Constellation of the Commons

Technical Documentation:

Overview of Code Base and Integration of Custom Code into Omeka Instance

Table of Contents

1. Project Overview

We've built out this project using Omeka Classic, which is an open source web publishing platform that is commonly used for digital scholarly projects and known for its rich metadata features. You can think of it as a content management system suited well for academic institutions. Generally speaking, Omeka is great for this project—storing records related to a set of social justice collectives and grouping them thematically—as it encourages record collecting and categorizing in an intuitive, not-too-techy interface called **Omeka Site Manager**. It should be noted, though, that a bit of customization is needed to express to a visitor (user) the notion of a thematic 'constellation' across these various collectives.

The main visualization [coded in **JavaScript** and implementing the **D3.js** library] displayed on the homepage is intended to be an interactive tool for visitors, to help them explore the different thematic connections and mission overlaps across the collectives. Through engagement, the visitor can start to grasp the desired "constellation effect" of the project dataset and scope.

We organize our data for themes and organizations on Omeka Site Manager, as well as structure our code base in such a way that the Site Manager data input-to-visualization workflow

involving (Omeka Site Manager input → Omeka API → D3.js → homepage data-driven visualization of themes and orgs) is a functional as well as optimized automation.

2. GitHub Repository: Where Does the Project Live

The source code for this project lives at: https://github.com/musicalforks/constellation-of-the-commons

A few notes about the GitHub repo:

(1) currently the source code can be integrated with Omeka's Big Picture and Center Row themes only. The GitHub copy of the source code represents the Big Picture integration.

(2) Virtually the same code exists for the two different theme integrations and minor differences are in the Omeka theme .php files to tailor the source code to a particular theme.

(3) The repo only includes custom files and modified files from the Omeka instance build of the Big Picture theme, i.e. the entire Omeka instance was not uploaded to GitHub to emphasize the components of the Omeka instance that were modified and augmented in this project. The full build can be downloaded at: https://omeka.org/classic/download/

3. Organization of Code Base

There are three high-level components of the code base: viz.html, the .js files, the css styling files, and the Omeka theme-specific .php files.

viz.html: the code to insert into the Omeka theme-specific .php file for rendering the homepage

.js files: set of JavaScript files that retrieve data from the Omeka API, process it, and display it in a graph visualization using D3.js library

.css files: set of .css files that determine the styling for the visualization and also the homepage at large

Omeka theme .php files: set of .php files included in the original Omeka instance build that are modified to include the viz code and display custom page elements

We'll look at each component individually in order to get a better idea of how they work together.

3.1 viz.html

This code is injected into the index.php file of the Omeka theme build with the following line:

```
<?php require('viz.html'); ?>
```

The file loads in relevant css stylesheets, d3.js, and all of the js files needed for visualization rendering in place – in between the header div and the footer div of the homepage HTML markup.

3.2 .js files

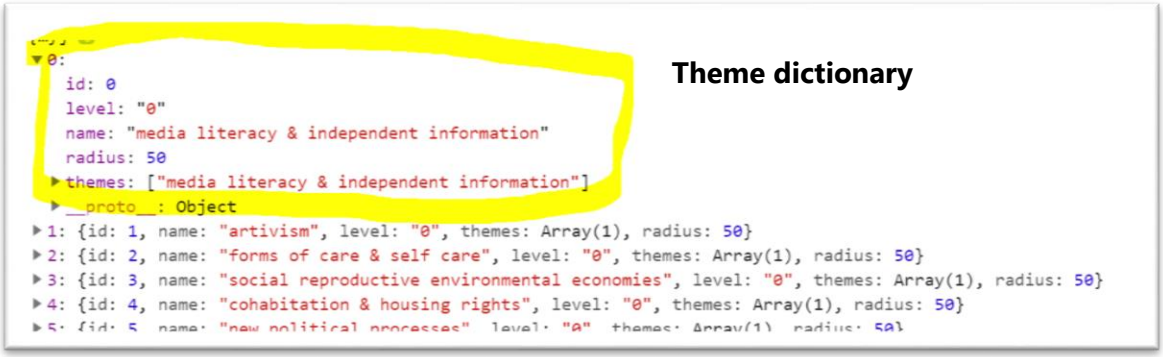
The .js files can be found in the /js folder and will be elaborated in call order, i.e. the order that they are called in the code base from the viz.html point of origin.

viz.js

- Initialize global vars to store nodes, links, theme data, etc.
- Make requests to Omeka API for theme data, collective data, color key data, and center node data
- Call processing and visualization .js files

dataFormatting.js

- File with one function `dataFormatting(themesJSON, orgsJSON, otherInfoJSON)` – parameters are (1) array of theme data, (2) array of collective data, (3) array of data pertaining to color key and center node
- Format API JSON returns to correct data format for D3 physics-driven visualization
    - Transfer fields correctly in API return to dictionary object represented node, e.g. 'title', 'url', 'logo orientation'
    - Create dictionaries representing themes, collectives, and center node and link each dictionary to relevant themes (primary themes) by ID



Collective dictionary

Center node dictionary

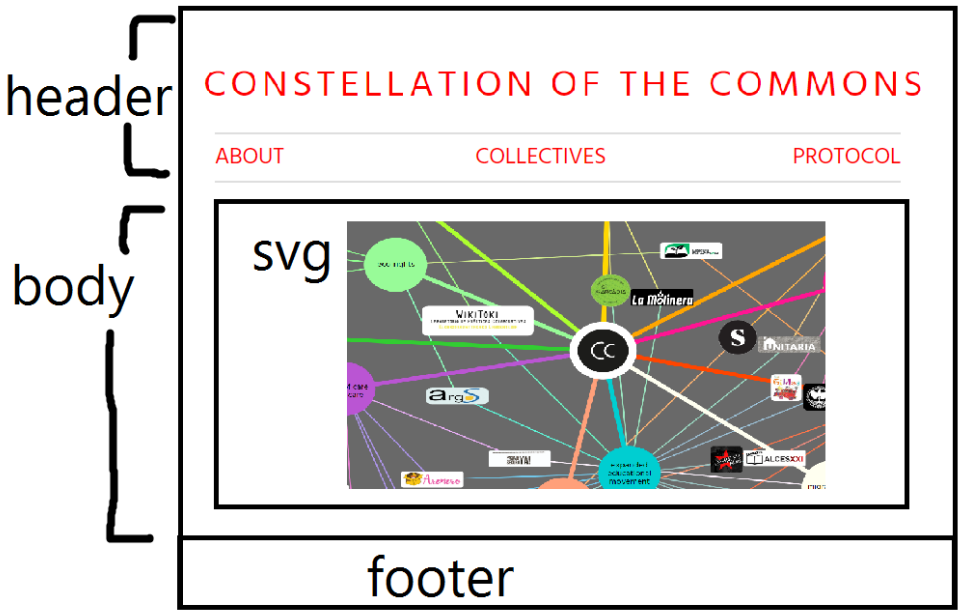
- Alternate version of this file is `dataFormattingUsingActualThemes.js` which can be found in same directory and links each collective to its full list of themes via *Tags* field instead of its list of primary themes (usually 2-3 themes) via *Primary Themes* field

defineColors() function in stylingd3.js

- Function that creates the theme color key from the color key stored in the *Color Key* field of the center node JSON (item named 'constellation of the commons' in Omeka Site Manager)
    - Converts standard HTML colornames to RGB equivalents

displayViz() function in display.js

- Function that displays visualization inside svg pane of homepage body using D3.js library



- Add svg to <body> HTML tag of homepage layout
- Set up physics engine using D3 built-in funcs (documentation: https://github.com/d3/d3-force)
- Display links between nodes, & color and weight them as defined by styling functions in stylingd3.js
- Class nodes as a theme node, collective node, or center node
- Display theme nodes as a circle colored by the color average/blended color of corresponding themes
- Display collective nodes as a logo image with hyperlink to the collective's Exhibit page
- Display center node as site logo image with concentric black circle around it, and with site title along semi circle arc
- Turn on physics engine force, user interactions for dragging, and mobile responsive design. Force center node to center of viz and add invisible nodes along links for spacing/repelling purposes

stylingd3.js elaboration

- This .js file includes various helper functions called by displayViz() in display.js (described above) for styling. Here's the condensed documentation for each helper function.
- **`isThemeOrOrg(node)` :** determines if node is theme or org
- Color keying suite
  - **`defineColors(colorKeyJSON)` :** iterates through the theme color key JSON and replaces color names with RGB equivalent; if a color name is invalid, then default the theme to white

    - **generateColor(colorName) :** parent func to convert a single HTML standard color name to its RGBA equivalent
    - **validColorName(string) :** validates input, which should be a standard HTML colorname, as CSS color name/string (https://http://jsfiddle.net/WK_of_Angmar/xgA5C/)
    - **colorToRGBA(color)** : Converts CSS color name to RGBA equivalent (https://stackoverflow.com/questions/1573053/javascript-function-to-convert-color-names-to-hex-codes/1573141)
    - **avgRgb(rgb1, rgb2, weight) :** average two RGB values by first squaring them, then averaging the squared values, (here the first value can be weighted), and then finally taking the square root of that average
- **circleColor(node) :** determines node color using theme color key
- **linkColor(node) :** determines link color for link btwn two nodes by averaging the colors of related themes
- **linkWeight(node) :** Determines stroke weight of a link btwn two nodes by idxing nodes' class fields
- Suite for positioning and sizing of collective nodes, i.e. logo images, depending on logo orientation (rectangular, square, or circle) and dynamically sized to window size reference
    - **imgParamX(node) :** determines x coordinate position of node
    - **imgParamY(node) :** determines y coordinate position of node
    - **imgParamWidth(node) :** determines width of node
    - **imgParamHeight(node) :** determines height of node
- **wrap(text, width) :** D3 text wrapping given maximum line width (https://bl.ocks.org/mbostock/7555321)
- **Array.prototype.unique** : Array prototype func definition to delete duplicate elements from array (https://stackoverflow.com/questions/1584370/how-to-merge-two-arrays-in-javascript-and-de-duplicate-items)
- **collide(alpha) :** Resolve collisions and overlapping between nodes through iterative relaxation, treating each node as a circle with a radius—"two nodes a and b are separated so that the distance between a and b is at least radius(a) + radius(b)." (http://bl.ocks.org/fabiovalse/bf9c070d0fa6bab79d6a)
- **d3.selection.prototype.moveToFront :** D3 selection prototype definition to move a D3 element in front of all others on a svg pane
- **d3.selection.prototype.moveToBack :** D3 selection prototype definition to move a D3 element behind all others on a svg pane

3.3 css files

viz.css

Styling guide for elements in svg pane (visualization pane) only – for svg elements like .body, .node text, .square logos, .rectangular logos, etc.

style.css

Added custom styling to Omeka theme-specific style.css file to position, size, and style navigation bar, search box, other page elements, etc. By modifying the Big Picture theme's style.css sheet, its default hamburger menu nav bar is currently replaced by a regular bar menu (mimicking the default nav bar menu style of Center Row theme).

3.4 Omeka theme .php files

""NOTE**: Applying the code base to themes besides Big Picture and Center Row may require additional modifications to the ones mentioned below.

The following default theme .php files, part of the Omeka instance build, incorporate the following changes to integrate the viz code + styling changes:

index.php

Includes an Access-Control-Allow-Origin header in the body HTML tag of homepage so the Omeka API endpoints can be accessed

```
<?php header('Access-Control-Allow-Origin: https://constellation.carletonds.com/');
header('Access-Control-Allow-Methods: GET, POST, PATCH, PUT, DELETE, OPTIONS');
header('Access-Control-Allow-Headers: Origin, Content-Type, X-Auth-Token'); ?>
```

Omeka Requires the viz.html code in the body HTML tag of homepage between header and footer to render the homepage visualization in place

```
<?php require('viz.html'); ?>
```

header.php

Only modified in the Big Picture-adapted version of the code base to convert the theme's default hamburger menu-style nav bar to a regular style nav bar (as defaulted in Center Row theme)

setUlClass php function organizes site page titles to display in a horizontal line across nav bar.

4. Applying Source Code to Other Omeka Themes

There is a Big Picture-adapted version of the code base uploaded to the Big Picture theme repository of the website hosting, and there is a Center Row-adapted version of the code base uploaded to the Center Row theme repository of the website hosting. To adapt the code base to other Omeka themes / integrate it into other theme repositories, the Access-Control-Allow-

Origin header and viz.html require must be added to the theme's index.php file. Additionally, there may be other changes that need to be made in index.php and to other theme .php files, e.g. header.php and footer.php to display the viz in the correct HTML div on the homepage and to implement custom styling that may not be the default styling of the theme.

Lastly, to set the source code as the Omeka Site homepage, the source code repository should be uploaded as the raw code for an Omeka Simple Page that is set as the Omeka site's homepage.

5. Known Issues

As of February 11, 2019, issues to note with the current state of the code base revolve around mobile responsiveness. Currently, the homepage visualization displays on mobile screens the same way it is displayed on desktop and tablet screens, except all elements are drastically resized to fit the whole visualization within the mobile screen view (without scrolling). Because all elements are scaled down by a factor determined by viewing screen width, the visualization ends up looking very crowded and dense and with illegible text and logos. Additionally, the colors do not display correctly on Safari mobile due to Safari's incompatibility with certain HTML standard color names; solutions have not been found. Moving forward, perhaps intentional discussions surrounding mobile design / what a user should ideally see on the mobile version of the site will contribute toward addressing these issues.

6. Concluding Remarks: Benefits of Omeka-D3 Integration

In early stages of this project, we explored alternate options for data visualization within the Omeka framework, primarily focusing on Omeka's Neatline plugin. As the project progressed, we came to find that static visualization tools like Neatline would not be the best fit for a project that involves live-data updating and long-term maintenance/prolonged project longevity. Currently we have decided the best course of action for the project is to utilize an Omeka-D3 integration environment for data visualization purposes. This is what Omeka-D3.js integration accomplishes through a physics/gravity-based graph model:

- Displays collective nodes as connected to and gravitating toward relevant theme nodes via lines
- Displays collective logos for collective nodes and theme name labels on circles for theme nodes, with each node colored by the color of its primary themes
- Allows for easy linking between Omeka collections and exhibits URLs and corresponding node displays (so if a user double-clicks on a theme node circle labelled Ecofeminism, user should be redirected to Omeka ecofeminism tags exploration screen).
- Allows for the process of drawing and sizing nodes in relation to one another to be automated/live-updated when new data is added to Omeka site, versus manual redrawing, resizing, and reuploading by a human on Omeka Neatline plugin every time a

new piece of data is added – to speak to the integration's emphasis on scalability and streamlined workflows

We hope that these design decisions will prove the code base efficient, powerful, and maintainable in the future of the project!