## **Tutorial 4 – Conditional AND and Informational Methods**

Estimated Time: 60 minutes

### **Prerequisite**

If statements, Else statements, Unity familiarity

### **Learning Objectives**

Students will learn the conditional and statement. The students will learn the informational API methods, (methods that return information about objects or the player).

### **API Methods Covered**

* getObjectPositionX()
* getObjectPositionY()
* getObjectPositionZ()
* getPlayerPositionX()
* getPlayerPositionY()
* getPlayerPositionZ()
* setObjectSpin()

### **Activity**

1. Describe the X ,Y, Z coordinates to the students to that they understand 3D space
2. Describe the different informational groups
   1. Information about the object position
   2. Information about the player position
3. Describe the conditional AND
4. Set a plane and a cube object
5. Set a script on the capsule (cubeScript.cs)
   1. if the player moves to a certain position on the plane, causes the object to tether to controller, else the object is not tethered and the object spins
   2. When the object moves to a certain position, change the object’s color.

### **Scripts**

**cubeScript.cs**

void updateGame () {

if (getPlayerPositionZ() > 2 && getPlayerPositionZ() < 6)

{

setTetherObjectToRightController(true);

}

else

{

setTetherObjectToRightController(false);

setObjectSpin(true);

}

if (getObjectPositionX() > 0 && getObjectPositionX() < 5)

{

setObjectColor("red");

}

else

{

setObjectColor("blue");

}

}

### **Optional Tutorial 4B**

Estimated Time: 20 minutes

1. Create a Rock (from Rocks catagory of complex objects)
2. Add script (rock.cs)
   1. If right trigger and left trigger are down, make the object spin
   2. Else, the object doesn’t spin

#### ***Source Code***

**rock.cs**

void updateGame () {

if (isControllerRightTriggerDown() && isControllerLeftTriggerDown())

{

setObjectSpin(true);

}

else

{

setObjectSpin(false);

}

}

### **Optional Tutorial 4C**

Estimated Time: 20 minutes

1. Create chair (from Furniture Category)
2. Add script (throwObject.cs)
3. Make object throwable
4. If the object is thrown into a specific corner, set it to not throwable

#### ***Source Code***

**throwObject.cs**

void buildGame () {

setObjectThrowable(true);

}

void updateGame () {

if (getObjectPositionX() > 1 && getObjectPositionZ() > 1)

{

setObjectThrowable(false);

}

}