Game Development with OpenXR

Christoph Haag
christoph.haag@collabora.com
01/02/2020

FOSDEM’20
Game Development with OpenXR

- OpenXR from a user/game developer view
- Getting Started with OpenXR
  - List of Examples
  - C simple example
  - Godot Engine Plugin
- TODOs
OpenXR from a user/game developer view

- **VR Inputs**
  - Headset, Controller position/rotation ("pose")
  - Buttons, touch, force, touchpad, thumbstick, …

- **VR Output**
  - Stereoscopic, tailored to hardware parameters
  - Lenses (distortion)
  - Extremely sensitive to latency/stutter

- **Vendor VR runtime**
  - Hardware drivers (USB data → pose, button events)
  - Presentation of Frames (distorted, async)
  - Provide API (vendor specific or OpenXR)
  - Think: libinput/X.org/Wayland
OpenXR Runtimes/Hardware today

- Windows
  - Oculus (beta) => Rift, Rift S, Quest
  - Windows MR [D3D only] => Windows MR
- Linux
  - Monado =>
    - OpenHMD, FLOSS, often reverse engineered & incomplete drivers
    - Native drivers in development (psmove, psvr, hydra, OSVR HDK)
    - libsurvive

FOSS Virtual & Augmented Reality
The Monado project & OpenXR

OpenHMD Community meeting 2020
Open Source VR, XR, AR community meeting by the OpenHMD Community!

The year of the virtual Linux desktop
OpenXR from a user/game developer view

- Derived from Vulkan spec
  - Inherits some API concepts
  - Similar loader/header layout (provided by Khronos: OpenXR-SDK)
    - `libopenxr_loader.so`
    - `openxr.h, openxr_platform.h`
  - Extensions (KHR/EXT, 3D API support, unusual display configurations, input ...)
  - API Layers (validation, ...)
- No compile time references to specific runtime, only loader
- At runtime, loader finds & dispatches to any OpenXR compliant runtime
Code Examples

- [https://github.com/KhronosGroup/OpenXR-SDK-Source/tree/master/src/tests/hello_xr](https://github.com/KhronosGroup/OpenXR-SDK-Source/tree/master/src/tests/hello_xr)
  - C++, abstracted, cross platform, OpenGL/Vulkan/D3D
  - C++, integrated/abstracted, windows only
- [https://github.com/Microsoft/OpenXR-SDK-VisualStudio](https://github.com/Microsoft/OpenXR-SDK-VisualStudio)
  - C++, windows only, D3D
- [https://github.com/jherico/OpenXR-Samples/](https://github.com/jherico/OpenXR-Samples/)
  - C++, abstracted, windows only, OpenGL
- [https://gitlab.freedesktop.org/xrdesktop/gxr](https://gitlab.freedesktop.org/xrdesktop/gxr)
  - C, glib, linux only, Vulkan with OpenGL memory object
- [https://gitlab.freedesktop.org/monado/demos/xrgears](https://gitlab.freedesktop.org/monado/demos/xrgears)
  - C++, mildly abstracted, linux only, Vulkan
- [https://gitlab.freedesktop.org/monado/demos/openxr-simple-example](https://gitlab.freedesktop.org/monado/demos/openxr-simple-example)
  - C, no abstraction, linux only, OpenGL
Simple example

- gcc app.c -o app `pkg-config --libs --cflags glx openxr`
- XR_RUNTIME_JSON=/path/openxr_monado-dev.json ./app
- Symlink: /etc/xdg/openxr/1/active_runtime.json
  - ./app

- Simple example: code & live demo
Godot Plugin

- live demo & code
TODOs

• Packaging OpenXR Loader/Headers in Distributions
  – Like Vulkan Loader/Headers

• Action binding UI for godot
  – Shared UI with OpenVR plugin?
  – https://github.com/GodotVR/godot_openvr/issues/71
Todos

- **Graphics Binding**
  - `glXGetCurrentContext();`
  - `glXGetCurrentDrawable()`

- **Windows Port**
  - Some `#defines`
  - Graphics binding
  - Testing

```c
typedef struct XrGraphicsBindingOpenGLXlibKHR {
   .XrStructureType type;
    const void* next;
    Display* xDisplay;
    uint32_t visualid;
    GLXFBConfig glxFBConfig;
    GLXDrawable glxDrawable;
    GLXContext glxContext;
} XrGraphicsBindingOpenGLXlibKHR;
```

```c
typedef struct XrGraphicsBindingOpenGLWin32KHR {
   .XrStructureType type;
    const void* next;
    HDC hDC;
    HGLRC hGLRC;
} XrGraphicsBindingOpenGLWin32KHR;
```
FOSDEM’20

Game Development with OpenXR

Any questions?

FossXR Conference 2020
https://fossxr.dev/
https://twitter.com/FossXR/

Psst... We're hiring!

Open First