

A ten-page sample document

Headings, paragraphs, lists and tables across ten A4 pages. By Technology Channel.

A good test file is boring in the best possible way. It opens everywhere, it is exactly what its name says, and it behaves the same way every time you use it. When a bug appears, you want to be sure the problem is in your code and not in a half-broken file you found somewhere online. That is why every file in this collection is generated from scratch, checked after it is written, and labelled with the details that matter: size, dimensions, duration, page count or row count.

Good test files also come in families. A single PDF tells you that your viewer can open a PDF; a family of PDFs tells you how it copes with one page and a hundred pages, with a few kilobytes and ten megabytes, with plain text, scanned images, form fields and passwords. Testing across a family is how you find the edges of a system before your users do.

About this document

Property	Value
Pages	10
Page size	A4
Contains	Headings, lists, tables, page numbers
Fonts	Poppins, Inter (embedded)
Author	Technology Channel

Inside the Portable Document Format

PDF was designed to make a document look the same on every screen and every printer. A PDF file is a collection of numbered objects, such as pages, fonts, images and annotations, tied together by a cross-reference table that lets a reader jump straight to any object without reading the whole file. That structure is why a viewer can show page 80 of a long report almost instantly.

Text in a PDF is not stored as flowing paragraphs. Each piece of text is placed at exact coordinates on the page, often a few words at a time. Extracting readable text therefore means reassembling lines, columns and reading order, which is harder than it sounds. Multi-column layouts, tables and footnotes are classic sources of extraction bugs, and they are worth including in any test set.

PDFs can also carry much more than text. Interactive forms, bookmarks, links, embedded files, digital signatures and encryption are all part of the format. Each feature is optional, and each one is supported unevenly across viewers and libraries. If your product accepts PDFs from users, assume you will eventually receive every one of them.

Images on the web

JPEG remains the workhorse for photographs. It uses lossy compression that throws away detail the eye is unlikely to miss, which keeps files small but means every re-save loses a little quality. PNG is lossless and supports transparency, which makes it the usual choice for screenshots, logos and interface graphics, at the cost of larger files for photographic content.

Things to check

- When testing images, dimensions and file size are only the beginning.
- Check how your system handles transparency, color profiles, CMYK files from print workflows, very large images that strain memory, and EXIF orientation tags that tell a viewer to rotate a photo taken with a sideways phone.
- Animated images and their first frame.
- Very small and very large dimensions.

Newer formats such as WebP and HEIC promise similar quality at a fraction of the size. WebP is now supported by every major browser. HEIC is the default photo format on iPhones, yet many web services still reject it or fail to convert it, which makes it one of the most common causes of upload support tickets.

Video containers and codecs

A video file has two layers that are easy to confuse. The container, such as MP4, MOV, WebM or MKV, is the box that holds the streams, timing information and metadata. The codec, such as H.264, HEVC, VP9 or AV1, is the method used to compress the pictures inside the box. Two MP4 files can contain completely different codecs, and a player that handles one may refuse the other.

Common resolutions

Name	Resolution	Aspect	Typical H.264 bitrate
360p	640 × 360	16:9	0.8 Mbps
480p	854 × 480	16:9	1.5 Mbps
720p	1280 × 720	16:9	3 Mbps
1080p	1920 × 1080	16:9	6 Mbps
4K UHD	3840 × 2160	16:9	20 Mbps

H.264 video with AAC audio in an MP4 container is still the safest combination for playback everywhere. For streaming tests, keep a range of resolutions from 360p to 4K and a range of bitrates, because adaptive players switch between them as network conditions change.

Useful test videos carry visible information: a running timecode, a frame counter, the resolution printed on screen and a short beep or flash at regular intervals. With those clues you can spot dropped frames, audio drift and scaling problems by eye, without special tools.

Sound and audio formats

Digital audio starts as a stream of samples, typically 44,100 or 48,000 per second for each channel. WAV files usually store those samples uncompressed, which makes them large but simple and lossless. MP3 and AAC use perceptual compression to shrink the same audio by a factor of ten or more while keeping it pleasant to listen to.

Audio bugs often hide in the details: an unexpected sample rate, a mono file where stereo was assumed, 24-bit samples in a pipeline built for 16-bit, or a long ID3 tag with cover art that pushes the first audio frame further into the file than a naive parser expects.

Testing a media player or transcription service with several short clips, one long track and a few unusual encodings will surface most of these problems quickly. Keep at least one silent file and one very short file in the set, because edge cases at both ends of the scale are where players tend to fail.

Structured data: JSON, CSV and XML

JSON has become the default language of web APIs. It is easy to read, maps naturally onto objects in most programming languages and supports nesting. Its simplicity also hides traps: very large numbers lose precision in JavaScript, duplicate keys are technically allowed, and a single missing comma makes the whole document invalid.

A quick importer checklist

- 1 Accept UTF-8 with and without a byte-order mark.
- 2 Handle quoted fields that contain commas.
- 3 Handle quoted fields that contain line breaks.
- 4 Detect semicolon delimiters.
- 5 Report the line number of the first error.

CSV looks even simpler, with values separated by commas and one record per line, but there is no single CSV standard in everyday use. Fields may contain commas, quotes or line breaks; files may use semicolons in European locales; spreadsheet programs add a byte-order mark; and old systems still write legacy code pages instead of UTF-8. A robust importer must handle all of these.

Office documents are ZIP files

Modern Word, Excel and PowerPoint files are ZIP archives in disguise. Rename a .docx file to .zip and you will find folders of XML describing the text, styles, images and relationships between them. This design, known as Office Open XML, made the formats far easier for other software to read and write.

Because they are archives, office files can contain surprises: embedded images that make a short document many megabytes in size, macros in the macro-enabled variants, comments and tracked changes that reveal earlier drafts, and hidden sheets in spreadsheets. Document viewers, search indexers and data-loss-prevention tools should all be tested against these cases.

When you generate office documents yourself, open them in more than one application. A file that one program tolerates may trigger a repair prompt in another, and that prompt is enough to make users distrust the export feature you worked hard to build.

Testing file uploads

File upload is one of the riskiest features in any web application. It accepts arbitrary bytes from strangers and often passes them to image processors, document converters and storage services. A good test plan covers the happy path and the many ways it can go wrong.

Upload test cases

Case	Input	Expected
Size limit	File 1 byte over the limit	Clear error message
Empty file	0-byte file	Rejected or handled
Wrong extension	PNG renamed to .jpg	Detected by content
Unicode name	résumé 2026 .pdf	Name preserved
Duplicate	Same file twice	No data loss

Start with limits. Try a file just under the maximum size, exactly at it and just over it. Try an empty file, a file with no extension, a file whose extension does not match its contents and a file whose name contains spaces, accents or emoji. Check what happens when a user uploads the same file twice, or cancels halfway through.

Character encodings

Every text file is a sequence of bytes, and an encoding is the rule that turns those bytes into characters. UTF-8 is now the dominant encoding on the web because it can represent every character in Unicode while remaining compatible with plain ASCII.

Problems appear when software guesses the encoding wrong. The result is garbled text: accented letters turned into strange pairs of symbols, or Asian scripts reduced to question marks. UTF-16 files, byte-order marks and legacy code pages are frequent culprits, and so are emoji that need four bytes in UTF-8 and two code units in UTF-16.

Line endings are a quieter source of trouble. Windows traditionally ends lines with a carriage return and line feed, while macOS and Linux use a line feed alone. Diff tools, parsers and checksums can all behave differently depending on which one they meet.

Automating tests with sample files

Manual testing with sample files is a good start, but the real value comes from automation. Store a small set of representative files alongside your test suite, or download them by URL during a continuous-integration run, and assert on the results: the number of pages extracted, the dimensions of a generated thumbnail, the rows imported from a spreadsheet.

Stable URLs and checksums make this reliable. If a file changes unexpectedly, the checksum comparison fails loudly instead of letting a test pass for the wrong reason. Keep the files small where possible so the suite stays fast, and reserve the largest samples for scheduled performance runs.

Over time, every bug you fix should leave behind a test file that reproduces it. That growing collection becomes one of the most valuable assets a team owns: a precise, executable memory of everything that has gone wrong before.

End of document. This file was generated by Sample Files by Technology Channel.