Documenting Labs

Labs

Labs are more involved; therefore they require detailed documentation in all files. When you turn in each Lab, you'll want to make sure that it contains all the necessary code and files in **one zip file** named with your Lab number and classID (e.g., **Lab1bjones4242.zip**).

Please note that all lab deliverables will include the following:

- Source Code
- Screenshots of Program Output (if requested)
- Documentation
- Other Files (as needed by a specific lab, such as images)

All Deliverables

The compressed file **must be in zip file format (not .rar, .7z, etc.).**
Deliverables should also include your Name, classID, Exercise/Lab Number, Date, and Explanation(s) in the header block **in all files.**

General Information on the Lab sections

Program Accuracy (60%)

Your program will be submitted in electronic form to me. Your program will be evaluated with the following **potential scores**:

- Program runs with no validation errors: **full credit** (also takes into account the use of proper structures).
- Program runs but errors occur: **points off for each error** (amount depends on severity of error).
- Program does not run or display (syntax errors): **points off for each syntax error** (amount depends on severity of error).

Programs that do not run or display as expected ***will receive extra consideration*** if the problems are well documented and you provide detailed descriptions, as well as the code used, of your attempts in order to provide me with insight into the problem and how you tried to solve it.

Poor design, logic, or script layout (to include code presentation) will result in the loss of points.

Report and/or Screen Design (20%)

This will be graded objectively (spelling and grammar) as well as subjectively (easy to read and follow, labels, user friendliness, theme, layout, etc.).

Remember, design is an essential component for this class given the nature of what we are doing.

Internal Documentation (20%)

A complete description of the program should be commented at the top of **each** program file along with all pertinent personal information. **This is the header block.**

If a lab has multiple files, each file requires a header block with specific information about to the file's functionality in relation to the program as a whole.

In JavaScript programs, each variable should be documented with an explanation of what that variable is used for. HTML tags should be discussed in terms of choice, context, etc. CSS needs to use self-documenting classes and IDs, as well as include comments.

Documentation should be used throughout the program to provide the instructor with insight as to what is happening at that point. Major modules should be discussed, as well as logic (in the case of JavaScript).

It is also critical to discuss your design choices to include layout, themes, etc.

Cryptic variable names, missing documentation, etc., will result in loss of points.

If you do not document your files, the lab will not be evaluated.

Important Instructions for the Labs and Exercises

1. Each file should begin with a comment that contains your name, classID, Lab# (or Exercise#), course number, semester, due date, and date completed. You should also include a description of the program.

Do make sure to place your DOCTYPE declaration before the header block or there may be issues.

An example **header block** is shown below:

```
<!DOCTYPE html>

<!--

****************************************

    *  Programmer: Bubba Jones
    *  Class ID: bjones4242
    *  Lab #5: HTML5 for World Domination
    *  CIS 3900: Business Web Architecture
    *  Fall 2016
    *  Due date: 12/09/16
    *  Date completed: 12/07/16
****************************************
```

```
    * Program Explanation
    * Here you provide a detailed explanation of the program.
    * Explain its purpose, what it does, and how it does it.
**************************************
-->
```

Make sure to use the correct documentation style for each language:

```
<!-- HTML5 Comment -->

/*

CSS3/JavaScript Comment

*/
```

2. Program comments should be provided for each program, user- defined function, variable declaration, selection, code block, loop, design choice, etc.

3. Use proper indentation in all scripting. Check your textbook for the style of code indentation.

4. When you need to create a variable, create custom tags, use CSS classes and IDs, etc., they all should have a meaningful name. Do not define any single-letter variable like **a, b, c, x, y, z,** etc. but descriptive names such as **studentCounter**, **averageSalary,** and **.navigationHeader** are good ones.

5. In JavaScript (and CSS classes and IDs), always write a comment for each at the time of declaration. Always use **camel casing** (e.g., camelCasing) for variables and wherever else necessary (e.g., **.leftImagePosition).**

6. Non-obvious code and script should be explained and commented. However, do not over comment your code or script.

7. Always backup your files and programs before turning in any assignment. **Save a dated copy for your records.**

8. Failure to comply with the above will result in a loss of points.