# A02: Enemies and Spawners

Programming for Commericial Engines

# Assignment goal

1. Add the Mole enemy to the game;
2. Add the PiPi enemy to the game;
3. Extend Spawner so you can optionally specify spawn regions;

# TaskS

## Grading

*Use the checkbox to mark which tasks you completed; If it is not marked, I will not attempt to grade it. If it is marked, but not attempted, it will be double penalized (ie, if a task is worth 10 and you marked it as complete, but did not attempt it, you will not only not get the 10 points, but will suffer an additional 10-point penalty).*

*As an example, you can either X the row, highlight it, or both.*

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Sample Category** | | **7** |
| A0 |  | Task Not Attempted | 5 |
| A1 | **X** | Task Attempted | 5 |

## Checklist

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CHECKLIST** | | | | |
| **Notes** | | | **POINTS** | |
|  | **General** | | | **20** |
| A0 | **X** | FaceDirectionOfMovement component | | 10 |
| A1 | **X** | Spawner supports spawn regions | | 10 |
|  | **Mole** | | | **30** |
| B0 | **X** | Mole Implemented to Spec (see below) | | 20 |
| B1 | **X** | Break World into Background, Stage, and Foreground Layers (Mole appears behind stage, in front of background) | | 10 |
|  | **Pipi & Copipi** | | | **35** |
| C0 | **X** | Pipi enemy itself. | | 5 |
| C1 | **X** | PipiEgg enemy which Pipi can drop; | | 3 |
| C2 | **X** | Egg behaviour | | 7 |
| C3 | **X** | Copipi behaviour | | 20 |
|  | **Extras** | | | **15** |
| X01.00 | **X** | Spawner picks regions weighted by area of the region (see below notes) | | 5 |
| X01.01 |  | SpawnOncoming | | 5 |
| X01.02 | **X** | Copipi picks a new direction if they cause damage. | |  |
| **Total** | | | | 100 |

# Assigment Details

## FaceDirection

Component, when added to an enemy, should flip a transform to face the direction it is moving; You must also implement GetFacing public member of it;

Should be located: Scripts/Components/FaceDirectionOfMovement.cs

// This script can assume sprites are authored to face "right"

class FaceDirectionOfMovement : MonoBehaviour

{

// if not set - assume it flips itself;

public GameObject objectToFlip;

// Returns 1.0f if facing right, -1.0f if facing left.

public float GetFacing()

{

// ToDo

return 1.0f;

}

// declare any other methods, events, or members you need

// to make this work;   
}

## Spawner supports regions

The Spawner script should be extended to include a list of optional spawning regions. If these regions exist, when picking a spawn location, it will pick randomly within one of these regions. If no regions are specified, it uses the spawner’s location like current behaviour.

Recommend using *List<Bounds>* objects in Unity to specify these regions – ignoring the Z dimension when picking spawn positions;

These regions should be rendered to the user when the Spawner is selected; Use the event *OnDrawGizmosSelected*, and you can use *OnDrawGizmos* as an example on how to use the Gizmos functions. Use DrawWireCube to render the regions.

### Extra: Weight region selection by area

When selecting a region to spawn from – select weighted by area. That is, if there are two areas, and the first area has twice the area as the first area, it should be twice as likely to be picked;

## Mole Enemy

*Do not worry about Animations for this assignment. Use single frame sprites. Animation is usually not done during a prototype phase anyway;*

Create a sprite from the existing sheet – the Mole sprite is: Mm2molesprite

A mole is an enemy that will spawn, and move either up or down depending on the player’s position. This direction is picked at spawn time, and is constant for the lifetime of the enemy; For example, if the player is below the Mole when it spawns, the Mole will move downward.

On contact, should cause 4 damage;

It should take 3 hits to kill.

It should move slower when in contact with the ground. Due to Tilemap collision properties, it is fine for this to only move slower while in contact with an edge, as it moving faster while hidden behind Terrain actually gets it off the screen faster). You can use any speeds you want (as long as they’re visually distinct). 2 (in air) and 1 (in ground) is a fine starting point;

Should spawn a *small\_explosion* effect on death. (*Prefabs/Effects/small\_explosion*)

Should play an enemy damage sound when damaged.

Setup a few regions that spawn this enemy to showcase it;

## Pipi & Copipi

This enemy is actually three different enemies;

Pipi – the bird; Mm2pippisprite  
The egg – the egg he is carrying (separate entity)  
Copipi – The tiny bird; Mm2feathersprite

For reference, see this gameplay video (let me know if the link is broken!)  
<https://www.youtube.com/watch?v=2R3MIZCfxmE&t=70>

The Pipis show up around 1:15.

### pipi

Pipi is simple. She spawns, moves either left or right depending on player’s location, and despawns off screen. Like Mole, her movement speed is picked at spawn time, and is constant for her lifetime;

She has one health.

She does 4 contact damage;

She plays a *small\_explosion* on death.

She spawns with an Egg which she can drop if the player is within a certain **x**-distance from her.

### Egg

The Egg has one hit point.

It causes 4 contact damage.

If killed by a bullet, spawns a *small\_explosion*.

If released, should fall straight toward the ground, otherwise moves with Pipi.

If contacts the ground, destroys itself, spawning a number of Copipis. This number should be editable from the Inspector.

### Copipi

Copipi has one hitpoint, and will spawn the *small\_explosion* prefab on death.

Copipi does 2 contact damage.

Copipi has two stages to its behaviour (making it a good candidate for a specialty controller).

At first, will move from its spawn location in a random direction for a random length. This length min and max should be editable from inspector.

Once they reach this point, they then make a beeline toward Mega Man™’s center. Once this direction is picked, it is constant – meaning they do not home in on him. Again see video for reference.

#### Copipi Direction change

Copipis, if they are the one to cause damage, will pick a random new direction to travel. Supporting this behaviour will likely require code changes to existing components – for example, the ColliderOnDamage and Health (ColliderOnDamage to report damage was done, and Health to report that damage was applied).

## Spawn oncoming

Spawner was written to constantly spawn enemies. Good for testing, but not a common feature in most Mega Man™ games.

The more common behaviour is to spawn an enemy once Megaman reaches a certain point in the level. We will call this the *SpawnOncoming* spawner.

This spawner works that once Mega Man crosses a threshold, an enemy is spawned as long as there is no other enemy on screen associated with this spawner (see Spawner for one way to track this).

The spawn location is separate from the location of the SpawnOncoming object. It should be specified using additional empty transforms or regions.

The spawn location can be influenced by the direction Mega Man is facing when crossing the threshold, so you may want two lists. For example, Pipi in Air Man’s stage will spawn from the left side of the screen if you’re moving Right to Left at the same location in the video. If you have a copy of the game.

The SpawnOncoming’s location should dictate *where Mega Man has to be*for the spawn to occur. How you hand this trigger is up to you. However you do it though, be sure that it is visualized in the Scene view using Gizmos.

# Additional notes

Make use of what Unity or I offer out the gate. For example, I have a component that will tell you if you’re touching the ground that I use for detecting when airborn – this can be re-used by both the Egg and Mole.

Also the Mathf and Random libraries Unity supplies are full of useful methods.

## Additional Reading

**Unity3D Tutorials: Prefabs - Concept & Usage**[https://unity3d.com/learn/tutorials/topics/interface-essentials/prefabs-concept-usage (Links to an external site.)Links to an external site.](https://unity3d.com/learn/tutorials/topics/interface-essentials/prefabs-concept-usage)  
*Entire series is actually a good introduction of common Unity concepts.  Prefabs are of particular import to this assignment.*

**Unity3D Tutorials: 2D Game Creation**  
[https://unity3d.com/learn/tutorials/s/2d-game-creation (Links to an external site.)Links to an external site.](https://unity3d.com/learn/tutorials/s/2d-game-creation)  
*Collection of tutorials that more or less cover the material in class.  The first tutorial (2D Game Development Walkthrough) does a good job walking through code and explaining it in English*

**Unity3D Tutorials: The Sprite Editor**  
<https://www.youtube.com/watch?v=gbgIA3pwpHc>*Video explaining how to use the Sprite Editor;  Needed for adding new sprites for these enemies;*