|  |
| --- |
| **STUDENT NAME** |
| Alley Chaggar |

**LAB #8**

[ACTIVITY 1 2](#_Toc49113775)

[ACTIVITY 2 6](#_Toc49113776)

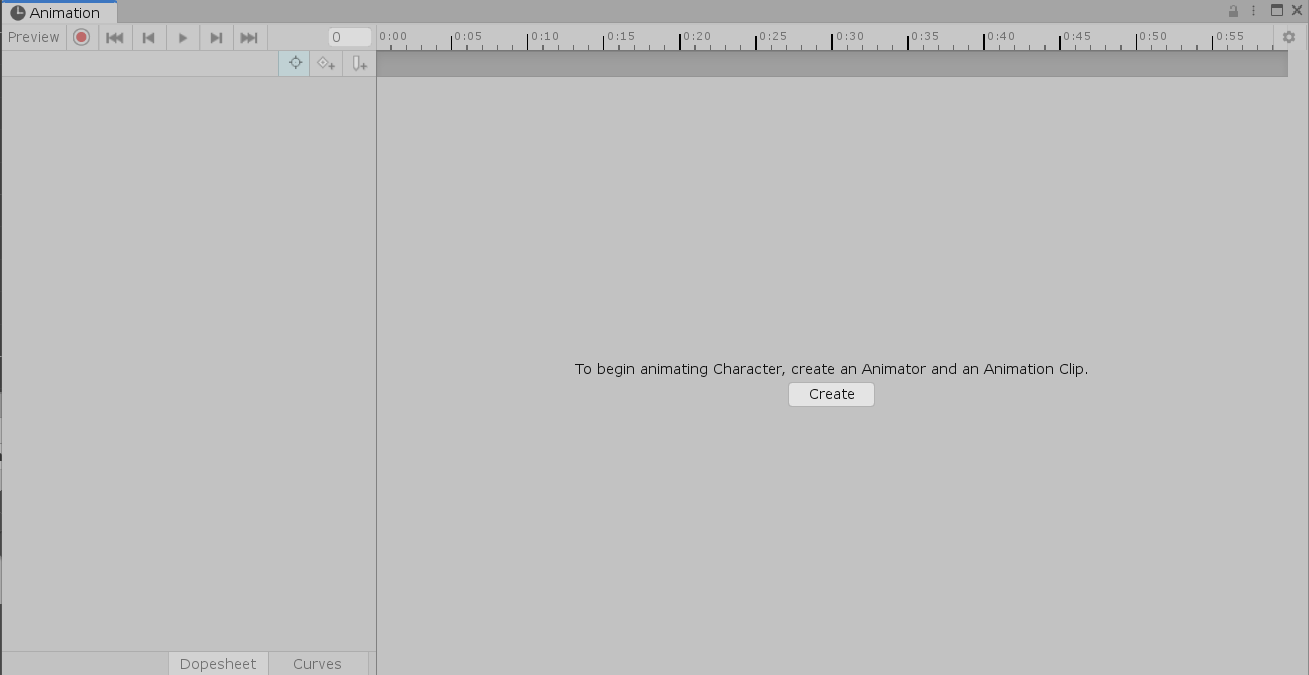
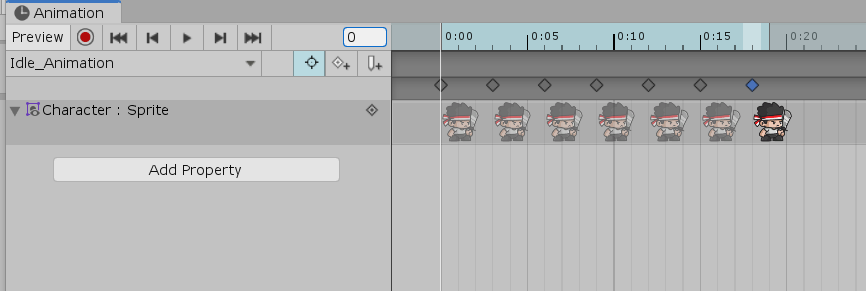
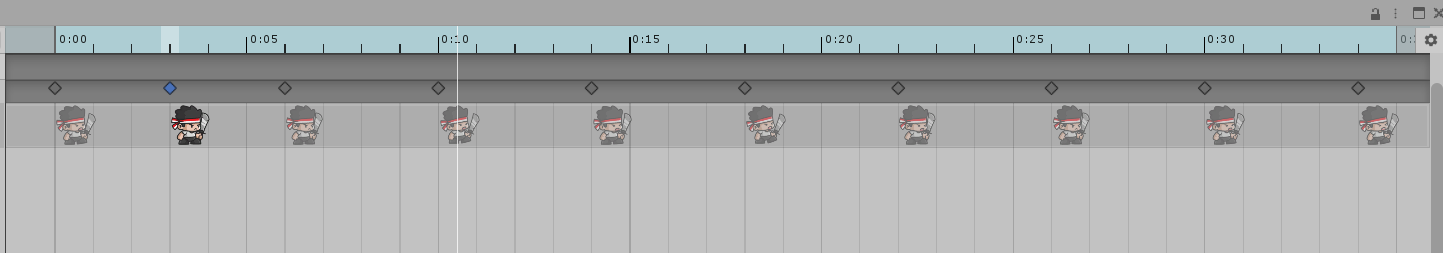
[ACTIVITY 3 12](#_Toc49113777)

# ACTIVITY 1

## ANIMATING A CHARACTER IN UNITY

This tutorial demonstrates how to import and animate a character in Unity. The character is imported as a Spritesheet, then it is slice and finally used for the animation.

Follow the steps below:

1. Double – click on the Unity Hub icon.
2. Create a New 2D project and name it CharacterAnimation.
3. Right-click on the Assets 🡪 Create 🡪 Folder.
4. Name the new folder Character.
5. Download the HeroGuySpriteSheet.png file from E-Centennial.
6. Drag and Drop the Spritesheet to the new Unity Project under the Character folder.
7. Click on the newly imported asset and under the Inspector select:
   1. Texture Type: Sprite (2D and UI).
   2. Sprite Mode: Multiple.
8. Click Apply.
9. Click Sprite Editor.
10. Click Slide 🡪 Automatic 🡪 Slice.
11. Click on the 10 sprites and name them Idle1, Idle2…Idle10.
12. Click on the next 10 and name them Hurt1, Hurt2… Hurt10.
13. Click on the next and name it Jump1.
14. Click on the next 10 and name them Run1, Run2… Run10.
15. Click on the rest and name them Attack1, Attack2… Attack10.
16. Click Apply.
17. Under the Hierarchy right – click 🡪 Create Empty.
18. Name the new GameObject Character.
19. Click on the Character object.
20. Click Window 🡪 Animation and this window should appear:  
      
    
21. Click Create.
22. Double – click on the Character folder.
23. Create a new folder and name it, Animations.
24. Name the new animation Idle\_Animation and press OK.
25. Select the Idle1 sprite, hold Shift and click on the Idle10, selecting all the Idle sprites (1 to 10).
26. Drag and drop all the selected Sprite in the animation window.
27. Move the last keyframe to a distance of 21 and set all the other to distances of 3 seconds, as shown below:   
      
    
28. Click on the Idle\_Animation name.
29. Click Create New Clip.
30. Drag and drop the Hurt sprites.
31. Modify its frame rate and create a smooth animation, such as shown below:  
      
    
32. Repeat steps 28 – 31 for the Run sprites and name the animation Run\_Animation.
33. Repeat steps 28 – 31 for the Run sprites, but set the animation rate to a slower rate to simulate a walking speed and name the animation Walk\_Animation.
34. Repeat steps 28 – 31 for the Attack sprites and name the animation Attack\_Animation.
35. Repeat steps 28 – 31 for the Jump Sprites plus 2 Run Sprites before and 2 Run Sprites after and name the animation Jump\_Animation.

Once you complete the previous steps:

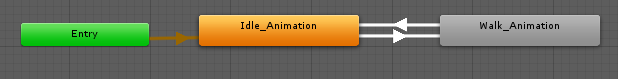
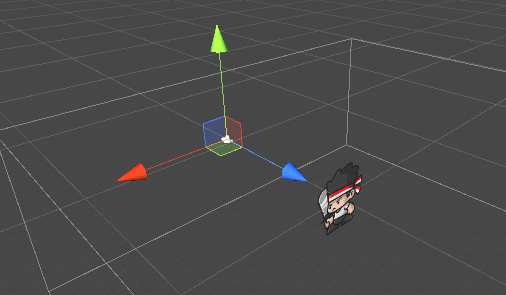
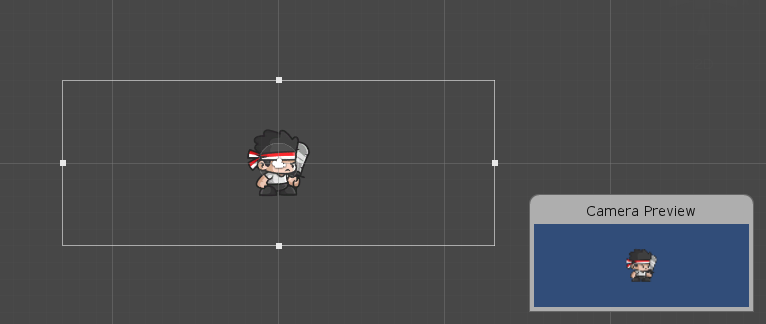
|  |
| --- |
| **TASK 1.1:**  Take a screenshot of the Unity Animation panel and paste it below: |
| Graphical user interface  Description automatically generated |

# ACTIVITY 2

## TRANSITIONS

If we play the game the animation seem pretty clumsy. We need to somehow connect them. This is called Transition. In Unity, the Transitions are created in the Animator window. This tutorial demonstrates how to create the transitions between your animations.

Follow the steps below:

1. Click Windows 🡪 Animation (keep the Game Character selected).
2. The Idle\_Animation should be by default highlighted as the default one.
3. If not, right – click on it 🡪 Set Default.  
     
   
4. Right – Click select Set Transition.
5. Connect the Arrow to the Walk\_Animation.
6. Click on the Camera under the Hierarchy.
7. Ensure that the character is inside the camera’s frustrum:
   1. in 3D projection:  
        
      
   2. in 2D projection:  
        
      
8. Press the Play button and see, if the character appears.
9. If the sprite disappears after playing the game, click on the Character and in the Inspect set the Sprite option under the Sprite Renderer tab to the Idle1.
10. Under the Animator click on the Parameters tab.
11. Click on the + button.
12. Name the new parameter Speed and set its value to 0.1.
13. Click on the Transition arrow from the Idle to the Walk animation.
14. Under the Inspector scroll down and find the Conditions section.
15. Click on the + symbol.
16. Select the Speed and set to Greater than 0.1.
17. Repeat the steps 12 – 15 but for the opposite Transition arrow and set the new condition to Lower than 0.1.
18. In the Animator set the speed to 0.
19. Play the Game.

Once you complete the previous steps:

|  |
| --- |
| **TASK 2.1:**  Take a screenshot of the Game View and paste it below: |
| A screenshot of a computer  Description automatically generated with medium confidence |

1. Set the Speed to 0.2 (while the game is played).

|  |
| --- |
| **TASK 2.2:**  Take a screenshot of the Game View and paste it below: |
| A screenshot of a computer  Description automatically generated with medium confidence |

|  |
| --- |
| **TASK 2.3:**  What do you see? What animation is played? Answer the questions in the cell below. |
| The Idle animation is the only one that is playing and it’s slower than when the speed was at 1. |

1. Stop the Game.
2. Return to the Animator’s windows.
3. Connect the Walk animation with the Run animation and vice versa.
4. Repeat the steps 13 – 18, for these animations setting the limit to 1.0.
5. Play the Game, changing the speed from 0.5 to 1.1.

Once you complete the previous steps:

|  |
| --- |
| **TASK 2.4:**  Take a screenshot of the Game View and paste it below: |
| Graphical user interface  Description automatically generated |

|  |
| --- |
| **TASK 2.5:**  In what speed the animation is shifted? Answer the question in the cell below. |
| The speed is shifted 0.6 |

1. Create the Parameter FireButtonPressed as Boolean, JumpButtonPressed as Boolean, Damage as Float.
2. Connect the Animations with transitions using the above created parameters, as you see fit. Try to cover all the cases.

|  |
| --- |
| **TASK 2.6:**  Take a screenshot of the Animator window and paste it below: |
| Graphical user interface, website  Description automatically generated |

# ACTIVITY 3

## TESTING THE ANIMATION

This tutorial demonstrates how to test the animation introducing some scripting to the project. The walk script contains a very basic movement scripts, that checks if the movement and animation are coordinated.

Follow the steps below:

1. Right – click on the Assets 🡪 Create 🡪 Folder.
2. Name the new folder Scripts.
3. Right – click on the scripts and select C# Script.
4. Name the new script Walk.
5. Add the code as shown below:

|  |
| --- |
| **CODE** |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  public class walk : MonoBehaviour  {  Animator anim;  void Start()  {  anim = GetComponent<Animator>();  }  void Update()  {  Movement();  {  float move = Input.GetAxis("Horizontal");  anim.SetFloat("Speed", move);  }  }  void Movement()  {  if (Input.GetKey(KeyCode.D))  {  transform.Translate(Vector2.right \* 3f \* Time.deltaTime);  transform.eulerAngles = new Vector2(0, 0);  }  if (Input.GetKey(KeyCode.A))  {  transform.Translate(Vector2.left \* 3f \* Time.deltaTime);  transform.eulerAngles = new Vector2(0, 0);  }  }  } |

1. Press Ctrl+S.
2. Return to the Editor.
3. Click on the Character and make the Animator window visible.
4. Press the Play Button.
5. Press the D and A buttons and see the changes at the Animator.

Once you complete the previous steps:

|  |
| --- |
| **TASK 3.1:**  Take a screenshot of the Animator windows and paste it below: |
| Graphical user interface  Description automatically generated |

1. Stop the Game.
2. Open the walk script again.
3. Make a few changes as shown below:

|  |
| --- |
| **CODE** |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  public class walk : MonoBehaviour  {  Animator anim;  int JumpButtonPressed = Animator.StringToHash("JumpButtonPressed");  void Start()  {  anim = GetComponent<Animator>();  }  void Update()  {  float move = Input.GetAxis("Horizontal");  anim.SetFloat("Speed", move);  AnimatorStateInfo stateInfo = anim.GetCurrentAnimatorStateInfo(0);  if (Input.GetKeyDown(KeyCode.Space))  {  anim.SetTrigger(JumpButtonPressed);  }  } |

1. Press Ctrl+S.
2. Return to the Editor.
3. Start the Game.
4. Use the A, D and Space keys.

Once you complete the previous steps:

|  |
| --- |
| **TASK 3.2:**  Take a screenshot of the Animator windows and paste it below: |
| Graphical user interface  Description automatically generated |

1. Stop the Game.
2. Open the walk script again.
3. Make a few changes as shown below:

|  |
| --- |
| **CODE** |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  public class walk : MonoBehaviour  {  //Animator anim;  //bool jump = false;  Animator anim;  int JumpButtonPressed = Animator.StringToHash("JumpButtonPressed");  int FireButtonPressed = Animator.StringToHash("FireButtonPressed");  public float damageTaken;  float timeLeft = 3.0f;  // do later var runStateHash : int = Animator.StringToHash("Base Layer.Run");  void Start()  {  //anim = GetComponent<Animator>();  anim = GetComponent<Animator>();  }  void Update()  {  Movement();  // float horizontalMove = Input.GetAxis("Horizontal");  // anim.SetFloat("Speed", Mathf.Abs(horizontalMove));  // anim.SetBool("JumpButonPressed", jump);  float move = Input.GetAxis("Horizontal");  anim.SetFloat("Speed", move);  anim.SetFloat("Damage", damageTaken);  AnimatorStateInfo stateInfo = anim.GetCurrentAnimatorStateInfo(0);  if (Input.GetKeyDown(KeyCode.Space))  {  anim.SetTrigger(JumpButtonPressed);  }  else if (Input.GetKeyUp(KeyCode.Space))  {  anim.ResetTrigger(JumpButtonPressed);  }  else if (Input.GetKeyDown(KeyCode.F))  {  anim.SetTrigger(FireButtonPressed);  }  else if (Input.GetKeyUp(KeyCode.F))  {  anim.ResetTrigger(FireButtonPressed);  }  else if (Input.GetKey(KeyCode.L))  {  Damage();  }  else if (Input.GetKey(KeyCode.K))  {  RepairDamage();  }  }  void Movement()  {  if (Input.GetKey(KeyCode.D))  {  transform.Translate(Vector2.right \* 3f \* Time.deltaTime);  transform.eulerAngles = new Vector2(0, 0);  }  if (Input.GetKey(KeyCode.A))  {  transform.Translate(Vector2.left \* 3f \* Time.deltaTime);  transform.eulerAngles = new Vector2(0, 0);  }  if (Input.GetButtonDown("Jump"))  {  transform.Translate(Vector2.up \* 10f \* Time.deltaTime);  }  }  void Damage()  {  if (Input.GetKey(KeyCode.L))  {  damageTaken = 1.0f;  }  }  void RepairDamage()  {  if (Input.GetKey(KeyCode.K))  {  damageTaken = 0.0f;  }  }  } |

1. Use the A, D, Space, L and K keys.

Once you complete the previous steps:

|  |
| --- |
| **TASK 3.3:**  Take a screenshot of the Animator windows and paste it below: |
| Graphical user interface, website  Description automatically generated |

|  |
| --- |
| **TASK 3.4:**  Compress this Unity project and add it to the submission folder. |
| upload iconIn the LMS, add the file to the assignment Lab #8 submission folder. You can submit multiple files at a time. |

FINAL STEP: Save this document as a PDF. Upload the PDF to the Lab #8 submission folder.