Exercise04: Javascript

Objectives:

To learn to use **JQuery**, **JSON**, and **Object-oriented Javascript**.

Work with your group (or by yourself). Each group is to upload only one submission.

Credits (for edits): Zahra Hosseini ,Naresh Somisetty, Isaac Martin, Sai Nemalikanti, Alexandra Lowry.

EXAMPLES

00: show the order in which js is loaded and executed and ready and onload events are fired.

01: shows self-invocation of functions. Also, shows use of call, apply, and bind.

02: shows JSON stringify and parse methods.

03: Shows adding action with Jquery to DOM

03-2: Shows adding action with Jquery to DOM via id attribute, tag and class

03-3: create table with Jquery (i.e. use jquery to create DOM elements)

04: two ways to create objects

05: factory pattern of creating objects and why it does not work.

06: shows constructor pattern for creating js objects and the problem with that.

07: shows the prototype pattern for creating js objects and the problem with that.

08: shows the constructor+prototype approach of creating js objects.

1. Warm Up: Try Some Examples

1. First, open blackboard, go to Course Contents, and then download exercise04.zip file into your workspace (U:\workspace or something like that!). Then, unzip.

2. Play with each of the given examples (in examples directory). Open them using a text editor of your choice and modify parts of the html or js files.

Please do the TODO segments for each example.

You will need to also learn how to use the available tools for JS debugging.
Safari has Develop menu with "show error console" etc, Firefox has tools->WebDeveloper->Debugger, Chrome has Tools->Developer Tools.

IT IS REQUIRED THAT YOU TRY EACH EXAMPLE AND DO THE "TODO" SECTIONS.

3. ADDITIONAL RESOURCES

Please read the concepts in below link. And also try examples provided for best practices.

1) This article has a lot of information on internal details of javascript:
http://dmitrysoshnikov.com/ecmascript/javascript-the-core/

2) https://developer.mozilla.org/en-US/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript#Object-oriented_programming

3) JQuery Documentation and Examples:

a) w3schools.com and

b) https://learn.jquery.com/using-jquery-core/ https://learn.jquery.com/about-jquery/

2. JQuery

This is a really simple exercise to have you practice jquery.

Create an HTML file named code.html. Add the following snippet to the html file

```
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js"></script>
<script src="code.js"></script>
<script>
  // page-specific code goes here
</script>
```

Create a file named code.js that demonstrates jQuery features. (Note that you will need to create and add HTML content to the code.html file also so that you can see your demos working). Create a small demo of each of following jQuery features:

1. Manipulate the CSS (i.e. style) of an element with jQuery. Show FIVE different style elements being changed.
2. Use an effect on an element. Show FIVE different jquery effects. http://learn.jquery.com/effects/.
3. Use jQuery events. Show FIVE different events. http://learn.jquery.com/events/

Comment your code.html file so that before each html element with an effect you have a single line of comment.
Example: <!-- Html Element: button id:..effect:…-->

You will need to submit your code.html and code.js files. Attach every required file and remove unrelated files from your submission.

GO TO NEXT PAGE

3. Library

The goal of this exercise is to design and implement a books library using object-oriented Javascript. Use class syntax to develop your objects (look at examples folder to see what we mean). At a minimum, your design should have the three classes Library, Shelf, and Book. Write all the code for these objects in booksLibrary.js file.

1. Assume we have 25 books in the Library.

2. Create random three digit ID for each book. (Note: when librarian adds a new book, they should use a new random three digit ID).

3. There are four categories of books : Art, Science, Sport, literature. you should categorize book based on this categories. The algorithm for categorizing should be as below:

```
If (BookID%4==0)  categorize book as Art

if (BookID%4==1)  categorize book as science

If (BookID%4==2)  categorize book as Sport

if (BookID%4==3)   categorize book as Literature
```

4. Books of different categories should be on different shelves.

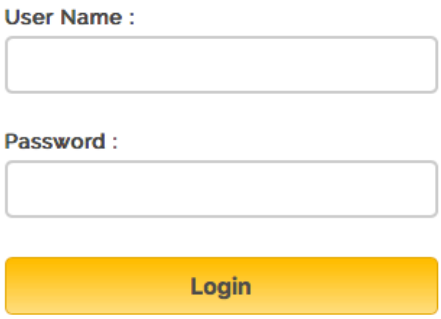
5. Each book should have an attribute borrowedBy which shows the user name of the student who borrowed that book. Initially, none of books are borrowed by students.

6. Each book should have an attribute availability which shows whether a book is available or not. A value of 1 means it is available whereas a value of 0 means it has been borrowed. Initially, all books are available.

7. Use local storage in order to save availability and borrowedBy attributes for each book.

The library System Operations:

1. Login page:



- User Interface: Should consist of Username and password fields and the login button.

- Functionality:
  - If the Username and password are equal to admin, login as a librarian.
  - If the username start with U, login as an Undergraduate student.
  - Otherwise, Show an alert that this is not a correct username and password.

2. Librarian View:

- User interface: Show a visual representation of the library with current shelves and books. HTML table is to be generated dynamically where each column is a shelf (for convenience, we will go with columns for shelves – extra credit if you can do horizontal shelves). Each cell represents a book (and the Book name is displayed).

Art	Science	Sport	Literature
B0	B1	B2	B3
B4	B5	B6	B7
B8	B9	B10	B11
B12	B13	B14	B15
B16	B17	B18	B19
B20	B21	B22	B23
B24			

Book Name | Shelf (literature, science, spo | Add Book

- Functionality:
    - Ability to add specific book to specific shelf (i.e. more than the existing 25 books): Create a text field for book name, shelf category for the book, and the add button. When librarian press add button the book add to the specific shelf. Note that BookID is to be randomly generated as discussed earlier in page 4.
    - Book name can be anything. It is not necessary to follow a specific pattern. Shelf category must be among the given 4 categories.

Art	Science	Sport	Literature
B0	B1	B2	B3
B4	B5	B6	B7
B8	B9	B10	B11
B12	B13	B14	B15
B16	B17	B18	B19
B20	B21	B22	B23
B24			

Shakespeare | Art | Add Book

- Clicking on the cell gives that book's details. Book's detail to be displayed may include book name, book id, and shelf category as shown below.

Art	Science	Sport	Literature
B0	B1	B2	B3
B4	B5	B6	B7
B8	B9	B10	B11
B12	B13	B14	B15
B16	B17	B18	B19
B20	B21	B22	B23
B24			
Shakespeare			

Shakespeare is a(n) Ordinary book on shelf: Art

Book Name | Shelf (literature, science, sp | Add Book

3. Undergraduate View:

1. User Interface: Undergraduate student should have an UI similar to the librarian.

2. Functionality:

    1. Undergraduate student can borrow at most two books each time. So, a student can select on at most two books and change the color of that cell to red (i.e. the book is borrowed, so change the borrowedBy attribute to the student's username).

Art	Science	Sport	Literature
B0	B1	B2	B3
B4	B5	B6	B7
B8	B9	B10	B11
B12	B13	B14	B15
B16	B17	B18	B19
B20	B21	B22	B23
B24			
Shakespeare			

    2. Student should be able to return the book which he/she borrowed. When student return a book the color cell related to that book should be changed back to white (note: availability attribute should be 1 for that book).

Note: Book ID is different from book name (initial book names can be as shown in the above figures).

MAKE SURE TO:

- Write a README.txt file – to describe what you did. Will help in grading.
- Use javascript classes to create objects.
- It should contain minimum three classes Library, Shelf, Book.
- Don't use any global variables or global functions.
- You should decide the members and operations of each class.
- Make sure to have the Library object perform all the required operations described earlier.
- Write client HTML code in a file named booksLibrary.html, that includes the booksLibrary.js file (i.e. <script src="booksLibrary.js"></script>) and uses it (i.e. create library objects and populate them and display them etc).
- **You will need to submit your booksLibrary.js and booksLibrary.html files.**

Submission:

Make sure your solutions work on Chrome (which is what TAs will use to grade the assignment). Your submission must include the following files.

1.code.js and code.html

2.login.html and login.js (if you already include login functionality in booksLibrary(html/js), then there is no need to submit login.js and login.html files)

3.booksLibrary.html and booksLibrary.js

4.Readme.txt file

5. Participation file (i.e. who worked on which part or if you worked together). Participation file is a simple txt file, which clarifies the specific participation of the two members.

Zip your html, js files. Then, submit this zip file on black board. Remember there is only one submission per group. Make sure to include all the files that are needed in order to run your program. DO NOT INCLUDE EXTRANEOUS FILES as they make it harder to grade.