1 Typography

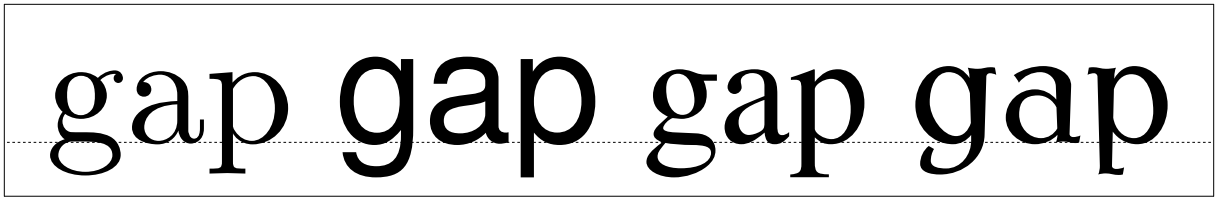
TODO: Rationale for this chapter: it should contain everything you need to know about **using** fonts, the title may change later. The following chapter will be about **defining** fonts.

1.1 Introduction

Throughout the millennia humans have developed and adapted methods for storing facts and thoughts on a variety of different mediums. A very efficient way of doing this is using logograms, like Chinese have done for ages. Another method is to represent each syllable in a word by a symbol, like the Japanese do when writing telegrams. However, the most common way of storing characters is by using a limited set of shapes representing basic sounds (a.k.a. phonemes). Such a collection is called an *alphabet*, and the shapes are called *letters*.

TEX is primarily meant for typesetting languages that use this third method. The other two methods can also be dealt with, but some extra effort is needed. In this chapter we will focus on languages that use alphabets, the other methods will be explained in later chapters.

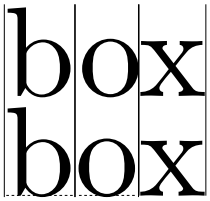
The shapes representing the characters that make up an alphabet are more or less standardized, and thereby can be recognized by readers even if their details differ. A collection of pictures matching character shapes is called a *font*, and the pictures in a font are called *glyphs*.



From left to right we see a Computer Modern font, a Helvetica lookalike, a Times Roman lookalike and the Antiqua Torunska font, all scaled to 60pt. As you can see, quite some variation is possible and when intermixed, the result is not always pleasing to look at. The term *fonts collection* refers to a set of fonts combined together in such a way that the overall appearance on a page looks good and reading is as comfortable as possible.

Even within a single font design there can be variations. In the example above we see a light, a bold, an italic, and a bold italic *alternative* of a single font. Such a set of fonts with the same basic design is known as a *font family*.

The distance between the individual glyphs in a word depends on the combinations of these glyphs. In the next sample, the gap between the b and the o as well as the distance between the o and the x is slightly altered. This is called kerning.



The font shown here is Computer Modern, the default TEX font. This font is designed by Donald Knuth. The Computer Modern has many kerning pairs, while the Palatino–like font that is used for most of the text in this manual has only a few.

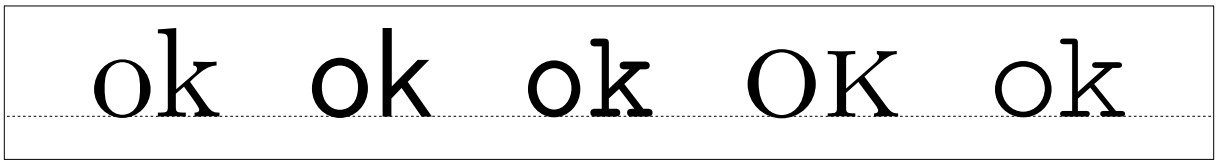
Micro–typography like kerning pairs are not to be altered by the user, it is part of the font design and the required data is stored inside the font file, together with the drawing routines for the actual pictures. It *is* possible for the user to alter fonts and interline spacing and some more aspects on the level of macro–typography. The choice of font is the main topic of this chapter.

There are many different methods that can be used to classify fonts. There are classification systems based on the period in which the style was first developed; on the characteristics of the font; or the font application, like a newspaper or a book. Often, classification systems mix these characteristics up to some point.

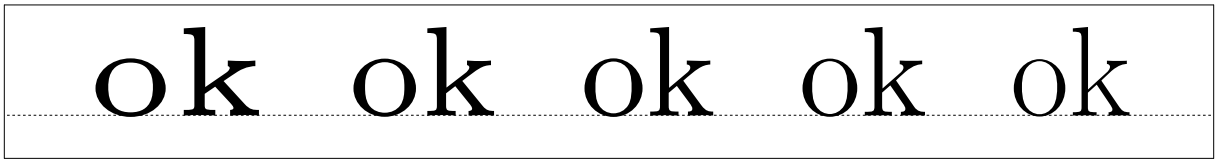
For example, the Computer Modern family can be classified as a 'modern' font. This is a classication that primarily indicates a period (late 18^{th} century), but it also implies a particular shape: 'modern' fonts have a high contrast between thick and thin strokes, and their stress axis is perfectly vertical.

At the same time, specific fonts in the Computer Modern family can be classified as 'serif' (glyphs strokes have embellishments at the end), 'sans serif' (shapes end abruptly), or 'monospaced' (all glyphs have the same width).

The Computer Modern family is in fact inspired by one font in particular: 'Modern 8a' by the Monotype corporation. Knuth implemented Computer Modern in METAFONT using parameters so that he could generate a whole collection of fonts all closely matching eachother in style. In ConTEXt you will normally use a reimplementation of Computer Modern using a more modern file format (Type 1 or OpenType). This new version is called 'Latin Modern', and also features an extended glyph set making it usable for languages that could not be typeset with Knuth's original fonts.



In this example we see five font styles of Latin Modern: the Roman, Sans, Typewriter, Smallcaps and Variable Typewriter. Computer Modern is one of the few font families that comes with dedicated design sizes. The example below shows the differences of a 5, 7, 9, 12 and 17 point design scaled up to 48 points. Such nuances in font size are seldom seen these days.



As explained earlier, the general appearance of a font style can be classified according to many schemes. In table 1.1 we see some examples of the naming of font styles that are often found together in a single document.

Serif	Sans	Mono
Regular	Support	TeleType
Roman	Sans	Type

Table 1.1 Some ways of classifying the styles in a font.

The top two series are normally used by typographers, the bottom series is what was traditionally used in plain TEX. In CONTEXT all three series of terms can be used because they are remapped to the same set of internal commands. As we will see, the command `\rm` is used to switch to a roman/serif/regular style, and `\tt` for switching to mono spaced or typewriter style, etcetera.

Text can be typeset in different font sizes. We often use the unit `pt` to specify the size. The availability of these font sizes are defined in definition files. Traditionally font designers used to design a glyph collection for each font size, but nowadays most fonts have a single design size of 10 points, or small set of sizes with names indicating their proposed use, like *caption*, *text*, and *display*.

In the next sections we will go into switching of font styles and fonts in your documents. Be warned that the font switching mechanism is rather complex. This is caused by the different modes like math mode and text mode in CONTEXT. If you want to be understand the mechanism fully, you will have to acquaint yourself with the concept of encoding vectors and obtain some knowledge on fonts and their peculiarities.

1.2 The mechanism

Font switching is one of the eldest features of CONTEXT because font switching is indispensable in a macro package. During the years extensions to the font switching mechanism were inevitable. We have chosen the following starting points during the development of this mechanism:

- To change a *style* must be easy, this means switching to: roman (serif, regular), sans serif (support), teletype (monospaced) etc. (`\rm`, `\ss`, `\tt` etc.)
- More than one *alternative* set of glyphs shapes must be available like slanted and bold (`\sl` and `\bf`).
- Different font *families* like Latin Modern Roman and Lucida Bright must be supported.

- It must be possbile to combine different families into font *collections*.
- Different sub– and superscripts must be available. These script sizes have to be retained across the switching of family, style and alternative.
- It should be possible to combine all of these requirements into a single definition unit called a *body font*.
- Changing the global font collection as wel as the size must also be easy, and so sizes between 8pt and 14.4pt must be available by default.

Before reading further, please stop for a moment to make sure you thoroughly comprehend the above paragraphs. ConTeXt's terminology probably differs from what you are accustomed to, especially if you were previously a LaTeX user.

1.3 Font switching

The mechanism to switch from one style to another is somewhat complex, not in the least because the terminology is a bit fuzzy. A quick recap: we call a collection of fonts, like Lucida or Computer Modern Roman, a *family*. Within such a family, the members can be grouped according to characteristics. Such a group is called a *style*. Examples of styles within a family are: 'roman', 'sans serif' and 'teletype'. We saw that there can be alternative classifications, but they all refer to the presence of serifs and the glyphs having equal widths. Within a style there can be *alternatives*, like 'boldface' and 'slanted'.

There are different ways to change into a new a style or alternative. You can use `\ss` to switch to a sans serif font style and `\bf` to get a bold alternative. When a different style is chosen, the alternatives adapt themselves to this style. Often we will typeset the document in one family and style. This is called the bodyfont.

Consistent use of commands like `\bf` and `\sl` in the text will automatically result in the desired bold and slanted altermatives when you change the family or style in the setup area of your input file.

1.3.1 Font style switching

Switching to another font style is done by one of five two-letter commands that are listed in table 1.2.

`\rm`	serif, regular, roman, rm
`\ss`	sans, support, sansserif, ss
`\tt`	mono, type, teletype, tt
`\hw`	handwritten, hw
`\cg`	calligraphic, cg
–	mm

Table 1.2 Font style switching commands

The 'handwritten' and 'calligraphic' font styles are sometimes useful when dealing with very elaborate document layout definitions. In the ConTeXt distribution, only the Lucida font family uses these styles, in any other font set they are simply ignored. You could use them in your own font setups if you so desire (see the next chapter for font setup definitions).

There is a sixth internal style that is only ever referenced as 'mm'. This style handles math fonts. It does not make sense to use this style directly so there is no command attached to it, but it is quite important internally so it makes sense to introduce it right away.

1.3.2 Font alternative switching

The alternatives within a style are given in table 1.3. Not all fonts have both italic and slanted or the bold alternatives of each. Some other fonts do not have small caps or only one set of digits. When an alternative is not known, ConTeXt will attempt to choose a suitable replacement automatically. For instance, the italic alternative may be used for if slanted is not available or vice versa.

`\bf`	bold
`\sl`	slanted
`\it`	italic
`\bs`	boldslanted, slantedbold
`\bi`	bolditalic, italicbold
`\sc`	smallcaps
`\os`	mediaeval
`\tf`	normal

Table 1.3 Font alternative switching commands and their keyword equivalents. With `\os` we tell ConTeXt that we prefer mediaeval or old–style numbers 139 over 139.

Besides these two-letter commands, there is a series of font selector commands with a suffix attached. Some examples of that are:

```
\tfx \bfx \slx \itx
\tfa \tfb \tfc \tfd \tfxx
```

Each of the ordered alphabetic suffixes `a`, `b`, ... select a somewhat larger actual font than the previous one. The `x` and `xx` suffixes select smaller and yet smaller versions.

`\bfx`	smallbold
`\slx`	smallslanted
`\itx`	smallitalic
`\bsx`	smallboldslanted, smallslantedbold
`\bix`	smallbolditalic, smallitalicbold
`\tfx`	small, smallnormal

Table 1.4 Small alternative switching commands and their keyword equivalents.

Besides the 'small' switches that are mentioned in table 1.4, it depends on the completeness of the font definition files whether commands like `\ita`, `\bfxx`, `\bfc`, etc. are available. For

the core ConTeXt fonts, you can count on at least \tfa, \tfb, \tfc, \tfd, and \tfxx being defined. For the others, just try and see what happens.

When you have chosen a larger charactersize, for example \tfb, then \tf equals \tfb, \bf equals \bfb, etc. This method is almost always preferable over returning to the original character size, but it may catch you off-guard.

More generic font scaling commands are also available:

```
\tx \txx
\setsmallbodyfont \setbigbodyfont
```

The command \tx adapts itself to both the style and the alternative. This command is rather handy when one wants to write macros that act like a chameleon. Going one more step smaller, is possible too: \txx. Using \tx when \tx is already given, is equivalent to \txx.

The commands \setsmallbodyfont and \setbigbodyfont switch to the 'small' and 'big' body font sizes. These relative sizes are defined via the 'body font environment', see section 1.6.4.

The various commands will adapt themselves to the actual setup of font and size. For example:

```
{\rm test {\sl test} {\bf test} \tfc test {\tx test} {\bf test}}
{\ss test {\sl test \tx test} {\bf test \tx test}}
```

will result in:

test *test* **test** test test **test**
test *test* *test* **test** **test**

When the \rm style is active, ConTeXt will interpret the command \tfd as if it was \rmd, when the style \ss is active, \tfd as is treated as \ssd. All default font setups use tf–setups so they will automatically adapt to the current font style.

The remainder of this paragraph is for completeness' sake only. Use of the following commands in new documents is discouraged.

Frequent font switching leads to longer processing times. When no sub- or superscripts are used and you are very certain what font you want to use, you can perform fast font switches with: \rmsl, \ssbf, \tttf, etc.

The plain TeX compatible font switches \vi, \vii, \viii, \ix, \x, and \xii are also defined, these have local effects like \tfx and \tfa.

1.3.3 Switching font styles in setup commands

A number of ConTeXt commands use the parameter style to set the used font. The parameter mechanism is rather flexible so that within the parameter style you can use any of the font switching commands like or bf or \switchtobodyfont, but also a number of keywords like

```
normal  bold  slanted  boldslanted  italic  bolditalic  type
small  smallbold  smallslanted  ...  smallitalic  ...  smalltype
capital
```

Most of these keywords have alread been listed in the tables 1.3 and 1.4, but a few predefined ones are still missing. These are displayed in table 1.5, together with the commands they execute. As is normal in ConTeXt, you can extend the list of accepted keywords by defining your own. This will be explained in section ?? in the next chapter.

`\tt`	type, mono
`\ttx`	smalltype
`\ss`	sans, sansserif
`\ss \bf`	sansbold
`\setsmallbodyfont`	smallbodyfont
`\setbigbodyfont`	bigbodyfont
`\smallcapped`	cap, capital
`\WORD`	WORD

Table 1.5 Remaining font alternative keywords.

1.4 Emphasize

Within most macropackages the command `\em` is available. This command behaves like a chameleon which means that it will adapt to the actual typeface. In ConTeXt `\em` has the following characteristics:

- a switch to *slanted* or *italic* is possible
- a switch within `\bf` results in ***bold slanted*** or ***bold italic*** (when available)
- a so called *italic correction* is performed automatically (`\/`)

The bold italic or bold slanted characters are supported only when `\bs` and `\bi` are available.

```
The mnemonic {\em em} means {\em emphasis}.
{\em The mnemonic {\em em} means {\em emphasis}.}
{\bf The mnemonic {\em em} means {\em emphasis}.}
{\em \bf The mnemonic {\em em} {\em emphasis}.}
{\it The mnemonic em {\em means \bf emphasis}.}
{\sl The mnemonic em {\em means \bf emphasis}.}
```

This results in:

The mnemonic *em* means *emphasis*.
The mnemonic em *means* emphasis.
The mnemonic *em* means *emphasis*.
***The mnemonic* em emphasis.**
The mnemonic em means **emphasis**.
The mnemonic em means **emphasis**.

The advantage of the use of `\em` over `\it` and/or `\sl` is that consistent typesetting is enforced.

By default emphasis is set at *slanted*, but in this text it is set at *italic*. This setting is made via `\setupbodyfontenvironment`, see section 1.6.4 for more details:

```
\setupbodyfontenvironment[default][em=italic]
```

1.5 Capitals

Words and abbreviations can be typeset in capitals. Both small and big characters are converted into capitals. When `\cap` is used to typeset a capital the size is that of an `\tx`. When we switch to slanted (`\sl`), bold (`\bf`), etc. the capital letter will also change. Since `\cap` has a specific meaning in math mode, the formal implementation is called `\smallcapped`. However in text mode one can use `\cap`.

```
\cap {.*.}

*   TEXT
```

```
\Cap {.*.}

*   TEXT
```

```
\CAP {.*.}

*   TEXT
```

```
\Caps {.. .*. ..}

*   WORD
```

The first command converts all letters to a capital. We advise you not to type capital letters in your source file because real small caps distinguishes between small and big letters.

```
Capitals for \cap {UK} are \cap {OK} and capitals for \cap {USA} are
okay. But what about capitals in \cap {Y2K}.
```

this results in:

Capitals for UK are OK and capitals for USA are okay. But what about capitals in Y2K.

A `\cap` within a `\cap` will not lead to any problems:

```
\cap {People that have gathered their \cap {capital} at the cost of other
people are not seldom \nocap {decapitated} in revolutionary times.}
```

or:

PEOPLE THAT HAVE GATHERED THEIR CAPITAL AT THE COST OF OTHER PEOPLE ARE NOT SELDOM decapitated IN REVOLUTIONARY TIMES.

In this example we see that `\cap` can be temporarily revoked by `\nocap`.

```
\nocap {.*.}

*   TEXT
```

The command \Cap changes the first character of a word into a capital and \CAP changes letters that are preceded by \\ into capital letters. With \Caps you can change the first character of several words into a capital letter.

```
\setupcapitals [..,.*=.,..]

*   title  =  yes  no
    sc     =  yes  no
```

With this command the capital mechanism can be set up. The key sc=yes switches to real Small Caps. With title we determine whether capitals in titles are changed.

Next to the former \cap–commands we have:

```
\Word {.*.}

*   WORD
```

and

```
\Words {.. .*. ..}

*   WORD
```

These commands switch the first characters of words into capitals. All characters in a word are changed with:

```
\WORD {.*.}

*   WORD
```

We end this section with real small capitals. When these are available the real small caps \sc are preferred over the pseudo–capital in abbreviations and logos.

```
In a manual on \TeX\ and Con\TeX t  there is always the question whether
to type \cap{\TeX} and \cap{Con\TeX t} or {\sc \TeX} and {\sc Con\TeX t}.
Both are defined as a logo in the style definition so we type \type {\TEX}
and \type {\CONTEXT}, which come out as \TEX\ and \CONTEXT.
```

Results in:

In a manual on TEX and ConTEXt there is always the question whether to type TEX and CONTEXT or TEX and CONTEXT. Both are defined as a logo in the style definition so we type `\TEX` and `\CONTEXT`, which come out as TEX and CONTEXT.

IT IS ALWAYS POSSIBLE TO TYPESET TEXT IN SMALL CAPITALS. HOWEVER, REALIZE THAT LOWER CASE CHARACTERS DISCRIMINATE MORE AND MAKE FOR AN EASIER READ.

An important difference between `\cap` and `\sc` is that the last command is used for a specific designed font type. The command `\cap` on the other hand adapts itself to the actual typeface: *KAP*, **KAP**, ***KAP***, etc.

Some typesetting packages stretch words (inter character spacing) to reach an acceptable alignment. In CONTEXT this not supported. On purpose! Words in titles can be stretched by:

```
\stretched {.*.}

*   WORD
```

```
\hbox to \hsize {\stretched{there\\is\\much\\stretch\\in ...}}
\hbox to 20em   {\stretched{... and\\here\\somewhat\\less}}
```

With `\\` we enforce a space (`{}` is also allowed).

t h e r e i s m u c h s t r e t c h i n . . .
. . . a n d h e r e s o m e w h a t l e s s

These typographically non permitted actions are only allowed in heads. The macros that take care of stretching do this by processing the text character by character.

We will not go into the typographical sins of underlining. These commands are discussed in section ?? (“??”).

1.6 Selecting bodyfonts

The bodyfont (main font), font style and size is set up with:

```
\setupbodyfont [...,*,...]

*    IDENTIFIER  serif  regular  roman  sans  support  sansserif  mono  type  teletype
    handwritten  calligraphic  5pt  ...  12pt
```

The various identifiers

In a running text a temporary font switch is done with the command:

```
\switchtobodyfont [...,*,...]

*   5pt  ...  12pt  small  big
```

This command doesn’t change the bodyfont in headers and footers. With `small` and `big` you switch to a smaller or larger font.

In most cases, the command `\setupbodyfont` is only used once: in the style definition, and font switching inside the document is done with `\switchtobodyfont`. Don't confuse these two because that may lead to some rather strange but legitimate effects.

1.6.1 Body font sizes

Body font sizes actually consist of two components. Of course if you specify a size it directly specifies the size at which the main font is loaded, but a number of indirect parameters have to taken care of as well. Think of things like the font size used in headers, footers, footnotes, sub– and superscripts, as well as the interline space and a few others.

This is why in ConTeXt there is the concept of a *body font environment* (expressed as a dimension), and that is what you pass as an argument to `\setupbodyfont` or `\switchtobodyfont`. The definitions as presented above use the indication `5pt ... 12pt` for the body font environment, but actually any dimension is acceptable.

The most frequently used sizes are predefined as body font environments: 4pt ... 12pt, 14.4pt, and 17.3pt. But when you use a different, not-yet-defined size specification —for example for a titlepage— ConTeXt will define a body font environment for that size automatically. While doing so, ConTeXt normally works with a precision of 1 decimal to prevent unnecessary loading of fontsizes with only small size differences.

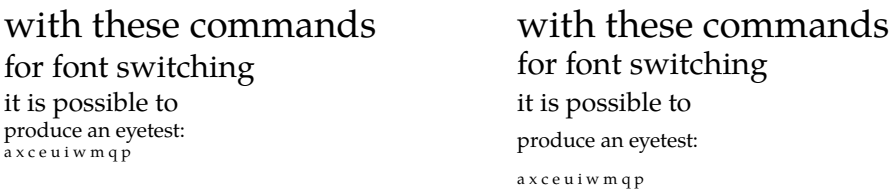
Be warned that in this case, the results may be a less than ideal. The reason is that ConTeXt not just has to load the actual font, but it also has to guess at the various other settings like the relative font sizes and the interline space. It does so by using the values from the nearest smaller body font environment is that is already defined.

You can extend the list of predefined body font environments and even alter the precision in body font matching. See the section 1.6.4 for detailed information about how to tweak or define your own body font sizes.

To end this section, the example below demonstrates how the interline space is adapted automatically, when changing the size of the bodyfont. Consider this input:

```
{\switchtobodyfont[14.4pt] with these commands \par}
{\switchtobodyfont[12pt] for font switching  \par}
{\switchtobodyfont[10pt]  it is possible to   \par}
{\switchtobodyfont[8pt]  produce an eyetest: \par}
{\switchtobodyfont[6pt]  a x c e u i w m q p \par}
```

The actual ConTeXt behaviour is shown below on the left. On the right you can see what would have happened if the interline space were not automatically adapted.



1.6.2 Body font identifiers

In the definition block of `setupbodyfont` there was a list of words given besides the special marker `IDENTIFIER`. These words are the symbolic ConTeXt names for the font styles that we

ran into earlier, with a few aliases so that you do not have to worry about the actual naming convention used. The symbolic names are mapped to two-letter internal style abbreviations that are used internally, see table 1.2 for an overview.

Although the macro syntax does not say so, you can use two-letter internal style abbreviations (`ss`, `rm`) as well as the longer names, if you prefer.

We have seen already that there are other and easier ways to switch the font style, so if `\setupbodyfont` could only be used for this purpose it would not be all that useful. But luckily there is more: the optional `IDENTIFIER` can be a 'body font name' (aka 'typeface'). Such names have to be predefined, perhaps in a font support file, or simply on earlier lines in the style definition.

A 'typeface' is a symbolic name that links a single font style to actual font families. Such symbolic names are typically grouped together in a definition block that sets up values that link the four styles `\rm`, `\ss`, `\tt` and `\mm` to fonts in a 'font collection', and such definition blocks are called 'typescripts'.

ConTEXt expects you to define your own font setups, but there are quite a few examples predefined in various typescript files. Not all of those are perpetually loaded, so you usually have to execute a typescript explicitly to get the typeface names predefined. To this end, typescripts *themselves* also have names.

Executing a typescript is done by `\usetypescript`. We will get back to `\usetypescript` later because it is in fact a very flexible command, but let's discuss simple usage first.

A typical input sequence for selecting the predefined 'palatino' set of typefaces in MkII will look like this:

```
\usetypescript[palatino][ec]
\setupbodyfont[palatino,12pt]
```

In this example the typescript named `palatino` is asked for in the `ec` font encoding, and that defines a set of typefaces under the name `palatino`. These are then used by `\setupbodyfont` and eventually this makes pdfTEX load the free Type 1 font URW Palladio in the correct encoding. URW Palladio is a font that looks a lot like the commercial font Linotype Palatino by Hermann Zapf, which explains the name of the typescript and typefaces.

Font encodings will be handled fully in the section 1.11. For now, please take for granted the fact that pdfTEX needs a second argument to `\usetypescript` that specifies an encoding name, and that there is a fixed set of acceptable names that depends on the typescript that is being requested.

In XETEX and MkIV the situation is a little bit different because fonts are reencoded to match Unicode whenever that is possible. That in turn means that XETEX and MkIV prefer to use OpenType fonts over Type 1 fonts, so different typescript definitions are used behind the scenes, and the second argument to `\usetypescript` becomes optional.

For example,

```
\usetypescript[palatino]
\setupbodyfont[palatino,12pt]
```

will make XETEX and LuaTEX load the OpenType font Pagella. This is a free font from the TEX Gyre project, that also looks just like the commercial font Linotype Palatino. You may as well

leave the second argument in place: while it will always be ignored by LuaTEX, XETEX will actually use that encoding if the typescript uses Type 1 fonts instead of the more modern OpenType or TrueType font formats.

All predefined typescripts attach meaning to (at least) the three basic text font styles, so you can e.g. do this:

```
\usetypescript[times][texnansi]
\setupbodyfont[times,sans,12pt]
```

and end up using the OpenType font TEX Gyre Heros or the Type 1 font URW Nimbus Sans L. Both fonts are very similar in appearance to Linotype Helvetica, by the way.

The typescripts that come with the ConTEXt distribution are placed in source files that have names that start with `type-`. Some of these files are automatically loaded, but most have to be loaded explicitly. Here is a list

File	Loaded by pdfTEX	Loaded by XETEX	Loaded by MkIV	Description
type-akb	no	no	no	PostScript fonts using psnfss names (Type 1)
type-buy	no	no	no	Various commercial fonts (Type 1)
type-cbg	no	no	no	Greek free fonts (Type 1)
type-cow	no	no	no	The ConTEXt cow font (Type 1)
type-exp	no	no	no	Commercial Zapf fonts (OpenType)
type-fsf	no	no	no	Commercial Fontsite 500 fonts (Type 1)
type-ghz	no	no	no	Commercial Zapf fonts (Type 1)
type-gyr	no	no	no	The TEX Gyre project fonts (Type 1)
type-hgz	no	no	no	Commercial Zapf fonts (OpenType)
type-msw	no	no	no	Fonts that come with Microsoft Windows (Type 1)
type-omg	no	no	no	Omega free fonts (Type 1)
type-one	yes	no	no	Various free fonts (Type 1)
type-otf	no	yes	yes	Various free fonts (OpenType)
type-xtx	no	yes	no	Fonts that come with MacOSX (OpenType)

Explicit loading one of those files is done via the macro `\usetypescriptfile`.

The predefined typescripts, the typefaces they define, the files they are contained in inside the ConTEXt distribution, and the encodings they support in MkII mode are listed in table 1.6. In the following section there is a table (1.7) that explains for each typescript what font set it attaches to each of the font styles.

For example, the following

```
\usetypescriptfile[type-buy]
\usetypescript[lucida][texnansi]
\setupbodyfont[lucida,12pt]
```

will make pdfTEX use the Lucida Bright font family. Because this is a commercial font, this only works correctly if you have actually bought and installed the fonts. This uses the `texnansi` encoding because that is the preferred encoding of the actual fonts.

This is a good moment to explain a little trick: because the various `type-xxx` files define the building blocks for typescripts as well as the actual typescripts, it is sometimes possible to alter the effect of a typescript by loading an extra typescript file. For example,

Typescript	Typeface	File	Encodings
OmegaArab	omarb	type-omg	(unspecified)
OmegaLGC	omlgc	type-omg	(unspecified)
antykwa-torunska	antykwa	type-one, type-otf	texnansi,ec,8r,uc,t2a
cbgreek	cbgreek	type-cbg	(unspecified)
cbgreek-all	cbgreek-all	type-cbg	(unspecified)
cbgreek-medium	cbgreek-medium	type-cbg	(unspecified)
cow	cow	type-cow	default
fourier	fourier	type-one	ec
iwona	iwona	type-one, type-otf	texnansi,ec,8r,uc,t2a
iwona-heavy	iwona-heavy	type-one, type-otf	texnansi,ec,8r,uc,t2a
iwona-light	iwona-light	type-one, type-otf	texnansi,ec,8r,uc,t2a
iwona-medium	iwona-medium	type-one, type-otf	texnansi,ec,8r,uc,t2a
lucida	lucida	type-buy	texnansi,ec,8r,uc
lucidabfm	lucida	type-buy	texnansi,ec,8r,uc
lucidabfm	lucidabfm	type-buy	texnansi,ec,8r,uc
lucidaboldmath	lucida	type-buy	texnansi,ec,8r,uc
lucidaboldmath	lucidaboldmath	type-buy	texnansi,ec,8r,uc
modern	modern	type-one, type-otf	texnansi,ec,qx,t5,default
modern-base	modern	type-one, type-otf	texnansi,ec,qx,t5,default,t2a,t2b,t2c,x2
modernvariable	modernvariable	type-one, type-otf	texnansi,ec,qx,8r,t5
optima	optima	type-one	texnansi,ec,qx
optima	optima	type-ghz	texnansi,ec,qx
optima-nova	optima	type-ghz, type-hgz	texnansi,ec
optima-nova-os	optima-os	type-ghz, type-hgz	texnansi,ec
palatino	palatino	type-hgz	(cannot be used in MkII)
palatino	palatino	type-one, type-otf	texnansi,ec,qx,8r,t5,uc
palatino-informal	palatino-informal	type-hgz	(cannot be used in MkII)
palatino-light	palatino-light	type-exp	(cannot be used in MkII)
palatino-medium	palatino-medium	type-exp	(cannot be used in MkII)
palatino-normal	palatino-normal	type-exp	(cannot be used in MkII)
palatino-nova	palatino	type-hgz	(cannot be used in MkII)
palatino-sans	palatino	type-hgz	(cannot be used in MkII)
postscript	postscript	type-one, type-otf	texnansi,ec,qx,8r,t5,uc
sheep	sheep	type-cow	default
times	times	type-one, type-otf	texnansi,ec,qx,8r,t5,uc

Table 1.6 The typescripts. Typescripts that use commercial fonts are typeset in bold.

```
\usetypescriptfile[type-gyr]
\usetypescript[palatino][ec]
\setupbodyfont[palatino,12pt]
```

will result in pdfTeX using the Type 1 font Pagella from the TeX Gyre project instead of the older and less complete URW Palladio, because the definition of the building blocks for the `palatino` typescript that is in the `type-gyr` file overwrites the preloaded definition from the `type-one` file.

Two of the files in the ConTeXt distribution exist precisely for this reason:

`type-gyr.tex`
: maps the typical PostScript font names for the free URW fonts to the TeX Gyre set;

`type-akb.tex`
: maps the same names to the commercial Adobe fonts.

For the defintions in the second file to work, you also need to execute an extra typescript:

```
\usetypescriptfile[type-akb]
\usetypescript[adobekb][ec]
\usetypescript[palatino][ec]
\setupbodyfont[palatino,12pt]
```

1.6.3 Typeface definitions

Defining a typeface goes like this:

```
\starttypescript [palatino] [texnansi,ec,qx,t5,default]

\definetypeface[palatino] [rm] [serif] [palatino] [default]
\definetypeface[palatino] [ss] [sans]  [modern]   [default] [rscale=1.075]
\definetypeface[palatino] [tt] [mono]  [modern]   [default] [rscale=1.075]
\definetypeface[palatino] [mm] [math]  [palatino] [default]

\stoptypescript
```

This defines a typescript named `palatino` in five different encodings. When this typescript is executed via `\usetypescript`, it will define four typefaces, one of each of the four basic styles `rm`, `ss`, `tt`, and `mm`.

The third and fourth arguments to `\definetypeface` are pointers to already declared font sets, these are defined elsewhere. Table 1.7 gives the full list of predefined typescripts (the first argument of `\starttypescript`) and font sets that are attached to the styles (the third and fourth argument of each `\definetypeface`).

The names in the third argument (like `serif` and `sans`) do *not* have the same meaning as the names used in `\setupbodyfont`. Inside `\setupbodyfont`, they were keywords that were internally remapped to one of the two-letter internal styles. Inside `\definetypeface`, they are nothing more than convenience names that are attached to a group of fonts by the person that wrote the font definition. They only reflect a grouping that the person believed that could be a single font style. Oftentimes, these names are identical to the official style keywords, just as the typescript and typeface names are often the same, but there can be (and are) different names altogether.

How to define your own font sets will be explained in the next chapter, but there are quite a few predefined font sets that come with ConTeXt; these are all listed in the four tables 1.8, 1.9, 1.10, and 1.11.

For everything to work properly in MkII, the predefined font sets also have to have an encoding attached, you can look those up in the relevant tables as well.

The fifth argument to `\definetypeface` specifies specific font size setups (if any), these will be covered in section `??` in the next chapter. Almost always, specifying `default` will suffice.

The optional sixth argument is used for tweaking font settings like the specification of font features or adjusting parameters. In this case, the two `modern` font sets are loaded with a small magnification, this evens out the visual heights of the font styles.

There are four possible keys in the sixth argument:

Typescript	Style rm	Style ss	Style tt	Style mm
OmegaArab	omega naskh	–	–	–
OmegaLGC	omega	–	omega	–
antykwa-torunska	antykwa-torunska	modern	modern	antykwa-torunska
cbgreek	cbgreek	cbgreek	cbgreek	–
cbgreek-all	cbgreek	cbgreek	cbgreek	–
cbgreek-medium	cbgreek	cbgreek	cbgreek	–
cow	cow	cow serif	modern	cow
fallback	modern	modern	modern	modern
fourier	fourier	modern	modern	fourier
iwona	modern	iwona	modern	iwona
iwona-heavy	modern	iwona-heavy	modern	iwona-heavy
iwona-light	modern	iwona-light	modern	iwona-light
iwona-medium	modern	iwona-medium	modern	iwona-medium
lucida	lucida	lucida	lucida	lucida
lucidabfm	lucida	lucida	lucida	lucida bfmath
lucidaboldmath	lucida	lucida	lucida	lucida boldmath
modern	modern	modern	modern	modern
modern-base	(computer-)modern	(computer-)modern	(computer-)modern	(computer-)modern
modernvariable	simple	modern	modern	modern
optima	palatino	optima-nova	modern	palatino
optima-nova	optima-nova sans	optima-nova	latin-modern	latin-modern
optima-nova-os	optima-nova-os sans	optima-nova-os	latin-modern	latin-modern
palatino	palatino-nova	palatino-sans	latin-modern	latin-modern
palatino	palatino	modern	modern	palatino
palatino-informal	palatino-nova	palatino-informal	latin-modern	latin-modern
palatino-light	palatino-nova	palatino-sans-light	latin-modern	latin-modern
palatino-medium	palatino-nova	palatino-sans-medium	latin-modern	latin-modern
palatino-normal	palatino-nova	palatino-sans-normal	latin-modern	latin-modern
palatino-nova	palatino-nova	palatino-sans	latin-modern	latin-modern
palatino-sans	palatino-nova	palatino-sans	latin-modern	latin-modern
postscript	times	helvetica	courier	times
sheep	sheep	sheep serif	modern	sheep
times	times	helvetica	modern	times

Table 1.7 The typescripts.
Unless stated otherwise, style **rm** uses a group named serif, style **ss** uses sans, style **tt** uses mono, and style **mm** uses math. A single dash in a cell means that the typescript does not define that style, you should refrain from using the style. The lucida, lucidabfm, and lucidaboldmath typescripts also define **hw** and **cg** as 'lucida handwring' and 'lucida calligraphy'. The modern-base typescript switches back to computer-modern for a few legacy encodings: t2a, t2b, t2c, and x2.

key	default value	explanation
rscale	1	a scaling factor for this typescript relative to the selected body font size
encoding	`\defaultencoding`	the encoding for the typeface, normally inherited from the typescript automatically
features		this applies a predefined font feature set (see section 1.7)
text		sets up the forced math text style

A note for the lazy: if the sixth argument is not given and the fifth argument happens to be `default`, then the fifth argument can be omitted as well.

Identifier	file	Encodings	Supported styles
modern	type-one	ec, qx, texnansi, t5, uc	serif, sans, mono, math, boldmath, bfmath
latin-modern	type-one	ec, qx, texnansi, t5, uc	serif, sans, mono, math, boldmath, bfmath
computer-modern	type-one	cyr, lcy, t2a, t2b, t2c, x2	serif, sans, mono, math, boldmath, bfmath
simple	type-one	– synonyms only –	serif
concrete	type-one	– hardcoded –	serif
euler	type-one	– hardcoded –	math, boldmath, bfmath
ams	type-one	– hardcoded –	math
fourier	type-one	ec	math, serif
courier	type-one	8r, ec, qx, texnansi, t5	mono
helvetica	type-one	8r, ec, qx, texnansi, t5	sans
times	type-one	8r, ec, qx, texnansi, t5, uc	serif, math
palatino	type-one	8r, ec, qx, texnansi, t5, uc	serif, math
bookman	type-one	8r, ec, qx, texnansi, t5	serif
schoolbook	type-one	8r, ec, texnansi, t5	serif
chancery	type-one	8r, ec, qx, texnansi	calligraphy
charter	type-one	8r, ec, texnansi	serif
utopia	type-one	ec, texnansi	serif
antykwa-torunska	type-one	texnansi, qx, t5, ec, t2a/b/c, greek	serif, math
antykwa-torunska-light	type-one	texnansi, qx, t5, ec, t2a/b/c, greek	serif, math
antykwa-torunska-cond	type-one	texnansi, qx, t5, ec, t2a/b/c, greek	serif, math
antykwa-torunska-lightcond	type-one	texnansi, qx, t5, ec, t2a/b/c, greek	serif, math
antykwa-poltawskiego	type-one	8r, ec, texnansi	serif
iwona	type-one	ec, qx, texnansi, t5	sans, math
iwona-light	type-one	ec, qx, texnansi, t5	sans, math
iwona-medium	type-one	ec, qx, texnansi, t5	sans, math
iwona-heavy	type-one	ec, qx, texnansi, t5	sans, math
iwona-cond	type-one	ec, qx, texnansi, t5	sans
iwona-light-cond	type-one	ec, qx, texnansi, t5	sans
iwona-medium-cond	type-one	ec, qx, texnansi, t5	sans
iwona-heavy-cond	type-one	ec, qx, texnansi, t5	sans
kurier	type-one	ec, qx, texnansi, t5	sans, math
kurier-light	type-one	ec, qx, texnansi, t5	sans, math
kurier-medium	type-one	ec, qx, texnansi, t5	sans, math
pagella	type-gyr	ec, texnansi, qx, t5, t2a/b/c, l7x	serif
palatino	type-gyr	ec, texnansi, qx, t5, t2a/b/c, l7x	serif
termes	type-gyr	ec, texnansi, qx, t5, t2a/b/c, l7x	serif
times	type-gyr	ec, texnansi, qx, t5, t2a/b/c, l7x	serif
bonum	type-gyr	ec, texnansi, qx, t5, t2a/b/c, l7x	serif
bookman	type-gyr	ec, texnansi, qx, t5, t2a/b/c, l7x	serif
schola	type-gyr	ec, texnansi, qx, t5, t2a/b/c, l7x	serif
schoolbook	type-gyr	ec, texnansi, qx, t5, t2a/b/c, l7x	serif
heros	type-gyr	ec, texnansi, qx, t5, t2a/b/c, l7x	sans
helvetica	type-gyr	ec, texnansi, qx, t5, t2a/b/c, l7x	sans
adventor	type-gyr	ec, texnansi, qx, t5, t2a/b/c, l7x	sans
cursor	type-gyr	ec, texnansi, qx, t5, t2a/b/c, l7x	mono
courier	type-gyr	ec, texnansi, qx, t5, t2a/b/c, l7x	mono
omega	type-omg	– hardcoded –	naskh, serif, mono
cbgreek	type-cbg	– hardcoded –	serif, sans, mono
cbgreek-medium	type-cbg	– hardcoded –	serif, sans, mono
cbgreek-all	type-cbg	– hardcoded –	serif, sans, mono
cow	type-cow	– hardcoded –	math, serif
sheep	type-cow	– hardcoded –	math, serif

Table 1.8 The predefined body font identifiers for free Type 1 and METAFONT fonts

Identifier	file	Encodings	Supported styles
lucida	type-buy	8r, ec, texnansi, uc	serif, sans, mono, handwriting, calligraphy, math, boldmath, bfmath, casual, fax
informal	type-buy	– hardcoded –	casual, math
officina	type-buy	8r, ec, texnansi	serif, sans
meta	type-buy	8r, ec, texnansi	serif, sans, expert
meta-medium	type-buy	8r, ec, texnansi	sans
meta-lf	type-buy	8r, ec, texnansi	sans
meta-book	type-buy	8r, ec, texnansi	sans
meta-book-lf	type-buy	8r, ec, texnansi	sans
meta-bold	type-buy	8r, ec, texnansi	sans
meta-bold-lf	type-buy	8r, ec, texnansi	sans
meta-normal	type-buy	8r, ec, texnansi	sans
meta-normal-lf	type-buy	8r, ec, texnansi	sans
meta-medium	type-buy	8r, ec, texnansi	sans
meta-medium-lf	type-buy	8r, ec, texnansi	sans
meta-black	type-buy	8r, ec, texnansi	sans
meta-black-lf	type-buy	8r, ec, texnansi	sans
univers	type-buy	8r, ec, texnansi	sans
univers-light	type-buy	8r, ec, texnansi	sans
univers-black	type-buy	8r, ec, texnansi	sans
mendoza	type-buy	8r, ec, texnansi	serif
frutiger	type-buy	8r, ec, texnansi	sans
kabel	type-buy	8r, ec, texnansi	sans
thesans	type-buy	8r, ec, texnansi	sans, mono, expert
sabon	type-buy	8r, ec, texnansi	serif
stone	type-buy	ec, texnansi	serif, sans
stone-oldstyle	type-buy	– synonyms only –	serif, sans
industria	type-buy	ec, texnansi	sans
bauhaus	type-buy	ec, texnansi	sans
swift	type-buy	ec, texnansi	serif
swift-light	type-buy	– synonyms only –	serif
syntax	type-buy	ec, texnansi	sans
linoletter	type-buy	ec, texnansi	serif
zapfino	type-ghz	8r, ec, texnansi	serif, handwriting
palatino-sans-light	type-exp	texnansi, ec	sans
palatino-sans-normal	type-exp	texnansi, ec	sans
palatino-sans-medium	type-exp	texnansi, ec	sans
opus	type-fsf	8r, ec, texnansi	sans
typewriter	type-fsf	8r, ec, texnansi	mono
garamond	type-fsf	8r, ec, texnansi	serif
optima	type-ghz	8r, ec, texnansi	sans
optima-nova	type-ghz	8r, ec, texnansi	sans
optima-nova-os	type-ghz	8r, ec, texnansi	sans
optima-nova-light	type-ghz	8r, ec, texnansi	sans
optima-nova-medium	type-ghz	8r, ec, texnansi	sans
palatino	type-ghz	8r, ec, texnansi	serif
palatino-nova	type-ghz	8r, ec, texnansi	serif
palatino-nova-os	type-ghz	8r, ec, texnansi	serif
palatino-nova-light	type-ghz	8r, ec, texnansi	serif
palatino-nova-medium	type-ghz	8r, ec, texnansi	serif
aldus-nova	type-ghz	8r, ec, texnansi	serif
melior	type-ghz	8r, ec, texnansi	serif
verdana	type-msw	texnansi	sans
arial	type-msw	texnansi	sans

Table 1.9 The predefined body font identifiers for commercial Type 1 fonts

Identifier	file	Supported styles
modern	type-otf	serif, sans, mono, math, boldmath, bfmath
latin-modern	type-otf	serif, sans, mono, math, boldmath, bfmath
modern-vari	type-otf	mono
latin-modern-vari	type-otf	mono
modern-cond	type-otf	mono
latin-modern-cond	type-otf	mono
computer-modern	type-otf	serif, sans, mono, math, boldmath, bfmath
concrete	type-otf	serif
euler	type-otf	math, boldmath, bfmath
ams	type-otf	math
pagella	type-otf	serif
termes	type-otf	serif
bonum	type-otf	serif
schola	type-otf	serif
chorus	type-otf	serif
heros	type-otf	sans
adventor	type-otf	sans
cursor	type-otf	sans
palatino	type-otf	serif, math
times	type-otf	serif, math
bookman	type-otf	serif
schoolbook	type-otf	serif
chancery	type-otf	calligraphy
helvetica	type-otf	sans
courier	type-otf	mono
antykwa-torunska	type-otf	serif, math
antykwa-torunska-light	type-otf	serif, math
antykwa-torunska-cond	type-otf	serif, math
antykwa-torunska-lightcond	type-otf	serif, math
antykwa-poltawskiego	type-otf	serif
iwona-light	type-otf	sans, math
iwona	type-otf	sans, math
iwona-medium	type-otf	sans, math
iwona-heavy	type-otf	sans, math
iwona-cond	type-otf	sans
iwona-light-cond	type-otf	sans
iwona-medium-cond	type-otf	sans
iwona-heavy-cond	type-otf	sans
kurier	type-otf	sans, math
kurier-light	type-otf	sans, math
kurier-medium	type-otf	sans, math
charter	type-otf	serif
gentium	type-xtx	serif

Table 1.10 The predefined body font identifiers for free Unicode (Opentype) fonts

In table 1.11 you will see three very special items: `Xserif`, `Xsans` and `Xmono`. These belong to a special XeTeX-only trick called 'wildcard typescripts'.

XeTeX offers some nice features in terms of automatically finding related fonts in a family, namely the italic, bold, and bolditalic alternatives. To take advantage of that, there's a set of wildcard typescripts that take an arbitrary Macintosh font name as input, and provide as many of the alternatives it can find. To set these typescripts (and the calling conventions) apart from the familiar ones, the typescripts are identified with `Xserif`, `Xsans`, and `Xmono`.

Identifier	file	Supported styles
zapfino	type-hgz	serif, handwriting
optima-nova	type-hgz	sans
optima-nova-os	type-hgz	sans
optima-nova-light	type-hgz	sans
optima-nova-medium	type-hgz	sans
palatino-nova	type-hgz	serif
palatino-nova-os	type-hgz	serif
palatino-nova-light	type-hgz	serif
palatino-nova-medium	type-hgz	serif
palatino-sans	type-hgz	sans
palatino-informal	type-hgz	sans
melior	type-hgz	serif
– all four-variant fonts –	type-xtx	Xserif
– all four-variant fonts –	type-xtx	Xsans
– all four-variant fonts –	type-xtx	Xmono
times	type-xtx	serif
palatino	type-xtx	serif
helvetica	type-xtx	sans
courier	type-xtx	mono
hoefler	type-xtx	serif
lucidagrande	type-xtx	sans
optima	type-xtx	sans
gillsans	type-xtx	sans
gillsanslt	type-xtx	sans
zapfino	type-xtx	handwriting, serif
applechancery	type-xtx	calligraphy, serif
timesnewroman	type-xtx	serif
arial	type-xtx	sans
lucida	type-xtx	serif, sans, mono, handwriting, calligraphy, fax

Table 1.11 The predefined body font identifiers for commercial Unicode (Opentype) fonts

To call the typescripts, it's most convenient to define a typeface that uses these features. The named font slot should contain the display name of the Regular alternative (not the family name) of the font in question. For example, you could have the following mix:

```
\starttypescript[myface]
\definetypeface[myface][rm][Xserif][Baskerville]    [default]
\definetypeface[myface][tt][Xmono] [Courier]        [default][rscale=.87]
\definetypeface[myface][ss][Xsans] [Optima Regular][default]
\stoptypescript
```

As you can see, you can activate relative scaling of face sizes. The above definitions look very much like any other typeface definition, except that the serif/sans/mono identifier is preceded with X, and that there is no underlying "Optima Regular" defined anywhere. Those missing bits of the definitions are handled by typescript and XeTeX magic.

1.6.4 Body font environments

Earlier we have seen that within a font family there are different font sizes. The relations between these sizes are defined by *body font environments*.

For all regular font sizes, environments are predefined that fulfill their purpose adequately. However when you want to do some extra defining yourself there is:

```
\definebodyfontenvironment [.1.] [.2.] [..,.3=.,..]
                           OPTIONAL OPTIONAL     OPTIONAL
1    IDENTIFIER
2   5pt   ...  12pt  default
3   text          =  DIMENSION
    script        =  DIMENSION
    scriptscript  =  DIMENSION
    x             =  DIMENSION
    xx            =  DIMENSION
    small         =  DIMENSION
    big           =  DIMENSION
```

The first argument is optional, and specifier the typeface identifier that this particular body font environment setup is for. It defaults to the current typeface.

The second argument is the size of the body font environment that is being defined. This argument is not really optional, the macro syntax description is a little misleading.

The third argument once again is optional, and contains the actual settings as key-value pairs. If it is missing, defaults will be guessed at by ConTeXt itself.

text	Math text size
script	Math script size
scriptscript	Math scriptscript size
x	The size used for the `\tfx` etc. commands
xx	The size used for the `\tfxx` command
big	The 'larger' font size
small	The 'smaller' font size
interlinespace	Distance between lines in a paragraph, can be ignored because a reasonable default is always set up

So, when you want to have a somewhat bigger fontsize for just a few words (e.g. for a book title) you can type:

```
\definebodyfontenvironment [24pt]
\switchtobodyfont[24pt]
```

but longer stretches of text, you will need to set up most of the values, using something like this

```
\definebodyfontenvironment
  [22pt]
  [        text=22pt,
         script=17.3pt,
   scriptscript=14.4pt,
              x=17.3pt,
             xx=14.4pt,
            big=28pt,
          small=17.3pt]
```

To tweak already defined sizes, there is an accompagnying setup command with the same parameter conventions:

```
\setupbodyfontenvironment [.1.] [.2.] [..,.3.,..]
                          OPTIONAL OPTIONAL OPTIONAL
1    inherits from \definebodyfontenvironment
2   inherits from \definebodyfontenvironment
3   inherits from \definebodyfontenvironment
```

1.7 Font feature sets

As we have seen already, some fonts contain extra information besides the actual glyph shapes. In traditional TEX fonts, the extra information is roughly limited to kerning pairs and ligature information, and both of these 'features' are automatically applied to the text that is being typeset. In the odd case where one of the two needs to be suppressed, a little bit of macro trickery can do the job without too many complicating factors.

But with the new OpenType font format that is used by XETEX and LUATEX, the list of possible features is increased enormously. OpenType fonts have not just kerning information and ligature information, but there can also be other features like optional oldstyle figures, caps and smallcaps glyphs, decorative swashes, etc. all inside a single font file.

Not only that, but some of these features are not even supposed to be active all the time. Certain features should only be activated if the user asks for it, other features depend on the script and language that is in use for the text that is being typeset.

This is a big step forward in that there are now far fewer fonts needed to achieve the same level of quality than before, all that extra font information also poses a big challenge for macro writers. And add to that the fact that at the core, the two engines (XETEX and LUATEX) handle OpenType fonts completely different from each other.

CONTEXT has a new subsystem called 'font features' to create order in this forest of features. The most important command is `\definefontfeature`. This command can be used to group various font features under a single symbolic name, that can then be used as e.g. the argument to the `features` key of `\definetypeface`.

```
\definefontfeature
   [default-base]
   [script=latn,language=dflt,liga=yes,kern=yes,tlig=yes,trep=yes]
```

As you can probably guess, the first argument is the symbolic name that is being defined. The second argument is a mix of a-hoc settings and OpenType font features.

script	An OpenType script identifier
language	An OpenType script language identifier
tlig, texligatures	A virtual feature for legacy (TEX-style) automatic ligatures
trep, texquotes	A virtual feature for legacy (TEX-style) automatic ligatures (only works in MKIV)

mode	Processing mode for MkIV. `node` and `base` allowed, `node` is default
<tag>	Any OpenType feature tag is acceptable, but in MkIV only a 'known' subset actually has any effect, and then only in `node` mode. This list is given in table 1.12. In XeTeX, processing depends on the internal subengine that is used by XeTeX, and that is outside of ConTeXt's control.

A few fontfeatures are predefined by context:

default	liga=yes,kern=yes,tlig=yes,trep=yes
smallcaps	liga=yes,kern=yes,tlig=yes,trep=yes,smcp=yes
oldstyle	liga=yes,kern=yes,tlig=yes,trep=yes,onum=yes

1.8 Displaying the current font setup

With the command `\showbodyfont` an overview is generated of the available characters, and an overview of the different fontsizes within a family can be summoned with `\showbodyfontenvironment`.

```
\showbodyfont [...,*,...]
                OPTIONAL
*   inherits from \setupbodyfont
```

```
\showbodyfontenvironment [...,*,...]
                           OPTIONAL
*   inherits from \setupbodyfont
```

Specifiying actual `IDENTIFIER`s to these commands is currently unreliable because they internally are still counting on an older system of body font definitions, but you can safely use a size argument to get the information for the current font set.

Below an example of the possible output is shown, for `\showbodyfont[12pt]`

[palatino] [12pt]													\mr : Ag
	\tf	\sc	\sl	\it	\bf	\bs	\bi	\tfx	\tfxx	\tfa	\tfb	\tfc	\tfd
\rm	Ag	Ag	*Ag*	*Ag*	**Ag**	***Ag***	***Ag***	Ag	Ag	Ag	Ag	Ag	Ag
\ss	Ag	Ag	*Ag*	*Ag*	**Ag**	***Ag***	***Ag***	Ag	Ag	Ag	Ag	Ag	Ag
\tt	Ag	Ag	*Ag*	*Ag*	**Ag**	***Ag***	***Ag***	Ag	Ag	Ag	Ag	Ag	Ag

And the output of `\showbodyfontenvironment[12pt]` is:

aalt	Access All Alternates	med2	Medial Forms #2
abvm	Above-Base Mark Positioning	mgrk	Mathematical Greek
afrc	Alternative Fractions	mkmk	Mark to Mark Positioning
akhn	Akhands	nalt	Alternate Annotation Forms
blwm	Below-Base Mark Positioning	nlck	NLC Kanji Forms
c2pc	Petite Capitals From Capitals	nukt	Nukta Forms
c2sc	Small Capitals From Capitals	numr	Numerators
calt	Contextual Alternates	onum	Old Style Figures
case	Case-Sensitive Forms	ordn	Ordinals
ccmp	Glyph Composition/Decomposition	ornm	Ornaments
clig	Contextual Ligatures	pnum	Proportional Figures
cpsp	Capital Spacing	pref	Pre-base Forms
cswh	Contextual Swash	pres	Pre-base Substitutions
curs	Cursive Positioning	pstf	Post-base Forms
dlig	Discretionary Ligatures	rlig	Required Ligatures
dnom	Denominators	rphf	Reph Form
expt	Expert Forms	rtla	Right-To-Left Alternates
fina	Terminal Forms	salt	Stylistic Alternates
fin2	Terminal Forms #2	sinf	Scientific Inferiors
fin3	Terminal Forms #3	smcp	Small Capitals
frac	Fractions	smpl	Simplified Forms
fwid	Full Width	ss01	Stylistic Set 1
haln	Halant Forms	ss02	Stylistic Set 2
hist	Historical Forms	ss03	Stylistic Set 3
hkna	Horizontal Kana Alternates	ss04	Stylistic Set 4
hlig	Historical Ligatures	ss05	Stylistic Set 5
hngl	Hangul	ss06	Stylistic Set 6
hwid	Half Width	ss07	Stylistic Set 7
init	Initial Forms	ss08	Stylistic Set 8
isol	Isolated Forms	ss09	Stylistic Set 9
ital	Italics	subs	Subscript
jp78	JIS78 Forms	sups	Superscript
jp83	JIS83 Forms	swsh	Swash
jp90	JIS90 Forms	titl	Titling
kern	Kerning	tnam	Traditional Name Forms
liga	Standard Ligatures	tnum	Tabular Figures
lnum	Lining Figures	trad	Traditional Forms
locl	Localized Forms	unic	Unicase
mark	Mark Positioning	zero	Slashed Zero
medi	Medial Forms		

Table 1.12 The OpenType features that are understood by MkIV in `mode=node` processing mode

[palatino] [12pt]							
text	script	scriptscript	x	xx	small	big	interlinespace
20.7pt	14.4pt	12pt	17.3pt	14.4pt	17.3pt	20.7pt	not set
17.3pt	12.1pt	8.6pt	13.8pt	10.3pt	13.8pt	20.7pt	not set
14.4pt	10pt	7.2pt	11.5pt	8.6pt	11.5pt	17.2pt	not set
12pt	8.3pt	6pt	9.6pt	7.2pt	9.6pt	14.3pt	not set
11pt	7.6pt	5.5pt	8.8pt	6.6pt	8.8pt	13.1pt	not set
10pt	6.9pt	5pt	8pt	6pt	8pt	11.9pt	not set
9pt	6.2pt	4.5pt	7.2pt	5.4pt	7.2pt	10.7pt	not set
8pt	5.5pt	4pt	6.4pt	4.8pt	6.4pt	9.5pt	not set
7pt	4.8pt	3.5pt	5.6pt	4.2pt	5.6pt	8.3pt	not set
6pt	4.1pt	3pt	4.8pt	3.6pt	4.8pt	7.1pt	not set
5pt	3.4pt	2.5pt	4pt	3pt	4pt	5.9pt	not set
4pt	2.7pt	2pt	3.2pt	2.4pt	3.2pt	4.7pt	not set

1.9 Math fonts

FIXME: This is paragraph is a mess

There are only a few font families that can handle math. There is the Computer Modern Roman, the very beautiful Lucida Bright that we prefer in electronic documents, and of course one can use the 'prefered by publishers font' Times. These fonts carry a complete set of characters and symbols for mathematical typesetting. Among these, the Computer Modern Roman distinguishes itself by its many design sizes, which pays off when typesetting complicate math. On this design there are a few variations called Euler and Concrete.[1]

Many TEX users have chosen TEX for its superb math type setting. The math oriented character of TEX has also influenced the font mechanism. We will not go into any details but the central key is the *family*. There is a font family for `\bf`, `\it`, etc. Within a family we distinguish three members: text, script and scriptscript, or a normal, smaller and smallest font. The normal font size is used for running text and the smaller ones for sub and superscripts. The next example will show what the members of a font family can do.

```
$\tf x^2+\bf x^2+\sl x^2+\it x^2+\bs x^2+ \bi x^2 =\rm 6x^2$
$\tf x^2+\bf x^2+\sl x^2+\it x^2+\bs x^2+ \bi x^2 =\tf 6x^2$
$\tf x^2+\bf x^2+\sl x^2+\it x^2+\bs x^2+ \bi x^2 =\bf 6x^2$
$\tf x^2+\bf x^2+\sl x^2+\it x^2+\bs x^2+ \bi x^2 =\sl 6x^2$
```

When this is typeset you see this:

$x^2 + \mathbf{x}^2 + x^2 + x^2 + x^2 + x^2 = 6x^2$
$x^2 + \mathbf{x}^2 + x^2 + x^2 + x^2 + x^2 = 6x^2$
$x^2 + \mathbf{x}^2 + x^2 + x^2 + x^2 + x^2 = 6\mathbf{x}^2$

[1] See Concrete Mathematics by Knuth cs., an outstanding book from the perspective of typography and didactically.

$x^2 + \mathbf{x}^2 + x^2 + x^2 + \boldsymbol{x}^2 + \boldsymbol{x}^2 = 6x^2$

We can see that the characters adapt but that the symbols are typeset in the same font. Technically this means that the symbols are set in font family 0 (there are 16 families) and in this case that is default `\tf`.

It can also be done somewhat differently as we will see in the next example. A new command is used: `\mf`, which stands for *math font*. This command takes care of the symbols in such a way that they are set in the actual font.[2]

$x^2 + x^2 + x^2 + x^2 + x^2 + x^2 = 6x^2$
$\mathbf{x^2 + x^2 + x^2 + x^2 + x^2 + x^2 = 6x^2}$
$x^2 + x^2 + x^2 + x^2 + x^2 + x^2 = 6x^2$
$\boldsymbol{x^2 + x^2 + x^2 + x^2 + x^2 + x^2 = 6x^2}$
$x^2 + x^2 + x^2 + x^2 + x^2 + x^2 = 6x^2$
$\boldsymbol{x^2 + x^2 + x^2 + x^2 + x^2 + x^2 = 6x^2}$

You should take into account that TEX typesets a formula as a whole. In some cases this means that setups at the end of the formula have effect at the beginning.

```
$\tf\mf x^2 + x^2 + x^2 + x^2 + x^2 + x^2 = 6x^2$
$\bf\mf x^2 + x^2 + x^2 + x^2 + x^2 + x^2 = 6x^2$
$\sl\mf x^2 + x^2 + x^2 + x^2 + x^2 + x^2 = 6x^2$
$\bs\mf x^2 + x^2 + x^2 + x^2 + x^2 + x^2 = 6x^2$
$\it\mf x^2 + x^2 + x^2 + x^2 + x^2 + x^2 = 6x^2$
$\bi\mf x^2 + x^2 + x^2 + x^2 + x^2 + x^2 = 6x^2$
```

The exact location of `\mf` is not that important. We also could have typed:

```
$\bf x^2 + x^2 + x^2 + x^2 + x^2 + x^2 = \mf 6x^2$
```

One other aspect of fonts in math mode is the way reserved names like `\sin` and `\cos` are typeset.

```
$\bf x^2 + \hbox{whatever} + \sin(2x)$
```

Unlike plain TEX, the sin is also set bold.

$\mathbf{x^2 + whatever + \sin(2x)}$

1.10 Em and Ex

In specifying dimensions we can distinguish physical units like `pt` and `cm` and internal units like `em` and `ex`. These last units are related to the actual fontsize. When you use these internal units in specifying for example horizontal and vertical spacing you don't have to do any recalculating when fonts are switched in the style definition.

Some insight in these units does not hurt. The width of an `em` is not the with of an M, but that of an — (an em–dash). When this glyph is not available in the font another value is used. Table 1.13 shows some examples. We see that the width of a digit is about `.5em`. In Computer Modern Roman a digit is excactly half an em wide.

[2] We also see a strange visual effect. It seems as if the lines are sloped.

\tf	\bf	\sl	\tt	\ss	\tfx
12 M	**12** **M**	*12* *M*	12 M	12 M	12 M

Table 1.13 The width of an em.

In most cases we use em for specifying width and and ex for height. Table 1.14 shows some examples. We see that the height equals the height of a lowercase x.

\tf	\bf	\sl	\tt	\ss	\tfx
x	**x**	*x*	x	x	x

Table 1.14 The height of an ex.

1.11 Encodings and mappings

This paragraph only applies to pdfTEX. If you are exclusively using XETEX or MKIV, you can safely ignore the following text.

Not every language uses the (western) latin alphabet. Although in most languages the basic 26 characters are somehow used, they can be combined with a broad range of accents placed in any place.

In order to get a character representation, also called glyph, in the resulting output, you have to encode it in the input. This is no problem for a..z, but other characters are accessed by name, for instance \eacute. The glyph é can be present in the font but when it's not there, TEX has to compose the character from a letter e and an accent `.

In practice this means that the meaning of \eacute depends on the font and font encoding used. There are many such encodings, each suited for a subset of languages.

encoding	usage	status
ec	the prefered encoding of TEX distributions	okay
texnansi	a combination of TEX and Adobe standard encoding	okay
qx	an encoding that covers most eastern european languages	okay
t5	an encoding dedicated to vietnamese (many (double) accents)	okay
t2a	a cyrillic TEX font encoding	?
t2b	another cyrillic TEX font encoding	?
t2c	another another cyrillic TEX font encoding	?
x2	another another another cyrillic TEX font encoding	?
default	the 7 bit ASCII encoding as used by plain TEX	obsolete
il2	iso latin 2 encoding as needed for Czech and Slovak	obsolete
pl0	a native Polish encoding	obsolete
uc	a 16-bit encoding that can fake the Unicode base plane	obsolete

`8r`	a (strange) mixture of encodings	useless
`l7x`	?	?

These encodings are font related as is demonstrated in figure 1.1, 1.2, 1.3, and 1.4. Here we used the `\showfont` command.

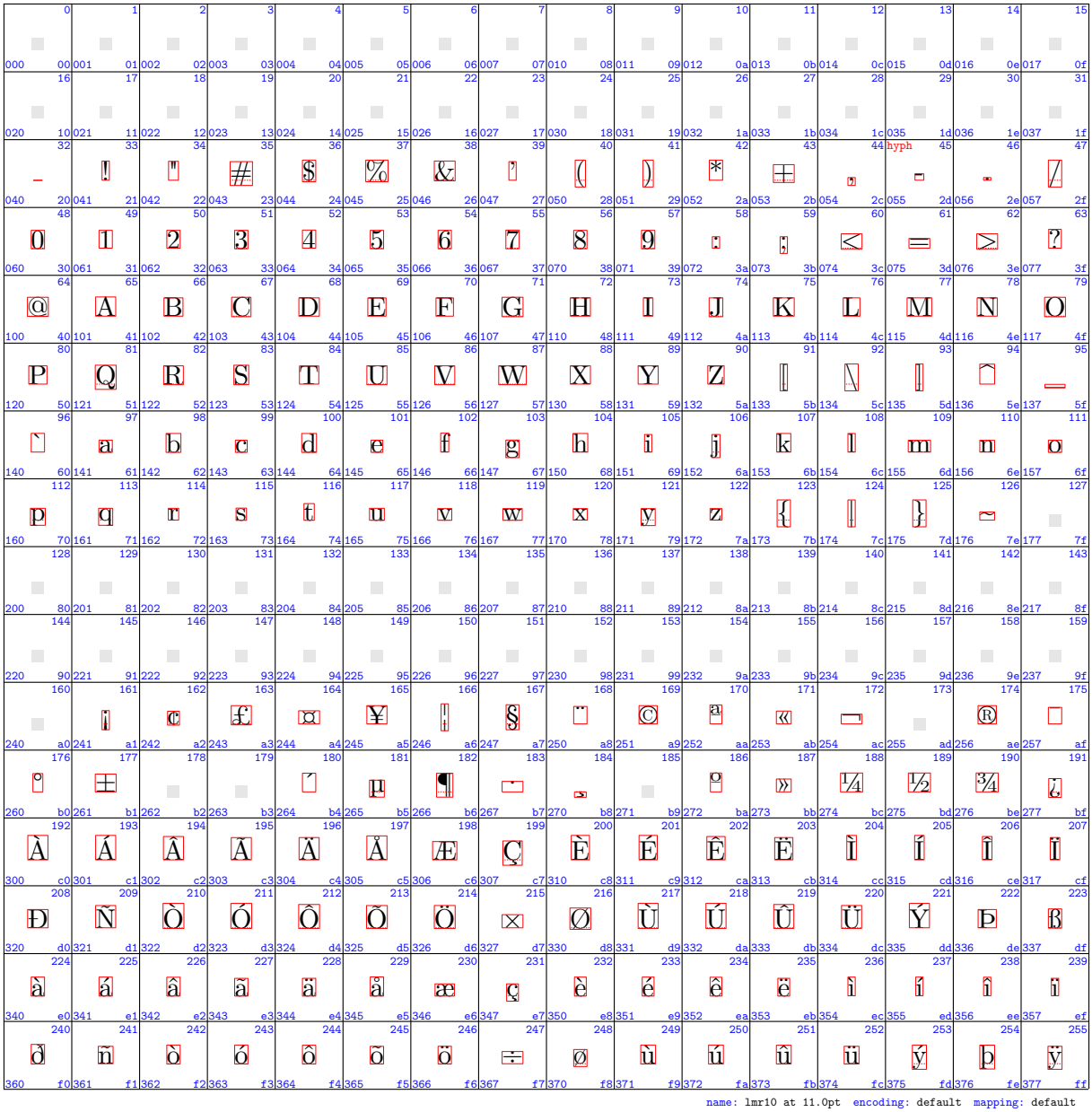


Figure 1.1 The Latin Modern Roman font in ec encoding.

The situation is even more complicated than it looks, since the font may be virtual, that is, built from several fonts.

The advantage of using specific encodings is that you can let TEX hyphenate words in the appropriate way. The hyphenation patterns are applied to the internal data structures that represent the sequence of glyphs. In spite of what you may expect, they are font–dependent! Even more

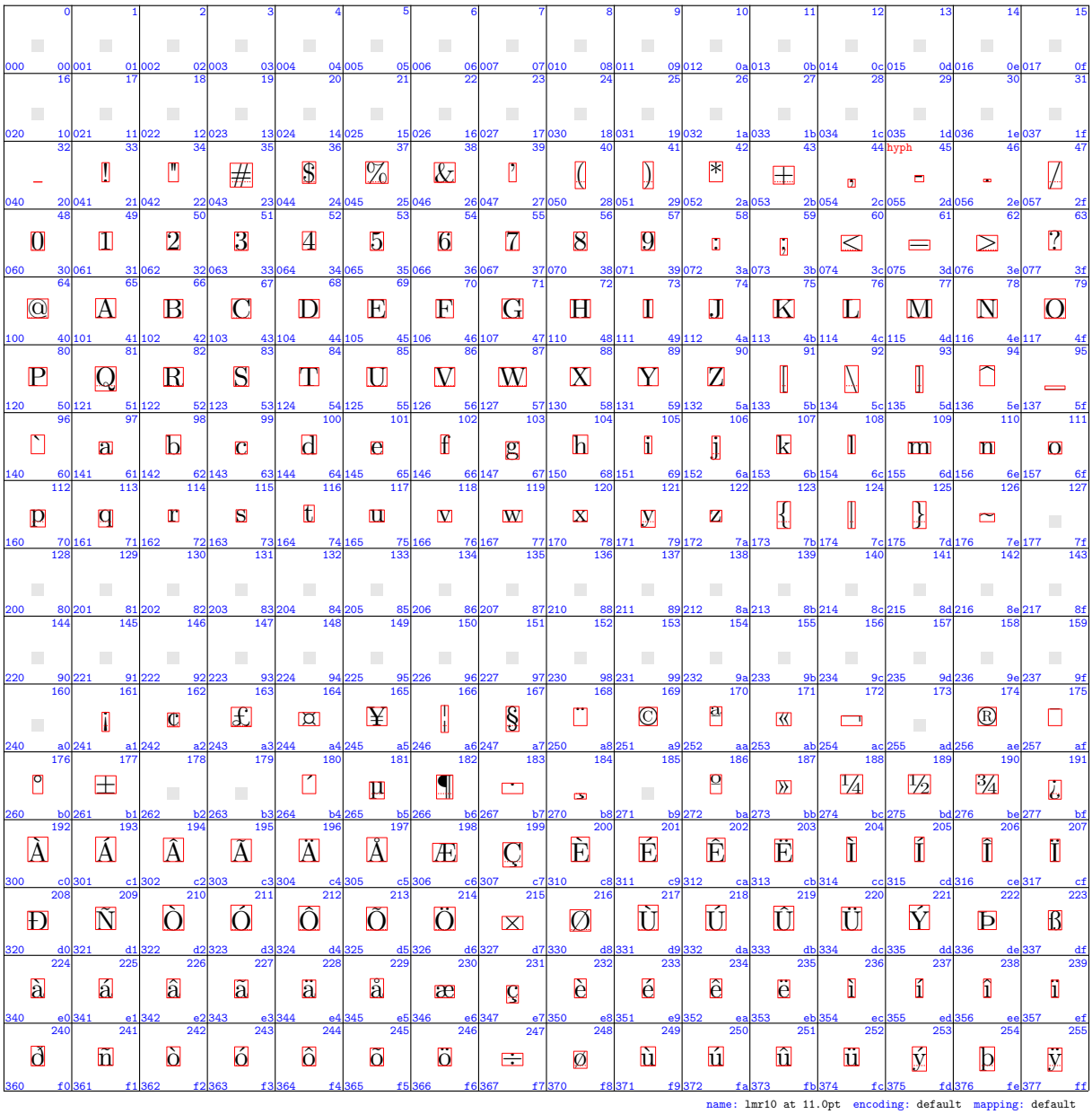


Figure 1.2 The Latin Modern Roman font in texnansi encoding.

confusing: they not only depend on the font encoding, but also on the mapping from lower to uppercase characters, or more precise, on the existence of such a mapping.

Unless you want to play with these encodings and mappings, in most cases you can forget their details and rely on what other TEX experts tell you to do. Normally switching from one to another encoding and/or mapping takes place with the change in fonts or when some special output encoding is needed, for instance in PDF annotations and/or unicode vectors that enable searching in documents. So, to summarize this: encodings and mappings depend on the fonts used as well have consequences for the language specific hyphenation patterns. Fortunately CONTEXT handles this for you automatically.

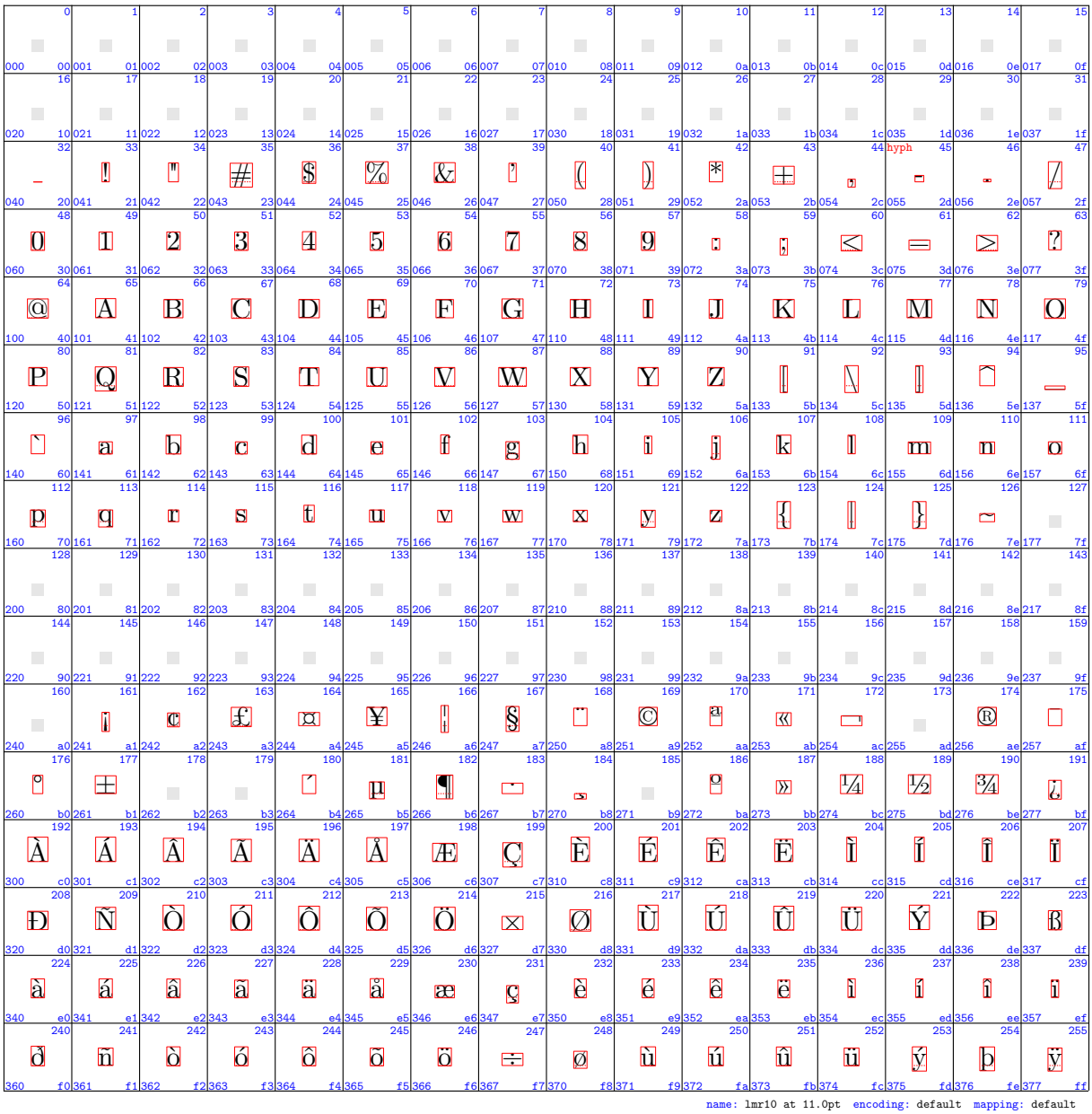


Figure 1.3 The Latin Modern Roman font in qx encoding.

If you want to know to what extent a font is complete and characters need to be composed on the fly, you can typeset a a couple of tables. The (current) composition is shown by `\showaccents`:

—— `default /opt/texmf-local/fonts/data/e-foundry/tex-gyre/texgyrepagella-regular`

\'	á	b́	ć	d́	é	f́	ǵ	h́	í	j́	ḱ	Í	ḿ	ń	ó	ṕ	q́	ŕ	ś	t́	ú	v́	ẃ	x́	ý	ź
	Á	B́	Ć	D́	É	F́	Ǵ	H́	Í	J́	Ḱ	Ĺ	Ḿ	Ń	Ó	Ṕ	Q́	Ŕ	Ś	T́	Ú	V́	Ẃ	X́	Ý	Ź
\`	à	b̀	c̀	d̀	è	f̀	g̀	h̀	ì	j̀	k̀	l̀	m̀	ǹ	ò	p̀	q̀	r̀	s̀	t̀	ù	v̀	ẁ	x̀	ỳ	z̀
	À	B̀	C̀	D̀	È	F̀	G̀	H̀	Ì	J̀	K̀	L̀	M̀	Ǹ	Ò	P̀	Q̀	R̀	S̀	T̀	Ù	V̀	Ẁ	X̀	Ỳ	Z̀
\^	â	b̂	ĉ	d̂	ê	f̂	ĝ	ĥ	î	ĵ	k̂	l̂	m̂	n̂	ô	p̂	q̂	r̂	ŝ	t̂	û	v̂	ŵ	x̂	ŷ	ẑ
	Â	B̂	Ĉ	D̂	Ê	F̂	Ĝ	Ĥ	Î	Ĵ	K̂	L̂	M̂	N̂	Ô	P̂	Q̂	R̂	Ŝ	T̂	Û	V̂	Ŵ	X̂	Ŷ	Ẑ

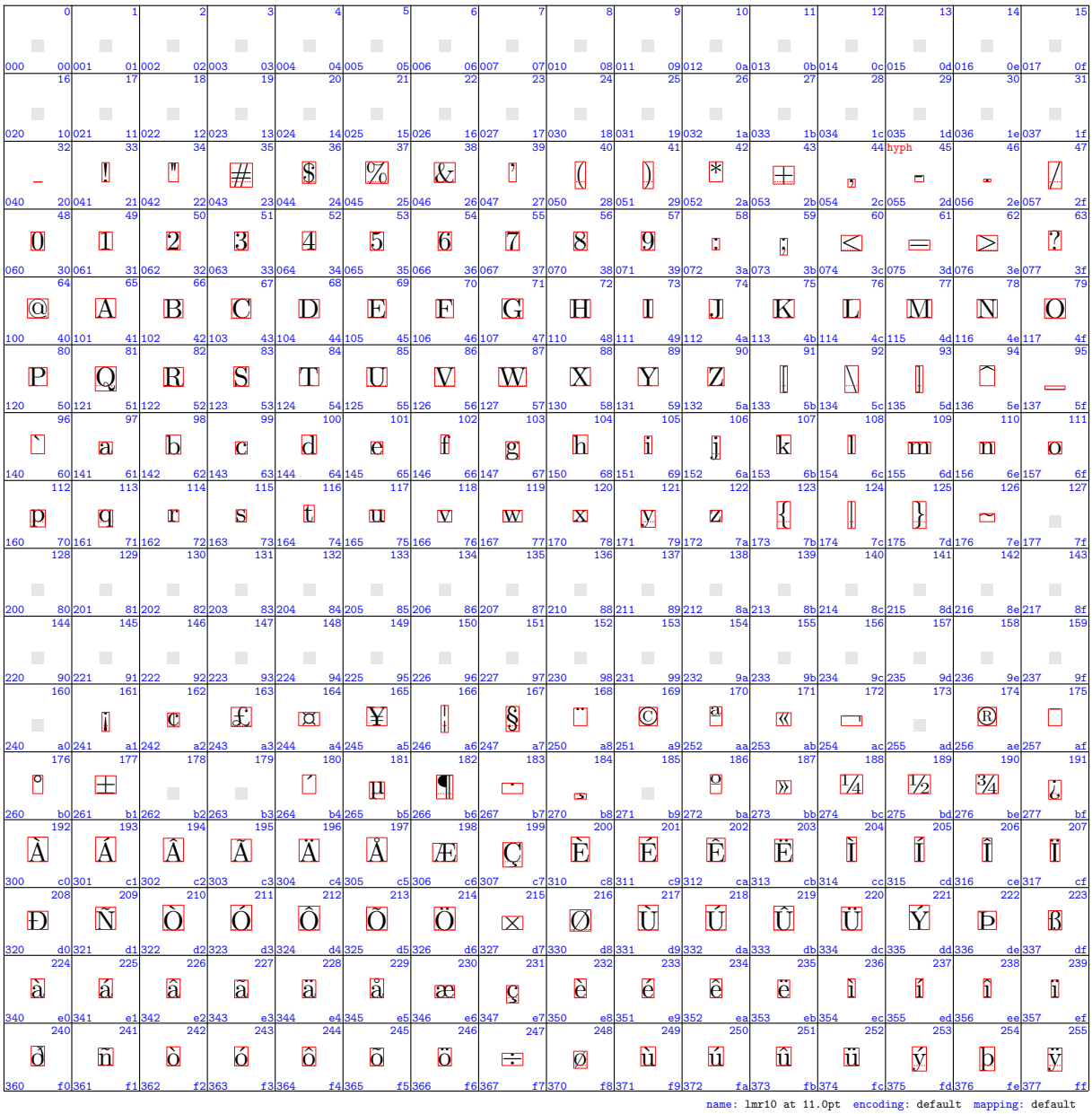


Figure 1.4 The Latin Modern Roman font in t5 encoding.

\~	ã	b̃	c̃	d̃	ẽ	f̃	g̃	h̃	ĩ	j̃	k̃	l̃	m̃	ñ	õ	p̃	q̃	r̃	s̃	t̃	ũ	ṽ	w̃	x̃	ỹ	z̃
	Ã	B̃	C̃	D̃	Ẽ	F̃	G̃	H̃	Ĩ	J̃	K̃	L̃	M̃	Ñ	Õ	P̃	Q̃	R̃	S̃	T̃	Ũ	Ṽ	W̃	X̃	Ỹ	Z̃
\"	ä	b̈	c̈	d̈	ë	f̈	g̈	ḧ	ï	j̈	k̈	l̈	m̈	n̈	ö	p̈	q̈	r̈	s̈	ẗ	ü	v̈	ẅ	ẍ	ÿ	z̈
	Ä	B̈	C̈	D̈	Ë	F̈	G̈	Ḧ	Ï	J̈	K̈	L̈	M̈	N̈	Ö	P̈	Q̈	R̈	S̈	T̈	Ü	V̈	Ẅ	Ẍ	Ÿ	Z̈
\H	á	b̋	c̋	d̋	é	f̋	g̋	h̋	í	j̋	k̋	l̋	m̋	n̋	ő	p̋	q̋	r̋	s̋	t̋	ű	v̋	ẅ	x̋	ý	z̋
	Á	B̋	C̋	D̋	É	F̋	G̋	H̋	Í	J̋	K̋	L̋	M̋	N̋	Ő	P̋	Q̋	R̋	S̋	T̋	Ű	V̋	Ẅ	X̋	Ý	Z̋
\r	å	b̊	c̊	d̊	e̊	f̊	g̊	h̊	i̊	j̊	k̊	l̊	m̊	n̊	o̊	p̊	q̊	r̊	s̊	t̊	ů	v̊	ẘ	x̊	ẙ	z̊
	Å	B̊	C̊	D̊	E̊	F̊	G̊	H̊	Í	J̊	K̊	L̊	M̊	N̊	O̊	P̊	Q̊	R̊	S̊	T̊	Ů	V̊	W̊	X̊	Y̊	Z̊
\v	ǎ	b̌	č	ď	ě	f̌	ǧ	ȟ	ǐ	ǰ	ǩ	ľ	m̌	ň	ǒ	p̌	q̌	ř	š	ť	ǔ	v̌	w̌	x̌	y̌	ž
	Ǎ	B̌	Č	Ď	Ě	F̌	Ǧ	Ȟ	Ǐ	J̌	Ǩ	Ľ	M̌	Ň	Ǒ	P̌	Q̌	Ř	Š	Ť	Ǔ	V̌	W̌	X̌	Y̌	Ž

\u	ă	b̆	č	d̆	ĕ	f̆	ğ	h̆	ĭ	j̆	k̆	l̆	m̆	n̆	ŏ	p̆	q̆	r̆	s̆	t̆	ŭ	v̆	w̆	x̆	y̆	ž
	Ă	B̆	Č	D̆	Ĕ	F̆	Ğ	H̆	Ĭ	J̆	K̆	L̆	M̆	N̆	Ŏ	P̆	Q̆	R̆	S̆	T̆	Ŭ	V̆	W̆	X̆	Y̆	Ž
\=	ā	b̄	c̄	d̄	ē	f̄	ḡ	h̄	ī	j̄	k̄	l̄	m̄	n̄	ō	p̄	q̄	r̄	s̄	t̄	ū	v̄	w̄	x̄	ȳ	z̄
	Ā	B̄	C̄	D̄	Ē	F̄	Ḡ	H̄	Ī	J̄	K̄	L̄	M̄	N̄	Ō	P̄	Q̄	R̄	S̄	T̄	Ū	V̄	W̄	X̄	Ȳ	Z̄
\.	ȧ	ḃ	ċ	ḋ	ė	ḟ	ġ	ḣ	i	j̇	k̇	l̇	ṁ	ṅ	ȯ	ṗ	q̇	ṙ	ṡ	ṫ	u̇	v̇	ẇ	ẋ	ẏ	ż
	Ȧ	Ḃ	Ċ	Ḋ	Ė	Ḟ	Ġ	Ḣ	İ	J̇	K̇	L̇	Ṁ	Ṅ	Ȯ	Ṗ	Q̇	Ṙ	Ṡ	Ṫ	U̇	V̇	Ẇ	Ẋ	Ẏ	Ż
\b	a̲	b̲	c̲	d̲	e̲	f̲	g̲	h̲	i̲	j̲	k̲	l̲	m̲	n̲	o̲	p̲	q̲	r̲	s̲	t̲	u̲	v̲	w̲	x̲	y̲	z̲
	A̲	B̲	C̲	D̲	E̲	F̲	G̲	H̲	I̲	J̲	K̲	L̲	M̲	N̲	O̲	P̲	Q̲	R̲	S̲	T̲	U̲	V̲	W̲	X̲	Y̲	Z̲
\d	ạ	ḅ	ç	ḍ	ẹ	f̣	ḡ	ḥ	ị	j̣	ḳ	ḷ	ṃ	ṇ	ọ	ṗ	q̣	ṛ	ṣ	ṭ	ụ	ṿ	ẉ	ẋ	ỵ	ẓ
	Ạ	Ḅ	Ç	Ḍ	Ẹ	F̣	Ḡ	Ḥ	Ị	J̣	Ḳ	Ḷ	Ṃ	Ṇ	Ọ	Ṗ	Q̣	Ṛ	Ṣ	Ṭ	Ụ	Ṿ	Ẉ	Ẋ	Ỵ	Ẓ
\k	ą	b̨	ç	d̨	ę	f̨	g̨	h̨	į	j̨	k̨	l̨	m̨	n̨	o̧	p̨	q̨	r̨	s̨	t̨	ų	v̨	w̨	x̨	y̨	z̨
	Ą	B̨	Ç	D̨	Ę	F̨	G̨	H̨	Į	J̨	K̨	L̨	M̨	N̨	O̧	P̨	Q̨	R̨	S̨	T̨	Ų	V̨	W̨	X̨	Y̨	Z̨
\c	a̧	b̧	ç	ḑ	ȩ	f̧	ģ	ḩ	i̧	j̧	ķ	ļ	m̧	ņ	o̧	p̧	q̧	ŗ	ş	ţ	u̧	v̧	w̧	x̧	y̧	z̧
	A̧	B̧	Ç	Ḑ	Ȩ	F̧	Ģ	Ḩ	I̧	J̧	Ķ	Ļ	M̧	Ņ	O̧	P̧	Q̧	Ŗ	Ş	Ţ	U̧	V̧	W̧	X̧	Y̧	Z̧

with `\showcharacters`, you get a list of named characters (and glyphs) as known to the system.

```
——— default /opt/tex/texmf-local/fonts/data/e-foundry/tex-gyre/texgyrepagella-regular
,   textcomma                œ   oeligature            ¥   textyen
.   textperiod               Œ   OEligature           ª   ordfeminine
´   textacute                ß   ssharp               º   ordmasculine
.   textbottomdot            SS  Ssharp               %   percent
˘   textbreve                þ   thorn                ‰   perthousand
ˇ   textcaron                Þ   Thorn                -   softhyphen
¸   textcedilla              ■   eth                  ·   periodcentered
^   textcircumflex           Ð   Eth                      compoundwordmark
¨   textdiaeresis            ¡   exclamdown            ^   textasciicircum
˙   textdotaccent            ¿   questiondown         ~   textasciitilde
`   textgrave                ©   copyright            /   textslash
˝   texthungarumlaut         ®   registered           \   textbackslash
¯   textmacron               ™   trademark            {   textbraceleft
˛   textogonek               §   sectionmark          }   textbraceright
°   textring                 ¶   paragraphmark        _   textunderscore
~   texttilde                ¼   onequarter           ␣   textvisiblespace
,   textbottomcomma          ½   onehalf              ¦   textbrokenbar
ı   dotlessi                 ¾   threequarter         •   textbullet
    dotlessj                 ¹   onesuperior          †   textdag
I   dotlessI                 ²   twosuperior          ‡   textddag
J   dotlessJ                 ³   threesuperior        °   textdegree
–   endash                   ¢   textcent            ÷   textdiv
—   emdash                   ¤   textcurrency        …   textellipsis
æ   aeligature               $   textdollar          /   textfraction
Æ   AEligature               €   texteuro            ¬   textlognot
ij  ijligature               ƒ   textflorin             textminus
IJ  IJligature               £   textsterling        µ   textmu
```

×	textmultiply	ù	ugrave	Ý	Yacute
±	textpm	Ỳ	Ygrave	ý	yacute
"	quotedbl	ỳ	ygrave	Ź	Zacute
„	quotedblbase	Ã	Atilde	ź	zacute
“	quotedblleft	ã	atilde	đ	dstroke
”	quotedblright	Ĩ	Itilde	Đ	Dstroke
`	quotesingle	ĩ	itilde	Ħ	Hstroke
‚	quotesinglebase	Ñ	Ntilde	ħ	hstroke
‘	quoteleft	ñ	ntilde		Tstroke
’	quoteright	Õ	Otilde		tstroke
‹	guilsingleleft	õ	otilde	Ċ	Cdotaccent
›	guilsingleright	Ũ	Utilde	ċ	cdotaccent
«	leftguillemot	ũ	utilde	Ė	Edotaccent
»	rightguillemot	Ỹ	Ytilde	ė	edotaccent
Â	Acircumflex	ỹ	ytilde	Ġ	Gdotaccent
â	acircumflex	Ä	Adiaeresis	ġ	gdotaccent
Ĉ	Ccircumflex	ä	adiaeresis	İ	Idotaccent
ĉ	ccircumflex	Ë	Ediaeresis	i	idotaccent
Ê	Ecircumflex	ë	ediaeresis	Ż	Zdotaccent
ê	ecircumflex	Ï	Idiaeresis	ż	zdotaccent
Ĝ	Gcircumflex	ï	idiaeresis	Ā	Amacron
ĝ	gcircumflex	Ö	Odiaeresis	ā	amacron
Ĥ	Hcircumflex	ö	odiaeresis	Ē	Emacron
ĥ	hcircumflex	Ü	Udiaeresis	ē	emacron
Î	Icircumflex	ü	udiaeresis	Ī	Imacron
î	icircumflex	Ÿ	Ydiaeresis	ī	imacron
Ĵ	Jcircumflex	ÿ	ydiaeresis	Ō	Omacron
ĵ	jcircumflex	Á	Aacute	ō	omacron
Ô	Ocircumflex	á	aacute	Ū	Umacron
ô	ocircumflex	Ć	Cacute	ū	umacron
Ŝ	Scircumflex	ć	cacute	Ç	Ccedilla
ŝ	scircumflex	É	Eacute	ç	ccedilla
Û	Ucircumflex	é	eacute	Ķ	Kcedilla
û	ucircumflex	Í	Iacute	ķ	kcedilla
Ŵ	Wcircumflex	í	iacute	Ļ	Lcedilla
ŵ	wcircumflex	Ĺ	Lacute	ļ	lcedilla
Ŷ	Ycircumflex	Í	lacute	Ņ	Ncedilla
ŷ	ycircumflex	Ń	Nacute	ņ	ncedilla
À	Agrave	ń	nacute	Ŗ	Rcedilla
à	agrave	Ó	Oacute	ŗ	rcedilla
È	Egrave	ó	oacute	Ş	Scedilla
è	egrave	Ŕ	Racute	ş	scedilla
Ì	Igrave	ŕ	racute	Ţ	Tcedilla
ì	igrave	Ś	Sacute	ţ	tcedilla
Ò	Ograve	ś	sacute	Ő	Ohungarumlaut
ò	ograve	Ú	Uacute	ő	ohungarumlaut
Ù	Ugrave	ú	uacute	Ű	Uhungarumlaut

ű	uhungarumlaut
Ą	Aogonek
ą	aogonek
Ę	Eogonek
ę	eogonek
Į	Iogonek
į	iogonek
Ų	Uogonek
ų	uogonek
Å	Aring
å	aring
Ů	Uring
ů	uring
Ă	Abreve
ă	abreve
Ĕ	Ebreve
ĕ	ebreve
Ğ	Gbreve
ğ	gbreve
Ĭ	Ibreve
ĭ	ibreve
Ŏ	Obreve
ŏ	obreve
Ŭ	Ubreve
ŭ	ubreve
Č	Ccaron
č	ccaron
Ď	Dcaron
ď	dcaron
Ě	Ecaron
ě	ecaron
Ľ	Lcaron
ľ	lcaron
Ň	Ncaron
ň	ncaron
Ř	Rcaron
ř	rcaron
Š	Scaron
š	scaron
Ť	Tcaron
ť	tcaron
Ÿ	Ycaron
ÿ	ycaron
Ž	Zcaron
ž	zcaron
Ł	Lstroke
ł	lstroke
Ø	Ostroke
ø	ostroke
ä	aumlaut
ë	eumlaut
ï	iumlaut
ö	oumlaut
ü	uumlaut
Ä	Aumlaut
Ë	Eumlaut
Ï	Iumlaut
Ö	Oumlaut
Ü	Uumlaut
ș	scommaaccent
Ș	Scommaaccent
ț	tcommaaccent
Ț	Tcommaaccent
ļ	lcommaaccent
Ļ	Lcommaaccent
Ẽ	Etilde
ẽ	etilde
Ả	Ahook
ả	ahook
Ẻ	Ehook
ẻ	ehook
Ỉ	Ihook
ỉ	ihook
Ỏ	Ohook
ỏ	ohook
Ủ	Uhook
ủ	uhook
Ỷ	Yhook
ỷ	yhook
Ầ	Acircumflexgrave
Ấ	Acircumflexacute
Ẫ	Acircumflextilde
Ẩ	Acircumflexhook
ầ	acircumflexgrave
ấ	acircumflexacute
ẫ	acircumflextilde
ẩ	acircumflexhook
Ề	Ecircumflexgrave
Ế	Ecircumflexacute
Ễ	Ecircumflextilde
Ể	Ecircumflexhook
ề	ecircumflexgrave
ế	ecircumflexacute
ễ	ecircumflextilde
ể	ecircumflexhook
Ồ	Ocircumflexgrave
Ố	Ocircumflexacute
Ỗ	Ocircumflextilde
Ổ	Ocircumflexhook
ồ	ocircumflexgrave
ố	ocircumflexacute
ỗ	ocircumflextilde
ổ	ocircumflexhook
Ằ	Abrevegrave
Ắ	Abreveacute
Ẵ	Abrevetilde
Ẳ	Abrevehook
ằ	abrevegrave
ắ	abreveacute
ẵ	abrevetilde
ẳ	abrevehook
Ạ	Adotbelow
ạ	adotbelow
Ẹ	Edotbelow
ẹ	edotbelow
Ị	Idotbelow
ị	idotbelow
Ọ	Odotbelow
ọ	odotbelow
Ụ	Udotbelow
ụ	udotbelow
Ỵ	Ydotbelow
ỵ	ydotbelow
Ợ	Ohorndotbelow
ợ	ohorndotbelow
Ự	Uhorndotbelow
ự	uhorndotbelow
Ậ	Acircumflexdotbelow
ậ	acircumflexdotbelow
Ệ	Ecircumflexdotbelow
ệ	ecircumflexdotbelow
Ộ	Ocircumflexdotbelow
ộ	ocircumflexdotbelow
Ặ	Abrevedotbelow
ặ	abrevedotbelow
Ơ	Ohorn
Ờ	Ohorngrave
Ớ	Ohornacute
Ỡ	Ohorntilde

Ở	`Ohornhook`	Ư	`Uhorn`	ừ	`uhorngrave`
ơ	`ohorn`	Ừ	`Uhorngrave`	ứ	`uhornacute`
ờ	`ohorngrave`	Ứ	`Uhornacute`	ữ	`uhorntilde`
ớ	`ohornacute`	Ữ	`Uhorntilde`	ử	`uhornhook`
ỡ	`ohorntilde`	Ử	`Uhornhook`		
ở	`ohornhook`	ư	`uhorn`		

If you want to know what patterns are used, you can try to hyphenate a word with `\showhyphenations`.

language : en (internal code:2)
font : /opt/tex/texmf-local/fonts/data/e-foundry/tex-gyre/texgyrepagella-regular.otf at 11.0pt
encoding : not set
mapping : not set
handling : not set

sample : abra-cadabra