Swinburne University of Technology

COS10011/60004
Creating Web Applications

Assignment Part 2

Develop an Interactive Website

Important Dates

Due Date	**10am on Monday in Week 8** (Late submission penalty: 10% of total available marks per day)
Demonstration	**Your tutorial: Week 8**
Contribution to Final Assessment	**30%**

Note: Do *not* use JavaScript libraries or frameworks (e.g. jQuery, Angular) in the main part of this assignment. You may create an additional alternative implementation using a library/framework as an enhancement (see enhancements section below).

Purpose

In this assignment you will further enhance the website you developed in Assignment Part 1 by using JavaScript to mark the quiz you created. You will:

- Use JavaScript to check data entered into HTML forms and provide user feed back
- Use client-side storage to transfer data between pages.

As in Part 1, there will be an opportunity to enhance your website beyond the basic requirements.

Essential Requirements

In Part 1 of the assignment you created an HTML form **`quiz.html`** that consisted of at least five questions related to your topic. In this part of the assignment we will use JavaScript to mark those questions. JavaScript should be in a file called **`quiz.js`** located in a **`scripts`** folder.

Create a marking scheme for your questions so that each question is allocated one or more marks.

When the user clicks the Submit button on the form use Javascript to:

1. Check they have a selected an answer for any questions where a 'required' attribute was not set in HTML (e.g. check boxes)
2. compare their answers with the correct answers you have defined and calculate their score.

If the user gets zero for the quiz they should not be able to submit their answers. Provide the user with some feedback if this is the case.

Once a positive score is calculated, display the results of the attempt on a webpage called `result.html`. Use HTML5 local storage to do transfer the information between pages.

This web page will display:

- The username and id.
- The score achieved for this attempt.
- The number of attempts the user has made doing the quiz using their browser.
- A hyperlink that allows them to have another attempt at the quiz (only if they have had less than 2 attempts).
- The maximum number of attempts a user (identified by their id) can make is two. You need to implement a way to prevent the user doing the quiz more than twice.

Hint: change the form you created in Part 1 of the Assignment to set the **`method`** attribute to **`result.html`**. While this file is just a static HTML page that does not do any processing of the form data, the web page will be returned from the server to the browser where you can initialize it with the local information you have stored.
We will further enhance this by adding server-side processing in Part 3 of the Assignment.

***Note**: There should be **no** JavaScript embedded in your HTML files. This precludes both event registration (e.g. `<form onsubmit="return validate()" ...`) and function definitions, in the HTML.*

All pages should be styled appropriately using CSS as in Part 1,and should be valid CSS3.

If you wish to make alterations to the HTML and CSS in your Assignment Part 1 that is OK (but you must keep your assigned web topic).
Remember: You need to implement your website in standard HTML5.

Enhancements

You should complete the above requirements before attempting any enhancements.

As with Part 1 you have an opportunity to implement enhancements to your website using techniques not covered in the tutorials. Each enhancement must be described on a page called **`enhancements2.html`**. The entries on this page should:

- briefly describe the interaction required to trigger the event ***and*** what a programmer has to do to implement the feature.
- provide a hyperlink to the page where the enhancement is implemented in your website.
- reference any 3rd party contribution to the enhancement

The JavaScript enhancements themselves should be in a separate **`enhancements.js`** file. Make sure there are adequate comments to explain the enhancement (including its source if applicable).
Examples of JavaScript and other enhancements you might make include (but are not limited to):

- Have your quiz questions written in JavaScript and dynamically display the questions on the quiz page (you can extend this enhancement in Part 3).
- Have multiple versions of quiz questions that are randomly displayed.
- Implement a timer so that the user only has a limited time to complete the quiz .
- *Re-implement* your JavaScript using a library such as jQuery or a framework like AngularJS. Add some enhancements the library/framework provides. **No** library code should be included in your **`quiz.js`** file. This alternative implementation should be in the file **`enhancements.js`** file. Explain the difference in approach using the library and using plain JavaScript.

Smaller enhancements (less than 5 marks) might include:

- Use the JavaScript methods `querySelector()` that take a CSS selector as an argument to manipulate the web page in response to user action.

- Create an extra client side JavaScript dynamic effect: e.g. Slideshow, random image displayed onload, etc. The code and structure of this is open, but must be documented and explained as clearly as possible.
- Use JavaScript to change the Menu display, to reflect the current page being viewed.
- …

Any enhancements that are not listed and linked on the page **`enhancements2.html`** ***and*** implemented in **`enhancements.js`** will not be assessed.

Up to 10 marks will be allocated to each enhancement. A maximum of 2 **enhancements will be assessed.**

Website Folder Structure and Deployment Requirements

Your website folder structure should follow a similar structure as Assignment 1.
All files should be under a folder `/assign2`. JavaScript should sit in an `assign2/scripts` folder.

```
assign2/              You must have this folder – case sensitive!
  index.html
  topic.html
  quiz.html
  result.html
  enhancements.html
  enhancements2.html
  ...other html pages

  scripts/            Folder for your JavaScript
  images/             Folder for images for your page content
  styles/             Folder for style.css other css files
  styles/images/      Folder for images referred to by your css files e.g. background
```

Notes:

- HTML files should only be in the base "assign2/" folder – not anywhere else.
- All links to your files (JavaScript, CSS or images) should be ***relative***. ***Do not use absolute links***, as these links will be broken when files are transferred for marking. No marks will be allocated if links are broken.

Assignment Submission (Canvas + Mercury)

Your assignment should be uploaded to Mercury on or before your deadline.

An electronic copy of your assignment should be submitted through Canvas on or before your deadline.

- Make sure all your files are in the correct folders and compress your root folder with all your sub-folders with HTML, CSS, and images into a zip file named "assign1.zip". Submit this to Canvas. When the zip file is decompressed, the entire website should be able to be run from index.html without needing to move any files.
- You can submit more than once through Canvas. Your last submission will be marked.
- Note that all deliverables must be submitted electronically. There is no need to submit an assignment cover sheet.

Make sure you complete your Canvas submission process.

Assignment Demonstration

1. Make sure you attend your allocated lab. You will demonstrate your assignment to a tutor in your allocated tutorial in week 5, 8 or 12 for Parts 1, 2, and 3 respectively. *You must attend this session to receive a mark for this assignment.* If you cannot attend your allocated tutorial due to illness you must provide a copy of the medical certificate to the convenor.

2. What you demonstrate in the tutorial ***must*** be the same as you submitted to Canvas. The tutor will check the time stamps on your files in Mercury ensure this.

3. **Before** your demonstration starts
   a. Fill in and sign the Declaration on the Marking Sheet your tutor will give you.
   b. Make sure your website is running **on the server** Mercury. (Your marker will check the URL).
   c. Load the website into **Firefox**. All demonstrations will be done on Firefox.
   d. In separate windows display the **source code** for ***all*** your HTML and CSS files.
   e. **Validate *all*** your HTML and CSS files in separate tabs in the browser so they can be quickly checked by your marker. (It might be helpful to load Firefox with the appropriate add-in for validating the HTML and CSS.)

4. **During** the demonstration load *index.html* and your tutor will get you to run through the functionality of the website and check the validations. As you demonstrate your website your tutor ***will ask you to show and explain the source code*** in your browser and explain how you have implemented various aspects of it.

5. **After** the demonstration your tutor will mark your source code. Your code will be checked in detail to ensure that it is your own work and that it meets standard (e.g. no style defined in the HTML).

Mark Sheet

Declaration: Fill this in before you start

I hereby confirm that the assignment to be demonstrated is identical to that I submitted to Canvas.

Student number ……. Student name ……..

Signature …….. Date …….

Essential Requirements

Essential Requirements	Mark

• All questions (e.g. check-boxes) have answers (**5**).
• The score correctly calculated using JavaScript (4 for each question counted = **20**).
• The following information *saved* on submit from **`quiz.html`** in local storage and *displayed* on **`result.html.`**
  ○ The username and id. (**10**)
  ○ The score achieved for this attempt. (**10**)
  ○ The number of attempts the user has had at doing the quiz. (**10**)
• Maximum 2 attempts a user can have at doing the quiz (**10**)
• Quiz result not submitted if zero (**5**)
• A hyperlink on result.html to allow user to have another attempt at the quiz if attempts less than 2. This link is not visible if the user has had two attempts (**10**)
	/80

Subtotal	**/80**

Enhancement Requirements

A maximum of 2 enhancements will be assessed if listed and linked from enhancements2.html. Up to 10 marks are available per feature. Poorly implemented or trivial enhancements may receive less or zero marks.

Enhancement	Described	Linked to implementation on website	Source (if applicable)	Mark
	Y/N	Y/N	Y/N/na	/10
	Y/N	Y/N	Y/N/na	/10
Sub-total				**/20**

Other Deductions based on demonstration, documentation, code, file inspection, etc.

Requirement	Deduction if not met	Deduction
HTML5		
- Valid HTML5	-10 (per web page)	
- Meta-data follows in-house standard	-5	
- HTML has no Style information embedded	- 5	
- HTML form elements follow in-house standard	-5	
- No deprecated elements/attributes used	-5	
- Comments adequate	-5	
JavaScript		
- All JavaScript is in an external file	- 10	
- "use strict" directive present	-5	
- No 3rd party libraries/frameworks used in quiz.js	-20	
- Header comments as per in-house standard	-5	
- Line comments as appropriate	-5	
Website		
- Directory Structure as required	-10	
- All third-party content acknowledged properly*	up to -100	
- Other deductions	up to -100	
Total Deductions		

*** Note: Failure to acknowledge third party code or content at all is plagiarism and may result in zero marks for this assessment or other penalties in accordance with Swinburne policy.**

A final assignment mark will not be provided during the demonstration. All code is inspected after the demonstration by your tutor before a final mark is allocated.