Assignment 7 (NF, minor subject)

Due: Fri 11.12.2015; 12:00h (1 Week)

Goals

After doing these exercises,

- You know what lazy loading is, and when and why it is useful.
- You are able to work with an API

Task 1: Lazy Loading

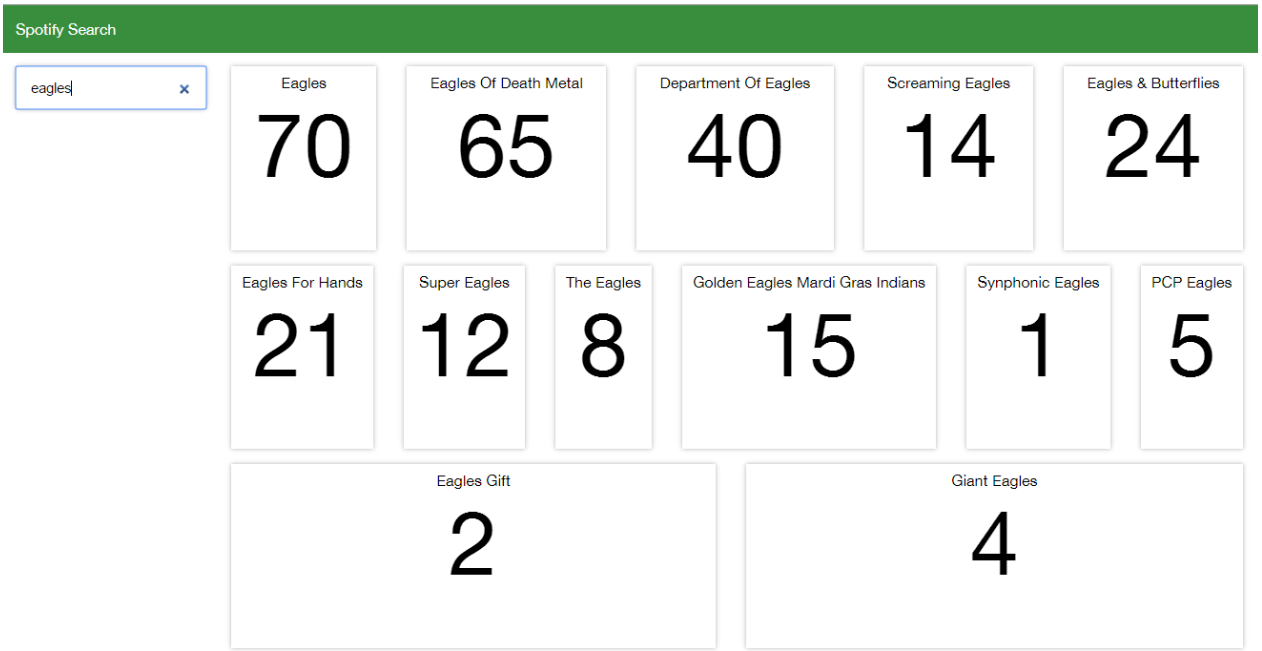
Research online what “lazy loading” means. What does it have to do with AJAX? When is it necessary? Write 300 words.

Put your text into the folder ‘task1’.

Task 2: Spotify Search

In tutorial 07, we implemented a movie search during the breakout session. Now you should create a similar web-page with the Spotify Search API to search for artists.

Goal:
The page displays a search box. With every key stroke, an AJAX request to the Spotify search API is started (`https://api.spotify.com/v1/search`).



Spotify Search

eagles ×

Eagles 70
Eagles Of Death Metal 65
Department Of Eagles 40
Screaming Eagles 14
Eagles & Butterflies 24
Eagles For Hands 21
Super Eagles 12
The Eagles 8
Golden Eagles Mardi Gras Indians 15
Synphonic Eagles 1
PCP Eagles 5
Eagles Gift 2
Giant Eagles 4

We provided a skeleton that you can complete[1]. Take a look at all lines that are marked with a **TODO** and read the comments above the TODO. Also make sure to take a look at the tutorial slides, which have all the necessary functions in them. You do **not** need PHP.

Here is an overview of what you need to do:

1. Line 81: Select the search-box and add an event listener to it. Use jQuery's .on() method for this. Find out the current content of the input field with the correct jQuery method (line 88).
2. Create an AJAX request to the Spotify search API (line 107). As you can see in its official documentation, it requires these two parameters with each request:
   `q`: the search term, in our case the artist name.
   `type`: what kind of search we want to perform, in our case `'artist'`.
   Use this information to set the data for the settings in the AJAX request (cf. tutorial slides).
3. Provide a success handler (callback function) that handles the response of the API. This function should take care of inserting the result of the search into the webpage. This is where the function `insertResultsIntoWebpage(…)` comes into play. What we need is in the **`"items"`** attribute of the response. The code helps you with looping through this array.

Example response for https://api.spotify.com/v1/search?q=deez&type=artist (shortened):

```
{
  "artists": {
    "items": [
      {
        "external_urls": {
          "spotify": "https://open.spotify.com/artist/2w0Dmj9GV9ZrokNRcnRwav"
        },
        "href": "https://api.spotify.com/v1/artists/2w0Dmj9GV9ZrokNRcnRwav",
        "id": "2w0Dmj9GV9ZrokNRcnRwav",
        "images": [
          {
            "height": 563,
            "url": "https://i.scdn.co/image/08031fd21228ae148f4180873b2f61e4596dd621",
            "width": 1000
          }
        ],
        "name": "Deez Nuts",
        "popularity": 45,
        "type": "artist"
      }
    ]
  }
}
```

4. Complete the code for `insertResultsIntoWebpage(…)`. First, clear the contents of the result-container (line 41). In the `forEach` loop, construct an element that will be used as a container for the artist's name and popularity (line 53). Hint: wrapping HTML markup in a jQuery function $(<div>) does the trick. Then, create an element for the name and also for the popularity in the same way (lines 58 + 59).
5. Append the name and popularity to the container (lines 62+62) and finally, append the container to the result (line 66)

Put all source code files into the folder 'task2'.

[1] Of course you can challenge yourself and create the solution from scratch. We will be happy to correct this, too!

Task 3: Participate in our Evaluation

We want to continually improve the tutorials. This is why we conduct a short survey to collect your feedback and suggestions.

Please take 5 minutes time to fill out this anonymous survey

http://goo.gl/forms/DTdGh4TYaC

Thank you!

Submission

Please turn in your solution via UniWorX as a ZIP archive. You can form groups of up to three people. We look forward to your solutions!