QuarkXPress Server

Brilliant content that works

quark.com

A Guide to QuarkXPress Server 2024

April 2024

Contents

Conventions 1

Getting started with QuarkXPress Server 2

QuarkXPress Server architecture 2

Understanding QuarkXPress Server 2

Server templates and static projects 3

The document pool and the Streaming Document Provider 3

Projects and layouts 4

Job Jackets and resources 4

Caching 5

QuarkXPress Server preferences 5

Quark License Administrator 5

Master-renderer environment 5

Changing logging levels in “log4j2.xml” 6

Understanding QuarkXPress Server XTensions software 7

The QuarkXPress Server XTensions API 7

Configuring QuarkXPress Server for launch 8

Launching QuarkXPress Server 10

Quitting QuarkXPress Server 10

The QuarkXPress Server user interface 11

Navigation pane 12

Administration menu 18

General Preferences dialog box 18

General Preferences — Server 18

General Preferences — Log 20

General Preferences — Email 21

General Preferences — Authentication 21

Renderer Preferences dialog box 21
Preferences — Display 22
Preferences — Input Settings 22
Preferences — Font Fallback 22
Preferences — Open and Save 23
Preferences — Fonts 23
Preferences — EPS 23
Preferences — PDF 24
Preferences — Project General Settings 24
Preferences — Print Layout Settings 25
Preferences — Print Layout Measurements 25
Preferences — Paragraph 26
Preferences — Character 27
Preferences — Trapping 28
Preferences — Color Manager 30
Preferences — Layers 30
Preferences — Kindle 31
Preferences — Modifier 31
Job Jackets dialog box 31
App Studio preferences 32
Check Out License dialog box 32

Using QuarkXPress Server 33
Creating URL requests 33
Understanding URL requests 33
Understanding QuarkXPress Server namespaces 34
Looking up a namespace 35
Understanding QuarkXPress Server parameters 35
Looking up a parameter 35
Supported interfaces 36

The Dynamic Publishing Process (DPP) 36
Getting started 36
Getting started: HTTP and HTTPS 37
Dissecting a QXP Server URL 37
Interpreting the QXP Server Manager response 38
HTTP GET and POST Requests 39
Getting started: Web services 44
QRequestContext 45
RequestService 47
QRequest 54
RequestParameters 54
NameValueParam 55
QContentData 56
QException 57
QXP Server Manager 58
Using the Web interface 59
Understanding rendering 59
Understanding logging 60
Understanding render types 61
appstudio 62
ave 65
eps 67
epub 71
jpeg 72
kindle 75
literal 76
pdf 77
png 87
postscript 90
qcddoc 94

qxpdoc 97
screenpdf 99
html5 107
Understanding render modifiers 109
annotateerrors 110
appenderrors 111
box 111
boxes 112
compositionzone 113
layer 114
layout 118
movepages 118
page 120
pages 121
scale 122
spread 122
spreads 123
suppresserrors 123
Using content modifiers 124
Inserting text 124
Applying a font at import 126
Inserting a picture 127
Saving a projects with a new name 129
Importing XML with placeholders 131
Updating article geometry and content 134
Highlighting text in rendered output 136
Using XML modify 136
Modifying box properties and content 138
Creating boxes 142
Deleting boxes 144

Grouping and ungrouping items 146
Modifying text attributes 147
Modifying picture properties 151
Importing data 155
Exporting Job Jackets files during deconstruction 159
Using XML deconstruct and construct 159
Deconstructing a project 160
Constructing a project 163
Working with pages and spreads 167
Working with layouts 168
Working with layers 171
Working with boxes 172
Working with groups 178
Working with pictures 180
Working with text 181
Working with tables 185
Column Attributes 191
Table Span 192
Working with sections 194
Working with Composition Zones 195
Using XSL transformation 196
Working with lists 197
Working with anchored boxes 198
Working with placeholders 199
Working with metadata 201
Working with hidden text 202
Using interactivity 205
Specifying colors 212
Working with indexes 212
Marking and rendering indexes 212

Working with cross references 215
Creating and using hyperlinks 216
Web hyperlinks 216
Anchor hyperlinks 217
Page hyperlinks 217
Using the Streaming Document Provider 217
Using administrative request handlers 218
Addfile 219
Delete 224
Evaluate 225
Exportprefsasjj 226
Fileinfo 227
Flush 228
Flushall 229
Getdocinfo 229
Getdocpoollist 231
Getlogs 232
Getprefs 232
Getprocessid 233
Getprojinfo 233
Getrendererprefs 235
Getserverinfo 236
Jobjacket 237
Preflight 238
Setprefs 239
Setrendererprefs 240
updateprefsfromjj 246
Using the QXPSM SDK 247
Writing a Java QXPSM client 247
Java sample: Deconstructing a project 248

Java sample: Rendering a PDF 248
Java sample: Chained request 249
Java sample: AddFile request 249
Writing a .NET QXPSM client 250
.NET sample: Deconstructing a project 252
.NET sample: Rendering a PDF 252
.NET sample: Chained request 252
Writing an Objective-C client for macOS or iOS 253
Objective-C sample: Deconstructing a project 255
Objective-C sample: Rendering a PDF 255
Objective-C sample: Chained request 256
Extending QuarkXPress Server Manager 257
Writing special request handlers 257
Implementing a custom load balancer 258
Keep document open (sessions) 259
Using the Extensibility tool 259
Understanding ManagerSDK.xml 260
Using the RequestParameters class 262
Sample applications 263
Sample applications: QXP Server Manager 263
ASP.NET samples 263
C# samples 263
Java samples 264
JSP samples 264
Objective-C samples 264
Sample applications legal notice 264

QuarkXPress Server Features 274
Dynamic Pagination and Flow 274
Dynamic Pagination and Flow Problem 275

Dynamic Pagination and Flow Solution 276
Landscape pagination 280
Automatic callout stacking 285
Automatic callout stacking example 286
Nested anchoring 287
Nested anchoring examples 288
Nesting an INLINEBOX within an INLINEBOX 288
Nesting an INLINEBOX within an ENTRY of an INLINETABLE 288
Nesting an INLINETABLE within an INLINETABLE 288
Nesting an INLINETABLE within an INLINEBOX 289

Modifier schema (annotated) 290
Entities (Modifier schema) 290
ADDCELLS (Modifier schema) 293
ALIGNHORSETTINGS (Modifier schema) 294
ALIGNVERSETTINGS (Modifier schema) 295
ALLOWBOXOFFPAGE (Modifier schema) 296
ALLOWBOXONTOPASTEBOARD (Modifier schema) 297
ANCHOREDBOXREF (Modifier schema) 297
ARTICLE (Modifier schema) 299
AUTHOR (Modifier schema) 300
BNSTYLE (Modifier schema) 300
BORDER (Modifier schema) 301
BOTTOM (Modifier schema) 308
BOTTOMGRID (Modifier schema) 309
BOX (Modifier schema) 310
BOXATTRIBUTE (Modifier schema) 315
BOXREF (Modifier schema) 318
CALLOUTANCHOR (Modifier schema) 319
CALLOUTBOXREF (Modifier schema) 321

CELL (Modifier schema) 322
CHILDID (Modifier schema) 325
CLIPPING (Modifier schema) 326
COLGROUP (Modifier schema) 331
COLSPEC (Modifier schema) 331
COLUMN (Modifier schema) 332
COLUMNFLOW (Modifier schema) 334
COLUMNSPAN (Modifier schema) 335
COLUMNSPLIT (Modifier schema) 336
COMPONENT (Modifier schema) 337
COMPOSITIONZONE (Modifier schema) 340
CONDITIONALMASTERPAGEREFERENCE (Modifier schema) 344
CONTENT (Modifier schema) 345
CONTENTPH (Modifier schema) 348
CONTINUEDHEADER (Modifier schema) 349
CONTINUEDTROWSTYLE (Modifier schema) 349
CONTOUR (Modifier schema) 350
CONTOURS (Modifier schema) 352
COPYFIT (Modifier schema) 352
COPYRIGHT (Modifier schema) 353
DATAPROVIDER (Modifier schema) 353
DEL (Modifier schema) 354
DELETECELLS (Modifier schema) 354
DESCRIPTION (Modifier schema) 355
DROPCAP (Modifier schema) 355
EBOOKMETADATA (Modifier schema) 356
ENTRY (Modifier schema) 356
EVENTCOLSTYLE (Modifier schema) 358
EVENTROWSTYLE (Modifier schema) 359
FIRSTTCOLSTYLE (Modifier schema) 360

FIT (Modifier schema) 360
FITTEXT (Modifier schema) 361
FOOTER (Modifier schema) 364
FOOTERTROWSTYLE (Modifier schema) 364
FORMAT (Modifier schema) 366
FRAME (Modifier schema) 369
GEOMETRY (Modifier schema) 372
GRADIENT (Modifier Schema) 375
GRID (Modifier schema) 377
GRIDLINE (Modifier schema) 377
GROUP (Modifier schema) 379
GROUPCHARACTERS (Modifier schema) 380
GROWACROSS (Modifier schema) 382
GROWDOWN (Modifier schema) 382
GUIDE (Modifier Schema) 382
HEADER (Modifier schema) 384
HEADTROWSTYLE (Modifier schema) 384
HEIGHT(Modifier schema) 385
HIDDEN (Modifier schema) 385
HYPERLINK (Modifier schema) 388
ID (Modifier schema) 388
INDEXTERM (Modifier schema) 390
INLINEBOX (Modifier schema) 391
INLINETABLE (Modifier schema) 395
INS (Modifier schema) 396
INSET (Modifier schema) 396
INTERACTIVITY (Modifier schema) 397
ISBN (Modifier schema) 398
KEEPLINESTOGETHER (Modifier schema) 398
KEYWORDS (Modifier schema) 399

LASTTCOLSTYLE (Modifier schema) 400
LAYER (Modifier schema) 400
LAYOUT (Modifier schema) 402
LAYOUTMETADATA (Modifier schema) 406
LAYOUTREF (Modifier schema) 407
LEFT (Modifier schema) 407
LEFTCONTROLPOINT (Modifier schema) 407
LEFTGRID (Modifier schema) 408
LINESTYLE (Modifier schema) 410
LINKEDBOX (Modifier schema) 411
LIST (Modifier schema) 413
LOCATION (Modifier schema) 413
LOCKTOGRID (Modifier schema) 414
MASTERPAGESEQUENCE (Modifier schema) 415
MATHEQUATION (Modifier schema) 415
MAX (Modifier schema) 416
METADATA (Modifier schema) 416
MIN (Modifier schema) 416
MOVEDOWN (Modifier schema) 417
MOVELEFT (Modifier schema) 417
MOVERIGHT (Modifier schema) 417
MOVEUP (Modifier schema) 418
NOTE (Modifier schema) 418
ODDTROWSTYLE (Modifier schema) 418
ODDTCOLSTYLE (Modifier schema) 419
ORIGIN (Modifier schema) 420
OVERMATTER (Modifier schema) 421
PAGE (Modifier schema) 421
PAGEBREAK (Modifier schema) 423
PAGEREF (Modifier schema) 424

PAGESEQUENCE (Modifier schema) 425
PARAGRAPH (Modifier schema) 427
PARENTTABLE (Modifier schema) 430
PICTURE (Modifier schema) 430
PLACEHOLDER (Modifier schema) 435
POSITION (Modifier schema) 436
PROJECT (Modifier schema) 437
PUBLICATION (Modifier schema) 439
PUBLICATIONCHANNEL (Modifier schema) 439
PUBLISHER (Modifier schema) 440
REFNOTE (Modifier schema) 440
RELPOSITION (Modifier schema) 441
REPEATABLEMASTERPAGEALTERNATIVES (Modifier schema) 442
REPEATABLEMASTERPAGEREFERENCE (Modifier Schema) 442
RGBCOLOR (Modifier schema) 443
RICHTEXT (Modifier schema) 444
RIGHT (Modifier schema) 464
RIGHTCONTROLPOINT (Modifier schema) 464
RIGHTGRID (Modifier schema) 466
ROW (Modifier schema) 467
RUBI (Modifier schema) 470
RUBITEXT (Modifier schema) 471
RULE (Modifier schema) 473
RUNAROUND (Modifier schema) 476
SAVEAS (Modifier schema) 483
SCALETO (Modifier schema) 484
SECTION (Modifier schema) 485
SECTIONNUMBERFORMAT (Modifier schema) 486
SHADINGSTYLE (Modifier schema) 487
SHADOW (Modifier schema) 490

SHRINKACROSS (Modifier schema) 494
SHRINKDOWN (Modifier schema) 495
SINGLEMASTERPAGEREFERENCE (Modifier schema) 495
SIZE (Modifier schema) 496
SPINEIMAGE (Modifier schema) 496
SPLINESHAPE (Modifier schema) 497
SPREAD (Modifier schema) 498
STACKINGORDER (Modifier schema) 499
STATICCONTENT (Modifier schema) 499
STOP (Modifier Schema) 500
STORY (Modifier schema) 501
SUPPRESSOUTPUT (Modifier schema) 503
TAB (Modifier schema) 504
TABLE (Modifier schema) 505
TABLEBREAK (Modifier schema) 509
TABLESTYLE (Modifier schema) 510
TABSPEC (Modifier schema) 511
TBODY (Modifier schema) 511
TCOL (Modifier schema) 511
TCOLSTYLE (Modifier schema) 513
TCONTINUED (Modifier schema) 514
TEXT (Modifier schema) 514
TEXTATTRIBUTE (Modifier schema) 517
TEXTNODEPH (Modifier schema) 518
TEXTPH (Modifier schema) 519
TFOOT (Modifier schema) 520
THEAD (Modifier schema) 520
TITLE (Modifier schema) 520
TOP (Modifier schema) 521
TOPGRID (Modifier schema) 521

TROW (Modifier schema) 523
TROWSTYLE (Modifier schema) 523
VALUE (Modifier schema) 525
VERTEX (Modifier schema) 525
VERTEXPOINT (Modifier schema) 526
VERTICES (Modifier schema) 527
WIDTH (Modifier schema) 528
XREF (Modifier schema) 528

Using SSL 530
Secure Sockets Layer (SSL) support 530
Enabling SSL 530
Enabling HTTP and HTTPS 531
Verifying and using SSL 531
Keystores and SSL certificates 531

QuarkXPress Server XTensions software 533
CopyDeskArticle XTensions software 533
Rendering articles 533
Exporting articles 533
Adding articles to projects 534
Creating and deleting components 534
PDF Filter XTensions software 535
Modifier XTensions software 535
Using Modifier XTensions software 536
Creating XML for Modifier XTensions software 537
Layer XTensions software 537
InteractiveDesigner Server XTensions software 538
App Studio XTensions software 538

Telegraph XTensions software 540

Setting Telegraph preferences 540
Specifying a server for template upload 541
Using Telegraph XTensions software 542
Identifying QuarkXPress items and groups 542
Naming items and groups 543
Uploading templates 543
Uploading missing or modified pictures 545

QuarkXPress Server Manager 546
Understanding QuarkXPress Server Manager 546
Load balancing 546
Request timeout interval 547
Determining QuarkXPress Server instance availability 548
Ping 548
Ping document 548
Logging with QXP Server Manager 548
Caching 549
Web services 549
Working with QuarkXPress Server Manager 550
Starting QuarkXPress Server Manager 550
Starting the Manager server application 551
Starting the Manager Web client 551
Request handler binding 551
Configuring QuarkXPress Server instances 552
Adding and enabling a QuarkXPress Server instance 552
Editing a QuarkXPress Server instance 554
Enabling and disabling routing to QuarkXPress Server instances 554
Controlling load balancing 555
Deleting a QuarkXPress Server instance 555
Managing the cache 556

Viewing a QuarkXPress Server Manager server cache 556
Deleting files from the cache 557
Configuring cache options 557
Managing logs 558
Viewing log file details 558
Deleting log files 558
Configuring logging options 559
Using a proxy server 560
Generating automatic e-mail messages 560
Using other global settings 561
Saving a server configuration 563
Using custom error messages 564
Creating a custom error code 564
Flagging an error code to generate an e-mail message 564
Creating custom error text 565
Sending requests from a browser 565
Additional parameters 565

The XTensions Developer Kit (XDK) 568

Glossary 569

Legal Notices 570

Conventions

Formatting conventions highlight information to help you quickly find what you need.

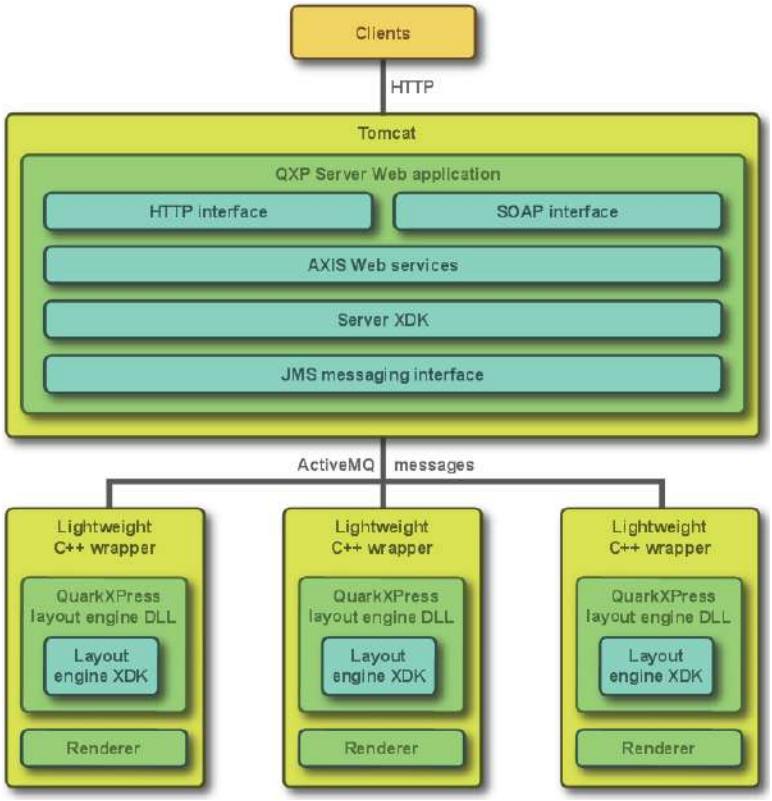
- Bold type style: The names of all dialog boxes, fields, and other controls are set in bold type. For example: “Click **OK**.”
- References: In descriptions of features, parenthetical
- l references guide you in accessing those features. For example: “The **Find/Change** dialog box (**Edit** menu) lets you find and replace text.”
- Arrows: You will often see arrows (**>**), which map out the menu path to a feature. For example: “Choose **Edit > Style Sheets** to display the **Style Sheets** dialog box.”
- Icons: Although many tools and buttons are referenced by name, which you can see by displaying ToolTips, in some cases icons are shown for easy identification. For example, “Click the button on the **Measurements** palette to center text.”
- Cross-platform issues: This application is quite consistent across operating systems. However, some labels, buttons, key combinations, and other aspects of the application must differ between macOS® and Windows® because of user interface conventions or other factors. In such cases, both the macOS and Windows versions are presented, separated by a slash, with the macOS version presented first. For example, if the macOS version of a button is labeled **Select**, and the Windows version is labeled **Browse**, you are directed to “Click **Select/Browse**.” More complex cross-platform differences are mentioned in notes or parenthetical statements.

Getting started with QuarkXPress Server

QuarkXPress Server lets you render QuarkXPress projects in a variety of formats.

QuarkXPress Server architecture

The QuarkXPress Server architecture is shown in the following diagram.



QuarkXPress Server architecture diagram

Understanding QuarkXPress Server

QuarkXPress Server helps automate the generation of QuarkXPress layouts to a variety of output formats—including JPEG, Portable Document Format (PDF), and PostScript®—all from a centralized Web application. To send a request to QuarkXPress Server, all you need to do is enter a URL into your Web browser's address field. For example, the following URL instructs the QuarkXPress Server application named "QXPServer" to return the file "MyProject.qxp" as a PDF file:

http://QXPServer:8080/pdf/MyProject.qxp

The QuarkXPress Server application receives these requests, renders (creates) the requested projects in the requested formats, and then returns the rendered file to the client application—the Web browser in this case.

➡ The format of QuarkXPress Server URL requests is described in detail in "*Creating URL requests*".

➡ You can also create custom applications that communicate with a QuarkXPress Server application using HTTP, Simple Object Access Protocol (SOAP), or another protocol.

You can think of QuarkXPress Server as a special version of QuarkXPress that runs on a server with the following main differences:

- Instead of accepting input from a keyboard and mouse, QuarkXPress Server accepts input in the form of URLs and other types of requests.
- Instead of sending a project to a printer, QuarkXPress Server renders (creates) output in a particular format and sends the rendered file to a client.

Server templates and static projects

QuarkXPress Server can open, render, and serve two types of projects:

- *Static projects* are QuarkXPress projects that can be rendered and served as-is in a variety of formats by the QuarkXPress Server application. For example, you might make a product manual available as a static project and allow customers to download that manual in a variety of formats.
- *Server templates* are QuarkXPress projects that can be manipulated by the QuarkXPress Server application before being rendered and served. For example, you might make a sales piece available as a server template so that each person who downloads it receives a personalized copy.

➡ QuarkXPress Server can open QuarkXPress documents, projects, and templates created in any language edition of QuarkXPress 7.0 or later. QuarkXPress Server can save and export projects in QuarkXPress 8.x, 9.x and 10.x format.

The document pool and the Streaming Document Provider

QuarkXPress Server has two main ways to find or receive content for rendering.

- The document pool
- The Streaming Document Provider

QuarkXPress Server can read templates and static projects from a directory called the *document pool*. The document pool can be any directory that is available to QuarkXPress Server through a file system or an FTP server. You can use any of the following methods to place files in the document pool:

- Drag the files to the document pool directory.
- Use the **Add Files** command in the **Document Pool** screen of the QuarkXPress Server Web interface.
- Upload the files using FTP to the document pool directory.
- Use Telegraph XTensions® software to upload the files to the document pool from within QuarkXPress. (For more information about Telegraph XTensions software, see "*Telegraph XTensions software*.")

For more information about the document pool, see "*The QuarkXPress Server User Interface*."

➡ The document pool directory cannot be an encrypted directory.

QuarkXPress Server can also receive templates, projects, and other files as part of a multipart HTTP request. For more information, see "*Using the Streaming Document Provider*."

In addition, QuarkXPress Server can serve documents from a database, a content management system, or other sources. Collectively, the document pool and any other source of files to be served are referred to as *document providers*.

Projects and layouts

QuarkXPress projects can contain one or more layouts, and only one layout can be rendered at any given time. If you do not specify a layout when you send a rendering request, QuarkXPress Server renders the first layout in the project.

Job Jackets and resources

In QuarkXPress, *resources* are things such as style sheets, colors, H&Js, output styles, and item styles. Resources are stored in a *Job Jackets structure*, which can be either embedded in a project or stored in a separate Job Jackets file.

QuarkXPress Server uses a default Job Jackets file to make a default set of resources available to all projects handled by QuarkXPress Server, regardless of whether they are included in the projects and articles you render. You can update this file in two ways:

- Using the QuarkXPress Server Web interface. For more information, see "*Job Jackets dialog box*."
- Using request handlers. For more information, see "*jobjacket*" and "*updateprefsfromjj*."

Once you've downloaded the default Job Jackets file, you can update its resources using QuarkXPress. For more information, see "Job Jackets" in *A Guide to QuarkXPress*.

The location of the default Job Jackets file is stored in the QuarkXPress Server preferences folder. For more information, see "*QuarkXPress Server preferences*."

Caching

To maximize efficiency, QuarkXPress Server uses cached versions of all rendered projects whenever possible. You can configure projects so that they are never cached, so that they are cached for a particular amount of time, or so that they are rendered every time they are served.

QuarkXPress Server preferences

When you launch QuarkXPress Server, the application creates preferences files that are functionally and structurally equivalent to the preferences files created by QuarkXPress. These preference files reside in the QuarkXPress Server "Preferences" folder. QuarkXPress Server also creates a "QuarkXPress Server.prf" file in the "Preferences" folder. This file contains preference settings that are specific to QuarkXPress Server.

QuarkXPress Server uses these preferences the same way QuarkXPress uses them. If an XTensions module creates a project in QuarkXPress Server, that project draws its settings from the QuarkXPress Server preferences just as a new QuarkXPress project draws its settings from the QuarkXPress preferences.

Preferences files are stored in the following locations:

- macOS: [User]/Library/Preferences/Quark/QuarkXPress Server Renderer/
- Windows: C:\Windows\System32|config\systemprofile\AppData\Roaming\Quark\QuarkXPressServer Renderer [version]

For more information about preferences, see the "*Administration menu*."

Quark License Administrator

To prevent unauthorized use, launching QuarkXPress Server requires the presence of a Quark® License Administrator (QLA) server. QuarkXPress Server follows the configuration and control rules that are enforced by QLA. For more information about QLA, see the QLA documentation included with QuarkXPress Server.

Master-renderer environment

Requests for project renders are stored in a *connection queue*. The requests in the rendering queue can be processed by a single QuarkXPress Server application, or by a master QuarkXPress Server application and several renderers (additional instances of QuarkXPress Server). The master

QuarkXPress Server application launches the available number of renderers and then passes the requests in the connection queue to those renderers as they become available. The number of renderers available for launch is determined by the number of licenses available from the QLA server.

The master QuarkXPress Server process and all the renderers it launches share the following elements:

- The same application preferences (each renderer has its own preferences files, but QuarkXPress Server keeps them synchronized)
- The same document cache in memory
- The same memory cache
- The same server XTensions modules (a separate instance of each XTensions module runs with each renderer)
- The same server document pool (if defined in the QuarkXPress Server preferences or if a document provider is used in place of the document pool)

If a renderer unexpectedly quits, the master QuarkXPress Server restarts the renderer without requiring any action from you.

Changing logging levels in "log4j2.xml"

You can change the logging levels for QuarkXPress Server. Options include ERROR, INFO, WARN, DEBUG, and TRACE.

- ERROR = includes messages that indicate disrupted and failed requests.
- INFO = includes messages that indicate the state of services.
- WARN = includes non-critical service error messages
- DEBUG = includes messages that indicate server resource usage.
- TRACE = includes messages according to activity related to requests.

Refer to Java documentation for more information about logging levels.

To change logging levels:

1. Open the "conf" folder in your QuarkXPress Server folder.
2. Open "log4j2.xml" in a text-editing application.
3. To define the logging level for QuarkXPress Server errors, scroll to <logger name=com.quark.qxps. The structure is as follows:

```
<logger name="com.quark.qxps">
  <level value="ERROR" />
</logger>
```

4 To define the logging level for QuarkXPress Server transactions, scroll to <logger name=QXPSTransactionLogger. The structure is as follows:

```
<logger name="com.quark.qxps" additivity="false">
 <level value="INFO" />
 <appender-ref ref="QxpsTransactionFileAppender" />
</logger>
```

5 To define the logging level for other activity, scroll to the <root>. The structure is as follows:

```
<root>
 <priority value="ERROR" />
 <appender-ref ref="QxpsServerAsyncAppender" />
</root>
```

6 Save and close "log4j2.xml."

Understanding QuarkXPress Server XTensions software

QuarkXPress Server ships with a collection of XTensions software that adds capabilities to QuarkXPress Server. For example, PDF Export XTensions software lets QuarkXPress Server serve content in PDF format; Modifier XTensions software lets you retrieve, manipulate, and reconstruct XML representations of projects; and QuarkCopyDesk® Renderer XTensions software lets you create QuarkCopyDesk articles. Telegraph XTensions software works with QuarkXPress to allow designers to name boxes in template files so that those boxes can be addressed by URLs.

For more information about XTensions software included with QuarkXPress Server, see "*XTensions software*" and "*Telegraph XTensions software*."

The QuarkXPress Server XTensions API

In addition to the XTensions modules included with QuarkXPress Server, developers can create custom XTensions software that add features. The complete server XTensions Application Programming Interface (API) documentation is available in the QuarkXPress Server XTensions Developer's Kit (XDK).

➡ As of version 8.0, the QuarkXPress Server XDK is Unicode-compliant.

The QuarkXPress Server XDK lets you create XTensions modules that provide the following abilities:

- The ability to register request handlers.
- The ability to register project providers.
- The ability to register new render formats.

- The ability to add items to the list of response properties, cookies, and HTTP header items.
- The ability to log messages in log files.
- The ability to initiate a new transaction to be processed by the server.
- The ability to completely control how projects are processed by the server.

In addition, QuarkXPress Server XTensions software can register for the following basic callbacks:

- Pre-processing
- Content loading
- Layout modification
- Post-processing
- Removing slugs while running the QuarkXPress project renderer
- Analyzing the server after a transaction is complete
- Pre and post-transaction callback

Configuring QuarkXPress Server for launch

To configure QuarkXPress Server prior to launch, open the file [QuarkXPress Server application folder]/conf/ServerApp.properties folder) and modify it as follows:

- To make the server run without loading any network interface, use the -nonetwork option with the qxpservercore.serverRendererParameters. In this mode, the only transactions a server can run are those passed to it by another process.
- To control whether renderers are monitored, set qxpservercore.monitorrenderers.value to true or false.
- To specify the query interval for monitoring renderers, set qxpservercore.monitorrenderers.queryinterval.value to a value in seconds.
- To specify the number of retries for monitoring renderers, set qxpservercore.monitorrenderers.noofretries.value to an integer. If a renderer process has been attempting to fulfill a request for the specified number of retries (with the specified query interval in seconds between retries), the renderer monitor recycles that process.
- To control how many renderers the master process launches, specify a number for qxpserver.subrenders. Note that the number of renderers you can launch depends on your license.

- To force the renderers to restart on a periodic basis, specify a value in hours for qxps.render.recycle.interval. The default value is 24, or 24 hours. Decimal values are permitted. Renderers restart serially, so one renderer doesn't restart until the other is finished restarting. If a renderer is busy, the master process waits for 15 minutes, and then if the renderer is still busy, postpones the restart until the next interval elapses. Set this value to zero to turn the automatic restart feature off.
- The request timeout at Renderer lets you specify the timeout value for rendering. It is set to 10 minutes by default.

qxpserver.renderer.request.timeout.name = requestTimeout

qxpserver.renderer.request.timeout.value = 10

- To control whether the server should collect performance metrics, set qxpserver.CollectPerformanceMetrics=true to collect and false to not collect metrics.
- To set the number to transactions after which performance metrics data will be flushed to file: qxpserver.CollectPerformanceMetrics.flush.transactioncount = 100
- To send JMX notifications when performance metrics change set: qxpserver.CollectPerformanceMetrics.sendjmxnotifications = true
- Enable this flag to automatically refresh admin ui through JMX notifications qxpserver.CollectPerformanceMetrics.adminui.autorefresh = true
- To reset metrics on every launch of qxps. If set to true, the QuarkXPress Server will clear old metrics and start capturing fresh metrics on every launch. qxpserver.CollectPerformanceMetrics.resetmetricsonserverlaunch = false
- To accelerate the time to troubleshoot a failed rendering request, QuarkXPress Server creates the rescue folder and collects the data for failed and timed out transactions by default into a folder name QXPSRescuedata<ver> relative to doc pool.

Setting this value to false means the rescue folder will not be created and no data will be collected.

qxpserver.rescuefolder.manage=true

- Specify the rescue folder path for failed transactions. If no value is given here, the document pool path would be considered by default: qxpserver.rescuefolder.path = <Specify custom folder path>

Launching QuarkXPress Server

On Windows, you can install QuarkXPress Server as an application or as a service (Quark recommends that you always run it as a service). On macOS, QuarkXPress Server always runs as an application.

If you install QuarkXPress Server on Windows as an application, you can launch it using the **Start** menu or by double-clicking the "ServerStartup.bat" in the QuarkXPress Server application folder.

For information on launching QuarkXPress Server in a separate Tomcat installation, see "Deploying QuarkXPress Server externally" in the QuarkXPress Server *Getting Started*.

➡ QuarkXPress Server offers a browser-based user interface instead of a conventional user interface.

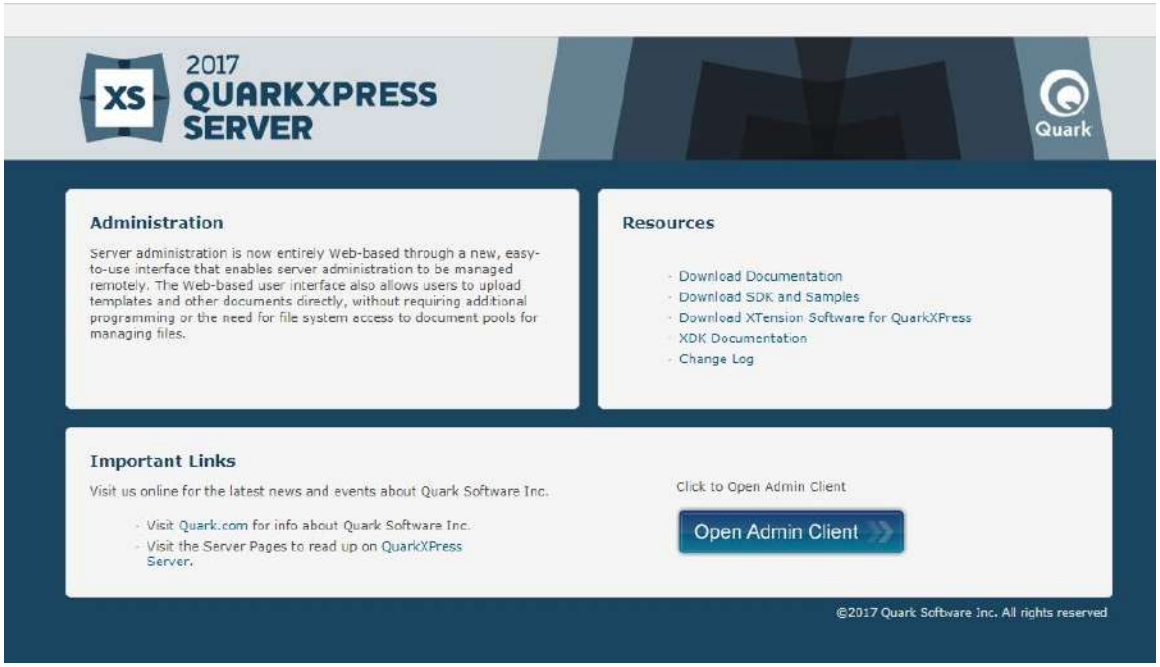
Quitting QuarkXPress Server

To quit QuarkXPress Server, press Control+C.

The QuarkXPress Server user interface

QuarkXPress Server offers a browser-based user interface. This chapter describes how you can use it to configure and customize your QuarkXPress Server application and manage your server XTensions modules.

To view the QuarkXPress Server welcome page, launch a Web browser and enter the URL http://[server]:[port] (where [server] is the IP address or domain name of the server and [port] is the TCP/IP port on which the server is running). The welcome screen displays.



The QuarkXPress Server welcome page

To display the administrative client, click **Open Admin Client**. If the server has realm verification enabled, you will be asked to enter your user name and password. The administrative client displays.

Navigation pane

Status Monitor screen

The navigation pane on the left side has two areas shown in the lower left. The **Server Information** area lets you view server information and the transaction log, and the **Document Pool** area lets you view the contents of the document pool. You can collapse and expand this pane with the button at the right end of the **Navigation Pane** header.

If you click **Server Information**, the **Status Monitor** screen displays.

This screen shows the following usage reporting metrics:

1. **Server Name:** Name or IP of the machine running QuarkXPress Server.

2. **Server Startup time:** Shows the date and time at which the QuarkXPress Server Service was launched.

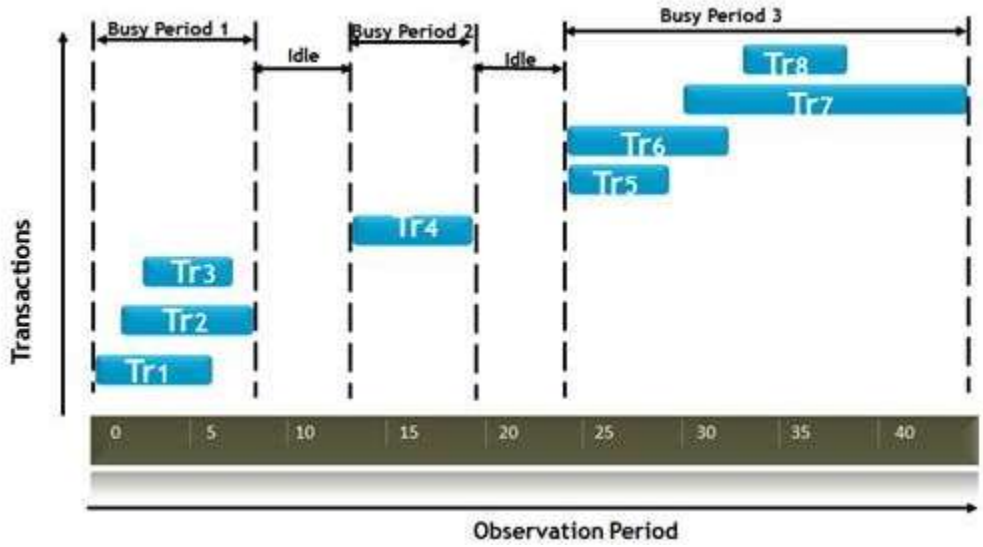
3. **Active Renderers:** Represents the number of renderers busy processing a rendering job. This updates dynamically.

4. **Idle Renderers:** Represents the number of renderers in an idle state. This updates dynamically.

5. **Observation Time:** The time period during which the server's activity is monitored to collect performance metrics.

6. **Busy Time:** Sum of all time periods when the server is busy processing at least one rendering transaction.

- For Example: Busy Time = Busy Period 1 + Busy Period 2 + Busy Period 3 + … + Busy Period n, where Tr1, Tr2, …Trn represent incoming rendering transactions.



Busy Time: Transactions Vs. Observation Period

7. **Server Utilization:** The percentage of server capacity used during the **Observation Time**.

Utilization (%) = (Busy Time /Observation Time) * 100.

- For Example: For an Observation Time of 50 minutes if the server is busy for 40 minutes, the Server Utilization(%) = (40 / 50) * 100 = 80%.

8. **Rendering Throughput**: The average number of rendering transactions completed during **Busy Time.**

Rendering Throughput = total rendering transactions completed / Busy Time.

- For example, if 150 rendering transactions are completed in a busy time of 50 minutes, then the throughput would be 150/50 = 3 transactions / min.

9. **Arrival Rate:** The number of arrivals per unit of time.

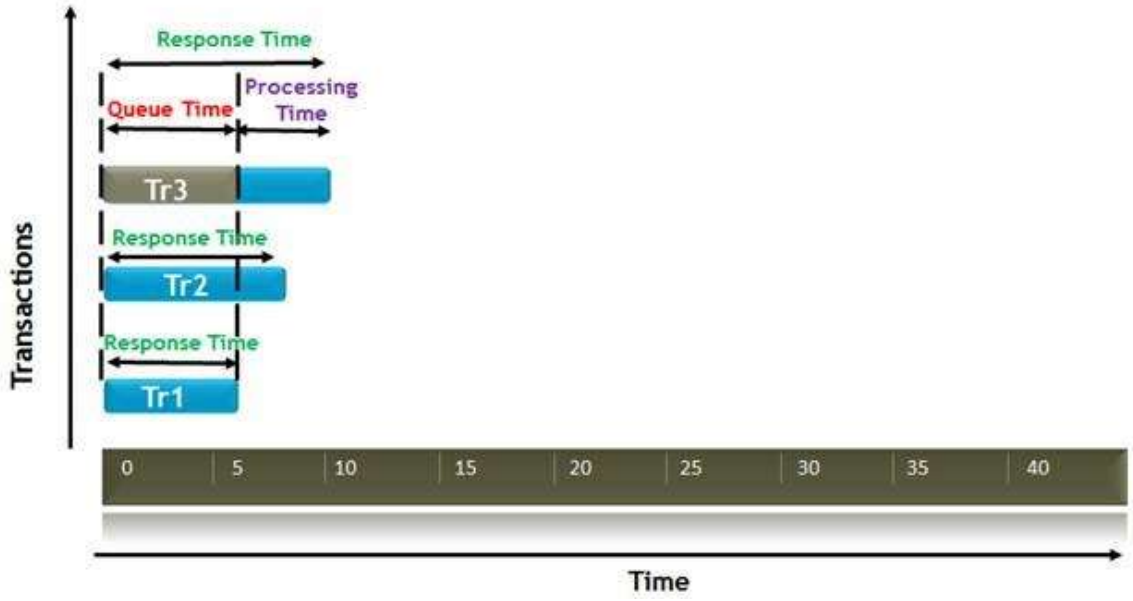
Arrival Rate = Total transactions arrived / Observation Time

- For example, for a total of 1500 rendering requests arrived, during 50 minutes Observation Time, the Arrival Rate = 1500/50 = 30 transactions / minute.

10. **Response Time for a rendering transaction:** It is the total time taken to complete a transaction from its submission to completion.

Response Time= Queue Time + Processing Time

- For example if a server running with 2 the rendering transaction Tr1 goes to renderer1, Tr2 goes to renderer2, Tr3 waits in queue until one of the renderer becomes free. The response time for each rendering transaction is as shown below.



Response Time: Transactions Vs. Time

11. **Average Response Time:** Average amount of time taken to complete a transaction from its submission to completion.

For Example:

Time taken to complete transaction 1 = Tr1

Time taken to complete transaction 2 = Tr2

…

…

Time taken to complete transaction n = Trn

Average Response Time = (Tr1 + Tr2 + … + Trn) / n

12. **Peak Response Time:** The highest response times recorded during a given observation time.

For example, if during a given **Observation Time**, if response times for various rendering transactions are as below,

Tr1 = 7 sec

Tr2 = 15 sec

Tr3 = 2 sec

Tr4 = 2 sec

Tr5 = 20 sec

…

…

Trn = 4 sec

Peak Response Time = MAX(7,15,2,2,20 …4) = 20 sec.

13. **Average Queue Time:** The average time a rendering job stays in the queue before it gets processed. The average queue time = total queue time / total rendering transactions completed.

- For example, if the total queue time is 3 minutes (i.e. 180 seconds) and 150 transactions are completed, then the Average Queue Time = (3 * 60) /150 = 1.2 seconds

14. **Peak Queue Time:** The highest waiting time a rendering job stays in the queue.

For example, if during a given Observation Time, the waiting times are:

Tr1 = 0 sec

Tr2 = 5 sec

Tr3 = 2 sec

…

…

Trn = 4 sec

Peak Queue Time = MAX(0,5,2, …4) = 5 sec.

15. **Current Queue Length:** The number of rendering jobs waiting in the queue to be processed.

16. **Peak Queue Length:** The highest queue lengths recorded during a given observation time.

17. **Rendering Requests:** The total number of transactions delegated to renderers during a given observation time (e.g. PDF, JPG, /getprojinfo, /jobjacket, /getrendererprefs, /setrendererprefs, etc).

18. **Pages Rendered:** Sum total of all pages rendered for each rendering transaction during a busy time.

19. **Images Processed:** Sum total of all image files either imported or rendered as a part of the rendering transaction during the observation time.

20. **Average Rendering Time Per Page:** The average time taken to render a page. Average rendering time per page = busy time / total pages rendered.

➡ For example, if a total of 15000 pages are rendered when the server was busy for 50 minutes (i.e. 3000 sec), the Average Rendering Time Per Page = (50*60)/15000 = 0.2 seconds

21. **Total Renditions:** The number of renditions of various formats rendered by the server during the observation time. By default, this includes PDF, JPEG, HTML5, EPUB, KINDLE, APPSTUDIO, EPS, XML, PNG, CopyDesk Articles, QuarkXPress Projects.

22. **Failed Transactions Count:** The number of failed transactions.

23. **Error Rate**: The percentage of failed transactions during the Observation Time.

➡ For example if out of 500 rendering transactions, if 10 transactions failed, then the Error Rate (%) = (10 / 500) * 100 = 2%.

24. **RESET button:** Clicking on the reset button results in the following sequence of actions:

- The server usage information recorded during the Observation Time is backed up to a backup file (Telemetry.json) within the QuarkXPress Server logs folder.
- Clears all the server usage information displayed in the Status Monitor page of Admin UI by resetting the values to zero.
- Starts a fresh Observation Time under which all the usage information will be freshly recorded again.

The icons in the area below represent the renderers that are currently running and show which requests are being processed by each renderer in real time. This screen also shows the number of active and idle renderers.

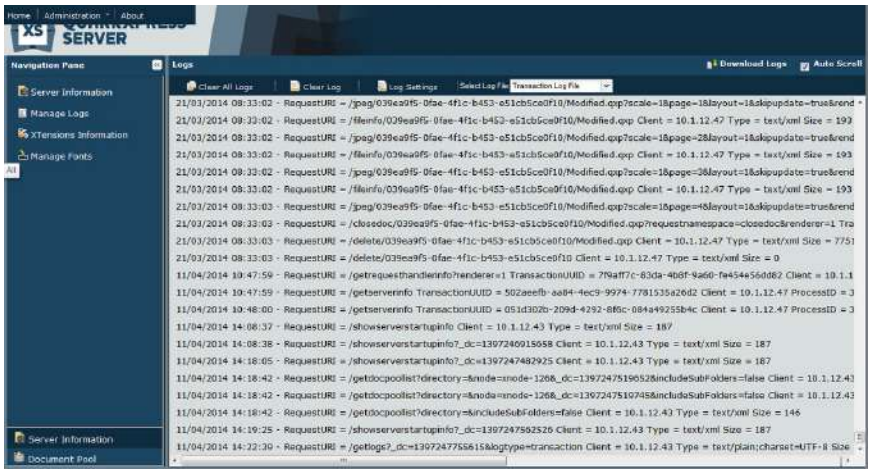
➡ The information for the Observation Time can be downloaded as a backup (step-5). A file named “Telemetry.json" is downloaded in association with the .zip archive downloaded using the **Download Logs** button under **Manage Logs**.

REST interface for usage data:

http://<servernameorip>:8080/telemetry

http://<servernameorip>:8080/cleartelemetry

If you click **Manage Logs**, the **Logs** screen displays the current transaction log.

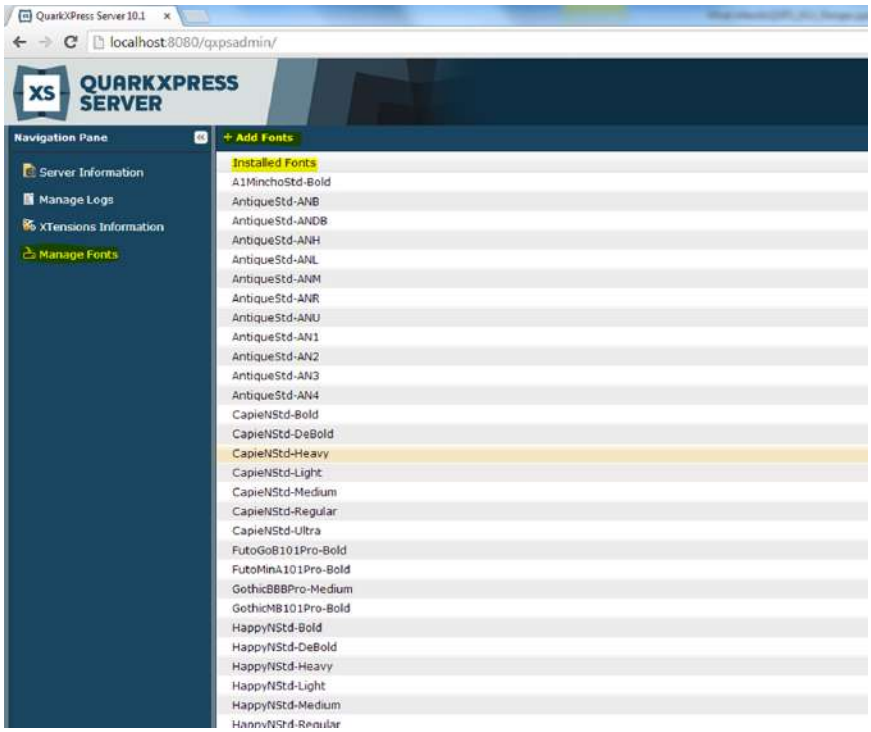


Logs screen

- To download the active log including recent crash reports, click **Download Logs** at the top of the **Logs** header.
- To make the log scroll automatically as events occur, check **Auto Scroll**.
- To clear all logs, click **Clear All Logs**.
- To clear the current log, click **Clear Log**.
- To display the **Log Settings** dialog box, click **Log Settings**. In this dialog box, you can change the maximum log file size, the maximum rolling count, and the logging level for the general QuarkXPress Server log, the QuarkXPress Server transaction log, and the QuarkXPress Server fatal log.
- Each transaction entry in the QuarkXPress Server transaction log has some useful information related to request URI, time taken for rendering, rendered output size, pages generated, images processed, average time per page, time/duration a transaction has waited in the queue before it got processed(queue time), queue length, and the number of other transactions still waiting in the queue at the time it was processed.
- To view a particular log file, choose an option from the **Select Log File** drop-down menu.

If you click **Show XTensions Information**, the **XTensions Information** screen displays, allowing you to view the status of all installed XTensions modules.

If you click **Manage Fonts**, a list of fonts installed on the Server machine is displayed.



Fonts screen

To search fonts, use the list search box in the top right corner.

To add fonts to the Server machine, click the **Add Fonts** button. This will copy the fonts to the "privatefonts" folder in the QuarkXPress Renderer folder.

Administration menu

The administration menu lets you manage QuarkXPress Server.

➡ You do not have to restart the server in GUI mode to set preferences.

General Preferences dialog box

The **General Preferences** dialog box (**Administration > Preferences > General**) lets you set preferences that are not related to rendering. It includes the tabs described in the following topics.

➡ You can also set general preferences using the setprefs request handler. For more information, see "*setprefs*."

General Preferences — Server

The **Server** tab (**Administration > Preferences > General > Server**) includes the following controls.

Use the **Document Root Folder** field to specify the location of the document pool.

➡ Putting the document pool on a network connected drive is not recommended, because this negatively impacts the performance of QuarkXPress Server.

Use the **Allow Memory Caching** check box to control whether memory is cached.

Use the **Max Memory Cache Size** field to specify the maximum memory size allocated to the cache memory. Valid values are from 10MB to 1024MB.

Use the **Force Served Documents Closed** check box to control whether QuarkXPress Server closes projects from the document pool after it renders them, regardless of the Telegraph XTensions software setting. Uncheck this box to keep such projects open on the server.

Use the **Default Renderer Type** drop-down menu to specify the default rendering format for the server.

- **Appstudio**: Returns a .zip file containing an HTML5 App Studio article.
- **Appstudio Upload**: Exports an HTML5 App Studio article and uploads it to the App Studio Publishing Portal.
- **ePUB**: Returns an ePUB file.
- **EPS Document**: Returns an Encapsulated PostScript (EPS) file.
- **JPEG**: Returns a JPEG file.
- **PDF**: Returns a PDF file.
- **PNG**: Returns a Portable Network Graphics (PNG) file.
- **PostScript**: Returns a PostScript file.
- **QCDDOC**: Returns a QuarkCopyDesk article.
- **QuarkXPress Document**: Returns a QuarkXPress project.
- **Raw Custom**: Returns a file in internal QuarkXPress format for use by server XTensions software developers.
- **RLE Raw Custom**: Returns a file in internal QuarkXPress format (compressed using Run Length Encoding) for use by server XTensions software developers.

Use the **Scale** field to specify the default scale percentage at which QuarkXPress Server should render projects.

Use the **Disable QXD Return** check box to specify whether QuarkXPress Server can return QuarkXPress projects to clients.

General Preferences — Log

The **Log** tab (**Administration > Preferences > General > Log**) includes the following controls.

Use the **Log Document Problems** check box to specify whether to include problem descriptions in transaction log files. The "Log" folder inside the QuarkXPress Server application folder contains three log files:

- **QuarkXPress Server Fatal Log.log**: This log lists all fatal errors.
- **QuarkXPress Server Log.log**: The log for the Java process. This log contains source code-level logging information that can be useful in troubleshooting.
- **QuarkXPress Server Transaction Log.log**: This log lists all transactions and all errors.

➡ Detailed application logging is enabled for these log files only if it is enabled in the "log4j.xml" file.

To log detailed transaction timing data, check **Log Timing Data**.

Logged problems include the following:

- **Missing Fonts**: If fonts are missing from a project that has been requested for rendering, a one-line description of each missing font is added to the error log. If QuarkXPress Server receives a request to render a project and does not have access to the fonts required by the project, it uses the fonts specified in the **Fonts** tab of the **Preferences** dialog box (**Administration > Preferences > General**). If these fonts are also unavailable, QuarkXPress Server substitutes Helvetica (macOS) or Arial (Windows). This behavior is the same as it is in QuarkXPress.
- **Missing Pictures**
- **Missing SXTs**: If a required server XTensions module is missing when a rendering request is received, a one-line description of each missing module is added to the error log. If the name of the missing module is not returnable, the XTensions module ID number is returned.
- **Text Encoding/Character Set Problems**: If text is sent to a text box in the template and the system does not have access to the correct font glyph, the issue is logged. The log data indicates the character set that the system attempted to convert. For example, the log entry might show that a request for Japanese characters was sent to an English project.

➡ Information about missing fonts and missing pictures is also recorded in the "QuarkXPressServerRenderer.log" file. This file also contains detailed timing information, including the transaction UID for each transaction. This log file can be found here:

- macOS: [drive]/Users/[user name]/Library/Logs/Quark

- *Windows*: [drive]:\Program Files\Quark\QuarkXPress Server 2017\log

➡ To download all logs to a non-server computer, click **Show Transaction Log** in the QuarkXPress Server Web interface, then click **Download Logs** on the home page.

General Preferences — Email

QuarkXPress Server can automatically notify someone by e-mail when the checked-out QLA license is about to expire. The **Email** tab (**Administration > Preferences > General > Email**) lets you specify where such e-mail messages should be sent.

Use the **Server** field to specify the domain name or IP address of the e-mail server that QuarkXPress Server should use to send messages (for example, mail.quark.com).

Use the **Port** field to specify the port number of the e-mail server that QuarkXPress Server should use to send messages. Valid values are from 0 to 255.

Use the **From** to specify the source e-mail address the QuarkXPress Server application should use to send messages (for example, QXPSserver1@quark.com).

Use the **To** to specify the e-mail address to which the QuarkXPress Server application should send messages (for example, QXPServerAdmin@quark.com).

General Preferences — Authentication

The **Authentication** tab (**Administration > Preferences > General > Authentication**) includes the following controls.

Check **Authenticate For Admin Requests** to enable the **Username** and **Password** fields. These fields let you control access to administrative parameters and features. You can enter up to 32 characters.

Renderer Preferences dialog box

Use the renderer **Preferences** dialog box (**Administration > Preferences > Renderer**) to set up default preferences for use when projects are created in QuarkXPress Server. It includes the tabs described in the following topics.

➡ Many of these preferences come into play only when you create a project in QuarkXPress Server, either using a construct request or through a request handled by a custom QuarkXPress Server XTensions module.

➡ You can also set renderer preferences using the setrendererprefs request handler. For more information, see"*setrendererprefs*."

Preferences — Display

The **Display** pane (**Administration > Preferences > Renderer > Display**) includes the following controls.

Use the **Color TIFFs** drop-down list to specify the color depth of screen previews created for color TIFFs when they are imported.

Use the **Gray TIFFs** drop-down list to specify the resolution of screen previews created for grayscale TIFFs when they are imported.

Choose a profile that corresponds to your monitor from the **Monitor Profile** drop-down menu, or choose **Automatic**. Profiles can be placed in the "Profiles" folder in the QuarkXPress Server application folder.

Preferences — Input Settings

The **Input Settings** tab (**Administration > Preferences > Renderer > Input Settings**) includes the following controls.

Check **Smart Quotes** to force QuarkXPress to replace feet (')and inches (") marks automatically with the specified quotation marks.

Use the **Format** drop-down menu to specify the default characters to be used with the **Smart Quotes** feature and **Convert Quotes** option in the **Get Text** dialog box (**File > Get Text**).

To change the separators used for indicating sequential ranges for output, enter a value in the **Sequential Page Range Separator**. This value override the preferences set for a project.

To change the separators used for indicating nonsequential ranges for output, enter a value in the **Non Sequential Page Range Separator**. This value override the preferences set for a project.

Preferences — Font Fallback

The **Font Fallback** pane (**Administration > Preferences > Renderer > Font Fallback**) includes the following controls.

Check **Font Fallback** to activate the Font Fallback feature. When this feature is active, if the application encounters a character it cannot display in the current font, it attempts to find a font that can display the character.

If the application encounters a missing font when opening a project, it uses the preferences in this pane to determine which substitute fonts to use.

➡ If you add characters to an existing project and the font cannot support those characters, the application will search the system for a font that can display the characters.

Check **Search** to have the application search for a suitable font that is used in the active project. To restrict the search to a particular range, choose an option from the **Search Type** drop-down menu. To search the entire story where a missing font occurs, choose **Active Story**. To search a particular number of paragraphs in both directions, choose **Paragraph** and enter a number in the **Search Limit** field.

To indicate which fallback fonts should be used when no other font can be found (taking the **Search** settings into account), choose options from each of the drop-down menus in the **Font List** area.

To indicate which font should be used for the slug line when a layout is printed with registration marks turned on, choose an option from the **Slug Line Font** drop-down menu.

Preferences — Open and Save

The **Open and Save** pane (**Administration > Preferences > Renderer > Open and Save**) includes the following controls.

Choose an option from the **Encoding** drop-down menu to indicate how the applications should display characters in non-Unicode text.

Preferences — Fonts

The **Fonts** pane (**Administration > Preferences > Renderer > Fonts**) includes the following controls.

To specify default replacement fonts, check **Specify Default Replacement Font** and choose options from the **Roman** and **East Asian** drop-down menu.

To highlight characters that are in a Traditional Chinese encoding's UDA/VDA (User Defined Area/Vendor Defined Area) range so that these characters can be visually verified, check **Highlight character ranges defined by Traditional Chinese font vendors**.

Preferences — EPS

The **EPS** pane (**Administration > Preferences > Renderer > EPS**) includes the following controls.

To control whether the application should generate a preview of an EPS file or use the preview (if any) embedded in the file, choose an option from the **Preview** drop-down list. The option specified in this pane is used only when the EPS preview is being created. If you change the preference, you need to reimport the EPS file.

Preferences — PDF

Use the **PDF** pane of the **Preferences** dialog box (**Administration > Preferences > Renderer > PDF**) to set preferences for rendering in PDF format.

➡ The **PDF** pane displays only if PDF Filter XTensions software is loaded. For more information, see "*PDF Filter XTensions software.*"

Use this pane to specify a PDF workflow:

- Click **DirectPDF** to generate PDF output in the browser. This is the default option.
- Click **PDFtoFolder** to generate and save PDF files to a folder. Click **Select/Browse** to specify a location for the folder in the **Watched Folder** field.
- Click **PS4D** (PostScript File for Later Distilling) to generate a PostScript file. Click **Select/Browse** to specify a location for the folder in the **Watched Folder** field.

Use this pane to set the desired PDF output style. Choose from the following output styles:

- **Default PDF Output Style**
- **Print - Medium Quality/Medium Resolution**
- **PDF/X-1a:2001**
- **PDF/X-3:2002**
- **Press - High Quality/High Resolution**
- **Print - Medium Quality/Medium Resolution**
- **Screen - Low Quality/Low Resolution**
- **Screen - Medium Quality/Low Resolution**

Use this pane to specify the folder path for the distiller error log file. The path is used by the PDFFilter XTension software to create the log file to log the errors that occur during the distillation process.

➡ The default path to the log file is <users>/Documents. If you choose to change the default, the log folder specified must be a pre-existing folder on the system.

Preferences — Project General Settings

The **Project General Settings** pane (**Administration > Preferences > Renderer > Project General Settings**) includes the following controls.

Check **Use OpenType Kerning** to activate the default kerning values for OpenType fonts. When OpenType kerning is active, it overrides any kerning specified through **Kerning Table Edit** (**Utilities** menu) for OpenType fonts.

To disable OpenType kerning for full-width characters, check **Do Not Kern Full Width Characters**.

Preferences — Print Layout Settings

The **Print Layout Settings** pane (**Administration > Preferences > Renderer > Print Layout Settings**) includes the following controls.

Use the **Master Page Items** drop-down menu to control what happens to master items when master pages are applied.

- Click **Keep Changes** if you intend modified master items on your layout pages to remain when a new master page is applied. The items that are kept are no longer master items.
- Click **Delete Changes** if you want modified master items on your layout pages to be deleted when a new master page is applied.

Use the **Framing** drop-down menu to specify whether frames are placed inside or outside text and picture boxes.

- When you click **Inside**, the distance between the text and the frame is determined by the box's **Text Inset** values (**Item > Modify**). When you place a frame inside a picture box, the frame overlaps the picture.
- When you click **Outside**, the frame is placed outside the box, increasing the box's width and height. The frame cannot extend beyond a constraining box or the pasteboard.

Use the **Auto Page Insertion** drop-down menu (Print layouts only) to determine whether pages are inserted automatically to contain text overflow from an automatic text box or a chain of text boxes (on a page associated with a master page that contains an automatic text box). The drop-down menu also enables you to determine where any pages will be inserted.

Preferences — Print Layout Measurements

The **Print Layout Measurements** pane (**Administration > Preferences > Renderer > Default Print Layout Measurements**) includes the following controls.

Use the **Horizontal** and **Vertical** drop-down menus to specify the measurement system for the rulers displayed along the top and left of the layout window. **Horizontal** corresponds to the top ruler; **Vertical** corresponds to the left ruler.

Use the **Points/Inch** field to override the default value of 72 points per inch. The application uses the value here as the basis for all point and pica measurements, as well as for all point- and pica-to-inch conversions. The desktop publishing standard for points per inch is 72. However, the traditional typographic standard used on most metal typographic rulers is usually

approximately 72.27 or 72.307 points per inch (range = 60 to 80 pt, measurement system = points, smallest increment = .001).

Use the **Ciceros/cm** field to specify a ciceros-to-centimeter conversion value different from the standard 2.1967 (range = 2 to 3 c, measurement system = ciceros, smallest increment = .001).

Preferences — Paragraph

The **Paragraph** pane (**Administration > Preferences > Renderer > Paragraph**) includes the following controls.

Use the **Auto Leading** feature to automatically set line spacing. Unlike paragraphs with absolute leading (identical line spacing above every line), paragraphs with auto leading may include lines with different leading when fonts and font sizes are mixed in the same paragraph.

Auto leading starts with a base amount of leading, which the application calculates by examining the ascent and descent values built into the fonts used in an auto-leaded line and the line above it; however, the user-specified text size plays the largest part in determining this base amount. Finally, a value specified by the user in the **Auto Leading** field is added to the base amount to arrive at the total amount of leading.

To specify percentage-based auto leading, enter a value from 0% to 100% in 1% increments. This value determines the amount of leading between two lines of text as follows: The largest font size in the line above is multiplied by the percentage value. This outcome is added to the base amount of auto leading between the two lines. Although the design of certain fonts complicates the process, here is a simplified example: 10-point text styled consistently in a "standard" font with **Auto Leading** set to 20% has 12 points of leading (10 pts + [20% of 10] = 12 pts).

Use the **Maintain Leading** check box to control the placement of a line of text that falls immediately below an obstruction in a column or box. If **Maintain Leading** is checked, the line's baseline is placed according to its applied leading value. If **Maintain Leading** is unchecked, the ascent of the line will abut the bottom of the obstruction or any applied runaround value.

In the **Lock to Grid Based On** area:

- Click **Ascent and Descent** to lock text to grid based on the ascenders and descenders of characters.
- Click **Font Size (Em Box)** to lock text to grid based on the size of the em boxes of the characters.

Preferences — Character

The **Character** pane (**Administration > Preferences > Renderer > Character**) includes the following controls.

Use the **Superscript** fields to control the placement and scale (size) of superscript characters. The **Superscript Offset** value determines how far below the baseline the application places a superscript character. The **Superscript Offset** value is measured as a percentage of font size. The default value is 33%. The **Superscript VScale** value determines the vertical size of the character and is a percentage of font size. The **Superscript HScale** value determines width and is a percentage of the normal character width (as specified by the font designer). The default value for both scales is 60% (range = 0 to 100%, measurement system = percentage, smallest increment = .1).

Use the **Subscript** fields to control the placement and scale (size) of subscript characters. The **Subscript Offset** value determines how far above the baseline the application places a subscript character. The **Subscript Offset** value is measured as a percentage of font size. The default value is 33%. The **Subscript VScale** value determines the vertical size of the character and is a percentage of font size. The **Subscript HScale** value determines width and is a percentage of the normal character width (as specified by the font designer). The default value for both scales is 100% (range = 0 to 100%, measurement system = percentage, smallest increment = .1).

Use the **Small Caps** fields to control the scale of characters with the **Small Caps** type style applied to them. The **Small Caps VScale** value determines the vertical size of the character and is measured as a percentage of font size. The **Small Caps HScale** value determines width and is measured as a percentage of the normal character width (as specified by the font designer). The default value for both scales is 75% (range = 0 to 100%, measurement system = percentage, smallest increment = .1).

Use the **Superior** fields to control the scale of superior characters. The **Superior VScale** value determines the vertical size of the character and is measured as a percentage of font size. The **Superior HScale** value determines width and is measured as a percentage of the normal character width (as specified by the font designer). The default value for both scales is 60% (range = 0 to 100%, measurement system = percentage, smallest increment = .1).

Use the **Ligatures Break Above** field to use ligatures built into a font. A ligature is a typographic convention in which certain characters are combined into a single glyph. Most fonts contain ligatures for the characters “f” followed by “i” and “f” followed by “l”. The **Ligatures Break Above** field enables you to specify the kerning or tracking value (measured in 1/200 em space increments) above which characters will not be combined into ligatures. For example, a headline with a large tracking value would probably not contain ligatures. The default value is 1 (range = 0 to 10, measurement system = .005 [1/200] em space,

smallest increment = .001). To prevent the second two letters in “ffi” and “ffl” (as in office and waffle) from being combined into ligatures, check **Not “ffi” or “ffl”**. Three-character ligatures for these combinations, common in traditional typesetting systems, are not standardized in fonts designed for macOSmacOS, so some typographers prefer to keep all three letters separate rather than combine only two of them. Note that many PostScript fonts do not have “ffi” and “ffl” ligatures, but most OpenType fonts do. This option is unchecked by default.

Check **Auto Kern** to specify that the application should use kerning tables, which are built into most fonts, to control intercharacter spacing. The **Auto Kern Above** field enables you to specify the point size above which automatic kerning must be used. The **Auto Kern Above** feature also implements custom tracking information specified in the **Tracking Values** dialog box for a selected font (**Utilities > Edit Tracking**) in QuarkXPress. This option is checked by default, with a 4-point threshold (range = 0 to 72 pt, measurement system = various [“, pt, cm, etc.], smallest increment = .001).

Check **Standard Em Space** to specify an em-space equivalent to the point size of the text (for example, 24pt text has a 24pt em space). If **Standard Em Space** is unchecked, the application uses the width of the two zeros in the current font as the em-space width. This option is checked by default. You can insert an em space in text by pressing Option+space/Ctrl+Shift+6.

Use the **Flex Space Width** field to change the 50% default width of a flexible space. To create a breaking flexible space, press Option+Shift+space/Ctrl+Shift+5; to create a nonbreaking flexible space, press Command+Option+Shift+space/Ctrl+Alt+Shift+5. The **Flex Space Width** value is expressed as a percentage of the normal en space for a given font and font size (range = 0 to 400%, measurement system = percentage, smallest increment = .1).

Use the **Accents for All Caps** check box to specify whether to include accent marks on accented characters with the All Caps type style applied. This option is checked by default.

Use the **Space between CJK & R** field to indicate how much space should be included by default between a Chinese, Japanese, or Korean character and an adjacent Roman character.

Preferences — Trapping

The **Trapping** pane (**Administration > Preferences > Renderer > Trapping**) includes the following controls.

Select a **Trapping Method**:

- Click **Absolute** to trap using the values in the **Auto Amount** and **Indeterminate** fields according to the object and background colors involved. If the object color is darker, the object is choked by the background using the

Auto Amount value. If the object color is lighter, the object is spread into the background using the **Auto Amount** value.

- Click **Proportional** to trap using the value in the **Auto Amount** field multiplied by the difference between the luminance (lightness or brightness) of the object color and the background color.
- Click **Knockout All** to turn off trapping and print objects with a zero trap amount.

Check **Process Trapping** to trap each process separation plate individually when a page contains overlapping process colors.

Check **Ignore White** to specify that an object color in front of multiple background colors (including white) does not take white into account when trapping.

Enter a trapping value in the **Auto Amount** field or choose **Overprint**:

- Enter a value in the **Auto Amount** field to control the amount of trapping that QuarkXPress applies to object and background colors that have an **Auto Amount** specified in the **Trap Specifications** dialog box (**Edit > Colors > Edit Trap**), and to control the amount of trapping applied to items with an **Auto Amount (+) Trap Information** or **(–)** specified in the **Trap Information** palette (**Window > Trap Information**).
- Choose **Overprint** to cause overprinting for object and background colors with an **Auto Amount** specified in the **Trap Specifications** dialog box (**Edit > Colors > Edit Trap**), as well as for items with an **Auto Amount (+)** or **(–)** specified in the **Trap Information** palette (**Window > Trap Information**).

Enter a trapping value in the **Indeterminate** field or choose **Overprint**:

- Enter a value in the **Indeterminate** field to control the amount of trapping that QuarkXPress applies to object colors that are in front of indeterminate backgrounds (multiple colors with conflicting trapping relationships).
- Choose **Overprint** to cause an object color to overprint an indeterminate background.

Enter a **Knockout Limit** value. The knockout limit is the value (expressed as a percentage of darkness of the object color) that enables you to control the point at which an object color knocks out a background color.

Enter an **Overprint Limit** value. Overprint limit is a trapping setting that allows an object set to overprint to trap according to the **Auto Amount** value if the object's shade is less than a particular percentage.

Preferences — Color Manager

The **Color Manager** pane (**Administration > Preferences > Renderer > Color Manager**) includes the following controls.

To specify an engine for color transformation, choose an option from the **Color Engine** drop-down menu.

To achieve the darkest possible blacks in all output methods, check **Black Point Compensation**.

Use the **Source Setup** drop-down menu to specify the source color space of pictures and colors used in the application.

To enable the **Profile Information** command in the **Window** menu and the **Color Management** tab in the **Import Picture** dialog box, check **Enable Access to Picture Profiles**. This option allows you to view information about profiles.

To specify a default proof output setup, choose an option from the **Proof Output** drop-down menu.

To specify a rendering intent for soft proofing, choose an option from the **Rendering Intent** drop-down list. **Perceptual** scales all the colors in the source gamut so that they all fit within the destination gamut. **Relative Colorimetric** retains colors that are in both the source gamut and the destination gamut. The only source colors that are changed are those that are not within the destination gamut. **Saturation** considers the saturation of source colors and changes them to colors with the same relative saturation in the destination gamut. **Absolute Colorimetric** retains colors that are in both the source gamut and the destination gamut. Colors that are outside the destination gamut are adjusted in relation to how they would look when printed on white paper. **Defined by Sources** uses the rendering intents defined in source setup for all colors and images.

To color manage vector content in imported EPS and PDF files, check **Color Manage Vector EPS/PDF**. Note that this preference applies only to EPS and PDF files imported after this box is checked.

To color manage vector content in EPS and PDF files that have already been imported in the active project, check **Include Existing Vector EPS/PDF in Layout**.

Preferences — Layers

The **Layers** pane (**Administration > Preferences > Renderer > Layers**) has the following controls.

To make new layers visible by default, check **Visible**.

To suppress the printout of new layers by default, check **Suppress Output**.

To make new layers locked by default, check **Locked**.

To maintain runaround on new layers so that text on visible layers flows around items on hidden layers, click **Keep Runaround**.

Preferences — Kindle

Use the **Kindle** pane (**Administration > Preferences > Renderer > Kindle**) to specify the location of the KindleGen tool, which is required for Kindle output.

To get a free copy of KindleGen, visit *https://kdp.amazon.com/self-publishing/help?topicId=A3IWA2TQYMZ5J6*.

Preferences — Modifier

Use the **Modifier** pane (**Administration > Preferences > Renderer > Modifier**) to control whether and where errors are displayed in rendered layouts.

To include descriptions of rendering errors in the layout, check **Annotate errors in the Output Document**. (For more information, see "*annotateerrors*.") In rendered QuarkXPress files, errors are displayed as notes. In rendered PDFs, errors are displayed as comments. In XML output, errors are displayed as notes XML markup.

To append descriptions of rendering errors after the last page, check **Append errors into the Output Document**. (For more information, see "*appenderrors*.") Descriptions of rendering errors are formatted in 10-point magenta Arial.

Job Jackets dialog box

The **Job Jackets** dialog box lets you edit the Job Jackets file used by QuarkXPress Server. To edit the QuarkXPress Server Job Jackets file:

1. In the QuarkXPress Server Web interface, choose **Administration > Job Jackets**. The **Manage Job Jacket** dialog box displays.
2. Click the **Get Job Jacket for editing** button and save the Job Jackets file to the desktop.
3. Open the downloaded Job Jackets file in QuarkXPress and make any necessary changes to the **QXPSJobTicket** Job Ticket.
4. In the **Job Jacket** dialog box, click **Choose File** and select the modified Job Jackets file.
5. Click **Submit**. The QuarkXPress Server Job Jackets file is replaced with the modified version.

App Studio preferences

The **AppStudio** dialog box lets you specify credentials and proxy settings so that QuarkXPress Server can upload HTML5 App Studio articles to the App Studio Publishing Portal. Enter your App Studio Publishing Portal user name and password, then enter the proxy settings for your proxy server (if any).

Check Out License dialog box

To check the QuarkXPress Server license out of Quark License Administrator, choose **Administration > Check Out License**. To specify the number of days for checkout, enter a value in days in the **Check out for** field. To be warned in advance of license expiration, check **Warn me** and use the corresponding fields.

Using QuarkXPress Server

The xml namespace deconstructs a project according to the Modifier schema. The construct namespace lets the server turn an XML representation of a QuarkXPress project back into an actual project. With these namespaces, you can deconstruct a project into an XML representation, change the XML in accordance with the Modifier schema, and then have the server generate an updated version of the QuarkXPress project. You can even create new QuarkXPress projects from scratch using XML.

In addition, you can use the construct namespace to:

- Create a page based on master page
- Create a project from XML, using a Job Jackets™ file as the basis for the project
- Modify text font and style, including OpenType® styles
- Apply style sheets and local formatting to text
- Create and populate tables
- Import pictures into picture boxes and specify picture attributes

The schema used for XML construction and deconstruction is completely Unicode®-compliant, making it ideal for use in international publishing. Furthermore, the use of this schema ensures that the schema of XML output created by Constructor does not change when server preferences change. This DTschema is provided in the QuarkXPress Server application folder and fully documented in "*Modifier schema (annotated)*."

➡ Deconstructor XTensions software and the deconstruct namespace are no longer supported.

Creating URL requests

You can use URL requests to make QuarkXPress Server render projects in a variety of formats, to use the features of server XTensions modules, and to control the server. The topics below provide an overview of how to construct server requests and use URL parameters.

This chapter also lists functions that let you control the server.

Understanding URL requests

QuarkXPress Server URL requests should use the following format:

http://*server:port/namespace/path/projectname?parameter=value*

- server: Indicates the name or IP address of the QuarkXPress Server computer.
- port: Indicates the QuarkXPress Server application's port number. The default port number is 8080.
- namespace: Sets the render type (or indicates another server functionality to access). For more information, see "*Understanding QuarkXPress Server namespaces.*"
- path: Indicates the path to the directory where the target project file is stored. The project to be rendered can either be located in the document pool (in which case paths are evaluated relative to the document pool directory), or can be streamed as part of a multipart HTTP Post request. When the project is streamed as part of the request, the project name will correspond to the name given to the HTTP request part which contains the project data.
- projectname: Identifies the project to be rendered.
- parameter=value: Optional parameters that provide more detailed control over how the target project should be rendered. Multiple parameter/value pairs, separated by the "&" character, can be included.

For example, the following URL asks the QuarkXPress Server application named "QXPServer" to return the file "MyProject.qxp" as a PDF file with hyperlinks and all fonts embedded:

http://QXPServer:8080/pdf/MyProject.qxp?includehyperlinks=1&embedallfonts=1

Some URL parameters require Boolean arguments. For such parameters, valid values include 1 or 0, true or false, y or n, and yes or no.

➡ You can also send requests to QuarkXPress Server using the HTTP GET and POST protocols and using XML with XSLT.

Understanding QuarkXPress Server namespaces

QuarkXPress Server namespaces differentiate among types of requests that are otherwise identical. For example, consider the following three URLs:

http://QXPServer:8080/project1.qxp

http://QXPServer:8080/*pdf*/project1.qxp

http://QXPServer:8080/*postscript*/project1.qxp

These requests are identical, except each uses a different namespace (in italic). (The first request does not specify a namespace, but this simply means the project is to be rendered using the server's default render type.)

Namespaces can be used to determine the format in which a rendered project is returned, as indicated above, but they can also be used to direct a request to

XTensions software that performs other functions. For example, if you use Modifier XTensions software's xml namespace, Modifier XTensions software can return an XML representation of the project.

Looking up a namespace

This guide lists the namespaces for every QuarkXPress Server function. There is no single list of namespaces because some functions do not require a particular namespace or are available in multiple namespaces. To determine which namespace you want to use:

1. In this Guide, go to the page that documents the render type you want to use. (For more information, see "*Understanding render types*.")
2. Locate the **Namespace** row. If the render type or function has an associated namespace, that namespace is listed here.

➡ Third-party XTensions can add their own namespaces. For information about a third-party namespace, see the documentation for the XTensions module that adds that namespace.

Understanding QuarkXPress Server parameters

Parameters let you control the details of how a request is executed. For example, you can use the page parameter to create a request that returns only the third page of a project:

http://QXPServer:8080/jpeg/project1.qxp?page=3

You can include multiple parameters in the same request; simply separate them with an ampersand (&). For example, here's a new version of the above URL that returns page three at a scale of 50%:

http://QXPServer:8080/jpeg/project1.qxp?page=3&scale=.5

Looking up a parameter

This Guide lists the parameters that are available for every QuarkXPress Server function. To determine which parameters you can use with a request:

1. In this Guide, go to the page that documents the render type you want or the function you want to use. (For more information, see "*Understanding render types*" and "*Understanding render modifiers*.")
2. Locate the **Parameters** row. This row lists all available parameters, and includes a description and a list of valid values for each parameter.

Supported interfaces

The following interfaces are available in QuarkXPress Server:

- **HTTP**: Lets you interact with the server using URLs that contain calls or point to XML files that contain calls. You can write client applications in any language that supports HTTP requests. For more information, see "*Getting started: HTTP*".
- **HTTPS**: Provides secure HTTP access.
- **Web services**: Lets you interact with the server via Web services using the QuarkXPress Server Manager object model. You can write client applications in Java, .NET, or any other programming language that can consume SOAP-based Web services. For more information, see "*Getting started: Web services*".

➡ To develop a custom load balancer or a custom application in Java, you must have version 1.5 or 1.6 of the JDK.

The Dynamic Publishing Process (DPP)

The Dynamic Publishing Process (DPP) has several stages. You may not need to use all of these stages every time, but this the order in which they occur:

- *Pre-Processing Stage*: During this stage, QuarkXPress Server performs any necessary initial steps, such as creating style sheets, colors, and H&J rules for a new QuarkXPress project.
- *Content Loading Stage*: During this stage, QuarkXPress Server loads dynamic content into boxes in the project.
- *Layout Modification Stage*: During this stage, QuarkXPress Server modifies the layout of the project.
- *Post-Processing Stage*: During this stage, QuarkXPress Server examines the project and performs maintenance tasks.

Getting started

The topics below describe how to create requests for the QuarkXPress Server Web interface.

➡ For information about the options available in such requests, see "*Using the Web interface*."

Getting started: HTTP and HTTPS

You can submit HTTP and HTTPS requests to QuarkXPress Server as URLs, either manually from a browser or automatically from an HTTP client application. QuarkXPress Server processes such requests and returns rendered content in the HTTP or HTTP responses. Depending on the type of request, the QuarkXPress Server preferences, and the type of content returned, the rendered content may be downloaded by the end user, displayed in the end-user's browser, or saved to a file system location accessible to QuarkXPress Server.

You can write a QuarkXPress Server client application in almost any language that can generate HTTP GET/POST requests. A QuarkXPress Server HTTP-based solution typically consists of QuarkXPress Server (running on a server computer connected to a network) plus a front-end application (usually Web-based) that provides a graphical user interface (GUI) for end users. The front-end application translates end users' input into HTTP or HTTPS requests and sends the requests to QuarkXPress Server or QuarkXPress Server Manager, which processes the requests and returns rendered content.

Dissecting a QXP Server URL

To interact with QuarkXPress Server from a Web browser, use a URL like the following:

http://[server]:[port]/[namespace]/[directory]/[DocumentName]?[parameter]=Value

- [server]: The name or IP address of the computer for QuarkXPress Server or QuarkXPress Server Manager.
- [port]: The port number on which to contact QuarkXPress Server or QuarkXPress Server Manager. The default port is 8080 for QuarkXPress Server and 8090 for QuarkXPress Server Manager.
- [namespace]: Defines what the URL action will be and any parameters and conditions available to that namespace.
- [directory]: The path in the document pool where the project is stored, relative to the QuarkXPress Server document pool. To access the root level, no directory path is necessary. (Note that you can also supply assets as part of a multipart HTTP request. For more information, see "*Using HTTP POST with QXP Server*.")
- [DocumentName]: The name of the QuarkXPress project to be processed.
- [parameter]: Further defines the URL action with attributes and values allowed for the namespace or general call. Pass parameters in the form attribute=value, with parameters separated by the "&" character.

For QuarkXPress Server Manager, use a URL like the following:

http://[server]:[port]/qxpsm/request/[namespace]/[directory]/[DocumentName]?[parameter]=Value

- Prior to QuarkXPress Server 9.0, you had to use different URL constructions when sending requests to an instance of QuarkXPress Server Manager in a QPS installation than you did when sending requests to a free-standing instance of QuarkXPress Server Manager. In versions 9.0 and later, both can use /qxpsm/request/ after [port]/ .
- You can now use both absolute and relative paths when you modify a project with SDK objects or classes. Relative paths are almost always relative to the document pool. If you use multiple QuarkXPress Server instances, you should use a common document pool.

Interpreting the QXP Server Manager response

When QuarkXPress Server Manager successfully processes a request through the HTTP interface, the response is the same as QuarkXPress Server's response unless the user has supplied additional parameters to QuarkXPress Manager. For more information, see "Working with QuarkXPress Server Manager" in *A Guide to QuarkXPress Server*.

If an error occurs, QuarkXPress Server Manager retries the request, either on the same QuarkXPress server instance or a different one (depending on the error and global settings established in the QuarkXPress Server Manager client). If QuarkXPress Server Manager cannot process the request, it returns an XML response describing the error, plus any header error codes returned by QuarkXPress Server. For example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<error>
  <httpresponsecode>500</httpresponsecode>
  <xpressservererrorcode>-43</xpressservererrorcode>
  <xpressservererrormessage>File not
found.</xpressservererrormessage>
  <xpressserverextendedmessage> <![CDATA[ Error #-43 - File not
found. ]]>
    </xpressserverextendedmessage>

<xpressservermanagererrorcode>M8000001</xpressservermanager
errorcode>
  <xpressservermanagererrormessage>The server could not locate
the specified file.
    </xpressservermanagererrormessage>
</error>
```

HTTP GET and POST Requests

The topics below describe how you can use HTML to interact with QuarkXPress Server.

- QuarkXPress Server supports both the GET and POST HTML methods. When you use the GET method, the browser encodes form data into a URL. When you use the POST method, form data is passed within the message body. Use the GET method only when the form processing is idempotent. In short: GET is for retrieving data, whereas POST can involve storing or updating data, ordering a product, or sending an e-mail.

Using HTTP GET with QXP Server

Use HTML like the following to specify a server and port where you want to send a request. You can specify the name of the target project, the output type, and a scaling value. You can specify the name of a box and the path of a text or picture files to import into that box, as long as the file's path is on the server's file system. You can also use HTML like the following to specify the page number and layout number of the project.

The form section of the HTML should begin with the following line of code:

```
<form id = form1 method="GET" enctype="application/x-www-form-urlencoded">
```

For both GET and POST, the browser constructs a form data set and encodes it according to the ENCTYPE attribute (you can use multipart/form-data for POST and application/x-www-form-urlencoded (the default) for both POST and GET).

To create fields that let the user specify the server IP address, the port, and the project name, use HTML like the following:

```
<TABLE cellSpacing=1 cellPadding=1 border=1 id=TABLE1 >
 <TBODY>
 <TR>
  <TD>
  <INPUT id=ServerTxt name=ServerTxt value="Server ID"
   readOnly size=13 style="WIDTH: 107px; HEIGHT: 22px">
  </TD>
  <TD>
  <INPUT id=Server maxLength=50 size=16 value=localhost
name=Server
    style="WIDTH: 170px; HEIGHT: 22px">
  </TD>
 </TR>
 <TR>
  <TD>
  <INPUT id=PortTxt name=PortTxt value="Port Number"
   readOnly size=13 style="WIDTH: 107px; HEIGHT: 22px">
```

```
    </TD>
    <TD>
    <INPUT id=Port maxLength=50 size=17 value=8080 name=Port
     style="WIDTH: 170px; HEIGHT: 22px">
    </TD>
   </TR>
   <TR>
   <TBODY>
</TABLE>
<TR>
  <p></p>
  <TD>
  <INPUT id=DocTxt name=DocTxt value="Document Name"
   readOnly size=13 style="WIDTH: 107px; HEIGHT: 22px">
  </TD>
  <TD>
  <INPUT id=Doc maxLength=50 size=18 name=Doc style=
   "WIDTH: 170px; HEIGHT: 22px">
  </TD>
</TR>
```

To create a drop-down menu that lets the end user specify a render format, use HTML like the following:

```
<SELECT id="select1" name="returntype">
  <OPTION value="jpeg">JPEG</OPTION>
  <OPTION value="pdf">PDF</OPTION>
  <OPTION value="qxpdoc">QuarkXPress document</OPTION>
  <OPTION value="eps">EPS Document</OPTION>
  <OPTION value="postscript">POSTSCRIPT</OPTION>
  <OPTION value="png">PNG</OPTION>
</SELECT><td/>
```

To create a drop-down menu that lets the end user specify a rendering scale, use HTML like the following:

```
<SELECT id="select2" name="scale">
  <OPTION value="1">100%</OPTION>
  <OPTION value="2">200%</OPTION>
  <OPTION value="3">300%</OPTION>
  <OPTION value="5">500%</OPTION>
  <OPTION value=".5">50%</OPTION>
</SELECT><p/>
```

To create input fields that let the end user specify a box name and the name of a file to be imported into that box, use HTML like the following:

```
<TD>
<INPUT id=box1Txt value="Box Name"
  readOnly style="WIDTH: 181px; HEIGHT: 22px" size=16>
</TD>
```

```
<TD>
<INPUT id=box1 maxLength=256 size=43 style="
  WIDTH: 293px; HEIGHT: 22px"></TD>
</TR>

<TR>
<TR>
<TD>
<INPUT id=box1FileTxt value="File on Server"
  readOnly style="WIDTH: 181px; HEIGHT: 22px" >
</TD>

<TD>
<INPUT id=box1File maxLength=256 size=43 style="
  WIDTH: 293px; HEIGHT: 22px">
</TD>
</TR>
```

To create fields that let the end user enter a page number a layout number, use HTML like the following:

```
<TABLE cellSpacing=1 cellPadding=1 border=1 style="WIDTH: 188px;
HEIGHT: 61px">
  <TR>
    <TD>
    <INPUT id=PageTxt value = "Page"
      readOnly style="WIDTH: 50px; HEIGHT: 22px" size=3>
    </TD>
    <TD>
    <input id=Page size="16" maxlength="256"
      style="WIDTH: 147px; HEIGHT: 22px">
    </TD>
  </TR>
  <TR>
    <TD>
    <INPUT id=LayoutTxt value = "Layout"
      readOnly style  ="WIDTH: 50px; HEIGHT: 22px" size=4>
    </TD>
    <TD>
    <input id=Layout size="16" maxlength="256"
      style="WIDTH: 147px; HEIGHT: 22px">
    </TD>
  </TR>
</TABLE>
```

To create a button that lets the end user submit the request, use HTML like the following:

```
<input type="submit" value="Render document"
```

```
name="Submit" LANGUAGE="javascript"
onclick="return Submit_onclick()"/>
```

The above HTML calls a function named Submit_onclick(). You can add such a function to the <HEAD> section of the HTML. For example:

```
<head>
<TITLE>Quark Stream</TITLE>
<script ID="clientEventHandlersJS" LANGUAGE="javascript">
 function Submit_onclick() {
 var prefix;
 var renderer;
 var file;
 var url;
 var box1Name;
 var dataImportStamp = "@dataimport";
 prefix = "http://" + document.getElementById("Server").value + ":";
 port = document.getElementById("Port").value + "/";
 renderer = document.getElementById("select1").value + "/";
 file = document.getElementById("Doc").value;
 box1Name = document.getElementById("box1").value;
 if (box1Name != "") {
  document.getElementById("box1File").name = box1Name +
dataImportStamp;
  } else {
  document.getElementById("box1File").name = "";
  }
 document.getElementById("Page").name = "Page";
 document.getElementById("Layout").name = "Layout";
 url = prefix + port + renderer + file;
 document.getElementById("form1").action = url;
}
</script>
</head>
```

The Submit_onclick() function reads the values from the formand builds a request URL using the server, port, and render type.

- If the end user specifies a file name in the “File on Server” text box, he or she must add file: to the beginning of the file path (for example, file:C:\data.txt).
- The code above adds @dataimport to the end of the box name to accommodate data import.

The action of the form is defined by this line:

```
document.getElementById("form1").action = url;
```

This form's method is GET. The user agent gets the value (the URL) of the action, appends a ? to it, adds the form data set, and submits the URL.

- In this scenario, form data must be in ASCII.

Using HTTP GET with QXP Server Manager

HTTP GET with QuarkXPress Server Manager works the same way as HTTP GET with QuarkXPress Server (see "*Using HTTP GET with QXP Server*"), except that Quark does not recommend using GET if you are working with non-ASCII characters. The behavior of GET requests with characters is highly browser-dependent, and there is no standard that all browsers follow. Instead, use POST.

Using HTTP POST with QXP Server

Use HTML like the HTML in "*Using HTTP GET with QXP Server*" to specify a server and port where you want to send a request. You can specify the name of the target project, the output type, and a scaling value. You can specify the name of a box and the path of a text or picture files to import into that box, as long as the file's path is on the server's file system. You can also use HTML like the HTML in "*Using HTTP GET with QXP Server*" to specify the page number and layout number of the project. Differences between the GET method and the POST method are described below.

The form section of the HTML should begin with the following line of code:

```
<form id = form1 method="post" enctype="multipart/form-data">
```

The following HTML creates input fields that let the end user specify the name of a file to be imported into a box:

```
<TD><INPUT id=box1FileTxt value="File on Client"
   readOnly style="WIDTH: 180px; HEIGHT: 22px" ></TD>
<TD><input id=box1File type="file"
  size="32" maxlength="256" style="WIDTH: 293px;
  HEIGHT: 22px">
</TD></TR>
```

The action of the form is defined by this line:

```
document.getElementById("form1").action = url;
```

The form's method is POST. The user agent conducts an HTTP post transaction using the value of the action attribute (the URL), and a message is created according to the content type specified by the enctype attribute.

When you use a multipart HTTP post request, you can include in the request any files which are required by the rendering process, including QuarkXPress templates, picture files, modifier XML, and digital publishing assets. For more information, see "*Using the Streaming Document Provider*."

Using HTTP POST with QXP Server Manager

HTTP POST with QuarkXPress Server Manager works the same way as HTTP POST with QuarkXPress Server (see "*Using HTTP POSTwith QXP Server*"), except that with QuarkXPress Server Manager, you must use UTF-8.

Getting started: Web services

The Web services interface is a collection of request classes. You can download the SDK WSDL class definitions from http://[server]:[port]/qxpsm/services/RequestService?wsdl (replace [server] with the QuarkXPress Server Manager computer's IP address and [port] with the QuarkXPress Server Manager port number.

These classes can be chained together to form compound QuarkXPress Server requests. The sample applications (see "*Sample Applications*") show how to use these classes to invoke a QuarkXPress Server command and manipulate the response.

For more information, see "*Functions*." In addition to the classes listed there, the Web services interface includes the following:

- RequestService processes QuarkXPress Server requests. This object's generic processRequest() method takes a QRequestContext argument and returns a QContentData object containing the response. For more information, see the sample applications and "*Functions*."
- QRequestContext is the argument you pass to RequestService's generic processRequest() method.This object contains settings which must be set once per request. Set all chained requests inside the request context.
- QRequest is the base class for all request objects (such as PDFRenderRequest). Consequently, all request objects share some common data members.
- RequestParameters is a generic class for executing any request and for adding dynamic properties to a request.
- NameValueParam is a generic class for adding dynamic properties to a request. This class is specifically for requests that take a box's name and/or ID as the parameter name and the box's content as the value.
- QContentData is the response returned when a request is executed. QContentData is a hyperlink that follows the same pattern as the classes above.
- QException is the exception class for QuarkXPress Server Manager. Web services returns a QException object if an error occurs with any Web service method. You can use try/catch blocks to handle QException objects.

If you've written a Server XTensions module, you can extend the XML interface to include any changes it makes to the Modifier schema by simply modifying an XML file and regenerating the stubs.

➡ To exclude empty tags in the request HTML, set the value of the appropriate variable to null.

➡ For Javadocs, WSDL schemas, and JSP samples, see the Welcome page that displays when you launch QuarkXPress Server Manager.

The following topics describe the general Web services classes.

QRequestContext

Description	An argument passed to RequestService. Contains settings that must be set once per request. All chained requests must be set inside the request context.		
Type	Web service data object		
Members	Name	Types	Description
	documentName	String	File or object name on which the command will be rendered.
	serverName	String	Server name. Default is NULL. Load balancer searches for the host itself in this case.
	serverPort	Integer	Port at which the desired server is listening.
	userName	String	Server admin username.
	userPassword	String	Server admin password.
	maxRetries	Integer	Max number of times to try executing the command before returning failure.
	requestTimeout	Integer	Max time out in milliseconds.
	useCache	Boolean	Indicates whether the cache should

			be checked for an existing result or if the command should be executed again.
	responseAsURL	Boolean	This value indicates whether the server should send the response as-is (text or binary) or store the response on the server and return its location as a URL. Because the object model works on SOAP, which can be slow when transferring large binary files, you might choose to set this value to “true” if you suspect that the response is going to be several megabytes or larger.
	bypassFileInfo	Boolean	Indicates whether file info should be fetched before executing the command.
	context	String	Context in which the command is being executed.

	request	QRequest	QuarkXPress Server request is instances of request objects chained together.
Example, object model	com.quark.qxpsm.QRequestContext rc = new com.quark.qxpsm.QRequestContext(); rc.documentName = this.DocumentSettings1.documentName.Text; rc.responseAsURL = this.DocumentSettings1.responseAsURL.Checked; rc.useCache = this.DocumentSettings1.useCache.Checked; rc.bypassFileInfo = this.DocumentSettings1.bypassFileInfo.Checked; //Create the service and call it QRequestContext object RequestService svc = new RequestService(); com.quark.qxpsm.QContentData qc = svc.processRequest(rc);		

RequestService

Description	Web service called to process the QuarkXPress Server request. RequestService has a generic method named processRequest() that takes QRequestContext as an argument and returns QContentData as the QuarkXPress Server response.
Type	Web service

<table>
<tr><td rowspan="11">Methods</td><td>processRequest</td><td colspan="3">Processes the request context and returns the result.</td></tr>
<tr><td></td><td>Parameter</td><td>Type</td><td>Description</td></tr>
<tr><td></td><td>requestCmd</td><td>QRequestContext</td><td>Argument passed to RequestService. Contains settings that must be set once per request. All chained requests are set inside the request context.</td></tr>
<tr><td>createSession</td><td colspan="3">Creates a new session and returns a session ID.</td></tr>
<tr><td></td><td>Parameter</td><td>Type</td><td>Description</td></tr>
<tr><td></td><td>timeout</td><td>Long</td><td>Timeout for the session in milliseconds. If no call is executed in that time, session is expired and all the open documents in that session are closed without saving. If 0 is passed as value of timeout, default timeout is used. If a negative value is passed as timeout, the session never expires.</td></tr>
<tr><td>closeAlldocs</td><td colspan="3">Closes all open documents in the session without saving them. If the session does not exist, an error is returned. If an error occurs while closing the document, it is logged and the document is marked closed in the internal cache. No error is returned.</td></tr>
<tr><td></td><td>Parameter</td><td>Type</td><td>Description</td></tr>
<tr><td></td><td>sessionId</td><td>String</td><td>Session whose documents are to be closed.</td></tr>
<tr><td>closeDoc</td><td colspan="3">Closes the specified document without saving it. If</td></tr>
</table>

<table>
<tr><td></td><td></td><td>the session does not exist, an error is returned. If the document is not open, and error is returned. If the document is open in another session, an error is returned. If an error occurs while closing the document, it is logged and the document is marked closed in the internal cache. No error is returned.<table><tr><td>Parameter</td><td>Type</td><td>Description</td></tr><tr><td>docName</td><td>String</td><td>Document to be closed.</td></tr><tr><td>sessionId</td><td>String</td><td>Session in which document was opened.</td></tr></table></td></tr>
<tr><td></td><td>closeSession</td><td>Closes the specified session. If the session does not exist, an error is returned. If any documents are still open in the session, an error is returned.<table><tr><td>Parameter</td><td>Type</td><td>Description</td></tr><tr><td>sessionId</td><td>String</td><td>Session to be closed.</td></tr></table></td></tr>
<tr><td></td><td>getOpenDocs</td><td>Gets all the open documents in the session. If the session does not exist, an error is returned.<table><tr><td>Parameter</td><td>Type</td><td>Description</td></tr><tr><td>sessionId</td><td>String</td><td>Session whose open documents are sought.</td></tr></table></td></tr>
<tr><td></td><td>getOpenSessions</td><td>Gets all open sessions.</td></tr>
<tr><td></td><td>getPreferences</td><td>Gets QuarkXPress Server preferences.</td></tr>
<tr><td></td><td>setPreferences</td><td>Sets QuarkXPress Server preferences.</td></tr>
<tr><td></td><td>getXPressDOM</td><td>Creates a DOM for the specified document.</td></tr>
<tr><td></td><td>newDoc</td><td>Creates a new document for modification and keeps it open until further notice. The document is created with a single layout. To create a more complex document, use the processRequestEx API. If a document with the same name is already open, an error is returned. If the session does not exist, an error is returned.</td></tr>
</table>

	Parameter	Type	Description
	docName	String	Document to be opened for modification. Provide the name only. You can provide a relative path when you save the document.
	jobJacketName	String	Name of the Job Jackets file to be used. The Job Jackets file is assumed to be already available on the QuarkXPress server computer.
	jobTicketName	String	Name of the Job Ticket to be used.
	host	String	The QuarkXPress Server instance that should be used for this document modification. If null, this value is supplied by the load balancer. If the indicated server is not an active registered server, an error is thrown.
	port	Integer	The port for the server specified in the host parameter.
	sessionId	String	Session in which the document should be opened.
openDoc	Opens the specified document and keeps it open until further notice. If the document is already open, an error is returned. If the session does not exist, an error is returned.		

	Parameter	Type	Description
	docName	String	Document (along with relative path if required) to be opened for modification.
	host	String	QuarkXPress Server instance which should be used for this document modification. If null, this value is supplied by the load balancer. If the indicated server is not an active registered server, an error is thrown.
	port	Integer	The port for the server specified in the host parameter.
	sessionId	String	Session in which the document should be opened.
processRequestEx	Executes the request context. If a session ID is specified, the document is kept open after the request is executed. If no session ID is specified, the request is executed normally without keeping the document open. If the document is open in another session, an error is returned. If the document is marked dirty, an error is returned (a document is marked dirty when the server that opened the document has become inactive; in such a case, the document must be closed and opened again).		
	Parameter	Type	Description
	reqContextObj	QRequestContext	Request to be executed.
	sessionId	String	Session in which the

<table>
<tr><td></td><td></td><td></td><td>request should be executed. This value may be null. If a session ID is provided, the document is kept open. If no session ID is provided, the request is executed normally, as if processRequest had been called.</td></tr>
<tr><td></td><td>saveAllDocs</td><td colspan="3">Saves all open documents in the session. The documents are saved one by one. If error occurs while saving a document, an error is returned immediately and the rest of the documents remain unsaved. If a document is marked dirty, an error is returned (a document is marked dirty when the server that opened the document has become inactive; in such a case, the document must be closed and opened again).
<table><tr><th>Parameter</th><th>Type</th><th>Description</th></tr><tr><td>relativePath</td><td>String</td><td>Relative path where open documents should be saved. If this value is provided, copies of open documents with changes made so far are saved in the new location. The open documents are not saved but have all of the changes made so far.</td></tr><tr><td>sessionId</td><td>String</td><td>Session in which the document exists.</td></tr></table></td></tr>
<tr><td></td><td>saveDoc</td><td colspan="3">Saves the open document. If a document is marked dirty, an error is returned (a document is marked dirty when the server that opened the</td></tr>
</table>

<table>
<tr><td></td><td></td><td>document has become inactive; in such a case, the document must be closed and opened again).<table><tr><th>Parameter</th><th>Type</th><th>Description</th></tr><tr><td>docName</td><td>String</td><td>Document to be saved. Must be the same name that was used when opening or creating the document.</td></tr><tr><td>newName</td><td>String</td><td>New name of the document. If null, the document is saved with the old name.</td></tr><tr><td>relativePath</td><td>String</td><td>Relative path where the document should be saved. The relative path can also contain the new name of the document. If this is provided, a copy of the open document with changes made so far is saved in the new location. The open document is not saved but has all of the changes made so far.</td></tr><tr><td>sessionId</td><td>String</td><td>Session in which the document exists.</td></tr></table></td></tr>
<tr><td></td><td>getXPressDOMEx</td><td>Lets you create a DOM of a particular layout or portion of a layout.</td></tr>
<tr><td></td><td>getXMLFromXPressDOM</td><td>Creates an XML string out of the DOM.</td></tr>
<tr><td></td><td>getXPressDOMFromXML</td><td>Takes a raw XML representation of a project as a string and returns an object model representing that project, with Project as the root class.</td></tr>
<tr><td>Example, object</td><td colspan="2">QRequestContext rc = new QRequestContext(); rc.documentName = “test.qxp”; rc.responseAsURL = false; JPEGRenderRequest</td></tr>
</table>

model	jpegRequest = new JPEGRenderRequest(); rc.request = jpegRequest; RequestService svc = new RequestService(); QContextData response = svc.processRequest(rc);

QRequest

Description	Base class for all request objects (such as PDFRenderRequest). All request objects share some common data members, which are described below.		
Type	Web service data object		
Members	Name	Types	Description
	request	QRequest	QuarkXPress Server request that includes instances of request objects chained together.

RequestParameters

Description	Generic class for executing any request and for adding dynamic properties to a request.		
Type	Web service data object		
Members	Name	Type	Description
	requestNamespace	String	Namespace of the request (for example, jpeg).
	params	NameValueParam[]	Parameter array for the specified request (for example, jpegquality).
Additional comments	You can use this class to send any request for which a specific class does not exist. When this request exists in the chain, its namespace is concatenated with the namespaces of other requests. That means the namespace provided here can be null. The parameters of this class can		

	be used to parameterize a request being sent to the server.
Example, object model	QRequestContext rc = new QRequestContext(); RequestParameters request = new RequestParameters(); request.setRequestNamespace("jpeg"); rc.setRequest = request; NameValueParam p1 = new NameValueParam(); p1.setParamName = "jpegquality"; p1.setTextValue = "4"; request.setParams(new NameValueParam[]{p1});

NameValueParam

Description	Generic class for adding dynamic properties to a request. This class is specifically for requests that take a box name/id as the parameter name and the box content as the parameter value.		
Type	Web service data object		
Members	Name	Type	Description
	paramName	String	Name of the parameter. In most cases this will be the name/ID of the box.
	textValue	String	Text value of the box. (You can set either textValue or streamValue.)
	streamValue	byte[]	Stream value of the box. (You can set either textValue or streamValue.)
	contentType	String	The MIME content type of the parameter.

QContentData

Description	A response to a Web Services call to QuarkXPress Server.		
Type	Web service data object		
Members	Name	Types	Description
	contentType	String	The type of the response. For example, “text/xml” or “text/plain.”
	textData	String	If the response type is text, this contains the text. Otherwise, this value is null.
	responseURL	String	If the responseAsURL parameter was set to “true” in the request, this contains the URL of the response. Otherwise, this value is null.
	streamValue	binary	If the response type is binary, this contains the byte array. Otherwise, this value is null.
	encodingType	String	If the response type is text, this value indicates the encoding of the text (for example, UTF-8 or ANSI).
	actualServerPortUsed	String	Identifies the server port.

	actualServerUsed	String	Identifies the server.
	headers	String	If the response returned by the server is a set of headers, this array contains the header response.
	multipartResponse	String	If the response returned by the server is multipart, this array contains the multipart response parts.
Example, object model	QRequestContext context = new QRequestContext(); context.setDocumentName("sample.qxp"); context.setResponseAsURL(true); JPEGRenderRequest request = new JPEGRenderRequest(); request.setJPEGQuality("4"); context.setRequest(request); RequestService requestService = new RequestServiceStub(); QContentData response = requestService.processRequest(context); System.out.println(response.getResponseURL());		

QException

Description	Exception class for QuarkXPress Manager.		
Type	Exception		
Members	Name	Types	Description
	httpResponseCode	String	HTTP response code.
	managerErrorCode	String	QuarkXPress Server Manager error code.
	managerErrorMessage	String	QuarkXPress Server Manager

			localized error message.
	serverErrorCode	String	QuarkXPress Server error code.
	serverErrorMessage	String	QuarkXPress Server response message.
	serverExtendedMessage	String	QuarkXPress Server extended error message.
Example, object model	String docName = “notexisting.qxp”; try { QRequestContext ctx = getRequestContext(docName); QRequest request = getJPEGRequest(); ctx.setRequest(ctx); QContentData response = getService().processRequest(ctx); System.out.println(response.getResponseURL()); } catch (QException ex) { // QuarkXPress Manager threw an QException and it is not // a runtime exception. QException object will be returned. System.out.println(ex.getServerErrorCode()); }		

QXP Server Manager

The following topics are for people who want to enhance QuarkXPress Server Manager or integrate it with other software.

Please refer to http://localhost:8090/qxpsmdocs/apidocs/index.html for manager API documentation. (Note that the port number used to retrieve the API documentation is 8090 by default, but you should use whatever port number you specified when installing QuarkXPress Server Manager.)

QuarkXPress Server Manager was developed using interface-based programming and uses the Spring Framework to instantiate pluggable objects. When QuarkXPress Server Manager starts up, it reads the contents of a Spring context definition file named “ManagerContainerConfig.xml” and instantiates all of the beans listed in the file. QuarkXPress Server Manager then initializes by reading various configuration options from a file named “ManagerConfig.xml.”

You can deploy QuarkXPress Server Manager in its own Tomcat container, in an external Tomcat container, or in a shared Spring context. For more information, see “Deploying QuarkXPress Server Manager” in the *QuarkXPress Server ReadMe*.

Using the Web interface

The topics below describe the features available via the QuarkXPress Server Web interface. The topics covered here include the following:

- *Render types* are namespaces you can use to return a QuarkXPress project in a specified file format.
- *Render modifiers* let you control which parts of a project are rendered and set the scale of the returned renderings.
- *Content modifiers* let you alter the content and formatting of boxes in layouts without using the XML modify parameter.
- *XML modify* lets you modify QuarkXPress projects using XML.
- The xml namespace deconstructs a project according to the Modifier schema. The construct namespace lets you turn an XML representation of a QuarkXPress project back into a QuarkXPress project.
- *Administrative request handlers* let you change the behavior of QuarkXPress Server.

➡ QuarkXPress Server uses case-sensitive XML.

Understanding rendering

Rendering is the process in which QuarkXPress Server opens a QuarkXPress project, transforms it into a different format (the *render type*), and then sends a response to the requestor. Depending on the type of rendering operation, the response may be a message or a rendered file.

For information on how to submit a render request, see "*Getting Started*."

Alerts	Cannot open this document type. Please select a QuarkXPress document or template.	HTTP Error #500 This alert displays if you try to render a file that is not a QuarkXPress project.
	File not found	HTTP Error #404 QuarkXPress Server Error #–43 This alert displays if you try to render a project that does not exist.
	I/O error trying to	HTTP Error #500 QuarkXPress Server Error #–36 This alert displays if QuarkXPress Server is

	read or write to disk.	running on Windows and a shared network folder was selected as the document pool, but the folder is no longer shared. What to do: In the QuarkXPress Server administration interface, choose Administration > Preferences > General and set Document Root Folder to a shared folder.
	Cannot find required volume or folder.	HTTP Error #404 QuarkXPress Server Error #–35 This alert displays if QuarkXPress Server is running on macOS and a shared network volume was selected as the document pool, but the volume is no longer shared. What to do: In the QuarkXPress Server administration interface, choose Administration > Preferences > General and set Document Root Folder to a shared folder.
Logs	See “Understanding logging."	
Example, GET URL	http://localhost:8080/sample.qxp	
Notes	There are two ways to specify a render format:1. Enter the render type directly in the browser address field:http://localhost:8080/pdf/project.qxp.2. In the QuarkXPress Server administration interface, choose Administration > Preferences > General and choose the default render type from the Default Renderer Type drop-down menu.	

Understanding logging

If a request succeeds, a transaction success message is written to the QuarkXPress Server transaction log file. This message includes the transaction ID, date, time, request type, project name, response type, response size in bytes, and client IP address. For example:

```
07/03/2011 14:37:47 - RequestURI = /xml/sample.qxp TransactionUUID = afb6f457-80ae-4d5d-a434-ce9f3e089761  Client = 10.91.30.216  Type = text/xml  Size = 4846
```

If an alert is displayed, an error message is written to the QuarkXPress Server transaction log file. The transaction entry contains the date and time of the request, the error code, and the error message. The following is a sample of an error log transaction entry:

```
09/03/2011 13:54:33 - RequestURI = /sample.qcd TransactionUUID = dffc3a7e-11fd-4d97-b3fe-8f2129353d58  Client = 10.91.30.216 Error #10120 - Cannot open this document type. Please select a
```

QuarkXPress document or template.

The "QuarkXPress Server Log.log" file also contains system-level log information. For example, if a request makes a renderer stop working, you can figure out which request it was using the transaction ID and the transaction log.

09/03/2011 014:00:07 ERROR [com.quark.qxps.core.server.ServerRendererMonitor][pool-1-thread-1] - The QuarkXPress Server Renderer with processId 2620 had quit while processing the transaction 87212dae-6ba3-4b3f-97bb-ea8f0c255bf9.

➡ To download all logs to a non-server computer, click **Show Transaction Log** in the QuarkXPress Server Web interface, then click **Download Logs** on the upper right.

Understanding render types

Render types are namespaces you can use to return a QuarkXPress project in a specified file format. The topics covered here include the following:

Function	Description	QuarkXPress Server Manager object model classes
appstudio	Returns a .zip file containing an App Studio article rendered from the App Studio layout(s) in the source project.	AppStudioRenderRequest
ave	Returns a .zip file containing an AVE issue file and its corresponding manifest.	AVERenderRequest
eps	Returns an EPS file.	EPSRenderRequest
epub	Returns an ePUB file.	EPubRenderRequest
jpeg	Returns a JPEG image.	JPEGRenderRequest
PDF	Returns a PDF file.	PDFRenderRequest
png	Returns a PNG image.	PNGRenderRequest
postscript	Returns a PostScript file.	PostScriptRenderRequest
qcddoc	Returns a QuarkCopyDesk article.	CopyDeskDocRequest
qxpdoc	Returns a QuarkXPress project file.	QuarkXPressRenderRequest

Function	Description	QuarkXPress Server Manager object model classes
qxpr	Returns an RLE Raw Custom format image.	RLERawCustomRenderRequest
raw	Returns a project in a QuarkXPress internal format.	RawCustomRenderRequest
screenpdf	Returns a low-resolution PDF file.	ScreenPDFRenderRequest
swf	Returns a SWF file.	SWFRenderRequest
html5	Returns a .zip file containing HTML5 output.	HTML5RenderRequest

- The default render type is JPEG.
- Developers can implement additional rendering formats through server XTensions software.

appstudio

The appstudio render type returns a .zip file containing an App Studio article rendered from the App Studio layout(s) in the source project. It also includes mechanisms for uploading an article to the App Studio Publishing Portal, retrieving and updating server settings, and presenting and clearing App Studio Publishing Portal credentials.

Namespace	appstudio		
Parameters	upload	String	Lets you generate an HTML5 article and upload it to the App Studio Publishing Portal. For example:http://localhost:8080/appstudio/upload/template.qxp?organization=XXX&publication=YYY&

			issue=ZZZ&article=AAA
	html	String	Lets you generate an HTML5 article and returns it as a .zip file. For example:http://localhost:8080/appstudio/html/template.qxp
	setcredential	String	Lets you log in to the App Studio Publishing Portal. For example:http://localhost:8080/appstudio/setcredential?username=XXX&password=YYY
	clearcredential	String	Lets you log out of the App Studio Publishing Portal. For example:http://localhost:8080/appstudio/clearcredential
	config	String	Lets you retrieve the current publication hierarchy from the App Studio Publishing Portal. For example:http://qxpserver:port/appstudio/config

Render modifier parameters	layout	String	Lets you specify a layout by name or ID. The first layout is Layout 1.
	page	Integer	Lets you specify a page.
	pages	String (page range)	Lets you specify a range of pages.
Response	A .zip file containing an HTML5 article.		
Alerts	The renderer for this image type has no way of rendering the desired objects.	HTTP Error #406 This alert displays if you submit a render request with the box parameter.	
	Cannot open this document type. Please select a QuarkXPress document or template.	HTTP Error #10120This alert displays if you submit an appstudio request for a QuarkCopyDesk article.	
Logs	See “Understanding logging."		
Example, GET URL	http://localhost:8080/appstudio/sample.qxp		
Example, object model	Request object name: AppStudioRenderRequest // STEP 1: Create the QuarkXPress Server Request // Context and set the necessary properties com.quark.qxpsm.QRequestContext requestCtx = new com.quark.qxpsm.QRequestContext(); Boolean responseAsURL = false; requestCtx.setDocumentName(docName); // STEP 2 (SPECIFIC TO REQUESTS): // Create the App Studio renderer // request and embed it in the request context. AppstudioRenderRequest req = new AppstudioRenderRequest(); req.setAppStudioData(request.getParameter(“AppStudioData")); requestCtx.setRequest(req); // STEP 3: Create the service and call the // processRequest() API		

```
RequestService service = new RequestServiceStub();
com.quark.qxpsm.QContentData data =
service.processRequest(requestCtx);
```

ave

The ave render type returns a .zip file containing an AVE issue file and its corresponding manifest.

Namespace	AVE		
Parameters	outputstyle	stylename	Lets you specify an output style. To use a named output style, use the name of that output style. For example:http://localhost:8080/ave/sample.qxp?outputstyle=stylename To use settings that have been captured with the Capture Settings in the QuarkXPress Export AVE for iPad dialog box, use document. For example:http://localhost:8080/ave/sample.qxp?outputstyle=document
Render modifier parameters	format	String	Lets you specify an output format. Use avemag for

			AVE-Mag or avedoc for AVE-Doc. The default is avemag.
	layout	String	Lets you specify a layout by name or ID. The first layout is Layout 1.
	page	Integer	Lets you specify a page.
	pages	String (page range)	Lets you specify a range of pages.
Response	A .zip file containing an AVE issue file and its corresponding manifest.		
Alerts	The renderer for this image type has no way of rendering the desired objects.	HTTP Error #406 This alert displays if you submit a render request with the box parameter.	
	This Output Style does not exist.	This alert displays if you specify a nonexistent output style.	
	This Output Style cannot be used with this render type.	This alert displays if you specify an output style that is incompatible with this render type.	
	Cannot open this document type. Please select a QuarkXPress document or template.	HTTP Error #10120This alert displays if you submit an ave request for a QuarkCopyDesk article.	
	AVE-Doc for an App Studio layout is not supported.	HTTP Error #10545This alert displays if you submit an ave request with format=avedoc for an App Studio layout.	
Logs	See “Understanding logging."		

Example, GET URL	http://localhost:8080/ave/sample.qxp?format=avemag&layout=2
Example, object model	Request object name: AVERenderRequest //STEP1: Create the QuarkXPress Server Request //Context and set the necessary properties com.quark.qxpsm.QRequestContext requestCtx = new com.quark.qxpsm.QRequestContext(); Boolean responseAsURL = false; requestCtx.setDocumentName(docName); //STEP 2(SPECIFIC TO REQUESTS): //Create the AVE renderer //request and embed it in the request context. AVERenderRequest avereq = new AVERenderRequest(); avereq.setAVEData(request.getParameter("AVEData")); avereq.setFormat(request.getParameter("Format")); avereq.setLayout(request.getParameter("Layout")); requestCtx.setRequest(avereq); //STEP3: Create the service and call the //processRequest() API RequestService service = new RequestServiceStub(); com.quark.qxpsm.QContentData data = service.processRequest(requestCtx);
Notes	The default AVE output style is used.

eps

The eps render type returns an EPS rendering of a page or spread.

Namespace	EPS		
Parameters	outputstyle	stylename	Lets you specify an output style. To use a named output style, use the name of that output style. For example:http://localhost:8080/pdf/sample.qxp?outputstyle=stylename To use settings that have been captured with the

			Capture Settings in the QuarkXPress Print dialog box, use document. For example:http://localhost:8080/pdf/sample.qxp?outputstyle=document
		epsformat	color	Lets you specify an EPS format. The default value is color.
		epspreview	tiff | none	Lets you include or omit a TIFF preview. The default value is tiff.
		epsdata	ascii | binary | clean8bit	Lets you specify a data type for the EPS file. The default value is clean8bit.
		epstransparent	1 | 0 | true | false | yes | no	Lets you specify whether the EPS can include transparent areas.
		updateimage	true | false	Lets you specify whether to update imported pictures.
		updateflow	true | false	Lets you specify whether to update the text flow version of a project to the

			current version.
Render modifier parameters	page	Integer	Lets you specify a page.
	produceblankpages	1 | 0 | true | false | yes | no	Lets you specify whether to render blank pages.
	scale	Float .1 to 6.92 for Windows .1 to 8 on macOS	Lets you specify a scaling percentage. The valid values are from .1 (10%) to 8 (800%) on macOS or 6.92 (692%) on Windows.
	spread	Integer	Lets you specify a spread. The first spread is spread 1. In a facing-page document, spread 1 consists of the first page.
	layout	String	Lets you specify a layout by name or ID. The first layout is Layout 1.
	downloadlayoutFonts	1 | 0 | true | false | yes | no	Lets you specify whether to download all fonts used in the layout and all system fonts.
	downloadImportedPdfEpsFonts	1 | 0 | true | false | yes | no	Lets you specify whether to download all fonts required by imported PDF and EPS files.
Response	An EPS file.		

Alerts	The renderer for this image type has no way of rendering the desired objects.	HTTP Error #406 This alert displays if you submit a render request with the pages or box parameter.
	This Output Style does not exist.	This alert displays if you specify a nonexistent output style.
	This Output Style cannot be used with this render type.	This alert displays if you specify an output style that is incompatible with this render type.
Logs	See “Understanding logging."	
Example, GET URL	http://localhost:8080/eps/sample.qxp?epsformat=color&epsdata=clean8bit&epspreview=tiff&epsbleed=0&epstransparent=0	
Example, object model	Request object name: EPSRenderRequest //STEP1: Create the QuarkXPress Server Request //Context and set the necessary properties com.quark.qxpsm.QRequestContext requestCtx = new com.quark.qxpsm.QRequestContext(); Boolean responseAsURL = false; requestCtx.setDocumentName(docName); //STEP 2(SPECIFIC TO REQUESTS): //Create the EPS renderer //request and embed it in the request context. EPSRenderRequest epsreq = new EPSRenderRequest(); epsreq.setEPSData(request.getParameter(“EPSData")); epsreq.setEPSFormat(request.getParameter(“EPSFormat")); epsreq.setEPSPreview(request.getParameter(“EPSPreview")); requestCtx.setRequest(epsreq); //STEP3: Create the service and call the //processRequest() API RequestService service = new RequestServiceStub(); com.quark.qxpsm.QContentData data = service.processRequest(requestCtx);	
Notes	You can specify an output style and set additional local parameters of that output style. For example, if no bleed setting is specified in the output style named “mystylename", you can specify a bleed setting with a URL	

	like the following:http://localhost:8080/eps/sample.qxp?outputstyle=mystylename?bleed=symmetric You can override settings in an output style. For example, if an asymmetric bleed is specified in the output style named “mystylename," you could override it with the same URL.If you do not specify an EPS output style, the default EPS output style is used.

epub

The epub render type returns an ePUB rendering of a layout.

Namespace	ePUB		
Render modifier parameters	layout	String	Lets you specify a layout by name or ID. The first layout is Layout 1.
	outputstyle	String	Lets you specify an ePUB output style by name or ID.
Response	An ePUB (.epub) file.		
Alerts	The renderer for this image type has no way of rendering the desired objects.	HTTP Error #406 This alert displays if you submit a render request with the pages or box parameter.	
	ePub not created. There is no reflow layout in the document.	HTTP Error #10543This error appears if there is no reflow layout.	
Logs	See “Understanding logging."		
Example, GET URL	http://localhost:8080/epub/sample.qxp?outputstyle=epub1&layout=2		
Example, object model	Request object name: EPubRenderRequest //STEP1: Create the QuarkXPress Server Request //Context and set the necessary properties com.quark.qxpsm.QRequestContext requestCtx = new com.quark.qxpsm.QRequestContext();		

	Boolean responseAsURL = false; requestCtx.setDocumentName(docName); //STEP 2(SPECIFIC TO REQUESTS): //Create the ePUB renderer //request and embed it in the request context. EPubRenderRequest epubreq = new EPubRenderRequest(); epubreq.setEPubData(request.getParameter("EPubData")); epubreq.setCreateTOC(request.getParameter("CreateTOC")); epubreq.setLayout(request.getParameter("Layout")); requestCtx.setRequest(epubreq); //STEP3: Create the service and call the //processRequest() API RequestService service = new RequestServiceStub(); com.quark.qxpsm.QContentData data = service.processRequest(requestCtx);
Notes	You can only create an ePUB file from a project that includes a reflow article.

jpeg

The jpeg render type returns a JPEG rendering of a page or spread.

Namespace	JPEG		
Parameters	jpegquality	1 | 2 | 3 | 4	Lets you specify the image quality of a rendered JPEG image. The valid values are: 1 (highest quality), 2 (high quality), 3 (medium quality), and 4 (lowest quality). The default value is 1.
	upadateimage	true | false	Lets you specify whether to update imported pictures.
	updateflow	true | false	Lets you specify whether to update the text flow

			version of a project to the current version.
	pasteboard	true | false	Lets you specify whether to display pasteboard items. Works only with spread parameter. The default value is true. For example:http://localhost:8080/jpeg/document.qxp?spread=1&pasteboard=true
	showboxoutline	true | false	Lets you specify whether to include bounding box outlines in the response JPEG image even if the boxes have no content.. The default value is false.
	download	true | false	Renders the images in a Zip package for download.
	pages	true | false	Lets you specify a range of pages for request. (5-8,1-3 etc.)
Render modifier parameters	boxes	String	Lets you request multiple boxes.
	page	Integer	Lets you request a single page.

	scale	Float .1 to 6.92 for Windows .1 to 8 on macOS)	Lets you specify a scaling percentage. The valid values are from .1 (10%) to 8 (800%) on macOS or 6.92 (692%) on Windows.
	box	String	Lets you request a single box.
	spread	Integer	Lets you specify a spread. The first spread is spread 1. In a facing-page document, spread 1 consists of the first page.
	layout	String	Lets you specify a layout by name or ID. The first layout is Layout 1.
Response	A JPEG file.		
Logs	See “Understanding logging."		
Example, GET URL	http://localhost:8080/jpeg/sample.qxp?jpegquality=1		
Example, object model	Request object name: JPEGRenderRequest // STEP1: Create the QuarkXPress Server Request // Context and set the necessary properties com.quark.qxpsm.QRequestContext requestCtx = new com.quark.qxpsm.QRequestContext(); Boolean responseAsURL = false; requestCtx.setDocumentName(docName); // STEP2: Create the JPEG renderer request and attach it // to the request context. JPEGRenderRequest jpreq = new JPEGRenderRequest(); jpreq.setJPEGQuality(request.getParameter(“jpegQuality")); jpreq.setLayout(request.getParameter(“Layout")); requestCtx.setRequest(jpreq); // STEP3: Create the service		

	and // call the processRequest() API RequestService service = new RequestServiceStub(); com.quark.qxpsm.QContentData data = service.processRequest(requestCtx);

kindle

The kindle render type returns a rendering of a layout that can be viewed on Amazon Kindle readers.

Namespace	kindle		
Render modifier parameters	layout	String	Lets you specify a layout by name or ID. The first layout is Layout 1.
	outputstyle	String	Lets you specify a Kindle output style by name or ID.
Response	A Kindle (.mobi) file.		
Alerts	The renderer for this image type has no way of rendering the desired objects.	HTTP Error #406 This alert displays if you submit a render request with the pages or box parameter.	
	Kindle not created. There is no reflow layout in the document.	HTTP Error #10543This error appears if there is no reflow layout.	
Logs	See “Understanding logging."		
Example, GET URL	http://localhost:8080/kindle/sample.qxp?outputstyle=kindle&layout=2		
Example, object model	Request object name: KindleRenderRequest //STEP1: Create the QuarkXPress Server Request //Context and set the necessary properties com.quark.qxpsm.QRequestContext requestCtx = new com.quark.qxpsm.QRequestContext(); Boolean responseAsURL = false;		

	requestCtx.setDocumentName(docName); //STEP 2(SPECIFIC TO REQUESTS): //Create the Kindle renderer //request and embed it in the request context. KindleRenderRequest kindlereq = new KindleRenderRequest(); kindlereq.setKindleData(request.getParameter("KindleData")); kindlereq.setCreateTOC(request.getParameter("CreateTOC")); kindlereq.setLayout(request.getParameter("Layout")); requestCtx.setRequest(kindlereq); //STEP3: Create the service and call the //processRequest() API RequestService service = new RequestServiceStub(); com.quark.qxpsm.QContentData data = service.processRequest(requestCtx);
Notes	You can only create a Kindle file from a project that includes a reflow article.

literal

The literal render type returns the contents of a file without any attempt to process it as a template. Depending on the file's MIME type, the requested project can be displayed within the browser (for example, if the response is a JPEG file) or saved to disk (for example, if the response is a Microsoft Word document).

Namespace	literal	
Response	The requested file returned in the HTTP response.	
Alerts	Incorrect administration realm username and password.	HTTP Error #401 This alert displays if you specify an invalid administrator user name and password.What to do: Use the user name and password set in the Authentication pane of the General Preferences dialog box (Administration > Preferences > General) in the QuarkXPress Server Web interface.
Logs	See "Understanding logging."	
Example, GET URL	http://localhost:8080/literal/Story.doc	
Example, object model	Request object name: LiteralRequestcom.quark.qxpsm.QRequestContext rc = new com.quark.qxpsm.QRequestContext();	

	if(!this.DocumentSettings1.documentName.Text.Equals("")) rc.documentName = this.DocumentSettings1.documentName.Text; rc.request = new LiteralRequest(); //Create the service and call it with QRequestContext object RequestService svc = new RequestService(); com.quark.qxpsm.QContentData qc = svc.processRequest(rc);

pdf

The pdf render type returns a PDF rendering of a project.

Namespace	PDF		
Parameters	outputstyle	stylename, document	Lets you specify an output style. To use a named output style, use the name of that output style. For example:http://localhost:8080/pdf/sample.qxp?outputstyle=stylename To use settings that have been captured with the Capture Settings in the QuarkXPress Export as PDF dialog box, use document. For example:http://localhost:8080/pdf/sample.qxp?outputstyle=document
	title	String	Lets you specify the title of the PDF file.

	subject	String	Lets you specify the subject of the PDF file.
	author	String	Lets you specify the author of the PDF file.
	keywords	String	Lets you specify keywords for the PDF file.
	includehyperlinks	1 | 0 | true | false | yes | no	Lets you specify whether to include hyperlinks in the PDF file.
	exportlistsashyperlinks	1 | 0 | true | false | yes | no	Lets you specify whether to export lists as hyperlinks. To use this parameter, you must set includehyperlinks to true.
	exportindexesashyperlinks	1 | 0 | true | false | yes | no	Lets you specify whether to export the index as hyperlinks. To use this parameter, you must set includehyperlinks to true.
	exportlistsasbookmarks	1 | 0 | true | false | yes | no	Lets you specify whether to export lists as bookmarks. To use this parameter, you must set includehyperlinks to true.

	mode	composite or separations	Lets you specify whether the PDF file is a composite or includes separations.
	printcolors	cmyk, rgb, grayscale, cmykandspot, asis	Lets you specify the color space of the PDF file. This option is available only when mode is set to composite.
	plates	inripseps	Lets you specify a separation method. This option is available only when mode is set to separations.
	produceblankpages	1 | 0 | true | false | yes | no	Lets you specify whether to include blank pages. This option is available only when mode is set to composite.
	useopi	1 | 0 | true | false | yes | no	Lets you specify whether to use OPI.
	images	includeimages, omittiff, omittiffandeps	Lets you specify whether to include TIFF and EPS images from an OPI server.
	registration	off, centered, offcenter	Lets you include, omit, and configure

			registration marks.
	offset	0–30 (in points)	Lets you specify the offset of registration marks.
	bleed	pageitemsonly, symmetric	Lets you specify a bleed type.
	offsetbleed	0–6 (in inches)	Lets you specify a bleed offset to use. This option is available only when bleed is set to symmetric.
	spreads	1 | 0 | true | false | yes | no	Lets you specify whether to output spreads.
	lowresolution	1 | 0 | true | false | yes | no	Lets you request a low-resolution (36 dpi) PDF.
	colorimagedownsample	9–2400	Lets you specify the resolution of color images.
	grayscaleimagedownsample	9–2400	Lets you specify the resolution of grayscale images.
	monochromeimagedownsample	9–2400	Lets you specify the resolution of monochrome images.
	colorcompression	true | false	Lets you specify whether medium-quality manual JPEG compression should be applied to color images.

	grayscalecompression	true | false	Lets you specify whether medium-quality manual JPEG compression should be applied to grayscale images.
	monochromecompression	true | false	Lets you specify whether ZIP compression should be applied to monochrome images.
	pdffile	String	Lets you specify the PDF name. This option is available only when PDF to Folder is selected in QuarkXPress Server PDF preferences.
	psfile	String	Lets you specify the PostScript file name. This option is available only when PostScript for later Distilling is selected in QuarkXPress Server PDF preferences.
	thumbnail	bw | color	Lets you embed a thumbnail in the PDF file.
	mode	composite | separations	Lets you specify the PDF file's

			color mode.
	fontdownload	yes | no	Lets you turn font download on or off. You cannot specify which fonts are downloaded.
	layers	String	Lets you specify which layers should be included, as a comma-separated list.
	verification	pdfx1a | pdfx3	Lets you use PDF/X–1a or PDF/X–3 verification.
	separate	yes | no	Lets you specify whether to output each page as a separate file.
	produceblankplates	yes | no	Lets you specify whether to include blank plates.
	download	1 | 0 | true | false	When download is true, the browser always displays a dialog box that lets the end user save the returned file, even if the browser can display it.When download is false, the browser attempts to display the

			returned file. If the browser cannot display the file, it lets the end user save the returned file. The default value is false.
	layoutstart	1 | 0 | true | false | yes | no	Lets you specify the number of the first layout to render when you render multiple layouts as separate PDF files. PDF files are saved at the location specified in QuarkXPress Server preferences (Administration > Preference > General > Server > Document Root Folder). The first layout in a project is layout 0. For example:http://localhost:8080/pdf/multilayout.qxp?layoutstart=0&layoutend=3
	layoutend	Integer	Lets you specify the number of the last layout to render when you render multiple layouts as separate PDF

			files. PDF files are saved at the location specified in QuarkXPress Server preferences (Administration > Preference > General > Server > Document Root Folder). The first layout in a project is layout 0. For example:http://localhost:8080/pdf/multilayout.qxp?layoutstart=0&layoutend=3
	updateimage	true | false	Lets you specify whether to update imported pictures.
	updateflow	true | false	Lets you specify whether to update the text flow version of a project to the current version.
Render modifier parameters	page	Integer	Lets you specify a single page.
	pages	String (page range)	Lets you specify a range of pages.
	spread	Integer	Lets you specify a spread. The first spread is spread 1. In a facing-page document, spread 1 consists

			of the first page.
	layout	String	Lets you specify a layout by name or ID. The first layout is Layout 1.
	spreads	Boolean 1 | 0 | true | false | yes | no	Lets you specify that the output use spreads.
Response	A PDF file.		
Alerts	This page range is invalid	HTTP Error #500QuarkXPress Server Error #147This alert displays if you try to render an invalid page range.	
	No file produced. The project requested contains only blank pages.	HTTP Error #500This alert displays if you try to render a a project that contains only blank pages.	
	This Output Style does not exist.	This alert displays if you specify a nonexistent output style.	
	This Output Style cannot be used with this render type.	This alert displays if you specify an output style that is incompatible with this render type.	
Logs	See “Understanding logging."		
Example, GET URL	This URL renders “sample.qxp” as a PDF with a symmetric bleed: http://localhost:8080/pdf/sample.qxp?bleed=symmetric&offsetbleed=2 This URL renders a PDF in which color images are downsampled to a resolution of 300 dpi and manual medium-quality JPEG compression is applied: http://localhost:8080/pdf/sample.qxp?colorimagedownsample=300&colorcompression=true		
Example, object model	Request object name: PDFRenderRequest // STEP1: Create the QuarkXPress Server Request Context // and set the nescessary properties com.quark.qxpsm.QRequestContext requestCtx = new com.quark.qxpsm.QRequestContext(); Boolean responseAsURL = false;		

	requestCtx.setDocumentName(docName); // STEP 2(SPECIFIC TO REQUESTS): // Create the PDF renderer request // and embed it in the request context. PDFRenderRequest pdfreq = new PDFRenderRequest(); pdfreq.setAuthor(request.getParameter("Author")); pdfreq.setTitle(request.getParameter("Title")); pdfreq.setLayout(request.getParameter("Layout")); pdfreq.setSpread(request.getParameter("Spread")); pdfreq.setPage(request.getParameter("mPage")); pdfreq.setPages(request.getParameter("Pages")); if(strLowResolution !=null && strLowResolution.equals("True")) pdfreq.setLowResolution("true"); requestCtx.setRequest(pdfreq); // STEP3: Create the service and // call the processRequest() API RequestService service = new RequestServiceStub(); com.quark.qxpsm.QContentData data = service.processRequest(requestCtx); For more information about the object model, see the samples.
Notes	There are three ways to generate PDF files with QuarkXPress Server. You can generate a PDF file in QuarkXPress Server and return it to the end user, generate the PDF in QuarkXPress server and save it to a folder on the server computer, or generate a PostScript file for later distilling and save it to a folder on the server computer. To choose one of these output methods in QuarkXPress Server, choose Administration > Preferences > Renderer > PDF) and then click DirectPDF, PDFtoFolder, or PS4D (PostScript for Later Distilling). If you choose either of the last two options, click Browse and navigate to the target folder, then choose an option from the Default Name drop-down menu.You can specify an output style and set additional local parameters of that output style. For example, if no bleed setting is specified in the output style named "mystylename", you can specify a bleed setting with a URL like the following:http://localhost:8080/pdf/sample.qxp?outputstyle=mystylename&bleed=symmetric You can override settings in an output style. For example, if an

asymmetric bleed is specified in the output style named “mystylename," you could override it with the same URL. If you do not specify a PDF output style, the default PDF output style is used. The default PD output style is Screen - Low Quality/Low Resolution.You can still obtain a high resolution PDF by specifying the output style. Output style names are case-sensitive and should be precise. The following will return a list of output styles from the server: http://<server><port>/getserverinfo <OUTPUTSTYLES> <OUTPUTSTYLE TYPE = “PDF">Default PDF Output Style</OUTPUTSTYLE> <OUTPUTSTYLE TYPE = “PDF">Press - High Quality/High Resolution</OUTPUTSTYLE> <OUTPUTSTYLE TYPE = “PDF">Print - Medium Quality/Medium Resolution</OUTPUTSTYLE> <OUTPUTSTYLE TYPE = “PDF">Screen - Medium Quality/Low Resolution</OUTPUTSTYLE> <OUTPUTSTYLE TYPE = “PDF">Screen - Low Quality/Low Resolution</OUTPUTSTYLE> <OUTPUTSTYLE TYPE = “PDF">PDF/X-3:2002</OUTPUTSTYLE> <OUTPUTSTYLE TYPE = “PDF">PDF/X-1a:2001</OUTPUTSTYLE> <OUTPUTSTYLE TYPE = “Print">Default Print Output Style</OUTPUTSTYLE> <OUTPUTSTYLE TYPE = “EPS">Default EPS Output Style</OUTPUTSTYLE> <OUTPUTSTYLE TYPE = “ePub">Default ePub Output Style</OUTPUTSTYLE> <OUTPUTSTYLE TYPE = “Kindle">Default Kindle Output Style</OUTPUTSTYLE> <OUTPUTSTYLE TYPE = “PDF">Default PDF For AVE</OUTPUTSTYLE> <OUTPUTSTYLE TYPE = “AVE">Default AVE Output Style</OUTPUTSTYLE> </OUTPUTSTYLES >

png

The png render type returns a PNG rendering of a page or spread.

Namespace	PNG		
Parameters	pngcompression	1 | 2 | 3 | 4	Lets you specify the compression of a PNG

			response. The valid values are: 1 (lowest compression), 2 (medium compression), 3 (high compression), and 4 (highest compression). The default value is 1.
	transparentpng	true | false	Lets you specify whether to generate a PNG file that uses transparency.
	upadateimage	true | false	Lets you specify whether to update imported pictures.
	updateflow	true | false	Lets you specify whether to update the text flow version of a project to the current version.
Render modifier parameters	boxes	String	Lets you request multiple boxes.
	page	Integer	Lets you specify a single page.
	scale	Float .1 to 6.92 for Windows .1 to 8 on macOS	Lets you specify a scaling percentage. The valid values are from .1 (10%) to 8 (800%) on macOSmacOSor

			6.92 (692%) on Windows.
	box	String	Lets you request a single box.
	spread	Integer	Lets you specify a spread. The first spread is spread 1. In a facing-page document, spread 1 consists of the first page.
	layout	String	Lets you specify a layout by name or ID. The first layout is Layout 1.
Response	A PNG file.		
Logs	See “Understanding logging."		
Example, GET URL	http://localhost:8080/png/sample.qxp?pngcompression=1		
Example, object model	Request object name: PNGRenderRequest // STEP1: Create the QuarkXPress Server Request // Context and set the nescessary properties com.quark.qxpsm.QRequestContext rc = new com.quark.qxpsm.QRequestContext(); Boolean responseAsURL = false; rc.setDocumentName(docName); // STEP 2(SPECIFIC TO REQUESTS):Create the PNG renderer // request and embed it in the request context. PNGRenderRequest pngreq = new PNGRenderRequest(); pngreq.setPNGCompression(request.getParameter(“PNGCompression")); pngreq.setLayout(request.getParameter(“Layout")); pngreq.setSpread(request.getParameter(“Spread")); pngreq.setPage(request.getParameter(“mPage")); rc.setRequest(pngreq); // STEP3: Create the service and // call the processRequest() API RequestService service = new RequestServiceStub(); com.quark.qxpsm.QContentData data = service.processRequest(rc);		

postscript

The postscript render type returns a PostScript rendering of a project.

Namespace	PostScript		
Parameters	prntbleed	Page | asym, clip<Boolean>, top<float>, bottom<float>, left<float>, right<float> | sym, clip<Boolean>, amount<float>	Lets you specify bleed values for a page.To specify an asymmetric bleed, use the following format:prntbleed=asym,clip,top,bottom, left, rightThe clip value is Boolean (yes/no). The top, bottom, left, and right values are float values. For example:http://localhost:8080/postscript/Sample.qxp?prntbleed=asym,true,1,2,2,1 The above example results in an asymmetric bleed of 1 on the top, 2 on the bottom, 2 on the left, and 1 on the right. To specify a symmetric bleed, use the following format:prntbleed=sym,clip,amount The clip value is Boolean (yes/no).

			The amount value is a float value. For example:http://localhost:8080/postscript/Sample.qxp?prntbleed=sym,true,1 The above example results in a symmetric bleed of 1 on all sides. Default: prntbleed=sym,yes,0
	outputstyle	stylename, document	Lets you specify an output style. To use a named output style, use the name of that output style. For example:http://localhost:8080/postscript/sample.qxp?outputstyle=stylename To use settings that have been captured with the Capture Settings in the QuarkXPress Print dialog box, use document. For example:http://localhost:8080/postscript/sample.qxp?outp

			utstyle=document
	updateimage	true | false	Lets you specify whether to update imported pictures.
	updateflow	true | false	Lets you specify whether to update the text flow version of a project to the current version.
Render modifier parameters	page	Integer	Lets you specify a single page.
	pages	String (page range)	Lets you specify a range of pages.
	spread	Integer	Lets you specify a spread. The first spread is spread 1. In a facing-page document, spread 1 consists of the first page.
	layout	String	Lets you specify a layout by name or ID. The first layout is Layout 1.
Response	A PostScript file.		
Alerts	This page range is invalid.	HTTP Error #500 QuarkXPress Server Error #147 This alert displays if you try to render an invalid page range.	
	No file produced. The document requested contains only blank pages.	HTTP Error #500 This alert displays if you try to render a a project that contains only blank pages.	
	PostScript printer mapped	HTTP Error #500 This alert displays if the PostScript printer or driver is not set to	

	to file not found	Print to File.
	This Output Style does not exist.	This alert displays if you specify a nonexistent output style.
	This Output Style cannot be used with this render type.	This alert displays if you specify an output style that is incompatible with this render type.
Logs	See “Understanding logging."	
Example, GET URL	http://localhost:8080/postscript/Sample.qxp	
Example, object model	Request object name: PostScriptRenderRequest // STEP1: Create the QuarkXPress Server Request // Context and set the nescessary properties com.quark.qxpsm.QRequestContext requestCtx = new com.quark.qxpsm.QRequestContext(); Boolean responseAsURL = false; requestCtx.setDocumentName(docName); // STEP 2(SPECIFIC TO REQUESTS): // Create the Post Script renderer // request and embed it in the request context. PostScriptRenderRequest pscreq = new PostScriptRenderRequest(); pscreq.setPrintBleed(request.getParameter(“PrintBleed")); pscreq.setPrintPPD(request.getParameter(“PrintPPD")); pscreq.setPages(request.getParameter(“Pages")); requestCtx.setRequest(pscreq); // STEP3: Create the service and call the // processRequest() API RequestService service = new RequestServiceStub(); com.quark.qxpsm.QContentData data = service.processRequest(requestCtx);	
Notes	To create a PostScript file, you must have a PostScript driver on the server computer.You can specify an output style and set additional local parameters of that output style. For example, if no bleed setting is specified in the output style named “mystylename", you can specify a bleed setting with a URL like the following:http://localhost:8080/eps/sample.qxp?outputstyle=mystylename&bleed=symmetric You can	

	override settings in an output style. For example, if an asymmetric bleed is specified in the output style named “mystylename," you could override it with the same URL.If you do not specify a PostScript-compatible output style, the default PostScript-compatible output style is used.

qcddoc

The qcddoc render type returns a QuarkCopyDesk article.

Namespace	qcddoc		
Parameters	article	String	Lets you specify which article in a project to render. For example:http://localhost:8080/qcddoc/abc.qxp?article=article1
	component	String	Lets you specify which component in an article to render. For example:http://localhost:8080/copydesk/abc.qcd?component=comp1
	format	lightweight | fullfeatured	Lets you render an article in lightweight or full-featured format. For example:http://localhost:8080/qcddoc/abc.qxp?article=article1&

			format=fullfeatured
	saveastemplate	true | false	Lets you save a copy of an article that was created in QuarkCopyDesk as a template. The default value is true. For example:http://QXPServer8:8080/saveas/qcddoc/article.qcd?saveastemplate=trueYou can also use this parameter to save a copy of a template as an article. For example:http://QXPServer8:8080/saveas/qcddoc/template.qct?saveastemplate=false
	includepagepicture	true | false | 1 | 0	Lets you include a page picture when you export an article from a QuarkXPress layout. Valid options are:picformat (embedded or separate)quality (blackandwhite or color)picdpi (72,

			144, or 200)spreadrange (all or first)For example:http://localhost:8080/saveas/qcddoc/4.qxp?includepagepicture=1&quality=blackandwhite&picdpi=144&spreadrange=first http://localhost:8080/saveas/qcddoc/PagePicture.qxp?includepagepicture=true
Render modifier parameters	modify	XML	Lets you modify the article with XML. For more information, see “XML modify."
Response	A QuarkCopyDesk article.		
Alerts	There is no box with the specified identifier.	This alert displays if the box corresoponding to a referenced component does not exist.	
	The number of characters in the article name can't be greater than max limit.	This alert displays if an article name is longer than 32 characters.	
	The article/component name is not unique.	This alert displays if you create or change the name of an article or component so that it is the same as the name of an existing article or component.	

Logs	See “Understanding logging."
Example, GET URL	http://localhost:8080/qcddoc/copydesk/sample.qcd
Example, object model	Request object name: CopyDeskDocRequest

qxpdoc

The qxpdoc render type returns a QuarkXPress project.

Namespace	qxpdoc		
Parameters	qxpdocver	8 | 9	Indicates the QuarkXPress version format to use. For example:http://localhost:8080/qxpdoc/construct/project1.qxp?qxpdocver=8
	upadateimage	true | false	Lets you specify whether to return modified pictures in the response or not. If set to false, modified pictures are not returned. If set to true, modified pictures are returned. The default value is true.
	saveastemplate	true | false	Lets you save a copy of a project as a template. The default value is true. For example:http://lo

			calhost:8080/saveas/qxpdoc/project.qxp?saveastemplate=trueYou can also use this parameter to save a copy of a template as a project. For example:http://localhost:8080/saveas/qxpdoc/template.qpt?saveastemplate=false
Render modifier parameters	layout	String	Lets you specify a layout by name or ID. The first layout is Layout 1.
Response	A QuarkXPress project.		
Alerts	QuarkXPress document return is disabled.	HTTP Error #500 This alert displays if Disable QXD Return is checked in the QuarkXPress Server administration interface (Administration > Preferences > General > Server).	
	The renderer for this image type has no way of rendering the desired objects.	HTTP Error #406 This alert displays if you submit a qxpdoc render request with the page, pages, box, or spread parameter.	
	Cannot save a QuarkXPress Project down to an earlier version.	HTTP Error #500 This alert displays if you attempt to save a QuarkXPress 6.x project to an earlier version of QuarkXPress with the qxpdocver parameter.	
Logs	See “Understanding logging.”		
Example, GET	http://localhost:8080/qxpdoc/sample.qxp		

URL	
Example, object model	Request object name: QuarkXPressRenderRequest // STEP1: Create the QuarkXPress Server Request // Context and set the nescessary properties com.quark.qxpsm.QRequestContext requestCtx = new com.quark.qxpsm.QRequestContext(); Boolean responseAsURL = false; requestCtx.setDocumentName(docName); // STEP 2(SPECIFIC TO REQUESTS):Create the QuarkXPress // renderer request and embed it in the request context. QuarkXPressRenderRequest qxpreq = new QuarkXPressRenderRequest(); qxpreq.setDocumentVersion(request.getParameter(“XpressDocVersion”)); qxpreq.setLayout(request.getParameter(“Layout”)); requestCtx.setRequest(qxpreq); // STEP3: Create the service and call the processRequest() API RequestService service = new RequestServiceStub(); com.quark.qxpsm.QContentData data = service.processRequest(requestCtx);

screenpdf

The screenpdf render type returns a low-resolution PDF rendering of a project. This render type overrides the **PDF Workflow** setting in the QuarkXPress Server administration interface (**Administration > Preferences > Renderer > PDF**) and always sends the PDF file to the browser.

Namespace	Screenpdf

Parameters	outputstyle	stylename	Lets you specify an output style. To use a named output style, use the name of that output style. For example:http://localhost:8080/screenpdf/sample.qxp?outputstyle=stylename To use settings that have been captured with the Capture Settings in the QuarkXPress Print dialog box, use document. For example:http://localhost:8080/screenpdf/sample.qxp?outputstyle=document
	title	String	Lets you specify the title of the PDF file.
	subject	String	Lets you specify the subject of the PDF file.
	author	String	Lets you specify the author of the PDF file.
	keywords	String	Lets you specify keywords of the PDF file.
	includehyperlink	1 | 0 | true | false | yes	Lets you specify

	s	| no	whether to include hyperlinks in the PDF file.
	exportlistsashyperlinks	1 | 0 | true | false | yes | no	Lets you specify whether to export lists as hyperlinks. To use this parameter, you must set includehyperlinks to true.
	exportindexesashyperlinks	1 | 0 | true | false | yes | no	Lets you specify whether to export the index as hyperlinks. To use this parameter, you must set includehyperlinks to true.
	exportlistsasbookmarks	1 | 0 | true | false | yes | no	Lets you specify whether to export lists as bookmarks. To use this parameter, you must set includehyperlinks to true.
	mode	composite or separations	Lets you specify whether the PDF file is a composite or includes separations.
	printcolors	cmyk, rgb, grayscale, cmykandspot, asis	Lets you specify the color space of the PDF file. This option is available only when mode

			is set to composite.
	plates	inripseps	Lets you specify a separation method. This option is available only when mode is set to separations.
	produceblankpages	1 | 0 | true | false | yes | no	Lets you specify whether to include blank pages. This option is available only when mode is set to composite.
	useopi	1 | 0 | true | false | yes | no	Lets you specify whether to use OPI.
	images	includeimages, omittiff, omittiffandeps	Lets you specify whether to include TIFF and EPS images from an OPI server.
	registration	off, centered, offcenter	Lets you include, omit, and configure registration marks.
	offset	0–30 (in points)	Lets you specify the offset of registration marks.
	bleed	pageitemsonly, symmetric	Lets you specify a bleed type.
	offsetbleed	0–6 (in inches)	Lets you specify a

			bleed offset to use. This option is available only when bleed is set to symmetric.
	spreads	1 | 0 | true | false | yes | no	Lets you specify whether to output spreads.
	lowresolution	1 | 0 | true | false | yes | no	Lets you request a low-resolution (36 dpi) PDF.
	colorimagedownsample	9–2400	Lets you specify the resolution of color images.
	grayscaleimagedownsample	9–2400	Lets you specify the resolution of grayscale images.
	monochromeimagedownsample	9–2400	Lets you specify the resolution of monochrome images.
	colorcompression	true | false	Lets you specify whether medium-quality manual JPEG compression should be applied to color images.
	grayscalecompression	true | false	Lets you specify whether medium-quality manual JPEG compression should be applied to grayscale images.
	monochromeco	true | false	Lets you specify

	mpression		whether ZIP compression should be applied to monochrome images.
	pdffile	String	Lets you specify the PDF name. This option is available only when PDF to Folder is selected in QuarkXPress Server PDF preferences.
	psfile	String	Lets you specify the PostScript file name. This option is available only when PostScript for later Distilling is selected in QuarkXPress Server PDF preferences.
	thumbnail	bw | color	Lets you embed a thumbnail in the PDF file.
	mode	composite | separations	Lets you specify the PDF file's color mode.
	fontdownload	yes | no	Lets you turn font download on or off. You cannot specify which fonts are downloaded.
	layers	String	Lets you specify which layers

			should be included, as a comma-separated list.
	transparencyres	Integer value from 36 to 3600	Lets you specify the resolution for flattened content.
	verification	pdfx1a | pdfx3	Lets you use PDF/X–1a or PDF/X–3 verification.
	separate	yes | no	Lets you specify whether to output each page as a separate file.
	produceblankplates	yes | no	Lets you specify whether to include blank plates.
	updateimage	true | false	Lets you specify whether to update imported pictures.
	updateflow	true | false	Lets you specify whether to update the text flow version of a project to the current version.
Render modifier parameters	page	Integer	Lets you specify a single page.
	pages	String (page range)	Lets you specify a range of pages.
	spread	Integer	Lets you specify a spread. The first spread is spread 1. In a facing-

			page document, spread 1 consists of the first page.
	layout	String	Lets you specify a layout by name or ID. The first layout is Layout 1.
	spreads	Boolean (1 | 0 | true | false | yes | no)	Lets you specify that the output use spreads.
Response	A screen-resolution PDF file		
Alerts	This page range is invalid.	HTTP Error #500QuarkXPress Server Error #147This alert displays if you try to render an invalid page range.	
	No file produced. The document requested contains only blank pages.	HTTP Error #500This alert displays if you try to render a a project that contains only blank pages.	
Logs	See “Understanding logging."		
Example, GET URL	http://localhost:8080/screenpdf/sample.qxp?colorimagedownsample=72&colorcompression=0		
Example, object model	Request object name: ScreenPDFRenderRequest // STEP1: Create the QuarkXPress Server Request Context // and set the nescessary properties com.quark.qxpsm.QRequestContext requestCtx = new com.quark.qxpsm.QRequestContext(); String docName = request.getParameter(“documentName") ; requestCtx.setDocumentName(docName); // STEP 2(SPECIFIC TO REQUESTS): // Create the QuarkXPress renderer // request and embed it in the request context. ScreenPDFRenderRequest screenpdfRequest = new ScreenPDFRenderRequest(); screenpdfRequest.setColorImageDownSample(request.getParameter(“ColorImageDownSample")); screenpdfRequest.setCompression(request.getParameter(“Compression"));		

```
requestCtx.setRequest(screenpdfRequest);   // STEP3:
Create the service and // call the processRequest() API
RequestService service = new RequestServiceStub();
com.quark.qxpsm.QContentData data =
service.processRequest(requestCtx);
```

html5

To Export as responsive HTML5 publication(Multi Device Output):

- Create/Duplicate different layouts in a QXP project where each layout corresponds to a digital device/orientation.
- Optionally create an HTML5 output style.
- Optionally specify a name for the section to be exported as a Table of Contents.
- While exporting as HTML5 Publication, select layouts to be included in the publication and select the HTML5 output style to be applied.

To Create Multiple Layouts in a QXP Project,one for each device/orientation please refer to the "HTML5 Publication output styles" section in '*A Guide to QuarkXPress 2017*' , found here: "*https://files.quark.com/download/documentation/QuarkXPress/2017/English/QXP -2017-User-Guide-EN.pdf*"

The html5 render type returns a .zip file containing HTML5 output.

Modifier XML Markup:

Use the Markup for the respective layout type (iPad(V), iPhone5 (V), iPhone6 (V), Android (V)) as shownin the following figure:

```
<PROJECT>
	<LAYOUT>
		<ID NAME="iPad (V)"/>
		<PAGESEQUENCE MASTERREFERENCE="G-TOC">
		.
		</PAGESEQUENCE>
		<PAGESEQUENCE MASTERREFERENCE="A-Introduction">
		.
		</PAGESEQUENCE>
		.
		.
	</LAYOUT>
	<LAYOUT>
		<ID NAME="iPhone5 (V)"/>
		<PAGESEQUENCE MASTERREFERENCE="G-TOC">
		.
		</PAGESEQUENCE>
		<PAGESEQUENCE MASTERREFERENCE="A-Introduction">
		.
		</PAGESEQUENCE>
		.
		.
	</LAYOUT>
		<LAYOUT>
		<ID NAME="iPhone6 (V)"/>
		<PAGESEQUENCE MASTERREFERENCE="G-TOC">
		.
		</PAGESEQUENCE>
		<PAGESEQUENCE MASTERREFERENCE="A-Introduction">
		.
		</PAGESEQUENCE>
		.
		.
	</LAYOUT>
		<LAYOUT>
		<ID NAME="Android (V)"/>
		<PAGESEQUENCE MASTERREFERENCE="G-TOC">
		.
		</PAGESEQUENCE>
		<PAGESEQUENCE MASTERREFERENCE="A-Introduction">
		.
		</PAGESEQUENCE>
		.
		.
	</LAYOUT>
</PROJECT>
```

HTML5 Modifier XML Markup

The following table pertains to HTML5 publications:

Namespace	html5		
Parameters	html	String	Lets you generate an HTML5 article and returns it as a .zip file. For example:http://localhost:8080/appstudio/html/template.qxp
Render modifier parameters	layout	String	Lets you specify a layout by name or

			ID. The first layout is Layout 1.
	page	Integer	Lets you specify a page.
	pages	String (page range)	Lets you specify a range of pages.
Response	A .zip file containing an HTML5 article.		
Alerts	The renderer for this image type has no way of rendering the desired objects.	HTTP Error #406 This alert displays if you submit a render request with the box parameter.	
	Cannot open this document type. Please select a QuarkXPress document or template.	HTTP Error #10120 This alert displays if you submit an appstudio request for a QuarkCopyDesk article.	
Logs	See “Understanding logging."		
URL Request to export as HTML5 Multi Device Output	http://<QXPS Server>:<port>/html5/<QXP Template>.qxp?modify=file:<Modifier File>&layouts=iPad (V),iPad (H),iPhone5 (V),iPhone5 (H)		
Example, object model	Request object name: html5RenderRequest		

- If you want to use a custom HTML5 output style, then it must be specified in the URL request: http://<QXPS Server>:<port>/html5/<QXP Template>?modify=file:<Modifier File>&layouts=iPad (V),iPad (H),iPhone5 (V),iPhone5 (H)&outputstyle=<MyHTML5Style>

Understanding render modifiers

Render modifiers let you control which parts of a project are rendered and set the scale of the returned renderings. The topics covered here include the following:

Property	Description
box	The box render modifier lets you render a single box.
boxes	The boxes render modifier lets you render multiple boxes.

Property	Description
layer	The layer render modifier lets you show and hide layers prior to rendering. This render modifier also lets you add and remove layers from a project on the server.
layout	The layout render modifier lets you render a particular layout.
movepages	The movepages render modifier lets you move pages prior to rendering.
page	The page render modifier lets you render a single page.
pages	The pages render modifier lets you render multiple pages.
scale	The scale render modifier lets you specify the scale at which content is rendered.
spread	The spread render modifier lets you render a single spread.
spreads	The spreads render modifier lets you render multiple spreads.

Additional render-type-specific parameters are listed on each render type's page.

- In the QuarkXPress Server Manager API, render modifiers are properties of render request classes.
- Render modifier names are not case-sensitive.

annotateerrors

The annotateerrors render modifier lets you include descriptions of rendering errors as notes in the layout itself. In rendered QuarkXPress files, errors are displayed as notes. In rendered PDFs, errors are displayed as comments. In XML output, errors are displayed as notes XML markup.

Parameters	annotateerrors	String	Includes descriptions of rendering errors as notes in the layout.
Compatible with	pdf, qxpdoc, xml, postscript		
Logs	See “Understanding logging."		
Example GET URL	http://localhost:8080/png/sample.qxp?box=annotateerrors		
Notes	Descriptions of rendering errors are formatted with the default character style sheet.		

appenderrors

The appenderrors render modifier lets you include descriptions of rendering errors after the last page in the layout. Descriptions of rendering errors are formatted in 10-point magenta Arial.

Parameters	appenderrors	String	Includes descriptions of rendering errors after the last page in the layout.
Compatible with	pdf, qxpdoc, xml, postscript		
Logs	See “Understanding logging."		
Example GET URL	http://localhost:8080/png/sample.qxp?box=appenderrors		
Notes	Descriptions of rendering errors are formatted in 10-point magenta Arial.		

box

The box render modifier lets you render a single box.

Parameters	box	String	Lets you specify which box to render.
	overlap	String	Lets you specify whether to show the area overlapped by the specified box.
Compatible with	jpeg, png, raw		
Alerts	There is no box with the specified identifier.	HTTP Error #500 This alert displays if you request a box that does not exist.	
	Cannot render box. The box must be within the page	HTTP Error #500 This alert displays if you request a box that is outside the page boundary.	

	boundaries.	
	The renderer for this image type has no way of rendering the desired objects.	HTTP Error #406 This alert displays if you try to use the box parameter with the eps, pdf, or qxpdoc render types.
Logs	See “Understanding logging."	
Example GET URL	http://localhost:8080/png/sample.qxp?box=pictbox	
Notes	To render a box in a particular layout, use a URL like the following:http://localhost:8080/png/sample.qxp?layout=2&page=3&box=textboxWhen you render using the box parameter, the box ID has a higher priority than the box name.	

boxes

The boxes render modifier lets you render multiple boxes.

Parameters	boxes	String	Lets you specify which boxes to render.
	overlap	String	Lets you specify whether to show the area overlapped by the specified boxes.
Compatible with	jpeg, png, raw		
Alerts	There is no box with the specified identifier.	HTTP Error #500 This alert displays if you request a box that does not exist.	
	Cannot render box. The box must be within the page boundaries.	HTTP Error #500 This alert displays if you request a box that is outside the page boundary. .	
	The renderer for	HTTP Error #406 This alert displays if	

	this image type has no way of rendering the desired objects.	you try to use the boxes parameter with the eps, pdf, or qxpdoc render types.
Logs	See “Understanding logging."	
Example GET URL	http://server:port/jpeg/doc.qxp?boxes=box1,box2	
Notes	To render boxes in a particular layout, use a URL like the following:http://localhost:8080/png/sample.qxp?layout=2&page=3&box=textboxWhen you render using the box parameter, the box ID has a higher priority than the box name.	

compositionzone

The compositionzone parameter lets you return an XML representation of one or more Composition Zones items.

Parameters	compositionzone	String	Lets you specify which Composition Zones item to return. For example:http://localhost:8080/xml/sample.qxp?compositionzone=czbox
	compositionzones	String	Lets you specify which Composition Zones items to return. For example:http://localhost:8080/xml/sample.qxp?compoxitionzones=czbox1, czbox2
Compatible with	xml		

Alerts	Invalid box given in Box Param.	Error #10401 This alert displays if you request a box that is not a Composition Zones item.
Logs	See “Understanding logging."	

layer

The layer render modifier lets you show and hide layers prior to rendering. This render modifier also lets you add and remove layers from a project on the server.

Parameters	layer	String	Lets you specify which layer to render. You can specify multiple layer names in one request.
	addlayer	String	Lets you add a new layer. You can add one layer per request.
	deletelayer	String	Lets you delete a layer and the items on that layer. You can delete one layer per request.
	alllayers	Boolean (1 | 0 | true | false | yes | no)	Lets you render every layer in the project, including hidden and suppressed layers.
	layerattribute	String	Lets you modify the attributes of a layer. You can modify one layer per request.
	name	String	Lets you specify a

			new name for a layer. You must use this parameter in conjunction with the layerattribute parameter.
		visible	Boolean (1 | 0 | true | false | yes | no)	Lets you make a layer visible or invisible. You can use this parameter in conjunction with the addlayer and layerattribute parameters. This parameter overrides QuarkXPress layer visibility preferences.
		suppressoutput	Boolean (1 | 0 | true | false | yes | no)	Lets you suppress or allow the output of a layer. You can use this parameter in conjunction with the addlayer and layerattribute parameters. This parameter overrides QuarkXPress suppress output preferences.
		locked	Boolean (1 | 0 | true | false | yes | no)	Lets you lock or unlock a layer. You can use this parameter in

			conjunction with the addlayer and layerattribute parameters. This parameter overrides QuarkXPress layer locking preferences.
	keeprunaround	Boolean (1 | 0 | true | false | yes | no)	Lets you set or change a layer's Keep Runaround setting. You can use this parameter in conjunction with the addlayer and layerattribute parameters. This parameter overrides QuarkXPress Keep Runaround preferences.
Compatible with	eps, jpeg, png, postscript, qcddoc, qxpdoc, raw, pdf, screenpdf, swf, xml		
Alerts	This layer does not exist. Please verify the layer name.	HTTP Error #500 This alert displays if you specify an invalid layer name with the layer, layerattribute, or deletelayer parameter.	
	Specify a layer name.	HTTP Error #500 This alert displays if you do not specify a layer name with the layer, layerattribute, addlayer, or deletelayer parameter.	
	A layer with the same name already exists.	HTTP Error #500 This alert displays if you try to add a layer that already exists or change the name of a layer to a name is already used in the project.	

	Cannot change the name of the default layer.	HTTP Error #500 This alert displays if you try to change the name of the default layer.
	Cannot delete the default layer.	HTTP Error #500 This alert displays if you try to delete the default layer.
	Invalid parameter value.	HTTP Error #500 This alert displays if you do not specify additional attributes or specify attributes with invalid values in an addlayer or layerattribute request.
	This layer has been locked and cannot be modified.	HTTP Error #500 This alert displays if you try to add or modify an item on a locked layer.
Logs	See “Understanding logging."	
Example GET URL	To render a single layer, use a URL like the following:http://localhost:8080/doc.qxp?layer=layer1 To add a layer, use a URL like the following:http://localhost:8080/qxpdoc/doc.qxp?addlayer=NewLayer&visible=yes&suppressoutput=yes&locked=no To delete a layer, use a URL like the following: http://localhost:8080/qxpdoc/doc.qxp?deletelayer=Layer1 To render all layers in a project, use a URL like the following: http://localhost:8080/qxpdoc/doc.qxp?alllayers=true To set layer attributes, use a URL like the following:http://localhost:8080/qxpdoc/doc.qxp?layerattribute=Layer1&name=Layer2&visible=true&keeprunaround=true	
Example, object model	To add a new layer to a project, use code like the following:Layer layer = new Layer(); layer.name = “New Layer"; layer.operation = “CREATE"; RGBColor rgbcolor = new RGBColor(); layer.RGBColor = rgbcolor; layout.layer = new Layer[]{layer};To edit the properties of an existing layer, use the following object hierarchy:ModifierRequest < Project < Layout < LayerTo delete a layer, set its operation attribute to “DELETE".	
Notes	You cannot add, modify, or delete multiple layers in a single request.You cannot print layers whose visible and suppressoutput properties are set to false.You can render a	

	hidden or suppressed layer by referencing it with the layer parameter. Suppressed layers are rendered for the jpeg, png, and qxpdoc render types, but not for the pdf, postscript, and eps render types.You can use the deconstruct and getdocinfo request handlers to view information about the layers in a project.When you add a layer using addlayer, any unspecified attributes use the settings in the QuarkXPress Server layer preferences (Administration > Preferences > Renderer > Layers).If the visible property is set to false, the suppressoutput property is automatically set to true.		

layout

The layout render modifier lets you render a specific layout.

Parameters	layout	String	Lets you specify which layout to render. The first layout is layout 1.
Compatible with	eps, jpeg, png, postscript, raw, pdf, screenpdf		
Alerts	The requested layout does not exist.	HTTP Error #500This alert displays if you supply an invalid layout value.	
Logs	See “Understanding logging."		
Example GET URL	To render a layout by its layer ID, use a URL like the following:http://localhost:8080/png/sample.qxp?layout=2 To render a layout by its name, use a URL like the following:http://localhost:8080/png/sample.qxp?layout=Layout 2		

movepages

The movepages render modifier lets you move pages prior to rendering.

Parameters	movepages	String	Lets you specify which pages to move. You can use a single page

			number (for example, 2) or a range of pages with the starting and ending page numbers separated by a hyphen (for example, 2–5).
	afterpage	String	Lets you specify the page after which the page or pages should be moved. To move pages to the beginning of a layout, use afterpage=start. To move pages to the end of a layout, use afterpage=end.
Compatible with	eps, jpeg, png, postscript, qcddoc, qxpdoc, raw, pdf, screenpdf, xml		
Alerts	This page does not exist.	QuarkXPress Server Error #61	
	Invalid page range.	QuarkXPress Server Error #62	
	The specified page range cannot be moved there.	QuarkXPress Server Error #51	
	This page range is invalid.	QuarkXPress Server Error #146	
	Invalid parameter value.	QuarkXPress Server Error #10108	
Logs	See “Understanding logging."		

Example GET URL	To move pages 2–3 to after page 5, use a URL like the following:http://localhost:8080/abc.qxp?movepages=2–3&afterpage=5 To move page 7 to the beginning of a layout, use a URL like the following:http://localhost:8080/abc.qxp?movepages=7&afterpage= start
Example, object model	To move pages before rendering a layout, use code like the following:// STEP1: Create the QuarkXPress Server Request Context // and set the nescessary properties com.quark.qxpsm.QRequestContext requestCtx = new com.quark.qxpsm.QRequestContext(); Boolean responseAsURL = false; requestCtx.setDocumentName(docName); // STEP 2(SPECIFIC TO REQUESTS):Create the PDF // renderer request and embed it in the request context. the request context. PDFRenderRequest pdfreq = new PDFRenderRequest(); pdfreq.setMovePages("2-4"); pdfreq.setAfterPage("7"); requestCtx.setRequest(pdfreq); // STEP3: Create the service and call the // processRequest() API RequestService service = new RequestServiceStub(); com.quark.qxpsm.QContentData data = service.processRequest(requestCtx);
Notes	The movepages operation executes only after all other modifications are complete. For example, if you use movepages in a modify request, the pages are moved only after the modify request is complete.

page

The page render modifier lets you render a single page.

Parameters	page	Integer	Lets you specify which page to render.
Compatible with	eps, jpeg, png, postscript, qcddoc, raw, pdf, screenpdf		
Alerts	The requested page does not exist.	HTTP Error #500 This alert displays if you attempt to render a page that does not exist.	
	The renderer for	HTTP Error #406 This alert displays if	

	this image type has no way of rendering the desired objects.	you use a page parameter with the qxpdoc render type.
Logs	See “Understanding logging."	
Example GET URL	http://localhost:8080/png/sample.qxp?page=2	
Example, object model	To add a new page to an existing spread in a project, use code like the following:Spread spread = new Spread(); Page page = new Page(); page.UID = “5"; page.operation = “CREATE"; spread.page = new Page[]{page}; To edit the properties of an existing page, use the following object hierarchy:ModifierRequest < Project < Layout < Spread < PageTo delete a page, set its operation attribute to “DELETE".	
Notes	To render a page in a particular layout, use a URL like the following:http://localhost:8080/png/sample.qxp?layout=2&page=3	

pages

The pages render modifier lets you render multiple pages. The pdf and postscript namespaces support this parameter.

Parameters	pages	String (page range)	Lets you specify which pages to render.
Compatible with	eps, jpeg, png, postscript, raw, pdf, screenpdf		
Alerts	This page range is invalid.	HTTP Error #500 QuarkXPress Server Error #147 This alert displays if you try to render a page range that exceeds the number of pages in the project.	
	The renderer for this image type has no way of rendering the desired objects.	HTTP Error #406 This alert displays if you use the pages parameter with the jpeg, eps, png, or qxpdoc render type.	
Logs	See “Understanding logging."		

Example, GET URL	http://localhost:8080/pdf/sample.qxp?pages=2–4
Notes	To render pages in a particular layout, use a URL like the following:http://localhost:8080/pdf/sample.qxp?layout=2&pages=2,3

scale

The scale render modifier lets you specify the scale at which content is rendered.

Parameters	scale	Float	Lets you specify a scaling percentage. The valid values are from .1 (10%) to 8 (800%) on macOS or 6.92 (692%) on Windows.
Compatible with	eps, jpeg, png, raw		
Alerts	Invalid scale parameter.	HTTP Error #500 This alert displays if an invalid scale value is provided. What to do: Enter a valid scale value.	
Logs	See “Understanding logging."		
Example, GET URL	http://localhost:8080/png/sample.qxp?scale=2		

spread

The spread render modifier lets you render a single spread.

Parameters	spread	Integer	Lets you specify which spread to render. Spread numbers start with 1. The first spread is spread 1. In a facing-page document,

			spread 1 consists of the first page.
Compatible with	eps, jpeg, png, postscript, raw, pdf, screenpdf		
Alerts	The requested spread does not exist.	HTTP Error #500 This alert displays if you specify an invalid spread.	
Logs	See “Understanding logging."		
Example, GET URL	http://localhost:8080/png/sample.qxp?spread=2		
Example, Object Model	To add a spread to a project, use code like the following:Spread spread = new Spread(); spread.UID = “5”; spread.operation = “CREATE”; layout.spread = new Spread[]{spread};Spread is located at the following place in the object hierarchy:ModifierRequest < Project < Layout < SpreadTo delete a spread, set its operation attribute to “DELETE”.		

spreads

The spreads render modifier lets you render layouts in spreads mode, so that pages in spreads are rendered side-by-side rather than as individual pages.

Parameters	spreads	Boolean (1 | 0 | true | false | yes | no)	Lets you specify whether to render spreads (true) or individual pages (false).
Compatible with	eps, jpeg, png, postscript, raw, pdf, screenpdf		
Logs	See “Understanding logging."		
Example, GET URL	http://localhost:8080/pdf/sample.qxp?spreads=true		

suppresserrors

The suppresserrors render modifier forces QuarkXPress Server to render as much of a layout as it can, despite any errors that occur.

Parameters	suppresserrors	String	Forces the layout to render despite

			any errors that may occur.
Compatible with	pdf, qxpdoc, xml, postscript		
Logs	See “Understanding logging."		
Example GET URL	http://localhost:8080/png/sample.qxp?box=suppresserrors		

Using content modifiers

Content modifiers let you alter the content and formatting of boxes in layouts without using the XML modify parameter.

Inserting text

This topic explains how to import text into a box. Any existing text in the box is replaced.

Parameters	[box name]	String	The name of the target box.Specify the name and location of the imported file with the file: prefix. The imported file must be available to QuarkXPress Server.To import a file that is in a subfolder of the document pool on macOS, use a path like the following: file:subfolder:MyFile.extTo import a file that is in a subfolder of the document pool on Windows, use a path like the

		following: file:subfolder\MyFile.ext
Response	A preview of the project with the imported text.	
Alerts	File not found.	HTTP Error #404 QuarkXPress Server Error #–43 This alert displays if the imported file is not available to QuarkXPress Server.
Logs	If the request succeeds, a transaction success message is written to the QuarkXPress Server transaction log file. For example: 8/3/2005 11:27:42 — jpeg/sample.qxp — Type: image/jpeg — Size: 31715 — Client: 127.0.0.1If an alert displays, an error message is written to the QuarkXPress Server error log file. For example:8/10/2005 10:32:57 — Error — Error Code: –43 — File not found.	
Example, GET URL	http://localhost:8080/sample.qxp?Author=NewText http://localhost:8080/sample.qxp?TopStory=file:TopStory.doc	
Example, object model	Request object name: RequestParameterscom.quark.qxpsm.QRequestContext rc = new com.quark.qxpsm.QRequestContext();; if(!this.DocumentSettings1.documentName.Text.Equals("")) rc.documentName = this.DocumentSettings1.documentName.Text; // STEP 2 (SPECIFIC TO REQUESTS):Create the Box Param // renderer request and embed it in RequestParameters request = new RequestParameters(); NameValueParam nameValue1 = new NameValueParam(); nameValue1.paramName = this.boxname1.Text; if(!this.boxvalue1.Text.Equals("")) nameValue1.textValue = this.boxvalue1.Text; request.params = new NameValueParam[]{nameValue1}; rc.request = request; // Create the service and // call it with QRequestContext object RequestService svc = new RequestService(); com.quark.qxpsm.QContentData qc = svc.processRequest(rc);	
Notes	Box names are case-sensitive.Use “&" to change the contents of multiple boxes in one request. The general URL for the multiple-box request	

	is:http://localhost:8080/sample.qxp?text1=NewText1&text2=NewText2 where text1 and text2 are the names of the two different boxes.You can use “&" to change the contents of multiple boxes in one request. For example:http://localhost:8080/sample.qxp?Headline=headline.txt&Story=file:Story.docYou can import an XTags file generated by QuarkXPress.

Applying a font at import

This topic explains how to apply a font to a new text flow. When you use this method, QuarkXPress Server ignores the original font of the target text box and inserts the new text with the font specified by the parameter.

Parameters	fontname	String	The name of the font to be applied.
Response	A preview of the project with the font applied to the imported text.		
Alerts	The specified font is not available.	This alert displays if you specify a font that is unavailable.	
Logs	If the request succeeds, a transaction success message is written to the QuarkXPress Server transaction log file. For example:12/2/2005 16:24:13 — project2.qxp — Type: image/jpeg — Size: 11380 — Client: 127.0.0.1If an error occurs, the error message is written to the QuarkXPress Server Error Log. The transaction entry in the error log contains the date and time of the request, the error code, and the error message. The following is a sample of an error transaction log entry:12/2/2005 16:16:26 — Error — Error Code: –43 — File not found.		
Example, GET URL	To apply Comic Sans MS to text in the box named “HeadBox," use a URL like the following:http://localhost:8080/png/sample.qxp?HeadBox=Headline&fontname=Comic Sans MS		
Example, object model	Request object name: RequestParameterscom.quark.qxpsm.QRequestContext rc = new com.quark.qxpsm.QRequestContext(); if(!this.DocumentSettings1.documentName.Text.Equals(“”))		

	rc.documentName = this.DocumentSettings1.documentName.Text; // STEP 2(SPECIFIC TO REQUESTS):Create the fontname // renderer request and embed it in RequestParameters request = new RequestParameters(); NameValueParam nameValue1 = new NameValueParam(); nameValue1.paramName = this.boxname.Text; if(!this.boxvalue1.Text.Equals("")) nameValue1.textValue = this.fontname.Text; request.params = new NameValueParam[]{nameValue1}; rc.request = request; // Create the service and // call it with QRequestContext object RequestService svc = new RequestService(); com.quark.qxpsm.QContentData qc = svc.processRequest(rc);

Inserting a picture

This topic explains how to import a picture into an empty box or replace an existing picture with a new one.

Parameters	[box name[	String	The name of the target box.Specify the name and location of the imported file with the file: prefix. The imported file must be available to QuarkXPress Server.To import a file that is in a subfolder of the document pool on macOS, use a path like the following: file:subfolder:MyFile.extTo import a file that is in a subfolder of the document pool on

		Windows, use a path like the following: file:subfolder\MyFile.ext
Response	A preview of the project with the imported picture.	
Alerts	File not found.	HTTP Error #404 QuarkXPress Server Error #–43 This alert displays if the imported file is not available to QuarkXPress Server.
	The specified file failed to load in the picture box.	HTTP Error #500 This alert displays if you attempt to import an invalid picture file.
Logs	If the request succeeds, a transaction success message is written to the QuarkXPress Server transaction log file. For example:8/3/2005 11:27:42 — jpeg/sample.qxp — Type: image/jpeg — Size: 31715 — Client: 127.0.0.1If an alert is displayed, an error message is written to the QuarkXPress Server error log. The following is a sample of the error log entry:8/10/2005 10:39:07 — Error — Error Code: 10339 — The specified file failed to load in the picture box.	
Example, GET URL	http://localhost:8080/sample.qxp?PictureBox=file:FrenchOpen.pdf	
Example, object model	Request object name: RequestParameterscom.quark.qxpsm.QRequestContext rc = new com.quark.qxpsm.QRequestContext(); if(!this.DocumentSettings1.documentName.Text.Equals("")) rc.documentName = this.DocumentSettings1.documentName.Text; //STEP 2(SPECIFIC TO REQUESTS):Create the Box Param //renderer request and embed it in RequestParameters request = new RequestParameters(); NameValueParam nameValue1 = new NameValueParam(); nameValue1.paramName = this.boxname1.Text; if(!this.boxvalue1.Text.Equals("")) nameValue1.textValue = this.boxvalue1.Text; request.params = new NameValueParam[]{nameValue1}; rc.request = request; //Create the service and call it with QRequestContext object	

	RequestService svc = new RequestService(); com.quark.qxpsm.QContentData qc = svc.processRequest(rc);
Notes	Box names are case-sensitive.You can use “&” to change the contents of multiple boxes in one request. For example:http://localhost:8080/sample.qxp?Logo=file:logo.jpeg&TopPicture=file:TopPicture.eps

Saving a projects with a new name

The saveas content modifier lets you save modified QuarkXPress projects in any supported format to the document pool or to any network location accessible to QuarkXPress Server.

If you send a saveas request to QuarkXPress Server Manager using HTTP or the Web services interface while the common doc pool switch is set to off in the QuarkXPress Server Manager client, the file is saved to all registered QuarkXPress Server instances. If the common doc pool is enabled, the file can be saved to any one registered QuarkXPress server instance.

Parameters	newname	String	Lets you specify a name for the saved-as project.
	path	String	Lets you specify a location for the saved-as project (other than the document pool).
	savetopool	true | false	Lets you specify whether the project should be saved to the document pool.The default value for this paramter is true. However, if you specify a path value, the default value changes to

			false, which means if you want the project saved to the document pool, you must explicitly set savetopool to true.
	replace	true | false	Lets you specify whether the saved project should replace a project with the same name. The default value is true.
Response	The message “Document successfully saved."		
Alerts	File not found.	HTTP Error #404 QuarkXPress Server Error #–43 This alert displays if you supply an incorrect file name or the file is not available to QuarkXPress Server.	
	Bad filename/ pathname.	HTTP Error #404 QuarkXPress Server Error #–43 This alert displays if you supply an incorrect file name or the file is not available to QuarkXPress Server.	
	The file path is invalid.	HTTP Error #500 This alert displays if you supply an invalid path parameter. What to do: Specify the correct file path with the path parameter.	
	The specified folder is Read-Only.	HTTP Error #500 This alert displays if you try to save a project to a folder with read-only access.	
Logs	If the request succeeds, a transaction success message is written to the QuarkXPress Server transaction log file. For example: 11/16/2005 15:41:42 — saveas/5mb.qxp — Type: — Size: 28 — Client: 127.0.0.1If an alert displays, an error message is written to the QuarkXPress Server error log file.		

	For example:11/16/2005 15:42:12 — Error — Error Code: 10371 — The file path is invalid.
Example, GET URL	To save a PDF file named “Customer1.pdf" in the folder HDD:temp and also in the document pool, use a URL like the following. Note that this URL will cause the saved-as file to replace any existing file with the same name.http://localhost:8080/saveas/pdf/sample.qxp?newname=Customer1&path=HDD:temp&savetopool=true
Example, object model	Request object name: SaveAsRequestcom.quark.qxpsm.QRequestContext rc = new com.quark.qxpsm.QRequestContext(); if(!this.DocumentSettings1.documentName.Text.Equals(“”)) rc.documentName = this.DocumentSettings1.documentName.Text; // STEP 2 (SPECIFIC TO REQUESTS): // Create the Save as request and chain it to the document context SaveAsRequest saveasreq = new SaveAsRequest(); saveasreq.newName = this.newname.Text; if((this.path.Text != null) && (!this.path.Text.Equals(“”))) saveasreq.newFilePath = this.path.Text; saveasreq.replaceFile = this.replace.Checked.ToString(); saveasreq.saveToPool = this.savetopool.Checked.ToString(); rc.request = saveasreq; // Create the service and call it with QRequestContext object RequestService svc = new RequestService(); com.quark.qxpsm.QContentData qc = svc.processRequest(rc);

Importing XML with placeholders

This topic explains how to import XML data into boxes using QuarkXPress placeholders.

To use this feature, you must have a QuarkXPress project that has been set up with placeholders that correspond to the element types in a source XML file. For more information, see *A Guide to XML Import*.

Parameters	thexmldoc	XML	Lets you specify the XML file containing the data to import.

		The path can be absolute or relative to the location of the XML file. You can also supply XML as a string.
layout	String	Lets you specify which layout to render. The first layout is layout 1. You can also specify a layout by name.
paginate	XML	Lets you specify the XML file containing the data to import. The paginate parameter reates enough pages in the target layout to accommodate the records in the XML.This parameter works only with the pdf, postscript, and qxp render types. If you use it with any other render type, the server returns only the first page of the paginated layout.If you do not supply an XML string or file (for example:

		http://localhost:8080/pdf/Sample.qxp?paginate), QuarkXPress Server attempts to use the XML file that was associated with the layout in QuarkXPress.
Response	The layout with the imported XML.	
Alerts	Invalid XML String	HTTP Error #500 This alert displays if you supply an invalid XML string in the thexmldoc parameter.
Logs	If the project is successfully rendered, a transaction success message is written to the QuarkXPress Server transaction log file. The transaction entry consists of the date and time of the request, the render type, the project name, the type of response produced by the server, the size of the response returned in bytes, and the client IP address. The following is a sample of a transaction entry: 8/5/2005 18:11:54 — sample.qxp — Type: image/jpeg — Size: 65982 — Client: 127.0.0.1If an alert displays, an error message is written to the QuarkXPress Server error log file. For example:8/9/2005 12:38:42 — Error — Error Code: 10396 — Invalid XML String.	
Example, GET URL	When QuarkXPress Server is running on Windows, use a URL like the following:http://localhost:8080/Sample.qxp?thexmldoc=<?xml version="1.0"?> <BookReview><Book><Title>C:\Autumn.jpg</Title> <Author> Brian Kernighan and Dennis Ritchie</Author> </Book></BookReview> When QuarkXPress Server is running on macOS, use a URL like the following: http://localhost:8080/Sample.qxp?thexmldoc=<?xml version= “1.0"?> <BookReview><Book><Title>/Volumes/MacHD/Pictures/abc.tiff</ Title> <Author> Brian Kernighan and Dennis	

	Ritchie</Author> </Book></BookReview>Alternatively, you can specify a path to a file containing the XML: http://localhost:8080/Sample.qxp?paginate= file:MacHD:Sample.xml
Example, object model	Request object names: XMLImportRequestcom.quark.qxpsm.QRequestContext rc = new com.quark.qxpsm.QRequestContext(); if(!this.DocumentSettings1. documentName.Text.Equals("")) rc.documentName = this.DocumentSettings1.documentName.Text; // STEP 2 (SPECIFIC TO REQUESTS): Create the XML Import request XMLImportRequest xmlimportreq = new XMLImportRequest(); xmlimportreq.XMLDocument = this.thexmldoc.Text; rc.request = xmlimportreq; // STEP 3(SPECIFIC TO REQUESTS): Create the JPEG renderer request JPEGRenderRequest jpreq = new JPEGRenderRequest(); xmlimportreq.request = jpreq; // Create the service and call it with QRequestContext object RequestService svc = new RequestService(); com.quark.qxpsm.QContentData qc = svc.processRequest(rc);

Updating article geometry and content

This topic explains how to update the geometry and contents of a QuarkCopyDesk article using another article file or an article in a QuarkXPress project.

Parameter	updategeometry	String	If you use this parameter with updatefromfile, this lets you specify the file in which you want to update the geometry. If you use this parameter with updatetofile, this lets you specify

			the QuarkXPress project with which you want to update the geometry of an article.
	updatecontent	String	Lets you specify the file in which you want to update the content. Can be used only with updatefromfile.
Response	WHAT DOES THIS RETURN?		
Alerts	WHAT ERRORS APPLY?		
Logs	If the project is successfully rendered, a transaction success message is written to the QuarkXPress Server transaction log file. The transaction entry consists of the date and time of the request, the render type, the project name, the type of response produced by the server, the size of the response returned in bytes, and the client IP address. The following is a sample of a transaction entry: 8/5/2005 18:11:54 — sample.qxp — Type: image/jpeg — Size: 65982 — Client: 127.0.0.1If an alert displays, an error message is written to the QuarkXPress Server error log file. For example:8/9/2005 12:38:42 — Error — Error Code: 10396 — Invalid XML String.		
Example, GET URL	To update the geometry of an article using the geometry of another article, use a URL like the following:http://localhost:8080/updategeometry/destination.qcd?updatefromfile=source.qcd To update the geometry of an article in a QuarkXPress file using the geometry of a QuarkCopyDesk article file, use a URL like the following:http://localhost:8080/updatecontent/destination.qxp?updatefromfile=source.qcd&articleid=1 To update the geometry of a QuarkCopyDesk article file using the geometry of an article in a QuarkXPress project, use a URL		

	like the following:http://localhost:8080/updategeometry/source.qxp?updatetofile=destination.qcd&articleid=1
Example, object model	CAN YOU DO THIS WITH QXPSM? IF SO, HOW?Request object names: XMLImportRequestcom.quark.qxpsm.QRequestContext rc = new com.quark.qxpsm.QRequestContext(); if(!this.DocumentSettings1. documentName.Text.Equals("")) rc.documentName = this.DocumentSettings1.documentName.Text; // STEP 2 (SPECIFIC TO REQUESTS): Create the XML Import request XMLImportRequest xmlimportreq = new XMLImportRequest(); xmlimportreq.XMLDocument = this.thexmldoc.Text; rc.request = xmlimportreq; // STEP 3(SPECIFIC TO REQUESTS): Create the JPEG renderer request JPEGRenderRequest jpreq = new JPEGRenderRequest(); xmlimportreq.request = jpreq; // Create the service and call it with QRequestContext object RequestService svc = new RequestService(); com.quark.qxpsm.QContentData qc = svc.processRequest(rc);

Highlighting text in rendered output

To apply highlighting to text in rendered output, use XML like the following.

```
<RICHTEXT BACKGROUNDCOLOR="Yellow">This text is highlighted.</RICHTEXT>
```

➡ Highlighting is applied only to rendered output. It is not retained in the QuarkXPress project.

Using XML modify

The modify parameter lets you modify a QuarkXPress project using XML.

➡ This topic covers the modify parameter when it is used without the construct namespace. You can also use the modify parameter to specify an XML file to use when constructing a project; for more information, see "*Constructing a project*".

The xml namespace takes two arguments: the name of the project to be modified, and a modify parameter with the string or the path of the XML file that describes how to create the project:

http://QXPServer8:8080/project1.qxp?modify=
file:path to XML file on server http://QXPServer8 :8080/
project1.qxp?modify=XML string

You can also modify QuarkCopyDesk articles. To modify a QuarkCopyDesk article:

http://localhost:8080/copydesk/abc.qcd?modify=
file:XMLfile.xml

schema	Modifier schema		
Parameters	modify	String	Lets you specify an XML file or string that describes how to create a project. The path can be absolute or a relative path in the document pool. Use the file: indicator to specify the path.Note that you can also include an XML file as part of a multipart HTTP request.
Example GET URL	http://QXPServer8:8080/project1.qxp?modify=file:sample.xml		
Example XML	This XML deletes page 2 of a QuarkXPress layout: <PROJECT> <LAYOUT> <ID NAME="Layout 1" /> <SPREAD> <ID UID="1" /> <PAGE OPERATION="DELETE"> <ID UID="2" /> </PAGE> </SPREAD> </LAYOUT> </PROJECT>		
Response	The updated QuarkXPress project.		
Logs	If the request succeeds, a transaction success message is written to the QuarkXPress Server transaction log file. For example:8/3/2005 11:27:42 — jpeg/sample.qxp — Type: image/jpeg — Size: 31715 — Client: 127.0.0.1If an alert is		

displayed, an error message is written to the QuarkXPress Server error log. The following is a sample of the error log entry:8/10/2005 10:39:07 — Error — Error Code: 10339 — The specified file failed to load in the picture box.

Modifying box properties and content

To modify box properties and content, use the following parameters in the Modifier schema:

- "*BOX*"
- "*ID*"
- "*TEXT*"
- "*PICTURE*"
- "*GEOMETRY*"
- "*CONTENT*"
- "*SHADOW*"
- "*FRAME*"
- "*BORDER*"
- "*PLACEHOLDER*"
- "*METADATA*"

The following XML shows how some of these parameters work.

```
<?xml version="1.0" encoding="UTF-8"?>
<PROJECT>
 <LAYOUT>
  <ID NAME="Layout 1"/>
  <SPREAD>
   <ID UID="1"/>
   <BOX BOXTYPE="CT_TEXT">
    <ID NAME="SERVICES"/>
    <GEOMETRY>
     <MOVEUP>50</MOVEUP>
     <MOVELEFT>30</MOVELEFT>

<ALLOWBOXONTOPASTEBOARD>true</ALLOWBOXONTOPASTEBOA
RD>
    </GEOMETRY>
    <CONTENT CONVERTQUOTES="true">
     HD:QuarkXPress:DocPool:Services.txt</CONTENT>
```

```
    </BOX>
    <BOX BOXTYPE="CT_TEXT">
     <ID NAME="FAMILY"/>
     <GEOMETRY>
      <MOVERIGHT>20</MOVERIGHT>
      <MOVEDOWN>30</MOVEDOWN>

<ALLOWBOXONTOPASTEBOARD>true</ALLOWBOXONTOPASTEBOA
RD>
       <ALLOWBOXOFFPAGE>true</ALLOWBOXOFFPAGE>
     </GEOMETRY>
    </BOX>
    <BOX BOXTYPE="CT_TEXT">
     <ID NAME="PRODUCTS"/>
     <GEOMETRY>
      <GROWACROSS>44</GROWACROSS>
      <GROWDOWN>30</GROWDOWN>

<ALLOWBOXONTOPASTEBOARD>false</ALLOWBOXONTOPASTEBO
ARD>
     </GEOMETRY>
    </BOX>
    <BOX BOXTYPE="CT_PICT">
     <ID NAME="MAP"/>
     <GEOMETRY>
      <SHRINKACROSS>30</SHRINKACROSS>
      <SHRINKDOWN>30</SHRINKDOWN>
     </GEOMETRY>
    </BOX>
    <BOX COLOR="Blue" BOXTYPE="CT_PICT">
     <ID NAME="CONTACT"/>
     <GEOMETRY>
      <STACKINGORDER>BRINGTOFRONT</STACKINGORDER>
      <RUNAROUND TYPE="ITEM" TOP="4" RIGHT="4"
       LEFT="4" BOTTOM="4"/>
      <ALLOWBOXOFFPAGE>false</ALLOWBOXOFFPAGE>
     </GEOMETRY>
    </BOX>
   </SPREAD>
  </LAYOUT>
</PROJECT>
```

If you know the UID attribute of a <CONTENT> element, you can insert content into that <CONTENT> element without having to specify where it is. For example:

```
<PROJECT>
 <CONTENT UID="0">NewPicture.jpg</CONTENT>
</PROJECT>
```

You can also use a <CONTENT> element to insert additional text between two <RICHTEXT> elements, like so:

```
<PROJECT>
  <STORY>
    <ID UID="0"/>
    <PARAGRAPH PARASTYLE="Normal">
      <RICHTEXT>Text before external file</RICHTEXT>
      <CONTENT>file:NewText.doc<CONTENT>
      <RICHTEXT>Text after external file</RICHTEXT>
    </PARAGRAPH>
  </STORY>
</PROJECT>
```

Response	A preview of the QuarkXPress project with a new box created in the specified position.	
Alerts	File not found.	HTTP Error #404 QuarkXPress Server Error #–43 This alert displays if you specify an invalid XML file or request a document that is not available to QuarkXPress Server.
	Bad filename/pathname.	HTTP Error #404 QuarkXPress Server Error #–37 This alert displays if you specify an invalid file name or path.
	The XML document is not valid or well formed.	HTTP Error #500 This alert displays if the XML you supply is not well-formed or does not adhere to the Modifier schema.
	The XML document contains an invalid tag value.	HTTP Error #500 This alert displays if you supply an invalid value in the XML.
Logs	If the request succeeds, a transaction success message is written to the QuarkXPress Server transaction log file. For example: 8/3/2005 11:27:42 — jpeg/sample.qxp — Type: image/jpeg — Size: 31715 — Client: 127.0.0.1If an alert displays, an error message is written to the QuarkXPress Server error log file. For example:	
Example GET URL	When QuarkXPress Server is running on Windows, use a URL like the following:http://localhost:8080/sample.qxp?modify=	

	file:C:\updateBox.xml When QuarkXPress Server is running on macOS, use a URL like the following:http://localhost:8080/sample.qxp?modify= file:MacHD:xml:updateBox.xml You can also supply a string that consists of valid XML commands. For example:http://localhost:8080/sample.qxp?modify= <PROJECT><LAYOUT><ID UID="Layout1"/><SPREAD><ID UID="1"/> <BOX BOXTYPE="CT_PICT" COLOR="Blue" SHADE="50" OPACITY="50"> <ID NAME="MOUNTAINS"/><CONTENT> file:Services.eps</CONTENT> </BOX></SPREAD></LAYOUT></PROJECT>
Example 1, object model	Request object names:ModifierRequest ModifierRequestContents Layout ID Box Geometry Runaround ModifierFileRequestFor ModifierFileRequest, the member contents are used to set the file path or send the XML itself. com.quark.qxpsm.QRequestContext rc = new com.quark.qxpsm.QRequestContext(); if(!this.DocumentSettings1.documentName.Text.Equals("")) rc.documentName = this.DocumentSettings1.documentName.Text; //STEP 2(SPECIFIC TO REQUESTS): //Create the BOX modifier renderer request and //embed it in request context ModifierRequest request = new ModifierRequest(); Project contents = new Project(); Geometry geo = new Geometry(); geo.moveUp = this.moveup.Text; geo.color = this.color.Text; geo.growDown = this.growdown.Text; geo.shrinkAcross = this.shrinkacross.Text; Box box = new Box(); box.UID = this.Boxid.Text; box.geometry = geo; Layout layout1 = new Layout(); layout1.name = this.layout.Text; layout1.boxes = new Box[]{box}; if(this.runaround.Checked == true) { Runaround runaround = new Runaround(); runaround.type = this.runaroundtype.Text; runaround.top = this.top.Text; runaround.left = this.left.Text; runaround.right = this.right.Text; geo.runaround = runaround; } contents.layouts = new Layout[]{layout1}; request.project = contents; rc.request = request; //Create the service and call it with QRequestContext object RequestService svc = new RequestService(); com.quark.qxpsm.QContentData qc

	= svc.processRequest(rc);
Example 2, object model	To edit the geometrical properties of an existing box in a QuarkXPress project, use the following object hierarchy:ModifierRequest < Project < Layout < Spread < Box < Geometry The Geometry object has the following properties:allowBoxOffPage allowBoxOnToPasteBoard angle growAcross growDown layer linestyle (of type 'Linestyle') moveDown moveLeft moveRight moveUp page position (of type 'Position') runaround (of type 'Runaround') shape shrinkAcross shrinkDown stackingOrder suppressOutput The Runaround object has the following properties:bottom edited invert left noise outset outsideOnly pathName restrictToBox right smoothness threshold top type

Creating boxes

To create a new box, use the following parameters in the Modifier schema:

- "*BOX*"
- "*ID*"
- "*TEXT*"
- "*PICTURE*"
- "*GEOMETRY*"
- *CONTENT*"
- "*SHADOW*"
- "*FRAME*"
- "*BORDER*"

The following XML shows how some of these parameters work.

```
<PROJECT>
 <LAYOUT>
  <ID UID="layout 1"/>
  <SPREAD>
  <ID UID="1"/>
   <ID/>
   <BOX OPERATION="CREATE" BOXTYPE="CT_PICT">
    <ID NAME="PRODUCTS"/>
    <GEOMETRY PAGE="2" SHAPE="SH_RECT">
```

```
        <POSITION>
         <TOP>5</TOP>
         <LEFT>5</LEFT>
         <BOTTOM>10</BOTTOM>
         <RIGHT>10</RIGHT>
        </POSITION>
       </GEOMETRY>
      </BOX>
     </SPREAD>
    </LAYOUT>
</PROJECT>
```

Response	A preview of the QuarkXPress project with new box created in specified position.	
Alerts	File not found.	HTTP Error #404 QuarkXPress Server Error #–43 This alert displays if you specify an invalid XML file or request a document that is not available to QuarkXPress Server.
	Bad filename/ pathname.	HTTP Error #404 QuarkXPress Server Error #–37 This alert displays if you specify an invalid file name or path.
	The XML document is not valid or well formed.	HTTP Error #500This alert displays if the XML you supply is not well-formed or does not adhere to the Modifier schema.
	The XML document contains an invalid tag value.	HTTP Error #500 This alert displays if you supply an invalid value in the XML.
Logs	If the request succeeds, a transaction success message is written to the QuarkXPress Server transaction log file. For example:The following is a sample of a transaction entry: 8/3/2005 11:27:42 — jpeg/sample.qxp — Type: image/jpeg — Size: 31715 — Client: 127.0.0.1If an alert displays, an error message is written to the QuarkXPress Server error log file. For example: 4/12/2007 14:51:50 — Error — Error Code: 10207 — The XML document is not valid or well formed. Project: /table.qxp	
Example, GET URL	When QuarkXPress Server is running on Windows, use a URL like the following:http://localhost:8080/sample.qxp?modify=	

	file:C:\createBox.xml When QuarkXPress Server is running on macOS, use a URL like the following:http://localhost:8080/sample.qxp?modify=file:MacHD:xml:createBox.xml You can also supply a string that consists of valid XML commands. For example:http://localhost:8080/sample.qxp?modify=<PROJECT><LAYOUT> <ID UID="layout 1"/><SPREAD><ID UID="1"/><ID/> <BOX OPERATION="CREATE" BOXTYPE="CT_PICT"><ID NAME="PRODUCTS"/> <GEOMETRY PAGE="2" SHAPE="SH_RECT"><POSITION><TOP>5</TOP> <LEFT>5</LEFT><BOTTOM>10</BOTTOM><RIGHT>10</RIGHT></POSITION> </GEOMETRY></BOX></SPREAD></LAYOUT></PROJECT>
Example, object model	To create a new box, use code like the following:Spread spread = new Spread(); Box box = new Box(); box.name = “textbox1”; Geometry geometry = new Geometry(); Position position = new Position(); position.top = “110”; position.left = “89”; position.bottom = “220”; position.right = “300”; geometry.position = position; geometry.shape = “SH_RECT”; geometry.page = “1”; geometry.layer = “Default”; box.geometry = geometry; box.boxType = “CT_TEXT”; box.operation = “CREATE”; spread.box = new Box[]{box}; Use the following object hierarchy:ModifierRequest < Project < Layout < Spread < Box < Geometry

Deleting boxes

To delete a box, use the following parameters in the Modifier schema:

- "*BOX*"
- "*ID*"

The following XML shows how these parameters work.

```
<PROJECT>
 <LAYOUT>
  <ID UID="Layout 1"/>
  <SPREAD>
   <ID UID="1"/>
   <BOX OPERATION="DELETE">
    <ID NAME="SERVICES"/>
   </BOX>
  </SPREAD>
```

</LAYOUT>
</PROJECT>

Response	A preview of the QuarkXPress project with the box deleted.	
Alerts	File not found.	HTTP Error #404 QuarkXPress Server Error #–43 This alert displays if you specify an invalid XML file or request a document that is not available to QuarkXPress Server.
	Bad filename/ pathname.	HTTP Error #404 QuarkXPress Server Error #–37 This alert displays if you specify an invalid file name or path.
	The XML document is not valid or well formed.	HTTP Error #500 This alert displays if the XML you supply is not well-formed or does not adhere to the Modifier schema.
	The XML document contains an invalid tag value.	HTTP Error #500 This alert displays if you supply an invalid value in the XML.
Logs	If the request succeeds, a transaction success message is written to the QuarkXPress Server transaction log file. For example: 8/3/2005 11:27:42 — jpeg/sample.qxp — Type: image/jpeg — Size: 31715 — Client: 127.0.0.1If an alert displays, an error message is written to the QuarkXPress Server error log file. For example:	
Example GET URL	When QuarkXPress Server is running on Windows, use a URL like the following:http://localhost:8080/sample.qxp?modify= file:C:\deleteBox.xml When QuarkXPress Server is running on macOS, use a URL like the following:http://localhost:8080/sample.qxp?modify= file:MacHD:xml:deleteBox.xml You can also supply a string that consists of valid XML commands. For example: http://localhost:8080/sample.qxp?modify= <PROJECT><LAYOUT><ID UID="Layout1"/><SPREAD> <ID UID="1"/><BOX OPERATION="DELETE"> <ID NAME="HISTORY"/></BOX></SPREAD> </LAYOUT></PROJECT>	

Notes	You can use the xml namespace or Telegraph XTensions software to determine the ID or name of the box you want to delete.

Grouping and ungrouping items

To group boxes using XML modify, use XML like the following:

```
<BOX BOXTYPE="CT_TEXT" COLOR="White">
 <ID NAME="MainStoryText" UID="217"/>
</BOX>

<BOX BOXTYPE="CT_PICT">
 <ID NAME="MainStoryPhoto" UID="218"/>
</BOX>

<GROUP>
 <ID NAME="MainStoryGroup" UID="300"
OPERATION="CREATE"
/>
 <BOXREF NAME="MainStoryText" UID="217"/>
 <BOXREF NAME="MainStoryPhoto" UID="218"/>
</GROUP>
```

To add a box to an existing group, use XML like the following:

```
<GROUP>
 <ID NAME="MainStoryGroup" UID="300"/>
 <BOXREF NAME="MainStoryText" UID="217"/>
 <BOXREF NAME="MainStoryPhoto" UID="218"
OPERATION="CREATE"
/>
</GROUP>
```

To remove a box from an existing group, use XML like the following:

```
<GROUP>
 <ID NAME="MainStoryGroup" UID="300"/>
 <BOXREF NAME="MainStoryHead" UID="216"/>
 <BOXREF NAME="MainStoryText" UID="217"/>
 <BOXREF NAME="MainStoryPhoto" UID="218"
OPERATION="DELETE" />
</GROUP>
```

To ungroup an existing group, use XML like the following:

```
<GROUP>
 <ID NAME="MainStoryGroup" UID="300" OPERATION="DELETE" />
</GROUP>
```

To proportionally scale all of the items in a group, add a <GEOMETRY> element that indicates the new size of the group, like so:

```
<GROUP>
 <ID NAME="MainStoryGroup" UID="300"/>

 <GEOMETRY>  <POSITION>  <TOP>10.0</TOP>
<LEFT>10.0</LEFT>  <BOTTOM>50.0</BOTTOM>
<RIGHT>70.0</RIGHT>  </POSITION>  </GEOMETRY>

</GROUP>
```

- The order of the <BOXREF> elements in a <GROUP> indicates the order in which the boxes were selected prior to grouping. The z-order of boxes in the layout is determined by the order of the <BOX> elements in the XML, from rearmost to frontmost.
- XML representations of groups created by versions of QuarkXPress Server prior to 8.1 are ignored during construct and modify calls, as they were in earlier versions of QuarkXPress Server.

Modifying text attributes

You can use the modify parameter to change the attributes of text in a QuarkXPress project. All modifications are done on a text box basis. To modify text properties, use the following parameters in the Modifier schema:

- "*BOX*"
- "*ID*"
- "*TEXT*"
- "*STORY*"
- "*PARAGRAPH*"
- "*FORMAT*"
- "*SHADINGSTYLE (Modifier schema)*"
- "*DROPCAP*"
- "*TABSPEC*"
- "*TAB*"
- "*RULE*"
- "*RICHTEXT*"

The following XML shows how some of these parameters work.

```
<PROJECT>
 <LAYOUT>
  <ID UID="Layout 1"/>
  <SPREAD>
```

```
    <ID UID="1"/>
    <BOX BOXTYPE="CT_TEXT">
     <ID NAME="ABOUT"/>
     <TEXT>
      <STORY CLEAROLDTEXT="true" FITTEXTTOBOX="true"
       CONVERTQUOTES="true">
       <RICHTEXT FONT="Castellar" PLAIN="true"/>
      </STORY>
     </TEXT>
    </BOX>
    <BOX BOXTYPE="CT_TEXT">
     <ID NAME="HISTORY"/>
     <TEXT>
      <STORY>
       <PARAGRAPH>
        <FORMAT ALIGNMENT="RIGHT">
         <SHADINGSTYLE COLOR="Yellow"
                SHADE="30%"
                OPACITY="20%"
                LENGTH="COLUMN"
                CLIPTOBOX="true"
                RIGHTPADDING="4pt"
                BOTTOMPADDING="2pt"
                LEFTPADDING="2pt"
                TOPPADDING="1pt"/>
        </FORMAT>
        <RICHTEXT SIZE="12">This text is 12pt and right
         justified.</RICHTEXT>
       </PARAGRAPH>
      </STORY>
     </TEXT>
    </BOX>
    <BOX BOXTYPE="CT_TEXT">
     <ID NAME="PRODUCTS"/>
     <TEXT>
      <STORY>
       <RICHTEXT BOLD="true">This is bold text.</RICHTEXT>
       <RICHTEXT BOLD="true" COLOR="Red" ITALIC="true"
        SIZE="20">This text is bold, red, italic, and 20pt.
       </RICHTEXT>
      </STORY>
     </TEXT>
    </BOX>
   </SPREAD>
  </LAYOUT>
</PROJECT>
```

Response	A preview of a QuarkXPress project with the values in the ModifierXT tags applied on text boxes.	
Alerts	File not found.	HTTP Error #404 QuarkXPress Server Error #–43 This alert displays if you specify an invalid XML file or request a document that is not available to QuarkXPress Server.
	Bad filename/ pathname.	HTTP Error #404 QuarkXPress Server Error #–37 This alert displays if you specify an invalid file name or path.
	The XML document is not valid or well formed.	HTTP Error #500 This alert displays if the XML you supply is not well-formed or does not adhere to the Modifier schema.
	There is no box with the specified identifier.	HTTP Error #500 This alert displays if the box specified by the child text node of an <ID> element does not exist.
	The text size value is outside the valid range.	HTTP Error #500 This alert displays if the value specified in a <SIZE> element is invalid. What to do: Specify a value between 2 and 720 points.
	The specified color is not available to the document	HTTP Error #500 This alert displays if the value specified in a <COLOR> element is invalid.
	The specified font is not available	HTTP Error #500 This alert displays if the value specified in a <FONT> element is invalid or the specified font is not present on the server.
	The XML document contains an invalid tag value.	HTTP Error #500 This alert displays if you supply an invalid value in the XML.
	The specified box cannot be modified.	HTTP Error #500 This alert displays if you try to modify text properties on a box that is not a text box.

Logs	If the request succeeds, a transaction success message is written to the QuarkXPress Server transaction log file. For example: 8/3/2005 11:27:42 — jpeg/sample.qxp — Type: image/jpeg — Size: 31715 — Client: 127.0.0.1If an alert displays, an error message is written to the QuarkXPress Server error log file. For example:8/5/2005 13:32:10 — Error — Error Code: 10006 — There is no box with the specified identifier.
Example GET URL	When QuarkXPress Server is running on Windows, use a URL like the following:http://localhost:8080/sample.qxp?modify= file:C:\modifier.xml When QuarkXPress Server is running on macOS, use a URL like the following:http://localhost:8080/sample.qxp?modify= file:MacHD:xml:modifier.xml You can also supply a string that consists of valid XML commands. For example: http://localhost:8080/sample.qxp?modify= <PROJECT><LAYOUT><ID UID="1"/><SPREAD><ID UID="1"/> <BOX BOXTYPE="CT_TEXT"><ID NAME="BACKGROUND"/> <TEXT><STORY><RICHTEXT FONT="Castella" PLAIN="true"> This is text.</RICHTEXT></STORY></TEXT></BOX> </SPREAD></LAYOUT></PROJECT>
Example 1, object model	Request object names: ModifierRequest ModifierStreamRequest Project RichText Text ID Box Layout ModifierFileRequestFor ModifierFileRequest, the member contents are used to set the file path or send the XML itself. com.quark.qxpsm.QRequestContext rc = new com.quark.qxpsm.QRequestContext(); if(!this.DocumentSettings1.documentName.Text.Equals("")) rc.documentName = this.DocumentSettings1.documentName.Text; // STEP 2 (SPECIFIC TO REQUESTS):Create the Text Modifier // renderer request and embed it in request context ModifierRequest textReq = new ModifierRequest(); Project contents = new Project(); RichText richText1 = new RichText(); richText1.value = this.text1.Text; richText1.color = this.color1.Text; Text boxText1 = new Text(); Story story = new Story(); story.richText = new RichText[]{richText1};

	boxText1.story = story; if(this.fittextbox1.Checked) boxText1.fitTextToBox = “true”; if(this.clearoldtext1.Checked) boxText1.clearOldText = “true”; Box box1 = new Box(); box1.UID = txtBox1; box1.text = boxText1; Layout layout1 = new Layout(); layout1.name = layoutText; layout1.boxes = new Box[]{box1}; contents.layouts = new Layout[]{layout1}; textReq.contents = contents; rc.request = textReq; // Create the service and call it with QRequestContext object RequestService svc = new RequestService(); com.quark.qxpsm.QContentData qc = svc.processRequest(rc);
Example 2, object model	To edit the properties of an existing text box in a QuarkXPress project, use the following object hierarchy:ModifierRequest < Project < Layout < Spread < Box < Text < Story < Paragraph < RichText For a list of the RichText object's properties, see the JavaDoc installed with QuarkXPress Manager. The Story object also contains some text-related properties: fitTextToBox, includeStylesheets, convertQuotes, and clearOldText.
Notes	The <FITTEXTTOBOX> attribute depends on two preferences: Allow Text to Grow and Font Size. To set these preferences in QuarkXPress Server, choose QuarkXPress > Server > Preferences and then click Modifier in the list on the left.

Modifying picture properties

You can modify the properties (such as origin, scale, angle, skew, and orientation) of pictures in a QuarkXPress project with XML. To modify picture properties, use the following parameters in the Modifier schema:

- "*BOX* “
- "*ID*“
- "*PICTURE*“

The following XML shows how some of these parameters work.

```
<PROJECT>
 <LAYOUT>
  <ID UID="1"/>
  <SPREAD>
   <ID UID="1"/>
```

```
    <BOX BOXTYPE="CT_PICT">
     <ID NAME="PEOPLE"/>
     <PICTURE SCALEACROSS="50" SCALEDOWN="50"
OFFSETACROSS="20"
      OFFSETDOWN="20"/>
    </BOX>
    <BOX BOXTYPE="CT_PICT">
     <ID NAME="MOUNTAINS"/>
     <PICTURE FIT="CENTERPICTURE" ANGLE="30" SKEW="30"
      FLIPHORIZONTAL="false"/>
    </BOX>
    <BOX BOXTYPE="CT_PICT">
     <ID NAME="OFFICES"/>
     <PICTURE FIT="FITPICTURETOBOX" ANGLE="30" SKEW="30"
      FLIPHORIZONTAL="false"/>
    </BOX>
    <BOX BOXTYPE="CT_PICT">
     <ID NAME="PRODUCTS"/>
     <PICTURE FIT="FITPICTURETOBOX" ANGLE="30" SKEW="30"
      FLIPHORIZONTAL="false"/>
    </BOX>
    <BOX BOXTYPE="CT_PICT">
     <ID NAME="SERVICES"/>
     <PICTURE FIT="FITPICTURETOBOXPRO"/>
    </BOX>
   </SPREAD>
  </LAYOUT>
</PROJECT>
```

Response	A preview of the QuarkXPress project with image modifier tags applied to the picture boxes.	
Alerts	File not found.	HTTP Error #404 QuarkXPress Server Error #–43 This alert displays if you specify an invalid XML file or request a document that is not available to QuarkXPress Server.
	Bad filename/ pathname.	HTTP Error #404 QuarkXPress Server Error #–37 This alert displays if you specify an invalid file name or path.
	The XML document is not valid or well formed.	HTTP Error #500 This alert displays if the XML you supply is not well-formed or does not adhere to the Modifier schema.

	There is no box with the specified identifier.	HTTP Error #500 This alert displays if the box specified by the child text node of the <ID> element does not exist.
	The value of Scale Across should be between 10% and 1000%.	HTTP Error #500 This alert displays if the value of the child text node of a <SCALEACROSS> element is invalid.
	The Value of Scale Down should be between 10% and 1000%.	HTTP Error #500 This alert displays if the value of the child text node of a <SCALEDOWN> element is invalid.
	The value of Offset Across is in invalid range.	HTTP Error #500 This alert displays if the value of the child text node of the <OFFSETACROSS> element is invalid.
	The value of Offset Down is in invalid range	HTTP Error #500 This alert displays if the value of the child text node of the <OFFSETDOWN> element is invalid.
	The value of Picture Angle must be between –360 and 360 degrees.	HTTP Error #500 This alert displays if the value of the child text node of the <ANGLE> element is invalid.
	The value of Picture Skew must be between –75 and 75 degrees.	HTTP Error #500 This alert displays if the value of the child text node of the <SKEW> element is invalid.
	The XML document contains an invalid tag value.	HTTP Error #500 This alert displays if you supply an invalid value in the XML.
	The specified box cannot be	HTTP Error #500 This alert displays if you try to modify picture properties on a

	modified. box that is not a picture box.
Logs	If the request succeeds, a transaction success message is written to the QuarkXPress Server transaction log file. For example: 8/3/2005 11:27:42 — jpeg/sample.qxp — Type: image/jpeg — Size: 31715 — Client: 127.0.0.1If an alert displays, an error message is written to the QuarkXPress Server error log file. For example:8/10/2005 10:39:07 — Error — Error Code: 10339 — The specified file failed to load in the picture box.
Example GET URL	When QuarkXPress Server is running on Windows, use a URL like the following:http://localhost:8080/sample.qxp?modify= file:C:\imageProperties.xml When QuarkXPress Server is running on macOS, use a URL like the following:http://localhost:8080/sample.qxp?modify= file:MacHD:xml:imageProperties.xml You can also supply a string that consists of valid XML commands. For example: http://localhost:8080/sample.qxp?modify= <PROJECT><LAYOUT><ID UID="1"/><SPREAD> <ID UID="1"/><BOX BOXTYPE="CT_PICT"> <ID NAME="EVEREST"/> <PICTURE SCALEACROSS="50" OFFSETDOWN="20" ANGLE="30" FIT="CENTERPICTURE" SKEW="30" FLIPHORIZONTAL="false"/></BOX></SPREAD> </LAYOUT></PROJECT>
Example 1, object model	Request object names:ModifierRequest ModifierStreamRequest Project Box Picture Layout ModifierFileRequestFor ModifierFileRequest, the member contents are used to set the file path or send the XML itself. com.quark.qxpsm.QRequestContext rc = new com.quark.qxpsm.QRequestContext(); if(!this.DocumentSettings1.documentName.Text.Equals("")) rc.documentName = this.DocumentSettings1.documentName.Text; // STEP 2(SPECIFIC TO REQUESTS):Create the Image // Modifier renderer request and embed it in ModifierRequest imgReq = new ModifierRequest(); Project contents = new Project(); Picture picture1 = new Picture(); picture1.scaleAcross = this.scaleacross1.Text; picture1.scaleDown = this.scaledown1.Text; if(this.fitpicturebox1.Checked ==

	true) picture1.fitPictureToBox = “true”; if(this.flipvertical1.Checked == true) picture1.flipVertical = “true”; if(this.fliphorizontal1.Checked == true) picture1.flipHorizontal = “true”; Box box1 = new Box(); box1.UID = txtBox1; box1.picture = picture1; Layout layout1 = new Layout(); layout1.name = layoutText; imgReq.contents = contents; contents.layouts = new Layout[]{layout1}; layout1.boxes = new Box[]{box1}; rc.request = imgReq; // Create the service and call it with QRequestContext object RequestService svc = new RequestService(); com.quark.qxpsm.QContentData qc = svc.processRequest(rc);
Example 2, object model	To edit the properties of an existing text box in a QuarkXPress project, use the following object hierarchy:ModifierRequest < Project < Layout < Spread < Box < Picture For a list of the Picture object's properties, see the JavaDoc installed with QuarkXPress Manager.
Notes	You cannot replace an image with the Modifier XTensions software.If you specify <FITPICTURETOBOX>, <FITBOXTOPICTURE>, and <FITPICTURETOBOXPRO> for a picture, only the first of these elements will be applied.

Importing data

Imports text or image data into a project. You can use import any text or picture file format supported by QuarkXPress, including XPress Tags files.

➡ You can import .doc, .docx, .dot, .dotx, and .docm files.

To import text or image data into a project, use the following parameters in the Modifier schema:

- "*BOX* “
- "*ID*“
- "*PICTURE*“ (this is not a required element when importing data)
- "*TEXT*“
- "*STORY*“
- "*CONTENT*“

The following XML shows how some of these parameters work.

<PROJECT>

```
  <ID NAME="Layout 1"/>
  <SPREAD>
   <ID UID="1"/>
   <BOX BOXTYPE="CT_PICT">
    <ID NAME="ABOUT"/>
    <PICTURE/>
    <CONTENT>C:\docs\file1.jpg</CONTENT>
   </BOX>
   <BOX BOXTYPE="CT_TEXT">
    <ID NAME="PRODUCTS"/>
    <CONTENT>file:C:\docs\file2.txt</CONTENT>
   </BOX>
   <BOX BOXTYPE="CT_TEXT">
    <ID NAME="SERVICES"/>
    <TEXT>
     <STORY FILE="file:C:\docs\file3.doc" CONVERTQUOTES="true"
      INCLUDESTYLESHEETS="true"/>
    </TEXT>
   </BOX>
  </SPREAD>
 </LAYOUT>
</PROJECT>
```

Response	A preview of a QuarkXPress project with a value in the data import XML tags applied to the text boxes.	
Alerts	File not found.	HTTP Error #404 QuarkXPress Server Error #–43 This alert displays if you specify an invalid XML file or request a document that is not available to QuarkXPress Server.
	The XML document is not valid or well formed.	HTTP Error #500 This alert displays if the XML you supply is not well-formed or does not adhere to the Modifier schema.
	There is no box with the specified identifier.	HTTP Error #500 This alert displays if the box specified by the child text node of the <ID> element does not exist.
	The specified box is not	HTTP Error #500 This alert displays if you request a box that is not a text box or a picture box.

	a picture or text box.	
	A locked layer cannot be manipulated.	HTTP Error #500 This alert displays if you request data from a box on a locked layer. What to do: Open the project in QuarkXPress, display the Layers palette, and unlock the box's layer.
	Unable to read picture (#106)	HTTP Error #500 QuarkXPress Server Error #–109 This alert displays if you try to import a text file into a picture box.
	Bad filename/ pathname	HTTP Error #404 QuarkXPress Server Error #–37 This alert displays if you try to import an invalid or nonexistent file into a box.
Logs	If the request succeeds, a transaction success message is written to the QuarkXPress Server transaction log file. For example: 8/5/2005 18:11:54 — sample.qxp — Type: image/jpeg — Size: 65982 — Client: 127.0.0.1If an alert displays, an error message is written to the QuarkXPress Server error log file. For example:8/5/2005 18:01:59 — Error — Error Code: 10343 — A locked Layer cannot be manipulated.	
Example GET URL	When QuarkXPress Server is running on Windows, use a URL like the following:http://localhost:8080/Sample.qxp?modify= file:c:\file.xml When QuarkXPress Server is running on macOS, use a URL like the following:http://localhost:8080/Sample.qxp?modify= file:HDD:file.xml You can also supply a string that consists of valid XML commands. For example: http://localhost:8080/sample.qxp?modify= <PROJECT><LAYOUT><ID UID="Layout1"/><SPREAD><ID UID="1"/> <BOXBOXTYPE="CT_TEXT"><ID NAME="TREES"/> <CONTENT>C:\docs\file1.jpg</CONTENT> </BOX></SPREAD></LAYOUT></PROJECT> When specifying a path, use URLs like the following: http://localhost:8080/Sample.qxp? textboxname@dataimport=file:c:\file.txt http://localhost:8080/Sample.qxp?	

	pictureboxname@dataimport=c:\file.jpg You can import text directly into a box from the URL string. For example: http://localhost:8080/Sample.qxp? textboxname@dataimport=Newdata When you import a file that uses style sheets, you can control how those style sheets are handled. For example: http://localhost:8080/Documentname? textboxname@dataimport=file:c:\file.doc& textboxnameincludestylesheets@dataimport=yes You can control how quotation marks are handled at import. For example: http://localhost:8080/Documentname? textboxname@dataimport=file:c:\file.doc& textboxnameconvertquotes@dataimport=yes
Example, object model	Request object names: ModifierRequest ModifierStreamRequest Project RichText Text ID Box Layout ModifierFileRequestFor ModifierFileRequest, the member contents are used to set the file path or send the XML itself. com.quark.qxpsm.QRequestContext rc = new com.quark.qxpsm.QRequestContext(); if(!this.DocumentSettings1.documentName.Text.Equals("")) rc.documentName = this.DocumentSettings1.documentName.Text; // STEP 2 (SPECIFIC TO REQUESTS):Create the data import // request and embed it in request context ModifierRequest request = new ModifierRequest(); Project requestContents = new Project(); Content boxContent1 = new Content(); Box box1 = new Box(); box1.UID = txtBox1; box1.content = boxContent1; Layout layout1 = new Layout(); layout1.name = layoutText; if(!this.content1.Text.Equals("")) { boxContent1.value = this.content1.Text; Text text1 = new Text(); text1.font = this.fontname1.Text; box1.text = text1; if(this.includestylesheets1.Checked == false) boxContent1.includeStylesheets = "false"; if(this.convertquotes1.Checked == false) boxContent1.convertQuotes = "false"; } else if (null != uplTheFile.PostedFile) { Stream theStream = uplTheFile.PostedFile.InputStream; StreamReader reader = new StreamReader(theStream); boxContent1.value = reader.ReadToEnd(); } layout1.boxes = new Box[]{box1}; requestContents.layouts = new Layout[]{layout1}; request.contents = requestContents; rc.request = request; //

	Create the service and call it with QRequestContext object RequestService svc = new RequestService(); com.quark.qxpsm.QContentData qc = svc.processRequest(rc);
Notes	BoxParam XTensions software lets you import only files in the document pool. Modifier XTensions software, however, lets you import files that are located anywhere on the server computer, at any accessible network location, or supplied as part of a multipart HTTP request.

Exporting Job Jackets files during deconstruction

While using the xml namespace to deconstruct a QuarkXPress project, you can specify the jjname parameter in the same request to output the Job Jackets file to the document pool. For example:

http://localhost:8080/xml/project.qxp?jjname=jjfilename.xml

You can then use the construct namespace to create new QuarkXPress projects that are based on that Job Jackets file's resources and layout specifications.

➡ The jjname parameter exports QuarkXPress project resources and layout specifications to a Job Ticket. Resources defined at the Job Jackets level are not exported to the Job Ticket.

Using XML deconstruct and construct

The xml namespace deconstructs a project according to the Modifier schema. The construct namespace lets you turn an XML representation of a QuarkXPress project back into a QuarkXPress project.

This means you can deconstruct a project into an XML representation, change the XML in accordance with the Modifier schemaschema, and then have the server generate an updated version of the QuarkXPress project. You can even create new QuarkXPress projects from scratch using XML.

In addition, you can use the construct namespace to:

- Create a page based on a master page
- Create a project from XML, using a Job Jackets file as the basis for the project
- Modify text font and style, including OpenType styles
- Apply style sheets and local formatting to text
- Create and populate tables
- Import pictures into picture boxes and specify picture attributes

The schema used for XML construction and deconstruction is completely Unicode-compliant, making it ideal for use in international publishing. Furthermore, the use of this schema ensures that the schema of XML output created by Constructor does not change when server preferences change. For more information, see "*Modifier schema (annotated)*."

- Some minor QuarkXPress features are not available through the Modifier schema. However, this schema represents the majority of all user-editable aspects of a QuarkXPress project.
- The deconstruct namespace/request no longer exists. If you try to use it in this version of QuarkXPress Server, an error is returned.

Deconstructing a project

The xml namespace returns an XML representation of the target project. To use this namespace, use a URL like the following:

```
http://QXPServer8:8080/xml/project1.qxp
```

When you use the xml namespace, QuarkXPress Server returns an XML file that represents the deconstructed project. This XML file adheres to the Modifier schema (see "*Modifier schema (annotated)*").

An XML file that represents a deconstructed project does not contain all of the information necessary to reconstruct the project. The definitions of the project's resources (such as style sheets, colors, and master page definitions) are stored in a Job Jackets file. For example, you can apply a style sheet to a paragraph by indicating the style sheet's name, like so:

```
<PARAGRAPH PARASTYLE="BodyText">
  <RICHTEXT>The sun has risen.</RICHTEXT>
</PARAGRAPH>
```

The above information is included in the deconstructed project's XML file. The *definition* of the "BodyText" style sheet, however, is stored in the Job Jackets file.

The URL of a deconstructed Job Jackets file is indicated by the PROJECT@JOBJACKET attribute. If you need access to new colors, style sheets, master pages, or other resources, add them to the Job Jackets file indicated by this URL.

- Projects can also refer to resources defined with the QuarkXPress Server **Document Controls** submenu (**Server/QuarkXPress Server** menu). QuarkXPress Server looks for resources first in the Job Jackets file and then in the server-defined resources.

XML

Creates an XML file from a QuarkXPress project. The XML is returned in a fixed format that adheres to the Modifier schema. You can use the returned XML to

create or modify a QuarkXPress document using the construct namespace or modify parameter.

Namespace	xml	
schema	Modifier schema	
Parameters	box	Returns XML only for the box with the given ID or name.
	boxes	Returns XML only for the boxes with the IDs or names supplied as a comma -separated list.
	XSL	Specifies the path of an XSL file for transforming the returned XML. Use the file: indicator to specify the path.
	layout	Specifies the name or number of the layout containing the box to render. The first layout is layout 1. Note that this parameter works only with the box parameter.
	relativegeometry	Tells the xml namespace to describe <GEOMETRY> elements using <RELPOSITION> rather than <POSITION>. This allows an item's position to be defined either in relation to the page or in relation to the entire spread.
	relativetopage	Use only with the relativegeometry parameter. Tells the xml namespace to describe <GEOMETRY> elements using <RELPOSITION> elements in which ORIGIN@RELATIVETO="page" (as opposed to “spread”).
	copyfitinfo	QuarkXPress Server returns copyfitting information for QuarkCopyDesk articles by default. To retrieve copyfitting information when deconstructing a QuarkXPress project, include copyfitinfo=true in the xml request. For example:http://localhost:8080/xml/sample.qxp ? copyfitinfo=true
	Refer to the Modifier schema	
Response	Sample response:<?xml version="1.0" encoding="UTF-8" standalone="no"?> <PROJECT JOBJACKET="Macintosh	

HD:QuarkXPress DocPool: default job jackets:New Job Jacket.xml" JOBTICKET="Default Job Ticket" PROJECTNAME="project1.qxp"> <LAYOUT MEDIATYPE="PRINT"> <ID NAME="Layout 1" UID="1"/> <LAYER KEEPRUNAROUND="false" LOCKED="false" SUPPRESS="false" VISIBLE="true"> <ID NAME="Default" UID="-1"/> <RGBCOLOR BLUE="231" GREEN="231" RED="231"/> </LAYER> <SPREAD> <ID UID="1"/> <PAGE MASTER="3" POSITION="RIGHTOFSPINE" FORMATTEDNAME="1"> <ID UID="1"/> </PAGE> <BOX BOXTYPE="CT_TEXT" COLOR="None" OPACITY="100%" SHADE="100%"> <ID NAME="Introduction" UID="5"/> <GEOMETRY LAYER="Default" PAGE="1" SHAPE="SH_RECT"> <POSITION> <TOP>39.064</TOP> <LEFT>39.026</LEFT> <BOTTOM>63.951</BOTTOM> <RIGHT>214.611</RIGHT> </POSITION> <SUPPRESSOUTPUT>false</SUPPRESSOUTPUT> <RUNAROUND TYPE="NONE"/> </GEOMETRY>pre <FRAME GAPCOLOR="White" GAPOPACITY="100%" GAPSHADE="100%"OPACITY="100%" SHADE="100%" STYLE="Solid" WIDTH="0 pt"/> <TEXT> <STORY> <COPYFIT FITAMOUNT="0.033"“ NUMBEROFCHARACTERS="6" NUMBEROFLINES="1" NUMBEROFWORDS="1" STATE="underFit"/> <PARAGRAPH PARASTYLE="launch"> <RICHTEXT CHARSTYLE="launch">LAUNCH</RICHTEXT> </PARAGRAPH> </STORY> </TEXT> </BOX> <BOX BOXTYPE="CT_PICT" COLOR="None" OPACITY="100%" SHADE="100%"> <ID NAME="Sunrise" UID="6"/> <PICTURE SCALEACROSS="100%" SCALEDOWN="100%"/> <CONTENT> Macintosh HD:QuarkXPress Server Documents:sunrise.tif </CONTENT> <GEOMETRY LAYER="Default" PAGE="1" SHAPE="SH_RECT"> <POSITION> <TOP>0</TOP> <LEFT>0</LEFT> <BOTTOM>800</BOTTOM> <RIGHT>600</RIGHT> </POSITION> <SUPPRESSOUTPUT>false</SUPPRESSOUTPUT> <RUNAROUND BOTTOM="0" LEFT="0" RIGHT="0" TOP="0" TYPE="ITEM"/> </GEOMETRY> <FRAME GAPCOLOR="White" GAPOPACITY="100%" GAPSHADE="100%"

	OPACITY="100%" SHADE="100%" STYLE="Solid" WIDTH="0"/> <PICTURE/> </BOX> </SPREAD> </LAYOUT> </PROJECT>
Logs	If the request succeeds, a transaction success message is written to the QuarkXPress Server transaction log file. For example: 8/3/2004 17:16:11 — xml/sample.qxp — Type: text/xml — Size: 2364 — Client: 127.0.0.1
Example GET URL	http://localhost:8080/xml/sample.qxp You can also deconstruct QuarkCopyDesk articles. To deconstruct a QuarkCopyDesk article, use the following: http://localhost:8080/xml/copydesk/abc.qcd
Example, Object Model	Request object name: XMLRequestXMLRequest xmlRequest = new XMLRequest(); QRequestContext context = new QRequestContext(); context.setDocumentName("SAMPLE_DOCUMENT.qxp"); context.setResponseAsURL(false); context.setRequest(xmlRequest); QContentData response = new RequestServiceStub().processRequest(context); System.out.println(response.getTextData());

Constructing a project

The construct namespace takes two arguments: The name of the project to be created, and a modify parameter that points to the XML file or string that describes how to create the project. For example:

http://QXPServer8:8080/construct/project1.qxp?modify=file:path to XML file on server

or:

http://QXPServer8:8080/construct/project1.qxp?modify=XML string

- There is a length limitation of 4096 characters on URLs, so you will probably want to use an XML file rather than an XML string.
- If you are using QuarkXPress Server Manager, you can send a similar command with a QuarkXPress Server Manager URL or through Web services.

Every project created with the construct namespace must be based on a Job Ticket in a Job Jackets file. Using construct to create a project is roughly equivalent to using the **File > New > Project from Ticket** command in QuarkXPress.

When you create a project using the construct namespace, you must supply the path to the Job Jackets file that will supply the project's resources. To do so, indicate the URL of the Job Jackets file in the PROJECT@JOBJACKET attribute and the name of the Job Ticket in the PROJECT@JOBTICKET attribute.

(<PROJECT> is the root element of the Modifier schema. For more information, see "*Modifier schema (annotated)*.")

For example, to create a project from a Job Ticket named "Tall US Brochure Ticket" in a Job Jackets file named "BrochureJJ.xml," use XML like the following:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<PROJECT JOBJACKET="MacintoshHD:brochures:BrochureJJ.xml"
    JOBTICKET="Tall US Brochure Ticket"
    PROJECTNAME="project1.qxp">
```

Construct

The construct namespace lets you create a QuarkXPress project using XML.

Namespace	construct		
schema	Modifier schema		
Parameters	modify	String	The string or the path of the XML file that describes how to create the project. Use the file: indicator to specify the path.
	qxpdocver	8 | 9	Indicates the QuarkXPress version format to use. For example:http://QXPServer8:8080/construct/qxpdoc/project1.qxp?qxpdocver=8
Example GET URL	http://QXPServer8:8080/construct/project1.qxp?modify=file:sample.xml		
Example XML	<?xml version="1.0" encoding="UTF-8"?> <PROJECT JOBJACKET="C:\XML\New Job Jacket 3.xml" JOBTICKET="Default Job Ticket" PROJECTNAME="project1.qxp"> <LAYOUT> <ID NAME="Layout 1"/> <SPREAD> <ID UID="1"/> <PAGE> <ID UID="1"/> </PAGE> </SPREAD>		

	</LAYOUT> </PROJECT>	
Response	A new QuarkXPress project.	
Alerts	File not found.	HTTP Error #404 QuarkXPress Server Error #–43 This alert displays if you specify an invalid XML file or request a document that is not available to QuarkXPress Server. For example, this error can occur if an image or text file file mentioned in a <CONTENT> element is invalid or missing.
	Bad filename/pathname.	HTTP Error #404 QuarkXPress Server Error #–37 This alert displays if you specify an invalid file name or path.
	The XML document is not valid or well formed.	HTTP Error #500 This alert displays if the XML you supply is not well-formed or do not adhere to the Modifier schema.
	The XML document contains an invalid tag value.	HTTP Error #500 This alert displays if you supply an invalid value in the XML.
Logs	If the request succeeds, a transaction success message is written to the QuarkXPress Server transaction log file. For example:8/3/2005 11:27:42 — jpeg/construct/table.qxp — Type: image/jpeg — Size: 31715 — Client: 127.0.0.1If an alert is displayed, an error message is written to the QuarkXPress Server error log. The following is a sample of the error log entry:8/10/2005 10:39:07 — Error — Error Code: 10339 — The specified file failed to load in the picture box.	
Example, object model	Request Object Names:XMLRequest ConstructRequest ConstructFileRequest ConstructStreamRequest To construct a new QuarkXPress project by editing an existing document, first deconstruct a QuarkXPress project using code like the following:XMLRequest dcnstrq = new XMLRequest(); rc.request = dcnstrq; Next, alter the project by manipulating the XML. When you're done, pass the modified XML document to ConstructStreamRequest to create a new QuarkXPress project. For	

	example:ConstructStreamRequest cnstrq = new ConstructStreamRequest(); cnstrq.modify = Buffer; // Byte[] for the modified XML rc.request = cnstrq; QuarkXPressRenderRequest qxprq = new QuarkXPressRenderRequest(); cnstrq.request = qxprq; Alternatively, you can deconstruct a QuarkXPress project using code like the following:RequestServiceService svc = new RequestServiceService() Project proj = svc.getDOM("document.qxp"); Next, alter the project by manipulating the XML. When you're done, pass the modified Project instance to ConstructRequest to create a new QuarkXPress project. For example:ConstructRequest cnstrq = new ConstructRequest(); cnstrq.project = proj; QRequestContext rc = new QRequestContext(); rc.request = cnstrq; QuarkXPressRenderRequest qxprq = new QuarkXPressRenderRequest(); cnstrq.request = qxprq;
Notes	The construct namespace takes two arguments: The name of the project to be created and a modify parameter with the string or the path of the XML file that describes how to create the project: http://localhost:8080/qxpdoc/construct/project1.qxp?modify=file:path to XML file on server http://localhost:8080/qxpdoc/construct/project1.qxp?modify=<xml-string>

Construct and modify

The modify parameter lets you modify existing projects. For example:

http://QXPServer8:8080/project1.qxp?
modify=file:path to XML file on server

or:

http://QXPServer8:8080/project1.qxp?modify=XML string

It's important to understand that although the construct namespace uses the same schema that you use when you modify an existing project, the construct namespace uses it differently. When you use the construct namespace, the XML you pass simply contains a description of everything in the document you want to create — much as an HTML file describes a page you want to display in a browser. There is no need to use a command and create elements such as ADDCELLS, OPERATION, and MOVERIGHT; you simply describe each item in the layout with elements such as <BOX> and <TABLE>, and specify each item's

position with the <POSITION> element type. When you use the modify attribute without the construct namespace, however, the XML you pass must contain commands that show how you want QuarkXPress Server to modify the project.

For more information, see "*Modifier schema (annotated)*."

Working with pages and spreads

The root element of a deconstructed QuarkXPress project is <PROJECT>. Within each <PROJECT> element are one or more <LAYOUT> elements. Each layout contains one or more <SPREAD> elements, and each <SPREAD> contains one or more <PAGE> elements. Each layout, spread, and page has a unique name, indicated by its <ID> element.

Each layout can have a unique name, indicated by its <ID> element's NAME attribute. You can use a layout's name when referring to that layout in a non-construct call that uses the MODIFY attribute. The ID@NAME attribute is ignored for <SPREAD> and <PAGE> elements, but you can refer to them numerically with their <ID> element's UID attribute, with "1" being the first, "2" being the second, and so forth.

➡ With most element types, it is best to assign an ID@NAME value to an element and use that to refer to the element, because ID@UID values are defined by QuarkXPress Server and thus ignored for construct calls. <PAGE> and <SPREAD> are exceptions to this rule.

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<PROJECT JOBJACKET=" MacintoshHD:brochures:BrochureJJ.xml"
      JOBTICKET="Tall US Brochure Ticket"
      PROJECTNAME="project1.qxp">
   <LAYOUT>
     <ID NAME="Layout 1" />
     <SPREAD>
       <ID UID="1" />
       <PAGE POSITION="RIGHTOFSPINE" MASTER="3">
         <ID UID="2" />
       </PAGE>
       ...
```

Each page has a POSITION attribute that indicates which side of the spine it is on. (In single-sided layouts, every page is given a POSITION of RIGHTOFSPINE).

You can assign items to a page using the GEOMETRY element, which is a child of the BOX and TABLE elements. For example:

```
<BOX BOXTYPE="CT_TEXT" COLOR="White">
     <ID NAME="Title Box" />
   <GEOMETRY LAYER="Default"
PAGE="1"
 SHAPE="SH_RECT">
```

```
      <POSITION>
        <TOP>90</TOP>
        <LEFT>95</LEFT>
        <BOTTOM>190</BOTTOM>
        <RIGHT>195</RIGHT>
      </POSITION>
    </GEOMETRY>
  </BOX>
```

Master pages are stored in a deconstructed project's Job Jackets file. To create a page from this master page, insert a MASTER attribute into the PAGE element and indicate the number of the target master page. Master page numbering is as follows:

1 = blank single page

2 = blank facing-page

3 = the first user-defined master page in the Job Jackets file (by default, the master page named "A-Master A")

For example, to create a master page based on the first user-defined master page in the Job Jackets file, you could use XML like the following:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<PROJECT JOBJACKET=" file://brochures/BrochureJJ.xml"
      JOBTICKET="Tall US Brochure Ticket"
      PROJECTNAME="project1.qxp">
  <LAYOUT>
    <ID NAME="Layout 1"/>
    <SPREAD>
      <ID UID="1" />
      <PAGE MASTER="3" POSITION="LEFTOFSPINE">
        <ID UID="2" />
      </PAGE>
      ...
```

Note that each page has a POSITION attribute that indicates where that page falls with regard to the spine.

Working with layouts

QuarkXPress Server lets you create layouts from scratch in several ways:

- You can create a layout using the default layout properties, as specified in the server Job Jackets file.
- You can create a layout using a layout specification in a template's Job Jackets structure.
- You can create a layout using a specific height and width.

To create a layout using the server Job Jackets file's default settings, use XML like the following:

```
<PROJECT>
 <LAYOUT OPERATION="CREATE">
  <ID NAME="New Layout"/>
  <SPREAD>
   <ID UID="1"/>
   <BOX>
    <ID NAME="Box5"/>
    <TEXT>
     <STORY>
      <PARAGRAPH PARASTYLE="Normal">
       <RICHTEXT>Scrollable Layout</RICHTEXT>
      </PARAGRAPH>
     </STORY>
    </TEXT>
   </BOX>
  </SPREAD>
 </LAYOUT>
</PROJECT>
```

To create a layout using a layout specification in the template's Job Jackets structure, use XML like the following:

```
<PROJECT>
 <LAYOUT OPERATION="CREATE"
LAYOUTSPECIFICATION="NewLayoutSpec"
>
  <ID NAME="New Layout"/>
  <SPREAD>
   <ID UID="1"/>
   <BOX>
    <ID NAME="Box5"/>
    <TEXT>
     <STORY>
      <PARAGRAPH PARASTYLE="Normal">
       <RICHTEXT>Scrollable Layout</RICHTEXT>
      </PARAGRAPH>
     </STORY>
    </TEXT>
   </BOX>
  </SPREAD>
 </LAYOUT>
</PROJECT>
```

To create a layout using a specific height and width, use XML like the following:

```
<PROJECT>
 <LAYOUT OPERATION="CREATE"
```

```
HEIGHT="900" WIDTH="500"
>
   <ID NAME="New Layout"/>
   <SPREAD>
    <ID UID="1"/>
    <BOX>
     <ID NAME="Box5"/>
     <TEXT>
      <STORY>
       <PARAGRAPH PARASTYLE="Normal">
        <RICHTEXT>Scrollable Layout</RICHTEXT>
       </PARAGRAPH>
      </STORY>
     </TEXT>
    </BOX>
   </SPREAD>
  </LAYOUT>
</PROJECT>
```

To create a layout and flow content into it in one go, without a spread/page context, use XML like the following:

```
<PROJECT>
 <LAYOUT OPERATION="CREATE" HEIGHT="900" WIDTH="500">
 <ID NAME="New Layout"/>

  <!--Create a TOC-->
  <BOX>
   <ID NAME="Flow"></ID>
   <TEXT>
    <STORY>
     <LIST LISTSTYLE="TOC" OPERATION="CREATE">
          ...
       </LIST>

     <!-- Insert a page break -->
     <PAGEBREAK></PAGEBREAK>

    </STORY>
   </TEXT>
  </BOX>

  <!-- Add a heading -->
  <INLINEBOX>
   <RICHTEXT>This is the Heading</RICHTEXT>
  </INLINEBOX>

  <!-- Add text -->
  <INLINEBOX>
```

```
 <PARAGRAPH INDENTLEVEL="2" PARASTYLE="MyStyle">
  <RICHTEXT>This is the text</RICHTEXT>
 </PARAGRAPH>
</INLINEBOX>

<!-- Import a picture -->
<INLINEBOX>
 <CONTENT>file:MyFile.jpg</CONTENT>
</INLINEBOX>

<!-- Add a  table -->
<INLINETABLE>
 <TBODY>
  <TROW>
   <ENTRY>Column1</ENTRY>
   <ENTRY>Column2</ENTRY>
  </TROW>
 </TBODY>
</INLINETABLE>

</LAYOUT>
</PROJECT>
```

When QuarkXPress Server creates a new layout this way, it adds an automatic text box. If you want to flow content into this automatic text box, address it by any name you like, and QuarkXPress Server will assign that name to the automatic text box on the first page.

Working with layers

To create a layer in XML, use the LAYER element. For example:

```
<LAYER KEEPRUNAROUND="true" LOCKED="false"
    SUPPRESS="false" VISIBLE="true">
  <ID NAME="Layer 1" />
</LAYER>
```

The RGBCOLOR element defines the layer's color as displayed in the **Layers** palette.

You can assign items to a layer using the GEOMETRY element, which is a child of the BOX and TABLE elements. For example:

```
BOX BOXTYPE="CT_TEXT" COLOR="White">
    <ID NAME="Main Layer" />
  <GEOMETRY
LAYER="Default"
 PAGE="1" SHAPE="SH_RECT">
    <POSITION>
      <TOP>90</TOP>
```

```
      <LEFT>95</LEFT>
      <BOTTOM>190</BOTTOM>
      <RIGHT>195</RIGHT>
    </POSITION>
  </GEOMETRY>
</BOX>
```

Working with boxes

To add text and pictures to a project, you must add text boxes and picture boxes to the project's <SPREAD> element. Both are represented by <BOX> elements, but text boxes have a BOXTYPE attribute of CT_TEXT, and picture boxes have a BOXTYPE attribute of CT_PICT. You can read about how <BOX> elements are put together in the Modifier schema, but for purposes of illustration, the sample XML below describes a spread that contains a text box and a picture box.

```
<SPREAD>
  <ID UID="1" />

  <!-- TEXT BOX -->
  <BOX BOXTYPE="CT_TEXT" COLOR="White">
    <ID NAME="Headline Box" />
    <GEOMETRY LAYER="Default" PAGE="1" SHAPE="SH_RECT">
      <POSITION>
        <TOP>200</TOP>
        <LEFT>80</LEFT>
        <BOTTOM>450</BOTTOM>
        <RIGHT>475</RIGHT>
      </POSITION>
    </GEOMETRY>
    <TEXT>
      <STORY>
        <PARAGRAPH PARASTYLE="Normal">
          <RICHTEXT>This is text in a box.</RICHTEXT>
        </PARAGRAPH>
      </STORY>
    </TEXT>
  </BOX>

  <!-- PICTURE BOX -->
  <BOX BOXTYPE="CT_PICT">
    <ID NAME="Main Story Photo" />
    <GEOMETRY LAYER="Default" PAGE="1" SHAPE="SH_RECT">
      <POSITION>
        <TOP>90</TOP>
        <LEFT>95</LEFT>
        <BOTTOM>190</BOTTOM>
        <RIGHT>195</RIGHT>
```

```
      </POSITION>
    </GEOMETRY>
    <PICTURE ANGLE="0°" FLIPHORIZONTAL="false"
      FLIPVERTICAL="false" FULLRES="false" MASK="None"
      OFFSETACROSS="0 OFFSETDOWN="0" OPACITY="100%"
      SCALEACROSS="100%" SCALEDOWN="100%" SHADE="100%"
      SKEW="0°" SUPRESSPICT="false"/>
    <CONTENT>Macintosh HD:DocPool:flower1.jpg</CONTENT>
  </BOX>

</SPREAD>
```

This example will work for a construct request. For a modify request, add the attribute value OPERATION="CREATE" in the BOX element.

All BOX elements can contain a GEOMETRY element that indicates the position and size of the box, a BORDER element that describes the four borders of a box, a FRAME element that describes the box's frame (if any), and a SHADOW element that describes the box's drop shadow. Additional BOX elements are described in the following sections.

➡ The z-order (stacking order) of boxes in the layout is determined by the order of the <BOX> elements in the XML, from rearmost to frontmost.

Fitting a box to text or a picture

The <FIT> element type lets you automatically adjust the size of a box to fit the text or picture in that box.

The default behaivior is to not fix a box to its content. To use this feature, you must supply <MAX> and <MIN> elements. Each <MAX> or <MIN> element lets you specify a maximum or minimum size for the box, a maximum or minimum location for the resized box, or a maximum or minimum scale percentage for the box. Note that you can use different types of <MAX> and <MIN> elements in a <FIT> element, but you can use only one <MAX> element and one <MIN> element per <FIT> element.

The FIT@POINT attribute lets you indicate the direction in which the box should grow or shrink. The available options are TOPLEFT, BOTTOMLEFT, TOPRIGHT, and BOTTOMRIGHT.

The FIT@AVOIDBOXESBY attribute lets you specify the distance between the POINT side or corner of a resized box and any other items around it. A box will expand only until it is this distance from an adjacent item.

The FIT@PROPORTIONAL attribute lets you specify whether the resized box should have the same aspect ratio as the original box.

For example:

```
<BOX>
```

```
  <ID UID="5"/>
  <GEOMETRY>
    <POSITION>
      <TOP>224.001</TOP>
      <LEFT>110.003</LEFT>
      <BOTTOM>381</BOTTOM>
      <RIGHT>253.253</RIGHT>
    </POSITION>

    <FIT POINT="BOTTOMLEFT" PROPORTIONAL="true">
<MAX>          <LOCATION X="320" Y="560"/>          </MAX>
<MIN>          <SIZE HEIGHT="100" WIDTH="10"/>          </MIN>
</FIT>

  </GEOMETRY>
<BOX/>
```

- To use this feature, you must have FitBoxToContent XTensions software loaded.
- For pictures, <FIT> is equivalent to PICTURE@FIT="FITBOXTOPICTURE". <MAX> and <MIN> have no effect.

Using inline boxes

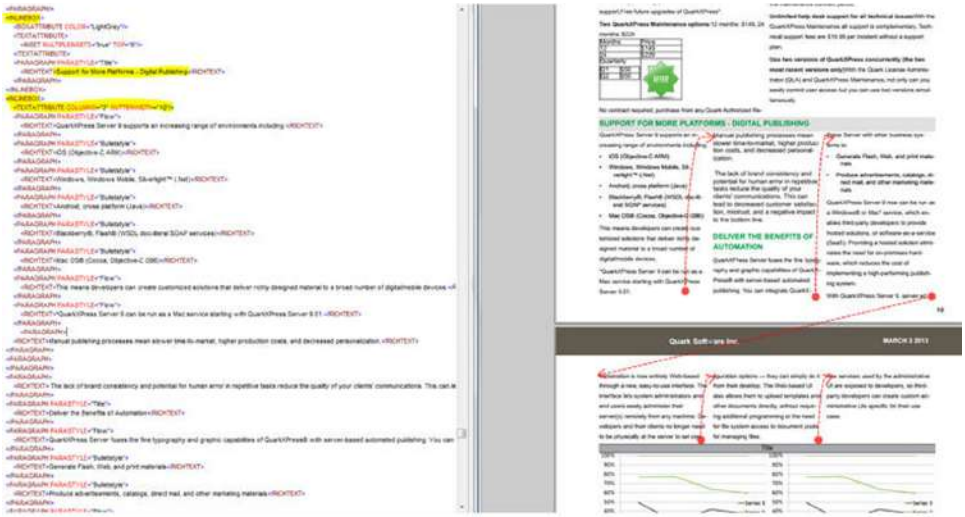
The Inline Boxes feature makes it easy to create an anchored box. Rather than having to describe every aspect of a box and then reference that box from an <ANCHOREDBOXREF> or <CALLOUTANCHOR> element, you can simply specify the content for a box inline, like so:

```
<PROJECT>
 <LAYOUT>
  <ID UID="123"/>
  <BOX COLOR="Cyan" OPACITY="50">
   <ID UID="456"/>
   <TEXT>
    <STORY>
     <RICHTEXT>Test before an anchored text box.</RICHTEXT>
     <INLINEBOX>
      <TEXTATTRIBUTE COLUMNS="2" GUTTERWIDTH="10">
       <INSET ALLEDGES="5"/>
      </TEXTATTRIBUTE>
      <CONTENT>file:example.docx</CONTENT>
     </INLINEBOX>
     <RICHTEXT>Test between anchored text boxes.</RICHTEXT>
     <INLINEBOX>
      <CONTENT>
       <PARAGRAPH PARASTYLE="Normal">
        <RICHTEXT>Text in the second anchored box.</RICHTEXT>
```

```
            </PARAGRAPH>
            <RICHTEXT>More text in the second anchored
box.</RICHTEXT>
          </CONTENT>
        </INLINEBOX>
        <RICHTEXT>Test between anchored boxes.</RICHTEXT>
        <INLINEBOX WIDTH="50" SCALEUP="false">
          <SHADOW ANGLE="166" BLUR="6" COLOR="Yellow"
DISTANCE="6"
            INHERITOPACITY="true" MULTIPLYSHADOW="true"
OPACITY="40%"
            SCALE="100% "SHADE="62%" SKEW="0"/>
          <CONTENT>file:example.jpg</CONTENT>
        </INLINEBOX>
        <RICHTEXT>Text after an anchored picture box.</RICHTEXT>
      </STORY>
    </TEXT>
  </BOX>
 </LAYOUT>
</PROJECT>
```

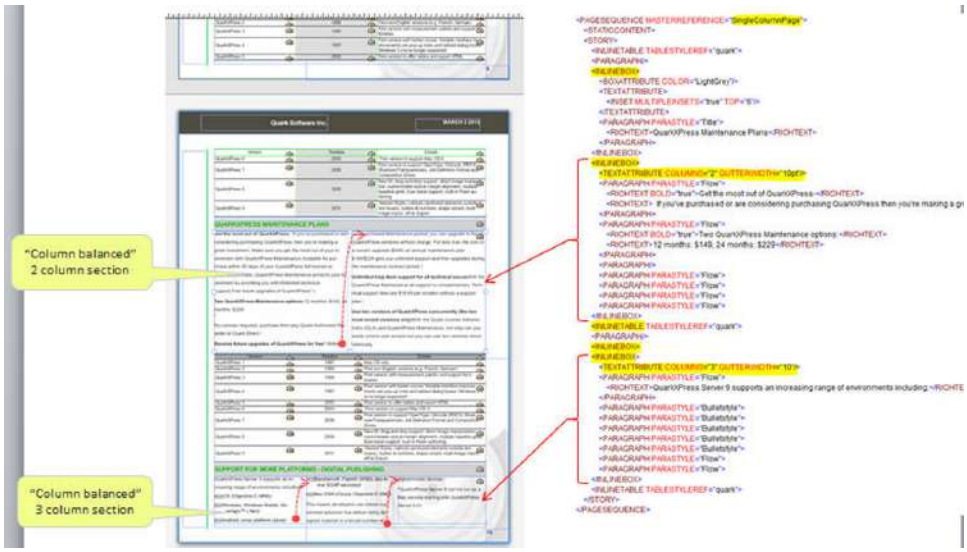
Because an <INLINEBOX> element's content comes in the form of a <CONTENT> element, you can fill such an anchored box with Modifier-formatted text, with text from a text file, or with a picture from a picture file.

If the volume of text (represented by PARAGRAPH) is potentially large, the INLINEBOX automatically continues to flow the remaining text across pages.



Automatic text flow example

The INLINEBOX element allows to have autommatic column balanced sections. This allows you to have the text flow and bottom-align across multiple columns on a page automatically.



Automatic column balancing

The INLINEBOX@WIDTH attribute lets you specify the width of the anchored box as a percentage of its parent column or box. If you fill an anchored box with so much text that it expands to the height of its parent box, the text is adjusted to fit in the box.

The INLINEBOX@SCALEUP attribute lets you control sizing for picture boxes.

The childSHADOW element lets you specify drop shadow effects.

A child BOXATTRIBUTE element lets you control the formatting of the boxes. If you're creating an inline text box, a child TEXTATTRIBUTE element lets you control the formatting of the boxes.

A child Gradient element lets you apply a gradient color to a box or a box frame, as shown below.

```
<INLINEBOX>
 <BOXATTRIBUTE>
  <GRADIENT ANGLE="90" TYPE="AXIAL">
   <STOP COLOR="Cyan" CONTROLPOINT="27.41%"
Location="25.00%"
     OPACITY="100%" SHADE="100%">
   <STOP COLOR="White" CONTROLPOINT="63.70%"
Location="25.00%"
     OPACITY="100%" SHADE="100%">
   <STOP COLOR="Magenta" CONTROLPOINT="28.89%"
Location="50.00%"
     OPACITY="50%" SHADE="100%">
   <STOP COLOR="White" CONTROLPOINT="68.15%"
Location="75.00%"
     OPACITY="100%" SHADE="100%">
   <STOP COLOR="Cyan" CONTROLPOINT="50.00%"
```

```
Location="100.00%"
      OPACITY="100%" SHADE="100%">
    <STORY BOXNAME="Flow">
   </GRADIENT>
  </BOXATTRIBUTE>
</INLINEBOX>
```

A child INTERACTIVITY element facilitates the application of interactivity on boxes. This allows for the deconstruct and modifiability of App Studio Interactivities, as shown below:

```
<PROJECT>
 <LAYOUT>
  <ID NAME="Layout 1"/>
  <PAGESEQUENCE MASTERREFERENCE="A-Master-A">
   <STORY BOXNAME="Flow">
    <INLINEBOX>
     <!--Preview image-->
     <CONTENT>file:Images/2pdf.PNG</CONTENT>
     <INTERACTIVITY AUTHORXTID="1131430225" NAME="Button
1"
                OWNERXTID="1129333841" TYPE="Button">
      <Settings InitiallyHidden="False">
       <settings>
        <actions>
         <action>
          <type>openfile</type>
          <name/>
          <sourcesettings>
           <sourcetype>1</sourcetype>
           <sourcepath>PDF/EconomicResearch.pdf</sourcepath>
          </sourcesettings>
         </action>
        <actions>
       </settings>
      </Settings>
      <DATAPROVIDER DATAPROVIDERXTID="1131430225"/>
     </INTERACTIVITY>
    </INLINEBOX>
   </STORY>
  </PAGESEQUENCE>
 </LAYOUT>
</PROJECT>
```

This applies button interactivity and associates an Open File action. The generated App Studio issue, when previewed on a device, would display the “EconomicResearch.pdf”. The Open File action can open media files as well.

The child BOXATTRIBUTE lets you specify an angle of rotation of the box through the use of the <ANGLE> element, like so:

```
<INLINEBOX>
 <BOXATTRIBUTE ANGLE="30"/>
  <CONTENT>file:tmp.bmp</CONTENT>
 ...
```

This would rotate the box by 30 degrees.

For more information, see "*INLINEBOX (Modifier schema)*," *BOXATTRIBUTE (Modifier schemaschema)*," and "*TEXTATTRIBUTE (Modifier schema)*."

- If you deconstruct an anchored box that was created with an <INLINEBOX> element, the resulting XML describes the box as a <BOX> element, not an <INLINEBOX> element.

Working with groups

To add boxes to a group, create a <GROUP> element and then insert <BOXREF> elements that refer to the boxes you want in the group. For example, the group described below includes the two boxes described above it:

```
 <BOX BOXTYPE="CT_TEXT" COLOR="White">
   <ID NAME="MainStoryText" UID="217"/>
</BOX>

<BOX BOXTYPE="CT_PICT">
   <ID NAME="MainStoryPhoto" UID="218"/>
</BOX>

<GROUP>
   <ID NAME="MainStoryGroup" UID="300"/>
   <BOXREF NAME="MainStoryText" UID="217"/>
   <BOXREF NAME="MainStoryPhoto" UID="218"/>
</GROUP>
```

You can nest one group within another by adding a <BOXREF> that refers to the child group, like so:

```
<GROUP>
   <ID NAME="MainStoryGroup" UID="300"/>
   <BOXREF NAME="MainStoryText" UID="217"/>
   <BOXREF NAME="MainStoryPhoto" UID="218"/>
</GROUP>

<BOX BOXTYPE="CT_PICT">
    <ID NAME="Masthead" UID="001"/>
</BOX>

<GROUP>
   <ID NAME="MainStoryPage" UID="218"/>
   <BOXREF NAME="Masthead" UID="001"/>
```

```
<BOXREF NAME="MainStoryGroup" UID="300"/>

</GROUP>
```

To anchor a group in a text box, use XML like the following. Note that you must set BOX@ANCHOREDGROUPMEMBER="true" for all boxes in the group, and set GROUP@ANCHOREDIN for the anchored group.

```
<BOX BOXTYPE="CT_TEXT" COLOR="White"
ANCHOREDGROUPMEMBER="true"
 >
   <ID NAME="MainStoryText" UID="217"/>
</BOX>

<BOX BOXTYPE="CT_PICT"
ANCHOREDGROUPMEMBER="true"
>
   <ID NAME="MainStoryPhoto" UID="218"/>
</BOX>

<GROUP
ANCHOREDIN="MainStoryText"
>
   <ID NAME="MainStoryGroup" UID="300"/>
   <BOXREF NAME="MainStoryText" UID="217"/>
   <BOXREF NAME="MainStoryPhoto" UID="218"/>
</GROUP>

<BOX BOXTYPE="CT_TEXT" COLOR="White">
   <ID NAME="MainStoryText" UID="217"/>
   <TEXT>
      <STORY>
         <PARAGRAPH>

<ANCHOREDBOXREF OFFSET="0">MainStoryGroup
</ANCHOREDBOXREF>

         </PARAGRAPH>
      </STORY>
   </TEXT>
</BOX>
```

➡ The order of the <BOXREF> elements in a <GROUP> indicates the order in which the boxes were selected prior to grouping. The z-order of boxes in the layout is determined by the order of the <BOX> elements in the XML, from rearmost to frontmost.

Working with pictures

The <PICTURE> element supports a variety of features, including the ability to specify runaround, opacity, and drop shadow characteristics. For more information, see the Modifier schema.

```
<PROJECT>
 <LAYOUT>
  <ID NAME="Layout 1"/>
  <SPREAD>
   <ID UID="1"/>
   <BOX COLOR="Magenta" SHADE="50%" OPACITY="100%">
    <ID NAME="pict1"/>
    <PICTURE MASK="Test Alpha1"/>
    <FRAME STYLE="Triple" WIDTH ="5" COLOR="Cyan"
SHADE="100%"
     OPACITY="100%" GAPCOLOR="Yellow"
     GAPSHADE="80%" GAPOPACITY="100%"/>
    <CONTENT UID="0">image.jpg</CONTENT>
   </BOX>
   <BOX>
    <ID NAME="pict2"/>
    <PICTURE SUPRESSPICT="true" FULLRES="true"
PICCOLOR="Cyan"
     SHADE="90" OPACITY="90"/>
    <SHADOW COLOR="Cyan" SHADE="90" ANGLE="130"
OPACITY="100"
     DISTANCE="5" SKEW="10"
      SCALE="90" BLUR="3"/>
    <CONTENT UID="0">image.jpg</CONTENT>
   </BOX>
   <BOX>
    <ID NAME="pict3"/>
    <GEOMETRY>
     <RUNAROUND TYPE="NONWHITEAREAS"  OUTSET="10"
NOISE="5"
      SMOOTHNESS="5"
       THRESHOLD="10" INVERT="true" OUTSIDEONLY="true"
       RESTRICTTOBOX="true"/>
    </GEOMETRY>
    </BOX>
    <BOX>
     <ID NAME="pict4"/>
     <PICTURE FIT="FITPICTURETOBOX" SCALEACROSS="40"
      SCALEDOWN="50" FLIPVERTICAL="true"
      FLIPHORIZONTAL="false" ANGLE="40" SKEW="20"/>
    <CONTENT UID="0">image.jpg</CONTENT>
   </BOX>
```

```
  </SPREAD>
 </LAYOUT>
</PROJECT>
```

If you know the UID attribute of a box, you can insert a picture into that box without having to specify where the <BOX> element is. For example:

```
<PROJECT>
 <CONTENT UID="0">ChangedPict.jpg</CONTENT>
</PROJECT>
```

➡ Content IDs are unique across layouts.

Working with text

Every <BOX> element for text contains a <TEXT> element, and every <TEXT> element contains a <STORY> element. A <STORY> element can contain <PARAGRAPH> elements, each of which contains <RICHTEXT> elements. A <STORY> element can also simply contain <RICHTEXT> elements.

A text <BOX> element can also contain a <CONTENT> element that indicates the origin of the text in that box.

A text <BOX> element in a deconstructed project can also contain <PLACEHOLDER> elements, which allow XML Import XTensions software to insert text from a different XML source.

➡ <PLACEHOLDER> elements are ignored by the construct namespace and the modify parameter; placeholders must be inserted in QuarkXPress using XML Import XTensions software.

If you know the UID attribute of a box or story, you can insert text into that box or story without having to specify where the <BOX> or <STORY> element is. For example:

```
<PROJECT>
 <BOX>
  <ID UID="4"/>
  <STORY>
   <RICHTEXT MERGE="false" FONT="20">New text</RICHTEXT>
  </STORY>
 </BOX>
</PROJECT>
<PROJECT>
 <STORY UID="0">
  <RICHTEXT MERGE="false" FONT="20">New text</RICHTEXT>
 </STORY>
</PROJECT>
```

➡ Story IDs are unique across layouts.

Applying style sheets

Like other resources, style sheets are defined in a deconstructed project's Job Jackets file. To apply a paragraph style sheet to text, use the PARASTYLE attribute of the <PARAGRAPH> element. For example, to apply the paragraph style sheet named "BodyText" to a paragraph, use XML like the following:

```
<PARAGRAPH PARASTYLE="BodyText">
  <RICHTEXT MERGE="true">The sun has risen.</RICHTEXT>
</PARAGRAPH>
```

To apply a character style sheet to text, use the CHARSTYLE attribute of the <RICHTEXT> element. For example, to apply the character style sheet named "Emphasis" to a word, use XML like the following:

```
<PARAGRAPH PARASTYLE="BodyText">
  <RICHTEXT>The </RICHTEXT>
  <RICHTEXT CHARSTYLE="Emphasis">sun</RICHTEXT>
  <RICHTEXT> has risen.</RICHTEXT>
</PARAGRAPH>
```

Applying local formatting

To apply local formatting to text, use the attributes of the <RICHTEXT> element. For example:

```
<PARAGRAPH>
  <RICHTEXT
  SIZE="10" COLOR="Magenta" BOLD="true" OPACITY="50%"
  >The sun has risen.</RICHTEXT>
</PARAGRAPH>
```

To apply paragraph formatting, use a <FORMAT> element. For example:

```
<PARAGRAPH>
  <FORMAT SPACEBEFORE="6" SPACEAFTER="2" LEADING="24"
      ALIGNMENT="LEFT" KEEPWITHNEXT="true">
    <RICHTEXT>The sun has risen.<RICHTEXT>
  </FORMAT>
</PARAGRAPH>
```

The MERGE attribute lets you control whether formatting from one <RICHTEXT> or <PARAGRAPH> element is carried forward to the next. For example, the following XML would result in "has risen" being italicized:

```
<PARAGRAPH PARASTYLE="BodyText">
  <RICHTEXT SIZE="10">The </RICHTEXT>
  <RICHTEXT SIZE="12"ITALIC="TRUE">sun</RICHTEXT>
  <RICHTEXT MERGE="true" SIZE="10"> has risen.</RICHTEXT>
</PARAGRAPH>
```

However, this XML would result in "has risen" being plain:

```
<PARAGRAPH PARASTYLE="BodyText">
```

```
    <RICHTEXT SIZE="10">The </RICHTEXT>
    <RICHTEXT SIZE="12" ITALIC="TRUE">sun</RICHTEXT>
    <RICHTEXT MERGE="false" SIZE="10"> has risen.</RICHTEXT>
</PARAGRAPH>
```

The default value for <MERGE> is “false."

To combine local formatting with style sheets, simply add attributes to the <RICHTEXT> elements within a <PARAGRAPH> element. For example:

```
<PARAGRAPH PARASTYLE="BodyText">
    <RICHTEXT COLOR="Red">The </RICHTEXT>
    <RICHTEXT COLOR="Yellow"
CHARSTYLE="Emphasis">sun</RICHTEXT>
    <RICHTEXT COLOR="Red"> has risen.</RICHTEXT>
</PARAGRAPH>
```

Applying paragraph/text shading

➡ To learn more about shading (paragraph, text) functionality, refer to the “Working with text shading" section of the *QuarkXPress 2017 User Guide* found here: *https://files.quark.com/download/documentation/QuarkXPress/2017/English/Q XP-2017-User-Guide-EN.pdf*. The newly introduced Modifier XML markup maps/corresponds to the QuarkXPress 2017 functionality.

There are several ways to apply paragraph/text shading in QuarkXPress Server:

- Referencing a pre-defined text shading style in a paragraph/character style sheet. For example, the “BodyText” paragraph style references the pre-defined “BodyTextVariations" text shading style:

```
<PARAGRAPH PARASTYLE="BodyText">
...
</PARAGRAPH>
```

- Specify a pre-defined text shading style using the SHADINGSTYLENAME attribute of the <RICHTEXT> element.

```
<PARAGRAPH>
  <RICHTEXT
SHADINGSTYLENAME="BodyTextVariations"</RICHTEXT>
</PARAGRAPH>
```

- Specify a pre-defined text shading style using the NAME attribute of the <SHADINGSTYLE> element, within the PARAGRAPH/FORMAT nodes.

```
<PARAGRAPH PARASTYLE="BodyText">
  <FORMAT>
    <SHADINGSTYLE NAME="BodyTextVariations"/>
  </FORMAT>
  <RICHTEXT>BODY TEXT AND BODY TEXT
VARIATIONS</RICHTEXT>
```

```
</PARAGRAPH>
```

- Specify attributes in the <SHADINGSTYLE> element within the PARAGRAPH/FORMAT element for adhoc styling.

```
<PARAGRAPH PARASTYLE="BodyText">
 <FORMAT>
  <SHADINGSTYLE COLOR="Yellow"
        Shade="30%"
        OPACITY="20%"
        LENGTH="COLUMN"
        CLIPTOBOX="true"
        RIGHTPADDING="4pt"
        BOTTOMPADDING="2pt"
        LEFTPADDING="2pt"
        TOPPADDING="1pt"
 </FORMAT>
 <RICHTEXT>BODY TEXT AND BODY TEXT
VARIATIONS</RICHTEXT>
</PARAGRAPH>
```

Formatting across paragraph boundaries

You can use two methods to describe a run of formatting that crosses a paragraph boundary. The first is to simply close the first <PARAGRAPH> element and then open a new one. For example:

```
<PARAGRAPH>
   <RICHTEXT SIZE="10">The sun has risen.</RICHTEXT>
</PARAGRAPH>
<PARAGRAPH>
   <RICHTEXT SIZE="10">The sun has set.</RICHTEXT>
</PARAGRAPH>
```

The second is to use a

entity to create the paragraph break. For example:

```
<PARAGRAPH>
   <RICHTEXT SIZE="10"
   >The sun has risen.
The sun has set.</RICHTEXT>
</PARAGRAPH>
```

Retrieving copyfitting information

In deconstructed projects, a <BOX> element can contain a <LINKEDBOX> element. The <LINKEDBOX> element indicates the point where text has overflowed the current box and identifies the box where the text continues. The <LINKEDBOX> element also contains attributes that indicate where in the text the break occurs.

In a <STORY> element, the <OVERMATTER> element indicates where the current box overflows when there is no subsequent box for text to flow into. A <STORY> element also contains a <COPYFIT> element indicating how many words, characters, and lines should be allowed to fit in that box and whether the text currently fits in the box, is too short, or is too long. This information can be useful for on-the-fly copyfitting.

- The elements described in this section occur only in deconstructed project XML generated by the xml namespace. Do not use these elements when using the construct namespace.
- QuarkXPress Server returns copyfitting information for QuarkCopyDesk articles by default. To retrieve copyfitting information when deconstructing a QuarkXPress project, include copyfitinfo=true in the xml request.

Working with tables

To construct tables in XML, use a structure like the following:

```
<TABLE COLUMNS="2" ROWS="2">
 <ID NAME="MyTable"/>
 <GEOMETRY PAGE="1">
  <POSITION>
   <TOP>100</TOP>
   <LEFT>100</LEFT>
   <BOTTOM>600</BOTTOM>
   <RIGHT>400</RIGHT>
  </POSITION>
 </GEOMETRY>
   <COLSPEC>
    <COLUMN AUTOFIT="false" COLUMNCOUNT="1"
COLUMNWIDTH="134.667">
     <GRIDLINE COLOR="Black" GAPCOLOR="none" OPACITY="100%"
      SHADE="100%" STYLE="Solid" TYPE="LEFT" WIDTH="1"/>
     <GRIDLINE COLOR="Black" GAPCOLOR="none" OPACITY="100%"
      SHADE="100%" STYLE="Solid" TYPE="RIGHT" WIDTH="1"/>
    </COLUMN>
    <COLUMN AUTOFIT="false" COLUMNCOUNT="2"
COLUMNWIDTH="134.667">
     <GRIDLINE COLOR="Black" GAPCOLOR="none" OPACITY="100%"
      SHADE="100%" STYLE="Solid" WIDTH="1"/>
    </COLUMN>
    <COLUMN AUTOFIT="false" COLUMNCOUNT="3"
COLUMNWIDTH="134.667">
     <GRIDLINE COLOR="Black" GAPCOLOR="none" OPACITY="100%"
      SHADE="100%" STYLE="Solid" WIDTH="1"/>
     </COLUMN>
    </COLSPEC>
```

```
    <ROW ROWCOUNT="1">
    <CELL COLUMNCOUNT ="1">
     ...
    </CELL>
    <CELL COLUMNCOUNT ="2">
     ...
    </CELL>
   </ROW>
  </TABLE>
```

Note that the position of each row and column within the table is indicated by the ROWCOUNT and COLUMNCOUNT attributes, respectively. <CELL> elements can describe text cells or picture cells; see the following sections for details.

To specify horizontal and vertical lines in a table, use XML like the following:

```
<TABLE>
 <GRID TYPE="ALLGRID">
  <LINE COLOR="Black" GAPCOLOR="none"
   OPACITY="100%" SHADE="100%"
   STYLE="Solid" WIDTH="0"/>
 </GRID>
 ...
</TABLE>
```

Creating tables

To create a new table, use the following parameters in the Modifier schema:

- "*SPREAD*"
- "*TABLE*"
- "*COLSPEC*"
- "*COLUMN*"
- "*ROW*"
- "*CELL*"

The following XML shows how some of these parameters work.

```
<PROJECT>
 <LAYOUT>
  <ID UID="Layout 1"/>
  <SPREAD>
   <ID UID="1"/>
   <TABLE OPERATION="CREATE" ROWS="5" COLUMNS="3">
    <ID NAME="STATS"/>
    <GEOMETRY PAGE="1"/>
     <POSITION>
      <TOP>5</TOP>
```

```
        <LEFT>5</LEFT>
        <BOTTOM>30</BOTTOM>
        <RIGHT>30</RIGHT>
       </POSITION>
      </GEOMETRY>
      <FRAME WIDTH="1" COLOR="Gray"/>
     </TABLE>
    </SPREAD>
   </LAYOUT>
  </PROJECT>
```

➡ Rather than creating tables manually, you can use the Inline Tables feature, which is much easier to use. For more information see "*Using inline tables*."

Response	A preview of the QuarkXPress project with new table created in the specified position.
Logs	If the request succeeds, a transaction success message is written to the QuarkXPress Server transaction log file. For example: 4/10/2007 17:54:37 — tab.qxp — Type: image/jpeg — Size: 9049 — Client: 127.0.0.1
Example GET URL	When QuarkXPress Server is running on Windows, use a URL like the following:http://localhost:8080/sample.qxp?modify= file:C:\createTable.xml When QuarkXPress Server is running on macOS, use a URL like the following:http://localhost:8080/sample.qxp?modify= file:MacHD:xml:createTable.xml You can also supply a string that consists of valid XML commands. For example: http://localhost:8080/sample.qxp?modify= <LAYOUT><ID UID="Layout1"/><SPREAD><ID UID="1"/> <TABLE OPERATION="CREATE" ROWS="5" COLUMNS="3"> <ID NAME="STATS"/><GEOMETRY PAGE="1"/><POSITION> <TOP>5</TOP><LEFT>5</LEFT><BOTTOM>30</BOTTOM> <RIGHT>30</RIGHT></POSITION></GEOMETRY> </TABLE>SPREAD></LAYOUT></PROJECT>
Example, object model	To add a new table to an existing spread, use code like the following:Spread spread = new Spread(); Table table = new Table(); table.name = "textbox1"; Geometry geometry = new Geometry(); Position position = new Position(); position.top = "110"; position.left = "89"; position.bottom = "220"; position.right = "300"; geometry.position = position; geometry.shape = "SH_RECT"; geometry.page = "1"; geometry.layer = "Default"; table.geometry = geometry; table.rows = "2"; table.columns = "4";

table.maintainGeometry = “true”; table.operation = “CREATE”; spread.tables = new Table []{table};Use the following object hierarchy:ModifierRequest < Project < Layout < Spread < TableTo delete a table, provide the table's name or ID and set the operation attribute to “DELETE”.

Adding text and picture cells to tables

To add a text cell, use XML like the following:

```
<CELL BOXTYPE="CT_TEXT" COLUMNCOUNT ="1">
  <TEXT>
    <STORY>
      <RICHTEXT>Text goes here.</RICHTEXT>
    </STORY>
  </TEXT>
</CELL>
```

Note that the <TEXT> element must always contain a <STORY> element. A <STORY> element can contain <PARAGRAPH> elements or simply <RICHTEXT> elements.

To add a picture cell, use XML like the following:

```
<CELL BOXTYPE="CT_PICT" COLUMNCOUNT ="1">
  <CONTENT>MacintoshHD:DocPool:flower1.jpg</CONTENT>
  <PICTURE FIT="CENTERPICTURE" />
</CELL>
```

Merging and splitting table cells

To merge table cells, use XML like the following:

```
<TABLE>
 <ID NAME="table1"/>
 <ROW ROWCOUNT="1" MERGEROWSPAN=”1” >
  <CELL COLCOUNT="1"><TEXT>...</TEXT></CELL>
  <CELL COLCOUNT="2"><TEXT>...</TEXT></CELL>
 </ROW>
 <ROW ROWCOUNT="2">
  <CELL COLCOUNT="1"><TEXT>...</TEXT></CELL>
  <CELL COLCOUNT="2"><TEXT>...</TEXT></CELL>
 </ROW>
 <ROW ROWCOUNT="3">
  <CELL COLCOUNT="1"><TEXT>...</TEXT></CELL>
  <CELL COLCOUNT="2"><TEXT>...</TEXT></CELL>
 </ROW>
</TABLE>
```

To split table cells, use XML like the following:

```
<TABLE>
 <ID NAME="table1"/>
 <ROW AUTOFIT="false" ROWCOUNT="5" ROWHEIGHT="60.9">
  <CELL BOXTYPE="CT_TEXT" COLUMNCOUNT="2" SPLIT="true"/>
 </ROW>
</TABLE>
```

Breaking a table across pages

To break a table across pages, use XML like the following:

```
<SPREAD>
 <ID UID="1"/>
 <PAGE MASTER="A-Master A" POSITION="RIGHTOFSPINE">
  <ID UID="1"/>
 </PAGE>
 <TABLE COLOR="none" COLUMNS="2"
MAINTAINGEOMETRY="false"
  ROWS="3" AUTOFIT="rows">
  <ID NAME="Table1"/>
  <TABLEBREAK BREAKHEIGHT="140.251" MAINTAINLINK="true">
   <HEADER>
    <ROW ROWCOUNT="1" ROWHEIGHT="68.625">
    ...
    </ROW>
   </HEADER>
  </TABLEBREAK>
  <ROW ROWCOUNT="1" ROWHEIGHT="68.625">
  ...
  </ROW>
  <ROW ROWCOUNT="2" ROWHEIGHT="68.625">
  ...
  </ROW>
  <FRAME .../>
  <GEOMETRY LAYER="Default" PAGE="1" SHAPE="SH_RECT">
  ...
  </GEOMETRY>
  <COLSPEC>
  ...
  </COLSPEC>
 </TABLE>
</SPREAD>
```

Using inline tables

The Inline Tables feature makes it easy to create an anchored table. Rather than having to specify every attribute of a table, you can simply specify the content

for a table as a series of <TROW> and <ENTRY> elements in an <INLINETABLE> element, like so:

```
<STORY>
 <INLINETABLE>
  <THEAD>
   <TROW>
    <ENTRY>Year</ENTRY>
    <ENTRY>2012</ENTRY>
    <ENTRY>2013</ENTRY>
    <ENTRY>2014</ENTRY>
    <ENTRY>2015</ENTRY>
   </TROW>
  </THEAD>
  <TBODY>
   <TROW>
    <ENTRY>Revenue</ENTRY>
    <ENTRY>000</ENTRY>
    <ENTRY>100</ENTRY>
    <ENTRY>200</ENTRY>
    <ENTRY>300</ENTRY>
   </TROW>
   <TROW>
    <ENTRY>Liabilities</ENTRY>
    <ENTRY>000</ENTRY>
    <ENTRY>100</ENTRY>
    <ENTRY>200</ENTRY>
    <ENTRY>300</ENTRY>
   </TROW>
  </TBODY>
 </INLINETABLE>
 ...
</STORY>
```

The number of rows in such a table is determined by the number of <TROW> elements. The number of columns is determined by the maximum number of <ENTRY> elements in a <TROW>.

In the <TBODY> element, each <TROW> contains one or more <ENTRY> elements. If you don't style the text in an <ENTRY> element, it uses the default styling, which can be defined in a <TROWSTYLE> or <TCOLSTYLE> element (see below).

The <THEAD> element lets you create a repeating header for the table. The <TCONTINUED> element lets you create a "continued" row for the table. If you don't supply either of these elements, you must create the header row manually as a <TROW> in the <TBODY>.

For each row and column, you can specify the following things:

- COLOR: Cell background color.
- SHADE: Cell background shade.
- STORYDIRECTION: Story direction.

You can automatically adjust and position pictures in table cells. The VALIGN@ENTRY and ALIGNMENT@ENTRY attributes lets you specify the alignment of a cell, including picture cells.

The child attribute ORIENTATION lets you specify the orientation of the table, like so:

```
<INLINETABLE> ORIENTATION="LANDSCAPE">
 ...
```

On a portrait page, this would effectively rotate the table in a clock wise direction, while the page itself is not rotated.

The childPICTUREATTRIBUTES element lets you specify several picture attributes, including flip horizontal, flip vertical, angle, background and color, on picture cells of the table.

- This does not create a breakable table. The table will be confined to a single page.
- If the number of rows is greater than the available rectangular size of the parent box, an error is returned stating the table cannot be fit into the available size.
- If you deconstruct a table that was created with an <INLINETABLE> element, the resulting XML describes the table as a <TABLE> element, not an <INLINETABLE> element.

Column Attributes

An <INLINETABLE> can also include optional <COLGROUP> elements, which allow you to specify column attrbutes in the form of <TCOL> elements, like so:

```
<INLINETABLE>
 <COLGROUP>
  <TCOL COLINDEX="1" WIDTH="250"/>
  <TCOL COLINDEX="2" MINWIDTH ="30pt" MAXWIDTH = "90pt"/>
  <TCOL COLINDEX="3" MINWIDTH ="20%"  MAXWIDTH = "50%"/>
 </COLGROUP>
 ...
```

- The COLINDEX value indicates the column number.
- You can specify the WIDTH of a column in points by omitting a unit indicator, or as a percentage of the table width by including a % after the number.

- You can specify the MINWIDTH and MAXWIDTH of a column in points by omitting a unit indicator, or as a percentage of the table width by including a % after the number.

If no width is indicated, the column widths will be automatically computed depending on the length of the text in a cell or image in a cell. The sum total width of all columns would remain within the bounds of the parent box column width (typically the Automatic text box into which the table is laid out).

Table Span

An <INLINETABLE> can include an optional SPAN attribute. Values for SPAN can be:

- PAGEWIDTH - the table will span the width of the page.
- ALLCOLUMNS - the table will span all columns on the page.
- the number of columns to span.

```
<INLINETABLE SPAN="PAGEWIDTH">
  ....
  ....
</INLINETABLE>
```

Using table styles

Table styles make it easy to style inline tables. Rather than applying formatting directly, you can define a table style, then apply the table style to inline tables like so:

```
<INLINETABLE TABLESTYLEREF="TableStyle1">
```

For example, assume you want to create a table where alternating rows are shaded, the grid is a particular color, the insets are a particular amount, and so forth. Instead of specifying the formatting for such a table manually for every row, you can define the table's qualities in a table style, like so:

```
<PROJECT>
 <TABLESTYLE WIDTH="95">
  <ID NAME="tableStyle10"/>
  <TROWSTYLE INSET="2">
   <TOPGRID COLOR="none"/>
   <BOTTOMGRID COLOR="none"/>
  </TROWSTYLE>
  <HEADTROWSTYLE COLOR="red" SHADE="30">
   <TOPGRID COLOR="red" WIDTH="1"/>
   <BOTTOMGRID COLOR="black" WIDTH="1"/>
  </HEADTROWSTYLE>
  <ODDTROWSTYLE COLOR="black" SHADE="20">
   <TOPGRID COLOR="none"/>
```

```
    <BOTTOMGRID COLOR="none"/>
   </ODDTROWSTYLE>
   <EVENTROWSTYLE COLOR="magenta" SHADE="60">
    <TOPGRID COLOR="none"/>
    <BOTTOMGRID COLOR="none"/>
   </EVENTROWSTYLE>
   <TCOLSTYLE>
    <LEFTGRID COLOR="none"/>
    <RIGHTGRID COLOR="none"/>
   </TCOLSTYLE>
   <FIRSTTCOLSTYLE COLOR="Cyan" SHADE="90"/>
   <LASTTCOLSTYLE COLOR="Cyan" SHADE="50"/>
  </TABLESTYLE>
  ...
 </PROJECT>
```

A <TABLESTYLE> lets you specify the following things:

- <TROWSTYLE>: A row style to be applied to every row in the table. One of the two mandatory elements of <TABLESTYLE>. Includes the INSET attribute, which lets you specify the inset to apply on all four sides.
- <HEADTROWSTYLE>: A row style to be applied only to the header row.
- <ODDTROWSTYLE> and <EVENTROWSTYLE>: Row styles that let you format odd and even rows differently.
- <TCOLSTYLE>: A column style. One of the two mandatory elements of <TABLESTYLE>. Note that when the table is created, column styles override row styles.
- <FIRSTTCOLSTYLE> and <LASTTCOLSTYLE>: Column styles that let you style the first and last column of a table differently.
- <TOPGRID> and <BOTTOMGRID>: A grid line at the top or bottom of a row's cells.
- <LEFTGRID> and <RIGHTGRID>: A grid line at the left or right edge of a column's cells.

To apply a table style to an inline table, add a TABLESTYLEREF attribute to the <INLINETABLE> element, like so:

```
<INLINETABLE TABLESTYLEREF="tableStyle10">
```

You can also override <TABLESTYLE> attributes by specifying them as part of the table, like so:

```
<TROW>
 <TOPGRID COLOR="black" WIDTH="1"/>
 <BOTTOMGRID COLOR="red" WIDTH="1"/>
 <ENTRY COLSPAN="5">Statements</ENTRY>
</TROW>
```

Here, we've created a cell that spans five columns by supplying only one <ENTRY>, and we've specified a black, one-point top line and a red, one-point bottom line for that row only.

Working with sections

The Section feature lets you change the numbering system for a layout or a range of pages in a layout. To use this feature, you create a section start on a particular page. In that section start, you can specify a number format, a starting page number, and an optional prefix. For example:

```
<PAGE FORMATTEDNAME="A1" MASTER="A-Master A"
POSITION="RIGHTOFSPINE">
   <ID UID="1"/>

<SECTION FORMAT="ROMAN" NUMBER="1" PREFIX="A"
OPERATION="CREATE"/>

</PAGE>
```

Once you have inserted a <SECTION> element, QuarkXPress Server will apply section-specific numbering and formatting to automatic page numbers. To insert automatic page numbers, use the RICHTEXT@PAGENUMBERCHAR attribute:

```
<TEXT>
   <STORY STORYDIRECTION="HORIZONTAL">
      <PARAGRAPH MERGE="false" PARASTYLE="Normal">
         <RICHTEXT MERGE="false">This is page </RICHTEXT>

<RICHTEXT MERGE="false"
PAGENUMBERCHAR="CURRENTPAGE"/>

         <RICHTEXT MERGE="false">. The story continues on page
          </RICHTEXT>

<RICHTEXT MERGE="false" PAGENUMBERCHAR="NEXTPAGE"/>

         <RICHTEXT MERGE="false">. This story is continued from
page
          </RICHTEXT>

<RICHTEXT MERGE="false"
PAGENUMBERCHAR="PREVIOUSPAGE"/>

      </PARAGRAPH>
   </STORY>
</TEXT>
```

To remove a section break, use XML like the following:

```
<PAGE FORMATTEDNAME="A1" MASTER="A-Master A"
POSITION="RIGHTOFSPINE">
   <ID UID="1"/>

<SECTION OPERATION="DELETE"/>

</PAGE>
```

Working with Composition Zones

A Composition Zones item in a deconstructed project is represented in XML by a <COMPOSITIONZONE> element. Like the <BOX> element type, this element type supports the <GEOMETRY>, <SHADOW>, and <FRAME> elements.

The content of each Composition Zones item is provided by a layout called the *composition layout*, which can be internal or external. Each <COMPOSITIONZONE> element includes a TYPE attribute that indicates whether its composition layout is internal or external.

- For internal Composition Zones items, each Composition Zones item is represented as an additional <LAYOUT> element within the <PROJECT> element. The LAYOUTREF element within the <COMPOSITIONZONE> element indicates the name of the <LAYOUT> that corresponds to that particular Composition Zones item.
- For external Composition Zones items, the PATH attribute indicates the location of the project containing the associated composition layout. However, a copy of the layout is also stored within the project as an additional <LAYOUT> element.

Composition Zones items must be created in QuarkXPress. <COMPOSITIONZONE> elements are ignored by the construct namespace and the modify parameter.

```
<PROJECT>
 <LAYOUT>
  <ID UID="Layout 1"/>
  <SPREAD>
   <ID/>
   <COMPOSITIONZONE BLENDSTYLE="SOLID"
BOXTYPE="CT_USER" COLOR="none"
    LAYOUTREF="Layout 2" PATH="/projects/ExternalZone1.qxp"
TYPE="EXTERNAL">
     <ID NAME="Box9" UID="9"/>
     ...
   </COMPOSITIONZONE>
  </SPREAD>
 </LAYOUT>
 <LAYOUT SHAREDSTATUS="ALLPROJECTS">
```

```
    <ID NAME="Layout 2" UID="2"/>
    <SPREAD>...</SPREAD>
  </LAYOUT>
</PROJECT>
```

You can create a shared layout for use in a Composition Zones item like so:

```
<PROJECT>
  <LAYOUT OPERATION="CREATE"
SHAREDSTATUS="THISPROJECT">
    <ID NAME="ScrollableLayout"/>
    <SPREAD>
      <ID UID="1"/>
      ...
    </SPREAD>
  </LAYOUT>
</PROJECT>
```

If you are creating App Studio issues, you can use this technique to create scrollable layouts on the fly. When doing so, use the HORIZONTALBINDING and VERTICALBINDING attributes to indicate which direction the layout should scroll.

You can modify the following aspects of an existing <COMPOSITIONZONE>:

- HORIZONTALBINDING and VERTICALBINDING
- LAYOUTOPACITY
- LAYOUTREF
- PREVIEWPAGE

For example:

```
<COMPOSITIONZONE HORIZONTALBINDING="false"
LAYOUTOPACITY="100%"
    LAYOUTREF="Layout 6" PREVIEWPAGE="3"
VERTICALBINDING="true">
  <PAGEREF ANGLE="0" NUMBER="1" OFFSETACROSS="0"
OFFSETDOWN="0" SCALE="100%"/>
</COMPOSITIONZONE>
```

Using XSL transformation

You can use an XSLT file to transform the XML returned by the xml namespace into other formats. You might find this feature useful if you want the xml namespace to return an XML representation that uses a different schema or a subset of the returned data.

To use this feature, use the XSL parameter in the request URL. If the XSL parameter specifies the absolute path to an XSLT file on the server, QuarkXPress Server uses that XSLT file to transform the response to that call. For example:

```
http://QXPServer8:8080/xml/project1.qxp?XSL=
```

```
path to XSLT file on server
```

➡ When you use this feature, "XSL" must be in all caps.

To make the returned XML use the Modifier schema, uncheck **Use default XSLT** and do not use the XSL parameter in your calls to the construct namespace.

➡ QuarkXPress Server currently supports only XML output from XSL transformation.

Working with lists

The <LISTS> element allows you to construct and deconstruct QuarkXPress lists. Lists allow a user to automatically create a table of contents (TOC) or list of figures. For more information, see the Modifier schema.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<PROJECT JOBJACKET="Project2 Job Jacket"
  JOBTICKET="Default Job Ticket 1:Project2"
 PROJECTNAME="lis1.qxp" XMLVERSION="8.0">
 <LAYOUT POINTSPERINCH="72">
  <ID NAME="Layout 1"/>
  <LAYER>
   <ID NAME="Default"/>
   <RGBCOLOR BLUE="90" GREEN="90" RED="90"/>
  </LAYER>
  <SPREAD>
   <ID UID="1"/>
   <PAGE MASTER="A-Master A" POSITION="RIGHTOFSPINE">
    <ID UID="1"/>
   </PAGE>
   <BOX BOXTYPE="CT_TEXT" COLOR="none">
    <ID NAME="Box5"/>
    <GEOMETRY>
     <POSITION>
      <TOP>56</TOP>
      <LEFT>56</LEFT>
      <BOTTOM>200</BOTTOM>
      <RIGHT>300</RIGHT>
     </POSITION>
    </GEOMETRY>
    <TEXT>
     <STORY>
      <LIST LISTSTYLE="New List" OPERATION="CREATE">
      </LIST>
     </STORY>
    </TEXT>
   </BOX>
  </SPREAD>
```

```
</LAYOUT>
</PROJECT>
```

LIST is a child of the STORY element. The value of LISTSTYLE will be the name of the list that had been created in QuarkXPress. When a project containing a list is deconstructed in XML, the XML will contain the text of the list, as well as a reference back to the LIST.

Working with anchored boxes

To create an anchored box within a text box, use a structure like the following:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<PROJECT JOBJACKET="Macintosh HD:Server:Project1 Job Jacket"
 JOBTICKET="Default Job Ticket 1:Project2"
  PROJECTNAME="anchor.qxp" XMLVERSION="8.0">
  <LAYOUT POINTSPERINCH="72">
    <ID NAME="Layout 1"></ID>
    <LAYER>
      <ID NAME="Default"/>
      <RGBCOLOR BLUE="90" GREEN="90" RED="90"/>
    </LAYER>
    <SPREAD>
      <ID UID="1"/>
      <PAGE MASTER="A-Master A" POSITION="RIGHTOFSPINE">
        <ID UID="1"/>
      </PAGE>
      <BOX BOXTYPE="CT_TEXT" COLOR="none">
        <ID NAME="Box5"/>
        <GEOMETRY LAYER="Default" PAGE="1">
          <POSITION>
            <TOP>36</TOP>
            <LEFT>36</LEFT>
            <BOTTOM>112</BOTTOM>
            <RIGHT>210</RIGHT>
          </POSITION>
        </GEOMETRY>
        <TEXT>
          <STORY>
            <PARAGRAPH MERGE="false" PARASTYLE="Normal">
              <RICHTEXT MERGE="false">Hello </RICHTEXT>
              <ANCHOREDBOXREF ALIGNWITHTEXT="BASELINE"
                OFFSET="0">Box7</ANCHOREDBOXREF>
              <RICHTEXT MERGE="false">, world</RICHTEXT>
            </PARAGRAPH>
          </STORY>
        </TEXT>
      </BOX>
```

```
            <BOX ANCHOREDIN="Box5" BOXTYPE="CT_TEXT"
        COLOR="none">
              <ID NAME="Box7" UID="7"/>
              <GEOMETRY PAGE="1" SHAPE="SH_RECT">
                <POSITION>
                  <TOP>0</TOP>
                  <LEFT>0</LEFT>
                  <BOTTOM>50</BOTTOM>
                  <RIGHT>75</RIGHT>
                </POSITION>
              </GEOMETRY>
              <TEXT>
                <STORY>
                  <PARAGRAPH MERGE="false" PARASTYLE="Normal">
                    <RICHTEXT MERGE="false">anchored box
                     </RICHTEXT>
                  </PARAGRAPH>
                </STORY>
              </TEXT>
            </BOX>
          </SPREAD>
        </LAYOUT>
      </PROJECT>
```

Note that there are two BOX elements. One is the parent box that has the element ANCHOREDBOXREF, which points to the name of the anchored box. The anchored box itself has the attribute ANCHOREDIN, which points to the name of the parent box.

Working with placeholders

Placeholders allow a region of text in a QuarkXPress project to hold non-printing metadata. You can use placeholders to store information from other systems, or to provide information to third-party XTensions software or other tools that operate on QuarkXPress projects.

Placeholders are used by technologies within QuarkXPress, such as XML import. Modifier XT allows placeholder data to be added to a QuarkXPress project from your application, and the placeholder data can be read from a project using the xml namespace.

- Unless a third-party XTensions software module for QuarkXPress is created to manage the placeholders inserted by your application using Modifier XML, a user is not prohibited from deleting placeholders from within the QuarkXPress user interface. In fact, users are not alerted to the presence of placeholders through the QuarkXPress user interface. You can use APIs in the QuarkXPress Server XTensions Software XDK to allow a suitable user interface for managing the placeholders inserted by your application.

Contact QuarkAlliance for details about the XTensions software developer program.

There are two types of placeholders supported in Modifier XML: Text placeholders and Text Node placeholders. Text placeholders can be placed around a run of text to identify particular metadata with that text content.

```
<PROJECT>
  <LAYOUT>
    <ID UID="1"/>
    <SPREAD>
      <ID UID="1"/>
      <BOX>
        <ID NAME="name"/>
        <TEXT>
          <STORY CLEAROLDTEXT="true">
            <PARAGRAPH PARASTYLE="Normal"/>
            <RICHTEXT>This is text that</RICHTEXT>
            <TEXTPH NAME="SOURCE_UID"
OWNER="1347639377">
              <RICHTEXT>has a placeholder</RICHTEXT>
            </TEXTPH>
          </STORY>
        </TEXT>
      </BOX>
    </SPREAD>
  </LAYOUT>
</PROJECT>
```

When a Text placeholder spans multiple paragraphs, the PARAGRAPH and RICHTEXT hierarchy is flattened. A new paragraph can be started using an empty PARAGRAPH element.

Text Node placeholders can represent a hierarchical structure of meta-tagging around text. This can allow more complex meta-tagging of data placed into a QuarkXPress project. Also, it allows some structure to be preserved within the QuarkXPress project format.

```
<PROJECT>
  <LAYOUT>
    <ID UID="1"/>
    <SPREAD>
      <ID UID="1"/>
      <BOX>
        <ID NAME="name"/>
        <TEXT>
          <STORY CLEAROLDTEXT="true">
            <PARAGRAPH PARACHAR="HARDRETURN"/>
            <TEXTNODEPH NAME="ARTICLE"
OWNER="1347639377">
```

```
                <TEXTPH NAME="HEADLINE">
                    <PARAGRAPH PARASTYLE="Headline"/>
                    <RICHTEXT>Text</RICHTEXT>
                </TEXTPH>
                <TEXTPH NAME="STANDFIRST">
                    <PARAGRAPH PARACHAR="HARDRETURN"
                      PARASTYLE="1st para"/>
                    <RICHTEXT>Text</RICHTEXT>
                </TEXTPH>
                <TEXTPH NAME="BODY">
                    <PARAGRAPH PARACHAR="HARDRETURN"
                      PARASTYLE="Body"/>
                    <RICHTEXT>Text</RICHTEXT>
                </TEXTPH>
                <METADATA>
                  <VALUE KEY="ARTICLE_ID">1145</VALUE>
                  <VALUE KEY="ARTICLE_TYPE">Press Release
                  </VALUE>
                  <VALUE KEY="AUTHOR">M.Gutherie</VALUE>
                </METADATA>
              </TEXTNODEPH>
            </STORY>
          </TEXT>
        </BOX>
      </SPREAD>
    </LAYOUT>
</PROJECT>
```

- To avoid hierarchy conflicts between the placeholder hierarchy and the paragraph hierarchy, the paragraph structure is flattened, which means that PARAGRAPH and RICHTEXT elements become siblings. In this case, the PARACHAR attribute is not applied, and the Modifier XML should include the
- entity to represent paragraph break characters.
- The OWNER attribute of the TEXTPH and TEXTNODEPH elements refers to the ID of the XTensions software that is responsible for the placeholder. The xml namespace returns all placeholders from all XTensions software. The default value for placeholders is “1347639377” (this is the XTension ID of PlaceholderSXT XT). If you want to create placeholders for your own XTensions software, use that XTensions software ID here.

Working with metadata

You can attach box-level metadata to a QuarkXPress project created from XML using the Modifierschema. For example, if you import a picture from a content management system into a box, you can store the unique ID of that picture (and other information, such as the last-modified date) with the box containing that

picture. When you deconstruct the project, you can read the metadata (for example, to track the usage of licensed pictures).

You can attach metadata to picture boxes, text boxes, tables, lines, and text paths. QuarkXPress Server metadata takes the form of key/value pairs. For more information, see the Modifier schema.

To create a new box with metadata, use XML like the following. In this example, QuarkXPress Server creates a box named “box1” and asssociates Asset, Date, and Password key-value pairs with it.

```
<BOX OPERATION="CREATE" BOXTYPE="CT_TEXT">
  <ID NAME="box1"/>
  <METADATA>
    <VALUE KEY="Asset" ><![CDATA[1234567890]]>
    </VALUE>
    <VALUE KEY="Date" ><![CDATA[08.06.07]]>
    </VALUE>
    <VALUE KEY="Password" ><![CDATA[Hello World]]>
    </VALUE>
  </METADATA>
  <GEOMETRY SHAPE="SH_RECT" PAGE="1">
    <POSITION>
      <TOP>5</TOP>
      <LEFT>5</LEFT>
      <BOTTOM>10</BOTTOM>
      <RIGHT>10</RIGHT>
    </POSITION>
  </GEOMETRY>
</BOX>
```

To delete metadata that is associated with a box, use XML like the following:

```
<BOX>
<ID NAME="BoxWithMetadata"/>
<METADATA>
<VALUE KEY="Asset"></VALUE>
  </METADATA>
</BOX>
```

Working with hidden text

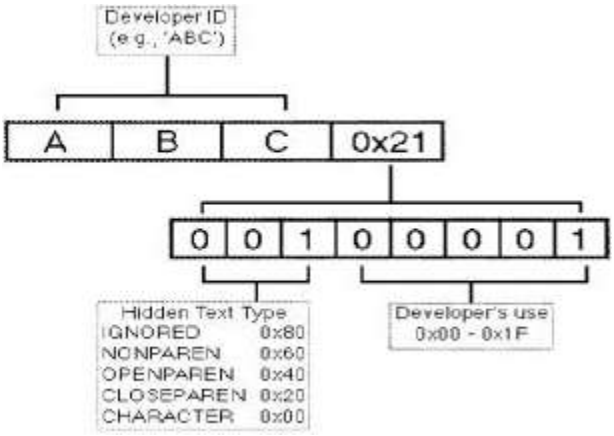
In QuarkXPress, hidden text is used by features which need to include information within the flow of text without that information being visible in its raw form, either on screen or at output. XTensions modules for QuarkXPress and QuarkXPress Server can use the data area in hidden text to store their custom data without changing the surrounding text. The custom data in the hidden text is simply invisible when opened in a copy of QuarkXPress that does not have the corresponding XTensions module. You can use hidden text in Modifier XML to interpret information added by a custom QuarkXPress XTension

or to send instructions to a QuarkXPress Server XTensions during a modify or construct request.

Each piece of hidden text is identified by an *opcode*. An opcode is a four-digit hexadecimal number which specifies:

- The developer ID of the developer who created the XTensions module.
- The unique identifier of the hidden text type, as defined by the XTensions developer.
- The type of hidden text entry (OPENPAREN, CLOSEPAREN, NONPAREN, or IGNORED)

An opcode is constructed as follows:



The structure of a hidden text marker

In Modifier XML, hidden text is represented by the HIDDEN element. By default, hidden text is not output from the xml namespace. To output hidden text, specify the opcode= parameter in your request, like so:

http://server:port/xml/projectname.qxp?opcode=51434450

➡ This example URL outputs all of the hidden text inserted by the XTensions software with this ID. To avoid byte order issues when cross-platform rendering is enabled, the XTID is represented decimally, rather than with the usual char[4] representation.

If you specify opcode=*, QuarkXPress Server returns all hidden text in the XML output. If you want only specific opcodes for a particular developer ID, you must pass the developer ID + the unique ID (more often than not, 1) + the sum of types of paren you wish to see (for example, to see OPENPAREN and CLOSEPAREN, you would calculate 0x20 + 0x40 = 0x60). For example, if you wanted to get only hidden text from the Custom Underline XTensions module, you would pass the request with the additional request parameter opcode=51526B61. The data you receive in the deconstructed hidden text is a base64 encoded version of the binary data which is stored in the hidden text. To interpret this, you must know the data structure which the XTension uses.

Similarly, when you pass data back to an XTensions module through a modify or construct request, the data passed in the <HIDDEN> element must be base-64 encoded, and must be a valid structure in the format which the XTensions module is expecting.

You can use hidden text in different ways by using different hidden text types. For example, the Notes XTensions module uses the OPENPAREN and CLOSEPAREN hidden text type. This XTensions module lets users embed user comments at particular locations in text and view these comments can in a "sticky note" window. To accomplish this, the XTensions module embeds two hidden text markers in the text, and the text of the note goes between them. The piece of hidden text at the start of the note has the type OPENPAREN, and the piece at the end has the type CLOSEPAREN.

```
<PARAGRAPH MERGE="false" PARACHAR="HARDRETURN"
  PARASTYLE="001-TEXT">
   <RICHTEXT MERGE="false">
      The population of Iceland is 500,000,000.
   </RICHTEXT>
   <HIDDEN DATALEN="100" OPCODE="51434450"
     OWNER="514344" TYPE="CHARACTERTYPE">
      <RICHTEXT LANGUAGE="USEnglish" MERGE="false">
         VGhpcyBpcyB0aGUgdGV4dCBvZiBhIENvcHlEZXNrIG5vdGU=
      </RICHTEXT>
   </HIDDEN>
   <RICHTEXT MERGE="false">
      Iceland is located north of the Equator.
   </RICHTEXT>
</PARAGRAPH>
```

The example XML extract above shows the output from the xml namespace of text that contains a note inserted by the Notes XT XTensions software. The note contains "This is the text of a CopyDesk note," which is represented as VGhpcyBpcyB0aGUgdGV4dCBvZiBhIENvcHlEZXNrIG5vdGU=. If this text is passed back to QuarkXPress Server in a modify or construct request, the hidden text inserted by the Notes XT XTensions software is preserved and can be read by the Notes XT XTensions software if the project is opened in QuarkXPress.

The Custom Underline XTensions module feature also uses this approach, but also stores the custom underline definitions in a binary data structure within the data of the CLOSEPAREN hidden text entry:

```
<PARAGRAPH MERGE="false" PARASTYLE="Normal">
   <RICHTEXT MERGE="false">
      The population of Iceland is 500,000,000.
   </RICHTEXT MERGE="false">regular text</RICHTEXT>
   <HIDDEN DATALEN="0" OPCODE="51526B41"
     OWNER="51526B" TYPE="OPENPAREN">
   </RICHTEXT MERGE="false">text with custom
```

```
underline</RICHTEXT>
   <HIDDEN DATALEN="20" OPCODE="51526B21"
     OWNER="51526B" TYPE="CLOSEPAREN">
      <RICHTEXT
MERGE="false">/////wACAAAAAKj2AAIAAABqAAU=</RICHTEXT>
   </HIDDEN>
   <RICHTEXT MERGE="false">regular text</RICHTEXT>
</PARAGRAPH>
```

The data within the RICHTEXT element inside a HIDDEN element is a Base 64-encoded representation of the raw data that is stored within the hidden text. Considering that hidden text in QuarkXPress can contain any type of data, and the structure of that data is specified by the XTensions software that creates it, this method ensures that the data can be safely represented in XML. Also, this data can be converted back into the same raw data structure so that it can be read by the destination XTensions software. If the content is edited, the destination XTensions software may not be able to interpret it. Only XTensions software developers should attempt to interpret data from their own XTensions software.

Using interactivity

The <INTERACTIVITY> element describes an asset used as an interactive element for a format such as App Studio and ePUB.

The specific schema for an interactive element is determined by the XTensions module that owns that element, so such schemas are not defined here. The best way to create or modify an <INTERACTIVITY> element is to deconstruct it and then use the deconstructed XML as a template. Below are some examples of App Studio interactivity.

Button interactivity

Below is an example of App Studio **Button** interactivity.

```
<ID UID="8"/>
<BOX BLENDSTYLE="SOLID" BOXTYPE="CT_PICT" COLOR="none">
<ID NAME="Button 2" UID="36"/>
<INTERACTIVITY AUTHORXTID="1131430225"
OWNERXTID="1129333841" TYPE="Button">
<Settings>
<settings>
<actions>
<action>
<type>gotofirstpage</type>
<name/>
</action>
</actions>
```

```
</settings>
</Settings>
<DATAPROVIDER DATAPROVIDERXTID="1131430225"/>
</INTERACTIVITY>
</BOX>
```

Scroll Zone interactivity

Below is an example of App Studio **Scroll Zone** interactivity.

```
<COMPOSITIONZONE LAYOUTREF="Scroll" OPERATION="CREATE">
<ID NAME="ScrollZone 1" UID="58"/>
<GEOMETRY CORNERRADIUS="0" CORNERSTYLE="RECTANGLE"
LAYER="Default" PAGE="7" SHAPE="SH_RECT" SKEW="0">
<POSITION LOCKPROPORTIONS="false">
<TOP>216</TOP>
<LEFT>149</LEFT>
<BOTTOM>668</BOTTOM>
<RIGHT>630</RIGHT>
</POSITION>
<SUPPRESSOUTPUT>false</SUPPRESSOUTPUT>
<RUNAROUND TYPE="NONE"/>
</GEOMETRY>
<FRAME COLOR="Black" GAPCOLOR="none" OPACITY="100%"
SHADE="100%" STYLE="Solid" WIDTH="0"/>
<PAGEREF ANGLE="0" NUMBER="1" OFFSETACROSS="0"
OFFSETDOWN="0" SCALE="100%"/>
<INTERACTIVITY AUTHORXTID="1131430225"
OWNERXTID="1129333841" TYPE="Scrollable Content">
<Settings>
<scrollzonesettings>
<defaultposition>1</defaultposition>
<fadeatends>true<fadedistance>60</fadedistance>
</fadeatends>
<showscrollbar>true</showscrollbar>
<automaticarrows>false</automaticarrows>
<loop>false</loop>
<docuid>4</docuid>
   <scrolldirection>1</scrolldirection>
</scrollzonesettings>
</Settings>
<DATAPROVIDER DATAPROVIDERXTID="1131430225"/>
</INTERACTIVITY>
</COMPOSITIONZONE>
```

- `<scrolldirection>1</scrolldirection>` Applies the Horizontal Scrolling
- `<scrolldirection>2</scrolldirection>` Applies the Vertical Scrolling

Slideshow interactivity

Below is an example of App Studio **Slideshow** interactivity.

```
<BOX BLENDSTYLE="SOLID" BOXTYPE="CT_PICT" COLOR="none">
<ID NAME="Slideshow 1" UID="18"/>
<PICTURE ANGLE="0" DPI="144" FLIPHORIZONTAL="false"
FLIPVERTICAL="false" FULLRES="false" MASK="Composite"
OFFSETACROSS="0" OFFSETDOWN="0" OPACITY="100%"
SCALEACROSS="100%" SCALEDOWN="100%" SKEW="0"
SUPRESSPICT="false">
<CLIPPING TYPE="ITEM"/>
</PICTURE>
<CONTENT PICTURECONTENTLOCK="true"
UID="9">Images\Slideshow01.jpg</CONTENT>
<INTERACTIVITY AUTHORXTID="1131430225"
OWNERXTID="1129333841" TYPE="Slideshow">
<Settings>
<slideshowsettings>
<allowfullscreen>true</allowfullscreen>
<allowinteraction>true</allowinteraction>
<uncroppedinfullscreen>false</uncroppedinfullscreen>
<autoplay>false</autoplay>
<sourcesettings>
<sourcetype>1</sourcetype>
</sourcesettings>
<animate>true<animationduration>6.000000</animationduration>
</animate>
<slides>
<slide slidetype="imagetype">
<imagepath>Images\slideshow1.jpg</imagepath>
<animatesettings>
<startcrop angle="0" xoffset="0" xscale="65536" yoffset="0"
yscale="65536"/>
<endcrop angle="0" xoffset="0" xscale="65536" yoffset="0"
yscale="65536"/>
</animatesettings>
<actions/>
</slide>
<slide slidetype="imagetype">
<imagepath>Images\slideshow2.jpg</imagepath>
<animatesettings>
<startcrop angle="0" xoffset="0" xscale="65536" yoffset="0"
yscale="65536"/>
<endcrop angle="0" xoffset="0" xscale="65536" yoffset="0"
yscale="65536"/>
</animatesettings>
<actions/>
```

```
</slide>
<slide slidetype="imagetype">
<imagepath>Images\slideshow3.jpg</imagepath>
<animatesettings>
<startcrop angle="0" xoffset="0" xscale="65536" yoffset="0"
yscale="65536"/>
<endcrop angle="0" xoffset="0" xscale="65536" yoffset="0"
yscale="65536"/>
</animatesettings>
<actions/>
</slide>
</slides>
</slideshowsettings>
</Settings>
<DATAPROVIDER DATAPROVIDERXTID="1131430225"/>
</INTERACTIVITY>
</BOX>
```

Video interactivity

Below is an example of App Studio **Video** interactivity.

```
<BOX BLENDSTYLE="SOLID" BOXTYPE="CT_PICT" COLOR="none">
<ID NAME="Video 1" UID="21"/>
<PICTURE/>
<INTERACTIVITY AUTHORXTID="1131430225"
OWNERXTID="1129333841" TYPE="Video">
<Settings>
<videosettings>
<autoplay>false</autoplay>
<fullscreenonly>false</fullscreenonly>
<loop>false</loop>
<hidecontroller>false</hidecontroller>
<sourcesettings>
<sourcetype>1</sourcetype>
<sourcepath>Video\abc.mp4</sourcepath>
</sourcesettings>
<usevideoframe>false</usevideoframe>
<useofflineimage>false</useofflineimage>
</videosettings>
</Settings>
<DATAPROVIDER DATAPROVIDERXTID="1131430225"/>
</INTERACTIVITY>
</BOX>
```

Audio interactivity

Below is an example of App Studio **Audio** interactivity.

```
<BOX BLENDSTYLE="SOLID" BOXTYPE="CT_PICT" COLOR="none">
<ID NAME="Audio 1" UID="24"/>
<PICTURE/>
<INTERACTIVITY AUTHORXTID="1131430225"
OWNERXTID="1129333841" TYPE="Audio">
<Settings>
<audiosettings>
<autoplay>false</autoplay>
<loop>false</loop>
<hidecontroller>false</hidecontroller>
<stopatpageturn>true</stopatpageturn>
<stopatarticleend>true</stopatarticleend>
<sourcesettings>
<sourcetype>1</sourcetype>
<sourcepath>Audio\abc.mp3</sourcepath>
</sourcesettings>
<useofflineimage>false</useofflineimage>
</audiosettings>
</Settings>
<DATAPROVIDER DATAPROVIDERXTID="1131430225"/>
</INTERACTIVITY>
</BOX>
```

Go to URL interactivity

Below is an example of App Studio **Go to URL** interactivity.

```
<BOX BLENDSTYLE="SOLID" BOXTYPE="CT_PICT" COLOR="none">
<ID NAME="Button 1" UID="6"/>
<PICTURE/>
<INTERACTIVITY AUTHORXTID="1131430225"
OWNERXTID="1129333841" TYPE="Button">
<Settings>
<settings>
<actions>
<action>
<type>gotourl</type>
<name/>
<gotourl>http://www.google.com</gotourl>
<switchtobrowser>false</switchtobrowser>
</action>
</actions>
</settings>
</Settings>
<DATAPROVIDER DATAPROVIDERXTID="1131430225"/>
</INTERACTIVITY>
</BOX>
```

Web View interactivity

Below is an example of App Studio **Web View** interactivity.

```
<BOX BLENDSTYLE="SOLID" BOXTYPE="CT_PICT" COLOR="none">
<ID NAME="WebView 1" UID="27"/>
<INTERACTIVITY AUTHORXTID="1131430225"
OWNERXTID="1129333841" TYPE="Embedded HTML">
<Settings>
<webviewsettings>
<allowuserinteraction>true</allowuserinteraction>
<scrollable>false</scrollable>
<allowzoom>false</allowzoom>
<sourcesettings>
<sourcetype>2</sourcetype>
<sourcepath>http://www.gsmarena.com</sourcepath>
</sourcesettings>
<useofflineimage>true<offlineimagepath/>
</useofflineimage>
</webviewsettings>
</Settings>
<DATAPROVIDER DATAPROVIDERXTID="1131430225"/>
</INTERACTIVITY>
</BOX>
```

Picture Zoom interactivity

Below is an example of App Studio **Picture Zoom** interactivity.

```
<BOX BLENDSTYLE="SOLID" BOXTYPE="CT_PICT" COLOR="none">
<ID NAME="InteractivePicture 1" UID="39"/>
<PICTURE ANGLE="0" DPI="144" FLIPHORIZONTAL="false"
FLIPVERTICAL="false" FULLRES="false" MASK="Composite"
OFFSETACROSS="0" OFFSETDOWN="27.233" OPACITY="100%"
SCALEACROSS="83.2%" SCALEDOWN="83.2%" SKEW="0"
SUPRESSPICT="false">
<CLIPPING TYPE="ITEM"/>
</PICTURE>
<CONTENT PICTURECONTENTLOCK="true"
UID="9">Images\Slideshow01.jpg</CONTENT>
<INTERACTIVITY AUTHORXTID="1131430225"
OWNERXTID="1129333841" TYPE="Full-screen Image">
<Settings>
<picturezoomsettings>
<allowfullscreen>true</allowfullscreen>
<allowpinchzoom>false</allowpinchzoom>
<allowpanning>true</allowpanning>
<animatepanandzoom>false</animatepanandzoom>
<zoomsetting>0</zoomsetting>
```

```
</picturezoomsettings>
</Settings>
<DATAPROVIDER DATAPROVIDERXTID="1131430225"/>
</INTERACTIVITY>
</BOX>
```

360 degree interactivity

Below is an example of App Studio **360 degree** interactivity.

```
<ID UID="8"/>
 <BOX BLENDSTYLE="SOLID" BOXTYPE="CT_PICT" COLOR="none">
  <ID NAME="360Degree 1" UID="31"/>
   <PICTURE/>
   <INTERACTIVITY AUTHORXTID="1131430225" NAME="360°
Image 1"
    OWNERXTID="1129333841" TYPE="360 degree Image">
    <SettingsInitiallyHidden="False">
     <image360settings>
      <autoplay>true<spincount>2</spincount></autoplay>
      <allowinteraction>true</allowinteraction>
      <frames/>
     </image360settings>
    </Settings>
    <DATAPROVIDER DATAPROVIDERXTID="1131430225"/>
   </INTERACTIVITY>
 </BOX>
```

Animation interactivity

Below is an example of App Studio **Animation** interactivity.

```
<ID UID="8"/>
 <BOX BLENDSTYLE="SOLID" BOXTYPE="CT_PICT" COLOR="none">
  <ID NAME="animation 1" UID="32"/>
  <PICTURE/>
  <INTERACTIVITY AUTHORXTID="1131430225" NAME="Animation
1"
   OWNERXTID="1129333841" TYPE="Animation">
   <Settings InitiallyHidden="False">
    <animationsettings>
     <animationtype>11</animationtype>
     <autoplay>true</autoplay>
     <allowinteraction>true</allowinteraction>
     <initiallyhidden>false</initiallyhidden>
     <loop>false<loopcount>1</loopcount></loop>
     <duration>5</duration>
     <delay>0</delay>
     <timingfunc>
```

```
      <functype>0</functype>
      <func/>
     </timingfunc>
     <pathname/>
     <direction>0</direction>
     <endsettings>
      <hidden>false</hidden>
      <opacity>65536</opacity>
      <angle>0</angle>
      <xscale>65536</xscale>
      <yscale>65536</yscale>
      <scaleproportionally>true</scaleproportionally>
     </endsettings>
    </animationsettings>
   </Settings>
   <DATAPROVIDER DATAPROVIDERXTID="1131430225"/>
  </INTERACTIVITY>
</BOX>
```

Specifying colors

When specifying colors, you can use named Web colors such as “Silver” and “MediumSlateBlue” by name. You can also use RGB colors by specifying their hexadecimal values. For example:

```
<RICHTEXT COLOR="Teal" MERGE="false" BOLD="true">This text is teal.</RICHTEXT>
<FRAME COLOR="#006699" GAPCOLOR="#996600" WIDTH="7"/>
```

Working with indexes

Indexes are a list of words or phrases (‘Index Terms’) and associated pointers to where useful material relating to that index term can be found in a document or on a page.

Index terms can be nested up to four levels and each level can have a separate style applied. The scope of the index term can also be set. Preferences for indexes can be defined, including setting the index marker color, the index term separator, the page range separator and the page list separator.

Marking and rendering indexes

Marking:

The INDEXTERM tag is used to mark indexes within the flow.

```
<PARAGRAPH PARASTYLE="Sub-Section Title">
  <RICHTEXT>Text containing the term</RICHTEXT>
  <INDEXTERM>the term to be marked for the index</INDEXTERM>
```

```
</PARAGRAPH>
```

The RANGE element is used to mark the beginning and end of the scope of the occurrence of the index within the flow.

```
<INDEXTERM ID="Unique ID" RANGE="START">Menu</INDEXTERM>
<PARAGRAPH>
  <RICHTEXT>....</RICHTEXT>
</PARAGRAPH>
<INDEXTERM ID="Unique ID" RANGE="END"/>
```

Use the RANGE element with a value of UNTILPARASTYLE to specify that a range should end at a particular style sheet.

```
<PARAGRAPH PARASTYLE="Sub-Section Title">
  <RICHTEXT>File Menu</RICHTEXT>
</PARAGRAPH>
<INDEXTERM RANGE="UNTILPARASTYLE" VALUE="Sub-Section
Title">File</INDEXTERM>
<PARAGRAPH>
  <RICHTEXT>....</RICHTEXT>
</PARAGRAPH>
< PARAGRAPH PARASTYLE="Sub-Section Title">
  < RICHTEXT>Edit Menu</RICHTEXT>
</PARAGRAPH>
```

Use the RANGE element with a value of NUMBEROFPARAGRAPHS to specify that a range should end after a specified number of paragraphs.

```
<PARAGRAPH PARASTYLE="Sub-Section Title">
  <RICHTEXT>File Menu</RICHTEXT>
</PARAGRAPH>
<INDEXTERM RANGE="NUMBEROFPARAGRAPHS"
VALUE="12">File</INDEXTERM>
<PARAGRAPH>
  <RICHTEXT>....</RICHTEXT>
</PARAGRAPH>
<PARAGRAPH>
  <RICHTEXT>Edit Menu</RICHTEXT>
</PARAGRAPH>
```

Use the RANGE element with a value of ENDOFSTORY to specify that a range should end at the end of the story.

```
<PARAGRAPH PARASTYLE="Sub-Section Title">
  <RICHTEXT>File Menu</RICHTEXT>
</PARAGRAPH>
<INDEXTERM RANGE="ENDOFSTORY">File</INDEXTERM>
<PARAGRAPH>
  <RICHTEXT>....</RICHTEXT>
</PARAGRAPH>
<PARAGRAPH>
  <RICHTEXT>Edit Menu</RICHTEXT>
```

```
</PARAGRAPH>
```

Use the ADDALL element to indicate that all instances of the index term in the document should be added to the index.

```
<INDEXTERM ID="Unique ID" RANGE="START"
ADDALL="TRUE">Menu</INDEXTERM>
   <RICHTEXT>Term</RICHTEXT>
<INDEXTERM ID="UniqueID" RANGE="END"/>
```

Use the MAINTERM, SUBTERM1, SUBTERM2, and SUBTERM3, elements to specify the level of the index term.

```
<RICHTEXT>Menus of QXP</RICHTEXT>
<INDEXTERM>
   <MAINTERM>Menus</MAINTERM>
   <SUBTERM1>Sub menu</SUBTERM1>
   <SUBTERM2>Utilities menu</SUBTERM1>
   <SUBTERM3>Insert Placeholder Text</SUBTERM1>
</INDEXTERM>
```

The previous XML snippet will produce the following results in the index:

```
Menus
 Sub menu
   Utilities menu
     Insert Placeholder Text 1
```

Use the CROSSREFERENCETOINDEX element to create cross references between index terms. (Valid PREFIX values: see, see also, see herein)

```
<PARAGRAPH>
   <INDEXTERM>
      <MAINTERM>Menu</MAINTERM>
      <CROSSREFERENCETOINDEX PREFIX="see
also">Palettes</CROSSREFERENCETOINDEX>
   </INDEXTERM>
   <RICHTEXT>Term</RICHTEXT>
</PARAGRAPH>
```

Use the INDEXSPECIFICATIONS tag to define the index preferences, including setting the style and the index term separators.

```
<INDEXSPECIFICATIONS>
   <INDEXSTYLE NAME ="IndexStyle">
   <SEPARATORS BETWEEN-PAGE-NUMBERS=""
BETWEEN-PAGE-RANGE=""
         FOLLOWING-ENTRIES=""
         BEFORE-CROSS-REFERENCE=""
         CROSS-REFERENCE-STYLE=""
         LEVELFORMAT="NESTED"/>
   <LEVELSTYLE FIRSTLEVEL ="Style1"
         SECONDLEVEL ="Style2"
         THIRDLEVEL ="Style3"
```

```
                FOURTHLEVEL =" Style4"
                LETTERHEADSTYLE ="Style5"/>
    </INDEXSTYLE>
</INDEXSPECIFICATIONS>
```

Rendering:

The INDEX OPERATION element is used to render indexes.

```
<PARAGRAPH>
  <INDEX OPERATION="CREATE" Style="IndexStyle"/>/>
</PARAGRAPH>
```

Working with cross references

The cross references feature makes it easy to create cross references. There are three types of cross references that can be created in QuarkXPress:

- Footnotes
- Endnotes
- Numbered Items

Use the <XREF> elements in a <REFNOTE> element, to specify cross references.

To create a numbered item cross reference:

```
<PAGESEQUENCE MASTERREFERENCE="B-Body">
 <STORY>
  <PARAGRAPH PARASTYLE="Normal">
   <RICHTEXT MERGE="false">Paragraph text (see Section
</RICHTEXT>
    <XREF HREF="#d83c7191-fbb9-4565-8364-4de14c77c44a"
      CHARSTYLE="CrossRef" INCLUDEABOVEBELOW="true"
HYPERLINK="FALSE"
      SEPARATOR="->" />
    <RICHTEXT MERGE="false">and Appendix A) Paragraph text
continues.</RICHTEXT>
  </PARAGRAPH>
   . . .
  <PARAGRAPH PARASTYLE="Heading1">
   <RICHTEXT MERGE="false">Support</RICHTEXT>
  </PARAGRAPH>
  <PARAGRAPH PARASTYLE="Heading2" XREFLABEL="d83c7191-
fbb9-4565-8364-4de14c77c44a">
   <RICHTEXT MERGE="false">Loss History</RICHTEXT>
  </PARAGRAPH>
 </STORY>
</PAGESEQUENCE>
```

- Use the CHARSTYLE element to identify a character style sheet to be applied to the cross reference.

- Use the HYPERLINK element to enable/disable the hyperlink in the PDF output.
- Use the XREFSTYLE element to specify the type of cross reference.
- Use the SEPARATOR element to specify the separator text.

You can show custom text as the cross reference:

```
<PAGESEQUENCE MASTERREFERENCE="B-Body">
 <STORY>
  <PARAGRAPH PARASTYLE="Normal">
   <RICHTEXT MERGE="false">Paragraph text (see Section
</RICHTEXT>
    <XREF HREF="#d83c7191-fbb9-4565-8364-4de14c77c44a"
CHARSTYLE="CrossRef/>CUSTOM TEXT</XREF>
    <RICHTEXT MERGE="false">and Appendix A) Paragraph text
continues.</RICHTEXT>
  </PARAGRAPH>
   . . .
  <PARAGRAPH PARASTYLE="Heading1">
   <RICHTEXT MERGE="false">Support</RICHTEXT>
  </PARAGRAPH>
  <PARAGRAPH PARASTYLE="Heading2" XREFLABEL="d83c7191-
fbb9-4565-8364-4de14c77c44a">
   <RICHTEXT MERGE="false">Loss History</RICHTEXT>
  </PARAGRAPH>
 </STORY>
</PAGESEQUENCE>
```

Creating and using hyperlinks

There are three types of hyperlink:

- Web (WWWURL).
- Anchor (ANCHOR). You must define Anchor hyperlinks at the <LAYOUT> level.
- Page (PAGE). You must define Anchor hyperlinks at the <LAYOUT> level.

Web hyperlinks

You must define Web hyperlinks at the <PROJECT> level. For example, to create a Web hyperlink named Quark-dot-com, you could add the following as a child of the <PROJECT> element:

```
<HYPERLINK HLTYPE="WWWURL" NAME="Quark-dot-com"
TARGET="http://www.quark.com"/>
```

To add a Web hyperlink to a layout, add HYERLINKREF and HLTYPE attributes to a <BOX> or <RICHTEXT> element. For example, to use the Quark-dot-com hyperlink defined above, you could do something like this:

```
<RICHTEXT>this is a hyperlink to </RICHTEXT>
<RICHTEXT COLOR="Cyan" UNDERLINE="true"
HYPERLINKREF="Quark-dot-com" HLTYPE="WWWURL"
>quark.com</RICHTEXT>
```

- You can use a Web hyperlink *without* creating it at the <PROJECT> level, but this is not the preferred method.

Anchor hyperlinks

To indicate the target of an Anchor hyperlink, use a <RICHTEXT> element like this:

```
<RICHTEXT HLANCHORREF="MyAnchor" />
```

To make sure the Anchor hyperlink works correctly, add something like this to the <LAYOUT> element:

```
<HYPERLINK HLTYPE="ANCHOR" TARGET="#somewhere" />
```

To link to this Anchor hyperlink, use something like this:

```
<RICHTEXT HLTYPE="ANCHOR"
HYPERLINKREF="#somewhere">link</RICHTEXT>
```

- You can use an Anchor hyperlink *without* creating it at the <PROJECT> level, but this is not the preferred method.

Page hyperlinks

To make sure a Page hyperlink works correctly, add something like this to the <LAYOUT> element:

```
<HYPERLINK HLTYPE="PAGE" NAME="Page 2" TARGET="2" />
```

To link to this Page hyperlink, use something like this:

```
<RICHTEXT HLTYPE="PAGE" HYPERLINKREF="Page 2"
>Page2</RICHTEXT>
```

- You can use a Page hyperlink *without* creating it at the <PROJECT> level, but this is not the preferred method.

Using the Streaming Document Provider

The Streaming Document Provider feature allows all of the assets required for a transaction to be provided as part of a multi-part HTTP request. Assets that can be streamed include:

- QuarkXPress templates.
- Picture files used in the template.

- Modifier XML.
- Picture and text files used in the Modifier XML.
- Assets used by digital publishing enrichments.

➡ The Streaming Document Provider feature also supports keepdocopen requests.

QuarkXPress Server searches for assets used in a call in the following order:

1. In the HTTP request.
2. At the supplied file path (if specified).
3. In the document pool.

If QuarkXPress Server does not find the required assets at any of these locations:

- If the image is being changed by the request, a “File not found” error occurs.
- If the image is not being changed by the request, it renders at preview resolution.

To use this feature, include a part in the HTTP request that has the same name as the asset to be streamed. For example:

```
<html>
 <body>
  <form enctype="multipart/form-data"
   action="http://localhost:8082/pdf/pic.qxp" method="post">
    <input type="file" name="picture.jpg"/><br/>
    <input type="file" name="pic.qxp"/><br/>
    <input type="hidden" name="modify"
value="<PROJECT><LAYOUT>
     <ID UID=111/><BOX><ID
NAME='picbox'/><CONTENT>picture.jpg
     </CONTENT></BOX></LAYOUT></ PROJECT>"/>
    <input type="submit"/>
  </form>
 </body>
</html>
```

Using administrative request handlers

Administrative request handlers let you change the behavior of QuarkXPress Server. The built-in administrative request handlers are described in the topics below

➡ You can add your own request handlers. During the DDSSETUPCBCODE callback, QuarkXPress Server XTensions software registers itself as a request handler via AddCustomRequestHandler, using the QuarkXPress

Server XTensions API. The first parameter of this API is a pointer to a request handler function implemented in QuarkXPress Server XTensions software. The second parameter is a namespace string that identifies the request. When a user submits a request that has the same namespace string as a suffix to the request URL, QuarkXPress Server calls the request handler function with all the user-specified parameters in the ServerRequest structure. The request handler function then processes the request and submits the reply in a ServerReply structure, which QuarkXPress Server communicates back to the user agent.

Addfile

Use the addfile request handler to put a document or image file in the document pool. An addfile request is always a POST request because it uses binary content.

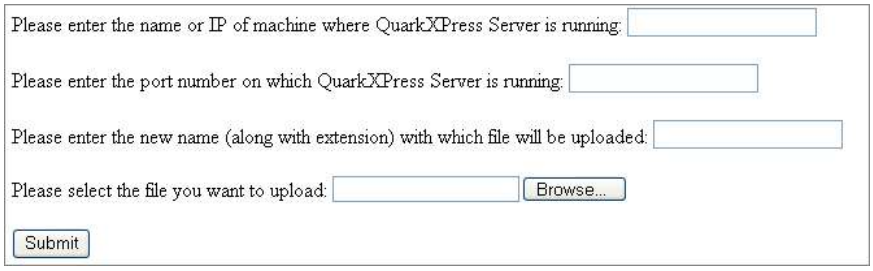
If you send an addfile request to QuarkXPress Server Manager using HTTP or the Web services interface while the common doc pool switch is set to off in the QuarkXPress Server Manager client, the file is uploaded to all registered QuarkXPress Server instances. If the common doc pool is enabled, the file can be uploaded to any one registered QuarkXPress server instance.

Namespace	addfile		
Parameters	uploadfile	Binary file or MIME-type file	Contains the actual binary content of the file to be uploaded. This can be a QuarkXPress file, a Word file, a text file, or a file with a MIME-type such as EPS, JPEG, PNG, or PICT.
Response	The message “File upload completed.”		
Alerts	The file system document pool is not enabled.	HTTP Error #404This alert displays if you attempt to upload a document when the file system document pool is not enabled. What to do: Check Enable File System Document Pool in the Server Configuration dialog box.	
	Incorrect	HTTP Error #401This alert displays if you	

	administration realm user name and password.	specify an invalid administrator user name and password.What to do: Use the user name and password set in theQuarkXPress Server Manager client Server Configuration dialog box.
	Cannot find required volume or folder	HTTP Error #500QuarkXPress Server Error #120This alert displays if you attempt to upload a document that is in a subfolder that does not exist in the document pool while Generate Hierarchy on Document Upload is unchecked in the Server Configuration dialog box. What to do: Check Generate Hierarchy on Document Upload in the Server Configuration dialog box.
Logs	See “Understanding logging."	
Example GET URL	To post a binary file in the root folder:http://localhost:8080/addfile/abc.qxpTo post a binary file in a subfolder:http://localhost:8080/addfile/sub1/abc.qxp	
Example, object model	Request object name: AddFileRequest // STEP 1 (COMMON FOR ALL REQUESTS): com.quark.qxpsm.QRequestContext rc = new com.quark.qxpsm.QRequestContext(); if(!this.DocumentSettings1.documentName.Text.Equals(“”)) rc.documentName = this.DocumentSettings1.documentName.Text; Stream theStream = uplTheFile.PostedFile.InputStream; long length = theStream.Length; Byte[] Buffer = new Byte[length]; const int BUFFER_SIZE = 10000; int nBytesRead = 0,iCount = 0; long remainingBytes = length — BUFFER_SIZE; if(remainingBytes > BUFFER_SIZE) { nBytesRead = theStream.Read(Buffer,iCount * BUFFER_SIZE,BUFFER_SIZE); while(0 != nBytesRead) { iCount++; remainingBytes = length — (iCount * BUFFER_SIZE); if(remainingBytes > BUFFER_SIZE) nBytesRead = theStream.Read(Buffer,iCount * BUFFER_SIZE,BUFFER_SIZE); else { nBytesRead = theStream.Read(Buffer,iCount *	

	BUFFER_SIZE,(int)remainingBytes); break; } } } else nBytesRead = theStream.Read(Buffer,iCount * BUFFER_SIZE,(int)remainingBytes); AddFileRequest addfilereq = new AddFileRequest(); addfilereq.fileData = Buffer; rc.request = addfilereq; // Create the service and call it with QRequestContext object RequestService svc = new RequestService(); QContentData qc = svc.processRequest(rc); The object model uses SOAP to transfer data, and SOAP encoding is not the most efficient way to transfer binary data. If you have to add a file using QuarkXPress Server Manager, the best way is to use a POST request in a QuarkXPress Server Manager URL. You might use QuarkXPress Manager to add a file if you wanted to add the file to all registered QuarkXPress Server instances at one time (assuming the instances are not sharing a single document pool).For more information, see the “AddFileRequest” .NET, Java, and Objective-C samples in the QuarkXPress Server Manager SDK samples.
Notes	The following is a sample of a POST request HTML form.<HTML> <HEAD><TITLE>Test Addfile</TITLE></HEAD> <BODY> File will always be uploaded with name new.qxp <FORM ACTION="http://localhost:8080/addfile/new.qxp" METHOD = “post” ENCTYPE="multipart/form-data"> Please select the file you want to upload: <INPUT TYPE=file NAME="uploadFile"> <INPUT TYPE=submit VALUE="Submit"> </FORM> </BODY> </HTML>

The following example demonstrates how to use an HTML form to create a POST request that uses the addfile request handler. The form looks like this:



To use this form:

1. Enter the name or IP address of the computer on which QuarkXPress Server is running.

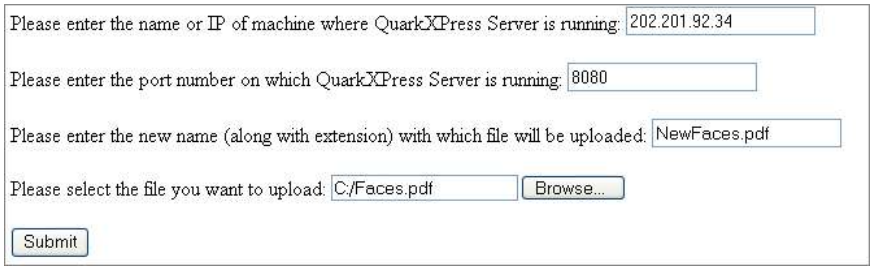
2 Enter the port number in the port number field.

3 Enter the file name along with the extension in the file field. Click **Browse** if you need to find the file on your computer. The file will be uploaded with this name.

4 Click **Submit**.

The file uploads to the document pool of the specified server. After the file is successfully uploaded, the “File upload completed." alert is displayed.

For example, assume you want to upload a file named “Faces.pdf" (located at the root of the C drive) to an instance of QuarkXPress Server running at IP address 202.201.92.34 and port 8080, and that you want the name of the uploaded file on the server to be “NewFaces.pdf." Here's how you would accomplish this in the HTML form:



The HTML code to generate the above sample file is as follows:

```
<HTML>
<HEAD>
 <TITLE>Test Addfile</TITLE>
 <SCRIPT LANGUAGE="JavaScript">
 function UploadDocument() {
  var URL;
  URL = "http://" + UploadForm.MachineIP.value + ":"  +
   UploadForm.Port.value + "/addfile/" +
UploadForm.NewName.value;
  UploadForm.action = URL;
 }
 </SCRIPT>
</HEAD>
<BODY>
 <FORM ID="UploadForm" METHOD = "post"
ENCTYPE="multipart/form-data"
   onSubmit="UploadDocument()">
 Please enter the name or IP of machine where QuarkXPress Server
is running:
  <INPUT TYPE="TextBox" NAME="MachineIP"><br><br>
 Please enter the port number on which QuarkXPress Server is
running:
  <INPUT TYPE="TextBox" NAME="Port"><br><br>
```

```
  Please enter the new name (along with extension) with which file
will be uploaded:
   <INPUT TYPE="TextBox" NAME="NewName"><br><br>
  Please select the file you want to upload: <INPUT TYPE=file
NAME="uploadFile">
  <br><br>
  <INPUT TYPE=submit VALUE="Submit">
</FORM>
</BODY>
</HTML>
```

The information entered in the form is created with the following tags:

```
<FORM ID="UploadForm" METHOD = "post"
  ENCTYPE="multipart/form-data" onSubmit="UploadDocument()">
  Please enter the name or IP of machine where QuarkXPress Server
is running:
  <INPUT TYPE="TextBox" NAME="MachineIP"><br><br>
  Please enter the port number on which QuarkXPress Server is
running:
  <INPUT TYPE="TextBox" NAME="Port"><br><br>
  Please enter the new name (along with extension) with which file
will be uploaded:
  <INPUT TYPE="TextBox" NAME="NewName"><br><br>
  Please select the file you want to upload:
  <INPUT TYPE=file NAME="uploadFile"><br><br>
  <INPUT TYPE=submit VALUE="Submit">
</FORM>
```

The FORM tag specifies that the method of the request is POST. This request is a "Multipart/form-data" request. When you submit the form, the UploadDocument() function is called.

Use the INPUT tag to create the text box and the **Browse** button.

- <INPUT TYPE="TextBox": To create text boxes only.

- <INPUT TYPE=file: To create a combination of text box and the **Browse** button in the form. When you click **Browse** and choose any file, the file path of the selected file displays in the text box linked with the **Browse** button.

You can use the INPUT tag to create the **Submit** button: <INPUT TYPE=submit VALUE="Submit">

When you click Submit, the UploadDocument() function is called. This function is defined inside a script tag. It combines the information that has been entered in the form to create a URL for the addfile request, then sends this URL to QuarkXPress Server for processing. The code for the UploadDocument() function is as follows:

```
<SCRIPT LANGUAGE="JavaScript">
 function UploadDocument() {
   var URL;
   URL = "http://" + UploadForm.MachineIP.value + ":"
    +  UploadForm.Port.value + "/addfile/" +
UploadForm.NewName.value;
   UploadForm.action = URL;
 }
 </SCRIPT>
```

Delete

The delete request handler removes a specified document or folder from the document pool.

If you send a delete request to QuarkXPress Server Manager using HTTP or the Web services interface while the common doc pool switch is set to off in the QuarkXPress Server Manager client, the file or folder is uploaded to all registered QuarkXPress Server instances. If the common doc pool is enabled, the file or folder can be deleted from any one registered QuarkXPress server instance.

Namespace	delete	
Response	The message “File deleted successfully."	
Alerts	File not found	HTTP Error #404 QuarkXPress Server Error #–43 This alert displays if you try to delete a file that does not exist in the document pool.
	Folder cannot be deleted. It may still contain files.	HTTP Error #405 This alert displays if you try to delete a folder that is not empty. What to do: First, delete all the files in the folder, and then resubmit the delete request to delete the folder.
	I/O error trying to read or write to disk.	HTTP Error #500 QuarkXPress Server Error #–36 This alert displays if you try to delete an open file.
	Incorrect administration realm user name and password.	HTTP Error #401 This alert displays if you specify an invalid administrator user name and password.What to do: Use the user name and password set in theQuarkXPress Server Manager client Server Configuration dialog box.
Logs	See “Understanding logging."	
Example	http://localhost:8080/delete/sample.qxp	

GET URL	
Example, object model	Request object name: DeleteRequestcom.quark.qxpsm.QRequestContext rc = new com.quark.qxpsm.QRequestContext(); if(!this.DocumentSettings1.documentName.Text.Equals("")) rc.documentName = this.DocumentSettings1.documentName.Text; rc.request = new DeleteRequest(); // Create the service and call it with QRequestContext object RequestService svc = new RequestService(); com.quark.qxpsm.QContentData qc = svc.processRequest(rc);

Evaluate

The evaluate request handler evaluates the document you specify using a rule set in the Job Jackets file you specify, and returns the results as an XML stream.

By default, this request handler evaluates the following rules:

- Platform mismatch
- Missing fonts
- Missing pictures

These rules are defined in the "Default Job Jacket.xml" file, which is generated by QuarkXPress Server in the preferences folder.

You can specify multiple rule sets in a comma-separated list.

To specify which layouts to evaluate, use the layout parameter.

To evaluate using an external Job Jackets file, use the jobjacket parameter. For example: jobjacket=customjj.xml

Namespace	evaluate	
Response	The default Job Jackets file.	
Alerts	Incorrect administration realm user name and password.	HTTP Error #401 This alert displays if you specify an invalid administrator user name and password. What to do: Use the user name and password set in the QuarkXPress Server Manager client Server Configuration dialog box.
Logs	See "Understanding logging."	

Example GET URL	http://localhost:8080/evaluate/MyProject.qxp?ruleset="MyRuleSet"	
Example, object model	Request object name: EvaluateRequestcom.quark.qxpsm.QRequestContext rc = new com.quark.qxpsm.QRequestContext(); if(!this.DocumentSettings1.documentName.Text.Equals("")) rc.documentName = this.DocumentSettings1.documentName.Text; rc.request = new EvaluateRequest(); //Create the service and call it with QRequestContext object RequestService svc = new RequestService(); com.quark.qxpsm.QContentData qc = svc.processRequest(rc);	
Notes	If a user name and password have been set in the Server Configuration dialog box, the browser requests that user name and password when you submit a getdocinfo parameter request.	

Exportprefsasjj

The exportprefsasjj request handler returns the default Job Jackets file as an XML stream. If you add ?download=true. the Job Jackets file is downloaded to the Web browser's default download location as an XML file.

Namespace	exportprefsasjj	
Response	The default Job Jackets file.	
Alerts	Incorrect administration realm user name and password.	HTTP Error #401 This alert displays if you specify an invalid administrator user name and password. What to do: Use the user name and password set in the QuarkXPress Server Manager client Server Configuration dialog box.
Logs	See “Understanding logging."	
Example GET URL	http://localhost:8080/exportprefsasjj	
Notes	If a user name and password have been set in the Server Configuration dialog box, the browser requests that user name and password when you submit a getdocinfo parameter request.	

Fileinfo

The fileinfo request handler returns XML that contains the creation date, modification date, and file size of a document.

Namespace	fileinfo	
Parameters	action=get	Lets you retrieve the creation date of a file in UTC format. For example:http://localhost:8080/fileinfo/sample.qxd?action=get&creationdate
	action=set	Lets you set the creation and modification dates of a file in UTC format. For example:http://localhost:8080/fileinfo/sample.qxp?action=set&creationdate=10-06-2007 12:12:37 UTC& modificationdate=10-06-2007 12:12:37 UTC
Response	The following XML code displays the creation date, modification date, and size of the document.<?xml version="1.0" encoding="UTF-8" ?> <FILEINFO> <CREATIONDATE>08-01-2004 06:14:07 UTC </CREATIONDATE> <MODIFICATIONDATE>08-01-2004 11:56:56 UTC </MODIFICATIONDATE> <SIZE>1519616</SIZE> </FILEINFO>	
Alerts	Incorrect administration realm user name and password.	HTTP Error #401 This alert displays if you specify an invalid administrator user name and password.What to do: Use the user name and password set in the QuarkXPress Server Manager client Server Configuration dialog box.
Logs	See “Understanding logging.”	
Example GET URL	http://localhost:8080/fileinfo/sample.qxp	
Example, object model	Request object name: FileInfoRequestcom.quark.qxpsm.QRequestContext rc = new com.quark.qxpsm.QRequestContext(); if(!this.DocumentSettings1.documentName.Text.Equals(“”)) rc.documentName = this.DocumentSettings1.documentName.Text; rc.request = new FileInfoRequest(); //Create the service and call it with QRequestContext object RequestService svc = new	

	RequestService(); com.quark.qxpsm.QContentData qc = svc.processRequest(rc);
Notes	If a user name and password have been set in the Server Configuration dialog box, the browser requests that user name and password when you submit a fileinfo parameter request.

Flush

The flush request handler flushes a document from the cache.

Namespace	flush	
Response	The message “CACHE FLUSH COMPLETED."	
Alerts	Incorrect administration realm user name and password.	HTTP Error #401 This alert displays if you specify an invalid administrator user name and password.What to do: Use the user name and password set in the QuarkXPress Server Manager client Server Configuration dialog box.
Logs	See “Understanding logging."	
Example GET URL	http://localhost:8080/flush/sample.qxp	
Example, object model	Request object name: FlushRequestsdk.QRequestContext rc = new sdk.QRequestContext(); if(!this.DocumentSettings1.documentName.Text.Equals(“”)) rc.documentName = this.DocumentSettings1.documentName.Text; rc.request = new FlushRequest(); //Create the service and call it with QRequestContext object RequestServiceService svc = new RequestServiceService(); com.quark.qxpsm.QContentData qc = svc.processRequest(rc);	
Notes	If a user name and password have been set in the Server Configuration dialog box, the browser requests that user name and password when you submit a flush parameter request.	

Flushall

Flushes all documents from the cache. When this request is sent to Server Manager using either HTTP or Web services, the cache of all registered QuarkXPress servers is flushed.

Namespace	flushall	
Response	The message “CACHE FLUSH COMPLETED."	
Alerts	Incorrect administration realm user name and password.	HTTP Error #401 This alert displays if you specify an invalid administrator user name and password. What to do: Use the user name and password set in the QuarkXPress Server Manager client Server Configuration dialog box.
Logs	See “Understanding logging."	
Example GET URL	http://localhost:8080/flushall	
Example, object model	Request object name: FlushAllRequestsdk.QRequestContext rc = new sdk.QRequestContext(); if(!this.DocumentSettings1.documentName.Text.Equals(“”)) rc.documentName = this.DocumentSettings1.documentName.Text; rc.request = new FlushAllRequest(); //Create the service and call it with QRequestContext object RequestServiceService svc = new RequestServiceService(); com.quark.qxpsm.QContentData qc = svc.processRequest(rc);	
Notes	If a user name and password have been set in the Server Configuration dialog box, the browser requests that user name and password when you submit a flushall parameter request.When you issue a flushall request, the memory usage value in the status monitor becomes zero.	

Getdocinfo

The getdocinfo request handler returns XML information about a QuarkXPress project that is in the document pool or has been supplied as part of a multipart HTTP request. The returned information includes the project version, the

platform on which it was saved, the number of layers, page properties, the length and width of the page in points, the number of pages, the names of imported picture files, the names of any required fonts, the names and IDs of any relevant XTensions modules, and (for documents saved in QuarkXPress 6.0 or later) information about synchronized content.

Namespace	getdocinfo	
Response	The XML response looks like the following:<? xml version="1.0" encoding="UTF-8" ?> <PROJINFO> <PLATFORM>WINDOWS</PLATFORM> <VERSION>7.0</VERSION> <NAME>Sample.qxp</NAME> <REQUIREDXTENSIONS /> <FONTUSAGE> <FONT> <NAME>ArialMT</NAME> </FONT> </FONTUSAGE> <LAYOUT> <NAME>Layout 1</NAME> <TYPE>Print</TYPE> <PAGES>4</PAGES> <PAGEPROPERTIES> <WIDTH>432</WIDTH> <LENGTH>756</LENGTH> </PAGEPROPERTIES> <LAYERS> <LAYER> <NAME>Default</NAME> </LAYER> </LAYERS> <HIRESGRAPHICS> <GRAPHICLINK> <FILEPATH>E:\pics\Jpeg\Autumn.jpg</FILEPATH> <USAGE PAGE="1" UNIQUEID="8" X="126.003" Y="116.967" /> </GRAPHICLINK> </HIRESGRAPHICS> </LAYOUT> </PROJINFO>	
Alerts	Incorrect administration realm user name and password.	HTTP Error #401 This alert displays if you specify an invalid administrator user name and password. What to do: Use the user name and password set in the QuarkXPress Server Manager client Server Configuration dialog box.
Logs	See “Understanding logging."	
Example GET URL	http://localhost:8080/getdocinfo/sample.qxp	
Example, object model	Request object name: GetDocInfoRequestsdk.QRequestContext rc = new sdk.QRequestContext(); if(!this.DocumentSettings1.documentName.Text.Equals(“”)) rc.documentName = this.DocumentSettings1.documentName.Text; rc.request = new GetDocInfoRequest(); //Create the service and call it with	

	QRequestContext object RequestServiceService svc = new RequestServiceService(); com.quark.qxpsm.QContentData qc = svc.processRequest(rc);
Notes	If a user name and password have been set in the Server Configuration dialog box, the browser requests that user name and password when you submit a getdocinfo parameter request.

Getdocpoollist

The getdocpoollist request handler returns an XML description of all files and folders in the local document pool, including name, size, type, modification date and time, and absolute and relative path.

Namespace	getdocpoollist	
Response	XML description of files and folders in the local document pool.	
Parameters	directory	Use this parameter to get information about a particular directory in the document pool. For example: http://server:port/getdocpoollist?directory=images
Alerts	Incorrect administration realm user name and password.	HTTP Error #401 This alert displays if you specify an invalid administrator user name and password. What to do: Use the user name and password set in the QuarkXPress Server Manager client Server Configuration dialog box.
Logs	See “Understanding logging.”	
Example GET URL	http://localhost:8080/getdocpoollist	
Example, object model	Request object name: GetDocPoolListRequest	
Notes	If a user name and password have been set in the Server Configuration dialog box, the browser requests that user name and password when you submit a fileinfo parameter request.	

Getlogs

The getlogs request handler returns the current preference settings for QuarkXPress Server in XML format. If you add ?download=true, the logs are returned in a .zip file.

Namespace	getlogs	
Response	The QuarkXPress Server transaction log.	
Alerts	Incorrect administration realm user name and password.	HTTP Error #401 This alert displays if you specify an invalid administrator user name and password. What to do: Use the user name and password set in the QuarkXPress Server Manager client Server Configuration dialog box.
Logs	See “Understanding logging."	
Example GET URL	http://localhost:8080/getlogs	
Notes	If a user name and password have been set in the Server Configuration dialog box, the browser requests that user name and password when you submit a getprefs parameter request.	

Getprefs

The getprefs request handler returns the current preference settings for QuarkXPress Server in XML format.

Namespace	getprefs	
Response	An XML description of QuarkXPress Server preference settings.	
Alerts	Incorrect administration realm user name and	HTTP Error #401 This alert displays if you specify an invalid administrator user name and password. What to do: Use the user name and password set in the QuarkXPress Server Manager client Server Configuration dialog box.

	password .
Logs	See “Understanding logging."
Example GET URL	http://localhost:8080/getprefs
Example, object model	// Create the service and call getPreferences method RequestService requestService = new RequestServiceStub(); Preferences preferences= requestService.getPreferences(“[host]", [port], “[username]", “[port]");
Notes	The getprefs request handler returns preference settings for server configuration and Status Monitor. It does not return other preference settings, such as the settings for Deconstruct and PDF workflow.If a user name and password have been set in the Server Configuration dialog box, the browser requests that user name and password when you submit a getprefs parameter request.

Getprocessid

The getprocessid request handler returns the process IDs of the master QuarkXPress Server instance and of all subrender processes running on the computer.

Namespace	getprocessid
Response	An XML description of the process IDs of the master QuarkXPress Server instance and of all subrender processes running on the computer. For example:<PROCESSID> <MASTER> <ID>3936</ID> <STATUS>BUSY</STATUS> </MASTER> <SUBRENDERERS> <SUBRENDERER> <ID>1736</ID> <STATUS>BUSY</STATUS> </SUBRENDERER> </SUBRENDERERS> </PROCESSID>
Example GET URL	http://localhost:8080/getprocessid

Getprojinfo

The getprojinfo request handler returns XML information about a QuarkXPress project that is in the document pool or has been supplied as part of a multipart HTTP request. The returned information identifies the operating system, the

version of QuarkXPress in which the project was created, the size of the project, the page properties for the project's layouts, and information about named boxes and synchronized text.

Namespace	getprojinfo	
Response	The XML response looks like the following:<? xml version="1.0" encoding="UTF-8" ?> <PROJINFO> <PLATFORM>WINDOWS</PLATFORM> <VERSION>6.0</VERSION> <NAME>Sample.qxp</NAME> <SIZE>1519616 Bytes</SIZE> <SYNCHRONIZED/> <LAYOUT> <NAME>Layout 1</NAME> <TYPE>Print</TYPE> <PAGES>4</PAGES> <PAGEPROPERTIES> <WIDTH>432</WIDTH> <LENGTH>756</LENGTH> </PAGEPROPERTIES> <NAMEDBOX> <BOX>box2</BOX> <BOX>box1</BOX> </NAMEDBOX> </LAYOUT> </PROJINFO>	
Alerts	The getprojinfo command can only be used for QuarkXPress 6 documents and later.	HTTP Error #500 This alert displays if you specify a QuarkXPress 4.0 or 5.0 document.
	Incorrect administration realm user name and password.	HTTP Error #401 This alert displays if you specify an invalid administrator user name and password. What to do: Use the user name and password set in the QuarkXPress Server Manager client Server Configuration dialog box.
Logs	See “Understanding logging."	
Example GET URL	http://localhost:8080/getprojinfo/sample.qxp	
Example, object model	Request object name: GetProjectInfoRequestsdk.QRequestContext rc = new sdk.QRequestContext(); if(!this.DocumentSettings1.documentName.Text.Equals(“”)) rc.documentName =	

	this.DocumentSettings1.documentName.Text; rc.request = new GetProjectInfoRequest(); //Create the service and call it with QRequestContext object RequestServiceService svc = new RequestServiceService(); QContentData qc = svc.processRequest(rc);
Notes	The getprojinfo parameter only works with projects saved in QuarkXPress 6.0 and later.If a user name and password have been set in the Server Configuration dialog box, the browser requests that user name and password when you submit a getprojinfo parameter request.

Getrendererprefs

The getrendererprefs request handler returns the current preference settings for QuarkXPress Server in XML format.

Namespace	getrendererprefs	
Response	An XML description of QuarkXPress Server renderer preference settings.	
Alerts	Incorrect administration realm user name and password.	HTTP Error #401 This alert displays if you specify an invalid administrator user name and password. What to do: Use the user name and password set in the QuarkXPress Server Manager client Server Configuration dialog box.
Logs	See “Understanding logging."	
Example GET URL	http://localhost:8080/getrendererprefs	
Notes	The schema for the returned XML stream can be found at webapps/ROOT/QuarkXPress Server Renderer/QXPSRendererPreferences.xsd.If a user name and password have been set in the Server Configuration dialog box, the browser requests that user name and password when you submit a getrendererprefs parameter request.	

Getserverinfo

The getserverinfo request handler returns XML information about QuarkXPress Server. The returned information includes the platform on which QuarkXPress Server is running, the version of QuarkXPress Server, a list of installed fonts and server XTensions modules, the relevant XTensions server XTensions module IDs, the startup parameters, and the output styles with which the server is running. Disabled server XTensions modules are not listed in the response.

Namespace	getserverinfo	
Response	XML containing information about this QuarkXPress server instance.	
Alerts	Incorrect administration realm user name and password.	HTTP Error #401 This alert displays if you specify an invalid administrator user name and password. What to do: Use the user name and password set in the QuarkXPress Server Manager client Server Configuration dialog box.
Logs	See “Understanding logging."	
Example GET URL	http://localhost:8080/getserverinfo	
Example, object model	Request object name: GetServerInfoRequestcom.quark.qxpsm.QRequestContext rc = new com.quark.qxpsm.QRequestContext(); if(!this.DocumentSettings1.documentName.Text.Equals(“”)) rc.documentName = this.DocumentSettings1.documentName.Text; rc.request = new GetServerInfoRequest(); //Create the service and call it with QRequestContext object RequestService svc = new RequestService(); com.quark.qxpsm.QContentData qc = svc.processRequest(rc);	
Notes	If a user name and password have been set in the Server Configuration dialog box, the browser requests that user name and password when you submit a getserverinfo parameter request.	

Jobjacket

The jobjacket request handler returns a Job Jackets file containing copies of all of the resources in the specified project. This is similar to what happens when you use jjname= in a modifier XML request, but it returns the Job Jackets file directly, rather than writing it to the document pool.

Namespace	jobjacket
Response	A Job Jackets file containing copies of all of the resources in the specified project.
Alerts	Incorrect administration realm user name and password. HTTP Error #401 This alert displays if you specify an invalid administrator user name and password. What to do: Use the user name and password set in the QuarkXPress Server Manager client Server Configuration dialog box.
Logs	See “Understanding logging."
Example GET URL	http://localhost:8080/jobjacket/myproject.qxp
Example, object model	Request object name: JobJacketRequestcom.quark.qxpsm.QRequestContext rc = new com.quark.qxpsm.QRequestContext(); if(!this.DocumentSettings1.documentName.Text.Equals(“”)) rc.documentName = this.DocumentSettings1.documentName.Text; rc.request = new JobJacketRequest(); //Create the service and call it with QRequestContext object RequestService svc = new RequestService(); com.quark.qxpsm.QContentData qc = svc.processRequest(rc);
Notes	If a user name and password have been set in the Server Configuration dialog box, the browser requests that user name and password when you submit a getrendererprefs parameter request.

Preflight

Use the preflight request handler to check a project for missing fonts and missing pictures prior to output. You can also use this request handler to determine if the platform on which a project was created is different from the platform on which QuarkXPress Server is running.

➡ The preflight request handler has been deprecated. Use the evaluate request handler instead. For more information, see "*Evaulate*."

Namespace	preflight	
Response	The XML response looks like the following:<?xml version="1.0" encoding="UTF-8" standalone="no" ?> <PREFLIGHT> <PLATFORMMISMATCH>TRUE</PLATFORMMISMATCH> <MISSINGFONT>MidashiGoPro-MB31</MISSINGFONT> <MISSINGPICTURE>/QuarkXPress Server Documents/images/illus_eps.eps </MISSINGPICTURE> </PREFLIGHT>	
Alerts	File not found	HTTP Error #404 QuarkXPress Server Error #–43 This alert displays if you try to delete a file that is not available to QuarkXPress Server.
Logs	See "Understanding logging."	
Example GET URL	To preflight a project in the root folder:http://localhost:8080/preflight/abc.qxpTo preflight a binary file in a subfolder:http://localhost:8080/preflight/sub1/abc.qxp	
Example, object model	Request object name: PreflightRequest com.quark.qxpsm.QRequestContext rc = new com.quark.qxpsm.QRequestContext(); if(!this.DocumentSettings1.documentName.Text.Equals("")) rc.documentName = this.DocumentSettings1.documentName.Text; rc.request = new PreflightRequest(); //Create the service and call it with QRequestContext object RequestService svc = new RequestService(); com.quark.qxpsm.QContentData qc = svc.processRequest(rc);	

Setprefs

The setprefs request handler lets you set server preferences. To use this request handler, issue a getprefs request, determine the name of the tag that needs to be modified, and then submit a setprefs request with the using the name of this tag. For example, to turn off memory caching, you would first submit a getprefs request to the server. In the resulting XML, you would note that the name of the tag for memory caching tag is AllowMemoryCaching. Finally, you would submit a setprefs request to the server, like so:

http://localhost:8080/setprefs?AllowMemoryCaching=false

➡ For a full list of preferences, see "*General preferences*" and "*Renderer preferences*."

Namespace	setprefs	
Response	The message "Preferences successfully set."	
Alerts	Incorrect administration realm user name and password.	HTTP Error #401 This alert displays if you specify an invalid administrator user name and password. What to do: Use the user name and password set in the QuarkXPress Server Manager client Server Configuration dialog box.
Logs	See "Understanding logging."	
Example GET URL	http://localhost:8080/setprefs?CacheSize=200	
Example, object model	EMailPreferences emailPrefs = new EMailPreferences(); emailPrefs.emailFrom = "from@email.com"; emailPrefs.emailTo = "to@email.com"; emailPrefs.smtpPort = "25"; emailPrefs.smtpServerIP = "SMTPServerIP"; Preferences prefs = new Preferences(); prefs.setEmailPreferences(emailPrefs); // Create the service and call setPreferences method RequestService svc = new RequestService(); svc.setPreferences("ServerName", "ServerPort", "Username", "Password", prefs);	
Notes	If a user name and password have been set in the Server Configuration dialog box, the browser requests that user name and password when you submit a setprefs parameter request.	

General preferences

The /getprefs handler lets clients remotely retrieve an XML representation of the current QuarkXPress Server preferences, as described below.

The /setprefs handler lets clients remotely control QuarkXPress Server preferences. For example, http://<servername>:<port>/setprefs?AddConnectionFilter=action=allow;ipaddr=206.195.80.80;mask=255.255.255.1;pos=0 permits connection filters and specifies the IP address and the subnet mask of the connection.

Attribute	Type	Description
CacheSize	integer	Specifies the memory cache size from 1MB to 1024MB.
DocumentRootFolder	string	Specifies the document root directory.
LogTiming	Boolean	Specifies whether to include timing information (such as processing time and opening time) in the transaction log.
ForceServedDocumentsClosed	Boolean	Causes QuarkXPress Server to close projects that are loaded into cache from the document pool after rendering them.
AllowMemoryCaching	Boolean	Specifies whether to store disk-based projects in a memory-resident cache.
DefaultRenderType	string	Sets the default. Valid values include PNG, PDF, EPS, PSCR, QXPD, RAW, RLER, and JPEG.
LogDocProblems	Boolean	Specifies whether to log any project problems.

Setrendererprefs

The setrendererprefs request handler lets you set rendering preferences. To use this request handler, issue a getrendererprefs request, determine the name of the tag that needs to be modified, and then submit a setrendererprefs request with the using the name of this tag. For example, set color TIFF and gray TIFF display preferences, you would first submit a getrendererprefs request to the server, then update these settings and submit a setrendererprefs request to the server, like so:

```
http://server:port/setrendererprefs?modify=<QXPSRENDERERPREFERENCES>
<DISPLAY><ColorTIFFs>8-bit</ColorTIFFs><GrayTIFFs>16 levels</GrayTIFFs></DISPLAY>
</QXPSRENDERERPREFERENCES>
```

➡ This request handler sets the preferences for *all* renderers.

Namespace	setrendererprefs	
Response	The message “Preferences successfully set."	
Alerts	Incorrect administration realm user name and password.	HTTP Error #401 This alert displays if you specify an invalid administrator user name and password. What to do: Use the user name and password set in the QuarkXPress Server Manager client Server Configuration dialog box.
Logs	See “Understanding logging."	
Example GET URL	http://localhost:8080/setrendererprefs?modify=<QXPSRENDERERPREFERENCES><DISPLAY><ColorTIFFs>8-bit</ColorTIFFs> <GrayTIFFs>16 levels</GrayTIFFs></DISPLAY></QXPSRENDERERPREFERENCES>	
Notes	The schema for the returned XML stream can be found at webapps/ROOT/QuarkXPress Server Renderer/QXPSRendererPreferences.xsd.If a user name and password have been set in the Server Configuration dialog box, the browser requests that user name and password when you submit a setrendererprefs parameter request.	

Renderer preferences

The /getrendererprefs handler lets clients remotely retrieve an XML representation of the current QuarkXPress Server renderer preferences, as described below. For a detailed description of each preference, see “General Preferences dialog box" in *A Guide to QuarkXPress Server*.

The /setrendererprefs handler lets clients remotely control QuarkXPress Server renderer preferences. For more information, see “*Setrendererprefs*."

Attribute	Type	Description
DISPLAY		
ColorTIFFs	string	Set to 8-bit or 24-bit.
GrayTIFFs	string	Set to 16 levels or 256 levels.

Attribute	Type	Description
MonitorProfile	string	Set to one of the options under MonitorProfileList.
MonitorProfileList	collection	Includes one <MonitorProfile> element for each available monitor profile.
INPUTSETTINGS		
SmartQuotes	Boolean	Set to true or false.
SmartQuoteFormat	integer	Set to the zero-based index of the desired format. For the list of available formats, see the Input Settings pane of the Preferences dialog box in the QuarkXPress Server administrative interface.
SequentialPageRangeSeparator	string	Set to one character only.
NonSequentialPageRangeSeparator	string	Set to one character only.
FONTFALLBACK		
ApplyFontFallback	Boolean	Set to true or false.
Search	Boolean	Set to true or false.
SearchType	Boolean	Set to Paragraph or Active Story.
SearchLimit	integer	Set to search limit number.
PreferredFont	string	Includes a <PreferredFont> element with each of the following attributes:ScriptOrLanguage="Cyrillic" ScriptOrLanguage="Greek" ScriptOrLanguage="Latin" ScriptOrLanguage="Japanese" ScriptOrLanguage="Korean" ScriptOrLanguage="Simplified Chinese" ScriptOrLanguage="Traditional Chinese"

Attribute	Type	Description
SlugLineFont	string	Set to the name of the slug line font.
OPENANDSAVE		
OpenSaveNonUnicodeEncoding	string	Set to Roman, Central European, Greek, Cyrillic, Turkish, Japanese, Korean, Simplified Chinese, or Traditional Chinese.
FONTS		
SpecifyDefaultFontReplacement	Boolean	Set to true or false.
ReplacementFontRoman	string	Set to the Roman replacement font.
ReplacementFontEastAsian	string	Set to the East Asian replacement font.
HighlightCharacterChanges	Boolean	Set to true or false.
EPS		
EPSPreview	string	Set to Embedded or Generate.
PDF		
PDFWorkflow	string	Set to PS4D, PDFtoFolder, or DirectPDF.
WatchedFolder	string	Set to the path of the watched folder.
DEFAULTPRINTLAYOUGENERAL		
MasterPageItems	string	Set to Keep Changes or Delete Changes.
Framing	string	Set to inside or outside.
AutoPageInsertionMode	string	Set to Off, End of Story, End of Section, or End of Document.
DEFAULTPRINTLAYOUTMEASUREMENTS		
HorizontalUnits	string	Set to Inches, Inches Decimal, Picas, Points, Millimeters, Centimeters, Ciceros, Agates, or Q.
VerticalUnits	string	Set to Inches, Inches Decimal, Picas, Points, Millimeters, Centimeters, Ciceros, Agates, or Q.
PointsPerInch	float	Set to the number of points per inch.
CicerosPerCM	float	Set to the number of ciceros per centimeter.
PARAGRAPH		

Attribute	Type	Description
AutoLeading	float	Set to the percentage to use for auto leading.
MaintainLeading	Boolean	Set to true or false.
LockToGridOption	string	Set to Ascent and Descent or Font Size (Em Box).
CHARACTER		
SuperScriptOffset	string	Set to the superscript offset percentage.
SuperScriptHScale	string	Set to the superscript horizontal scale percentage.
SuperScriptVScale	string	Set to the superscript vertical scale percentage.
SubScriptOffset	string	Set to the subscript offset percentage.
SubScriptHScale	string	Set to the subscript horizontal scale percentage.
SubScriptVScale	string	Set to the subscript vertical scale percentage.
SmallCapsHScale	string	Set to the small caps horizontal scale percentage.
SmallCapsVScale	string	Set to the small caps vertical scale percentage.
SuperiorHScale	string	Set to the superior horizontal scale percentage.
SuperiorVScale	string	Set to the superior vertical scale percentage.
LigatureBreakAbove	string	Set to the ligature-break-above value.
NotffiORffl	Boolean	Set to true or false.
AutoKern	Boolean	Set to true or false.
AutoKernAbove	integer	Set to the auto-kern-above value.
StandardEmSpace	Boolean	Set to true or false.

Attribute	Type	Description
FlexSpaceWidth	string	Set to the flex space width percentage.
AccentsForAllCaps	Boolean	Set to true or false.
SpaceCJKandR	float	Set to the amount of space between East Asian and Roman letters.
TRAPPING		
TrappingMethod	string	Set to Absolute, Proportional, or Knockout All.
ProcessTrapping	Boolean	Set to true or false.
IgnoreWhite	Boolean	Set to true or false.
AutoAmount	float	Set to the auto trapping amount value.
Indeterminate	float	Set to the indeterminate trapping value.
KnockoutLimit	float	Set to the knockout limit value.
OverPrintLimit	float	Set to the overprint limit value.
COLORMANAGER		
ColorEngine	string	Set to Automatic, ColorSync, Kodak, or LogoSync.
BlackPointCompensation	Boolean	Set to true or false.
SourceSetup	string	Set to the name of the default color source setup.
EnableAccessToPictureProfiles	Boolean	Set to true or false.
ProofOutPut	string	Set to the name of the default proofing color output setup.
RenderingIntent	string	Set to Perceptual, Relative Colorimentric, Saturation, Absolute Colorimetric, or Defined by Sources.
ColorManageVectorEPSPDF	Boolean	Set to true or false.
IncludeExistingVectorEPSPDF	Boolean	Set to true or false.

Attribute	Type	Description
LAYERS		
Visible	Boolean	Set to true or false.
SuppressOutput	Boolean	Set to true or false.
Locked	Boolean	Set to true or false.
KeepRunaround	Boolean	Set to true or false.
FULLRESPREVIEW		
MaxCacheFolderSize	integer	Set to the maximum full res preview cache folder size in MB.
DisableFullResPreviewsOnOpen	Boolean	Set to true or false.
INSTALLEDFONTS		
FontName	string	Include a <FontName> element for each font loaded on the server.

updateprefsfromjj

The updateprefsfromjj request handler lets you update the rendering preferences for QuarkXPress Server using a file named "QuarkXPressServerJobJacket.xml" that is in the document pool or has been supplied as part of a multipart HTTP request.

To modify QuarkXPress Server renderer preferences, first upload a modified Job Jackets file named "QuarkXPressServerJobJacket.xml" to the document pool using addfile, then call this request handler.

➡ This request handler sets the preferences for *all* renderers.

Namespace	updateprefsfromjj	
Response	The message "Preferences successfully set."	
Alerts	Incorrect administration realm	HTTP Error #401 This alert displays if you specify an invalid administrator user name and password. What to do: Use the user name and password set in the QuarkXPress Server Manager client Server

	user name and password .	Configuration dialog box.
Logs	See “Understanding logging."	
Example GET URL	http://localhost:8080/updateprefsfromjj	
Notes	If a user name and password have been set in the Server Configuration dialog box, the browser requests that user name and password when you submit a setrendererprefs parameter request.	

Using the QXPSM SDK

The QXPSM (QuarkXPress Server Manager) SDK lets you create applications that communicate with QuarkXPress Server Manager in a variety of languages, including the following:

- .NET
- Java
- Objective-C

The QXPSM SDK includes the following folders:

- Documentation: Includes Javadoc for the classes in the Java SDK.
- Extensibility: Includes the Extensibility tool, for extending QuarkXPress Server Manager. For more information, see “*Extending QuarkXPress Manager*."
- Samples: Includes sample applications in ASP.NET, C#, Java, JSP, and Objective-C.
- WebServiceStubs: Includes remoting stubs for .NET (C#), Java, and Objective-C.

➡ To use the QXPSM SDK in ASP.NET/Visual C#, you must have the .NET 3.5/4.0 framework and development environment (Visual Studio).

Writing a Java QXPSM client

To write a QuarkXPress Server Manager client in Java:

1. Include the QXPSM stub jar file in the project classpath. This jar file can be found at the following location:
   [QXPSM_Home]/XDK/WebServiceStubs/java/qxpsm-webservicestubs.jar
2. Include all client-side third-party-dependent jar files in the project classpath. These can be found at the following location:
   [QXPSM_Home]/XDK/WebServiceStubs/java/dependencies
3. Get a reference to the RequestService:

```
RequestService requestService = new RequestServiceStub(
  "http://[server]:[port]/qxpsm/services/RequestService");
```

4. Get a reference to the AdminService:

```
AdminService adminService = new AdminServiceStub(
"http://[server]:[port]/qxpsm/services/AdminService");
```

5. Use these two services to make requests.

For sample code, see the following topics.

➡ If QuarkXPress Server Manager is running over SSL,the client-side application must also use SSL. Invoke NoValidationTrustProvider.install(), where install() is the static method of the Java class NoValidationTrustProvider (provided with the Java samples).

Java sample: Deconstructing a project

```
QRequestContext qRequestContext = new QRequestContext();
qRequestContext.setDocumentName("MyDoc.qxp");

// Create XML Request
XMLRequest xmlRequest = new XMLRequest();
qRequestContext.setRequest(xmlRequest);

// Get reference to RequestService
RequestService service = new RequestServiceStub(
  "http://<server>:<port>/qxpsm/services/RequestService");

//Process Request using request service
QContentData data = service.processRequest(qRequestContext);
String deconstructXml = data.getTextData();
```

Java sample: Rendering a PDF

```
QRequestContext qRequestContext = new QRequestContext();
qRequestContext.setDocumentName("MyDoc.qxp");

// Setting responseAsURL to true generates the response as a URL
qRequestContext.setResponseAsURL(true);
```

```
// Create the PDFRenderRequest
PDFRenderRequest pdfRenderRequest = new PDFRenderRequest();
qRequestContext.setRequest(pdfRenderRequest);

// Get reference to RequestService
RequestService service = new RequestServiceStub(
  "http://<server>:<port>/qxpsm/services/RequestService");

// Process request using RequestService
QContentData data = service.processRequest(qRequestContext);

// Get URL from which resulting PDF can be fetched
String pdfUrl = data.getResponseURL();
```

Java sample: Chained request

```
QRequestContext qRequestContext = new QRequestContext();
qRequestContext.setDocumentName("Project.qxp");

// QXP doc render request
QuarkXPressRenderRequest qxpreq = new
QuarkXPressRenderRequest();

// Save as request that saves the file
SaveAsRequest saveAsRequest= new SaveAsRequest();
saveAsRequest.setNewName("NewDoc.qxp");

qxpreq.setRequest(saveAsRequest);

qRequestContext.setRequest(qxpreq);

// Get reference to RequestService
RequestService service = new RequestServiceStub(
  "http://[server]:[port]/qxpsm/services/RequestService");

//Process the request
service.processRequest(qRequestContext);
```

If made from QuarkXPress Server, this request would look like this:

```
http://[server]:[port]/saveas/qxpdoc/Project.qxp?newname="NewDoc
.qxp"
```

Java sample: AddFile request

This code snippet shows how to upload a file using a QuarkXPess Server Manager servlet request that uses the Apache HTTPClient 3.1 library.

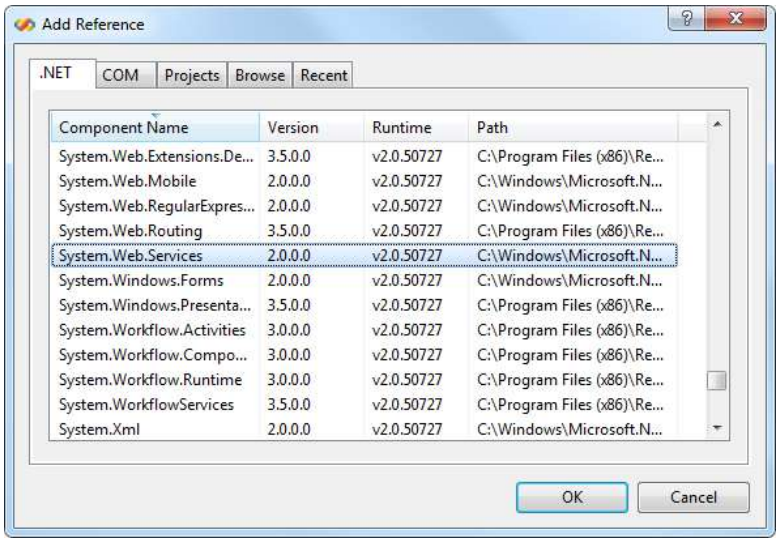
```
File file = new File("C:/FileToUpload.qxp");
```

```
// Create a post method to add file through QXPSM servlet request
PostMethod method = new PostMethod(
  "http://[server]:[port]/qxpsm/request/addfile/“+ file.getName());
try {

  // File stream passed as 'uploadFile' parameter to QXPSM
  FilePart part = new FilePart("uploadFile", file);
  part.setContentType("multipart/form-data");
  Part[] parts = { part };
  method.setRequestEntity(new MultipartRequestEntity(parts,
method.getParams()));

  HttpClient client = new HttpClient();
  int status = client.executeMethod(method);

  if (status == HttpStatus.SC_OK) {
    System.err.println("Upload complete..");
  } else {
    System.err.println("Upload failed, response=" +
HttpStatus.getStatusText(status));
  }
} catch (Exception ex) {
  System.err.println("Error: " + ex.getMessage());
} finally {
  method.releaseConnection();
}
```

Writing a .NET QXPSM client

To write a QuarkXPress Server Manager client in .NET:

1. Add a reference to the QuarkXPress Server Mananger .NET Web service stubs. The “QXPSMWebServiceStubs.dll” file can be found at the following location:
   [QXPSM_Home]/XDK/WebServiceStubs/dotnet/QXPSMWebServiceStubs.dll
2. Add a reference to Microsoft's System.Web.Services library.



Add Reference dialog box

3 Get a reference to the RequestService:

```
RequestService requestService = new RequestService();
requestService.Url =
"http://<server>:<port>/qxpsm/services/RequestService";
```

4 Get a reference to the AdminService:

```
AdminService adminService = new AdminService();
adminService.Url =
"http://<server>:<port>/qxpsm/services/AdminService";
```

5 Use these two services to make requests.

For sample code, see the following topics.

➡ If QuarkXPress Server Manager is running over SSL,the client-side application must also use SSL. Define a server certificate validation callback during application initialization using code like the following:

```
Imports System.Net
Imports System.Net.Security
Imports System.Security.Cryptography.X509Certificates
...
ServicePointManager.ServerCertificateValidationCallback +=
delegate(
  object senders,
  X509Certificate certificate,
  X509Chain chain,
  SslPolicyErrors sslPolicyErrors)
{
  return true;
}
```

.NET sample: Deconstructing a project

```
QRequestContext qRequestContext = new QRequestContext();
qRequestContext.documentName = "MyDoc.qxp";

// Create XML Request
XMLRequest xmlRequest = new XMLRequest();
qRequestContext.request = xmlRequest;

// Get reference to RequestService
RequestService service = new RequestService();
service.Url = "http://[server]:[port]/qxpsm/services/RequestService";

// Process request using RequestService
QContentData data = service.processRequest(qRequestContext);
String deconstructXml = data.textData;
```

.NET sample: Rendering a PDF

```
QRequestContext qRequestContext = new QRequestContext();
qRequestContext.documentName = "MyDoc.qxp";

// Setting responseAsURL to true generates the response as a URL
qRequestContext.responseAsURL = true;

// Create the PDFRenderRequest
PDFRenderRequest pdfRenderRequest = new PDFRenderRequest();
qRequestContext.request = pdfRenderRequest;

// Get reference to RequestService
RequestService service = new RequestService();
service.Url = "http://[server]:[port]/qxpsm/services/RequestService";

// Process request using RequestService
QContentData data = service.processRequest(qRequestContext);

// URL from which resulting PDF can be fetched
String pdfUrl = data.responseURL;
```

.NET sample: Chained request

```
QRequestContext qRequestContext = new QRequestContext();
qRequestContext.documentName = "Project.qxp";

// QXP doc render request
QuarkXPressRenderRequest qxpreq = new
QuarkXPressRenderRequest();
```

```
// Save as request that saves the file
SaveAsRequest saveAsRequest= new SaveAsRequest();
saveAsRequest.newName = "NewDoc.qxp";

qxpreq.request = saveAsRequest;
qRequestContext.request = qxpreq;

// Get reference to RequestService
RequestService service = new RequestService();
service.Url = "http://[server]:[port]/qxpsm/services/RequestService";

// Process the request
service.processRequest(qRequestContext);
```

If made from QuarkXPress Server, this request would look like this:

```
http://[server]:[port]/saveas/qxpdoc/Project.qxp?newname="NewDoc
.qxp"
```

Writing an Objective-C client for macOS or iOS

To write a QuarkXPress Server Manager client in objective C for macOS or iOS:

1. Include the QuarkXPress Server Manager stub header files path in the header search paths. These files can be found at the following location: [QXPSM_Home]/XDK/WebServiceStubs/objective-c/include
2. Include the Axis2c header files path in the header search paths. These header files can be found at the following location:
   - macOS: [QXPSM_Home]/XDK/WebServiceStubs/objective-c/lib/i86_64/axis2c/include
   - iOS: [QXPSM_Home]/XDK/WebServiceStubs/objective-c/lib/arm/axis2c/include
3. Include the QuarkXPress Server Manager stub libraries in the library search paths. These libraries can be found at the following locations:

macOS:

```
[QXPSM_Home]/XDK/WebServiceStubs/objective-c/lib/i86_64/
  c/libQXPSMSoapCBindings.a
  cpp/libQXPSMSoapCppBindings.a
  objc/libQXPSMSoapObjCBindings.a
```

iOS:

```
[QXPSM_Home]/XDK/WebServiceStubs/objective-c/lib/i86_64/
  c/libQXPSMSoapCBindings.a
  cpp/libQXPSMSoapCppBindings.a
  objc/libQXPSMSoapObjCBindings.a
```

4 Create an instance of QXPSMServiceManager. (QXPSMServiceManager is an Axis2c-based factory for QXPSM Web services stubs. It maintains shared instances of stubs corresponding to different QXPSM Web services.)

Initialise QXPSMServiceManager using setupForHost, with the required parameters:

- Host: Host name of QXPSM server.
- port: Web port configured at the server for HTTP (or HTTPS) communication.
- logFilePath: Location where Axis2 can generate a log file.
- logLevel: Number specifying log level for Web service communication (0 = critical, 1 = error, 2 = warning, 3 = info, 4 = debug, 5 = user-level debug message, 6 = trace).
- axis2Home: Client side Axis2 home folder location.
- useHttps: Set to true for secure API communication with QXPSM server
- serverCertificatePath: Certificate file path . For a non-secured connection, this argument is ignored.

For example:

```
[[QXPSMServiceManager sharedInstance]
  setupForHost:server port:port
  logFilePath:axisLogFile logLevel:axisLogLevel
axis2Path:axis2Home
  useHttps:usehttps serverCertificatePath:certFilePath];
```

5 Initialise QXPSMServiceManager using setupForHostWithProxy, with the required parameters:

- Host: Host name of QXPSM server.
- port: Web port configured at server for HTTP (or HTTPS) communication.
- logFilePath: Location where Axis2 can generate a log file.
- logLevel: Number specifying log level for Web service communication.
- axis2Home: Client side Axis2 home folder location.
- useHttps: Set to true for secure API communication with QXPSM server.
- serverCertificatePath: Certificate file path. For a non-secured connection, this argument is ignored.
- proxyHost: Host name of HTTP proxy. If null, proxy is not used.
- proxyPort: HTTP proxy port number.

- username: User name for proxy authentication. If null, no authentication is performed.
- password: Password for HTTP proxy authentication.

For example:

```
[[QXPSMServiceManager sharedInstance]
  setupForHost:server port:port
  logFilePath:axisLogFile logLevel:axisLogLevel
axis2Path:axis2Home
  useHttps:usehttps serverCertificatePath:certFilePath
  proxyHost:proxyHost proxyPort:proxyPort
  username:proxyUserName password:proxyPassword];
```

6 Get a reference to the RequestService using QXPSMServiceManager and perform the required API invocations on the referenced RequestService. For example:

```
[[QXPSMServiceManager requestService]
  getXPressDomForDocumentName:documentName];
```

7 To invoke a QXPSM service, get a reference to the AdminService using QXPSMServiceManager and perform the required API invocations on the referenced AdminService. For example:

```
[[QXPSMServiceManager requestService]
  getXPressDomForDocumentName:documentName];
```

For sample code, see the following topics.

Objective-C sample: Deconstructing a project

```
QXPSMQRequestContext *qRequestContext =
  [[[QXPSMQRequestContext alloc] init] autorelease];
[qRequestContext setDocumentName:@"MyDoc.qxp"];

// Create XML Request
QXPSMXMLRequest *xmlRequest = [[[QXPSMXMLRequest alloc] init]
autorelease];
[qRequestContext setRequest:xmlRequest];

// Get reference to RequestService and process request
QXPSMQContentData *data = [[QXPSMServiceManager
requestService]
  processRequestForRequestCmd:qRequestContext];
NSString *deconstructXml = [data getTextData];
```

Objective-C sample: Rendering a PDF

```
QXPSMQRequestContext *qRequestContext =
```

```
  [[[QXPSMQRequestContext alloc] init] autorelease];
[qRequestContext setDocumentName:@"MyDoc.qxp"];

// Setting responseAsURL to true generates the response as a URL
[qRequestContext setResponseAsURL:YES];

// Create the PDFRenderRequest
QXPSMPDFRenderRequest *pdfRenderRequest =
  [[[QXPSMPDFRenderRequest alloc] init] autorelease];
[qRequestContext setRequest:pdfRenderRequest];

// Get reference to RequestService and process request
QXPSMQContentData *data = [[QXPSMServiceManager
requestService]
  processRequestForRequestCmd:qRequestContext];

// URL from which resulting PDF can be fetched
NSString *pdfUrl = [data getResponseURL];
```

Objective-C sample: Chained request

```
QXPSMQRequestContext *qRequestContext =
  [[[QXPSMQRequestContext alloc] init] autorelease];
[qRequestContext setDocumentName:@"MyDoc.qxp"];

//QXP doc render request
QXPSMQuarkXPressRenderRequest *qxpReq =
  [[[QXPSMQuarkXPressRenderRequest alloc] init] autorelease];

//Save as request that saves the file.
QXPSMSaveAsRequest *saveAsRequest = [[[QXPSMSaveAsRequest
alloc] init] autorelease];
[saveAsRequest setNewname:@"NewDoc.qxp"];

[qxpReq setRequest:saveAsRequest];
[qRequestContext setRequest:qxpReq];

//Get reference to RequestService and process request
[[QXPSMServiceManager requestService]
processRequestForRequestCmd:qRequestContext];
```

If made from QuarkXPress Server, this request would look like this:

```
http://[server>]:[port]/saveas/qxpdoc/MyDoc.qxp?newname="NewDo
c.qxp"
```

Extending QuarkXPress Server Manager

Custom XTensions written for XPressServer can be used in the QuarkXPress Server Manager Web service interface in two ways:

- Using the Extensibility tool in the QXPSM SDK. With this tool, you can easily update QXPSM Web service objects to include objects corresponding to custom request handlers and their parameters.
- Using the RequestParameters class. This is a generic request class that can be used in lieu of any class, existing or otherwise.

The prerequisites for using the Extensibility tool are as follows:

- JDK 1.6
- Apache ANT 1.6.5 or later
- Perl 5.8.4 or later with the XML::DOM module
- Third-party libraries (available at [QXPSM application folder]\Server\dependencies)
- QXPSM libraries (avaiable at [QXPSM application folder]\Server\lib
- Microsoft .NET Framework 3.5 or later (required only for generating .NET stubs)

The Extensibility tool is located in the XDK/Extensibility folder. For instructions on how to use it, see the following topics.

Writing special request handlers

If you need to perform custom actions on specific flags, you need to define special flags and write handlers for them. These flags can then be passed as GET parameters to the servlet, as additional QParam parameters in QCommand (executed using QManagerSvc.executeCommand), or as additional NameValueParam parameters in a derived class of QRequest using RequestService.processRequest. The servlet will automatically create parameters out of these flags and set these in the command before sending it for execution.

To handle these special flags, you can write your request handler derived from the class QRequestHandler. You can then insert this new handler class anywhere in the chain of responsibility pattern, starting with QDocProviderImpl and ending with QHostRequestHandler.

➡ Try not to change end points. In your handler implementation, handle your special flags, then either return a response after handling or pass the control to the successor for further handling.

Implementing a custom load balancer

To implement a custom load balancer, first implement the com.quark.manager.lb.QLoadBalancer interface. To use this interface, add a reference to "managerengine.jar" to your project.

This interface method contains the following methods:

getLoadBalancerAlgorithm	
Signature	public String getLoadBalancerAlgorithm();
Description	Returns the name of the algorithm that is mapped to the current load balancer while loading the server.
Returns	The algorithm name used to load-balance the list of hosts.
getLoadBalancerDescription	
Signature	public String getLoadBalancerDescription();
Description	Gets the description of the load-balancing algorithm so it can be displayed in the QuarkXPress Server Manager client.
Returns	Description of the load balancer.
useFileInfo	
Signature	public Boolean useFileInfo();
Description	Gets a flag that indicates whether the load balancer uses file information to decide on which host to use.
Returns	True if the fileinfo command should be fired before rendering, otherwise false.
getAvailableHost	
Signature	public QHostProxy getAvailableHost(QHostProxy[] hosts, QCommand command);
Description	Gets an available host out of the provided list of hosts to execute the specified command.
Parameters	hosts: List of hosts that should be scanned for the most eligible host.command: Command for which host is being searched.
Returns	Available host. Can be used for next request.

Next:

1. Make a jar for the load balancer.
2. Deploy the jar to the following folder: [QXPSM_HOME]/dependencies

3. To configure “ManagerContainerConfig.xml” for bean mapping, first navigate to [QXPSM_HOME]/conf .
4. Open the “ManagerContainerConfig.xml” file and look for the XML tag bean whose id has the value ConfigurationManager.
5. Within that tag find the property name availableLoadBalancers.
6. In the <list> tag, add the following: <ref bean=[your newbeanID]/>
7. Above this ConfigureManager tag, define the bean ID as your new bean ID: <bean id=[your newbeanID] class=[yourLoadBalancerClass]/>
8. Restart the Tomcat server.
9. Log on with the QuarkXPress Server Manager client and choose **Global Setting > Load Balancer Method > Choose Load Balancer**.
10. Locate your new load balancer method, then click **Save**.

Keep document open (sessions)

In early versions of QuarkXPress Server Manager, the software opened a QuarkXPress project, performed a function, and then closed the project. To avoid the delays involved in repeatedly opening and closing a QuarkXPress project, QuarkXPress Server Manager can now keep QuarkXPress projects open until they need to be closed.

To keep projects open for a set period of time, create a session and then open one or more projects in that session. You can specify a timeout interval while creating the session. If the session is not used during the interval, all open projects in that session are closed.

An open project can be modified and saved at any time during the process. An open project can even be saved at another location relative to the QuarkXPress Server document pool. You can also create a new project and keep it open.

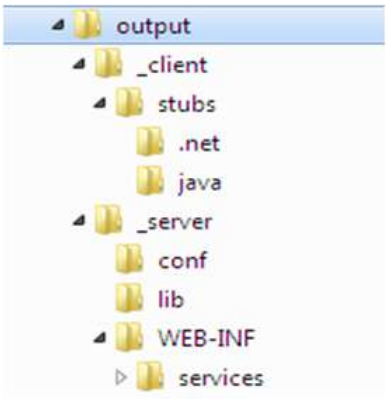
Using the Extensibility tool

To compile and generate artifacts of custom objects:

1. Update the file [QXPSM-XDK]/Extensibility/rogenerator/ManagerSDK.xml by adding the definitions of any custom objects being created by QuarkXPress Server. Make sure you do not use existing class names.
2. Open the file “Generate.command”/"Generate.bat" in a text editor and make the following changes:
    - Set the value of QXPSM_LIB_DIR to the path of the QXPSM libraries. For example: @set QXPSM_LIB_DIR="C:\Quark\QuarkXPressServer Manager\Server\lib"

- Set the value of QXPSM_DEPENDENCIES_DIR to the path of the third-party libraries. For example: @set QXPSM_DEPENDENCIES_DIR="C:\Quark\QuarkXPressServer Manager\Server\dependencies“
- Set the value of VS_COMMON_TOOLS to the path of the Visual Studio common tools. For example: @set VS_COMMON_TOOLS="C:\Program Files\Microsoft Visual Studio 9.0\Common7\Tools\"
- Set the value of QXPSM_OUTPUT_DIR to the output location. For example: QXPSM_OUTPUT_DIR= "c:\output"

3 Execute the file “Generate.command"/"Generate.bat". The resulting output uses the following structure.



4 To use the generated artifacts:
- Copy the contents of [output location]/_server/conf to [QXPSM application folder]/Server/conf.
- Copy the contents of [output location]/_server/lib to [QXPSM application folder]/Server/lib.
- Restart QuarkXPress Server Manager.

Compatible client side web-service stubs, for both Java and .NET can be found at [output location]/_client/stubs

Understanding ManagerSDK.xml

"ManagerSDK.xml" is used to generate client SDK classes for QuarkXPress Server requests. Each element in “ManagerSDK.xml" corresponds to a request handler, a render type, or an element in the schema.

A client SDK class is generated for each element in the XML. Each property in the schema and each parameter of the request handler or render type also corresponds to a unique element in the XML.

A Class variable is generated for each property, as follows.

- <Class>: One element for each SDK class generated. The class generated is derived from QRequest. Attributes are:
  - name: The name of the generated class.
  - namespace: The namespace recognized by QuarkXPress Server when this request class is translated into a QuarkXPress Server request.
  - description: A description of the class. Unless this value is null, the description forms the header of the generated class and is included in the generated API docs.
  - alias: The alias to be used as an element name if this request class is serialized to XML. For example, when the Project class is serialized to XML, the element used is Project.
  - serializeAs: Determines how the class is serialized. The valid values are:
    - nameValue indicates that all members of the class should be handled as name-value pairs in the request to QuarkXPress Server. (This is the default option in JPEGRenderRequest and ModifierStreamRequest.)
    - xml indicates that the class should be serialized as XML with the class name or alias as the element value. All of the fields of the class are serialized as child elements. If the field is a subclass of QRequest, it is processed recursively. If the field is an array, it must be an array of QRequest-derived classes.
    - mixed indicates that the class should be serialized as XML with the class name or alias as the element. All the primitive fields of the class are serialized as attributes. If the field is a subclass of QRequest, it is serialized as a child element and processed recursively. If the field is an array, it must be an array of QRequest-derived classes.
    - attribute indicates that the class should be serialized as XML with the class name or alias as the element. The class must be primitive. All such fields must be serialized as attributes of the element. Also, "value" fields must be serialized as values of the element. Valid only if the parent class has a serializeAs value of "xml" or "mixed."
- <Attribute>: One element for each class field.
  - name: The name of the generated class variable.
  - accessor: The name of the accessor that gets the property. If this value is null, the default accessor name is used. The default name is "get" + CamelCase(name) (for example, if the name of the property is "quality," the default accessor method is getQuality).
  - mutator: The name of the accessor that sets the property. If this value is null, the default mutator name is used. The default name is "set" +

CamelCase(name) (for example, if the name of the property is “quality,” the default mutator method is setQuality).

- description: A description of the attribute. Unless this value is null, the description is included in variable headers and accessor and mutator headers and is included in generated API docs.
- type: The type of the class variable. If this value is null, the default type (string) is used. If this is not a primitive data type, it should be defined as a separate Class element. If this attribute has a value of “reference,” it means the class defined by name is a reference that will be used by a reference attribute in the same Class element. Before serialization, the referring values are set in this instance.
- reference: Unless this attribute has a null value, during serialization the value of the field should be set in the reference class provided. Note that the reference class should be declared using “type=reference” as explained above.
- readonly: If this value is true, this field is for read-only purposes and should be ignored during serialization.
- hidden: If this value is true, this field should be generated as a private variable. As such, it would not be included in WSDL.
- deprecated: If this value is true, this field has been deprecated, should not be used, is not supported, and will be removed in a future version of QuarkXPress Server.
- cdata: If this value is true, the value of this field is to be wrapped in a cdata section before being sent to QuarkXPress Server. This is valid only if the field is “value”, that is value of the element in modifier XML.
- <others>: If any other attributes are defined, a class field with the name as <name>_<others> is created, and you can write your own implementation for it.

Using the RequestParameters class

RequestParameters is a generic request class that can be used in lieu of any class, existing or otherwise.

RequestParameters has a namespace property, which can be used to send any request. For example:

```
RequestParameters requestParameters = new RequestParameters();
requestParameters.setRequestNamespace("jpeg");
```

It also has an array of dynamic parameters, which can be used to parameterize the request. For example:

```
NameValueParam param = new NameValueParam();
```

```
param.setParamName("scale");
param.setTextValue("1");
requestParameters.setParams(new NameValueParam[]{param });
```

It can also be executed using the QuarkXPress Server Manager Web service API. For example:

```
QRequestContext qRequestContext = new QRequestContext();
qRequestContext.setRequest(requestParameters);
```

Sample applications

The topics below describe the sample applications distributed with QuarkXPress Server.

Sample applications: QXP Server Manager

These sample applications are available in the QuarkXPress Server installation package.

ASP.NET samples

This sample application consists of Web pages that demonstrate different ways the object model can be used to post QuarkXPress Server requests for various operations. To use this application:

1. Create a virtual directory (for example, "ClientSDKSamplesSite") in IIS.
2. Copy the sample from [QuarkXPress Server Manager application folder]\XDK\samples\asp.net\clientsdksamples and set the home path of the Web demo to the virtual directory.
3. Set the endpoint address for Web services calls in the "web.config" file like so: configuration - >appSettings - >add key="com.quark.qxpsm.RequestService" value= "End Point Address"
4. Restart IIS.
5. In a browser, enter the following URL: http://<IIS Server Name>:<Port>/ClientSDKSamplesSite/Index.htm

C# samples

These samples show how to use C# to take advantage of bullets and numbering, callouts, and conditional styles with QuarkXPress Server requests transmitted via the QuarkXPress Server Manager Web services interface. They use .NET Web service stubs provided by the QuarkXPress Server Manager SDK.

- The "AddFileRequest" sample shows how to make servlet requests to QuarkXPress Server Manager, instead of using QuarkXPress Server Manager

Web service stubs. Web services use SOAP to pass data, and SOAP is not designed to transfer large amounts of data, so Quark recommends using the servlet interface to upload and download files in a production environment.

Java samples

These samples show how to use Java to take advantage of bullets and numbering, callouts, and conditional styles with QuarkXPress Server requests transmitted via the QuarkXPress Server Manager Web services interface. They use Java Web service stubs provided by the QuarkXPress Server Manager SDK.

- The "AddFileRequest" sample shows how to use the Apache HttpClient library to make servlet requests to QuarkXPress Server Manager, instead of using QuarkXPress Server Manager Web service stubs. Web services use SOAP to pass data, and SOAP is not designed to transfer large amounts of data, so Quark recommends using the servlet interface to upload and download files in a production environment.

JSP samples

These samples have been developed using JSP, for deployment in the same Web server as that of QuarkXPress Server Manager. They show how to make local calls to QuarkXPress Server Manager's RequestService to perform various tasks.

By default, these samples are deployed as a separate webapp named "clientsdksamples". You can access this webapp from the QuarkXPress Server Manager home page.

Objective-C samples

These samples show how to use Objective-C to make QuarkXPress Server requests via the Web services interface provided by QuarkXPress Server Manager. They demonstrate document rendering and modifcation under both macOS and iOS.

- The "AddFileRequest" sample shows how to make servlet requests to QuarkXPress Server Manager, instead of using QuarkXPress Server Manager Web service stubs. Web services use SOAP to pass data, and SOAP is not designed to transfer large amounts of data, so Quark recommends using the servlet interface to upload and download files in a production environment.

Sample applications legal notice

©2012 Quark Software Inc. as to the content and arrangement of this material. All rights reserved.

©1986–2012 Quark Software Inc. and its licensors as to the technology. All rights reserved.

Protected by one or more of U.S. Patent Nos. 5,541,991, 5,907,704, 6,005,560, 6,052,514, 6,081,262, 6,947,959 B1, 6,940,518 B2, 7,116,843 and other patents pending.

Quark Products and materials are subject to the copyright and other intellectual property protection of the United States and foreign countries. Unauthorized use or reproduction without Quark's written consent is prohibited.

QUARK IS NOT THE MANUFACTURER OF THIRD PARTY SOFTWARE OR OTHER THIRD PARTY HARDWARE (HEREINAFTER "THIRD PARTY PRODUCTS") AND SUCH THIRD PARTY PRODUCTS HAVE NOT BEEN CREATED, REVIEWED, OR TESTED BY QUARK, THE QUARK AFFILIATED COMPANIES OR THEIR LICENSORS. (QUARK AFFILIATED COMPANIES SHALL MEAN ANY PERSON, BRANCH, OR ENTITY CONTROLLING, CONTROLLED BY OR UNDER COMMON CONTROL WITH QUARK OR ITS PARENT OR A MAJORITY OF THE QUARK SHAREHOLDERS, WHETHER NOW EXISTING OR FORMED IN THE FUTURE, TOGETHER WITH ANY PERSON, BRANCH, OR ENTITY WHICH MAY ACQUIRE SUCH STATUS IN THE FUTURE.)QUARK, THE QUARK AFFILIATED COMPANIES AND/OR THEIR LICENSORS MAKE NO WARRANTIES, EITHER EXPRESS OR IMPLIED, REGARDING THE QUARK PRODUCTS/SERVICES AND/OR THIRD PARTY PRODUCTS/SERVICES, THEIR MERCHANTABILITY, OR THEIR FITNESS FOR A PARTICULAR PURPOSE. QUARK, THE QUARK AFFILIATED COMPANIES AND THEIR LICENSORS DISCLAIM ALL WARRANTIES RELATING TO THE QUARK PRODUCTS/SERVICES AND ANY THIRD PARTY PRODUCTS/SERVICES. ALL OTHER WARRANTIES AND CONDITIONS, WHETHER EXPRESS, IMPLIED OR COLLATERAL, AND WHETHER OR NOT, MADE BY DISTRIBUTORS, RETAILERS, XTENSIONS DEVELOPERS OR OTHER THIRD PARTIES ARE DISCLAIMED BY QUARK, THE QUARK AFFILIATED COMPANIES AND THEIR LICENSORS, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF NON-INFRINGEMENT, COMPATIBILITY, OR THAT THE SOFTWARE IS ERROR-FREE OR THAT ERRORS CAN OR WILL BE CORRECTED. THIRD PARTIES MAY PROVIDE LIMITED WARRANTIES AS TO THEIR OWN PRODUCTS AND/OR SERVICES, AND USERS MUST LOOK TO SAID THIRD PARTIES FOR SUCH WARRANTIES, IF ANY. SOME JURISDICTIONS, STATES OR PROVINCES DO NOT ALLOW LIMITATIONS ON IMPLIED WARRANTIES, SO THE ABOVE LIMITATION MAY NOT APPLY TO PARTICULAR USERS.IN NO EVENT SHALL QUARK, THE QUARK AFFILIATED COMPANIES, AND/OR THEIR LICENSORS BE LIABLE FOR ANY SPECIAL, INDIRECT, INCIDENTAL, CONSEQUENTIAL OR PUNITIVE DAMAGES, INCLUDING, BUT NOT LIMITED TO, ANY LOST PROFITS, LOST TIME, LOST SAVINGS, LOST DATA, LOST FEES, OR EXPENSES OF ANY KIND ARISING FROM INSTALLATION OR USE OF THE QUARK PRODUCTS/SERVICES, IN ANY MANNER, HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY. IF, NOTWITHSTANDING THE FOREGOING, QUARK, THE QUARK AFFILIATED COMPANIES AND/OR THEIR LICENSORS ARE FOUND TO HAVE LIABILITY RELATING TO THE QUARK PRODUCTS/SERVICES OR THIRD PARTY PRODUCTS/SERVICES, SUCH LIABILITY SHALL BE LIMITED TO THE AMOUNT PAID BY THE USER TO QUARK

FOR THE SOFTWARE/SERVICES AT ISSUE (EXCLUDING THIRD PARTY PRODUCTS/SERVICES), IF ANY, OR THE LOWEST AMOUNT UNDER APPLICABLE LAW, WHICHEVER IS LESS. THESE LIMITATIONS WILL APPLY EVEN IF QUARK, THE QUARK AFFILIATED COMPANIES, THEIR LICENSORS AND/OR THEIR AGENTS HAVE BEEN ADVISED OF SUCH POSSIBLE DAMAGES. SOME JURISDICTIONS, STATES OR PROVINCES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THIS LIMITATION OR EXCLUSION MAY NOT APPLY. ALL OTHER LIMITATIONS PROVIDED UNDER APPLICABLE LAW, INCLUDING STATUTES OF LIMITATION, SHALL CONTINUE TO APPLY.IN THE EVENT ANY OF THESE PROVISIONS ARE OR BECOME UNENFORCEABLE UNDER APPLICABLE LAW, SUCH PROVISION SHALL BE MODIFIED OR LIMITED IN ITS EFFECT TO THE EXTENT NECESSARY TO CAUSE IT TO BE ENFORCEABLE.USE OF THE QUARK PRODUCTS IS SUBJECT TO THE TERMS OF THE END USER LICENSE AGREEMENT OR OTHER APPLICABLE AGREEMENTS FOR SUCH PRODUCT/SERVICE. IN THE EVENT OF A CONFLICT BETWEEN SUCH AGREEMENTS AND THESE PROVISIONS THE RELEVANT AGREEMENTS SHALL CONTROL.

Quark, the Quark logo, QuarkXPress, XTensions, QuarkCopyDesk, Job Jackets and Composition Zones, QuarkAlliance and QPS are trademarks or registered trademarks of Quark Software Inc. and its affiliates in the U.S. and/or other countries.

OpenType, Visual C#, Visual Studio, Microsoft and Windows are registered trademarks of Microsoft Corporation in the United States and/or other countries.

macOS is a trademark of Apple, Inc. registered in the U.S. and other countries.

Adobe, PostScript and Acrobat are registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Java and all Java based marks are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Unicode is a trademark of Unicode, Inc.

MySQL is a registered trademark of MySQL AB.

PANTONE® Colors displayed in the software application or in the user documentation may not match PANTONE-identified standards. Consult current PANTONE Color Publications for accurate color. PANTONE® and other Pantone, Inc. trademarks are the property of Pantone, Inc. ©Pantone, Inc., 2008.

Color Data is produced under license from Dainippon Ink and Chemicals, Inc.

FOCOLTONE and FOCOLTONE Colour System are registered trademarks of FOCOLTONE. The concept, structure, and form of FOCOLTONE material and intellectual property are protected by patent and copyright law. Any

reproduction in any form, in whole or in part, for private use or for sale, is strictly forbidden. Contact FOCOLTONE, Ltd. for specific patent information.

Toyo Ink Mfg. Co., Ltd. is the copyright owner of TOYO INK COLOR FINDER™ SYSTEM AND SOFTWARE which is licensed to Quark Software Inc. to distribute for use only in connection with QuarkXPress. TOYO INK COLOR FINDER™ SYSTEM AND SOFTWARE shall not be copied onto another diskette or into memory unless as part of the execution of QuarkXPress. TOYO INK COLOR FINDER™ SYSTEM AND SOFTWARE © TOYO INK MFG. CO., LTD., 1991. COLOR FINDER is in the process of registration as the registered trademark of Toyo Ink Mfg. Co., Ltd. COLOR FINDER™ computer video simulation used in the product may not match the COLOR FINDER™ book, and additionally some printer color used in the product may also not match. Please use the COLOR FINDER™ book to obtain the accurate color."

TRUMATCH, TRUMATCH Swatching System, and TRUMATCH System are trademarks of TRUMATCH, Inc.

As to tt2pt1 technology, Copyright ©1997–2003 by the AUTHORS: Andrew Weeks <ccsaw@bath.ac.uk> Frank M. Siegert <fms@this.net> Mark Heath <mheath@netspace.net.au> Thomas Henlich <thenlich@rcs.urz.tu-dresden.de> Sergey Babkin <babkin@users.sourceforge.net>, <sab123@hotmail.com> Turgut Uyar <uyar@cs.itu.edu.tr> Rihardas Hepas <rch@WriteMe.Com> Szalay Tamas <tomek@elender.hu> Johan Vromans <jvromans@squirrel.nl> Petr Titera <P.Titera@sh.cvut.cz> Lei Wang <lwang@amath8.amt.ac.cn> Chen Xiangyang <chenxy@sun.ihep.ac.cn> Zvezdan Petkovic <z.petkovic@computer.org> Rigel <rigel863@yahoo.com> All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. All advertising materials mentioning features or use of this software must display the following acknowledgement: This product includes software developed by the TTF2PT1 Project and its contributors. THIS SOFTWARE IS PROVIDED BY THE AUTHORS AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH

DAMAGE. For the approximate list of the AUTHORS' responsibilities see the project history. Other contributions to the project are: Turgut Uyar <uyar@cs.itu.edu.tr> The Unicode translation table for the Turkish language. Rihardas Hepas <rch@WriteMe.Com> The Unicode translation table for the Baltic languages. Szalay Tamas <tomek@elender.hu> The Unicode translation table for the Central European languages. Johan Vromans <jvromans@squirrel.nl> The RPM file. Petr Titera <P.Titera@sh.cvut.cz> The Unicode map format with names, the forced Unicode option. Frank M. Siegert <frank@this.net> Port to Windows Lei Wang <lwang@amath8.amt.ac.cn> Chen Xiangyang <chenxy@sun.ihep.ac.cn> Translation maps for Chinese fonts. Zvezdan Petkovic <z.petkovic@computer.org> The Unicode translation tables for the Cyrillic alphabet. Rigel <rigel863@yahoo.com> Generation of the dvips encoding files, modification to the Chinese maps. I. Lee Hetherington <ilh@lcs.mit.edu> The Type1 assembler (from the package 't1utils'), its full copyright notice: Copyright ©1992 by I. Lee Hetherington, all rights reserved. Permission is hereby granted to use, modify, and distribute this program for any purpose provided this copyright notice and the one below remain intact.

As to Apache technology, copyright ©1999–2008 The Apache Software Foundation. All rights reserved. Any Apache software which is distributed with this software is developed by the Apache Software Foundation (http://www.apache.org/). Licensed under the Apache License, Version 2.0 (the "License"); you may not use these files except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE–2.0. Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

As to MoreFiles software, ©1992–2002 by Apple, Inc., all rights reserved.

Portions of this product include technology used under license from Global Graphics.

As to ICU4J technology, ICU4J license — ICU4J 1.3.1 and later,

COPYRIGHT AND PERMISSION NOTICE, Copyright ©1995–2001 International Business Machines Corporation and others. All rights reserved. Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, provided that the above copyright notice(s) and this permission notice appear in all copies of the Software and that both the above copyright notice(s) and this permission notice appear in supporting documentation. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT

NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE BE LIABLE FOR ANY CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE. Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

This software is based in part on the work of the Independent JPEG Group.

As to Microsoft technology, ©1988–2008 Microsoft Corporation. All rights reserved.

As to Nodeka software, ©1999–2002 Justin Gottschlich. All rights reserved.

As to STLport technology, Copyright 1999,2000 Boris Fomitchev. This material is provided “as is", with absolutely no warranty expressed or implied. Any use is at your own risk. Permission to use or copy this sofware for any purpose is hereby granted without fee, provided the above notices are retained on all cpies. Permission to modify the code and to distribute modified code is granted, provided the above notices are retained, and a notice that the code was modified is included with the above copyright notice. The Licensee may distribute binaries compiled with STLport (whether original or modified) without any royalties or restrictions. The Licensee may distribute original or modified STLport sources, provided that: The conditions indicated in the above permission notice are met; The following copyright notices are retained when present, and conditions provided in accompanying permission notices are met: Copyright 1994 Hewlett-Packard Company. Copyright 1996,97 Silicon Graphics Computer Systems, Inc. Copyright 1997 Moscow Center for SPARC Technology.

Permission to use, copy, modify, distribute and sell this software and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation. Hewlett-Packard Company makes no representations about the suitability of this software for any purpose. It is provided “as is" without express or implied warranty.Permission to use, copy, modify, distribute and sell this software and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation. Silicon Graphics makes no representations about the suitability of this software for any purpose. It is provided “as is" without express or implied warranty.Permission to use, copy, modify, distribute and sell this software and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice

appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation. Moscow Center for SPARC Technology makes no representations about the suitability of this software for any purpose. It is provided “as is" without express or implied warranty.

As to Dr. Brian Gladman software, Copyright ©2001, Dr. Brian Gladman <brg@gladman.uk.net>, Worcester, UK. All rights reserved. LICENSE TERMS The free distribution and use of this software in both source and binary form is allowed (with or without changes) provided that: 1. distributions of this source code include the above copyright notice, this list of conditions and the following disclaimer; 2. distributions in binary form include the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other associated materials; 3. the copyright holder's name is not used to endorse products built using this software without specific written permission. DISCLAIMER This software is provided ‘as is’ with no explicit or implied warranties in respect of any properties, including, but not limited to, correctness and fitness for purpose.

As to cascading menus based on menu.js. by Gary Smith, July 1997, Copyright ©1997–1999 Netscape Communication Corp. Netscape grants you a royalty free license to use or modify the cascading menus software provided that this copyright notice appears on all copies. This software is provided “AS IS," without a warranty of any kind.

Portions of this software is based on the work of Jean-loup Gailly and Mark Adler and is ©1995–1998 Jean-loup Gailly and Mark Adler [ZIP library]

As to Sun technology, Copyright 2003–2006, Sun Microsystems, Inc. All rights reserved. Use is subject to license terms.

As to Apple technology, ©2002–2004 Apple, Inc. All rights reserved. This Apple software is supplied to you by Apple, Inc. (“Apple") in consideration of your agreement to the following terms, and your use, installation, modification or redistribution of this Apple software constitutes acceptance of these terms. If you do not agree with these terms, please do not use, install, modify or redistribute this Apple software. In consideration of your agreement to abide by the following terms, and subject to these terms, Apple grants you a personal, non-exclusive license, under Apple's copyrights in this original Apple software (the “Apple Software"), to use, reproduce, modify and redistribute the Apple Software, with or without modifications, in source and/or binary forms; provided that if you redistribute the Apple Software in its entirety and without modifications, you must retain this notice and the following text and disclaimers in all such redistributions of the Apple Software. Neither the name, trademarks, service marks or logos of Apple, Inc. may be used to endorse or promote products derived from the Apple Software without specific prior written permission from Apple. Except as expressly stated in this notice, no other rights or licenses, express or implied, are granted by Apple herein, including but not limited to any patent rights that may be infringed by your derivative works or by

other works in which the Apple Software may be incorporated. The Apple Software is provided by Apple on an “AS IS" basis. APPLE MAKES NO WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING THE APPLE SOFTWARE OR ITS USE AND OPERATION ALONE OR IN COMBINATION WITH YOUR PRODUCTS. IN NO EVENT SHALL APPLE BE LIABLE FOR ANY SPECIAL, INDIRECT, INCIDENTAL OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) ARISING IN ANY WAY OUT OF THE USE, REPRODUCTION, MODIFICATION AND/OR DISTRIBUTION OF THE APPLE SOFTWARE, HOWEVER CAUSED AND WHETHER UNDER THEORY OF CONTRACT, TORT (INCLUDING NEGLIGENCE), STRICT LIABILITY OR OTHERWISE, EVEN IF APPLE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

As to HTML Parsing code technology, Copyright ©1998 World Wide Web Consortium (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved. Contributing Author(s): Dave Raggett <dsr@w3.org> The contributing author(s) would like to thank all those who helped with testing, bug fixes, and patience. This wouldn't have been possible without all of you. COPYRIGHT NOTICE: This software and documentation is provided “as is," and the copyright holders and contributing author(s) make no representations or warranties, express or implied, including but not limited to, warranties of merchantability or fitness for any particular purpose or that the use of the software or documentation will not infringe any third party patents, copyrights, trademarks or other rights. The copyright holders and contributing author(s) will not be liable for any direct, indirect, special or consequential damages arising out of any use of the software or documentation, even if advised of the possibility of such damage. Permission is hereby granted to use, copy, modify, and distribute this source code, or portions hereof, documentation and executables, for any purpose, without fee, subject to the following restrictions: 1. The origin of this source code must not be misrepresented. 2. Altered versions must be plainly marked as such and must not be misrepresented as being the original source. 3. This Copyright notice may not be removed or altered from any source or altered source distribution. The copyright holders and contributing author(s) specifically permit, without fee, and encourage the use of this source code as a component for supporting the Hypertext Markup Language in commercial products. If you use this source code in a product, acknowledgment is not required but would be appreciated.

As to DOM4J software, Redistribution and use of this software and associated documentation (“Software"), with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain copyright statements and notices. Redistributions must also contain a copy of this document.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

The name “DOM4J” must not be used to endorse or promote products derived from this Software without prior written permission of MetaStuff, Ltd. For written permission, please contact dom4j-info@metastuff.com.

Products derived from this Software may not be called “DOM4J” nor may “DOM4J” appear in their names without prior written permission of MetaStuff, Ltd. DOM4J is a registered trademark of MetaStuff, Ltd.

Due credit should be given to the DOM4J Project — http://www.dom4j.org

THIS SOFTWARE IS PROVIDED BY METASTUFF, LTD. AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL METASTUFF, LTD. OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright 2001–2005 (C) MetaStuff, Ltd. All Rights Reserved.

As to Jaxen technology: Copyright 2003–2006 The Werken Company. All Rights Reserved.

License Text

$Id: LICENSE.txt,v 1.5 2006/02/05 21:49:04 elharo Exp $

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the ‘Jaxen' nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS 'AS IS' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

All other marks are the property of their respective owners.

Dynamic Pagination and Flow

➡ The *Evolved Mechanics* sample, demonstrating the flow automation constructs, can be downloaded from the QXPS Administration home page, using the **Download SDK and Samples** link.

There are three main division in the structure of any document:

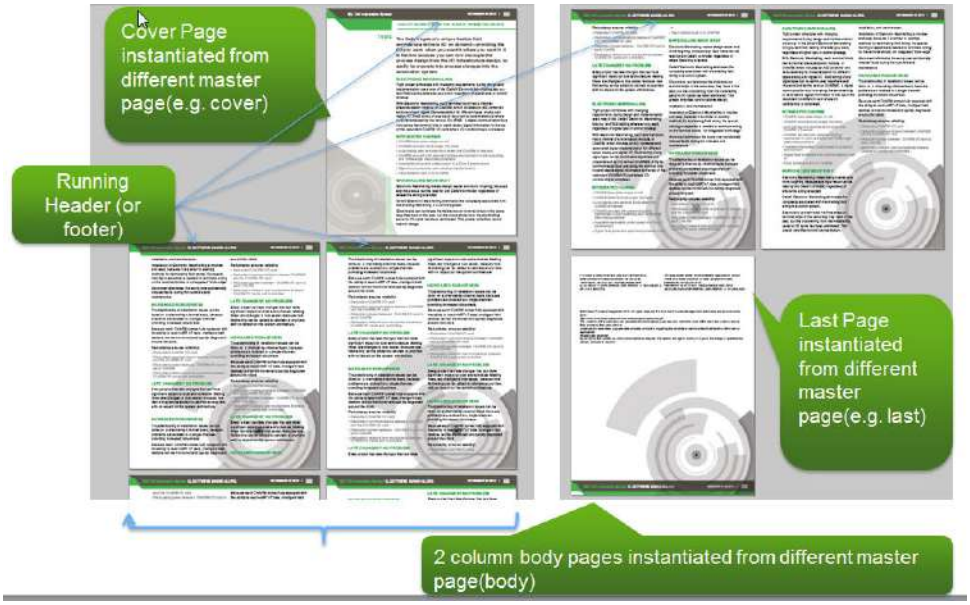
1. the front page content
2. the body content
3. the back page content

The Dynamic Pagination and Flow feature gives you the ability to define multiple page layouts in a single flow. The Page Sequences can be a logical section or a chapter comprised of a single flow. Different pages of a Page Sequence can be based on different master pages. Pages will be organized while adhering to defined constraints.

The following is a list of customer use cases that can be fullfilled using the new modifier XML markup:

- The ability to specify running headers and running footers.
- The ability to pre-define the sequence of the application of the available master pages on the various pages being created dynamically.
- The ability to express/segment the flow content into logical sections.
- Bulleted text, Section numbering, Text Styling, colors etc
- Components that are re-usable across multiple document types. For example, disclaimers and copy right notices.
- The ability to impose constraints on the page count (odd or even) per section, enabling you to automatically insert blank pages for print versions of documentation.
- The ability to specify the desired page numbering formats for various sections (for example, index or glossary pages having roman numerals).
- The ability to have repeatable interactivy aspects.
- The ability to present tablular content in various ways:
- inline
- spanned across columns

- full width spanned
- full page - rotated / un-rotated.
- The ability to format tables in the following various ways:
- horizontal and vertical
- gridlines
- grouping headers
- shading rows and columns
- the inclusion of table notes with the table
- The ability to break content across pages along with headers.
- The ability to specify the placement of images and charts in the following scenarios:
- inline
- spanned across columns
- full page
- The ability to specify a different Table of Contents for digital and print issues.



Example User Pagination Requirement

Dynamic Pagination and Flow Problem

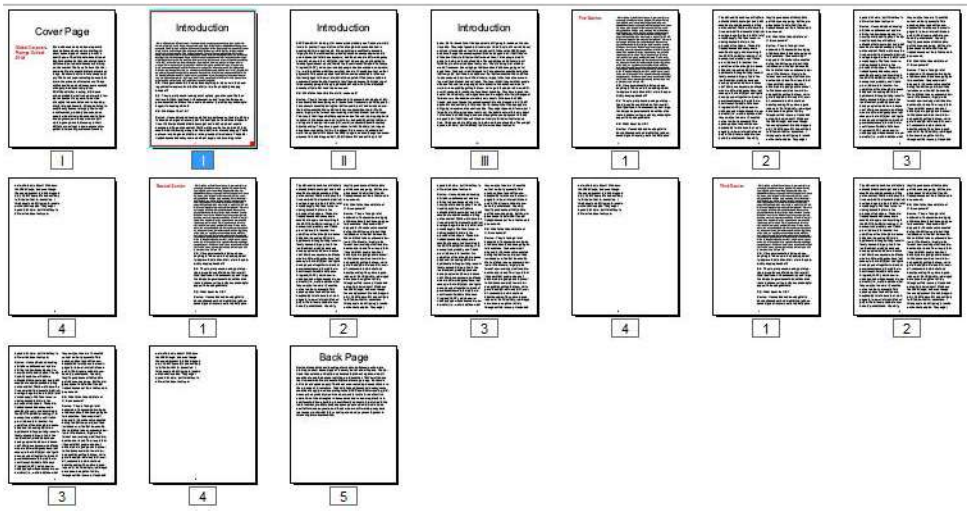
Consider the following pagination requirement.

The template has 5 master pages:

- Cover - a master page for the cover page
- Introduction - a master page for introduction page(s)
- Section First Page - a master page for the first page of each section (a section's cover page)
- Section Content - a master page to be used for each page that has body content
- Back Page - a master page for the back page (a back cover)

The pagination requirement is:

1. Cover
2. Introduction
3. Multiple sections
    - Section First Page
    - Section Content
4. Back Page



Example User Pagination Requirement

Previously, when a desktop user created this document, he accomplished this manually by dragging and dropping the desired master pages onto each page. The user was not able to achieve the desired pagination using the automated dynamic publishing workflows (XML--> PDF or XML-->iPad).

Dynamic Pagination and Flow Solution

The Dynamic Pagination and Flow feature introduced in the 9.5.1 release of QuarkXPress Server, solves this problem.

New Parts of the ModifierXML:

1. The MASTERPAGESEQUENCE element gives you the ability to describe the use and application of master pages by the QuarkXPress Server Modifier XML processor and further by its layout engine. It also allows the user to define the pagination pattern with a name.
2. The PAGESEQUENCE element contains the actual flow content with reference to the MASTERPAGESEQAUENCE via a @MASTERPAGESEQUENEREF and a @MASTERPAGEREF respectively. Each PAGESEQUENCE element will have a certain non-variable static content segment (represented by STATICCONTENT) containing the content that is intended to be repeated across multiple flow pages.

Each PAGESEQUENCE has a child object STORY which acts as a container for the flow content containing multiple PARAGRAPH elements.

Each PARAGRAPH object holds RICHTEXT giving you the ability to apply a wide variety of styling.

3. The SINGLEMASTERPAGEREFERENCE element defines a sequence in which a particular master page will be applied to a singles page in a page sequence. It contains the name of the master page in the QuarkXPress template to be used. The given master page is applied to one page of a page sequence. This is useful for front matter, back matter and section start pages.

The following is an example of the use of the SINGLEMASTERPAGEREFERENCE

```
<MASTERPAGESEQUENCE NAME="SingleMaster">
  <SINGLEMASTERPAGEREFERENCE NAME="A-
MasterA"></SINGLEMASTERPAGEREFERENCE>
  <SINGLEMASTERPAGEREFERENCE NAME="B-
MasterB"></SINGLEMASTERPAGEREFERENCE>
  <SINGLEMASTERPAGEREFERENCE NAME="C-
MasterC"></SINGLEMASTERPAGEREFERENCE>
</MASTERPAGESEQUENCE>
```

This applies the A-MasterAmaster page to the first page, B-MasterB master page to the second page, and the C-MasterC master page to the third page and onwards.

4. The REPEATABLEMASTERPAGEREFERENCE element defines a sequence in which a master page will be applied to multiple pages in a page sequence. It contains the name of the master page in the QuarkXPress template to be used. This is useful for constructing runs of identical pages and causes a bounded or unbounded sequence of pages to be generated using the same master page.

This element is a super set of the SINGLEMASTERPAGEREFERENCE element. It allows an application of a particular master page upto n number of pages using the MAXREPEATS attribute.

The following is an example of the use of the REPEATABLEMASTERPAGEREFERENCE.

```
<MASTERPAGESEQUENCE NAME="RepeatableMaster">
 <REPEATABLEMASTERPAGEREFERENCE MAXREPEATS="5"
NAME="B-MasterB"/>
 <REPEATABLEMASTERPAGEREFERENCE NAME="C-MasterC"/>
</MASTERPAGESEQUENCE>
```

This applies the B-MasterB master page to the first 5 pages. From the sixth page onward, the C-MasterC master page will be applied.

5 The REPEATABLEMASTERPAGEALTERNATIVES element defines a master page along with conditions that must be satisfied to apply the given master page on a page. It contains one or more CONDITIONALMASTERPAGEREFERENCE elements that define the conditions.

This is a super set of the first two master page sequences. It allows an application of a particular master page on even pages, a different master page on odd pages and a different master page on first and last pages.

The following is an example of the use of the REPEATABLEMASTERPAGEALTERNATIVES.

```
<MASTERPAGESEQUENCE NAME="ConditionalMaster">
 <REPEATABLEMASTERPAGEALTERNATIVES>
  <CONDITIONALMASTERPAGEREFERENCE NAME="D-MasterD"
POSITION="FIRST"/>
  <CONDITIONALMASTERPAGEREFERENCE NAME="A-MasterA"
ODDOREVEN="EVEN" POSITION="REST"/>
  <CONDITIONALMASTERPAGEREFERENCE NAME="C-MasterC"
ODDOREVEN="ODD" POSITION="REST"/>
  <CONDITIONALMASTERPAGEREFERENCE NAME="B-MasterB"
POSITION="LAST"/>
 </REPEATABLEMASTERPAGEALTERNATIVES>
</MASTERPAGESEQUENCE>
```

This applies the D-MasterD master page to the first page, the A-MasterA master page to all even numbered pages, the C-MasterC master page to all odd numbered pages, and the B-MasterB master page to the last page.

6 The CONDITIONALMASTERPAGEREFERENCE element gives you the ability to specify the master page along with the conditions that must be satisfied to apply the given master page on a page. The following conditions may be specified:

- a page's position within a sequence
- the odd or even page number property
- whether or not a particular page is blank

contains the actual flow content with reference to the MASTERPAGESEQAUENCE via a @MASTERPAGESEQUENEREF and a

@MASTERPAGEREF respectively. Each PAGESEQUENCE element will have a certain non-variable static content segment (represented by STATICCONTENT) containing the content that is intended to be repeated across multiple flow pages.

7 The STATICCONTENT element gives you the ability to specify content chunks that are intended to be repeated across multiple flow pages. For example, running headers or footers.

8 The SECTIONNUMBERFORMAT element gives you the ability to specify the page number format.

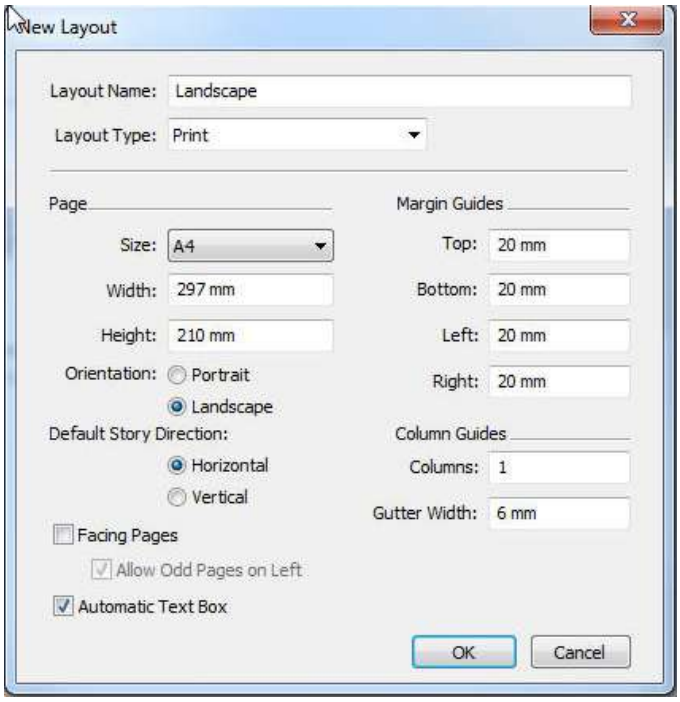
```
<PROJECT>
<LAYOUT>
 <ID UID="1"/>
 <MASTERPAGESEQUENCE NAME="APSMasterPages">
  <REPEATABLEMASTERPAGEALTERNATIVES>
   <CONDITIONALMASTERPAGEREFERENCE POSITION="FIRST"
NAME="C-ArticleFirst"/>
   <CONDITIONALMASTERPAGEREFERENCE POSITION="REST"
NAME="D-ArticleRest"/>
   <CONDITIONALMASTERPAGEREFERENCE POSITION="LAST"
NAME="E-ArticleLast"/>
   <CONDITIONALMASTERPAGEREFERENCE
BLANKORNOTBLANK="BLANK"
    NAME="F-BlankMaster"/>
  </REPEATABLEMASTERPAGEALTERNATIVES>
 </MASTERPAGESEQUENCE>
 <PAGESEQUENCE
MASTERPAGESEQUENCEREF="APSMasterPages"
FORCEPAGECOUNT="ENDONEVEN">
  <SECTIONNUMBERFORMAT FORMAT="NUMERIC"
INITIALPAGENUMBER="1"/>
  <STATICCONTENT>
   <BOX>
    <ID NAME="logo"/>
    <TEXT>
     <STORY>
      <PARAGRAPH PARASTYLE="Article SubHead">
       <RICHTEXT BOLD="true">Article Sub heading goes here
</RICHTEXT>
      </PARAGRAPH>
     </STORY>
    </TEXT>
   </BOX>
   <BOX>
    <ID NAME="title"/>
```

```
      <TEXT>
       <STORY>
        <PARAGRAPH MERGE="FALSE" PARASTYLE="NORMAL">
         <FORMAT ALIGNMENT="RIGHT"/>
         <RICHTEXT BOLD="true" COLOR="White" SIZE="14">
          Automation System</RICHTEXT>
        </PARAGRAPH>
       </STORY>
      </TEXT>
     </BOX>
    </STATICCONTENT>
    <STORY BOXNAME="flow">
     <PARAGRAPH PARASTYLE="Article Opening Paragraph">
     <PARAGRAPH PARASTYLE="Article Section Start">
     <PARAGRAPH PARASTYLE="Article Body Copy">
     <PARAGRAPH PARASTYLE="Article Body Copy">
     <PARAGRAPH PARASTYLE="Article Body Copy">
     <PARAGRAPH PARASTYLE="Article Section Start">
     <PARAGRAPH PARASTYLE="Article Body Copy">
     <PARAGRAPH PARASTYLE="Article Body Copy">
    </STORY>
   </PAGESEQUENCE>
  </LAYOUT>
  </PROJECT>
```

Landscape pagination

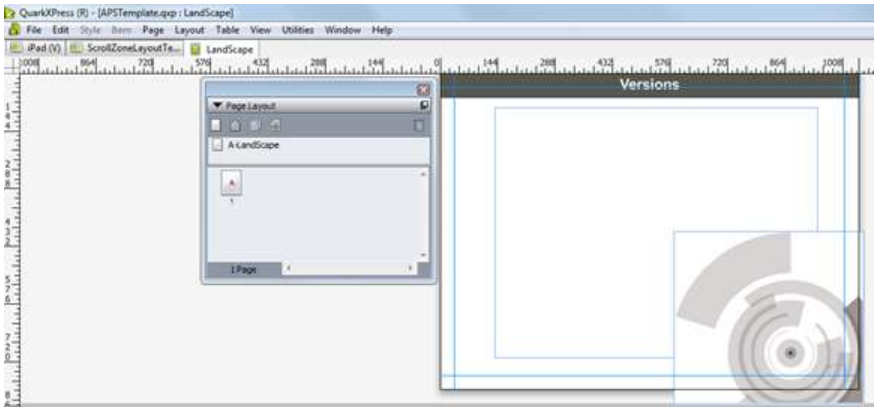
The support for mixed orientation in the PDF output gives you the ability to define different page orientations in a single flow. This can be accomplished by using the PAGESEQUENCE element of the ModifierXML. The PAGESEQUENCE element contains a new attribute ORIENTATION which allows you to specifiy a page orientation for each page in the flow. Setting the value of ORIENTATION to *Landscape* would cause the contents of the STORY element of the page sequence to flow into a Landscape layout made available in the template. The following steps allow you to achieve mixed orientation pages in the PDF output:

1. Add a landscape layout in the template, giving the page width and height as:
   - Page Height = The width of a page in portrait mode
   - Page Width = The height of a page in portrait mode



2 Designate the new layout as the Landscape layout by giving it the name Landscape.

3 Use this new layout to create pages with the Landscape orientation.



Creating pages with the Landscape layout

The following is an example of the MASTERPAGESEQUENCE to be used for Landscape mode.

```
<MASTERPAGESEQUENCE NAME="Landscape">
  <SINGLEMASTERPAGEREFERENCE NAME="A-
Landscape"></SINGLEMASTERPAGEREFERENCE>
</MASTERPAGESEQUENCE>
```

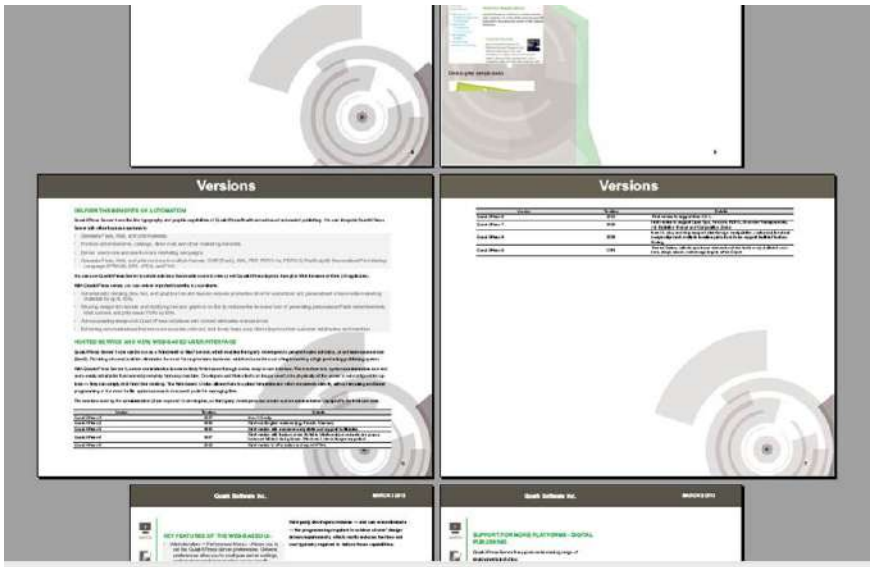
The following is an example of the PARAGRAPH element with the PARASTYLE attribute set to flow. This cause the landscape flow content under the PAGESEQUENCE that has ORIENTATION set to *LANDSCAPE* to be flown into the flow boxes in the layout with the name *Landscape*.

```
<PARAGRAPH PARASTYLE="Flow">
 <RICHTEXT BACKGROUNDCOLOR="yellow">Text here</RICHTEXT>
</PARAGRAPH>
```

The following is an example of a nested PAGESEQUENCE depicting Landscape mode. The landscape flow content under the PAGESEQUENCE that has ORIENTATION set to *LANDSCAPE* is flown into the flow boxes in the layout with the name *Landscape*.

```
<PAGESEQUENCE
ORIENTATION="LANDSCAPE"MASTERREFERENCE="Landscape">
 <STORY BOXNAME="flow">
  <PARAGRAPH PARASTYLE="Title">
   <RICHTEXT>Deliver the Benefits of Automation</RICHTEXT>
  </PARAGRAPH>
  <PARAGRAPH PARASTYLE="Flow">
   <RICHTEXT>Text Here</RICHTEXT>
  </PARAGRAPH>
  <PARAGRAPH PARASTYLE="Bulletstyle">
  <PARAGRAPH PARASTYLE="Bulletstyle">
  <PARAGRAPH PARASTYLE="Bulletstyle">
  <PARAGRAPH PARASTYLE="Bulletstyle">
  <PARAGRAPH PARASTYLE="Flow">
  <PARAGRAPH PARASTYLE="Flow">
  <PARAGRAPH PARASTYLE="Bulletstyle">
  <PARAGRAPH PARASTYLE="Bulletstyle">
  <PARAGRAPH PARASTYLE="Bulletstyle">
  <PARAGRAPH PARASTYLE="Bulletstyle">
  <PARAGRAPH PARASTYLE="Title">
  <PARAGRAPH PARASTYLE="Flow">
  <PARAGRAPH PARASTYLE="Flow">
  <PARAGRAPH PARASTYLE="Flow">
  <PARAGRAPH/>
  <PARAGRAPH/>
  <INLINETABLE TABLESTYLEREF= "quark">
   <COLGROUP>
   <THREAD>
    <TROW>
     <ENTRY>
      <PARAGRAPH PARASTYLE="TableCOntent">
       <FORMAT ALIGNMENT= "CENTERED"/>
       <RICHTEXT>Text Here</RICHTEXT>
      </PARAGRAPH>
     </ENTRY>
     <ENTRY>
      <PARAGRAPH PARASTYLE="TableCOntent">
       <FORMAT ALIGNMENT= "CENTERED"/>
       <RICHTEXT>Text Here</RICHTEXT>
```

```
        </PARAGRAPH>
       </ENTRY>
      </THROW>
     </THREAD>
    </INLINETABLE>
   </STORY>
</PAGESEQUENCE>
<!---Portrait Mode Resumed--->
<PARAGRAPH PARASTYLE="Title">
 <RICHTEXT>Text here</RICHTEXT>
</PARAGRAPH>
```

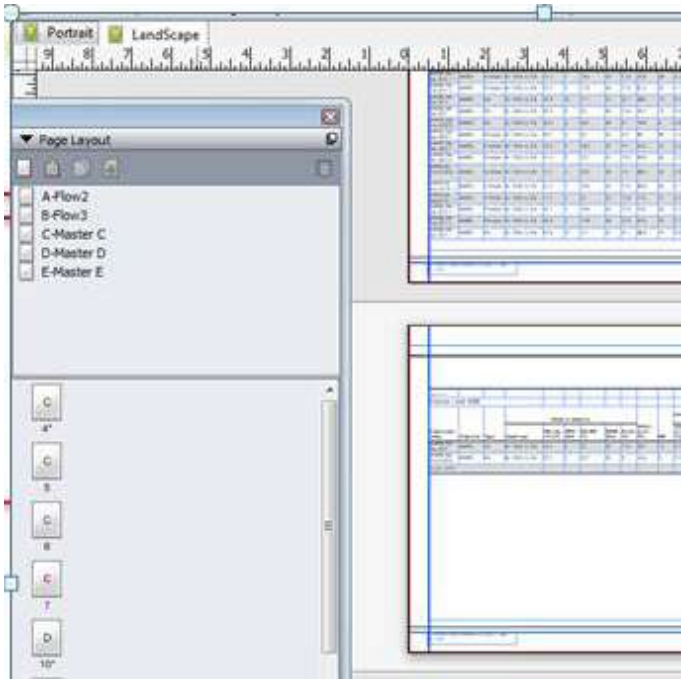


Flowing content onto pages created with the Landscape layout

The following is an example of a PROJECT element. The landscape flow content under the PAGESEQUENCE that has ORIENTATION set to *LANDSCAPE* is flown into the flow boxes in the layout with the name *Landscape*.

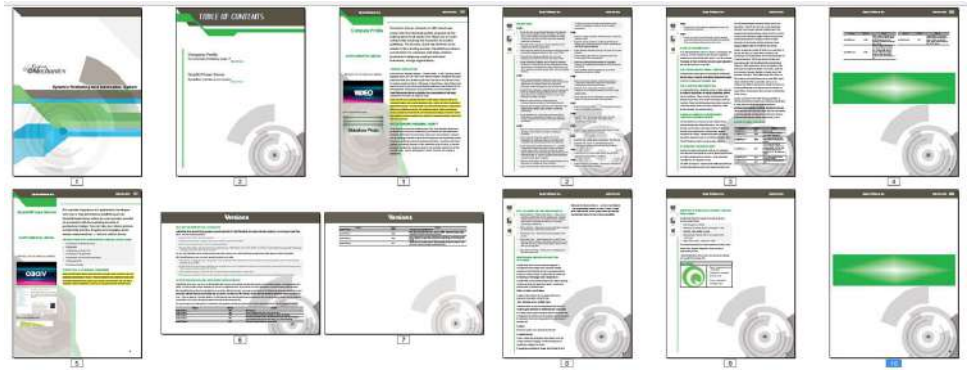
```
<PROJECT>
 <TABLESTYLE>
 <TABLESTYLE>
 <LAYOUT>
  <ID UID="1"/>
  <MASTERPAGESEQUENCE NAME= "Cover">
  <MASTERPAGESEQUENCE NAME= "Flow">
  <MASTERPAGESEQUENCE NAME= "Flow2">
   <SINGLEMASTERPAGEREFERENCE NAME= "A-Flow2"/>
  </MASTERPAGESEQUENCE>
  <MASTERPAGESEQUENCE NAME= "Flow3">
   <SINGLEMASTERPAGEREFERENCE NAME= "B-Flow3"/>
  </MASTERPAGESEQUENCE>   <PARAGRAPH
PARASTYLE="Bulletstyle">
  <MASTERPAGESEQUENCE NAME= "Disclaimer">
```

```
    <PAGESEQUENCE MASTERREFERENCE="Cover">
    <PAGESEQUENCE MASTERREFERENCE="Flow">
    <PAGESEQUENCE MASTERREFERENCE="Flow">
    <PAGESEQUENCE MASTERREFERENCE="Flow2" ORIENTATION=
"LANDSCAPE">
      <STORY BOXNAME= "StoryFlow2">
        <INLINETABLE TABLESTYLEREF= "Rating">
      </STORY>
    </PAGESEQUENCE>
    <PAGESEQUENCE MASTERREFERENCE="Flow">
    <PAGESEQUENCE MASTERREFERENCE="Flow">
    <PAGESEQUENCE MASTERREFERENCE="Flow3" ORIENTATION=
"LANDSCAPE">
      <STATICCONTENT>
      <STORY BOXNAME= "StoryFlow2">
        <INLINETABLE TABLESTYLEREF= "NoGrids">
      </STORY>
    </PAGESEQUENCE>
    <PAGESEQUENCE MASTERREFERENCE="Flow3" ORIENTATION=
"LANDSCAPE">
      <STATICCONTENT>
      <STORY BOXNAME= "StoryFlow2">
        <INLINETABLE TABLESTYLEREF= "NoGrids">
      </STORY>
    </PAGESEQUENCE>
    <PAGESEQUENCE MASTERREFERENCE="Disclaimer">
  </LAYOUT>
</PROJECT>
```



The landscape flow content under the PAGESEQUENCE that has ORIENTATION = *LANDSCAPE* is flown into the flow boxes in the layout with the name *Landscape*.

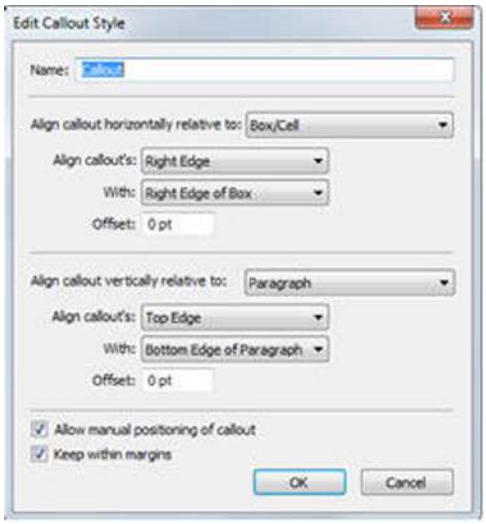
The following is the resultant PDF from the above example:



A PDF with a mixed orientation.

Automatic callout stacking

Automatic callout stacking allows for the tiling of multiple callouts. You use the CALLOUTANCHOR element to automate your layout design. The CALLOUTANCHOR@CALLOUTSTYLE requires a callout style to be created in the template. The callout style defines the placement of the graphics object.



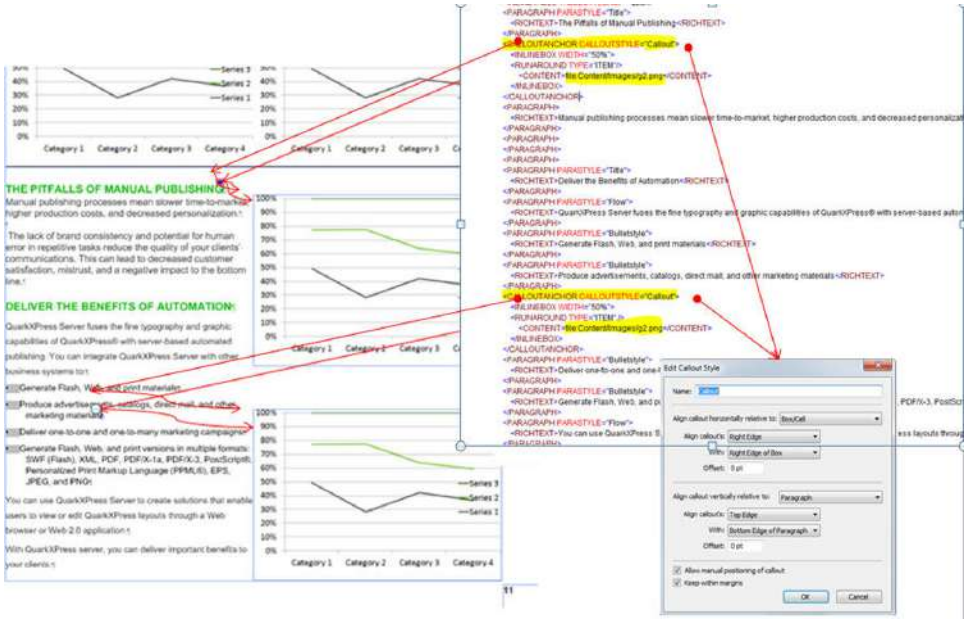
Edit Callout Style dialog box

Automatic callout stacking example

In the following example the 2 graphic objects on the right side are related to the sub-sections on the left side.

Example Automatic Callout Stacking

The callout anchors applied with the same callout style will be stacked automatically.



Result Automatic Callout Stacking

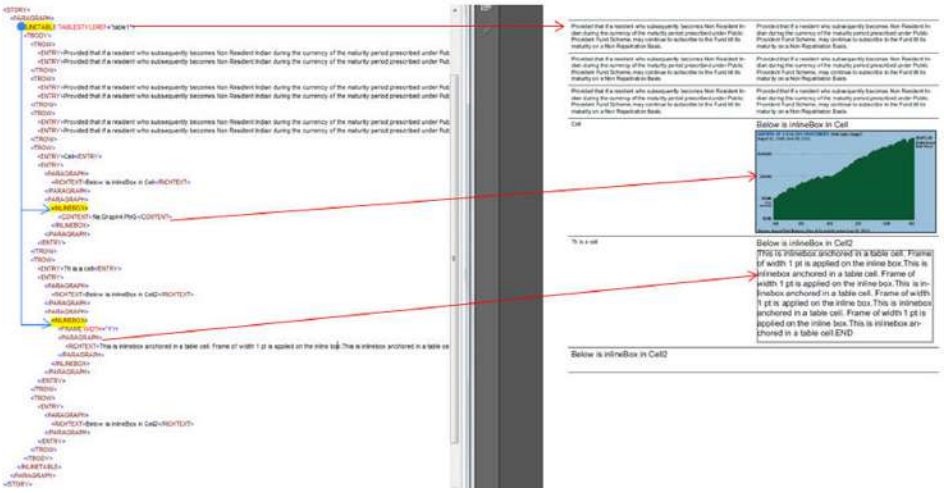
Nested anchoring

Nested anchoring allows you to anchor an item within another anchored item. This allows for nesting of INLINEBOX and INLINETABLE elements.

Nested anchoring examples

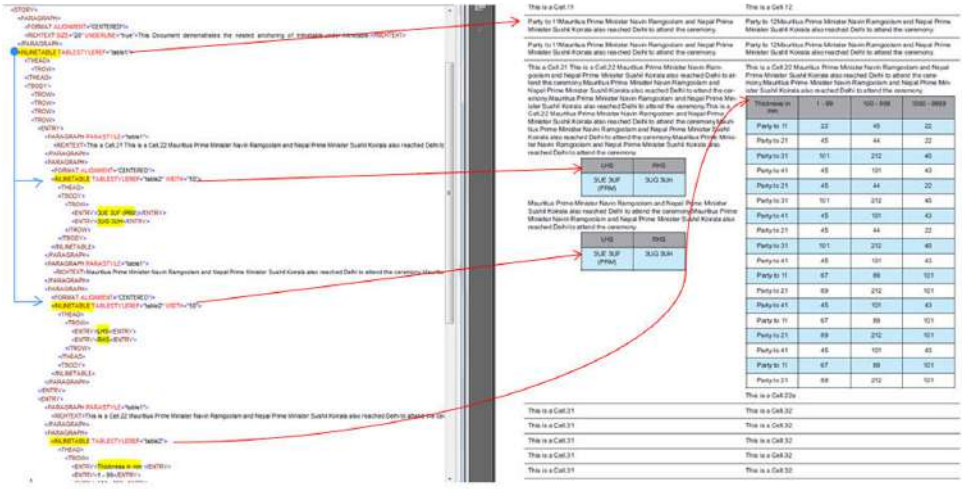
Nesting an INLINEBOX within an INLINEBOX

Nesting an INLINEBOX within an ENTRY of an INLINETABLE

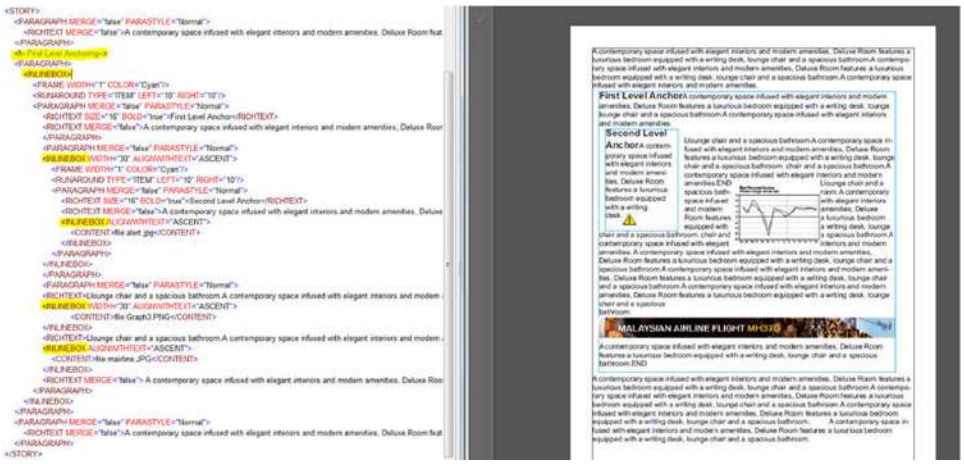


An INLINEBOX **within an** ENTRY **of an**INLINETABLE

Nesting an INLINETABLE within an INLINETABLE

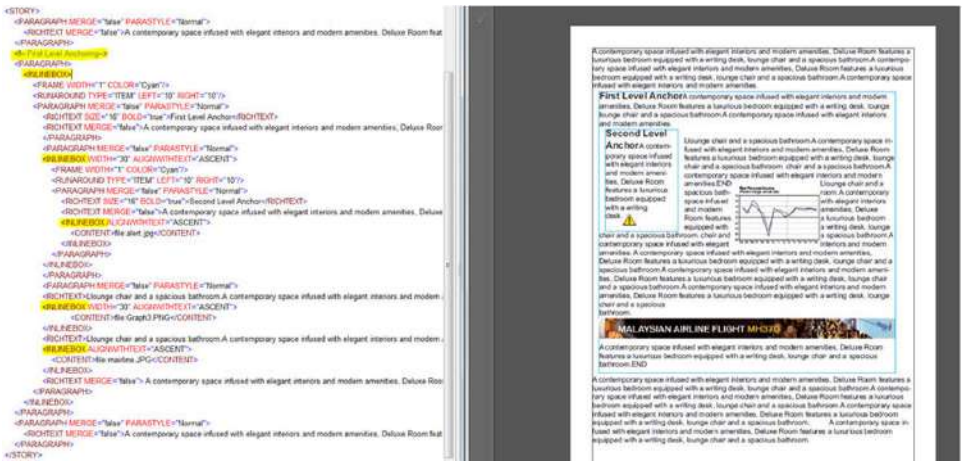


An INLINETABLE **within an** INLINETABLE



An INLINETABLE **within an** INLINETABLE

Nesting an INLINETABLE within an INLINEBOX



An INLINETABLE **within an** INLINEBOX

Modifier schema (annotated)

The topics below provide an annotated version of the Modifier schema, which is available both as a schema and in XML Schema format. Details are provided for how to form XML code that uses the construct namespace, modify parameter, and xml namespace. The XML sent to or from these functions is case-sensitive and validated by the Modifier schema, thereby providing well-formed XML code that is compatible between each function.

In the following topics:

- The "Construct" column refers to constructing a QuarkXPress project using the construct namespace.
- The "Modify" column refers to modifying a QuarkXPress project using the modify parameter.
- The "Deconstruct" column refers to deconstructing a QuarkXPress project using the xml namespace.

To conserve space, the notation used in the following topics is schema notation. See the "Modifier.xsd" file for XML Schema definitions.

➡ Measurement values do not require units. For example, "25pt" should be submitted as "25".

Entities (Modifier schema)

Element type	Construct	Modify	Deconstruct
<!ENTITY sot "&#x0002;"><!ENTITY etx "&#x0003;"><!ENTITY eot "&#x0004;"><!ENTITY enq "&#x0005;"><!ENTITY ack "&#x0006;"><!ENTITY softReturn "&#x0007;"><!ENTITY bs "&#x0008;"><!ENTITY hTab	Entities that represent QuarkXPress special characters. Note: Some entities, such as softreturn, are different for QuarkXPress than they are in the Unicode® specification.	Entities that represent QuarkXPress special characters. Note: Some entities, such as softreturn, are different for QuarkXPress than they are in the Unicode specification.	Entities that represent QuarkXPress special characters. Note: Some entities, such as softreturn, are different for QuarkXPress than they are in the Unicode specification.

Element type	Construct	Modify	Deconstruct
" "><!ENT ITY lineFeed "
"><!EN TITY vTab "&#x000B;"><!EN TITY boxBreak ""><!EN TITY hardReturn "
"><!EN TITY so "&#x000E;"><!ENT ITY flexSpace "&#x000F;"><!ENT ITY dle "&#x0010;"><!ENT ITY dcOne "&#x0011;"><!ENT ITY dcTwo "&#x0012;"><!ENT ITY dcThree "&#x0013;"><!ENT ITY dcFour "&#x0014;"><!ENT ITY nak "&#x0015;"><!ENT ITY syn "&#x0016;"><!ENT ITY etb "&#x0017;"><!ENT ITY can "&#x0018;"><!ENT ITY em "&#x0019;"><!ENT ITY sub "&#x001A;"><!EN TITY esc "&#x001B;"><!EN TITY csMarker			

Element type	Construct	Modify	Deconstruct
"&#x001C;"><!EN TITY discReturn "&#x001D;"><!EN TITY indentHere "&#x001E;"><!ENT ITY discHyphen ""><!EN TITY shy "&#x001F;"><!ENT ITY ensp " "><!ENT ITY emsp " "><!ENT ITY threePerEmSpac e " "><!ENT ITY fourPerEmSpace " "><!ENT ITY sixPerEmSpace " "><!ENT ITY figureSpace " "><!ENT ITY punctSpace " "><!ENT ITY thinsp " "><!ENT ITY hairSpace " "><!EN TITY zeroWidthSpace ""><!EN TITY hyphen "‐"><!ENT ITY ndash "–"><!ENT			

Element type	Construct	Modify	Deconstruct
ITY mdash "—"><!ENT ITY wordJoiner ""><!ENT ITY nbsp " "><!EN TITY ideographicSpace " ">			

ADDCELLS (Modifier schema)

Element type	Construct	Modify	Deconstruct
ADDCELLS (empty)	Not applicable.	Adds cells to an existing table. Note: If you add a column, you must also define every ROW and CELL element in that column.	Not applicable.
Attributes			
TYPE (ROW | COLUMN | HEADER | FOOTER) #REQUIRED	Not applicable.	Specifies whether to add rows, columns, headers, or footers.	Not applicable.
BASEINDEX CDATA #REQUIRED	Not applicable.	Specifies the index number of the cell before or after which the new cells should be inserted. See the INSERTPOSITION attribute.	Not applicable.

Element type	Construct	Modify	Deconstruct
INSERTCOUNT CDATA #REQUIRED	Not applicable.	Specifies how many cells to add.	Not applicable.
INSERTPOSITION (AFTER | BEFORE) "AFTER"	Not applicable.	Specifies whether to add the new cells before or after the cell indicated in the BASEINDEX attribute.	Not applicable.
KEEPATTRIBUTE (true | false) "false"	Not applicable.	Specifies whether an inserted row or column should adopt the same attributes as the BASEINDEX cell.	Not applicable.

ALIGNHORSETTINGS (Modifier schema)

Element type	Construct	Modify	Deconstruct
ALIGNHORSETTINGS (empty)	Specifies horizontal alignment settings for a callout.	Specifies horizontal alignment settings for a callout.	Specifies horizontal alignment settings for a callout.
Attributes			
ALIGNRELATIVETO (ANCHOR | PARAGRAPH | BOXCELL | PAGE | SPREAD) #IMPLIED	Indicates what a callout should be aligned relative to.	Indicates what a callout should be aligned relative to.	Indicates what a callout should be aligned relative to.
ALIGNCALLOUTS (CENTER | LEFTEDGE | RIGHTEDGE | INSIDEEDGE | OUTSIDEEDGE)	Indicates which part of a callout to align.	Indicates which part of a callout to align.	Indicates which part of a callout to align.

Element type	Construct	Modify	Deconstruct
#IMPLIED			
ALIGNWITH (ANCHORPOSITION | LEFTEDGE | RIGHTEDGE | INSIDEEDGE | OUTSIDEEDGE | CENTER | CENTERMARGIN | LEFTMARGIN | RIGHTMARGIN | INSIDEMARGIN | OUTSIDEMARGIN) #IMPLIED	Indicates how a callout should be aligned.	Indicates how a callout should be aligned.	Indicates how a callout should be aligned.
OFFSET CDATA #IMPLIED	Indicates a callout's horizontal offset.	Indicates a callout's horizontal offset.	Indicates a callout's horizontal offset.

ALIGNVERSETTINGS (Modifier schema)

Element type	Construct	Modify	Deconstruct
ALIGNVERSETTINGS (empty)	Specifies vertical alignment settings for a callout.	Specifies vertical alignment settings for a callout.	Specifies vertical alignment settings for a callout.
Attributes			
ALIGNRELATIVETO (ANCHOR | PARAGRAPH | BOXCELL | PAGE | SPREAD) #IMPLIED	Indicates what a callout should be aligned relative to.	Indicates what a callout should be aligned relative to.	Indicates what a callout should be aligned relative to.
ALIGNCALLOUTS (CENTER | TOPEDGE | BOTTOMEDGE)	Indicates which part of a callout to align.	Indicates which part of a callout to align.	Indicates which part of a callout to align.

Element type	Construct	Modify	Deconstruct
#IMPLIED			
ALIGNWITH (ANCHORPOSITION | CENTER | TOPEDGE | BOTTOMEDGE | CENTERMARGIN | TOPMARGIN | BOTTOMMARGIN | TOP | BOTTOM | TEXTASCENT | TEXTBASELINE | EMBOXTOP | EMBOXBOTTOM) #IMPLIED	Indicates how a callout should be aligned.	Indicates how a callout should be aligned.	Indicates how a callout should be aligned.
OFFSET CDATA #IMPLIED	Indicates a callout's vertical offset.	Indicates a callout's vertical offset.	Indicates a callout's vertical offset.

ALLOWBOXOFFPAGE (Modifier schema)

Element type	Construct	Modify	Deconstruct
ALLOWBOXOFFPAGE (#PCDATA)	Not applicable.	Specifies whether a box is allowed to be moved completely off of a page and onto the pasteboard by, for example, a MOVERIGHT element.Only accepts true or false values; default value is true.	Not applicable.

ALLOWBOXONTOPASTEBOARD (Modifier schema)

Element type	Construct	Modify	Deconstruct
ALLOWBOXONTOPASTEBOARD (#PCDATA)	Not applicable.	Specifies whether a box is allowed to be moved partially off of a page and onto the pasteboard by, for example, a MOVERIGHT element.Only accepts true or false values; default value is true.	Not applicable.

ANCHOREDBOXREF (Modifier schema)

Element type	Construct	Modify	Deconstruct
ANCHOREDBOXREF (#PCDATA)	Specifies id of anchored box that is part of the story.	Specifies id of anchored box that is part of the story.	Specifies id of anchored box that is part of the story.
Attributes			
ALIGNWITHTEXT (ASCENT | BASELINE) "BASELINE"	Determines whether the top of the anchored box will align with the top of the text (ascent) or the bottom of the text (baseline). Note that if you want to anchor the table and have it continue in the next box or column, this	Determines whether the top of the anchored box will align with the top of the text (ascent) or the bottom of the text (baseline). Note that if you want to anchor the table and have it continue in the next box or column, this value must be set to BASELINE.The alignwithtext	Determines whether the top of the anchored box will align with the top of the text (ascent) or the bottom of the text (baseline). Note that if you want to anchor the table and have it continue in the next box or column, this value must be set to

Element type	Construct	Modify	Deconstruct
	value must be set to BASELINE.	attribute should be set to baseline when the breakwhenanchored attribute is set to true for an anchored table.	BASELINE.
OFFSET CDATA #IMPLIED	Determines the offset when ALIGNWITHTEXT is set to BASELINE. Default is 0.	Determines the offset when ALIGNWITHTEXT is set to BASELINE. Default is 0.	Determines the offset when ALIGNWITHTEXT is set to BASELINE. Default is 0.
BASELINESHIFT CDATA #IMPLIED	Shifts the anchored box up or down without affecting paragraph line spacing. A positive value raises the anchored box, and a negative value lowers it.	Shifts the anchored box up or down without affecting paragraph line spacing. A positive value raises the anchored box, and a negative value lowers it.	The baseline shift value applied to the anchored box.
TRACKAMOUNT CDATA #IMPLIED	Adjusts the amount of space between the anchored box and surrounding characters and words.	Adjusts the amount of space between the anchored box and surrounding characters and words.	The tracking value applied to the anchored box and surrounding characters and words.
SENDING CDATA #IMPLIED	A character spacing attribute used in East Asian typography. Similar to	A character spacing attribute used in East Asian typography. Similar to kerning, but applicable as a fixed value over a	A character spacing attribute used in East Asian typography. Similar to kerning, but applicable as

Element type	Construct	Modify	Deconstruct
	kerning, but applicable as a fixed value over a range of text containing the anchored box.	range of text containing the anchored box.	a fixed value over a range of text containing the anchored box.
BOXUID CDATA #IMPLIED	The ID of the anchored box.	The ID of the anchored box.	The ID of the anchored box.

ARTICLE (Modifier schema)

Element type	Construct	Modify	Deconstruct
ARTICLE (ID, RGBCOLOR?, COMPONENT+)	Describes an article (a series of one or more COMPONENT elements).New articles should not be created in a QuarkXPress project in systems working directly with QPS. Instead, create an article only within a QuarkCopyDesk ® file. To assign an article in QPS®, use the QPS SDK.	Describes an article (a series of one or more COMPONENT elements).	Describes an article (a series of one or more COMPONENT elements).
Attributes			
OPERATION (CREATE | DELETE) #IMPLIEDDOCFO	Describes the type of Article.	Not applicable.	Describes the type of Article. “LIGHTWEIGHT” and

Element type	Construct	Modify	Deconstruct
RMAT (LIGHTWEIGHT | FULLFEATURED) “LIGHTWEIGHT"EXPORTARTICLE (true | false) “false"ARTICLETEMPLATENAME CDATA #IMPLIED			“FULLFEATURED" articles are forms of QuarkCopyDesk articles that can be constructed/modified through QuarkXPress Server.

AUTHOR (Modifier schema)

Element type	Construct	Modify	Deconstruct
AUTHOR (#PCDATA)	Not applicable.	Part of the <EBOOKMETADATA> element. Specifies the author of an e-book.	Specifies the author of an e-book.

BNSTYLE (Modifier schema)

Element type	Construct	Modify	Deconstruct
BNSTYLE (empty)	Describes a bullet, numbering, or outline style	BNSTYLE element typeDescribes a bullet, numbering, or outline style	BNSTYLE element typeDescribes a bullet, numbering, or outline style
Attributes			
TYPE (BULLET | NUMBERING | OUTLINE) #IMPLIED	Specifies whether this is a bullet, numbering, or outline style.	Specifies whether this is a bullet, numbering, or outline style.	Specifies whether LOCKTOGRID is enabled.
NAME CDATA #IMPLIED	Specifies the name of the style.	Specifies the name of the style.	Specifies the name of the style.

Element type	Construct	Modify	Deconstruct
MINDISTFROMTEXT CDATA #IMPLIED	Specifies the minimum distance between the bullet or number and the text.	Specifies the minimum distance between the bullet or number and the text.	Specifies the minimum distance between the bullet or number and the text.
RESTARTNUMBERING (true | false) #IMPLIED	Specifies whether renumbering should restart with this style.	Specifies whether renumbering should restart with this style.	Specifies whether renumbering should restart with this style.
STARTAT CDATA #IMPLIED	Specifies the first number for numbering.	Specifies the first number for numbering.	Specifies the first number for numbering.

BORDER (Modifier schema)

Element type	Construct	Modify	Deconstruct
BORDER (empty)	Describes the borders of a box. Gives you the ability to apply a different border to each	Describes a grid line on the bottom edge of a row in an <INLINETABLE>.	Not applicable.
Attributes for Width ➡ This attribute is specified in points as a floating point value.			
BORDERTOPWIDTH CDATA #IMPLIED	Specifies the thickness of the top border of the box.	Specifies the thickness of the top border of the box.	Not applicable.
BORDERLEFTWIDTH CDATA #IMPLIED	Specifies the thickness of the left border of the box.	Specifies the thickness of the left border of the box.	Not applicable.
BORDERRIG	Specifies the thickness	Specifies the	Not applicable.

Element type	Construct	Modify	Deconstruct
HTWIDTH CDATA #IMPLIED	of the right border of the box.	thickness of the right border of the box.	
BORDERBOTTOMWIDTH CDATA #IMPLIED	Specifies the thickness of the bottom border of the box.	Specifies the thickness of the bottom border of the box.	Not applicable.
Attributes forStyle			
BORDERTOPSTYLE CDATA #IMPLIED	Specifies the style of the top border of a box (i.e. Dashes, Stripes etc.).	Specifies the style of the top border of a box (i.e. Dashes, Stripes etc.).	Not applicable.
BORDERLEFTSTYLE CDATA #IMPLIED	Specifies the style of the left border of a box (i.e. Dashes, Stripes etc.).	Specifies the style of the left border of a box (i.e. Dashes, Stripes etc.).	Not applicable.
BORDERIGHTSTYLE CDATA #IMPLIED	Specifies the style of the right border of a box (i.e. Dashes, Stripes etc.).	Specifies the style of the right border of a box (i.e. Dashes, Stripes etc.).	Not applicable.
BORDERBOTTOMSTYLE CDATA #IMPLIED	Specifies the style of the bottom border of a box (i.e. Dashes, Stripes etc.).	Specifies the style of the bottom border of a box (i.e. Dashes, Stripes etc.).	Not applicable.
Attributes for Color			

➡ Only the name of a color is included in these attributes. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.

Element type	Construct	Modify	Deconstruct
BORDERTOPCOLOR CDATA #IMPLIED	Identifies the color of the top border of a box.	Identifies the color of the top border of a box.	Not applicable.
BORDERLEFTCOLOR CDATA #IMPLIED	Identifies the color of the left border of a box.	Identifies the color of the left border of a box.	
BORDERRIGHTPCOLOR CDATA #IMPLIED	Identifies the color of the right border of a box.	Identifies the color of the right border of a box.	
BORDERBOTTOMCOLOR CDATA #IMPLIED	Identifies the color of the bottom border of a box.	Identifies the color of the bottom border of a box.	
Attributes for Shade			
➡ This attribute is expressed as an integer percentage from 0 to 100.			
BORDERTOPSHADE CDATA #IMPLIED	Specifies the shade of the color applied to the top border of a box.	Specifies the shade of the color applied to the top border of a box.	
BORDERLEFTSHADE CDATA #IMPLIED	Specifies the shade of the color applied to the left border of a box.	Specifies the shade of the color applied to the left border of a box.	

Element type	Construct	Modify	Deconstruct
BORDERRIGHTSHADE CDATA #IMPLIED	Specifies the shade of the color applied to the right border of a box.	Specifies the shade of the color applied to the right border of a box.	
BORDERBOTTOMSHADE CDATA #IMPLIED	Specifies the shade of the color applied to the bottom border of a box.	Specifies the shade of the color applied to the bottom border of a box.	
Attributes for Opacity			
➡ This attribute is expressed as an integer percentage from 0 to 100.			
BORDERTOPOPACITY CDATA #IMPLIED	Specifies the opacity of the top border of a box.	Specifies the opacity of the top border of a box.	
BORDERLEFTOPACITY CDATA #IMPLIED	Specifies the opacity of theleft border of a box.	Specifies the opacity of the left border of a box.	
BORDERRIGHTOPACITY CDATA #IMPLIED	Specifies the opacity of the right border of a box.	Specifies the opacity of the right border of a box.	
BORDERBOTTOMOPACITY CDATA #IMPLIED	Specifies the opacity of the bottom border of a box.	Specifies the opacity of the bottom border of a box.	
Attributes for Blend Mode			

Element type	Construct	Modify	Deconstruct
BORDERTOPBLENDMODE CDATA #IMPLIED	Specifies the blend mode of the top border of a box	Specifies the blend mode of the top border of a box	
BORDERLEFTBLENDMODE CDATA #IMPLIED	Specifies the blend mode of the left border of a box	Specifies the blend mode of the left border of a box	
BORDERRIGHTBLENDMODE CDATA #IMPLIED	Specifies the blend mode of the right border of a box	Specifies the blend mode of the right border of a box	
BORDERBOTTOMLENDMODE CDATA #IMPLIED	Specifies the blend mode of the bottom border of a box	Specifies the blend mode of the bottom border of a box	
Attributes for Gap Color			
➡ Only the name of a color is included in these attributes. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.			
BORDERTOPGAPCOLOR	Identifies the color of a border gap for the top border of a box.	Identifies the color of a border gap for the top border of a	

Element type	Construct	Modify	Deconstruct
CDATA #IMPLIED		box.	
BORDERLEFTGAPCOLOR CDATA #IMPLIED	Identifies the color of a border gap for the left border of a box.	Identifies the color of a border gap for the left border of a box.	
BORDERRIGHTGAPCOLOR CDATA #IMPLIED	Identifies the color of a border gap for the right border of a box.	Identifies the color of a border gap for the right border of a box.	
BORDERBOTTOMGAPCOLOR CDATA #IMPLIED	Identifies the color of a border gap for the bottom border of a box.	Identifies the color of a border gap for the bottom border of a box.	
Attributes for Gap Shade ➡ This attribute is expressed as an integer percentage from 0 to 100.			
BORDERTOPGAPSHADE CDATA #IMPLIED	Identifies the shade of a border gap for the top border of a box.	Identifies the shade of a border gap for the top border of a box.	
BORDERLEFTGAPSHADE CDATA #IMPLIED	Identifies the shade of a border gap for the left border of a box.	Identifies the shade of a border gap for the left border of a box.	

Element type	Construct	Modify	Deconstruct
BORDERRIGHTGAPSHADE CDATA #IMPLIED	Identifies the shade of a border gap for the right border of a box.	Identifies the shade of a border gap for the right border of a box.	
BORDERBOTTOMGAPSHADE CDATA #IMPLIED	Identifies the shade of a border gap for the bottom border of a box.	Identifies the shade of a border gap for the bottom border of a box.	
Attributes for Gap Opacity ➡ This attribute is expressed as an integer percentage from 0 to 100.			
BORDERTOPGAPOPACITY CDATA #IMPLIED	Identifies the opacity of a border gap for the top border of a box.	Identifies the opacity of a border gap for the top border of a box.	
BORDERLEFTGAPOPACITY CDATA #IMPLIED	Identifies the opacity of a border gap for the left border of a box.	Identifies the opacity of a border gap for the left border of a box.	
BORDERRIGHTGAPOPACITY CDATA #IMPLIED	Identifies the opacity of a border gap for the right border of a box.	Identifies the opacity of a border gap for the right border of a box.	
BORDERBOTTOMGAPOPACITY	Identifies the opacity of a border gap for the bottom border of a box.	Identifies the opacity of a border gap for the bottom	

Element type	Construct	Modify	Deconstruct
CDATA #IMPLIED		border of a box.	
Attributes for Gap Blend Mode			
BORDERTOPGAPBLENDMODE CDATA #IMPLIED	Identifies the blend mode of a border gap for the top border of a box.	Identifies the blend mode of a border gap for the top border of a box.	
BORDERLEFTGAPBLENDMODE CDATA #IMPLIED	Identifies the blend mode of a border gap for the left border of a box.	Identifies the blend mode of a border gap for the left border of a box.	
BORDERRIGHTGAPBLENDMODE CDATA #IMPLIED	Identifies the blend mode of a border gap for the right border of a box.	Identifies the blend mode of a border gap for the right border of a box.	
BORDERBOTTOMGAPBLENDMODE CDATA #IMPLIED	Identifies the blend mode of a border gap for the bottom border of a box.	Identifies the blend mode of a border gap for the bottom border of a box.	

BOTTOM (Modifier schema)

Element type	Construct	Modify	Deconstruct
BOTTOM (#PCDATA)	The distance between the box	The distance between the box or	The distance between the box

Element type	Construct	Modify	Deconstruct
	or line's bottom edge and the bottom of the page, in points.	line's bottom edge and the bottom of the page, in points.	or line's bottom edge and the bottom of the page, in points.

BOTTOMGRID (Modifier schema)

Element type	Construct	Modify	Deconstruct
BOTTOMGRID (empty)	Describes a grid line on the bottom edge of a row in an <INLINETABLE>.	Describes a grid line on the bottom edge of a row in an <INLINETABLE>.	Not applicable.
Attributes			
TYPE (TOP | LEFT | BOTTOM | RIGHT) #IMPLIED	Specifies the location of the grid line.	Specifies the location of the grid line.	Not applicable.
STYLE CDATA #IMPLIED	Identifies the <TABLESTYLE> that styles this grid line. If you specify this value, you do not have to specify the remaining attributes. If you specify the remaining attributes, those attribute values override the corresponding <TABLESTYLE> values.	Identifies the <TABLESTYLE> that styles this grid line. If you specify this value, you do not have to specify the remaining attributes. If you specify the remaining attributes, those attribute values override the corresponding <TABLESTYLE> values.	Not applicable.
WIDTH CDATA	Specifies the	Specifies the width of	Not applicable.

Element type	Construct	Modify	Deconstruct
#IMPLIED	width of the grid line in points.	the grid line in points.	
COLOR CDATA #IMPLIED	Specifies the color of the grid line.	Specifies the color of the grid line.	Not applicable.
SHADE CDATA #IMPLIED	Specifies the shade of the grid line.	Specifies the shade of the grid line.	Not applicable.
OPACITY CDATA #IMPLIED	Specifies the opacity of the grid line.	Specifies the opacity of the grid line.	Not applicable.
GAPCOLOR CDATA #IMPLIED	Specifies the color of the gap (if any) between the lines that make up the grid line.	Specifies the color of the gap (if any) between the lines that make up the grid line.	Not applicable.
GAPSHADE CDATA #IMPLIED	Specifies the shade of the gap (if any) between the lines that make up the grid line.	Specifies the shade of the gap (if any) between the lines that make up the grid line.	Not applicable.
GAPOPACITY CDATA #IMPLIED	Specifies the opacity of the gap (if any) between the lines that make up the grid line.	Specifies the opacity of the gap (if any) between the lines that make up the grid line.	Not applicable.

BOX (Modifier schema)

Element type	Construct	Modify	Deconstruct
BOX (ID, METADATA?, (TEXT | PICTURE	Describes a text box or picture box. Note: On	Identifies a text box or picture box to be modified. You can	Describes a text box or picture box.

Element type	Construct	Modify	Deconstruct
| GEOMETRY | CONTENT | SHADOW | FRAME | BORDER | PLACEHOLDER | CONTENTPH | STATICCONTENT)*, INTERACTIVITY?)	construct, you must provide a box name in the ID@NAME attribute; QuarkXPress Server assigns an ID@UID to each BOX you create. Note: When a box is created, its page number is inferred from the GEOMETRY@PAGE attribute.	use either the ID@UID or ID@NAME value to identify the box. Note: Named boxes can be easily identified by an XPath search for //BOX[@NAME]).	If a NAME value exists, the NAME displays in the content of the ID element: <ID UID=456NAME=Name of box>Name of box</ID>If a NAME value does not exist, the UID displays in the content of the ID element: <ID UID=457>457</ID>
Attributes			
AUTOMATICTEXTBOX (true|false)	Indicates that an automatic text box should be created.	Indicates that an automatic text box should be created.	Indicates that an automatic text box should be created.
MASTERBOX	Refer to the Master Page box.	Refer to the Master Page box.	Refer to the Master Page box.
OPERATION (CREATE | DELETE) #IMPLIED	No applicable.	Specifies whether to create or delete the indicated box.	Not applicable.
BOXTYPE (CT_NONE | CT_TEXT | CT_PICT) #IMPLIED	The box type:CT_NONE = No box type specified.CT_TEXT = Text boxCT_PICT = Picture box	The box type:CT_NONE = No box type specified.CT_TEXT = Text boxCT_PICT = Picture box	The box type:CT_NONE = No box type specified.CT_TEXT = Text boxCT_PICT = Picture box
COLOR CDATA #IMPLIED	Identifies the background	Identifies the background color of	Identifies the background color

Element type	Construct	Modify	Deconstruct
	color of a box. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the project's Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	a box. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the project's Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server. The color definition can also be based on an existing color created and saved in the project.	of a box. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the project's Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server. The color definition can also be based on an existing color created and saved in the project.
SHADE CDATA #IMPLIED	Specifies the shade of a box's background color, specified as a float value from 0 to 100.	Specifies the shade of a box's background color, specified as an integer percentage from 0 to 100.	Specifies the shade of a box's background color, specified as an integer percentage from 0 to 100.
OPACITY CDATA #IMPLIED	Specifies the opacity of a box's background color, specified as a float value from 0 to 100.	Specifies the opacity of a box's background color, specified as an integer percentage from 0 to 100.	Indicates the opacity of a box's background color, specified as an integer percentage from 0 to 100.
BLENDSTYLE (SOLID | LINEAR | MIDLINEAR |	Specifies the type of blend applied to this	Specifies the type of blend applied to this box (linear, circular,	Specifies the type of blend applied to this box (linear,

Element type	Construct	Modify	Deconstruct
RECTANGULAR | DIAMOND | CIRCULAR | FULLCIRCULAR | none) “none”	box (linear, circular, rectangular, etc.).	rectangular, etc.).	circular, rectangular, etc.).
BLENDANGLE CDATA #IMPLIED	Specifies the angle of the blend.	Specifies the angle of the blend.	Specifies the angle of the blend.
BLENDCOLOR CDATA #IMPLIED	Specifies the second color of the blend. The first color of the blend is the color applied to the box.	Specifies the second color of the blend. The first color of the blend is the color applied to the box.	Specifies the second color of the blend. The first color of the blend is the color applied to the box.
BLENDSHADE CDATA #IMPLIED	Specifies the shade applied to the second color of the blend. The shade of the first color of the blend is the shade of the color applied to the box.	Specifies the shade applied to the second color of the blend. The shade of the first color of the blend is the shade of the color applied to the box.	Specifies the shade applied to the second color of the blend. The shade of the first color of the blend is the shade of the color applied to the box.
BLENDOPACITY CDATA #IMPLIED	Specifies the opacity applied to the second color of the blend. The opacity of the first color of the blend is the opacity of the color applied to the box.	Specifies the opacity applied to the second color of the blend. The opacity of the first color of the blend is the opacity of the color applied to the box.	Specifies the opacity applied to the second color of the blend. The opacity of the first color of the blend is the opacity of the color applied to the box.
ANCHOREDIN	Not applicable.	Not applicable.	Indicates an

Element type	Construct	Modify	Deconstruct
CDATA #IMPLIED			anchored box and identifies its parent box.
ANCHOREDGROUPMEMBER CDATA #IMPLIED	Specifies that this box is a member of the indicated anchored group.	Specifies that this box is a member of the indicated anchored group.	Specifies that this box is a member of the indicated anchored group.
HYPERLINKREF CDATA #IMPLIED	Not applicable.	Not applicable.	Specifies that this box is a hyperlink by referring to a HYPERLINK.
HLTYPE (WWWURL | PAGE | ANCHOR) #IMPLIED	Not applicable.	Not applicable.	Specifies the type of hyperlink this box hyperlinks to. Options include WWWURL (a URL on the Web), PAGE (the top of a page in the same layout), and ANCHOR (an anchor).WWWURL is global to all layouts in the project and is specified directly under the project node.
HLANCHORREF CDATA #IMPLIED	Not applicable.	Not applicable.	If this box is a hyperlink of the HLTYPE ANCHOR, this attribute identifies the anchor by name.
ADDTOREFLOW (true | false)	Not applicable.	If true, adds this box to the project's	Not applicable.

Element type	Construct	Modify	Deconstruct
#IMPLIED		reflow article. Equivalent to the Digital Publishing > Add to Reflow command in QuarkXPress.	
ARTICLENAME CDATA #IMPLIED	Not applicable.	Specifies the name of the project's reflow article (to which this box is being added as a component). If no reflow article exists and you do not include this attribute, the default reflow article name is used.	Not applicable.

BOXATTRIBUTE (Modifier schema)

Element type	Construct	Modify	Deconstruct
BOXATTRIBUTE (empty)	Specifies the attributes of a box created with the INLINEBOX element type.	Specifies the attributes of a box created with the INLINEBOX element type.	Not applicable.
Attributes			
COLOR CDATA #IMPLIED	Identifies the background color of a box. Note: Only the name of a color is included in this attribute. The definition of the color is	Identifies the background color of a box. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the project's Job Jackets	Not applicable.

Element type	Construct	Modify	Deconstruct
	stored in the project's Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	file or defined using the Document Controls submenu in QuarkXPress Server.	
SHADE CDATA #IMPLIED	Specifies the shade of a box's background color, specified as a float value from 0 to 100.	Specifies the shade of a box's background color, specified as a float value from 0 to 100.	Not applicable.
OPACITY CDATA #IMPLIED	Specifies the opacity of a box's background color, specified as a float value from 0 to 100.	Specifies the opacity of a box's background color, specified as a float value from 0 to 100.	Not applicable.
BLENDSTYLE (SOLID | LINEAR | MIDLINEAR | RECTANGULAR | DIAMOND | CIRCULAR | FULLCIRCULAR | none) “none”	Specifies the type of blend applied to this box (linear, circular, rectangular, etc.).	Specifies the type of blend applied to this box (linear, circular, rectangular, etc.).	Not applicable.
BLENDANGLE CDATA #IMPLIED	Specifies the angle of the blend.	Specifies the angle of the blend.	Not applicable.
BLENDCOLOR CDATA #IMPLIED	Specifies the second color of the blend. The first color of the	Specifies the second color of the blend. The first color of the blend is the color	Not applicable.

Element type	Construct	Modify	Deconstruct
	blend is the color applied to the box.	applied to the box.	
BLENDSHADE CDATA #IMPLIED	Specifies the shade applied to the second color of the blend. The shade of the first color of the blend is the shade of the color applied to the box.	Specifies the shade applied to the second color of the blend. The shade of the first color of the blend is the shade of the color applied to the box.	Not applicable.
BLENDOPACITY CDATA #IMPLIED	Specifies the opacity applied to the second color of the blend. The opacity of the first color of the blend is the opacity of the color applied to the box.	Specifies the opacity applied to the second color of the blend. The opacity of the first color of the blend is the opacity of the color applied to the box.	Not applicable.
BLENDMODE (NORMAL|MULTIPLY | SCREEN|OVERLAY | DARKEN | LIGHTEN | COLORDODGE| COLORBURN | HARDLIGHT | SOFTLIGHT | DIFFERENCE | EXCLUSION) #IMPLIED	Specifies the type of blendmode applied to the box.	Specifies the type of blendmode applied to the box.	Specifies the type of blendmode applied to the box.

Element type	Construct	Modify	Deconstruct
CORNERRADIUS CDATA #IMPLIED	Specifies the radius of the corner style of the box.	Specifies the radius of the corner style of the box.	Not applicable.
CORNERSTYLE (ROUNDED | CONCAVE | RECTANGLE | BEVELED) #IMPLIED	Specifies the style of the corners of the box.	Specifies the style of the corners of the box.	Not applicable.
ANGLE CDATA #IMPLIED	Specifies a rotation angle for a box as a floating-point value between -360 degrees and 360 degrees.	Specifies a rotation angle for a box as a floating-point value between -360 degrees and 360 degrees.	Not applicable.

BOXREF (Modifier schema)

Element type	Construct	Modify	Deconstruct
BOXREF (empty)	Identifies a box that is a member of a <GROUP>.	Identifies a box that is a member of a <GROUP>.	Identifies a box that is a member of a <GROUP>.
Attributes			
UID CDATA #IMPLIED	The ID@UID of a <BOX> that is part of a group.	The ID@UID of a <BOX> that is part of a group.	The ID@UID of a <BOX> that is part of a group.
NAME CDATA #IMPLIED	The ID@NAME of a <BOX> that is part of a group.	The ID@NAME of a <BOX> that is part of a group.	The ID@NAME of a <BOX> that is part of a group.
OPERATION (CREATE | DELETE) #IMPLIED	Creates or deletes the link that makes the target box part	Creates or deletes the link that makes the target box part of a group. Note that	Not applicable.

Element type	Construct	Modify	Deconstruct
	of a group. Note that deleting a <BOXREF> does not remove the corresponding box from the layout.	deleting a <BOXREF> does not remove the corresponding box from the layout.	

CALLOUTANCHOR (Modifier schema)

Element type	Construct	Modify	Deconstruct
CALLOUTANCHOR ((CALLOUTBOXREF? | INLINETABLE? | INLINEBOX?), ALIGNHORSETTINGS?, ALIGNVERSETTINGS?)	Identifies a callout anchor.	Identifies a callout anchor.	Identifies a callout anchor.
Attributes			
UID CDATA #IMPLIED	Unique identifier for a callout anchor.	Unique identifier for a callout anchor.	Unique identifier for a callout anchor.
CALLOUTSTYLE CDATA #IMPLIED	Identifies the callout style to be associated with a callout anchor.	Identifies the callout style to be associated with a callout anchor.	Identifies the callout style associated with a callout anchor.
OPERATION (RELEASE | DELETE) #IMPLIED	Lets you release a callout anchor's callout or delete the callout anchor.	Lets you release a callout anchor's callout or delete the callout anchor.	Not applicable.
ALLOWMANUAL	Indicates	Indicates whether	Indicates whether

Element type	Construct	Modify	Deconstruct
POS (true | false) #IMPLIED	whether the callout associated with this callout anchor can be manually repositioned.	the callout associated with this callout anchor can be manually repositioned.	the callout associated with this callout anchor can be manually repositioned.
KEEPWITHINMARGINS (true | false) #IMPLIED	Indicates whether the callout associated with this callout anchor can be positioned outside the margin.	Indicates whether the callout associated with this callout anchor can be positioned outside the margin.	Indicates whether the callout associated with this callout anchor can be positioned outside the margin.
VERTICALPADDING	If TILINGDIRECTION = VERTICAL, the vertical padding specified signifies the spacing to be maintained between two callout boxes. This value will be used during the time of automatic stacking of callouts in the event when they attempt to overlap upon each other as a result of	If TILINGDIRECTION = VERTICAL, the vertical padding specified signifies the spacing to be maintained between two callout boxes. This value will be used during the time of automatic stacking of callouts in the event when they attempt to overlap upon each other as a result of application of same callout style on them.	If TILINGDIRECTION = VERTICAL, the vertical padding specified signifies the spacing to be maintained between two callout boxes. This value will be used during the time of automatic stacking of callouts in the event when they attempt to overlap upon each other as a result of application of same callout style on them.

Element type	Construct	Modify	Deconstruct
	application of same callout style on them.		
HORIZONTALPADDING	If TILINGDIRECTION = HORIZONTAL, the horizontal padding value specified signifies the spacing to be maintained between two the callout boxes. This value will be used during the time of automatic stacking of callouts in the event when they attempt to overlap upon each other as a result of application of same callout style on them.	If TILINGDIRECTION = HORIZONTAL, the horizontal padding value specified signifies the spacing to be maintained between two the callout boxes. This value will be used during the time of automatic stacking of callouts in the event when they attempt to overlap upon each other as a result of application of same callout style on them.	If TILINGDIRECTION = HORIZONTAL, the horizontal padding value specified signifies the spacing to be maintained between two the callout boxes. This value will be used during the time of automatic stacking of callouts in the event when they attempt to overlap upon each other as a result of application of same callout style on them.

CALLOUTBOXREF (Modifier schema)

Element type	Construct	Modify	Deconstruct
CALLOUTBOXREF (empty)	Identifies an item or group as a callout.	Identifies an item or group as a callout.	Identifies an item or group as a callout.
Attributes			

Element type	Construct	Modify	Deconstruct
UID CDATA #IMPLIED	Unique identifier for a callout box reference.	Unique identifier for a callout box reference.	Unique identifier for a callout box reference.
NAME CDATA #IMPLIED	Name for a callout box reference.	Name for a callout box reference.	Name for a callout box reference.

CELL (Modifier schema)

Element type	Construct	Modify	Deconstruct
CELL (ID? (CONTENT| CONTENTPH | TEXT | PICTURE | PLACEHOLDER)*)	Describes a table cell.	Describes a table cell.	Describes a table cell.
Attributes			
COLUMNCOUNT CDATA #REQUIRED	Specifies the column index position of a cell, with the first cell being cell 1.	Specifies the column index position of a cell, with the first cell being cell 1.	Specifies the column index position of a cell, with the first cell being cell 1.
BOXTYPE (CT_NONE | CT_TEXT | CT_PICT) #IMPLIED	Specifies a cells type:CT_NONE = No-content cellCT_TEXT = Text cellCT_PICT = Picture cell	Specifies a cells type:CT_NONE = No-content cellCT_TEXT = Text cellCT_PICT = Picture cell	Specifies a cells type:CT_NONE = No-content cellCT_TEXT = Text cellCT_PICT = Picture cell
COLOR CDATA #IMPLIED	Identifies the color of a cell. Note: Only the name of a color is included in this attribute. The definition of the color is	Identifies the color of a cell. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets	Identifies the color of a cell. Note: Only the name of a color is included in this attribute. The definition of the color is stored in

Element type	Construct	Modify	Deconstruct
	stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	file or defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.	the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.
SHADE CDATA #IMPLIED	Specifies the shade of the color applied to a cell, as an integer percentage from 0 to 100.	Specifies the shade of the color applied to a cell, as an integer percentage from 0 to 100.	Specifies the shade of the color applied to a cell, as an integer percentage from 0 to 100.
OPACITY CDATA #IMPLIED	Specifies the opacity of the color applied to a cell, specified as an integer percentage from 0 to 100.	Specifies the opacity of the color applied to a cell, specified as an integer percentage from 0 to 100.	Specifies the opacity of the color applied to a cell, specified as an integer percentage from 0 to 100.
BLENDSTYLE (SOLID | LINEAR | MIDLINEAR | RECTANGULAR | DIAMOND | CIRCULAR | FULLCIRCULAR | none) “none”	Specifies the type of blend applied to this cell (linear, circular, rectangular, etc.).	Specifies the type of blend applied to this cell (linear, circular, rectangular, etc.).	Specifies the type of blend applied to this cell (linear, circular, rectangular, etc.).
BLENDANGLE CDATA #IMPLIED	Specifies the angle of the blend.	Specifies the angle of the blend.	Specifies the angle of the blend.

Element type	Construct	Modify	Deconstruct
BLENDCOLOR CDATA #IMPLIED	Specifies the second color of the blend. The first color of the blend is the color applied to the cell, as in QuarkXPress.	Specifies the second color of the blend. The first color of the blend is the color applied to the cell, as in QuarkXPress.	Specifies the second color of the blend. The first color of the blend is the color applied to the cell, as in QuarkXPress.
BLENDSHADE CDATA #IMPLIED	Specifies the shade applied to the second color of the blend. The shade of the first color of the blend is the shade of the color applied to the cell.	Specifies the shade applied to the second color of the blend. The shade of the first color of the blend is the shade of the color applied to the cell.	Specifies the shade applied to the second color of the blend. The shade of the first color of the blend is the shade of the color applied to the cell.
BLENDOPACITY CDATA #IMPLIED	Specifies the opacity applied to the second color of the blend. The opacity of the first color of the blend is the opacity of the color applied to the cell.	Specifies the opacity applied to the second color of the blend. The opacity of the first color of the blend is the opacity of the color applied to the cell.	Specifies the opacity applied to the second color of the blend. The opacity of the first color of the blend is the opacity of the color applied to the cell.
MERGEROWSPAN CDATA #IMPLIED	Attribute used for merging cells and rows.	Attribute used for merging cells and rows.	If a table includes merged cells, then the MERGECOLSPAN value is shown in the xml output.
MERGECOLSPAN CDATA #IMPLIED	Attribute used for merging cells	Attribute used for merging cells and	Not applicable.

Element type	Construct	Modify	Deconstruct
	and columns.	columns.	
SPLIT (true | false) #IMPLIED	Not applicable.	Attribute used for splitting rows and columns.	Not applicable.
ADDTOREFLOW (true | false) #IMPLIED	Not applicable.	If true, adds the contents of this cell to the project's reflow article. Equivalent to the Digital Publishing > Add to Reflow command in QuarkXPress.	Not applicable.
ARTICLENAME CDATA #IMPLIED	Not applicable.	Specifies the name of the project's reflow article (to which the contents of this cell is being added as a component). If no reflow article exists and you do not include this attribute, the default reflow article name is used.	Not applicable.

CHILDID (Modifier schema)

Element type	Construct	Modify	Deconstruct
CHILDID (empty)	Specifies a child of a parent TABLE element.	Specifies a child of a parent TABLE element.	Specifies a child of a parent TABLE element.
Attributes			
NAME CDATA #IMPLIED	Indicates the user-assigned name of the	Not applicable.	Indicates the user-assigned name of the

Element type	Construct	Modify	Deconstruct
	CHILD element of the parent table.		CHILD element of the parent table.
UID CDATA #IMPLIED	Not applicable.	Indicates the ID of the CHILD element of the parent table assigned from QuarkXPress Server.	Indicates the ID of the CHILD element of the parent table assigned from QuarkXPress Server.

CLIPPING (Modifier schema)

Element type	Construct	Modify	Deconstruct
CLIPPING (empty)	Describes a clipping path.	Describes a clipping path.	Describes a clipping path.
Attributes			
TYPE (ITEM | EMBEDDEDPATH | ALPHACHANNEL | NONWHITEAREAS | PICTUREBOUNDS) “ITEM”	Specifies the type of clipping applied to a picture item:ITEM = Runs along the edges of the item.EMBEDDEDPATH = Runs along a path embedded in the picture file.ALPHACHANNEL = Runs along an alpha channel embedded in the picture file.NONWHITEA	Specifies the type of clipping applied to a picture item:ITEM = Runs along the edges of the item.EMBEDDEDPATH = Runs along a path embedded in the picture file.ALPHACHANNEL = Runs along an alpha channel embedded in the picture file.NONWHITEAREAS = Runs along a path based on the dark and light areas of the picture file.	Specifies the type of clipping applied to a picture item:ITEM = Runs along the edges of the item.EMBEDDEDPATH = Runs along a path embedded in the picture file.ALPHACHANNEL = Runs along an alpha channel embedded in the picture file.NONWHITEAREAS = Runs along a path based on the dark and light

Element type	Construct	Modify	Deconstruct
	REAS = Runs along a path based on the dark and light areas of the picture file. See the THRESHOLD attribute.PICTUREBOUNDS = Runs along the rectangular canvas area of the picture, regardless of the size and shape of the picture box.	See the THRESHOLD attribute.PICTUREBOUNDS = Runs along the rectangular canvas area of the picture, regardless of the size and shape of the picture box.	areas of the picture file. See the THRESHOLD attribute.PICTUREBOUNDS = Runs along the rectangular canvas area of the picture, regardless of the size and shape of the picture box.
TOP CDATA #IMPLIED	Valid when CLIPPING@TYPE = ITEM or PICTUREBOUNDS. Moves the top edge of the clipping path by the specified number of points (positive=up, negative=down).	Valid when CLIPPING@TYPE = ITEM or PICTUREBOUNDS. Moves the top edge of the clipping path by the specified number of points (positive=up, negative=down).	Valid when CLIPPING@TYPE = ITEM or PICTUREBOUNDS. Moves the top edge of the clipping path by the specified number of points (positive=up, negative=down).
RIGHT CDATA #IMPLIED	Valid when CLIPPING@TYPE = ITEM or PICTUREBOUNDS. Moves the right edge of the clipping path by the specified number of	Valid when CLIPPING@TYPE = ITEM or PICTUREBOUNDS. Moves the right edge of the clipping path by the specified number of points (positive=right,	Valid when CLIPPING@TYPE = ITEM or PICTUREBOUNDS. Moves the right edge of the clipping path by the specified number of points

Element type	Construct	Modify	Deconstruct
	points (positive=right, negative=left).	negative=left).	(positive=right, negative=left).
LEFT CDATA #IMPLIED	Valid when CLIPPING@TYPE = ITEM or PICTUREBOUNDS. Moves the left edge of the clipping path by the specified number of points (positive=left, negative=right).	Valid when CLIPPING@TYPE = ITEM or PICTUREBOUNDS. Moves the left edge of the clipping path by the specified number of points (positive=left, negative=right).	Valid when CLIPPING@TYPE = ITEM or PICTUREBOUNDS. Moves the left edge of the clipping path by the specified number of points (positive=left, negative=right).
BOTTOM CDATA #IMPLIED	Valid when CLIPPING@TYPE = ITEM or PICTUREBOUNDS. Moves the bottom edge of the clipping path by the specified number of points (positive=down, negative=up).	Valid when CLIPPING@TYPE = ITEM or PICTUREBOUNDS. Moves the bottom edge of the clipping path by the specified number of points (positive=down, negative=up).	Valid when CLIPPING@TYPE = ITEM or PICTUREBOUNDS. Moves the bottom edge of the clipping path by the specified number of points (positive=down, negative=up).
PATHNAME CDATA #IMPLIED	Identifies a path embedded in a picture for use as the clipping path.	Identifies a path embedded in a picture for use as the clipping path.	Identifies a path embedded in a picture for use as the clipping path.
OUTSET CDATA #IMPLIED	Valid when CLIPPING@TYPE = EMBEDDEDPATH,	Valid when CLIPPING@TYPE = EMBEDDEDPATH, ALPHACHANNEL, or NONWHITEAREAS.	Valid when CLIPPING@TYPE = EMBEDDEDPATH, ALPHACHANNEL,

Element type	Construct	Modify	Deconstruct
	ALPHACHANNEL, or NONWHITEAREAS. Specifies a single outset or inset integer value in points to be used on all sides.	Specifies a single outset or inset integer value in points to be used on all sides.	or NONWHITEAREAS. Specifies a single outset or inset integer value in points to be used on all sides.
NOISE CDATA #IMPLIED	Valid when CLIPPING@TYPE = ALPHACHANNEL or NONWHITEAREAS. Specifies that areas smaller than this number of points should be ignored when creating a clipping path.	Valid when CLIPPING@TYPE = ALPHACHANNEL or NONWHITEAREAS. Specifies that areas smaller than this number of points should be ignored when creating a clipping path.	Valid when CLIPPING@TYPE = ALPHACHANNEL or NONWHITEAREAS. Specifies that areas smaller than this number of points should be ignored when creating a clipping path.
THRESHOLD CDATA #IMPLIED	Valid when CLIPPING@TYPE = ALPHACHANNEL or NONWHITEAREAS. Specifies the maximum integer percentage of darkness that should be considered white when creating a clipping path.	Valid when CLIPPING@TYPE = ALPHACHANNEL or NONWHITEAREAS. Specifies the maximum integer percentage of darkness that should be considered white when creating a clipping path.	Valid when CLIPPING@TYPE = ALPHACHANNEL or NONWHITEAREAS. Specifies the maximum integer percentage of darkness that should be considered white when creating a clipping path.

Element type	Construct	Modify	Deconstruct
SMOOTHNESS CDATA #IMPLIED	Valid when CLIPPING@TYPE = ALPHACHANNEL or NONWHITEAREAS. Specifies the smoothness, in points, of an automatically created clipping path.	Valid when CLIPPING@TYPE = ALPHACHANNEL or NONWHITEAREAS. Specifies the smoothness, in points, of an automatically created clipping path.	Valid when CLIPPING@TYPE = ALPHACHANNEL or NONWHITEAREAS. Specifies the smoothness, in points, of an automatically created clipping path.
OUTSIDEONLY (true | false | none) “none”	Valid when CLIPPING@TYPE = EMBEDDEDPATH, ALPHACHANNEL, or NONWHITEAREAS. Indicates that only the outer edges of the clipping path should be used.	Valid when CLIPPING@TYPE = EMBEDDEDPATH, ALPHACHANNEL, or NONWHITEAREAS. Indicates that only the outer edges of the clipping path should be used.	Valid when CLIPPING@TYPE = EMBEDDEDPATH, ALPHACHANNEL, or NONWHITEAREAS. Indicates that only the outer edges of the clipping path should be used.
RESTRICTTOBOX (true | false | none) “none”	Valid when CLIPPING@TYPE = EMBEDDEDPATH, ALPHACHANNEL, or NONWHITEAREAS. Indicates whether the clipping path is restricted to the inside of the	Valid when CLIPPING@TYPE = EMBEDDEDPATH, ALPHACHANNEL, or NONWHITEAREAS. Indicates whether the clipping path is restricted to the inside of the box.	Valid when CLIPPING@TYPE = EMBEDDEDPATH, ALPHACHANNEL, or NONWHITEAREAS. Indicates whether the clipping path is restricted to the inside of the box.

Element type	Construct	Modify	Deconstruct
	box.		
INVERT (true | false | none) "none"	Valid when CLIPPING@TYPE = EMBEDDEDPATH, ALPHACHANNEL, or NONWHITEAREAS. Reverses the shape of the clipping path.	Valid when CLIPPING@TYPE = EMBEDDEDPATH, ALPHACHANNEL, or NONWHITEAREAS. Reverses the shape of the clipping path.	Valid when CLIPPING@TYPE = EMBEDDEDPATH, ALPHACHANNEL, or NONWHITEAREAS. Reverses the shape of the clipping path.
EDITED (true | false | none) "none"	Not applicable.	Not applicable.	Indicates whether the clipping path has been manually edited in QuarkXPress.

COLGROUP (Modifier schema)

Element type	Construct	Modify	Deconstruct
COLGROUP (TCOL+)	Identifies a group of columns in an <INLINETABLE>.	Identifies a group of columns in an <INLINETABLE>.	Not applicable.

COLSPEC (Modifier schema)

Element type	Construct	Modify	Deconstruct
COLSPEC (COLUMN+)	Describes the columns in a table. Note: If the COLSPEC element is missing for a table, then the	Describes the columns in a table. Note: If the COLSPEC element is missing for a new table, then the table is created using columns of	Describes the columns in a table.

Element type	Construct	Modify	Deconstruct
	table is created using columns of equal width, based on the number of columns in the row with the most columns.	equal width, based on the number of columns in the row with the most columns.	

COLUMN (Modifier schema)

Element type	Construct	Modify	Deconstruct
COLUMN (LINE*)	Describes a column in a table.	Describes a column in a table.	Describes a column in a table.
Attributes			
COLUMNCOUNT CDATA #REQUIRED	Specifies the index position of a column beginning from the left. For example, COLUMNCOUNT = 1 indicates the first column from the left, and COLUMNCOUNT = 2 indicates the second column from the left.	Specifies the index position of a column beginning from the left. For example, COLUMNCOUNT = 1 indicates the first column from the left, and COLUMNCOUNT = 2 indicates the second column from the left.	Specifies the index position of a column beginning from the left. For example, COLUMNCOUNT = 1 indicates the first column from the left, and COLUMNCOUNT = 2 indicates the second column from the left.
COLUMNWIDTH CDATA #IMPLIED	Specifies the width of a column.	Specifies the width of a column.	Specifies the width of a column.
COLOR CDATA #IMPLIED	Identifies the color of a column.	Identifies the color of a column. Overrides the TABLE@COLOR	Identifies the color of a column. Overrides the

Element type	Construct	Modify	Deconstruct
	Overrides the TABLE@COLOR attribute. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	attribute. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.	TABLE@COLOR attribute. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.
SHADE CDATA #IMPLIED	Specifies the shade of the color applied to a column, as an integer percentage from 0 to 100.	Specifies the shade of the color applied to a column, as an integer percentage from 0 to 100.	Specifies the shade of the color applied to a column, as an integer percentage from 0 to 100.
OPACITY CDATA #IMPLIED	Specifies the opacity of the color applied to a column, specified as an integer percentage from 0 to 100.	Specifies the opacity of the color applied to a column, specified as an integer percentage from 0 to 100.	Specifies the opacity of the color applied to a column, specified as an integer percentage from 0 to 100.
MERGECOLSPAN CDATA #IMPLIED	Attribute used for merging cells, rows, and	Attribute used for merging cells, rows, and columns.	If a table includes merged cells, then the

Element type	Construct	Modify	Deconstruct
	columns.		MERGECOLSPAN value is shown in the xml output.
SPLIT (true | false) #IMPLIED	Not applicable.	Attribute used for splitting merged cells.	Not applicable.
AUTOFIT (true | false | none) "none"	Specifies whether the rows or columns will adjust size to fit the content.	Specifies whether the rows or columns will adjust size to fit the content.	Indicates whether the rows or columns will adjust size to fit the content.
AUTOFITMAXLIMIT CDATA #IMPLIED	Max limit for autofit.	Max limit for autofit.	Max limit for autofit.

COLUMNFLOW (Modifier schema)

➡ To learn more about column flow (split & span) functionality in QuarkXPress 2017, refer to the "Controlling column flow" and "Column flow examples" sections of the *QuarkXPress 2017 User Guide* found here: *http://files.quark.com/download/documentation/QuarkXPress/2017/English/QXP-2017-User-Guide-EN.pdf*. The newly introduced Modifier XML markup maps/corresponds to the QuarkXPress 2017 functionality.

Element type	Construct	Modify	Deconstruct
COLUMNFLOW (COLUMNSPAN | COLUMNSPLIT)	The COLUMNFLOW element is used to specify how text flows in relation to columns.	The COLUMNFLOW element is used to specify how text flows in relation to columns.	The COLUMNFLOW element is used to specify how text flows in relation to columns.
Attributes			
FLOWORDER	Specifies the column flow order. Allowed	Specifies the column flow order. Allowed options are	Specifies the column flow order. Allowed

Element type	Construct	Modify	Deconstruct
	options are CONTINUOUS to leave the paragraph in its original column or RESTART to move the paragraph to the start of the column block.	CONTINUOUS to leave the paragraph in its original column or RESTART to move the paragraph to the start of the column block.	options are CONTINUOUS to leave the paragraph in its original column or RESTART to move the paragraph to the start of the column block.

COLUMNSPAN (Modifier schema)

Element type	Construct	Modify	Deconstruct
COLUMNSPAN	The COLUMNSPAN element is used to specify how a paragraph spans over multiple columns.	The COLUMNSPAN element is used to specify how a paragraph spans over multiple columns.	The COLUMNSPAN element is used to specify how a paragraph spans over multiple columns.
Attributes			
COLUMNCOUNT	Specifies the number of columns you want the paragraph to span. You can choose All or a number from 2 to 30.	Specifies the number of columns you want the paragraph to span. You can choose All or a number from 2 to 30.	Specifies the number of columns you want the paragraph to span. You can choose All or a number from 2 to 30.
SPACEAFTER	Specifies the space after the paragraphs.	Specifies the space after the paragraphs.	Specifies the space after the paragraphs.
SPACEBEFORE	Specifies the	Specifies the space	Specifies the

Element type	Construct	Modify	Deconstruct
	space before the paragraphs.	before the paragraphs.	space before the paragraphs.

COLUMNSPLIT (Modifier schema)

➡ To learn more about column flow (split & span) functionality in QuarkXPress 2017, refer to the "Controlling column flow" and "Column flow examples" sections of the *QuarkXPress 2017 User Guide* found here: *http://files.quark.com/download/documentation/QuarkXPress/2017/English/QXP-2017-User-Guide-EN.pdf*. The newly introduced Modifier XML markup maps/corresponds to the QuarkXPress 2017 functionality.

Element type	Construct	Modify	Deconstruct
COLUMNSPLIT	The COLUMNSPLIT element is used to specify how a paragraph is split between multiple columns.	The COLUMNSPLIT element is used to specify how a paragraph is split between multiple columns.	The COLUMNSPLIT element is used to specify how a paragraph is split between multiple columns.
Attributes			
COLUMNCOUNT	Specifies the number of columns you want the paragraph to be split among. You can choose a number from 2 to 30.	Specifies the number of columns you want the paragraph to be split among. You can choose a number from 2 to 30.	Specifies the number of columns you want the paragraph to be split among. You can choose a number from 2 to 30.
SPACEAFTER	Specifies the space after the paragraphs.	Specifies the space after the paragraphs.	Specifies the space after the paragraphs.
SPACEBEFORE	Specifies the space before the paragraphs.	Specifies the space before the paragraphs.	Specifies the space before the paragraphs.

Element type	Construct	Modify	Deconstruct
COLUMNGAP	Specifies the gutter width for split columns.	Specifies the gutter width for split columns.	Specifies the gutter width for split columns.
COLUMNRULEENABLED (true | false)	Indicate that you want to apply a separator line between coloumns.	Indicate that you want to apply a separator line between coloumns.	Indicate that you want to apply a separator line between coloumns.
COLUMNRULESTYLE	Specifies the style of the column rule.	Specifies the style of the column rule.	Specifies the style of the column rule.
COLUMNRULEWIDTH	Specifies the width of the column rule.	Specifies the width of the column rule.	Specifies the width of the column rule.
COLUMNRULECOLOR	Specifies the color of the column rule.	Specifies the color of the column rule.	Specifies the color of the column rule.
COLUMNRULESHADE	Specifies the shade of the color of the column rule.	Specifies the shade of the color of the column rule.	Specifies the shade of the color of the column rule.
COLUMNRULEOPACITY	Specifies the opacity of the color of the column rule.	Specifies the opacity of the color of the column rule.	Specifies the opacity of the color of the column rule.

COMPONENT (Modifier schema)

Element type	Construct	Modify	Deconstruct
COMPONENT (empty)	The component(s) that make up an article. Required for ARTICLE element.	The component(s) that make up an article. Required for ARTICLE element.	The component(s) that make up an article.
Attributes			

Element type	Construct	Modify	Deconstruct
OPERATION (CREATE | DELETE) #IMPLIED	Not applicable.	Specifies whether to create or delete the specified component from the ARTICLE.	Not applicable.
NAME CDATA #IMPLIED	The name given to a specific component in an ARTICLE. Required for COMPONENT.	The name given to a specific component in an ARTICLE. Required for COMPONENT.	Specifies the name of the component in the ARTICLE.
UID CDATA #IMPLIED	QuarkXPress Server automatically assigns a unique ID to components.	The unique ID of the COMPONENT to be modified.	Specified the unique ID of the COMPONENT to be modified.
BOXNAME CDATA #IMPLIED	Specifies the name of the user-assigned box to which the COMPONENT belongs.	Specifies the name of the user-assigned box to which the COMPONENT belongs.	Specifies the name of the user-assigned box to which the COMPONENT belongs.
BOXUID CDATA #IMPLIED	Not applicable.	Specifies the ID of the QuarkXPress Server-assigned box to which the COMPONENT belongs.	Specifies the ID of the QuarkXPress Server-assigned box to which the COMPONENT belongs.
COMPONENTCLASS (CT_TEXT | CT_PICT | CT_GROUP) "CT_TEXT"	Describes whether the component resides in a text box, picture box, or group.	Describes whether the component resides in a text box, picture box, or group.	Describes whether the component resides in a text box, picture box, or group.
COMPONENTTYPE CDATA #IMPLIED	Not applicable.	Indicates the label applied to a component. Valid	Indicates the label applied to a component. Valid

Element type	Construct	Modify	Deconstruct
		only for reflow articles and QuarkCopyDesk articles. Valid values include: Body Byline FigureCaption FigureCredit Headline Headline2 IndentedParagraph Pullquote SectionChapterName Title Title2 OrderedList UnorderedList	only for reflow articles and QuarkCopyDesk articles. Valid values include: Body Byline FigureCaption FigureCredit Headline Headline2 IndentedParagrap h Pullquote SectionChapterNa me Title Title2 OrderedList UnorderedList
ROWNUM CDATA #IMPLIED	If the component resides in a Table cell, the value will describe the row number.	If the component resides in a Table cell, the value will describe the row number.	If the component resides in a Table cell, the value will describe the row number.
COLNUM CDATA #IMPLIED	If the component resides in a Table cell, the value will describe the column number.	If the component resides in a Table cell, the value will describe the column number.	If the component resides in a Table cell, the value will describe the column number.
COMPONENTIND EX CDATA #IMPLIED	Not applicable.	Specifies the index (position) of this component in the project's reflow article. The first component has a value of zero. If you specify an invalid value, the component	Specifies the index (position) of this component in the project's reflow article. The first component has a value of zero. If you specify an invalid

Element type	Construct	Modify	Deconstruct
		is placed at the end.	value, the component is placed at the end.

COMPOSITIONZONE (Modifier schema)

Element type	Construct	Modify	Deconstruct
COMPOSITIONZONE (ID, (FRAME | GEOMETRY | SHADOW | PAGEREF)*, INTERACTIVITY?)	Not applicable.	Not applicable.	Describes a Composition Zones item. (Applies only to the xml namespace.)
Attributes			
LAYOUTREF CDATA #IMPLIED	Not applicable.	Not applicable.	Identifies the layout referenced by this Composition Zones item.
BOXTYPE (CT_TEXT | CT_PICT | CT_USER) #IMPLIED	Not applicable.	Not applicable.	Indicates CT_USER as the box type for a Composition Zones item.
TYPE (INTERNAL | EXTERNAL) #IMPLIED	Not applicable.	Not applicable.	Indicates the Composition Zones items type.INTERNAL = A Composition Zones item that uses a layout within the same project.EXTERNAL = A Composition

Element type	Construct	Modify	Deconstruct
			Zones item that uses a layout in a different project.
PATH CDATA #IMPLIED	Not applicable.	Not applicable.	Indicates the absolute path to an external composition layout.
COLOR CDATA #IMPLIED	Not applicable.	Not applicable.	Identifies a color applied to a Composition Zones item. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.
SHADE CDATA #IMPLIED	Not applicable.	Not applicable.	Specifies the shade of a color applied to a Composition Zones object, as an integer percentage from 0 to 100.

Element type	Construct	Modify	Deconstruct
OPACITY CDATA #IMPLIED	Not applicable.	Not applicable.	Specifies the opacity of a color applied to a Composition Zones item, as an integer percentage from 0 to 100.
ANCHOREDIN CDATA #IMPLIED	Not applicable.	Not applicable.	Indicates an anchored Composition Zones and identifies its parent Composition Zones.
BLENDSTYLE (SOLID | LINEAR | MIDLINEAR | RECTANGULAR | DIAMOND | CIRCULAR | FULLCIRCULAR | none) “none”	Not applicable.	Not applicable.	Specifies the type of blend applied to this box (linear, circular, rectangular, etc.).
BLENDANGLE CDATA #IMPLIED	Not applicable.	Not applicable.	Specifies the angle of the blend.
BLENDCOLOR CDATA #IMPLIED	Not applicable.	Not applicable.	Specifies the second color of the blend. The first color of the blend is the color applied to the box.
BLENDSHADE CDATA #IMPLIED	Not applicable.	Not applicable.	Specifies the shade applied to the second color

Element type	Construct	Modify	Deconstruct
			of the blend, as an integer percentage from 0 to 100. The shade of the first color of the blend is the shade of the color applied to the box.
BLENDOPACITY CDATA #IMPLIED	Not applicable.	Not applicable.	Specifies the opacity applied to the second color of the blend, as an integer percentage from 0 to 100. The opacity of the first color of the blend is the opacity of the color applied to the box.
HORIZONTALBINDING (true | false) #IMPLIED	Not applicable.	Not applicable.	If true, specifies that the Composition Zones item is constrained only horizontally to the size of its composition layout.
VERTICALBINDING (true | false) #IMPLIED	Not applicable.	Not applicable.	If true, specifies that the Composition Zones item is constrained only vertically to the size of its

Element type	Construct	Modify	Deconstruct
			composition layout.
LAYOUTOPACITY CDATA #IMPLIED	Not applicable.	Not applicable.	Specifies the opacity of a Composition Zones item, as an integer percentage from 0 to 100.
SUPPRESSOUTPUT (true | false) #IMPLIED	Not applicable.	Not applicable.	If true, specifies that this Composition Zones item should not be included in output.
PREVIEWPAGE CDATA #IMPLIED	Not applicable.	Not applicable.	Identifies the page shown by default in the layout.
OPERATION (CREATE | DELETE) #IMPLIED	Lets you create or delete a Composition Zones item.	Lets you create or delete a Composition Zones item.	Not applicable.

CONDITIONALMASTERPAGEREFERENCE (Modifier schema)

Element type	Construct	Modify	Deconstruct
CONDITIONALMASTERPAGEREFERENCE ()	Not applicable.	CONDITIONALMASTERPAGEREFERENCE element typeLets you specify the master page along with the conditions that must be satisfied to apply	Not applicable.

Element type	Construct	Modify	Deconstruct
		the given master page on a page.	
Attributes			
NAME	Not applicable.	Specifies the name of the master page in QuarkXPress template to be used.	Not applicable.
POSITION	Not applicable.	Specifies the position of the page to which this master page will be applied. FIRST, LAST, REST, ANY	Not applicable.
ODDOREVEN	Not applicable.		Not applicable.
BLANKORNOTBLANK	Not applicable.	Specifies if the master page to be applied on the page should be blank or not blank. BLANKORNOTBLANK	Not applicable.

CONTENT (Modifier schema)

Element type	Construct	Modify	Deconstruct
CONTENT (#PCDATA)	Specifies the path of an image or text file that you want to associate with the parent box. The CONTENT element also supports relative paths for images or text files.	Specifies the path of an image or text file that you want to import into the parent box. Note: If you use the CONTENT element to import text, the imported text is appended to the end of any existing text in the box.	Specifies the path of the image or text file (if any) associated with the parent box.

Element type	Construct	Modify	Deconstruct
Attributes			
PICTURECONTENTLOCK (true | false) “true”	Specifies whether picture content is locked.	Specifies whether picture content is locked.	Specifies whether picture content is locked.
CONVERTQUOTES (true | false) “true”	If true, straight quotation marks in an imported text file are converted to typesetter's quotation marks and double hyphens are converted to em dashes.	If true, straight quotation marks in an imported text file are converted to typesetter's quotation marks and double hyphens are converted to em dashes.	Not applicable.
INCLUDESTYLESHEETS (true | false) “true”	If true, any style sheets in an imported text file are added to the QuarkXPress project.	If true, any style sheets in an imported text file or document are added to the QuarkXPress project.	Not applicable.
FONTNAME CDATA #IMPLIED	Specifies a font to apply to imported text.	Specifies a font to apply to imported text.	Not applicable.
PAGETOIMPORT CDATA #IMPLIED	Indicates the page number of an imported PDF.	Indicates the page number of an imported PDF.	Indicates the page number of an imported PDF.
BOUNDINGBOX (MEDIABOX | CROPBOX | BLEEDBOX | TRIMBOX) #IMPLIED	Identifies the bounding box type for an imported PDF. MEDIABOX includes the full imageable area. CROPBOX is the	Identifies the bounding box type for an imported PDF. MEDIABOX includes the full imageable area. CROPBOX is the portion of the page that should be	Identifies the bounding box type for an imported PDF. MEDIABOX includes the full imageable area. CROPBOX is the

Element type	Construct	Modify	Deconstruct
	portion of the page that should be visible (not typically used for prepress). BLEEDBOX is the area defined by the crop marks, plus 3-5 additional millimeters. TRIMBOX is the area defined by the crop marks (in other words, the finished page size).	visible (not typically used for prepress). BLEEDBOX is the area defined by the crop marks, plus 3-5 additional millimeters. TRIMBOX is the area defined by the crop marks (in other words, the finished page size).	portion of the page that should be visible (not typically used for prepress). BLEEDBOX is the area defined by the crop marks, plus 3-5 additional millimeters. TRIMBOX is the area defined by the crop marks (in other words, the finished page size).
MAINTAINPICTUREATTRIBUTES (true | false) #IMPLIED	Not applicable.	If true, maintains the picture attributes when importing a new picture.	Not applicable.
MAINTAINCLIPPINGANDRUNAROUND (true | false) #IMPLIED	Not applicable.	If true, maintains the runaround and clipping applied to a picture when reimporting the picture. Default value is false. Clipping and runaround are not maintained if you do not specify this attribute in the CONTENT element.	Not applicable.
LOCKPICTURE (true | false) #IMPLIED	Specifies whether the picture should be locked.	Specifies whether the picture should be locked.	Specifies whether the picture should be locked.

Element type	Construct	Modify	Deconstruct
CONDITIONALSTYLE CDATA #IMPLIED	Identifies the conditional style (if any) to be associated with the content specified in the CONTENT node.	Identifies the conditional style (if any) to be associated with the content specified in the CONTENT node.	Not applicable.

CONTENTPH (Modifier schema)

Element type	Construct	Modify	Deconstruct
CONTENTPH ((CONTENT), METADATA?)	Placeholder that will contain either text or picture data from a linked file.	Placeholder that will contain either text or picture data from a linked file.	Placeholder that will contain either text or picture data from a linked file.
Attributes			
NAME CDATA #REQUIRED	The name of the content placeholder (CONTENTPH).	The name of the content placeholder (CONTENTPH).	The name of the content placeholder (CONTENTPH).
OWNER (1347639377) "1347639377"	The XTensions ID of the XTensions that created this placeholder. The default XT ID is PlaceHolderSXT ID (1347639377). All placeholders created through Modifier should use this ID. This ID is assigned by	The XTensions ID of the XTensions that created this placeholder. The default XT ID is PlaceHolderSXT ID (1347639377). All placeholders created through Modifier should use this ID. This ID is assigned by default by the schema, so there is no need to specify	The XTensions ID of the XTensions that created this placeholder.

Element type	Construct	Modify	Deconstruct
	default by the schema, so there is no need to specify this manually. schema validation will add this attribute.	this manually. schema validation will add this attribute.	

CONTINUEDHEADER (Modifier schema)

Element type	Construct	Modify	Deconstruct
CONTINUEDHEADER (ROW*)	Identifies a continued table header.	Identifies a continued table header.	Identifies a continued table header.
Attributes			
APPLIEDTO (NOHEADERROW | FIRSTHEADERROW | ALLHEADERROWS) #IMPLIED	Specifies which header rows are continued header rows.	Specifies whether only the first header row or all header rows should be considered a continued header.	Specifies which header rows are continued header rows.

CONTINUEDTROWSTYLE (Modifier schema)

Element type	Construct	Modify	Deconstruct
CONTINUEDTROWSTYLE (TOPGRID?, BOTTOMGRID?)	Defines a style for “continued” rows in an <INLINETABLE>.	Defines a style for “continued” rows in an <INLINETABLE>.	Not applicable.
Attributes			
PARASTYLE CDATA #IMPLIED	Identifies the paragraph style	Identifies the paragraph style	Not applicable.

Element type	Construct	Modify	Deconstruct
	sheet for the row style.	sheet for the row style.	
ALIGNMENT (LEFT | RIGHT | CENTER | JUSTIFIED | FORCED) #IMPLIED	Identifies the paragraph alignment for the row style.	Identifies the paragraph alignment for the row style.	Not applicable.
ANGLE CDATA #IMPLIED	Identifies the text angle for the row style.	Identifies the text angle for the row style.	Not applicable.
VALIGN (TOP | CENTER | BOTTOM) #IMPLIED	Specifies the vertical alignment of the row style.	Specifies the vertical alignment of the row style.	Not applicable.
COLOR CDATA #IMPLIED	Specifies the background color of the row style.	Specifies the background color of the row style.	Not applicable.
SHADE CDATA #IMPLIED	Specifies the background shade of the row style.	Specifies the background shade of the row style.	Not applicable.
INSET CDATA #IMPLIED	Specifies the text inset of the row style.	Specifies the text inset of the row style.	Not applicable.

CONTOUR (Modifier schema)

Element type	Construct	Modify	Deconstruct
CONTOUR (VERTICES)	A single contour within a spline shape.	A single contour within a spline shape.	A single contour within a spline shape.
Attributes			
CURVEDEDGES	Specifies	Specifies whether	Specifies whether

Element type	Construct	Modify	Deconstruct
(true | false) “false”	whether there are any curved edges in the contour.	there are any curved edges in the contour.	there are any curved edges in the contour.
RECTCONTOUR (true | false) “false”	Specifies whether this contour is rectangular.	Specifies whether this contour is rectangular.	Specifies whether this contour is rectangular.
INVERTEDCONTOUR (true | false) “false”	Specifies whether the points describe a hole instead of an outside contour.	Specifies whether the points describe a hole instead of an outside contour.	Specifies whether the points describe a hole instead of an outside contour.
TOPLEVEL (true | false) “false”	Specifies whether the contour has no containing contours.	Specifies whether the contour has no containing contours.	Specifies whether the contour has no containing contours.
SELFINTERSECTED (true | false) “false”	Specifies whether the contour intersects itself.	Specifies whether the contour intersects itself.	Specifies whether the contour intersects itself.
POLYCONTOUR (true | false) “false”	Specifies whether this is a polycontour (as opposed to a spline contour).	Specifies whether this is a polycontour (as opposed to a spline contour).	Specifies whether this is a polycontour (as opposed to a spline contour).
VERTEXTAGEXISTS (true | false) “false”	Specifies whether there are vertex tags associated with the contour.	Specifies whether there are vertex tags associated with the contour.	Specifies whether there are vertex tags associated with the contour.

CONTOURS (Modifier schema)

Element type	Construct	Modify	Deconstruct
CONTOURS (CONTOUR+)	A group of contours which, combined, make a spline shape.	A group of contours which, combined, make a spline shape.	A group of contours which, combined, make a spline shape.

COPYFIT (Modifier schema)

Element type	Construct	Modify	Deconstruct
COPYFIT (empty)	Not applicable.	Not applicable.	Indicates whether the copy in this text box or chain fits the available space.
Attributes			
STATE (fit | overFit | underFit) “fit”	Not applicable.	Not applicable.	Indicates whether the text currently fits in the box (fit), is too long (overFit), or is too short (underFit).
FITAMOUNT CDATA #IMPLIED	Not applicable.	Not applicable.	Indicates the vertical distance in points by which text in a text box is overFit or underFit. See the STATE element.
NUMBEROFCHARACTERS CDATA #IMPLIED	Not applicable.	Not applicable.	Indicates how many characters are included in the story.
NUMBEROFWORDS CDATA #IMPLIED	Not applicable.	Not applicable.	Indicates how many words are included in the

Element type	Construct	Modify	Deconstruct
			story.
NUMBEROFLINES CDATA #IMPLIED	Not applicable.	Not applicable.	Indicates how many lines are included in the story.
FITLINEAMOUNT CDATA #IMPLIED	Not applicable.	Not applicable.	Indicates how many lines the text is overfit or underfit.
STORYDEPTHAMOUNT CDATA #IMPLIED	Not applicable.	Not applicable.	Not applicable.

COPYRIGHT (Modifier schema)

Element type	Construct	Modify	Deconstruct
COPYRIGHT (#PCDATA)	Not applicable.	Part of the <EBOOKMETADATA> element. Specifies copyright information for an e-book.	Specifies copyright information for an e-book.

DATAPROVIDER (Modifier schema)

Element type	Construct	Modify	Deconstruct
DATAPROVIDER (empty)	Not applicable.	Not applicable.	Provides information about the XTensions module through which interactivity is applied to a box.
Attributes			
DATAPROVIDERX	Not applicable.	Not applicable.	Provides

Element type	Construct	Modify	Deconstruct
TID CDATA #IMPLIED			information about the XTensions module through which interactivity is applied to a box. For example, if interactivity is provided through a built-in QuarkXPress XTensions module, this value is QDPC (Quark Digital Publishing Core).

DEL (Modifier schema)

Element type	Construct	Modify	Deconstruct
DEL (#PCDATA)	Describes a tracked deletion in text.	Describes a tracked deletion in text.	Describes a tracked deletion in text.
Attributes			
CREATEDBY CDATA #IMPLIED	Not applicable.	Not applicable.	The username of the deleter.
CREATEDON CDATA #IMPLIED	Not applicable.	Not applicable.	The deletion date.

DELETECELLS (Modifier schema)

Element type	Construct	Modify	Deconstruct
DELETECELLS (empty)	Not applicable.	Deletes cells from an existing table.	Not applicable.
<!ATTLIST DELETECELLS			

Element type	Construct	Modify	Deconstruct
TYPE (ROW | COLUMN | HEADER | FOOTER) #REQUIRED	Not applicable.	Specifies whether to delete rows, columns, headers, or footers.	Not applicable.
BASEINDEX CDATA #REQUIRED	Not applicable.	Specifies the index number of the first cell to be deleted.	Not applicable.
DELETECOUNT CDATA #REQUIRED	Not applicable.	Specifies how many cells to delete.	Not applicable.

DESCRIPTION (Modifier schema)

Element type	Construct	Modify	Deconstruct
DESCRIPTION (#PCDATA)	Not applicable.	Part of the <EBOOKMETADATA> element. Specifies a description for an e-book.	Specifies a description for an e-book.

DROPCAP (Modifier schema)

Element type	Construct	Modify	Deconstruct
DROPCAP (empty)	Describes a drop-capital effect at the beginning of a paragraph, which is when initial characters display at a large size and hang two or more lines below the first line of a	Describes a drop-capital effect at the beginning of a paragraph, which is when initial characters display at a large size and hang two or more lines below the first line of a paragraph.	Describes a drop-capital effect at the beginning of a paragraph, which is when initial characters display at a large size and hang two or more lines below the first line of a paragraph.

Element type	Construct	Modify	Deconstruct
	paragraph.		
Attributes			
CHARCOUNT CDATA #REQUIRED	Specifies how many characters should be included in a drop-cap effect.	Specifies how many characters should be included in a drop-cap effect.	Specifies how many characters are included in a drop-cap effect.
LINECOUNT CDATA #REQUIRED	Specifies the number of lines a drop-caps should hang in the paragraph.	Specifies the number of lines a drop-caps should hang in the paragraph.	Specifies the number of lines drop-caps hang in the paragraph.

EBOOKMETADATA (Modifier schema)

Element type	Construct	Modify	Deconstruct
EBOOKMETADATA (TITLE | AUTHOR | PUBLISHER | COPYRIGHT | ISBN | DESCRIPTION | KEYWORDS | SPINEIMAGE)*	Not applicable.	Defines a variety of metadata for a layout to be exported as an e-book.	Defines a variety of metadata for a layout to be exported as an e-book.

ENTRY (Modifier schema)

Element type	Construct	Modify	Deconstruct
ENTRY (#PCDATA | RICHTEXT | PARAGRAPH | CONTENT | PICTURE)*	Describes a cell in an <INLINETABLE>.	Describes a cell in an <INLINETABLE>.	Not applicable.
Attributes			

Element type	Construct	Modify	Deconstruct
WIDTH CDATA #IMPLIED	Specifies the width of the cell, either as an absolute measurement or as a percentage of the table width. To specify a percentage, use %. If you do not specify a width, cell widths are distributed evenly.	Specifies the width of the cell, either as an absolute measurement or as a percentage of the table width. To specify a percentage, use %. If you do not specify a width, cell widths are distributed evenly.	Not applicable.
COLOR CDATA #IMPLIED	Specifies the background color of the cell.	Specifies the background color of the cell.	Not applicable.
SHADE CDATA #IMPLIED	Specifies the background shade of the cell.	Specifies the background shade of the cell.	Not applicable.
COLSPAN CDATA #IMPLIED	If specified, indicates that this cell is merged with the following number of cells. For example, COLSPAN="2" merges this cell with the next one.	If specified, indicates that this cell is merged with the following number of cells. For example, COLSPAN="2" merges this cell with the next one.	Not applicable.
ROWSPAN CDATA #IMPLIED	If specified, indicates that this cell is merged with the following	If specified, indicates that this cell is merged with the following number of cells. For example,	Not applicable.

Element type	Construct	Modify	Deconstruct
	number of cells. For example, ROWSPAN="2" merges this cell with the one below it.	ROWSPAN="2" merges this cell with the one below it.	
ANGLE	Specifies the angle of the cell.	Specifies the angle of the cell.	Not applicable.
VALIGN	Specifies the vertical alignment of the cell.	Specifies the vertical alignment of the cell.	Not applicable.
ALIGNMENT	Specifies the alignment of the cell.	Specifies the alignment of the cell.	Not applicable.

EVENTCOLSTYLE (Modifier schema)

Element type	Construct	Modify	Deconstruct
EVENTCOLSTYLE (LEFTGRID?, RIGHTGRID?)	Defines a style for even columns in an <INLINETABLE>.	Defines a style for even columns in an <INLINETABLE>.	Not applicable.
Attributes			
WIDTH CDATA #IMPLIED	Specifies the width of the column style.	Specifies the width of the column style.	Not applicable.
COLOR CDATA #IMPLIED	Specifies the background color of the column style.	Specifies the background color of the column style.	Not applicable.
SHADE CDATA #IMPLIED	Specifies the background shade of the column style.	Specifies the background shade of the column style.	Not applicable.

EVENTROWSTYLE (Modifier schema)

Element type	Construct	Modify	Deconstruct
EVENTROWSTYLE (TOPGRID?, BOTTOMGRID?)	Defines a style for even rows in an <INLINETABLE>.	Defines a style for even rows in an <INLINETABLE>	Not applicable.
Attributes			
PARASTYLE CDATA #IMPLIED	Identifies the paragraph style sheet for the row style.	Identifies the paragraph style sheet for the row style.	Not applicable.
ALIGNMENT (LEFT | RIGHT | CENTER | JUSTIFIED | FORCED) #IMPLIED	Identifies the paragraph alignment for the row style.	Identifies the paragraph alignment for the row style.	Not applicable.
ANGLE CDATA #IMPLIED	Identifies the text angle for the row style.	Identifies the text angle for the row style.	Not applicable.
VALIGN (TOP | CENTER | BOTTOM) #IMPLIED	Specifies the vertical alignment of the row style.	Specifies the vertical alignment of the row style.	Not applicable.
COLOR CDATA #IMPLIED	Specifies the background color of the row style.	Specifies the background color of the row style.	Not applicable.
SHADE CDATA #IMPLIED	Specifies the background shade of the row style.	Specifies the background shade of the row style.	Not applicable.
INSET CDATA #IMPLIED	Specifies the text inset of the row style.	Specifies the text inset of the row style.	Not applicable.

FIRSTTCOLSTYLE (Modifier schema)

Element type	Construct	Modify	Deconstruct
FIRSTTCOLSTYLE (LEFTGRID?, RIGHTGRID?)	Defines a style for the first column in an <INLINETABLE>.	Defines a style for the first column in an <INLINETABLE>.	Not applicable.
Attributes			
WIDTH CDATA #IMPLIED	Specifies the width of the column style.	Specifies the width of the column style.	Not applicable.
COLOR CDATA #IMPLIED	Specifies the background color of the column style.	Specifies the background color of the column style.	Not applicable.
SHADE CDATA #IMPLIED	Specifies the background shade of the column style.	Specifies the background shade of the column style.	Not applicable.

FIT (Modifier schema)

Element type	Construct	Modify	Deconstruct
FIT (MAX, MIN)	Lets you resize a box to fit its text or picture, within the limitations specified by the <MAX> and <MIN> elements. A box will expand or shrink only until it reaches the <MIN> or <MAX> size.	Lets you resize a box to fit its text or picture, within the limitations specified by the <MAX> and <MIN> elements. A box will expand or shrink only until it reaches the <MIN> or <MAX> size.	Not applicable.
Attributes			

Element type	Construct	Modify	Deconstruct
POINT (TOPLEFT | TOP | TOPRIGHT | RIGHT | BOTTOMRIGHT | BOTTOM | BOTTOMLEFT | LEFT | CENTER) #REQUIRED	Lets you specify the direction in which the box should be resized. To resize the box from the center, use “CENTER”.	Lets you specify the direction in which the box should be resized. To resize the box from the center, use “CENTER”.	Not applicable.
AVOIDBOXESBY CDATA #IMPLIED	Lets you specify the distance between the POINT side or corner of a resized box and any other items around it. A box will expand only until it is this distance from an adjacent item.	Lets you specify the distance between the POINT side or corner of a resized box and any other items around it. A box will expand only until it is this distance from an adjacent item.	Not applicable.
PROPORTIONAL (true | false) “false”	Lets you specify whether the resized box should have the same aspect ratio as the original box.	Lets you specify whether the resized box should have the same aspect ratio as the original box.	Not applicable.

FITTEXT (Modifier schema)

Element type	Construct	Modify	Deconstruct
FITTEXT (EMPTY)	Lets you control text-fitting options at the story level, rather than	FITTEXT element type Lets you control text-fitting options at the story level, rather than using	Not applicable.

Element type	Construct	Modify	Deconstruct
	using application-level text-fitting options. Note: The FITTEXTTOBOX attribute fits text to a box with the application-level text fitting defaults.	application-level text-fitting options. Note: The FITTEXTTOBOX attribute fits text to a box with the application-level text fitting defaults.	
Attributes			
ALLOWTEXTTOGROW (true | false) “false”	If true, text will grow if the box is underfit and shrink if the box is overfit. If false, text can shrink but cannot grow.	If true, text will grow if the box is underfit and shrink if the box is overfit. If false, text can shrink but cannot grow.	Not applicable.
SCALINGINCREMENT CDATA “5”	Indicates the percentage by which text size, baseline shift, tracking, and so forth will be incremented or decremented until the text fits.	Indicates the percentage by which text size, baseline shift, tracking, and so forth will be incremented or decremented until the text fits.	Not applicable.
MAXFONTSIZE CDATA #IMPLIED	Indicates the maximum font size that can be used for text fitting. The default value is the maximum text size allowed by QuarkXPress	Indicates the maximum font size that can be used for text fitting. The default value is the maximum text size allowed by QuarkXPress (3184 pt).	Not applicable.

Element type	Construct	Modify	Deconstruct
	(3184 pt).		
MINFONTSIZE CDATA #IMPLIED	Indicates the minimum font size that can be used for text fitting. The default value is the minimum text size allowed by QuarkXPress (2 pt).	Indicates the minimum font size that can be used for text fitting. The default value is the minimum text size allowed by QuarkXPress (2 pt).	Not applicable.
SCALETRACKING (true | false) "false"	If true, tracking can be changed during text fitting operations. If false, tracking cannot be changed during text fitting operations.	If true, tracking can be changed during text fitting operations. If false, tracking cannot be changed during text fitting operations.	Not applicable.
SCALEBASELINE SHIFT (true | false) "false"	If true, baseline shift can be changed during text fitting operations. If false, baseline shift cannot be changed during text fitting operations.	If true, baseline shift can be changed during text fitting operations. If false, baseline shift cannot be changed during text fitting operations.	Not applicable.
SCALETEXTSCALING (true | false) "false"	If true, horizontal/vertical scale can be changed during text fitting operations. If	If true, horizontal/vertical scale can be changed during text fitting operations. If false,	Not applicable.

Element type	Construct	Modify	Deconstruct
	false, horizontal/vertical scale cannot be changed during text fitting operations. If horizontal scaling has been applied to text, only horizontal scaling will be changed. The same is true for vertical scaling. Horizontal and vertical scaling cannot be adjusted simultaneously.	horizontal/vertical scale cannot be changed during text fitting operations. If horizontal scaling has been applied to text, only horizontal scaling will be changed. The same is true for vertical scaling. Horizontal and vertical scaling cannot be adjusted simultaneously.	

FOOTER (Modifier schema)

Element type	Construct	Modify	Deconstruct
FOOTER (ROW*)	Specifies if the row is to be a footer row.	Specifies if the row is to be a footer row.	Indicates if the row is to be a footer row.
Attributes			
FOOTERROWS CDATA #IMPLIED	Specifies number of footer row.	Specifies number of footer row.	Specifies number of footer row.

FOOTERTROWSTYLE (Modifier schema)

Element type	Construct	Modify	Deconstruct
FOOTERTROWST	Defines a style	Defines a style for	Not applicable.

Element type	Construct	Modify	Deconstruct
YLE (TOPGRID?, BOTTOMGRID?)	for footer rows in an <INLINETABLE>.	footer rows in an <INLINETABLE>.	
Attributes			
PARASTYLE CDATA #IMPLIED	Identifies the paragraph style sheet for the row style.	Identifies the paragraph style sheet for the row style.	Not applicable.
ALIGNMENT (LEFT | RIGHT | CENTER | JUSTIFIED | FORCED) #IMPLIED	Identifies the paragraph alignment for the row style.	Identifies the paragraph alignment for the row style.	Not applicable.
ANGLE CDATA #IMPLIED	Identifies the text angle for the row style.	Identifies the text angle for the row style.	Not applicable.
VALIGN (TOP | CENTER | BOTTOM) #IMPLIED	Specifies the vertical alignment of the row style.	Specifies the vertical alignment of the row style.	Not applicable.
COLOR CDATA #IMPLIED	Specifies the background color of the row style.	Specifies the background color of the row style.	Not applicable.
SHADE CDATA #IMPLIED	Specifies the background shade of the row style.	Specifies the background shade of the row style.	Not applicable.
INSET CDATA #IMPLIED	Specifies the text inset of the row style.	Specifies the text inset of the row style.	Not applicable.

FORMAT (Modifier schema)

Element type	Construct	Modify	Deconstruct
FORMAT (KEEPLINESTOGETHER?, DROPCAP?, LOCKTOGRID?, BNSTYLE?, SHADINGSTYLE?)	Describes formatting for a PARAGRAPH element.	Describes formatting for a PARAGRAPH element.	Describes formatting for a PARAGRAPH element.
Attributes			
SPACEBEFORE CDATA #IMPLIED	Describes the amount of space before a paragraph.	Describes the amount of space before a paragraph.	Describes the amount of space before a paragraph.
SPACEAFTER CDATA #IMPLIED	Describes the amount of space after a paragraph.	Describes the amount of space after a paragraph.	Describes the amount of space after a paragraph.
LEFTINDENT CDATA #IMPLIED	Describes the amount of space in a paragraphs left indent.	Describes the amount of space in a paragraphs left indent.	Describes the amount of space in a paragraphs left indent.
RIGHTINDENT CDATA #IMPLIED	Describes the amount of space in a paragraphs right indent.	Describes the amount of space in a paragraphs right indent.	Describes the amount of space in a paragraphs right indent.
FIRSTLINE CDATA #IMPLIED	Describes the amount of space in a paragraphs first-line indent.	Describes the amount of space in a paragraphs first-line indent.	Describes the amount of space in a paragraphs first-line indent.
LEADING CDATA #IMPLIED	Describes a paragraphs line spacing.	Describes a paragraphs line spacing.	Describes a paragraphs line spacing.
ALIGNMENT (LEFT | RIGHT | CENTERED |	Indicates whether a paragraph	Indicates whether a paragraph should be left-aligned, right-	Indicates whether a paragraph is left-aligned, right-

Element type	Construct	Modify	Deconstruct
JUSTIFIED | FORCED) “LEFT”	should be left-aligned, right-aligned, centered, justified, or force-justified.Note: JUSTIFIED aligns the text in a paragraph to the left and right indentations, except for the last line. FORCED justifies every line, including the last line.	aligned, centered, justified, or force-justified. Note: JUSTIFIED aligns the text in a paragraph to the left and right indentations, except for the last line. FORCED justifies every line, including the last line.	aligned, centered, justified, or force-justified. Note: JUSTIFIED aligns the text in a paragraph to the left and right indentations, except for the last line. FORCED justifies every line, including the last line.
HANDJ CDATA #IMPLIED	Identifies a hyphenation and justification specification to be applied to a paragraph. Note: Only the name of an H&J specification is included in this attribute. The definition of the H&J specification is stored in the projects Job Jackets file or defined using the Document Controls	Identifies a hyphenation and justification specification to be applied to a paragraph. Note: Only the name of an H&J specification is included in this attribute. The definition of the H&J specification is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	Identifies the hyphenation and justification specification applied to a paragraph. Note: Only the name of an H&J specification is included in this attribute. The definition of the H&J specification is stored in the projects Job Jackets file or defined using the Document Controls submenu in

Element type	Construct	Modify	Deconstruct
	submenu in QuarkXPress Server.		QuarkXPress Server.
KEEPWITHNEXT (true | false | none) “none”	Specifies whether the last lines of a paragraph should always appear on the same page as the next paragraph.	Specifies whether the last lines of a paragraph should always appear on the same page as the next paragraph.	Specifies whether the last lines of a paragraph should always appear on the same page as the next paragraph.
HANGINGCHARACTERS CDATA #IMPLIED	Describes the hanging character set used by this paragraph.	Describes the hanging character set used by this paragraph.	Describes the hanging character set used by this paragraph.
CHARACTERALIGNMENT (ROMANBASELINE | EMBOXTOP | EMBOXCENTER | EMBOXBOTTOM | ICFBOXTOP | ICFBOXBOTTOM) “ROMANBASELINE”	Defines the character alignment used by this paragraph.For a story with horizontal direction, EMBOXTOP, EMBOXBOTTOM, ICFBOXTOP, ICFBOXBOTTOM are applicable.For a story with vertical direction, EMBOXRIGHT, EMBOXLEFT, ICFBOXRIGHT,	Defines the character alignment used by this paragraph.For a story with horizontal direction, EMBOXTOP, EMBOXBOTTOM, ICFBOXTOP, ICFBOXBOTTOM are applicable.For a story with vertical direction, EMBOXRIGHT, EMBOXLEFT, ICFBOXRIGHT, ICFBOXLEFT are applicable.	Defines the character alignment used by this paragraph.For a story with horizontal direction, EMBOXTOP, EMBOXBOTTOM, ICFBOXTOP, ICFBOXBOTTOM are applicable.For a story with vertical direction, EMBOXRIGHT, EMBOXLEFT, ICFBOXRIGHT, ICFBOXLEFT are applicable.

Element type	Construct	Modify	Deconstruct
	ICFBOXLEFT are applicable.		
MOJIGUMISET CDATA #IMPLIED	Identifies the mojigumi set (if any) applied to this paragraph.	Identifies the mojigumi set applied to this paragraph.	Identifies the mojigumi set (if any) applied to this paragraph.
BACKGROUNDCOLOR CDATA #IMPLIED	Specifies a background color to be inserted behind the text. This color displays only in rendered output, and is not saved with the project file.	Specifies a background color to be inserted behind the text. This color displays only in rendered output, and is not saved with the project file.	Not applicable.
OPACITY CDATA #IMPLIED	Specifies the opacity of a background color to be inserted behind the text. This color displays only in rendered output, and is not saved with the project file.	Specifies the opacity of a background color to be inserted behind the text. This color displays only in rendered output, and is not saved with the project file.	Not applicable.

FRAME (Modifier schema)

Element type	Construct	Modify	Deconstruct
FRAME (empty)	Describes a box frame.	Describes a box frame.	Describes a box frame.
Attributes			
STYLE CDATA #IMPLIED	Specifies a Dashes & Stripes	Specifies a Dashes & Stripes style for a	Specifies a Dashes & Stripes

Element type	Construct	Modify	Deconstruct
	style for a frame.	frame.	style for a frame.
WIDTH CDATA #IMPLIED	Specifies the thickness of a frame in points as a floating point value.	Specifies the thickness of a frame in points as a floating point value.	Specifies the thickness of a frame in points as a floating point value.
COLOR CDATA #IMPLIED	Identifies the color of a frame.Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	Identifies the color of a frame.Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.	Identifies the color of a frame.Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.
SHADE CDATA #IMPLIED	Specifies the shade of the color applied to a frame, as an integer percentage from 0 to 100.	Specifies the shade of the color applied to a frame, as an integer percentage from 0 to 100.	Specifies the shade of the color applied to a frame, as an integer percentage from 0 to 100.
OPACITY CDATA #IMPLIED	Specifies the opacity of a frame, specified as an integer	Specifies the opacity of a frame, specified as an integer percentage from 0 to	Specifies the opacity of a frame, specified as an integer

Element type	Construct	Modify	Deconstruct
	percentage from 0 to 100.	100.	percentage from 0 to 100.
GAPCOLOR CDATA #IMPLIED	Identifies the color of a frame gap. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	Identifies the color of a frame gap. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.	Identifies the color of a frame gap. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.
GAPSHADE CDATA #IMPLIED	Specifies the shade of the color applied to a frame gap, as an integer percentage from 0 to 100.	Specifies the shade of the color applied to a frame gap, as an integer percentage from 0 to 100.	Specifies the shade of the color applied to a frame gap, as an integer percentage from 0 to 100.
GAPOPACITY CDATA #IMPLIED	Specifies the opacity of the gap color of a frame, specified as an integer percentage from 0 to 100.	Specifies the opacity of the gap color of a frame, specified as an integer percentage from 0 to 100.	Specifies the opacity of the gap color of a frame, specified as an integer percentage from 0 to 100.

GEOMETRY (Modifier schema)

Element type	Construct	Modify	Deconstruct
GEOMETRY ((POSITION | RELPOSITION)? | MOVEUP | MOVEDOWN | MOVELEFT | MOVERIGHT | GROWACROSS | GROWDOWN | SHRINKACROSS | SHRINKDOWN | ALLOWBOXONTOPASTEBOARD | ALLOWBOXOFFPAGE | STACKINGORDER | SUPPRESSOUTPUT | RUNAROUND | LINESTYLE | SPLINESHAPE | FIT)*)	Describes the geometric characteristics of a box or line.	Describes the geometric characteristics of a box or line, and allows you to change its position and size.	Describes the geometric characteristics of a box or line.
Attributes			
SHAPE (SH_RECT | SH_CONVEXRRECT | SH_CONCAVERRECT | SH_STRAIGHTRRECT | SH_OVAL | SH_LINE | SH_ORTHLINE | SH_SPLINEBOX | SH_NONE |	Describes the shape of a box or line.SH_RECT = Rectangular boxSH_CONVEXRRECT = Box with convex cornersSH_CONCAVERRECT = Box with concave cornersSH_STR	Describes the shape of a box or line.SH_RECT = Rectangular boxSH_CONVEXRRECT = Box with convex cornersSH_CONCAVERRECT = Box with concave cornersSH_STRAIGHTRRECT = Box with beveled	Describes the shape of a box or line.SH_RECT = Rectangular boxSH_CONVEXRRECT = Box with convex cornersSH_CONCAVERRECT = Box with concave cornersSH_STRAIGHTRRECT = Box

Element type	Construct	Modify	Deconstruct
SH_ORTHPOLYLINE | SH_SPLINELINE | SH_ORTHPOLYBOX | SH_USER) “SH_RECT”	AIGHTRRECT = Box with beveled cornersSH_OVAL = Elliptical boxSH_LINE = LineSH_ORTHLINE = Orthogonal line (restricted to 45-degree angles)SH_SPLINEBOX = Freehand shapeSH_NONE = Available to define in XDK APISH_ORTHPOLYLINE = Can be defined in XDKSH_SPLINELINE = Freehand lineSH_ORTHPOLYBOX = Available to define in XDK APINote: You cannot specify PICTURE content for a box if its SHAPE attribute is set to SH_LINE.	cornersSH_OVAL = Elliptical boxSH_LINE = LineSH_ORTHLINE = Orthogonal line (restricted to 45-degree angles)SH_SPLINEBOX = Freehand shapeSH_NONE = Available to define in XDK APISH_ORTHPOLYLINE = Can be defined in XDKSH_SPLINELINE = Freehand lineSH_ORTHPOLYBOX = Available to define in XDK APINote: You cannot specify PICTURE content for a box if its SHAPE attribute is set to SH_LINE.	with beveled cornersSH_OVAL = Elliptical boxSH_LINE = LineSH_ORTHLINE = Orthogonal line (restricted to 90-degree angles)SH_SPLINEBOX = Freehand shapeSH_NONE = Available to define in XDK APISH_ORTHPOLYLINE = Can be defined in XDKSH_SPLINELINE = Freehand lineSH_ORTHPOLYBOX = Available to define in XDK APISH_USER = Available to define in XDK API
PAGE CDATA #IMPLIED	Specifies the number of the page where the upper left corner of this box or line should be created. If the	Specifies the number of the page where the upper left corner of this box or line is located. If the page number is followed by *, the box origin is	Specifies the number of the page where the upper left corner of this box or line is located. If the page number is

Element type	Construct	Modify	Deconstruct
	page number is followed by *, the box origin is on the left pasteboard. If the page number is followed by **, the box origin is on the right pasteboard. Note: This attribute determines where to create a box or line, regardless of which PAGE element the box or line occurs within.	on the left pasteboard. If the page number is followed by **, the box origin is on the right pasteboard. Note: This attribute determines where a box or line is, regardless of which PAGE element the box or line occurs within.	followed by *, the box origin is on the left pasteboard. If the page number is followed by **, the box origin is on the right pasteboard. Note: This attribute determines where a box or line is, regardless of which PAGE element the box or line occurs within.
ANGLE CDATA #IMPLIED	Specifies a rotation angle for a box or line as a floating-point value between –360 degrees and 360 degrees.	Specifies a rotation angle for a box or line as a floating-point value between –360 degrees and 360 degrees.	Specifies a rotation angle for a box or line as a floating-point value between –360 degrees and 360 degrees.
LAYER CDATA #IMPLIED	Identifies the layer where a box or line should be created.	Identifies the layer where a box or line is located.	Identifies the layer that a box resides on. Note: Boxes on non-displayed layers are not included. This means you can use the LAYER URL parameter as a

Element type	Construct	Modify	Deconstruct
			filter when a layout contains multiple layers.
CORNERSTYLE (ROUNDED | CONCAVE | RECTANGLE | BEVELED) #IMPLIED	Identifies the corner style (if any) applied to this box.	Identifies the corner style (if any) applied to this box.	Identifies the corner style (if any) applied to this box.
SKEW CDATA #IMPLIED	Specifies a skew value for the contents of this box or line as a floating-point value between –75 degrees and 75 degrees.	Specifies a skew value for the contents of this box or line as a floating-point value between –75 degrees and 75 degrees.	Specifies a skew value for the contents of this box or line as a floating-point value between –75 degrees and 75 degrees.

GRADIENT (Modifier Schema)

Element type	Construct	Modify	Deconstruct
GRADIENT (ELEMENT)	Describes a gradient color.	Describes a gradient color.	Describes a gradient color.
Attributes			
TYPE (RADIAL| AXIAL| RECTANGULAR | FULL RADIAL) #IMPLIED	Specifies the type of gradient.	Specifies the type of gradient.	Specifies the type of gradient.
ASPECTRATIO	Specifies the width of a column.	Specifies the width of a column.	Specifies the width of a column.
ANGLE	Identifies the color of a column. Overrides the	Identifies the color of a column. Overrides the TABLE@COLOR attribute.Note: Only	Identifies the color of a column. Overrides the TABLE@COLOR

Element type	Construct	Modify	Deconstruct
	TABLE@COLOR attribute.Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.	attribute.Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.
STOP	Specifies the shade of the color applied to a column, as an integer percentage from 0 to 100.	Specifies the shade of the color applied to a column, as an integer percentage from 0 to 100.	Specifies the shade of the color applied to a column, as an integer percentage from 0 to 100.
OPACITY CDATA #IMPLIED	Specifies the opacity of the color applied to a column, specified as an integer percentage from 0 to 100.	Specifies the opacity of the color applied to a column, specified as an integer percentage from 0 to 100.	Specifies the opacity of the color applied to a column, specified as an integer percentage from 0 to 100.
MERGECOLSPAN CDATA #IMPLIED	Attribute used for merging cells, rows, and columns.	Attribute used for merging cells, rows, and columns.	If a table includes merged cells, then the MERGECOLSPAN

Element type	Construct	Modify	Deconstruct
			value is shown in the xml output.
SPLIT (true | false) #IMPLIED	Not applicable.	Attribute used for splitting merged cells.	Not applicable.
AUTOFIT (true | false | none) “none”	Specifies whether the rows or columns will adjust size to fit the content.	Specifies whether the rows or columns will adjust size to fit the content.	Indicates whether the rows or columns will adjust size to fit the content.
AUTOFITMAXLIMIT CDATA #IMPLIED	Max limit for autofit.	Max limit for autofit.	Max limit for autofit.

GRID (Modifier schema)

Element type	Construct	Modify	Deconstruct
GRID (GRIDLINE)	Element used for specifying a grid in a table.	Element used for specifying a grid in a table.	Element used for specifying a grid in a table.
Attributes			
TYPE (HGRID | VGRID | ALLGRID) #IMPLIED	Not applicable.	Attribute used for selecting a horizontal or vertical grid (or both).	Not applicable.

GRIDLINE (Modifier schema)

Element type	Construct	Modify	Deconstruct
GRIDLINE (empty)	Element used to define line attributes.	Element used to define line attributes.	Element used to define line attributes.
Attributes			
STYLE CDATA	Identifies a	Identifies a Dashes &	Identifies a

Element type	Construct	Modify	Deconstruct
#IMPLIED	Dashes & Stripes style (LINESTYLE) for a rule. Note: Only the name of a Dashes & Stripes style is included in this attribute. The definition of the Dashes & Stripes style is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	Stripes style (LINESTYLE) for a rule. Note: Only the name of a Dashes & Stripes style is included in this attribute. The definition of the Dashes & Stripes style is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	Dashes & Stripes style (LINESTYLE) for a rule. Note: Only the name of a Dashes & Stripes style is included in this attribute. The definition of the Dashes & Stripes style is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.
WIDTH CDATA #IMPLIED	Specifies the thickness of a line as a floating point value (measured in points).	Specifies the thickness of a line as a floating point value (measured in points).	Specifies the thickness of a line as a floating point value (measured in points).
COLOR CDATA #IMPLIED	Identifies the color of a line.	Identifies the color of a line.	Identifies the color of a line.
SHADE CDATA #IMPLIED	Specifies the shade of the color applied to a line, as an integer percentage from 0 to 100.	Specifies the shade of the color applied to a line, as an integer percentage from 0 to 100.	Specifies the shade of the color applied to a line, as an integer percentage from 0 to 100.
OPACITY CDATA #IMPLIED	Specifies the opacity of a line,	Specifies the opacity of a line, specified as	Specifies the opacity of a line,

Element type	Construct	Modify	Deconstruct
	specified as an integer percentage from 0 to 100.	an integer percentage from 0 to 100.	specified as an integer percentage from 0 to 100.
GAPCOLOR CDATA #IMPLIED	Identifies the color of a line gap.	Identifies the color of a line gap.	Identifies the color of a line gap.
GAPSHADE CDATA #IMPLIED	Specifies the shade of the color applied to a line gap, as an integer percentage from 0 to 100.	Specifies the shade of the color applied to a line gap, as an integer percentage from 0 to 100.	Specifies the shade of the color applied to a line gap, as an integer percentage from 0 to 100.
GAPOPACITY CDATA #IMPLIED	Specifies the opacity of the gap color of a line, specified as an integer percentage from 0 to 100.	Specifies the opacity of the gap color of a line, specified as an integer percentage from 0 to 100.	Specifies the opacity of the gap color of a line, specified as an integer percentage from 0 to 100.

GROUP (Modifier schema)

Element type	Construct	Modify	Deconstruct
GROUP (ID, BOXREF*,GEOMETRY?, SHADOW?)	Not applicable.	Not applicable.	Describes a group of items.
Attributes			
OPERATION (CREATE | DELETE) #IMPLIED	Creates or deletes the group in the layout.	Creates or deletes the group in the layout.	Not applicable.
ANCHOREDIN CDATA #IMPLIED	Not applicable.	Not applicable.	Indicates an anchored box in a text box and

Element type	Construct	Modify	Deconstruct
			identifies its parent box.
ANCHOREDGROUPMEMBER CDATA #IMPLIED	Specifies that this group is a member of the indicated anchored group.	Specifies that this group is a member of the indicated anchored group.	Specifies that this group is a member of the indicated anchored group.
GROUPOPACITY CDATA #IMPLIED	Specifies the opacity of the group as a whole.	Specifies the opacity of the group as a whole.	Specifies the opacity of the group as a whole.
ADDTOREFLOW (true | false) #IMPLIED	Not applicable.	If true, adds this group (of text and/or picture boxes) to the project's reflow article. Equivalent to the Digital Publishing > Add to Reflow command in QuarkXPress.	Not applicable.
ARTICLENAME CDATA #IMPLIED	Not applicable.	Specifies the name of the project's reflow article (to which this group is being added as a component). If no reflow article exists and you do not include this attribute, the default reflow article name is used.	Not applicable.

GROUPCHARACTERS (Modifier schema)

Element type	Construct	Modify	Deconstruct
GROUPCHARACT	Combines a	Combines a series of	Combines a

Element type	Construct	Modify	Deconstruct
ERS ((RICHTEXT | HIDDEN)+)	series of characters into a unit always runs horizontally even if the story direction is vertical. Grouped characters do not break at the end of a line.	characters into a unit always runs horizontally even if the story direction is vertical. Grouped characters do not break at the end of a line.	series of characters into a unit always runs horizontally even if the story direction is vertical. Grouped characters do not break at the end of a line.
Attributes			
SCALEDIRECTION (HORIZONTAL | VERTICAL) #IMPLIED	Specifies the direction in which text is scaled. Works only when the story direction is vertical.	Specifies the direction in which text is scaled. Works only when the story direction is vertical.	Specifies the direction in which text is scaled. Works only when the story direction is vertical.
SCALEAMOUNT CDATA #IMPLIED	Specifies the scaling percentage. Works only when the story direction is vertical.	Specifies the scaling percentage. Works only when the story direction is vertical.	Specifies the scaling percentage. Works only when the story direction is vertical.
SENDING CDATA #IMPLIED	Specifies the sending amount. (Sending is similar to kerning, but applicable as a fixed value over a range of text.)	Specifies the sending amount. (Sending is similar to kerning, but applicable as a fixed value over a range of text.)	Specifies the sending amount. (Sending is similar to kerning, but applicable as a fixed value over a range of text.)
TRACKAMOUNT CDATA #IMPLIED	Specifies the amount of tracking applied to text, in	Specifies the amount of tracking applied to text, in 1/200ths of an em space.	Specifies the amount of tracking applied to text, in

Element type	Construct	Modify	Deconstruct
	1/200ths of an em space.		1/200ths of an em space.

GROWACROSS (Modifier schema)

Element type	Construct	Modify	Deconstruct
GROWACROSS (#PCDATA)	Not applicable.	Expands a box horizontally to the right by the specified number of points. Note: A box can be expanded on the same page or on other spreads and pages.	Not applicable.

GROWDOWN (Modifier schema)

Element type	Construct	Modify	Deconstruct
GROWDOWN (#PCDATA)	Not applicable.	Expands a box vertically toward the bottom of the page by the specified number of points. Note: A box can be expanded on the same page or on other spreads and pages.	Not applicable.

GUIDE (Modifier Schema)

Element type	Construct	Modify	Deconstruct
GUIDE(LINE*)	Describes a guide on a page	Describes a guide on a page or spread.	Describes a guide on a page or

Element type	Construct	Modify	Deconstruct
	or spread.		spread.
Attributes			
INDEX CDATA #REQUIRED	Specifies the index of the guide.	Specifies the index of the guide.	Specifies the index of the guide.
LOCATION CDATA #IMPLIED	Specifies the location of the guide.	Specifies the location of the guide.	Specifies the location of the guide.
ORIENTATION (HORIZONTAL | VERTICAL) #IMPLIED	Specifies the orientation of the guide.	Specifies the orientation of the guide.	Specifies the orientation of the guide.
COLOR CDATA #IMPLIED	Specifies the color of the guide.	Specifies the color of the guide.	Specifies the color of the guide.
VIEWSCALE CDATA #IMPLIED	Specifies the viewscale percentage of the guide, as an integer percentage from 0 to 100.	Specifies the viewscale percentage of the guide, as an integer percentage from 0 to 100.	Specifies the viewscale percentage of the guide, as an integer percentage from 0 to 100.
LOCKED (true | false) #IMPLIED	Specifies whether the guide is locked on the page/spread.	Specifies whether the guide is locked on the page/spread.	Specifies whether the guide is locked on the page/spread.

HEADER (Modifier schema)

Element type	Construct	Modify	Deconstruct
HEADER (ROW*, CONTINUEDHEADER)	Specifies if the row is to be a header row.	Specifies if the row is to be a header row.	Indicates if the row is to be a header row.
Attributes			
HEADERROWS CDATA #IMPLIED	Specifies number of header row.	Specifies number of header row.	Specifies number of header row.

HEADTROWSTYLE (Modifier schema)

Element type	Construct	Modify	Deconstruct
HEADTROWSTYLE (TOPGRID?, BOTTOMGRID?)	Defines a style for header rows in an <INLINETABLE>.	Defines a style for header rows in an <INLINETABLE>.	Not applicable.
Attributes			
PARASTYLE CDATA #IMPLIED	Identifies the paragraph style sheet for the row style.	Identifies the paragraph style sheet for the row style.	Not applicable.
ALIGNMENT (LEFT | RIGHT | CENTER | JUSTIFIED | FORCED) #IMPLIED	Identifies the paragraph alignment for the row style.	Identifies the paragraph alignment for the row style.	Not applicable.
ANGLE CDATA #IMPLIED	Identifies the text angle for the row style.	Identifies the text angle for the row style.	Not applicable.
VALIGN (TOP | CENTER | BOTTOM) #IMPLIED	Specifies the vertical alignment of the row style.	Specifies the vertical alignment of the row style.	Not applicable.
COLOR CDATA	Specifies the	Specifies the	Not applicable.

Element type	Construct	Modify	Deconstruct
#IMPLIED	background color of the row style.	background color of the row style.	
SHADE CDATA #IMPLIED	Specifies the background shade of the row style.	Specifies the background shade of the row style.	Not applicable.
INSET CDATA #IMPLIED	Specifies the text inset of the row style.	Specifies the text inset of the row style.	Not applicable.

HEIGHT(Modifier schema)

Element type	Construct	Modify	Deconstruct
HEIGHT (#PCDATA)	Indicates the height of an item.	Indicates the height of an item.	Indicates the height of an item.

HIDDEN (Modifier schema)

➡ For more information, see "*Working with hidden text*."

Element type	Construct	Modify	Deconstruct
HIDDEN (RICHTEXT)*	Given the OPCODE and OWNER, this will specify hidden text within the project.	Given the OPCODE and OWNER, this will specify hidden text within the project.	Given the OPCODE and OWNER, this will specify hidden text within the project.
Attributes			
DATALEN CDATA #IMPLIED	Not applicable.	Not applicable.	Number of characters the hidden text spans. Note that if the DATALEN attribute does not

Element type	Construct	Modify	Deconstruct
			match the length of the hidden data provided as a RICHTEXT child of HIDDEN, then the text in subsequent RICHTEXT elements will be included within the hidden text created in QuarkXPress, up to the length specified for DATALEN. This can result in data intended for the page being included in the hidden text data, and therefore being hidden from view in QuarkXPress and in output. It is critical to ensure that the length of data intended as hidden text matches the value of the DATALEN attribute to avoid data loss.
OPCODE CDATA #REQUIRED	Hidden text opcode is a four-byte field	Hidden text opcode is a four-byte field that contains	Hidden text opcode is a four-byte field that

Element type	Construct	Modify	Deconstruct
	that contains ownerId, opcodeId,and hiddenTextType. The Hidden text opcode is usually the originating XTensions ID of the XTensions that owns this hidden text. Note that you MUST be certain that the handling XTensions will correctly understand the data being passed, and handle any errors. XTensions that are not designed to handle inappropriate data may cause QuarkXPress Server to unexpectedly quit.	ownerId, opcodeId,and hiddenTextType. The Hidden text opcode is usually the originating XTensions ID of the XTensions that owns this hidden text. Note that you MUST be certain that the handling XTensions will correctly understand the data being passed, and handle any errors. XTensions that are not designed to handle inappropriate data may cause QuarkXPress Server to unexpectedly quit.	contains ownerId, opcodeId,and hiddenTextType. The Hidden text opcode is usually the originating XTensions ID of the XTensions that owns this hidden text.
OWNER CDATA #IMPLIED	Represents the XTensions ID of the XTensions software that owns this hidden text.	Represents the XTensions ID of the XTensions software that owns this hidden text.	
TYPE	The type of	The type of hidden	The type of

Element type	Construct	Modify	Deconstruct
(OPENPAREN | CLOSEPAREN | NONPAREN | CHARACTERTYPE) #IMPLIED	hidden text, as described in the XDK.	text, as described in the XDK.	hidden text, as described in the XDK.

HYPERLINK (Modifier schema)

Element type	Construct	Modify	Deconstruct
HYPERLINK (empty)	Not applicable.	Not applicable.	Defines a hyperlink.
Attributes			
NAME CDATA #IMPLIED	Not applicable.	Not applicable.	The name of the hyperlink.
TARGET CDATA #REQUIRED	Not applicable.	Not applicable.	The name of the hyperlink target.
HLTYPE (WWWURL | PAGE | ANCHOR) #REQUIRED	Not applicable.	Not applicable.	Specifies the type of hyperlink. Options include WWWURL (a URL on the Web), PAGE (the top of a page in the same layout), and ANCHOR (an anchor).

ID (Modifier schema)

Element type	Construct	Modify	Deconstruct
ID (empty)	Lets you specify a name for a LAYOUT, LAYER, BOX, LINKEDBOX,	Identifies an object by its UID or NAME. Note: QuarkXPress Server evaluates the ID element for a	Identifies an object by its unique ID and by its name (if any).If a NAME value

Element type	Construct	Modify	Deconstruct
			exists, the NAME displays in the content of the ID element: <ID UID=456 NAME=Name of box>Name of box</ID>If a NAME value does not exist, the UID displays in the content of the ID element: <ID UID=457>457</ID >Note: If a NAME value does not exist for a box, the word Box and the box UID are concatenated and display in the XML.
	TABLE, GROUP, or COMPOSITIONZONE.Lets you specify a unique ID for a SPREAD or PAGE.	NAME value first and for a UID second. If a NAME is found, the UID is ignored.	
Attributes			
NAME CDATA #IMPLIED	The name of the parent element. The NAME is assigned to QuarkXPress elements during document construction. For example, NAME="BOX1" would be assigned to a box after it has been constructed.Req	The name of the LAYOUT, LAYER, SPREAD, BOX, TABLE, GROUP, or element to be modified.	The name of the parent element.

Element type	Construct	Modify	Deconstruct
	uired for LAYOUT, LAYER, BOX, TABLE, GROUP, and COMPOSITIONZONE elements. QuarkXPress Server automatically assigns a UID to such elements.Ignored for spreads and pages.		
UID CDATA #IMPLIED	Required for PAGE and SPREAD elements. Ignored for all other element types.	The unique ID of the element to be modified.	Specifies the unique ID of an element in the QuarkXPress project.

INDEXTERM (Modifier schema)

Element type	Construct	Modify	Deconstruct
INDEXTERM (RICHTEXT)	Lets you mark indexes within the flow.	Lets you mark indexes within the flow.	Not applicable.
Attributes			
RANGE (UNTILPARASTYLE | NUMBEROFPARAGRAPHS | ENDOFSTORY | START)	Specifies the beginning and end of the scope of the occurrence of the index within the flow.	Specifies the beginning and end of the scope of the occurrence of the index within the flow.	Not applicable.

Element type	Construct	Modify	Deconstruct
ADDALL (true | false) #IMPLIED	Specifies that all instances of the index term in the document should be added to the index.	Specifies that all instances of the index term in the document should be added to the index.	Not applicable.
MAINTERM CDATA #IMPLIED	Specifies the level of the index term.	Specifies the level of the index term.	Not applicable.
SUBTERM1 CDATA #IMPLIED	Specifies the level of the index term.	Specifies the level of the index term.	Not applicable.
SUBTERM2 CDATA #IMPLIED	Specifies the level of the index term.	Specifies the level of the index term.	Not applicable.
SUBTERM3 CDATA #IMPLIED	Specifies the level of the index term.	Specifies the level of the index term.	Not applicable.
CROSSREFERENCETOINDEX CDATA #IMPLIED	Creates a cross references between index terms.	Creates a cross references between index terms.	Not applicable.

INLINEBOX (Modifier schema)

Element type	Construct	Modify	Deconstruct
INLINEBOX (BOXATTRIBUTE?, FRAME?, RUNAROUND?, SHADOW?, TEXTATTRIBUTE?, PICTURE?, (CONTENT? | (PARAGRAPH | RICHTEXT)*),	Lets you create an anchored text or picture box inline rather than referencing it.	Lets you create an anchored text or picture box inline rather than referencing it.	Not applicable.

Element type	Construct	Modify	Deconstruct
INTERACTIVITY?)			
Attributes			
WIDTH CDATA #IMPLIED	Specifies the width of the box as a percentage of the width of the parent column or text box, in all supported measurement units. The default WIDTH value is 100%.The default height for a picture box is the height of the picture it contains at a scale of 100% (or after SCALEUP is applied, if specified). The default width is the width of the picture at that scale. If SCALEUP="false " and the picture is too big to fit the WIDTH value, the picture is scaled down until it fits. The height of a text	Specifies the width of the box as a percentage of the width of the parent column or text box, in all supported measurement units. The default WIDTH value is 100%.The default height for a picture box is the height of the picture it contains at a scale of 100% (or after SCALEUP is applied, if specified). The default width is the width of the picture at that scale. If SCALEUP="false" and the picture is too big to fit the WIDTH value, the picture is scaled down until it fits. The height of a text box is determined by the amount of content it contains after text-fitting (if any).	Not applicable.

Element type	Construct	Modify	Deconstruct
	box is determined by the amount of content it contains after text-fitting (if any).		
SCALEUP CDATA (true | false) #IMPLIED	Applicable only if you are inserting a picture into the box, and only if the picture is narrower in width than the box. If SCALEUP="true", the picture is sized up to fill the full width of the box. If SCALEUP="false", the picture is not resized. In both cases, the height of the box is adjusted to match the height of the picture.	Applicable only if you are inserting a picture into the box, and only if the picture is narrower in width than the box. If SCALEUP="true", the picture is sized up to fill the full width of the box. If SCALEUP="false", the picture is not resized. In both cases, the height of the box is adjusted to match the height of the picture.	Not applicable.
ALIGNWITHTEXT (ASCENT | BASELINE) "BASELINE"	Specifies whether the anchored box is aligned with the ascent or the baseline of the line in which it occurs.	Specifies whether the anchored box is aligned with the ascent or the baseline of the line in which it occurs.	Not applicable.

Element type	Construct	Modify	Deconstruct
OFFSET CDATA #IMPLIED	Specifies the offset of the anchored box from the ascent or baseline.	Specifies the offset of the anchored box from the ascent or baseline.	Not applicable.
ITEMSTYLE	Specifies the pre-defined item style to be applied to the inline box. To learn more about Item Style functionality in QuarkXPress 2017, refer to the "Item Styles" section of the *QuarkXPress 2017 User Guide* found here: *https://files.quark.com/download/documentation/QuarkXPress/2017/English/QXP-2017-User-Guide-EN.pdf*. The newly introduced Modifier XML markup maps/corresponds to the QuarkXPress 2017 functionality.	Specifies the pre-defined item style to be applied to the inline box.	Not applicable.

INLINETABLE (Modifier schema)

Element type	Construct	Modify	Deconstruct
INLINETABLE (FRAME?, COLGROUP?, THEAD?, TBODY,TFOOT?,SHADOW RUNAROUND)	Describes a table that is anchored in a text box.	Describes a table that is anchored in a text box.	Not applicable.
Attributes			
WIDTH CDATA #IMPLIED	Specifies the width of the table, in all supported measurement units. Can be specified as a percentage of the width of the parent text box.	Specifies the width of the table, in all supported measurement units. Can be specified as a percentage of the width of the parent text box.	Not applicable.
SPAN CDATA #IMPLIED	Specifies how much of the page the table will span. (PAGEWIDTH, ALLCOLUMNS, or any number of columns.).	Specifies how much of the page the table will span. (PAGEWIDTH, ALLCOLUMNS, or any number of columns.).	Not applicable.
TABLESTYLEREF CDATA #IMPLIED	Identifies the <TABLESTYLE> that should be used to style the table.	Identifies the <TABLESTYLE> that should be used to style the table.	Not applicable.
ORIENTATION (LANDSCAPE | PORTRAIT) #IMPLIED	Specifies the orientation of the table.	Specifies the orientation of the table.	Not applicable.

INS (Modifier schema)

Element type	Construct	Modify	Deconstruct
INS (empty)	Describes the beginning or end of a tracked insertion in text.	Describes the beginning or end of a tracked insertion in text.	Describes the beginning or end of a tracked insertion in text.
Attributes			
CREATEDBY CDATA #IMPLIED	Not applicable.	Not applicable.	The username of the creator of the insertion.
CREATEDON CDATA #IMPLIED	Not applicable.	Not applicable.	The creation date of the insertion.
STATUS (BEGIN | END) #IMPLIED	If BEGIN, this is the beginning of an insertion. If END, this is the end of an insertion.	If BEGIN, this is the beginning of an insertion. If END, this is the end of an insertion.	If BEGIN, this is the beginning of an insertion. If END, this is the end of an insertion.

INSET (Modifier schema)

Element type	Construct	Modify	Deconstruct
INSET (empty)	Specifies the distance between the inside border of a text box and the text.	Specifies the distance between the inside border of a text box and the text.	Indicates the distance between the inside border of a text box and the text.
Attributes			
MULTIPLEINSETS (true | false | none) “none”	Specifies multiple insets.	Specifies multiple insets.	Indicates multiple insets.
TOP CDATA #IMPLIED	Specifies the distance between the top inside border of	Specifies the distance between the top inside border of a text box and the text.	Indicates the distance between the top inside border of a text

Element type	Construct	Modify	Deconstruct
	a text box and the text.		box and the text.
BOTTOM CDATA #IMPLIED	Specifies the distance between the bottom inside border of a text box and the text.	Specifies the distance between the bottom inside border of a text box and the text.	Indicates the distance between the bottom inside border of a text box and the text.
RIGHT CDATA #IMPLIED	Specifies the distance between the right inside border of a text box and the text.	Specifies the distance between the right inside border of a text box and the text.	Indicates the distance between the right inside border of a text box and the text.
LEFT CDATA #IMPLIED	Specifies the distance between the right inside border of a text box and the text.	Specifies the distance between the right inside border of a text box and the text.	Indicates the distance between the right inside border of a text box and the text.
ALLEDGES CDATA #IMPLIED	Specifies the distance between the inside border of all sides of a text box and the text.	Specifies the distance between the inside border of all sides of a text box and the text.	Indicates the distance between the inside border of all sides of a text box and the text.

INTERACTIVITY (Modifier schema)

Note that the content of an <INTERACTIVITY> element is defined by the XTensions module that owns it. Consequently, no schema for that content is presented here.

Element type	Construct	Modify	Deconstruct
INTERACTIVITY (DATAPROVIDER ?)	Describes AVE interactivity added to an App	Describes AVE interactivity added to an App Studio layout	Describes AVE interactivity added to an App

Element type	Construct	Modify	Deconstruct
	Studio layout or Print layout.	or Print layout.	Studio layout or Print layout.
Attributes			
OWNERXTID CDATA #IMPLIED	Identifies the XTensions module that enables this type of interactivity.	Identifies the XTensions module that enables this type of interactivity.	Identifies the XTensions module that enables this type of interactivity.
TYPE CDATA #IMPLIED	Identifies the type of interactivity.	Identifies the type of interactivity.	Identifies the type of interactivity.

ISBN (Modifier schema)

Element type	Construct	Modify	Deconstruct
ISBN (#PCDATA)	Not applicable.	Part of the <EBOOKMETADATA> element. Specifies the ISBN code of an e-book.	Specifies the ISBN code of an e-book.

KEEPLINESTOGETHER (Modifier schema)

Element type	Construct	Modify	Deconstruct
KEEPLINESTOGETHER (empty)	The Keep Lines Together feature specifies whether lines in paragraphs flow together or are separated when they reach the bottoms of columns.	The Keep Lines Together feature specifies whether lines in paragraphs flow together or are separated when they reach the bottoms of columns.	The Keep Lines Together feature specifies whether lines in paragraphs flow together or are separated when they reach the bottoms of columns.
Attributes			

Element type	Construct	Modify	Deconstruct
ENABLED (true | false | none) “none”	Specifies whether or not this feature is enabled.	Specifies whether or not this feature is enabled.	Specifies whether or not this feature is enabled.
ALLLINESINPARA (true | false | none) “none”	Specifies whether this is for all lines in the paragraph or has a specific start and end.	Specifies whether this is for all lines in the paragraph or has a specific start and end.	Specifies whether this is for all lines in the paragraph or has a specific start and end.
STARTLINE CDATA #IMPLIED	Specifies the number of lines at the beginning of a paragraph before wrapping text to keep lines together.	Specifies the number of lines at the beginning of a paragraph before wrapping text to keep lines together.	Specifies the number of lines at the beginning of a paragraph before wrapping text to keep lines together.
ENDLINE CDATA #IMPLIED	Specifies the number of lines at the end of a paragraph before wrapping text to keep lines together.	Specifies the number of lines at the end of a paragraph before wrapping text to keep lines together.	Specifies the number of lines at the end of a paragraph before wrapping text to keep lines together.

KEYWORDS (Modifier schema)

Element type	Construct	Modify	Deconstruct
KEYWORDS (#PCDATA)	Not applicable.	Part of the <EBOOKMETADATA> element. Specifies a comma-separated list of keywords for an e-book.	Specifies a list of keywords for an e-book.

LASTTCOLSTYLE (Modifier schema)

Element type	Construct	Modify	Deconstruct
LASTTCOLSTYLE (LEFTGRID?, RIGHTGRID?)	Defines a style for the last column in an <INLINETABLE>.	Defines a style for the last column in an <INLINETABLE>.	Not applicable.
Attributes			
WIDTH CDATA #IMPLIED	Specifies the width of the column style.	Specifies the width of the column style.	Not applicable.
COLOR CDATA #IMPLIED	Specifies the background color of the column style.	Specifies the background color of the column style.	Not applicable.
SHADE CDATA #IMPLIED	Specifies the background shade of the column style.	Specifies the background shade of the column style.	Not applicable.

LAYER (Modifier schema)

Element type	Construct	Modify	Deconstruct
LAYER (ID, RGBCOLOR)	Describes a layer.	Describes a layer.	Describes a layer.
Attributes			
OPERATION (CREATE | DELETE) #IMPLIED	Not applicable.	Specifies whether to create or delete the indicated layer. Note that when you delete a layer, all items on the layer are deleted.	Not applicable.
VISIBLE (true | false | none) “none”	Specifies whether a layer is visible. Note: In QuarkXPress, this parameter	Specifies whether a layer is visible. Note: In QuarkXPress, this parameter overrides the Visible setting in	Specifies whether a layer should be visible. Note: In QuarkXPress, this parameter

Element type	Construct	Modify	Deconstruct
	overrides the Visible setting in the Layers pane of the Preferences dialog box (QuarkXPress/Edit menu).	the Layers pane of the Preferences dialog box (QuarkXPress/Edit menu).	overrides the Visible setting in the Layers pane of the Preferences dialog box (QuarkXPress/Edit menu).
KEEPRUNAROUND (true | false | none) “none”	Specifies whether text on visible layers runs around text on hidden layers. Note: In QuarkXPress, this parameter overrides the Keep Runaround setting in the Layers pane of the Preferences dialog box (QuarkXPress/Edit menu).	Specifies whether text on visible layers runs around text on hidden layers. Note: In QuarkXPress, this parameter overrides the Keep Runaround setting in the Layers pane of the Preferences dialog box (QuarkXPress/Edit menu).	Specifies whether text on visible layers runs around text on hidden layers. Note: In QuarkXPress, this parameter overrides the Keep Runaround setting in the Layers pane of the Preferences dialog box (QuarkXPress/Edit menu).
LOCKED (true | false | none) “none”	Specifies whether a layer is locked. Note: In QuarkXPress, this parameter overrides the Locked setting in the Layers pane of the Preferences dialog box (QuarkXPress/Edit menu).	Specifies whether a layer is locked. Note: In QuarkXPress, this parameter overrides the Locked setting in the Layers pane of the Preferences dialog box (QuarkXPress/Edit menu).	Specifies whether a layer is locked. Note: In QuarkXPress, this parameter overrides the Locked setting in the Layers pane of the Preferences dialog box (QuarkXPress/Edit menu).

Element type	Construct	Modify	Deconstruct
SUPPRESS (true | false | none) “none”	Specifies whether output of a layer is suppressed. Note: In QuarkXPress, this parameter overrides the Suppress Output setting in the Layers pane of the Preferences dialog box (QuarkXPress/Edit menu).	Specifies whether output of a layer is suppressed. Note: In QuarkXPress, this parameter overrides the Suppress Output setting in the Layers pane of the Preferences dialog box (QuarkXPress/Edit menu).	Specifies whether output of a layer is suppressed. Note: In QuarkXPress, this parameter overrides the Suppress Output setting in the Layers pane of the Preferences dialog box (QuarkXPress/Edit menu).

LAYOUT (Modifier schema)

Element type	Construct	Modify	Deconstruct
LAYOUT (ID, ARTICLE*, LAYER*, EBOOKMETADATA?, HYPERLINK*, (SPREAD | BOX | TABLE | INLINETABLE | INLINEBOX | MASTERPAGESEQUENCE | PAGESEQUENCE)*)	Describes a layout in a project.	Identifies a layout to be modified. Use the ID@NAME or ID@UID attribute to indicate the target layout. Layout numbers start with 1; layout=1 refers to the first layout in the project.If you want to modify existing boxes, regardless of where the boxes appear in the project, boxes to modify can be specified as direct children of the LAYOUT element,	Specifies the layout number in the ID@UID element and the layout name in the ID@NAME element.

Element type	Construct	Modify	Deconstruct
		rather than being enclosed within a specific SPREAD.	
Attributes			
POINTSPERINCH CDATA #IMPLIED	Not applicable.	Not applicable.	Specifies how many points to use per inch for measurements.
MATHSUPERSET CDATA #IMPLIED	Identifies the XPressMath superset (if any) used by this project.	Identifies the XPressMath superset (if any) used by this project.	Identifies the XPressMath superset (if any) used by this project.
MEDIATYPE (PRINT | DIGITAL) #IMPLIED	Not applicable.	Not applicable.	Specifies whether the layout is a Print or App Studio layout.
SHAREDSTATUS (LAYOUT | ALLPROJECTS | THISPROJECT) #IMPLIED	Not applicable.	Not applicable.	Specifies the sharing status of the layout, as specified in the Layout > Advanced Layout Properties dialog box in QuarkXPress.
REFLOWVIEWLAYOUTUID CDATA #IMPLIED	Not applicable.	Not applicable.	Specifies the UID of the Print layout for this Reflow layout. This attribute is valid only for reflow views for Print layouts.
LAYOUTHIDDEN (true | false)	Not applicable.	Not applicable.	If true, the layout is hidden. If false,

Element type	Construct	Modify	Deconstruct
#IMPLIED			the layout is visible.
REFLOWVIEWLAYOUT (true | false) #IMPLIED	Not applicable.	Not applicable.	If true, this layout is a Reflow layout. If false, this layout is not a Reflow layout.
LAYOUT_ID CDATA #IMPLIED	Not applicable.	Not applicable.	Specifies the internal layout UID value for the layout. This value does not change when the project gets modified — unlike the UID value, which is reset when, for example, an intermediate layout is deleted. Note that the LAYOUT/ID@UID attribute represents the index of the layout, rather than the internal UID used by QuarkXPress.
OPERATION (CREATE | DELETE) #IMPLIED	Allows you to create or delete a layout.	Allows you to create or delete a layout.	Not applicable.
LAYOUTSPECIFICATION CDATA #IMPLIED	Lets you specify a specification to be used when you create a	Lets you specify a specification to be used when you create a layout with	Not applicable.

Element type	Construct	Modify	Deconstruct
	layout with the OPERATION attribute.	the OPERATION attribute.	
WIDTH CDATA #IMPLIED	Valid only when you are creating a layout with the OPERATION attribute and you are not using a layout specification. Lets you specify the width of the layout.	Valid only when you are creating a layout with the OPERATION attribute and you are not using a layout specification. Lets you specify the width of the layout.	Not applicable.
HEIGHT CDATA #IMPLIED	Valid only when you are creating a layout with the OPERATION attribute and you are not using a layout specification. Lets you specify the height of the layout.	Valid only when you are creating a layout with the OPERATION attribute and you are not using a layout specification. Lets you specify the height of the layout.	Not applicable.
DUPLICATEFROMLAYOUT CDATA #IMPLIED	Creates a duplicate of the layout with this name. For example:<LAYOUT OPERATION="CREATE" DUPLICATEFROMLAYOUT= “Layout 1"> <ID NAME="new"/>	Creates a duplicate of the layout with this name. For example:<LAYOUT OPERATION="CREATE" DUPLICATEFROMLAYOUT= “Layout 1"> <ID NAME="new"/> </LAYOUT>	Not applicable.

Element type	Construct	Modify	Deconstruct
	</LAYOUT>		

LAYOUTMETADATA (Modifier schema)

Element type	Construct	Modify	Deconstruct
LAYOUTMETADATA (ID)	Specifies the metadata to apply to the layout for eBook and tagged PDF export.	Not applicable.	Not applicable..
Attributes			
TITLE CDATA #IMPLIED	Specifies the title of the e-book or PDF.	Not applicable.	Not applicable.
SUBJECT CDATA #IMPLIED	Specifies the subject of the e-book or PDF.	Not applicable.	Not applicable.
AUTHOR CDATA #IMPLIED	Specifies the author of the e-book or PDF.	Not applicable.	Not applicable.
KEYWORDS CDATA #IMPLIED	Specifies keywords for the e-book or PDF.	Not applicable.	Not applicable.
DESCRIPTION CDATA #IMPLIED	Specifies a description for the e-book or PDF.	Not applicable.	Not applicable.
ISBN CDATA #IMPLIED	Specifies the ISBN of the e-book or PDF.	Not applicable.	Not applicable.
PUBLISHER CDATA #IMPLIED	Specifies the publisher of the e-book or PDF.	Not applicable.	Not applicable.
COPYRIGHT	Specifies the	Not applicable.	Not applicable.

Element type	Construct	Modify	Deconstruct
CDATA #IMPLIED	copyright of the e-book or PDF.		

LAYOUTREF (Modifier schema)

Element type	Construct	Modify	Deconstruct
LAYOUTREF (ID)	Not applicable.	Not applicable.	Identifies a layout that is a member of a layout family (a PUBLICATIONCHANNEL).
Attributes			
ORIENTATION (HORIZONTAL | VERTICAL) #IMPLIED	Not applicable.	Not applicable.	Identifies the orientation of the layout.

LEFT (Modifier schema)

Element type	Construct	Modify	Deconstruct
LEFT (#PCDATA)	The distance between the box or lines left edge and the left edge of the page, in points.	The distance between the box or lines left edge and the left edge of the page, in points.	The distance between the box or lines left edge and the left edge of the page, in points.

LEFTCONTROLPOINT (Modifier schema)

Element type	Construct	Modify	Deconstruct
LEFTCONTROLPOINT (empty)	Each point on a curve is described by three geometric	Each point on a curve is described by three geometric positions: the x,y coordinate of	Each point on a curve is described by three geometric

Element type	Construct	Modify	Deconstruct
	positions: the x,y coordinate of the vertex point (this coordinate is relative to the bounding geometry of the shape, not the page), and the left and right control handles–as you would see onscreen in the QuarkXPress user environment. For more information on drawing and manipulating bezier curves, please see A Guide to QuarkXPress.	the vertex point (this coordinate is relative to the bounding geometry of the shape, not the page), and the left and right control handles–as you would see onscreen in the QuarkXPress user environment. For more information on drawing and manipulating bezier curves, please see A Guide to QuarkXPress.	positions: the x,y coordinate of the vertex point (this coordinate is relative to the bounding geometry of the shape, not the page), and the left and right control handles–as you would see onscreen in the QuarkXPress user environment. For more information on drawing and manipulating bezier curves, please see A Guide to QuarkXPress.

LEFTGRID (Modifier schema)

Element type	Construct	Modify	Deconstruct
LEFTGRID (empty)	Describes a grid line on the left edge of a cell in an <INLINETABLE>.	Describes a grid line on the left edge of a cell in an <INLINETABLE>.	Not applicable.
Attributes			
TYPE (TOP | LEFT | BOTTOM | RIGHT)	Specifies the location of the grid line.	Specifies the location of the grid line.	Not applicable.

Element type	Construct	Modify	Deconstruct
#IMPLIED			
STYLE CDATA #IMPLIED	Identifies the <TABLESTYLE> that styles this grid line. If you specify this value, you do not have to specify the remaining attributes. If you specify the remaining attributes, those attribute values override the corresponding <TABLESTYLE> values.	Identifies the <TABLESTYLE> that styles this grid line. If you specify this value, you do not have to specify the remaining attributes. If you specify the remaining attributes, those attribute values override the corresponding <TABLESTYLE> values.	Not applicable.
WIDTH CDATA #IMPLIED	Specifies the width of the grid line in points.	Specifies the width of the grid line in points	Not applicable.
COLOR CDATA #IMPLIED	Specifies the color of the grid line.	Specifies the color of the grid line.	Not applicable.
SHADE CDATA #IMPLIED	Specifies the shade of the grid line.	Specifies the shade of the grid line.	Not applicable.
OPACITY CDATA #IMPLIED	Specifies the opacity of the grid line.	Specifies the opacity of the grid line.	Not applicable.
GAPCOLOR CDATA #IMPLIED	Specifies the color of the gap (if any) between the lines that make up the grid line.	Specifies the color of the gap (if any) between the lines that make up the grid line.	Not applicable.

Element type	Construct	Modify	Deconstruct
GAPSHADE CDATA #IMPLIED	Specifies the shade of the gap (if any) between the lines that make up the grid line.	Specifies the shade of the gap (if any) between the lines that make up the grid line.	Not applicable.
GAPOPACITY CDATA #IMPLIED	Specifies the opacity of the gap (if any) between the lines that make up the grid line.	Specifies the opacity of the gap (if any) between the lines that make up the grid line.	Not applicable.

LINESTYLE (Modifier schema)

Element type	Construct	Modify	Deconstruct
LINESTYLE (empty)	Describes a Dashes & Stripes style that can be applied to lines or frames.	Describes a Dashes & Stripes style that can be applied to lines or frames.	Describes a Dashes & Stripes style that can be applied to lines or frames.
Attributes			
ARROWHEADS (PLAINLINE | LEFTARROW | RIGHTARROW | LEFTFARROW | RIGHTFARROW | DOUBLEARROW) “PLAINLINE”	Specifies whether a line should have arrows on its ends:PLAINLINE = No arrows LEFTARROW = Arrow head on left endRIGHTARROW = Arrow head on right endLEFTFARROW = Arrow head	Specifies whether a line should have arrows on its ends:PLAINLINE = No arrows LEFTARROW = Arrow head on left endRIGHTARROW = Arrow head on right endLEFTFARROW = Arrow head on left end, arrow tail on right endRIGHTFARROW =	Specifies whether a line has arrows on its ends:PLAINLINE = No arrowsLEFTARROW = Arrow head on left endRIGHTARROW = Arrow head on right endLEFTFARROW = Arrow head on left end, arrow tail

Element type	Construct	Modify	Deconstruct
	on left end, arrow tail on right endRIGHTFARROW = Arrow head on right end, arrow tail on left endDOUBLEARROW = Arrow heads on both ends	Arrow head on right end, arrow tail on left endDOUBLEARROW = Arrow heads on both ends	on right endRIGHTFARROW = Arrow head on right end, arrow tail on left endDOUBLEARROW = Arrow heads on both ends

LINKEDBOX (Modifier schema)

Element type	Construct	Modify	Deconstruct
LINKEDBOX (ID)	Represents a box or table cell into which text flows from the parent box. The child ID element identifies the box or table.To force text to run into the next box or cell in a chain, insert the boxbreak character entity where you want the text to break.	Represents a box or table cell into which text flows from the parent box. The child ID element identifies the box or table.To force text to run into the next box or cell in a chain, insert the boxbreak character entity where you want the text to break.	Identifies the point where the text has overflowed the current box and identifies the box or table cell where the text continues. Example:<BOX> <ID NAME="Box5" UID="5"/> <TEXT> <STORY STORYDIRECTION="HORIZONTAL"> <LINKEDBOX ENDOFFSET="94" STARTOFFSET="55"> <ID NAME="Box6" UID="6"/> </LINKEDBOX>

Element type	Construct	Modify	Deconstruct
			<LINKEDBOX ENDOFFSET="108" STARTOFFSET="95"> <ID NAME="Box7" UID="7"/> </LINKEDBOX> <PARAGRAPH MERGE="false" PARASTYLE="Normal"> <RICHTEXT MERGE="false"> Text is here.</RICHTEXT> </PARAGRAPH> </STORY> </TEXT> </BOX>
Attributes			
STARTOFFSET CDATA #IMPLIED	Not applicable.	Not applicable.	Offset of the first character in the next box or cell in the chain.
ENDOFFSET CDATA #IMPLIED	Not applicable.	Not applicable.	Offset of the last character in the next boxor cell in the chain.
ROWCOUNT CDATA #IMPLIED	If a LINKEDBOX is a table cell, this attribute identifies the row of the cell. Otherwise, not applicable.	If a LINKEDBOX is a table cell, this attribute identifies the row of the cell. Otherwise, not applicable.	If a LINKEDBOX is a table cell, this attribute identifies the row of the cell. Otherwise, not applicable.
COLUMNCOUNT CDATA #IMPLIED	If a LINKEDBOX is a table cell, this attribute	If a LINKEDBOX is a table cell, this attribute identifies	If a LINKEDBOX is a table cell, this attribute identifies

Element type	Construct	Modify	Deconstruct
	identifies the column of the cell. Otherwise, not applicable.	the column of the cell. Otherwise, not applicable.	the column of the cell. Otherwise, not applicable.

LIST (Modifier schema)

Element type	Construct	Modify	Deconstruct
LIST ((PARAGRAPH | RICHTEXT)*,OVERMATTER?)	Specifies a List in a QuarkXPress project.	Specifies a List in a QuarkXPress project.	Identifies a List in a QuarkXPress project.
Attributes			
OPERATION (CREATE | DELETE) #IMPLIED	Not applicable.	Specifies whether to create a list or delete a list.	Not applicable.
LISTSTYLE CDATA #REQUIRED	Name of the List as defined in QuarkXPress.	Name of the List as defined in QuarkXPress.	Name of the List as defined in QuarkXPress.

LOCATION (Modifier schema)

Element type	Construct	Modify	Deconstruct
LOCATION (empty)	Lets you specify the maximum or minimum location on the page of a box's upper-left corner for a fit-box-to-content operation.	Lets you specify the maximum or minimum location on the page of a box's upper-left corner for a fit-box-to-content operation.	Not applicable.
Attributes			
X CDATA	The largest or	The largest or	Not applicable.

Element type	Construct	Modify	Deconstruct
#REQUIRED	smallest allowable coordinate for the left side of the resized box.	smallest allowable coordinate for the left side of the resized box.	
Y CDATA #REQUIRED	The largest or smallest allowable coordinate for the top side of the resized box.	The largest or smallest allowable coordinate for the top side of the resized box.	Not applicable.

LOCKTOGRID (Modifier schema)

Element type	Construct	Modify	Deconstruct
LOCKTOGRID (empty)	Specifies whether this paragraph is locked to the baseline grid. You can choose to lock to the page grid or the text box grid.	Specifies whether this paragraph is locked to the baseline grid. You can choose to lock to the page grid or the text box grid.	Specifies whether this paragraph is locked to the baseline grid. You can choose to lock to the page grid or the text box grid.
Attributes			
ENABLED (true | false | none) “none”	Specifies whether LOCKTOGRID is enabled.	Specifies whether LOCKTOGRID is enabled.	Specifies whether LOCKTOGRID is enabled.
GRIDLEVEL (PAGE | TEXTBOX) “PAGE”	Specifies whether GRID applies on page level or text box level.	Specifies whether GRID applies on page level or text box level.	Specifies whether GRID applies on page level or text box level.
GRIDTYPE (TOPLINE |	Specifies grid type applied on	Specifies grid type applied on page level	Specifies grid type applied on

Element type	Construct	Modify	Deconstruct
BOTTOMLINE | LEFTLINE | RIGHTLINE | CENTERLINE | BASELINE) “BASELINE”	page level or text box level grid.	or text box level grid.	page level or text box level grid.

MASTERPAGESEQUENCE (Modifier schema)

Element type	Construct	Modify	Deconstruct
MASTERPAGESEQUENCE (NAME (SINGLEMASTERPAGEREFERENCE | REPEATABLEMASTERPAGEREFERENCE | REPEATABLEMASTERPAGEALTERNATIVES)	Not applicable.	Enables you to define the application of master pages during the pagination process. It is identified by a unique name and can be referenced by name as many times as needed. The children of MASTERPAGESEQUENCE are termed sub-sequences.	Not applicable.
Attributes			
NAME	Not applicable.	Specifies the unique name of the MASTERPAGESEQUENCE	Not applicable.

MATHEQUATION (Modifier schema)

Element type	Construct	Modify	Deconstruct
MATHEQUATION (#PCDATA)	Contains an XPressMath equation in	Contains an XPressMath equation in	Contains an XPressMath equation in

Element type	Construct	Modify	Deconstruct
	XPressMath ASCII language format.	XPressMath ASCII language format.	XPressMath ASCII language format.

MAX (Modifier schema)

Element type	Construct	Modify	Deconstruct
MAX (LOCATION | SIZE | SCALETO)	Lets you specify the maximum location, size, or scale of a box for a fit-box-to-content operation.	Lets you specify the maximum location, size, or scale of a box for a fit-box-to-content operation.	Not applicable.

METADATA (Modifier schema)

Element type	Construct	Modify	Deconstruct
METADATA (VALUE+)	Specifies if the box will have metadata associated with it. Metadata takes the form of key/value pairs.	Specifies if the box will have metadata associated with it. Metadata takes the form of key/value pairs.	Describes the metadata associated with the box.

MIN (Modifier schema)

Element type	Construct	Modify	Deconstruct
MIN (LOCATION | SIZE | SCALETO)	Lets you specify the minimum location, size, or scale of a box for a fit-box-to-content	Lets you specify the minimum location, size, or scale of a box for a fit-box-to-content operation.	Not applicable.

Element type	Construct	Modify	Deconstruct
	operation.		

MOVEDOWN (Modifier schema)

Element type	Construct	Modify	Deconstruct
MOVEDOWN (#PCDATA)	Not applicable.	Moves a box down by the specified number of points. Note: You can move a box or line onto another page.	Not applicable.

MOVELEFT (Modifier schema)

Element type	Construct	Modify	Deconstruct
MOVELEFT (#PCDATA)	Not applicable.	Moves a box to the left by the specified number of points. Note: You can move a box or line onto another page.	Not applicable.

MOVERIGHT (Modifier schema)

Element type	Construct	Modify	Deconstruct
MOVERIGHT (#PCDATA)	Not applicable.	Moves a box to the right by the specified number of points. Note: You can move a box or line onto another page.	Not applicable.

MOVEUP (Modifier schema)

Element type	Construct	Modify	Deconstruct
MOVEUP (#PCDATA)	Not applicable.	Moves a box up by the specified number of points. Note: You can move a box or line onto another page.	Not applicable.

NOTE (Modifier schema)

Element type	Construct	Modify	Deconstruct
NOTE (#PCDATA)	Describes a note in text.	Describes a note in text.	Describes a note in text.
Attributes			
CREATEDBY CDATA #IMPLIED	Not applicable.	Not applicable.	The username of the creator of the note.
CREATEDON CDATA #IMPLIED	Not applicable.	Not applicable.	The creation date of the note.
BACKGROUNDCOLOR CDATA #IMPLIED	The background color of the note.	The background color of the note.	The background color of the note.

ODDTROWSTYLE (Modifier schema)

Element type	Construct	Modify	Deconstruct
ODDTROWSTYLE (TOPGRID?, BOTTOMGRID?)	Defines a style for odd rows in an <INLINETABLE>.	Defines a style for odd rows in an <INLINETABLE>	Not applicable.
Attributes			
PARASTYLE CDATA #IMPLIED	Identifies the paragraph style sheet for the row	Identifies the paragraph style sheet for the row	Not applicable.

Element type	Construct	Modify	Deconstruct
	style.	style.	
ALIGNMENT (LEFT | RIGHT | CENTER | JUSTIFIED | FORCED) #IMPLIED	Identifies the paragraph alignment for the row style.	Identifies the paragraph alignment for the row style.	Not applicable.
ANGLE CDATA #IMPLIED	Identifies the text angle for the row style.	Identifies the text angle for the row style.	Not applicable.
VALIGN (TOP | CENTER | BOTTOM) #IMPLIED	Specifies the vertical alignment of the row style.	Specifies the vertical alignment of the row style.	Not applicable.
COLOR CDATA #IMPLIED	Specifies the background color of the row style.	Specifies the background color of the row style.	Not applicable.
SHADE CDATA #IMPLIED	Specifies the background shade of the row style.	Specifies the background shade of the row style.	Not applicable.
INSET CDATA #IMPLIED	Specifies the text inset of the row style.	Specifies the text inset of the row style.	Not applicable.

ODDTCOLSTYLE (Modifier schema)

Element type	Construct	Modify	Deconstruct
ODDTCOLSTYLE (LEFTGRID?, RIGHTGRID?)	Defines a style for odd columns in an <INLINETABLE>.	Defines a style for odd columns in an <INLINETABLE>.	Not applicable.
Attributes			
WIDTH CDATA	Specifies the	Specifies the width of	Not applicable.

Element type	Construct	Modify	Deconstruct
#IMPLIED	width of the column style.	the column style.	
COLOR CDATA #IMPLIED	Specifies the background color of the column style.	Specifies the background color of the column style.	Not applicable.
SHADE CDATA #IMPLIED	Specifies the background shade of the column style.	Specifies the background shade of the column style.	Not applicable.

ORIGIN (Modifier schema)

Element type	Construct	Modify	Deconstruct
ORIGIN (empty)	Specifies an item's size and its position relative to the upper left corner of its page or spread.	Specifies an item's size and its position relative to the upper left corner of its page or spread.	Specifies an item's size and its position relative to the upper left corner of its page or spread.
Attributes			
X CDATA #REQUIRED	The distance between the left side of the item and the left edge of the page or spread.	The distance between the left side of the item and the left edge of the page or spread.	The distance between the left side of the item and the left edge of the page or spread.
Y CDATA #REQUIRED	The distance between the top side of the item and the top edge of the page or spread.	The distance between the top side of the item and the top edge of the page or spread.	The distance between the top side of the item and the top edge of the page or spread.
RELATIVETO (PAGE | SPREAD)	Indicates whether the	Indicates whether the item's position is	Indicates whether the item's

Element type	Construct	Modify	Deconstruct
“SPREAD"	item's position is relative to the page or to the spread.	relative to the page or to the spread.	position is relative to the page or to the spread.

OVERMATTER (Modifier schema)

Element type	Construct	Modify	Deconstruct
OVERMATTER (PARAGRAPH | RICHTEXT | ANCHOREDBOXREF | GROUPCHARACTERS | HIDDEN | RUBI | CALLOUTANCHOR | INLINEBOX | MATHEQUATION)*	Not applicable.	Not applicable.	Identifies where the current box overflows when there is no subsequence box for text to flow into.

PAGE (Modifier schema)

Element type	Construct	Modify	Deconstruct
PAGE (ID, SECTION?)	PA page to be created.	The page to be created or deleted. Note: To locate a page, for example, for creating a box, you use the GEOMETRY@PAGE attribute in the BOX element.	Indicates a page's absolute page number (in the ID@UID element)Note: Page names are not returned.
Attributes			
OPERATION (CREATE |	Not applicable.	Specifies whether to create or delete the	Not applicable.

Element type	Construct	Modify	Deconstruct
DELETE) #IMPLIED		indicated page.	
MASTER CDATA #IMPLIED	Identifies the master page from which to create a page. This value should be specified as a number, with 3 indicating the first master page. Note: Only the number of a master page is included in this attribute. The definition of the master page is stored in the project's Job Jackets file.	Identifies the master page from which to create a page. This value should be specified as a number, with 3 indicating the first master page.	Identifies the master page that is applied to a page. Specified as a number, with "1" indicating the first master page. Note: Only the number of a master page is included in this attribute. The definition of the master page is stored in the project's Job Jackets file.
POSITION (LEFTOFSPINE | RIGHTOFSPINE) "RIGHTOFSPINE"	Specifies whether a page should be on the left or right side of the spine.	Specifies whether a page should be on the left or right side of the spine.	Specifies whether a page is on the left or right of the spine.
FORMATTEDNAME CDATA #IMPLIED	Not applicable.	Not applicable.	The string that displays in automatically created page numbers. A combination of the PREFIX, FORMAT, and NUMBER for this page's

Element type	Construct	Modify	Deconstruct
			<SECTION> element.

PAGEBREAK (Modifier schema)

Element type	Construct	Modify	Deconstruct
PAGEBREAK (empty)	Allows you to change the master page applied to the next page. You can also control whether the next page is on the left or right, and specify a master page to be inserted if necessary.	Allows you to change the master page applied to the next page. You can also control whether the next page is on the left or right, and specify a master page to be inserted if necessary.	Not applicable.
Attributes			
NEXTPAGE (RECTO | VERSO) "RECTO"	Specifies whether the next page should be right-facing (recto) or left-facing (verso).	Specifies whether the next page should be right-facing (recto) or left-facing (verso).	Not applicable.
BLANKPAGE (YES | NO) "NO"	Indicates whether to insert a blank page between this page and the next page. Applicable only if this page and the next page are both recto or	Indicates whether to insert a blank page between this page and the next page. Applicable only if this page and the next page are both recto or both verso.	Not applicable.

Element type	Construct	Modify	Deconstruct
	both verso.		
BLANKMASTER CDATA #IMPLIED	Specifies the name of the master page to be applied to the inserted blank page (if any).	Specifies the name of the master page to be applied to the inserted blank page (if any).	Not applicable.
NEXTMASTER CDATA #IMPLIED	Specifies the name of the master page to be applied to the next page.	Specifies the name of the master page to be applied to the next page.	Not applicable.

PAGEREF (Modifier schema)

Element type	Construct	Modify	Deconstruct
PAGEREF (empty)	Not applicable.	Not applicable.	Identifies a page within the layout corresponding to a multi-page Composition Zones item.
Attributes			
NUMBER CDATA #IMPLIED	Not applicable.	Not applicable.	Specifies the number of the page currently displayed. You can set this value using the COMPOSITIONZONE element's PREVIEWPAGE attribute.
ANGLE CDATA #IMPLIED	Not applicable.	Not applicable.	Specifies the angle applied to the Composition

Element type	Construct	Modify	Deconstruct
			Zones item.
OFFSETACROSS CDATA #IMPLIED	Not applicable.	Not applicable.	Specifies the horizontal offset applied to the Composition Zones item.
OFFSETDOWN CDATA #IMPLIED	Not applicable.	Not applicable.	Specifies the vertical offset applied to the Composition Zones item.
SCALE CDATA #IMPLIED	Not applicable.	Not applicable.	Specifies the scale applied to the Composition Zones item.

PAGESEQUENCE (Modifier schema)

Element type	Construct	Modify	Deconstruct
PAGESEQUENCE (MASTERREFERENCE*, FORCEPAGECOUNT* (SECTIONNUMBERFORMAT | STATICCONTENT | STORY))	Not applicable.	Enables you to define a sequence of pages which can be a logical section, chapter, article etc. @MASTERREFERENCE allows you to refer to MASTERPAGESEQUENCE or a master page in template.	Not applicable.
Attributes			
MASTERREFERENCE	Not applicable.	Unique name of the MASTERPAGESEQUENCE.	Not applicable.
FORCEPAGECOUNT	Not applicable.	Imposes a condition on the number of	Not applicable.

Element type	Construct	Modify	Deconstruct
		pages in a PAGESEQUENCE. In the event that this constraint is not satisfied, an additional blank page will be added to the end of the sequence. Valid values include:AUTO (Default) The action taken depends on the existence of a succeeding PAGESEQUENCE and the value of the INITIALPAGENUMBER property specified within the SECTIONNUMBERFORMAT element.Even Forces an even page count for the page sequence. Odd Forces an odd page count for the page sequenceENDONEVEN Forces the last page to have an even page numberENDONODD forces the last page to have an odd page numberNOFORCE Does not force any page count	
ORIENTATION	Not applicable.	Specifies the page orientation. The	Not applicable.

Element type	Construct	Modify	Deconstruct
		value can be set to LANDSCAPE or PORTRAIT. If the value is set to LANDSCAPE, the project template must already contain a layout with the name Landscape, because the content of the page sequence will flow into a landscape layout made available in the template.If the value is set to LANDSCAPE, this would result in a mixed mode PDF output.	

PARAGRAPH (Modifier schema)

Element type	Construct	Modify	Deconstruct
PARAGRAPH ((TABSPEC | REFNOTE | RULE | FORMAT | RICHTEXT | COLUMNFLOW | ANCHOREDBOXREF | HIDDEN | GROUPCHARACTERS | RUBI | CALLOUTANCHOR | CONTENT | INLINEBOX |	Describes a paragraph.	Describes a paragraph.	Describes a paragraph.

Element type	Construct	Modify	Deconstruct
XREF | NOTE | INS | DEL | PAGEBREAK | MATHEQUATION)*)			
Attributes			
PARASTYLE CDATA #IMPLIED	Applies a paragraph style sheet to text. Note: Only the name of a paragraph style sheet is included in this attribute. The definition of the style sheet is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	Applies a paragraph style sheet to text. Note: Only the name of a paragraph style sheet is included in this attribute. The definition of the style sheet is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	Identifies the paragraph style sheet applied to a paragraph. Note: Only the name of a paragraph style sheet is included in this attribute. The definition of the style sheet is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.
XREFLABEL CDATA #IMPLIED	Defines a label (name) for a cross reference.	Defines a label (name) for a cross reference.	Defines a label (name) for a cross reference.
PARACHAR (HARDRETURN | VTAB | BOXBREAK) “HARDRETURN”	Defines a breaking character for a paragraph.	Defines a breaking character for a paragraph.	Defines a breaking character for a paragraph.
MERGE (true | false) “false”	Specifies whether formatting from a previous	Specifies whether formatting from a previous PARAGRAPH or	Indicates whether formatting from a previous PARAGRAPH or

Element type	Construct	Modify	Deconstruct
	PARAGRAPH or RICHTEXT element should be carried over to the next.	RICHTEXT element should be carried over to the next.	RICHTEXT element is carried over to the next.
FAUXSTYLE (BOLD | ITALIC | BOLDITALIC | NONE) #IMPLIED	Not applicable.	Not applicable.	Indicates whether a paragraph contains a faux type style (such as a bold face that is constructed by software, as opposed to a bold font).
CONDITIONALSTYLE CDATA #IMPLIED	Applies a conditional style to this paragraph.	Applies a conditional style to this paragraph.	Identifies the conditional style (if any) associated with this paragraph.
INDENTLEVEL CDATA #IMPLIED	Sets the indent level for this paragraph.	Sets the indent level for this paragraph.	Indicates the indent level of this paragraph.
TEXTTAGTYPE CDATA #IMPLIED	Not applicable.	Indicates the label applied to a paragraph. Valid only for reflow articles and QuarkCoypDesk articles. Valid values include: Body Byline FigureCaption FigureCredit Headline Headline2 IndentedParagraph Pullquote SectionChapterName Title Title2	Indicates the label applied to a paragraph. Valid only for reflow articles and QuarkCopyDesk articles. Valid values include: Body Byline FigureCaption FigureCredit Headline Headline2 IndentedParagrap

Element type	Construct	Modify	Deconstruct
		OrderedList UnorderedList	h Pullquote SectionChapterName Title Title2 OrderedList UnorderedList

PARENTTABLE (Modifier schema)

Element type	Construct	Modify	Deconstruct
PARENTTABLE (empty)	Identifies the originating table when a table has been broken.	Identifies the originating table when a table has been broken.	Identifies the originating table when a table has been broken.
Attributes			
NAME CDATA #IMPLIED	Specifies the name of the parent table.	Specifies the name of the parent table.	Specifies the name of the parent table.
UID CDATA #IMPLIED	Not applicable.	Specifies the ID of the parent table assigned from QuarkXPress Server.	Specifies the ID of the parent table assigned from QuarkXPress Server.

PICTURE (Modifier schema)

Element type	Construct	Modify	Deconstruct
PICTURE (empty)	Describes the properties of a picture box.	Describes the properties of a picture box.	Describes the properties of a picture box.
Attributes			
FIT (CENTERPICTURE | FITPICTURETOBOX |	Specifies how a picture should fit within a picture box.CENTERPICTURE = Shifts a	Specifies how a picture should fit within a picture box.CENTERPICTURE = Shifts a picture to	Not applicable.

Element type	Construct	Modify	Deconstruct
FITBOXTOPICTURE | FITPICTURETOBOXPRO | NONE) “NONE”	picture to the center of the picture box without changing the pictures scale.FITPICTURETOBOX = Scales a picture to fit in its box exactly. The picture cannot be reduced to a size smaller than 10% or increased to a size larger than 1000%, both horizontally and vertically.FITBOXTOPICTURE = Resizes a box to fit its picture.FITPICTURETOBOXPRO = Scales a picture in a picture box in such a way that the x scale and y scale of a picture remain the same. The picture cannot be reduced to a size smaller than 10% or increased to a size larger than 1000%,	the center of the picture box without changing the pictures scale.FITPICTURETOBOX = Scales a picture to fit in its box exactly. The picture cannot be reduced to a size smaller than 10% or increased to a size larger than 1000%, both horizontally and vertically.FITBOXTOPICTURE = Resizes a box to fit its picture.FITPICTURETOBOXPRO = Scales a picture in a picture box in such a way that the x scale and y scale of a picture remain the same. The picture cannot be reduced to a size smaller than 10% or increased to a size larger than 1000%, both horizontally and vertically.	

Element type	Construct	Modify	Deconstruct
	both horizontally and vertically.		
SCALEACROSS CDATA #IMPLIED	Specifies the horizontal scale of a picture as an integer percentage from 10 to 1000.	Specifies the horizontal scale of a picture as an integer percentage from 10 to 1000.	Specifies the horizontal scale of a picture as an integer percentage from 10 to 1000.
SCALEDOWN CDATA #IMPLIED	Specifies the vertical scale of a picture as an integer percentage from 10 to 1000.	Specifies the vertical scale of a picture as an integer percentage from 10 to 1000.	Specifies the vertical scale of a picture as an integer percentage from 10 to 1000.
OFFSETACROSS CDATA #IMPLIED	Specifies a horizontal offset for the content of a picture box.	Specifies a horizontal offset for the content of a picture box.	Specifies a horizontal offset for the content of a picture box.
OFFSETDOWN CDATA #IMPLIED	Specifies a vertical offset for the content of a picture box.	Specifies a vertical offset for the content of a picture box.	Specifies a vertical offset for the content of a picture box.
ANGLE CDATA #IMPLIED	Specifies a rotation angle for a picture as a floating-point value between –360 degrees and 360 degrees.	Specifies a rotation angle for a picture as a floating-point value between –360 degrees and 360 degrees.	Specifies a rotation angle for a picture as a floating-point value between –360 degrees and 360 degrees.
SKEW CDATA #IMPLIED	Specifies a skew angle for a picture as a floating-point value from –75 degrees to 75 degrees.	Specifies a skew angle for a picture as a floating-point value from –75 degrees to 75 degrees.	Specifies a skew angle for a picture as a floating-point value from –75 degrees to 75 degrees.

Element type	Construct	Modify	Deconstruct
PICCOLOR CDATA #IMPLIED	Identifies a color to be applied to a grayscale picture. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	Identifies a color to be applied to a grayscale picture. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.	Identifies a color applied to a grayscale picture. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.
SHADE CDATA #IMPLIED	Specifies the shade of the color applied to a grayscale picture, as an integer percentage from 0 to 100.	Specifies the shade of the color applied to a grayscale picture, as an integer percentage from 0 to 100.	Specifies the shade of the color applied to a grayscale picture, as an integer percentage from 0 to 100.
OPACITY CDATA #IMPLIED	Specifies the opacity of a picture, as an integer percentage from 0 to 100.	Specifies the opacity of a picture, as an integer percentage from 0 to 100.	Specifies the opacity of a picture, as an integer percentage from 0 to 100.
PICBACKGROUNDCOLOR CDATA	Identifies the background	Identifies the background color	Identifies the background color

Element type	Construct	Modify	Deconstruct
#IMPLIED	color applied to a grayscale picture.	applied to a grayscale picture.	applied to a grayscale picture.
PICBACKGROUNDSHADE CDATA #IMPLIED	Specifies the shade of the background color applied to a grayscale picture, as an integer percentage from 0 to 100.	Specifies the shade of the background color applied to a grayscale picture, as an integer percentage from 0 to 100.	Specifies the shade of the background color applied to a grayscale picture, as an integer percentage from 0 to 100.
PICBACKGROUNDOPACITY CDATA #IMPLIED	Specifies the opacity of the background color applied to a grayscale picture, as an integer percentage from 0 to 100.	Specifies the opacity of the background color applied to a grayscale picture, as an integer percentage from 0 to 100.	Specifies the opacity of the background color applied to a grayscale picture, as an integer percentage from 0 to 100.
FLIPVERTICAL (true | false | none) “none”	Flips a picture vertically.	Flips a picture vertically. If a picture is already flipped vertically, then this flips the picture back.	Indicates whether a picture has been flipped vertically.
FLIPHORIZONTAL (true | false | none) “none”	Flips a picture horizontally.	Flips a picture horizontally. If a picture is already flipped horizontally, then this flips the picture back.	Indicates whether a picture has been flipped horizontally.
SUPRESSPICT (true | false) “false”	Prevents a picture from being included in output.	Prevents a picture from being included in output.	Prevents a picture from being included in output.
FULLRES (true |	Causes imported	Causes imported	Causes imported

Element type	Construct	Modify	Deconstruct
false | none) "none"	pictures to display at full resolution in QuarkXPress if the picture files are available.	pictures to display at full resolution in QuarkXPress if the picture files are available.	pictures to display at full resolution in QuarkXPress if the picture files are available.
MASK CDATA #IMPLIED	Identifies an alpha channel in the picture file to be used to mask the picture file.	Identifies an alpha channel in the picture file to be used to mask the picture file.	Identifies an alpha channel in the picture file that is being used to mask the picture file.
CLEARPICTURE (true | false) "false"	Not applicable.	Removes the picture (if any) from the box.	Not applicable.
DPI CDATA #IMPLIED	Not applicable.	Not applicable.	Indicates the picture resolution in dots per inch (DPI).
ALTTEXT CDATA #IMPLIED	Specifies alternate text for each picture view.	Not applicable.	Not applicable.

PLACEHOLDER (Modifier schema)

Element type	Construct	Modify	Deconstruct
PLACEHOLDER (#PCDATA)	Not applicable.	Not applicable.	Describes a placeholder inserted in QuarkXPress for use with XML Import XTensions software. Note: To replace placeholders with

Element type	Construct	Modify	Deconstruct
			XML content, use XML Import XTensions software with QuarkXPress, or refer to the thexmldoc and paginate parameters.
Attributes			
OWNER CDATA #REQUIRED	Not applicable.	Not applicable.	The name of the element in the XML or schema that created the Placeholder.

POSITION (Modifier schema)

➡ Rather than using the POSITION element type, you can use the RELPOSITION element type to describe the position of <GEOMETRY> elements relative to the page or to the spread. To return item positions as RELPOSITION elements, use the relativegeometry parameter when deconstructing. For more information, see "*XML*."

Element type	Construct	Modify	Deconstruct
POSITION (TOP, LEFT, BOTTOM, RIGHT)	Specifies the absolute position of a box or line on the page, using coordinates measured in points from the upper-left corner of the page.	Specifies the absolute position of a box or line on the page, using coordinates measured in points from the upper-left corner of the page.	Specifies the absolute position of a box or line on the page, using coordinates measured in points from the upper-left corner of the page.
Attributes			
LOCKPROPORTI	Specifies	Specifies whether	Specifies whether

Element type	Construct	Modify	Deconstruct
ONS (true | false) "false"	whether proportions are locked for an item.	proportions are locked for an item.	proportions are locked for an item.
LOCKPOSITION (true | false) "false"	Specifies whether the position is locked for an item.	Specifies whether the position is locked for an item.	Specifies whether the position is locked for an item.

PROJECT (Modifier schema)

Element type	Construct	Modify	Deconstruct
PROJECT (SAVEAS? | PUBLICATION* | HYPERLINK* | TABLESTYLE* | LAYOUT* | STORY* | CONTENT*)*	Describes the QuarkXPress project using one or more LAYOUT elements and allows you to save a copy of the project.	Identifies the QuarkXPress project being modified and allows you to save a copy of that project.	Identifies the QuarkXPress project being deconstructed.
Attributes			
PROJECTNAME CDATA #IMPLIED	Specifies the name of the file to construct.	Not applicable.	Identifies the QuarkXPress project being deconstructed.
JOBJACKET CDATA #IMPLIED	The name and absolute path (on the server computer) of the Job Jackets file to use during construct.If the Job Jackets file cannot be	Not applicable.	The name and path of the Job Jackets file associated with the deconstructed project.

Element type	Construct	Modify	Deconstruct
	located, cannot be read, or contains invalid XML, an error is returned. Note: You cannot create or modify Job Jackets files using the construct namespace and the modify attribute. To create or modify Job Jackets files, use the Job Jackets Manager dialog box (Utilities menu) in QuarkXPress.		
JOBTICKET CDATA #IMPLIED	The name of the Job Ticket that contains the resources for this project. Note: All resources in the Job Ticket will be added to the project.	Not applicable.	The name of the Job Ticket associated with the deconstructed project.
XMLVERSION CDATA #IMPLIED	Not applicable.	Not applicable.	Identifies the version of QuarkXPress Server from which the XML is being returned. Ensures compatibility with

Element type	Construct	Modify	Deconstruct
			future versions of the schema. For example, the value 8.0 is returned for QuarkXPress Server 8.0.

PUBLICATION (Modifier schema)

Element type	Construct	Modify	Deconstruct
PUBLICATION (ID, PUBLICATIONCHANNEL*)	Not applicable.	Not applicable.	Identifies a set of one or more related layout families (PUBLICATIONCHANNEL elements). For example, a PUBLICATION may contain a PUBLICATIONCHANNEL for iPad and another PUBLICATIONCHANNEL for a different target device.

PUBLICATIONCHANNEL (Modifier schema)

Element type	Construct	Modify	Deconstruct
PUBLICATIONCHANNEL (ID, LAYOUTREF*)	Not applicable.	Not applicable.	Defines a layout family — that is, a set of related layouts (LAYOUTREFs).

Element type	Construct	Modify	Deconstruct
Attributes			
HEIGHT CDATA #IMPLIED	Not applicable.	Not applicable.	Specifies the height of the screen for layout family's target device.
WIDTH CDATA #IMPLIED	Not applicable.	Not applicable.	Specifies the width of the screen for layout family's target device.

PUBLISHER (Modifier schema)

Element type	Construct	Modify	Deconstruct
PUBLISHER (#PCDATA)>	Not applicable.	Specifies the publisher of an e-book.	Specifies the publisher of an e-book.

REFNOTE (Modifier schema)

Element type	Construct	Modify	Deconstruct
REFNOTE	Lets you insert a reference note of type footnote or endnote.	Lets you insert a reference note of type footnote or endnote.	Lets you insert a reference note of type footnote or endnote.
Attributes			
STYLE	Specifies the name of the footnote style to be applied.	Specifies the name of the footnote style to be applied.	Specifies the name of the footnote style to be applied.
RESTARTNUMBERINGVALUE	Allows you to restart the footnote numbering.	Allows you to restart the footnote numbering.	Allows you to restart the footnote numbering.

Element type	Construct	Modify	Deconstruct
REFNOTEBODY	Comprised of PARAGRAPH(S) that form the body of either the footnote or endnote.	Comprised of PARAGRAPH(S) that form the body of either footnote or endnote.	Comprised of PARAGRAPH(S) that form the body of either footnote or endnote.
REFERENCEDTEXTMARKER	Allows you to pass a custom marker corresponding to reference superscript.	Allows you to pass a custom marker corresponding to reference superscript.	Allows you to pass a custom marker corresponding to reference superscript.
REFNOTEBODYMARKER	Allows you to pass a custom marker corresponding to a reference body.	Allows you to pass a custom marker corresponding to a reference body.	Allows you to pass a custom marker corresponding to a reference body.
XREFLABEL	Allows you to specify a label (name) for the reference.	Allows you to specify a label (name) for the reference.	Allows you to specify a label (name) for the reference.

RELPOSITION (Modifier schema)

➡ To return item positions as RELPOSITION elements, use the relativegeometry parameter when deconstructing. For more information, see "*XML*."

Element type	Construct	Modify	Deconstruct
RELPOSITION (ORIGIN, WIDTH, HEIGHT)	Specifies the position of a box or line, using coordinates measured in points from the upper-left corner of the page or	Specifies the position of a box or line, using coordinates measured in points from the upper-left corner of the page or spread.	Specifies the position of a box or line, using coordinates measured in points from the upper-left corner of the page or

Element type	Construct	Modify	Deconstruct
	spread.		spread.

REPEATABLEMASTERPAGEALTERNATIVES (Modifier schema)

Element type	Construct	Modify	Deconstruct
REPEATABLEMASTERPAGEALTERNATIVES (MAXREPEATS (CONDITIONALMASTERPAGEREFERENCE)*)	Not applicable.	This sub-sequence is useful for the conditional application of master pages. The children of the REPEATABLEMASTERPAGEALTERNATIVES element are known as alternatives. Each alternative is represented by a CONDITIONALMASTERPAGEREFERENCE element. The REPEATABLEMASTERPAGEALTERNATIVES element may contain one or more alternatives.	Not applicable.
Attributes			
MAXREPEATS	Not applicable.	Can be used to set an upper limit on the number of pages that may be generated using each of the CONDITIONALMASTERPAGEREFERENCE	Not applicable.

REPEATABLEMASTERPAGEREFERENCE (Modifier

Schema)

Element type	Construct	Modify	Deconstruct
REPEATABLEMASTERPAGEREFERENCE (NAME*, MAXREPEATS*)	Not applicable.	This sub-sequence is useful for constructing runs of identical pages. It allows you to define a sequence in which a master page is applied to multiple pages in the page sequence. Causes a bounded or unbounded sequence of pages to be generated using the same master page. (*Unbounded = Default is hardcoded to 50000)	Not applicable.
Attributes			
NAME	Not applicable.	The name of the master page in the QuarkXPress template to be used.	Not applicable.
MAXREPEATS	Not applicable.	Used to set an upper limit on the number of pages that may be generated using this specifier.	Not applicable.

RGBCOLOR (Modifier schema)

Element type	Construct	Modify	Deconstruct
RGBCOLOR (empty)	Describes an RGB color that can be	Describes an RGB color that can be associated with a	Describes an RGB color that can be associated with a

Element type	Construct	Modify	Deconstruct
	associated with a layer, as displayed in the Layers palette in QuarkXPress.	layer, as displayed in the Layers palette in QuarkXPress.	layer, as displayed in the Layers palette in QuarkXPress.
Attributes			
RED CDATA #IMPLIED	An integer from 0 to 255, indicating the red component of an RGB color.	An integer from 0 to 255, indicating the red component of an RGB color.	An integer from 0 to 255, indicating the red component of an RGB color.
GREEN CDATA #IMPLIED	An integer from 0 to 255, indicating the green component of an RGB color.	An integer from 0 to 255, indicating the green component of an RGB color.	An integer from 0 to 255, indicating the green component of an RGB color.
BLUE CDATA #IMPLIED	An integer from 0 to 255, indicating the blue component of an RGB color.	An integer from 0 to 255, indicating the blue component of an RGB color.	An integer from 0 to 255, indicating the blue component of an RGB color.

RICHTEXT (Modifier schema)

Element type	Construct	Modify	Deconstruct
RICHTEXT (#PCDATA)	Describes formatting for text. Use this element to apply additional formatting besides formatting applied with a paragraph or	Describes formatting for text. Use this element to apply additional formatting besides formatting applied with a paragraph or style sheet. Note: The RICHTEXT element replaces the TYPE	Describes formatting for text, other than formatting applied with a paragraph or style sheet. Note: The RICHTEXT element replaces the TYPE element

Element type	Construct	Modify	Deconstruct
	style sheet. Note: The RICHTEXT element replaces the TYPE element in QuarkXPress Server 7.2 and later.	element in QuarkXPress Server 7.2 and later.	in QuarkXPress Server 7.2 and later.
Attributes			
CHARSTYLE CDATA #IMPLIED	Identifies a character style sheet to be applied to text. Note: Only the name of an H&J specification is included in this attribute. The definition of the H&J specification is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	Identifies a character style sheet to be applied to text.	Identifies the character style sheet applied to text. Note: Only the name of an H&J specification is included in this attribute. The definition of the H&J specification is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.
SHADINGSTYLE NAME	Identifies a text shading style to be applied to text.	Identifies a text shading style to be applied to text.	Identifies a text shading style to be applied to text.
PLAIN (true | false | none)	Removes existing	Removes existing formatting and	Removes existing formatting and

Element type	Construct	Modify	Deconstruct
“none”	formatting and renders text as plain text.	renders text as plain text.	renders text as plain text.
MERGE (true | false) “false”	Specifies whether the formatting from the previous RICHTEXT tag should be carried into this RICHTEXT tag.	Specifies whether the formatting from the previous RICHTEXT tag should be carried into this RICHTEXT tag.	Specifies whether the formatting from the previous RICHTEXT tag should be carried into this RICHTEXT tag.
BOLD (true | false | none) “none”	Applies the bold type style to text.	Applies the bold type style to text.	Identifies the bold type style applied to text.
ITALIC (true | false | none) “none”	Applies the italic type style to text.	Applies the italic type style to text.	Identifies the italic type style applied to text.
FONT CDATA #IMPLIED	Identifies a font to be applied to text.	Identifies a font to be applied to text.	Identifies a font applied to text.
MISSINGFONT (true | false) “false”	If the font is missing on rendering, then this attribute is set to true. This allows you to identify when rendering a portion of text that the original font is missing on the machine where the rendering is taking place, and allows your application to	If the font is missing on rendering, then this attribute is set to true. This allows you to identify when rendering a portion of text that the original font is missing on the machine where the rendering is taking place, and allows your application to substitute the font (overriding the inbuilt font mapping functionality in	If the font is missing on rendering, then this attribute is set to true. This allows you to identify when rendering a portion of text that the original font is missing on the machine where the rendering is taking place, and allows your application to

Element type	Construct	Modify	Deconstruct
	substitute the font (overriding the inbuilt font mapping functionality in QuarkXPress Server). If the font specified in the XML is missing and if the MISSINGFONT attribute is present then this becomes the basis for applying font fallback on the particular text run if the FontFallBack preference is enabled. Otherwise this would cause an error because the required font is missing.	QuarkXPress Server).	substitute the font (overriding the inbuilt font mapping functionality in QuarkXPress Server).
PSFONTNAME CDATA #IMPLIED	Some fonts have different postscript and menu display names. The FONTNAME attribute describes the menu name of the font, and	Some fonts have different postscript and menu display names. The FONTNAME attribute describes the menu name of the font, and PSFONTNAME describes the internal postscript	Some fonts have different postscript and menu display names. The FONTNAME attribute describes the menu name of the font, and

Element type	Construct	Modify	Deconstruct
	PSFONTNAME describes the internal postscript name of the font family.	name of the font family.	PSFONTNAME describes the internal postscript name of the font family.
SIZE CDATA #IMPLIED	Specifies a size for text, from 2 to 720 points.	Specifies a size for text, from 2 to 720 points.	Identifies the size of the text, from 2 to 720 points.
FONTSET CDATA #IMPLIED	Identifies a font set that has been applied to text. Note that you can apply font sets during a Construct operation, but you cannot create them.	Identifies a font set that has been applied to text. Note that you can apply font sets during a Modify operation, but you cannot create them.	Identifies a font set that has been applied to text.
FONTSETSIZE CDATA #IMPLIED	Specifies the size of the font set that has been applied to text. (The base size of text can be different from its font set size.)	Specifies the size of the font set that has been applied to text. (The base size of text can be different from its font set size.)	Specifies the size of the font set that has been applied to text. (The base size of text can be different from its font set size.)
COLOR CDATA #IMPLIED	Identifies the color for text. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job	Identifies the color for text. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document	Identifies the color for text. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or

Element type	Construct	Modify	Deconstruct
	Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.	defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.
SHADE CDATA #IMPLIED	Specifies the shade of text color, as an integer percentage from 0 to 100.	Specifies the shade of text color, as an integer percentage from 0 to 100.	Identifies the shade of text color, as an integer percentage from 0 to 100.
OPACITY CDATA #IMPLIED	Specifies the opacity of text, specified as an integer percentage from 0 to 100.	Specifies the opacity of text, specified as an integer percentage from 0 to 100.	Identifies the opacity of text, specified as an integer percentage from 0 to 100.
STROKEWIDTH CDATA #IMPLIED	Specifies the width of the text stroke applied to the text, as an integer percentage between 0 and 100.	Specifies the width of the text stroke applied to the text, as an integer percentage between 0 and 100.	Specifies the width of the text stroke applied to the text, as an integer percentage between 0 and 100.
STROKECOLOR CDATA #IMPLIED	Specifies the color of the text stroke applied to the text.	Specifies the color of the text stroke applied to the text.	Specifies the color of the text stroke applied to the text.
STROKESHADE CDATA #IMPLIED	Specifies the shade of the text stroke applied to	Specifies the shade of the text stroke applied to the text, as	Specifies the shade of the text stroke applied to

Element type	Construct	Modify	Deconstruct
	the text, as an integer percentage between 0 and 100.	an integer percentage between 0 and 100.	the text, as an integer percentage between 0 and 100.
STROKEJOIN (MITERJOIN | ROUNDJOIN | BEVELJOIN |	Specifies the type of join of the text stroke applied to the text.	Specifies the type of join of the text stroke applied to the text.	Specifies the type of join of the text stroke applied to the text.
STROKEMITERLIMIT CDATA #IMPLIED	If the type of join is a MITERJOIN, this specifies the miter limit of the text stroke applied to the text.	If the type of join is a MITERJOIN, this specifies the miter limit of the text stroke applied to the text.	If the type of join is a MITERJOIN, this specifies the miter limit of the text stroke applied to the text.
STROKEFILL (true | false)	If this is false, it changes the color of the text to no color. This is only relevant if the width of the text stroke is > 0	If this is false, it changes the color of the text to no color. This is only relevant if the width of the text stroke is > 0	If this is false, it changes the color of the text to no color. This is only relevant if the width of the text stroke is > 0
NONBREAKING (true | false | none) “none"	Specifies if the text will be nonbreaking or not. Used for special character (e.g., for a hyphen:<RICHTEXT NONBREAKING="true"> - </RICHTEXT>)	Specifies if the text will be nonbreaking or not. Used for special characters (e.g., for a thinspace:<RICHTEXT NONBREAKING="true"> </RICHTEXT>)	Specifies if the text will be nonbreaking or not. Used for special characters (e.g., for a thinspace:<RICHTEXT NONBREAKING="true"> </RICHTEXT>)

Element type	Construct	Modify	Deconstruct
UNDERLINE (true | false | none) “none”	Applies the underline type style to text.	Applies the underline type style to text.	Identifies the underline type style applied to text.
WORDUNDERLINE (true | false | none) “none”	Applies the word underline type style to text.	Applies the word underline type style to text.	Identifies the word underline type style applied to text.
SMALLCAPS (true | false | none) “none”	Applies small caps to text.	Applies small caps to text.	Identifies small caps applied to text.
ALLCAPS (true | false | none) “none”	Applies all caps to text.	Applies all caps to text.	Identifies all caps applied to text.
SUPERSCRIPT (true | false | none) “none”	Applies the superscript type style to text.	Applies the superscript type style to text.	Identifies the superscript type style applied to text.
SUBSCRIPT (true | false | none) “none”	Applies the subscript type style to text.	Applies the subscript type style to text.	Identifies the subscript type style applied to text.
SUPERIOR (true | false | none) “none”	Applies the superior type style to text.	Applies the superior type style to text.	Identifies the superior type style applied to text.
OUTLINE (true | false | none) “none”	Applies the outline type style to text.	Applies the outline type style to text.	Identifies the outline type style applied to text.
SHADOW (true | false | none) “none”	Applies the shadow type style to text.	Applies the shadow type style to text.	Identifies the shadow type style applied to text.
STRIKETHRU (true | false | none) “none”	Applies the strikethru type style to text.	Applies the strikethru type style to text.	Identifies the strikethru type style applied to text.

Element type	Construct	Modify	Deconstruct
EMPHASISMARK (NONE | DOT | BLACKCIRCLE | WHITECIRCLE | WHITESQUARE | FISHEYE | COMMA | BLACKSESAME | WHITESESAME | BLACKTRIANGLE) “NONE”	Allows an emphasis mark to be placed on this RICHTEXT.	Allows an emphasis mark to be placed on this RICHTEXT.	Allows an emphasis mark to be placed on this RICHTEXT.
BASELINESHIFT CDATA #IMPLIED	Shifts text up or down without affecting paragraph line spacing. A positive value raises text; a negative value lowers text.	Shifts text up or down without affecting paragraph line spacing. A positive value raises text; a negative value lowers text.	Identifies a shift of text up or down without affecting paragraph line spacing. A positive value raises text; a negative value lowers text.
HORIZONTALSCALE CDATA #IMPLIED	Applies a horizontal scale to text, which makes characters narrower or wider.	Applies a horizontal scale to text, which makes characters narrower or wider.	Identifies a horizontal scale applied to text, which makes characters narrower or wider.
VERTICALSCALE CDATA #IMPLIED	Applies a vertical scale to text, which makes characters taller or shorter. Specified as an integer percentage from 25 to 400.	Applies a vertical scale to text, which makes characters taller or shorter. Specified as an integer percentage from 25 to 400.	Identifies a vertical scale applied to text, which makes characters taller or shorter. Specified as an integer percentage from 25 to 400.

Element type	Construct	Modify	Deconstruct
TRACKAMOUNT CDATA #IMPLIED	Adjusts the amount of space between characters and words.	Adjusts the amount of space between characters and words.	Identifies an amount of adjusted space applied between characters and words.
KERNAMOUNT CDATA #IMPLIED	Adjusts the amount of space between two characters.	Adjusts the amount of space between two characters.	Identifies an amount of adjusted space applied between two characters.
LIGATURES (true | false | none) “none”	Indicates whether standard ligatures should be applied.	Indicates whether standard ligatures should be applied.	Indicates whether standard ligatures are applied.
OT_STANDARD_ LIGATURES (true | false | none) “none”	Applies the OpenType standard ligatures type style to text.	Applies the OpenType standard ligatures type style to text.	Identifies the OpenType standard ligatures type style applied to text.
OT_DISCRETION ARY_LIGATURES (true | false | none) “none”	Applies the OpenType discretionary type style to text.	Applies the OpenType discretionary type style to text.	Identifies the OpenType discretionary type style applied to text.
OT_ORDINALS (true | false | none) “none”	Applies the OpenType ordinals type style to text.	Applies the OpenType ordinals type style to text.	Identifies the OpenType ordinals type style applied to text.
OT_TITLING_ALT ERNATES (true | false | none) “none”	Applies the OpenType titling alternates type style to text.	Applies the OpenType titling alternates type style to text.	Identifies the OpenType titling alternates type style applied to text.
OT_ALL_SMALL_	Applies the	Applies the	Identifies the

Element type	Construct	Modify	Deconstruct
CAPS (true | false | none) “none”	OpenType all small caps type style to text.	OpenType all small caps type style to text.	OpenType all small caps type style applied to text.
OT_FRACTIONS (true | false | none) “none”	Applies the OpenType fractions type style to text.	Applies the OpenType fractions type style to text.	Identifies the OpenType fractions type style applied to text.
OT_SWASHES (true | false | none) “none”	Applies the OpenType swashes type style to text.	Applies the OpenType swashes type style to text.	Identifies the OpenType swashes type style applied o text.
OT_SMALL_CAPS (true | false | none) “none”	Applies the OpenType small caps type style to text.	Applies the OpenType small caps type style to text.	Identifies the OpenType small caps type style applied to text.
OT_CONTEXTUAL_ALTERNATIVES (true | false | none) “none”	Applies the OpenType contextual alternates type style to text.	Applies the OpenType contextual alternates type style to text.	Identifies the OpenType contextual alternates type style applied to text.
OT_TABULAR_FIGURES (true | false | none) “none”	Applies the OpenType tabular figures type style to text.	Applies the OpenType tabular figures type style to text.	Identifies the OpenType tabular figures type style applied to text.
OT_PROPORTIONAL_FIGURES (true | false | none) “none”	Applies the OpenType proportional figures type style to text.	Applies the OpenType proportional figures type style to text.	Identifies the OpenType proportional figures type style applied to text.
OT_LINING_FIGURES (true | false |	Applies the OpenType lining	Applies the OpenType lining	Identifies the OpenType lining

Element type	Construct	Modify	Deconstruct
none) “none”	figures type style to text.	figures type style to text.	figures type style applied to text.
OT_NONE (true | false | none) “none”	Removes OpenType formatting from text.	Removes OpenType formatting from text.	Indicates the OpenType formatting has been removed from text.
OT_SUPERSCRIPT (true | false | none) “none”	Applies the OpenType superscript type style to text.	Applies the OpenType superscript type style to text.	Identifies the OpenType superscript type style applied to text.
OT_SUBSCRIPT (true | false | none) “none”	Applies the OpenType subscript type style to text.	Applies the OpenType subscript type style to text.	Identifies the OpenType subscript type style applied to text.
OT_NUMERATOR (true | false | none) “none”	Applies the OpenType numerator type style to text.	Applies the OpenType numerator type style to text.	Identifies the OpenType numerator type style applied to text.
OT_DENOMINATOR (true | false | none) “none”	Applies the OpenType denominator type style to text.	Applies the OpenType denominator type style to text.	Identifies the OpenType denominator type style applied to text.
OT_OLDSTYLE_FIGURES (true | false | none) “none”	Applies the OpenType old style figures type style to text.	Applies the OpenType old style figures type style to text.	Identifies the OpenType old style figures type style applied to text.
OT_SCIENTIFIC_INFERIOR_FEATURE (true | false | none) “none”	Replaces lining or old style figures with inferior figures	Replaces lining or old style figures with inferior figures (smaller glyphs	Replaces lining or old style figures with inferior figures (smaller

Element type	Construct	Modify	Deconstruct
	(smaller glyphs which sit lower than the standard baseline, primarily for chemical or mathematical notation). May also replace lowercase characters with alphabetic inferiors.	which sit lower than the standard baseline, primarily for chemical or mathematical notation). May also replace lowercase characters with alphabetic inferiors.	glyphs which sit lower than the standard baseline, primarily for chemical or mathematical notation). May also replace lowercase characters with alphabetic inferiors.
OT_ITALICS_FEATURE (true | false | none) “none”	Some fonts (such as Adobe® Pro Japanese fonts) have both Roman and Italic forms of some characters in a single font. This feature replaces the Roman glyphs with the corresponding Italic glyphs.	Some fonts (such as Adobe Pro Japanese fonts) have both Roman and Italic forms of some characters in a single font. This feature replaces the Roman glyphs with the corresponding Italic glyphs.	Some fonts (such as Adobe Pro Japanese fonts) have both Roman and Italic forms of some characters in a single font. This feature replaces the Roman glyphs with the corresponding Italic glyphs.
OT_HVKANA_ALTERNATES (true | false | none) “none”	Apply specially designed horizontal or vertical Kana forms that correspond with the story direction (vertical or horizontal).	Apply specially designed horizontal or vertical Kana forms that correspond with the story direction (vertical or horizontal).	Apply specially designed horizontal or vertical Kana forms that correspond with the story direction (vertical or horizontal).

Element type	Construct	Modify	Deconstruct
OT_RUBINOTATION_FORMS (true | false | none) “none”	Japanese typesetting often uses smaller kana glyphs, generally in superscripted form, to clarify the meaning of kanji which may be unfamiliar to the reader. These are called ruby, from the old typesetting term for four-point-sized type. This feature identifies glyphs in the font which have been designed for this use, substituting them for the default designs.	Japanese typesetting often uses smaller kana glyphs, generally in superscripted form, to clarify the meaning of kanji which may be unfamiliar to the reader. These are called ruby, from the old typesetting term for four-point-sized type. This feature identifies glyphs in the font which have been designed for this use, substituting them for the default designs.	Japanese typesetting often uses smaller kana glyphs, generally in superscripted form, to clarify the meaning of kanji which may be unfamiliar to the reader. These are called ruby, from the old typesetting term for four-point-sized type. This feature identifies glyphs in the font which have been designed for this use, substituting them for the default designs.
OT_LOCALIZED_FORMS (true | false | none) “none”	Replace default forms of glyphs with localized forms.	Replace default forms of glyphs with localized forms.	Replace default forms of glyphs with localized forms.
OT_ALTERNATE_WIDTHS_NONE (true | false | none) “none”	Apply alternate widths for heights based on story direction (vertical or horizontal).	Apply alternate widths for heights based on story direction (vertical or horizontal).	Apply alternate widths for heights based on story direction (vertical or horizontal).
OT_FULL_WIDTHS (true | false |	Replace glyphs set on other em	Replace glyphs set on other em widths	Replace glyphs set on other em

Element type	Construct	Modify	Deconstruct
none) “none”	widths with glyphs set on full-em widths.	with glyphs set on full-em widths.	widths with glyphs set on full-em widths.
OT_HALF_WIDTHS (true | false | none) “none”	Replace glyphs set on other em widths with half-em width glyphs.	Replace glyphs set on other em widths with half-em width glyphs.	Replace glyphs set on other em widths with half-em width glyphs.
OT_THIRD_WIDTHS (true | false | none) “none”	Replace glyphs set on other em widths with glyphs set on third-em widths.	Replace glyphs set on other em widths with glyphs set on third-em widths.	Replace glyphs set on other em widths with glyphs set on third-em widths.
OT_QUARTER_WIDTHS (true | false | none) “none”	Replace glyphs set on other em widths with glyphs set on quarter-em widths.	Replace glyphs set on other em widths with glyphs set on quarter-em widths.	Replace glyphs set on other em widths with glyphs set on quarter-em widths.
OT_PROPORTIONAL_WIDTHS (true | false | none) “none”	Fit glyphs to individual, proportional widths.	Fit glyphs to individual, proportional widths.	Fit glyphs to individual, proportional widths.
OT_ALTVERTMETRICS (true | false | none) “none”	Center glyphs inside a full-em height.	Center glyphs inside a full-em height.	Center glyphs inside a full-em height.
OT_PROPORTIONAL_ALTVERTMETRICS (true | false | none) “none”	Fit glyphs to individual, proportional heights.	Fit glyphs to individual, proportional heights.	Fit glyphs to individual, proportional heights.
OT_ALTERNATE_HALF_METRICS (true | false | none) “none”	Fit full-em height glyphs to half-em heights.	Fit full-em height glyphs to half-em heights.	Fit full-em height glyphs to half-em heights.
OT_ALTERNATE_FORMS_NONE	Alternate glyph forms, such as	Alternate glyph forms, such as	Alternate glyph forms, such as

Element type	Construct	Modify	Deconstruct
(true | false | none) “none"OT_JIS78FORMS (true | false | none) “none"OT_JIS83FORMS (true | false | none) “none"OT_JIS90FORMS (true | false | none) “none"OT_JIS04FORMS (true | false | none) “none"OT_SIMPLIFIED_FORMS (true | false | none) “none"OT_TRADITIONAL_FORMS (true | false | none) “none”	JIS2004, JIS78, JIS90, Simplified, and Traditional. These glyph forms are specially designed for some Japanese OpenType fonts.	JIS2004, JIS78, JIS90, Simplified, and Traditional. These glyph forms are specially designed for some Japanese OpenType fonts.	JIS2004, JIS78, JIS90, Simplified, and Traditional. These glyph forms are specially designed for some Japanese OpenType fonts.
LANGUAGE (SwissGerman | SwissGermanReformed | BrazilianPortuguese | Bulgarian | Croatian | Czech | Dutch | Danish | Finnish | French | German | ReformedGerman | Hungarian | Greek | Italian | BokmalNorwegian | Portuguese | Polish | Slovak |	Specifies the dictionary preference used for hyphenation.	Specifies the dictionary preference used for hyphenation.	Identifies the dictionary preference used for hyphenation.

Element type	Construct	Modify	Deconstruct
Russian | Romanian | Swedish | Turkish | Spanish | USEnglish | Catalan | Estonian | Lithuanian | Latvian | Icelandic | Slovenian | InternationalEnglish | SimplifiedChinese | TraditionalChinese | Japanese | Korean | Ukrainian | NynorskNorwegian | None | none) “none”			
SENDING CDATA #IMPLIED	Sending is a character spacing attribute used particularly in East Asian typography, similar to kerning, but applicable as a fixed value over a range of text.	Sending is a character spacing attribute used particularly in East Asian typography, similar to kerning, but applicable as a fixed value over a range of text.	Sending is a character spacing attribute used particularly in East Asian typography, similar to kerning, but applicable as a fixed value over a range of text.
APPLYSENDINGTONONCJK (true | false | none) “none”	Describes whether sending should be applied to both Roman and	Describes whether sending should be applied to both Roman and Chinese/Japanese/K	Describes whether sending should be applied to both Roman and

Element type	Construct	Modify	Deconstruct
	Chinese/Japanese/Korean glyphs (true) or just to Chinese, Japanese, and Korean Glyphs (false).	orean glyphs (true) or just to Chinese, Japanese, and Korean Glyphs (false).	Chinese/Japanese/Korean glyphs (true) or just to Chinese, Japanese, and Korean Glyphs (false).
UEGGLYPHID CDATA #IMPLIED	Unencoded Glyphs (UEG)Some glyphs, especially in legacy Korean documents, are not covered by the Unicode specification. These are referred to as UEG or Unencoded Glyphs. This attribute represents the font glyph ID for such characters that cannot be represented. Note that this is an empty element, as the glyph cannot be represented as text.	Some glyphs, especially in legacy Korean documents, are not covered by the Unicode specification. These are referred to as UEG or Unencoded Glyphs. This attribute represents the font glyph ID for such characters that cannot be represented. Note that this is an empty element, as the glyph cannot be represented as text.	Some glyphs, especially in legacy Korean documents, are not covered by the Unicode specification. These are referred to as UEG or Unencoded Glyphs. This attribute represents the font glyph ID for such characters that cannot be represented. Note that this is an empty element, as the glyph cannot be represented as text.
OTVARIANT CDATA #IMPLIED	Specifies which variant to use from among the multiple match	Specifies which variant to use from among the multiple match found (if any).	Specifies which variant to use from among the multiple match

Element type	Construct	Modify	Deconstruct
	found (if any).		found (if any).
OTFEATURE CDATA #IMPLIED	Contains the value of the OpenType feature applied on text like AlternateFractions (afrc), AlternateAnnotations, etc.	Contains the value of the OpenType feature applied on text like AlternateFractions (afrc), AlternateAnnotations, etc.	Contains the value of the OpenType feature applied on text like AlternateFractions (afrc), AlternateAnnotations, etc.
SCRIPT (Hira | Hani | Hrkt | Hang | Yiii | Kana |Bopo | none) “none”	Represents the script system used by this <RICHTEXT> element's content.	Represents the script system used by this <RICHTEXT> element's content.	Represents the script system used by this <RICHTEXT> element's content.
HALFWIDTHUPRIGHT (true | false | none) “none”	Specifies whether this character should be presented upright in a vertical story. This is specifically applicable to Roman characters within a vertical story.	Specifies whether this character should be presented upright in a vertical story. This is specifically applicable to Roman characters within a vertical story.	Specifies whether this character should be presented upright in a vertical story. This is specifically applicable to Roman characters within a vertical story.
FAUXSTYLE (BOLD | ITALIC | BOLDITALIC | NONE) #IMPLIED	Not applicable.	Not applicable.	Indicates whether the text contains a faux type style (such as a bold face that is constructed by software, as opposed to a bold

Element type	Construct	Modify	Deconstruct
			font).
PAGENUMBERCHAR (CURRENTPAGE | NEXTPAGE | PREVIOUSPAGE) #IMPLIED	Represents an automatic page number character. If a RICHTEXT element with this attribute occurs in a section, section-specific numbering and formatting is applied to the page number. For more information, see "Working with sections."	Represents an automatic page number character. If a RICHTEXT element with this attribute occurs in a section, section-specific numbering and formatting is applied to the page number. For more information, see "Working with sections."	Represents an automatic page number character.
HYPERLINKREF CDATA #IMPLIED	Not applicable.	Not applicable.	Specifies that this <RICHTEXT> element is a hyperlink by referring to a HYPERLINK.
HLTYPE (WWWURL | PAGE | ANCHOR) #IMPLIED	Not applicable.	Not applicable.	Specifies the type of hyperlink this <RICHTEXT> element hyperlinks to. Options include WWWURL (a URL on the Web), PAGE (the top of a page in the same layout), and ANCHOR (an

Element type	Construct	Modify	Deconstruct
			anchor).
HLANCHORREF CDATA #IMPLIED	Not applicable.	Not applicable.	If this <RICHTEXT> element is a hyperlink of the HLTYPE ANCHOR, this attribute identifies the anchor by name.
BACKGROUNDCOLOR CDATA #IMPLIED	Specifies a background color to be inserted behind the text. This color displays only in rendered output, and is not saved with the project file.	Specifies a background color to be inserted behind the text. This color displays only in rendered output, and is not saved with the project file.	Not applicable.

RIGHT (Modifier schema)

Element type	Construct	Modify	Deconstruct
RIGHT (#PCDATA)	The distance between the box or lines right edge and the right edge of the page, in points.	The distance between the box or lines right edge and the right edge of the page, in points.	The distance between the box or lines right edge and the right edge of the page, in points.

RIGHTCONTROLPOINT (Modifier schema)

Element type	Construct	Modify	Deconstruct
RIGHTCONTROLPOINT (empty)	Each point on a curve is described by	Each point on a curve is described by three geometric positions:	Each point on a curve is described by three

Element type	Construct	Modify	Deconstruct
	three geometric positions: the x,y coordinate of the vertex point (this coordinate is relative to the bounding geometry of the shape, not the page), and the left and right control handles–as you would see onscreen in the QuarkXPress user environment. For more information on drawing and manipulating bezier curves, please see A Guide to QuarkXPress.	the x,y coordinate of the vertex point (this coordinate is relative to the bounding geometry of the shape, not the page), and the left and right control handles–as you would see onscreen in the QuarkXPress user environment. For more information on drawing and manipulating bezier curves, please see A Guide to QuarkXPress.	geometric positions: the x,y coordinate of the vertex point (this coordinate is relative to the bounding geometry of the shape, not the page), and the left and right control handles–as you would see onscreen in the QuarkXPress user environment. For more information on drawing and manipulating bezier curves, please see A Guide to QuarkXPress.
Attributes			
X CDATA #IMPLIED	X coordinate of RIGHTCONTROLPOINT.	X coordinate of RIGHTCONTROLPOINT.	X coordinate of RIGHTCONTROLPOINT.
Y CDATA #IMPLIED	Y coordinate of RIGHTCONTROLPOINT.	Y coordinate of RIGHTCONTROLPOINT.	Y coordinate of RIGHTCONTROLPOINT.

RIGHTGRID (Modifier schema)

Element type	Construct	Modify	Deconstruct
RIGHTGRID (empty)	Describes a grid line on the right edge of a cell in an <INLINETABLE>.	Describes a grid line on the right edge of a cell in an <INLINETABLE>.	Not applicable.
Attributes			
TYPE (TOP | LEFT | BOTTOM | RIGHT) #IMPLIED	Specifies the location of the grid line.	Specifies the location of the grid line.	Not applicable.
STYLE CDATA #IMPLIED	Identifies the <TABLESTYLE> that styles this grid line. If you specify this value, you do not have to specify the remaining attributes. If you specify the remaining attributes, those attribute values override the corresponding <TABLESTYLE> values.	Identifies the <TABLESTYLE> that styles this grid line. If you specify this value, you do not have to specify the remaining attributes. If you specify the remaining attributes, those attribute values override the corresponding <TABLESTYLE> values.	Not applicable.
WIDTH CDATA #IMPLIED	Specifies the width of the grid line in points.	Specifies the width of the grid line in points.	Not applicable.
COLOR CDATA #IMPLIED	Specifies the color of the grid line.	Specifies the color of the grid line.	Not applicable.
SHADE CDATA #IMPLIED	Specifies the shade of the grid	Specifies the shade of the grid line.	Not applicable.

Element type	Construct	Modify	Deconstruct
	line.		
OPACITY CDATA #IMPLIED	Specifies the opacity of the grid line.	Specifies the opacity of the grid line.	Not applicable.
GAPCOLOR CDATA #IMPLIED	Specifies the color of the gap (if any) between the lines that make up the grid line.	Specifies the color of the gap (if any) between the lines that make up the grid line.	Not applicable.
GAPSHADE CDATA #IMPLIED	Specifies the shade of the gap (if any) between the lines that make up the grid line.	Specifies the shade of the gap (if any) between the lines that make up the grid line.	Not applicable.
GAPOPACITY CDATA #IMPLIED	Specifies the opacity of the gap (if any) between the lines that make up the grid line.	Specifies the opacity of the gap (if any) between the lines that make up the grid line.	Not applicable.

ROW (Modifier schema)

Element type	Construct	Modify	Deconstruct
ROW ((CELL | GRIDLINE)*)	Describes a row in a table.	Describes a row in a table.	Describes a row in a table.
Attributes			
ROWCOUNT CDATA #REQUIRED	Specifies the index position of a row from top to bottom. For example, ROWCOUNT = 1 indicates the	Specifies the index position of a row from top to bottom. For example, ROWCOUNT = 1 indicates the first column from the top,	Specifies the index position of a row from top to bottom. For example, ROWCOUNT = 1 indicates the first

Element type	Construct	Modify	Deconstruct
	first row from the top, and ROWCOUNT = 2 indicates the second row from the top.	and ROWCOUNT = 2 indicates the second row from the top.	column from the top, and ROWCOUNT = 2 indicates the second column from the top.
ROWHEIGHT CDATA #IMPLIED	Specifies the height of a row. Note: If this attribute is empty, the row is resized to fit its contents, unless RICHTEXT@ MAINTAINGEOMETRY is set to true, in which case any row that does not have a ROWHEIGHT attribute will be sized equally using the amount of space remaining after all the specified ROWHEIGHT attributes have been subtracted from the total height of the box.	Specifies the height of a row. Note: If this attribute is empty, the row is resized to fit its contents, unless RICHTEXT@ MAINTAINGEOMETRY is set to true, in which case any row that does not have a ROWHEIGHT attribute will be sized equally using the amount of space remaining after all the specified ROWHEIGHT attributes have been subtracted from the total height of the box.	Specifies the height a row.
COLOR CDATA #IMPLIED	Identifies the color of a row. Overrides the TABLE@COLOR attribute. Note:	Identifies the color of a row. Overrides the TABLE@COLOR attribute. Note: Only the name of a color	Identifies the color of a row. Overrides the TABLE@COLOR attribute. Note:

Element type	Construct	Modify	Deconstruct
	Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.	Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.
SHADE CDATA #IMPLIED	Specifies the shade of the color applied to a row, as an integer percentage from 0 to 100.	Specifies the shade of the color applied to a row, as an integer percentage from 0 to 100.	Specifies the shade of the color applied to a row, as an integer percentage from 0 to 100.
OPACITY CDATA #IMPLIED	Specifies the opacity of the color applied to a row, specified as an integer percentage from 0 to 100.	Specifies the opacity of the color applied to a row, specified as an integer percentage from 0 to 100.	Specifies the opacity of the color applied to a row, specified as an integer percentage from 0 to 100.
MERGEROWSPAN CDATA #IMPLIED	Attribute used for merging cells and rows.	Attribute used for merging cells and rows.	If a table includes merged cells, then the MERGECOLSPAN value is shown in the xml output.

Element type	Construct	Modify	Deconstruct
SPLIT (true | false) #IMPLIED	Not applicable.	Attribute used for splitting rows and columns.	Not applicable.
AUTOFIT (true | false | none) “none”	Specifies whether the rows or columns will adjust size to fit the content.	Specifies whether the rows or columns will adjust size to fit the content.	Specifies whether the rows or columns will adjust size to fit the content.
AUTOFITMAXLIMIT CDATA #IMPLIED	Max limit for autofit.	Max limit for autofit.	Max limit for autofit.

RUBI (Modifier schema)

Element type	Construct	Modify	Deconstruct
RUBI (RUBITEXT, (RICHTEXT | ANCHOREDBOXREF | HIDDEN | CALLOUTANCHOR | INLINEBOX | MATHEQUATION)+)	Specifies a region of base text and the rubi text to include with that text. Note the second and subsequent children of the RUBI element (RICHTEXT | ANCHOREDBOX | HIDDEN | CALLOUTANCHOR | INLINEBOX | MATHEQUATION)+ declare the base text to which the rubi text is to be applied.	Specifies a region of base text and the rubi text to include with that text. Note the second and subsequent children of the RUBI element (RICHTEXT | ANCHOREDBOX | HIDDEN | CALLOUTANCHOR | INLINEBOX | MATHEQUATION)+ declare the base text to which the rubi text is to be applied.	Specifies a region of base text and the rubi text to include with that text. Note the second and subsequent children of the RUBI element (RICHTEXT | ANCHOREDBOX | HIDDEN | CALLOUTANCHOR | INLINEBOX | MATHEQUATION) + declare the base text to which the rubi text is to be applied.

RUBITEXT (Modifier schema)

Element type	Construct	Modify	Deconstruct
RUBITEXT (RICHTEXT)	Specifies the rubi text to be applied to the specified base text.The RUBITEXT element is a container for a RICHTEXT element. All the usual character formatting attributes can be applied to the rubi text through this RICHTEXT element.	Specifies the rubi text to be applied to the specified base text. The RUBITEXT element is a container for a RICHTEXT element. All the usual character formatting attributes can be applied to the rubi text through this RICHTEXT element.	Specifies the rubi text to be applied to the specified base text. The RUBITEXT element is a container for a RICHTEXT element. All the usual character formatting attributes can be applied to the rubi text through this RICHTEXT element.
Attributes			
ALIGNMENT (LEFT | TOP | CENTERED | RIGHT | BOTTOM | JUSTIFIED | FORCED | ONETOONE | EQUALSPACE | ONERUBISPACE) “CENTERED”	Controls how non-overhanging rubi text aligns with the base text. For more information, see “Rubi alignment options" in the QuarkXPress documentation.	Controls how non-overhanging rubi text aligns with the base text. For more information, see “Rubi alignment options" in the QuarkXPress documentation.	Controls how non-overhanging rubi text aligns with the base text. For more information, see “Rubi alignment options" in the QuarkXPress documentation.
OVERHANGALIGNMENT (none | LEFT | TOP | CENTERED | RIGHT | BOTTOM | JUSTIFIED | FORCED |	Defines how far the rubi text can overhang base text that is unrelated to the rubi text. For more	Defines how far the rubi text can overhang base text that is unrelated to the rubi text. For more information, see “Rubi overhang	Defines how far the rubi text can overhang base text that is unrelated to the rubi text. For more information,

Element type	Construct	Modify	Deconstruct
ONETOONE | EQUALSPACE) “none”	information, see “Rubi overhang options."	options."	see “Rubi overhang options."
PLACEMENT (ABOVE | BELOW | RIGHT | LEFT) “ABOVE”	This attribute specifies whether rubi text displays above or below the base text (in a horizontal story) or to the left of or right of the base text (in a vertical story).	This attribute specifies whether rubi text displays above or below the base text (in a horizontal story) or to the left of or right of the base text (in a vertical story).	This attribute specifies whether rubi text displays above or below the base text (in a horizontal story) or to the left of or right of the base text (in a vertical story).
RELATIVESIZE CDATA “50”	Defines the size of the rubi text compared to the base text.	Defines the size of the rubi text compared to the base text.	Defines the size of the rubi text compared to the base text.
OFFSET CDATA “0”	Use this attribute to control how far the rubi text is offset from the base text.	Use this attribute to control how far the rubi text is offset from the base text.	Use this attribute to control how far the rubi text is offset from the base text.
OVERHANG (none | UNRESTRICTED | HALFRUBI | FULLRUBI | HALFBASE | FULLBASE) “HALFRUBI”	Defines how far the rubi text can overhang base text that is unrelated to the rubi text. For more information, see “Rubi overhang options."	Defines how far the rubi text can overhang base text that is unrelated to the rubi text. For more information, see “Rubi overhang options."	Defines how far the rubi text can overhang base text that is unrelated to the rubi text. For more information, see “Rubi overhang options."
AUTOALIGNATLINEEDGES (true | false) “true”	Automatically aligns rubi text with the border	Automatically aligns rubi text with the border of a text box	Automatically aligns rubi text with the border of

Element type	Construct	Modify	Deconstruct
	of a text box when the rubi text overhangs the base text and touches the edge of the text box.	when the rubi text overhangs the base text and touches the edge of the text box.	a text box when the rubi text overhangs the base text and touches the edge of the text box.
ANNONATIONS (true | false) "true"	Applicable for OT fonts applied to rubi. If the font supports annotations, then that is applied on the rubi text.	Applicable for OT fonts applied to rubi. If the font supports annotations, then that is applied on the rubi text.	Applicable for OT fonts applied to rubi. If the font supports annotations, then that is applied on the rubi text.

RULE (Modifier schema)

Element type	Construct	Modify	Deconstruct
RULE (empty)	Describes a rule above or below a paragraph.	Describes a rule above or below a paragraph.	Describes a rule above or below a paragraph.
Attributes			
ENABLED (true | false | none) "none"	Specifies whether to add a rule to a paragraph or not.	Specifies whether to add a rule to a paragraph or not.	Specifies whether a rule is applied to a paragraph or not.
POSITION (ABOVE | BELOW) "BELOW"	Specifies whether a rule should be above or below a paragraph.	Specifies whether a rule should be above or below a paragraph.	Specifies whether a rule is above or below a paragraph.
LENGTH (TEXT | COLUMN | INDENTS)	Specifies the length of a rule.TEXT = Rule	Specifies the length of a rule.TEXT = Rule is the same length as	Specifies the length of a rule.TEXT = Rule

Element type	Construct	Modify	Deconstruct
“INDENTS”	is the same length as the first line of text in the paragraph (for rule above) or the last line of text in the paragraph (for rule below).COLUMN = Rule extends to edges of parent box or column.INDENTS = Rule extends from the paragraph's left indent to its right indent.	the first line of text in the paragraph (for rule above) or the last line of text in the paragraph (for rule below).COLUMN = Rule extends to edges of parent box or column.INDENTS = Rule extends from the paragraph's left indent to its right indent.	is the same length as the first line of text in the paragraph (for rule above) or the last line of text in the paragraph (for rule below).COLUMN = Rule extends to edges of parent box or column.INDENTS = Rule extends from the paragraph's left indent to its right indent.
LEFT CDATA #IMPLIED	Specifies a distance to indent a rule farther from the left. A positive number moves the end-point to the right; a negative number moves the end-point to the left.	Specifies a distance to indent a rule farther from the left. A positive number moves the end-point to the right; a negative number moves the end-point to the left.	Specifies a distance a rule is indented farther from the left. A positive number moves the end-point to the right; a negative number moves the end-point to the left.
RIGHT CDATA #IMPLIED	Specifies a distance to indent a rule farther from the right. A positive number moves the end-point to the left; a	Specifies a distance to indent a rule farther from the right. A positive number moves the end-point to the left; a negative number moves the end-point	Specifies a distance a rule is indented farther from the right. A positive number moves the end-point to the left; a negative number

Element type	Construct	Modify	Deconstruct
	negative number moves the end-point to the right.	to the right.	moves the end-point to the right.
OFFSET CDATA #IMPLIED	Specifies the amount of space between a rule and the paragraph to which it is attached.	Specifies the amount of space between a rule and the paragraph to which it is attached.	Specifies the amount of space between a rule and the paragraph to which it is attached.
WIDTH CDATA #IMPLIED	Specifies the thickness of a rule.	Specifies the thickness of a rule.	Specifies the thickness of a rule.
COLOR CDATA #IMPLIED	Identifies the color for a rule. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	Identifies the color for a rule. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.	Identifies the color for a rule. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.
SHADE CDATA #IMPLIED	Specifies the shade of a rules color, as an	Specifies the shade of a rules color, as an integer percentage	Specifies the shade of a rules color, as an

Element type	Construct	Modify	Deconstruct
	integer percentage from 0 to 100.	from 0 to 100.	integer percentage from 0 to 100.
OPACITY CDATA #IMPLIED	Specifies the opacity of a rules color, specified as an integer percentage from 0 to 100.	Specifies the opacity of a rules color, specified as an integer percentage from 0 to 100.	Specifies the opacity of a rules color, specified as an integer percentage from 0 to 100.
STYLE CDATA #IMPLIED	Identifies a Dashes & Stripes style (LINESTYLE) for a rule. Note: Only the name of a Dashes & Stripes style is included in this attribute. The definition of the Dashes & Stripes style is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	Identifies a Dashes & Stripes style (LINESTYLE) for a rule. Note: Only the name of a Dashes & Stripes style is included in this attribute. The definition of the Dashes & Stripes style is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	Identifies a Dashes & Stripes style (LINESTYLE) for a rule. Note: Only the name of a Dashes & Stripes style is included in this attribute. The definition of the Dashes & Stripes style is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.

RUNAROUND (Modifier schema)

Element type	Construct	Modify	Deconstruct
RUNAROUND (empty)	Describes a runaround	Describes a runaround applied to	Describes a runaround applied

Element type	Construct	Modify	Deconstruct
	applied to a box or line.	a box or line.	to a box or line.
Attributes			
TYPE (NONE | ITEM | EMBEDDEDPATH | ALPHACHANNEL | NONWHITEAREAS | PICTUREBOUNDS | SAMEASCLIPPING | AUTOIMAGE | MANUAL) "NONE"	Specifies the type of runaround applied to a box or line: NONE = Text runs behind the box or line. ITEM = Text runs around the edges of the box or line. EMBEDDEDPATH = Text runs around a path embedded in the picture file. ALPHACHANNEL = Text runs around an alpha channel embedded in the picture file. NONWHITEAREAS = Text runs around a path based on the dark and light areas of the picture file. See the THRESHOLD attribute.	Specifies the type of runaround applied to a box or line: NONE = Text runs behind the box or line. ITEM = Text runs around the edges of the box or line. EMBEDDEDPATH = Text runs around a path embedded in the picture file. ALPHACHANNEL = Text runs around an alpha channel embedded in the picture file. NONWHITEAREAS = Text runs around a path based on the dark and light areas of the picture file. See the THRESHOLD attribute. PICTUREBOUNDS = Text runs around the rectangular canvas area of the picture, regardless of the size and shape of the picture box.	Specifies the type of runaround applied to a box or line: NONE = Text runs behind the box or line. ITEM = Text runs around the edges of the box or line. EMBEDDEDPATH = Text runs around a path embedded in the picture file. ALPHACHANNEL = Text runs around an alpha channel embedded in the picture file. NONWHITEAREAS = Text runs around a path based on the dark and light areas of the picture file. See the THRESHOLD attribute. PICTUREBOUNDS = Text runs

Element type	Construct	Modify	Deconstruct
	PICTUREBOUNDS = Text runs around the rectangular canvas area of the picture, regardless of the size and shape of the picture box. SAMEASCLIPPING = Text runs around the pictures clipping path, if any. AUTOIMAGE = Text runs around a clipping path created based on the dark and light areas in the picture file. See the THRESHOLD attribute.	SAMEASCLIPPING = Text runs around the pictures clipping path, if any. AUTOIMAGE = Text runs around a clipping path created based on the dark and light areas in the picture file. See the THRESHOLD attribute.	around the rectangular canvas area of the picture, regardless of the size and shape of the picture box. SAMEASCLIPPING = Text runs around the pictures clipping path, if any. AUTOIMAGE = Text runs around a clipping path created based on the dark and light areas in the picture file. See the THRESHOLD attribute.
TOP CDATA #IMPLIED	Valid when RUNAROUND@TYPE = ITEM or PICTUREBOUNDS. Moves the top edge of the runaround by the specified number of points (positive=up, negative=down).	Valid when RUNAROUND@TYPE = ITEM or PICTUREBOUNDS. Moves the top edge of the runaround by the specified number of points (positive=up, negative=down).	Valid when RUNAROUND@TYPE = ITEM or PICTUREBOUNDS. Moves the top edge of the runaround by the specified number of points (positive=up, negative=down).
RIGHT CDATA	Valid when	Valid when	Valid when

Element type	Construct	Modify	Deconstruct
#IMPLIED	RUNAROUND@TYPE = ITEM or PICTUREBOUNDS. Moves the right edge of the runaround by the specified number of points (positive=right, negative=left).	RUNAROUND@TYPE = ITEM or PICTUREBOUNDS. Moves the right edge of the runaround by the specified number of points (positive=right, negative=left).	RUNAROUND@TYPE = ITEM or PICTUREBOUNDS. Moves the right edge of the runaround by the specified number of points (positive=right, negative=left).
LEFT CDATA #IMPLIED	Valid when RUNAROUND@TYPE = ITEM or PICTUREBOUNDS. Moves the left edge of the runaround by the specified number of points (positive=left, negative=right).	Valid when RUNAROUND@TYPE = ITEM or PICTUREBOUNDS. Moves the left edge of the runaround by the specified number of points (positive=left, negative=right).	Valid when RUNAROUND@TYPE = ITEM or PICTUREBOUNDS. Moves the left edge of the runaround by the specified number of points (positive=left, negative=right).
BOTTOM CDATA #IMPLIED	Valid when RUNAROUND@TYPE = ITEM or PICTUREBOUNDS. Moves the bottom edge of the runaround by the specified number of points (positive=down, negative=up).	Valid when RUNAROUND@TYPE = ITEM or PICTUREBOUNDS. Moves the bottom edge of the runaround by the specified number of points (positive=down, negative=up).	Valid when RUNAROUND@TYPE = ITEM or PICTUREBOUNDS. Moves the bottom edge of the runaround by the specified number of points (positive=down, negative=up).
PATHNAME CDATA #IMPLIED	Identifies a clipping path	Identifies a clipping path embedded in a	Identifies a clipping path

Element type	Construct	Modify	Deconstruct
	embedded in a picture for use as the runaround path.	picture for use as the runaround path.	embedded in a picture for use as the runaround path.
OUTSET CDATA #IMPLIED	Valid when RUNAROUND@TYPE = AUTOIMAGE, EMBEDDEDPATH, ALPHACHANNEL, NONWHITEAREAS, or SAMEASCLIPPING. Specifies a single outset or inset integer value in points to be used on all sides.	Valid when RUNAROUND@TYPE = AUTOIMAGE, EMBEDDEDPATH, ALPHACHANNEL, NONWHITEAREAS, or SAMEASCLIPPING. Specifies a single outset or inset integer value in points to be used on all sides.	Valid when RUNAROUND@TYPE = AUTOIMAGE, EMBEDDEDPATH, ALPHACHANNEL, NONWHITEAREAS, or SAMEASCLIPPING. Specifies a single outset or inset integer value in points to be used on all sides.
NOISE CDATA #IMPLIED	Valid when RUNAROUND@TYPE = AUTOIMAGE, ALPHACHANNEL, or NONWHITEAREAS. Specifies that areas smaller than this number of points should be ignored when creating a runaround path.	Valid when RUNAROUND@TYPE = AUTOIMAGE, ALPHACHANNEL, or NONWHITEAREAS. Specifies that areas smaller than this number of points should be ignored when creating a runaround path.	Valid when RUNAROUND@TYPE = AUTOIMAGE, ALPHACHANNEL, or NONWHITEAREAS. Specifies that areas smaller than this number of points should be ignored when creating a runaround path.
THRESHOLD CDATA #IMPLIED	Valid when RUNAROUND@T	Valid when RUNAROUND@TYPE	Valid when RUNAROUND@TY

Element type	Construct	Modify	Deconstruct
	YPE = AUTOIMAGE, ALPHACHANNEL, or NONWHITEAREAS. Specifies the maximum integer percentage of darkness that should be considered white when creating a runaround path.	= AUTOIMAGE, ALPHACHANNEL, or NONWHITEAREAS. Specifies the maximum integer percentage of darkness that should be considered white when creating a runaround path.	PE = AUTOIMAGE, ALPHACHANNEL, or NONWHITEAREAS. Specifies the maximum integer percentage of darkness that should be considered white when creating a runaround path.
SMOOTHNESS CDATA #IMPLIED	Valid when RUNAROUND@TYPE = AUTOIMAGE, ALPHACHANNEL, or NONWHITEAREAS. Specifies the smoothness, in points, of an automatically created runaround path.	Valid when RUNAROUND@TYPE = AUTOIMAGE, ALPHACHANNEL, or NONWHITEAREAS. Specifies the smoothness, in points, of an automatically created runaround path.	Valid when RUNAROUND@TYPE = AUTOIMAGE, ALPHACHANNEL, or NONWHITEAREAS. Specifies the smoothness, in points, of an automatically created runaround path.
OUTSIDEONLY (true | false | none) “none”	Valid when RUNAROUND@TYPE = AUTOIMAGE, EMBEDDEDPATH, ALPHACHANNEL, or NONWHITEAREAS. Indicates that only the outer	Valid when RUNAROUND@TYPE = AUTOIMAGE, EMBEDDEDPATH, ALPHACHANNEL, or NONWHITEAREAS. Indicates that only the outer edges of the runaround path should be used.	Valid when RUNAROUND@TYPE = AUTOIMAGE, EMBEDDEDPATH, ALPHACHANNEL, or NONWHITEAREAS. Indicates that only the outer edges of the runaround path is

Element type	Construct	Modify	Deconstruct
	edges of the runaround path should be used.		used.
RESTRICTTOBOX (true | false | none) “none”	Valid when RUNAROUND@TYPE = AUTOIMAGE, EMBEDDEDPATH, ALPHACHANNEL, or NONWHITEAREAS. Indicates whether the runaround path is restricted to the inside of the box.	Valid when RUNAROUND@TYPE = AUTOIMAGE, EMBEDDEDPATH, ALPHACHANNEL, or NONWHITEAREAS. Indicates whether the runaround path is restricted to the inside of the box.	Valid when RUNAROUND@TYPE = AUTOIMAGE, EMBEDDEDPATH, ALPHACHANNEL, or NONWHITEAREAS. Indicates whether the runaround path is restricted to the inside of the box.
INVERT (true | false | none) “none”	Valid when RUNAROUND@TYPE = EMBEDDEDPATH, ALPHACHANNEL, or NONWHITEAREAS. Reverses the shape of the runaround path.	Valid when RUNAROUND@TYPE = EMBEDDEDPATH, ALPHACHANNEL, or NONWHITEAREAS. Reverses the shape of the runaround path.	Valid when RUNAROUND@TYPE = EMBEDDEDPATH, ALPHACHANNEL, or NONWHITEAREAS. Reverses the shape of the runaround path.
EDITED (true | false | none) “none”	Not applicable.	Not applicable.	Indicates whether the runaround path has been manually edited in QuarkXPress.

SAVEAS (Modifier schema)

Element type	Construct	Modify	Deconstruct
SAVEAS (empty)	Lets you save a constructed QuarkXPress project to a specific location on the server computer. Roughly equivalent to choosing File > Save As in QuarkXPress.	Lets you save a modified QuarkXPress project to a specific location on the server computer. Roughly equivalent to choosing File > Save As in QuarkXPress.	Not applicable.
Attributes			
NEWNAME CDATA #IMPLIED	Specifies a name for the project being saved.	Specifies a name for the project being saved. Can be a relative path to the document pool.	Not applicable.
PATH CDATA #IMPLIED	The absolute path on the server computer for saving the project.	The absolute path on the server computer for saving the project.	Not applicable.
SAVETOPOOL (true | false) “true”	Specifies whether the project should be saved to the document pool, in addition to saving it in the location specified in the PATH attribute.	Specifies whether the project should be saved to the document pool, in addition to saving it in the location specified in the PATH attribute.	Not applicable.
REPLACE (true | false) “true”	Indicates whether the	Indicates whether the saved project	Not applicable.

Element type	Construct	Modify	Deconstruct
	saved project should replace any existing file with the same name in the specified location.An index number gets appended to the file name if this value is set to false and a file with the supplied name exists at the specification location.For example, if NEWNAME = file.qxp and the REPLACE value is set to false, the file is saved as file1.qxp when a file with the same name exists at the specified location.	should replace any existing file with the same name in the specified location.An index number gets appended to the file name if this value is set to false and a file with the supplied name exists at the specification location.For example, if NEWNAME = file.qxp and the REPLACE value is set to false, the file is saved as file1.qxp when a file with the same name exists at the specified location.	

SCALETO (Modifier schema)

Element type	Construct	Modify	Deconstruct
SCALETO (empty)	Lets you specify the maximum or minimum size of a box for a fit-	Lets you specify the maximum or minimum size of a box for a fit-box-to-	Not applicable.

Element type	Construct	Modify	Deconstruct
	box-to-content operation.	content operation.	
Attributes			
X CDATA #REQUIRED	The largest or smallest allowable width for the resized box, as an integer percentage.	The largest or smallest allowable width for the resized box, as an integer percentage.	Not applicable.
Y CDATA #REQUIRED	The largest or smallest allowable height for the resized box, as an integer percentage.	The largest or smallest allowable height for the resized box, as an integer percentage.	Not applicable.

SECTION (Modifier schema)

Element type	Construct	Modify	Deconstruct
SECTION (empty)	Describes a section break in a layout.	Describes a section break in a layout.	Specifies a section in a QuarkXPress layout.
Attributes			
PREFIX CDATA #IMPLIED	The prefix to be added before each automatic page number inserted in this section.	The prefix to be added before each automatic page number inserted in this section.	The prefix to be added before each automatic page number inserted in this section.
OPERATION (CREATE | DELETE) #IMPLIED	Not applicable.	Specifies whether to create or delete the indicated section.	Not applicable.

Element type	Construct	Modify	Deconstruct
FORMAT (NUMERIC | ROMAN | SMALLROMAN | ALPHA | SMALLALPHA | ASIANNUMBERS) #IMPLIED	The format of each automatic page number inserted in this section.	The format of each automatic page number inserted in this section.	The format of each automatic page number inserted in this section.
NUMBER CDATA #IMPLIED	The starting page number for this section.	The starting page number for this section.	The starting page number for this section.

SECTIONNUMBERFORMAT (Modifier schema)

Element type	Construct	Modify	Deconstruct
SECTIONNUMBERFORMAT (empty)	Not applicable.	Allows you to specify the page number format.	Not applicable.
Attributes			
FORMAT (NUMERIC | ROMAN | SMALLROMAN | ALPHA | SMALLALPHA | ASIANNUMBERS) #IMPLIED	Not applicable.	The format of each automatic page number inserted in this section.	Not applicable.
PREFIX CDATA #IMPLIED	Not applicable.	The prefix to be added before each automatic page number inserted in this section.	Not applicable.
INITIALPAGENUMBER	Not applicable.	Fixes the page number for the first page of the PAGESEQUENCE to	Not applicable.

Element type	Construct	Modify	Deconstruct
		which it applies.	

SHADINGSTYLE (Modifier schema)

Element type	Construct	Modify	Deconstruct
SHADINGSTYLE	Allows you to specify the shading style applied to the text.	Allows you to specify the shading style applied to the text.	Allows you to specify the shading style applied to the text.
Attributes			
NAME	Specifies the name of the pre-defined text shading style to be applied to the text.	Specifies the name of the pre-defined text shading style to be applied to the text.	Specifies the name of the pre-defined text shading style to be applied to the text.
COLOR	Specifies the color of the text shading to be applied to the text.	Specifies the color of the text shading to be applied to the text.	Specifies the color of the text shading to be applied to the text.
SHADE	Specifies the shade of the text shading color.	Specifies the shade of the text shading color.	Specifies the shade of the text shading color.
OPACITY	Specifies the opacity of the text shading color.	Specifies the opacity of the text shading color.	Specifies the opacity of the text shading color.
LENGTH	Specifies the length of the text shading to be applied to the text. Allowed options are:	Specifies the length of the text shading to be applied to the text. Allowed options	Specifies the length of the text shading to be applied to the

Element type	Construct	Modify	Deconstruct
	INDENTS = text shading will be applied to the entire line between the defined indents. TEXT = text shading will be applied only the length of the text in each line. COLUMN = text shading will be applied to the entire length of the box.	are: INDENTS = text shading will be applied to the entire line between the defined indents. TEXT = text shading will be applied only the length of the text in each line. COLUMN = text shading will be applied to the entire length of the box.	text. Allowed options are: INDENTS = text shading will be applied to the entire line between the defined indents. TEXT = text shading will be applied only the length of the text in each line. COLUMN = text shading will be applied to the entire length of the box.
CLIPTOBOX (true | false)	Confines the text shading to the bounding box.	Confines the text shading to the bounding box.	Confines the text shading to the bounding box.
RIGHTPADDING	Specifies the right offset.	Specifies the right offset.	Specifies the right offset.
BOTTOMPADDING	Specifies the bottom offset.	Specifies the bottom offset.	Specifies the bottom offset.
LEFTPADDING	Specifies the left offset.	Specifies the left offset.	Specifies the left offset.
TOPPADDING	Specifies the top offset.	Specifies the top offset.	Specifies the top offset.
BORDERWIDTH	Specifies the width of the	Specifies the width of the frame.	Specifies the width of the

Element type	Construct	Modify	Deconstruct
	frame.		frame.
BORDERSTYLE	Specifies the frame style.	.Specifies the frame style.	Specifies the frame style.
BORDERCOLOR	Specifies the color of the frame.	Specifies the color of the frame.	Specifies the color of the frame.
BORDERSHADE	Specifies the shade of the frame color.	Specifies the shade of the frame color.	Specifies the shade of the frame color.
BORDEROPACITY	Specifies the opacity of the frame color.	Specifies the opacity of the frame color.	Specifies the opacity of the frame color.
LEFTBORDER (true | false)	Indicate that you want the frame to appear on the left of the text.	Indicate that you want the frame to appear on the left of the text.	Indicate that you want the frame to appear on the left of the text.
RIGHTBORDER (true | false)	Indicate that you want the frame to appear on the right of the text.	Indicate that you want the frame to appear on the right of the text.	Indicate that you want the frame to appear on the right of the text.
TOPBORDER (true | false)	Indicate that you want the frame to appear above the text.	Indicate that you want the frame to appear above the text.	Indicate that you want the frame to appear above the text.
BOTTOMBORDER (true | false)	Indicate that you want the frame to appear below the text.	Indicate that you want the frame to appear below the text.	Indicate that you want the frame to appear below the text.
BORDERRIGHTPADDING	Specifies the frames right offset.	Specifies the frames right offset.	
BORDERBOTTOMPADDING	Specifies the frames bottom offset.	Specifies the frames bottom offset.	Specifies the frames bottom offset.
BORDERLEFTPA	Specifies the	Specifies the frames	Specifies the

Element type	Construct	Modify	Deconstruct
DDING	frames left offset.	left offset.	frames left offset.
BORDERTOPPADDING	Specifies the frames top offset.	Specifies the frames top offset.	Specifies the frames top offset.

SHADOW (Modifier schema)

Element type	Construct	Modify	Deconstruct
SHADOW (empty)	Describes an automatic drop shadow.	Describes an automatic drop shadow.	Describes an automatic drop shadow.
Attributes			
COLOR CDATA #REQUIRED	Identifies the color of a drop shadow. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	Identifies the color of a drop shadow. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file, defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.	Identifies the color of a drop shadow. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.
SHADE CDATA #REQUIRED	Specifies the shade of the color applied to	Specifies the shade of the color applied to a drop shadow, as	Specifies the shade of the color applied to a drop

Element type	Construct	Modify	Deconstruct
	a drop shadow, as an integer percentage from 0 to 100.	an integer percentage from 0 to 100.	shadow, as an integer percentage from 0 to 100.
OPACITY CDATA #REQUIRED	Specifies the opacity of a drop shadow, specified as an integer percentage from 0 to 100.	Specifies the opacity of a drop shadow, specified as an integer percentage from 0 to 100.	Specifies the opacity of a drop shadow, specified as an integer percentage from 0 to 100.
ANGLE CDATA #REQUIRED	Specifies an angle in degrees for a drop shadow. Should be a floating point value between –180 and 180.	Specifies an angle in degrees for a drop shadow. Should be a floating point value between –180 and 180.	Specifies an angle in degrees for a drop shadow. Should be a floating point value between –180 and 180.
DISTANCE CDATA #REQUIRED	Specifies the distance in points from the edge of an item to the edge of the items drop shadow as a floating point value.	Specifies the distance in points from the edge of an item to the edge of the items drop shadow as a floating point value.	Specifies the distance in points from the edge of an item to the edge of the items drop shadow as a floating point value.
SKEW CDATA #REQUIRED	Specifies a skew angle for a drop shadow as a floating-point value from –75 degrees to 75 degrees	Specifies a skew angle for a drop shadow as a floating-point value from –75 degrees to 75 degrees	Specifies a skew angle for a drop shadow as a floating-point value from –75 degrees to 75 degrees
SCALE CDATA #REQUIRED	Specifies the size of an items	Specifies the size of an items drop	Specifies the size of an items drop

Element type	Construct	Modify	Deconstruct
	drop shadow as an integer percentage of the size of the item. The valid values are from 10 to 1000 percent.	shadow as an integer percentage of the size of the item. The valid values are from 10 to 1000 percent.	shadow as an integer percentage of the size of the item. The valid values are from 10 to 1000 percent.
BLUR CDATA #REQUIRED	Specifies the blur distance for a drop shadow, from 0 to 144 points, with higher values creating blurrier edges.	Specifies the blur distance for a drop shadow, from 0 to 144 points, with higher values creating blurrier edges.	Specifies the blur distance for a drop shadow, from 0 to 144 points, with higher values creating blurrier edges.
KNOCKOUTSHADOW (true | false) “false”	Specifies whether a shadow displays through semi-opaque areas of its item.	Specifies whether a shadow displays through semi-opaque areas of its item.	Specifies whether a shadow displays through semi-opaque areas of its item.
SYNCHRONIZEANGLE (true | false) “false”	Specifies whether to synchronize the angle of a drop shadow with the angles of other drop shadows in the layout.	Specifies whether to synchronize the angle of a drop shadow with the angles of other drop shadows in the layout.	Specifies whether to synchronize the angle of a drop shadow with the angles of other drop shadows in the layout.
RUNAROUNDSHADOW (true | false) “false”	Specifies whether to include a drop shadow with the text runaround specified in the RUNAROUND	Specifies whether to include a drop shadow with the text runaround specified in the RUNAROUND element. Note: The OUTSET attribute of	Specifies whether to include a drop shadow with the text runaround specified in the RUNAROUND element. Note:

Element type	Construct	Modify	Deconstruct
	element. Note: The OUTSET attribute of the RUNAROUND element is measured from the edges of the drop shadow. For example, if text is wrapping around a rectangular pull-out quote with a drop shadow, text will not overlap the drop shadow if RUNAROUNDSHADOW is set to true.	the RUNAROUND element is measured from the edges of the drop shadow. For example, if text is wrapping around a rectangular pull-out quote with a drop shadow, text will not overlap the drop shadow if RUNAROUNDSHADOW is set to true.	The OUTSET attribute of the RUNAROUND element is measured from the edges of the drop shadow. For example, if text is wrapping around a rectangular pull-out quote with a drop shadow, text will not overlap the drop shadow if RUNAROUNDSHADOW is set to true.
MULTIPLYSHADOW (true | false) "true"	Specifies how a drop shadow is combined with its background.When true, the shadow color is combined with the background color or colors using a "multiply" blending mode, producing a darker result (similar to an overprint).When false, the color	Specifies how a drop shadow is combined with its background.When true, the shadow color is combined with the background color or colors using a "multiply" blending mode, producing a darker result (similar to an overprint).When false, the color of the background is combined with the color of the shadow to create the	Specifies how a drop shadow is combined with its background.When true, the shadow color is combined with the background color or colors using a "multiply" blending mode, producing a darker result (similar to an overprint).When false, the color of the background is combined with

Element type	Construct	Modify	Deconstruct
	of the background is combined with the color of the shadow to create the intermediate shades you see on screen.In general, set to true if the shadow is a lighter color, and set to false if the shadow is black.	intermediate shades you see on screen.In general, set to true if the shadow is a lighter color, and set to false if the shadow is black.	the color of the shadow to create the intermediate shades you see on screen.
INHERITOPACITY (true | false) "false"	Specifies whether the drop shadow reflects the opacity or opacities of the item, such as differences in opacity between the box background and frame.	Specifies whether the drop shadow reflects the opacity or opacities of the item, such as differences in opacity between the box background and frame.	Specifies whether the drop shadow reflects the opacity or opacities of the item, such as differences in opacity between the box background and frame.

SHRINKACROSS (Modifier schema)

Element type	Construct	Modify	Deconstruct
SHRINKACROSS (#PCDATA)	Not applicable.	Shrinks a box horizontally to the left by the specified number of points. Note: A box can shrink on the same page or on other	Not applicable.

Element type	Construct	Modify	Deconstruct
		spreads and pages.	

SHRINKDOWN (Modifier schema)

Element type	Construct	Modify	Deconstruct
SHRINKDOWN (#PCDATA)	Not applicable.	Shrinks a box vertically toward the top of the page by the specified number of points. Note: A box can shrink on the same page or on other spreads and pages.	Not applicable.

SINGLEMASTERPAGEREFERENCE (Modifier schema)

Element type	Construct	Modify	Deconstruct
SINGLEMASTER PAGEREFERENCE (NAME*)	Not applicable.	Defines a sequence in which a single master page will be applied to pages in a page sequence. The given master page is applied to one page of the page sequence.	Not applicable.
Attributes			
NAME	Not applicable.	Specifies the name of the master page in the QuarkXPress template to be used.	Not applicable.

SIZE (Modifier schema)

Element type	Construct	Modify	Deconstruct
SIZE (empty)	Lets you specify the maximum or minimum size of a box for a fit-box-to-content operation.	Lets you specify the maximum or minimum size of a box for a fit-box-to-content operation.	Not applicable.
Attributes			
WIDTH CDATA #REQUIRED	The largest or smallest allowable width for the resized box.	The largest or smallest allowable width for the resized box.	Not applicable.
HEIGHT CDATA #REQUIRED	The largest or smallest allowable height for the resized box.	The largest or smallest allowable height for the resized box.	Not applicable.

SPINEIMAGE (Modifier schema)

Element type	Construct	Modify	Deconstruct
SPINEIMAGE (#PCDATA)	Not applicable.	Part of the <EBOOKMETADATA> element. Specifies the path of the spine image (if any).	Specifies the path of the spine image (if any).
Attributes			
INCLUDE (true | false) #REQUIRED	Not applicable.	If true, a spine image is included with Blio eBook output.	If true, a spine image is included with Blio eBook output.

SPLINESHAPE (Modifier schema)

Element type	Construct	Modify	Deconstruct
SPLINESHAPE (CONTOURS)	Specifies a complex spline shape in QuarkXPress (i.e., the curve of a Bezier box or Bezier text path).	Specifies a complex spline shape in QuarkXPress (i.e., the curve of a Bezier box or Bezier text path).	Specifies a complex spline shape in QuarkXPress (i.e., the curve of a Bezier box or Bezier text path).
Attributes			
RECTSHAPE (true | false) "false"	Specifies whether the shape is a pure rectangle.	Specifies whether the shape is a pure rectangle.	Specifies whether the shape is a pure rectangle.
INVERTEDSHAPE (true | false) "false"	Specifies whether the shape encodes the inverse of its area ("inside out").	Specifies whether the shape encodes the inverse of its area ("inside out").	Specifies whether the shape encodes the inverse of its area ("inside out").
HASSPLINES (true | false) "false"	Specifies whether any of the contours in the shape contains a spline.	Specifies whether any of the contours in the shape contains a spline.	Specifies whether any of the contours in the shape contains a spline.
HASHOLES (true | false) "false"	Specifies whether any of the contours is inside another.	Specifies whether any of the contours is inside another.	Specifies whether any of the contours is inside another.
NEWFORMAT (true | false) "false"	Specifies whether incompatible with "old" (3.31 and below) shapes.	Specifies whether incompatible with "old" (3.31 and below) shapes.	Specifies whether incompatible with "old" (3.31 and below) shapes.
MORETHANONE	Specifies	Specifies whether	Specifies whether

Element type	Construct	Modify	Deconstruct
TOPLEVELCONTOUR (true | false) “false”	whether there is more than one top-level contour.	there is more than one top-level contour.	there is more than one top-level contour.
CLOSEDSHAPE (true | false) “false”	Specifies whether all its contours are closed. (Polylines might not be.)	Specifies whether all its contours are closed. (Polylines might not be.)	Specifies whether all its contours are closed. (Polylines might not be.)
WELLFORMED (true | false) “false”	Specifies whether the shape does not intersect itself other than at the vertex.	Specifies whether the shape does not intersect itself other than at the vertex.	Specifies whether the shape does not intersect itself other than at the vertex.
TAGSALLOCATED (true | false) “false”	Specifies whether the vertex tags are set correctly.	Specifies whether the vertex tags are set correctly.	Specifies whether the vertex tags are set correctly.
INCOMPLETE (true | false) “false”	Specifies whether shape is associated with UNFINISHED box.	Specifies whether shape is associated with UNFINISHED box.	Specifies whether shape is associated with UNFINISHED box.
VERTSELECTED (true | false) “false”	Specifies whether one or more verts are selected.	Specifies whether one or more verts are selected.	Specifies whether one or more verts are selected.

SPREAD (Modifier schema)

Element type	Construct	Modify	Deconstruct
SPREAD (ID, PAGE*, (BOX | TABLE | GROUP |	Describes a spread (a series of one or more	Identifies the spread to be modified.	Describes a spread (a series of one or more

Element type	Construct	Modify	Deconstruct
COMPOSITIONZONE)*)	PAGE elements, divided by a SPINE)		PAGE elements, divided by a SPINE).
Attributes			
OPERATION (CREATE | DELETE) #IMPLIED	Not applicable.	Specifies whether to create or delete the indicated spread.	Not applicable.

STACKINGORDER (Modifier schema)

Element type	Construct	Modify	Deconstruct
STACKINGORDER (#PCDATA)	Lets you control whether a box or line is in front of or behind other items on the page.Only accepts SENDBACKWARD, SENDTOBACK, BRINGFORWARD, BRINGTOFRONOT.	Lets you control whether a box or line is in front of or behind other items on the page.Only accepts SENDBACKWARD, SENDTOBACK, BRINGFORWARD, BRINGTOFRONOT.	Not applicable.

STATICCONTENT (Modifier schema)

Element type	Construct	Modify	Deconstruct
STATICCONTENT (BOX*)	Not Applicable.	Enables you to specify content chunks that are intended to be repeated across multiple flow page (i.e. running headers	Not Applicable.

Element type	Construct	Modify	Deconstruct
		and footers).	

STOP (Modifier Schema)

Element type	Construct	Modify	Deconstruct
STOP	Describes a stopping point for a <GRADIENT> color.	Describes a stopping point for a <GRADIENT> color.	Describes a stopping point for a <GRADIENT> color.
Attributes			
COLOR	Specifies the name of the color.	Specifies the name of the color.	Specifies the name of the color.
CONTROLPOINT CDATA #REQUIRED	Specifies the control point of the stop, as an integer percentage from 0 to 100.	Specifies the control point of the stop, as an integer percentage from 0 to 100.	Specifies the control point of the stop, as an integer percentage from 0 to 100.
LOCATION CDATA #REQUIRED	Specifies the location of the stop, as an integer percentage from 0 to 100.	Specifies the location of the stop, as an integer percentage from 0 to 100.	Specifies the location of the stop, as an integer percentage from 0 to 100.
OPACITY CDATA #REQUIRED	Specifies the opacity of the color applied at this stop, as an integer percentage from 0 to 100.	Specifies the opacity of the color at this stop, as an integer percentage from 0 to 100.	Specifies the opacity of the color at this stop, as an integer percentage from

Element type	Construct	Modify	Deconstruct
			0 to 100.
SHADE CDATA #REQUIRED	Specifies the shade of the color applied at this stop, as an integer percentage from 0 to 100.	Specifies the shade of the color at this stop, as an integer percentage from 0 to 100.	Specifies the shade of the color.

STORY (Modifier schema)

Element type	Construct	Modify	Deconstruct
STORY (COPYFIT?, FITTEXT?,(PARAGRAPH | RICHTEXT | ANCHOREDBOXREF | LINKEDBOX | TEXTNODEPH | TEXTPH | HIDDEN | LIST | RUBI | CALLOUTANCHOR | INLINETABLE | INLINEBOX | PAGEBREAK | MATHEQUATION)*, OVERMATTER?)	Describes a text story in a text box or a chain of text boxes.	Describes a text story in a text box or a chain of text boxes.	Describes a text story in a text box or a chain of text boxes.
Attributes			
CLEAROLDTEXT (true | false) "true"	Not applicable.	Clears any existing text from the box.	Not applicable.
FITTEXTTOBOX (true | false)	Increases or decreases the	Increases or decreases the size of	Not applicable.

Element type	Construct	Modify	Deconstruct
“false”	size of the text to fit into the text box or text chain. Note: Text size increases only if Allow Text to Grow is checked in Text Modifier preferences (QuarkXPress Server/Edit > Preferences) in QuarkXPress Server. Note: To control how text fits to a box on a story-by-story basis, use the <FITTEXT> element type (for more information, see “FITTEXT (Modifier schema)”).	the text to fit into the text box or text chain. Note: Text size increases only if Allow Text to Grow is checked in Text Modifier preferences (QuarkXPress Server/Edit > Preferences) in QuarkXPress Server. Note: To control how text fits to a box on a story-by-story basis, use the <FITTEXT> element type (for more information, see “FITTEXT (Modifier schema)”).	
FILE CDATA #IMPLIED	The absolute path (on the server computer) to import a text document from.	The absolute path (on the server computer) to import a text document from.	Not applicable.
CONVERTQUOTES (true | false) “true”	Converts straight quotation marks to typesetter's quotation marks and double	Converts straight quotation marks to typesetter's quotation marks and double hyphens to em dashes in an	Not applicable.

Element type	Construct	Modify	Deconstruct
	hyphens to em dashes in an imported text file.	imported text file.	
INCLUDESTYLESHEETS (true | false) “true”	Adds any style sheets in an imported text file or document to the QuarkXPress project.	Adds any style sheets in an imported text file or document to the QuarkXPress project.	Not applicable.
STORYDIRECTION (HORIZONTAL | VERTICAL) #IMPLIED	Specified direction of this story.	Specified direction of this story.	Specified direction of this story.
LOCKSTORY (true | false) “true”	Specifies whether the story should be locked.	Specifies whether the story should be locked.	Specifies whether the story should be locked.
UID CDATA #IMPLIED	Unique identifier of this story.	Unique identifier of this story.	Unique identifier of this story.
TEXTOVERFLOW (ELLIPSIS | CLIP | CUSTOM STRING)	Specifies how to represent over flow text.	Specifies how to represent over flow text.	Specifies how to represent over flow text.

SUPPRESSOUTPUT (Modifier schema)

Element type	Construct	Modify	Deconstruct
SUPPRESSOUTPUT (#PCDATA)	Specifies whether a box is included in output.A true value does not include the box; a false value includes the box.	Specifies whether a box is included in output.A true value does not include the box; a false value includes the box.	Specifies whether a box is included in output.A true value does not include the box; a false value includes the box.

TAB (Modifier schema)

Element type	Construct	Modify	Deconstruct
TAB (empty)	Describes a single tab stop.	Describes a single tab stop.	Describes a single tab stop.
Attributes			
POSITION CDATA #REQUIRED	Specifies the position of a tab stop.	Specifies the position of a tab stop.	Specifies the position of a tab stop.
FILL CDATA #IMPLIED	Identifies one or two characters to repeat in order to fill the space between text and a tab stop.	Identifies one or two characters to repeat in order to fill the space between text and a tab stop.	Identifies one or two characters that repeat in order to fill the space between text and a tab stop.
ALIGNMENT (LEFT | RIGHT | CENTER | COMMA | DECIMAL | ALIGNON) "LEFT"	Indicates how a tab stop should be aligned.LEFT = Aligns text flush left on the tab stop.RIGHT = Aligns text flush right on the tab stop.CENTER = Aligns text centrally on the tab stop.DECIMAL = Aligns text on a decimal point (period).COMMA = Aligns text on a first comma.ALIGN ON = Aligns text on any character you specify in	Indicates how a tab stop should be aligned.LEFT = Aligns text flush left on the tab stop.RIGHT = Aligns text flush right on the tab stop.CENTER = Aligns text centrally on the tab stop.DECIMAL = Aligns text on a decimal point (period).COMMA = Aligns text on a first comma.ALIGN ON = Aligns text on any character you specify in the ALIGNON attribute.	Indicates how a tab stop is aligned.LEFT = Aligns text flush left on the tab stop.RIGHT = Aligns text flush right on the tab stop.CENTER = Aligns text centrally on the tab stop.DECIMAL = Aligns text on a decimal point (period).COMMA = Aligns text on a first comma.ALIGN ON = Aligns text on any character you specify in the ALIGNON

Element type	Construct	Modify	Deconstruct
	the ALIGNON attribute.		attribute.
ALIGNON CDATA #IMPLIED>	Specifies a specific character to align a tab stop on.	Specifies a specific character to align a tab stop on.	Specifies a specific character a tab stop is aligned on.
ENABLED (true | false) “true”			

TABLE (Modifier schema)

Element type	Construct	Modify	Deconstruct
TABLE (ID, METADATA?, (PARENTTABLE | TABLEBREAK | ADDCELLS | DELETECELLS | COLSPEC | ROW | FRAME | GEOMETRY | SHADOW | GRID)*)	Describes a table. Note: The size and position of a table are defined using the GEOMETRY element.	Describes a table. Note: The size and position of a table are defined using the GEOMETRY element.	Describes a table. Note: The size and position of a table are defined using the GEOMETRY element.
Attributes			
OPERATION (CREATE | DELETE) #IMPLIED	Not applicable.	Specifies whether to create or delete the indicated table.	Not applicable.
ROWS CDATA #IMPLIED	Specifies the number of rows in a table, including the header.	Specifies the number of rows in a table, including the header.	Specifies the number of rows in a table, including the header.
COLUMNS CDATA #IMPLIED	Specifies the number of	Specifies the number of columns in a table.	Specifies the number of

Element type	Construct	Modify	Deconstruct
	columns in a table.		columns in a table.
MAINTAINGEOMETRY (true | false | none) “none”	Controls whether inserted rows or columns affect the entire table's width and height.true = Table height and width remain the same.false = Table height and width change to accommodate new rows and columns.	Controls whether inserted rows or columns affect the entire table's width and height.true = Table height and width remain the same.false = Table height and width change to accommodate new rows and columns.	Controls whether inserted rows or columns affect the entire table's width and height.true = Table height and width remain the same.false = Table height and width change to accommodate new rows and columns.
COLOR CDATA #IMPLIED	Identifies the color of a table. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server.	Identifies the color of a table. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.	Identifies the color of a table. Note: Only the name of a color is included in this attribute. The definition of the color is stored in the projects Job Jackets file or defined using the Document Controls submenu in QuarkXPress Server, or an existing color created and saved in the project.
SHADE CDATA	Specifies the	Specifies the shade	Specifies the

Element type	Construct	Modify	Deconstruct
#IMPLIED	shade of the color applied to a table, as an integer percentage from 0 to 100.	of the color applied to a table, as an integer percentage from 0 to 100.	shade of the color applied to a table, as an integer percentage from 0 to 100.
OPACITY CDATA #IMPLIED	Specifies the opacity of the color applied to a table, specified as an integer percentage from 0 to 100.	Specifies the opacity of the color applied to a table, specified as an integer percentage from 0 to 100.	Specifies the opacity of the color applied to a table, specified as an integer percentage from 0 to 100.
BLENDSTYLE (SOLID | LINEAR | MIDLINEAR | RECTANGULAR | DIAMOND | CIRCULAR | FULLCIRCULAR | none) “none”	Specifies the type of blend applied to this table (linear, circular, rectangular, etc.).	Specifies the type of blend applied to this table (linear, circular, rectangular, etc.).	Specifies the type of blend applied to this table (linear, circular, rectangular, etc.).
BLENDANGLE CDATA #IMPLIED	Specifies the angle of the blend.	Specifies the angle of the blend.	Specifies the angle of the blend.
BLENDCOLOR CDATA #IMPLIED	Specifies the second color of the blend. The first color of the blend is the color applied to the table, as in QuarkXPress.	Specifies the second color of the blend. The first color of the blend is the color applied to the table, as in QuarkXPress.	Specifies the second color of the blend. The first color of the blend is the color applied to the table, as in QuarkXPress.
BLENDSHADE CDATA #IMPLIED	Specifies the shade applied to the second color of the blend. The shade of the first	Specifies the shade applied to the second color of the blend. The shade of the first color of the blend is	Specifies the shade applied to the second color of the blend. The shade of the first

Element type	Construct	Modify	Deconstruct
	color of the blend is the shade of the color applied to the table.	the shade of the color applied to the table.	color of the blend is the shade of the color applied to the table.
BLENDOPACITY CDATA #IMPLIED	Specifies the opacity applied to the second color of the blend. The opacity of the first color of the blend is the opacity of the color applied to the table.	Specifies the opacity applied to the second color of the blend. The opacity of the first color of the blend is the opacity of the color applied to the table.	Specifies the opacity applied to the second color of the blend. The opacity of the first color of the blend is the opacity of the color applied to the table.
ANCHOREDIN CDATA #IMPLIED	Not applicable.	Not applicable.	Indicates an anchored box and identifies its parent box.
AUTOFIT (rows | columns | all | none) “none”	Specifies whether the rows or columns will adjust size to fit the content.	Specifies whether the rows or columns will adjust size to fit the content.	Specifies whether the rows or columns will adjust size to fit the content.
AUTOFITMAXLIMIT CDATA #IMPLIED	Max limit for AUTOFIT.	Max limit for AUTOFIT.	Max limit for AUTOFIT.
ANCHOREDGROUPMEMBER CDATA #IMPLIED	Specifies that this table is a member of the indicated anchored group.	Specifies that this table is a member of the indicated anchored group.	Specifies that this table is a member of the indicated anchored group.
ADDTOREFLOW (true | false) #IMPLIED	Not applicable.	If true, adds this table to the project's reflow article.	Not applicable.

Element type	Construct	Modify	Deconstruct
		Equivalent to the Digital Publishing > Add to Reflow command in QuarkXPress.	
ARTICLENAME CDATA #IMPLIED	Not applicable.	Specifies the name of the project's reflow article (to which this table is being added as a component). If no reflow article exists and you do not include this attribute, the default reflow article name is used.	Not applicable.

TABLEBREAK (Modifier schema)

Element type	Construct	Modify	Deconstruct
TABLEBREAK (CHILDID | HEADER | FOOTER)*	Sets a table break for a HEADER or FOOTER or both.	Sets a table break for a HEADER or FOOTER or both.	Sets a table break for a HEADER or FOOTER or both.
Attributes			
BREAKHEIGHT CDATA #IMPLIED	Specifies the height at which a table is set to break.	Specifies the height at which a table is set to break.	Indicates the height at which a table is set to break.
MAINTAINLINK (true | false) “true”	Specifies whether a child table will maintain a link to its parent.	Specifies whether a child table will maintain a link to its parent.	Specifies whether a child table will maintain a link to its parent.
BREAKWHENANCHORED (true |	Specifies whether the	Specifies whether the table will break when	Specifies whether the table will

Element type	Construct	Modify	Deconstruct
false) #IMPLIED	table will break when anchored in a text flow.	anchored in a text flow.	break when anchored in a text flow.

TABLESTYLE (Modifier schema)

Element type	Construct	Modify	Deconstruct
TABLESTYLE ((ID, FRAME?, TROWSTYLE, HEADTROWSTYLE?, CONTINUEDTROWSTYLE?, FOOTERTROWSTYLE?, ODDTROWSTYLE?, EVENTROWSTYLE?, TCOLSTYLE, FIRSTTCOLSTYLE?, LASTTCOLSTYLE?, ODDTCOLSTYLE?, EVENTCOLSTYLE?))	Describes a style that can be applied to an <INLINETABLE>.	Describes a style that can be applied to an <INLINETABLE>.	Not applicable.
Attributes			
WIDTH CDATA #IMPLIED	Specifies the default width of the <TABLESTYLE>, expressed as a percentage of the width of the parent column	Specifies the default width of the <TABLESTYLE>, expressed as a percentage of the width of the parent column or as an absolute	Not applicable.

Element type	Construct	Modify	Deconstruct
	or as an absolute measurement. To specify a percentage, use a number without a unit indicator.	measurement. To specify a percentage, use a number without a unit indicator.	

TABSPEC (Modifier schema)

Element type	Construct	Modify	Deconstruct
TABSPEC (TAB)+	Describes a group of tab stops.	Describes a group of tab stops.	Describes a group of tab stops.

TBODY (Modifier schema)

Element type	Construct	Modify	Deconstruct
TBODY (TROW+)	Identifies the body portion of an <INLINETABLE>.	Identifies the body portion of an <INLINETABLE>.	Not applicable.

TCOL (Modifier schema)

Element type	Construct	Modify	Deconstruct
TCOL (LEFTGRID?, RIGHTGRID?)	Describes a column in an <INLINETABLE>.	Describes a column in an <INLINETABLE>.	Not applicable.
Attributes			
COLINDEX CDATA #REQUIRED	Identifies the position of the column, with the first column	Identifies the position of the column, with the first column being	Not applicable.

Element type	Construct	Modify	Deconstruct
	being column 1.	column 1.	
WIDTH CDATA #IMPLIED	Specifies the width of the column, either as an absolute measurement or as a percentage of the table width. To specify a percentage, use %. If you do not specify a width, column widths are distributed evenly.	Specifies the width of the column, either as an absolute measurement or as a percentage of the table width. To specify a percentage, use %. If you do not specify a width, column widths are distributed evenly.	Not applicable.
MINWIDTH CDATA #IMPLIED	Specifies the minimum width of the column, either as an absolute measurement or as a percentage of the table width. To specify a percentage, use %.	Specifies the minimum width of the column, either as an absolute measurement or as a percentage of the table width. To specify a percentage, use %.	Not applicable.
MAXWIDTH CDATA #IMPLIED	Specifies the maximum width of the column, either as an absolute measurement or as a percentage of the table width. To specify a percentage,	Specifies the maximum width of the column, either as an absolute measurement or as a percentage of the table width. To specify a percentage, use %.	Not applicable.

Element type	Construct	Modify	Deconstruct
	use %.		
COLOR CDATA #IMPLIED	Specifies the background color of the column.	Specifies the background color of the column.	Not applicable.
SHADE CDATA #IMPLIED	Specifies the background shade of the column.	Specifies the background shade of the column.	Not applicable.
STORYDIRECTION (HORIZONTAL | VERTICAL) #IMPLIED	Specifies the default story direction of the column.	Specifies the default story direction of the column.	Not applicable.

TCOLSTYLE (Modifier schema)

Element type	Construct	Modify	Deconstruct
TCOLSTYLE (LEFTGRID?, RIGHTGRID?)	Defines a style for a column in an <INLINETABLE>.	Defines a style for a column in an <INLINETABLE>.	Not applicable.
Attributes			
WIDTH CDATA #IMPLIED	Specifies the width of the column style.	Specifies the width of the column style.	Not applicable.
COLOR CDATA #IMPLIED	Specifies the background color of the column style.	Specifies the background color of the column style.	Not applicable.
SHADE CDATA #IMPLIED	Specifies the background shade of the column style.	Specifies the background shade of the column style.	Not applicable.

TCONTINUED (Modifier schema)

Element type	Construct	Modify	Deconstruct
TCONTINUED (TROW+)	Identifies a continued" indicator row in the header of an <INLINETABLE>.	Identifies a continued" indicator row in the header of an <INLINETABLE>.	Not applicable.

TEXT (Modifier schema)

Element type	Construct	Modify	Deconstruct
TEXT ((INSET)*, STORY)	Container for an INSET and STORY element.	Container for an INSET and STORY element.	Container for an INSET and STORY element.
Attributes			
ANGLE CDATA #IMPLIED	Specifies a rotation angle for text as a floating-point value between –360 degrees and 360 degrees.	Specifies a rotation angle for text as a floating-point value between –360 degrees and 360 degrees.	Indicates a rotation angle for text as a floating-point value between –360 degrees and 360 degrees.
SKEW CDATA #IMPLIED	Specifies a skew angle for text as a floating-point value from –75 degrees to 75 degrees.	Specifies a skew angle for text as a floating-point value from –75 degrees to 75 degrees.	Indicates a skew angle for text as a floating-point value from –75 degrees to 75 degrees.
COLUMNS CDATA #IMPLIED	Specifies the number of columns in a text box.	Specifies the number of columns in a text box.	Indicates a number of columns in a text box.
GUTTERWIDTH CDATA #IMPLIED	Specifies the width of the gutter(s) in a multi-column	Specifies the width of the gutter(s) in a multi-column text box.	Specifies the width of the gutter(s) in a multi-column text

Element type	Construct	Modify	Deconstruct
	text box.		box.
FLIPVERTICAL (true | false | none) “none”	Flips the text vertically in a text box.	Flips the text vertically in a text box.	Indicates the text is flipped vertically in a text box.
FLIPHORIZONTAL (true | false | none) “none”	Flips the text horizontally in a text box.	Flips the text horizontally in a text box.	Indicates the text is flipped horizontally in a text box.
VERTICALALIGNMENT (TOP | CENTERED | BOTTOM | JUSTIFIED | none) “none”	Vertically aligns the text.	Vertically aligns the text.	Indicates the vertical alignment of text.
INTERPARAGRAPHMAX CDATA #IMPLIED	Specifies the space between two consecutive paragraphs	Specifies the space between two consecutive paragraphs	Specifies the space between two consecutive paragraphs
FIRSTBASELINEMIN (ASCENT | CAPHEIGHT | CAPACCENT | none) “none”	Specifies the minimum distance between the top edge of a text box and the baseline of the first line of text.ASCENT = Specifies the distance based on the space needed for the accent mark of the tallest character.CAPHEIGHT= Specifies the distance	Specifies the minimum distance between the top edge of a text box and the baseline of the first line of text.ASCENT = Specifies the distance based on the space needed for the accent mark of the tallest character.CAPHEIGHT= Specifies the distance based on the cap height of the tallest character.CAPACCENT = Specifies the	Indicates the minimum distance between the top edge of a text box and the baseline of the first line of text.ASCENT = Specifies the distance based on the space needed for the accent mark of the tallest character.CAPHEIGHT= Specifies the distance based on the cap

Element type	Construct	Modify	Deconstruct
	based on the cap height of the tallest character.CAPACCENT = Specifies the distance based on the cap height of the tallest character plus the space required for an accent mark over an uppercase character.	distance based on the cap height of the tallest character plus the space required for an accent mark over an uppercase character.	height of the tallest character.CAPACCENT = Specifies the distance based on the cap height of the tallest character plus the space required for an accent mark over an uppercase character.
OFFSET CDATA #IMPLIED	Specifies the distance between the first text baseline in the text box and the top inside edge of the text box.	Specifies the distance between the first text baseline in the text box and the top inside edge of the text box.	Indicates the distance between the first text baseline in the text box and the top inside edge of the text box.
RUNTEXTAROUNDALLSIDES (true | false | none) “none”	Indicates text runaround on all sides of an item.	Indicates text runaround on all sides of an item.	Indicates text runaround on all sides of an item.
TEXTORIENTATION (ROTATE | SKEW | ROTATEANDSKEW | NOROTATEANDSKEW | none) “none”	Specifies how the text should be attached to a line.	Specifies how the text should be attached to a line.	Indicates how the text is attached to a line.
TEXTALIGN (ASCENT |	Specifies the part of a font to	Specifies the part of a font to use for	Indicates the part of a font being

Element type	Construct	Modify	Deconstruct
CENTER | BASELINE | DESCENT | none) “none”	use for positioning characters on a line.	positioning characters on a line.	used for positioning characters on a line.
TEXTALIGNWITH LINE (TOP | CENTER | BOTTOM | none) “none”	Specifies how to align text to a line.	Specifies how to align text to a line.	Indicates text is aligned to a line.
FLIPTEXT (true | false | none) “none”	Flips the characters horizontally on a line.	Flips the characters horizontally on a line.	Indicates characters are horizontally flipped on a line.
GRIDSTYLE CDATA #IMPLIED	Identifies the grid style applied to the text.	Identifies the grid style applied to the text.	Identifies the grid style applied to the text.

TEXTATTRIBUTE (Modifier schema)

Element type	Construct	Modify	Deconstruct
TEXTATTRIBUTE (INSET)	Specifies the text-related attributes of a box created with the INLINEBOX element type.	Specifies the text-related attributes of a box created with the INLINEBOX element type.	Not applicable.
Attributes			
COLUMNS NMTOKEN #IMPLIED	Specifies the number of columns in a text box.	Specifies the number of columns in a text box.	Not applicable.
GUTTERWIDTH NMTOKEN #IMPLIED	Specifies the width of the gutter(s) in a multi-column	Specifies the width of the gutter(s) in a multi-column text box.	Not applicable.

Element type	Construct	Modify	Deconstruct
	text box.		

TEXTNODEPH (Modifier schema)

Element type	Construct	Modify	Deconstruct
TEXTNODEPH ((TEXTNODEPH | PARAGRAPH | RICHTEXT | OVERMATTER | TEXTPH)*, METADATA?)	A text node placeholder allows metadata to be defined hierarchically on a region of text, and can contain further text node placeholders and text placeholders.	A text node placeholder allows metadata to be defined hierarchically on a region of text, and can contain further text node placeholders and text placeholders.	A text node placeholder allows metadata to be defined hierarchically on a region of text, and can contain further text node placeholders and text placeholders.
Attributes			
NAME CDATA #REQUIRED	The name of the text node placeholder. A placeholder name may not be Unique within the Box or XML Hierarchy.	The name of the text node placeholder. A placeholder name may not be Unique within the Box or XML Hierarchy.	The name of the text node placeholder. A placeholder name may not be Unique within the Box or XML Hierarchy.
OWNER (1347639377) "1347639377"	The XTensions ID of the XTensions that created this placeholder. The default XT ID is PlaceHolderSXT ID (1347639377). All placeholders created through	The XTensions ID of the XTensions that created this placeholder. The default XT ID is PlaceHolderSXT ID (1347639377). All placeholders created through Modifier should use this ID. This ID is assigned	The XTensions ID of the XTensions that created this placeholder.

Element type	Construct	Modify	Deconstruct
	Modifier should use this ID. This ID is assigned by default by the schema, so there is no need to specify this manually. schema validation will add this attribute.	by default by the schema, so there is no need to specify this manually. schema validation will add this attribute).	

TEXTPH (Modifier schema)

Element type	Construct	Modify	Deconstruct
TEXTPH ((PARAGRAPH | RICHTEXT | OVERMATTER)*, METADATA?)	A text placeholder allows metadata to be defined on a region of text.	A text placeholder allows metadata to be defined on a region of text.	A text placeholder allows metadata to be defined on a region of text.
Attributes			
NAME CDATA #REQUIRED	The name of the text node placeholder.	The name of the text node placeholder.	The name of the text node placeholder.
OWNER (1347639377) "1347639377"	The XTensions ID of the XTensions that created this placeholder. The default XT ID is PlaceHolderSXT ID (1347639377). All placeholders created through	The XTensions ID of the XTensions that created this placeholder. The default XT ID is PlaceHolderSXT ID (1347639377). All placeholders created through Modifier should use this ID. This ID is assigned	The XTensions ID of the XTensions that created this placeholder.

Element type	Construct	Modify	Deconstruct
	Modifier should use this ID. This ID is assigned by default by the schema, so there is no need to specify this manually. schema validation will add this attribute.	by default by the schema, so there is no need to specify this manually. schema validation will add this attribute.	

TFOOT (Modifier schema)

Element type	Construct	Modify	Deconstruct
TFOOT (TROW+)	Identifies the footer portion of an <INLINETABLE>.	Identifies the footer portion of an <INLINETABLE>.	Not applicable.

THEAD (Modifier schema)

Element type	Construct	Modify	Deconstruct
THEAD (TROW+, TCONTINUED?)	Identifies a header in an <INLINETABLE>.	Identifies a header in an <INLINETABLE>.	Not applicable.

TITLE (Modifier schema)

Element type	Construct	Modify	Deconstruct
TITLE (#PCDATA)	Not applicable.	Specifies the title of an e-book.	Specifies the title of an e-book.

TOP (Modifier schema)

Element type	Construct	Modify	Deconstruct
TOP (#PCDATA)	The distance between the box or lines top edge and the top of the page, in points.	The distance between the box or lines top edge and the top of the page, in points.	The distance between the box or lines top edge and the top of the page, in points.

TOPGRID (Modifier schema)

Element type	Construct	Modify	Deconstruct
TOPGRID (empty)	Describes a grid line on the top edge of a cell in an <INLINETABLE>.	Describes a grid line on the top edge of a cell in an <INLINETABLE>.	Not applicable.
Attributes			
TYPE (TOP | LEFT | BOTTOM | RIGHT) #IMPLIED	Specifies the location of the grid line.	Specifies the location of the grid line.	Not applicable.
STYLE CDATA #IMPLIED	Identifies the <TABLESTYLE> that styles this grid line. If you specify this value, you do not have to specify the remaining attributes. If you specify the remaining attributes, those attribute values override the corresponding	Identifies the <TABLESTYLE> that styles this grid line. If you specify this value, you do not have to specify the remaining attributes. If you specify the remaining attributes, those attribute values override the corresponding <TABLESTYLE> values.	Not applicable.

Element type	Construct	Modify	Deconstruct
	<TABLESTYLE> values.		
WIDTH CDATA #IMPLIED	Specifies the width of the grid line in points.	Specifies the width of the grid line in points.	Not applicable.
COLOR CDATA #IMPLIED	Specifies the color of the grid line.	Specifies the color of the grid line.	Not applicable.
SHADE CDATA #IMPLIED	Specifies the shade of the grid line.	Specifies the shade of the grid line.	Not applicable.
OPACITY CDATA #IMPLIED	Specifies the opacity of the grid line.	Specifies the opacity of the grid line.	Not applicable.
GAPCOLOR CDATA #IMPLIED	Specifies the color of the gap (if any) between the lines that make up the grid line.	Specifies the color of the gap (if any) between the lines that make up the grid line.	Not applicable.
GAPSHADE CDATA #IMPLIED	Specifies the shade of the gap (if any) between the lines that make up the grid line.	Specifies the shade of the gap (if any) between the lines that make up the grid line.	Not applicable.
GAPOPACITY CDATA #IMPLIED	Specifies the opacity of the gap (if any) between the lines that make up the grid line.	Specifies the opacity of the gap (if any) between the lines that make up the grid line.	Not applicable.

TROW (Modifier schema)

Element type	Construct	Modify	Deconstruct
TROW (TOPGRID?, BOTTOMGRID?, ENTRY*)	Describes a row in an <INLINETABLE>.	Describes a row in an <INLINETABLE>.	Not applicable.
Attributes			
COLOR CDATA #IMPLIED	Identifies the background color of the cells in the <TROW>.	Identifies the background color of the cells in the <TROW>.	Not applicable.
SHADE CDATA #IMPLIED	Identifies the background shade of the cells in the <TROW>.	Identifies the background shade of the cells in the <TROW>.	Not applicable.
STORYDIRECTION (HORIZONTAL | VERTICAL) #IMPLIED	Specifies the story direction of the cells in the <TROW>.	Specifies the story direction of the cells in the <TROW>.	Not applicable.
ANGLE	Specifies the angle of the row	Specifies the angle of the row	Not applicable.
VALIGN	Specifies the vertical alignment of the row	Specifies the vertical alignment of the row	Not applicable.
ALIGNMENT	Specifies the alignment of the row	Specifies the alignment of the row	Not applicable.
ROWHEIGHT	Specifies the height of the row	Specifies the height of the row	Not applicable.

TROWSTYLE (Modifier schema)

Element type	Construct	Modify	Deconstruct
TROWSTYLE	TROWSTYLE	Defines a style for	Not applicable.

Element type	Construct	Modify	Deconstruct
(TOPGRID?, BOTTOMGRID?)	element typeDefines a style for rows in an <INLINETABLE>.	rows in an <INLINETABLE>.	
Attributes			
PARASTYLE CDATA #IMPLIED	Identifies the paragraph style sheet for the row style.	Identifies the paragraph style sheet for the row style.	Not applicable.
ALIGNMENT (LEFT | RIGHT | CENTER | JUSTIFIED | FORCED) #IMPLIED	Identifies the paragraph alignment for the row style.	Identifies the paragraph alignment for the row style.	Not applicable.
ANGLE CDATA #IMPLIED	Identifies the text angle for the row style.	Identifies the text angle for the row style.	Not applicable.
VALIGN (TOP | CENTER | BOTTOM) #IMPLIED	Specifies the vertical alignment of the row style.	Specifies the vertical alignment of the row style.	Not applicable.
COLOR CDATA #IMPLIED	Specifies the background color of the row style.	Specifies the background color of the row style.	Not applicable.
SHADE CDATA #IMPLIED	Specifies the background shade of the row style.	Specifies the background shade of the row style.	Not applicable.
INSET CDATA #IMPLIED	Specifies the text inset for all four sides of cells that use the row style.	Specifies the text inset for all four sides of cells that use the row style.	Not applicable.

VALUE (Modifier schema)

Element type	Construct	Modify	Deconstruct
VALUE (#PCDATA)	pecifies the VALUE of the key/value pair. The value can be given in CDATA form only, such as:<METADATA><VALUE KEY="myKey"><![CDATA[METADATAVALUE]]></VALUE></METADATA>	Specifies the VALUE of the key/value pair. The value can be given in CDATA form only, such as:<METADATA><VALUE KEY="myKey"><![CDATA[METADATAVALUE]]></VALUE></METADATA>	Specifies the VALUE of the key/value pair. The value can be given in CDATA form only, such as:<METADATA><VALUE KEY="myKey"><![CDATA[METADATAVALUE]]></VALUE></METADATA>
Attributes			
KEY CDATA #REQUIRED	Specifies the KEY attribute of the key/value pair.	Specifies the KEY attribute of the key/value pair. Metadata that contains a value for KEY but no value for VALUE will delete any metadata matching the value for KEY.	Specifies the KEY attribute of the key/value pair.

VERTEX (Modifier schema)

Element type	Construct	Modify	Deconstruct
VERTEX (LEFTCONTROLPOINT?, VERTEXPOINT, RIGHTCONTROLPOINT?)	A single vertext (i.e. Line segment) in a bezier curve.	A single vertext (i.e. Line segment) in a bezier curve.	A single vertext (i.e. Line segment) in a bezier curve.
Attributes			
SMOOTHVERTEX	Specifies	Specifies whether the	Specifies whether

Element type	Construct	Modify	Deconstruct
(true | false) “false”	whether the given vertex is “straight” — i.e. C1 continuous.	given vertex is “straight” — i.e. C1 continuous.	the given vertex is “straight” — i.e. C1 continuous.
STRAIGHTEDGE (true | false) “false”	Specifies whether the following edge is “straight”.	Specifies whether the following edge is “straight”.	Specifies whether the following edge is “straight”.
SYMMVERTEX (true | false) “false”	Specifies whether the given vertex is also symmetrical — i.e., C2 continuous.	Specifies whether the given vertex is also symmetrical — i.e., C2 continuous.	Specifies whether the given vertex is also symmetrical — i.e., C2 continuous.
CUSPVERTEX (true | false) “false”	Specifies whether the vertex is not smooth or symmetric.	Specifies whether the vertex is not smooth or symmetric.	Specifies whether the vertex is not smooth or symmetric.
TWISTED (true | false) “false”	Specifies whether the following (splined) edge intersects itself.	Specifies whether the following (splined) edge intersects itself.	Specifies whether the following (splined) edge intersects itself.
VERTEXSELECTED (true | false) “false”	Specifies whether the given vertex is selected.	Specifies whether the given vertex is selected.	Specifies whether the given vertex is selected.

VERTEXPOINT (Modifier schema)

Element type	Construct	Modify	Deconstruct
VERTEXPOINT (empty)	Each point on a curve is described by	Each point on a curve is described by three geometric positions:	Each point on a curve is described by three

Element type	Construct	Modify	Deconstruct
	three geometric positions: the x,y coordinate of the vertex point (this coordinate is relative to the bounding geometry of the shape, not the page), and the left and right control handles–as you would see onscreen in the QuarkXPress user environment. For more information on drawing and manipulating bezier curves, please see A Guide to QuarkXPress.	the x,y coordinate of the vertex point (this coordinate is relative to the bounding geometry of the shape, not the page), and the left and right control handles–as you would see onscreen in the QuarkXPress user environment. For more information on drawing and manipulating bezier curves, please see A Guide to QuarkXPress.	geometric positions: the x,y coordinate of the vertex point (this coordinate is relative to the bounding geometry of the shape, not the page), and the left and right control handles–as you would see onscreen in the QuarkXPress user environment. For more information on drawing and manipulating bezier curves, please see A Guide to QuarkXPress.

VERTICES (Modifier schema)

Element type	Construct	Modify	Deconstruct
VERTICES (VERTEX+)	A collection of vertexes which, combined, make up a contour.	A collection of vertexes which, combined, make up a contour.	A collection of vertexes which, combined, make up a contour.

WIDTH (Modifier schema)

Element type	Construct	Modify	Deconstruct
WIDTH (#PCDATA)	Indicates the width of an item.	Indicates the width of an item.	Indicates the width of an item.

XREF (Modifier schema)

Element type	Construct	Modify	Deconstruct
XREF (#PCDATA)	The XREF element is used to describe cross references.	XREF element typeThe XREF element is used to describe cross references.	XREF element typeThe XREF element is used to describe cross references.
Attributes			
XREFSTYLE (REFNOTENUMBER | PAGENUMBER | FORMATTEDREFNOTENUMBER | PARANUMBERWITHOUTCONTEXT | PARANUMBERWITHCONTEXT | BODYTEXT)	Specifies the type of cross reference.	Specifies the type of cross reference.	Specifies the type of cross reference.
HREF CDATA #IMPLIED			
HYPERLINK (true | false) “true”	Indicates whether or not the hyperlink is enabled/disabled in the PDF output.	Indicates whether or not the hyperlink is enabled/disabled in the PDF output.	Indicates whether or not the hyperlink is enabled/disabled in the PDF output.
INCLUDEABOVEBELOW (true |	Indicates whether the text	Indicates whether the text above and	Indicates whether the text above

Element type	Construct	Modify	Deconstruct
false) “false”	above and below the text should be included in the cross reference scope.	below the text should be included in the cross reference scope.	and below the text should be included in the cross reference scope.
SEPARATOR CDATA #IMPLIED	Identifies the text to be used as the separator.	Identifies the text to be used as the separator.	Identifies the text to be used as the separator.
CHARSTYLE CDATA #IMPLIED	Identifies a character style sheet to be applied to the cross reference.	Identifies a character style sheet to be applied to the cross reference.	Identifies a character style sheet to be applied to the cross reference.

Using SSL

You can configure QuarkXPress Server with different security options. In addition to your own network security specifications, you can specify Secure Sockets Layer (SSL) protocol for client applications.

Secure Sockets Layer (SSL) support

You can configure Tomcat (and therefore all QuarkXPress Server clients) to run in secure mode with Secure Sockets Layer (SSL) technology. This section explains the configuration process.

➡ It is also possible to run QuarkXPress Server without embedding Tomcat in the JVM. For more information, see the QuarkXPress Server *ReadMe* file.

To manage Web applications in the QuarkXPress Server environment, QuarkXPress Server embeds an instance of Apache Tomcat 6.18 in its JVM.

When you enable SSL, it applies to all QuarkXPress Server client applications.

Enabling SSL

The instructions below address two scenarios. The "server.xml" file you edit contains XML tags for both scenarios, which you need to enable or disable by "commenting" and "uncommenting" specific tags.

To enable SSL for secure HTTP for all QuarkXPress Server applications:

1. Open the "conf" folder in your QuarkXPress Server folder.
2. Open "server.xml" in a text-editing application.
3. Locate the following tag (preceded by the comment <!-- Define a non-SSL HTTP/1.1 Connector on port 8080 -->) and comment it out.

```
<Connector port="8080" maxHttpHeaderSize="8192"
maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
enableLookups="false" redirectPort="61399" acceptCount="100"
connectionTimeout="20000" disableUploadTimeout="true"
URIEncoding="UTF-8"/>
```

4. Locate the following tag (preceded by the comment <!-- Define a SSL HTTP/1.1 Connector on port 61399 -->) and uncomment it.

```
<Connector port="61399"
maxHttpHeaderSize="8192"MaxThreads="150"
minSpareThreads="25"
maxSpareThreads="75"enableLookups="false"
disableUploadTimeout="true"acceptCount="100" scheme="https"
secure="true"clientAuth="false" sslProtocol="TLS" />
```

5 Replace 61399 with 61400 (or any port on which Tomcat will be listening for secure connections).

6 Save and close "server.xml."

7 Open the "ServerApp.properties" file (in the "conf" folder) and enter the port number from step 5 for qxpswebserver.port.

➡ This change means QuarkXPress Server client applications can use HTTPS. For example, the URL for a QuarkXPress Server user would be as follows: https://[server name]:[port]/.

Enabling HTTP and HTTPS

To enable HTTP and HTTPS:

1 Open the "conf" folder in your QuarkXPress Server folder.

2 Open "server.xml" in a text-editing application.

3 Uncomment the following tag:

```
<Connector port="61399"
maxHttpHeaderSize="8192"MaxThreads="150"
minSpareThreads="25"
maxSpareThreads="75"enableLookups="false"
disableUploadTimeout="true"acceptCount="100" scheme="https"
secure="true"clientAuth="false" sslProtocol="TLS" />
```

4 Save and close "server.xml."

➡ This feature allows QuarkXPress Server application users to access QuarkXPress Server with HTTPS or HTTP.

Verifying and using SSL

To verify and use SSL:

1 Start the QuarkXPress Server

2 Test QuarkXPress Server access by navigating to the QuarkXPress Server Web interface HTTPS. For example: https://[server]:[port]/qxpsadmin

Keystores and SSL certificates

A *certificate* is a file on a Web server that is used in encryption and confirmation between two endpoints to establish a secure connection. A *keystore* is essentially a database of digital certificates on the Web server.

You can obtain an SSL certificate from a trusted Certificate Authority (CA). Import the certificate into the keystore used by QuarkXPress Server's JVM.

For more information about the importance of keystores, use the following URL: *http://tomcat.apache.org/tomcat-6.0-doc/ssl-howto.html*.

QuarkXPress Server XTensions software

Just as XTensions software provides additional functionality to QuarkXPress, XTensions software enables QuarkXPress Server to do things it can't do by default. The XTensions modules included with QuarkXPress Server allow clients to render projects as PDF files, apply QuarkVista picture effects to pictures, dynamically update pictures in picture boxes and text in text boxes (as well as boxes themselves), import data on the fly, manipulate layers in projects, and more.

CopyDeskArticle XTensions software

CopyDeskArticle XTensions software allows QuarkXPress Server to do the following things:

- Render QuarkCopyDesk articles
- Export QuarkCopyDesk articles from a QuarkXPress project
- Add a QuarkCopyDesk article to an existing QuarkXPress project
- Create and delete components in a QuarkCopyDesk article within a QuarkXPress project

Rendering articles

To render QuarkCopyDesk articles, use the copydesk namespace, as follows:

http://*[server]*:*[port]*/*[render type]*/copydesk/*[articlename]*

For example, to render "Article.qcd" as a PDF file, you could use a URL like the following:

http://QXPServer.8080/pdf/copydesk/Article.qcd

Exporting articles

To export an article from a QuarkXPress project, use the QCDDOC namespace, as follows:

http://*[server]*:*[port]*/qcddoc/*[project name]*?article=*[article name or ID]*

For example, to export the article named "Article1" from the project named "Project1.qxp," you could use a URL like the following:

http://QXPServer.8080/qcddoc/Project1.qxp?article=Article1

➡ You cannot export an article from a QuarkXPress project unless the article has been created and named in QuarkXPress using CopyDeskArticleXT XTensions software.

To export an article in a particular format, use the format parameter.

- To export a standard article, use format=fullfeatured.
- To export an article in lightweight mode, use format=lightweight.This format can be useful in situations where file size is an issue. The lightweight file format includes only those items that a QuarkCopyDesk user is supposed to work on.

For example:

http://QXPServer.8080/qcddoc/Project1.qxp?article=Article1?format=lightweight

➡ You cannot export page pictures in lightweight mode.

Adding articles to projects

You can use Modifier XTensions software to create a QuarkCopyDesk article within an existing QuarkXPress project. For example, to add an article named "New Article" to the project named "Project1.qxp," you could use XML like the following:

```
<PROJECT PROJECTNAME="Project1.qxp" XMLVERSION="8.0">
<LAYOUT POINTSPERINCH="72">
<ID NAME="Layout 1" UID="1"/>
<ARTICLE OPERATION="CREATE" DOCFORMAT="FULLFEATURED">
<ID NAME="New Article" UID="4"/>
<RGBCOLOR BLUE="98" GREEN="254" RED="0"/>
<COMPONENT BOXNAME="Box1" BOXUID="1"
COMPONENTCLASS="CT_TEXT"
NAME="Component 1" UID="1"/>
</ARTICLE>
</LAYOUT>
</PROJECT>
```

Creating and deleting components

You can use Modifier XTensions software to create and delete components in QuarkCopyDesk articles. For example, to add a component named "New Component" to the article named "Article 1" in the project named "Project1.qxp," you could use XML like the following:

```
<PROJECT JOBJACKET="Project1 Job Jacket" JOBTICKET="Default
Job Ticket 1:Project1"
PROJECTNAME="Project1.qxp" XMLVERSION="8.0">
<LAYOUT POINTSPERINCH="72">
<ID NAME="Layout 1" UID="1"/>
```

```
<ARTICLE DOCFORMAT="FULLFEATURED">
<ID NAME="Article 1" UID="1"/>
<COMPONENT OPERATION="CREATE" BOXUID="9"
COMPONENTCLASS="CT_TEXT"
NAME="New Component"/>
</ARTICLE>
</LAYOUT>
</PROJECT>
```

To delete the component named “New Component” from the article named “Article1.qcd,” you could use XML like the following:

```
<PROJECT JOBJACKET="Project1 Job Jacket" JOBTICKET="Default
Job Ticket 1:Project1"
PROJECTNAME="Project1.qxp" XMLVERSION="8.0">
<LAYOUT POINTSPERINCH="72">
<ID NAME="Layout 1" UID="1"/>
<ARTICLE DOCFORMAT="FULLFEATURED">
<ID NAME="Article 1" UID="1"/>
<COMPONENT OPERATION="DELETE" NAME="New Component"/>
</ARTICLE>
</LAYOUT>
</PROJECT>
```

PDF Filter XTensions software

PDF Filter XTensions software allows QuarkXPress Server to render a QuarkXPress project as a PDF file. To render QuarkXPress projects as PDF files when PDF is not the QuarkXPress Server default render type, use the PDF namespace, as follows:

http://[server]:[port]/pdf/[projectname]

For information about PDF preferences, see “*Preferences — PDF*.” To take advantage of more detailed preferences, create a PDF output style and use that output style when rendering projects as PDF files.

Modifier XTensions software

Modifier XTensions software lets clients perform all of the following tasks using XML:

- Modify the properties of pictures in a QuarkXPress project
- Modify the text in text boxes within a QuarkXPress project
- Modify the properties of text boxes and picture boxes in a QuarkXPress project
- Create and delete picture boxes and text boxes in a QuarkXPress project

- Import text or text strings into text boxes within a QuarkXPress project
- Import pictures into picture boxes within a QuarkXPress project
- Save modified QuarkXPress projects in any supported format to any location on the network (and also in the QuarkXPress Server document pool)
- Create and delete pages
- Create and delete layers
- Move items within layers
- Create and delete tables
- Modify tables and their contents
- Create QuarkCopyDesk articles and components
- Create lines, anchored boxes, and Bézier boxes
- Create lists
- Group and ungroup boxes
- Divide layouts into sections

To use Modifier XTensions software, a client creates an XML file indicating the actions to be taken and sends that XML file to the QuarkXPress Server application, where Modifier XTensions software reads the XML and makes the requested changes. Clients can use a single XML file or string to manipulate multiple documents and boxes.

➡ Modifier XTensions software supports both GET and POST functionality.

➡ Modifier XTensions software supports XML containing code that uses Unicode UTF-8 and UTF-16 encodings. Use the encoding attribute of the XML declaration to specify an encoding, as you would with any other XML file.

Using Modifier XTensions software

To use Modifier XTensions software:

1. Create a QuarkXPress project. Note the IDs or names of any text and picture boxes you want to manipulate.
2. Upload the project to the QuarkXPress Server document pool.
3. Create XML that describes the changes you want, as described in "*Creating XML for Modifier XTensions software*."
4. Send the XML to the server in one of the following ways:

- Put the XML in a file on the server and then use a URL to point to the file, as follows:

http://*server:port/namespace/path/projectname*?modify=file:[absolute path to XML file on server or relative path to XML file on server relative to the document pool]

- Put the XML in the URL, as follows:

http://[server]:[port]/[namespace]/path/projectname?modify=[XML string]

➡ Clients can also send XML in the form of a POST request.

Creating XML for Modifier XTensions software

All XML used with Modifier XTensions software uses the Modifier schema. This schema is documented in full in "*Modifier schema (annotated)*."

In general, the structure for addressing items in a particular layout is as follows:

```
<PROJECT>
<LAYOUT>
<ID NAME="
[name of layout]
">
<
[Item being addressed]
>
<
[Parameters of item]
>
</
[Item being addressed]
>
</LAYOUT>
</PROJECT>
```

Layer XTensions software

You can use QuarkXPress Server Layer XTensions software to control the visibility of specific layers in a rendered QuarkXPress project. You can also add layers, delete layers, edit layer attributes, and control whether layers are rendered.

➡ When you use the getdocinfo namespace, QuarkXPress Server returns information about items on all of the layers in the QuarkXPress project, including layers that are not visible.

Clients can use the layer parameter to specify a layer (even a hidden layer) to be rendered. For example, the URL http://QXPServer:8080/doc.qxp?layer=layer1 renders only the layer named "layer1" in the project named "doc.qxp."

Clients can specify more than one layer in a single URL. For example, the URL http://QXPServer:8080/doc.qxp?layer=layer1,layer2 renders the layers named "layer1" and "layer2."

For information about layer preferences, see "*Preferences — Layers*."

➡ If **Suppress Output** is selected for a layer, QuarkXPress Server does not render that layer when producing PDF, EPS, or PostScript files.

InteractiveDesigner Server XTensions software

InteractiveDesigner XTensions software allows QuarkXPress Server to render both Print and Interactive layouts in QuarkXPress projects as SWF (Flash) files. If you render a Print layout as an SWF file, you can use the right and left arrow keys to navigate between pages in the SWF file.

To render an Interactive layout in a QuarkXPress project as a SWF file when SWF is not the QuarkXPress Server default render type, use the SWF namespace, as follows:

```
http://[server]:[port]/swf/[projectname]?layout="[layoutname]"
```

App Studio XTensions software

The App Studio XTensions modules allow QuarkXPress Server to render Print and App Studio layouts in QuarkXPress projects as AVE issue files.

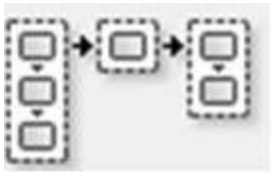
In QuarkXPress 9.5.1.1, an option was added to allow the App Studio output to convert sections to page stacks. A new URL parameter convertsectionstopagestacks has been added to provide this ability.

The XML example below demonstrates how to apply sections using the ModifierXML:

```
<PAGESEQUENCE MASTERREFERENCE="APSMasterPages"
FORCEPAGECOUNT="NOFORCE">
SECTIONNUMBERFORMAT FORMAT="NUMERIC"
INITIALPAGENUMBER="1"/>
```

The element SECTIONNUMBERFORMAT creates a section start which applied to all pages in the page sequence. Each PAGESEQUENCE could have its own SECTIONNUMBERFORMAT.

If the modify request is sent to the server and each PAGESEQUENCE in the ModifyXML has its own SECTIONNUMBERFORMAT, creating sections, and the URL parameter contains convertsectionstopagestacks=true, the digital issue navigation would be as follows:



If the URL parameter contains convertsectionstopagestacks=false, the digital issue navigation would be as follows:

➡ By default, convertsectionstopagestacks is **false.**

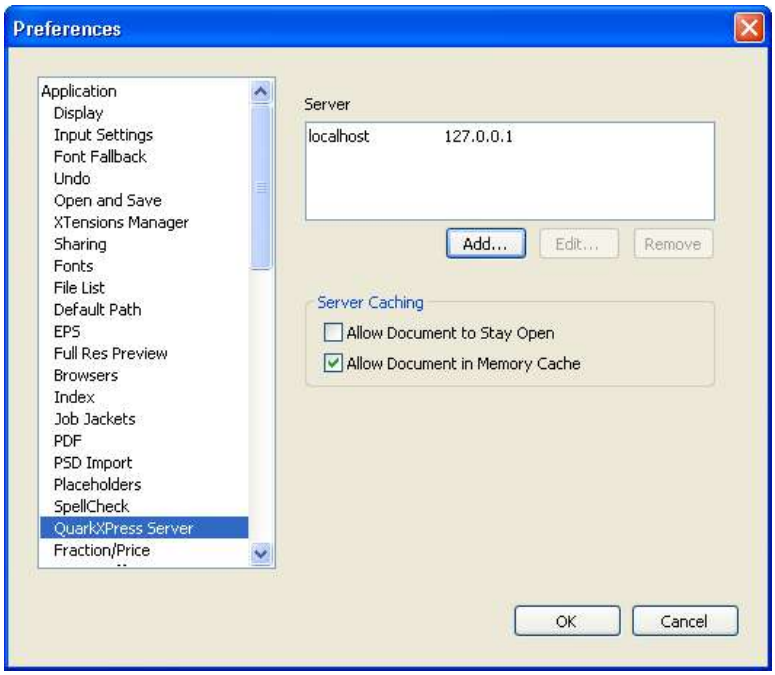
Telegraph XTensions software

Unlike the other XTensions software described in this guide, Telegraph XTensions software works with QuarkXPress, rather than with QuarkXPress Server. You can use Telegraph XTensions software to create QuarkXPress projects that can serve as templates in QuarkXPress Server. Using Telegraph XTensions software, you can assign unique names to individual items, define server caching parameters, and upload the template directly to a QuarkXPress Server computer.

➡ These topics explain how to use Telegraph XTensions software. It is assumed that you are already familiar with the functionality and user interface of QuarkXPress.

Setting Telegraph preferences

Telegraph XTensions software adds the **QuarkXPress Server** pane to the QuarkXPress **Preferences** dialog box (**QuarkXPress/Edit** menu). You can use this pane to configure settings for QuarkXPress Server templates, specify where to store your projects on the server, and control how projects are cached.



QuarkXPress Server pane of Preferences dialog box (QuarkXPress/Edit menu)

To edit the properties of a server in the **Server** list, select the server name and then click **Edit**.

To remove a server from the **Server Setup** list, select the server and then click **Remove**.

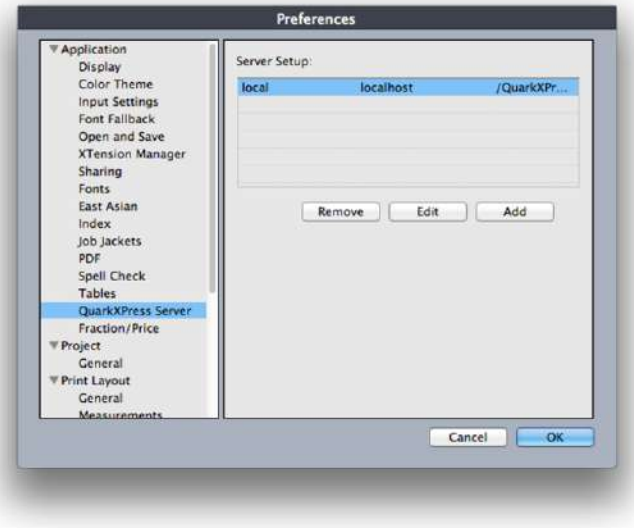
To allow projects checked in from this copy of QuarkXPress to remain open on the server after it has been served, check **Allow Document to Stay Open**.

To load projects checked in from this copy of QuarkXPress into the server memory cache, check **Allow Document in Memory Cache**.

Specifying a server for template upload

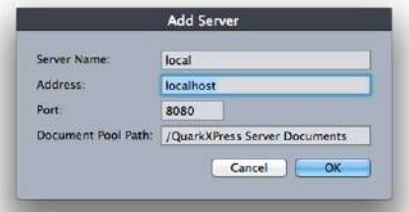
Before you can upload a template to a server with Telegraph XTensions software, you must add that server to the QuarkXPress preferences. To do so:

1. Choose **QuarkXPress/Edit > Preferences**. The **Preferences** dialog box displays.
2. Click **QuarkXPress Server** in the list on the left. The **QuarkXPress Server** pane displays.



QuarkXPress Server pane of Preferences dialog box

3. Click **Add**. The **Add Server** dialog box displays.



Add Server dialog box

4. Enter a human-readable name for the server in the **Server Name** field.
5. Enter the server's name or IP address in the **Address** field.

6 Enter the server's port number in the **Port** field. The default port number is 8080. Valid values are from 1 to 65535.

7 In the **Document Pool Path** field, enter the path to the document pool directory on the server, or to a subdirectory within the document pool. If you leave this field blank, the path defaults to the document pool directory path specified in the **QuarkXPress Server Document Root** field (**QuarkXPress Server > Server Configuration > Server** tab).

➡ If you enter a folder path that does not exist, QuarkXPress Server can create the folders in the path when you upload the template to a QuarkXPress Server. To create folders when you upload, check **Generate Hierarchy On Document Upload** in the **Server** tab of the QuarkXPress Server **Server Configuration** dialog box **(QuarkXPress Server > Server Configuration)** before you upload the template to the server.

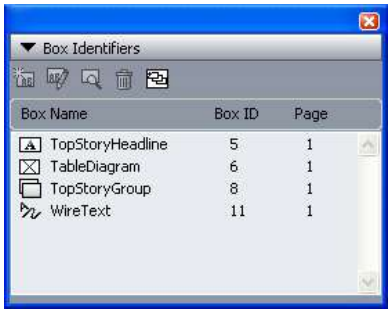
8 Click **OK**, then close the **Preferences** dialog box.

Using Telegraph XTensions software

Once you have configured preferences for Telegraph QuarkXTensions software, you can begin creating QuarkXPress Server templates. After you complete a template, Telegraph XTensions software can upload the file to a QuarkXPress Server computer.

Identifying QuarkXPress items and groups

The **Box Identifiers** palette lets you associate names with items and groups. To display this palette, choose **Window > Box Identifiers**.



Box Identifiers palette

To edit the name of an item or group, select its name in the **Box Identifiers** palette and then click **Edit Box Name**.

To scroll to the location of a named item or group, double-click the target item or group's name in the **Box Identifiers** palette. Alternatively, you can select the name and click **Go To**.

To delete an item's or group's name (without deleting the item or group itself), select the name in the **Box Identifiers** palette and then click **Delete**.

To make sure that all named boxes display in the **Box Identifiers** palette, click **Populate Named Boxes**.

Each QuarkXPress item also has an identification number that you can use when you want to render individual project items in QuarkXPress Server. The number displays in the **Box ID** field in the upper right corner of the palette.

➡ In a chain of text boxes, all boxes use the same **Box Name**. However, each box has a unique **Box ID** number.

Naming items and groups

To specify a name for an item or group:

1. Using the Item tool, select the target item or group.
2. In the **Box Identifiers** palette, click **New Box Name**. The **New Box Name** dialog box displays.

New Box Name dialog box

3. Enter a name for the item or group in the **Box Name** field.
4. Click **OK**. The name of the item or group displays on the **Box Identifiers** palette, next to the item's item ID and page number.

➡ In addition to items on layout pages, you can also name items on master pages. Items on layout pages that are based on items on master pages have a default name of *"<Item name on master page><New box UID>"*

Uploading templates

Telegraph QuarkXTensions software can upload a project to the QuarkXPress Server computer in one simple step. You can also upload any required pictures and fonts, if you choose to collect the fonts and pictures during the upload.

➡ QuarkXPress Server automatically generates any file hierarchy necessary when you upload content to the document pool from Telegraph XTensions software.

➡ You must have Telegraph XTensions software 10.0 or later to upload templates to QuarkXPress Server 9.0 or later. (You can use previous versions

of Telegraph with previous versions of QuarkXPress, but if you do so you must upload projects to the server using the document pool upload capability in the QuarkXPress Server Web interface.)

➡ This feature does not upload assets that are used in App Studio AVE interactivity. You must upload such assets manually.

To upload the active project:

1 Choose **Utilities > Upload Template**. The **Upload Template** dialog box displays.

➡ If you have edited the project since you last saved it, QuarkXPress prompts you to save the project.

Upload Template dialog box

2 Choose a server from the **Server** drop-down menu. This drop-down menu includes the servers listed in the **QuarkXPress Server** pane of the **Preferences** dialog box (**QuarkXPress/Edit** menu).

3 If you specified a directory path for the server, that path automatically displays in the **Directory Path** field. If you did not specify a directory path in the preferences, this field remains blank. This path defaults to the document pool directory specified in QuarkXPress Server.

4 If you are uploading to QuarkXPress Server, and you have configured that server to require authentication, enter a valid user name and password in the **User Name** and **Password** fields. (If the QuarkXPress Server application does not require authentication, leave these fields empty.)

➡ To specify authentication information for a QuarkXPress Server application, choose **Administration > Preferences > General** in the QuarkXPress Server Web interface, check **Authenticate for Admin Requests** box and enter a user name and password.

5 To indicate which protocol to use for uploading, click **HTTP** or **HTTPS**.

6 Enter the port for QuarkXPress Server in the **Port** field.

7 To collect and upload fonts used by the project, check **Fonts**.

8 To collect and upload pictures used by the project, check **Pictures**. This will upload high- or low-resolution pictures that are linked to or embedded in the project.

➡ If picture files are missing or have been modified since they were imported into the project, an alert displays. For more information about missing or modified picture files, see "*Uploading Missing or Modified Pictures*."

9 Click **OK**.

➡ If you check **Fonts**, an alert reminds you of possible restrictions regarding copying font software. Click **OK** to continue uploading the project with the fonts, click **Do Not Collect Fonts** to upload the project without the fonts, or click **Cancel** to stop the upload.

The **Upload Status** window displays a progress bar that displays the status of the upload. When the upload is complete, a message notifies you whether the project uploaded successfully.

Uploading missing or modified pictures

If the picture files linked to the project are missing or have been modified since they were imported into the project, an alert displays at upload. Choose from among the following options:

- To continue the upload with low-resolution versions of the pictures, click **OK**.
- To stop the upload, click **Cancel**.
- To locate missing pictures or update modified pictures, click **List Pictures**.

If you click **List Pictures**, the **Missing/Modified Pictures** dialog box displays:

- To view a picture in the project, select the picture's name in the list and click **Show**.
- To locate a missing picture file, select it and click **Update**. The **Find** dialog box displays. Locate and choose the appropriate file, and then click **Open**.
- To update a modified picture file, click **Update**. Every instance of the modified picture in the project is updated.
- When **OK** displays in the **Status** column for each picture, click **Collect**. If the status of any picture is still **Missing** or **Modified** when you click **Collect**, that picture file will not be uploaded, but a low-resolution preview will remain in the project.
- To stop the upload and return to the project window, click **Cancel**.

QuarkXPress Server Manager

QuarkXPress Server Manager is a server application that efficiently routes rendering requests in an environment that uses one or more QuarkXPress Server applications. QuarkXPress Server Manager uses load-balancing methods to determine which server in the QuarkXPress Server pool can best process a document request, and uses caching to improve speed and efficiency. QuarkXPress Server Manager also provides failsafe capability by reliably resubmitting failed requests, either to the same QuarkXPress Server instance or to a different one (depending on the error message returned by the server instance).

QuarkXPress Server does not require QuarkXPress Server Manager, but a QuarkXPress Server Manager server can coordinate multiple QuarkXPress Server applications so that they work together with maximum speed, reliability, and availability.

QuarkXPress Server Manager also provides a Web services interface that allows developers to use QuarkXPress Server features without having to use the HTTP interface.

To configure a QuarkXPress Server Manager server application, you must use the QuarkXPress Server Manager Web client. The topics below explain how the QuarkXPress Server Manager Web client works and provide examples for using it.

Understanding QuarkXPress Server Manager

Before you begin, take time to review the topics below so that you understand how this chapter is structured and how you can get the most out of it.

Load balancing

Load balancing ensures that each rendering request is sent to a server that is likely to be able to handle it quickly. QuarkXPress Server Manager lets you use choose from three load-balancing settings:

- **Dynamic Load Balancer**: The QuarkXPress Server Manager server considers file size and throughput requirements for each request. For example, assume the following series of requests is sent to QuarkXPress Server Manager in an environment that uses two QuarkXPress Server instances:

Request	Size
1	8MB

Request	Size
2	1MB
3	2MB

- **Random Load Balancer**: Each rendering request is sent to a random server.
- **Round-robin Load Balancer**: Requests are sent to servers in a set order. For example, if you have three QuarkXPress Server instances and QuarkXPress Server Manager receives ten rendering requests, the requests are distributed as follows:

Request	QuarkXPress Server instance used
1	1
2	2
3	3
4	1
5	2
6	3
7	1

The first request is assigned to server #1, and the second request is assigned to server #2. When the third request arrives, QuarkXPress Server Manager checks the loads that the two servers are already handling and assigns the task to the server with the smallest load — in this case, server #2.

If a request fails because a server stops responding or because of a "File Not Found" error, QuarkXPress Server Manager does not resubmit that request to that server.

➡ The **Dynamic** setting is typically the most efficient setting for environments with more than one QuarkXPress Server instance.

For information about choosing a load-balancing setting, see "*Controlling load balancing*."

Request timeout interval

QuarkXPress Server Manager attempts to send each request to a QuarkXPress Server instance that can promptly handle that request. However, in some situations a QuarkXPress Server instance might be unable to process a request in a reasonable amount of time (for example, if the server is working on a large rendering job, or if the server computer has stopped functioning). If you specify a certain period of time as the request timeout interval, QuarkXPress Server Manager will wait for the response until that period of time elapses, and then

send the request to a different QuarkXPress Server instance. This ensures that a request does not get "lost" if its assigned QuarkXPress Server instance does not become available promptly.

For information about setting a request timeout interval, see "*Using other global settings*."

➡ If a client request fails despite being sent to multiple QuarkXPress Server instances, QuarkXPress Server sends the end user a customizable error message or exception so that the end user can appropriately handle the failure. A QuarkXPress Server Manager server can also automatically send e-mail to an administrator in the event of a problem; for more information, see "*Generating automatic e-mail messages*."

Determining QuarkXPress Server instance availability

QuarkXPress Server Manager uses two methods to determine the availability of a QuarkXPress Server instance: *ping* and *ping document*.

Ping

QuarkXPress Server Manager periodically sends a ping request to all of its QuarkXPress Server instances to determine whether they are available. Ping requests use the following format:

http://[Server]:[Port]/getprocessid

Ping document

QuarkXPress Server Manager periodically sends a render request to all of its QuarkXPress Server instances to determine whether they can render a document. Ping document requests use the following format:

http://[Server]:[Port]/[PingDocumentName.qxp]

You can specify the document to be used for this render request. To avoid long ping document rendering times, use a simple document.

You can specify the interval between pings and ping documents in the **Other Settings** pane in the **Global Settings** pane of the **QuarkXPress Server Manager** window (see "*Using other global settings*").

Logging with QXP Server Manager

QuarkXPress Server Manager logs all interactions with QuarkXPress Server instances. QuarkXPress Server Manager log files contain the following information:

- Render requests

- QuarkXPress Server responses
- Information about events (such as alerts) that occur during the render-request process
- Details about requests that were sent to a different QuarkXPress Server instance after the first assigned QuarkXPress Server instance was unable to process the request

Within the logs, each QuarkXPress Server instance is identified by its IP address and port number.

You can export log files in XML (Extensible Markup Language) or comma-separated values (CSV) format.

Caching

To increase speed and efficiency, QuarkXPress Server Manager caches information in memory. If the response to a render request, URL request, or file request is included in the QuarkXPress Server Manager memory cache, QuarkXPress Server Manager returns the response from the disk cache instead of sending the request to a QuarkXPress Server instance. For more information, see "*Managing the cache*."

- Requests that contain a binary parameter and multipart responses are not cached, regardless of whether global caching or command-specific caching is enabled.
- When the QuarkXPress Server Manager server application receives response data from a QuarkXPress Server instance, QuarkXPress Server Manager can return that response directly or write it as a file and return the file's URL. The second approach maximizes efficiency for SOAP-based clients, because SOAP transfers binary data very slowly. Cached response files have names that begin with "TMP_", and they are removed when they reach the age specified in the cache settings (see "*Managing the cache*"). QuarkXPress Server Manager uses the cache file this way regardless of whether caching is turned on or off; however, you can override this behavior by setting the responseasurl parameter to false for every request.

Web services

QuarkXPress Server Manager provides a Web services interface that makes it easy for developers to create applications that use QuarkXPress Server. This Web services interface provides the same functionality that is available through the QuarkXPress Server HTTP interface.

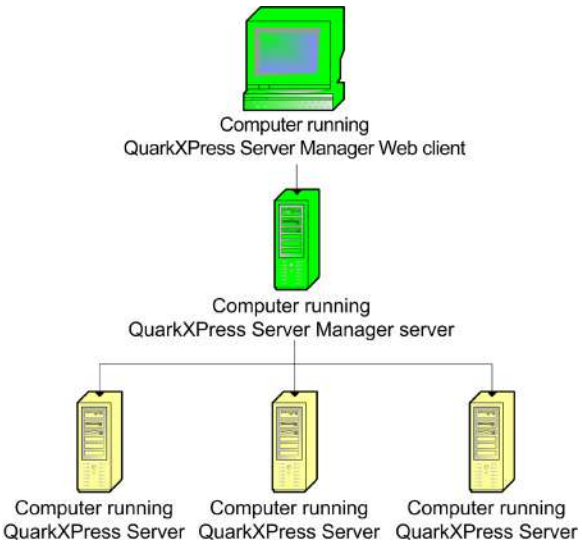
Working with QuarkXPress Server Manager

The general process for using QuarkXPress Server Manager is as follows:

1. Install QuarkXPress Server Manager software.
2. Launch one or more QuarkXPress Server instances on your network.
3. Launch the QuarkXPress Server Manager Console server application (see *Starting the Manager server application*").
4. Launch the QuarkXPress Server Manager Web client (see *Starting the Manager Web client*").
5. Use the **Manage Servers** pane to add QuarkXPress Server instances, specify information about those servers (see "*Configuring QuarkXPress Server instances*"), and choose a load-balancing method (see "*Controlling load balancing*").
6. Configure proxy server settings, automatic e-mail settings, and various other settings in the **Global Settings** pane (see "*Using a proxy server*," "*Generating automatic e-mail messages*," and "*Using other global settings*").
7. As necessary, delete cache items and clear the QuarkXPress Server Manager server cache using the **Manage Cache** pane (see "*Managing the cache*").

Starting QuarkXPress Server Manager

Each QuarkXPress Server Manager server can handle multiple QuarkXPress Server instances.



QuarkXPress Server Manager diagram

Starting the Manager server application

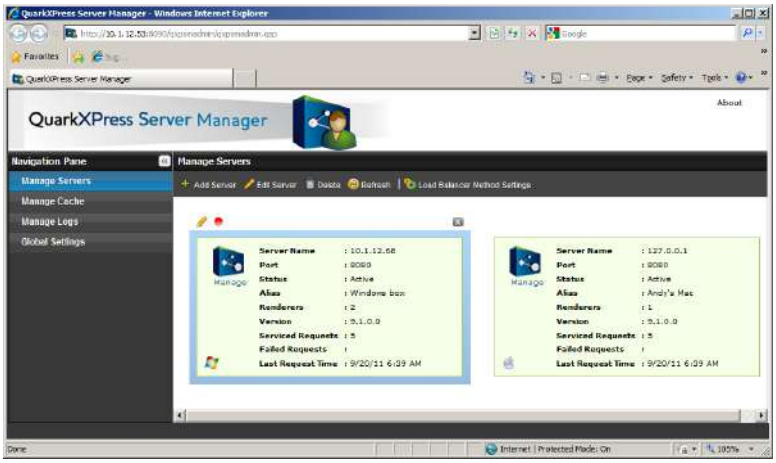
To launch the QuarkXPress Server Manager server application:

- macOS: Open the QuarkXPress Server Manager/Server folder inside the applications folder and double-click "QXPSMServerStart.command."
- *Windows (if you have not installed QuarkXPress Server Manager as a service)*: Choose **Start > Programs > QuarkXPress Server Manager 9 > Start QuarkXPress Server Manager**. Alternatively, open the "Server" folder in the QuarkXPress Server Manager application folder and double-click the "QXPSMServerStart.bat" file as an administrator.

You can access API documentation in HTML format by navigating to http://[server]:[port], where [server] identifies the computer on which QuarkXPress Server Manager is running and [port] identifies the port it is running on.

Starting the Manager Web client

To launch the QuarkXPress Server Manager Web application, click **Open Admin Client** on the QuarkXPress Server Manager home page. The QuarkXPress Server Manager administration Web client displays.



The QuarkXPress Server Manager Web client

Request handler binding

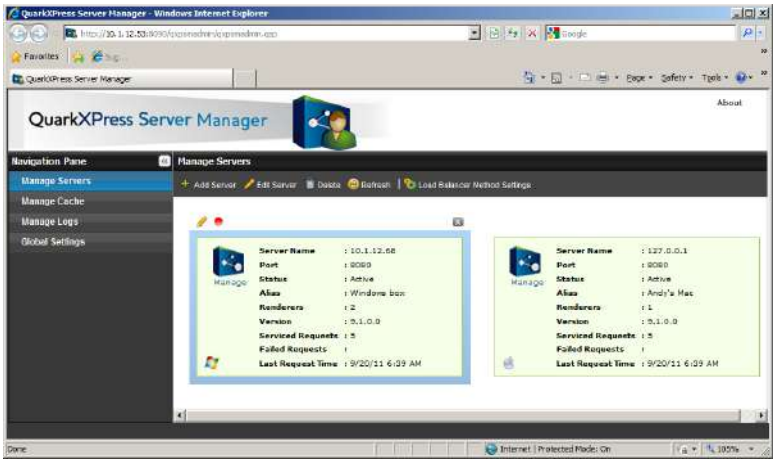
When you add a QuarkXPress Server instance to QuarkXPress Server Manager, you can choose to bind that server instance to particular rendering type or request type. When QuarkXPress Server Manager receives a matching request, it will send the request to only those server instances that are bound to that rendering type or request type. If multiple server instances are bound to a particular rendering type or request type, QuarkXPress Server Manager balances the load of such requests between the server instances.

An instance of QuarkXPress Server Manager that is not bound to any specific type of request is called a *generic server*. If a request is not bound to a particular server instance, QuarkXPress Server Manager sends that request to a generic server. If more than one generic server is available, QuarkXPress Server Manager balances the load of such requests between them.

You can bind a server to more than one rendering type or request type.

Configuring QuarkXPress Server instances

The **Manage Servers** pane lists the QuarkXPress Server instances the QuarkXPress Server Manager server is handling. You can use this pane to add QuarkXPress Server instances, edit the description of existing QuarkXPress Server instances, delete QuarkXPress Server instances, and choose a load balancing method.



Manage Servers pane

- To configure an individual QuarkXPress Server instance, click **Manage** under the icon for that instance. The QuarkXPress Server Web interface for that instance displays.

Adding and enabling a QuarkXPress Server instance

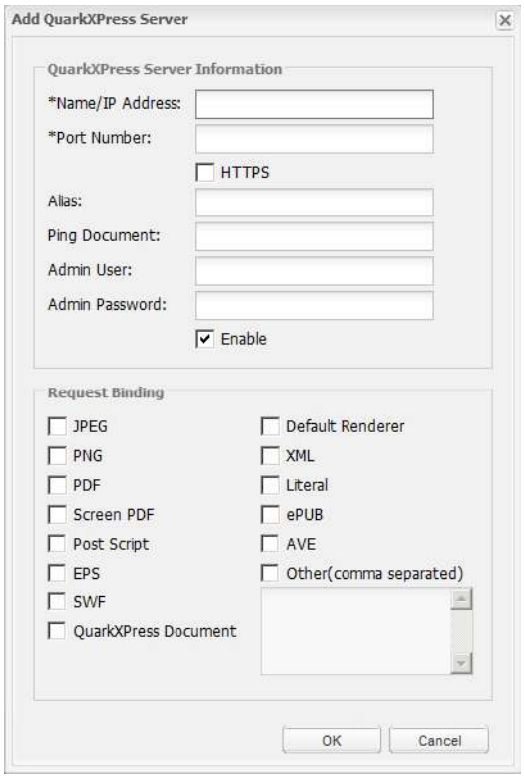
When you add and enable a QuarkXPress Server instance in the **Manage Servers** pane of the **QuarkXPress Server Manager** interface, the QuarkXPress Server Manager server begins routing rendering requests to that QuarkXPress Server instance.

- If you add and enable a QuarkXPress Server instance in this pane, be sure that clients are no longer sending rendering requests directly to that QuarkXPress Server instance; otherwise, the server will be handling both direct requests and routed requests, and the server might become overly busy. Note also that requests sent directly to such a QuarkXPress Server

instance do not benefit from QuarkXPress Server Manager features such as load balancing, caching, and logging.

To add and enable a QuarkXPress Server instance:

1. Display the **Manage Servers** pane of the **QuarkXPress Server Manager** interface.
2. Click **Add Server**. The **Add QuarkXPress Server** dialog box displays.



Add QuarkXPress Server dialog box

3. Enter the QuarkXPress Server instance's DNS name or IP address in the **Name/IP Address** field.
4. Enter the QuarkXPress Server instance's port number in the **Port Number** field.
5. If the QuarkXPress Server instance is running with the HTTPS protocol, check **HTTPS**.
6. To specify an alternate name for the server, enter a value in the **Alias** field. The **Alias** value displays in the **Manage Servers** pane of the **QuarkXPress Server Manager** interface.
7. If you choose to use a particular ping document for this server (see "*Ping document*"), make sure the project file is in the QuarkXPress Server instance's document pool and then enter the project's file name in the **Ping Document** field. The ping document is used only if the global **Ping Type** is set to **Ping Document** (see "*Using other global settings*"). Note that if you do not

set a ping document here, and no global ping document is set (see "*Using other global settings*"), an error message might display to indicate that the server is registered but inactive.

8. Enter the QuarkXPress Server instance user name and password in the **Admin User** and **Admin Password** fields.
9. To specify that QuarkXPress Server Manager should begin sending rendering requests to this QuarkXPress Server instance, check **Enable**.
10. To restrict this server to one or more particular types of rendering, check the appropriate boxes in the **Request Binding** area. To add additional render types (for example, render types provided by QuarkXPress Server XTensions software), check **Other** and enter the appropriate namespaces in the corresponding field as a comma-separated list. For more information, see "*Request handler binding*."
11. Click **OK**.

Editing a QuarkXPress Server instance

To edit the description of a QuarkXPress Server instance, display the **Manage Servers** pane of the **QuarkXPress Server Manager** interface, select the server in the list, and then click **Edit Server**. You can also display the **Edit QuarkXPress Server** dialog box by hovering the mouse cursor over the server and then clicking the **Edit Server** button on the upper left.

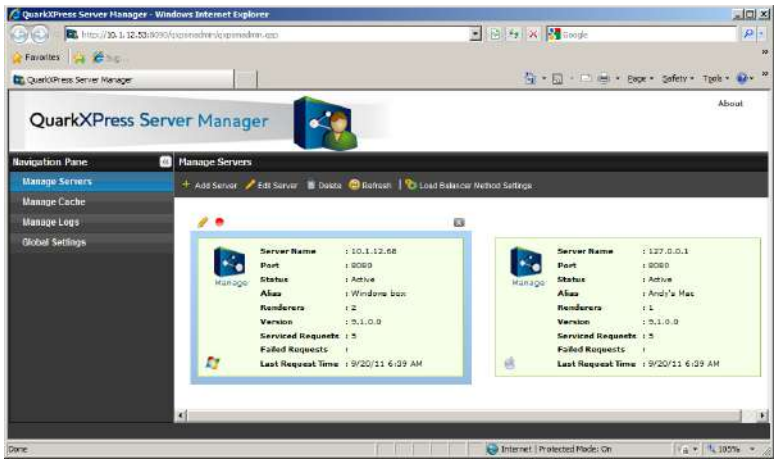
The options in the **Edit QuarkXPress Server** dialog box are the same as the options in the **Add QuarkXPress Server** dialog box (see "*Adding and enabling a QuarkXPress Server instance*").

Enabling and disabling routing to QuarkXPress Server instances

The **Status** field for each server in the **Manage Servers** pane of the **QuarkXPress Server Manager** interface shows the status of each QuarkXPress Server instance.

➡ The status fields are not updated automatically. To update the **Status** field for all servers, click **Refresh**.

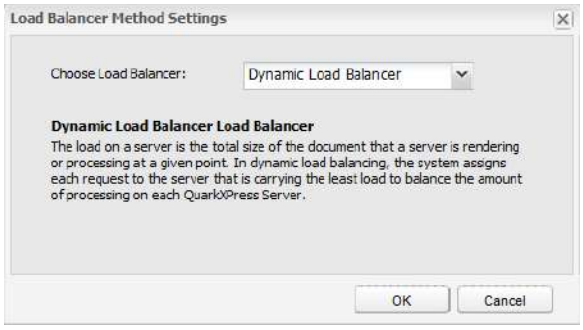
To enable or disable routing to a QuarkXPress Server instance, select the server, click **Edit Server** to display the **Edit QuarkXPress Server** dialog box, check or uncheck **Enable**, and then click **OK**. You can also enable or disable a QuarkXPress Server instance by hovering the mouse cursor over the server and then clicking the **Enable the Server** button or **Disable the Server** button on the upper left.



Manage Servers pane

Controlling load balancing

Load balancing ensures that each rendering request is sent to a QuarkXPress Server instance that is most likely to be able to handle it quickly. To define a load-balancing setting for the QuarkXPress Server Manager server, display the **Manage Servers** pane of the **QuarkXPress Server Manager** interface and click **Load Balancer Method Settings**. The **Load Balancer Method Settings** dialog box displays.



Load Balancer Method Settings dialog box

QuarkXPress Server Manager lets you use choose from three load-balancing settings:

- **Dynamic Load Balancer**: Sends requests to servers based on file size
- **Random Load Balancer**: Sends each rendering request to a random server
- **Round-robin Load Balancer**: Sends requests to servers in a set order

For more information about these load-balancing settings, see "*Load balancing.*"

Deleting a QuarkXPress Server instance

To delete a QuarkXPress Server instance from the list of available servers in the **QuarkXPress Server Manager** interface, display the **Manage Servers** pane, select

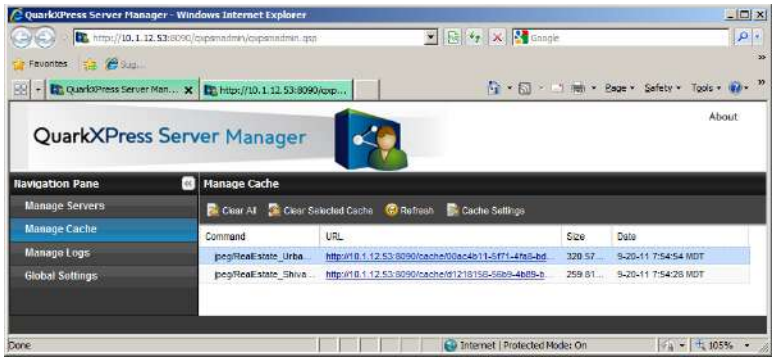
the server name or IP address, and then click **Delete**. You can also delete a server by hovering the mouse cursor over the server and then clicking the **Delete the Server** button on the upper left.

- Deleting a QuarkXPress Server instance from this dialog box does not shut down the QuarkXPress Server instance, but it does prevent the QuarkXPress Server Manager server from routing rendering requests to the QuarkXPress Server instance.

Managing the cache

Each QuarkXPress Server Manager server has an in-memory cache (in which it stores the keys to recently accessed items) and a disk-based cache (in which the items themselves are stored). If a request for a recently used item arrives, and a QuarkXPress Server Manager server has that request in its memory cache, the server can simply return the response from its disk cache instead of having to send the request to a QuarkXPress Server instance.

To manage the QuarkXPress Server Manager cache, display the **Manage Cache** pane of the **QuarkXPress Server Manager** interface.



Manage Cache pane

- Requests are stored in the cache only if the cache is turned on. For more information, see "*Configuring cache options*."
- The cache stores only the results of requests that do not deliberately bypass the cache.

Viewing a QuarkXPress Server Manager server cache

To view QuarkXPress Server Manager cache information, display the **Manage Cache** pane of the **QuarkXPress Server Manager** interface. For each file in the cache, this pane lists the command, URL, size, and time and date it was generated. To view a cache file, double-click the file name in the list.

Deleting files from the cache

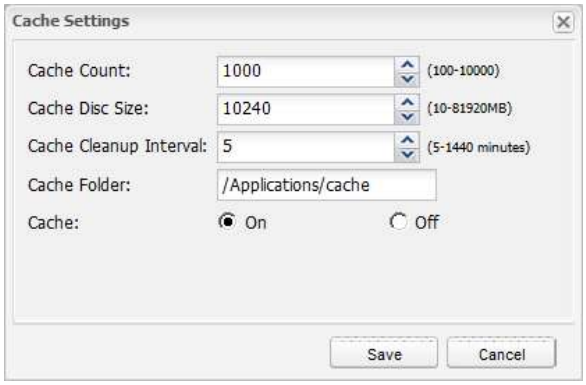
There is usually no need to manually delete files from a QuarkXPress Server Manager server's cache. When the cache reaches 95% of its capacity, QuarkXPress Server Manager automatically begins deleting the least recently used files in the cache to make room for new files. However, you can also manually clear files from the cache.

To manually delete cache files:

1. If you want to delete specific files, select those files in the list.
2. Click **Clear Selected Cache**. The **Clear Cache** alert displays.
3. Click **OK**.

Configuring cache options

To configure cache options, display the **Manage Cache** pane in the **QuarkXPress Server Manager** interface, then click **Cache Settings**. The **Cache Settings** dialog box displays.



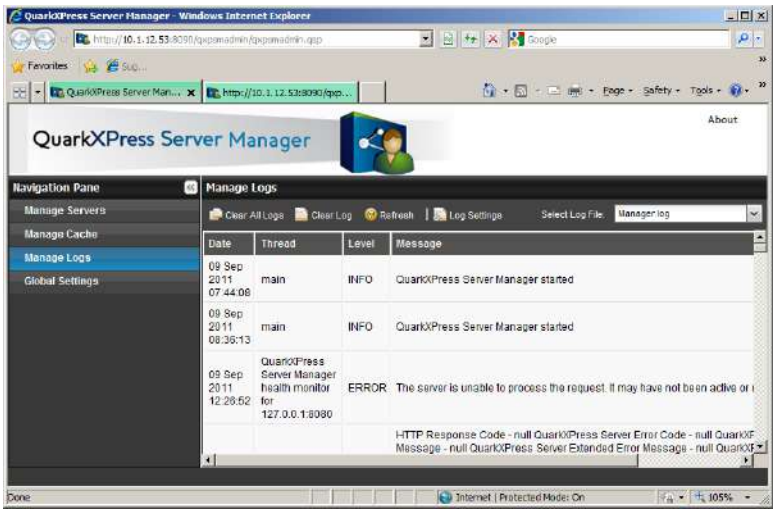
Manage Cache pane

- To set the maximum number of files allowed in the cache, enter a value in the **Cache Count** field. When the number of files in the cache reaches the number you set here, QuarkXPress Server Manager begins deleting the least recently used files to make room in the cache.
- To set the maximum disk cache size, enter a value in the **Cache Disk Size** field. When the disk cache reaches this size, QuarkXPress Server Manager begins deleting the least recently used files to make room in the cache.
- To specify an interval after which the cache should be periodically cleared, enter a value in the **Cache Cleanup Interval** field.
- To specify where cache files for the QuarkXPress Server Manager server should be stored, enter a path in the **Cache Folder** field.
- The **Cache** radio buttons let you control caching for the QuarkXPress Server Manager server. To turn caching on, click **On**. To turn caching off, click **Off**.

Managing logs

A QuarkXPress Server Manager server maintains logs of all of the requests it receives, the responses from the QuarkXPress Server instances, information about events (such as alerts) that occur during the render-request process, dates and times, and details about each request that was sent to a different QuarkXPress Server instance after its first assigned QuarkXPress Server instance was unable to process the request.

➡ To control what information is stored in the logs, use the **Manage Logs** pane of the **QuarkXPress Server Manager** interface. For more information, see "*Configuring logging options*."



Manage Logs pane

Viewing log file details

To view information about a specific log file, display the **Manage Logs** pane of the **QuarkXPress Server Manager** interface, then choose the log file name from the **Select Log File** drop-down menu.

Deleting log files

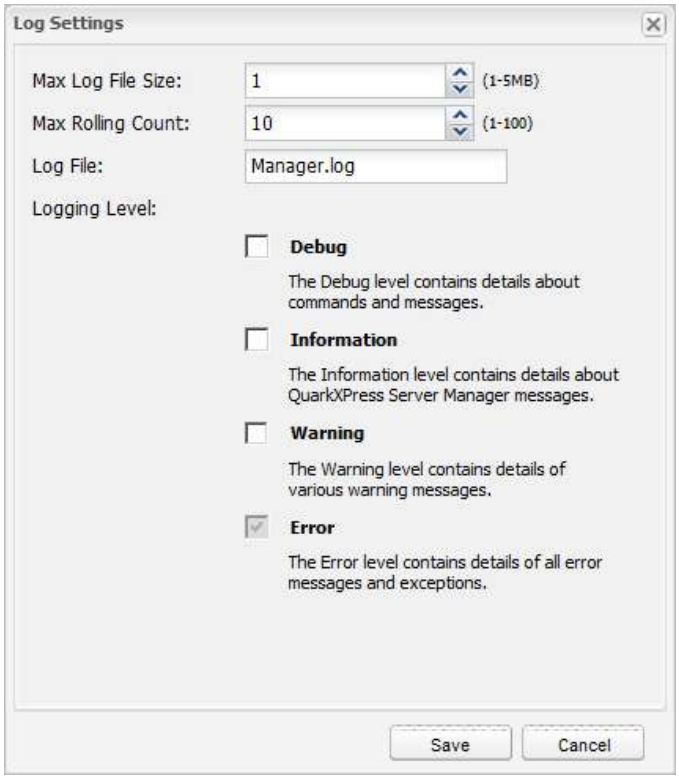
To delete a log file:

1. Display the **Manage Logs** pane of the **QuarkXPress Server Manager** interface.
2. Choose the target log file from the **Select Log File** drop-down menu.
3. Click **Clear Log**.

➡ To clear all log files, click **Clear All Logs**.

Configuring logging options

To configure logging options, click **Log Settings** pane in the **Manage Logs** pane of the **QuarkXPress Server Manager** interface. The **Log Settings** dialog box displays.



Log Settings dialog box

To set the maximum log file size, enter a value in the **Max Log FileSize** field. When a log file reaches this size, the current log file is closed and a new log file is created.

To specify the maximum number of log files to keep, enter a value in the **Max Rolling Count** field. When the number of log files reaches this limit, QuarkXPress Server Manager deletes the oldest log file each time a new log file is created.

To specify the root name of the log file for the QuarkXPress Server Manager server, enter that name in the **Log File** field. To place the log file in a particular directory, precede the file name with an absolute path. QuarkXPress Server Manager appends numbers to this name to create consecutively named log files.

To control how much information is stored in the log files, check a box in the **Logging Level** area:

- **Debug**: Stores information such as the commands executed and the servers where those commands are executed. Also stores all of the information that is stored when **Information, Warning**, and **Error** are checked.

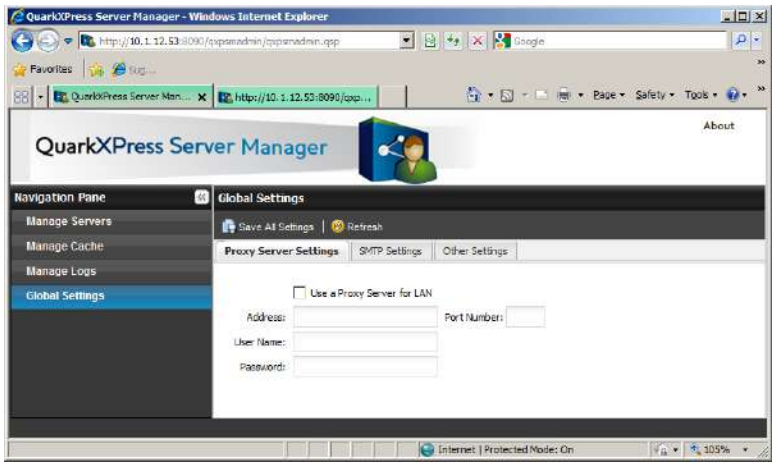
- **Information**: Stores informational messages such as startup messages and command-retry messages. Also stores all of the information that is stored when **Warning** and **Error** are checked.
- **Warning**: Stores warning messages. Also stores all of the information that is stored when **Error** is checked.
- **Error**: Stores error messages and stack traces for exceptions.

➡ The **Debug** and **Information** settings produce large logs that grow rapidly, so you might want to use these settings for troubleshooting only.

Using a proxy server

Some networks route network traffic through a proxy server for reasons of efficiency or security. To use a proxy server for all requests and responses between QuarkXPress Server Manager and QuarkXPress Server:

1 Display the **Proxy Server Settings** tab of the **Global Settings** pane of the **QuarkXPress Server Manager** interface.



Proxy Server Settings tab

2 Check **Use a Proxy Server for LAN**.

3 Enter the proxy server's DNS name or IP address in the **Address** field.

4 Enter the proxy server's port number in the **Port Number** field.

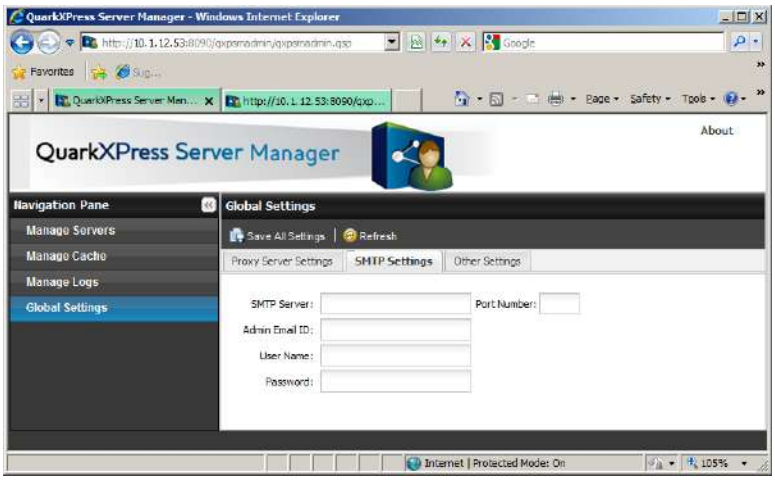
5 Enter the proxy server's user name in the **User Name** field.

6 Enter the proxy server's password in the **Password** field.

Generating automatic e-mail messages

You can configure a QuarkXPress Server Manager server to automatically generate and send e-mail messages if particular events occur. To configure the QuarkXPress Server Manager server to automatically send e-mail messages:

1. Display the **SMTP Settings** tab of the **Global Settings** pane of the **QuarkXPress Server Manager** interface.
2. Enter a valid SMTP server name or IP address in the **SMTP Server** field and then enter the corresponding port number in the **Port Number** field.
3. Enter the e-mail address to which messages should be sent in the **Admin Email ID** field.
4. If this SMTP server requires validation, enter a valid user name in the **User Name** field and a valid password in the **Password** field.



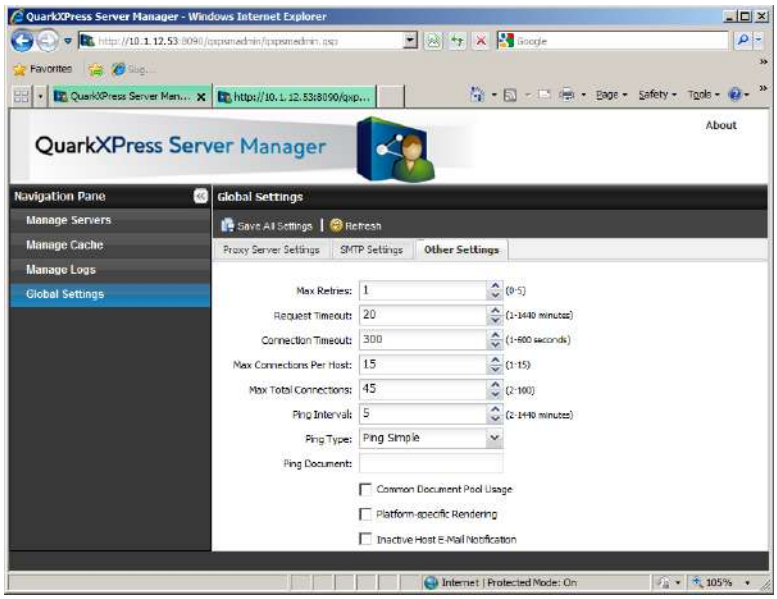
SMTP Settings pane

Two events can cause QuarkXPress Server Manager to generate an e-mail message:

- If **Inactive Host E-mail Notification** is checked (see "*Using other global settings*"), a message is sent when a QuarkXPress Server instance goes from the active state to the inactive state.
- If you have set up custom error messages (as described in "*Using custom error messages*"), certain QuarkXPress Server errors result in e-mail messages.

Using other global settings

To configure other global settings, display the **Other Settings** tab of the **Global Settings** pane of the **QuarkXPress Server Manager** user interface.



Other Settings tab of Global Settings page

- **Max Retries**: To specify the maximum number of times the QuarkXPress Server Manager server should submit a request to the QuarkXPress Server instances, enter a value in this field. When QuarkXPress Manager has unsuccessfully submitted a request this many times, the application returns an error message.
- **Request Timeout**: To specify the maximum number of minutes the QuarkXPress Server Manager server should wait for a response from a QuarkXPress Server instance, enter a value in this field. When this time has elapsed, the QuarkXPress Server Manager server retries the request (unless the **Max Retries** value has been reached).
- **Connection Timeout**: To specify the maximum number of seconds the QuarkXPress Server Manager server should spend attempting to establish a connection with a particular QuarkXPress Server instance, enter a value in this field.
- **Max Connections Per Host**: To specify the maximum number of connections the QuarkXPress Server Manager server should open for a particular QuarkXPress Server instance before it begins queuing requests, enter a value in this field.
- **Max Total Connections**: To specify the maximum number of connections the QuarkXPress Server Manager server should open for all hosts before it begins queuing requests, enter a value in this field.
- **Ping Interval**: To set the amount of time the QuarkXPress Server Manager server should wait between ping attempts (see "*Determining QuarkXPress Server instance availability*"), enter a value in this field.

- **Ping Type**: To indicate whether QuarkXPress Server Manager should use a simple ping or a ping document to test a QuarkXPress Server instance, choose an option from this drop-down menu.
- **Ping Document**: To indicate which QuarkXPress project the QuarkXPress Server Manager server should use for ping document requests, enter the project file name in this field. Make sure a copy of the project file is in each QuarkXPress Server instance's document pool. Note that individual QuarkXPress Server instances can override this value by providing another document name. Note also that a ping document is used only if **Ping Type** is set to **Ping Document**.
- **Common Document Pool Usage**: Check this box if all managed QuarkXPress Server instances are using the same document pool. If this box is checked, then upload, delete, and saveas requests are sent to one of the available servers. If this box is unchecked, then upload, delete, and saveas requests are sent to all managed servers. Note that you must manually set each server to point at the common document pool.
- **Platform-specific Rendering**: A QuarkXPress Server Manager server can send requests that involve macOS project files to a macOS-based QuarkXPress Server instance, and send requests that involve Windows project files to a Windows-based QuarkXPress Server instance. Setting up the server in this manner can be desirable if the project files involved use fonts that are available on only one platform or the other. To enable platform-specific rendering for the QuarkXPress Server Manager server, check this box.
- **Inactive Host E-mail Notification**: To automatically generate an e-mail message when a QuarkXPress Server instance becomes inactive, check this box. The e-mail message is sent to the address specified in the **SMTP Settings pane**.

Saving a server configuration

Changes that you make to a server configuration in the QuarkXPress Server Manager client are not made to the QuarkXPress Server Manager server until you click **Save All Settings** in the **Global Settings** pane of the QuarkXPress Server Manager interface.

To discard any changes you have made since logging on to the QuarkXPress Server Manager server, click **Refresh**. The configuration of the QuarkXPress Server Manager server remains as it was.

➡ Disabling or enabling a QuarkXPress Server instance from the QuarkXPress Server Manager client is not considered a configuration change.

Using custom error messages

You can control which errors cause the QuarkXPress Server Manager server application to send e-mail messages to the address specified in the **SMTP Settings** pane (see "*Generating automatic e-mail messages*"). You can also define which messages are sent when such errors occur. The first step is to create a custom error code that corresponds to a QuarkXPress Server error code. After you set up this custom error code, you can specify whether that code generates an e-mail message and then create custom error messages.

Creating a custom error code

To create a custom error code:

1. On the computer where the QuarkXPress Server Manager server application is running, open the following file in a text-editing application:

   [application folder]\server\conf\Manager_Server_ErrorCodeMapping.properties
2. Create a new line containing a QuarkXPress Server specific error code for which you want to generate automatic e-mail messages (with or without a custom text message). Follow the error code with an equals sign, a unique custom error code, and a return.
3. Save and close the file.

➡ This change will not take effect until you quit and restart the QuarkXPress Server Manager server application.

Flagging an error code to generate an e-mail message

To specify that a custom error code should cause an e-mail message to be generated:

1. Create a unique custom error code for the target QuarkXPress Server error. For more information, see "*Creating a custom error code*").
2. On the computer where the QuarkXPress Server Manager server application is running, open the following file in a text-editing application:

   [application folder]\server\conf\ManagerErrorCodeMailOption.properties
3. Create a new line containing the unique custom error code you defined in step 2. Follow the custom error code with a tab, enter a 1 (to send the message) or a 0 (to suppress the message), and then press Return.
4. Save and close the file.

➡ This change will not take effect until you quit and restart the QuarkXPress Server Manager server application.

Creating custom error text

To define the text that should be sent in an e-mail message when an error occurs:

1. Create a unique custom error code for the target QuarkXPress Server error (as described in "*Creating a custom error code*").
2. On the computer where the QuarkXPress Server Manager server application is running, open the following file in a text-editing application:

   [application folder]\server\conf\ManagerErrorCodeMessage.properties
3. Create a new line containing the unique custom error code you defined in step 2. Follow the custom error code with a tab and then enter the custom text to be returned for that error.
4. Save and close the file.

➡ This change will not take effect until you quit and restart the QuarkXPress Server Manager server application.

Sending requests from a browser

Like QuarkXPress Server, QuarkXPress Server Manager lets you send requests from a Web browser. This capability helps to ensure that you need to make only minimal changes when you update an application so that it sends requests to a QuarkXPress Server Manager server instead of a QuarkXPress Server instance.

Assume that a QuarkXPress Server instance expects requests in the following format:

http://[QXPServer]:[port]/[request]?[request_parameters]

If this is the case, a QuarkXPress Server Manager server will expect requests in the following format:

http://[QXPSManagerServer]:[port]/qxpsm/request/[request]?[request_parameters]

In other words, a QuarkXPress Server Manager server accepts requests in a format that is similar to the request format used with a QuarkXPress Server instance. That means you can get the benefits of QuarkXPress Server Manager without having to completely rewrite your applications.

Additional parameters

In addition to request-specific parameters, QuarkXPress Server Manager accepts the following request parameters. These parameters can be submitted in the standard QuarkXPress Server GET format joined by an ampersand (&) with the other parameters in the body of the request.

- qxpsm_bypassfileinfo: When QuarkXPress Server Manager receives a request, it executes a fileinfo request on the document to get the document's size and last-modified date and time. These values are used for load balancing and for determining whether to serve the document from the cache. If the value of this parameter is set to true, the fileinfo request is not made, file size is considered to be zero for load-balancing purposes, and the document in the cache is considered to have changed.
- qxpsm_context: Set this value if you need to use it in a custom load balancer.
- qxpsm_maxtries: This parameter lets you specify a maximum number of retries for a specific request. If this parameter is absent or set to 0, the global maximum retries value is used. To disable maximum retries for a request, use the value –1.
- qxpsm_password: The value of this parameter, if supplied, is used as the "Admin Password" value when the request is forwarded to a QuarkXPress Server instance.
- qxpsm_responseasurl: By default, QuarkXPress Server Manager writes responses as temporary files in the cache folder and returns a URL to the client. This approach prevents the performance degradation that can result from sending binary data using SOAP. However, you might want QuarkXPress Server Manager to return the response directly if you are creating an application that processes that response (simple or multipart) with its own logic. To make QuarkXPress Server Manager send a response to the browser rather than the URL of the temporary files in the cache, set this value to false. (Note that setting this value to false might result in decreased performance.)
- qxpsm_responseredirect: If you use servlet methodology to send a request to QuarkXPress Server Manager with qxpsm_responseasurl=true, QuarkXPress Server Manager returns a frameset page with one or more frames. If the response is not multipart, the frameset contains a single page with a URL pointing to a response file in the temporary cache. This can be problematic if, for example, you want to use the returned URL as an image link in an HTML page. In such situations, submit the request with both qxpsm_responseredirect=true and qxpsm_responseasurl=true. If you do this, QuarkXPress Server Manager returns the URL of the rendered file in the temporary cache instead of returning a frameset page. Note, however, that if the request results in a multipart response (such as the response returned by the boxes parameter), QuarkXPress Server Manager ignores the qxpsm_responseredirect=true parameter and returns the frameset page.
- qxpsm_servername: By default, the target QuarkXPress Server instance for each request is determined by the QuarkXPress Server Manager server's load-balancing system. To send a request to a specific QuarkXPress Server instance, set this parameter to the name or IP address of that QuarkXPress

Server instance. Note that if you use this parameter with an IP address, you must also submit the port number using the qxpsm_serverport parameter.

- qxpsm_serverport: If you use the qxpsm_servername parameter with an IP address, supply the corresponding port number as this parameter's value.
- qxpsm_timeout: This parameter lets you specify a timeout (in milliseconds) for a specific request. If this parameter is absent or set to 0, the global timeout value is used. To disable timeout for a request, use the value –1.
- qxpsm_usecache: If you set this value to false, the request will be rendered regardless of whether it is cached and regardless of whether caching is enabled at the global level.
- qxpsm_username: The value of this parameter, if supplied, is used as the "Admin User" value when the request is forwarded to a QuarkXPress Server instance.
- qxpsm_userpassword: The value of this parameter, if supplied, is used as the "Admin User" password when the request is forwarded to a QuarkXPress Server instance. Default value is null, which means no password information. If this is not null, you must also provide qxpsm_username.

The XTensions Developer Kit (XDK)

The QuarkXPress Server XDK lets software developers implement features that are not available in QuarkXPress Server, such as server-side processing and application-specific services.

➡ The QuarkXPress Server XDK requires knowledge of C or C++.

Glossary

Document pool: The document pool contains the projects that are available for rendering. By default, the document pool is a collection of discrete files or folders in a specific, identified folder located on the local server or on a connected network drive. When some type of external document provider (such as a content management system or database) is used, projects are not stored in the local document pool.

Document provider: The document provider is the source for projects that QuarkXPress Server renders. The most basic document provider is the local document pool. Other document providers can be enabled through the creation of Server XTensions software, which establish a virtual file system. Server XTensions software can register for control of a specified range of the QuarkXPress Server namespace. When a project is requested from this range, server XTensions software retrieves the file from the specified source and hands it to the server. Examples of document providers include content management systems such as a standard database,or a live data feed from an HTTP agent.

Layout: A layout is a sequence of same-sized pages in a QuarkXPress project. A project can contain one or more layouts. A layout is functionally equivalent to a QuarkXPress document in QuarkXPress 5 and earlier.

Project: A QuarkXPress project is a file created by QuarkXPress. A project can contain one or more layouts.

Rendering: Rendering is the process of generating a file in a particular format (such as JPEG, EPS, or PDF) from a QuarkXPress layout.

Rendering type: The rendering type is the format in which QuarkXPress Server can render QuarkXPress layouts. Some rendering types, such as JPEG and PNG, can be displayed in a Web browser, while others must be saved to the hard drive.

Server XTensions Software (SXT): Server XTensions software is XTensions software written specifically for QuarkXPress Server. For more information, see the QuarkXPress Server XTensions Developer's Kit.

Renderer: A renderer is a process launched by QuarkXPress Server to help process rendering requests. Renderers reside on the same server as QuarkXPress Server and share the same memory and preferences. When renderers are launched, QuarkXPress Server becomes a load-balancing “master server," passing incoming requests to renderers for faster response times.

Legal Notices

©2021 Quark Software Inc. and its licensors. All rights reserved.

Quark, the Quark logo, and QuarkXPress are trademarks or registered trademarks of Quark Software Inc. and its affiliates in the U.S. and/or other countries. All other marks are the property of their respective owners.

If a quark – in science – is the basis for all matter, then Quark – the company – is the basis for all content. The company revolutionized desktop publishing and today provides content design, automation and intelligence software for end-to-end content lifecycle management. Customers worldwide rely on Quark to modernize their content ecosystems so they can create complex print and digital layouts, automate omnichannel publishing of mission-critical documents, and analyze production and engagement insights for the greatest return on their content investments.

© 2024 Quark Software, Inc. All rights reserved. All trademarks, trade names, service marks and logos referenced herein belong to their respective owners.
This content is informational and does not imply any guarantee or warranty.