

## **PROJECT 2:**

### **Student as Teacher!:**

### **Creating an Online Quiz**

CSE100/INFO100 Fluency with Information Technology

#### **Table of Contents**

Introduction .....	1
Objectives .....	2
IMPORTANT!!! .....	2
Turn-in procedure:.....	3
Grading Criteria.....	3
Part A: Creating the Interface .....	4
Part B: Initial Quiz Logic .....	5
Part C: Decision-making for scoring multiple answer questions .....	6
Part D: Arrays (lists) for questions with more than one correct answer .....	7
Project Inventory .....	8
Please Recall the Class Policy on Cooperation and Collaboration.....	8

## **Introduction**

Every time you use the WWW, you probably encounter pages that ask you for personal information (name and address), or respond to questions you ask (search engines), or even allow you to practice your skills in some area by taking an online quiz. All of this interaction is done through the use of HTML combined with logic that is implemented using a variety of programming languages.

The ultimate goal of Project 2 is to create an interactive quiz using JavaScript and HTML. To achieve this goal, however, we will be dividing the work up into several distinct parts. Then at the end, each part will be put together to create a single working quiz program! You all will become the teachers for this project and over the course of the next 4 weeks, develop a series of quiz questions related to FIT 100 that the entire class can use to prep for the final! By now, you should be comfortable with the HTML component that makes up the interface design

aspect of the quiz. Use your skills and understanding of proper design to provide good instructions and valid metaphors to help people use your exam and to make it aesthetically pleasing.

In order for an online quiz to be interactive and independent (not requiring your constant attention), you need to provide the program with enough instructions that the computer can “correct” the quiz for the user and alert them to incorrect answers. This means giving explicit and unambiguous instructions on how to determine correct answers from incorrect ones without your involvement at every turn. The logic that allows the computer to “understand” is at the heart of programming. We will use JavaScript as the language for providing the instructions. This allows us to continue working with the web page environment and HTML while adding some new concepts that deal with logic and computer-user interaction.

The questions used for your quiz should be based off the reading material (the book or articles) or lecture material from our course. Make sure you test the quiz for correct answers as well as correct code so that it is useful to your classmates at the end of the quarter.

## Objectives

This project and its various pieces will require you to incorporate all programming concepts covered in FIT 100. By the end of this project you should be able:

- To combine HTML and JavaScript to add logic and interaction to a simple web page
- To identify elements and objects on a web page and use basic JavaScript to manipulate them
- To use the concept of variables, assignment, expressions and values to store questions, answers and scores for your quiz.
- To use the concept of branching (conditionals) to allow your program to make decisions based on user input.
- To use iteration as a way to search through lists (arrays) for matching terms
- To write functions that will perform certain instructions when triggered by certain events

## IMPORTANT!!!

The Web site should be in your Dante web space (public\_html or student\_html), but should be in a separate directory (folder) within that web area. Create a new directory within your public\_html. **Choose any directory name you want, but make sure**

**your project 2 files, parts A-D, are placed in it and the absolute path indicated to your TA on the paper copies you submit. *DO NOT* create any files with names that begin with "index", such as index.html, index.htm, etc in either of the folders.** Make sure the directory and the files in it all have general Unix Read permissions (you will know if they do not when you try to look at your web page on line and receive a "permission denied" message).

Once the due date has come and gone, do NOT alter your files again in any way. Files with a modification date later than the due date will be considered late, and your freebie will be used. Files that are modified more than 24 hours after deadline will not be graded. Any changes you make between the due date and receiving your grade for that project piece will eliminate any position you may have to question the grade.

#### ***Turn-in procedure:***

1. The deadline for making changes to your initial site is **11 PM** on the calendar date given
2. **After this time, make no changes whatsoever to your page**, or it will be graded as late. Changes made more than 24 hours later will mean your site is NOT GRADED.
3. Browse to your web site using Internet Explorer or Netscape Navigator. Print it off, from the web browser. This will confirm to YOU that your files are inside the public\_html directory in your Dante account.
4. From the browser, print out the HTML source for the site. ( Go to **View > Source** to see the HTML )
5. Staple all pages together, write your name and section on the first page, and bring them to the next lecture (Friday). Late papers will not be accepted.

#### ***Grading Criteria***

Project pieces will be graded based on 3 general areas:

- How well the Web Interface instructs users and the completeness of you HTML
- The functionality of the JavaScript code for that piece (does it work?)
- The completeness of the comments for each code section.

Each area will be worth roughly the same amount of points, so encountering problems just with one area and not the other two will cause you to have a lower grade, yes, but it should not cause you to fail if the other 2 are strong.

## Part A: Creating the Interface

In this part we will be creating the user interface and the foundation for our program. To create this foundation you should:

1. Create a webpage called project2a.html.
2. Create a form with 8 text input boxes and a submit button.
3. Create a question to go with each text box that is based on FIT 100 content. These questions should be answerable using one word or a number. Hint: Use a series of underscores to create a blank space. For example:
  - $1 + \_ = 2$  or
  - Ethernet is a technique for \_\_\_\_\_ area networking.

---

1. Ethernet is a technique for \_\_\_\_\_ area networking.

---

4. Your webpage should include a last modified date. Make sure to put text around it so that it is clear what the date refers to. A date alone on a page does not help your viewer understand what that date means.
5. Use JavaScript to change the background of your form. Use the appropriate object and property.
6. Comment your code. For each line of JavaScript that you write, write comments that explain what your code is doing. Feel free to explain blocks of code all at once. However, make sure that each line of your code is explained in detail. Include not only what the code does, but how it is doing it. If you do not include comments at every step in this process, you will be marked down!

**Part A Due Date:** October 31<sup>st</sup>, 11 PM. Paper copies due in class the next day.

**NOTE: Look to your labs if you are unsure how to begin any of the steps required in a project piece.**

## Part B: Initial Quiz Logic

In this part of the project we will be building on the form that you built in the first part. In this part we will be checking the answers from your user. However, do not modify the original file, project2a.html. Make a copy of the file, rename it project2b.html and add the code in these directions to that file.

1. Establish a variable to hold the score of the person taking the quiz.
2. Create a function that checks to see if the answers are correct. Realize that to check the answer placed in any text box, you will need to refer to the text box by name and look at its value.
  - If an answer is correct, add 1 to the persons score. Do this for each question.
3. In your form HTML tag, add an "onSubmit" attribute that calls the function you have just created.
4. If the score that the person gets is above a 6, change the background to one color otherwise make it turn a different color. (We leave the color choice to you!)
5. Create two web pages. One should display, "You Won!" Name this page won.html. The other should display, "Try Again!" Name this page lose.html. There does not need to be any additional content on these pages, unless you are feeling creative!
6. Find two images of the same size and create a mouse-over event. Because clicking on this linked image will call a function, link the image to project2b.html.
  - Make sure that the images are in the public domain. The easiest way to do this is to create your own image with a program such as Photoshop or Microsoft Paint.
7. Create a function that will check the score to see if the score is over a 7. If the score is a 7 or higher, have won.html open in a new window. If the score is lower then a 7, have lose.html open in a new window. Call this function when the linked image is clicked.
8. Comment your code as you did for Part A. Remember, commenting code shows you understand the logic and placement of your code. It can show you understand the concept, even if you are having problems with the implementation. Code without comments will be marked off quite a bit.

**Part B Due Date:** November 7<sup>th</sup>, 11 PM. Paper copies due in class the next day.

## Part C: Decision-making for scoring multiple answer questions

In this part of the project we will be creating a new webpage. So far you have project2a.html, project2b.html, lose.html and won.html. You are going to use this new webpage to test a new part of your project. We are going to be adding a question that has multiple correct answers and we will be using a series of checkboxes to do so.

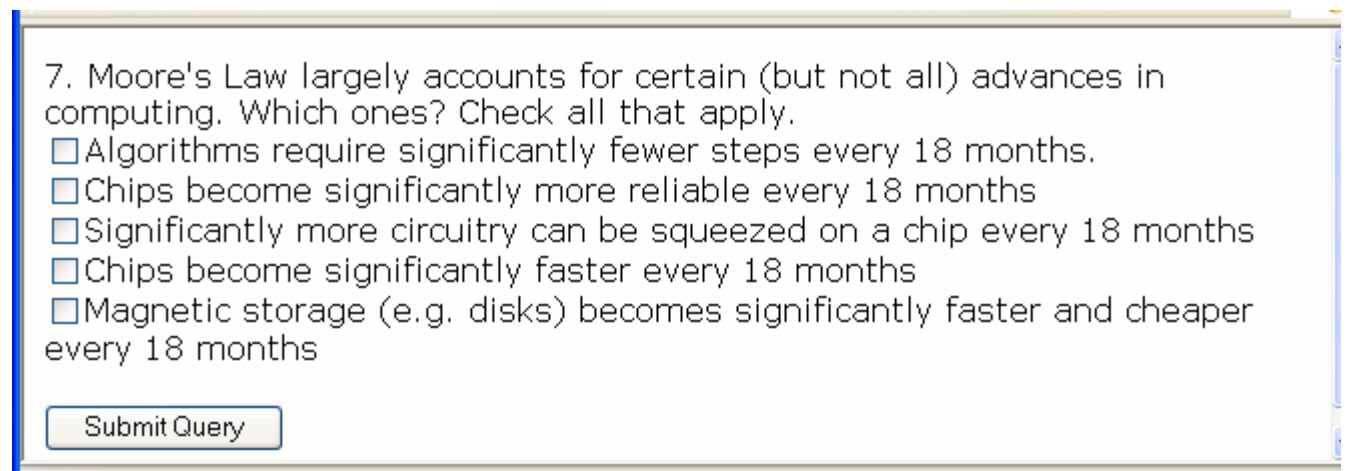
1. Create a new webpage called partc.html. (Make sure to add the head and the body, as well as the respective closing tags.)
2. Create a new form with a submit button.
3. Make up a question that has more than one answer. (Make sure that in order to get the question correct you must select more than one answer, but less than all of the answers.) For example:

Which of the following are valid HTML tags? (check all that apply):

- <html>
- <wordwrap>
- <h1>
- <p>

4. Establish a checkbox for each answer. Remember to name each input (Check Boxes, Text Boxes, etc) on your form.
5. Write a function that checks to make sure that the correct boxes are checked when you click the submit button. If they are, add 1 to a variable that will hold the score. **Use the same variable name that you used in project2b.html.**
6. Print the score to the webpage after the submit button is pushed.
7. Comment your code as you did for Parts A and B. Code without comments is marked down.

### Sample question with multiple correct answers 1



7. Moore's Law largely accounts for certain (but not all) advances in computing. Which ones? Check all that apply.

- Algorithms require significantly fewer steps every 18 months.
- Chips become significantly more reliable every 18 months
- Significantly more circuitry can be squeezed on a chip every 18 months
- Chips become significantly faster every 18 months
- Magnetic storage (e.g. disks) becomes significantly faster and cheaper every 18 months

Submit Query

**Part C Due Date:** November 14<sup>th</sup>, 11 PM. Paper copies due in class the next day.

## Part D: Arrays (lists) for questions with more than one correct answer

In this part of the project you will be doing two things: creating the final question for your quiz and then, once that is complete, putting all of the project parts together. The final question will be one that has more than one possible correct answer. You will create this question on a separate page, just as you did for Part C. After creating this item, you will integrate Part C and Part D into your final project.

1. Create a new webpage called partd.html. (Make sure to add the head and the body, as well as the respective closing tags.)
2. Create a new form with a submit button.
3. Make up a question that has more than one answer. This question should have at least three answers. For example, "Name one of the planets in our solar system" would have nine possible correct answers.
4. Create a text box for just as you did for the first 8 questions.
5. Create an array that will hold the possible answers for this question.
6. Create an iteration that will compare the answer that is entered in the text box to the items in your array when the submit button is clicked. If there is a match, add 1 to a variable to hold the score. **Use the same variable you used in project2b.html.**
7. Print the score to the webpage after the submit button is clicked.
8. Create a copy of your project2b.html page. Rename is **project2final.html**.
9. Add partc.html and partd.html to your project2final.html project page. You should remove the aspects of part C and D that ask you to print the score to the webpage.
10. Add a "close window" button to your win.html and lose.html files. Use the appropriate object and property.
11. Comment your code as you did for Parts A, B, and C. Code without comments is marked down.

**Part D Due Date:** November 21<sup>st</sup>, 11 PM. Paper copies due in class the next day.

# Project Inventory

When you complete part D, you should have the following:

1. **project2a.html**
  - 8 text box questions with one word (or number) answers.
  - A Submit button.
  - A last modified date.
2. **Project2b.html**
  - All the content from the Project2a.html file AND
  - A background that changes color depending on the final score.
  - An image that changes upon mouse over.
  - A conditional that opens a specific window when the linked image is clicked, depending on the score.
3. **win.html**
  - A header saying, "You Win!"
  - A button that will close the window when clicked.
4. **lose.html**
  - A header saying, "Try Again!"
  - A button that will close the window when clicked.
5. **partc.html**
  - A working multiple answer question with checkboxes.
  - A score that is printed when the submit button is clicked.
6. **partd.html**
  - A working multiple answer question with a text box.
  - A score that is printed when the submit button is clicked.
7. **Project2final.html**
  - All the content from the Project2b.html file, partC.html AND
  - An array that holds the possible answers for your question from Part D.
  - An iteration that checks the answer for your Part D question against the array.
  - 1 checkbox question to multiple answers.
  - 1 text box question with multiple answers.

## Please Recall the Class Policy on Cooperation and Collaboration

It is valuable to work with a friend or classmate when learning a new application or working out a problem. However, the work that you perform in FIT100 for a grade must be your own work unless "working in groups" is explicitly allowed. The Projects involved in FIT 100 are NOT intended to be group projects. It is OK to run ideas and scenarios past your friends or classmates, but the solution you create should be based on what is inside your own head. Talk to your classmates about how to approach the work, but do not let them do it for you. If you run into problems, talk to your instructor or your TA.