

MICHIGAN STATE

UNIVERSITY

Beta Presentation

Improve Firefox's Reader View

The Capstone Experience

Team Mozilla

Chad Burnham

Steve Hagopian

Jintao Hu

Tyler Kabaker

Noel Lefevre

Emily Michaels

Department of Computer Science and Engineering

Michigan State University

Spring 2022



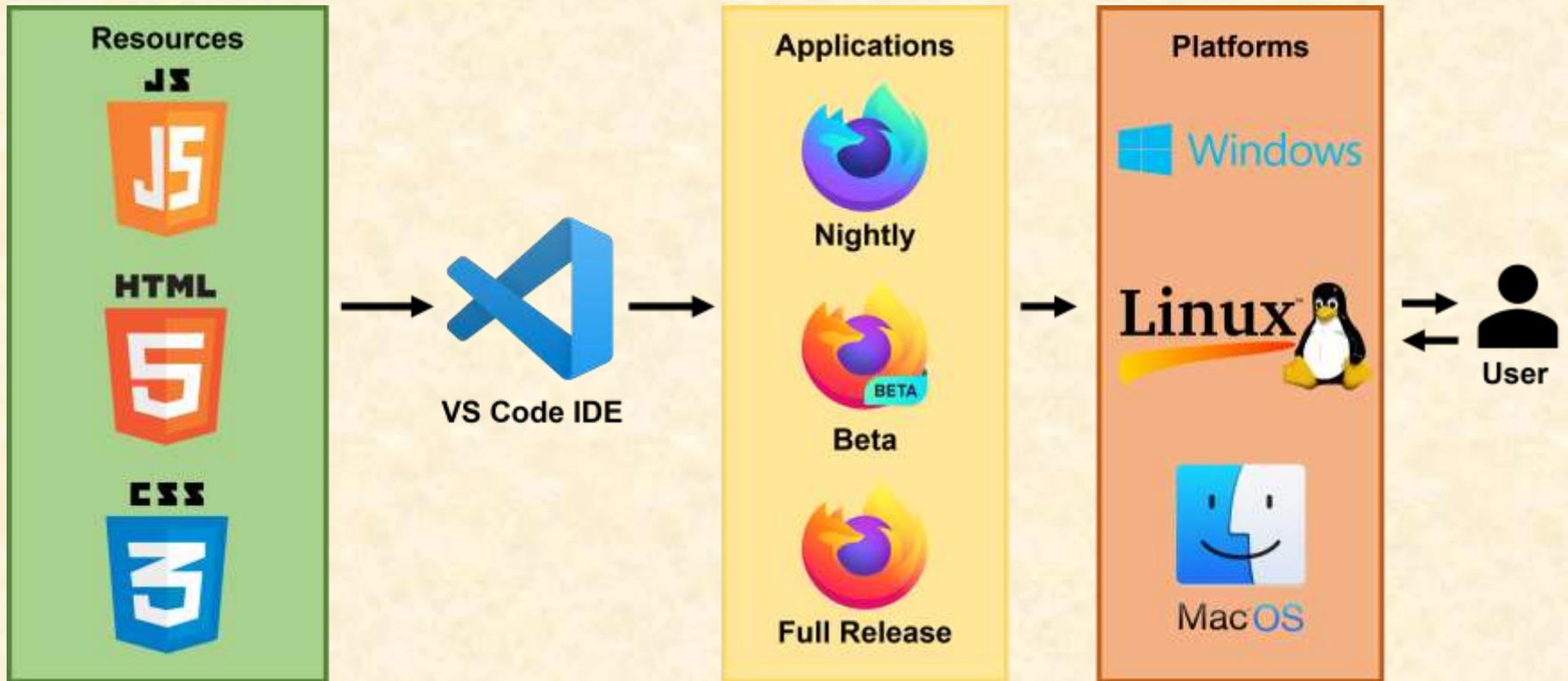
*From Students...
...to Professionals*

Project Overview

- Fixing long-standing issues on the about:reader page
- Features that are being improved upon or added:
 - Quality of life improvements
 - Cross-browser compatibility
 - Page formatting and styling
- Writing tests to verify integrity of existing code
- Investigating and fixing issues on top sites



System Architecture



Save to Pocket & Image Bugs

https://www.richz.com/fishing/blog/?m=202203

richz.com

Month: March 2022 | RichZ's Bass Blog

14-17 minutes



Click the Pocket Button to save any article, video or page from Firefox.

View in Pocket on any device, any time.

Learn more

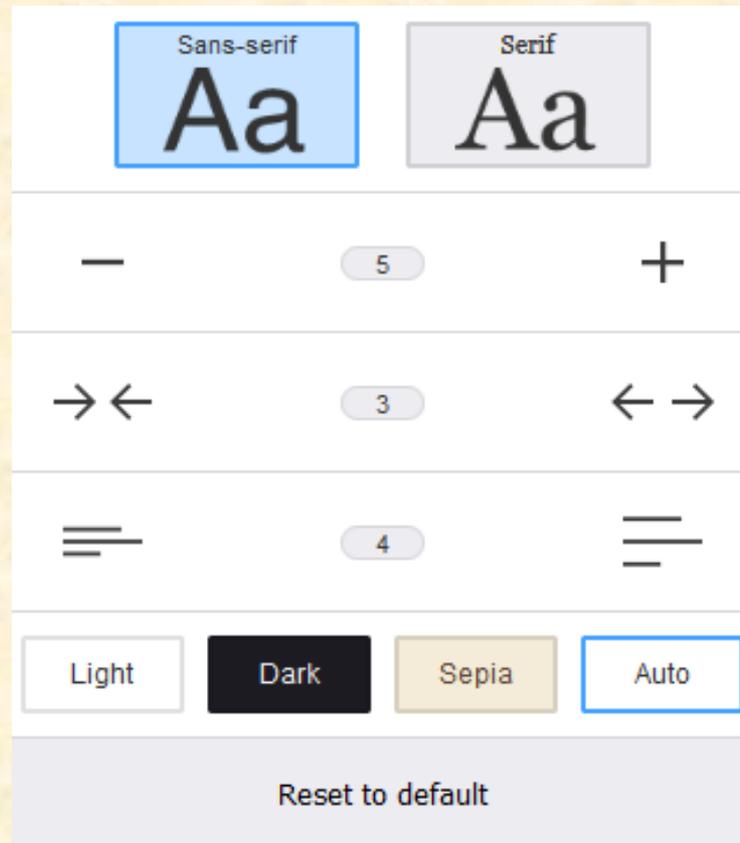
Sign up for Pocket. It's free.

 Sign up with Firefox

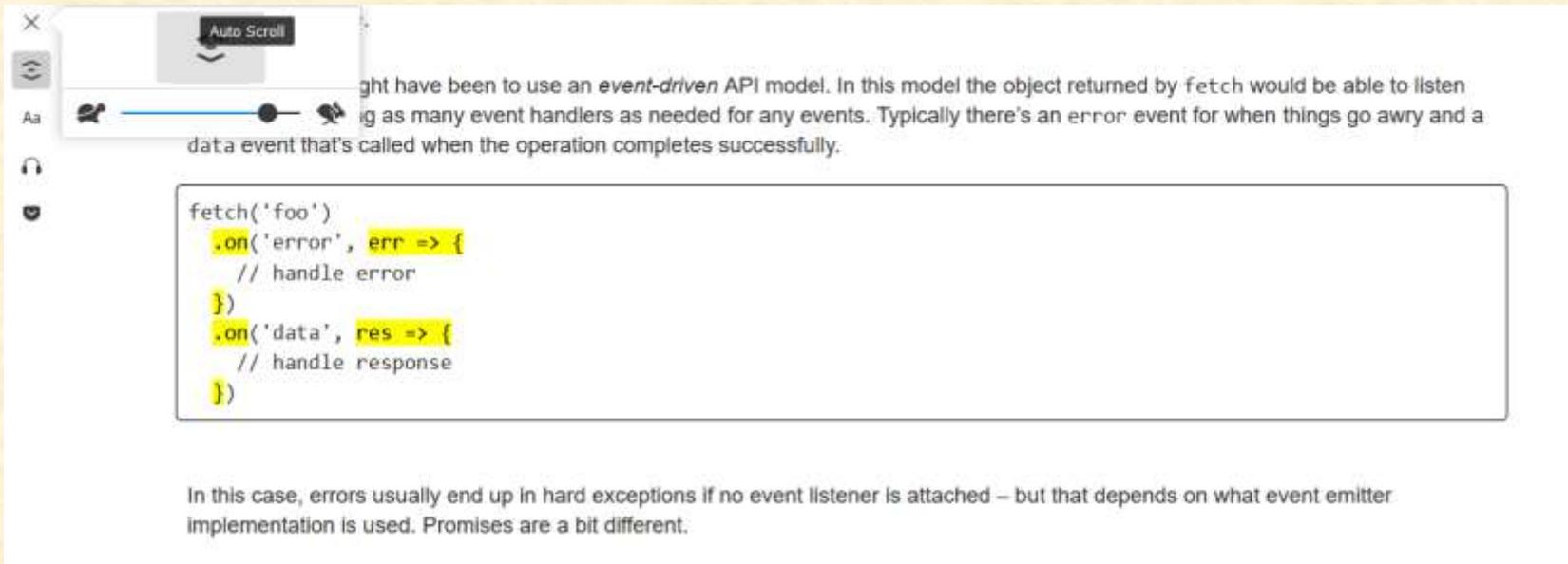
Already a Pocket user? [Log in.](#)



Reset to Default & Auto Feature



Auto Scroll & Code Blocks



might have been to use an *event-driven* API model. In this model the object returned by `fetch` would be able to listen as many event handlers as needed for any events. Typically there's an error event for when things go awry and a data event that's called when the operation completes successfully.

```
fetch('foo')  
  .on('error', err => {  
    // handle error  
  })  
  .on('data', res => {  
    // handle response  
  })
```

In this case, errors usually end up in hard exceptions if no event listener is attached – but that depends on what event emitter implementation is used. Promises are a bit different.



Narrate Popup



ponyfoo.com

ES6 Promises in Depth

Nicolás Bevacqua

21-26 minutes

Promises are a very involved paradigm, so we'll take it slow.

Here's a table of contents with the topics we'll cover in this article. Feel free to skip topics you're comfortable about.

- [What is a Promise?](#) – we define `Promise` and look at a simple example in JavaScript
- [Callbacks and Events](#) – alternative ways to handle asynchronous code flows
- [Gist of a Promise](#) – a first glimpse at how promises work
- [Promises in Time](#) – a brief history of promises



What's left to do?

- Features
 - Tweak remaining unresolved bugs
- Stretch Goals
 - Investigate and fix issues with top sites, such as:
 - Wikipedia
 - Github
 - Daily Mail



Questions?

?

?

?

?

?

?

?

?

?

