**Game Design**

Performance Task

# **Create — Game Mechanics**

## **Overview**

In this performance task, you will program an arcade-style Sumo battle with the objective of knocking increasingly difficult waves of enemies off of a floating island, using power ups to help defeat them. In creating this prototype, you will learn how to implement new gameplay mechanics into your projects, which are new rules or systems that make the game more interesting to play. On one hand, you will learn to program a powerup, which gives the player a temporary advantage. On the other hand, you will learn to program increasingly difficult enemy waves, which make survival more challenging for the player. A good balance of power ups and increasing difficulty make for a much more interesting gameplay experience.

## **Assessment**

You will be provided with 16 hours of class time to complete and submit the following:

* A video of your game being planned or a trailer for the game
* Written responses about your game and design process
* Digital Game Prototype

Your teacher will share submission guidelines that include suggestions for creating video and PDF files.

## **General Requirements**

You are required to:

* Analyze their own code and provide solutions to errors identified.
* Explain the characteristics of a nonlinear story
* Compare conflict and outcomes
* Build empathy maps that explore the range of different user perspectives and needs. Brainstorm solutions that will meet the needs of their users.
* Compare processes of creating interior versus exterior environments
* Compare level design of linear games to open world games
* Compare design considerations for environmental, assets, characters and User Interface (UI) creation
* Describe good quality assurance practices
* Produce a game pitch document
* Research various styles of game documentation
* Research victory condition mechanics of a game
* Differentiate control structures and modify program with boolean expression
* Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.
* Provide the definition of a data structure and its different functions (e.g., organizing, processing, retrieving, and storing data).
* Compare the use of constants and variables
* Differentiate between syntax and semantics errors
* Write responses to questions about your game.
* Include your entire prototype.

## **Submission Requirements**

### 1. **Video**

Submit one video in .mp4, .wmv, .avi, or .mov format that demonstrates the running of your game. Your video must not exceed 1 minute in length and must not exceed 30MB in size.

### 2**. Written Responses**

Submit one PDF document in which you respond directly to each prompt. Clearly label your responses. Your response to all prompts combined must not exceed 950 words.

You may use images to show off or to explain things in your written responses.

## **Purpose and Development**

1. Provide a written response or audio narration in your video that:

Identifies the changes you made to your prototype. Identifies the changes.

* Explains what the video illustrates.

(Approximately 150 words)

1. Explain the characteristics of a nonlinear story. Compare conflict and outcomes in the terms of digital game play. Build empathy maps that explore the range of different user perspectives and needs and how that affects game play. Brainstorm solutions that will meet the needs of their users. Describe good quality assurance practices.

(Approximately 200 words)

1. Compare processes of creating interior versus exterior environments. Compare level design of linear games to open world games. Compare design considerations for environmental, assets, characters and User Interface (UI) creation.

(Approximately 200 words)

1. Differentiate control structures and modify programs with boolean expressions. Provide the definition of a data structure and its different functions. Compare the use of constants and variables. Explain how events and handlers function in conjunction with one another along with why validation is important to program design. Differentiate between syntax and semantics errors.

(Approximately 200 words)

1. Research various styles of game documentation, and select a style which suits your needs. Research victory condition mechanics of a game and select one for the different ideas you currently have written down.

(Approximately 200 words)

## Tasks

### **Activity 1 - Explore**

**Description**

students learn about scene design and develop their own unique scene using a collection of polygon prefab assets.

* Building a Scene with Prefabs

Time To Complete: 2-3 Hours

### **Activity 2 - Research**

**Description**

Research environments similar to what you require for your game idea. Plan and design a level based on the selected environment. You may complete multiple environments if you have time to do so. You should research player character models for your game as well, look for something appropriate for your game.

* Design your Game Level(s)

Time To Complete: 5-6 Hours

### **Activity 3 - Ideate**

**Description**

Students develop working prototypes of their games, have peers test them and provide feedback, and complete the final games! This is an open project by design – we want you to be creative and build something you’re excited about! At the very minimum, your game needs to have the items below, but don’t hold back your creative mind:

**REQUIREMENTS**

* A simple game design document that includes a description of your game and how it meets the four good game elements (what you planned and chose).
* Demonstration of object and prefab transformation (position, rotation, and scale).
* Use of materials and colors.
* Player interaction.
* Planning Your Game Part 2

Time To Complete: 2-3 Hours

### **Activity 4 - Evaluate**

**Description**

Students will self evaluate their game idea, and then peer review the idea looking for input and helpful feedback to finalize the idea before building the prototype.

Time To Complete: 1-2 Hours

### **Activity 5 - Construct a Prototype**

**Description**

First things first, we will create a new prototype and download the starter files! You’ll notice a beautiful island, sky, and particle effects... all of which can be customized! Next you will allow the player to rotate the camera around the island in a perfect radius, providing a glorious view of the scene. The player will be represented by a sphere, wrapped in a detailed texture of your choice. Finally you will add force to the player, allowing them to move forwards or backwards in the direction of the camera.

The player can roll around to its heart’s content… but it has no purpose. In this lesson, we fill that purpose by creating an enemy to challenge the player! First we will give the enemy a texture of your choice, then give it the ability to bounce the player away... potentially knocking them off the cliff. Lastly, we will let the enemy chase the player around the island and spawn in random positions.

The enemy chases the player around the island, but the player needs a better way to defend themselves... especially if we add more enemies. In this lesson, we’re going to create a powerup that gives the player a temporary strength boost, shoving away enemies that come into contact! The powerup will spawn in a random position on the island, and highlight the player with an indicator when it is picked up. The powerup indicator and the powerup itself will be represented by stylish game assets of your choice.

We have all the makings of a great game; A player that rolls around and rotates the camera, a powerup that grants super strength, and an enemy that chases the player until the bitter end. In this lesson we will wrap things up by putting these pieces together!

First we will enhance the enemy spawn manager, allowing it to spawn multiple enemies and increase their number every time a wave is defeated. Lastly we will spawn the powerup with every wave, giving the player a chance to fight back against the ever-increasing horde of enemies.

* Watch where You’re Going
* Follow the Player
* PowerUp and CountDown
* For-Loops for Waves
* Challenge 4

Time To Complete: 6-8 Hours

### **Activity 6 - Improve the Design**

**Description**

Once the prototype is built the students will now test the games. Students should play 2-3 games and provide useful feedback to the creator. Students will then take their feedback and plan how to improve their game, play test it once more and get some final feedback.

* Extras

Time To Complete: 2-3 Hours

### **Activity 7 - Share**

**Description**

Students will share their feedback, prototype and whole design process with the class in a formal presentation, or gallery walk.

Time To Complete: 1-2 Hours

### **Activity 8 - Reflect**

**Description**

Students will reflect on their design process and how they went about creating their game, what struggles they encounter and how what worked and didn’t work for them. They will submit a video of game play, along with their written reflection.

Time To Complete: 1 Hour