WEB-215 Final Project

Objectives:

- Enhance two websites so that it is interactive by implementing jQuery.
- Make use of jQuery's built-in methods, UI, and themes.
- Abstract custom functions so they are usable as a library.
- Link two different sites to the same single jQuery file, to the same single custom JavaScript library, and to a unique "function calling" file.

Before You Begin Writing Code

- In your unzipped folder, you will find two website folders: *Final project Duck*, and *Final project Fish*
- Browse through the first two to familiarize yourself with the current non-JavaScript functionality
- You will begin with the *Final project Duck* site, then later implement your JavaScript on the *Final project Fish* site
- Let's identify details of the first site, *Final project Duck*:
    - There are no "current page" highlights in the navbar
    - FAQ – all content displays immediately
    - Resources – table is not striped; links open in the same browser window
    - Ducks – thumbnails link directly to JPGs, which open in the same window
    - Contact – default values are not sticky; there is no form validation; there is no submit button
    - Media – no external content is displayed on the page
- If you examine the details of the second site, *Final project Fish*, you will find similar behavior (and an identical lack of interactivity)
- Lastly, let's examine the CSS that is related to the JavaScript for the first site, *Final project Duck*:
    - The class *.page Highlight* should be used to mark the current page
    - At the bottom of the CSS, you will need to create three classes for your table row states, such as .odd-row, .even-row, .over-row. You may use any names you like.

What You May Not Do

You may not…

- …edit any of the HTML except to add a link in the HEAD to your jQuery file
    - Exception: You may need to modify the HTML structure so your content works with a jQuery accordion, tab, UI plug-in, interaction, etc…
    - These changes should be minimal and only performed if required by a jQuery plug-in.
    - You may not rearrange the order of existing content as it is optimized for SEO.
- …edit any of the CSS except to add your table row colors at the bottom of the CSS file for the main site

The JavaScript

- Use jQuery throughout
- Abstract any custom functions
- Do not place any JavaScript code in the head. Use an external JS file and link to it.
- Place both your jQuery file and your custom JavaScript file in a single folder that is used by both sites. Additionally, each site should link to a third JavaScript file that will simply call the functions and pass arguments. See the last item in these instructions for an example.
- Now let's examine page-by-page what you need to make happen.

All pages

- Current page should be identified in the navbar by dynamically applying the *.pageHighlight* class (the class already exists in the provided CSS file). Note you will need to change this for the Fish site where you will use the *.currentLink* class.
- Use a consistent jQuery theme that works well with the existing site design

FAQ

- Create/use a jQuery accordion to dynamically display the FAQ answers. They should all be hidden on page load.
- Accordion must:
    - Allow only one FAQ open at a time
    - Allow all FAQs to be hidden/collapsed
    - Auto-detect the content height; expanded FAQs should not all be the same height
    - Open on click, not mouseover
    - Animate a smooth slide

Resources

- Stripe the table rows using the CSS rules you created – this is the only edit to the CSS you should make
- Make the table links open in new tabs/windows, canceling the default link behavior
    - Make sure only external links open in new tabs/windows – not just links in this table!
    - The 'external links' function should work for ANY external link on the site – like the one on the home page.
- On hover, the cursor should change to a hand (even if pointing to a year)
- The entire row should be 'an active link'. You should be able to click a year (even though it's not a hyperlink) and have the external site open in a new window.
- Format the years to appear to be hyperlinks so they are underlined (a visual cue to the user) like the other text

Ducks

- Implement a jQuery image gallery (not just a rotating slide show)
- If the images automatically rotate, users should be able to start/stop it and still click any thumbnail to display the larger image

Contact

- Default values should disappear when a field has focus and should return if the field is left empty
- All fields except 'comments' should be required. They are already flagged in the HTML with class="required". You may edit the class in the HTML as necessary. No required field should be allowed to be left blank or contain the default value.
- Form should not submit unless all required fields pass validation
- Implement a jQuery Datepicker in place of the Date text field. Restrict the date so that users are not able to choose a date in the future. (Nothing later than 'today'.)
- Note the form points to a form handler, process_form.html. Do not edit process_form.html!
- The form should submit when the correct image is dragged to the correct target image. This will be a challenge. The hints below will help.
    - Only the top three images should be draggable
    - All three bottom images should be droppable
    - The dragged image should return to its original position after being dropped
    - As soon as dragging starts, the three drop images should be formatted with the class defined in the CSS
    - See next page for login hints.

Logic Hints for Drag and Drop Feature

- Pass one argument in your draggable function call:
    - …to identify which elements can be dragged (you will likely have to edit your HTML to help with this)
- Pass five arguments in your droppable function call:
    - …to identify which elements can be dropped onto (you will likely have to edit your HTML to help with this)
    - The ID of the correct image to drag (consult the existing HTML)
    - The ID of the correct image on which to drop (consult the existing HTML)
    - The ID of the form to submit (consult the existing HTML)
    - The class to assign the targets when dragging starts (consult the existing CSS)
- Code the abstracted drag function to (consult the jQuery Draggable API documentation):
    - Revert the dragged image to its original position after a drop
    - Display the cursor as a 'move' icon
    - Increase the z-index of the element being dragged
- Code the abstracted drop function to (consult the jQuery Droppable API documentation):
    - Set the tolerance so the dragged element simply has to intersect the drop element
    - Apply the correct CSS class to droppable elements on hover so users can see the drop element 'highlight' when it is okay to drop
    - Run an anonymous function on drop to:
        - Compare the dragged element's ID to the "correct ID" that was passed as an argument
        - Compare the ID of the element that was dropped onto to that of the "correct drop ID" that was passed as an argument
        - If they both match, then the correct item was dropped on the correct target, so submit the form

Media/Movies

- Similar to the New York Times JSON/API projects you did, this page should display the results of a movie review search using the New York Times' Movie Review Search API. However, the user should not be allowed to specify the search term. This should be hard-coded as 'duck' (or 'fish'). How you wish to display the results is up to you. You may choose to include pagination or simply display the first page of results only.
- At a minimum, the total number of movies available on your site must appear at the top of the results. Each movie must show its title, rating, short summary, opening date, and link to the full review that opens in a new tab/window. (If you do not include pagination, the total shown at the top should match the number of movies actually shown. If you do include pagination, the number up top will be more than what is show on the 'first page of results'.)

The Fish site

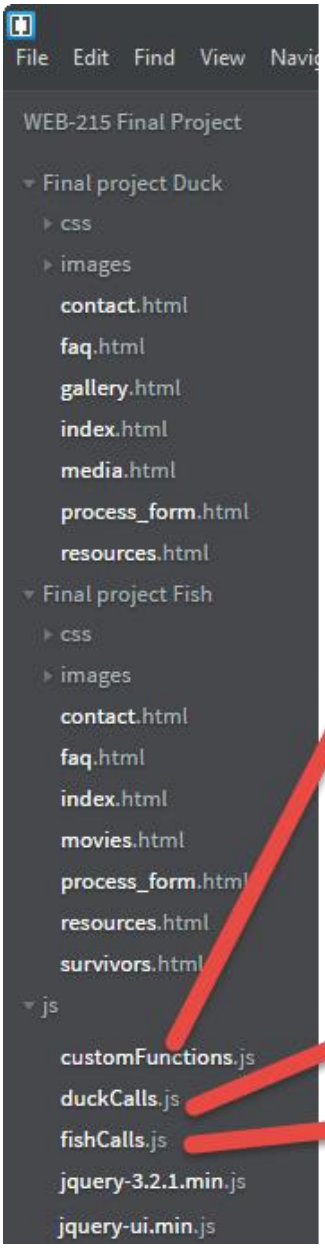
- Ensure your Fish site uses the same custom library JS file as the main site. Provide a unique "function caller" JS file to call the library functions while passing the necessary arguments.
- You may not edit any CSS in the CSS files! Table row colors are already created. Scroll to the bottom of the CSS to see them as you will need to know these classnames.

Restrictions and other notes

- JS should be 'lean and clean' and make use of jQuery
- Everything should work in current versions of Firefox, Chrome, Safari, IE, and Edge (versions installed on classroom computers)

What to turn in

- Create a root folder named **Last-First-jQuery-Final** (use your own first/last name).
- Place the **Final project Duck**, **Final project Fish** and **js** folders in this root folder.
- The **Final project Duck** folder should contain the HTML, CSS, and images necessary for the site. No JavaScript should be in this folder!
- The **Final project Fish** folder should contain the HTML, CSS, and images necessary for the site. No JavaScript should be in this folder!
- The **js** folder should contain all JavaScript files
    1. The shared jQuery file
    2. The custom library
    3. The "function caller" for Final project Duck
    4. The "function caller" for Final project Fish
- If you downloaded additional files for a jQuery plugin, be sure to place all JavaScript files in the shared **js** folder. If there are any images or CSS that needs to be shared, place those files in the shared **js** folder also inside their own subfolders. Even if it is CSS, place it in the **js** folder if that CSS is part of a shared JavaScript library. In other words – anything that is shared across both sites must reside in the shared **js** folder.
- See the next page for a diagram!




```
// CustomFunctions.js
// all your custom functions here

function myFunction1(parameter, paremeter){
    ...
}

function myFunction2(parameter, paremeter){
    ...
}
```

```
// Final project Duck function calls

$('document').ready(function(){
    ...call the functions and pass the arguments needed...
    ...
    ...
    ...
});
```

```
// Final project Fish function calls

$('document').ready(function(){
    ...call the functions and pass the arguments needed...
    ...
    ...
    ...
});
```