Course Assignment 3 in WEB1 – **Deadline can found on itslearning.**

A Browser Game

For your final project, you will design and build a browser-based game using JavaScript. This project is your chance to get creative, write a lot of JavaScript, and apply everything you've learned about the DOM, events, and local storage.

You are free to choose the type of game you want to build. The most important thing is that it's built using the technologies you've learned. Here's some inspiration:

- **Choose your own adventure** - A story game with choices that affect the outcome.
- **Quiz game** - Multiple-choice trivia game with score tracking.
- **Math trainer** - Random math problems with tracking of correct answers and speed.
- **Whack-a-mole** - Click moles quickly as they pop up to score points.
- **Typing speed test** - Type quickly and accurately while measuring speed and errors.
- **Simon says** - Repeat a growing sequence of colors and sounds.
- **Memory card game** - Players flip cards to find matching pairs.
- **Hangman** - A word guessing game where players guess letters to reveal a hidden word.
- **Clicker/idle game** - Click buttons to earn points and buy upgrades that boost progress.

Requirements

- The website must include a short description of the game and how to play it.
- The game must be based on the DOM API - that is, no Canvas or WebGL.
- The game must use localStorage to save game progress, scores, or settings.
- The game must display correctly and be fully playable on a desktop with a resolution of 1920×1080. There are no other requirements for responsiveness.

Animations are not a requirement, but they can bring your game to life. You can explore and use CSS animations and/or manually animate elements with JavaScript to add effects like blinking, fading, or moving elements, if you want an extra challenge!

External libraries, frameworks and game engines are not allowed.

Tips

- The projects will be discussed at the exam. Focus on applying what you've learned to make it easier to explain and showcase your skills. A more interesting project might lead to a more interesting discussion.
- Keep it simple! The DOM is not designed for fast-paced or graphics-heavy games, so focus on mechanics that rely on simple interaction and logic rather than constant motion. Ask, if you are unsure about the scope of your game idea!
- Don't jump straight into code. Plan your UI and game flow before implementing it.
- Start with a minimal viable product (MVP). What is the player doing most of the time? (guessing, reacting, choosing, etc.) Build that first, and then expand around it with UI, polish and extra game mechanics.
- Big problems needs to be broken down. Analyse the game that you want to make. What systems does it consist of? (UI, game logic, score board, persistence…) Can these systems be broken down further? Can you divide the work so everyone can develop their part without constantly depending on each other?
- Keep your code modular. Use functions for each piece of logic (e.g., checkAnswer(), updateScore(), resetGame()), so it's easier to understand and explain.
- Many game ideas require timing or repetition. Use setTimeout() to schedule something once after a delay, and setInterval() to repeat actions at fixed intervals (with can be stopped with clearInterval()).
- Consider collaborating directly through your GitHub repository, and/or use the Live Share plugin in VS Code to code together in real time.

Submission

This is a group project to be completed with your existing SEP group. The deadline can be found on itslearning. Hand in before the deadline if you want feedback.

You must host the game using GitHub Pages and **submit the link to the site** using the handin slot on itslearning.

Have fun ☺