FontTools

Reviving an Open Source project, ~~Re~~building a thriving community

Behdad Esfahbod

ATypI Barcelona, 17 September 2014

History

- Started in 1999ish
- by Just van Rossum
- of LettError fame
- Slowed down by 2004
- I picked it up last year

What is it?

- Two things: TTX and fontTools
- TTX converts fonts to XML and back
- fontTools is a Python library
- Font engineer's calculator
- Closely reflecting OpenType tables
- Minimum abstractions


TTX
ATypI Barcelona, 17 September 2014

```
<?xml version="1.0" encoding="utf-8"?>
<ttFont sfntVersion="\x00\x01\x00\x00" ttLibVersion="2.5">

  <GlyphOrder>
    <!-- The 'id' attribute is only for humans; it is ignored when parsed. -->
    <GlyphID id="0" name=".notdef"/>
    <GlyphID id="1" name="T"/>
    <GlyphID id="2" name="X"/>
  </GlyphOrder>

  <head>
    <!-- Most of this table will be recalculated by the compiler -->
    <tableVersion value="1.0"/>
    <fontRevision value="1.001"/>
    <checkSumAdjustment value="0x22ff1f02"/>
    <magicNumber value="0x5f0f3cf5"/>
    <flags value="00000000 00001011"/>
    <unitsPerEm value="1000"/>
    ...
  </head>

  <hhea>
    <tableVersion value="1.0"/>
    <ascent value="863"/>
    <descent value="-228"/>
    <lineGap value="0"/>
    ...
  </hhea>
```

```
<maxp>
  <!-- Most of this table will be recalculated by the compiler -->
  <tableVersion value="0x10000"/>
  <numGlyphs value="3"/>
  <maxPoints value="68"/>
  <maxContours value="7"/>
  ...
</maxp>

<OS_2>
  <version value="3"/>
  <xAvgCharWidth value="560"/>
  <usWeightClass value="400"/>
  <usWidthClass value="5"/>
  <fsType value="00000000 00000000"/>
  ...
</OS_2>
```

```
<hmtx>
  <mtx name=".notdef" width="260" lsb="0"/>
  <mtx name="T" width="576" lsb="22"/>
  <mtx name="X" width="584" lsb="26"/>
</hmtx>

<cmap>
  <tableVersion version="0"/>
  <cmap_format_4 platformID="3" platEncID="1" language="0">
    <map code="0x54" name="T"/><!-- LATIN CAPITAL LETTER T -->
    <map code="0x58" name="X"/><!-- LATIN CAPITAL LETTER X -->
  </cmap_format_4>
</cmap>

<loca>
  <!-- The 'loca' table will be calculated by the compiler -->
</loca>
```

```
<glyf>

  <!-- The xMin, yMin, xMax and yMax values
       will be recalculated by the compiler. -->

  <TTGlyph name=".notdef"/><!-- contains no outline data -->

  <TTGlyph name="T" xMin="22" yMin="0" xMax="554" yMax="715">
    <contour>
      <pt x="264" y="673" on="1"/>
      <pt x="22" y="673" on="1"/>
      <pt x="22" y="715" on="1"/>
      <pt x="554" y="715" on="1"/>
      <pt x="554" y="673" on="1"/>
      <pt x="312" y="673" on="1"/>
      <pt x="312" y="0" on="1"/>
      <pt x="264" y="0" on="1"/>
    </contour>
    <instructions><assembly>
      </assembly></instructions>
  </TTGlyph>

  <TTGlyph name="X" xMin="26" yMin="0" xMax="558" yMax="715">
    ...
  </TTGlyph>

</glyf>
```

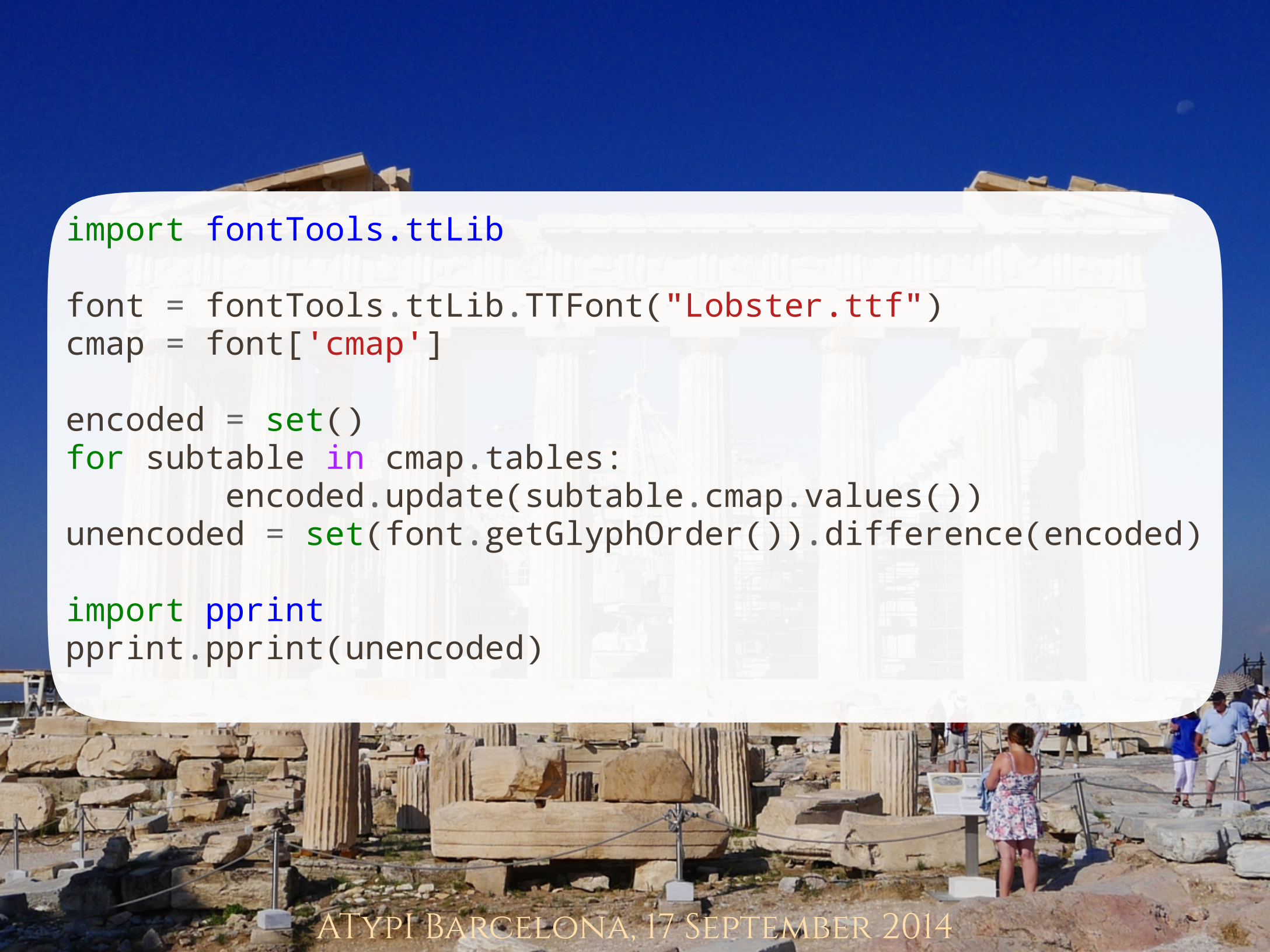
```
  <name>
    <namerecord nameID="1" platformID="3" platEncID="1" langID="0x409">
      Julius Sans One
    </namerecord>
    <namerecord nameID="2" platformID="3" platEncID="1" langID="0x409">
      Regular
    </namerecord>
  </name>

  <post>
    <formatType value="3.0"/>
    <italicAngle value="0.0"/>
    <underlinePosition value="-75"/>
    <underlineThickness value="50"/>
    <isFixedPitch value="0"/>
    <minMemType42 value="0"/>
    <maxMemType42 value="0"/>
    <minMemType1 value="0"/>
    <maxMemType1 value="0"/>
  </post>

</ttFont>
```

FontTools

ATypI Barcelona, 17 September 2014

from fontTools import

- afmLib
- cffLib
- ~~fondLib~~
- ~~nfntLib~~
- t1Lib
- **ttLib**



```
import fontTools.ttLib

font = fontTools.ttLib.TTFont("Lobster.ttf")
cmap = font['cmap']

encoded = set()
for subtable in cmap.tables:
        encoded.update(subtable.cmap.values())
unencoded = set(font.getGlyphOrder()).difference(encoded)

import pprint
pprint.pprint(unencoded)
```


set(['A.salt',
'B.salt',
'E.salt',
'E_x',
'F_i',
'N.salt',
'NULL',
'T_h',
'T_i',
'a.end',
...
'y_z'])
ATypI Barcelona, 17 September 2014

Framework

- Compiler / decompiler
- XML serialization
- Optimizer
- Library
- A tool to build products with
- Not an end product

Philosophy

- Minimal abstraction
- Hide byte layout, redundancy, etc
- Platform to build on

WHY?

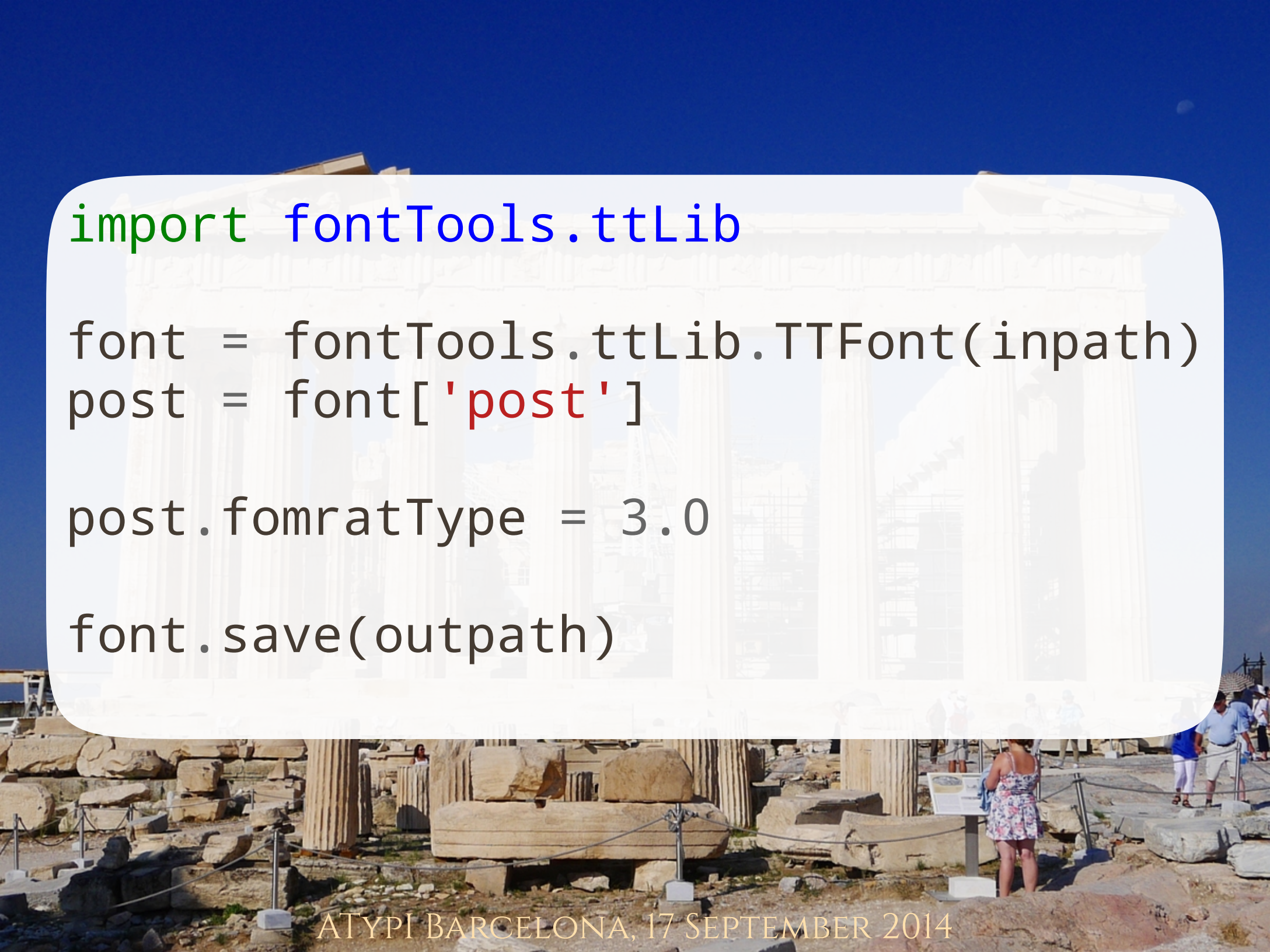
- Free Software
- Python (2.7+ & 3.x)
- Portable
- No dependencies
- Non-destructive
- Batch-friendly

What for?

- Figuring out what's inside a font
- Diff'ing font versions
- Final steps of producing fonts
- Hotfixing existing fonts
- Reporting and analysis over *many* fonts
- Server-side reporting and manipulation
- Readable VCS-friendly font source code
- Subsetting and merging fonts

Hotfixing

- XML and back
- Python recipes
- Interactive Python



```
import fontTools.ttLib

font = fontTools.ttLib.TTFont(inpath)
post = font['post']

post.fomratType = 3.0

font.save(outpath)
```

```
import fontTools.ttLib

font = fontTools.ttLib.TTFont(inpath)
glyphnames = font.getGlyphOrder()

# ...do the actual renaming in glyphnames...

font = fontTools.ttLib.TTFont(inpath) # load again
font.setGlyphOrder(glyphnames)
post = font['post'] # make sure post table is loaded

font.save(outpath)
```

New work

- Bitmap tables, color tables, misc tables
- WOFF1 read and write
- More stable XML format
- More optimized font files
- Bug fixes; many of them
- Major speed-up
- New tools

Tools

- pyftsubset
- pyftmerge
- pyftinspect

PYFTSUBSET

- OpenType font subsetter & optimizer
- TrueType and CFF flavored
- SFNT and WOFF1
- Full OpenType Layout support
- File-size optimizations
- Designed for webfont productions
- Used for sub-family instantiation

```
$ pyftsubset Lobster.ttf --text=Lobster --glyph-names --verbose
Text: 'Lobster'
Unicodes: []
Glyphs: []
Gids: []
cmap pruned
glyf pruned
...
Added gid0 to subset
Closing glyph list over 'GSUB': 8 glyphs before
Glyph names: ['.notdef', 'L', 'b', 'e', 'o', 'r', 's', 't']
Closed  glyph list over 'GSUB': 16 glyphs after
Glyph names: ['.notdef', 'L', 'b', 'b_s', 'b_s.end', 'e', 'e.end', 'o',
              'o.end', 'o_s', 'o_s.end', 'r', 's', 's.end', 't', 't.end']
Retaining 16 glyphs:
head subsetting not needed
...
hmtx subsetted
cmap subsetted
glyf subsetted
kern subsetted to empty; dropped
name subsetting not needed
GDEF subsetted
GPOS subsetted
GSUB subsetted
...
glyf pruned
GDEF pruned
GPOS pruned
GSUB pruned
Input  font: 140856 bytes: Lobster.ttf
Subset font:   4648 bytes: Lobster.ttf.subset
```

PYFTMERGE

- Font merging tool
- Very early prototype
- Fonts having same format, upem
- TrueType-flavored only
- Retains hinting from one font
- Handles conflicting characters
- Designed for Noto
- Used for merging Latin and non-Latin

```
$ pyftmerge NotoSansDevanagari-Regular.ttf NotoSansBengali-Regular.ttf
$ ls -l
-rw-r--r-- 1 behdad eng 266580 Sep 17 08:19 merged.ttf
-rw-r--r-- 1 behdad eng 188664 Sep 17 08:18 NotoSansBengali-Regular.ttf
-rw-r--r-- 1 behdad eng 177672 Sep 17 08:18 NotoSansDevanagari-Regular.ttf

$ pyftmerge NotoSansBengali-Regular.ttf NotoSansDevanagari-Regular.ttf
$ ls -l
-rw-r--r-- 1 behdad eng 293276 Sep 17 08:21 merged.ttf
-rw-r--r-- 1 behdad eng 188664 Sep 17 08:18 NotoSansBengali-Regular.ttf
-rw-r--r-- 1 behdad eng 177672 Sep 17 08:18 NotoSansDevanagari-Regular.ttf
```

FontTools

ATypI Barcelona, 17 September 2014

Strengths

- Supports wide array of tables
- Low-level, non-destructive
- Python
- Self-documenting
- Optimized output
- Extensible
- Supports all color tables

Weaknesses

- Supports wide array of tables
- Low-level, non-destructive
- Python
- Undocumented
- Always optimizes output
- Not well-extensible
- Supports many legacy tables

Users

- Google Fonts; misc foundries
- Font Bakery, AFDKO, etc
- Adam Twardoch; others
- …


Community
• https://github.com/behdad/fonttools/
• https://groups.google.com/forum/#!forum/fonttools
• http://sourceforge.net/projects/fonttools/

Future work

- More font optimization
- Better input: .fea / UFO
- ttx2ufo
- Lint tool
- TrueType / CFF conversion
- Subsetting more tables
- Better merging

Future work: optimization

- Optimal packing: glyf flags / PUSH / etc
- CFF outline operation specializer
- CFF subroutinizer
- TrueType outline optimizer? (fontcrunch)
- TrueType bytecode analysis
- UPEM reduction

Q?

ATypI Barcelona, 17 September 2014


ATypI Barcelona, 17 September 2014


cff = font['CFF '].cff.topDictIndex[0]
cff.ROS = ('Adobe','Identity',0)
mapping = {name:("cid"+str(n) if n else ".notdef") for n,name in enumerate(cff.charset)}
charstrings = cff.CharStrings
charstrings.charStrings = {mapping[name]:v for name,v in charstrings.charStrings.items()}
cff.charset = ["cid"+str(n) if n else ".notdef" for n in range(len(cff.charset))]
ATypI Barcelona, 17 September 2014

```
<cmap raw="1">
  <!-- Hand-coded format13 table mapping all Unicode
       characters to glyph00001. -->
  <hexdata>
    0000 0001 <!-- version numTables -->
      0003 000A <!-- platformID encodingID -->
      0000000C  <!-- offset -->
        000D 0000 <!-- format reserved -->
        0000001C  <!-- length -->
        00000000  <!-- language -->
        00000001  <!-- nGroups -->
          00000001  <!-- startCharCode -->
          0010FFFE  <!-- endCharCode -->
          00000001  <!-- glyphID -->
  </hexdata>
</cmap>
```