

Event Template Documentation

Project Overview

The purpose of this program is to maintain and present a carousel that transitions between events that appear for the museum of natural history using a new API. This program tracks the time, venue and description of events that are sourced from the API and displays them in a format given by the DCA in this template. This template can also be used to display data on exhibits. These are sourced from several sites that encompass several different locales and different types of events/exhibits given by the API.

Carousel

The carousel is designed using bootstrap, though it has been modified to be dynamic in size to fit the number of given events/exhibits using Javascript. The API feed is parsed via Javascript and all relevant information is then extracted via JSON format. This data includes information such as the starting time of events and the ending time of events, the event/exhibit location and the summary description of the event/exhibit. After the information has been extracted it is then inserted into containers (divs) in the carousel and displayed with the given timer interval that is set in Javascript.

Script

The Javascript portion of this project is the most influential. The majority of actions and design decisions occur in the Javascript portion of this program. The primary datapoints are represented by the feedURL, venue_name and event_or_exhibit variables that are determined at runtime but can be easily changed at the top of the script. The feedURL is the given link for the API that is to be parsed. The venue_name given is determinant of which location/venue of events/exhibits to extract from the JSON feed given by the feedURL. The event_or_exhibit variable defines whether to extract events or exhibits from the given feed. These can all be easily changed to suite your needs. Considering adding an interval variable for the time delay between carousel slides at the top of the script. After these are defined the program initiates an async function that awaits a promise from the server for the given API link. If the promised response is null, then a “no events” slide is shown. Otherwise, the response is then parsed via JSON into a variable named data to locally store the information obtained from the API.

Function

The `getData` function simply parses the API feed given at the top of the script and then extracts relevant information from the given feed to fit the template. After the feed has been stored in the data variable and the data is a non-null value, a function named `createCarousel` is called. This function simply establishes the carousel and the contents of the carousel. To start, the events/exhibits are looped through to find all the given events or exhibits determined by the `event_or_exhibit` variable. If the time of the event/exhibit is beyond the current date and time and if the event/exhibit venue matches the name of the given `venue_name` variable, given at the top of the script, as well as matching the given event/exhibit category, then a slide is created. This slide is an HTML div which is later inserted into the carousel. The div is then IDed by the ordered number of the given event in the loop. The number of the event/exhibit is checked to see whether it is the first slide. If it is the first slide, then its' class is set to active. This determines the first slide shown. Then the timing interval for that slide is set for the current slide in milliseconds. All relevant information for the current event/exhibit is then extracted to fill the current slide. This information includes an image URL for the given event image, the start date and end date, venue location and description. This data is then formatted and placed into divs and inserted into the slide's inner HTML. After this the slide is simply appended/added to the page's carousel div and displayed appropriately.

CSS

The CSS or cascading stylesheet aides in helping to design and format the given HTML. The margin and padding of the page is initially set to 0 pixels using the body and html tags in CSS. The width and height are set to 100% but may need to be changed to fit different screens in different orientations. A vertical or portrait orientation may require defining the pixel width and height of the display in use. The background color is also set to black. Each variable class simply uses the viewport width to maintain the size of the fonts for the page and determines the font family, as well as color of the font. Most of the formatting is done via Javascript so there isn't much done in the stylesheet.

HTML

The HTML is very minimal, as most of the design and operation of this page is done via Javascript. The head tag contains the title and links to the stylesheet, as well as external resources for using bootstrap. The body simply contains a "main" container div. This container contains a div for the event title "Events", which can be changed. The main container also contains the primary div container for the carousel. This carousel is IDed as "carouselSlides". This container is used to display the carousel and is used in the Javascript to insert other divs, information and formatting. After the carousel div there is a script tag for the local Javascript and for the external bootstrap resource.