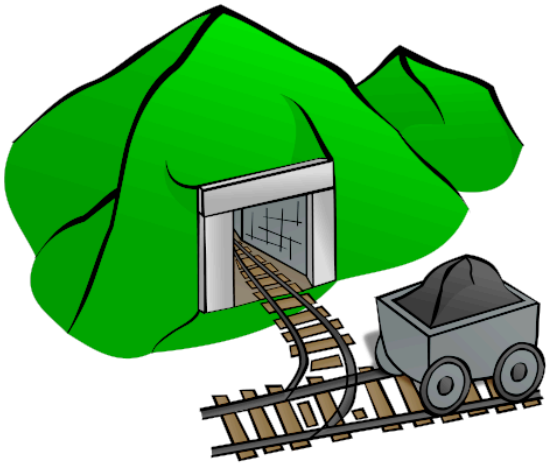
Mining



Part 1 Due before class on **9/18**
Part 2 Due before class on **9/25**
Part 3 Due before class on **10/2**
Part 4 Due before class on **10/11**
Final Due before class on **10/18**

For this project, you will create a
simulation of a mining operation that produces bronze, silver, and gold ore.

This project is intended to help you increase your skills in the following areas:

- Resourcefulness and Persistence
  - Searching for answers online, using bookmarks for commonly used webpages
  - Asking for help from Ira, the TAs, or other students when you're stuck
- Logical Thinking
  - Carefully thinking through step-by-step logic for the program
- Using Unity
  - Creating new projects, files, and assets
  - Attaching scripts to objects
- Using GitHub
  - Storing and sharing your program as it changes over time
- Publishing a game to your own website
  - How to create a basic website and the tools required
  - FTP, basic text editor, Unity
- Organization
  - Well-organized files and folders
  - Well-named files, projects, assets, and variables
  - Nicely formatted code with matching {} and indentation
  - Comments to help understand the code
- Types of variables
  - int, float, bool
  - How to create them and modify them
- Mathematical expressions
  - Comfortable with adding, subtracting, multiplying and dividing
- Conditional expressions
  - if(), else if(), else()

Wow, that's a lot! It's amazing what you'll be able to learn in just a few weeks of effort!

Since many of you are new programmers, it's very likely that you will be confused or get stuck along the way. Please remember that is a normal and natural part of learning! There are many resources at your disposal: the internet, your peers, your TA, and your professor. As you get more practice at programming, you will improve!

What To Hand In
For each part of this project, **deliver your work as follows**:

a) Within Unity, make a WebGL build (File -> Build Settings)
b) Upload the .html file, Build folder, and TemplateData folder to your website.
c) Upload your project files (especially .cs) to GitHub.
d) Find your tab on the spreadsheet below and add your website and GitHub links for the appropriate part of the project:

https://docs.google.com/spreadsheets/d/1mLWBSuBEUG6xb08D3WJqiw0eRso2Kqe37OwpDhCJAf8

We'll cover how to do all these things in class, and soon they will be second nature!

Part 1
By the deadline, accomplish the following tasks:

1) Create a free GitHub account. (http://github.com)
2) Add to your current project to GitHub, using an appropriate .gitignore

Modify your project to work as follows:

1) When the game starts, the player should see a blank scene.
2) After 3 seconds, print "It's been three seconds!" to the console.

If you complete this part before the deadline, go on to the next part.

Part 2
By the deadline, create a game that functions as follows:

1) When the game starts, the player should see a blank scene. The game should know the player has 0 bronze and 0 silver ore to start.
2) Every `miningSpeed` seconds, the player will mine ore, then the game should show how many ore the player has. For example, at the start of the game:
   Bronze: 0
   Silver: 0
3) There are `bronzeSupply` bronze ore and `silverSupply` silver ore available to be mined.
4) The player should mine all of the bronze first, then all of the silver.
5) Once all available ore is mined, the game should continue showing ore totals every `miningSpeed` seconds (but obviously no new ore will be added).

Use the values below for the variables mentioned, but they should be easily changeable in the Inspector:

```
miningSpeed = 3
bronzeSupply = 3
silverSupply = 2
```

Example: After the first 3 seconds, the player should have 1 bronze and 0 silver. After 6 seconds, the player should have 2 bronze and 0 silver.

In your code or in a separate document, describe any parts of the game design (i.e. instructions above) that were unclear, and how you decided to resolve that uncertainty yourself.

If you complete this part before the deadline, go on to the next part.

Part 3
By the deadline, modify your existing game as follows:
In addition to printing the player's ore totals, also create cubes on screen to represent the ore. Use red cubes for bronze and white cubes for silver. Note: It should be obvious to the player that more ore is appearing over time, so don't put the new bronze ore in the same place as existing ore, and ensure the new ore appears someplace the player can see it.

Part 4
By the deadline, create a game that functions the same as the previous part, with the following additions:

1) Instead of what's described previously, use the following algorithm to spawn the ore:
    a. Mine an ore every `miningSpeed` seconds.
    b. If there are <4 bronze ore, spawn a bronze.
    c. If there are 4 or more bronze, spawn a silver.
    d. If there are exactly 2 bronze and 2 silver, spawn a gold (yellow cube) instead (but just spawn 1 gold and then go back to the normal rules.)

In your code or in a separate document, describe any parts of the game design (i.e. instructions above) that were unclear, and how you decided to resolve that uncertainty yourself.

Final
By the deadline, create a game that functions the same as the previous part, with the following additions:

1) When the player clicks on an ore, it should disappear (and be removed from their total).
2) Whenever the player clicks an ore, give the player points as follows:
    a. `bronzePoints` for a bronze
    b. `silverPoints` for a silver
    c. `goldPoints` for a gold

Use the values below for the variables mentioned, but they should be easily changeable in the Inspector:

```
bronzePoints = 1
silverPoints = 10
goldPoints = 100
```

In your code or in a separate document, describe any parts of the game design (i.e. instructions above) that were unclear, and how you decided to resolve that uncertainty yourself.

Challenge by Choice: Do any or all of the following additional tasks. I've included categories below, but you can pick and choose however you want.

Feedback to the Player

- Whenever players mouse-over an ore (but before clicking on it), give a visual indication that the ore is clickable (change the color a bit, enlarge it slightly, etc. Your choice of what to do, but do something.)
- Somewhere on the screen, display a count-up timer that shows how long the game has been going.
    - If showing the game timer, make it fancy: After 59 seconds, show 1:00 instead of 60. After 60 minutes, show 1:00:00 instead of 60:00.
- Somewhere on screen, display the player's current score.
    - Along with the player's current score, show the word "Score."
    - At the moment the player earns points, display the points earned somewhere (+10!).

Sound Effects

- Whenever the player clicks an ore, play a sound effect (SFX).
- Whenever the player clicks an ore, play a different SFX depending on which type of ore it is.
- Play music that loops, so there's constantly music playing during the game.

Camera Control

- If there are too many ore to fit on screen, automatically zoom out the camera.
    - If doing the automatic zoom out, also do an automatic zoom in once there are few enough ore on screen.

Game Logic

- Add kryptonite ore (green) which only appears under these circumstances:
    - There can never be more than 2 kryptonite at once; only 0, 1 or 2.
    - If there are the same number of silver and gold ore, spawn a kryptonite.
    - When determining if there is the same number of silver and gold, kryptonite can be used a wildcard.
        - For example, if there were 2 silver, 1 gold, and 1 kryptonite, you would spawn a kryptonite (because the existing kryptonite could count as a gold, and then you have 2 silver and 2 gold, which meets the conditions of spawning a kryptonite.)
    - Kryptonite should be worth `kryptonitePoints`, initially set to `1000`.

Player Input

- Instead of automatically controlling the camera, let the player control the camera instead. Either zoom in and zoom out or pan left and right.
- Allow the player to modify `miningSpeed` while the game is going, perhaps by pressing the up and down keys, or whatever mechanism you choose.

Visual Polish

- Once an ore is mined, have it move or rotate slightly.
- Make the ore look prettier, instead of just being a simple colored cube. You can change the shape, texture, etc. The goal is to make it pretty. Feel free to add a background to the game, too.

Original assignment designed by Professor Ira Fay. Inspiration from Stone Librande, Sam Goodspeed, Sabrina Ko, and Sienna Cornish.