

# Installing JadeTeX

**Frank Atanassow Christoph**  
Next Solution Co., Ltd.

**Sebastian P Q Rahtz**  
Oxford University Computing Services

**Simon Pepping**  
Elsevier Science B.V.

## Introduction

This document is based on a note by FC describing his experience installing JadeTeX with OpenJade 1.3 and TeX 1.06 (on Unix). It has since been expanded and checked by SR and SP. It is aimed to people who have perhaps some experience using TeX and LaTeX, enough to write up technical reports using simple macros, but know little or nothing about format files or the particulars of the TeX distribution structure, and consequently have a rough time with JadeTeX<sup>o</sup>

This document relates to JadeTeX version 2.11 or later.

To use JadeTeX, you first of all should compile and install OpenJade. The resulting **openjade** executable, used with the `-t tex` flag, will format an SGML/XML file and yield a TeX output file. Using **jadetex** or **pdfjadetex**, you can transform this into DVI or PDF; from DVI, you can use a program like **dvips** to get PostScript output. This note describes how to build and install the former two programs, **jadetex** and **pdfjadetex**.

## Increasing TeX capacities

The TeX output file uses a macro package, JadeTeX, built on top of LaTeX, just as LaTeX is a macro package sitting on top of plain TeX. It is probably possible to just stick an `\input` at the top of your `.tex` file and use the JadeTeX macros this way, but that would be slow since TeX would need to parse and compile the macro definitions every time you format your `.tex` file. So what we do instead is to use TeX (actually, **initex**) to read in the LaTeX format, followed by the JadeTeX package, once, and dump the compiled image to what is called a “format file.” This is the same way that LaTeX is usually employed. Once the format file is built and installed, it is easy to arrange for TeX to read it in quickly and automatically when you process a `.tex` file.

But, as ever, there are complications, relating to TeX's capacity restrictions. TeX is designed to use a fixed amount of resources to process documents; for example, there is a maximum number of strings that can be allocated, and a maximum stack size. If a processing run exceeds the default limits, TeX will complain and refuse to continue. Unfortunately, OpenJade's TeX backend tends to exceed these default limits.

Fortunately, though, there is no need to recompile your TeX binary. If you are using teTeX (or any other Web2c-based distribution) there will be a file called `texmf.cnf` in your installation which sets the capacity parameters and is consulted every time TeX is run. By adding the right parameter settings here, you can ensure that JadeTeX will be unlikely to run out of memory.

Where is `texmf.cnf`? You can find it in the `web2c` directory under your `texmf` tree... which begs the question, where is `texmf`? The most straightforward way to find it is to make sure all the TeX executables are in your `PATH`; then do:

```
$ kpsewhich -expand-var '$TEXMFMAIN'  
/usr/share/texmf/web2c/texmf.cnf
```

and it will respond with the location of your `texmf` tree. As you can see, it is also known as `$TEXMFMAIN`, which is how I shall refer to it in the sequel.

If for some reason **kpsewhich** is not in your `PATH`, and you don't know where it is, here are some likely locations:

```
/usr/share/texmf  
/usr/local/share/texmf  
/usr/local/teTeX/texmf  
/usr/local/lib/teTeX/texmf  
/usr/local/lib/texmf  
/usr/lib/texmf  
/usr/lib/teTeX/texmf
```

In a minute we are going to modify `texmf.cnf` to increase the capacities, then build the JadeTeX format file and install the JadeTeX executable(s). But before we go further, check if you have an executable named **hugelatex**:

```
$ which hugelatex
```

If you are using teTeX, the answer is: probably not.

**hugelatex** is a version of `latex` with greater capacity settings than the usual one, which is named simply **latex**. You need to build your format file using **hugelatex** rather than **latex**, so that JadeTeX inherits the 'huge' settings. In addition, if you make any changes to the literate source of the JadeTeX macro package (see below), a normal **latex** will not do.

First we need to update `texmf.cnf` to ensure that **hugelatex** really is “huge”. First, make a backup:

```
$ cp texmf.cnf texmf.cnf.orig
```

Then take a look at the top of the file. It will probably say:

```
% original texmf.cnf - runtime path configuration file for kpathsea.  
% (If you change or delete 'original' on the previous line, the  
% distribution won't install its version over yours.)
```

If it says

```
% TeX Live texmf.cnf
```

then you can skip all this and proceed to the next section, as TeX Live is already set up for JadeTeX

If you have an ‘original’, follow those directions and delete the string `original` to ensure that future upgrades won’t obliterate your changes.

In the latter half of the file, you will find the capacity settings, which look something like this:

```
pool_size = 125000  
pool_size.context = 750000
```

Here, `pool_size` is the name of the parameter in both cases, but the second one is qualified with `.context`, which indicates that this setting will be preferred when using the ConTeXt macro package.

We need to make similar accommodations not only for **hugelatex** but also **jadetex** and **pdfjadetex**.

Unfortunately, we don’t know the minimal required values for every parameter and some of these values are probably ridiculously high, but no matter ...

```
% hugelatex settings  
main_memory.hugelatex = 1100000  
param_size.hugelatex = 1500  
stack_size.hugelatex = 1500  
hash_extra.hugelatex = 15000  
string_vacancies.hugelatex = 45000  
pool_free.hugelatex = 47500  
nest_size.hugelatex = 500  
save_size.hugelatex = 5000  
pool_size.hugelatex = 500000  
max_strings.hugelatex = 55000  
font_mem_size.hugelatex= 400000
```

```
% jadetex & pdfjadetex  
main_memory.jadetex = 1500000  
param_size.jadetex = 1500
```

```
stack_size.jadetex = 1500
hash_extra.jadetex = 50000
string_vacancies.jadetex = 45000
pool_free.jadetex = 47500
nest_size.jadetex = 500
save_size.jadetex = 5000
pool_size.jadetex = 500000
max_strings.jadetex = 55000

main_memory.pdfjadetex = 2500000
param_size.pdfjadetex = 1500
stack_size.pdfjadetex = 1500
hash_extra.pdfjadetex = 50000
string_vacancies.pdfjadetex = 45000
pool_free.pdfjadetex = 47500
nest_size.pdfjadetex = 500
save_size.pdfjadetex = 5000
pool_size.pdfjadetex = 500000
max_strings.pdfjadetex = 55000
```

Add these to the end of the file, or wherever makes you happy.

Now let's build some format files. Under `$TEXMFMAIN/tex/latex/config`, you will find the files necessary to rebuild **latex**. Copy them somewhere temporary and go there:

```
$ cp -R /usr/share/texmf/tex/latex/config /tmp
$ cd /tmp/config
```

Now do this:

```
$ tex -ini -programe=hugelatex latex.ini
```

This will produce a file `latex.fmt` in the same directory. Rename this to `hugelatex.fmt`, then become root and put it in `$TEXMFMAIN/web2c`. This is where all the format files are kept. (You can delete the other copied files from `config` afterwards.)

```
$ mv hugelatex.fmt /usr/share/texmf/web2c
```

Now just create a symbolic link from **tex** to **hugelatex**:

```
$ ln -s /usr/bin/tex /usr/local/bin/hugelatex
```

When **tex** is invoked, it looks at the name *X* it was invoked with, then loads the format file *X.fmt* from *\$TEXMFMAIN/web2c* before it starts processing the document. So creating this symbolic link is all that is needed to create the **hugelatex** executable.

## Creating the format files

Next, take a look at your OpenJade distribution. Under the directory *dsss1* you will find the files necessary to build *jadetex* and *pdfjadetex*, including a Makefile. If you don't have, and neglected to build, a **hugelatex**, then you will first need to edit the Makefile and replace **hugelatex** with **latex**. Otherwise, become root and just do:

```
$ make install
```

This creates *jadetex.fmt* and *pdfjadetex.fmt*, puts them in *\$TEXMFMAIN/web2c* for you (using **kpsewhich** to find *\$TEXMFMAIN*), and installs a few other auxiliary files under *\$TEXMFMAIN/tex/jadetex*.

All that's left to do is to create the links:

```
$ ln -s /usr/bin/tex /usr/bin/jadetex
$ ln -s /usr/bin/pdftex /usr/bin/pdfjadetex
```

(changing the location of the binary directory to wherever your TeX is) and run **mktexlsr** so that your TeX distribution becomes aware of the newly installed files in *\$TEXMFMAIN/tex/jadetex*.

```
$ mktexlsr
```

## Testing the installation

Finally, test your installation using the demonstration files in that directory:

```
$ openjade -t tex -d demo.dsl demo.sgm
$ jadetex demo.tex
$ pdfjadetex demo.tex
```

You're done!

## Frequently Asked Questions

1. I installed JadeTeX, but when I run it, it complains that a file named `unicode.sty` (or `dsssl.def`, etc) can't be found. What did I do wrong?

You didn't run the **mktexlsr**, so **kpathsea** doesn't know about the newly installed files.

2. I don't like JadeTeX's default behavior in some situations. How do I modify it?

If for some reason you want to modify the JadeTeX macro package, modify the file `jadetex.dtx`. This is the literate source for the format file and other files installed under `$TEXMFMAIN/tex/jadetex`.

To format it, use **hugelatex**:

```
$ hugelatex jadetex.dtx
```

You will get tons of overfull hboxes but if you are using **hugelatex** rather than **latex**, it will work.

Formatting the batch file `jadetex.ins` will produce stripped sources (`dsssl.def` and `jadetex.ltx`), which can be compiled into format files as before:

```
$ hugelatex jadetex.ins
$ make -f Makefile.jadetex
```

3. What fonts can I use?

Following is the names of the font families supported at the time of writing. Of course you must actually have these fonts installed to format the document (but not to produce the TeX output).

- Arial
- Helvetica
- Palatino
- Bookman
- Courier
- Symbol
- Wingdings
- WingDings
- LucidaSans
- LucidaBright
- Savoy
- ACaslon

Caslon  
Formata  
FranklinGothic  
OCRAbyBT  
AGaramond  
Avant-Garde  
Courier-New  
New-Century-Schoolbook  
Times-Roman  
Trade-Gothic  
Times-New-Roman  
Times-NR-MT  
Courier-New  
Zapf-Dingbats  
Gill-Sans  
iso-serif  
iso-sanserif  
iso-monospace  
LetterGothic12PitchBT  
Monospace821  
OCRB10PitchBT  
OCR-A  
OCR-B-10PitchBT  
Computer-Modern-Typewriter  
Computer-Modern-Sans  
Computer-Modern  
Computer-Modern-Caps-And-Small-Caps

#### 4. Why doesn't hyphenation work?

Remember that your text must be fully justified (`quadding: #t`), hyphenation must be on, `hyphenation?: #t`, and a current language must be selected (e.g., `language: 'EN`) for JadeTeX to perform hyphenation.

#### 5. I'm using Norman Walsh's DocBook stylesheets and my footnotes are coming up at the end of the document rather than at the foot of each page. Why?

The DocBook stylesheets need to be made aware that you are using the TeX backend. Add `-v tex-backend` to your **openjade** command-line.

