

FrontPanelDolphin 样本

# *\*本示例适用于 Xbox One XDK（2017 年 6 月）*

# 描述

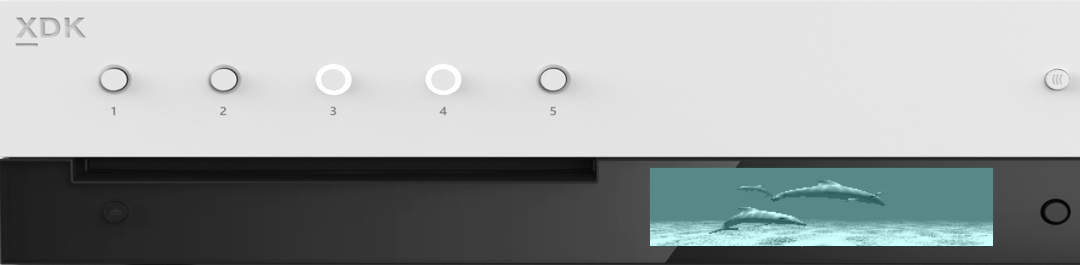
FrontPanelDolphin 演示了如何使用 GPU 渲染到 FrontPanel。我们预计大多数游戏开发人员将拥有大量使用 GPU 渲染到屏幕的代码，因此本样本的目的在于让您更轻易地利用现有代码来定位前面板显示器。

以下是一些使用案例：

* 您正在实验室设置中使用“无外设”开发套件运行游戏，并希望使用前面板渲染通常在屏幕上看到的内容。您可以一目了然地确定游戏是否健康。
* 许多游戏引擎已具有“开发 HUD”，能够实现游戏的诊断功能，这些功能通常在零售版本中不可访问。例如，通过执行秘密控制器按键组合，这将导致 HUD 出现。然后，HUD 提供在测试和开发过程中有用的其他选项，例如产生怪物、跳到特定级别或使角色无敌。此 HUD 通常由 GPU 进行渲染，因此如果能够将结果复制到前面板，则可重复使用现有的 HUD 代码并令其适应前面板。通过将其置于前面板，不必牺牲屏幕空间。此外，还可使用前面板上的方向键和按键，而不必使用游戏手柄。

# 使用样本

FrontPanelDolphin 样本适用于带有集成前面板的 Xbox One X DevKit。启动样本时，将在主显示屏上渲染海豚场景，同时也会在前面板 LCD 显示屏上进行。



每个按钮对应 1 个 LED 灯

5X 可编程按钮

256 x 64 x 4bpp OLED 显示屏

DPAD + 选择

## FrontPanelDolphin 主显示屏

## FrontPanelDolphin 前面板显示屏



除前面板方向键选择按键外，样本不处理任何输入。按下“选择”按键时，将从前面板显示屏捕获缓冲区，并将结果保存为 Title Scratch 文件夹中的 .dds 文件。

# 实施说明

样本使用名为 FrontPanelRenderTarget 的辅助类，顾名思义，是一个适合前面板显示屏的离屏渲染目标。FrontPanelRenderTarget 将所提供的纹理转换为灰度，然后将其渲染为四边形。这将使用非常简易的顶点着色器和非常简易的像素着色器完成。顶点着色器生成四边形，像素着色器对所提供的纹理进行采样，并使用点乘积将每个像素转换为灰度。渲染步骤在一个名为 GPUBlit(): 的方法中实现

//使用所提供的着色器资源视图作为源，来渲染灰度图像

void GPUBlit(ID3D11DeviceContext \*context, ID3D11ShaderResourceView \*srcSRV);

应从要渲染的纹理创建提供的 ID3D11ShaderResource 视图。

呼叫 GPUBlit() 后，然后需将结果复制回 CPU 上的缓冲区，然后将缓冲区提供给前面板显示屏。FrontPanelRendertTarget 类提供两种简化方法：

//将渲染目标复制到分段结构，然后将其复制回 CPU

void CopyToBuffer(ID3D11Device \*device,

ID3D11DeviceContext \*context, ATG::BufferDesc &desc);

//将渲染目标复制到分段结构，将结果复制回 CPU

//然后将其提供给前面板显示屏

//请注意，每次呼叫时都会分配一个额外的缓冲区。

void PresentToFrontPanel(ID3D11Device \*device, ID3D11DeviceContext \*context);

BufferDesc 是一种追踪 CPU 缓冲区宽度和高度的结构。FrontPanelRenderTarget::CopyToBuffer 可以复制到内存中的任何地址，只需一个 BufferDesc 描述缓冲区的尺寸。样本使用 FrontPanelDisplay 类，管理前面板缓冲区，并使用 FrontPanelDisplay::GetBufferDescriptor() 以获得前面板的 BufferDesc。然后呼叫 FrontPanelRenderTarget::CopyToBuffer() 从 FrontPanelRenderTarget 复制图像。最后，必须呼叫 FrontPanelDisplay::Present() ，以将图像实际显示在前面板显示屏上。

请注意，FrontPanelRenderTarget::PresentToFrontPanel() 方法负责两个步骤，复制到 CPU 并显示缓冲区。如果尚未使用 FrontPanelDisplay 类，这将非常方便。FrontPanelRenderTarget::PresentToFrontPanel() 的一个弊端是每次呼叫时必须分配一个缓冲区。

使用 FrontPanelRenderTarget 调适 Dolphin 样本以渲染前面板实际上非常简单。最简单的调适只需做出一些改变：

**在 Sample::Sample 中：**

//构造前面板渲染目标

m\_frontPanelRenderTarget = std::make\_unique<FrontPanelRenderTarget>();

// 获取默认前面板

DX::ThrowIfFailed(GetDefaultXboxFrontPanel(m\_frontPanelControl.ReleaseAndGetAddressOf()));

// 初始化 FrontPanelDisplay 对象

m\_frontPanelDisplay = std::make\_unique<FrontPanelDisplay>(m\_frontPanelControl.Get());

**在 Sample::CreateDeviceDependentResources 中：**

//设置通过 GPU 渲染到前面板

ComPtr<IXboxFrontPanelControl> frontPanelControl;

DX::ThrowIfFailed(GetDefaultXboxFrontPanel(frontPanelControl.ReleaseAndGetAddressOf()));

//创建前面板渲染目标资源

m\_frontPanelRenderTarget->CreateDeviceDependentResources(frontPanelControl.Get(),

device);

**在 Sample::CreateWindowSizeDependentResources 中：**

//为主渲染目标创建着色器资源视图

auto device = m\_deviceResources->GetD3DDevice();

DX::ThrowIfFailed(device->CreateShaderResourceView(m\_deviceResources->GetRenderTarget(),

nullptr, m\_mainRenderTargetSRV.GetAddressOf()));

**在 Sample::Render 中：**

//闪射到前面板渲染目标，然后提供给前面板

m\_frontPanelRenderTarget->GPUBlit(m\_deviceResources->GetD3DDeviceContext(),

m\_mainRenderTargetSRV.Get());

auto fpDesc = m\_frontPanelDisplay->GetBufferDescriptor();

m\_frontPanelRenderTarget->CopyToBuffer(m\_deviceResources->GetD3DDevice(),

m\_deviceResources->GetD3DDeviceContext(), fpDesc);

m\_frontPanelDisplay->Present();

# 更新历史记录

2017 年 4 月，本示例首次发布。

# 隐私声明

编译和运行示例时，示例可执行文件的文件名将发送给Microsoft以帮助跟踪示例使用情况。要选择退出此数据收集，您可以删除Main.cpp中标记为“Sample Usage Telemetry”的代码块。

有关 Microsoft 隐私政策的更多信息，请参阅 [Microsoft 隐私声明](https://privacy.microsoft.com/zh-cn/privacystatement/)。