



# Mastering PDF rendering complexities with C++

Nutrient renders documents pixel perfect. We use a custom renderer based on C++ that is shared across all our platforms. This guarantees the broadest test coverage and allows us to focus on a highly tuned codebase for all supported platforms. If you have a document that renders incorrectly or different than with Adobe Acrobat, [report it to us](#).

## Understanding the complexities

PDF has been around since 1993 and has evolved a lot since then. This also means that there's a lot of legacy baggage in the specification — and one of the main promises of PDF is that even your 20-year-old documents will still render as when they were first created.

[The specification, currently at version 1.7, has 756 pages](#). There are also quite a few extensions required ([Adobe® Supplement to ISO 32000-1, Extension Level 3](#), [Adobe® Supplement to ISO 32000-1, Extension Level 5](#)) to parse more modern documents correctly. Additionally, there's an amendment for Adobe Acrobat-specific quirks, and basically everyone else has to follow this to be correct. There's the older original 1.7 version of the document from November 2006, which includes many details — missing in the most current version of the spec — about correctly supporting older documents (1,310 pages) and the [Errata](#) and [Redaction](#) amendments. ([This page provides a good overview](#).)

PDF also has a JavaScript API that is most often used to make forms interactive and to validate input. It's critical that this API is supported by us, as it's also often used for things like turning pages or opening a hyperlink. The [API specification](#) is another whopping 769 pages. The specification for character required to convert PDF glyph data into Unicode for searching purposes, is in another separate document with more than 100 pages. And while Nutrient doesn't support XFA, some informa



handling mixed-mode documents with both AcroForms and XFA is part of the XML Forms Architecture (XFA) Specification, which is a document with about 1,600 pages. (To fully understand the details, sometimes we also need older versions of this document, and [this page provides a good overview.](#)) Since PDF is based on PostScript, the documentation often references the PostScript language reference (1999), which has about 900 pages, and the Errata from 2004. There are various additional documents required for certain encodings used, but these are rarely larger than 100 pages.

Along with the official specification, there’s also “real-world” PDF, which often has bugs based on misunderstandings or bugs in creation software that need specific workarounds and hacks. Adobe Acrobat is very forgiving in terms of invalid references, duplicate entries, and typos, and it allows various drawing command variations that aren’t documented anywhere — these are things that have to be found out via trial and error, and which also add to the complexity of a rendering engine. Our code around this engine is mostly written in modern C++; however, the codebase has to deal with many variants and edge cases, so the code is quite large and complex, resulting in a noticeable binary footprint.

PDF also supports many image formats that require additional libraries to decode — such as JPEG 2000 or [JBIG2](#) — which contributes to the binary size.

---

Was this helpful?

☒ YES

☐ NO

---

Questions? [Contact us](#)

