Redesigning BrowseEverything

Samvera Connect 2019

Thomas Scherz
Randall Floyd
Christina Chortaria
James Griffin

BrowseEverything

- Samvera Core Component
- Rails Engine
- Permits users to select files (or directories) for repository ingestion
- Supports multiple file/directory sources
  - File System
  - Google Drive
  - Dropbox
  - Amazon S3 Buckets
  - Box

BrowseEverything

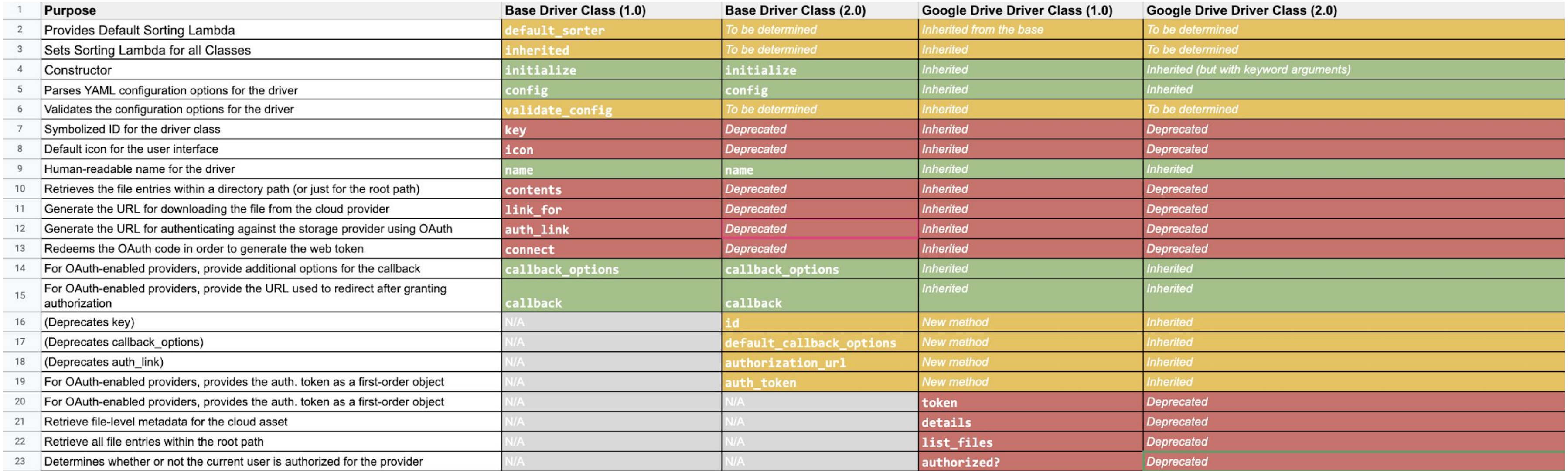
- Did not offer a 1.0.0 release until March 2019
- In use by many production Samvera repositories
- Problem: Maintaining drivers for cloud storage providers
  - Many cloud providers have different APIs
  - Feature requests submitted for organizations are provider-specific
  - How to define an API for a cloud provider?
- Before these challenges can be discussed, we will explore the **Driver API**

BrowseEverything: Driver API

- The Driver API wraps cloud provider clients
- Let us review the code base for drivers!
  - Base driver
  - Amazon S3 driver
  - Google Drive driver
- Drivers are inconsistent
  - Some cloud clients require OAuth2 authorization
  - Cloud clients search directories/folders
- **This needs to change**

BrowseEverything: Driver API

- We started prototyping API changes in supporting Google Drive
- Analysis



Purpose	Base Driver Class (1.0)	Base Driver Class (2.0)	Google Drive Driver Class (1.0)	Google Drive Driver Class (2.0)
Provides Default Sorting Lambda	`default_sorter`	*To be determined*	*Inherited from the base*	*To be determined*
Sets Sorting Lambda for all Classes	`inherited`	*To be determined*	*Inherited*	*To be determined*
Constructor	`initialize`	`initialize`	*Inherited*	*Inherited (but with keyword arguments)*
Parses YAML configuration options for the driver	`config`	`config`	*Inherited*	*Inherited*
Validates the configuration options for the driver	`validate_config`	*To be determined*	*Inherited*	*To be determined*
Symbolized ID for the driver class	`key`	*Deprecated*	*Inherited*	*Deprecated*
Default icon for the user interface	`icon`	*Deprecated*	*Inherited*	*Deprecated*
Human-readable name for the driver	`name`	`name`	*Inherited*	*Inherited*
Retrieves the file entries within a directory path (or just for the root path)	`contents`	*Deprecated*	*Inherited*	*Deprecated*
Generate the URL for downloading the file from the cloud provider	`link_for`	*Deprecated*	*Inherited*	*Deprecated*
Generate the URL for authenticating against the storage provider using OAuth	`auth_link`	*Deprecated*	*Inherited*	*Deprecated*
Redeems the OAuth code in order to generate the web token	`connect`	*Deprecated*	*Inherited*	*Deprecated*
For OAuth-enabled providers, provide additional options for the callback	`callback_options`	`callback_options`	*Inherited*	*Inherited*
For OAuth-enabled providers, provide the URL used to redirect after granting authorization	`callback`	`callback`	*Inherited*	*Inherited*
(Deprecates key)	N/A	`id`	*New method*	*Inherited*
(Deprecates callback_options)	N/A	`default_callback_options`	*New method*	*Inherited*
(Deprecates auth_link)	N/A	`authorization_url`	*New method*	*Inherited*
For OAuth-enabled providers, provides the auth. token as a first-order object	N/A	`auth_token`	*New method*	*Inherited*
For OAuth-enabled providers, provides the auth. token as a first-order object	N/A	N/A	`token`	*Deprecated*
Retrieve file-level metadata for the cloud asset	N/A	N/A	`details`	*Deprecated*
Retrieve all file entries within the root path	N/A	N/A	`list_files`	*Deprecated*
Determines whether or not the current user is authorized for the provider	N/A	N/A	`authorized?`	*Deprecated*

BrowseEverything: Rearchitecting APIs

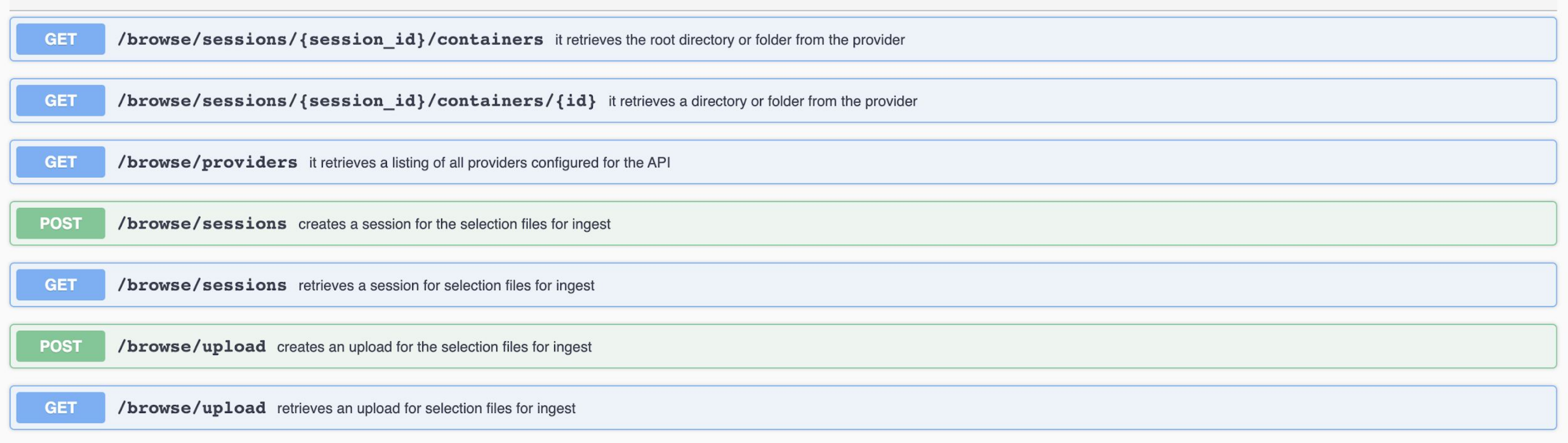
- Drivers need an API specification
- But there were also other requirements
  - User Interface improvements
  - Legacy jQuery Plugin dependencies
  - jQuery binding to server-rendered partial HTML
- Are there other approaches?
  - Can Rails Controllers just serve JSON data instead?

BrowseEverything: Rearchitecting APIs

- REST
  - Representational State Transfer (REST)
  - URL path structure combined with HTTP methods to define an API
    - `GET` “reads” an object
      - `GET https://institution.edu/repo/rest/v1/books/1`
    - `POST` “creates” an object
      - `POST https://institution.edu/repo/rest/v1/books`
    - `PATCH` “updates” an object
      - `PATCH https://institution.edu/repo/rest/v1/books/1`
    - `DELETE` “deletes” an object
      - `DELETE https://institution.edu/repo/rest/v1/books/1`

BrowseEverything: Rearchitecting APIs

- Swagger
  - Really, OpenAPI
  - Documents the RESTful API and HTTP methods
  - Gems and other libraries build upon Swagger utilities



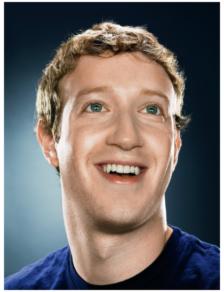
GET	`/browse/sessions/{session_id}/containers`	it retrieves the root directory or folder from the provider
GET	`/browse/sessions/{session_id}/containers/{id}`	it retrieves a directory or folder from the provider
GET	`/browse/providers`	it retrieves a listing of all providers configured for the API
POST	`/browse/sessions`	creates a session for the selection files for ingest
GET	`/browse/sessions`	retrieves a session for selection files for ingest
POST	`/browse/upload`	creates an upload for the selection files for ingest
GET	`/browse/upload`	retrieves an upload for selection files for ingest

BrowseEverything: Rearchitecting UI

- How does one consume data from our RESTful API?
- With Rails release 6.0.0, Webpack is supported for JavaScript assets
    - Without Sprockets, using ES6 or TypeScript with Node.js packages becomes easier
- BrowseEverything developers can also explore jQuery alternatives
    - Angular?
    - Vue?
    - React?
    - **Nothing at all?**

BrowseEverything: Rearchitecting UI

- Prototyping
  - After a spike, **React with Redux** was chosen
- React
  - JavaScript user interface framework
  - Logically groups a user interface into “components”
  - Oriented towards modularity
  - Supported by a major company
  - Competing only with Vue for dominance
- Redux
  - React is not concerned with fetching data from the server
  - **Redux provides these features**
  - Support for integrating React and Redux is mature



BrowseEverything: Web Components

- React is still an investment in a framework
  - Will we need to abandon React in the future?
- Solution: Web Components
  - https://github.com/w3c/webcomponents
  - **Web Components are not tied to frameworks**
  - We can build a UI from a collection of Web Components that can be reused in different frameworks (or no framework)
  - Still, developing them requires adoption of some Web Component library or toolset

BrowseEverything: Stencil

- Stencil is one such library that can help us create Web Components for use as custom elements anywhere
  - From Ionic Framework Team
- It is a compiler that takes things like the Virtual DOM, TypeScript and JSX and **generates standards compliant Web Components**
- We want to compose the main B-E UI out of smaller reusable granular Web Components
  - For example, a Stencil Web Component for the ByteStream row in B-E is being prototyped

BrowseEverything: Stencil

- Can Stencil Web Components, React, and Redux be integrated?
    - Yes, with caveats: each framework has their own quirks for direct integration of a Stencil Web Component
    - Ability to compile to popular frameworks as output targets is a work in progress

- But Material-UI, a React UI framework, makes composition using Web Components even easier
    - Material-UI allows you to replace the root node of most of their components via a property

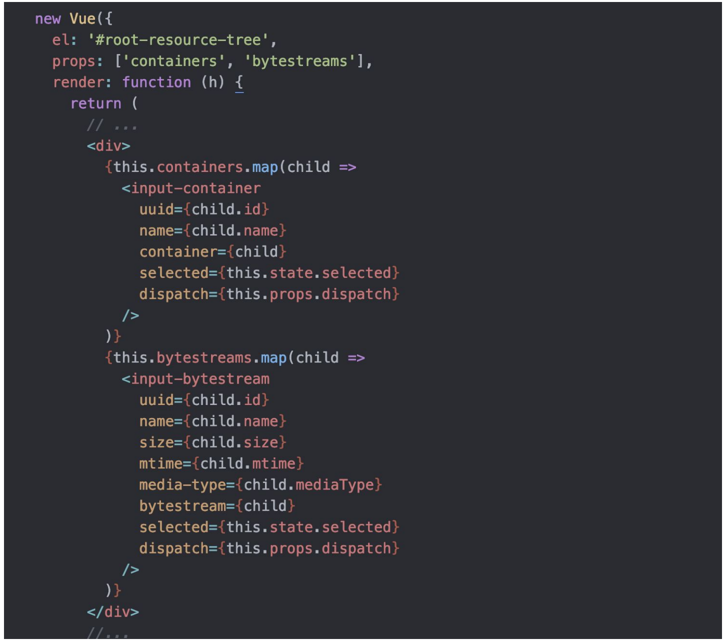
BrowseEverything: Stencil

- We can pass a Material-UI container a B-E Web Component like:

```
<div>
...
    <StyledChildResourceTree
  ...
      <ListItem   // A Material-UI component
        variant="contained"
        color="secondary"
        component="bytestream"   // From collection of BrowseEverything components
        onClick={this.handleClickAuthButton}
        ...
      />
    />
</div>
```

BrowseEverything: Stencil

- Can Stencil Web Components be used outside of React and Redux?



```
new Vue({
  el: '#root-resource-tree',
  props: ['containers', 'bytestreams'],
  render: function (h) {
    return (
      // ...
      <div>
        {this.containers.map(child =>
          <input-container
            uuid={child.id}
            name={child.name}
            container={child}
            selected={this.state.selected}
            dispatch={this.props.dispatch}
          />
        )}
        {this.bytestreams.map(child =>
          <input-bytestream
            uuid={child.id}
            name={child.name}
            size={child.size}
            mtime={child.mtime}
            media-type={child.mediaType}
            bytestream={child}
            selected={this.state.selected}
            dispatch={this.props.dispatch}
          />
        )}
      </div>
      //...
```

Vue.js

BrowseEverything: Stencil

- Stencil is still evolving and growing
- Use of Stencil in high-profile services, like Apple Music Web Beta, shows viable product and vibrant community
  - https://twitter.com/stenciljs/status/1169719595508408320?s=19
- Better React and Angular integration is coming
  - They are working on open sourcing their bindings to those so that you can compile to native React/Angular components
  - https://twitter.com/jthoms1/status/1176941324123201542?s=19

BrowseEverything: Prototyping

- Progress on this has been slow…
- Steadily prototyping of 2.0.0 and React/Redux frontend is underway
- ***We will now demonstrate the prototype***

BrowseEverything: Forthcoming

- Hooking into Hyrax will be supported
- To be determined: JavaScript dependencies
  - Webpack? Sprockets?
  - CDN?
  - Static asset?
- **Better user experience**
  - James is at fault for this (sorry)
- **A11y and i18n**
  - These were not priorities, but should be
- Community CfP
  - Maybe a community sprint?

BrowseEverything: Questions?

Q & A

Redesigning BrowseEverything

Thank you for your attention!

Please join us using:

#browse-everything

BrowseEverything Interest Group