

Lab 7

Objectives

- Get more practice using JavaScript to interact with HTML
- Read and write JavaScript code with variables, if-statements, loops, and objects

References / Resources

- JavaScript objects
 - https://www.w3schools.com/js/js_objects.asp
- Using JavaScript to perform actions to HTML:
 - https://www.w3schools.com/js/js_htmldom_methods.asp
- Getting started with jQuery:
 - https://www.w3schools.com/jquery/jquery_get_started.asp
- Translation between JavaScript and jQuery:
https://www.w3schools.com/js/js_jquery_selectors.asp

SETUP – Download the necessary files

1. Access the JSFiddle page at: <https://jsfiddle.net/Aestey/k1ab6vp5/>

Part I – Interpreting JavaScript objects

1. For Part I, you will be reading JavaScript code, and writing HTML code.
2. At the beginning of the JavaScript section of the JSFiddle page, an array of course objects is created. Read through the array to determine the details of each course in the array.
3. Based on the array of course objects, fill in the HTML table so it displays the attributes for each item in the array (similar to how I created tables based off of JavaScript object code in the PowerPoint slides).
4. After you have filled in all of the table cells, click run to see if they show up on the resulting web-page.

CHECKPOINT 1

Part II – Create a new course object

1. For Part II, you will be adding code to the JavaScript section.
2. In the HTML, scroll down to the section labeled `<!-- PART 2 -->`
3. In the HTML, there is a button created that calls a function when clicked. The function is called `addCourse()` and it should add a course to the array of courses.
4. In the JavaScript section, scroll down to the section labeled `/* Part 2 */`
5. The `addCourse` function has been started for you. It creates a **newCourse** variable, but the value for the variable isn't set up correctly. Erase the "fix me - this isn't a course object"; and update the line of code so that the **newCourse** variables creates a new course object. On the website (in the bottom right section), you'll notice there is a link to all of the different courses at UVic. Feel free to choose any one you like. If you are out of ideas, you can use CSC 464: Concurrency.
6. If you have created the **newCourse** variable correctly so that it holds information about a course object, clicking the 'Add Course' button should add a new course with the information you entered to the array (and HTML table):

Part 1

Subject	Number	Title
BIOL	355	Evolution
EOS	427	Geophysics
MATH	451	Probability
PHIL	371	Logic

Part 1

Subject	Number	Title
BIOL	355	Evolution
EOS	427	Geophysics
MATH	451	Probability
PHIL	371	Logic
CSC	464	Concurrency

Part 1

Add course

[Click here](#) for UVic course calendar

Part 2

Add course

[Click here](#) for UVic course calendar

CHECKPOINT 2

Part III – Add a course based on textbox input information

1. For Part III, you will be adding code to the JavaScript section.
2. In the HTML, scroll down to the section labeled `<!-- PART 3 -->`
3. In the HTML, there is a button created that calls a function when clicked. The function is called `addCourse2()` and it should add a course to the array of courses.
4. The difference between this and Part II is that the course information should be added in the text input boxes on the webpage, and that information should be used to add a new course to the course array.
5. In the JavaScript section, scroll down to the section labeled `/* Part 3 */`
6. The `addCourse2` function has been started for you. It creates a **newCourse** variable, but the value for the variable isn't set up correctly. Erase the `"fix me - this isn't a course object"`; and update the line of code so that the **newCourse** variables creates a new course object. The values for the course's subject, number, and title should be read in from the text boxes in the HTML. You will need to remember how to get the value entered into the text box (based on its id) in JavaScript.
7. If you have created the **newCourse** variable correctly so that it holds information about a course object, clicking the 'Add Course' button in the Part 3 section should add a new course with the information you entered to the array (and HTML table):

The diagram illustrates the process of adding a new course to a web application. It shows two states of the application: before and after the addition of a new course.

Left State (Before):

- Part 1:** A table with 4 rows of course data: BIOL 355 Evolution, EOS 427 Geophysics, MATH 451 Probability, and PHIL 371 Logic.
- Part 2:** An 'Add course' button and a link to the UVic course calendar.
- Part 3:** Input fields for Subject (SOC), Number (326), and Title (Social Networks), with an 'Add course' button below them.

Right State (After):

- Part 1:** The table now has 5 rows, including the new course: SOC 326 Social Networks (highlighted in red).
- Part 2:** Same as the left state.
- Part 3:** Same as the left state.

A red arrow points from the input fields in Part 3 of the left state to the new row in the table of the right state, indicating the data flow.

CHECKPOINT 3

Part IV – Loop through array of courses and display a result

1. For Part IV, you will be adding code to the JavaScript section.
2. In the HTML, scroll down to the section labeled `<!-- PART 4 -->`
3. In the HTML, there are three buttons created that call functions when clicked. The functions should display information in the Part 4 section of the webpage.
4. In the JavaScript section, scroll down to the section labeled `/* Part 4 */`
5. The first button calls the function `allClasses()`. The `allClasses()` function creates a variable named **`allClasses`** and adds information about each class to the end of it in the for-loop when it visits each course in the array. Read through the different pieces of this code, as you will be writing similar code for the next two functions.
6. Add code to the `fourthYearClasses()` function so that it adds information about fourth year classes to the variable valled `fourthYear`. Look to the `allClasses` function above to recall the syntax for looping through an array and adding text to the end of a variable. The main difference is for this function only want to add course information for 400-level courses. You will need to use an if-statement to determine whether a course is a 4th year course or not. After completing the functions, clicking the button should display information for 4th year courses.
7. Similar to step 6, the `mathClasses()` function should only display information for courses with subject MATH.

Part 1

Subject	Number	Title
BIOL	355	Evolution
EOS	427	Geophysics
MATH	451	Probability
PHIL	371	Logic
CSC	464	Concurrency
MATH	423	Graph Theory

Part 2

Add course

[Click here](#) for UVic course calendar

Part 3

Subject: Math
Number: 423
Title: Graph Theory
Add course

Part 4

List classes
List 400-level classes
List Math classes

Click

Part 1

Subject	Number	Title
BIOL	355	Evolution
EOS	427	Geophysics
MATH	451	Probability
PHIL	371	Logic
CSC	464	Concurrency
MATH	423	Graph Theory

Part 2

Add course

[Click here](#) for UVic course calendar

Part 3

Subject: Math
Number: 423
Title: Graph Theory
Add course

Part 4

Classes: BIOL355 EOS427 MATH451
PHIL371 CSC464 MATH423
List classes

400-level Classes: EOS427 MATH451
CSC464 MATH423
List 400-level classes

Math Classes: MATH451 MATH423
List Math classes

CHECKPOINT 4 – LAB COMPLETE