Milestone 2 - Group 102-5

Project Features List

- Physics Engine
    - Provides an approximate simulation of physical phenomenon (in this case gravity, velocity, acceleration, normal force). Allows in game characters to jump, land, run etc. in a realistic manner.
- Game Mechanics
    - Working hand-in-hand with the physics engine, the mechanics make up how the game operates and what the user needs to do to interact with the game itself.
- Art Style/Graphics
    - Dynamic graphics that change as the player transverses, and interacts with their environment. e.g jumping, running and picking up items animation.
- Levels
    - Levels will be either in a sequential order or have branches to create a web of levels that the user can interact with. Levels help keep the attention span of the user
- Leaderboard (and what it involves)
    - Keeps track of, and ranks high scores of players
- Music/Sounds
    - Governs how music will be integrated into the gameplay experience, and how sounds will be implemented (and activated) upon interaction with environments.
- Items/Power Ups
    - Gives the user the opportunity to interact with items that are placed on the map to enhance their movement speed, extend their time window or even give them an extra life

Requirements

- Physics Engine
    - Functional:
        - The characters on screen will move consistent to player inputs with little to no latency.
    - Non-Functional:
        - We will be developing the physics engine with unity, In which we have the ability to
- Game Mechanics
    - Functional:
        - Need to be seamless, and easy for the user to interact with, learn, and master.
            - However, create a steep mastery curve
    - Non-Functional:

        - Needs to work hand-in-hand with the physics engine to ensure that nothing crashes the game or makes the physics break.
        - We need to code everything correctly so that the mechanics are working correctly based on how far someone is in the game, what the current game state is, etc.
- Art Style/Graphics
    - Functional:
        - Style must be consistent throughout the map or level. The game must have smooth transitions between movement animations (i.e. Jumping to running, running to standing)
    - Non-Functional:
        - Animation files must be allocated properly and created with .json file type to integrate properly into a c# script for implementation
- Levels
    - Functional:
        - When a level is finished, moves on to the next appropriate level.
    - Non-Functional:
        - Some function controls the linear (or branching) sequence of levels
- Leaderboard
    - Functional:
        - Displays highscores in an organized list
    - Non-Functional:
        - Archives, retrieves, and sorts high score and name information from a database
- Items/Power Ups
    - Functional:
        - These items and power ups will need to have specific and noticeable attributes that the user can easily identify
    - Non-Functional:
        - Make sure that everything works with the physics engine, animation, and make sure that data within powerups is organized correctly
- Music/Sounds
    - Functional:
        - Each environment will have its own music or sound loop. Environments (i.e. enemies, hazards, interactable features) should also play sounds in order to make the player feel more immersed in the world and to provide immediate auditory feedback.
    - Non-Functional:

        - Game code needs to be able to retrieve sound files efficiently and appropriately. Sound should be made using a specialized software like Logic.

Project Plan

- We will be using Trello as our Project Plan environment. This will allow us to effectively collaborate.
- On Trello, we have “To Do”, “Doing”, and “Done”. Which will help our group identify the sequence of sprints.
- **We invited Chelsea to our Trello site via email (from Peter LeCavalier)**