HDR10 Sample

*This sample is compatible with the Microsoft Game Development Kit (June 2020)*

# Description

This sample switches a UHD TV into HDR mode and renders an HDR scene with values higher than 1.0f, which will be displayed as brighter than white on a UHD TV. 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.

A picture containing timeline

Description automatically generated

Text

Description automatically generated

# Building the sample

If using an Xbox One devkit, set the active solution platform to Gaming.Xbox.XboxOne.x64.

If using Project Scarlett, set the active solution platform to Gaming.Xbox.Scarlett.x64.

*For more information, see* Running samples*, in the GDK documentation.*

# Using the sample

The sample uses the following controls.

|  |  |
| --- | --- |
| Action | Gamepad |
| Toggle displaying ST.2084 curve | A button |
| Toggle displaying only paper white block | B button |
| Adjust brightness of paper white | D-Pad |
| Adjust values | Left/Right thumb stick |
| Exit | View button |

# Implementation notes

This sample uses an API to determine if the attached display is HDR capable. If so, it will switch the display to HDR mode. A very simple HDR scene, with values above 1.0f, is rendered to a FP16 backbuffer and outputs to two different swapchains, one for HDR and one for SDR. Even if the consumer uses an HDR display, the SDR signal is still required for GameDVR and screenshots.

This sample has a version of the [DeviceResources](https://github.com/Microsoft/DirectXTK12/wiki/DeviceResources) class which supports both the HDR and SDR swapchains.

Refer to the white paper “[HDR on Xbox One](http://aka.ms/hdr-on-xbox-one)”.

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/DirectXTK12/wiki/PostProcess) class in *DirectX Tool Kit for DirectX 12* 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 using the DPad up/down. 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

Note that the implementation for Project Scarlett here works, but a future update will provide an easier method to make use of auto-tone mapping. This will allow Project Scarlett titles to present just the HDR image and the SDR output and GameDVR will be fully handled by the system.

# 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/).