Simple HDR Sample (DirectX 11)

# *This sample is compatible with the Windows 10 April 2018 Update SDK (17134)*

# Description

This sample renders an HDR scene with values higher than 1.0f, which will be displayed as brighter than white on a UHD capable monitor with a supported video system (or tonemapped on standard displays). The goal of the sample is to show which APIs to use, how the HDR swapchain should be created, and how different values larger than 1.0f will look on a UHD TV.





# Using the sample

The sample uses the following controls.

|  |  |  |
| --- | --- | --- |
| Action | Gamepad | Keyboard |
| Toggle displaying ST.2084 curve | A button | Space |
| Toggle displaying only paper white block | B button | Enter |
| Adjust brightness of paper white | D-Pad | + and - |
| Adjust values | Left/Right thumb stick | Up and Down  (slowly with Shift) |
| Exit | View button | Esc |

# Implementation notes

A very simple HDR scene, with values above 1.0f, is rendered to a FP16 backbuffer. If on a HDR-capable system, it produces an HDR signal otherwise a tonemapped SDR signal is generated in the swapchain.

This sample uses the [DeviceResources](https://github.com/Microsoft/DirectXTK/wiki/DeviceResources) class which supports HDR-capable swapchains.

Up to now, games were outputting and SDR signal using Rec.709 color primaries and Rec.709 gamma curve. One new feature of UHD displays is a wider color gamut (WCG). To use this, we need to use a new color space, Rec.2020 color primaries. Another new feature of UHD displays is high dynamic range (HDR). To use this, we need to use a different curve, the ST.2084 curve. Therefore, to output an HDR signal, it needs to use Rec.2020 color primaries with ST.2084 curve.

For displaying the SDR signal, a simple tonemapping shader is applied to simply clip all values above 1.0f in the HDR scene, and outputs 8bit values using Rec.709 color primaries. See the [PostProcess](https://github.com/Microsoft/DirectXTK/wiki/PostProcess) class in *DirectX Tool Kit for DirectX 11* for additional tone mapping operators.

For displaying the HDR signal, a shader is used to rotate the Rec.709 color primaries to Rec.2020 color primaries, and then apply the ST.2084 curve to output 10bit values which the HDR display can correctly display. The whiteness and brightness of the output on an HDR display will be determined by the selected nits value for defining “paper white”. SDR specs define “paper white” as 80nits, but this is for a cinema with a dark environment. Consumers today are used to much brighter whites, e.g. ~550 nits for a smartphone(so that it can be viewed in sunlight), 200-300 nits for a PC monitor, 120-150 nits for an SDR TV, etc. The nits for “paper white” can be adjusted in the sample. Displaying bright values next to white can be deceiving to the eye, so you can use the A button to toggle if you only want to see the “paper white” block.

The sample has two modes:

* Render blocks with specific values in the scene
* Render ST.2084 curve with specific brightness values (nits)

# Known issues

This sample does not implement HDR on UWP on Xbox One

# Update history

Initial release October 2017

# Privacy Statement

When compiling and running a sample, the file name of the sample executable will be sent to Microsoft to help track sample usage. To opt-out of this data collection, you can remove the block of code in Main.cpp labeled “Sample Usage Telemetry”.

For more information about Microsoft’s privacy policies in general, see the [Microsoft Privacy Statement](https://privacy.microsoft.com/en-us/privacystatement/).