Introduction to Computer Science

# Project 2: Platform Game

*Adapted from Forest Ridge High School's Mario Project.*

Students will implement a side-scrolling platform game (a la Super Mario Bros.) in Snap!.

## Overview

Platform games are among the most widely recognized types of video games. Composing about one third of all console games at the peak of their popularity, platform games are characterized by their relative simplicity and by the common gameplay element of jumping across suspended platforms (hence the name) to avoid falling into a hazard. Platform games also typically include enemy characters, items that grant the hero special abilities ("power-ups"), and a"checkpoint" system that allows the player to restart from partway through a game or level when he or she dies.

## Details

1. **Controllable Sprite**
   Your program should have a "Mario" sprite. It does not necessarily need to look like Mario (a simple stick figure will suffice). Mario should respond to user input. Specifically:
   - Mario should clearly “face” the right when you push the right arrow key
   - Mario should clearly “face” the left when you push the left arrow key
   - Mario should perform an animated in-place walk when you hold left or right arrow key
   - Mario should jump based on some input (you can decide the key or mouse click)
2. **Moving Scenery Sprites**
   You should have scenery sprites that move based upon Mario’s traveling on the level. It is up to you to decide the level scenery, but you should meet the following requirements:
   - There should be at least two scenery sprites (Example: A mountain and a tree)
   - You should layer these sprites relative to Mario and each other. For example, Mario should always be “in front” of any background scenery sprites
   - Scenery sprites move relative to Mario as he moves. For example, when you hold down the right arrow key, the background sprites should move from right to left in the stage.
   - Following with the layers, scenery sprites should move at different speeds so that one seems farther away. For example, a faraway mountain should move more slowly than a nearby tree.
   - Scenery sprites should roll over/re-appear when they hit the edge of the stage. For example, when Mario is walking to the right, the scenery Sprites should re-appear on the right side of the stage when they roll off the left.

3. **On-Ground Enemy**
   There should be an on-ground enemy for Mario to contend with. Specific criteria here include:

   - There is at least one on-ground enemy
   - The enemy sprite moves towards Mario, independent of whether Mario is moving (e.g. regardless of whether the user is pressing an arrow key).
   - The enemy sprite reappears/rolls-over when he hits the edge of the stage
   - The enemy sprite is animated when he is moving towards Mario
   - If Mario does not jump he runs into the enemy and the game ends with an appropriate message (HINT: you can use the "STOP ALL" block to end all scripts)
   - It should be possible for Mario to jump over the enemy.

## Programming Habits

We will again look for you to incorporate good programming habits in your code:

- Using Start and Stop blocks.
- Making sure you initialize state appropriately so that your program is repeatable.
- Add comments to your code so it is easy to understand.
- Always keep in mind all Good Programming Skills you've been taught

## Additional Extensions

Once you complete the above, you can extend your program. Some suggestions:

- Include flying enemies for Mario to dodge or duck
- Keep score based on how many objects Mario gets by [Hint: Use a variable and show it on the screen]
- Have Mario jump to 'grab' an object that offers Mario extra points or more powerful abilities (such as jumping higher or not being killed when he runs into an enemy). The objects must appear at random times and move smoothly as Mario runs

**We will demo the most interesting projects in class, so be creative!**

## Grading Criteria

The detailed list for what we will use to grade your projects is below. Please review your projects before submitting to be sure you meet all of them. If you have any questions, please ask us!!

| **Requirement** | **Value** |
|---|---|
| Mario turns to the right on right arrow key | 5 |
| Mario turns to the left on left arrow key | 5 |
| Mario performs an in-place animated walk if you hold down either arrow key | 5 |
| Mario jumps based on some input | 5 |
| There are least two scenery sprites | 5 |
| Scenery sprites are layered with Mario (e.g. appropriate layering blocks are in your program) | 5 |
| Scenery sprites move based on Mario's movement | 5 |
| Scenery sprites move at different speeds (e.g. far away versus near) | 5 |
| Scenery sprites roll over when the fall of the stage | 5 |
| There is at least one on-ground enemy | 5 |
| Enemy sprite always moves towards Mario | 5 |
| Enemy sprite re-appears/rolls over correctly | 5 |
| Enemy sprite is animated when it moves | 5 |
| If Mario does not jump, he runs into the enemy and the game ends nicely and properly | 10 |
| Mario can jump over the enemy | 5 |
| *Good programming #1:* Program has clear start and stop | 5 |
| *Good programming #2:* Program is repeatable and initializes state correctly | 5 |
| *Good programming #3:* Use of comments in your code | 5 |
| *Good programming #4:* General skills you've been taught | 5 |
| *Good programming #5:* Custom blocks used to break down program into logical parts | 5 |
| **Extra Credit:** Include flying enemies for Mario to dodge or duck | 5 |
| **Extra Credit:** Keep score based on how many objects Mario gets by | 5 |
| **Extra Credit:** Have Mario jump to 'grab' an object that offers Mario extra points or more powerful abilities. The objects must appear at random times and move smoothly as Mario runs | 10 |
| **Extra Credit:** Create Your Own Extension | 0-10 |

| Requirement | Value |
| --- | --- |
| **TOTAL POINTS** | **105** (135) |