WEB222 Assignment 5 – DOM and CSS Programming

Due date

Monday August 9, 2021 by midnight

Grade value: 5% of your final course grade

Objectives

This assignment will help you learn and practice interactive DOM and CSS programming, and let you work on CSS layout. Please do this assignment on your own and submit only your own work. Gaining experience with JavaScript and the web takes a lot of personal practice and working on these problems yourself will help you build your skills.

We will continue to extend our application from Assignment 4, using iNaturalist (https://www.inaturalist.org/) wildlife observation data. In order to complete Assignment 5, you must have a working version of Assignment 4.

Setup

Begin by **making a copy of your Assignment 4**. Since you will be modifying your current code, it's a good idea to do your new work in a copy so you can go back to consult your previous work. Do not modify your original assignment directly: work only in your copy!

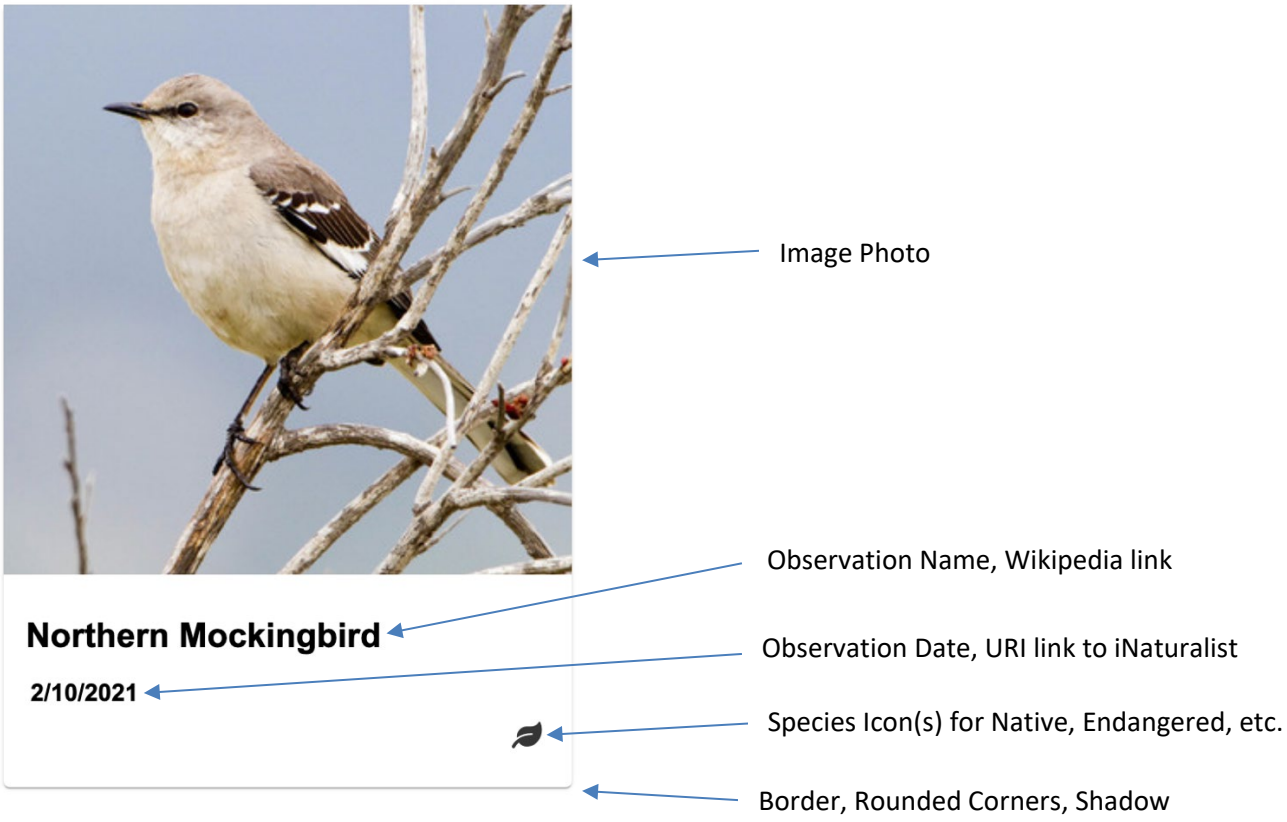
If you have not completed Assignment 4 already, please speak to your professor to make a plan.

Instructions

After the success of your work completing the lost website, you have been asked to make modifications and redesign the layout. See the video of the finished version to get a sense of how things should work.

The company is interested in moving away from using a table, and instead wants to use "cards" to show each observation with more emphasis on the visuals (i.e., less text). See https://www.nngroup.com/articles/cards-component/ for a discussion of cards in web design.

Each observation's card will look something like this:

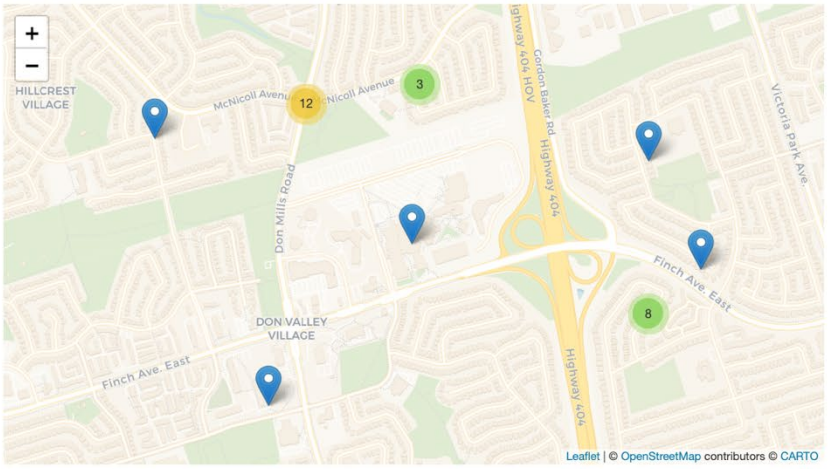


Notice the card's slightly **rounded corners**, **border**, and **slight shadow**. Also take note of the use of **whitespace for margins and padding** around the various elements and the **different sizes of the text and icon**.

The entire table will need to be replaced with a containing these "cards," along with a **legend** showing the different icons that might appear and what they mean:

Here is an example of how the final page should look (NOTE: only the map and some of the cards are shown):

Map



Wildlife Observations: All Species (29)

Legend

Cards

All Species | Only Native Species | Only Introduced Species

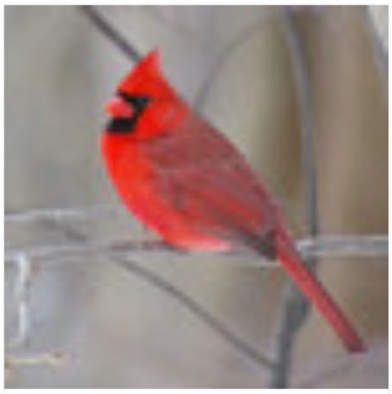
Native | Introduced | Threatened | Endangered

Muskrat

1/10/2021

Northern Mockingbird

2/10/2021



Northern Cardinal

11/5/2020

Common Eastern Bumble Bee

10/5/2020

Here is a sample of the final HTML used to create one of these cards. Each card is made from existing HTML5 elements and attributes, and needs to be generated dynamically using JavaScript, DOM methods, and CSS:

```
<div class="card" id="60706122">
  <div
    class="card-img"
    style="background-image: url(background-image: url('https://inaturalist-open-
data.s3.amazonaws.com/photos/10177220/medium.jpg?1545693877');"></div>
  <div class="card-body">
    <h3>
      <a href="https://en.wikipedia.org/wiki/Campsis_radicans">American Trumpet Vine</a>
    </h3>
    <h4>
      <a href="https://www.inaturalist.org/observations/60706122">9/25/2020</a>
    </h4>
  </div>
  <div class="card-icons">
    <i class="fas fa-leaf" title="Native"></i>
    <i class="fas fa-radiation-alt" title="Threatened"></i>
  </div>
</div>
```

Notice a few things:

- The **id** on the card matches the id of the observation
- The use of **classes** to identify and style various parts of the card for styling: **card, card-img, card-body, card-icons**
- The image is done using a **background-image** inline style on a vs. an <img>, so that we can fit them all in the same size area (see instructions below)
- The photo URL's name **"square.jpg"** has been changed to **"medium.jpg"** in order to load a higher-quality image
- The use of **<a>** elements throughout to link to different things like Wikipedia and iNaturalist
- The use of the **<i>** element to create icons using **Font Awesome** (see instructions below)

TODO Checklist

Here is a list of the things you need to do in order to implement this new design:

1. **src/index.html**

    - We will be using Font Awesome for our icons (see https://fontawesome.com/) and need to add its stylesheet. Add the Font Awesome Stylesheet to the document's head: https://use.fontawesome.com/releases/v5.12.1/css/all.css

    - Remove the <table> from the observation-data

- Above observation-data , add a new ** **. In it, create a for each legend label, as well as the associated Font Awesome icon. See https://fontawesome.com/how-to-use/on-the-web/referencing-icons/basic-use for instructions on how Font Awesome works. The icons and classes we are using are:
  - Native: fas fa-leaf
  - Introduced: fas fa-frog
  - Threatened: fas fa-radiation-alt
  - Endangered: fas fa-skull-crossbones

- Replace any references to "table" in the HTML document to something more appropriate. Remember what you change, since we'll have to update our JavaScript and CSS files to match.

2. **src/js/ui.js** needs to be updated to work with "cards" vs. table elements. Use the existing code for tables as a model for how you should write the new functions. You need to break down the HTML for a card (see above) into smaller components and write functions that convert observation data into DOM elements. Here are some suggestions:

   - **function buildCardForObservation(observation) {…}** should accept an observation Object and use it to call the functions below in order to create a card. It should return the completed card (i.e., the containing element).

   - **function cardImg(url) {…}** should accept an observation's photo URL. It should replace the word "square" for "medium" in the URL to use a better-quality image. It should create and return a with the card-img class name, and an inline style of background-image: url(…) using the modified URL. Once complete, it should return this .

   - **function cardBody(name, date, uri, wikipediaUrl) {…}** should accept observation properties and use them to create the card's body (a). Inside this, it should create <h4> and <h3> elements that include the name and formatted date. The name should link to the observation's Wikipedia article, and the date to the iNaturalist URI. Once all of this is complete, return the

   - **function cardIcons(*isNative*, *isIntroduced*, *isThreatened*, *isEndangered*) {…}** should accept observation properties and use them to create the icons (if any). For each property, check if its value is true, and if so, create the appropriate icon using Font Awesome and the <i> element (see notes above on

the classes to use). Add each icon to a and return that from your function.

3. **src/js/app.js** needs to be updated to replace all references to "table" with "card" or "cards" (e.g., in function names, comments, etc). Make sure do this carefully so things match in your HTML, CSS and other JavaScript files. You also need to modify the code to call the **buildCardForObservation()** function you wrote above to create cards for each observation, and add them to your observation-cards div.

4. **src/css/style.css** needs to be updated to style the cards and additional content on the index.html page. Pay attention to the screenshots above and assignment video, and make your page look as close to them as possible. Here are some tips on how to approach this:

    - Your **observation-cards** div should use **Flexbox** to fit the cards into the available space, see https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Flexible_Box_Layout/Basic_Concepts_of_Flexbox. When the page gets narrow, you can use **flex-wrap** to reduce the number that are shown on a line to 1 from 2, see https://developer.mozilla.org/en-US/docs/Web/CSS/flex-wrap. You can also use **justify-content** to evenly space your cards, see https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Flexible_Box_Layout/Basic_Concepts_of_Flexbox#justify-content

    - Your **card** class should use a height of **515 pixels**, and use **25 pixels** of margin on the bottom, to space cards apart from each other. Set a **border-radius** to round the corners a bit, and use a box-shadow to achieve the border and shading:

```
box-shadow: 0px 2px 1px -1px rgba(0,0,0,0.2),
            0px 1px 1px 0px rgba(0,0,0,0.14),
            0px 1px 3px 0px rgba(0,0,0,0.12);
```

    - Your **card-img** class should be **375 pixels** square (same width and height). To force the images to all be the same size, we use a CSS trick: set the image as a **background**, which we can manipulate more easily, see https://developer.mozilla.org/en-US/docs/Web/CSS/background-size (recall that we already set the background-image in the JavaScript via a style attribute):

```
background-position: 50% 50%;
background-repeat: no-repeat;
background-size: cover;
```

- Your **card-body** class should add **15 pixels** of padding to the top and bottom and use the **Arial font**. Additionally, links in the card-body need some special treatment. By default, all **underlines should be removed** unless the user **hovers** the link, and the link **colour should be black**.

- Your **card-body** contains h3 and h4 elements for the name and date. Make the h3 element a bit larger than the h4 and reduce the margins on the h4 element so that they sit closer to each other.

- Your **card-icons** class needs to **right align** its contents. It also needs to use **15 pixels** of padding on the left and right, but none on the top and bottom.

- The **i elements within the card-icons div** need a **5 pixel** margin and their font size increased a bit.

Debugging

No matter how experienced you get at programming, you will always deal with difficult bugs in your code. The difference between a beginner and an expert isn't so much how much they know about a language, as it is how well they know how to use their development tools. Learning to use the browser's built-in developer tools is a super power that you need to acquire.

This is true of JavaScript as well as CSS, and learning how to debug CSS issues in dev tools is very important. This video gives a good introduction to many of the techniques you can use https://www.youtube.com/watch?v=VYyQv0CSZOE.

Submission

When you have completed your assignment, you need to prepare your submission. To do so, use the npm command:

```
npm run prepare-submission
```

You can upload and submit the submission.zip to Blackboard.