- Due Dec 4, 2021 by 11:59pm
- Points 100
- Submitting a file upload

CS-546 Lab 11

AJAX

For this lab, you will be using HTML, JQuery and Client-side JavaScript on the user's browser to make a simple application that makes AJAX requests for the data needed and then inject the elements and data onto the page.

TV Maze API

For this lab, you will be using three endpoints of the TV Maze API for your AJAX calls. The list of shows: http://api.tvmaze.com/shows (Links to an external site.) and then you'll get an individual show using the endpoint http://api.tvmaze.com/shows/:id (Links to an external site.) where `:id` is the ID of the show you are looking up. You will also use http://api.tvmaze.com/search/shows?q=search_term (Links to an external site.)to search the API

The Server

Your server this week should not do any of the processing or calculations. Your server only exists to allow someone to get to the HTML Page and download the associated assets to run the application.

`/` The Whole Application

This route will respond with a static HTML file and all of the functionality will be done in a client-side JS file. You will make client-side AJAX requests to the API and use jQuery to target and create elements on the page.

Your page should have a few basic user interface elements:

- A header tag, with an h1 naming your site, with a title for your page
- A footer with your name, student ID, and any other info about yourself you wish to include
- An empty unordered list with an id of `showList` that is initially hidden.
- A div with an id of `show` that is initially hidden.
- A form with an id of `searchForm`.
- A text input with an id of `search_term`.
- A label with a `for` attribute referencing your input
- A button to submit the form
- A link that links back to the "/" route with the text "Back to All Shows" and that has an id of `homeLink` This link will initially be hidden. It will ONLY be displayed when the `show` element is being displayed or ONLY when a search is performed and search results are being displayed. You should NOT show this link on the initial show list. This link will simply trigger a reload of the page, which then will display the initial list of shows

AJAX Requests

Remember, you will be ONLY using client-side JavaScript!

1. **Page load**: When the page loads, you will query the TV Maze API to get the list of shows using an AJAX request. Once the AJAX request returns the data, you will then create list items of links for each show that is returned using jQuery. The link text will be the name of the show, and the `href` attribute will be set to the url for that show from the

TV Maze API (the url we need for the link is in the data, in the `_links.self.href` field.) For the link, you will need to call a function on the click event of the link (do not forget to preventDefault() for the default behavior for the link. You will then append each list item to the `showList` UL element and then show the `showList` element (make sure you hide the `show` element).

Endpoint to be used: http://api.tvmaze.com/shows

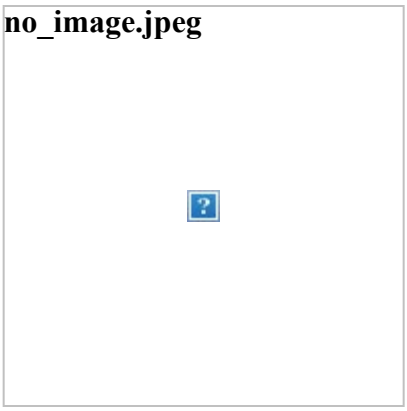
2. **Search Form Submission:** If there is no value for the `search_term` input when the form is submitted, you should not continue and instead should inform them of an error somehow. (don't forget to take into account if they just submit the form with a bunch of spaces as the value!) If there is a value, you will first empty the list item elements in the `showList` element (because there will be elements from the initial showList still there, they are just hidden), then query the API for that `search_term` using an AJAX request. Once the AJAX request returns the data, you will then create list items of links for each show that is returned using jQuery. The link text will be the name of the show, and the `href` will link to the url for that show. (the url we need is in the data in the `_links.self.href` field. You will then append each list item to the `showList` UL element and then show the `showList` element (make sure you hide the `show` element).

Endpoint to be used: http://api.tvmaze.com/search/shows?q=search_term_here

3. **Link Clicked**: For the link, you will need to call a function on the click event of the link and not the default link behavior (do not forget to use preventDefault()). When the link to a show is clicked, it will hide the `showList` element, it will then empty the `show` element (just in case there was show data previously loaded into the `show` element). It will then make an AJAX request to the URL and fetch the data for that show (that was the href in your link). You will parse through the show data returned from the AJAX request. You will create the following elements using jQuery: an `h1` with the show `name`, an `img` which the `src` is set to the value read from `image.medium` in the data which is a URL to an image for the show, and a `dl` (definition list) of the following properties of the matching show: `language`, `genres`(the entire array in an `ul`), `rating.average`, `network.name`, and `summary`. You will then show the `show` element.

NOTE: Not all shows have ALL data displayed on the `show` element, which will cause your application to not work correctly when a show link is clicked if it doesn't have all the needed data needed for the `show` element. You will be required to check each field needed for the `show` element. If there is no value for a field, you will show "N/A" instead of that field's value on the show detail element. For the image, if there is no image, you can load a generic "no image" image that is served from your public directory. You can save this one (right click and save the image):

no_image.jpeg

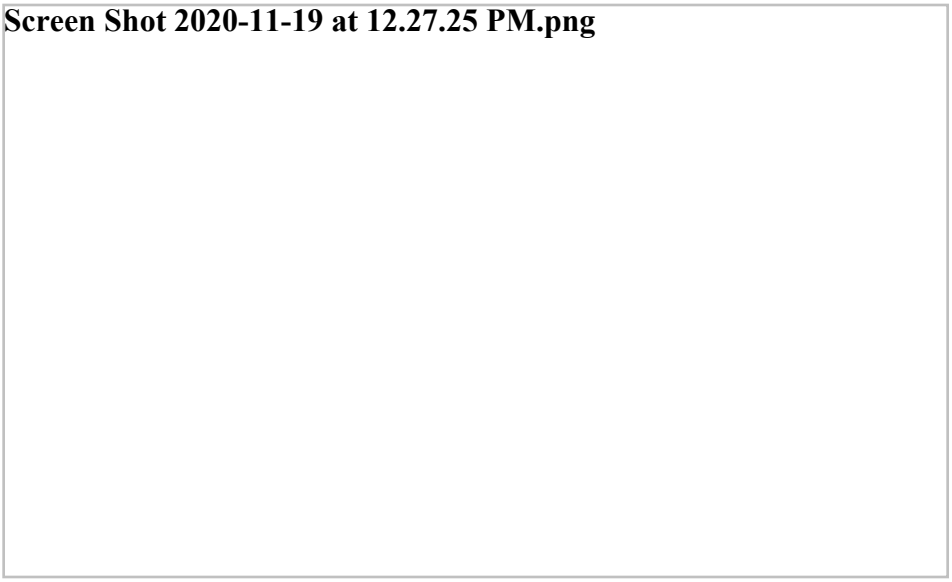


For example: The show "Hemlock Grove" is missing the network, "Anna und die Haustiere" is missing the image and the Average Rating, "Under Arrest" is missing the Language, the Average Rating and the Network. You would display them similar to shown below.

Screen Shot 2020-11-19 at 12.28.01 PM.png

Screen Shot 2020-11-19 at 12.27.46 PM.png

Screen Shot 2020-11-19 at 12.27.25 PM.png



Endpoint to be used: http://api.tvmaze.com/shows/:id (this is read from the href attribute of the link)

Style:

You are NOT required to use CSS for this lab (but you can if you want to).

Requirements

1. All previous requirements still apply.
2. You **must remember** to update your package.json file to set `app.js` as your starting script!
3. Your HTML must be valid (Links to an external site.) or you will lose points on the assignment.
4. Your HTML must make semantical sense; usage of tags for the purpose of simply changing the style of elements (such as `i`, `b`, `font`, `center`, etc) will result in points being deducted; think in terms of content first.
5. **You can be as creative as you'd like to fulfill front-end requirements**; if an implementation is not explicitly stated, however you go about it is fine (provided the HTML is valid and semantical). Design is not a factor in this course.
6. **Your client side JavaScript must be in its own file and referenced from the HTML accordingly**.
7. All inputs must be properly labeled!

Previous Next

Submission

Submitted!
Dec 4, 2021 at 5:44pm
Submission Details
Download YuFu_Liao_CS546_A-3.zip
Grade: 97 (100 pts possible)
Graded Anonymously: no

Comments:

-3 on the homePage ('/'), each list item should li, you had ol

Christina Li, Dec 6, 2021 at 3:11pm

Good Job!

Christina Li, Dec 6, 2021 at 3:11pm