Session 18

|  |
| --- |
| **Objectives**  The goals of this session |
| 1. Finish off the activities for session 17. 2. User input introduction:    1. Keyboard inputs 3. Game UI recap:    1. Changing the text of text components    2. Getting button inputs |

|  |
| --- |
| **Method Focuses**  New methods or functions that you will learn during this session |
| **Vector3**   * Vector3 MoveTowards(*Vector3 current, Vector3 target, float maxDistanceDelta*); * float Distance(*Vector3 a, Vector3 b*);   **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*); |
| **Property Focuses**  Properties (variables) of classes you will learn during this session |
| **Vector3**  float sqrMagnitude;  **TextMeshProUGUI**  String text |
| **Documentation Links** |
| Text Mesh Pro <http://digitalnativestudios.com/textmeshpro/docs/textmeshpro-component/>  Vector3.Distance  <https://docs.unity3d.com/2020.1/Documentation/ScriptReference/Vector3.Distance.html>  Vector3.MoveTowards  <https://docs.unity3d.com/2020.1/Documentation/ScriptReference/Vector3.MoveTowards.html>  Vector3.sqrMagnitude  <https://docs.unity3d.com/ScriptReference/Vector3-sqrMagnitude.html>  Input class  <https://docs.unity3d.com/2020.1/Documentation/ScriptReference/Input.html> |

## 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);  }  } |

### Activity 5

|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  using UnityEngine.EventSystems; // Implement this namespace if you are using interfaces to handle events  using TMPro; // Don't forget to import this  // We can detect button presses using an interface  // Add the IPointerClickHandler  public class Activities5 : MonoBehaviour  {  //////////////////////////////////////////////////////  // Session 18 Activities 5  // Goal: Change the text on a string and debug to the console when the button is pressed  // Change the text when the button is pressed  public TextMeshProUGUI textObject;  // This way is the simplest way!  // So the way buttons work is that they will run a certain method or function when an event is ran  public void ChangeText() {  textObject.text = "Anything we want.";  }  // This is the more complex way  // Uses IPointerClickHandler  public void OnPointerClick(PointerEventData eventData) {  ChangeText();  }  } |

#### Activities 4 (Complete as task if not completed already)

|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  public class Activities4 : MonoBehaviour  {  ////////////////////////////////////////////////////////////  // Activity 4  // Goal:  // Using the distance and the square magnitude function find what object is closer to the central object  // Variables provides  public Transform objectA;  public Transform objectB;  public Transform centralObject;    // Start is called before the first frame update  void Start()  {    }  // Update is called once per frame  void Update()  {  }  void DistanceWay()  {  // Find the distance between the central object and objectA  float distanceA = Vector3.Distance(centralObject.position, objectA.position);  // Find the distance between the central object and objectB  // Compare the distances and output  }  void SqrMagnitudeWay()  {  }  } |

#### Activities 3 (Complete as task if not completed already)

|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  public class Activities3 : MonoBehaviour  {  ////////////////////////////////////////////////////////////  // Activity 3  // Goal:  // Get this script to move towards a game object at a distance  // of atleast 5 metres  // The starting variables have been made for you  public float distanceToFollow = 5;  public GameObject objectToFollow;  // Your variables if required  // Start is called before the first frame update  void Start()  {    }  // Update is called once per frame  void Update()  {  }  } |

|  |  |
| --- | --- |
| **Session Information**  You can ignore this | |
| **Planned Information** | |
| Session Time Session Date | 5:20PM – 6:20PM AWST 27 January 2021 |
| **Real Information** | |
| Session Time Session Date | 5:26PM – 6:36AWST 27 January 2021 |
| Github Name | Session18Examples |
| Session Length | 1 hour 10 minutes |
| Activities Completed | |  |  | | --- | --- | | □ | Activity 3 | | □ | Activity 4 | | □ | Activity 5 | |
| Signature | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *By signing this you confirm that the session has occurred and the subjects that have been taught have been allocated on this document.* |