NON-PROGRAMMER GUIDE TO ADDING DUMMY DSIG

For arbitrary or unknown reasons, Microsoft insist on having a DSIG (Digitally Signature) code in OpenType-TrueType fonts - otherwise OpenType 'features' built into fonts are not accessible in applications.

If a font has no DSIG table it will support only the default letters, and OpenType features will not be recognized by the Windows operating system. Being an extension of the TrueType format, many OpenType tables are optional. Windows checks for the presence or absence of a DSIG table to decide whether a font is OpenType (OTF) or TrueType (TTF).

Digital Signatures can be obtained from various sources on a per annum paid-for basis. But if you are prepared to delve into the world of Shell scripting, there is a method that can insert a “dummy” DSIG into your existing TrueType fonts. You’ll notice that the new file no longer has its TT icon. But it’s a small sacrifice if it enables OpenType ‘features’ to be accessible in applications.

After many hours of trial, error, advice, more trials, more errors, more advice, and with grateful thanks to Michel Boyer at Typophile http://www.typophile.com for helping to create these scripts, here I describe how to use three recipes for success. You may want to just add a DSIG to one for, or to many hundreds of fonts in a Folder/Sub-folder structure. Pick which recipe suits your situation.

You’ll use Terminal (Mac OS) or Command Prompt (Windows) to run Shell scripts.

THE RECIPES
1. Single ttf (merges DSIG into a single font)
2. Folder ttf (merges DSIG into a all fonts in a Folder)
3. Multiple Folder ttf (merges DSIG into a all fonts in Folders and Sub-folders)

WINDOWS COMMAND PROMPT
Windows offers numerous ways to open a session in Command Prompt depending on your current needs.

Method 1
Press "Win-R," type "cmd" and press "Enter" to open a Command Prompt session using just your keyboard.

Method 2
Click the "Start | Program Files | Accessories | Command Prompt" to open a Command Prompt session using just your mouse.

Method 3
Click the "Start" button and type "cmd." Right-click "Cmd," select "Run as Administrator" and click "Yes" to open Command Prompt with elevated privileges.

Method 4
Hold the "Shift" key, right-click a folder in Windows Explorer and select "Open Command Window Here" to open the Command Prompt at the selected folder's hard drive location.

MAC OS TERMINAL
Step 1
Navigate to: Applications/Utilities/Terminal

Step 2
Double-click to launch.

Step 3
From the keyboard, press Command + N to open a new session window using just your mouse.

TTX
Check if ttx executable is installed in an FDK tools folder on your computer. If you get no answer, ttx is not installed and the first thing to do is to install ttx. You should have it if you properly installed AFDKO… http://www.adobe.com/devnet/opentype/afdko.html

In the Terminal window, after the "prompt" (probably %) you type "which ttx" then press Return to get a result as follows:

```
which ttx
/Users/yourname/bin/FDK/Tools/osx/ttx
```

A path ending with “ttx” means you’re good to go. You can then copy code snippet (the code between the cut lines) and paste it in the terminal window.

```
SINGLE TTF RECIPE-------------cut line
FLDR="${HOME}/Desktop/Dsig fldr"
mkdir -p "$FLDR"
cd "$FLDR"
cat > dsig.ttx<<EOF
<?xml version="1.0" encoding="ISO-8859-1"?>
<ttFont sfntVersion="\x00\x01\x00\x00" ttLibVersion="2.3">

  <DSIG>
    <hexdata>
      00000001 00000000
    </hexdata>
  </DSIG>

</ttFont>
EOF

open .
END-------------cut line
```

Dsig fldr
After the paste, you may need to type a carriage return. That will create a Folder on your Desktop entitled ‘Dsig fldr’ containing the file **dsig.ttx** which will be used in the recipes. Now you can begin the merging of DSIG and TTF files. You’ll need a few windows open.

SINGLE TTF RECIPE

1. Launch Terminal.

2. Type: which ttx

3. Return
You'll get something like this displayed (on an iMac for example):
```
/Users/yourhome/bin/FDK/Tools/osx/ttx
your-imac:Dsig fldr yourhome$
```

4. Drag TTX app to Terminal (since it accepted your Dsig fldr snippet) and displays a path:
```
/Applications/FDK/Tools/osx/ttx
```

4. Type: -m (that's space hyphen m space).
First remove all of the dsig#.ttf files in the Dsig fldr if there are any.
Then for each font file:

5. Drag the ttf font file to Terminal (say MyFontt.ttf), to see a path something like:
```
/Users/yourhome/Documents/\ CLIENTS\ MYJOBS/ClientA/Test\ Data/OT/OT-TT/
MyFont.ttf
```

6. Drag dsig.ttx to terminal to see something like:
```
/Users/yourhome/Desktop/Dsig\ fldr_orig/dsig.ttx
```

7. Return

That will compile only dsig.ttx, merge the DSIG into MyFont.ttf and produce a file named dsig.ttf in the Dsig fldr, so now…
8. Rename dsig.ttf as MyFont something.ttf (original TrueType filename something)
Don't switch steps 3 and 4. You need to put the .ttf file before the .ttx file.

FOLDER TTF RECIPE
Another way is to define a variable TTX that contains the full path of the application. For instance, to handle 70 fonts in some folder with Adrian's current configuration, what can be done is this (make sure there is no dsig.ttf font in those 70 fonts).

1. type cd with a space in the terminal window

2. drag the folder (not the fonts) containing the ttf fonts in the terminal window
3. return

4. paste the following lines in the terminal window:

```
FOLDER TTF RECIPE-------------cut line
FLDR="$HOME/Desktop/Dsig fldr"
TTX=$FDK_EXE/ttx
for ttffont in *.ttf
do
  ttxsig="$FLDR/${ttffont%.ttf}.ttx"
```

```
  cp "$FLDR"/dsig.ttx "$ttxsig"
  $TTX -m "$ttffont" "$ttxsig"
  rm -f "$ttxsig"
done
END-------------cut line
```

5. return

The 70 font files should now be in Dsig fldr with the SAME name they previously had but with a DSIG merged into them.

To be complete, I should add that it is assumed that the Dsig folder already exists on the desktop and contains dsig.ttx for ttx 2.4. Such a folder had previously been created by pasting in a terminal window the following lines:

MULTIPLE FOLDER TTF RECIPE
Usually scripts are not pasted in the terminal window. They are kept in files with a name and you simply type the name with the arguments. Now that you installed AFDKO (with TTX 2.4), you could install such a script addDSIG that adds a dummy DSIG to a ttf file. To make that installation, just paste the following lines in a terminal window:

```
#------ cut line
cat > "$FDK_EXE/addDSIG"<<'EOFA'
#!/bin/sh

if [[ $# -eq 0 || $1 == '-u' || $1 == '-h' ]]
then
  cat <<EOF
Usage: addDSIG <ttf font file>
       adds a dummy DSIG to the ttf font file
EOF
  exit 0
fi

cp "$FDK_EXE"/../SharedData/dsig24.ttx "${1%.ttf}.ttx"
mv "$1" "${1%.ttf}old.ttf"
"$FDK_EXE"/ttx -m "${1%.ttf}old.ttf" "${1%.ttf}.ttx"
rm -f "${1%.ttf}old.ttf" "${1%.ttf}.ttx"
EOFA
chmod 755 "$FDK_EXE/addDSIG"

cat > "$FDK_EXE/../SharedData/dsig24.ttx" <<'EOFB'
<?xml version="1.0" encoding="utf-8"?>
<ttFont sfntVersion="\x00\x01\x00\x00" ttLibVersion="2.4">
  <DSIG><tableHeader flag="0x0" numSigs="0" version="1"/></
DSIG>
</ttFont>
EOFB
#------ cut line
```

That adds the script **addDSIG** in the FDK_EXE folder entitled "**dsig24.ttx**"

Copy the file dsig24.ttx and place in the Dsig fldr (where the dsig.ttx file is too).

With the unix find command, you can process ALL the ttf fonts contained in the current folder AND its subfolders by just typing or pasting the one line command

```
find . -name "*.ttf" -exec adddsig "{}" \;
```

As usual, to get to that folder in the terminal window you:

1. Type cd with a space

2. Drag the folder

3. Return
Now your current directory is that folder;

4. Now you can type (or paste)
```
find . -name "*.ttf" -exec adddsig "{}" \;
```
to process all the ttf fonts therein (including all subfolders).

PS For those on Linux where case is relevant, always type addDSIG or rename $FDK_EXE/addDSIG as $FDK_EXE/adddsig to avoid the uppercase.

NOTE
If it is important for you too keep the original fonts, I should say that addDSIG replaces the original font file so it may be better to apply the command on a copy of the Folder/Sub-folder structure containg the font to be worked on.

The part of the script that actually does any processing is only four lines and one of those lines simply removes the original font (with "old" appended to its name) as well as the ttx file.