jQuery Interests Assignment

Ethan Busha

Goal of Assignment

The goal of this application is to use our previously gained HTML and JavaScript knowledge to build a webpage. We will combine that with use of our newly gained knowledge of jQuery to manipulate the DOM by dynamically creating and updating it with information. This assignment serves as a mock test for the final project, as we can strictly only compute with what examples we already have previously been given in the class.

Project Use

The way to use my code from the user’s viewpoint is simple. The page presents a list of games that I have enjoyed recently. Hover over the cards to see some more general information regarding the game. The hover function also shows a “More-Information” button to the user, which if they press will call upon a function that will write over the element with .html to be blank, display game information for the appropriate game in that element. The button click passes off the event object, and I use event.target.id to obtain the information I need to identify the correct game. The overwritten div contains a longer summary about the game along with some more images and below everything is a button to go back. This “Go-Back” button calls upon the function that is initially used once the document is ready at the start of the code. The start of this function overwrites the div before dynamically creating all the game information once more. Important to note is that the summary text for the games that the user sees are ai generated, and was the only time ai was used for this assignment.

Rubric Assessment

**Interface is Authentic, Professional, Balanced Interface design is authentic, looks professional, is balanced across the web page**

I have never had a great eye knowing how to make things look great, but out of all of the projects we have done so far, I have put by far the most amount of time for styling into this project. I have a wide array of styling for the webpage by default, but also styling for elements of the page when interacted with. This includes events with jQuery and changing styling for existing elements when a function is called

**Interface links to functional jQuery jQuery is included in application source**

jQuery is referenced and used thoroughly in my project. The source:  
<script src="./lib/jquery-3.7.1.js"></script> is included before any of my two JavaScript files or style files are referenced

**Events are Effective, jQuery jQuery is used to bind events to event handlers<br/> The event handling mechanism works to achieve goals.**

A screen shot of a computer program

Description automatically generated

This section of code is what happens when the user hovers over each game card. Certain aspects of that specific card are changed using (this) and changed back when mouse is off. The “More-Information” button of class “summary” appears when hovering over.

A computer screen shot of text

Description automatically generated

The above code is what happens when the user hovers over the button that appears as a result of hovering over the game card. Styling of the button will change with a mouse over before reverting with a mouse off.

When the “More Information” button is clicked the gameDetails function will be called upon shown with $(".summary").on("click", gameDetails)

A computer screen shot of text

Description automatically generated

The “Go-Back” button is of a different class so I made another effect for mouse over and mouse out

When the “Go-Back” button is clicked the loadGames function will be called upon shown with

$(".return").on("click", loadGames);

**DOM Access, jQuery The DOM is accessed with jQuery**

While I do have default style settings for the webpage, every change that is made with events such as mouse over or mouse out use jQuery to identify what needs to be changed. I also use jQuery to access the DOM for appending elements to one another

A screen shot of a computer code

Description automatically generated

I append element containing game image and game name to the interest element before appending that to the interests wrapper, and then appending that along with the instructions box to the parent container which is accessed using jQuery as I didn’t assign a variable to it. I could have accessed all elements here with their class name and it would have worked the same.

A black background with white text

Description automatically generated

Later on I append elements to the interest wrapper but this time identify it by its class rather than variable assigned to it.

**DOM Dynamically Built, jQuery The DOM is dynamically updated using jQuery calls**

There are elements that I created using html rather than JavaScript

A computer screen shot of text

Description automatically generated

Every element that was created in the body of the html document is in this image. Every other element is created using jQuery in the JavaScript file.

A computer screen shot of text

Description automatically generated

This may be the best example. While it doesn’t show the whole function, it shows that for each game I iterate through I create another instance of multiple elements, each instance having unique information. Later on in the gameDetails function I use

$(".interestsWrapper").html(" ");

To clear all previous information. It’s the same element each time, but after I clear it I populate that element with appropriate information for whatever game was clicked. And when I click the “Go-Back” button that same element is cleared again and filled with game information once again.

**Documentation: Program Code and Project Description .js file is fully annotated, explaining the code, not just stating that a call is being used. There is a document describing how the application works and how to use it, and the goal of the application**

There are plenty of comments in my JavaScript file to explain what is happening and my intention throughout the process. I have also added numerous console logs at critical points to make sure that I am getting a value and what that value is.

console.log("MOUSE IS OVER SUMMARY BUTTON")

every mouse over and mouse off has a command to notify the event, which did help me find an error in making a mouse over event happen correctly.

let index = event.target.id;

console.log(`summaryIndex is ${index}, which should be equal to event.target.id of ${event.target.id}`)

I made sure to console log the variable that was equal to event object id I got for gameDetails function, so I knew before moving forward that I had what I needed.

**Style, jQuery In at least one instance, style is modified or applied using jQuery**

The style of elements has been changed during hovers as shown before, but I also changed certain element styling as a part of functions clicked as a result of buttons which I have not shown yet

A computer screen shot of text

Description automatically generated

By default the interests wrapper has a border radius of 5% and a margin-left of 2%. When we get the game details though we hide the instructions box so there is more room, so I repositioned the element to reflect that.

A screen shot of a computer code

Description automatically generated

When I click the “Go-back” button these attributes are reset to their default.