介绍

Renderdoc 是一个逐帧的图形调试器，适用于Vulkan，D3D11，D3D12，OpenGL和OpenGL ES多种图形API，适用于构建于Windows，Linux，Android，Stadia，和Nintendo Switch多种的平台的应用。

License

使用

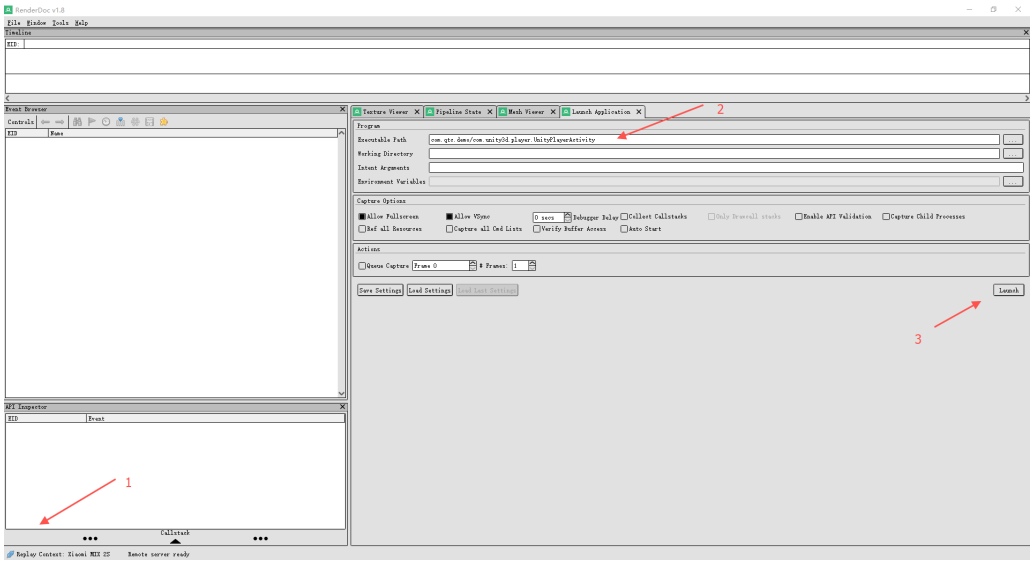
Renderdoc的工作流很简单。

1. 它在启动应用的时候加载调试器
2. 然后在你需要捕获帧的时候将当前绘制帧的所有信息都保存到一个文件中
3. 你可以打开这个文件，但前提是你的应用需要运行中，因为它需要你的机器的硬件和你的应用的一些资源信息去解析这个文件
4. 解析好之后你就可以在它内置的各种视图里调试了

在此之前，你需要准备一个允许加载调试器的包。在Unity应用上体现为打包的时候，将Development Build勾选上。

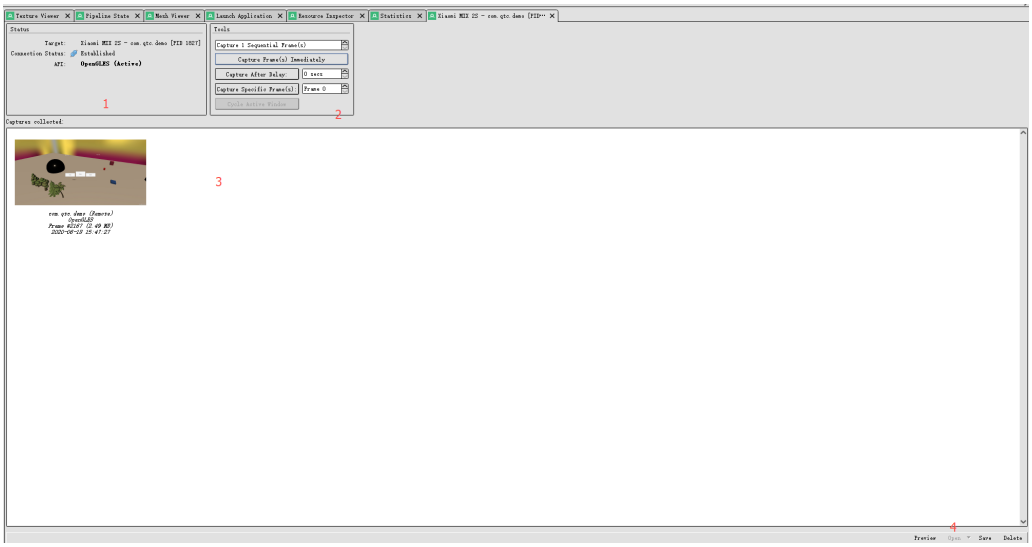
捕获帧

打开Renderdoc，点开 *Launch Application* 视图。



1. 在这里检查你是否连上了机器。Renderdoc会在手机上安装两个服务。如果这里显示大红×，说明连接出现了异常。
2. 在这里选择需要调试的应用的主活动。
3. 在这里启动你的应用。通过Renderdoc启动应用，Renderdoc会为待调试应用加载调试器。如果调试器成功附着到应用上，则应用的左上角应该显示一行白色的字，显示当前的一些信息。

等应用启动完毕，Renderdoc会跳到一个用于捕获帧的视图：



如上图所示，视图有四个部分组成，分别在图中用红色的数字标出：

1. Status 状态。在这里你可以看到以下信息：

- Target：调试的目标。由机器名称，调试的应用包名及其进程pid组成。
- Connection Status：Renderdoc与机器的连接状态。
- API：应用所使用的图形API。

2. Tools 工具。在这里设置如何捕获：

- Capture Frame(s) Immediately ：立即开始捕获。捕获多少帧在它上面的文本框中设置。
- Capture After Delay：n 秒之后开始捕获。捕获多少帧也是在最上面的文本框中设置。
- Capture Specific Frame(s)：捕获特定序号的帧。Renderdoc从启动应用开始，就为每一帧编了序号。当当前帧的序号等于你设置的捕获序号时，Renderdoc就会将该帧捕获下来。它其实发送了一个信号，这个信号通知Renderdoc的服务端等待特定序号的帧出现。由于帧的序号是递增的，所以如果设置的序号的帧已经渲染过了，那这个帧永远不会被捕获。

3. Captures collected 显示捕获的帧。每个帧都包含自身的序号，大小和被捕获时的物理时间。

4 底部的功能栏。功能有：

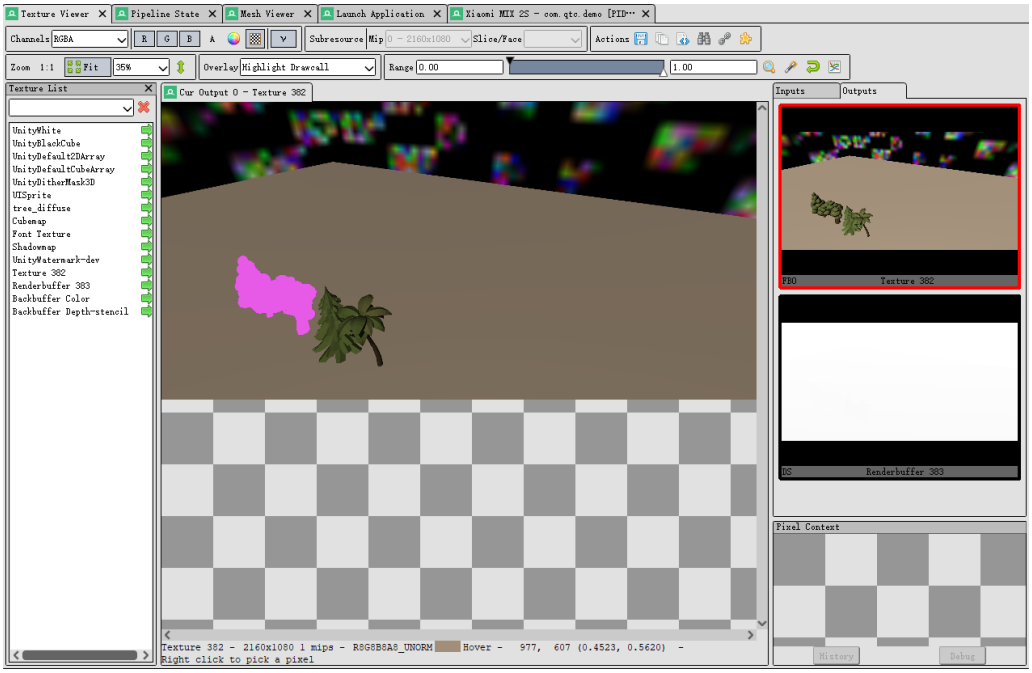
- Preview：在Captures Collected窗口选择一个帧，点击Preview会展开一个预览窗口。
- Open：打开当前选中的帧。
- Save：保存帧。
- Delete：删除帧。

调试帧

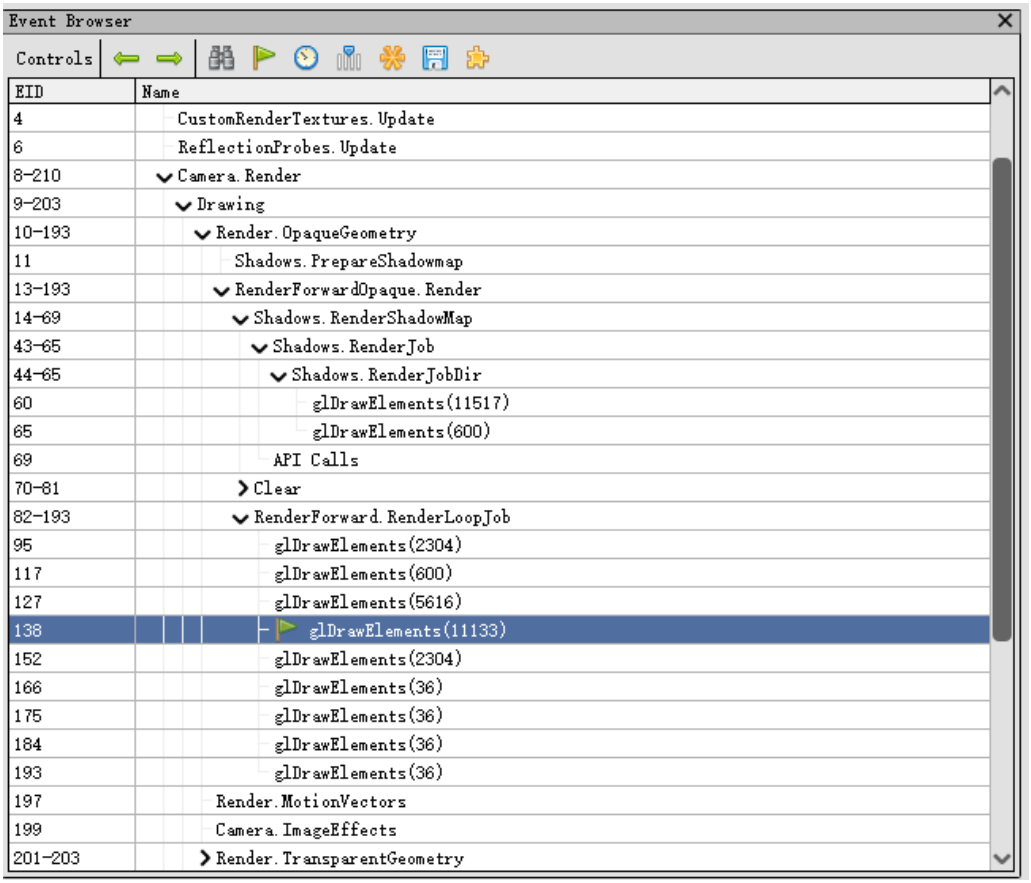
Renderdoc提供了很多视图为你调试提供便利。一些主要的视图包括：

Texture Viewer 纹理视图。在这个视图中你可以查看在绘制当前的帧的时每一步Drawcall在做的事情。从这个视图中，你可以知道GPU可能在：绘制网格，制作Mipmap，制作阴影贴图，渲染光照，应用后处理等等。你可以通过这个视图去查看当前帧所运用的纹理贴图或者RenderTexture，可以查看当前GPU在缓冲中的输出或者输入的纹理。更多信息请参考章节 Texture View

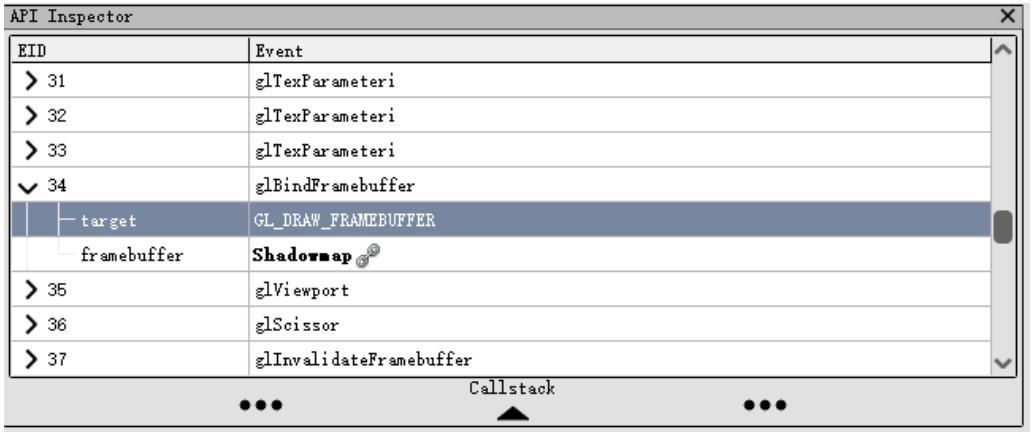
下图显示此刻GPU在绘制两个树形网格。



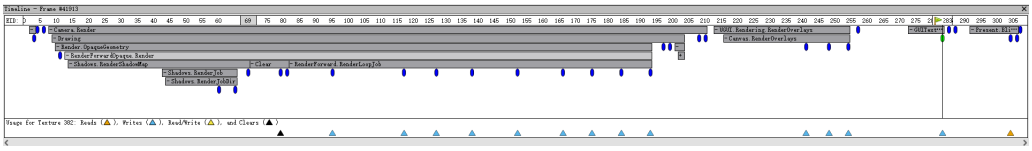
Event Browser 事件浏览器。在这个视图中你可以查看当前帧所包含的一些重要的渲染函数。其中DrawCall函数都处在叶节点的位置。你可以细数当前帧每个Drawcall所处的位置，并细数每个Drawcall提交的图元数量。更多信息请参考章节 Event Browser



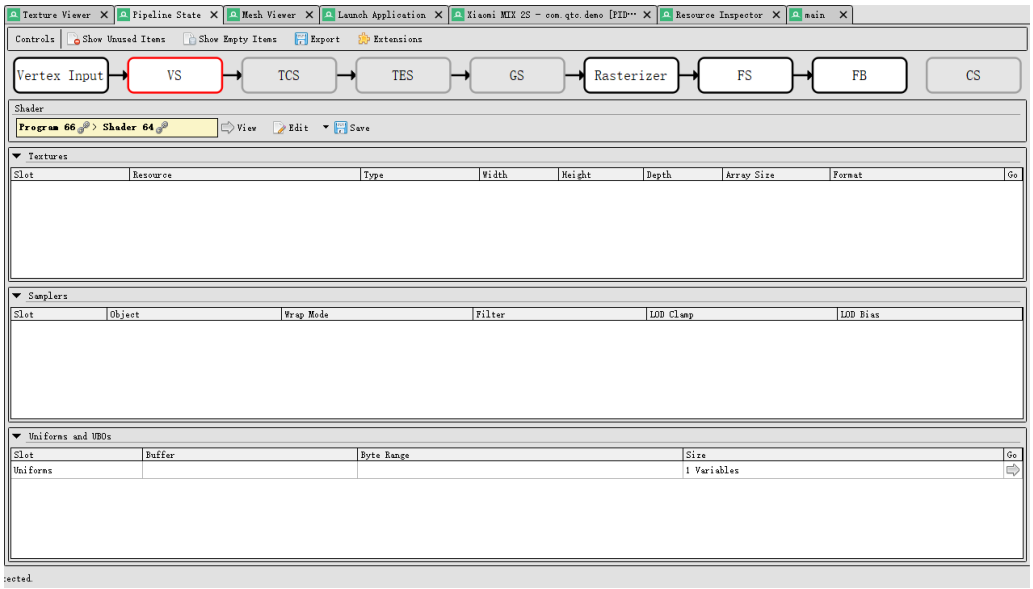
API Inspector API 视图。每当一个Drawcall在 Event Browser 中被选中，API 视图都会在此更新有关当前Drawcall的信息。这些信息包括从上次Drawcall到当前Drawcall所经历的一些函数（因为OpenGL的渲染机制是基于状态机的，所以在OpenGL中这些往往都是一些改变渲染状态的函数，包括纹理状态，顶点状态，颜色状态，等等）。更多信息请参考 API Inspector



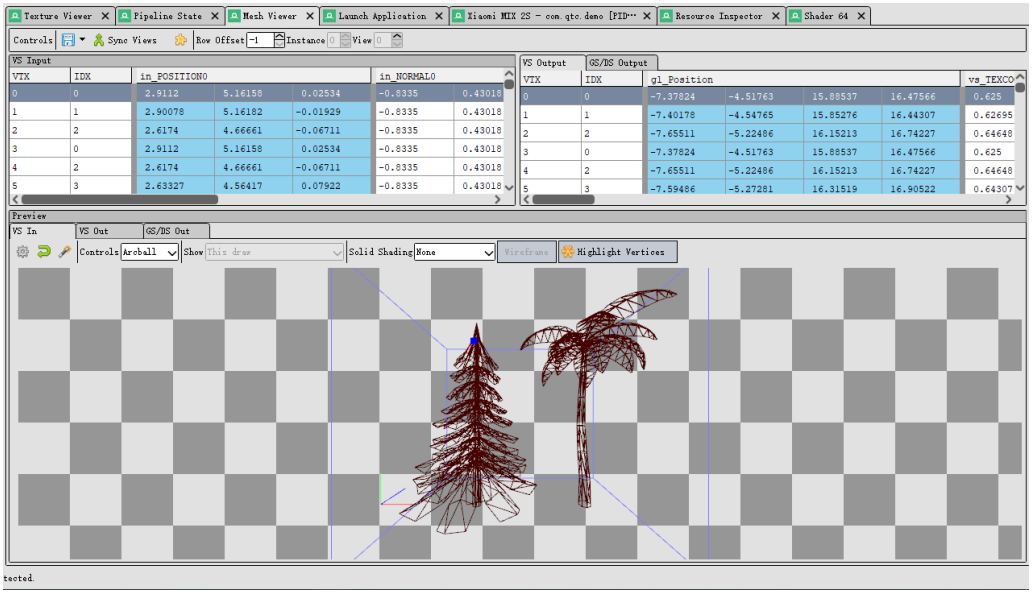
Timeline Bar 时间线视图。这个视图可是看作是 Event Browser 的另一个版本。它将 Event Browser 中的函数按照时间线进行排列，你可以在这个视图上清晰看到每个函数在当前帧中所占用的相对时间，以及每个Drawcall在时间线上所处的位置。更多信息请参考 Timeline Bar



Pipeline State 渲染管线状态视图。你可以在这个视图中看到对应Drawcall的渲染管线的状态，包括启用了哪个着色器，每个着色器的输入输出，等等。你甚至可以查看着色器对应的GLSL源码。更多信息请参考 Pipeline State



Mesh Viewer 网格视图。你可以在这个视图查看当前Drawcall提交的顶点信息，即当前Drawcall在顶点着色器中的输入。你还可以查看顶点着色器的输出，等等。更多信息请参考 Mesh Viewer



其他视图 上面主要列举一些调试用的最经常使用到的视图。一些高级视图可以查看章节 视图

视图

Renderdoc的每个视图都是一个功能的合集，它可能专门用来查看纹理，或者专门查看渲染管线，或者专门查看当前顶点着色器的输入的网格……如果你发现有某个视图没有显示出来，可以菜单栏中的`window`里打开它。

下面我们对每个视图进行详细的介绍：

Texture View

Texture View（纹理视图）