F28WP Coursework - Edinburgh Campus

Mark: **50%** of the overall mark of the course
Due Date: **25th November at 15:30**, Local Time (Week 11)

Overview

Key Points:

- Develop Multiplayer Online **2D** Game (MMOG)
- **Team** size 3-5
- **GitHub** (create manage project) - every team member must show contributions weekly (e.g., commits, tasks, bug fixes on GitHub) - standard industry practice (see GitHub vuejs/vue, d3d, angularjs - view insights, which includes, contributors, commits, code frequency, dependency graph, ..)
- The project must be **`YOUR' code** (i.e., avoid libraries, show your coding abilities, management, iteratively tuning, testing, web programming features, ..)
- Coursework offers **creative freedom** and room for innovation and improvement
- Demonstrate real world working practices
- Sufficiently complex system/implementation
- Manage security, data, **server-client**, connections, ...
- HTML5/CSS/Javascript Compliant (e.g., W3C Markup Validation Service)
- **Animated** and dynamic (e.g., can't be a static board game), utilize latest CSS/JavaScript (synchronized across network)
- Readme, outline start, goals and objectives, flexible, agile/iterative working,

Details

A develop an online multiplayer game that that lets players interact and play other people in real-time. Store and view information (e.g., high-scores, number of online active players). Play other people or join other games. On interaction should be communicated to a web server and recorded in server side storage (e.g. MongoDB, MySQL or files), as well as being displayed back to the user side (e.g., leaderboards, other player interaction). The front-end should also let a user list all or some game information that is available (e.g., player's past scores and other where they sit in the global high scores).

The assignment should be in a group pairs (3-5). Students on the Edinburgh campus who can't find a partner may contact

Benjamin Kenwright (Edinburgh) by e-mail (**b.kenwright@hw.ac.uk**) for help in doing so.

The following Web technologies should be used:

- HTML 5 (XHTML syntax), CSS 3, DOM 1+ and JavaScript
- Client and server side technologies and Database (or server storage system)

The project will examine web technologies and algorithms to allow you to develop a complex web-based system (emphasis on web development best practices).

Objectives

- Design and implementation web-based `multiplayer' game
- Theory and practice of server-side scripting and database for web games

- Code modulation in JavaScript
- Techniques to implement best practices in web development such as Test-Driven - Development, Version Control and JavaScript documentation API
- Minimum Viable Product (MVP) and testing

This coursework requires you to spend your time programming and doing practical development work. The coursework aims to develop a web-based skills through a complex web-based solution (interactive online multiplayer game) combined with the techniques covered in lectures and labs.

- Design and develop a sufficiently complex web-based game using advance web techniques such as design pattern and modularity
- Efficiently manage and document the code using appropriate software tools and techniques
- Incorporate back-end services into the game for asset management and game-state management
- Use appropriate web technologies to comply with best practices

Development environment

Access to a web server for online publication, access to a software version control system (GIT), access to a limited range of web services (e.g. a high-score service, game-state management services)

Date Due

The coursework for Edinburgh students is due in week 10 (show):

1. Version control (regular checking, show team work, ..)
2. Team work (distribution of workload, tasks, ..)
3. How you manage security (e.g., cheating)?
4. Reliable connection? (e.g., delays, intermittent spikes, ..)
5. Scalability (5 or 5000 connections/players)
6. Store and display the game statistics to players (e.g., number of live players,)
7. Complexity of the software
8. Testing (bugs, user interaction, software reliability, …)
9. Browser compatibility (Chrome, IE, Mobile, Desktop, …)

Criteria

- 2D Game (i.e., NOT 3D)
- Interactive and animated (move around/explore, can't be a static/board game)
- Store separate player profiles/histories/details (securely)
- Communicate data/information without reloading the browser

Submission

2 Page Report, include:

- Name, Student No
- List all other team members (percentage of work each member did)
- Youtube URL (1-2 Max Trailer):

- GitHub (Repo) URL (GitHub usernames and name for each member)
- Live Pages URL

Marking Criteria

Grid View | List View

	No Feature	Attempt	Working	Excellent
HTML, CSS and Javascript	0 (0.00%)	1 (4.7619%)	2 (9.5238%)	3 (14.28571%)
Client/Server	0 (0.00%)	1 (4.7619%)	2 (9.5238%)	3 (14.28571%)
Database/Security	0 (0.00%)	1 (4.7619%)	2 (9.5238%)	3 (14.28571%)
Code Structure	0 (0.00%)	1 (4.7619%)	2 (9.5238%)	3 (14.28571%)
Usability	0 (0.00%)	1 (4.7619%)	2 (9.5238%)	3 (14.28571%)
Development Progress	0 (0.00%)	1 (4.7619%)	2 (9.5238%)	3 (14.28571%)
Testing	0 (0.00%)	1 (4.7619%)	2 (9.5238%)	3 (14.28571%)

Details for each of the criteria

Demonstrate Understanding of HTML, CSS and Javascript

HTML, CSS and Javascript validates fully (no syntax errors, warning, issues)
Creative use of CSS styling
Advanced CSS3 layout
Creative use of Javascript
HTML5 semantic elements used appropriately
Game/site should be responsive (also responsive in a mobile/different browsers)
Hand written code (clean, structured, commented, ..) *Your Code*

Client/Server

Manage communication between multiple concurrent client/server
Handle issues (delays, network issues, corruption,)

Database/Security

Data is managed for the website effectively and efficiently (Server/Database)

Code Structure

Clear separate of code
Tidy code organisation, file and folder names
Appropriate code nesting and indentation
Informative code comments

Usability

Easy to navigate and use (Website/Game)
Clear awareness of accessibility principles
Effective navigation at all sizes, content easy to locate
Limitations and optimizations

Development Progress

Shows even work distribution (group work)

Evidence of task development, tuning, feature refinement
Live site (Github and webpage)
Project management

Testing

Testing integrated into the project from the start
Validation/verification process for ensuring reliability (e.g., code, standards, ..)