

1 Intro to Python
2 for Type Design!

3

4

5

6 Type@Cooper

7 2025

Why RoboFont?



UFO

Unified Font Object



UFO 3

metainfo.plist
fontinfo.plist
groups.plist
kerning.plist
featuresfea
lib.plist
layercontents.plist
glyphs
contents.plist
layerinfo.plist
glif
images
data
conventions

Open-source & editor- agnostic



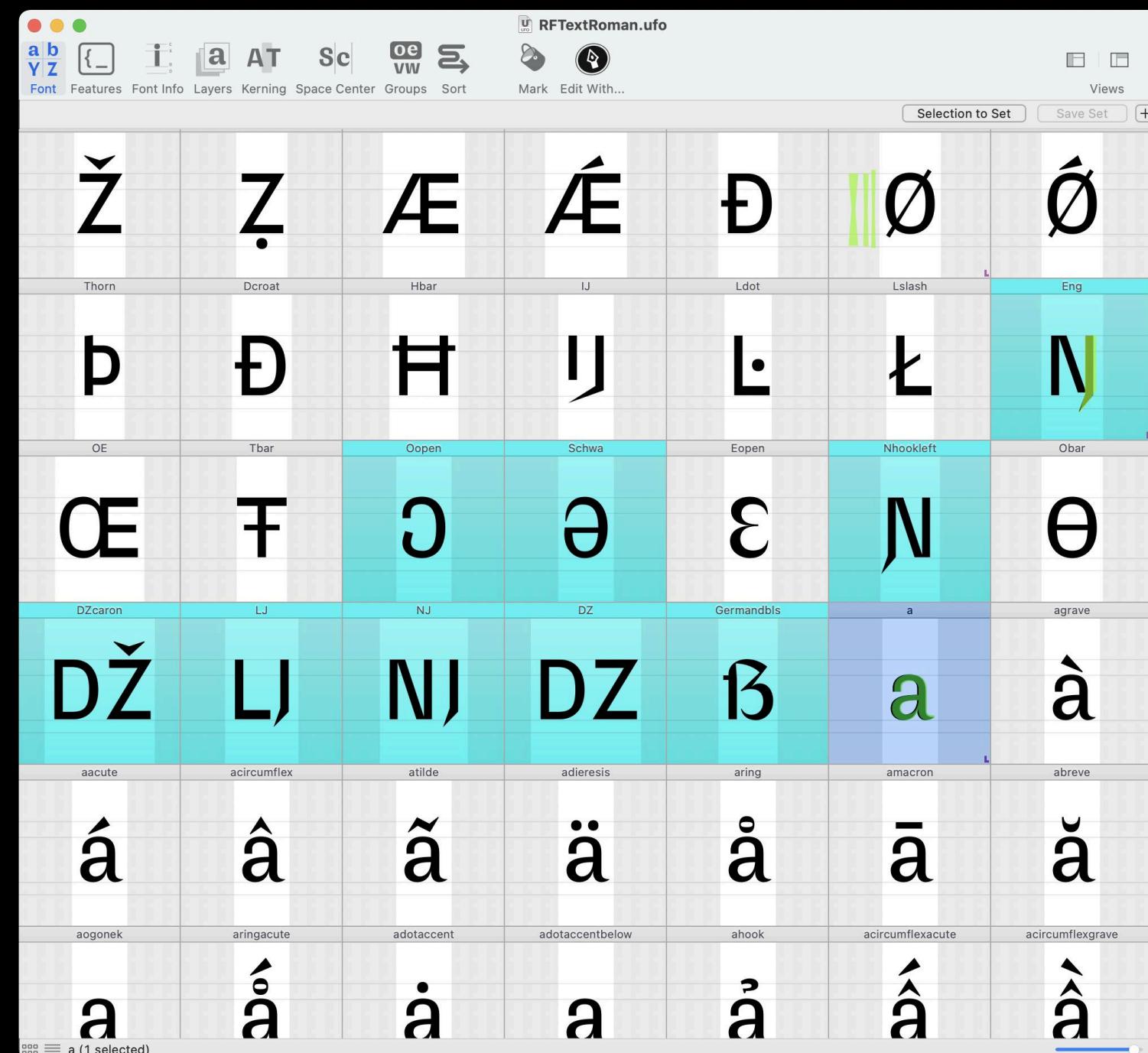
UFO 3

- metainfo.plist
- fontinfo.plist
- groups.plist
- kerning.plist
- featuresfea
- lib.plist
- layercontents.plist
- glyphs
- contents.plist
- layerinfo.plist
- glif
- images
- data
- conventions

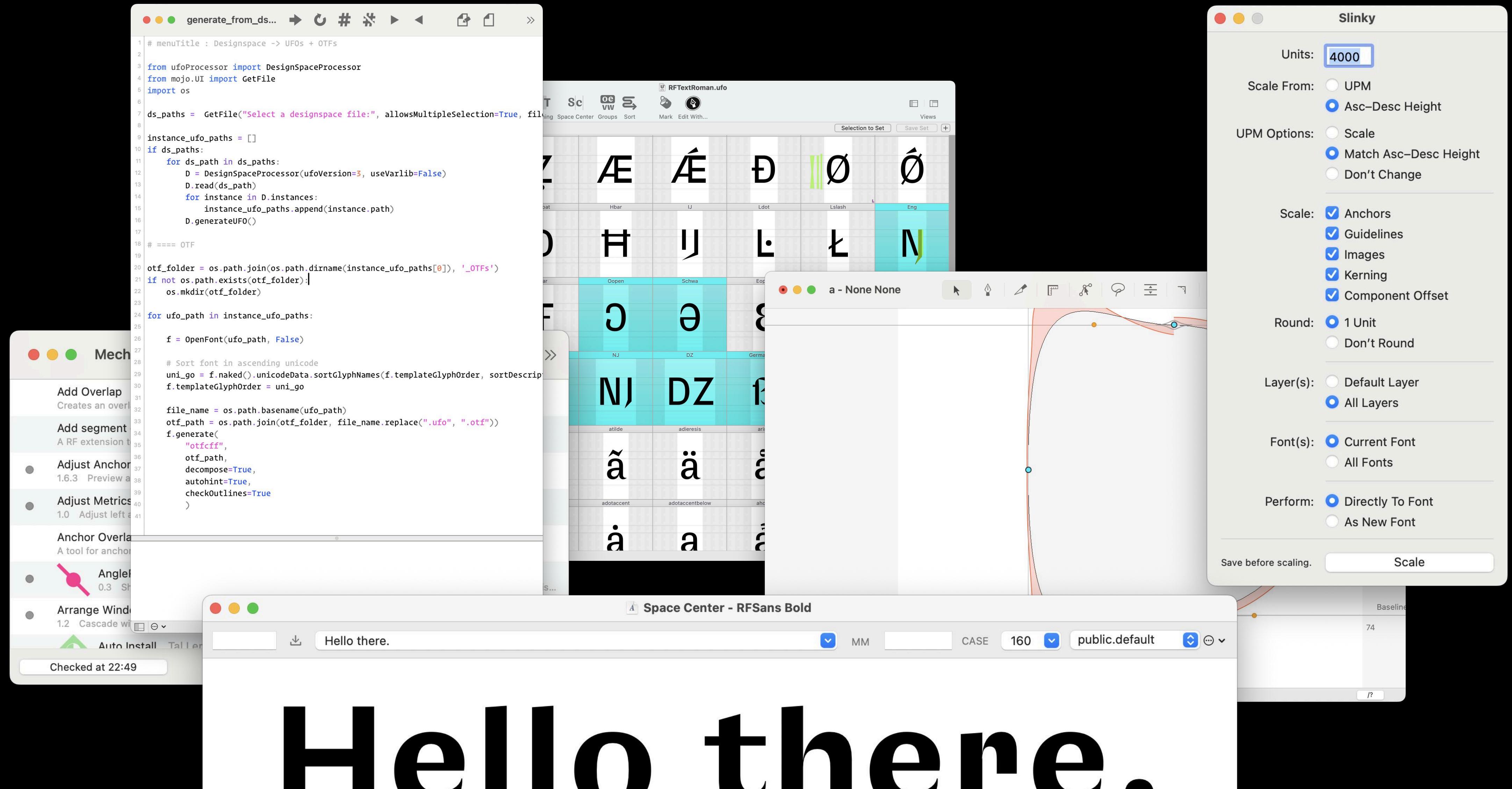
RoboFont is a
simple UFO editor.



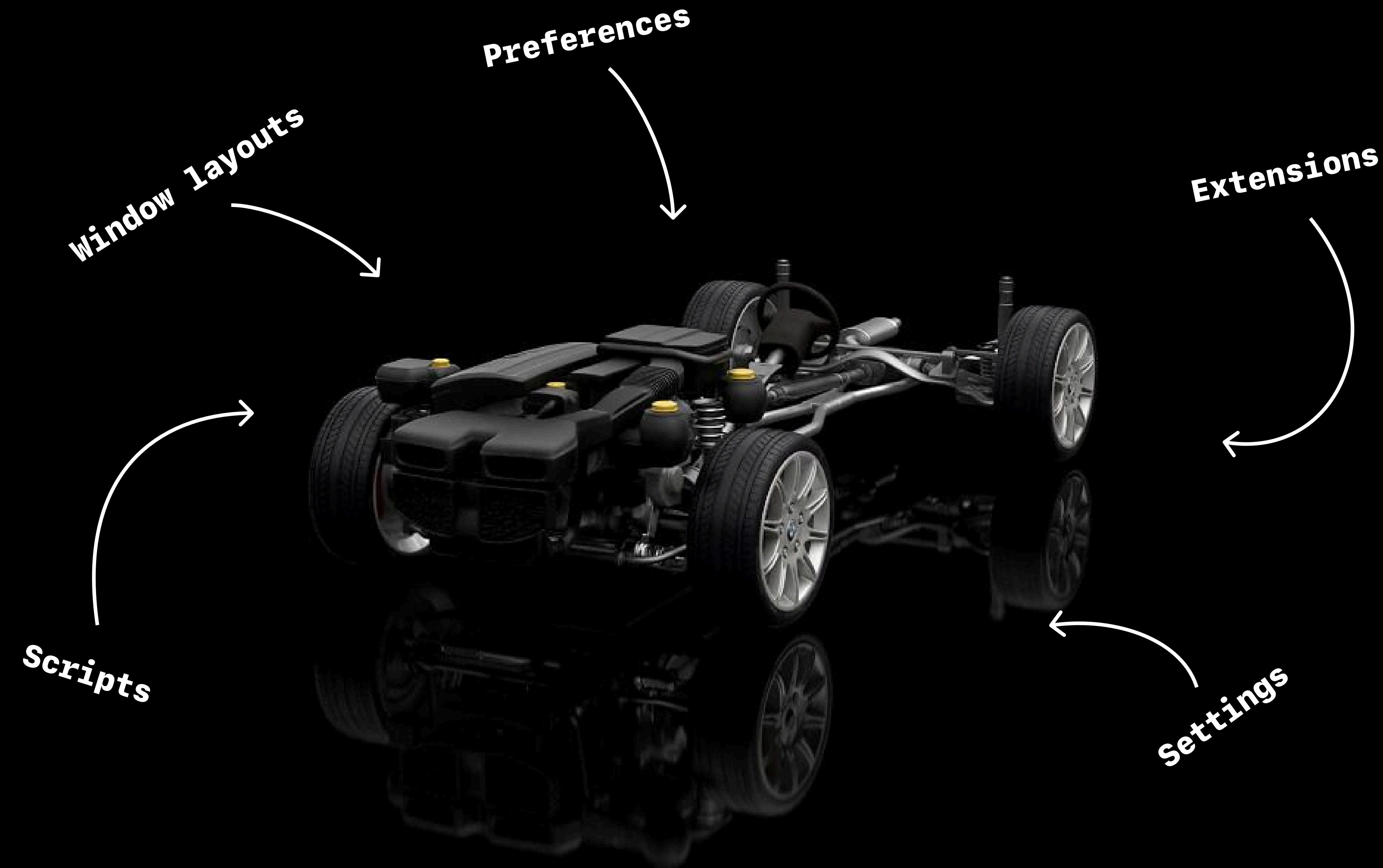
Built from scratch, in Python.



With flexibility in mind.









"Why is the steering wheel shaped this way?" ©

"There is only one cupholder and it's too small."

"I don't need a built-in water dispenser"

"The seat doesn't go high enough."

Let's use this
as a playground.



Font time!

1

General Python

2

**DrawBot-specific
Python**

3

**Font-specific
Python**

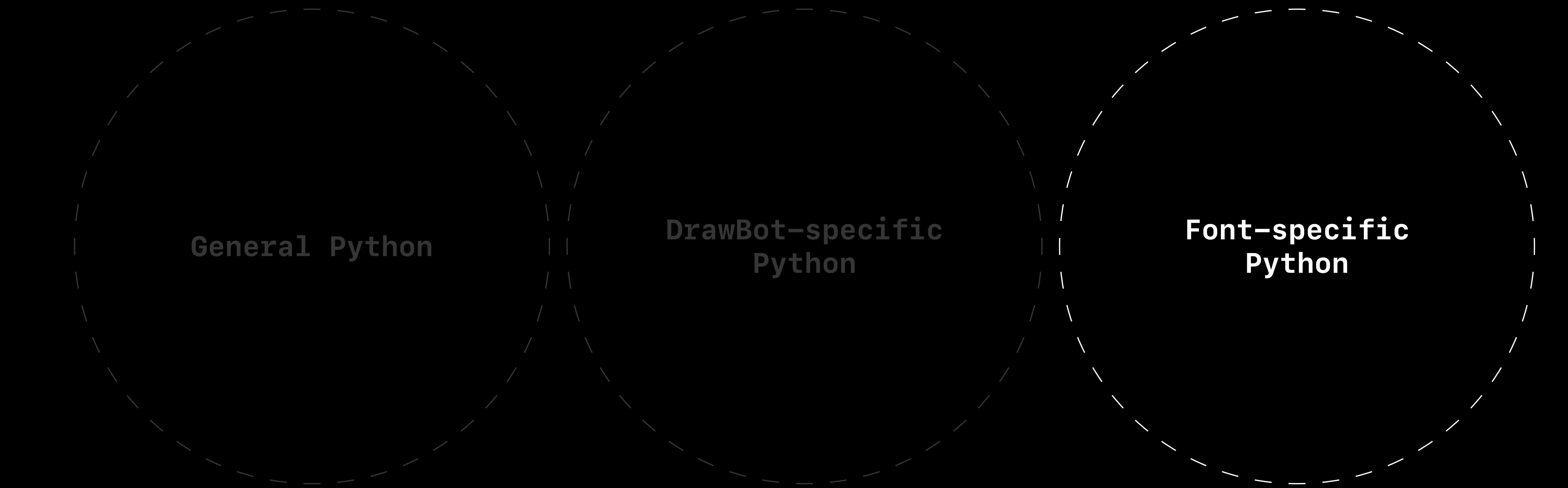
Done

Done

General Python

**DrawBot-specific
Python**

**Font-specific
Python**



Coding in RoboFont

General Python

DrawBot-specific
Python

Font-specific
Python

From DrawBot API to FontParts!

DrawBot

Shapes
Primitives
Drawing Paths
Path Properties
BezierPath
Colors
Canvas
Text
Images
Variables
Quick Reference

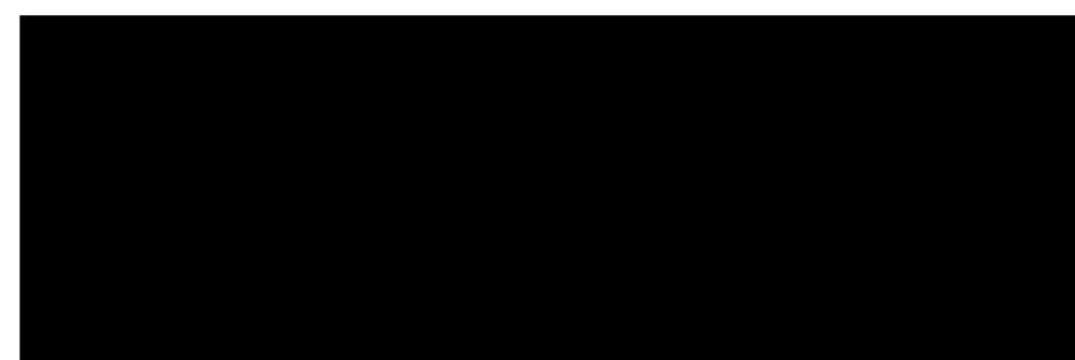
Primitives

rect(x, y, w, h)

Draw a rectangle from position x, y with the given width and height.

```
# draw a rectangle
#   x   y   w   h
rect(100, 100, 800, 800)
```

[Open in DrawBot: rect.py](#)
[Download: rect.py](#)



fontParts.base.BaseFont.copy

TYPE DESIGNERS

Getting Started

Object Reference

Objects
Font
Info
Groups
Kerning
Features
Lib
Layer
Glyph
Contour
Segment
bPoint
Point
Component
Anchor
Image
Guideline

Common Value Types
fontParts.world

SOFTWARE DEVELOPERS

Implementing FontParts
Developing FontParts

Overview

Copy

[BaseFont.copy](#) Copy the font into a new font.

File Operations

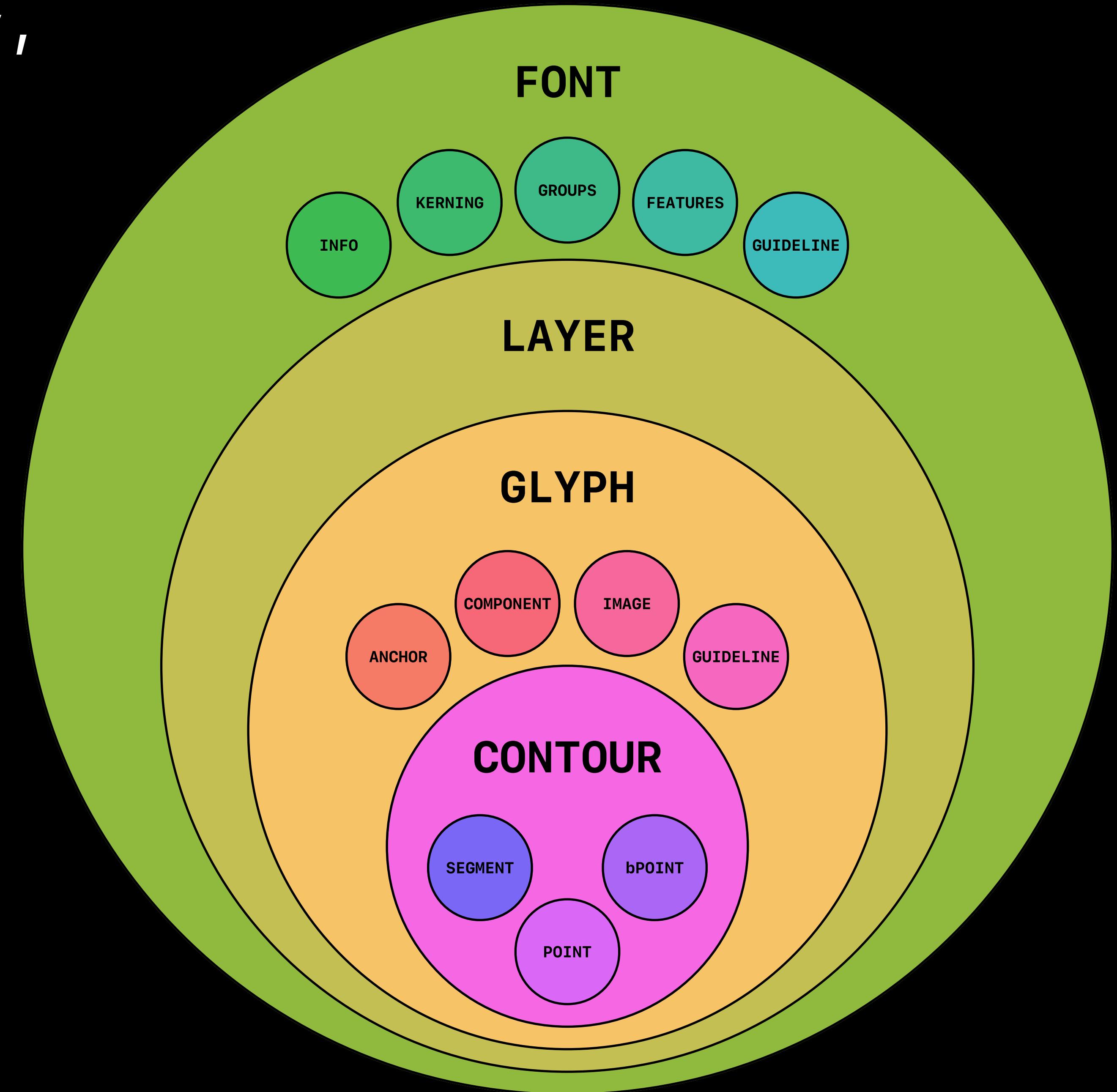
BaseFont.path	The path to the file this object represents.
BaseFont.save	Save the font to path .
BaseFont.generate	Generate the font to another format.

Sub-Objects

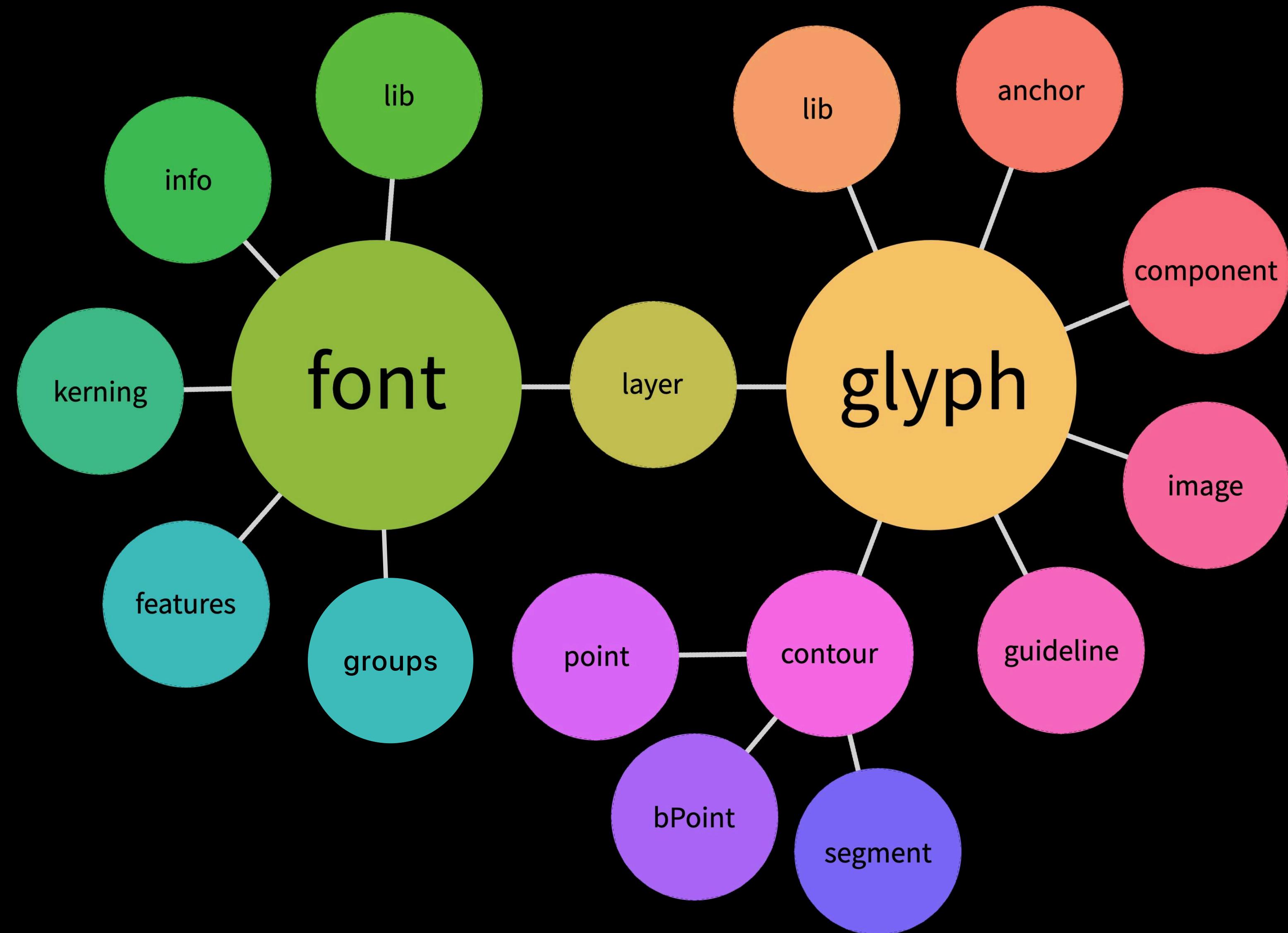
BaseFont.info	The font's BaseInfo object.
BaseFont.groups	The font's BaseGroups object.
BaseFont.kerning	The font's BaseKerning object.
BaseFont.features	The font's BaseFeatures object.
BaseFont.lib	The font's BaseLib object.
BaseFont.tempLib	The font's BaseLib object.

`fontparts.robotools.dev`

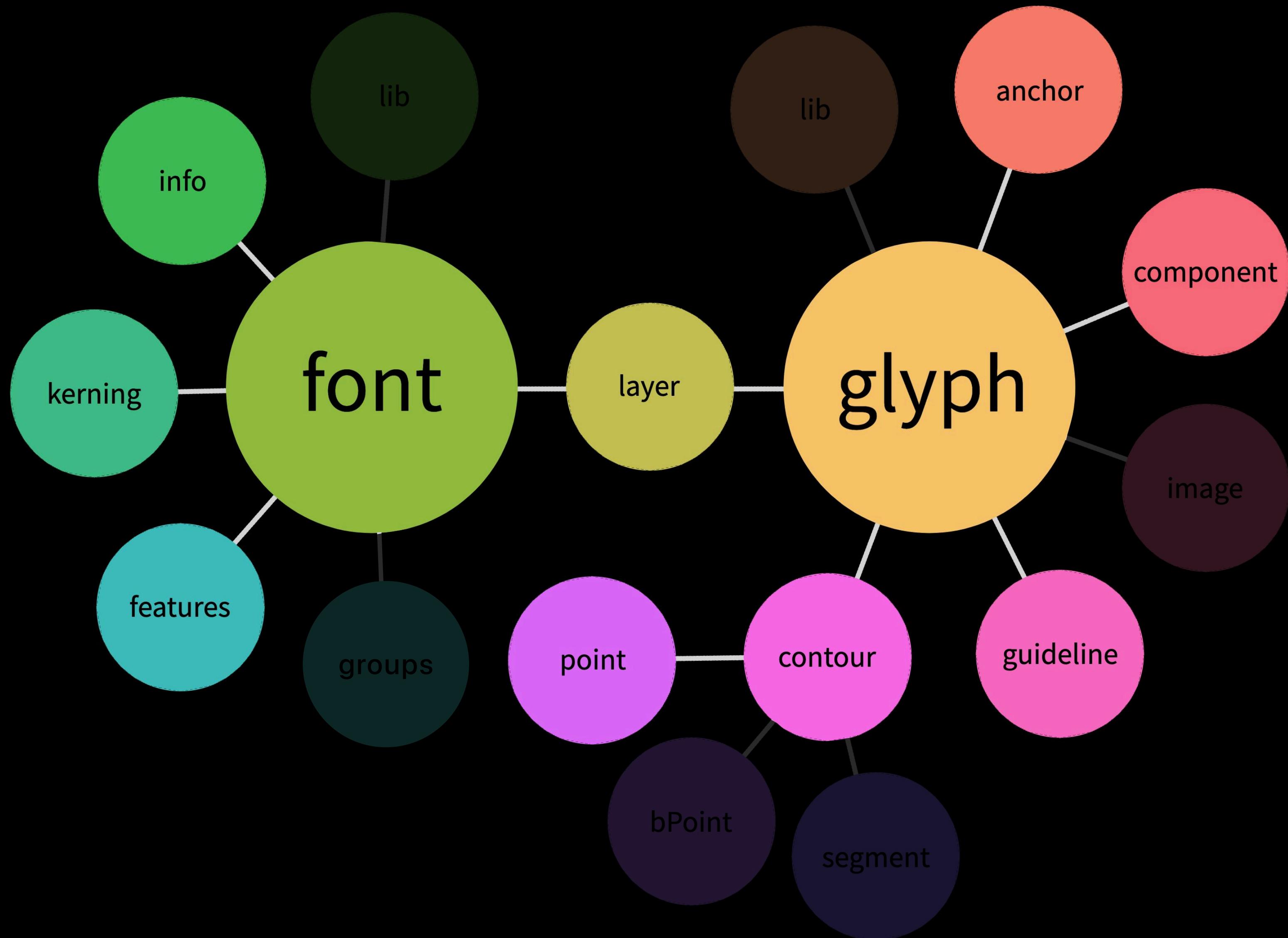
The hierarchy,
intuitively



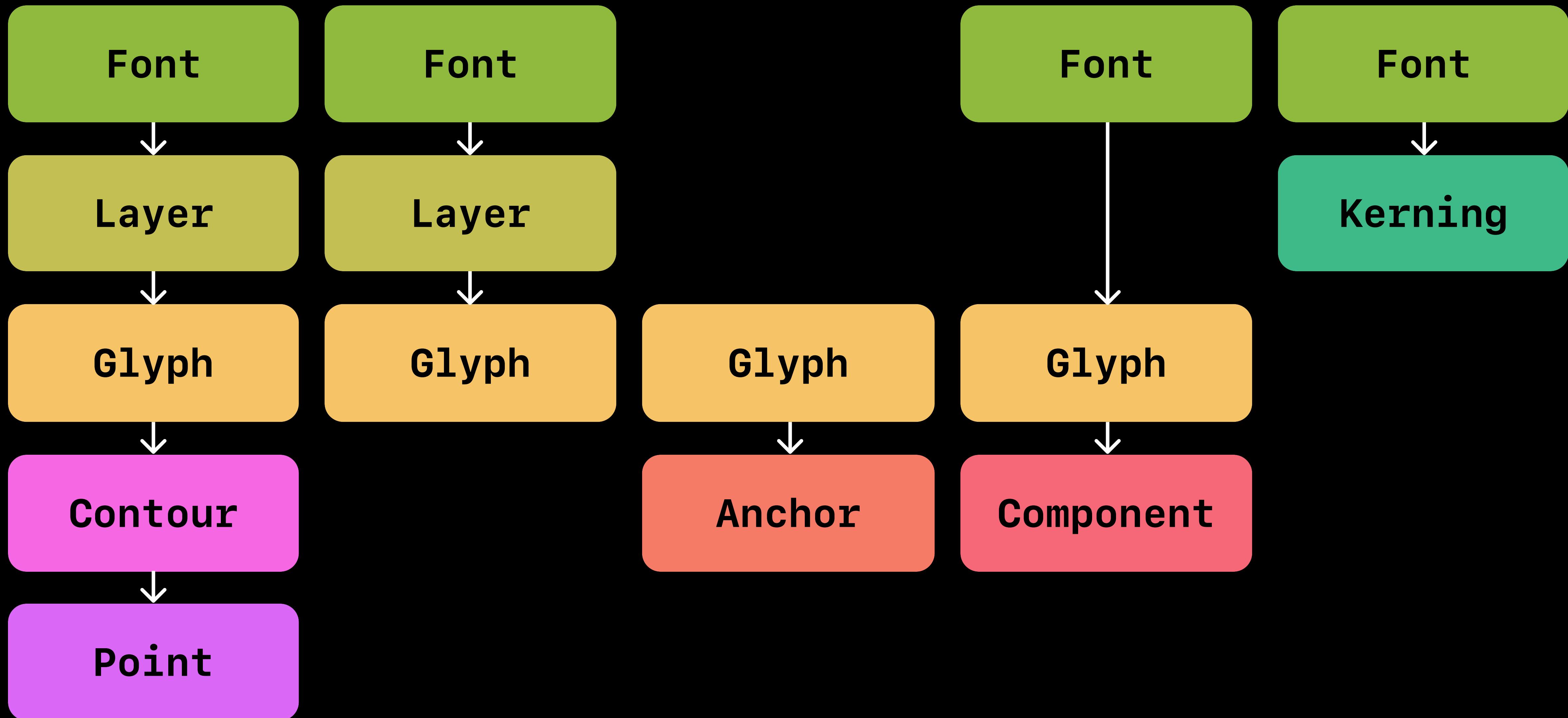
The map



Our main destinations



Some examples of looping with FontParts



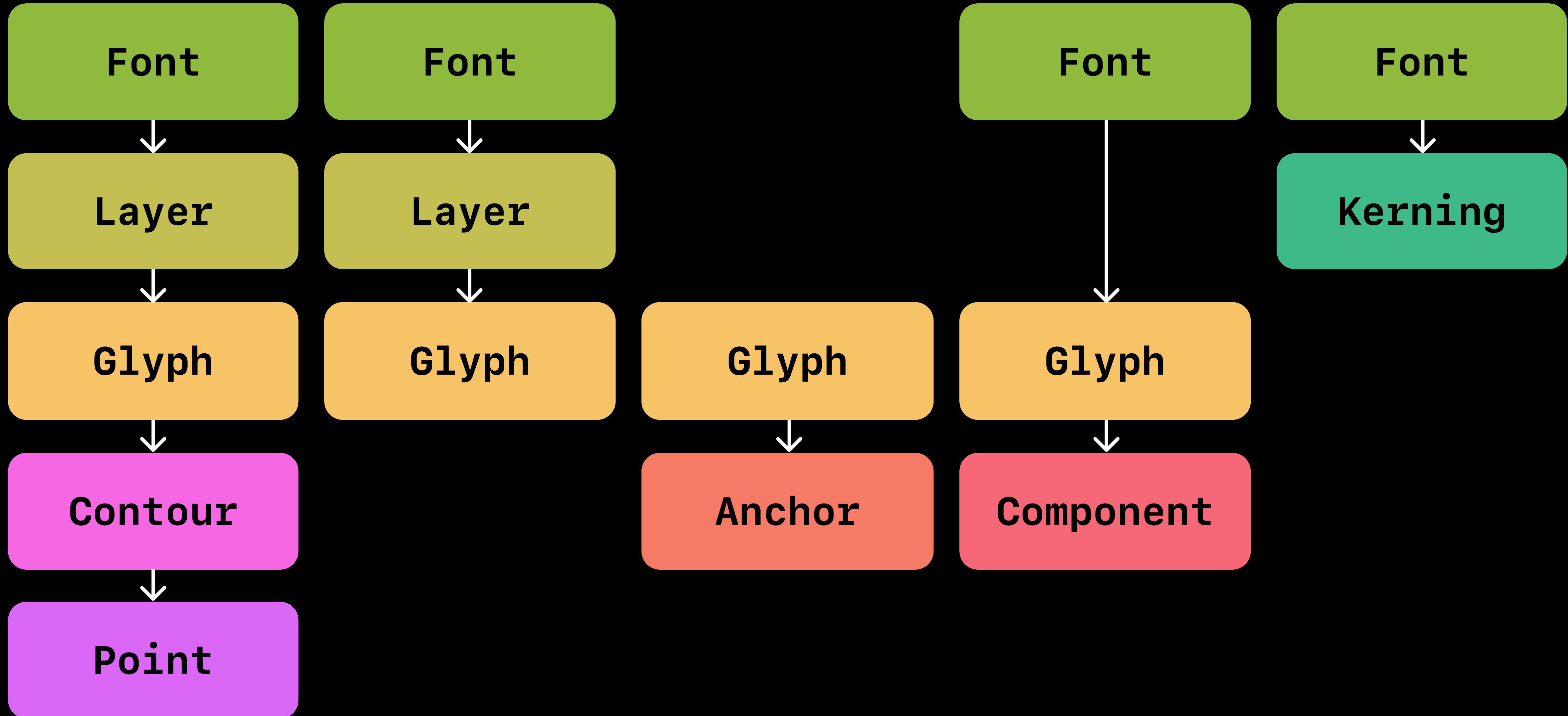
Do something with
every **point** in
this **font**.

Do something with
every **glyph** in
this **font**.

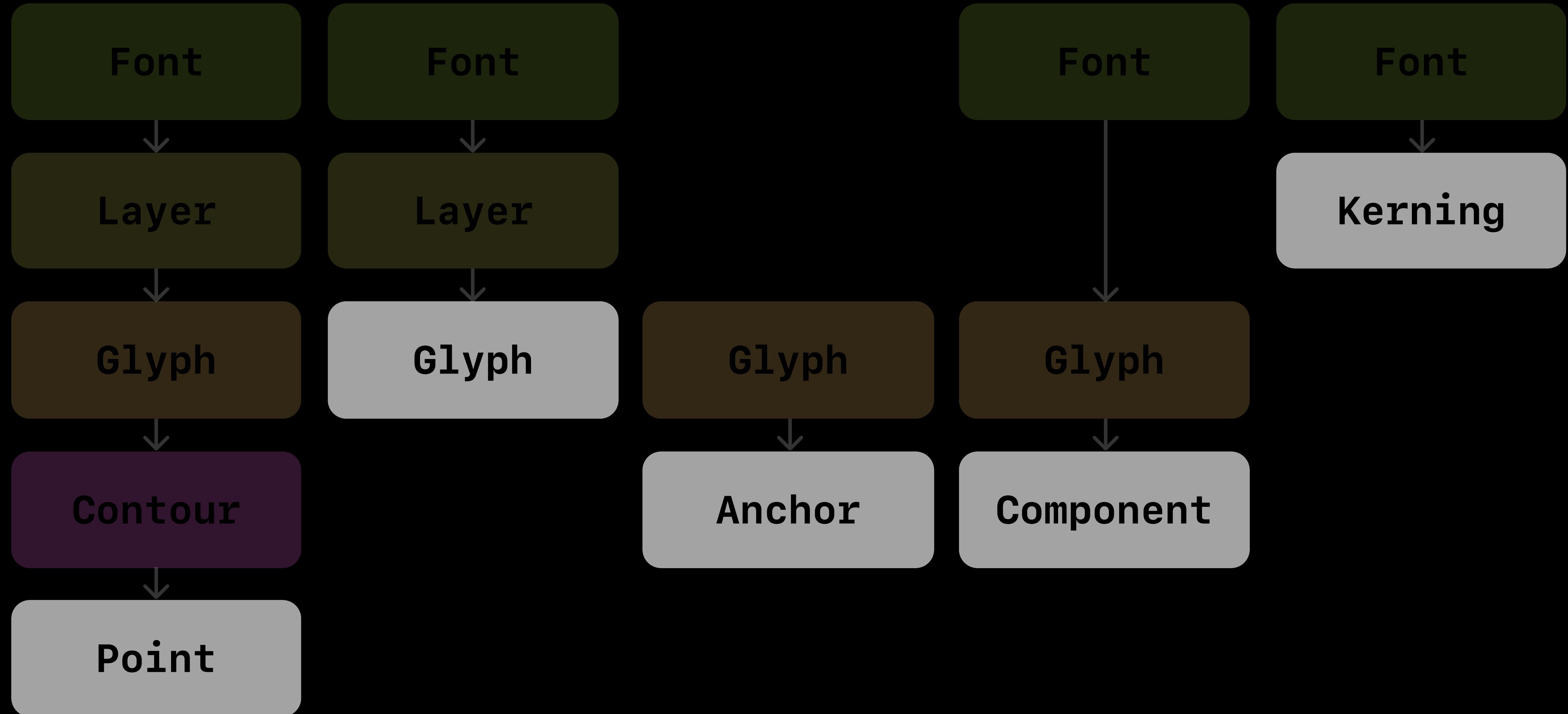
Do something with
every **anchor** in
this **glyph**.

Do something with
every **component** in
this **font's**
default layer.

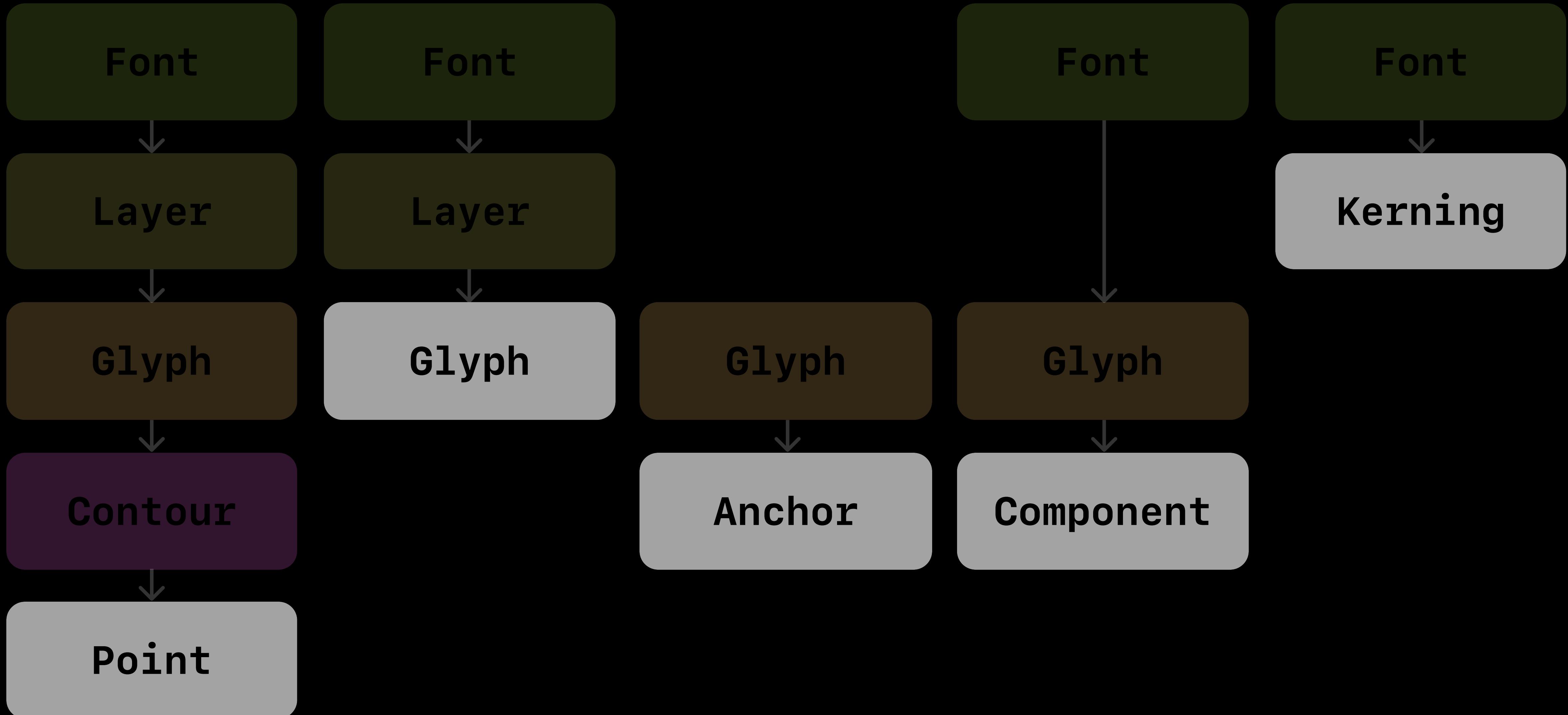
Do something with
the **kerning** in this
font.



What do we do here?



Do stuff or get stuff.



Methods / Functions

Attributes

Method / Function

Perform an action.

Attribute

Get or
set information.

Method / Function

`glyph.appendContour()`

Attribute

`glyph.contours`

Method / Function

`glyph.appendContour()`

*Methods and functions
have these parentheses.*

Attribute

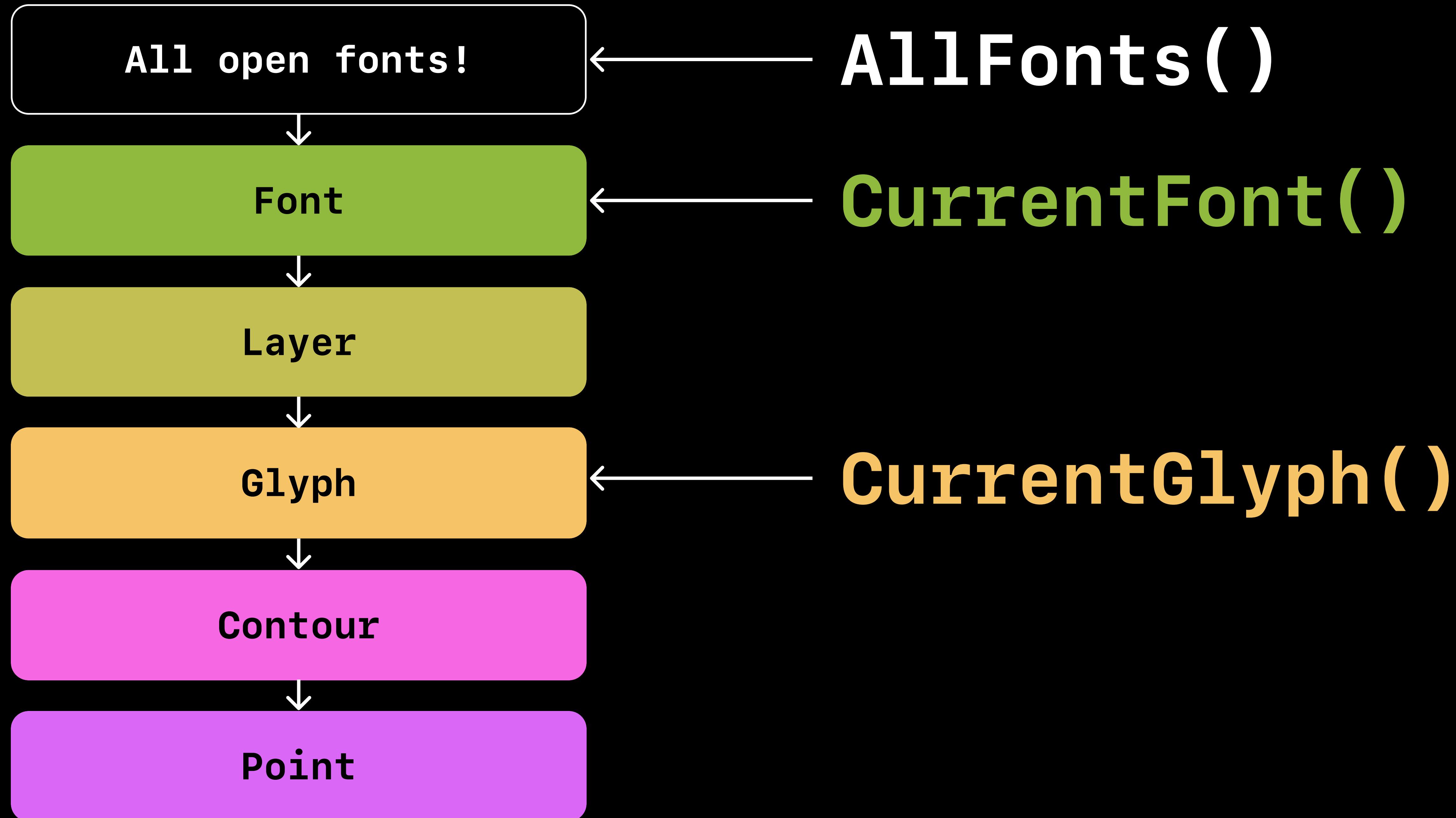
`glyph.contours`

Where do we start?

AllFonts()

CurrentFont()

CurrentGlyph()



All open fonts!

AllFonts()

Font

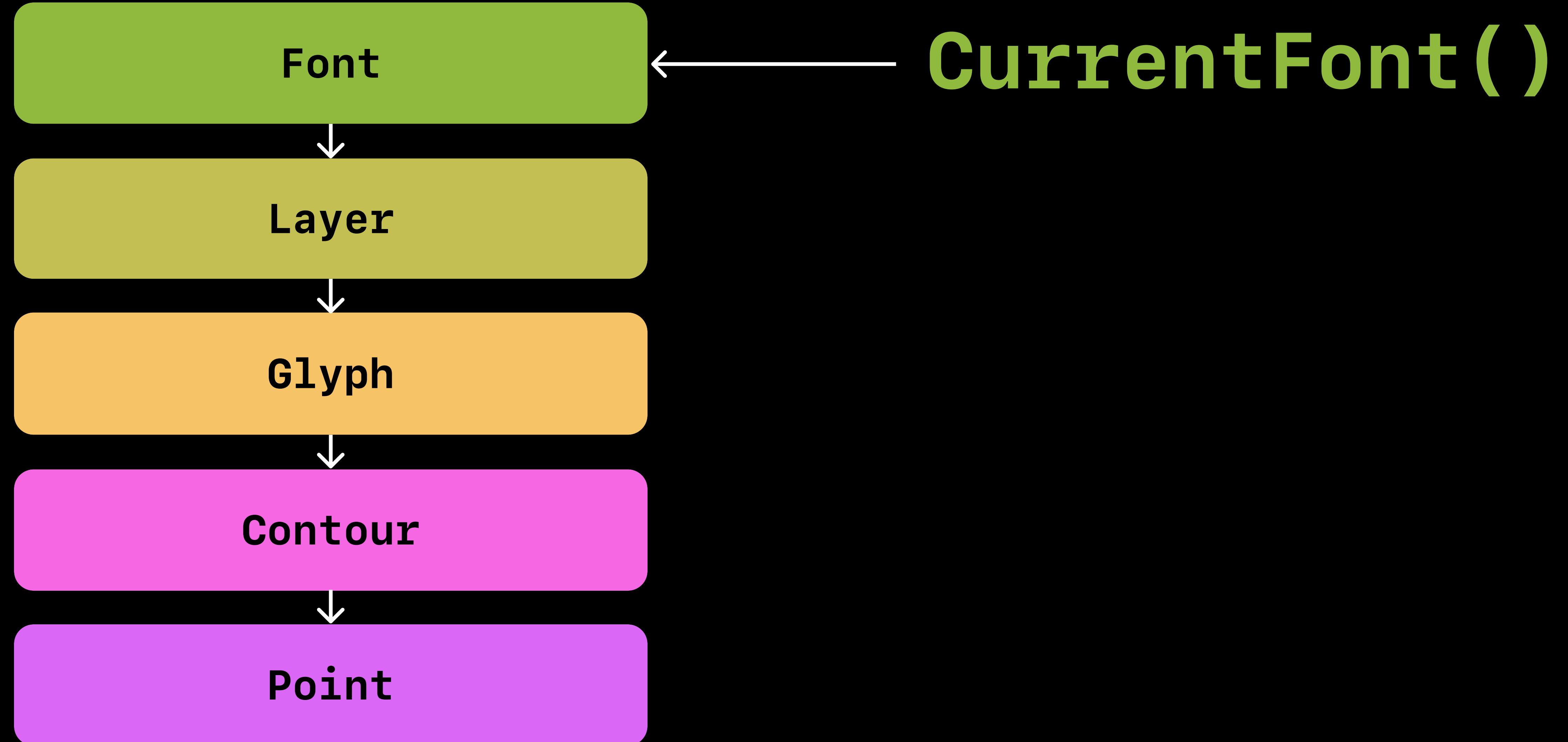
Layer

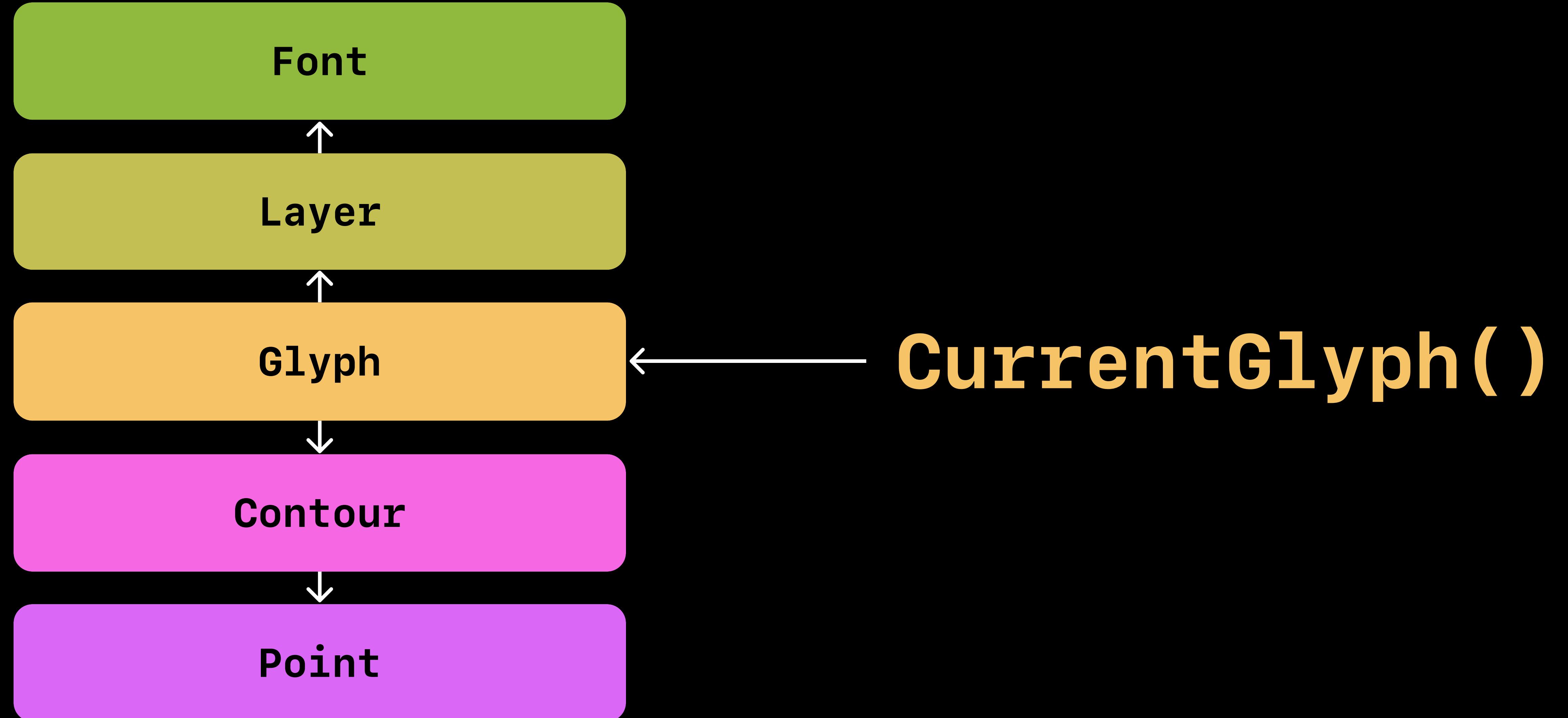
Glyph

Contour

Point







```
1 print('Let's go.')
```

2

3

4

5

6

7