# This Way

ConTEXt magazine #9
February 2005

## Using Platform Fonts

Hans Hagen
PRAGMA ADE

In this document I will explain how to use system fonts in CONTEXT in a convenient way. We will use the Verdana that ships with the Microsoft Windows environment as an example.

There was a time that TEX distributions shipped with a truckload of fonts compared to what your operating system came with. Times have changed and so it makes sense to look into ways to tap this new source of fonts.

Do you recognize this font? It's the Verdana that comes with Microsoft Windows. This truetype font is located in the systems font path, normally:

```
c:\windows\fonts
```

or:

```
%SYSTEMROOT%\fonts
```

In order to use this font, we need to generate the metrics. This can be done with TEXfont. First you need to generate the so called Adobe Font Metrics files:

```
ttf2afm -a verdana.ttf  > verdana.afm
ttf2afm -a verdanab.ttf > verdanab.afm
ttf2afm -a verdanai.ttf > verdanai.afm
ttf2afm -a verdanaz.ttf > verdanaz.afm
```

After this, you can use TEXfont to produce the TEX Font Metrics files:

```
texfont
    --vendor=microsoft
    --collection=verdana
    --encoding=texnansi
    --pattern=verdana*.afm
```

In order to use the font, you need to have the following files present on your system (four files on each of the first three paths):

```
<root>/afm/microsoft/verdana/verdana[|i|b|z].afm
<root>/tfm/microsoft/verdana/verdana[|i|b|z].tfm
<root>/truetype/microsoft/verdana/verdana[|i|b|z].ttf
<root>/map/pdftex/context/texnansi-microsoft-verdana.map
```

However, there is no need to keep duplicate copies of the TrueType fonts around, apart from possible copyright issues. First define an environment variable:

```
OSFONTDIR=%SYSTEMROOT%\fonts
```

Next, adapt a few font paths in your local copy of `texmf.cnf`:

```
OSFONTDIR =
AFMFONTS  = .;$TEXMF/fonts/afm//;$OSFONTDIR//
T1FONTS   = .;$TEXMF/fonts/{type1,pfb}//;$OSFONTDIR//
TTFONTS   = .;$TEXMF/fonts/{truetype,ttf}//;$OSFONTDIR//
OPENTYPEFONTS = .;$TEXMF/fonts/opentype//;$OSFONTDIR//
```

In the ConTEXt distribution you will find a file `context.cnf` that provides these definitions. The Verdana typescripts are part of the typescript collection `type-mws.tex`. As an exampe we show how they are defined:

```
\starttypescript [sans] [verdana] [name]

  \setups[font:fallback:sans]

  \definefontsynonym[Sans]           [Verdana]
  \definefontsynonym[SansBold]       [Verdana-Bold]
  \definefontsynonym[SansItalic]     [Verdana-Italic]
  \definefontsynonym[SansBoldItalic][Verdana-BoldItalic]

\stoptypescript

\starttypescript [sans] [verdana] [texnansi]

  \definefontsynonym [Verdana]
      [\typescriptthree-verdana]
      [encoding=\typescriptthree]
  \definefontsynonym [Verdana-Bold]
      [\typescriptthree-verdanab]
      [encoding=\typescriptthree]
  \definefontsynonym [Verdana-Italic]
      [\typescriptthree-verdanai]
      [encoding=\typescriptthree]
  \definefontsynonym [Verdana-BoldItalic]
      [\typescriptthree-verdanaz]
      [encoding=\typescriptthree]
```

```
\stoptypescript

\starttypescript [map] [verdana] [texnansi]

  \loadmapfile[texnansi-microsoft-verdana.map]

\stoptypescript
```

Before you can use these typesescripts, you need to import this typescript file. In your document (or style) a usable definition looks like:

```
\usetypescriptfile[type-msw]

\definetypeface [verdana]
  [ss] [sans]  [verdana]  [default]
  [encoding=texnansi]
\definetypeface [verdana]
  [rm] [serif] [palatino] [default]
  [encoding=texnansi,rscale=1.1]
\definetypeface [verdana]
  [mm] [math]  [palatino] [default]
  [encoding=texnansi,rscale=1.1]
\definetypeface [verdana]
  [tt] [mono]  [modern]   [default]
  [encoding=texnansi,rscale=1.25]
```

We used this definition for this document. This gives us the table shown in figure 1.

| [verdana] | | | | | | | | | | | | | \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 |

**Figure 1** \showbodyfont[verdana]

Those familiar with defining typescripts, may notice a new feature. Instead of defining each instance, we start with loading some fallbacks using the \setups command.

```
\setups[font:fallback:sans]
```

These setups are defined in type-def.tex. We could have used typescripts, but this is faster. The fallbacks:

```
\startsetups [font:fallback:sans]
  \definefontsynonym [Sans]             [DefaultFont]
  \definefontsynonym [SansBold]         [Sans]
  \definefontsynonym [SansItalic]       [Sans]
  \definefontsynonym [SansSlanted]      [SansItalic]
  \definefontsynonym [SansBoldItalic]   [Sans]
  \definefontsynonym [SansBoldSlanted] [SansBoldItalic]
  \definefontsynonym [SansCaps]         [Sans]
\stopsetups
```

Because we group the font definitions in (font) classes, these definitions are (as usual) local to the definition of the verdana typeface definition.

While playing with this font —actually, we needed the Verdana because one of our customers wanted to use this rather platform specific font— I also decided to speed up typescript processing a bit.

First of all, there is now an option to quit parsing typescripts once the one we're interested in is found and executed.

```
\starttypescript [sometypeface] [texnansi,ec,...]

  \definetypeface [sometypeface] [rm] ...
  \definetypeface [sometypeface] [ss] ...
  ...

  \quittypescriptscanning

\stoptypescript
```

Quitting only makes sense when the file is loaded early in the parsing stage, which happens to be true for the predefined typefaces.

Another speedup is to add the following directive to your local variant of `cont-sys.tex` or to your style.

```
\preloadtypescripts
```

If we define one typeface, we normally need four passes over the typescript files, one pass for rm, ss, tt and rm. Given:

```
\usetypescript [modern]  [texnansi]
```

On my machine, in Februari 2005, I gain quite some gross execution time as reported when I use the (command line) `–time-statistics` directive. Times are in milliseconds. The values are corrected for the time needed to process an empty, which takes about 500 milliseconds.

| **method** | **time** | **gain** | **percentage** |
|---|---|---|---|
| **normal** | 950 | | |
| **quiting** | 800 | 150 | 15% |
| **preloading** | 325 | 625 | 70% |

When we mix typefaces in documents, we get similar results:

```
\usetypescript [modern]  [texnansi]
\usetypescript [palatino][texnansi]
\usetypescript [times]   [texnansi]
```

We now get (rought estimates on multiple combined runs):

| **method** | **time** | **gain** | **percentage** |
|---|---|---|---|
| **normal** | 2700 | | |
| **quiting** | 2200 | 500 | 20% |
| **preloading** | 800 | 1900 | 70% |

Further speedups at the macro level are possible but don't make much sense because we would gain only a few milliseconds when handling many thousands of definitions, which is not the reality. We may also consider packing keywords in macros, which would definitely speed up things a bit as well as save memory, so maybe one day I will do this. Currently, on my machine, preloading takes some 112.000 extra memory words, and using a macros instead of strings can easily save us 20.000 memory words and at the same time give us about 5% speed improvement (of course only for typescript handling *loading*).

source code of this document

```
\usemodule[mag-01,abr-02]

\setvariables
  [magazine]
  [title={Using Platform Fonts},
   author=Hans Hagen,
   affiliation=PRAGMA ADE,
   date=February 2005,
   number=9]

\startbuffer[verdana]
\usetypescriptfile[type-msw]

\definetypeface [verdana]
  [ss] [sans]  [verdana]  [default]
  [encoding=texnansi]
\definetypeface [verdana]
  [rm] [serif] [palatino] [default]
  [encoding=texnansi,rscale=1.1]
\definetypeface [verdana]
  [mm] [math]  [palatino] [default]
  [encoding=texnansi,rscale=1.1]
\definetypeface [verdana]
  [tt] [mono]  [modern]   [default]
  [encoding=texnansi,rscale=1.25]
\stopbuffer

\getbuffer[verdana]

\startbuffer[abstract]
  In this document I will explain how to use system fonts in
  \CONTEXT\ in a convenient way. We will use the Verdana
  that ships with the Microsoft Windows environment as an
  example.
\stopbuffer

\starttext \setups [titlepage] \setups [title]

\start \switchtobodyfont[verdana]

There was a time that \TEX\ distributions shipped with a
truckload of fonts compared to what your operating system came
with. Times have changed and so it makes sense to look into
ways to tap this new source of fonts.
```

source code of this document

Do you recognize this font? It's the Verdana that comes with Microsoft Windows. This truetype font is located in the systems font path, normally:

```
\starttyping
c:\windows\fonts
\stoptyping
```

or:

```
\starttyping
%SYSTEMROOT%\fonts
\stoptyping
```

In order to use this font, we need to generate the metrics. This can be done with \TEXFONT. First you need to generate the so called Adobe Font Metrics files:

```
\starttyping
ttf2afm -a verdana.ttf  > verdana.afm
ttf2afm -a verdanab.ttf > verdanab.afm
ttf2afm -a verdanai.ttf > verdanai.afm
ttf2afm -a verdanaz.ttf > verdanaz.afm
\stoptyping
```

After this, you can use \TEXFONT\ to produce the \TEX\ Font Metrics files:

```
\starttyping
texfont
    --vendor=microsoft
    --collection=verdana
    --encoding=texnansi
    --pattern=verdana*.afm
\stoptyping
```

In order to use the font, you need to have the following files present on your system (four files on each of the first three paths):

```
\starttyping
<root>/afm/microsoft/verdana/verdana[|i|b|z].afm
<root>/tfm/microsoft/verdana/verdana[|i|b|z].tfm
<root>/truetype/microsoft/verdana/verdana[|i|b|z].ttf
```

source code of this document

```
<root>/map/pdftex/context/texnansi-microsoft-verdana.map
\stoptyping

However, there is no need to keep duplicate copies of the
TrueType fonts around, apart from possible copyright issues.
First define an environment variable:

\starttyping
OSFONTDIR=%SYSTEMROOT%\fonts
\stoptyping

Next, adapt a few font paths in your local copy of \type
{texmf.cnf}:

\starttyping
OSFONTDIR =
AFMFONTS  = .;$TEXMF/fonts/afm//;$OSFONTDIR//
T1FONTS   = .;$TEXMF/fonts/{type1,pfb}//;$OSFONTDIR//
TTFONTS   = .;$TEXMF/fonts/{truetype,ttf}//;$OSFONTDIR//
OPENTYPEFONTS = .;$TEXMF/fonts/opentype//;$OSFONTDIR//
\stoptyping

In the \CONTEXT\ distribution you will find a file \type
{context.cnf} that provides these definitions. The Verdana
typescripts are part of the typescript collection \type
{type-mws.tex}. As an exampe we show how they are defined:

\starttyping
\starttypescript [sans] [verdana] [name]

  \setups[font:fallback:sans]

  \definefontsynonym[Sans]           [Verdana]
  \definefontsynonym[SansBold]       [Verdana-Bold]
  \definefontsynonym[SansItalic]     [Verdana-Italic]
  \definefontsynonym[SansBoldItalic][Verdana-BoldItalic]

\stoptypescript

\starttypescript [sans] [verdana] [texnansi]

  \definefontsynonym [Verdana]
      [\typescriptthree-verdana]
      [encoding=\typescriptthree]
```

source code of this document

```
  \definefontsynonym [Verdana-Bold]
      [\typescriptthree-verdanab]
      [encoding=\typescriptthree]
  \definefontsynonym [Verdana-Italic]
      [\typescriptthree-verdanai]
      [encoding=\typescriptthree]
  \definefontsynonym [Verdana-BoldItalic]
      [\typescriptthree-verdanaz]
      [encoding=\typescriptthree]

\stoptypescript

\starttypescript [map] [verdana] [texnansi]

  \loadmapfile[texnansi-microsoft-verdana.map]

\stoptypescript
\stoptyping
```

Before you can use these typesescripts, you need to import this typescript file. In your document (or style) a usable definition looks like:

```
\typebuffer[verdana]
```

We used this definition for this document. This gives us the table shown in \in {figure} [fig:verdana].

```
\placefigure
    [here]
    [fig:verdana]
    {\type {\showbodyfont[verdana]}}
    {\showbodyfont[verdana]}
```

Those familiar with defining typescripts, may notice a new feature. Instead of defining each instance, we start with loading some fallbacks using the \type {\setups} command.

```
\starttyping
\setups[font:fallback:sans]
\stoptyping
```

These setups are defined in \type {type-def.tex}. We could have used typescripts, but this is faster. The fallbacks:

source code of this document

```
\starttyping
\startsetups [font:fallback:sans]
  \definefontsynonym [Sans]            [DefaultFont]
  \definefontsynonym [SansBold]        [Sans]
  \definefontsynonym [SansItalic]      [Sans]
  \definefontsynonym [SansSlanted]     [SansItalic]
  \definefontsynonym [SansBoldItalic]  [Sans]
  \definefontsynonym [SansBoldSlanted] [SansBoldItalic]
  \definefontsynonym [SansCaps]        [Sans]
\stopsetups
\stoptyping
```

Because we group the font definitions in (font) classes, these definitions are (as usual) local to the definition of the \type {verdana} typeface definition.

While playing with this font |<|actually, we needed the Verdana because one of our customers wanted to use this rather platform specific font|>| I also decided to speed up typescript processing a bit.

First of all, there is now an option to quit parsing typescripts once the one we're interested in is found and executed.

```
\starttyping
\starttypescript [sometypeface] [texnansi,ec,...]

  \definetypeface [sometypeface] [rm] ...
  \definetypeface [sometypeface] [ss] ...
  ...

  \quittypescriptscanning

\stoptypescript
\stoptyping
```

Quitting only makes sense when the file is loaded early in the parsing stage, which happens to be true for the predefined typefaces.

Another speedup is to add the following directive to your local variant of \type {cont-sys.tex} or to your style.

source code of this document

```
\starttyping
\preloadtypescripts
\stoptyping

If we define one typeface, we normally need four passes over the
typescript files, one pass for rm, ss, tt and rm. Given:

\starttyping
\usetypescript [modern]  [texnansi]
\stoptyping

On my machine, in Februari 2005, I gain quite some gross
execution time as reported when I use the (command line)
\type {-time-statistics} directive. Times are in
milliseconds. The values are corrected for the time needed
to process an empty, which takes about 500 milliseconds.

\starttabulate[|lBj2|cj2|cj2|c|]
\NC method     \NC  \bf time \NC \bf gain \NC \bf percentage \NC \NR
\NC normal     \NC   950     \NC          \NC                \NC \NR
\NC quiting    \NC   800     \NC 150      \NC 15\%           \NC \NR
\NC preloading \NC   325     \NC 625      \NC 70\%           \NC \NR
\stoptabulate

When we mix typefaces in documents, we get similar results:

\starttyping
\usetypescript [modern]  [texnansi]
\usetypescript [palatino][texnansi]
\usetypescript [times]   [texnansi]
\stoptyping

We now get (rought estimates on multiple combined runs):

\starttabulate[|lBj2|cj2|cj2|c|]
\NC method     \NC  \bf time \NC \bf gain \NC \bf percentage \NC \NR
\NC normal     \NC  2700     \NC          \NC                \NC \NR
\NC quiting    \NC  2200     \NC  500     \NC 20\%           \NC \NR
\NC preloading \NC   800     \NC 1900     \NC 70\%           \NC \NR
\stoptabulate

Further speedups at the macro level are possible but
don't make much sense because we would gain only a few
milliseconds when handling many thousands of definitions,
```

source code of this document

```
which is not the reality. We may also consider  packing
keywords in macros, which would definitely speed up things
a bit as well as save memory, so maybe one day I will do
this. Currently, on my machine, preloading takes some
112.000 extra memory words, and using a macros instead
of strings can easily save us 20.000 memory words and at
the same time give us about 5\% speed improvement (of
course only for typescript handling {\em loading}).

\stop \page

\setups [listing] \setups [lastpage] \stoptext
```