第四章　Fonts
字体

In text mode TEX takes characters from a 'current font'. This chapter describes how fonts are identified to TEX, and what attributes a font can have.

在文本模式下，TEX 从“当前字体”中获取字符。本章介绍了如何将字体标识给 TEX，以及字体可能具有的属性。

\font Declare the identifying control sequence of a font.
声明字体的标识控制序列。

\fontname The external name of a font.
字体的外部名称。

\nullfont Name of an empty font that TEX uses in emergencies.
TEX 在紧急情况下使用的空字体的名称。

\hyphenchar Number of the hyphen character of a font.
字体的连字符字符的编号。

\defaulthyphenchar Value of \hyphenchar when a font is loaded. Plain TEX default: `\-.
在加载字体时 \hyphenchar 的值。Plain TEX 的默认值为 `-。

\fontdimen Access various parameters of fonts.
访问字体的各种参数。

\char47 Italic correction.
斜体校正。

\noboundary Omit implicit boundary character.
忽略隐式边界字符。

4.1 Fonts
字体

In TEX terminology a font is the set of characters that is contained in one external font file. During processing, TEX decides from what font a character should be taken. This decision is taken separately for text mode and math mode.

在 TEX 的术语中，字体是包含在一个外部字体文件中的字符集合。在处理过程中，TEX 决定从哪个字体中取出一个字符。这个决定针对文本模式和数学模式分别进行。

When TEX is processing ordinary text, characters are taken from the 'current font'. External font file names are coupled to control sequences by statements such as

当 TEX 处理普通文本时，字符是从“当前字体”中取出的。外部字体文件名与控制序列相关联，类似于以下语句：

```
\font\MyFont=myfont10
```

which makes TEX load the file myfont10.tfm. Switching the current font to the font described in that file is then done by

这使得 TEX 加载文件 myfont10.tfm。然后，可以通过以下方式将当前字体切换为该文件描述的字体：

```
\MyFont
```

The status of the current font can be queried: the sequence

可以查询当前字体的状态：序列

```
\the\font
```

produces the control sequence for the current font.

会生成当前字体的控制序列。

Math mode completely ignores the current font. Instead it looks at the 'current family', which can contain three fonts: one for text style, one for script style, and one for scriptscript style. This is treated in Chapter ??.

数学模式完全忽略当前字体。相反，它关注的是“当前族”，其中可以包含三种字体：文本样式、上标样式和次上标样式的字体。这在第 ?? 章中讨论。

See [?] for a consistent terminology of fonts and typefaces.

有关字体和字型的一致术语，请参阅 [?]。

With ‘virtual fonts’ (see [?]) it is possible that what looks like one font to TEX resides in more than one physical font file. See further page ??.

使用“虚拟字体”（参见 [?]）可以实现以下情况：在 TEX 看来，一个字体可能存在于多个物理字体文件中。

请参阅第 ?? 页上的进一步内容。

4.2 Font declaration
字体声明

Somewhere during a run of TEX or IniTEX the coupling between an internal identifying control sequence and the external file name of a font has to be made. The syntax of the command for this is

在 TEX 或 IniTEX 运行期间的某个位置，需要将一个内部标识的控制序列与字体的外部文件名进行关联。此命令的语法如下：

\font⟨control sequence⟩⟨equals⟩⟨file name⟩⟨at clause⟩

where 其中，

⟨at clause⟩ ⟶ at ⟨dimen⟩ | scaled ⟨number⟩ | ⟨optional spaces⟩

Font declarations are local to a group.

字体声明在一个组内生效。

By the ⟨at clause⟩ the user specifies that some magnified version of the font is wanted. The ⟨at clause⟩ comes in two forms: if the font is given scaled f TEX multiplies all its font dimensions for that font by $f/1000$; if the font has a design size dpt and the ⟨at clause⟩ is at ppt TEX multiplies all font data by p/d. The presence of an ⟨at clause⟩ makes no difference for the external font file (the

.tfm file) that TEX reads for the font; it just multiplies the font dimensions by a constant.

通过 ⟨at clause⟩，用户可以指定要使用字体的某个放大版本。⟨at clause⟩ 有两种形式：如果字体给定了 scaledf/，TEX 会将该字体的所有字体尺寸乘以 $f/1000$；如果字体具有设计尺寸 d/pt，并且 ⟨at clause⟩ 为 at p/pt，TEX 会将所有字体数据乘以 p/d。⟨at clause⟩ 的存在对于 TEX 读取的外部字体文件（.tfm 文件）没有影响；它只是将字体尺寸乘以一个常数。

After such a font declaration, using the defined control sequence will set the current font to the font of the control sequence.

在这样的字体声明之后，使用所定义的控制序列将设置当前字体为该控制序列所对应的字体。

4.2.1 Fonts and tfm files
字体和 tfm 文件

The external file needed for the font is a tfm (TEX font metrics) file, which is taken independent of any ⟨at clause⟩ in the \font declaration. If the tfm file has been loaded already (for instance by IniTEX when it constructed the format), an assignment of that font file can be reexecuted without needing recourse to the tfm file.

字体所需的外部文件是一个 tfm（TEX 字体度量）文件，它与 \font 声明中的任何 ⟨at clause⟩ 无关。如果 tfm 文件已经被加载（例如由 IniTEX 在构建格式时加载），则可以重新执行该字体文件的赋值，而无需再次访问 tfm 文件。

Font design sizes are given in the font metrics files. The cmr10 font, for instance, has a design size of 10 point. However, there is not much in the font that actually has a size of 10 points: the opening and closing parentheses are two examples, but capital letters are considerably smaller.

字体设计尺寸在字体度量文件中给出。例如，cmr10 字体的设计尺寸为 10 点。然而，在字体中实际上没有太多尺寸为 10 点的内容：开闭括号就是其中的两个例子，但大写字母要小得多。

4.2.2 Querying the current font and font names
查询当前字体和字体名称

It was already mentioned above that the control sequence which set the current font can be retrieved by the command \the\font. This is a special case of

上面已经提到，可以通过命令 \the\font 获取设置当前字体的控制序列。这是

\the⟨font⟩

where 的一个特例，其中

⟨font⟩ ⟶ \font | ⟨fontdef token⟩ | ⟨family member⟩
⟨family member⟩ ⟶ ⟨font range⟩⟨4-bit number⟩
⟨font range⟩ ⟶ \textfont | \scriptfont | \scriptscriptfont

A ⟨fontdef token⟩ is a control sequence defined by \font, or the predefined control sequence \nullfont. The concept of ⟨family member⟩ is only relevant in math mode.

⟨fontdef token⟩ 是由 \font 定义的控制序列，或者是预定义的控制序列 \nullfont。在数学模式中，⟨family member⟩ 的概念只有在数学模式中才相关。

Also, the external name of fonts can be retrieved:

此外，还可以获取字体的外部名称：

\fontname⟨font⟩

gives a sequence of character tokens of category 12 (but space characters get category 10) that spells the font file name, plus an ⟨at clause⟩ if applicable.

返回一个类别码为 12 的字符记号序列（但空格字符的类别码为 10），拼写字体文件名，如果适用，则包含一个 ⟨at clause⟩。

After 在
\font\tenroman=cmr10 \tenroman
the calls \the\font and \the\tenroman both give \tenroman. The call \fontname\tenroman gives cmr10.
之后，调用 \the\font 和 \the\tenroman 都给出 \tenroman。调用 \fontname\tenroman 给出 cmr10。

4.2.3　\nullfont
\nullfont

TEX always knows a font that has no characters: the \nullfont. If no font has been specified, or if in math mode a family member is needed that has not been specified, TEX will take its characters from the nullfont. This control sequence qualifies as a ⟨fontdef token⟩: it acts like any other control sequence that stands for a font; it just does not have an associated tfm file.

TEX 总是知道一个没有字符的字体：\nullfont。如果没有指定字体，或者在数学模式下需要一个未指定的字族成员，TEX 将从 nullfont 获取其字符。这个控制序列符合 ⟨fontdef token⟩：它的行为类似于代表字体的任何其他控制序列；只是它没有关联的 tfm 文件。

4.3　Font information
字体信息

During a run of TEX the main information needed about the font consists of the dimensions of the characters. TEX finds these in the font metrics files, which usually have extension .tfm. Such files contain

在 TEX 的运行过程中，有关字体的主要信息是字符的尺寸。TEX 从字体度量文件中获取这些信息，这些文件通常具有扩展名 .tfm。这些文件包含以下内容：

- global information: the \fontdimen parameters, and some other information,
  全局信息：\fontdimen 参数和其他一些信息，
- dimensions and the italic corrections of characters, and
  字符的尺寸和斜体校正，
- ligature and kerning programs for characters.
  字符的连字和紧排程序。

Also, the design size of a font is specified in the tfm file; see above. The definition of the tfm format can be found in [?].

此外，字体的设计尺寸也在 tfm 文件中指定；请参见上文。关于 tfm 格式的定义可以在 [?] 中找到。

4.3.1 Font dimensions
字体尺寸

Text fonts need to have at least seven \fontdimen parameters (but TEX will take zero for unspecified parameters); math symbol and math extension fonts have more (see page ??). For text fonts the minimal set of seven comprises the following:

文本字体至少需要有七个 \fontdimen 参数（但对于未指定参数的情况，TEX 会取零）；数学符号字体和数学扩展字体有更多参数（参见第 ??页）。对于文本字体，最小的七个参数包括以下内容：

1. the slant per point; this dimension is used for the proper horizontal positioning of accents;
   每点的倾斜度；这个维度用于正确水平定位重音符号；
2. the interword space: this is used unless the user specifies an explicit \spaceskip; see Chapter ??;
   单词间距：如果用户没有指定显式的 \spaceskip，将使用此间距；请参见第?? 章；
3. interword stretch: the stretch component of the interword space;
   单词间距的可伸展部分；
4. interword shrink: the shrink component of the interword space;
   单词间距的可收缩部分；
5. the x-height: the value of the ⟨internal unit⟩ ex, which is usually about the height of the lowercase letter ‘x’;
   小写字母“x”的高度：⟨ 内部单位 ⟩ ex 的值，通常大约等于小写字母“x”的高度；
6. the quad width: the value of the ⟨internal unit⟩ em, which is approximately the width of the capital letter ‘M’; and
   四分宽度：⟨ 内部单位 ⟩ em 的值，大致等于大写字母“M”的宽度；
7. the extra space: the space added to the interword space at the end of sentences (that is, when \spacefactor ≥ 2000) unless the user specifies an explicit \xspaceskip.
   额外空格：在句子末尾（即当 \spacefactor ≥ 2000 时）加到单词间距的空格，除非用户指定了显式的 \x-space-skip。

Parameters 1 and 5 are purely information about the font and there is no point in varying them. The values of other parameters can be changed in order to adjust spacing; see Chapter ?? for examples of changing parameters 2, 3, 4, and 7.

参数 1 和 5 纯粹是字体的信息，没有改变它们的意义。可以改变其他参数的值以调整间距；请参见第 ?? 章中更改参数 2、3、4 和 7 的示例。

Font dimensions can be altered in a ⟨font assignment⟩, which is a ⟨global assignment⟩ (see page ??):

字体维度可以在 ⟨ 字体赋值 ⟩ 中进行更改，这是一个 ⟨ 全局赋值 ⟩（参见第 ?? 页）：

\fontdimen⟨number⟩⟨font⟩⟨equals⟩⟨dimen⟩

See above for the definition of ⟨font⟩.

⟨ 字体 ⟩ 的定义请参见上文。

4.3.2 Kerning
字距调整

Some combinations of characters should be moved closer together than would be the case if their bounding boxes were to be just abutted. This fine spacing is called kerning, and a proper kerning is as essential to a font as the design of the letter shapes.

某些字符的组合应该比它们的包围盒紧密地排列在一起，而不是紧邻在一起。这种精细的间距被称为字距调整，对于字体来说，适当的字距调整和字形设计一样重要。

Consider as an example

以一个例子来说明：

‘Vo’ versus the unkerned variant ‘Vo’

‘Vo’ 与未进行字距调整的变体 ‘Vo’

Kerning in TEX is controlled by information in the tfm file, and is therefore outside the influence of the user. The tfm file can be edited, however (see Chapter ??).

在 TEX 中，字距调整由 tfm 文件中的信息控制，因此不受用户的影响。然而，可以编辑 tfm 文件（请参见第 ?? 章）。

The \kern command has (almost) nothing to do with the phenomenon of kerning; it is explained in Chapter ??.

\kern 命令（几乎）与字距调整现象无关；它将在第 ?? 章中解释。

4.3.3 Italic correction
斜体校正

The primitive control symbol \/ inserts the 'italic correction' of the previous character or ligature. Such a correction may be necessary owing to the definition of the 'bounding box' of a character. This box always has vertical sides, and the width of the character as TEX perceives it is the distance between these sides. However, in order to achieve proper spacing for slanted or italic typefaces, characters may very well project outside their bounding boxes. The italic correction is then needed if such an overhanging character is followed by a character from a non-slanting typeface.

原始控制符号 \/ 插入前一个字符或连字的“斜体校正”。这种校正可能是由于字符的“边界框”定义而导致的。这个边界框总是具有垂直边，而 TEX 感知到的字符宽度是这些边之间的距离。然而，为了实现倾斜或斜体字体的正确间距，字符可能会超出其边界框。如果这样一个突出的字符后跟一个非倾斜字体的字符，就需要斜体校正。

Compare for instance

例如，比较

'TEX has' to 'TEX has',

where the second version was typed as

第二个版本的输入为

{\italic\TeX\/} has

The size of the italic correction of each character is determined by font information in the font metrics file; for the Computer Modern fonts it is approximately half

the ‘overhang’ of the characters; see [?]. Italic correction is not the same as \fontdimen1, slant per point. That font dimension is used only for positioning accents on top of characters.

每个字符的斜体校正大小由字体度量文件中的字体信息确定；对于 Computer Modern 字体，它大约是字符的“突出部分”的一半；详见 [?]。斜体校正与 \fontdimen1（点数斜率）不同。该字体尺寸仅用于将重音定位在字符之上。

An italic correction can only be inserted if the previous item processed by TEX was a character or ligature. Thus the following solution for roman text inside an italic passage does not work:

只有在 TEX 最后处理的前一项是字符或连字时，才能插入斜体校正。因此，在斜体段落中插入罗马文本的以下解决方案无效：

{\italic Some text {\/\roman not} emphasized}

The italic correction has no effect here, because the previous item is glue.

这里斜体校正没有任何效果，因为前一项是粘连。

4.3.4 Ligatures
连写

Replacement of character sequences by ligatures is controlled by information in the tfm file of a font. Ligatures are formed from ⟨character⟩ commands: sequences such as fi are replaced by ‘fi’ in some fonts.

字符序列的连写替换受控于字体的 tfm 文件中的信息。在某些字体中，⟨character⟩ 命令可以形成连写：例如，fi 这样的序列在某些字体中会被替换为 ‘fi’。

Other ligatures traditionally in use are between ff, ffi, fl, and ffl; in some older works ft and st can be found, and similarly to the fl ligature fk and fb can also occur.

传统上，在 TEX 中还使用其他一些连写，如 ff、ffi、fl 和 ffl；在一些较旧的作品中还可以找到 ft 和 st，类似于 fl 连写，也可以出现 fk 和 fb。

Ligatures in TEX can be formed between explicit character tokens, \char commands, and ⟨chardef token⟩s. For example, the sequence \char`f\char`i is replaced by the ‘fi’ ligature, if such a ligature is part of the font.

TEX 中的连写可以在显式字符记号、\char 命令和 ⟨chardef token⟩ 之间形成。例如，如果字体中包含 fi' 连写，序列 \charf\char`i 会被替换为连写。

Unwanted ligatures can be suppressed in a number of ways: the unwanted ligature 'halflife' can for instance be prevented by

可以通过多种方式来阻止不需要的连写：例如，通过以下方式可以阻止不需要的连写 'halflife'：

```
half{}life, half{l}ife, half\/life, or half\hbox{}life
```

but the solution using italic correction is not equivalent to the others.

但使用斜体修正的解决方案与其他解决方案不等价。

4.3.5 Boundary ligatures 边界连写

Each word is surrounded by a left and a right boundary character (TEX3 only). This makes phenomena possible such as the two different sigmas in Greek: one at the end of a word, and one for every other position. This can be realized through a ligature with the boundary character. A \noboundary command immediately before or after a word suppresses the boundary character at that place.

每个单词都由左边界字符和右边界字符包围（仅限于 TEX3）。这使得希腊语中的两个不同的 sigma 成为可能：一个用于单词的末尾，另一个用于其他位置。这可以通过与边界字符进行连写来实现。在单词之前或之后立即使用 \noboundary 命令可以取消该位置处的边界字符。

In general, the ligature mechanism has become more complicated with the transition to TEX version 3; see [?].

一般来说，随着过渡到 TEX 第 3 版，连写机制变得更加复杂；请参阅 [?]。

4.3.6 Ligatures
连写

Replacement of character sequences by ligatures is controlled by information in the tfm file of a font. Ligatures are formed from ⟨character⟩ commands: sequences such as fi are replaced by 'fi' in some fonts.

字符序列的连写替换受控于字体的 tfm 文件中的信息。在某些字体中，⟨character⟩ 命令可以形成连写：例如，fi 这样的序列在某些字体中会被替换为 'fi'。

Other ligatures traditionally in use are between ff, ffi, fl, and ffl; in some older works ft and st can be found, and similarly to the fl ligature fk and fb can also occur.

传统上，在 TEX 中还使用其他一些连写，如 ff、ffi、fl 和 ffl；在一些较旧的作品中还可以找到 ft 和 st，类似于 fl 连写，也可以出现 fk 和 fb。

Ligatures in TEX can be formed between explicit character tokens, \char commands, and ⟨chardef token⟩s. For example, the sequence \char`f\char`i is replaced by the 'fi' ligature, if such a ligature is part of the font.

TEX 中的连写可以在显式字符记号、\char 命令和 ⟨chardef token⟩ 之间形成。例如，如果字体中包含 fi' 连写，序列 \charf\char`i 会被替换为连写。

Unwanted ligatures can be suppressed in a number of ways: the unwanted ligature 'halflife' can for instance be prevented by

可以通过多种方式来阻止不需要的连写：例如，通过以下方式可以阻止不需要的连写 'halflife'：

half{}life, half{l}ife, half\/life, or half\hbox{}life

but the solution using italic correction is not equivalent to the others.

但使用斜体修正的解决方案与其他解决方案不等价。

4.3.7 Boundary ligatures
边界连写

Each word is surrounded by a left and a right boundary character (TEX3 only). This makes phenomena possible such as the two different sigmas in Greek: one

at the end of a word, and one for every other position. This can be realized through a ligature with the boundary character. A \noboundary command immediately before or after a word suppresses the boundary character at that place.

每个单词都由左边界字符和右边界字符包围（仅限于 TEX3）。这使得希腊语中的两个不同的 sigma 成为可能：一个用于单词的末尾，另一个用于其他位置。这可以通过与边界字符进行连写来实现。在单词之前或之后立即使用 \noboundary 命令可以取消该位置处的边界字符。

In general, the ligature mechanism has become more complicated with the transition to TEX version 3; see [?].

一般来说，随着过渡到 TEX 第 3 版，连写机制变得更加复杂；请参阅 [?]。