README for BasicRoom.prefab

Overview:

This prefab contains the two basic components for rooms in Zombie Dog Outbreak 2.

Video:

https://youtu.be/dhBcAJklQ_c

This video demonstrates the vent opening animation.

https://youtu.be/FBkZXyQ0b28

This video demonstrates one door being unlocked and another that is still blocking access to a room.

Child GameObjects:

-'door':

This object will be copied as needed for all doors on the map and should block the player and enemies from passing. It has a BoxCollider2D component and a child GameObject 'sprite' with a SpriteRenderer component that displays a basic gray door sprite.

-'vent':

This object will be copied as needed for all spawn points for enemies on the map. It has a SpriteRenderer component that displays the closed vent sprite taken from Among Us. It also has an Animator component that references the VentController animator controller. The animator controller contains three stages: Closed, Ready and Open. Closed is the default stage, where the vent displays the closed vent sprite. When Open is played, the vent open animation plays for roughly half a second and returns to Closed. The animator has one parameter: bool spawning, which causes the animator controller to wait in the Ready stage as long as it's true. Ready changes the normal closed sprite to a red glowing vent sprite.

Usage:

The prefab and its child GameObjects can be positioned manually or by modifying their transform components with code at runtime. The parent GameObject should have some Room script attached to it as a component for the door to be unlockable and the vent to spawn enemies. Use SetActive(false) on the 'door' GameObject to
unlock it, and SetActive(true) to lock it.

See my GameFinal scene as an example:

-MapManager creates a new AbstractRoomFactory of type LargeRoomFactory

-MapManager calls the createRoom method of LargeRoomFactory twice, providing room names

-The LargeRoomFactory instantiates a new copy of the BasicRoom prefab:

Copies of the door and vent child GameObjects are made,
The doors and vents are positioned at predefinied coordinates,
A LargeRoom component is added to the new room GameObject,
The LargeRoom script is provided with references to all of

doors and vents,

The LargeRoom component is returned to the MapManager for later reference.

-MapManager deletes the old factory component

-MapManager creates a new AbstractRoomFactory of type SmallRoomFactory

-MapManager calls the createRoom method of AbstractRoomFactory four times, providing room names

-The SmallRoomFactory instantiates a new copy of the BasicRoom prefab,

The door and vent are positioned at predefinied coordinates,

A SmallRoom component is added to the new room GameObject,

The SmallRoom script is provided with references to the door and vent

The SmallRoom component is returned to the MapManager for later reference.

-MapManager calls the unlockRoom method in the desired Room script, which then disables all of its doors

-MapManager calls the getSpawnPoints method in the desired Room script, which provides the positions of the vents

-MapManager provides the vent positions to GameManager, which spawns enemies at those locations

Troubleshooting:

-Code integration:

In order to get the child components dynamically, you have to use the Find method in Transform, not the more common GameObject Find.

If the child objects appear not to be positioned at the correct coordinates when transform.position is set, it's likely because your positions are switching from local space to world space. Use TransformPoint in Transform to turn the provided local coordinates into world space coordinates. InverseTransform works the same way, in the other direction.

If the door is still colliding with the player after it's been unlocked, make sure you are calling SetActive(false) on the parent object, named 'door', as disabling the child will leave the parent's box collider component active.

-Manual adjustments:

If you want to change the size of the door, it's easiest to use the scale tool on the parent object (named 'door'), as that changes the box collider as well. If you adjust the size or position of just the child object (named 'sprite'), then the collider will not be positioned correctly, and the door may have an offset from the parent position.