Offline Render

Overview

Legacy plugin included in bundle but not supported or **maintained**, if you need custom support please contact us.

Offline Render is an easy to use, realtime capture plugin for Unity. It allows you to capture the game view to a multi-channel OpenEXR or multiple PNG files, supporting not just the final output image, but also some common elements and other G-Buffers. Offline Render supports 360 mono, stereo and sbs rendering and output to MP4 for both offline and realtime capture.

Offline Render allows you to render your Unity scenes and take them into a standard post-production pipeline using your favorite compositing software.

Changes in 3.0

- Support for HDRP 7.x
- Rewrite of Offline Render Core
- Support for all HDRP passes
- Support for 360 mono, stereo and sbs
- Support for Object ID
- Deprecated Standard Pipeline

Limitations

- Requires Unity 2019.3.x
- Requires HDRP 7.x
- Windows 64-bits only

How to use

After importing the package to you project, you can add Offline Render component to your main camera or other cameras in the scene for multi-camera render. You can it via Add Component and writing “Offline Render” or “Offline Render HD” (depending on the version used), you should be presented with an UI like the one in Figure 1.

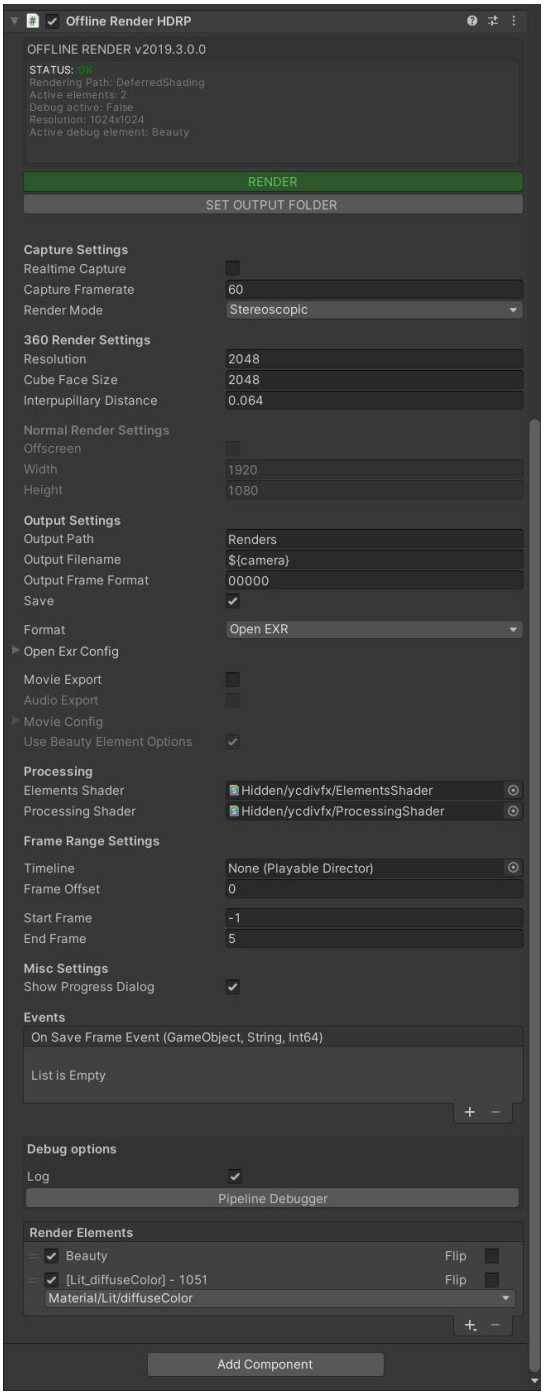

Figure 1 – Offline Render (HDSRP version) component.

Options

- Realtime Capture: Enables realtime game view capture (this setting will only generate the MP4 file);
- Capture framerate: Framerate to capture;
- Render Mode: Allows to set the type of output, Normal, 360 Monoscopic or 360 Stereoscopic;
- Resolution: Width size for 360 equirectangular output;
- Cube Face Size: The size for the cube faces that will generate the final 360 output;
- Interpupillary Distance: Distance between the eyes for stereo render;
- Offscreen: Renders the game view offscreen allowing you to render at higher resolutions;
- Width/Height: Resolution for offscreen output;
- Output path: location where the OpenEXR, will be saved;
- Output filename: is the base filename, for the saved frames, and accepts the token ${camera}, that corresponds to the current rendering camera;
- Output Frame Format: Controls the number padding and format of the output filename. It uses .NET formatting, check https://docs.microsoft.com/en-us/dotnet/standard/base-types/formatting-types;
- Save: Enable to save the frames when capturing;
- Format: Select either output to multi-layer EXR or multiple PNG files;
- Format Config: Options for the select file format;
- Movie Export: Outputs a MP4 file;
- Audio Export: Outputs audio into MP4 file (experimental);
- Audio/Video Config: Options for the Audio and Video output;
- Timeline: Uses the timeline range to control start and stop frames;
- Frame Offset: Controls the offset added to the frame numbering;
- Start: the start frame for the capture, when -1 this parameter is ignored;
- End: the end frame for the capture, when -1 this paramenter is ignored. Important, this determines when the script stops the editor play mode
- Elements: Render elements to capture. After setting all the options that fit your needs, and if there are no errors displayed, press Render button to start the capture process. If an end frame was set, the play mode will disable when this frame is reached.

Segmentation Pass Instructions

The new segmentation pass allows great flexibility, just add the Segmentation Manager (Figure 1) component, to a root of a geometry prefab that you want to capture and select one of the modes.

Figure 2 – Segmentation Manager component

If you want to override any given child object, it's possible to add the Segmentation Category component to that object and pick the color wanted to output. Then in the segmentation manager, enable "Allow overrides".

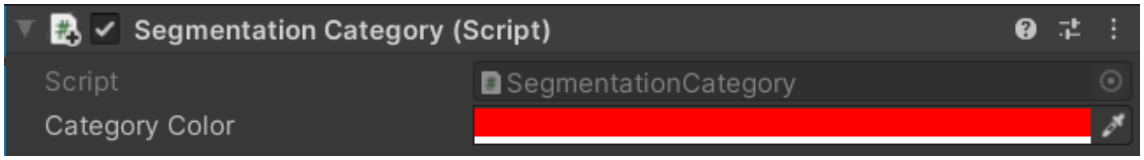

Figure 3 – Segmentation Category component

Don't forget to add the Segmentation Pass to OfflineRender to capture.

Release history

1.0 – Initial release

1.3 – Added developer API and documentation for creating custom passes

1.4 – Added support for ObjectID pass

1.5 – Added 360 mode

1.6 – Added support for MP4 output

1.7 – Added PNG file output, option to capture UI, API changes.

1.8 – Small fixes

1.9 – Updated UI; New Native plugin for handling conversion H264 codec options; Handle of some edge cases for some RenderTexture formats

2.0 – Check document for changes

2.6 – Support for the HDSRP

3.0 – Support for Unity 2019.3 and HDRP 7.x