备课教案：

讲课人：王航
讲课内容：3《Syntax》 3.1《Lexical Conventions》3.2《Objects》

第三章 语法(Syntax)
名词解释：Syntax
Computer Science The rules governing construction of a machine language.
【计算机科学】 语法：支配某种机器语言的构造的规则
在 PDF Reference 中语法指的是 PDF 文件构造的规则。PDF 语法规则。
没有规矩不成方圆所有的合法 PDF 文档都必须遵守这个规则。
This chapter covers everything about the syntax of PDF at the object, file, and document level. It sets the stage for subsequent chapters, which describe how the contents of a PDF file are interpreted as page descriptions, interactive navigational aids, and application-level logical structure.
第三章涵盖了 PDF 在 Object /file/ document 三个方面的语法规则。它讲述了 PDF 文件的页面描述，交互规则，和应用层面的逻辑结构，并为后面章节的学习打下基础。
PDF syntax is best understood by thinking of it in four parts, as shown in Figure 3.1:
•Objects. A PDF document is a data structure composed from a small set of basic types of data objects.
Section 3.1, “Lexical Conventions,” describes the character set used to write objects and other syntactic elements.
Section 3.2, “Objects,” describes the syntax and essential properties of the objects.
Section 3.2.7, “Stream Objects,” provides complete details of the most complex data type, the stream object.
PDF 语法最好通过以下 4 个部分来理解。
对象：PDF 文档是一个由一些小的基本数据类型组成的。
3.1 节 词法约定 介绍了对象和其他的语法元素使用的字符集。
3.2 节 对象 描述了语法和对象的本质属性。
3.2.7 节 流对象 可以用来完成大部分复杂的数据结构。
•File structure. The PDF file structure determines how objects are stored in a PDF file, how they are accessed, and how they are updated. This structure is independent of the semantics of the objects.
Section 3.4, “File Structure,” describesthe file structure.
Section 3.5, “Encryption,” describes a file-level mechanism for protecting a document’s contents from unauthorized access.
文件结构.PDF 文件结构规定了对象的存储方法，访问方法，以及校正方法。它与对象是两个完全不同的概念。
3.4 节 文件结构
3.5 节 加密技术 介绍了文档加密内容的保护机制。
•Document structure. The PDF document structure specifies how the basic object types are used to represent components of a PDF document: pages, fonts, annotations, and so forth.
Section 3.6, “Document Structure,” describes the overall document structure; later chapters address the detailed semantics of the components.
文档结构.PDF 文档结构用来描述如何使用基本对象类型来构造一个 PDF 文档：page/fonts/annotations 等等。
3.6 节 文档结构 是文档结构的总纲，以后的章节则是详细的语法解释。
•Content streams. A PDF content stream contains a sequence of instructions describing the appearance of a page or other graphical entity. These instructions, while also represented as objects, are conceptually distinct from the objects that represent the document structure and are described separately.
内容串。PDF 内容串让页面或别的实体按一定的次序表达出来。这部分的说明，也提到了对象，可是这里的对象要和文档结构中描述的对象区别开，是不同的两个概念。

Section 3.7, “Content Streams and Resources,” discusses PDF content streams and their associated resources.

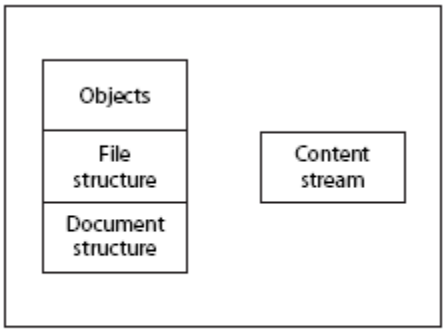

FIGURE 3.1 *PDF components*

In addition, this chapter describes some data structures, built from basic objects, that are so widely used that they can almost be considered basic object types in their own right. These objects are covered in Sections 3.8, “Common Data Structures”; 3.9, “Functions”; and 3.10, “File Specifications.” PDF’s object and file syntax is also used as the basis for other file formats. These include the Forms Data Format (FDF), described in Section 8.6.6, “Forms Data Format,” and the Portable Job Ticket Format (PJTF), described in Adobe Technical Note #5620, Portable Job Ticket Format.

3.7 节 《内容串和数据资源》讨论了 PDF 内容串和与它相关的资源。另外这个章节还介绍了一些由基本对象构造而成的数据结构，但由于广泛使用，所以也被我们把他们当做是基本的数据结构。这些对象将在 3.8 节 通用数据类型 3.9 节 函数 以及 3.10 节 文件格式中详细介绍。PDF 对象和文件的语法规则也被其它的文件格式作为基本规则。这些文件包括，在 8.6.6《表格数据格式》介绍的表格数据格式（*.FDF），以及轻便式工作标签格式（PJTF），这在 Adobe 技术笔记#5620，轻便式工作标签格式中有描述。

第一节 Lexical Conventions

词法约定

At the most fundamental level, a PDF file is a sequence of 8-bit bytes. These bytes can be grouped into tokens according to the syntax rules described below. One or more tokens are assembled to form higher-level syntactic entities, principally objects, which are the basic data values from which a PDF document is constructed.

在底层的角度看，PDF 文件是一串 8 位字节数据。这些字节根据下面的语法规则组合成语法标志。主要的对象就是由这样一个或者多个的语法标志通过更加高级的语法规则组装而成的，它们就是构成 PDF 文档的基本数据。

PDF can be entirely represented using byte values corresponding to the visible printable subset of the ASCII character set, plus white space characters such as space, tab, carriage return, and line feed characters. ASCII is the American Standard Code for Information Interchange, a widely used convention forencoding a specific set of 128 characters as binary numbers. However, a PDF file is not restricted to the ASCII character set; it can contain arbitrary 8-bit bytes, subject to the following considerations:

PDF 文件可以完全用字节值来表达，有 ASCII 字符集中可打印部分、以及间隔符号如空格、制表符、回车及换行符。 ASCII 是美国信息交换标准码，一个广泛使用的包含 128 个字符的二进制字符集。但是基于下面三点考虑，PDF 文件并不局限于 ASCII 字符集，它可以包含任意的 8 位字节。

•The tokens that delimit objects and that describe the structure of a PDF file are all written in the ASCII character set, as are all the reserved words and the names used as keys in standard dictionaries.

所有定义基本对象的语法标志和PDF文件结构描述符号,以及所有的保留字和标准字典中使用的ke关键字全都用的是 ACSCII 字符集。

•The data values of certain types of objects—strings and streams—can be but need not be written entirely in ASCII. For the purpose of exposition (as in this book), ASCII representation is preferred. However, in actual practice, data that is naturally binary, such as sampled images, is represented directly in binary for compactness and efficiency.

可是一些特定的对象类型,比如字符串和数据流不完全是采用 ASCII。为了便于说明（例如这本书），ASCII 是首选的表示方法。然而，实际操作上，例如一个图像采样，为了简洁有效考虑，就需要直

接用二进制来表达。

•A PDF file containing binary data must be transported and stored by means that preserve all bytes of the file faithfully; that is, as a binary file rather than a text file. Such a file is not portable to environments that impose reserved character codes, maximum line lengths, end-of-line conventions, or other restrictions.

包含二进制数据的 PDF 文件必须能够被传送和存储也就意味着，所有的字节信息必须被忠实的保护。更确切的说，二进制文件比一个文本文件更能做到这一点。如果一个文件受到了保留字、每行最大长度，行末结束符约定，或者别的约束，那么他就不容易移到别的环境了。

Note: In this chapter, the term character is synonymous with byte and merely refers to a particular 8-bit value. This usage 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.

注意：本章中，术语字符与字节意思相近，都是指一个有 8 位的值。可是当他作为数据内容的一部分时它不包含任何附加的逻辑含义，只有当人可以从一个字体字形上分辨出来是才表达出文字的含义。

3.1.1Character Set

字符集

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, except within strings, streams, and comments; different rules apply in those contexts.

PDF 字符集被分成三类，常规字符、分隔符和间隔符。除了字符串、数据串、和注释要依据环境采用不同的规则，其余的语法符号都使用了这种办法分类。

White-space characters (see Table 3.1) 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 3.1 White-space characters

DECIMAL	HEXADECIMAL	OCTAL	NAME
0	00	000	Null (NUL)
9	09	011	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)

间隔字符（参考表 3.1）按照了语法结构中的命名和数值来相互区分。除开注释，字符串，和数据流，所有的间隔字符都是等价的。在所有的别的对象结构中，PDF 语法把任何连续的间隔字符看作是一个字符。

The carriage return (CR) and line feed (LF) characters, also called newline characters, are treated as end-of-line (EOL) markers. The combination of a carriage return followed immediately by a line feed is treated as one EOL marker. For the most part, EOL markers are treated the same as any other white-space characters. However, sometimes an EOL marker is required or recommended—that is, the following token must appear at the beginning of a line.

回车符和行末结束符也称为换行符，被看作是一行结束的标记。回车符后紧跟着一个行末结束符的组合被

看成是一个换行标识 EOL（=End of Life）。大多时候，EOL 标识和别的间隔字符是同样看待的。然而，在需要表示下一个语法标志必须出现在行首时，那么。EOL 标识就需要被使用到了。

Note: The examples in this book illustrate a recommended convention for arranging tokens into lines. However, the examples' use of white space for indentation is purely for clarity of exposition and is not recommended for practical use.

注意：本书例子阐明一个把令牌排成行的用法。但是例子中为了缩进使用空格符仅仅只是为了说明清楚，在实际使用中并不推荐。

The delimiter characters (,), <, >, [,], {, }, /, and % are special. They delimit syntactic entities such as strings, arrays, names, and comments. Any of these characters terminates the entity preceding it and is not included in the entity.

All characters except the white-space characters and delimiters are referred to as regular characters. These characters include 8-bit binary characters that are outside the ASCII character set. A sequence of consecutive regular characters comprises a single token.

Note: PDF is case-sensitive; corresponding uppercase and lowercase letters are considered
distinct.

分隔符(,), <, >, [,], {, }, /,和 % 是特殊字符。它们将各个语法实体划分成诸如字符串，数组，名称，和注释等等。任何这些字符都可以用做前面提及实体的结束标志，并且这些字符不作为实体的内容。

除间隔符外，所有的字符和分隔符都是指常规字符。这些字符包括 ASCII 字符集外的 8 位二进制字符。一个个单个的语法符号可以组成一系列的常用字符。

注意：PDF 有区分大小写的。大小字母盘跟小写字母盘被看成是不同的。

3.1.2 Comments

注解

Any occurrence of the percent sign character (%) outside a string or stream introduces a comment. The comment consists of all characters between the percent sign and the end of the line, including regular, delimiter, space, and tab characters. PDF ignores comments, treating them as if they were single white-space characters. That is, a comment separates the token preceding it from the one following it; thus, the PDF fragment

abc% comment { /%) blah blah blah

123

is syntactically equivalent to just the tokens abc and 123.

Comments (other than the %PDF−n.m and %%EOF comments described in Section 3.4, "File Structure") have no semantics. They are not necessarily preserved by applications that edit PDF files (see implementation note 2 in Appendix H). In particular, there is no PDF equivalent of the PostScript document structuring conventions (DSC).

除了字符串或内容流外任何一个地方出现百分号都表示是一个注解。这个注解的有效域从百分号开始一直到本行结束，包括常规字符，定界符，空格符和 Tab。PDF 语法忽视注解，把注释看成一个间隔符。也就是，一个注解把位于它前后的语法标志分隔开；这样 PDF 片断

abc% comment { /%) blah blah blah

123

在语法解释上想当于 abc 123

注解（不同于 3.4 节，《文件结构》中描述的%PDF−n.m 和 %%EOF 注解）在语法上没有任何语义。编写一个 PDF 文件不一定非得保存注解（参看附录 H 执行注释 2）.实际上，这也是不是 PDF 特有的，PostScript 语言在文档结构约定中也是这么规定的。

名词解释：PostScript

http://developer.foxitsoftware.com/forums/read.php?tid=12

第二节 Objects
对象

PDF supports eight basic types of objects:
•Boolean values
•Integer and real numbers
•Strings
•Names
•Arrays
•Dictionaries
•Streams
•The null object

Objects may be labeled so that they can be referred to by other objects. A labeled object is called an indirect object.

The following sections describe each object type, as well as how to create and refer to indirect objects.

PDF 语法支持 8 大基本类型
•布尔型
•整数和实数类型
•字符串类型
•类别名类型
•数组类型
•映射类型
•内容串类型
•空值类型

3.2.1Boolean Objects
布尔对象

PDF provides boolean objects identified by the keywords true and false. Boolean objects can be used as the values of array elements and dictionary entries, and can also occur in PostScript calculator functions as the results of boolean and relational operators and as operands to the conditional operators if and ifelse (see Section 3.9.4, “Type 4 (PostScript Calculator) Functions”).

PDF 提供布尔类型对象，用关键字 true false。布尔对象可以作为数组元素和字典条目，也可以被用在 PostScript 计算中，作为布尔运算结果，或者做为 if ifelse 关系运算中的运算因子，参看（3.9.4 部分，“类型 4（PostScript 计算）”）。

3.2.2Numeric Objects

PDF provides two types of numeric objects: integer and real. Integer objects represent mathematical integers within a certain interval centered at 0. Real objects approximate mathematical real numbers, but with limited range and precision; they are typically represented in fixed-point form rather than floating-point form.The range and precision of numbers are limited by the internal representations used in the computer on which the PDF consumer application is running; Appendix C gives these limits for typical implementations.

PDF 提供两种数字对象：整数和实数。整数对象以零为中心表示一个特定间隔内的整数。实数对象是一个近似值只能在一个有限的范围内保证精确；用固定点描述比用浮点描述更加具有代表性。数字的范围和精确受限于计算机；附录 C 对阐述了这些限制。

An integer is written as one or more decimal digits optionally preceded by a sign:
一个整数也可以用一个符号加一个或多个十进制数字写成：

123 43445 +17 −98 0

The value is interpreted as a signed decimal integer and is converted to an integer object. If it exceeds the implementation limit for integers, it is converted to a real object.
这个值被解释为成是一个带有符号的十进制整数，并被转化成一个整数对象。如果它超过整数的注释限制，就会被转化成实数对象。
A real value is written as one or more decimal digits with an optional sign and a leading, trailing, or embedded period (decimal point):
34.5 −3.62 +123.6 4. −.002 0.0
The value is interpreted as a real number and is converted to a real object. If it exceeds the implementation limit for real numbers, an error occurs.
一个实数可以写成带有一个或者多个小数位数，前面也可以带符号，或者后面可以用一个小数点结尾。
Note: PDF does not support the PostScript syntax for numbers with nondecimal radices (such as 16#FFFE) or in exponential format (such as 6.02E23).
Throughout this book, the term number refers to an object whose type may be either integer or real. Wherever a real number is expected, an integer may be used instead and is automatically converted to an equivalent real value. For example, it is not necessary to write the number 1.0 in real format; the integer 1 is sufficient.
注意：PDF 语法不支持非 10 进制数,如 PostScript 就支持（16#FFFE）或者指数形式（6.02E23）。
贯穿全书，术语 “数字”指的是一个要么是整数要么是实数类型。无论何时，只要需要实数的地方，整数会自动转化成一个对等的实数值。比如，没有必需用实数的格式写数字 1.0，用整数 1 表达就足够了。
3.2.3String Objects
A string object consists of a series of bytes—unsigned integer values in the range 0 to 255. String objects are not integer objects, but are stored in a more compact format. The length of a string may be subject to implementation limits; see Appendix C.
String objects can be written in two ways:
•As a sequence of literal characters enclosed in parentheses ( ); see “Literal Strings,” below”
•As hexadecimal data enclosed in angle brackets < >; see “Hexadecimal Strings” on page 56
一个字符串对象是有一连串的无符号整数值（0—255）字节组成。字符串对象不是整数对象，但是它的存储结构十分的紧凑。实际运行中，字符串的长度可能受到的约束，详细参考附录 C。CImplementation Limits。
字符串对象可以有两种创建办法：
第一种，用在圆括号（）来引用含在里面的文字符号；参看下面的“文字字符串”。
第二种，使用 16 进制数据并用尖括号<>包含；参看 “16 进制字符串”。
This section describes only the basic syntax for writing a string as a sequence of bytes. Strings can be used for many purposes and can be formatted in a variety of ways. When a string is used for a specific purpose (to represent a date, for example),it is useful to have a standard format for that purpose (see Section 3.8.3, “Dates”). Such formats are merely conventions for interpreting the contents of a string and are not separate object types. The use of a particular format is described with the definition of the string object that uses that format.
Section 3.8.1, “String Types” describes the encoding schemes used for the contents of string objects.
这部分只介绍编写字符串的基本语法。字符串可以被用在多种途径上而且可以有很多种格式化的办法。当一个字符串被用来表示一个特定的目的时（比如去表示一个日期），那么有一个标准格式是很有用的（参看 3.8.3 部分，“日期”）。这些格式仅仅是解释字符串内容的习惯用法，可是并不把字符串这种数据类型分开，都叫做字符串。使用这些特殊格式是要按照格式的说明来使用。
3.8.1 节，《字符串类型》介绍了这些格式的说明。
Literal Strings
文字串
A literal string is written as an arbitrary number of characters enclosed in parentheses. Any characters may appear in a string except unbalanced parentheses and the backslash, which must be treated specially. Balanced pairs of parentheses within a string require no special treatment.

文字串是用含在圆括号内的任意字符编写成的。任何一个字符都可以出现在串里面，除了必需做特别处理的 单括号 和反斜线符号，串中成对的括号不需要特别处理。

The following are valid literal strings:

(This is a string)

(Strings may contain newlines
and such .)

(Strings may contain balanced parentheses ( ) and
special characters (* ! & } ^ % and so on) .)

(The following is an empty string .)

( )

(It has zero (0) length .)

Within a literal string, the backslash (\) is used as an escape character for various purposes, such as to include newline characters, nonprinting ASCII characters, unbalanced parentheses, or the backslash character itself in the string. The character immediately following the backslash determines its precise interpretation (see Table 3.2). If the character following the backslash is not one of those shown in the table, the backslash is ignored.

下面字符串是合法有效的：

在一个文字串中，反斜杠被用为转义字符用来各种不同的目的上，比如包括换行符，非打印 ASCII 字符，单括号，或反斜杠字符本身。参看表格 3.2。如果跟着反斜线的字符不在以下列表中，那么反斜线就会被忽视。

TABLE 3.2 Escape sequences in literal strings

SEQUENCE	MEANING
\n	Line feed (LF)
\r	Carriage return (CR)
\t	Horizontal tab (HT)
\b	Backspace (BS)
\f	Form feed (FF)
\(	Left parenthesis
\)	Right parenthesis
\\	Backslash
ddd	Character code *ddd* (octal)

\n 行结束符
\r 回车符
\t 制表符
\b 回退符
\f 表格结束符
\(左括弧
\) 右括弧
\\ 反斜杠

\ddd 八进制字符编码

If a string is too long to be conveniently placed on a single line, it may be split across multiple lines by using the backslash character at the end of a line to indicate that the string continues on the following line. The backslash and the end-of-line marker following it are not considered part of the string. For example:

如果串过于长，不适宜放在一个行内，在行末它有可能被反斜线字符跨行分开，表示这个串在下一行继续。反斜线和跟着它的行末标识不被认为是串的一部分。比如：

(These \
two strings \
are the same .)
(These two strings are the same .)

If an end-of-line marker appears within a literal string without a preceding backslash, the result is equivalent to \n (regardless of whether the end-of-line marker was a carriage return, a line feed, or both). For example:

如果一个行末标识在一个文字串中出现没有反斜线符号跟在前，结果是等同于 \n （不管行末标识是一个回车符，还是一个行填充符，或者两个都是。）比如：

(This string has an end−of−line at the end of it .
)
(So does this one .\n)

The \ddd escape sequence provides a way to represent characters outside the printable ASCII character set. For example:

\ddd 转义字符提供了一种方法来表达在可打印 ASCII 字符集外的字符。比如：

(This string contains \245two octal characters\307 .)

The number ddd may consist of one, two, or three octal digits, with high-order overflow ignored. It is required that three octal digits be used, with leading zeros as needed, if the next character of the string is also a digit. For example, the literal

(\0053)

denotes a string containing two characters, \005 (Control-E) followed by the digit 3, whereas both

数字 ddd 可以包含一个，两个或三个八进制阿拉伯数字，超出位数则忽略。如果下一个字符串也是一个数字，那么它需要使用三个八进制数字，要以 0 开头。比如：文字

(\0053)表示 2 个字符 \005 和 3

(\053)

and

(\53)

denote strings containing the single character \053, a plus sign (+).

都表示 \053 正的

This notation provides a way to specify characters outside the 7-bit ASCII character set by using ASCII characters only. However, any 8-bit value may appear in a string. In particular, when a document is encrypted (see Section 3.5, “Encryption”), all of its strings are encrypted and often contain arbitrary 8-bit values. Note that the backslash character is still required as an escape to specify unbalanced parentheses or the backslash character itself.

这种转义字符为我们提供了一种方法，我们仅仅只要使用 ASCII 字符，只能详细表达出在 7 位 ASCII 字符集外的字符。因为字符串中有可能也会出现 8 位值。特别是当一个文档被加密的时候（参看 3.5 部分，“加密”），文档所含的串都是加密的，经常包含任意 8 位值。

Hexadecimal Strings

Strings may also be written in hexadecimal form, which is useful for including arbitrary binary data in a PDF file. A hexadecimal string is written as a sequence of hexadecimal digits (0–9 and either A–F or a–f) enclosed within

angle brackets (< and >):
串也有可能用十六进制格式来编写，这对囊括 PDF 文件中的任意二进制数据很有用。一个十六进制串被编写成一序列十六进制的数字（0-9 和 A-F 或 a-f），用尖括号括起来：

< 4E6F762073686D6F7A206B6120706F702E >

Each pair of hexadecimal digits defines one byte of the string. White-space characters (such as space, tab, carriage return, line feed, and form feed) are ignored.
If the final digit of a hexadecimal string is missing—that is, if there is an odd number of digits—the final digit is assumed to be 0. For example:
每一对 16 进制数字都对应字符串的一个字节。间隔字符（比如空格键，Tab 键，回车键，行结束符，和表格结束符）都被忽略。
如果一个十六进制串的最后数字丢失，也就是说如果一个十六进制串的最后数字是奇数，那么这个最后数字就会被看成是 0.比如：

< 901FA3 >

is a 3-byte string consisting of the characters whose hexadecimal codes are 90, 1F, and A3, but

< 901FA >

is a 3-byte string containing the characters whose hexadecimal codes are 90, 1F, and A0.

3.2.4Name Objects

名称对象

A name object is an atomic symbol uniquely defined by a sequence of characters. Uniquely defined means that any two name objects made up of the same sequence of characters are identically the same object. Atomic means that a name has no internal structure; although it is defined by a sequence of characters, those characters are not considered elements of the name.
一个名称对象是由一系列字符唯一定义的最小符号。唯一定义意味着任何两个名称对象只要是有同一个序列字符构成的那么都是指同一个对象。最小的意思是一个名称没有更小的内部结构了，尽管它由一系列的字符定义，但是那些字符不被看成是名称的一个个元素。相当与宏

A slash character (/) introduces a name. The slash is not part of the name but is a prefix indicating that the following sequence of characters constitutes a name. There can be no white-space characters between the slash and the first character in the name. The name may include any regular characters, but not delimiter or white-space characters (see Section 3.1, “Lexical Conventions”). Uppercase and lowercase letters are considered distinct: /A and /a are different names. The following examples are valid literal names:
一个斜线字符（/）引导一个名称。斜线不是名称的一部分，但是它是名称的前缀表明它后面所跟的序列字符构成一个名称。名称中斜线和第一个字符间没有空格字符。名称可以包括任何常规字符，但没有分隔符或空格符（参看 3.1 部分“词汇常用法”）。大小写被看成是有区别的：/A 和/a 是两个不同的名称。以下例子是有效的文字名称：

/Name1
/ASomewhatLongerName
/A;Name_With−Various***Characters?
/1 . 2
/$$
/@pattern
/. Notdef

Note: The token / (a slash followed by no regular characters) is a valid name.
注意，斜杠后跟一个非常规字符是有效的。

Beginning with PDF 1.2, any character except null (character code 0) may be included in a name by writing its 2-digit hexadecimal code, preceded by the number sign character (#); see implementation notes 3 and 4 in Appendix H. This syntax is required to represent any of the delimiter or white-space characters or the number sign character itself; it is recommended but not required for characters whose codes are outside the range 33 (!) to 126 (~). The examples shown in Table 3.3 are valid literal names in PDF 1.2 and later.

从 PDF1.2 版本开始，除了空符(字符代码 0)外，通过写 2 位数字十六进制代码任何字符都有可能含在名称中，以字符#号在前引领；参看附录 H 执行注释 3 和 4。这个语法需要再次使用分隔符、空格符或数字符号；推荐使用 33（!）到 126(~)范围的字符。表格 3.3 中的例子在 PDF1.2 和更高的版本中为有效的文字名称。

TABLE 3.3 Examples of literal names using the # character

LITERAL NAME	RESULT
/Adobe#20Green	Adobe Green
/PANTONE#205757#20CV	PANTONE 5757 CV
/paired#28#29parentheses	paired()parentheses
/The_Key_of_F#23_Minor	The_Key_of_F#_Minor
/A#42	AB

The length of a name is subject to an implementation limit; see Appendix C. The limit applies to the number of characters in the name's internal representation. For example, the name /A#20B has four characters (/, A, space, B), not six.

名称的长度；参看附录 C。这个局限适用于计算一个名称内在的长度。 比如，名称/A#20B 有 4 个字符 (/, A, space, B)，而不是 6 个。

As stated above, name objects are treated as atomic symbols within a PDF file. Ordinarily, the bytes making up the name are never treated as text to be presented to a human user or to an application external to a PDF consumer. However, occasionally the need arises to treat a name object as text, such as one that represents a font name (see the BaseFont entry in Table 5.8 on page 413) or a structure type (see Section 10.6.2, "Structure Types").

如上面所述，在 PDF 文件中名称对象被看作是最小单元符号。通常，构成名称的字节不会被当作文本来显示给用户看或作为 PDF 用户的应用。但是，有时也需要把名称对象当作文本来显示，比如表示字体名称的对象（参看第 413 页表格 5.8 BaseFont ）或一个结构类型（参看 10.6.2 部分，"结构类型"）。

In such situations, it is recommended that the sequence of bytes (after expansion of # sequences, if any) be interpreted according to UTF-8, a variable-length byte-encoded representation of Unicode in which the printable ASCII characters have the same representations as in ASCII. This enables a name object to represent text in any natural language, subject to the implementation limit on the length of a name. (See implementation note 5 in Appendix H.)

在这些情况下，推荐使用根据 UTF-8 编码（在#之后，如果有的话），UTF-8 是一个 Unicode 可变长度字节编码表述法，在 Unicode 中可打印 ASCII 字符跟 ASCII 中的字符具有相同的表述法。这就使的名称对象能以自然语言来显示文本。实际使用中，名称的长度是有局限的。（参看附录 H 执行注释 5.）

Note: PDF does not prescribe what UTF-8 sequence to choose for representing any given piece of externally specified text as a name object. In some cases, multiple UTF-8 sequences could represent the same logical text. Name objects defined by different sequences of bytes constitute distinct name objects in PDF, even though the UTF-8 sequences might have identical external interpretations.

注意：PDF 并没有规定选择什么样的 UTF－8 序列来表述作为对象名称的外部给定文字。在某些情况中，

多个 UTF－8 序列可以表示同一个逻辑文本。在 PDF 文件内部，名称对象可以由构成不同的字节序列来定义，尽管 UTF－8 编码可能有相同的外部表述。
名词解释：UTF-8
地址：http://developer.foxitsoftware.com/forums/read.php?tid=15

In PDF, name objects always begin with the slash character (/), unlike keywords such as true, false, and obj. This book follows a typographic convention of writing names without the leading slash when they appear in running text and tables. For example, Type and FullScreen denote names that would actually be written in a PDF file (and in code examples in this book) as /Type and /FullScreen.
在 PDF 文件中，名称对象总是用反斜线符号(/)开始，不象关键字比如 true, false, 和 obj.本书按照印刷排版的习惯，当名称出现在一个可运行文本或表格时，省略了前缀反斜杠。比如，Type 和 FullScreen 实际上就是/Type 和/FullScreen。

3.2.5Array Objects
数组对象

An array object is a one-dimensional collection of objects arranged sequentially. Unlike arrays in many other computer languages, PDF arrays may be heterogeneous;that is, an array's elements may be any combination of numbers, strings, dictionaries, or any other objects, including other arrays. The number of elements in an array is subject to an implementation limit; see Appendix C.
数组对象是能够为了安排一组有序的对象的集合。与其它计算机语言不同，PDF 的数组元素可以是数字，字符串，字典包或其它对象，比如其它的数组。数组中元素编号规范；参看附录 C。
An array is written as a sequence of objects enclosed in square brackets ([and]):
[549 3.14 false (Ralph) /SomeName]
两个数字 一个 BOOL 一个字符串 一个 NAME
PDF directly supports only one-dimensional arrays. Arrays of higher dimension can be constructed by using arrays as elements of arrays, nested to any depth.
一个数组被写成是含在方括号([and])中的一序列对象
PDF 只支持一维数组。更高维的数组可以用数组嵌套数组的方式来构建。

3.2.6Dictionary Objects
字典包对象类型

A dictionary object is an associative table containing pairs of objects, known as the dictionary's entries. The first element of each entry is the key and the second element is the value. The key must be a name (unlike dictionary keys in PostScript, which may be objects of any type). The value can be any kind of object, including another dictionary. A dictionary entry whose value is null (see Section 3.2.8, "Null Object") is equivalent to an absent entry. (This differs from PostScript, where null behaves like any other object as the value of a dictionary entry.) The number of entries in a dictionary is subject to an implementation limit; see Appendix C.
Note: No two entries in the same dictionary should have the same key. If a key does appear more than once, its value is undefined.
字典对象是一个包含了一对 objects 的联合体，称为字典条目。每个条目的第一个元素为关键字，第二个元素是值。关键字必需是一个名称（与 PostScript 语言中的字典关键字不同，它可能是任何类型的对象）。值可以为任何对象，把另外一个字典包做为这个字典包的关键字或者值。一个字典条目如果值为空（参看 3.2.8 部分，"空值"）这样的字典条目相当于无条目。（这点与 PostScript 语言不同，在 PostScript 语言中空值条目跟其它有值的字典条目作用是一样的。）字典中的条目编码规范；参看附录 C。
注意：同一个字典中的两个条目不可以有相同的关键字。如果一个关键字多次出现，那么它的值就无法确定了。

A dictionary is written as a sequence of key-value pairs enclosed in double angle brackets (<< … >>). For example:

字典的编写方式是：Key value 成对出现 用<< >>包含。比如：

```
<< /Type /Example
/Subtype /DictionaryExample
/Version 0.01
/IntegerItem 12
/StringItem (a string)
/Subdictionary << /Item1 0.4
                  /Item2 true
                  /LastItem (not!)
                  /VeryLastItem (OK)
                  >>
>>
```

Note: Do not confuse the double angle brackets with single angle brackets (< and >), which delimit a hexadecimal string (see “Hexadecimal Strings” on page 56).

注意不要混淆( ) < >的用法！

Dictionary objects are the main building blocks of a PDF document. They are commonly used to collect and tie together the attributes of a complex object, such as a font or a page of the document, with each entry in the dictionary specifying the name and value of an attribute. By convention, the Type entry of such a dictionary identifies the type of object the dictionary describes. In some cases, a Subtype entry (sometimes abbreviated S) is used to further identify a specialized subcategory of the general type. The value of the Type or Subtype entry is always a name. For example, in a font dictionary, the value of the Type entry is always Font, whereas that of the Subtype entry may be Type1, TrueType, or one of several other values.

字典对象是构建 PDF 文档最主要的部分。由于字典中的每个条目指定一个属性的名称和值，这样它们就经常被用来把复杂对象的属性聚集并联系在一块，比如文档中的字体或页面。按照惯例，一个字典的 Type 条目只识别这个字典所描述的对象类型。在某些情况中，一个 Subtype 条目（有时简写成 S）被用来进一步的识别总类型中的子种类。Type 和 Subtype 条目的值都是名称对象。比如，在一个字体字典中，Type 条目的值总是 Font，而 Subtype 条目有可能是 Type1、TrueType，或者是某个其它值的值。

The value of the Type entry can almost always be inferred from context. The operand of the Tf operator, for example, must be a font object; therefore, the Type entry in a font dictionary serves primarily as documentation and as information for error checking. The Type entry is not required unless so stated in its description; however, if the entry is present, it must have the correct value. In addition, the value of the Type entry in any dictionary, even in private data, must be either a name defined in this book or a registered name; see Appendix E for details.

Type 条目的值几乎都是从上下文环境中推断得到的。比如 Tf 运算符的操作符必需是一个字体对象；因此在一个字典中的 Type 条目主要的作用就是可以检测出文档信息是否有错误。除非在描述中有特意指定，否则 Type 条目是不一定必需的；但是如果存在 Type 条目，那它必需有正确的值。O(∩_∩)o…哈哈!也就是一定要配对成功！另外，任何字典中的 Type 条目值，即使是在隐蔽数据中，都必须要么是本书所定义的一个名称， 要么是一个已经注册的名称；详细参看附录 E。

3.2.7Stream Objects

内容流对象

A stream object, like a string object, is a sequence of bytes. However, a PDF application can read a stream incrementally, while a string must be read in its entirety. Furthermore, a stream can be of unlimited length, whereas a string is subject to an implementation limit. For this reason, objects with potentially large amounts of data, such as images and page descriptions, are represented as streams.

流对象就像串对像，是一系列字节。然而，PDF 应用程序可以逐步读取一个流，而一个串必须全部一次性读取。而且，一个流的长度可以是不限制的，而串的长度是有约束的。正因为如此，大数据量的对象，比如图像和页面描述，都用流来表达。

Note: As with strings, this section describes only the syntax for writing a stream as a sequence of bytes. What those bytes represent is determined by the context in which the stream is referenced.

A stream consists of a dictionary followed by zero or more bytes bracketed between the keywords stream and endstream:

dictionary

stream

… Zero or more bytes …

Endstream

注意：本部分仅仅描述流的语法层面。至于那些字节表示的是什么，是由流所引用的语境决定的。一个流包含一个字典包对象，后面紧随的是 0 或更多的字节，头尾用关键字 Stream 和 endstream 括起来：

All streams must be indirect objects (see Section 3.2.9, “Indirect Objects”) and the stream dictionary must be a direct object. The keyword stream that follows the stream dictionary should be followed by an end-of-line marker consisting of either a carriage return and a line feed or just a line feed, and not by a carriagereturn alone. The sequence of bytes that make up a stream lie between the streamand endstream keywords; the stream dictionary specifies the exact number of bytes. It is recommended that there be an end-of-line marker after the data and before endstream; this marker is not included in the stream length.

所有的流都必须是间接引用对象（参看 3.2.9 部分，“间接引用对象”），而且流里面的字典包也必须是一个间接引用对象。紧随着流字典的关键字 stream 后面必须跟一个行末标识符（要么包含一个回车符和换行符，要么只是一个换行符），不会是单独一个回车。构成流的序列字节，位于关键字 stream 和 endstream 之间；流字典指定确切的字节编号。推荐在关键字 endstream 之前和数据之后使用行末标识符；这个标识符不含在流长度里。

Alternatively, beginning with PDF 1.2, the bytes may be contained in an external file, in which case the stream dictionary specifies the file, and any bytes between stream and endstream are ignored. (See implementation note 6 in Appendix H.)

另外，从 PDF1.2 版本开始，字节流可能被包在外部文件中，在这种场合下流字典内的值就是所包含的文件，那么在 stream 和 endstream 的任何字节都被忽略。

Note: Without the restriction against following the keyword stream by a carriage return alone, it would be impossible to differentiate a stream that uses carriage return as its end-of-line marker and has a line feed as its first byte of data from one that uses a carriage return–line feed sequence to denote end-of-line.

注意：那就不可能使用回车符作为行末标志和用换行符作为首个数据字节的流 ，来辨别流的开始和结束。

Table 3.4 lists the entries common to all stream dictionaries; certain types of streams may have additional dictionary entries, as indicated where those streams are described. The optional entries regarding filters for the stream indicate whether and how the data in the stream must be transformed (decoded) before it is used. Filters are described further in Section 3.3, “Filters.”

表格 3.4 列出对所有的流字典都通用的选项；某些类型的流可能有一些附加的字典选项，比如流的描述部分。可选选项需要过流滤器来表明流中的数据在被使用前，是否或者需要怎样被解码。就是 3.3SmallYan

同学要讲解的部分，“过滤器”。

TABLE 3.4 Entries common to all stream dictionaries

KEY	TYPE	VALUE
Length	integer	*(Required)* The number of bytes from the beginning of the line following the keyword **stream** to the last byte just before the keyword **endstream**. (There may be an additional EOL marker, preceding **endstream**, that is not included in the count and is not logically part of the stream data.) See “Stream Extent,” above, for further discussion.
Filter	name or array	*(Optional)* The name of a filter to be applied in processing the stream data found between the keywords **stream** and **endstream**, or an array of such names. Multiple filters should be specified in the order in which they are to be applied.
DecodeParms	dictionary or array	*(Optional)* A parameter dictionary or an array of such dictionaries, used by the filters specified by **Filter**. If there is only one filter and that filter has parameters, **DecodeParms** must be set to the filter’s parameter dictionary unless all the filter’s parameters have their default values, in which case the **DecodeParms** entry may be omitted. If there are multiple filters and any of the filters has parameters set to non-default values, **DecodeParms** must be an array with one entry for each filter: either the parameter dictionary for that filter, or the null object if that filter has no parameters (or if all of its parameters have their default values). If none of the filters have parameters, or if all their parameters have default values, the **DecodeParms** entry may be omitted. (See implementation note 7 in Appendix H.)
F	file specification	*(Optional; PDF 1.2)* The file containing the stream data. If this entry is present, the bytes between **stream** and **endstream** are ignored, the filters are specified by **FFilter** rather than **Filter**, and the filter parameters are specified by **FDecodeParms** rather than **DecodeParms**. However, the **Length** entry should still specify the number of those bytes. (Usually, there are no bytes and **Length** is 0.) (See implementation note 46 in Appendix H.)
FFilter	name or array	*(Optional; PDF 1.2)* The name of a filter to be applied in processing the data found in the stream’s external file, or an array of such names. The same rules apply as for **Filter**.
FDecodeParms	dictionary or array	*(Optional; PDF 1.2)* A parameter dictionary, or an array of such dictionaries, used by the filters specified by **FFilter**. The same rules apply as for **DecodeParms**.
DL	integer	*(Optional; PDF 1.5)* A non-negative integer representing the number of bytes in the decoded (defiltered) stream. It can be used to determine, for example, whether enough disk space is available to write a stream to a file. This value should be considered a hint only; for some stream filters, it may not be possible to determine this value precisely.

3.2.8Null Object

The null object has a type and value that are unequal to those of any other object. There is only one object of type null, denoted by the keyword null. An indirect object reference (see Section 3.2.9, “Indirect Objects”) to a nonexistent object is treated the same as a null object. Specifying the null object as the value of a dictionary entry (Section 3.2.6, “Dictionary Objects”) is equivalent to omitting the entry entirely.

空值对象也有一个 type 和 value 可是它不同于其他的对象。有一种由关键字 null 指定的空值对象。一个间接引用对象如果它的值不存在（看 3.2.9 部分“间接对象”）那么我们就把它看作是空值对象。如果某个字典的条目它的值是空值对象，那么相当与这个条目完全被忽视，相当与不存在（参看 3.2.6 部分“字典对象”）。

3.2.9Indirect Objects

间接对象

Any object in a PDF file may be labeled as an indirect object. This gives the object a unique object identifier by which other objects can refer to it (for example, as an element of an array or as the value of a dictionary entry). The object identifier consists of two parts:

PDF 文件中的任何对象都可以被标记为间接对象。这就给每个对象一个唯一对象标识符，通过这个标识符其它对象都可以引用它（比如，数组的一个元素或字典选项的值）。

间接引用对象标识符包括两个部分：

•A positive integer object number. Indirect objects are often numbered sequentially

within a PDF file, but this is not required; object numbers may be assigned in any arbitrary order.

•A non-negative integer generation number. In a newly created file, all indirect objects have generation numbers of 0. Nonzero generation numbers may be introduced

一个正正数对象编号。在 PDF 文件中间接对象经常被按顺序编号，但不是必须的；对象编号可以以任意顺序分配。

一个非负正数生成编号。在一个新生成的文件中，所有间接对象的生成编号都为 0。当文件在以后被更新时就会引进非 0 生成编号；

when the file is later updated; see Sections 3.4.3, “Cross-Reference Table,” and 3.4.5, “Incremental Updates.”

参看 3.4.3 部分，“参照表”和 3.4.5 部分，“逐步更新”。

Together, the combination of an object number and a generation number uniquely identifies an indirect object. The object retains the same object number and generation number throughout its existence, even if its value is modified.

The definition of an indirect object in a PDF file consists of its object number and generation number, followed by the value of the object bracketed between the keywords obj and endobj. For example, the definition

同时，一个对象编号和一个型号组合在一起唯一标识一个间接对象。即使对象的值被修改，都会始终保留同一个对象编号和生产编号。

PDF 文件中一个间接对象的定义包括它的对象编号和生产编号，和位于关键字 obj 和 endobj 之间括号中的值。比如，下面的定义

12 0 obj

(Brillig)

endobj

defines an indirect string object with an object number of 12, a generation number of 0, and the value Brillig.

The object can be referred to from elsewhere in the file by an indirect referenceconsisting of the object number, the generation number, and the keyword R:

12 0 R

Beginning with PDF 1.5, indirect objects may reside in object streams (see Section 3.4.6, “Object Streams”). They are referred to in the same way; however, their definition does not include the keywords obj and endobj.

An indirect reference to an undefined object is not an error; it is simply treated as a reference to the null object.

For example, if a file contains the indirect reference 17 0 R but does not contain the corresponding definition
间接引用一个未定义对象不会出错；它只是被看作引用一个空值对象。比如，如果一个文件包含间接引用 17 0 R 但没有包含相应的定义

17 0 obj

…

endobj

then the indirect reference is considered to refer to the null object.

Note: In the data structures that make up a PDF document, certain values are required to be specified as indirect object references. Except where this is explicitly called out, any object (other than a stream) may be specified either directly or as an indirect object reference; the semantics are entirely equivalent. Note in particular that content streams, which define the visible contents of the document, may not contain indirect references (see Section 3.7.1, “Content Streams”). Also, see implementation note 8 in Appendix H.

Example 3.1 shows the use of an indirect object to specify the length of a stream. The value of the stream’s Length entry is an integer object that follows the streamin the file. This allows applications that generate PDF in a single pass to defer specifying the stream’s length until after its contents have been generated.

Example 3.1

```
7 0 obj
<< /Length 8 0 R >>% An indirect reference to object 8
stream
BT
/F1 12 Tf
72 712 Td
( A stream with an indirect length ) Tj
ET
endstream
endobj
8 0 obj
77% The length of the preceding stream
endobj
```