the version is less than 1.7 with extension level 8 (except R5 is disabled if less than 1.7 with extension level 3), AES encryption is disabled if the version is less than 1.6, cleartext metadata and object streams are disabled if less than 1.5, 128-bit encryption keys are disabled if less than 1.4, and all encryption is disabled if less than 1.3. Even with these precautions, qpdf won't be able to do things like eliminate use of newer image compression schemes, transparency groups, or other features that may have been added in more recent versions of PDF.

As a general rule, with the exception of big structural things like the use of object streams or AES encryption, PDF viewers are supposed to ignore features in files that they don't support from newer versions. This means that forcing the version to a lower version may make it possible to open your PDF file with an older version, though bear in mind that some of the original document's functionality may be lost.

By default, when a stream is encoded using non-lossy filters that qpdf understands and is not already compressed using a good compression scheme, qpdf will uncompress and recompress streams. Assuming proper filter implements, this is safe and generally results in smaller files. This behavior may also be explicitly requested with **--stream-data=compress**.

When **--normalize-content=y** is specified, qpdf will attempt to normalize whitespace and newlines in page content streams. This is generally safe but could, in some cases, cause damage to the content streams. This option is intended for people who wish to study PDF content streams or to debug PDF content. You should not use this for “production” PDF files.

This paragraph discusses edge cases of content normalization that are not of concern to most users and are not relevant when content normalization is not enabled. When normalizing content, if qpdf runs into any lexical errors, it will print a warning indicating that content may be damaged. The only situation in which qpdf is known to cause damage during content normalization is when a page's contents are split across multiple streams and streams are split in the middle of a lexical token such as a string, name, or inline image. There may be some pathological cases in which qpdf could damage content without noticing this, such as if the partial tokens at the end of one stream and the beginning of the next stream are both valid, but usually qpdf will be able to detect this case. For slightly increased safety, you can specify **--coalesce-contents** in addition to **--normalize-content** or **--qdf**. This will cause qpdf to combine all the content streams into one, thus recombining any split tokens. However doing this will prevent you from being able to see the original layout of the content streams. If you must inspect the original content streams in an uncompressed format, you can always run with **--qdf --normalize-content=n** for a QDF file without content normalization, or alternatively **--stream-data=uncompress** for a regular non-QDF mode file with uncompressed streams. These will both uncompress all the streams but will not attempt to normalize content. Please note that if you are using content normalization or QDF mode for the purpose of manually inspecting files, you don't have to care about this.

Object streams, also known as compressed objects, were introduced into the PDF specification at version 1.5, corresponding to Acrobat 6. Some older PDF viewers may not support files with object streams. qpdf can be used to transform files with object streams to files without object streams or vice versa. As mentioned above, there are three object stream modes: **preserve**, **disable**, and **generate**.

In **preserve** mode, the relationship to objects and the streams that contain them is preserved from the original file. In **disable** mode, all objects are written as regular, uncompressed objects. The resulting file should be readable by older PDF viewers. (Of course, the content of the files may include features not supported by older viewers, but at least the structure will be supported.) In **generate** mode, qpdf will create its own object streams. This will usually result in more compact PDF files, though they may not be readable by older viewers. In this mode, qpdf will also make sure the PDF version number in the header is at least 1.5.

The **--qdf** flag turns on QDF mode, which changes some of the defaults described above. Specifically, in QDF mode, by default, stream data is uncompressed, content streams are normalized, and encryption is removed. These defaults can still be overridden by specifying the appropriate options as described above. Additionally, in QDF mode, stream lengths are stored as indirect objects, objects are laid out in a less efficient but more readable fashion, and the documents are interspersed with comments that make it easier for the user to find things and also make it possible for **fix-qdf** to work properly. QDF mode is intended for people, mostly developers, who wish to inspect or modify PDF files in a text editor. For details, please see Chapter 4, *QDF Mode*, page 22.