



Document

Quick brown



jumped over a lazy



<p>Quick brown fox jumped over a lazy dog. A very long line appears here because we need new line.</p>		<p>Lorem ipsum dolor sit amet,consectetur adipiscing elit.Pellentesque a lectus sit amet lectus accumsan aliquam.Quisque facilisis ullamcorper dolor, quis gravida leo faucibus in. Donec a dolor ligula, quis placerat nunc. Etiam enim velit, egestas in lacinia at, ultricies eu massa.Cras ornare felis id quam vehicula lobortis. Ut semper malesuada nulla, in vulputate dui eleifend at. Phasellus pulvinar nisl a lorem volutpat pellentesque. In vitae ligula et quam vestibulum iaculis eget vitae massa. Fusce vitae leo ut diam suscipit dictum in id sapien. Praesent mi ligula, auctor vitae ultrices in, venenatis nonodio. Nullam sit amet velit pellentesque lectus consectetur lacinia nec quis mi. In hac habitasse platea dictumst.Quisque facilisis ullamcorper dolor, quis gravida leo faucibus in.Donec a dolor ligula, quis placerat nunc.</p>
<p>jumped over a lazy</p>		<p>1. Etiam enim velit, egestas in lacinia at, ultricies eu massa. Cras ornare felis id quam vehicula lobortis. Ut semper malesuada nulla, in vulputate dui eleifend at.Phasellus pulvinar nisl a lorem volutpat pellentesque. In vitae ligula et quam vestibulum iaculis eget vitae massa. Fusce vitae leo ut diam suscipit dictum in idsapien. Praesent mi ligula, auctor vitae ultrices in, venenatis non odio. Nullam sit amet velit pellentesque lectus consectetur lacinia nec quis mi. In hachabitasse platea dictumst.</p> <p>2. Morbi euismod, nunc quis malesuada feugiat, dui nibh rhoncus leo, quis cursus erat tellus vel tortor. Mauris nibh dolor, iaculis et pharetra pretium, pellentesque vitae erat. Aenean enim nisi, euismod quis ultricies vel, convallis nec nulla. Suspendisse nisl purus, molestie et egestas ac, cursus in mauris.Aliquam erat volutpat. Donec at nulla in elit faucibus mollis ac vel enim. Nullam dapibus dui sit amet sem consectetur ac vulputate est sagittis. Aliquam luctus ornare nulla. Mauris adipiscing congue pharetra. Proin tempus, nibh sed pretium tempor, arcu est hendrerit est, et dignissim odio leo non purus.Suspendisse non elit massa. Vestibulum tincidunt ipsum vitae dui congue sagittis. Aenean porttitor tristique euismod. Nulla id justo in quam imperdiet facilisis ut non turpis. Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p> <p>3. Aliquam non elit ligula, nec</p>

<p> Lorem ipsum dolor sit amet,consectetur adipiscing elit.Pellentesque a lectus sit amet lectus accumsan aliquam.Quisque facilisis ullamcorper dolor, quis gravida leo faucibus in. Donec a dolor ligula, quis placerat nunc. Etiam enim velit, egestas in lacinia at, ultricies eu massa.Cras ornare felis id quam vehicula lobortis. Ut semper malesuada nulla, in vulputate dui eleifend at. Phasellus pulvinar nisl a lorem volutpat pellentesque. In vitaeligula et quam vestibulum iaculis eget vitae massa. Fusce vitae leo ut diam suscipit dictum in id sapien. Praesent mi ligula, auctor vitae ultrices in, venenatis nonodio. Nullam sit amet velit pellentesque lectus consectetur lacinia nec quis mi. In hac habitasse platea dictumst.Quisque facilisis ullamcorper dolor, quis gravida leo faucibus in.Donec a dolor ligula, quis placerat nunc. 1. Etiam enim velit, egestas in lacinia at, ultricies eu massa. Cras ornare felis id quam vehicula lobortis. Ut semper </p>	<p>Hello World</p>	<p> hendrerit urna. Mauris ut velit sapien. Sed in convallis diam. Nulla faucibus, purus a porttitor ultrices, est quam convallis magna,molestie aliquam sapien nulla eget metus. Integer nec enim mi, eu mattis massa. Integer quis sapien vel purus pretium ullamcorper ac id dui. Suspendissepellentesque tellus sit amet neque pulvinar egestas lacinia diam imperdiet. 4. Curabitur hendrerit, sem et facilisis vestibulum, massa felis vestibulum ligula, ut faucibus massa nisi in neque. Nulla facilisi. Etiam diam mauris, pellentesquelacinia dapibus at, lobortis non quam. Nullam et neque quis diam vestibulum scelerisque ullamcorper non mauris. Cras massa enim, commodo malesuadatincidunt ac, lobortis eu erat. Sed sed risus velit. Suspendisse tellus tortor, ullamcorper nec tristique ac, semper non nulla. Maecenas vitae diam orci, sedfermentum enim. Curabitur a libero nisl, vel laoreet nulla. Integer id volutpat sem. Pellentesque blandit, tellus at consequat dictum, urna sem elementum nisi,a bibendum nisi ipsum sit amet felis. Donec mattis ipsum nec metus lobortis eget volutpat nisl volutpat. 5. Fusce in aliquet nibh. Etiam quis varius ipsum. Vivamus sit amet mauris a libero iaculis semper in a neque. Nam faucibus congue posuere. Cras vitae nibhsed magna ultricies pretium. Proin eget lacus quis dui ullamcorper cursus commodo in lacus. Quisque et sem id leo venenatis dictum dignissim et felis.Vestibulum enim urna, vehicula vel dictum in, congue quis sapien. Quisque ac mauris tellus. Nulla cursus pellentesque mauris viverra bibendum. Fuscemolestie dui id sem blandit in convallis justo euismod. Curabitur velit nisi, adipiscing sed consequat et, dignissim eget dolor. Aenean malesuada quam id mivestibulum pulvinar. Nullam diam quam, lobortis sit amet semper vitae, tempus eget dolor. </p>
--	--------------------	--

convallis justo euismod. Curabitur velit nisi, adipiscing sed consequat et, dignissim eget dolor. Aenean malesuada quam id mivestibulum pulvinar. Nullam diam quam, lobortis sit amet semper vitae, tempus eget dolor.		
--	--	--

```

xref
0 6
0000000003 65535 f
0000000017 00000 n
0000000081 00000 n
0000000000 00007 f
0000000331 00000 n
0000000409 00000 n

```

EXAMPLE 3 The following shows a cross-reference section with four subsections, containing a total of five entries. The first subsection contains one entry, for object number 0, which is free. The second subsection contains one entry, for object number 3, which is in use. The third subsection contains two entries, for objects number 23 and 24, both of which are in use. Object number 23 has been reused, as can be seen from the fact that it has a generation number of 2. The fourth subsection contains one entry, for object number 30, which is in use.

```

xref
0 1
0000000000 65535 f
3 1
0000025325 00000 n
23 2
0000025518 00002 n
0000025635 00000 n
30 1
0000025777 00000 n

```

See H.7, "Updating Example", for a more extensive example of the structure of a PDF file that has been updated several times.

7.5.5 File Trailer

The *trailer* of a PDF file enables a conforming reader to quickly find the cross-reference table and certain special objects. Conforming readers should read a PDF file from its end. The last line of the file shall contain only the end-of-file marker, **%%EOF**. The two preceding lines shall contain, one per line and in order, the keyword **startxref** and the byte offset in the decoded stream from the beginning of the file to the beginning of the **xref** keyword in the last cross-reference section. The **startxref** line shall be preceded by the *trailer dictionary*, consisting of the keyword **trailer** followed by a series of key-value pairs enclosed in double angle brackets (<<...>>) (using LESS-THAN SIGNS (3Ch) and GREATER-THAN SIGNS (3Eh)). Thus, the trailer has the following overall structure:

```

trailer
<< key1 value1
    key2 value2
    ...
    keyn valuen
>>
startxref
Byte_offset_of_last_cross-reference_section
%%EOF

```

EXAMPLE 2 3 0 R

In an encrypted file (i.e., entire object stream is encrypted), strings occurring anywhere in an object stream shall not be separately encrypted.

A conforming writer shall store the first object immediately after the last byte offset. A conforming reader shall rely on the **First** entry in the stream dictionary to locate the first object.

An object stream itself, like any stream, shall be an indirect object, and therefore, there shall be an entry for it in a cross-reference table or cross-reference stream (see 7.5.8, "Cross-Reference Streams"), although there might not be any references to it (of the form 243 0 R).

The generation number of an object stream and of any compressed object shall be zero. If either an object stream or a compressed object is deleted and the object number is freed, that object number shall be reused only for an ordinary (uncompressed) object other than an object stream. When new object streams and compressed objects are created, they shall always be assigned new object numbers, not old ones taken from the free list.

EXAMPLE 3 The following shows three objects (two fonts and a font descriptor) as they would be represented in a PDF 1.4 or earlier file, along with a cross-reference table.

```

11 0 obj
  << /Type /Font
    /Subtype /TrueType
    ...other entries...
    /FontDescriptor 12 0 R
  >>
endobj

12 0 obj
  << /Type /FontDescriptor
    /Ascent 891
    ...other entries...
    /FontFile2 22 0 R
  >>
endobj

13 0 obj
  << /Type /Font
    /Subtype /Type0
    ...other entries...
    /ToUnicode 10 0 R
  >>
endobj

...

xref
0 32
0000000000 65535 f
... cross-reference entries for objects 1 through 10 ...
0000001434 00000 n
0000001735 00000 n
0000002155 00000 n
... cross-reference entries for objects 14 and on ...
trailer
  << /Size 32
    /Root ...
  >>

```

NOTE 8 For readability, the object stream has been shown unencoded. In a real PDF 1.5 file, Flate encoding would typically be used to gain the benefits of compression.

Document

Quick brown



jumped over a lazy



```

xref
0 6
0000000003 65535 f
0000000017 00000 n
0000000081 00000 n
0000000000 00007 f
0000000331 00000 n
0000000409 00000 n

```

EXAMPLE 3 The following shows a cross-reference section with four subsections, containing a total of five entries. The first subsection contains one entry, for object number 0, which is free. The second subsection contains one entry, for object number 3, which is in use. The third subsection contains two entries, for objects number 23 and 24, both of which are in use. Object number 23 has been reused, as can be seen from the fact that it has a generation number of 2. The fourth subsection contains one entry, for object number 30, which is in use.

```

xref
0 1
0000000000 65535 f
3 1
0000025325 00000 n
23 2
0000025518 00002 n
0000025635 00000 n
30 1
0000025777 00000 n

```

See H.7, "Updating Example", for a more extensive example of the structure of a PDF file that has been updated several times.

7.5.5 File Trailer

The *trailer* of a PDF file enables a conforming reader to quickly find the cross-reference table and certain special objects. Conforming readers should read a PDF file from its end. The last line of the file shall contain only the end-of-file marker, **%%EOF**. The two preceding lines shall contain, one per line and in order, the keyword **startxref** and the byte offset in the decoded stream from the beginning of the file to the beginning of the **xref** keyword in the last cross-reference section. The **startxref** line shall be preceded by the *trailer dictionary*, consisting of the keyword **trailer** followed by a series of key-value pairs enclosed in double angle brackets (<<...>>) (using LESS-THAN SIGNS (3Ch) and GREATER-THAN SIGNS (3Eh)). Thus, the trailer has the following overall structure:

```

trailer
<< key1 value1
    key2 value2
    ...
    keyn valuen
>>
startxref
Byte_offset_of_last_cross-reference_section
%%EOF

```


EXAMPLE 2 3 0 R

In an encrypted file (i.e., entire object stream is encrypted), strings occurring anywhere in an object stream shall not be separately encrypted.

A conforming writer shall store the first object immediately after the last byte offset. A conforming reader shall rely on the **First** entry in the stream dictionary to locate the first object.

An object stream itself, like any stream, shall be an indirect object, and therefore, there shall be an entry for it in a cross-reference table or cross-reference stream (see 7.5.8, "Cross-Reference Streams"), although there might not be any references to it (of the form 243 0 R).

The generation number of an object stream and of any compressed object shall be zero. If either an object stream or a compressed object is deleted and the object number is freed, that object number shall be reused only for an ordinary (uncompressed) object other than an object stream. When new object streams and compressed objects are created, they shall always be assigned new object numbers, not old ones taken from the free list.

EXAMPLE 3 The following shows three objects (two fonts and a font descriptor) as they would be represented in a PDF 1.4 or earlier file, along with a cross-reference table.

```

11 0 obj
  << /Type /Font
    /Subtype /TrueType
    ...other entries...
    /FontDescriptor 12 0 R
  >>
endobj

12 0 obj
  << /Type /FontDescriptor
    /Ascent 891
    ...other entries...
    /FontFile2 22 0 R
  >>
endobj

13 0 obj
  << /Type /Font
    /Subtype /Type0
    ...other entries...
    /ToUnicode 10 0 R
  >>
endobj

...

xref
0 32
0000000000 65535 f
... cross-reference entries for objects 1 through 10 ...
0000001434 00000 n
0000001735 00000 n
0000002155 00000 n
... cross-reference entries for objects 14 and on ...
trailer
  << /Size 32
    /Root ...
  >>

```

NOTE 8 For readability, the object stream has been shown unencoded. In a real PDF 1.5 file, Flate encoding would typically be used to gain the benefits of compression.

EXAMPLE 4 The following shows the same objects from the previous example stored in an object stream in a PDF 1.5 file, along with a cross-reference stream.

The cross-reference stream (see 7.5.8, "Cross-Reference Streams") contains entries for the fonts (objects 11 and 13) and the descriptor (object 12), which are compressed objects in an object stream. The first field of these entries is the entry type (2), the second field is the number of the object stream (15), and the third field is the position within the sequence of objects in the object stream (0, 1, and 2). The cross-reference stream also contains a type 1 entry for the object stream itself.

```

15 0 obj                                % The object stream
  << /Type /ObjStm
    /Length 1856
    /N 3                                % The number of objects in the stream
    /First 24                           % The byte offset in the decoded stream of the first object
  % The object numbers and offsets of the objects, relative to the first are shown on the first line of
  % the stream (i.e., 11 0 12 547 13 665).
  >>
stream
  11 0 12 547 13 665
  << /Type /Font
    /Subtype /TrueType
    ...other keys...
    /FontDescriptor 12 0 R
  >>

  << /Type /FontDescriptor
    /Ascent 891
    ...other keys...
    /FontFile2 22 0 R
  >>

  << /Type /Font
    /Subtype /Type0
    ...other keys...
    /ToUnicode 10 0 R
  >>
...
endstream
endobj

99 0 obj                                % The cross-reference stream
  << /Type /XRef
    /Index [0 32]                       % This section has one subsection with 32 objects
    /W [1 2 2]                         % Each entry has 3 fields: 1, 2 and 2 bytes in width,
                                       % respectively
    /Filter /ASCIISHexDecode           % For readability in this example
    /Size 32
    ...
  >>
stream
  00 0000 FFFF
  ... cross-references for objects 1 through 10 ...
  02 000F 0000
  02 000F 0001
  02 000F 0002
  ... cross-reference for object 14 ...
  01 BA5E 0000
  ...
endstream
endobj

startxref
  54321
%%EOF

```

NOTE 9 The number 54321 in Example 4 is the offset for object 99 0.

The **Size** entry of the trailer shall be large enough to include all objects, including those defined in the cross-reference stream referenced by the **XRefStm** entry. However, to allow random access, a main cross-reference section shall contain entries for all objects numbered 0 through **Size** - 1 (see 7.5.4, "Cross-Reference Table"). Therefore, the **XRefStm** entry shall not be used in the trailer dictionary of the main cross-reference section but only in an update cross-reference section.

When a conforming reader opens a hybrid-reference file, objects with entries in cross-reference streams are not hidden. When the conforming reader searches for an object, if an entry is not found in any given standard cross-reference section, the search shall proceed to a cross-reference stream specified by the **XRefStm** entry before looking in the previous cross-reference section (the **Prev** entry in the trailer).

Hidden objects, therefore, have two cross-reference entries. One is in the cross-reference stream. The other is a free entry in some previous section, typically the section referenced by the **Prev** entry. A conforming reader shall look in the cross-reference stream first, shall find the object there, and shall ignore the free entry in the previous section. A reader designed only to support versions of PDF before PDF 1.5 ignores the cross-reference stream and looks in the previous section, where it finds the free entry. The free entry shall have a next-generation number of 65535 so that the object number shall not be reused.

There are limitations on which objects in a hybrid-reference file can be hidden without making the file appear invalid to readers designed only to support versions of PDF before PDF 1.5. In particular, the root of the PDF file and the document catalog (see 7.7.2, "Document Catalog") shall not be hidden, nor any object that is *visible from the root*. Such objects can be determined by starting from the root and working recursively:

- In any dictionary that is visible, direct objects shall be visible. The value of any required key-value pair shall be visible.
- In any array that is visible, every element shall be visible.
- Resource dictionaries in content streams shall be visible. Although a resource dictionary is not required, strictly speaking, the content stream to which it is attached is assumed to contain references to the resources.

In general, the objects that may be hidden are optional objects specified by indirect references. A conforming reader can resolve those references by processing the cross-reference streams. In a reader designed only to support versions of PDF before PDF 1.5, the objects appear to be free, and the references shall be treated as references to the null object.

EXAMPLE 1 The **Outlines** entry in the catalog dictionary is optional. Therefore, its value may be an indirect reference to a hidden object. A reader designed only to support versions of PDF before PDF 1.5 treats it as a reference to the null object, which is equivalent to having omitted the entry entirely; a conforming reader recognizes it.

If the value of the **Outlines** entry is an indirect reference to a visible object, the entire outline tree shall be visible because nodes in the outline tree contain required pointers to other nodes.

Items that shall be visible include the entire page tree, fonts, font descriptors, and width tables. Objects that may be hidden in a hybrid-reference file include the structure tree, the outline tree, article threads, annotations, destinations, Web Capture information, and page labels,.

EXAMPLE 2 In this example, an **ASCIIHexDecode** filter is specified to make the format and contents of the cross-reference stream readable.

This example shows a hybrid-reference file containing a main cross-reference section and an update cross-reference section with an **XRefStm** entry that points to a cross-reference stream (object 11), which in turn has references to an object stream (object 2).

In this example, the catalog (object 1) contains an indirect reference (3 0 R) to the root of the structure tree. The search for the object starts at the update cross-reference table, which has no objects in it. The search proceeds depending on the version of the conforming reader.

<p> Lorem ipsum dolor sit amet,consectetur adipiscing elit.Pellentesque a lectus sit amet lectus accumsan aliquam.Quisque facilisis ullamcorper dolor, quis gravida leo faucibus in. Donec a dolor ligula, quis placerat nunc. Etiam enim velit, egestas in lacinia at, ultricies eu massa.Cras ornare felis id quam vehicula lobortis. Ut semper malesuada nulla, in vulputate dui eleifend at. Phasellus pulvinar nisl a lorem volutpat pellentesque. In vitaeligula et quam vestibulum iaculis eget vitae massa. Fusce vitae leo ut diam suscipit dictum in id sapien. Praesent mi ligula, auctor vitae ultrices in, venenatis nonodio. Nullam sit amet velit pellentesque lectus consectetur lacinia nec quis mi. In hac habitasse platea dictumst.Quisque facilisis ullamcorper dolor, quis gravida leo faucibus in.Donec a dolor ligula, quis placerat nunc. 1. Etiam enim velit, egestas in lacinia at, ultricies eu massa. Cras ornare felis id quam vehicula lobortis. Ut semper </p>	<p>Hello World</p>	<p> hendrerit urna. Mauris ut velit sapien. Sed in convallis diam. Nulla faucibus, purus a porttitor ultrices, est quam convallis magna,molestie aliquam sapien nulla eget metus. Integer nec enim mi, eu mattis massa. Integer quis sapien vel purus pretium ullamcorper ac id dui. Suspendissepellentesque tellus sit amet neque pulvinar egestas lacinia diam imperdiet. 4. Curabitur hendrerit, sem et facilisis vestibulum, massa felis vestibulum ligula, ut faucibus massa nisi in neque. Nulla facilisi. Etiam diam mauris, pellentesquelacinia dapibus at, lobortis non quam. Nullam et neque quis diam vestibulum scelerisque ullamcorper non mauris. Cras massa enim, commodo malesuadatincidunt ac, lobortis eu erat. Sed sed risus velit. Suspendisse tellus tortor, ullamcorper nec tristique ac, semper non nulla. Maecenas vitae diam orci, sedfermentum enim. Curabitur a libero nisl, vel laoreet nulla. Integer id volutpat sem. Pellentesque blandit, tellus at consequat dictum, urna sem elementum nisi,a bibendum nisi ipsum sit amet felis. Donec mattis ipsum nec metus lobortis eget volutpat nisl volutpat. 5. Fusce in aliquet nibh. Etiam quis varius ipsum. Vivamus sit amet mauris a libero iaculis semper in a neque. Nam faucibus congue posuere. Cras vitae nibhsed magna ultricies pretium. Proin eget lacus quis dui ullamcorper cursus commodo in lacus. Quisque et sem id leo venenatis dictum dignissim et felis.Vestibulum enim urna, vehicula vel dictum in, congue quis sapien. Quisque ac mauris tellus. Nulla cursus pellentesque mauris viverra bibendum. Fuscemolestie dui id sem blandit in convallis justo euismod. Curabitur velit nisi, adipiscing sed consequat et, dignissim eget dolor. Aenean malesuada quam id mivestibulum pulvinar. Nullam diam quam, lobortis sit amet semper vitae, tempus eget dolor. </p>
--	--------------------	--

<p>malesuada nulla, in vulputate dui eleifend at.Phasellus pulvinar nisl a lorem volutpat pellentesque. In vitae ligula et quam vestibulum iaculis eget vitae massa. Fusce vitae leo ut diam suscipit dictum in idsapien. Praesent mi ligula, auctor vitae ultrices in, venenatis non odio. Nullam sit amet velit pellentesque lectus consectetur lacinia nec quis mi. In hachabitasse platea dictumst. 2. Morbi euismod, nunc quis malesuada feugiat, dui nibh rhoncus leo, quis cursus erat tellus vel tortor. Mauris nibh dolor, iaculis et pharetra pretium,pellentesq ue vitae erat. Aenean enim nisi, euismod quis ultrices vel, convallis nec nulla. Suspendisse nisl purus, molestie et egestas ac, cursus in mauris.Aliquam erat volutpat. Donec at nulla in elit faucibus mollis ac vel enim. Nullam dapibus dui sit amet sem consectetur ac vulputate est sagittis. Aliquam luctusornare nulla. Mauris adipiscing congue pharetra. Proin tempus, nibh sed pretium tempor, arcu est hendrerit est, et</p>		
---	--	--

Document Header

Quick brown fox jumped over a lazy
dog. A very long line appears here
because we need new line.



jumped over a lazy



First Edition
2008-7-1

Document management — Portable document format — Part 1: PDF 1.7

4.11**conforming reader**

software application that is able to read and process PDF files that have been made in conformance with this specification and that itself conforms to requirements of conforming readers specified here [ISO 32000-1]

4.12**conforming product**

software application that is both a conforming reader and a conforming writer

4.13**conforming writer**

software application that is able to write PDF files that conform to this specification [ISO 32000-1]

4.14**content stream**

stream object whose data consists of a sequence of instructions describing the graphical elements to be painted on a page

4.15**cross reference table**

data structure that contains the byte offset start for each of the indirect objects within the file

4.16**developer**

Any entity, including individuals, companies, non-profits, standards bodies, open source groups, etc., who are developing standards or software to use and extend ISO 32000-1.

4.17**dictionary object**

an associative table containing pairs of objects, the first object being a name object serving as the key and the second object serving as the value and may be any kind of object including another dictionary

4.18**direct object**

any object that has not been made into an indirect object

4.19**electronic document**

electronic representation of a page-oriented aggregation of text, image and graphic data, and metadata useful to identify, understand and render that data, that can be reproduced on paper or displayed without significant loss of its information content

4.20**end-of-line marker (EOL marker)**

one or two character sequence marking the end of a line of text, consisting of a CARRIAGE RETURN character (0Dh) or a LINE FEED character (0Ah) or a CARRIAGE RETURN followed immediately by a LINE FEED

4.21**FDF file**

File conforming to the Forms Data Format containing form data or annotations that may be imported into a PDF file (see 12.7.7, "Forms Data Format")

4.22**filter**

an optional part of the specification of a stream object, indicating how the data in the stream should be decoded before it is used

level syntactic entities, principally *objects*, which are the basic data values from which a PDF document is constructed.

A non-encrypted PDF can be entirely represented using byte values corresponding to the visible printable subset of the character set defined in ANSI X3.4-1986, plus white space characters. However, a PDF file is not restricted to the ASCII character set; it may contain arbitrary bytes, subject to the following considerations:

- The tokens that delimit objects and that describe the structure of a PDF file shall use the ASCII character set. In addition all the reserved words and the names used as keys in PDF standard dictionaries and certain types of arrays shall be defined using the ASCII character set.
- The data values of strings and streams objects may be written either entirely using the ASCII character set or entirely in binary data. In actual practice, data that is naturally binary, such as sampled images, is usually represented in binary for compactness and efficiency.
- A PDF file containing binary data shall be transported as a binary file rather than as a text file to insure that all bytes of the file are faithfully preserved.

NOTE 1 A binary file is not portable to environments that impose reserved character codes, maximum line lengths, end-of-line conventions, or other restrictions

NOTE 2 In this clause, the usage of the term character is entirely independent of any logical meaning that the value may have when it is treated as data in specific contexts, such as representing human-readable text or selecting a glyph from a font.

7.2.2 Character Set

The PDF character set is divided into three classes, called *regular*, *delimiter*, and *white-space* characters. This classification determines the grouping of characters into tokens. The rules defined in this sub-clause apply to all characters in the file except within strings, streams, and comments.

The *White-space characters* shown in Table 1 separate syntactic constructs such as names and numbers from each other. All white-space characters are equivalent, except in comments, strings, and streams. In all other contexts, PDF treats any sequence of consecutive white-space characters as one character.

Table 1 – White-space characters

Decimal	Hexadecimal	Octal	Name
0	00	000	Null (NUL)
9	09	011	HORIZONTAL TAB (HT)
10	0A	012	LINE FEED (LF)
12	0C	014	FORM FEED (FF)
13	0D	015	CARRIAGE RETURN (CR)
32	20	040	SPACE (SP)

The CARRIAGE RETURN (0Dh) and LINE FEED (0Ah) characters, also called *newline characters*, shall be treated as *end-of-line* (EOL) markers. The combination of a CARRIAGE RETURN followed immediately by a LINE FEED shall be treated as one EOL marker. EOL markers may be treated the same as any other white-space characters. However, sometimes an EOL marker is required or recommended—that is, preceding a token that must appear at the beginning of a line.

NOTE The examples in this standard use a convention that arranges tokens into lines. However, the examples' use of white space for indentation is purely for clarity of exposition and need not be included in practical use.

(normative)

Implementation Limits 649

Annex D
(normative)

Character Sets and Encodings 651

Annex E
(normative)

PDF Name Registry 673

Annex F
(normative)

Linearized PDF 675

Annex G
(informative)

Linearized PDF Access Strategies 695

Annex H
(informative)

Example PDF Files 699

Annex I
(normative)

PDF Versions and Compatibility 727

Annex J
(informative)

FDF Rename Flag Implementation Example 729

Annex K
(informative)

PostScript Compatibility — Transparent Imaging Model 731

Annex L
(informative)

Colour Plates 733

Bibliography 745

Document

Quick brown



jumped over a lazy



convallis justo euismod. Curabitur velit nisi, adipiscing sed consequat et, dignissim eget dolor. Aenean malesuada quam id mivestibulum pulvinar. Nullam diam quam, lobortis sit amet semper vitae, tempus eget dolor.		
--	--	--