November 12th, 2019

Choices I’ve considered for Savitar’s Text Engine:

NSTextView

WKWebView

iTerm2Lib

I started down the path of using NSTextView until I got to the point where I wanted to start rendering ANSI codes and also thought about how things like <code> would need to be implemented. I wanted both of course. So, If I used NSTextView, I’d have to do an ANSI-to-attributedString parser as well as an HTML-to-attributedString parser. I found an ANSI-to-HTML parser (aha.c). This got me thinking of using WKWebView, because itcould handle the HTML natively and aha.c would provide the ANSI code support. Also, it gave room to doing other media types easily, including links of course.

I then thought WKWebView would be just too darn slow. So, I set my sights on iTerm2Lib, for it was performant, could handle ANSI codes, and well, I could live without HTML… I guess.

The problem with iTerm2Lib is I’m carving it up into smaller pieces, and the network aspect isn’t going to be a joy to deal with.

What if WKWebView was in fact performant? Wouldn’t that be the ideal means to render text for the next generation of Savitar? WKWebView claims to be: “*Boasting responsive 60fps scrolling, built-in gestures, streamlined communication between app and webpage, and the same JavaScript engine as Safari”*.

Hey, JavaScript would be a nice addition too. I think I’m back to exploring WKWebView.

November 13th, 2019

Next thing I want to do is do some local testing, get some ANSI-to-HTML conversion working.