# 3D Infinite Runner for Playmaker

Documentation

---

Thank you for purchasing the Playmaker 3D Infinite Runner Kit! Please make sure you have Playmaker installed, as this kit will not function without it. Please remember to leave a rating on the Unity Asset Store if you've found this kit useful.

Feel free to contact us if you have any questions or suggestions either by using the contact form on [3dsauce.com](3dsauce.com), or by email: [support@3dsauce.com](mailto:support@3dsauce.com)

## ***Contents***

# 1 - Getting Started

**Once you have installed both Playmaker and this kit, you must then import the included package called "IMPORT THIS After Playmaker.unitypackage".** You can now load the "3D Infinite Runner" example scene. If you experience any issues see note below. You will find fully commented logic within the FSMs inside the Playmaker Editor. Arraymaker, DOTween and Animator actions will be installed automatically when you import the extra package. This kit has been tested and is working as intended in Playmaker 1.9.0 and Unity 2018.4.6.

**NOTE:** In a few rare circumstances the project settings and global variables may not import properly. If you are experiencing strange behaviors or console errors after import, try the following solutions.

**1.** Select the "PlaymakerGlobals_EXPORTED" file in the 3D Infinite Runner directory. In the inspector press the "Import Globals" Button.

**2.** Go to "Edit / Project Settings / Physics." In the inspector set the Gravity Y value to "-30"

**For assistance in putting your own project together and customizing the look and feel of the game,** please refer to the following guidelines.

## 2 - Customization

### Enabling or disabling the swipe-touch mobile controls:

The swipe and touch controls are fully contained inside the “TouchCamera” gameobject in your scene hierarchy. Enable the gameobject if you require the mobile controls, disable it if you don’t need touch controls. If you removed it by accident you can simply re-add the “TouchCamera” prefab to the scene.

### Swapping the character models in Editor:

**Note:** You must use a mecanim ready model for this to work. Be sure to follow these steps exactly.

1. Inside the “PlayerController / CharacterMesh_TriggerCollider” prefab in your scene hierarchy, you will find a child gameobject called “PlayerModel.” Drag your new player model mesh into the “CharacterMesh_Trig- gerCollider” so that is at the same hierarchy level as the old “PlayerModel.”

2. In your new mecanim model you should see an animator component. Find and drag the “PlayerAnimController” from your project view into the empty “controller” slot on the animator component. Be sure to disable “Apply Root Motion” if it is checkmarked.
3. Copy the two playmaker fsm components and the audio source component from the old “PlayerModel” to your new model. You can do this by right clicking each component and selecting “Copy Component,” then in the new model right click in the inspector and select “Paste Component As New.” Once you have done this for all three components you can move on.

4. Make sure your new model is positioned directly over the old player model, in my case I had to change the Y position value to “-1.”

5. Rename your new model to “PlayerModel” and delete the old “PlayerModel.” Make sure you name it exactly like this, or you will run into problems.

6. Finally, make sure to apply the new texture in a material which uses the “curved” shader.

## Changing prop models:

Swapping the mesh in each prop prefab with your own is simple. Place a prefab in your scene and find the “Mesh” subobject of the prefab. on the mesh gameobject you will find a “Mesh Filter” component, drag your new model into the mesh slot. Make sure you apply the prefab and save the scene afterwards. All that remains is to replace the texture with your new corresponding texture in the material that is assigned to that mesh.

## Creating track chunks:

Begin by enabling the “PlacementGuide” gameobject in your scene hierarchy and centering on it in scene view by pressing the “F” key. You can start a chunk from scratch or simply duplicate a previous chunk and start working on it from there. If you want to create one from scratch, follow these steps:

Create an empty gameobject, move it inside the “ChunkManager” gameobject in the scene hierarchy, zero out it’s XYZ positions in the inspector and give it a chunk name. Drag a “ChunkBase” prefab into your newly created chunk. Continue dragging prefabs from the props directory into the newly created chunk and placing them until your chunk is laid out as desired.

**Tip:** Use the “PlacementGuide” object to snap prefabs and make item placement more accurate.

**Tip:** Instead of placing walls manually from scratch every time, I recommend duplicating the walls from a previous chunk and moving them into the new one.

**Note:** When putting obstacles in extremely close proximity you may encounter a bug where using a forgive key only gets rid of the first layer and leaves you instantly dead again. If you have 2 obstacles in close proximity that you wish to combine so that both of them get disabled when using a forgive key, simply drag the mesh and colliders of one obstacle into the other. You can delete the remaining empty object. Keep in mind that you will lose the prefab instance when doing this, so updates to the prefab won’t apply to this any longer. You can however make your own prefabs of these combined obstacles if you use them frequently.

**Starting chunk:** The “ChunkStart” gameobject in your scene hierarchy is the first chunk that the player will spawn in at the beginning of every single round. You can edit this chunk the same as the others, just keep in mind that it will never change from round to round.

## Swapping the character models at Runtime:

**Note:** You can see a working example of this by checking out the “Character Swap” FSM on the “Button_Swap” gameobject in the example scene.

Adding additional skins and meshes to swap between is as simple as adding additional player models as children to the “CharacterMesh_TriggerCollider” gameobject, and adding switches as needed to the Swap Button. You can find an example of this already within the example scene.

## Customizing sound effects:

The majority of sounds can be changed by using the “Sound Manager” FSM on the “SoundManager” gameobject in your scene hierarchy. There are a few exceptions to this which I will cover as well.

Begin by simply dragging each of your new sound effects into the desired slot within the “SoundManager” FSM component in the inspector. This will auto set these sounds when the level starts and no additional steps are required for these sounds to work. A few exceptions do apply as follows.

1. The first exception would be the coin prefab found in the Prefabs/Props directory. Select the coin prefab in the project window and observe the exposed audio clip variable within the Playmaker FSM component in the inspector. Simply drag your coin collection sound effect into the “Sound_CoinPinkup” slot and that is it.

2. The particle system prefabs in the Prefab/Particles directory also require special treatment. In each of the two particle systems see the corresponding FSM component in the inspector. Drag your new sound effects into the exposed audio clip slots as you did for the coin prefab.

3. In your scene hierarchy find the “Button_ForgiveKey” game object. In the Playmaker component you will again find an exposed audio clip variable called “Sound_FailedResume.” Simply drag your sound here and you are done.

## Customizing particle meshes:

You will probably want to change the particle meshes to accommodate whatever aesthetic you're going for. To do this select each “Particle System” component within the prefabs found in the “Prefab/Particles” directory. Scroll down and expand the renderer section. Simply replace the mesh with your own and make sure it is using a material with your texture and the curve shader applied.

## Unlit curve shader limitations:

In order to use the world curve shader you must be using unlit artwork and mesh based particle systems without opacity. The shader doesn't support shadows or lighting, if you need this functionality I recommend switching to another shader, however you will lose the world curving functionality. There are a few powerful curved world shaders available on the asset store that support these extra features if needed.

## Adding additional world curvature materials:

If you didn't atlas your meshes and textures, you may need to use more than the four materials included with the kit. Once you create a new material and apply the curve shader to it, you will have to follow a few additional steps to have the material curvature animate along with the rest of the world.

Navigate to the Curve World FSM in the "TrackCurveManager" game object. In the "Inactive" state, copy and paste one of the "Set Material Vector2 Floats" actions. Drag your newly created curve material into the Material slot of the newly pasted action. Now do the same in both the "Wait" and "Translate" states. Your new models should now animate properly.

## Adjusting powerup spawn probability:

The locations which power ups spawn are specified by placing the “PickupZone” prefab in strategic locations in each chunk. The probability of a powerup spawning on any given pickup zone is controlled by the “SpawnChancePercent” variable which you can find exposed on the Pickup Randomizer FSM component on the PickupZone prefab. Simply change this number to reflect the desired chance. At 100 a powerup will spawn at every single pickup zone so long as at least one remains free in the “PickupPool” object in your scene hierarchy.

## Adjusting speed change intervals and amounts:

Find the “SpeedControlTrigger” gameobject in your scene hierarchy. In the Speed Manager Playmaker FSM component you will see a few exposed variables. Distance_1 and Distance_2 define the placement of the triggers which will cause the game to speed up. Speed_1 and Speed_2 will define the speed at which the player will travel when it reaches the given distances.

“InitialSpeed” will determine the speed at which the player runs when starting a new round.

## Enabling the coin multiplier:

You may wish to enable a coin multiplier in your game that allows the player to multiply every coin picked up by a specified amount. To do this, navigate to the “GUICoinManager” in the scene hierarchy. In the Coin Manager FSM within the inspector, you will find an exposed variable called “CoinMultiplierINT.” Simply change this to whatever amount you want coins to be multiplied by. If the player purchased a coin doubler, you would simply have to set this Int value to 2 when the scene is started.

---

**If you’ve enjoyed the package and would like to encourage updates and new templates, please remember to rate it and leave a review on the Unity Asset Store.**

**The secret’s in the sauce!**

**3dsauce.com**