some cases. For example, some PDF writers split page contents into small streams at arbitrary points that may fall in the middle of lexical tokens within the content, and some PDF readers may get confused on such files. If you use qpdf to coalesce the content streams, such readers may be able to work with the file more easily. This can also be combined with QDF mode or content normalization to make it easier to look at all of a page's contents at once.

--flatten-annotations=*option*

This option collapses annotations into the pages' contents with special handling for form fields. Ordinarily, an annotation is rendered separately and on top of the page. Combining annotations into the page's contents effectively freezes the placement of the annotations, making them look right after various page transformations. The library functionality backing this option was added for the benefit of programs that want to create *n-up* page layouts and other similar things that don't work well with annotations. The *option* parameter may be any of the following:

- **all**: include all annotations that are not marked invisible or hidden
- **print**: only include annotations that indicate that they should appear when the page is printed
- **screen**: omit annotations that indicate they should not appear on the screen

Note that form fields are special because the annotations that are used to render filled-in form fields may become out of date from the fields' values if the form is filled in by a program that doesn't know how to update the appearances. If qpdf detects this case, its default behavior is not to flatten those annotations because doing so would cause the value of the form field to be lost. This gives you a chance to go back and resave the form with a program that knows how to generate appearances. QPDF itself can generate appearances with some limitations. See the **--generate-appearances** option below.

--generate-appearances

If a file contains interactive form fields and indicates that the appearances are out of date with the values of the form, this flag will regenerate appearances, subject to a few limitations. Note that there is not usually a reason to do this, but it can be necessary before using the **--flatten-annotations** option. Most of these are not a problem with well-behaved PDF files. The limitations are as follows:

- Radio button and checkbox appearances use the pre-set values in the PDF file. QPDF just makes sure that the correct appearance is displayed based on the value of the field. This is fine for PDF files that create their forms properly. Some PDF writers save appearances for fields when they change, which could cause some controls to have inconsistent appearances.
- For text fields and list boxes, any characters that fall outside of US-ASCII or, if detected, “Windows ANSI” or “Mac Roman” encoding, will be replaced by the ? character.
- Quadding is ignored. Quadding is used to specify whether the contents of a field should be left, center, or right aligned with the field.
- Rich text, multi-line, and other more elaborate formatting directives are ignored.
- There is no support for multi-select fields or signature fields.

If qpdf doesn't do a good enough job with your form, use an external application to save your filled-in form before processing it with qpdf.

--optimize-images

This flag causes qpdf to recompress all images that are not compressed with DCT (JPEG) using DCT compression as long as doing so decreases the size in bytes of the image data and the image does not fall below minimum specified dimensions. Useful information is provided when used in combination with **--verbose**. See also the **--**