COMP2068 – Advanced Web Programming

Final Project
The JavaScript Arcade Game

Due class #14 (Friday December 12, 2014) @ midnight.

Value 30%

The JavaScript Arcade Game **Maximum Mark: 66**

Overview: Either alone or as a Team of 2, you will create an original 2D arcade game. The game can be a scrolling game, a platform game, tower defense, ¾ down adventure or other classic arcade game. Puzzle games will not be accepted. The game must have a **start screen**, the **main game screen**, and a **game-end screen**. A **scoring system** must also be included. The game will include at least **3 difficulty levels**. You must use your own graphic and sound assets. You may choose a Web API such as CreateJS to create your interface.

IMPORTANT: Please keep in mind that the **Final Presentation** (which will be used in place of a final exam and is worth 10% of your final mark) is dependent on this project working and being completed on time.

Instructions :
(25 Marks: GUI, 25 Marks: Functionality, 5 Marks: Internal Documentation, 11 Marks: External Documentation)

1. Your application will have the following characteristics **(25 Marks: GUI, 25 Marks Functionality)**
    a. A **Start Screen (**the Game Start State**)** – will allow the user to get ready and displaying rules and instructions on how to win the game (2 Marks: GUI, 2 Marks: Functionality)
    b. A **Gameplay screen** (the Game Play State) where the main game occurs. (2 Marks: GUI, 2 Marks: Functionality)
    c. A **Game-End screen** (the Game End State) – this will display the player's final score and give the player the option to play again (2 Marks: GUI, 2 Marks: Functionality)
    d. Player control of an **Avatar** (a vehicle or character) – the main input may be a combination of mouse and keyboard clicks. The player's avatar may have **weapons** or other **devices** that he can use to defeat the computer controlled enemies (3 Marks: GUI, 3 Marks: Functionality).
    e. Computer control (AI) of the **enemies**. The enemies should be abundant enough to challenge the player but not be impossible to beat. (3 Marks: GUI, 3 Marks: Functionality)
    f. The game will have **3 levels of difficulty** – examples of this would be "Easy-Medium-Hard", several Game-Level screens or increasingly challenging computer AI. **Do not**

simply make the game "faster" to make it more difficult (6 Marks: GUI, 6 Marks: Functionality).

g. Random opportunities to generate points for the player aside from killing enemies (2 Marks: GUI, 2 Marks: Functionality)
h. A **Scoring system** – ensure that the player's score is accurately calculated and displayed somewhere on the **Gameplay screen** (1 Mark: GUI, 1 Mark: Functionality).
i. The player must have a **life counter** or **health status** that decreases each time his **avatar** is "killed" (1 Mark: GUI, 1 Mark: Functionality)
j. Add **sound effects** for collisions with enemies, collecting points, shooting attacks, explosions, etc. (2 Marks: GUI, 2 Mark: Functionality).
k. Add a **Game soundtrack** (1 Marks: GUI, 1 Mark: Functionality).

2. Include **Internal Documentation** for your program **(5 Marks: Internal Documentation):**
   a. Ensure you include a program header for each module of your game that indicates: the Source file name, Author's name, Last Modified by, Date last Modified, Program description, Revision History (2 Marks: Internal Documentation).
   b. Ensure you include a headers for all of your functions and classes (1 Marks: Internal Documentation
   c. Ensure your program uses contextual variable names that help make the program human-readable (1 Marks: Internal Documentation).
   d. Ensure you include inline comments that describe elements of your GUI Design for your arcade game (1 Marks: Internal Documentation)
3. Include **External Documentation** for your program that includes **(11 Marks: External Documentation)**:
   a. **A company Logo** (0.5 Marks: External Documentation).
   b. **Table of Contents** (0.5 Marks: External Documentation).
   c. **Version History** – ensure you include details for each version of your code (1 Mark: External Documentation).
   d. **Detailed Game Description** – describing how your game works (1 Mark: External Documentation).
   e. **Controls** (0.5 Mark: External Documentation).
   f. **Interface Sketch** – this section should include wireframes of each of your game screens with appropriate labels (1.5 Marks: External Documentation)
   g. **Screen Descriptions** – Include at least 6 screen shots for your game: 1 for your Start Screen, 1 for your Gameplay Screen, 1 for your Game-End Screen and 1 for each level of difficulty (2 Marks: External Documentation).
   h. **Game World** – Describe your game environment (0.5 Mark: External Documentation).
   i. **Levels** – Describe each of your game levels or challenge levels (0.5 Mark: External Documentation).
   j. **Characters / Vehicles** – Describe the character's Avatar (0.5 Mark: External Documentation).
   k. **Enemies** – Describe the computer-controlled enemies and how they function (0.5 Mark: External Documentation).

l. **Weapons** – Describe any weapons available to the player (0.5 Mark: External Documentation).
m. **Scoring** – Describe how the player can score and how the score is calculated (0.5 Mark: External Documentation).
n. **Sound Index** – Include an index of all your sound clips (0.5 Mark: External Documentation).
o. **Art / Multimedia Index** – Include examples of your image assets. Each image should be displayed as a thumbnail (0.5 Mark: External Documentation).

Optional Game Features (i.e. Potential Bonus Marks).

A. Include a final "boss monster" to defeat.
B. Add power-ups for the player's **avatar** (e.g. extra speed, a shield) that he can add to his "inventory" and use whenever he chooses.
C. Add Cheat Codes.
D. Create a mini-game in a section of the main arcade game (e.g. disarm a bomb, pick a lock, etc.).
E. Empower the player to gain an NPC (non-player character) computer-controlled ally.
F. Make it possible for the player to save / load his game.

SUBMITTING YOUR WORK

Your submission should include:

1. An external document (MS Word or PDF).
2. A zip archive of your Project files or a link to your Project on GitHub (preferred).

Program Code & Functionality

Technical Evaluation

Display / User Interface	The Display / User Interface elements meet the program requirements. All text is spelled correctly and appropriate space is allocated for user input. Graphics & Icons are appropriate and match the program's functions.	25
Functionality	The program's deliverables are all met and the program functions as it should. No errors appear as a result of execution. User Input does not crash the program.	25
Internal Documentation & Readability	A program header is present and includes the name of the program, the name of the student, a short revision history and a short description of the program. All procedures and classes include headers that describe their functionality and scope. Inline comments are used to indicate their function when code is new or unclear. Variable names are contextual wherever possible.	5
External Documentation	An external document (MS Word or PDF) has been created that includes a company logo, table of contents, version history, detailed program description, a sketch of the GUI and screenshot (if applicable), and other details outlined in the template provided.	11

Creative Evaluation		**Mark**
Creativity	The program's GUI / UI is attractive. The programmer has added additional elements outside of the scope of the program that enhance functionality, usability and fun.	0
	Total (/66)	66
	%	100.0%

This assignment is weighted **30%** of your total mark for this course.

Late submissions:

- 10% deducted for each additional day.

External code (e.g. from the internet or other sources) can be used for student submissions within the following parameters:

1. The code source (i.e. where you got the code and who wrote it) must be cited in your internal documentation.
2. It encompasses a maximum of 10% of your code (any more will be considered cheating).
3. You must understand any code you use and include documentation (comments) around the code that explains its function.
4. You must get written approval from me via email.