

# Bonus Features 3 Solution Walkthrough



Easy - Randomize Obstacles	2
Medium - Double Jump!	15
Hard - Dash Ability	17
Expert - Game Start Animation	26

### **Easy - Randomize Obstacles**

1. Navigate to the Scripts folder and open up the script **SpawnManager.cs**. We will need to update the obstaclePrefab variable to be an array and add in a new variable for spawning a random obstacle.

```
public GameObject[] obstaclePrefabs;
private Vector3 spawnPos = new Vector3(25, 0, 0);
private float startDelay = 2;
private float repeatRate = 2;
private PlayerController playerControllerScript;
private int randomObstacle;
```

2. We now need to update the **SpawnObstacle** method. First we will need to set the randomObstacle number, then spawn the object from that position in the array. The updated method will look like this:

```
void SpawnObstacle ()
{
    if(playerControllerScript.gameOver == false)
    {
        randomObstacle = Random.Range(0, obstaclePrefabs.Length);
        Instantiate(obstaclePrefabs[randomObstacle], spawnPos,
    obstaclePrefabs[randomObstacle].transform.rotation);
    }
}
```

Save the script and head back to Unity.

3. We now need to set up some more obstacle prefabs. First, create an empty GameObject by right-clicking in the Hierarchy and selecting **Create Empty**. Rename the new GameObject to *Obstacle\_Crate*.

'≔ Hierar	chy		<b>∂</b> *≡
Create *			
	otype 3		*≡
	ain Camera		
	rectional Light round		
	ackground		
🔹 🕨 🧊 Pla			>
	oawnManager		
	Com		
	Сору		
	Paste		
	Rename		
	Duplicate		
	Delete		
	Create Empty		
	3D Object	>	
	2D Object	>	
	Effects	>	
	Light	>	

4. On the ObstacleCrate GameObject, add a Box Collider. We will adjust the size of this later. In the Inspector, click **Add Component > Physics > Box Collider**.

Inspector	•			2	•≡
	ostacleCrate		_ st	tatic	•
Tag U	ntagged 🕴 Layer Default				\$
🔻 🦶 🛛 Trai	isform			井	\$,
Position	X 12 Y 0	Z	0		
Rotation	X 0 Y 0	Z	0		
Scale	X 1 Y 1	Z	1		
	Add Component	]			
	<u>(a)</u>				
	< Physics	1			
	🐺 Box Collider 🧧				
	🍯 Capsule Collider				
	🐥 Character Controller				

5. Next add a Rigidbody. In the Inspector, click **Add Component > Physics > Rigidbody**.

Inspector	r		<u> -</u> ≡
🕤 🗹 OI	bstacleCrate	9	🗌 🗌 Static 🔻
Tag U	ntagged	‡ Layer Default	\$
🔻 🙏 🛛 Trar	nsform		💽 🗟 🌣
Position		X 12 Y 0	Z 0
Rotation		X 0 Y 0	Z 0
Scale		X 1 Y 1	Z 1
🔻 📦 🗹 Вож	Collider		🔯 🕸 🏟
		🔥 Edit Collider	
Is Trigger			
Material		None (Physic Material)	0
Center		X 0 Y 0	ZO
Size		X 1 Y 1	Z 1
	_		1
	/	Add Component	
	<u> </u>		
	<	Physics	-
	-	er Controller	
	Characte		
	Cloth	ci sonic	
	♥ Configur	cable Jaint	1
	🖞 Constan		
	-		
	✓ Fixed Joi		
	≫Hinge Jo		
	🔠 Mesh Co		
	Å Rigidbod		
	😂 Sphere (	Collider	
	🥃 Spring J	oint	
	1.2		

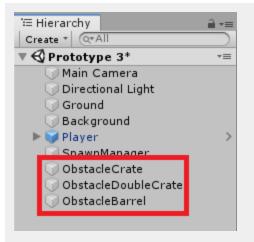
6. The last component we need on this GameObject, is the MoveLeft script. In the Inspector, click **Add Component** and search for **MoveLeft**, click on it to add it to the GameObject.

Inspector			et et i
ObstacleCra			Static
Tag Untagged	t Layer Default		
Transform			💽 큐
Position	X 12 Y 0	Z 0	
Rotation	X 0 Y 0	Z 0	
Scale	X 1 Y 1	Z 1	
🔻 💗 🗹 Вох Collider			「二」
	🔥 Edit Collider		
Is Trigger			
Material	None (Physic Material)		
Center	X 0 Y 0	Z 0	
Size	X 1 Y 1	Z 1	
🛚 🙏 Rigidbody			💽 井
Mass	1		
Drag	0		
Angular Drag	0.05		
Use Gravity			
Is Kinematic			
Interpolate	None		
Collision Detection	Discrete		
Constraints			
	Add Component	1	
Q Move	Left		
	Search	-	
œ Move L			
New sc		>	
New se	- ip -		

7. Change the Tag of the ObstacleCrate to **Obstacle**. This can be done by clicking on the **Untagged** dropdown and selecting **Obstacle**.

<li>Inspect</li>	tor	<b>≟</b> *≡
	ObstacleCrate	🗌 Static 🔻
Tag	Obstacle ‡	Layer Default +
Tr 🙏 🔻	Untagged	🔟 示 🌣
Position	Respawn	0 Z 0
Rotation	Finish	
Scale	EditorOnly	
🔻 🥡 🗹 B c	MainCamera	🛛 🖓 🎝
	Player	ider
Is Trigg Material	GameController	aterial) O
Center	✓ Obstacle	0 Z 0
Size	Ground	1 Z 1
▼ 🙏 🛛 Ri Mass	Add Tag	[] 규 �.
Drag	0	

 Now that we have the basis for our new Obstacles. Duplicate ObstacleCrate twice, rename one to ObstacleDoubleCrate and the other to ObstacleBarrel. You can duplicate a GameObject by either right-clicking on it and selecting Duplicate or selecting the GameObject and pressing Ctrl + D (PC) or Cmd + D (Mac).

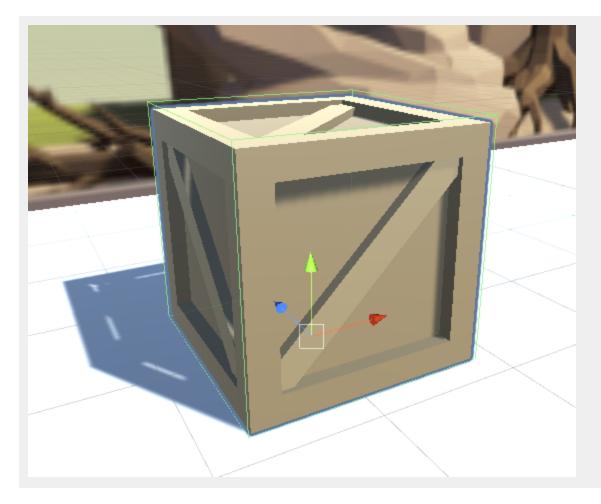


9. In the Project view, navigate to the Obstacles folder (Course Library > Obstacles). Drag Crate\_01 into ObstacleCrate in the Hierarchy. In the ObstacleDoubleCrate GameObject, drag in Crate\_01 twice. In the ObstacleBarrel GameObject drag in either of the Barrels, we chose Barrel\_02.



10. On the *ObstacleCrate* GameObject, Adjust the **Size** and **Center** of the BoxCollider so that it fits the crate.

Inspective	tor									ì.	•≡
	Obs	tacleCrat	e						Sta	tic	•
Tag	Obs	tacle		\$	Laye	r Defau	ılt				\$
V 🙏 – Tr	ans	form								킕	\$,
Position			X[	15.059	Y	0	Z	2 0			
Rotation	1		X[	0	Y	0	Z	2 0			
Scale			X	1	Y	1	Z	2 1			
🔻 🧊 🗹 B a	ож С	ollider								킕	\$,
			4	🍾 Edi	t Colli	der					
Is Trigg	er										
Material			No	one (Ph	ysic M	aterial)					0
Center			Х	0	Y	0.8	Z	2 0			
Size			X	1.6	Y	1.6	Z	1.	.6		
🔻 🙏 🛛 Ri	gidb	ody								킕	\$,
Mass			1								
Drag			0								
Angular	Drag	,	0.	05							
Use Gra	vity		$\checkmark$								
Is Kinen	natic										
Interpol	ate		N	one						_	ŧ
Collision	n Det	ection	D	iscrete						_	ŧ
▶ Constra	ints										
🔻 🖬 🗹 M	ove	Left (Sc	ript)	)						킕	\$,
				Marriela	EL.						0
Script				MoveLe	ert						-
Script				Comp			_				-



11. On the *ObstacleDoubleCrate* GameObject, adjust the crates so that one is stacked on the other. Then adjust the BoxCollider to fit around both crates.



Inspector		<b>∂</b> •≡
🕜 🗹 ObstacleDout	leCrate	🗌 🗌 Static 🔻
Tag Obstacle	‡ Layer Default	
▼人 Transform		🔯 다 🔅
Position	X 19.4 Y 0	Z 0
Rotation	X 0 Y 0	Z 0
Scale	X 1 Y 1	Z 1
🔻 🤪 🗹 Вох Collider		🛛 🖓 🐥
	🔥 Edit Collider	
Is Trigger		
Material	None (Physic Material)	0
Center	X 0 Y 1.6	Z 0
Size	X 1.6 Y 3.2	Z 1.6
🔻 🙏 Rigidbody		🔯 다 🔅
Mass	1	
Drag	0	
Angular Drag	0.05	
Use Gravity		
Is Kinematic		
Interpolate	None	+
Collision Detection	Discrete	+
▶ Constraints		
🔻 📾 🗹 Move Left (Sci	ript)	🔯 🕸 🖗
Script	MoveLeft	0
	Add Component	

12. On the *ObstacleBarrel* GameObject, adjust the **BoxCollider Size** and **Center** to fit the barrel.

<ol> <li>Inspector</li> </ol>		<b>∂</b> •≡
🕝 🗹 ObstacleBarr	el	🗌 🗌 Static 🔻
Tag Obstacle	+ Layer Default	+
▼人 Transform		🔯 🕸 🏟
Position	X 12 Y 0	Z 0
Rotation	X 0 Y 0	Z 0
Scale	X 1 Y 1	Z 1
🔻 🤪 🗹 Вох Collider		🔯 🕸 🖗
	🔥 Edit Collider	
Is Trigger		
Material	None (Physic Material)	
Center	X 0 Y 0.75	Z 0
Size	X 1 Y 1.5	Z 1
🔻 🙏 🛛 Rigidbody		🔯 🕸 🌾
Mass	1	
Drag	0	
Angular Drag	0.05	
Use Gravity		
Is Kinematic		
Interpolate	None	+
Collision Detection	Discrete	÷
Constraints		
🔻 📾 🗹 Move Left (Sci	ript)	🔯 🗟 🔅
Script	MoveLeft	0
	Add Component	]

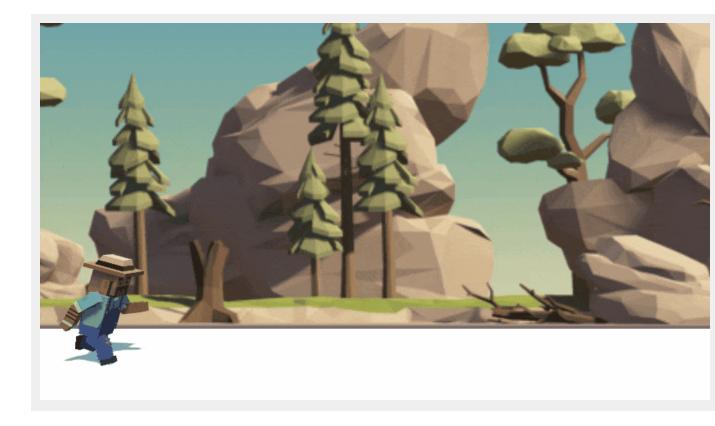
13. Navigate to the Prefabs folder in the Project view and drag the 3 new Obstacles from the Hierarchy to the Project view. Then delete the Obstacles from the Hierarchy.

Project 🛛 Cor Create T	nsole			
▶ 😭 Favorites	Assets > Pre	efabs		
▼ ■ Assets ▼ ■ Course Libr > ■ _Source_ > ■ Characte ■ Obstacles ■ Particles	1	ObstacleBa	ObstacleCr	ObstacleD
▶ 🚰 Sound 🚰 Prefabs 🚰 Scenes				
🚔 Scripts 🕨 🚔 Packages				

14. Select the SpawnManager in the Hierarchy. In the Inspector, add in the obstacles from the Prefabs folder to the **Obstacle Prefabs** array on the **Spawn Manager** component.

Inspector		<u></u> ==
SpawnManage	er	🗌 Static 🔻
Tag Untagged	‡ Layer Default	\$
▼人 Transform		💽 🗐 🖓 🔅
Position	X 9.90622 Y 4.817572 2	2 -4.914564
Rotation	X 0 Y 0 2	2 0
Scale	X 1 Y 1 Z	2 1
🔻 📾 🗹 Spawn Manage	r (Script)	💽 🗐 🖓 🔅
Script	🗟 SpawnManager	0
🔻 Obstacle Prefabs		
Size	4	
Element 0	🜍 Obstacle	0
Element 1	🜍 ObstacleBarrel	0
Element 2	🜍 ObstacleCrate	0
Element 3	🞯 ObstacleDoubleCrate	0
	Add Component	

15. Save the scene and press play. The new obstacles will now spawn



## Medium - Double Jump!

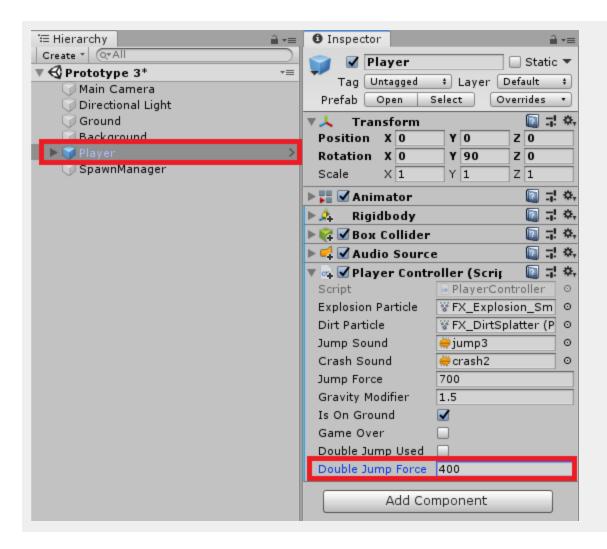
1. Navigate to the scripts folder and open up the **PlayerController.cs** file. By the variable declarations, add the following variables.

```
public bool doubleJumpUsed = false;
public float doubleJumpForce;
```

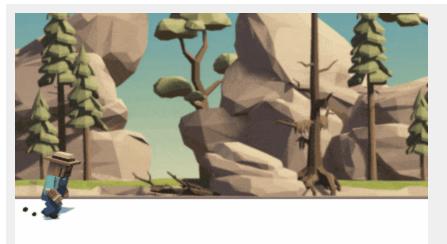
2. In the Update method, we first need to set the doubleJumpUsed boolean to false when the player does their first jump. We then check if the player has pressed the space key, doubleJump has not been used, and the player is in the air. If all of those checks pass, we will then add another force to the player like we did in the previous if statement and use the animation.Play() function to reset the jumping animation to the first frame. The updated Update method will look like this:

```
void Update()
{
  if (Input.GetKeyDown(KeyCode.Space) && isOnGround && !gameOver)
  {
    playerRb.AddForce(Vector3.up * jumpForce, ForceMode.Impulse);
    isOnGround = false;
    playerAnim.SetTrigger("Jump_trig");
    dirtParticle.Stop();
    playerAudio.PlayOneShot(jumpSound, 1.0f);
    doubleJumpUsed = false;
  }
  else if(Input.GetKeyDown(KeyCode.Space) && !isOnGround && !doubleJumpUsed)
    doubleJumpUsed = true;
    playerRb.AddForce(Vector3.up * doubleJumpForce, ForceMode.Impulse);
    playerAnim.Play("Running_Jump", 3, 0f);
    playerAudio.PlayOneShot(jumpSound, 1.0f);
 }
}
```

3. Save the script and head back to Unity. In the Hierarchy, select the *Player* GameObject. In the Inspector for the *Player*, adjust the **Double Jump Force** value on the **Player Controller (script)** component. Try playtesting with different values to find one that you like.



4. Save the scene and test the game. You can now press the spacebar twice to get a higher jump.



## Hard - Dash Ability

1. Navigate to the Scripts folder and open up the **PlayerController.cs** file. Below the variable declarations, add a new variable.

```
public bool doubleSpeed = false;
```

2. Next we will set this boolean within the **Update** method. After the jump code, add the following:

```
if(Input.GetKey(KeyCode.LeftShift))
{
    doubleSpeed = true;
    playerAnim.SetFloat("Speed_Multiplier", 2.0f);
}
else if (doubleSpeed)
{
    doubleSpeed = false;
    playerAnim.SetFloat("Speed_Multiplier", 1.0f);
}
```

This piece of code will check if the user has pressed down the left shift. If they have, it will set the doubleSpeed boolean to true, otherwise the boolean will be false. We also set the animator parameter *Speed\_Multiplier* according to if *doubleSpeed* is true or not. This parameter will be set up a bit later on.

Save the script and head back to Unity.

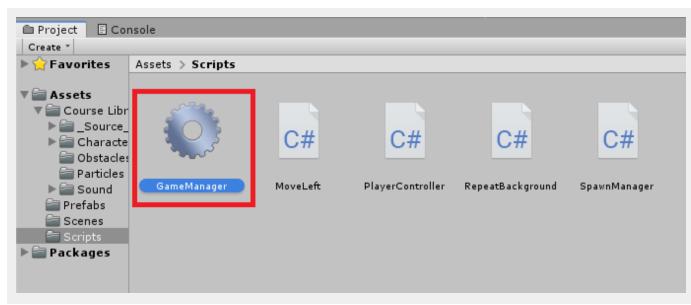
 Open up the MoveLeft.cs script. We will now tell the obstacles if the player is moving at double speed. We already have a variable for the player so all we need to do is to check the boolean we just created. Adjust the Update method to look like this.

```
void Update()
{
    if (playerControllerScript.gameOver == false)
    {
        if (playerControllerScript.doubleSpeed)
        {
            transform.Translate(Vector3.left * Time.deltaTime * (speed * 2));
        }
        else
        {
            transform.Translate(Vector3.left * Time.deltaTime * speed);
        }
    }
}
```

```
if (transform.position.x < leftBound && gameObject.CompareTag("Obstacle"))
{
    Destroy(gameObject);
}
</pre>
```

Save the script and head back to Unity.

4. Create a new script in the Scripts folder by either selecting Assets > Create > C# Script or right-clicking in the project view and selecting Create > C# script. Name the new script GameManager. Double-click the new script to open it.



5. Inside the class declaration, add two new variables. The first will store our score value, the second will be used to reference our player.

```
public float score;
private PlayerController playerControllerScript;
```

6. Next, inside the **Start** method, set the reference for the player and ensure the value of score is 0.

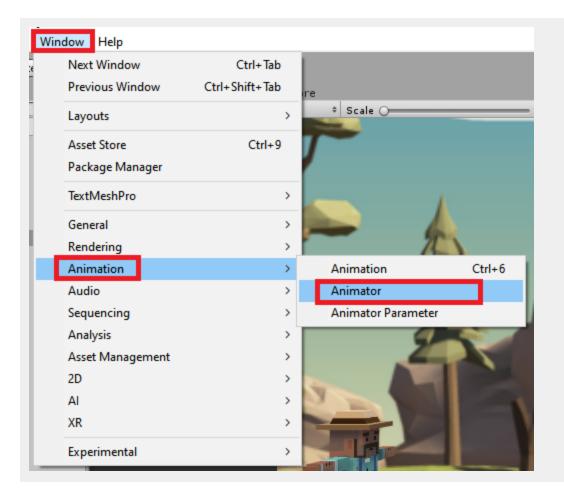
```
void Start()
{
    playerControllerScript =
GameObject.Find("Player").GetComponent<PlayerController>();
    score = 0;
}
```

7. The last thing we need to do in this script, is set up the **Update** method. We want to check if the game is over, if it isn't we will add to the score based on how fast the player is moving. We will also output to the console the current score.

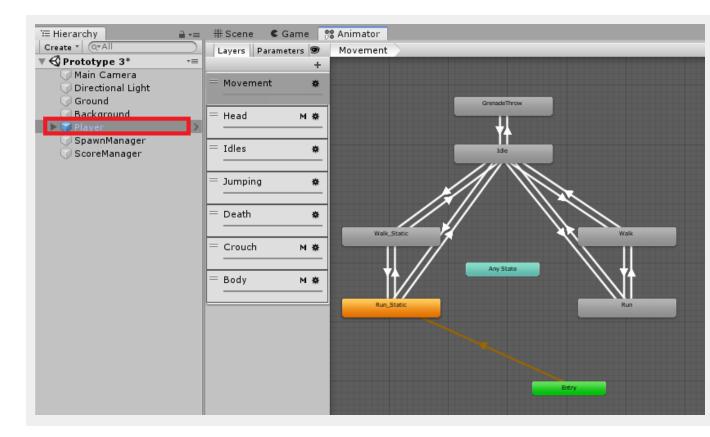
```
void Update()
{
    if(!playerControllerScript.gameOver)
    {
        if(playerControllerScript.doubleSpeed)
        {
            score += 2;
        }
        else
        {
            score++;
        }
        Debug.Log("Score: " + score);
    }
}
```

Save the script and head back to Unity.

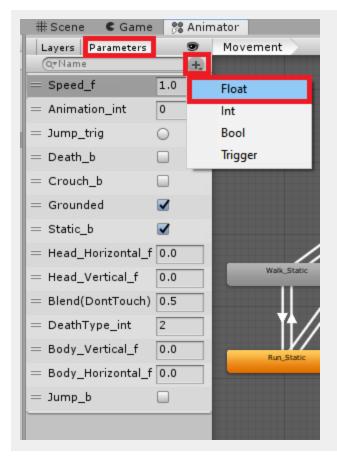
8. We now need to adjust the Player's Animator. Go to **Window > Animation > Animator** to open the animator window.



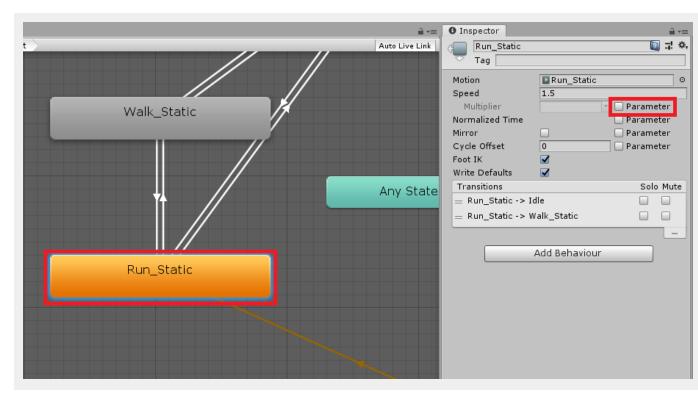
9. Once the window is open, dock it somewhere easy to access. Then select the player in the hierarchy



10. Inside the Animator window, select the parameters button. Next select the + button and select **Float**. Name the new parameter *Speed\_Multiplier*. Set it's default value to **1.0**.



11. Next select the **Run\_Static** state in the Animator window. In the Inspector, by the **Multiplier** parameter, check the **Parameter** checkbox.



12. Once checked, you can then select the Speed\_Multiplier parameter from the dropdown list.

Walk_Static       Tag         Walk_Static       Speed         Motion       Run_Static         Speed       1.5         Multiplier       Speed_f         Normalized Time       Speed_f         Head_Horizontal_f       Head_Vertical_f         Foot IK       Image: Speed_f         Write Defaults       Image: Speed_f         Run_Static -> Idle       Blend(DontTouch)         Body_Horizontal_f       Speed_Multiplier		<b>a</b> •≡	Inspector			<u>-</u> -≡
Walk_Static     Walk_Static     Speed     Normalized Time   Normalized Time   Speed_f   Head_Horizontal_f   Cycle Offset   Foot IK   Write Defaults   Transitions   = Run_Static -> Idle   = Run_Static -> Idle   = Run_Static -> Walk_S     Speed_f   Head_Horizontal_f   Blend(DontTouch)   Body_Horizontal_f   Speed_Multiplier		Auto Live Link			i∎ ‡	¦ ¢,
Any       Normalized Time       Speed_f         Head_Horizontal_f       Head_Vertical_f         Blend(DontTouch)       Body_Vertical_f         Body_Vertical_f       Body_Vertical_f         Body_Horizontal_f       Body_Horizontal_f         Speed_Multiplier       Speed_Multiplier	Walk Static		Speed	1.5		_ ⊙ 
Run_Static		Any	Mirror Cycle Offset Foot IK Write Defaults Transitions = Run_Static -> In	<b>✓</b> ✓	Head_Horizontal_f Head_Vertical_f Blend(DontTouch) Body_Vertical_f Body_Horizontal_f Speed_Multiplier	

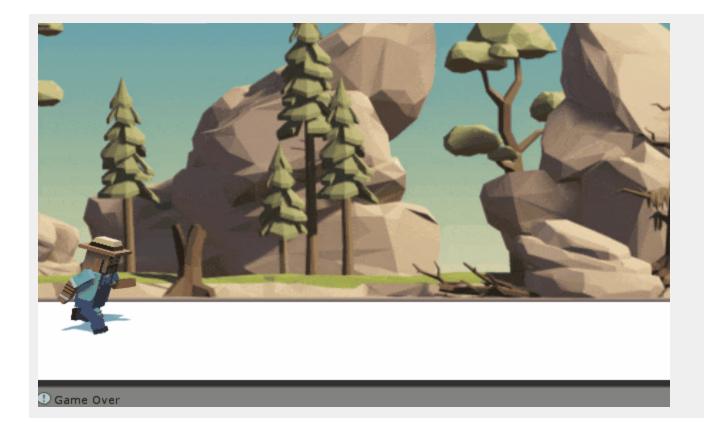
13. In the Hierarchy, create a new empty GameObject by either clicking **Create > Create Empty** or right-clicking and selecting **Create Empty**. Name the new GameObject *GameManager*.

≔ Hierarchy Create ▼	QTAIL
Create Empty	
Create Empty Child	
3D Object	>
2D Object	>
Effects	>
Light	>
Audio	>
Video	>
UI	>
Camera	

14. In the Inspector for the *GameManager*, click **Add Component** and search for **GameManager**, select the **GameManager** script to add it to the GameObject.

Inspe	ctor						<u>∩</u> •≡
	GameMana	ager				Sta	tic 🔻
Tag	Untagged		🕴 Laye	r Defau	ılt		\$
▼ 🙏 — т	ransform						다 후
Position	ו	X 0	Y	0	Z	0	
Rotatio	n	X 0	Y	0	Z	0	
Scale		X 1	Υ	1	Z	1	
	Ļ	Add Cor	nponent	:			
Q Game 🛞							
Search							
Game Manager							
	New script >						

15. Save the scene and press play. When you hold the shift key, the player will run faster and the objects will appear quicker!



### **Expert - Game Start Animation**

1. Navigate to the Scripts folder and open up the **GameManager.cs** script. By the variable declarations, add the following variables:

```
public Transform startingPoint;
public float lerpSpeed;
```

2. Update the **Start** method to look like the following:

```
void Start()
{
    playerControllerScript =
GameObject.Find("Player").GetComponent<PlayerController>();
    score = 0;

    playerControllerScript.gameOver = true;
    StartCoroutine(PlayIntro());
}
```

3. Below the **Update** method, create a new method called **PlayIntro**. This new method will move the player to the starting point.

```
IEnumerator PlayIntro()
{
    Vector3 startPos = playerControllerScript.transform.position;
    Vector3 endPos = startingPoint.position;
    float journeyLength = Vector3.Distance(startPos, endPos);
    float startTime = Time.time;
    float distanceCovered = (Time.time - startTime) * lerpSpeed;
    float fractionOfJourney = distanceCovered / journeyLength;
    playerControllerScript.GetComponent<Animator>().SetFloat("Speed_Multiplier",
0.5f);
    while (fractionOfJourney < 1)</pre>
    {
        distanceCovered = (Time.time - startTime) * lerpSpeed;
        fractionOfJourney = distanceCovered / journeyLength;
        playerControllerScript.transform.position = Vector3.Lerp(startPos, endPos,
fractionOfJourney);
        yield return null;
```

```
}
playerControllerScript.GetComponent<Animator>().SetFloat("Speed_Multiplier",
1.0f);
playerControllerScript.gameOver = false;
}
```

Let's break this method down. First we determine what the start and end positions are for our movement. Next, we determine what the length of the journey is. After that, we determine the distance we have covered so far, and what the fraction of the distance over the journey length is. We then do that calculation within a while loop to move the player forwards.

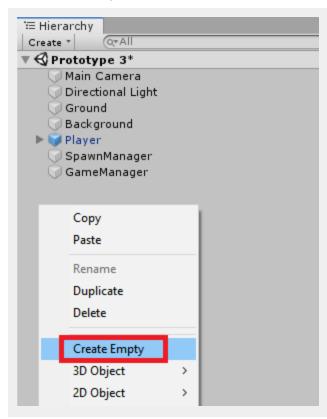
We can adjust the value of the lerpSpeed variable to determine how fast the movement happens.

Within this method we are also adjusting the speed of the player's animation using the Speed\_Multiplier variable we created earlier.

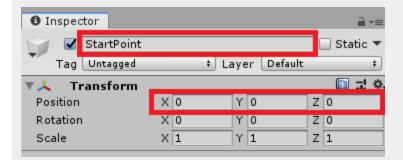
After the movement has been completed, we set the gameOver variable to false to allow the player to move.

Save the script and head back to Unity.

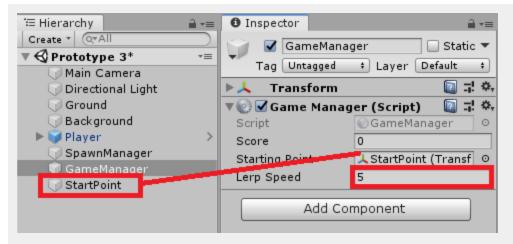
4. In the Hierarchy, right-click and select **Create Empty**.



5. Select the new GameObject, rename it to **StartPoint**. Adjust the **Position** of the StartPoint to **0,0,0**.



6. We will now set up the variables for the GameManager. In the Hierarchy, select the **GameManager**. Drag the **StartPoint** from the Hierarchy into the **Starting Point** field. Adjust the **Lerp Speed** variable to be **5**.



7. The last thing we need to do is adjust the players position. Select the Player in the Hierarchy. Adjust its position to -5,0,0.

O Inspector 🔒 +≡										
🍟 🗹 Player						Static	•			
👘 Tag 🗍	ag Untagged 🕴 Layer					Default ‡				
Prefab Open Select Ov						rides	•			
▼ 🙏 Tra <u>nsform</u> 🔯 式 🔅										
Position	Χ-	-5	Y	0	Z	0				
Rotation	X	D	Y	90	z	0				
Scale	X	L	Y	1	Z	1				

8. Save the scene and press Play. The player will now run in from the left before the game starts.

