- Add new generalized methods for reading and writing files from/to programmer-defined sources. The method *QPDF::processInputSource* allows the programmer to use any input source for the input file, and *QPDFWriter::setOutputPipeline* allows the programmer to write the output file through any pipeline. These methods would make it possible to perform any number of specialized operations, such as accessing external storage systems, creating bindings for qpdf in other programming languages that have their own I/O systems, etc.

- Add new method *QPDF::getEncryptionKey* for retrieving the underlying encryption key used in the file.

- This release includes a small handful of non-compatible API changes. While effort is made to avoid such changes, all the non-compatible API changes in this version were to parts of the API that would likely never be used outside the library itself. In all cases, the altered methods or structures were parts of the ***QPDF*** that were public to enable them to be called from either ***QPDFWriter*** or were part of validation code that was over-zealous in reporting problems in parts of the file that would not ordinarily be referenced. In no case did any of the removed methods do anything worse that falsely report error conditions in files that were broken in ways that didn't matter. The following public parts of the ***QPDF*** class were changed in a non-compatible way:

  - Updated nested ***QPDF::EncryptionData*** class to add fields needed by the newer encryption formats, member variables changed to private so that future changes will not require breaking backward compatibility.

  - Added additional parameters to *compute_data_key*, which is used by ***QPDFWriter*** to compute the encryption key used to encrypt a specific object.

  - Removed the method *flattenScalarReferences*. This method was previously used prior to writing a new PDF file, but it has the undesired side effect of causing qpdf to read objects in the file that were not referenced. Some otherwise files have unreferenced objects with errors in them, so this could cause qpdf to reject files that would be accepted by virtually all other PDF readers. In fact, qpdf relied on only a very small part of what flattenScalarReferences did, so only this part has been preserved, and it is now done directly inside ***QPDFWriter***.

  - Removed the method *decodeStreams*. This method was used by the **--check** option of the **qpdf** command-line tool to force all streams in the file to be decoded, but it also suffered from the problem of opening otherwise unreferenced streams and thus could report false positive. The **--check** option now causes qpdf to go through all the motions of writing a new file based on the original one, so it will always reference and check exactly those parts of a file that any ordinary viewer would check.

  - Removed the method *trimTrailerForWrite*. This method was used by ***QPDFWriter*** to modify the original QPDF object by removing fields from the trailer dictionary that wouldn't apply to the newly written file. This functionality, though generally harmless, was a poor implementation and has been replaced by having QPDFWriter filter these out when copying the trailer rather than modifying the original QPDF object. (Note that qpdf never modifies the original file itself.)

- Allow the PDF header to appear anywhere in the first 1024 bytes of the file. This is consistent with what other readers do.

- Fix the **pkg-config** files to list zlib and pcre in *Requires.private* to better support static linking using **pkg-config**.

3.0.2: September 6, 2012

- Bug fix: *QPDFWriter::setOutputMemory* did not work when not used with *QPDFWriter::setStaticID*, which made it pretty much useless. This has been fixed.

- New API call *QPDFWriter::setExtraHeaderText* inserts additional text near the header of the PDF file. The intended use case is to insert comments that may be consumed by a downstream application, though other use cases may exist.