



Best Practices: Transparency

©2009 Quark Inc. as to the content and arrangement of this material. All rights reserved.

©1986–2009 Quark Inc. and its licensors as to the technology. All rights reserved.

Unauthorized reproduction is a violation of applicable laws.

Quark, the Quark logo and QuarkXPress are trademarks or registered trademarks of Quark Inc. and its affiliates in the U.S. and/or other countries. All other marks are the property of their respective owners.

Best Practices: Transparency

Introduction	1
Understanding transparency	1
Transparency relationships	1
Transparency and output	2
Flattening	3
Native transparency output	5
Workflow requirements	6
Avoid importing transparent PDFs back into QuarkXPress	6
Transparency Rendering Options	6
Upsample Rotations control	7
Ignore Transparency Flattening check box	8
Overlapping resolution values	9
Flattening and color models	10
Stitching issues	11
PDF proofing	12
Best practices for transparency	13
Use transparency judiciously	13
Use transparency intelligently	13
Control color model usage	15
Understand transparency and OPI	15
Use bold with care	15
Watch for color and resolution stitching issues	16
Understand flattening in imported PDF and Illustrator files	17
Use Ignore Transparency Flattening for troubleshooting	18
Understand trapping in transparency relationships	18
Keep RIP overprint settings in mind	19
Watch for changes in text weight	21

INTRODUCTION

This document is intended to help you use the new transparency features in QuarkXPress® 8.1 in the most efficient manner. By following the guidelines in this document, you can avoid potential output problems and minimize the amount of time required to output layouts that use transparency features and to process the resulting output on a RIP.

UNDERSTANDING TRANSPARENCY

Transparency is a category of QuarkXPress 8 features that includes the following:

- The ability to control opacity for text, pictures, items, and groups.
- The ability to use alpha masking to realistically compose the subjects of imported pictures against layout items.
- The ability to apply feathered drop shadows.

Transparency lets you create unique and memorable layouts. But because transparency adds to the complexity of a layout, it also increases the amount of processing necessary to output that layout, as well as the amount of time required to process the output on a RIP. For this reason, it's important to understand how transparency works, so that you can minimize the impact of transparency on processing time without compromising your layout goals.

TRANSPARENCY RELATIONSHIPS

A transparency relationship occurs when a semiopaque object is positioned in front of another object. More specifically, a transparency relationship occurs when an area includes one of the following elements:

- An item that uses a semiopaque color (a color with an opacity between 0% and 100%, non-inclusive).
- A box that contains a blend where at least one of the colors is semiopaque, or where one of the colors is None and the other color is opaque or semiopaque.
- An imported picture with a picture opacity value between 0% and 100%, non-inclusive.
- An imported picture that uses an alpha mask.
- A drop shadow.
- A grayscale picture that uses None as its picture background color.
- A grayscale or 1-bit picture that is colored with a semiopaque color.

The following situations do not create a transparency relationship (except when they occur in conjunction with one of the above situations):

- An item with an opacity of 100%.
- An item with an opacity of 0%.
- An item with a color of None.



When an item in a Web layout causes a transparency relationship, QuarkXPress automatically selects the PNG export format for that item. The resulting exported HTML page uses PNG transparency to display the transparency relationship in the Web browser. (To force such items to go through raster flattening, change their export format from PNG to another format.)

TRANSPARENCY AND OUTPUT

When you send a layout that uses transparency to output, there are a few things you should keep in mind — whether you’re printing to a physical printer or creating a PostScript®, EPS, or PDF file.

QuarkXPress 8.1 can output transparency in two ways:

- *Flattening* is the default method for dealing with transparency at output, and was the only method available in QuarkXPress before version 8.1. During flattening, overlapping items in a transparency relationship are broken down and recombined into a single image before being sent to output. Once flattened, items cannot be moved. Flattening is described in detail in the following sections.
- *Native transparency*, introduced in QuarkXPress 8.1, simply means that items in a transparency relationship are sent to output as separate, overlapping objects. Each item has a transparency attribute, just as it has color and angle attributes. Natively transparent items are composed dynamically, meaning that they can be moved around the layout and will always resolve their transparency against whatever background they’re in front of. Because PostScript does not natively support transparency, this option is available only for PDF export.



Transparency in imported PDF files is a special case. For more information, see “Flattening in imported PDF and Illustrator files.”

FLATTENING

PostScript does not support semiopaque objects. Consequently, QuarkXPress must *flatten* layouts that use transparency before sending them to PostScript output. *Flattening* is the process of turning a layout that contains layered semiopaque objects into a non-layered (flat) layout composed of opaque objects. The QuarkXPress component that flattens layouts is called the *flattener*.

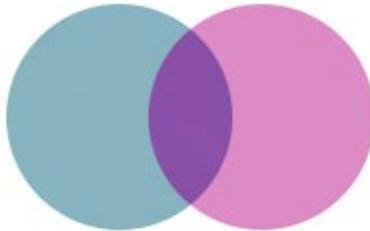
The process of flattening consists of two steps:

- 1 **Vector flattening:** In this step, the flattener breaks down the layout into non-transparent shapes that simulate the appearance of transparency.
- 2 **Rasterization:** In this step, the flattener rasterizes any regions that include raster data in a transparency relationship.

Both steps are described in detail in the following sections.

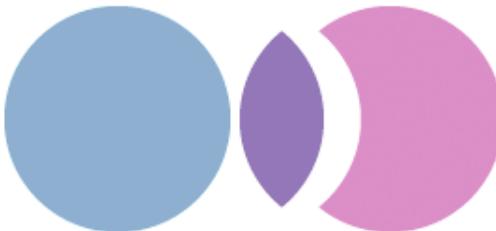
VECTOR FLATTENING

In vector flattening, the flattener breaks down objects in a transparency relationship into vector shapes. For example, assume you have a semiopaque magenta box that partially overlaps an opaque cyan box:



A semiopaque magenta box partially overlapping a cyan box.

During the vector flattening process, these objects are broken down into three vector shapes, as follows:



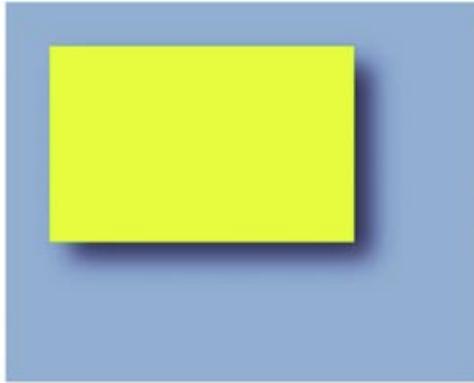
The overlapping boxes as broken down by the flattener.



No rasterization is required in this example because the colors in each of the flattened vector shapes can be rendered as fills in PostScript.

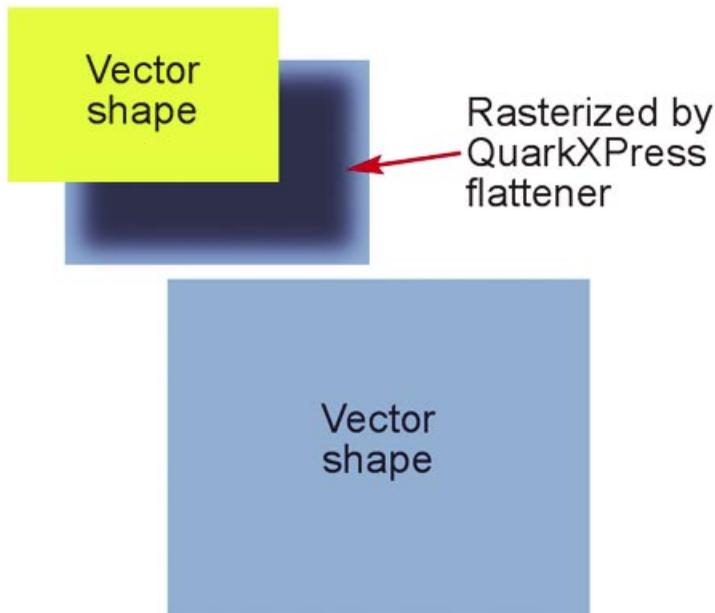
RASTERIZATION

After vector flattening, the flattener rasterizes and composites any non-vector objects that are included in a transparency relationship. For example, assume you have a yellow box with a drop shadow positioned in front of a cyan box:



A yellow box with a drop shadow in front of a cyan box.

During the vector flattening process, this portion of the layout is broken into three pieces, as shown below. Then, during the rasterization process, the portion of the cyan background that is overlapped by the drop shadow is converted into a raster image.



The above layout as broken down by the flattener.

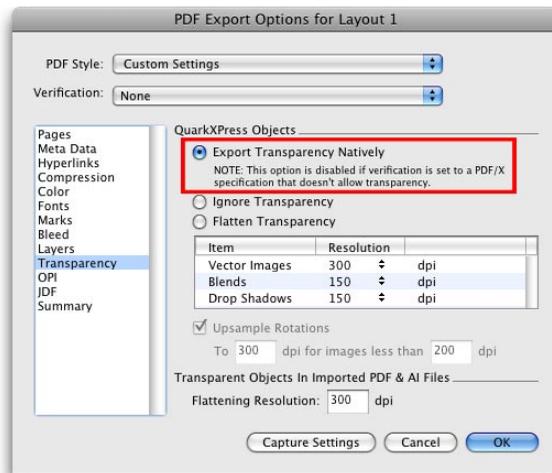
During the flattening process, the flattener also rasterizes any portions of imported vector pictures (such as EPS and PDF files) that are included in transparency relationships. The sole exception to this rule involves images that have been saved from Photoshop® in Photoshop EPS format with the **Include Vector Data** box unchecked; such images are treated the same way as other raster images, such as TIFF files. (For information about resolution considerations with such files, see “Flattening and resolution.”)

It is important to note that even though flattening involves rasterization, it is not the same thing as sending a layout to a RIP. When you flatten a layout, only the portions of the layout that involve raster data in a transparency relationship are rasterized; any vector objects without raster data remain vector objects. If you subsequently send that layout to a RIP, the entire layout is rasterized, meaning:

- Vector objects (including those created by vector flattening) are rasterized at RIP resolution.
- Rasterized regions and imported raster pictures are converted into halftones based on PostScript settings or RIP configuration.

NATIVE TRANSPARENCY OUTPUT

In QuarkXPress 8.1 and later, you can create print-ready non-flattened PDF files by using the **Export Transparency Natively** option in the **PDF Export Options** dialog box (**Transparency** tab). This feature is useful if you need to defer transparency flattening until as late as possible in the workflow, or if you’re experiencing flattening problems and want to test an alternative to the QuarkXPress flattener. This feature creates PDF objects that are inherently transparent and can be flattened later in the workflow (usually immediately before final output).



The Export Transparency Natively option.

WORKFLOW REQUIREMENTS

Not all workflows can support PDFs with native transparency, so check with your output service provider before submitting such PDFs for output. For example, you cannot use native transparency if:

- Your workflow requires PostScript files (.ps, .eps, or .dcs) created in QuarkXPress (because native transparency is supported only for PDF export).
- Your workflow is to create pre-separated files from QuarkXPress.

To consume transparent PDF files, you need a RIP that supports PDF 1.4 or later with transparency. You can use Adobe® Acrobat® to open transparent PDFs and print them directly to your device, or to convert them to flattened PostScript.



Quark does not recommend using native transparency if transparency has been applied directly to a duotone image or to an image meant for OPI replacement. For example, you can use native transparency on an item that interacts with an OPI image (such as a drop shadow over an OPI image) — but do not use OPI on an image that has a transparency feature (such as masking or picture opacity) applied to it.

AVOID IMPORTING TRANSPARENT PDFS BACK INTO QUARKXPRESS

QuarkXPress can flatten imported transparent PDFs during output. However, Quark does not recommend the practice because this functionality is still maturing. If you intend to export a PDF that can be imported into a QuarkXPress layout and then sent to output, we recommend that you use a flattened PDF rather than a PDF with native transparency. If you do import a PDF with native transparency, check the proofs carefully for flattening artifacts.

TRANSPARENCY RENDERING OPTIONS

The controls in the **Transparency** pane of the various output options dialog boxes give you separate control over the resolutions at which vector pictures, transparent blends, and drop shadows are rasterized.

Item	Resolution		
Vector Images	300	↕	dpi
Blends	150	↕	dpi
Drop Shadows	150	↕	dpi

The controls in the **Transparency** pane of the **Print** dialog box give you precise control over the resolution of flattened vector images, blends, and drop shadows.

- The **Vector Images** control lets you specify a resolution for rasterizing pictures that include vector data when they occur in a transparency relationship. It's generally a good idea to keep this value fairly high because vector images typically include sharp lines that will look jagged at lower resolutions. This field also controls the rendering resolution for bitmap frames in a transparency relationship.
- The **Blends** control lets you specify a resolution for blends when they occur in a transparency relationship. Blends can typically be rasterized at a relatively low resolution because they do not contain sharp edges.
- The **Drop Shadows** control lets you specify a resolution for rasterizing drop shadows. This value can be relatively low, unless you create drop shadows with a Blur of zero.

Choosing a lower resolution value for one or more of these fields can reduce the time required for flattening and can save processing time when you send the layout to output.



The resolution controls for bends and drop Shadows apply to natively transparent output, as well as to flattening.

UPSAMPLE ROTATIONS CONTROL

During flattening, rotated or skewed items that participate in a transparency relationship must be rasterized, taking into account any rotation or skew that has been applied to them. Because rotate and skew operations tend to degrade the quality of an image if they are performed at low resolutions, QuarkXPress can upsample such items prior to rotating or skewing them, thus minimizing image degradation.



Use the **Upsample Rotations** controls in the **Transparency** pane of the **Print** dialog box to control the flattening resolution of skewed and rotated items.

Check **Upsample Rotations** if you want to manually set the upsampled resolution for rotated or skewed items and images that are involved in a transparency relationship. If you're using low resolution values, and a rotated or skewed item appears blocky or degraded, check this box and then enter a value in the **To** field that is at least equal to the highest resolution value in the **Transparency Rendering Options** area.

The **dpi for images less than** field lets you specify a value above which rotated or skewed items are not upsampled. The purpose of this field is to prevent rotated or skewed items that are near the **Upsample Rotations To** value from being needlessly upsampled. In general, set this value to about 100 dpi less than the **Upsample Rotations To** value.

IGNORE TRANSPARENCY FLATTENING CHECK BOX

The **Transparency** pane of the **Print** dialog box (**File** menu) contains an **Ignore Transparency Flattening** check box. When you check this box, the flattening step in the output process is skipped. This option can be useful for troubleshooting transparency-related output issues.



The **Ignore Transparency Flattening** area in the **Transparency** pane of the **Print** dialog box lets you turn flattening off.

When using this feature, keep in mind:

- Checking this box might not eliminate transparency if you are printing to a non-PostScript device, especially on Mac OS®.
- Checking this box does not affect transparency within imported PSD files.



The flattener rasterizes an area only if that area includes a raster element such as a drop shadow, a blend, a semiopaque picture, or a picture masked with an alpha channel, or when a vector graphic contributes to a flattenable area. The flattener does not rasterize areas of solid color (regardless of whether they are the result of semiopaque layering) unless such areas are overlapped by a raster element.

The rasterizing engine in the flattener always uses the “nearest-neighbor” algorithm for image scaling to avoid the insertion of pixels with undesirable new colors.

By default, the shape of a drop shadow on a picture with an alpha mask or clipping path is based on the low-resolution preview displayed in a picture box. This works fine for soft-edged drop shadows—but if you are using hard-edged drop shadows (shadows with a very low blur value), use the **Full Res Preview** option for the picture (**View > Full Res Previews and Item > Preview Resolution > Full Resolution**) to avoid “jaggy” drop shadows.

OVERLAPPING RESOLUTION VALUES

Assume a drop shadow overlaps an imported picture. By default, the imported picture retains its own resolution, and the drop shadow rasterizes at the **Drop Shadow** resolution value. Which resolution is used for the area where the drop shadow overlaps the picture?



A drop shadow overlapping a picture.

In such situations, each element that brings a resolution to the transparency relationship is called a *contributor*. The picture contributes its effective resolution (its native resolution divided by its scaling percentage), and the drop shadow contributes the **Drop Shadow** resolution value.

When two contributors overlap, the higher of the two values is used to compose the area of overlap. So, if the **Drop Shadow** resolution value is 300 dots per inch (dpi), and the imported picture is a 300-dpi image scaled to 50% (effective resolution = $300/.5 = 600$ dpi), the area where the drop shadow overlaps the picture is composed at 600 dpi. (More precisely, the drop shadow is rasterized at 300 dpi and then upsampled to 600 dpi to match the resolution of the picture.)

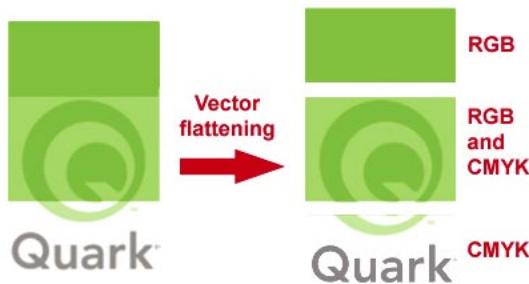
Keep in mind that either the image's effective resolution or the **Transparency Flattening Resolution** value might contribute the higher resolution value in a transparency relationship. Consequently, if you are unhappy with the output resolution of such an area, you might need to adjust one or the other.

- ➔ The flattener rasterizes an area only if that area includes a raster element such as a drop shadow, a blend, a semiopaque picture, or a picture masked with an alpha channel. The flattener does not rasterize areas of solid color (regardless of whether they are the result of semiopaque layering) unless such areas are overlapped by a raster element.

FLATTENING AND COLOR MODELS

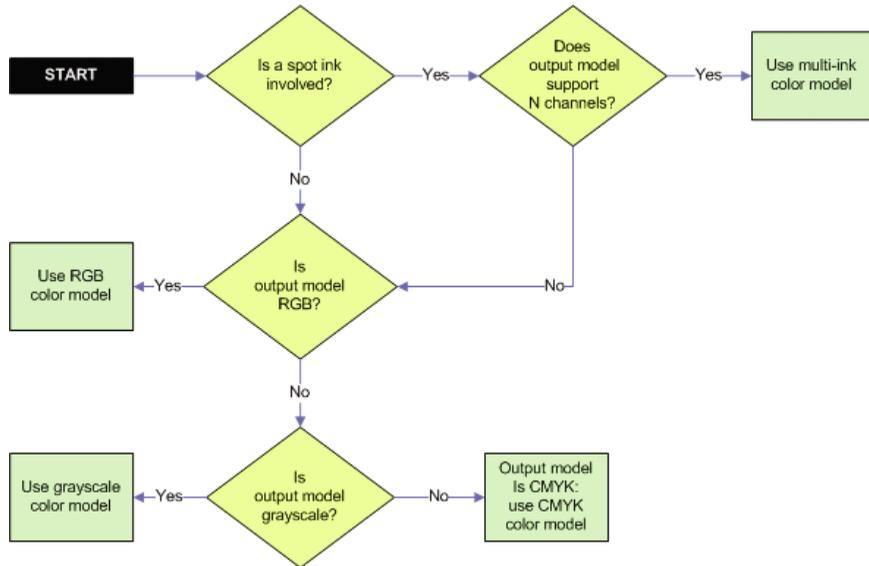
QuarkXPress has always supported a variety of color models, including RGB, CMYK, and “color book” models such as PANTONE®. Transparency features, however, introduce new complexity into color management at output.

For example, assume you’ve positioned a semiopaque CMYK picture in front of a box that uses an RGB color. During flattening, the overlapping area must be rasterized. But the area cannot be rasterized using *both* the RGB and CMYK color models; QuarkXPress must select a color model. How does QuarkXPress make such decisions?



A semiopaque CMYK picture overlapping an RGB box.

The answer depends on the color mode of the output device. The algorithm used by QuarkXPress in such situations can be graphed as follows:



Flowchart showing how the flattener determines which color model to use when multiple color models are involved in a transparency relationship.

This approach to color models ensures that color shifts are avoided during flattening. However, to achieve the best results *overall*, it is important to use discretion when mixing color models when you construct a layout. You can also avoid many issues by using the **View > Proof Output** command to preview the appearance of your layouts on the target output device before you go to output. (For more information about color management, see the “Color Management” chapter in *A Guide to QuarkXPress 8*.)



The **View > Proof Output** command cannot preview color shifts that might occur as a result of flattening. Consequently, it is a good idea to proof layouts that use transparency and multiple color models by exporting the layout in PDF format. For more information, see “PDF proofing.”

STITCHING ISSUES

A *stitching issue* is a situation where an artifact of the flattening process is visible at the edge between two shapes (paths). There are several kinds of stitching issues, including the following:

- *Color shifts* can occur where two abutting areas emerge from the flattener with different color modes. To prevent color shifts, make sure that all of the items in your layout use the same color model.
- *Pixel-alignment issues* can occur when two abutting areas emerge from the flattener with different raster resolutions or a rasterized area abuts a vector

area. If you encounter pixel-alignment issues, make sure the effective resolution of all images and the **Transparency Flattening Resolution** value are the same (or that the higher values are a multiple of the lower values).

- *Antialiasing artifacts* may be visible if you view flattened PDF output in a PDF renderer that antialiases (smoothes) paths on the screen. These artifacts appear as very thin white lines crossing through the design. The antialiasing that causes these lines is typically applied only by renderers that draw PDF files to a computer monitor — not by RIPs that image a page to a physical device — so you usually won't see such lines at print. However, there are several ways you can verify that they are merely artifacts. If you zoom in on such a line and the thickness doesn't increase, it's an artifact. If you disable screen smoothing (in Adobe Acrobat this control is located at **Preferences > Page Display > Rendering > Smooth Line Art**) and the problem disappears, it's an artifact. Of course, a printed proof is always the best test. (Note that because the flattener in QuarkXPress is optimized for pre-press output, you should consider manually flattening any transparency relationships if you're creating a PDF for screen display.)
- In rare cases you may notice that a section of a flattened area is missing. This can be caused by a flattening error that resulted in an incorrect stacking order for the decomposed paths. This can sometimes happen if multiple levels of narrow path overlap. There are three possible solutions to this problem. First, try moving one of the boxes in this flattened area slightly. Second, if your workflow allows it, output the layout as a natively transparent PDF instead of flattening it. Third, open the flattened PDF in an editing application such as Adobe Illustrator® and manually change the stacking order of the boxes.

Stitching issues are generated during the flattening process, so they cannot be previewed on-screen. However, you can proof for stitching areas using printed output.

PDF PROOFING

Because flattening occurs at output, there is no way to proof a flattened layout on the screen in QuarkXPress. However, you can proof a flattened layout by printing the layout or exporting the layout in PDF format.

If you choose to use PDF proofing, it's important to be aware of the following points:

- To optimize readability, most PDF viewers use different methods to draw text, pictures, and other items to the screen. Consequently, different parts of a PDF file might look different both within and between PDF viewers.
- The smoothing (anti-aliasing) feature included in many PDF viewer applications can create the illusion of problems where no problems actually exist.

To avoid the above issues, disable anti-aliasing and smoothing in the preferences for your PDF viewer.

BEST PRACTICES FOR TRANSPARENCY

This section includes a list of guidelines and tips that can help you to get the most out of QuarkXPress transparency features while optimizing the processing time required to output your layouts.

USE TRANSPARENCY JUDICIOUSLY

Wherever a transparency relationship that requires rasterization occurs, QuarkXPress must evaluate every pixel in the area to determine its flattened value. This can take quite a bit of time if the area is large. Even if no rasterization is necessary, QuarkXPress must perform vector flattening for areas that contain transparency relationships — a process that involves quite a bit of geometric conversion. This kind of processor-intensive work is a waste of math if there is nothing for a semiopaque object to interact with. Consequently, the most obvious way to improve the efficiency of a layout is to avoid using transparency where it isn't needed.

Note that the following tips are provided only as suggestions, not as rules. A layout might be simple enough that the processing cost of transparency features is negligible. However, if a layout's processing becomes onerous, you might want to consider the following suggestions:

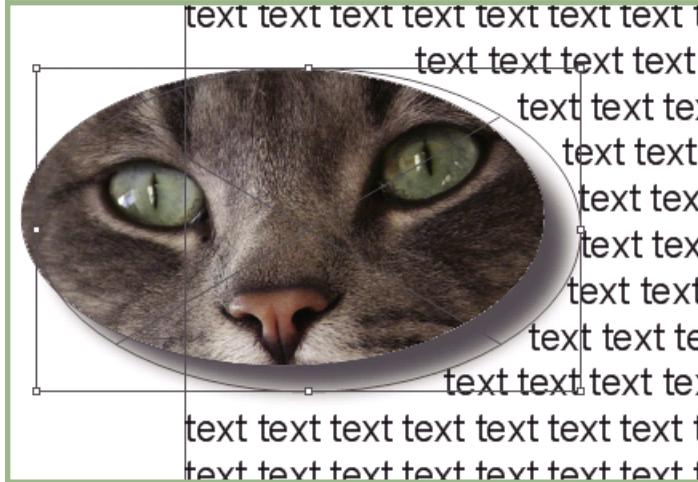
- Avoid using transparency features over page white. For example, if a semiopaque object overlaps only a blank page area, you can probably achieve the same effect by changing the shade of the object rather than its opacity. Similarly, if you are applying opacity to an imported picture against a plain white background, you can probably achieve the same effect by lightening the picture with an image-editing tool such as QuarkVista.
- Avoid using transparency features over solid-color backgrounds. For example, if a semiopaque red object overlaps a blue box, you can probably achieve the same effect by assigning a 100% opaque purple color to the foreground object.
- Avoid using alpha-masked images over plain backgrounds. In such situations, you can probably achieve the same effect by deleting the unmasked portions of the image in an image-editing application.
- If you are using a hard-edged alpha mask on a high-resolution picture, consider using a clipping path instead. You can convert such an alpha mask to a clipping path in an image-editing application, or even create the clipping path yourself in QuarkXPress (**Item > Modify > Clipping tab > Type drop-down menu > Alpha Channel**).

USE TRANSPARENCY INTELLIGENTLY

Depending on the way your layout is constructed, you might be able to use the following tips to minimize processing time:

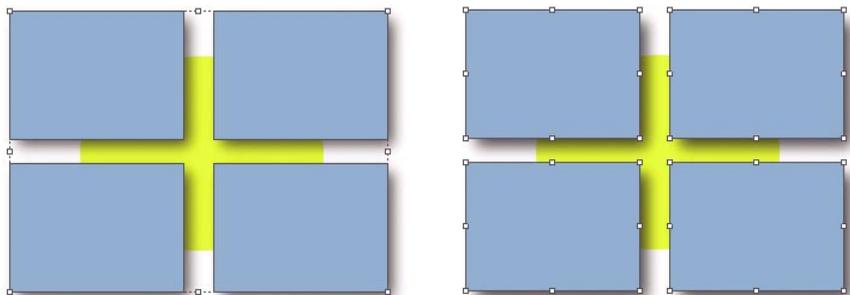
- If a semiopaque object is placed in front of a series of other objects, QuarkXPress must take all of the underlying objects into account during the flattening process. You can save processing time by moving a transparent object as far

back (**Item > Send Backward**) in the stacking order as possible. For example, assume you have imported a picture and applied a drop shadow to its box, and you want to run text around the box. QuarkXPress requires that a text box be behind the object that provides the runaround, so it's easy to assume you must put the text box behind the picture box. However, you can keep the text object in front of the box with the drop shadow by simply using another picture box with a background of None to create the runaround.



Use an empty picture box to create runaround for a drop-shadowed picture box at the rear of the stacking order.

- If you are applying drop shadows to several objects on a page, consider grouping the objects and then applying the drop shadow to the group. This approach consolidates the shadows into a single layer in the stacking order, allowing them to be processed as a single item.



A drop shadow applied to a group (left) produces the same results as four separate drop-shadowed boxes (right) but takes much less time to flatten.

CONTROL COLOR MODEL USAGE

In general, use discretion when mixing color models in a layout. The QuarkXPress color manager can help you avoid issues that can occur when color models are mixed, but the use of transparency introduces additional possibilities for error (see “Flattening and color models”). The best approach is always to make sure that all colors and images that contribute to a layout use the same color model from the beginning.

UNDERSTAND TRANSPARENCY AND OPI

In an OPI workflow, you import a low-resolution version of a picture, then the high-resolution version of the picture is swapped in when the PostScript output is sent to the RIP. What happens if you involve such a picture in a transparency relationship?

The QuarkXPress flattener does its work *before* a layout is output in PostScript format. That means the flattener does not have access to the high-resolution image file. Consequently, the flattener uses the low-resolution file in the flattening process, and the OPI image is never swapped in for any flattened portion of the image.



When part of an OPI image is involved in a transparency relationship, that portion of the image is rasterized by the QuarkXPress flattener. (Resolution in this picture has been exaggerated.)

USE BOLD WITH CARE

You can use two types of bold text in QuarkXPress:

- *Intrinsic bold* is when you use a font that contains bold characters, such as ITC Stone Serif® Std Bold.
- *Transformed bold* is when you use the Bold type style in QuarkXPress to create a simulated bold from a font that has no intrinsic bold in its family; for example, when you apply bold formatting to a specialty font such as Lucida Console.

If you include transformed bold text in a transparency relationship, the text will probably look bad in the output. Consequently, QuarkXPress displays an alert at output if transformed bold is included in a transparency relationship somewhere in the layout.

WATCH FOR COLOR AND RESOLUTION STITCHING ISSUES

Assume you import a grayscale image and then place a drop-shadowed box in front of it.



A grayscale picture with a drop-shadowed box in front of it.

If you've read the first part of this document, you already know that the flattener will first perform vector flattening, breaking the page down into vector shapes. You also know that some portions of the layout will be subsequently rasterized by the flattener, while other parts will be left as-is to be handled by the RIP.



The above layout as broken down by the flattener.

Because the drop shadow area can be rasterized by the flattener at one resolution and the background can be rasterized by the RIP at a different resolution, you have a potential for pixel-alignment stitching issues. You might also encounter stitching issues caused by color shifts, even when you are using a black drop shadow over a grayscale picture.



The area in red could be subject to pixel-alignment stitching issues.

The simplest solution to such issues is to make sure the entire picture is included in the transparency relationship so that all of it is rasterized by the flattener at the same resolution and with the same color model. You can do this by artificially introducing transparency to a wider area of the layout. For example, you might change the opacity of the picture itself to 99.9%, forcing it to be flattened in its entirety.

If the increase in processing time caused by this solution is unacceptable, you can try addressing the stitching issues individually, as follows:

- To address pixel-alignment stitching issues, make sure the effective resolution (native resolution/scaling) of the images in transparency relationships are either the same as or greater than the **Transparency Flattening Resolution** value. Alternatively, make sure that the higher of the two values is a multiple of the lower value.
- To address color-shift stitching issues in this situation, you can try converting the grayscale image into a CMYK image in which all image data is on the black plate.

UNDERSTAND FLATTENING IN IMPORTED PDF AND ILLUSTRATOR FILES

QuarkXPress can import PDF files and Adobe Illustrator (.ai) files that contain native transparency. Such imported images are flattened during output by the flattener built into the Jaws™ engine used by QuarkXPress to read and write PDF files — not by the QuarkXPress flattener. That means most of the controls in the **Transparency** pane of the output options dialog boxes (including the rendering resolution value controls) have no effect on such files.

The only applicable control for such images is the **Flattening Resolution** field under **Transparent Objects in Imported PDF & AI Files** (QuarkXPress 8.1 and later). This field lets you control the resolution of the path deconstruction performed by Jaws during flattening. If you experience errors that indicate this value is set too high, consider flattening the affected PDF or Illustrator file in an application such as Adobe Acrobat or the application that created the file.

QuarkXPress does not support compositing of semiopaque items in imported PDF and EPS files with other page items. If you require such a composition then create all of its elements directly in QuarkXPress or within the application you are using to create graphics.

USE IGNORE TRANSPARENCY FLATTENING FOR TROUBLESHOOTING

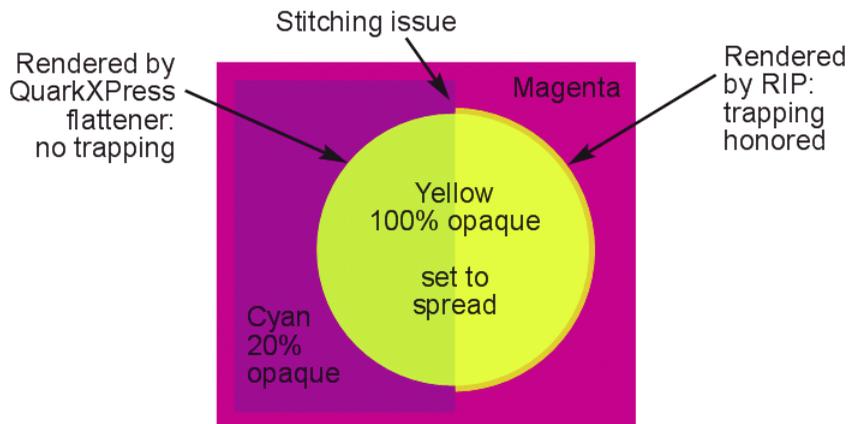
The **Transparency** pane of the **Print** dialog box (**File** menu) contains an **Ignore Transparency Flattening** check box. When you check this box, the flattening step in the output process is skipped. This feature can be useful if you're trying to troubleshoot performance problems at output or on the RIP.

Here are a couple of points to keep in mind when using this feature:

- Checking this box might not eliminate transparency when printing to non-PostScript devices, especially on Mac OS.
- Checking this box does not affect transparency within an imported PSD file.

UNDERSTAND TRAPPING IN TRANSPARENCY RELATIONSHIPS

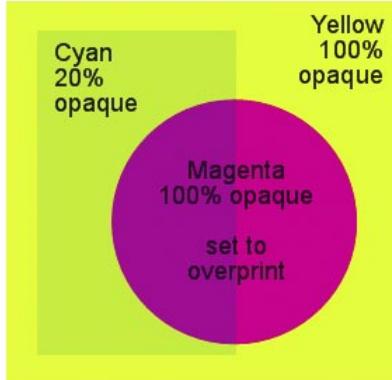
No trapping is applied within flattened areas. Most of the time, this should not be a problem. However, if you are using large trapping values, be aware of stitching issues that can arise when vector objects cross the boundary between areas with and without transparency relationships.



A heavily trapped vector shape that crosses between areas with and without transparency can introduce stitching issues.

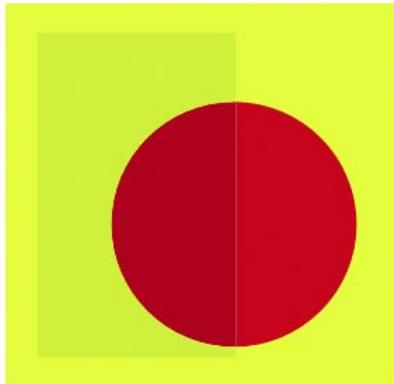
KEEP RIP OVERPRINT SETTINGS IN MIND

A given RIP might or might not honor overprints specified in QuarkXPress. If a RIP does *not* honor overprints specified in the PostScript, be careful about using transparency with objects set to overprint. For example, consider the following layout:



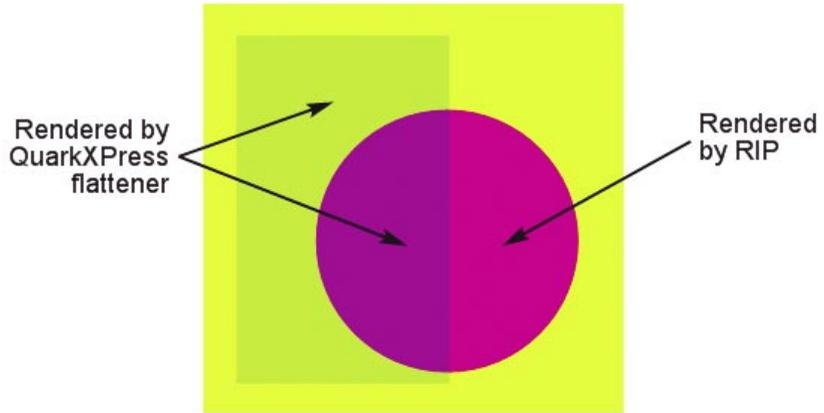
A layout that contains a circular box that has been manually set to overprint in QuarkXPress.

Because the right half of the circle is set to overprint the yellow background, you would expect the area to appear red on the printed page, as shown below.



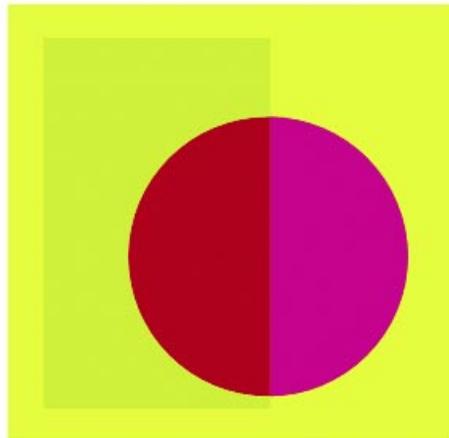
The above layout as it will print if the overprint is honored.

However, because the left side of the circle lies behind a semiopaque object, it will be flattened by the QuarkXPress flattener prior to PostScript output. (An overprint within a transparency relationship must be resolved by the flattener during output — so the overprint is included in the flattened output rather than being deferred to the RIP like a normal overprint.) The right side of the circle, on the other hand, is not part of a transparency relationship, and therefore will be handled by the RIP.



The QuarkXPress flattener will rasterize the area inside the semiopaque cyan box.

If the RIP does not honor the overprint specified in QuarkXPress, the right side of the circle will knock out rather than overprint, resulting in printed output that looks like the following example:



The layout as it will print if overprint is ignored by the RIP.

Although the failed overprint will occur regardless of whether transparency comes into play, the fact that parts of the layout are flattened by the QuarkXPress flattener introduces a chance for additional inconsistency in the printed layout.

WATCH FOR CHANGES IN TEXT WEIGHT

The first step in flattening is to deconstruct the shapes that contribute to the flattenable area. In the case of text, this is done by converting the text glyphs into generic Bézier paths. Although the shape of such a path is identical to the shape of the original font glyph, some RIPs are configured to render text differently than generic Bézier paths (particularly at very small sizes). As a result, the apparent weight of text may change when the text crosses into a flattened area.



Most PDF viewers have special methods for displaying text. As a result, it is not always possible to detect this kind of problem with a PDF proof. If you are working with a layout where text crosses into a flattened area, you may need to print a physical proof.

To address this issue, first confirm that the affected text really needs to contribute to the transparency relationship. If the text is opaque, you may be able to bring it to the top of the stacking order without affecting the design.

If the above solution is not an option, consider applying 99.9% opacity to the entire text run. This causes all of the text to be converted to generic Bézier paths during output, resulting in uniform text weight.
