- The **--generate-appearances** flag will cause qpdf to generate appearances for form fields if the PDF file indicates that form field appearances are out of date. This can happen when PDF forms are filled in by a program that doesn't know how to regenerate the appearances of the filled-in fields.

- The **--flatten-annotations** flag can be used to *flatten* annotations, including form fields. Ordinarily, annotations are drawn separately from the page. Flattening annotations is the process of combining their appearances into the page's contents. You might want to do this if you are going to rotate or combine pages using a tool that doesn't understand about annotations. You may also want to use **--generate-appearances** when using this flag since annotations for outdated form fields are not flattened as that would cause loss of information.

- The **--optimize-images** flag tells qpdf to recompresses every image using DCT (JPEG) compression as long as the image is not already compressed with lossy compression and recompressing the image reduces its size. The additional options **--oi-min-width**, **--oi-min-height**, and **--oi-min-area** prevent recompression of images whose width, height, or pixel area (width × height) are below a specified threshold.

- The **--show-object** option can now be given as **--show-object=trailer** to show the trailer dictionary.

- Bug Fixes and Enhancements

  - QPDF now automatically detects and recovers from dangling references. If a PDF file contained an indirect reference to a non-existent object, which is valid, when adding a new object to the file, it was possible for the new object to take the object ID of the dangling reference, thereby causing the dangling reference to point to the new object. This case is now prevented.

  - Fixes to form field setting code: strings are always written in UTF-16 format, and checkboxes and radio buttons are handled properly with respect to synchronization of values and appearance states.

  - The *QPDF::checkLinearization()* no longer causes the program to crash when it detects problems with linearization data. Instead, it issues a normal warning or error.

  - Ordinarily qpdf treats an argument of the form **@file** to mean that command-line options should be read from *file*. Now, if *file* does not exist but *@file* does, qpdf will treat *@file* as a regular option. This makes it possible to work more easily with PDF files whose names happen to start with the `@` character.

- Library Enhancements

  - Remove the restriction in most cases that the source QPDF object used in a *QPDF::copyForeignObject* call has to stick around until the destination QPDF is written. The exceptional case is when the source stream gets is data using a QPDFObjectHandle::StreamDataProvider. For a more in-depth discussion, see comments around *copyForeignObject* in *QPDF.hh*.

  - Add new method *QPDFWriter::getFinalVersion()*, which returns the PDF version that will ultimately be written to the final file. See comments in *QPDFWriter.hh* for some restrictions on its use.

  - Add several methods for transcoding strings to some of the character sets used in PDF files: *QUtil::utf8_to_ascii*, *QUtil::utf8_to_win_ansi*, *QUtil::utf8_to_mac_roman*, and *QUtil::utf8_to_utf16*. For the single-byte encodings that support only a limited character sets, these methods replace unsupported characters with a specified substitute.

  - Add new methods to ***QPDFAnnotationObjectHelper*** and ***QPDFFormFieldObjectHelper*** for querying flags and interpretation of different field types. Define constants in *qpdf/Constants.h* to help with interpretation of flag values.