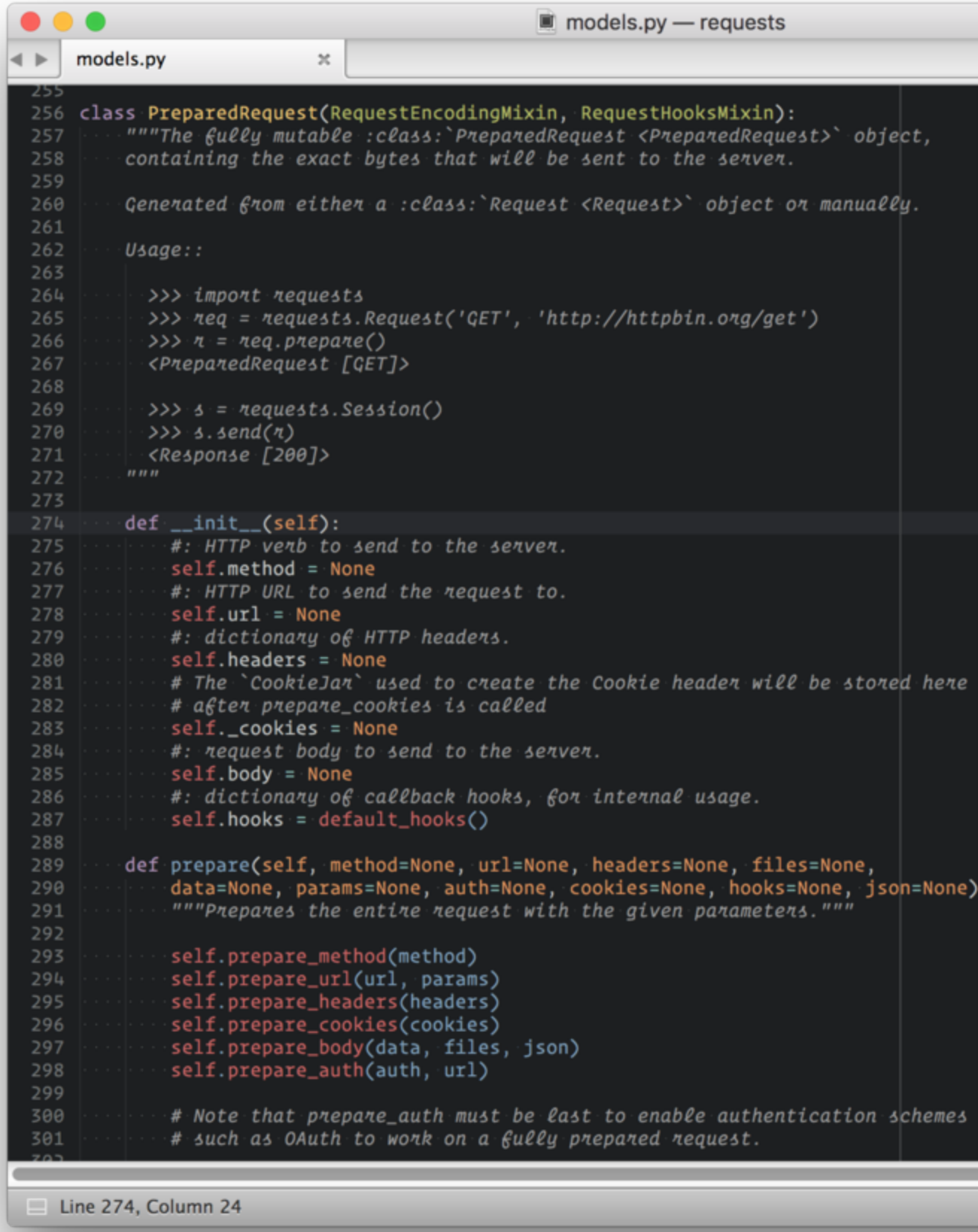
Test-Driving a $200 Coding Font: Operator Mono

2016

2 min read • 423 words

Themes: Consciousness Programming Spiritual


models.py — requests
models.py
class PreparedRequest(RequestEncodingMixin, RequestHooksMixin):
"""The fully mutable :class:`PreparedRequest <PreparedRequest>` object,
containing the exact bytes that will be sent to the server.
Generated from either a :class:`Request <Request>` object or manually.
Usage::
>>> import requests
>>> req = requests.Request('GET', 'http://httpbin.org/get')
>>> r = req.prepare()
<PreparedRequest [GET]>
>>> s = requests.Session()
>>> s.send(r)
<Response [200]>
"""
def __init__(self):
#: HTTP verb to send to the server.
self.method = None
#: HTTP URL to send the request to.
self.url = None
#: dictionary of HTTP headers.
self.headers = None
# The `CookieJar` used to create the Cookie header will be stored here
# after prepare_cookies is called
self._cookies = None
#: request body to send to the server.
self.body = None
#: dictionary of callback hooks, for internal usage.
self.hooks = default_hooks()
def prepare(self, method=None, url=None, headers=None, files=None,
data=None, params=None, auth=None, cookies=None, hooks=None, json=None):
"""Prepares the entire request with the given parameters."""
self.prepare_method(method)
self.prepare_url(url, params)
self.prepare_headers(headers)
self.prepare_cookies(cookies)
self.prepare_body(data, files, json)
self.prepare_auth(auth, url)
# Note that prepare_auth must be last to enable authentication schemes
# such as OAuth to work on a fully prepared request.
Line 274, Column 24

As programmers, typefaces surround us — everything we do, build, manage, and orchestrate is typically encoded with a pleasant and comforting monospace typeface emanating from our console or editor of choice.

Which Typeface to Use?

I've invested a lot of time in optimizing my development environment and workspace — and typefaces are a large part of that.

Over the years, I've cascaded through a number of monospaced typefaces for coding, including— Monaco, MS Consalas, Inconsolas, Inconsolata-dz, and Ubuntu Mono.

Ubuntu Mono was my favorite monospaced typeface, used longer than any other, until I started using Operator Mono about six months ago.

The Hoefler & Co. type foundry released Operator Mono around this time—a very thoughtfully handcrafted typeface inspired by the pica-ruled days of typewriters. It is beautiful, and I have found it to be a pleasure to use.

Operator Mono for Programming

I find that different typefaces (as well as color schemes) both set the mood for my development environment, as well as offer utility in the way that I parse programmable text. While using Operator Mono, I found that I scan entire words as I read code more easily; while in other typefaces, my parsing style is often more letter-by-letter.

This observation could be written off as a subjective one, but I am all about the subjective, above all else — especially when it comes to my environments.

Feature: Italics in Script

The most unique feature of this font is that it includes an italicized typeface that uses monospaced script characters.

While perhaps appearing novel at first, I have found that using this (beautiful) script typeface for code comments has improved both the quality of tone and frequency of my code annotations — something which I highly value.

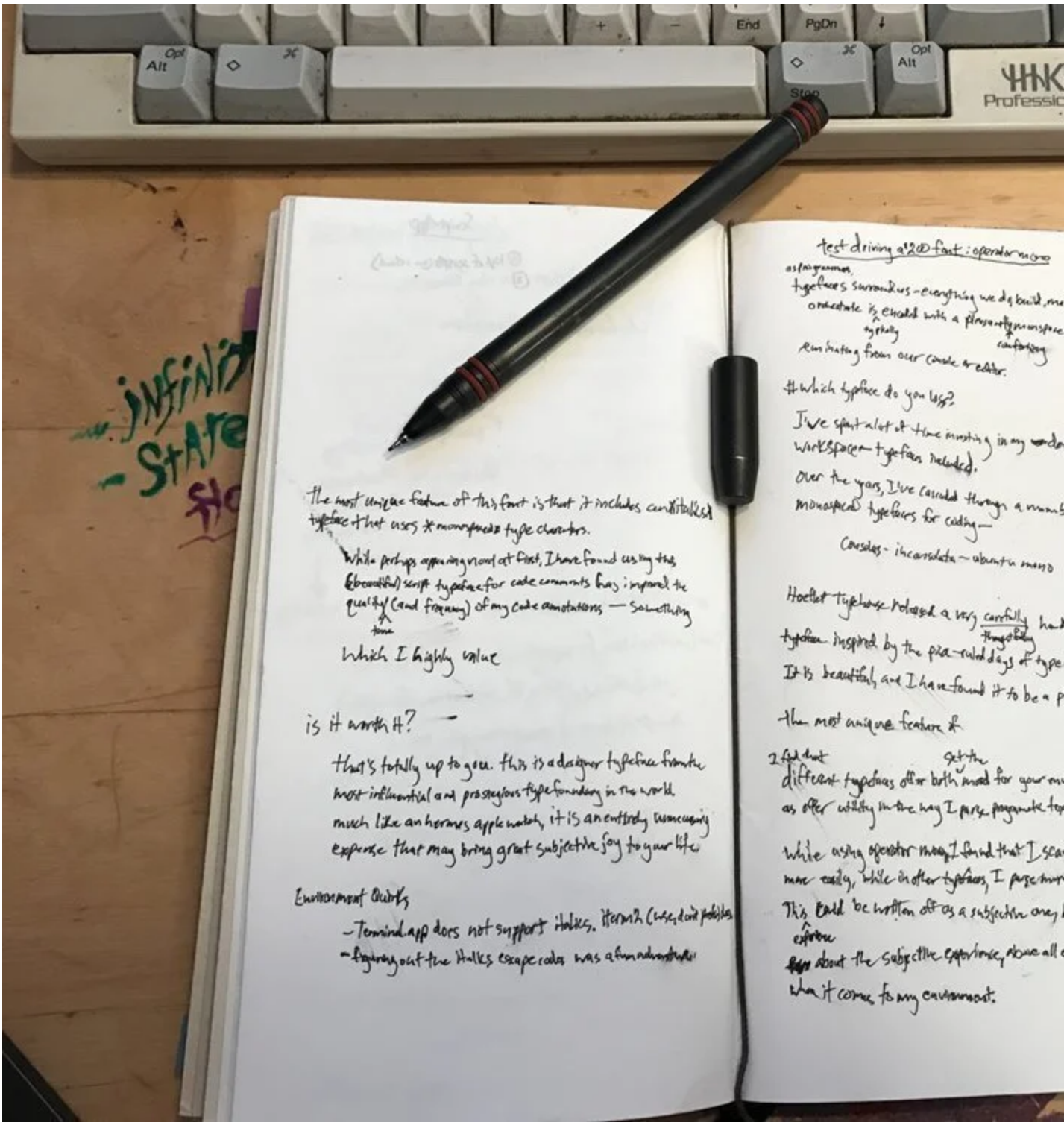
Is it worth it?

That's totally up to you.

This is a designer typeface from the most influential and prestigious type foundry in the world. Much like an Hermès Edition Apple Watch, it is an entirely unnecessary expense that may bring great subjective joy to your life.

To me, that's all that really matters—part of understanding programming as spiritual practice where the tools and environment we choose shape our consciousness and creative expression.

***P.S.** This blog post was written entirely with pen and ink—surprisingly fun.*



Relevant Links

- Introducing Operator by Hoefler & Co.

- Operator ScreenSmart Mono on Typography.com (1 computer license)
- pep8.org utilizes this typeface throughout (code examples)

Bonus Integration Quirks

- Terminal.app (my favorite console) does not support italics. iTerm2, however, does.
- Figuring out the italics escape codes for zsh was a fun adventure.
