Devils Pack PBR

VISIT WWW.INFINITYPBR.COM FOR THE LATEST UPDATES, FORUMS & MORE ASSETS.

** If you have any problems getting set up or using our models, please use the Unity forums to contact us and we'll get it sorted.*

** If you enjoy using our models,* ***please write a review*** *on the Unity Asset Store so other developers know how cool our stuff is :D*

1. Introduction

"Devils Pack PBR" is a 3d PBR Character Model designed for video games developers. The procedural aspect means there are virtually unlimited looks you can give to the monster, creating unique looks that no one else has. Physically Based Rendering means the looks can appear hyper realistic.

Due to all of this, there is a little setup involved. It shouldn't take long and maybe it'll be quite fun, as you'll get to fine-tune the look of your Devils.

In most cases the Quick Set Up section will be all that you need. If you're interested in knowing more about each of the values you're able to tweak, check out the Procedural Values section.

For advanced users, if you're interested in scripting run time changes in the texture of the model, refer to the Scripting section.

We plan on updating our assets periodically, so please check the Asset Store for available updates.

2. Quick Set Up

This quick guide will work for most users, and does not allow for run time changes in the look of the textures. *For videos, please visit our website at www.InfinityPBR.com where you will find much more detailed examples.* ***We highly suggest you create your maps in a new, empty project.***

***** For the latest known Unity bugs, please visit www.InfinityPBR.com and log in! *****

This video goes through all of this in much better detail:
https://www.youtube.com/watch?v=cF1U7wZyaJc *<— WATCH IT :)*

1. Load the Assets/SFBayStudios/SFB Demo Scenes/SFB Devils Texture Creation.unity scene
2. For each of the "CT2T - XYZ" objects, select the source material in the project, which you can find here: Assets/SFBayStudios/SFB Devils/Procedural Materials/ — Be sure to check the "Enable Texture Modification" box under the top "Main" section. Also make any adjustments to the Environment options, such as Ground Dirt, Damage etc.
3. In the scene view, select "CT2T" object with the source you just modified. This editor script (see Inspector) will copy the settings from the source to all the listed sections in the target materials. You can add or remove them as you'd like, but if you plan on making a custom Copy This To That controller, I suggest you create a new object, and keep the demo defaults as is.
4. Click the "Copy Settings" button, and wait until all materials are finished updating. You'll see them update in the scene view, but it may take a few moments.
5. Do the same for the other Copy This to That controllers.
6. If you want to make any more modifications to individual parts or materials, do so now. If you click "Copy Settings" again, any changes you make will be lost.
7. When you are satisfied with the look of your model, select the "Mass Exporter" object. In the inspector, confirm that all of the Procedural Materials are seen in the "Substances" array. Give your new character a descriptive name, such as "Blue #1", and choose whether to remove Emissive & Height maps. If you are going to share maps with a previous export, add the "Previous Group Name" and select which maps to share. If this is your first export, or you've changed so much of the look that you can't share maps, uncheck the boxes. Finally, click "Export Materials." It will take a few minutes for Unity to import all the .tga files that are created. There are three objects, separated into groups of materials, and one that has all of the materials.
8. When complete, you'll find your game-ready materials in Assets/SFBayStudios/Exported Materials/ [Group Name]
9. Don't forget to choose the correct LOD for your game, and play with the size settings of the textures to optimize their system resource usage.

* Masking the base texture will remove the Normal & AO maps for this part of the object. This could be used to make a more flat, cartoonish look.

Hey.... did you watch that YouTube video I linked above? You really should, if you're having trouble getting the hang of this... https://www.youtube.com/watch?v=cF1U7wZyaJc :)

3. Procedural Values

The included Run Time procedural materials are designed to be used in game. They start with "SFB_RT_". You'll find some basic scripting guidelines here, but we encourage you to use the Forums on the Unity website if you have more questions, as we aren't the best coders.

SFB_RT_TextureBlend: Use this to blend between two textures. It accepts two groups of exported maps, and has a float value to blend between the two. One potential use is to blend between a "clean" and "damaged" version of an object. Perhaps the more it is used, the more the "Damaged" version is shown.

SFB_RT_DirectionalBlood: Bring in your ready-to-go texture maps, along with the Positional Map for the object (included in the package). Choose the direction appropriate for your object, and then you can code the material to update in script. This could be used to add blood to an object whenever it's "used", and the blood can fade out over time.

SFB_RT_TextureBlend

Category	Name	ID | Type Min,Max	Description
N/A	Blend Amount	`blendAmount` `float (0.0,1.0)`	Blend amount between Texture group 1 and Texture group 2.

SFB_RT_DirectionalBlood

Category	Name	ID | Type Min,Max	Description
N/A	Axis	`axis` `int (1,6)`	Which direction should be blood go? X, X-inverted, Y, Y-inverted, Z, Z-inverted
Blood	Height	`bloodHeight` `float (0.0,1.0)`	How high does the blood extend
	Level	`bloodLevel` `float (0.0,1.0)`	How "thick" is the blood, works in conjunction with height
	Contrast	`bloodContrast` `float (0.0,1.0)`	The contrast of the pattern
	Color	`bloodColor` `Color()`	Color of the blood
	Roughness	`bloodRoughness` `float (0.0,1.0)`	How reflective the blood is, suggested range 0.25-0.8

SFB_RT_SnowIceMoss

Category	Name	ID | Type Min,Max	Description
SFX	Water Level	SFXWaterLevel float (0.0,1.0)	How much water
	Water Details	SFXWaterDetails float (0.0,1.0)	Details of the water
	Refraction	SFXRefraction float (0.0,1.0)	Refraction of the water. (Try 0 for an "ooze" look)
	Reflection	SFXReflection float (0.0,1.0)	Reflection amount
	Reflection Distance	SFXReflectionDistance float (0.0,1.0)	Reflection distance
	Flow Direction	SFXFlowDirection float (0.0,1.0)	Changes the direction the water appears to be flowing
	Water Color	SFXWaterColor Color	Color of the water (clear = default)
	Ice	SFXIce float (0.0,1.0)	How much of the water has turned to ice? Water is required for this to work.
	Ice Details	SFXIceDetails float (0.0,1.0)	Details of the ice
	Snow	SFXSnow float (0.0,1.0)	Amount of snow
	Moss	SFXMoss float (0.0,1.0)	Amount of moss
	Moss Scale	SFXMossScale int (1,4)	Scale of the moss texture
	Moss Color	SFXMossColor Color	Color of the moss

4. Scripting

It's possible to change values during run time. We include a few versions of the material, some of which are optimized for common run-time options. In those cases, you'll likely want to bake maps for the base materials you plan on using (which do not change at run time), and use the optimized versions. This will speed up the changes in game.

Please Note: We are not the best coders. There may be more ways of doing what we're doing, perhaps better ways. Please use the forums on our site and the Unity forums if you'd like to discuss or ask the community about various ways of doing this. ***We are also using Unity Script because, simply, it's what we currently understand.*** *Check out our demo scripts for more extensive examples.*

```
var substance                : ProceduralMaterial;

// Set an Int or a Float value
substance.SetProceduralFloat("Grunge2Volume", 0.5);

// Set a Color value
substance.SetProceduralColor("Grunge2Color", Color(1,1,1,1));

// Get a Vector2 value
var currentOffset   : Vector2 = substance.GetProceduralVector("Grunge2Offset");

// Set a Vector2 value
substance.SetProceduralVector("Grunge2Offset", Vector2(currentOffset[0],currentOffset[1]));
```

5. Animations

We've included a sizable selection of animations designed specifically for this character. Read about them here. Many, if not all, can be looped in various ways inside Unity to create animation combinations. **You can find the animations by expanding the main model in the "x_Model" folder. Drag the animations into an Animator Controller to use them. Select the main model and click the "Animations" tab to add events.**

Animation	Looped?	Description
Idle	Yes	Idle Loop
Idle Break	No	Break from the idle loop
Walk	Yes	A powerful if slightly awkward hoof walk forward
WalBack	Yes	A powerful if slightly awkward hoof walk backward
Dodge	No	Dodges an attack
Got Hit	No	Reaction to being hit
Death	No	Death comes to even the Devil
Attack1	No	Two sword attacks
Attack2	No	Both swords stab forward
Attack3	No	Head Bash
Attack4	No	Twisting double sword attack
Cast	Yes	Reaches to the sky like He-Man
Taunt	No	Taunt animations
Scream	No	Scream forward
Jump Left	No	It's just a jump to the left
Jump Right	No	And then a jump to the right

6. Level Of Details

There are multiple level of details available. The full resolution is of course the best to use in close up views. However each of the levels could be better for when the character is further away, or when they're not as visible based on your game.

7. Change Log

v1	Initial Version