CTI Runtime Components HDRP 7.2.

About this documentation

In case you want to use CTI trees along with the HDRP you have to assign the *CTI HDRP* shaders and use the *CTI_SRP_CustomWind* script. Both shaders and script are slightly different from the CTI Runtime Components for the built in RP.

Table of Content

Limitations

- Shaders need Unity 2019.3. and HDRP 7.2. or above. Other versions have not been tested.
- Only basic LOD trees are supported. No texture arrays, no tessellation.
- The shaders only accept wind from script.
- You can not author billboard textures using HDRP nor can you use the debug shader, so authoring should take place using the built in RP.

Changes

- The leaf shader uses the **built in transmission lighting.** So you have to add/edit a diffusion profile.
- **Specular color** has been dropped. The shaders use the default value.
- The **leaf shader** uses a **regular normal** or bump map. Lighting uses the built in transmission feature. An **alternative leaf shader** is added: Check it out.
- The shaders only except **wind from script**. → *The Tree component is not needed at all and should be removed.*

- The CTI LOD HDRP shaders need a slightly different input for the **wind from script**.
  → You have to use the CTI_SRP_CustomWind script instead of the old one.
- **Fade out Wind** has been dropped.
- **Fade out Translucency** has been dropped.
- **Tumbling** and **Turbulence** have slightly been reworked and optimized. **Leaf Noise** has been added. *→ You may have to adjust their settings.*
- **Wind multipliers** for primary and secondary bending as well as edge fluttering have been added. *→ Now you can tweak the bending without editing the tree. Make sure multipliers in the bark material match those in the leaf material.*

CTI HDRP LOD Shaders

The shaders were authored using Shader Graph to be most accessible. Only the billboard shader is in HLSL as it needs functions not supported by shader graph.

Shader Graphs/CTI SG Bark HDRP

Shader Inputs

Surface Options

Surface Type Opaque

Rendering Pass *Should be Default.*

Alpha Clipping *Should be unchecked.*

Double-Sided *Should be unchecked.*

Exposed Properties

Color Variation (RGB) Strength (A) Color variation in RGB. Alpha contains the strength.

Albedo (RGB) Smoothness (A) Diffuse texture which contains **smoothness** (unlike the leaf shader which expects transparency) in the alpha channel.

Normal Map (GA) Occlusion (B) contains the combined normal and occlusion map. *Red color channel should be black.*

Normal Strength Lets you adjust the strength of the normal.

Smoothness Multiplier for the smoothness as sampled from the *Albedo (RGB) Smoothness (A)* map.

Wind Multipliers

X Multiplier for the Primary Strength. *Must match the value in the leaf material.*

Y Multiplier for the Secondary Strength. *Must match the value in the leaf material.*

Z Multiplier for Edge Flutter. *Does not matter here.*

Shader Graphs/CTI SG Leaves HDRP

Shader inputs

Surface Options

Surface Type Opaque

Double-Sided Please check if your leaf geometry is only single sided (recommended)

Normal Mode Should be set to *Flip*.

Exposed Properties

Color Variation (RGB) Strength (A) Color variation in RGB. Alpha contains strength.

Albedo (RGB) Alpha (A) Diffuse texture which contains transparency in the alpha channel.

Alpha Cutoff If the alpha channel of the Base texture contains different shades of gray instead of just black and white, you can manually determine the cutoff point by adjusting the slider.

Normal Map The Normal Map.

Normal Scale Scale of the normal.

AO (G) Translucency (B) Smoothness (A) contains the combined ambient occlusion, normal, translucency and smoothness map.

Smoothness Multiplier for the smoothness as sampled from the *Normal (GA) Smoothness (B) Trans (R)* map. If this map is disabled *Smoothness* defines the final smoothness value.

Thickness Lets you remap the thickness as sampled from the *AO (G) Translucency (B) Smoothness (A)* texture.

Wind Multipliers

X Multiplier for the Primary Strength. *Must match the value in the bark material.*

Y Multiplier for the Secondary Strength. *Must match the value in the bark material.*

Z Multiplier for Edge Flutter.

Tumble Strength defines the strength of the tumbling animation.

Tumble Frequency lets you adjust the frequency of the tumbling.

Leaf Turbulence lets you adjust the strength of the turbulence.

Leaf Noise lets you adjust the strength of the edge flutter (stored in vertex color green) affecting the leaf turbulence. Using edge flutter influence values above 0.0 will most likely add some distortion to the leaf meshes – which in fact looks really nice.

Shader Graphs/CTI SG Leaves VS Normals HDRP

This shader is like the one above but smoothes normals in view space. Doing so may give you better normals in case these are “heavily” smoothed to reveal the original shape of the tree, as

the shader will not simply flip or mirror them but takes them as they are and forces them to point towards the camera.

Due to the custom normal correction it is a bit more heavy than the regular leaf shader but i think when using HDRP and a depth prepass this is more or less negligible (my fps dropped from 56 to 55 when flooding the screen with 1000 mesh trees).

Both trees use smoothed normals (projected from a simple sphere). The left tree uses the VS Normals shader so its shape is a bit more readable and it shows up less noise in lighting. The right tree uses the standard leaf shader and flipped normals which makes it a bit more difficult to read the light direction.

Real time shadows however minimize the benefit a bit.

Special shader inputs

Surface Options

Double-Sided Should be checked of course :)

Normal Mode Should be set to ***None***. The shader will “flip” the normals automatically in view space to make them point towards the camera. This is what we want. Using built in *Flip* or *Mirror* here would corrupt the effect.

CTI HDPR/Billboard HLSL

As mentioned above this is the only HLSL shader as shader graph simply does not support the functionality needed to make it work (namely calculating custom UVs or color variation in the vertex shader which can't be done in the fragment shader when it comes to billboards).

Shader inputs

Surface Options

Surface Type Opaque

Double-Sided Must be checked!

Normal Mode Must be set to *Flip*!

Exposed Properties

Color Variation (RGB) Strength (A) Color variation in RGB. Alpha contains strength.

Albedo (RGB) Alpha/Occlusion (A) This slot should contain the created albedo texture atlas.

Alpha Cutoff If the alpha channel of the Base texture contains different shades of gray instead of just black and white, you can manually determine the cutoff point by adjusting the slider. A value of 0.45 should just be fine.

Normal (AG) Translucency (R) Smoothness (B) This slot should contain the created texture atlas.

Normal Scale Scale of the normal.

Smoothness Multiplier for the smoothness as sampled from the *Normal (AG) Translucency (R) Smoothness (B)* map. If this map is disabled *Smoothness* defines the final smoothness value.

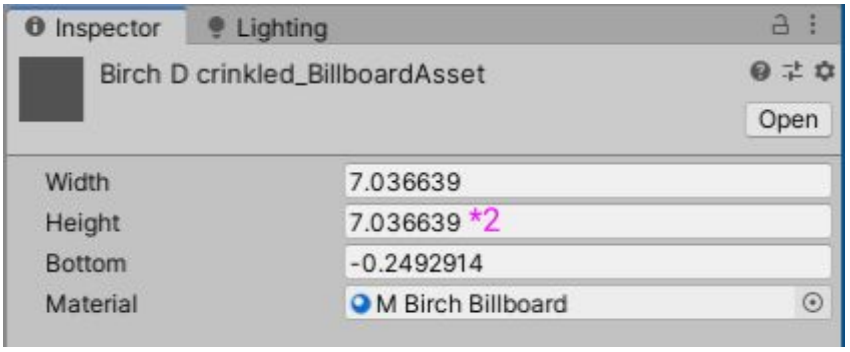
Wind Strength As Billboards do not have any baked wind information you may use this parameter to make the bending of the billboard better match the bending of the mesh tree.

Wind Power Power value which drives the wind strength along the y axis. Should match the power value used on importing the tree. Default is 1.5.

Thickness Remap Lets you remap the thickness as sampled from the translucency channel. Nevertheless the bark will never get fully opaque when using the built in Foliage diffusion profile… *Consider creating a diffusion profile just for billboards.*

Vertical Scale Actually CTI contains a bug: Billboard assets only have half the height they need (so they are most likely as high as they are wide) and for some reason i can't remember i never fixed this on the asset but simply added a fixed multiplier of 2 in the shader. While this is visually correct it might cause issues if the billboard leaves the screen at the lower border as it will be culled too early.
So with this factor exposed you can fix the billboards:

- Edit die billboard asset and add "*2" at the end of its height value. Unity will calculate the proper final height which in this case is 14.07328.



- Then edit the billboard material and set *Vertical Scale* to 1.