Yahoo Search FE Coding Exercise

Introduction

This coding exercise is intended to test your ability to implement a simple client-side feature using HTML, CSS and JS **without the aid of a framework**.

The end product of this exercise will be a single html file that contains all of your code. Please submit this html file as an email attachment to the address you have been provided.

If you are unclear about a requirement, take your best guess or improvise. It's ok if your final product does not look identical to the mocks provided.

The two most important things that you need to demonstrate are:

1. Your ability to implement a carousel using vanilla javascript.
2. Your ability to implement a complex multi-column page layout using CSS and HTML.

You should also feel free to add additional features or functionality if you wish.

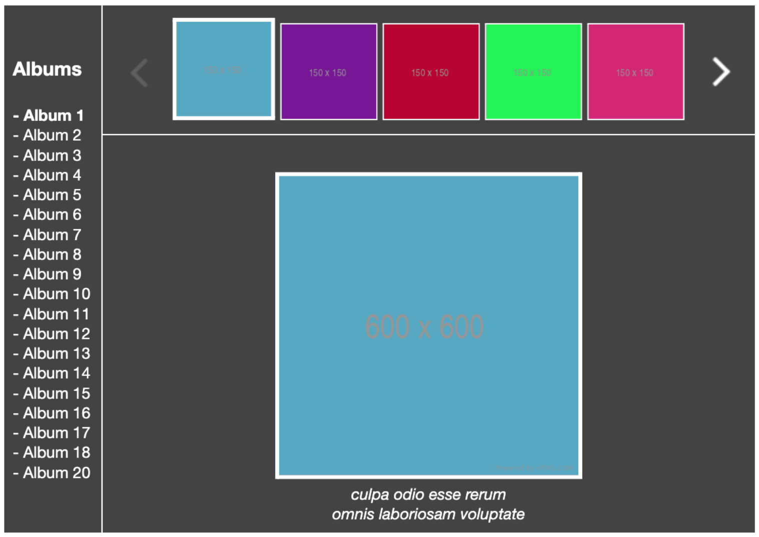
If you have any questions, feel free to reach out to the contact you've been provided.

Guidelines

- All work should be your own.
- Do not use any frameworks to complete this exercise. Use plain HTML, CSS and JS.
- You should expect to spend 1-3 hours on this exercise depending on your level of expertise.
- Your code should be production quality. Reusability, maintainability, performance and readability all matter.
- Use comments liberally so that it is easy for someone else to understand your code.

Requirements

You will be implementing a web page with a simple image viewer that uses https://jsonplaceholder.typicode.com/photos/ as a data source.



API

The structure of the json photo api is very basic and does not contain any nested objects. It also does not accept any parameters so the entire data set is returned for every call.

There are a total of 5000 image objects in the API response.

Each image object contains the following fields:

albumId
The album that the image belongs to. There are 100 albums and each album contains 50 images.

id
The id for the image. This value is unique and assigned sequentially. In other words, the id of the first object is ‘1’ and the id of the last object is ‘5000’.

title
The image caption.

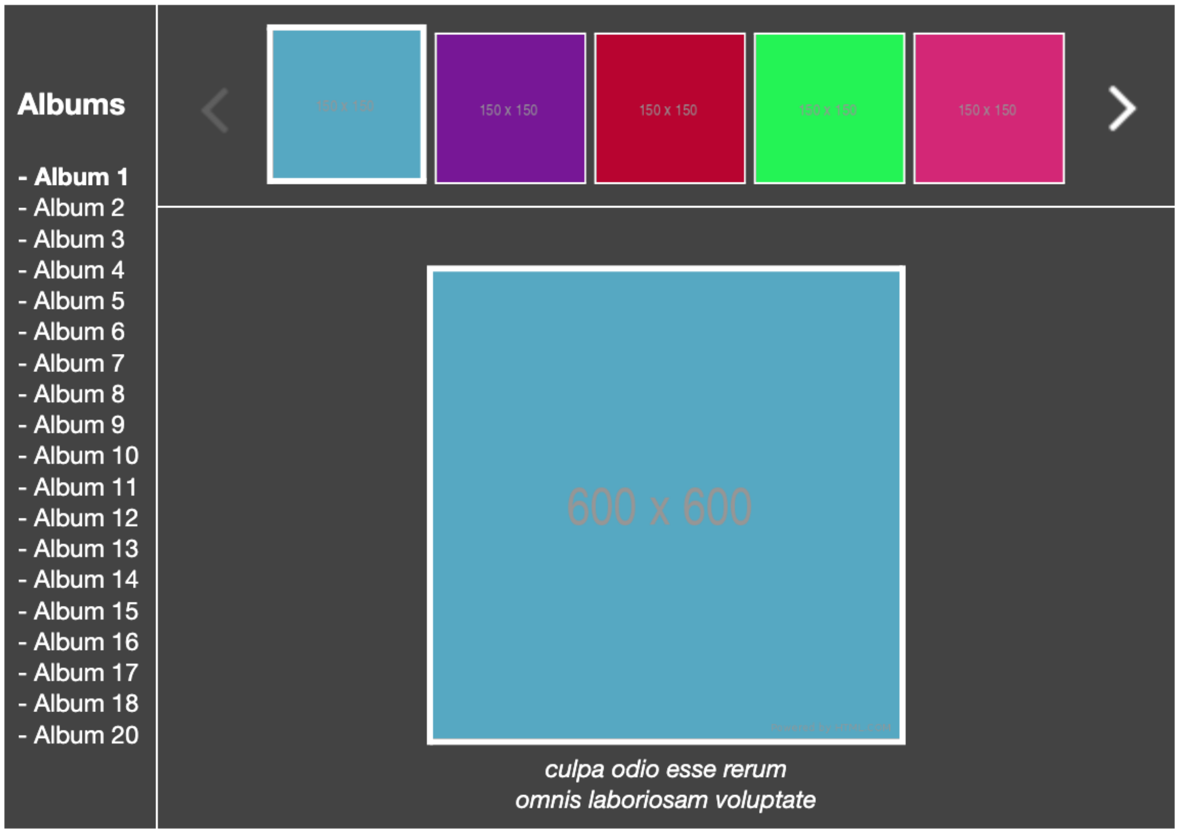
url
The src url for the full sized image. All images are 600 x 600 px.

thumbnailUrl
The src url for the thumbnail version of the image. All thumbnails are 150 x 150 px.

Sample Object

```
{
    "albumId": 1,
    "id": 1,
    "title": "accusamus beatae ad facilis cum similique qui sunt",
    "url": "https://via.placeholder.com/600/92c952",
    "thumbnailUrl": "https://via.placeholder.com/150/92c952"
  }
```

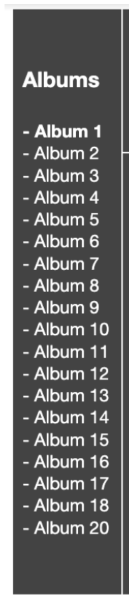
User Interface



The page is made up of three regions:

- Album List
- Thumbnail Carousel
- Image Viewer

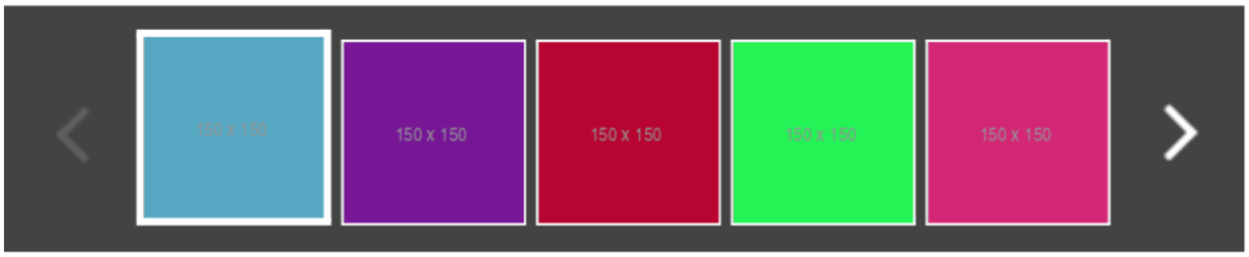
Album List



- ○ The album list displays links for the first 20 albums in the API in a fixed width column.

- ○ When the user clicks on the name of the album, three things should happen:
    - ■ The album link text should be bolded.
    - ■ The thumbnails for that album should load in the Thumbnail Carousel
    - ■ The full sized version of the first thumbnail should be displayed in the Image Viewer.

Thumbnail Carousel



- The Thumbnail Carousel should display the thumbnails associated with whichever album the user has selected from the Album List.

- The thumbnails should be displayed in a single row.

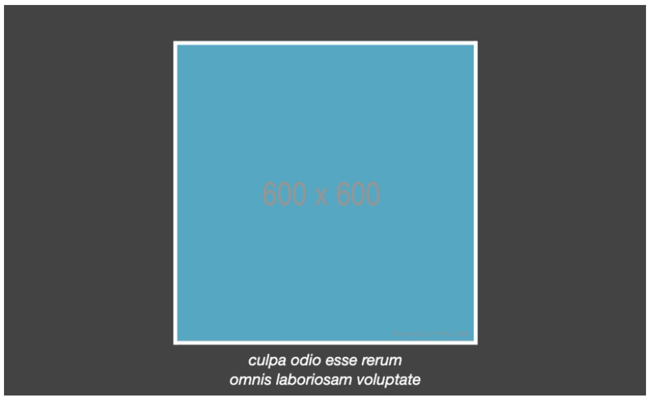
- The width of the carousel and the number of thumbnails displayed should be responsive based on the width of the browser window. The minimum number of thumbnails displayed should be 5.

- The user can scroll through the thumbnails for the selected album by clicking on the right and left arrows.
    - ○ When the user clicks on the right arrow, the next 5 thumbnails should be displayed.

    - When the user clicks on the left arrow, the previous 5 thumbnails should be displayed
    - When the user is at the beginning of the list, the left arrow should not be clickable and the opacity should be reduced to 15%.
    - When the user is at the end of the list, the right arrow should not be clickable and the opacity should be reduced to 15%.
    - Sprite for arrows. https://s.yimg.com/pv/static/img/phoenix2x-1639465063192.min.png

- When a user clicks on a thumbnail, two things should happen:
    - The full sized version of the thumbnail should be displayed in the Image Viewer.
    - The border width on the selected thumbnail should increase to 2px.

Image Viewer



- The Image Viewer should display the full sized image and image title that corresponds to the thumbnail selected in the Thumbnail Carousel.

- The width of the image viewer should flex along with the thumbnail carousel in response to the width of the browser window.

- The full sized image and image title should always be centered within the image viewer.

Default State

The default state of the page when it loads should be as follows:

- The ‘Album 1’ should be selected by default.
- The Image Carousel should be populated with the thumbnails for ‘Album 1’.
- The Image Viewer should be populated with the full sized image of the first thumbnail for ‘Album 1’.

Styles

Other than the layout of the page elements, do not spend a lot of time defining styles for this exercise. For the sake of simplicity use the following styles throughout:

- Font: helvetica neue, helvetica, 12px
- Background: #434343
- Border and text color: #FFFFFF
- Icon Sprite: https://s.yimg.com/pv/static/img/phoenix2x-1639465063192.min.png
    - Active state: 100% opacity
    - Inactive State: 15% opacity
- Spacing: Use whatever looks good to you.
- All thumbnails have a 1px border around them. In the selected state, they have a 2px border around them.
- Full size images have a 2px border around the,

Getting Started

In the real world, js and css are frequently stored in separate files, but for the sake of simplicity, please add them inside of the html document. Here is a basic template you can use.

```html
<!DOCTYPE html>
  <head>
    <title>title</title>

    <style>

      <!-- css goes here →

    </style>
  </head>
  <body>

    <!-- content goes here -->

    <script>

    <!-- js goes here →

    </script>

  </body>
</html>
```