Mojo Simplified WebVR Architecture

This document describes the mojo refactory architecture of WebVR.

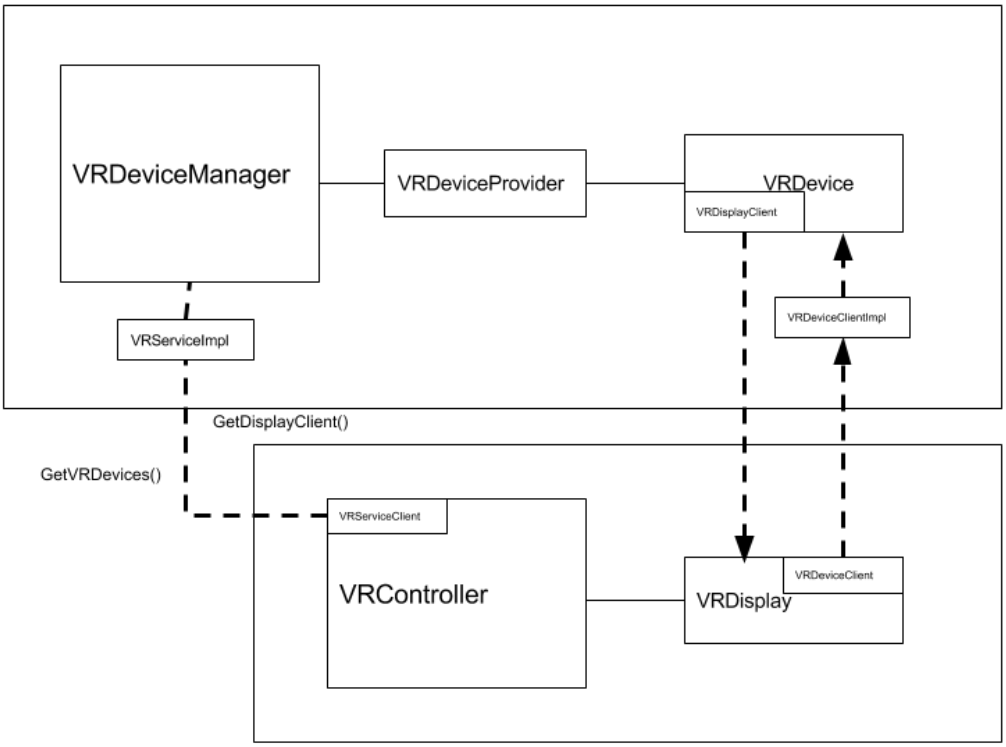
Problem

WebVR have two major manage class : VRDeviceManager manages VRDevices which offer real vr device functions on device side in browser process and VRController manages VRDisplays which offers to user as representation of vr device on blink side in render process. These two major class communicate through mojo message pipe and handle all the API calls.

Since most of the API calls are actually from VRDisplay to VRDevice or VRDevice to VRDisplay, VRDeviceManager and VRController need to implementation API in mojo interface with index attribute and this is not what mojo recommended.

In a recommended way, VRDevice should communicate with VRDisplay directly through a mojo message pipe to handle all this VRDevice to VRDisplay API call or VRDisplay to VRDevice API call. And that's why WebVR need to refactory.

Architectural overview



Mojo Interface

- VRService : Build message pipe between VRController and VRDeviceManager. It only contains GetVRDevices() API. This API functions the initial call from WebVR app. It will query vr devices and request update. It contains SetClient() API. This API will set VRServiceClient **It will be implemented by VRServiceImpl class and VRController will get VRServicePtr.**
- VRServiceClient : Build message pipe between VRDeviceManager and VRController. It only contains GetDisplayClient(VRDisplayClientPtr&). When device side has device created, it will be called to get a VRDisplayClientPtr for the device. **It will be implemented by VRController class and VRServiceImpl will get VRServiceClientPtr**
- VRDeviceClient : Build message pipe between VRDisplay and VRDevice, it contains most of the APIs which calls from VRDisplay and dealing in VRDevice. **It will be implemented by VRDeviceServiceImpl class and each VRDisplay will get an VRDeviceServicePtr.**
- VRDisplayClient : Build message pipe between VRDevice and VRDisplay. It contians most of the APIs which calls from VRDevice and dealing in VRDisplay, most of these APIs are event handlers. **It will be implemented by VRDisplay class itself, and VRDevice will get an VRDisplaySevicePtr.**

Binding Process

There are two part of Binding Process :

- VRDeviceManager and VRController Binding : It happens when VRController construct. VRController will get VRServicePtr from frame, at the same time VRServiceImpl class will be construct. And when VRController called SetClient, VRSeviceImpl will register to VRDeviceManager and VRServiceImpl will get VRServiceClientPtr.
- VRDevice and VRDisplay Binding: It happens after VRServiceClientPtr was held by VRServiceImpl class. On device side, each VRDevice created will called GetDisplayClient API through VRServiceImpl class. And on blink side, VRController will created a new VRDisplay and return VRDisplayClientPtr to devcie side when GetDisplayClient API is invoked. The new created VRDevice will get VRDisplayClientPtr and built a message pipe between VRDevice and VRDisplay.
  Then VRDevice will invoke SetDeviceClient() API during user called getDisplay API. SetDeviceClient() will created a VRDeviceClientPtr and passed to VRDisplay which the VRDevice built message pipe with. After that, the bi-direction message pipe with a pair of VRDevice and VRDisplay has finished its building process.