Breakout Task Overview

Spotify Search with jQuery AJAX

Goal

We want to asynchronously query the Spotify search API to quickly display artists, albums, tracks and playlists. Here's a video showing a possible implementation:
https://youtu.be/rJaZQ7tjLp0

Before you get started, familiarize yourself with the search method in the Spotify API:
https://developer.spotify.com/web-api/search-item/

The Skeleton

You need two files that we put on GitHub:
https://github.com/MIMUC-MMN/tutorials-15-16/blob/master/tutorial06/breakout/breakout.html

https://github.com/MIMUC-MMN/tutorials-15-16/blob/master/tutorial06/breakout/style.css

Your TODOs

AJAX Handling

1. Create a function "`search`". It takes three parameters: `search-term`, `type`, `callback`.
   `search-term`: word or multiple words that the API will be searched for.
   `type`: one of 'artist', 'album', 'playlist' or 'track'
   `callback`: A function that will be called with the result of the AJAX request.
2. Launch an AJAX request in the search function using jQuery. Mind the correct parameters that the Spotify API requires. This parameter needs to look something like this:
```
{
    q: searchTerm,
    type: type
}
```
3. Create a function "`resultHandler`". It takes a parameter "items", which is an array containing the result of the AJAX request. The `resultHandler` function is called in the "search" function. It is responsible for processing the results.
   - There is a `<div id="result">` in the skeleton, which you can use to show the results nicely.
   - Empty the result div everytime the `resultHandler` is called.
   - Loop through each item and construct a DOM node:
```
<div class="item flexChild">
    <img class="itemImage" src="…"/>
    <span>[Result name]</span>
</div>
```

- Fill the DOM node with the corresponding information. If possible, add the artist / album / playlist image.
- Don't forget to attach the new node to the target div.
- By default, the `.item` class has a `display:none` property. Fade the item in, once it was attached to the target div.

Event Handling

1. Make sure your script is only executed after the DOM has loaded. jQuery provides a very handy event listener for this: `$(document).ready(…)`.
2. Attach an appropriate event handler to the input field. The AJAX request should be triggered instantly when the user types. `$(this).val()` will give you the content of the input field if called inside the event handler. If possible, only react to character keys and not control or meta keys, e.g. the Shift-key. Use the `search` function you implemented earlier.
3. To determine the type of the search, you need to find out which radio-button is currently selected in the UI. The CSS pseudo-class `:checked` is very helpful here.
4. When the user selects a new search category by clicking a different radio button, we want to relaunch the search. Consequently, you need to attach a 'change' event handler to the radio buttons. It should also find out the current value of the search box.

Further remarks:

The video above shows a quite elaborate solution. Don't bother if your solution does not achieve the proposed functionality within the short timeframe (it also took us longer than 45 minutes to code that).

However, we encourage you to go beyond the tasks in this breakout session. Be creative and continue working on it when you are home.

Continue to ask questions, our tutors are happy to help you.