Session 19

|  |
| --- |
| **Objectives**  The goals of this session |
| 1. Finish off the activities for session 17. 2. User input introduction:    1. Keyboard inputs 3. Introduction of new activities    1. Create a big square for a ground    2. Create a tree prefab    3. Randomly spawn trees on top of the ground |

|  |
| --- |
| **Method Focuses**  New methods or functions that you will learn during this session |
| **Input**   * bool GetKey(*string name*); * bool GetKeyDown(*string name*); * bool GetKeyUp(*string name*); * bool GetMouseButton(*int button*); * bool GetMouseButtonDown(*int button*); * bool GetMouseButtonUp(*int button*);   **Randomness**  **Vectors** |
| **Property Focuses**  Properties (variables) of classes you will learn during this session |
| **N/A** |
| **Documentation Links** |
| Vector3 class  <https://docs.unity3d.com/ScriptReference/Vector3.html>  Input class  <https://docs.unity3d.com/2020.1/Documentation/ScriptReference/Input.html>  Random class  <https://docs.unity3d.com/ScriptReference/Random.html> |

## Focus Activity 1

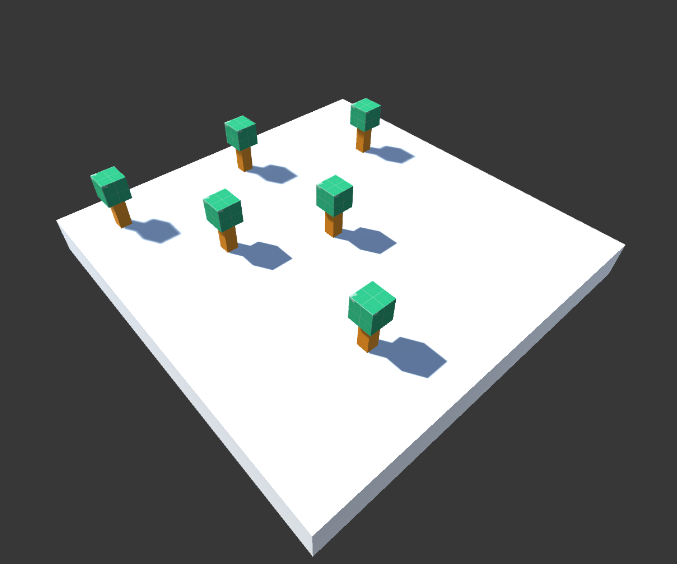
**Goal**

Create a platform and have trees randomly spawn on that platform.

This will require knowledge of:

* Spawning prefabs
* Making random numbers
* Loops
* Changing the position of an object
* Creating materials

The outcome should look like this:



## Example 7

|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  // You can't forget this when using text mesh pro objects  using TMPro;  public class Example7 : MonoBehaviour  {  //////////////////////////////////////////////////////  // Session 18 Example 7  // Updating the text on an object  public TextMeshProUGUI textObject;  public void UpdateText(string toUpdate) {  textObject.text = toUpdate;  }  // Debug to the console what mouse button was pressed  public void GetMouseButtonPressed() {  Input.GetMouseButton(0); // Returns true whenever the mouse button is held down  Input.GetMouseButtonDown(0); // Returns true when the player first presses their mouse  Input.GetMouseButtonUp(0); // Returns true when the player releases the button  // 0 = Left click  // 1 = Right click  // 2 = Middile click  if (Input.GetMouseButtonUp(0)) {  Debug.Log("Pressed left click.");  }  if (Input.GetMouseButtonUp(1)) {  Debug.Log("Pressed right click.");  }  if (Input.GetMouseButtonUp(2)) {  Debug.Log("Pressed middle click.");  }  }  // Getting what key is pressed  public void GetKeyPressed() {  // You can type the name of the key  if (Input.GetKeyUp("space")) {  print("Space key was released");  }  // Or you can get the keycode variable for the key  if (Input.GetKeyUp(KeyCode.Space)) {  print("Space key was released");  }  }  // It is advisable to use GetAxis because...  // The script below is a simple script for car movement  public float speed = 10.0f;  public float rotationSpeed = 100.0f;  public void SimpleMovement() {  // Get the horizontal and vertical axis.  // By default they are mapped to the arrow keys.  // The value is in the range -1 to 1  float translation = Input.GetAxis("Vertical") \* speed;  float rotation = Input.GetAxis("Horizontal") \* rotationSpeed;  // Make it move 10 meters per second instead of 10 meters per frame...  translation \*= Time.deltaTime;  rotation \*= Time.deltaTime;  // Move translation along the object's z-axis  transform.Translate(0, 0, translation);  // Rotate around our y-axis  transform.Rotate(0, rotation, 0);  }  } |

|  |  |
| --- | --- |
| **Session Information**  You can ignore this | |
| **Planned Information** | |
| Session Time Session Date | 7:00PM – 8:00PM AWST 31 January 2021 |
| **Real Information** | |
| Session Time Session Date | 7:06PM – 8:06PM AWST 31 January 2021 |
| Github Name | Session19Examples |
| Session Length | 1 hour |
| Activities Completed | |  |  | | --- | --- | | □ | Activity 3 | | □ | Focus Activity 1 | |
| Signature | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *By signing this you confirm that the session has occurred and the subjects that have been taught have been allocated on this document.* |