Why do I get junk characters for some PDF documents when extracting text using the WordFinder?

Not all PDF documents are authored properly for text extraction.

Documents that embed custom encoded fonts will often remap glyphs into non-standard positions. For example, in ASCII, the standard position for the letter A is 65. But a custom encoded font might have some other character in position 65.

The PDF can be rendered with the correct appearance by selecting the correct glyph index value in the embedded font. But the meaning of the character codes used in the text stream are custom codes and are not mapped to any known encoding names. Text returned from the WordFinder may not be recognizable text because there is not enough information provided in the file to convert its content stream into a recognizable character set. If you are using the Library to extract text from a PDF document the text might be rendered as gibberish if the characters in a custom font are not mapped to standard positions.

The PDF Library can still extract the text and return useful data from custom encoded fonts as long as the font contains an optional Differences array that describes the differences from the encoding specified by the font's BaseEncoding, if any. If the embedded font does not contain a Difference array or if the Difference array entries do not match up to standard encoding names, then text cannot be extracted reliably.

In these cases, you can try setting the unknownToStdEnc flag in your wordFinder config.

When extracting text to Unicode, fonts need to have a ToUnicode entry in the font dictionary. Without a ToUnicode table, text cannot be reliably extracted. Look at the Text Extraction sample program in the PDFLSnippetRunner.

Datalogics has run across PDF documents containing custom fonts that maps characters in its ToUnicode table to the Unicode private use area. The Private Use Area of the ToUnicode table, referring to characters between U+E000 and U+F8FF, is used to create custom font mappings. This area of the ToUnicode table allows a developer or vendor to assign Unicode values to glyphs they create for internal use.

If a custom font mapping appears within the Private Use Area within a PDF document, the Adobe PDF Library might not have enough information to interpret the characters completely. By definition, characters in the private-use area are vendor-specific or font-specific entities, and not for general purpose use. In these cases, you could scan the returned text for private-use characters and warn users that not all of the text could be translated.

You could perhaps convert the document by changing all the high-order bytes to the hexadecimal value 0x00. That may work for some PDFs and fonts, but it may not work for all custom fonts that map into the Unicode private-use area.

The high-order byte refers to the last character of a byte string. In a four byte stream, the first byte is the low order byte, and the fourth byte is the high order byte.

If no ToUnicode table exists and the font has an unknown or custom encoding, the Library will attempt to guess the encoding. You can disable that attempt with the noEncodingGuess flag in your wordFinder config, in which case it tries to provide the original characters without any encoding conversion.

To learn more about text extraction, see Section 9.10, “Extraction of Text Content,” in the ISO 32000 Reference, 1.7, page 292. Find this document on the web store of the International Standards Organization (ISO).