Spawning Enemies Concepts

Spawning Enemies Concepts

Spawning Functionality

1. We need to define which enemies should be spawned in different dungeon rooms.

2. We also need to allow for variation in the spawning to create some randomness.

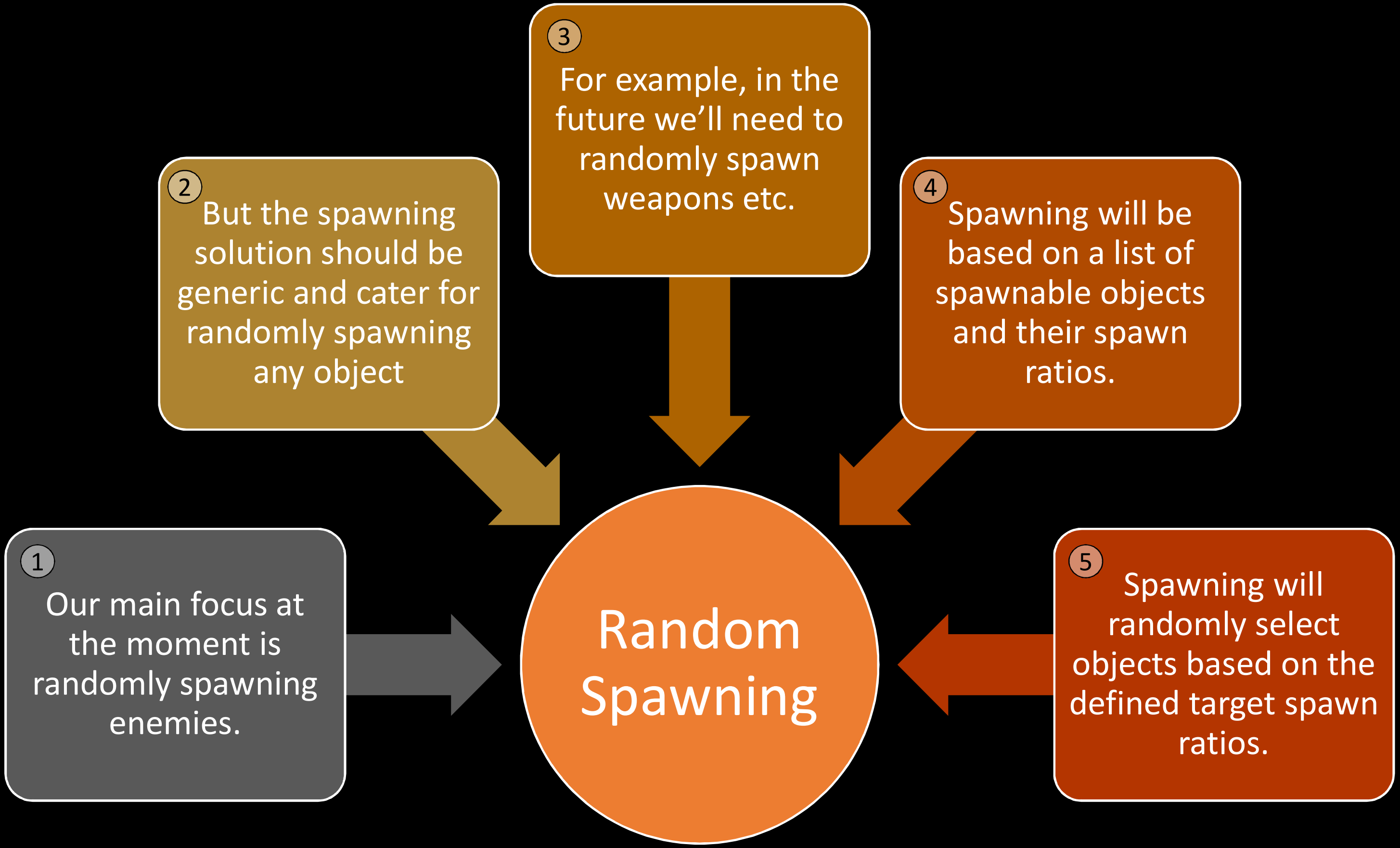
3. We'll define enemy spawning for each Dungeon Room Template.

4. Each template will define which enemies are to be spawned for different Levels and the spawn parameters.

5. A key concept for Rogue Like games is 'Randomness' – both for dungeon generation and for enemy spawning.

Spawning Enemies Concepts

Random Spawning



1 Our main focus at the moment is randomly spawning enemies.

2 But the spawning solution should be generic and cater for randomly spawning any object

3 For example, in the future we'll need to randomly spawn weapons etc.

4 Spawning will be based on a list of spawnable objects and their spawn ratios.

5 Spawning will randomly select objects based on the defined target spawn ratios.

Spawning Enemies Concepts

Enemy Spawning Example

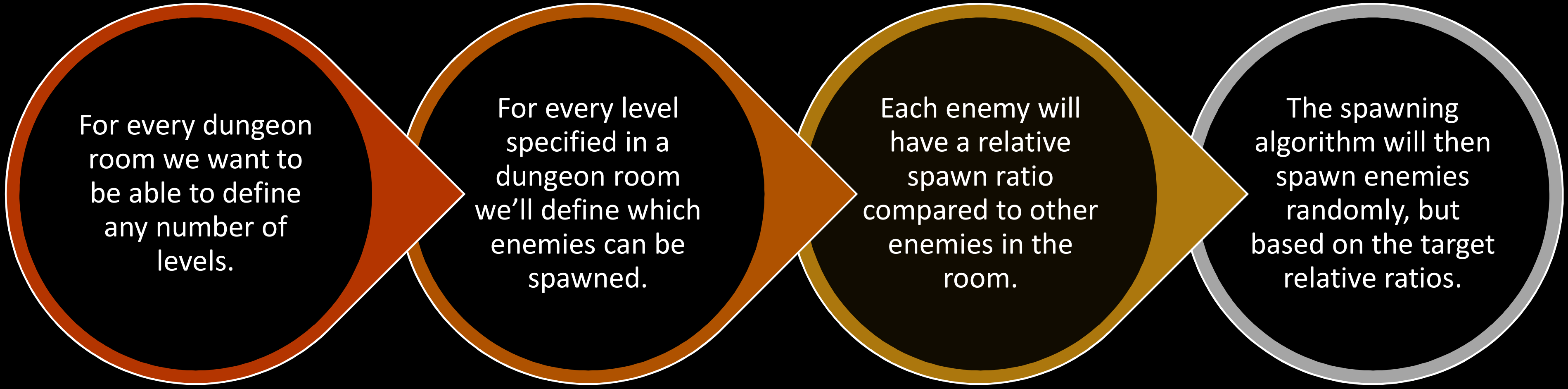
Catacombs Large Room 1

Level 1		Level 2		Level 3	
Enemy	Ratio	Enemy	Ratio	Enemy	Ratio
	10		5		5
	10		5		6
			5		6
			5		3
					3
					3
					6

The absolute value of the ratios doesn't matter, it's their value relative to the other enemy ratio values that's important (a ratio of 1:1 is the same as 10:10)

Spawning Enemies Concepts

Defining Enemy Types And Spawning Ratios



For every dungeon room we want to be able to define any number of levels.

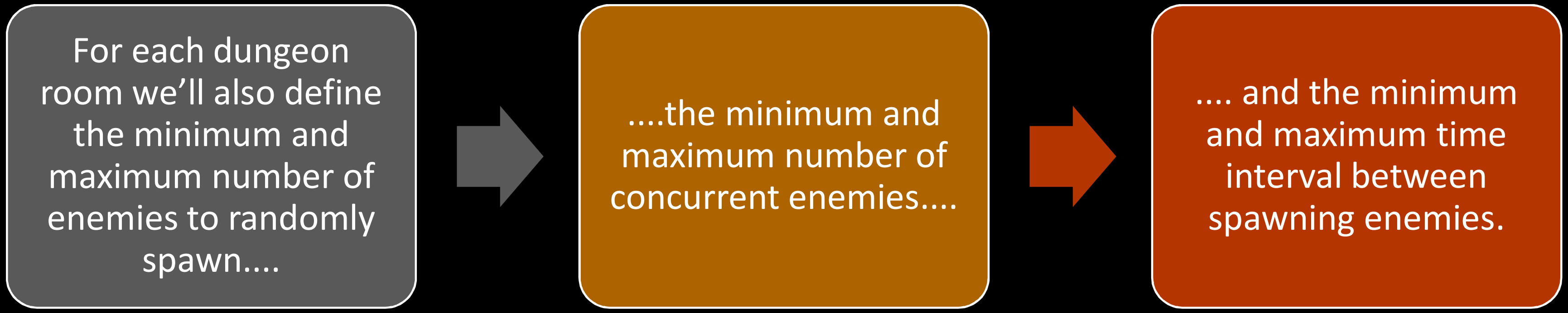
For every level specified in a dungeon room we'll define which enemies can be spawned.

Each enemy will have a relative spawn ratio compared to other enemies in the room.

The spawning algorithm will then spawn enemies randomly, but based on the target relative ratios.

Spawning Enemies Concepts

Enemy Number, Concurrent Enemies And Spawn Interval



For each dungeon room we'll also define the minimum and maximum number of enemies to randomly spawn....

....the minimum and maximum number of concurrent enemies....

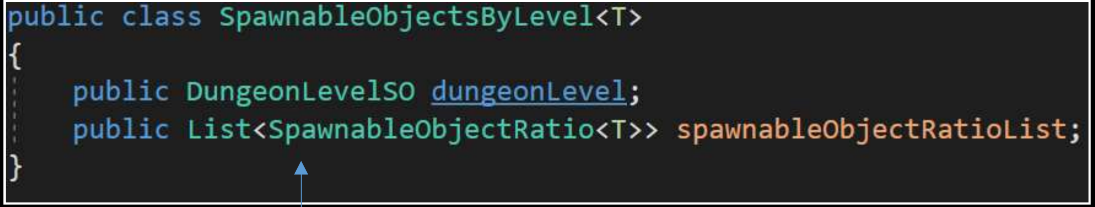
.... and the minimum and maximum time interval between spawning enemies.

Spawning Enemies Concepts

Generic Spawnable Objects By Level

2

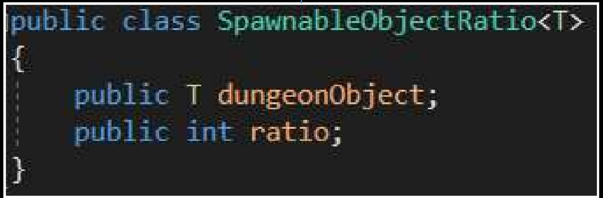
The SpawnableObjectsByLevel generic class can be used to define a list of SpawnableObjectRatio instances for a given dungeon level.



```
public class SpawnableObjectsByLevel<T>
{
    public DungeonLevelSO dungeonLevel;
    public List<SpawnableObjectRatio<T>> spawnableObjectRatioList;
}
```

1

For any object that we want to spawn in the dungeon we can create a SpawnableObjectRatio instance for that object using a generic class. This will allow us to define a spawn ratio for the object. This can be used for Enemies, Weapons ..... or any object that we want to spawn in the dungeon.



```
public class SpawnableObjectRatio<T>
{
    public T dungeonObject;
    public int ratio;
}
```

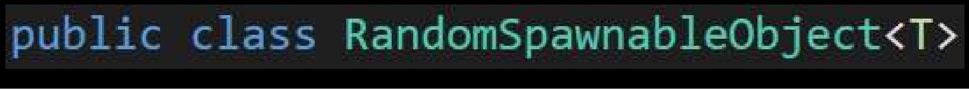
3

We'll use lists of SpawnableObjectsByLevel<EnemyDetailsSO> in the dungeon RoomTemplateSO scriptable object class to enable lists of enemies and their spawn ratios to be defined for all the relevant dungeon levels in the inspector.

Spawning Enemies Concepts

Generic Spawnable Objects By Level

This RandomSpawnableObject<T> class will allow us to easily get a random item to spawn

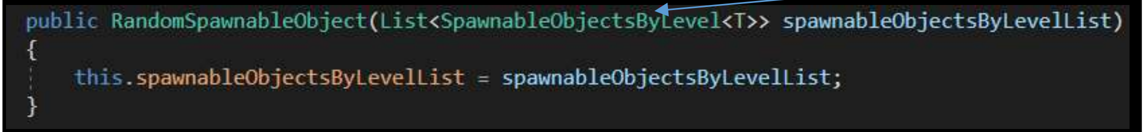


```
public class RandomSpawnableObject<T>
```

1

This class takes a List<SpawnableObjectsByLevel<T> as it's constructor parameter

CONSTRUCTOR

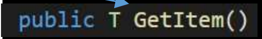


```
public RandomSpawnableObject(List<SpawnableObjectsByLevel<T>> spawnableObjectsByLevelList)
{
    this.spawnableObjectsByLevelList = spawnableObjectsByLevelList;
}
```

2

The GetItem() method will return a random object from the List<SpawnableObjectsByLevel<T> that was passed in as the constructor parameter

PUBLIC METHOD TO RETURN A RANDOM OBJECT



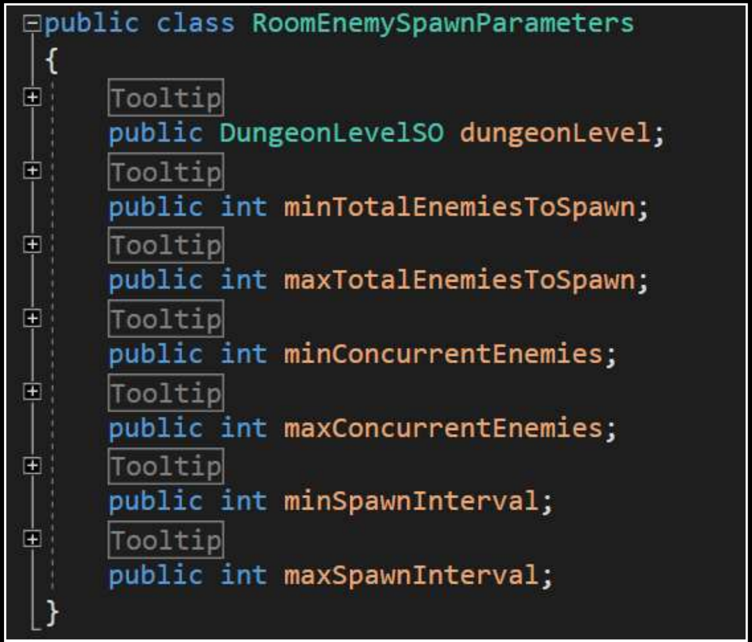
```
public T GetItem()
```

Spawning Enemies Concepts

Room Enemy Spawn Parameters

The previous classes allow us to define what objects to spawn by dungeon level.

We’ll also create the class ‘RoomEnemySpawnParameters’ to define how many enemies should be spawned, how many concurrent enemies there should be, and what the spawn interval should be between enemies spawning. We’ll use lists of this class in the dungeon RoomTemplateSO scriptable object class to enable the spawn parameters to be defined for all the relevant dungeon levels in the inspector.



```
public class RoomEnemySpawnParameters
{
        Tooltip
        public DungeonLevelSO dungeonLevel;
        Tooltip
        public int minTotalEnemiesToSpawn;
        Tooltip
        public int maxTotalEnemiesToSpawn;
        Tooltip
        public int minConcurrentEnemies;
        Tooltip
        public int maxConcurrentEnemies;
        Tooltip
        public int minSpawnInterval;
        Tooltip
        public int maxSpawnInterval;
}
```

Spawning Enemies Concepts

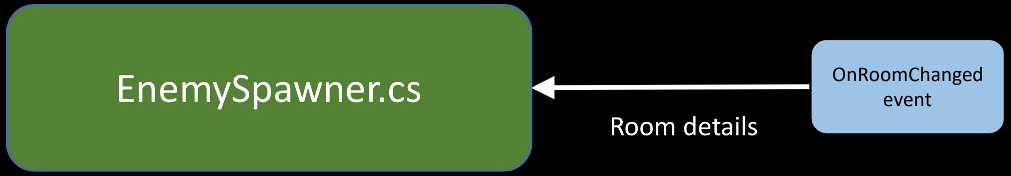
Enemy Spawner Singleton

The EnemySpawner will be a singleton gameobject in the scene.

It subscribes to the OnRoomChanged event, and when the player enters a room it checks to see if the room has already been cleared of enemies.

If it hasn't and there are enemies defined to be spawned in the room for this level, then the spawner will start spawning the enemies defined in the dungeon RoomTemplateSO.

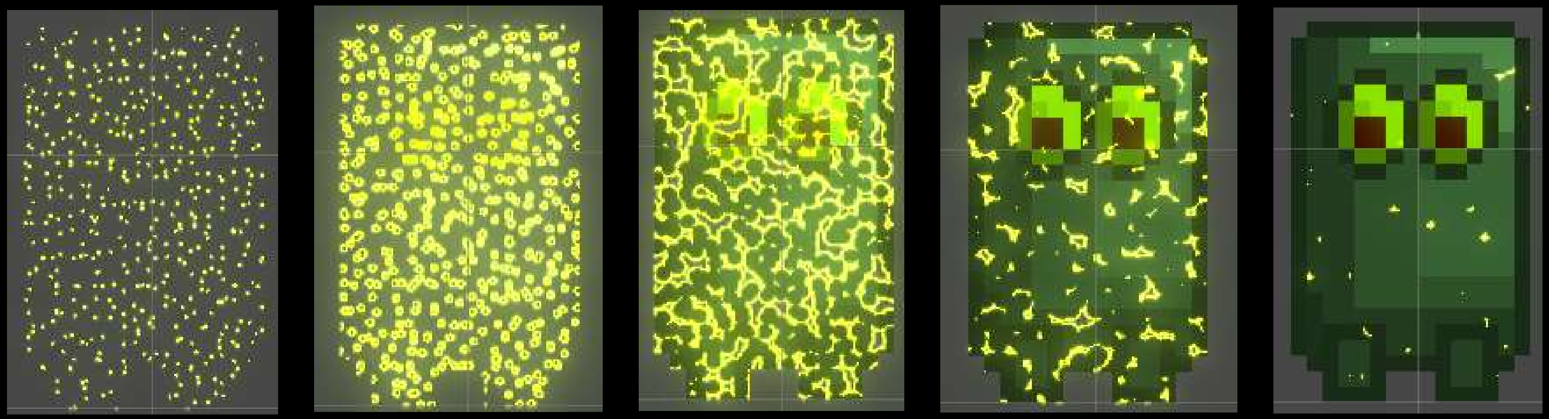
The spawner will keep track of how many enemies need to be spawned, how many have been spawned, how many have been destroyed, and thus it can work out if the room has been cleared of enemies.



Spawning Enemies Concepts

Materializing Enemies

When enemies are spawned in a room, they will be spawned randomly at the previously defined spawn positions in the room spawnPositionsArray.



We’ll use a MaterializeShader to create a materialize effect when enemies are spawned