SOSCON 2018

IMMERSIVE WEB

JINHO BANG, BYUNGKWON KO
zino@chromium.org, codeimpl@gmail.com

WHO ARE WE?

Samsung Electronics
Chromium/Blink OWNER
W3C Spec Editor
Node.js Contributor

Samsung Electronics
Chromium/Blink Member

INTRODUCTION

Play VR & AR Movies

CONTENTS

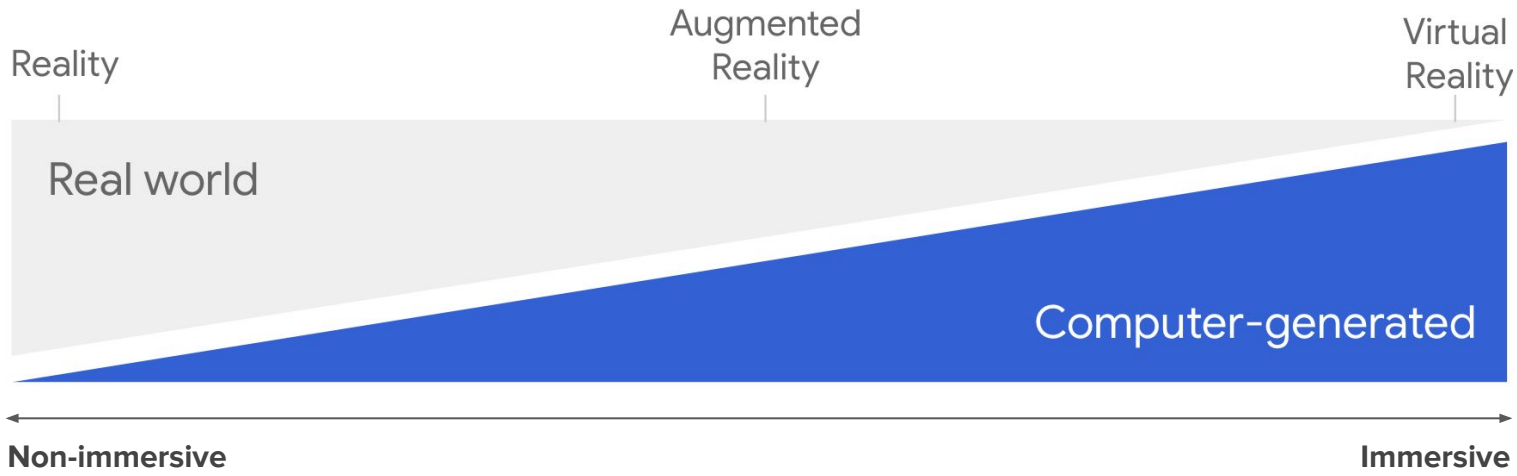
- What’s the “Immersive Web”?
- WebXR’s GOAL
- How WebXR App Works?
- How to implement WebXR App
- WebXR Internals in Chromium

NON-GOAL

- How to write/render WebXR contents?
- Complex graphics theory
- Specific frameworks or XR engine

WHAT'S THE IMMERSIVE WEB

Immersive..


Reality
Augmented Reality
Virtual Reality
Real world
Computer-generated
Non-immersive
Immersive

The immersive web means virtual world experiences hosted through the browser.

WebAR + WebVR + ⋯ + = **WebXR**

WebAR + WebVR + ⋯ + = **WebXR**

= **“Immersive Web”**

Immersive Web Working Group

Our Mission

> *We help bring high-performance Virtual Reality (VR) and Augmented Reality (AR) (collectively known as XR) to the open Web via APIs to interact with XR devices and sensors in browsers.*
>
> *We work together with the Immersive Web Community Group in building the immersive web platform. Most new ideas start there.*

The Co-Chairs of the Working Group are Ada Rose Cannon (Samsung) and Chris Wilson (Google). The W3C Team Contact for the Immersive Web Working Group is Dominique Hazaël-Massieux.

October 25-26 at TPAC 2018

We'll be meeting for the first time at TPAC 2018.

Open Web Platform

You can contribute to..

- WebXR Standard Spec
- Web Platform(Browser) Implementation
- Web Platform Tests
- MDN (Documents)

WEB XR'S GOAL

세상에는 다양한 XR 디바이스
제조사 및 브랜드가 존재합니다

또한 세상에는

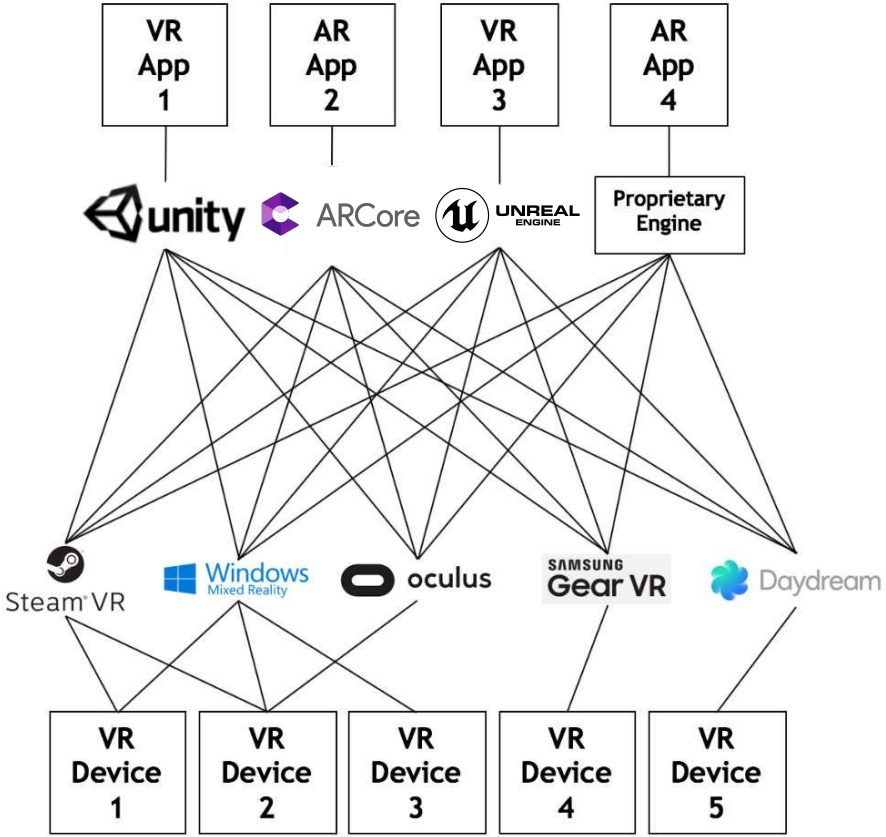
다양한 XR 디바이스들이 있습니다

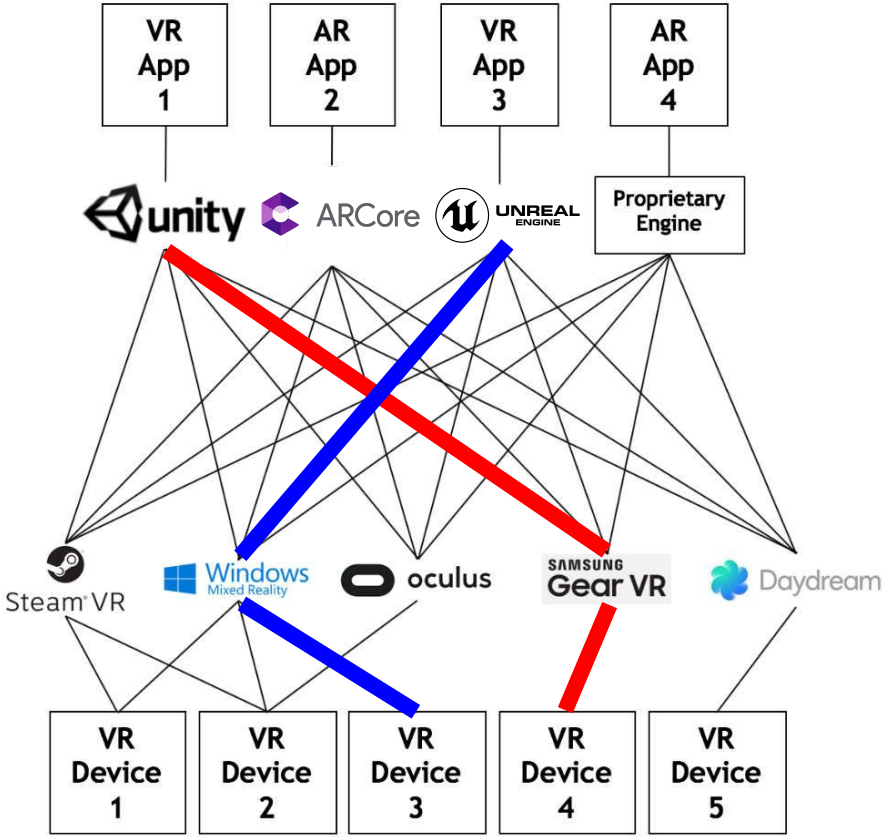



각 제조사 및 브랜드 기기마다 지원하는

XR 엔진 또한 다양합니다

XR 컨텐츠 개발자들은 각각의 엔진 위에서
앱이나 서비스를 개발합니다


VR App 1
AR App 2
VR App 3
AR App 4
unity
ARCore
UNREAL ENGINE
Proprietary Engine
Steam VR
Windows Mixed Reality
oculus
SAMSUNG Gear VR
Daydream
VR Device 1
VR Device 2
VR Device 3
VR Device 4
VR Device 5


VR App 1
AR App 2
VR App 3
AR App 4
unity
ARCore
UNREAL ENGINE
Proprietary Engine
Steam VR
Windows Mixed Reality
oculus
SAMSUNG Gear VR
Daydream
VR Device 1
VR Device 2
VR Device 3
VR Device 4
VR Device 5

XR시장 성장감소↓ 야기

한 번의 XR 컨텐츠 작성으로
여러 디바이스와 플랫폼을 지원할 수 없을까?

문제 해결의 핵심

Open Web Platform

웹 브라우저는 데스크탑이든 모바일이든

플랫폼에 관계 없이 동작합니다

Your XR App/Service

Windows Mixed Reality

UNREAL ENGINE

SAMSUNG Gear VR

STEAM

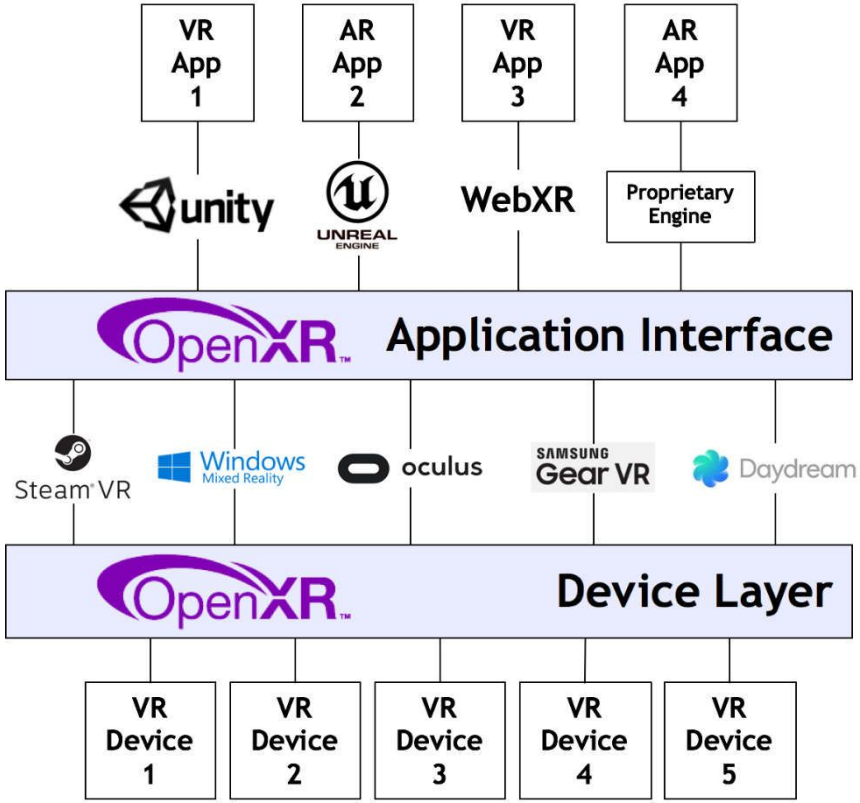
Daydream

oclulus

unity

ARCore

WebXR vs OpenXR


VR App 1
AR App 2
VR App 3
AR App 4
unity
UNREAL ENGINE
WebXR
Proprietary Engine
OpenXR Application Interface
Steam VR
Windows Mixed Reality
oculus
SAMSUNG Gear VR
Daydream
OpenXR Device Layer
VR Device 1
VR Device 2
VR Device 3
VR Device 4
VR Device 5

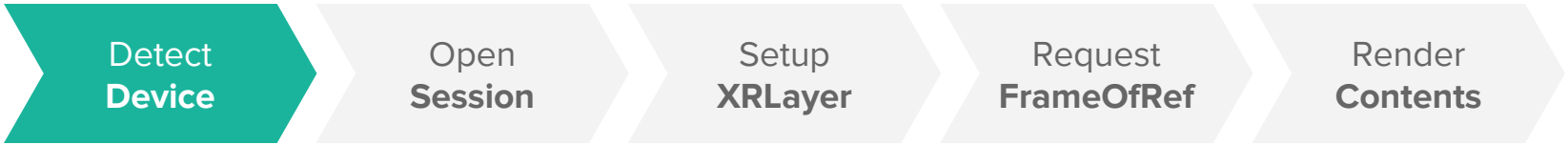
WebXR은 **한 번의 XR 컨텐츠 작성**으로
광범위한 디바이스와 플랫폼을 커버할 수 있다!

HOW WEB XR APP WORKS?

How WebXR App works

Detect **Device** → Open **Session** → Setup **XRLayer** → Request **FrameOfRef** → Render **Contents**

STEP1: Detect Device



```
navigator.xr.requestDevice();
```

SOSCON 2018

XRDevice

XRDevice

PC/Mobile Web

```
navigator.xr.requestDevice();
```

XRDevice

XRDevice

Default
XRDevice

XRDevice

PC/Mobile Web

```
navigator.xr.requestDevice();
```

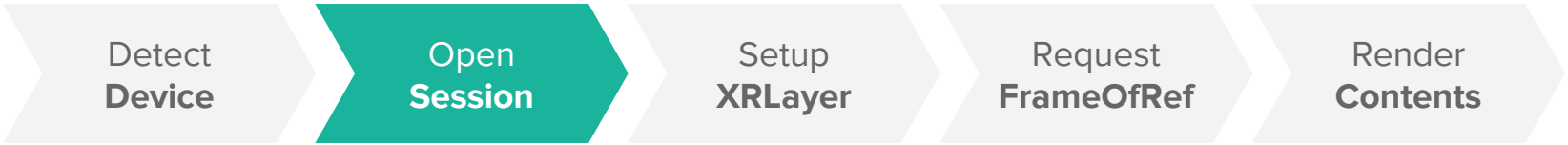
XRDevice

XRDevice

"XRDevice" 객체는

XR 디바이스들의 추상화이다

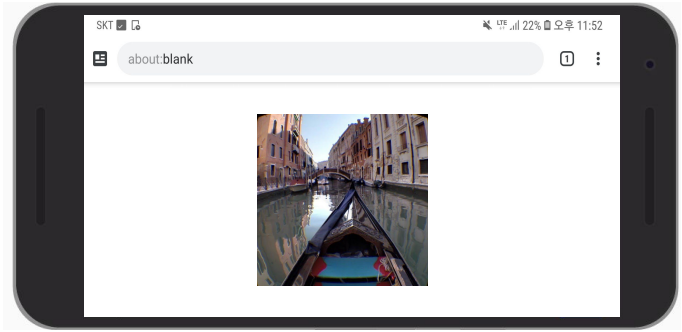
STEP2: Open Session



"XRSession"을 생성하면

XR 디바이스와 상호작용을 할 수 있다

```
device.requestSession();
```




```
device.requestSession({
    immersive: false
});
```

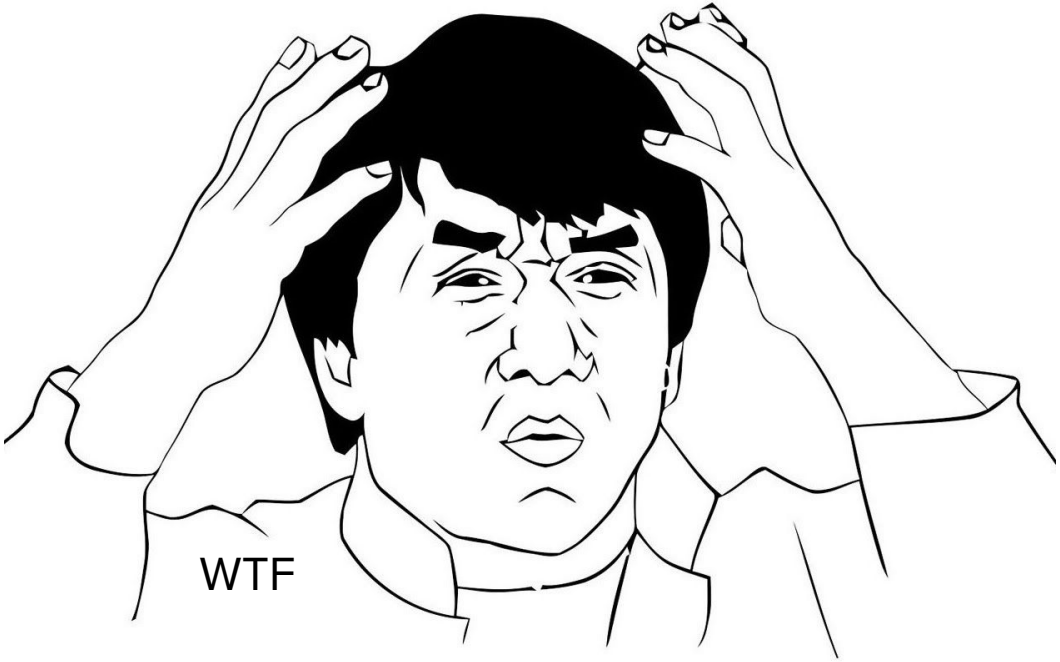
```
device.requestSession({
    immersive: true
});
```

```
device.requestSession({ immersive: true });
```

```
SecurityError: The requested session requires
               user activation.
```

User Activation

- `event.isTrusted` should be `true`
- event's type is one of:
  - change
  - click
  - contextmenu
  - dblclick
  - mouseup
  - pointerup
  - reset
  - submit
  - touchend

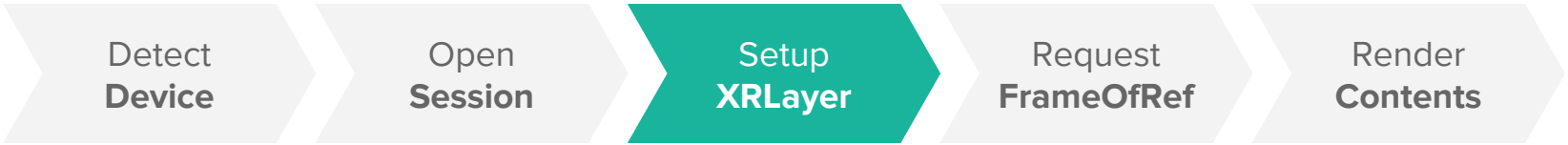

WTF

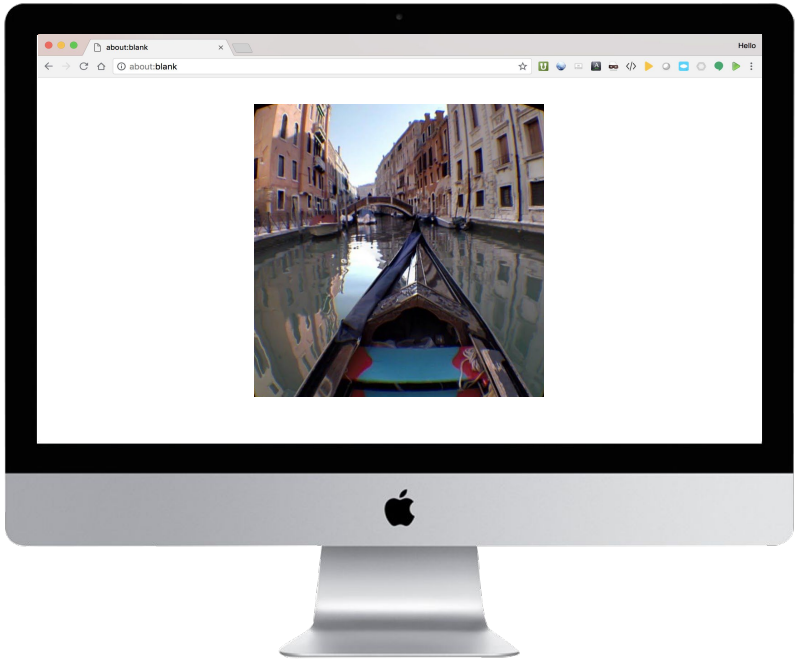
User Interaction이 요구된다

Open WebXR

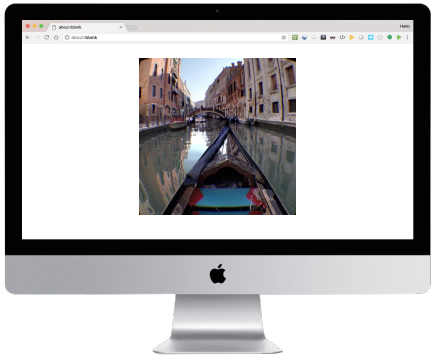
```
device.requestSession({
    immersive: true
});
```

STEP3: Setup XRLayer

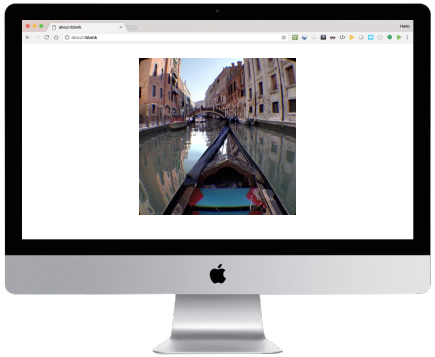



about:blank
Hello
about:blank

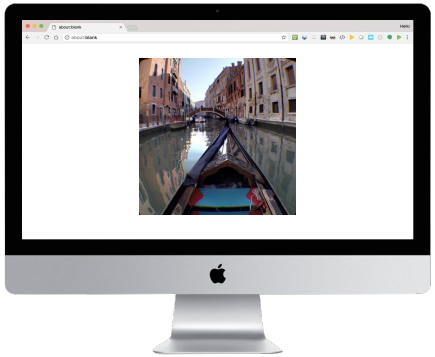
<canvas></canvas>



XRLayer



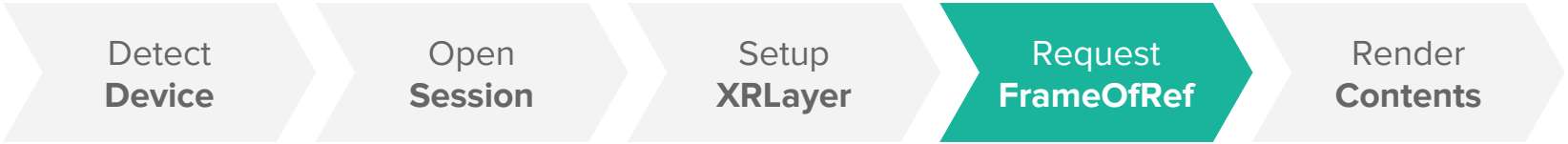
```
session.baseLayer = new XRWebGLLayer(session, gl);
```



```
drawScene(gl);
```

XRWebGLLayer

STEP4: Request FrameOfReference



FrameOfReference란 무엇일까?

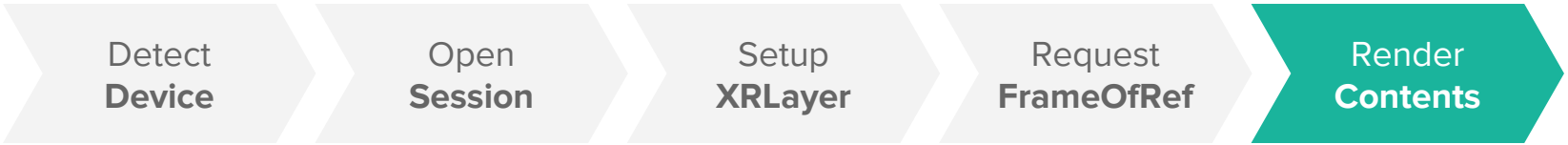
Frame Of Ref == Ref Coordinate System

Frame Of Reference Types

- head-model
  - The origin is approximately the location of the viewer's head and does not change if the viewer moves.
- eye-level
  - The origin is the viewer's head and moves with the viewer.
- stage
  - The origin is implied to be the center of the room at floor level and does not change if the viewer moves.

```
session.requestFrameOfReference('eye-level');
```

STEP5: Render Contents



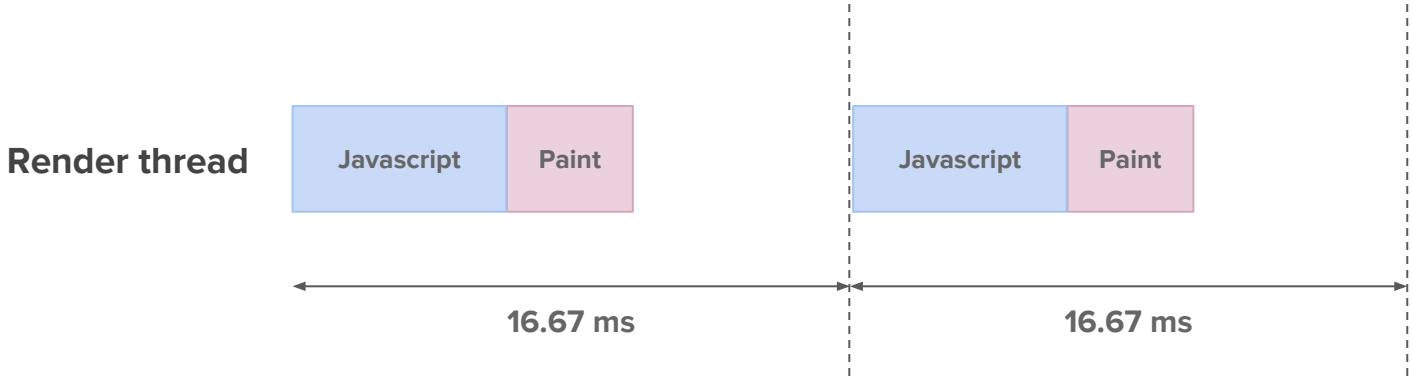
그냥 WebGL을 사용해서 그리면 된다

```
requestAnimationFrame(onDrawFrame);
```

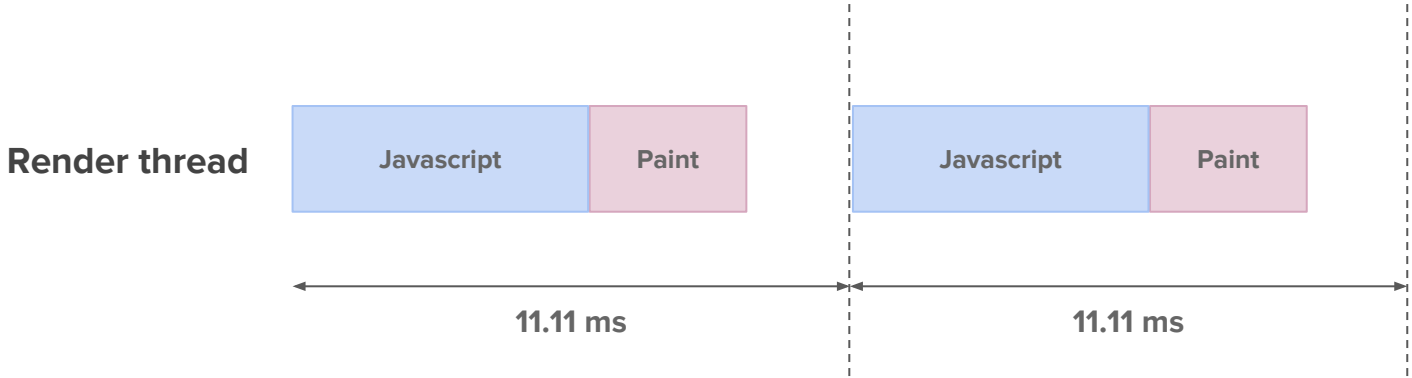
60 MHz

90 MHz

window.requestAnimationFrame()



session.requestAnimationFrame()



```
function onDrawFrame(time, frame) {
    ...
}
```

XRFrame’s informations

- Pose
  - In general, the position and orientation of a thing in AR/VR is called a pose.
- Views
  - Each view corresponds to a display or portion of a display used by an XR device to present imagery to the user.

HOW TO IMPLEMENT WEB XR APP

Summary: How WebXR App Works

Detect **Device** → Open **Session** → Setup **XRLayer** → Request **FrameOfRef** → Render **Contents**

Request Device

```
// Detect default XRDevice
const device = await navigator.xr.requestDevice();
```

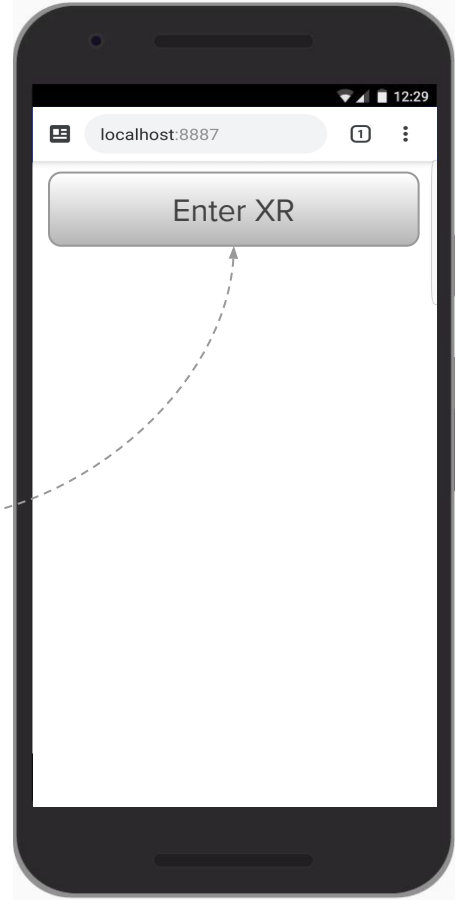
Not Found

Available XRDevice

Available XRDevice

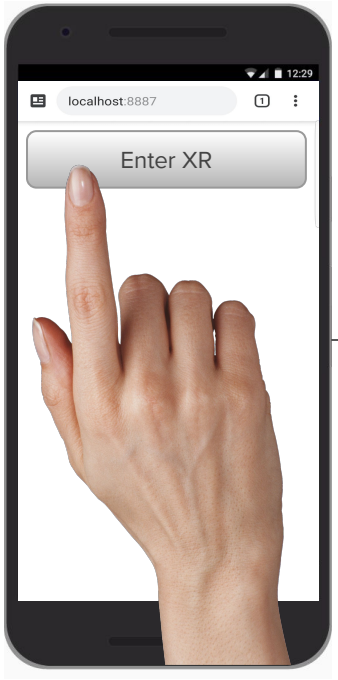
Request Session

```
const xrButton = document.querySelector(‘button’);
try {
  await device.supportsSession({ immersive: true });
  xrButton.innerText = ‘Enter VR’;
  xrButton.disabled = false;
  xrButton.addEventListener('click', onEnterXR);
} catch(error) {
  // Not support session
}
```



Request Session

```
function onEnterXR() {
  session = await device.requestSession({
    immersive: true
  });
  ...
}
```



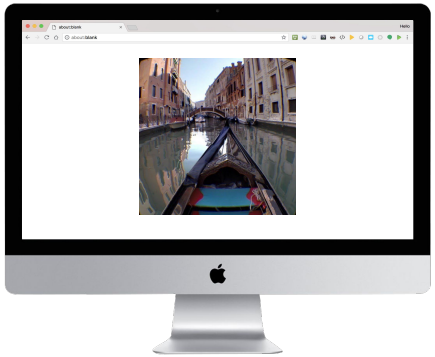

```
device.requestSession({
    immersive: true
});
```

Setup XRLayer

```
function setupXRLayer() {
  const canvas = document.createElement('canvas');
  gl = canvas.getContext('webgl', {
    compatibleXRDevice: session.device
  });
}
```

Setup XRLayer

```
function setupXRLayer() {
  const canvas = document.createElement('canvas');
  gl = canvas.getContext('webgl', {
    compatibleXRDevice: session.device
  });
  session.baseLayer = new XRWebGLLayer(session, gl);
}
```



```
drawScene(gl);
```

XRWebGLLayer

Request FrameOfReference

```
frameOfRef = await session.requestFrameOfReference('stage');
```

Rendering Loop

```
function onXRFrame(time, frame) {
  session.requestAnimationFrame(onXRFrame);
}

session.requestAnimationFrame(onXRFrame);
```

Get Device Pose

```
function onXRFrame(time, frame) {
  session.requestAnimationFrame(onXRFrame);
  const pose = frame.getDevicePose(frameOfRef);
  if (!pose)
    return;
}

session.requestAnimationFrame(onXRFrame);
```

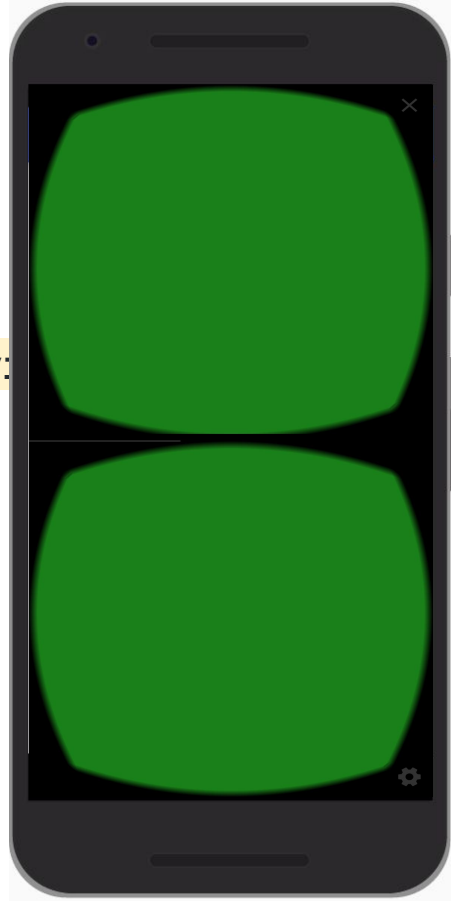
Bind Frame Buffer

```
function onXRFrame(time, frame) {
  session.requestAnimationFrame(onXRFrame);
  const pose = frame.getDevicePose(frameOfRef);
  if (!pose)
    return;
  gl.bindFramebuffer(session.baseLayer.framebuffer);
}

session.requestAnimationFrame(onXRFrame);
```

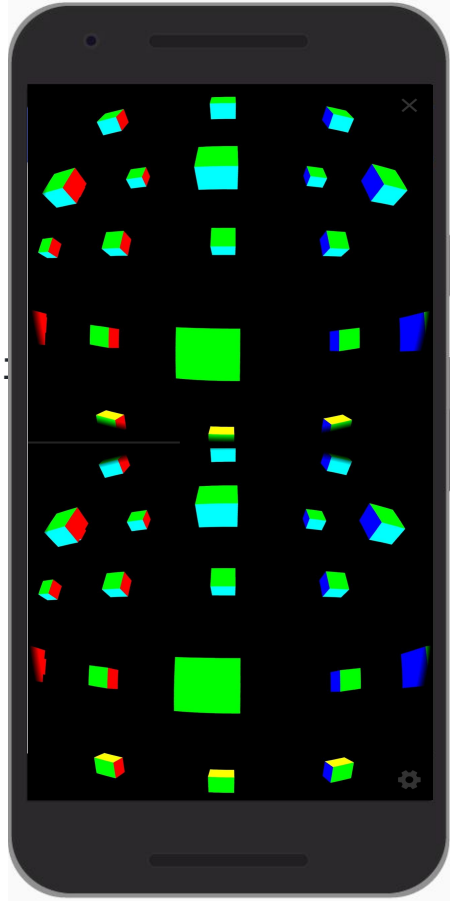
Rendering

```
function onXRFrame(time, frame) {
  ...
  for(let view of frame.views) {
    const viewport = session.baseLayer.getViewport(v
    gl.viewport(viewport.x, viewport.y,
                viewport.width, viewport.height);
    gl.clearColor(0.1, 0.5, 0.1, 1.0);
    gl.clear(gl.COLOR_BUFFER_BIT);
  }
}
```



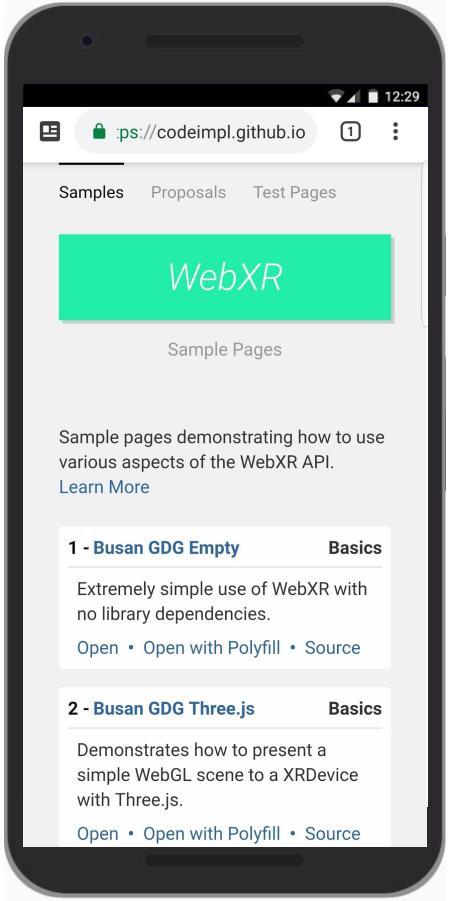
Rendering with Three.js

```
function onXRFrame(time, frame) {
  ...
  for(let view of frame.views) {
    const viewport = session.baseLayer.getViewport(vi
    gl.viewport(viewport.x, viewport.y,
                viewport.width, viewport.height);
    draw(view.projectionMatrix,
         pose.getViewMatrix(view), gl);
  }
}
```



Demo & Source Code

https://codeimpl.github.io/webxr-samples/



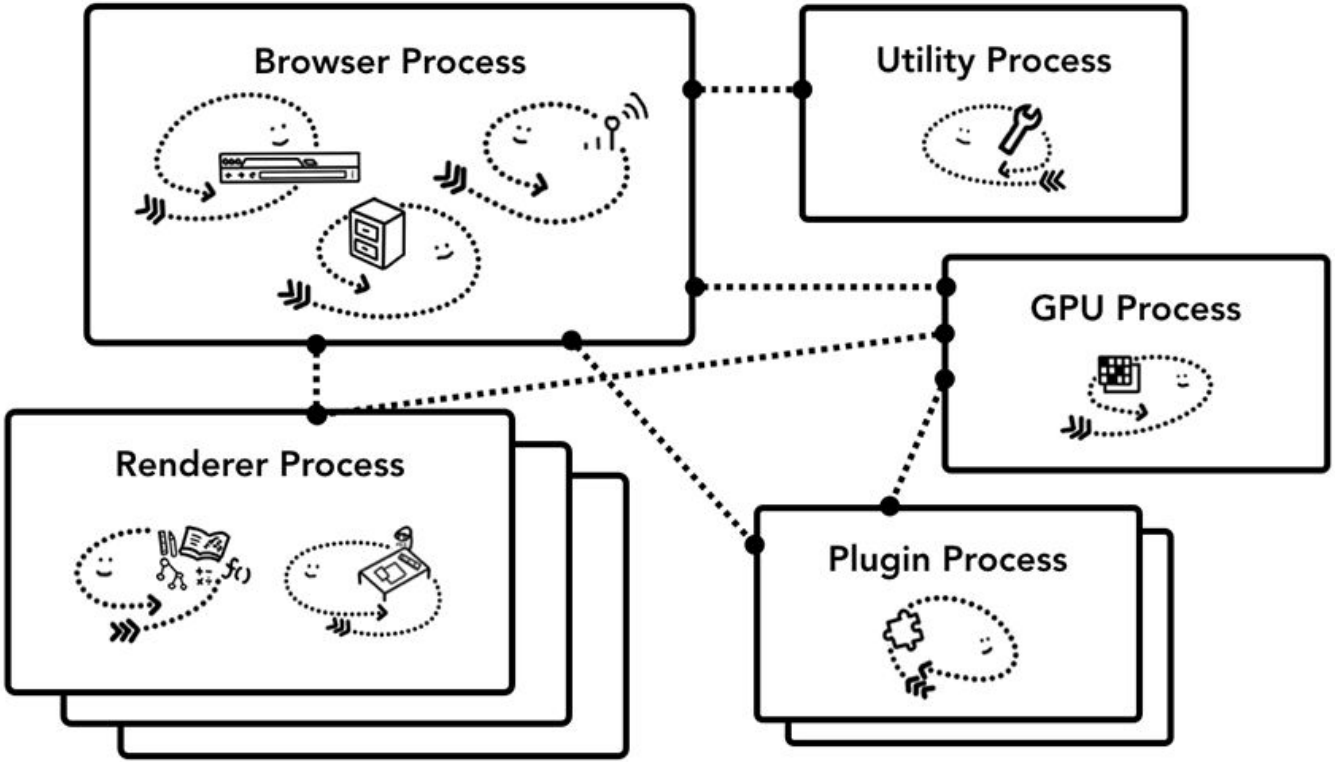
WEB XR INTERNALS IN CHROMIUM

왜 Chromium인가?

WebXR의 가장 최신 구현체이다

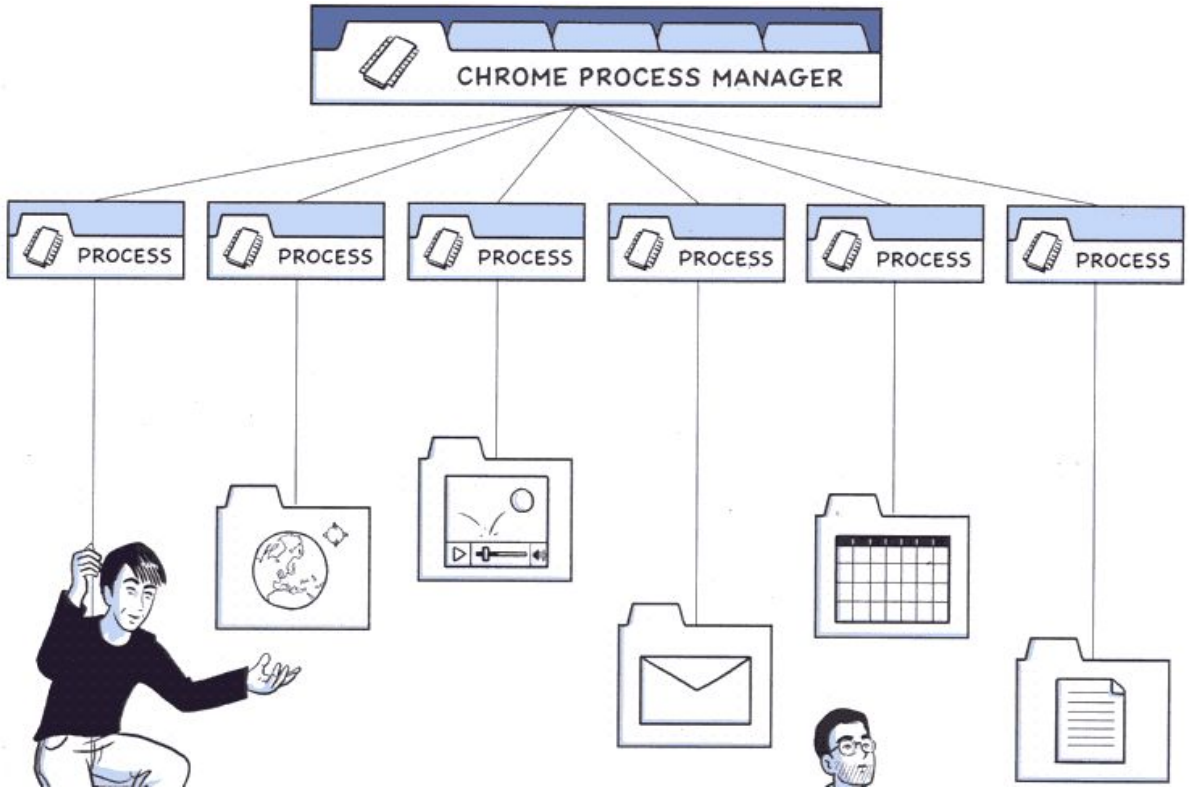
Chromium은 널리알려진 것처럼

Multi-process architecture를 사용한다


Browser Process
Utility Process
GPU Process
Renderer Process
Plugin Process

Browser

Renderer



WebXR의 관점에서..

Browser Process

XRRuntime Process

Renderer Process

XRService

Renderer와 Device를
중계해주는 역할

XRRuntime

GVR, OpenVR 등
다양한 XRDevice의
드라이버 역할

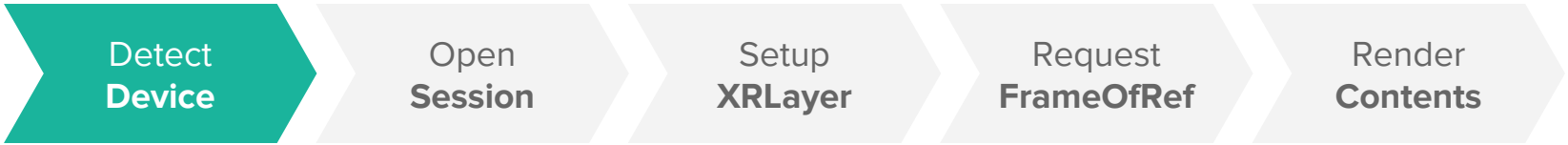
Renderer

Web개발자에게
제공되는 WebXR API의
Javascript Binding 구현

How WebXR App works

Detect **Device** → Open **Session** → Setup **XRLayer** → Request **FrameOfRef** → Render **Contents**

STEP1: Detect Device



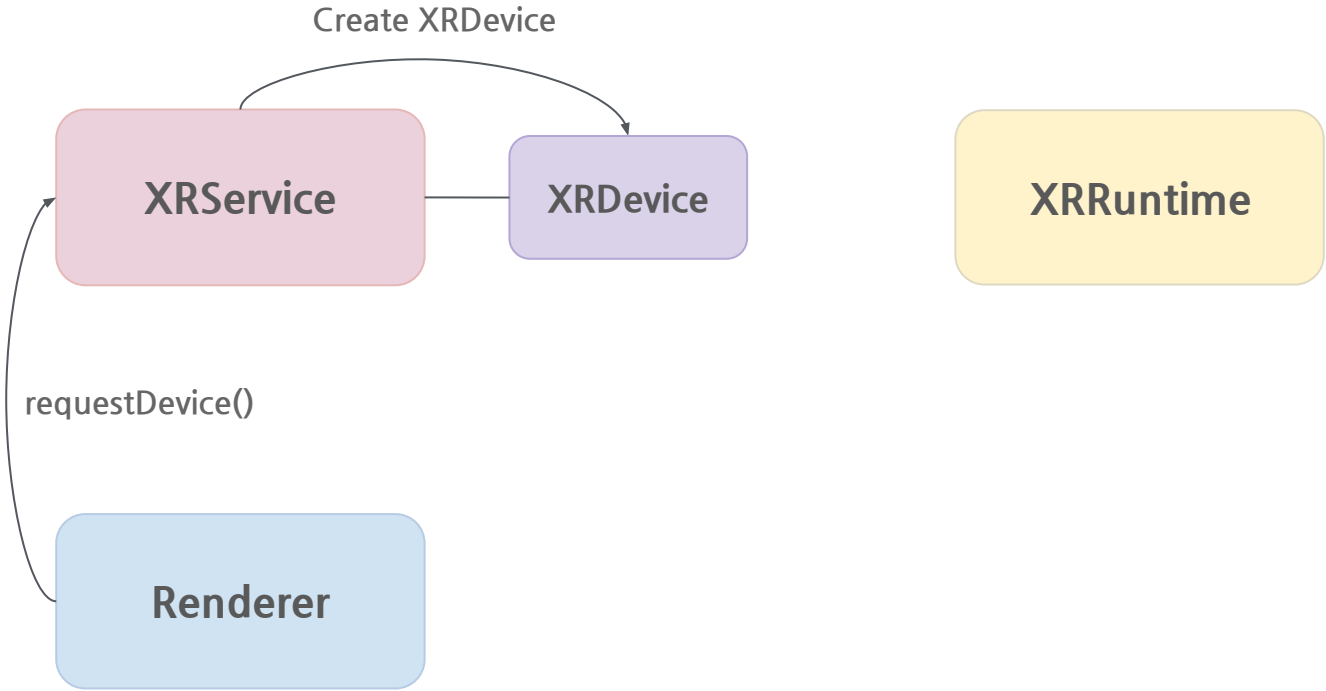
```
navigator.xr.requestDevice();
```

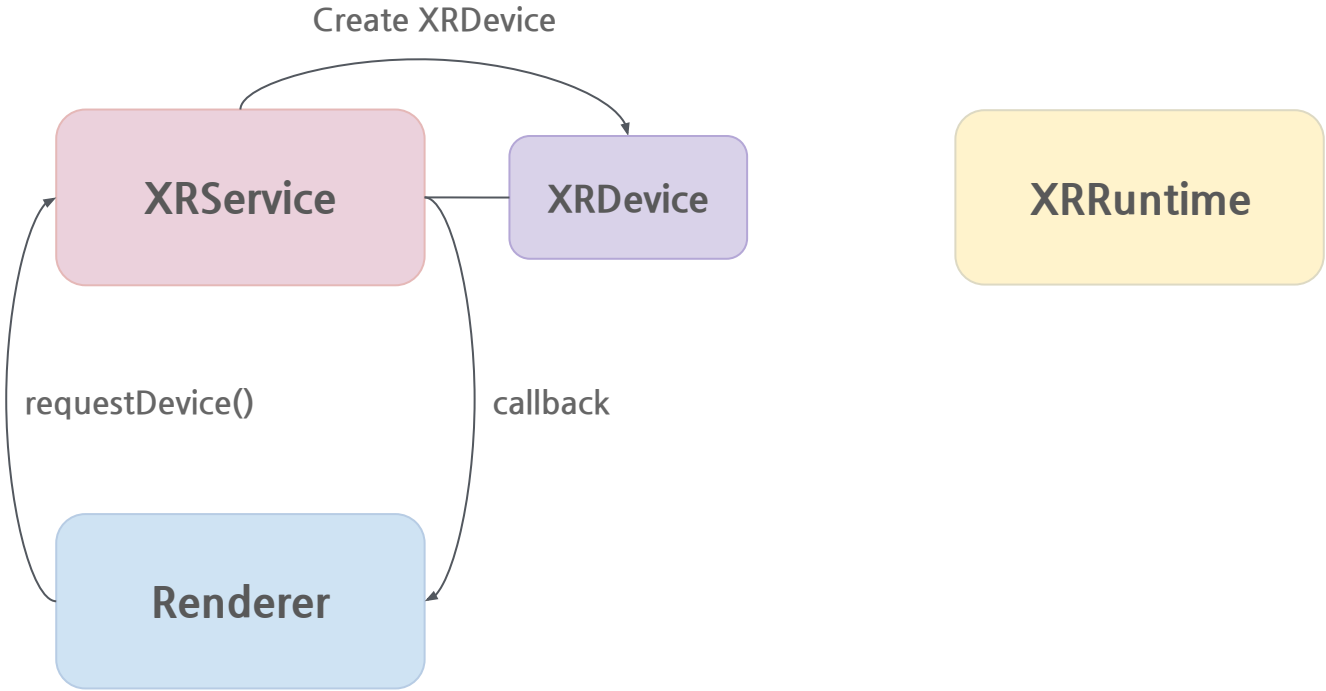
XRService

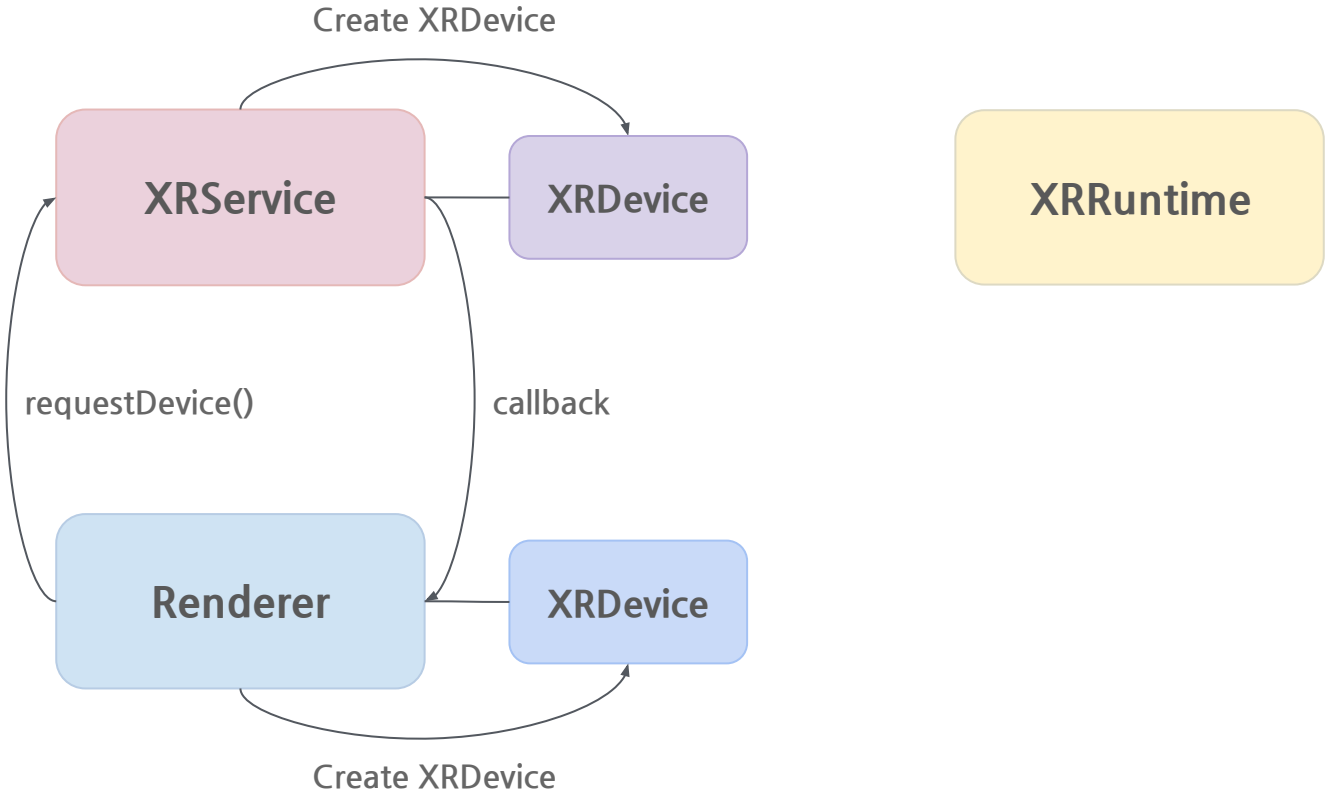
XRRuntime

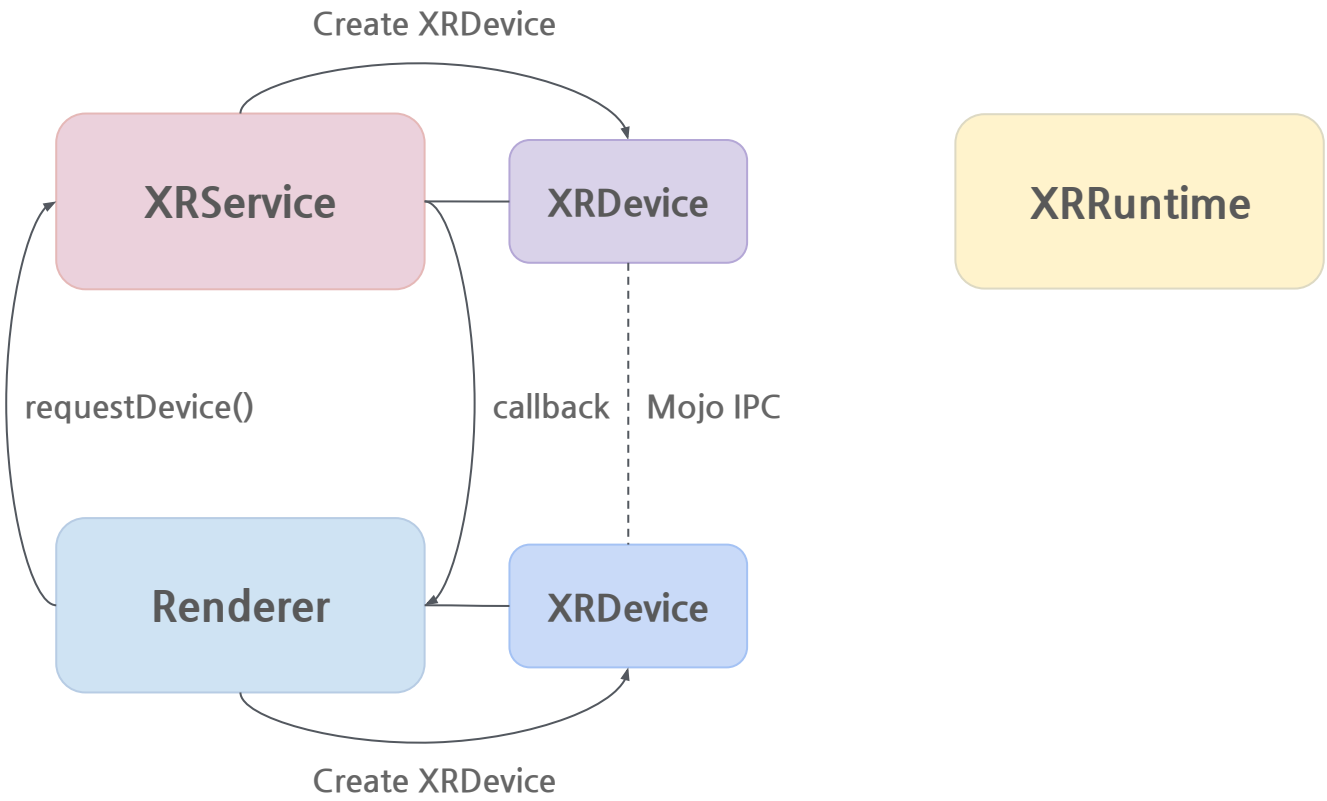
Renderer


XRService
XRRuntime
requestDevice()
Renderer

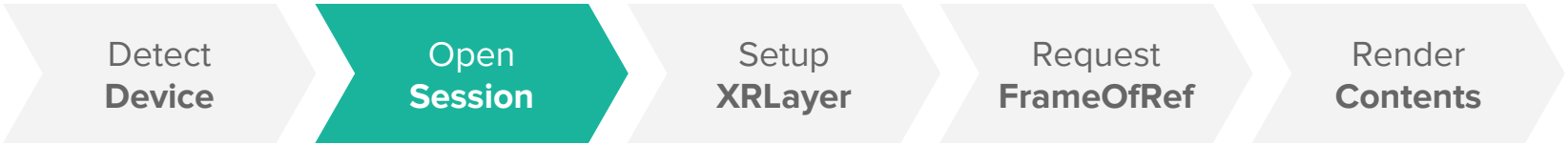

Create XRDevice
XRService
XRDevice
XRRuntime
requestDevice()
Renderer


Create XRDevice
XRService
XRDevice
XRRuntime
requestDevice()
callback
Renderer

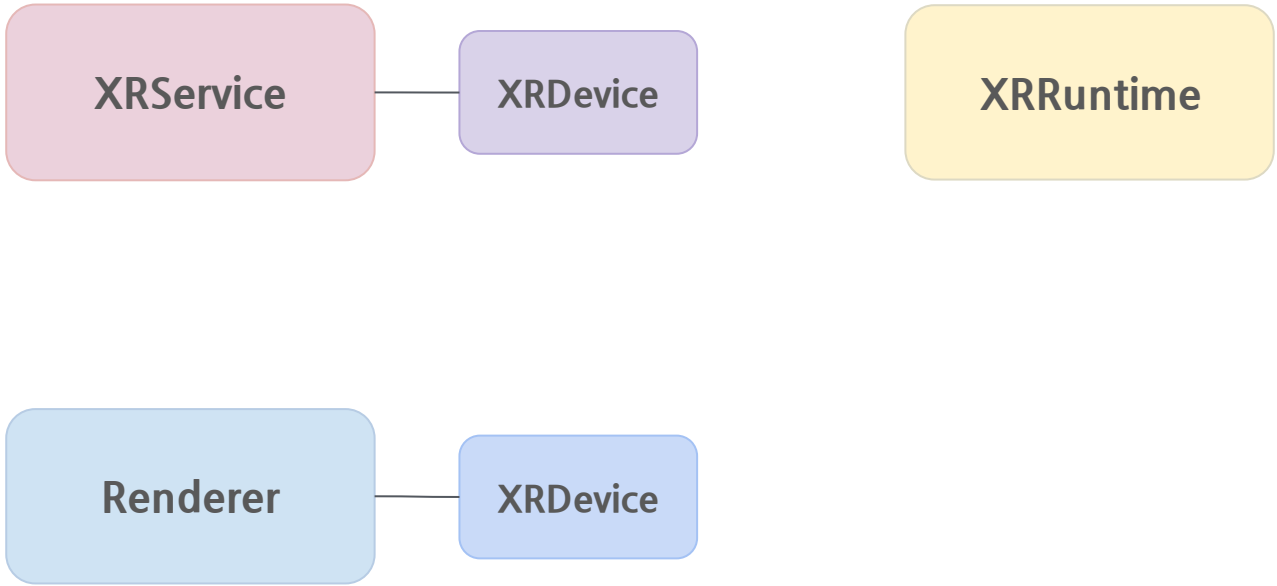

Create XRDevice
XRService
XRDevice
XRRuntime
requestDevice()
callback
Renderer
XRDevice
Create XRDevice

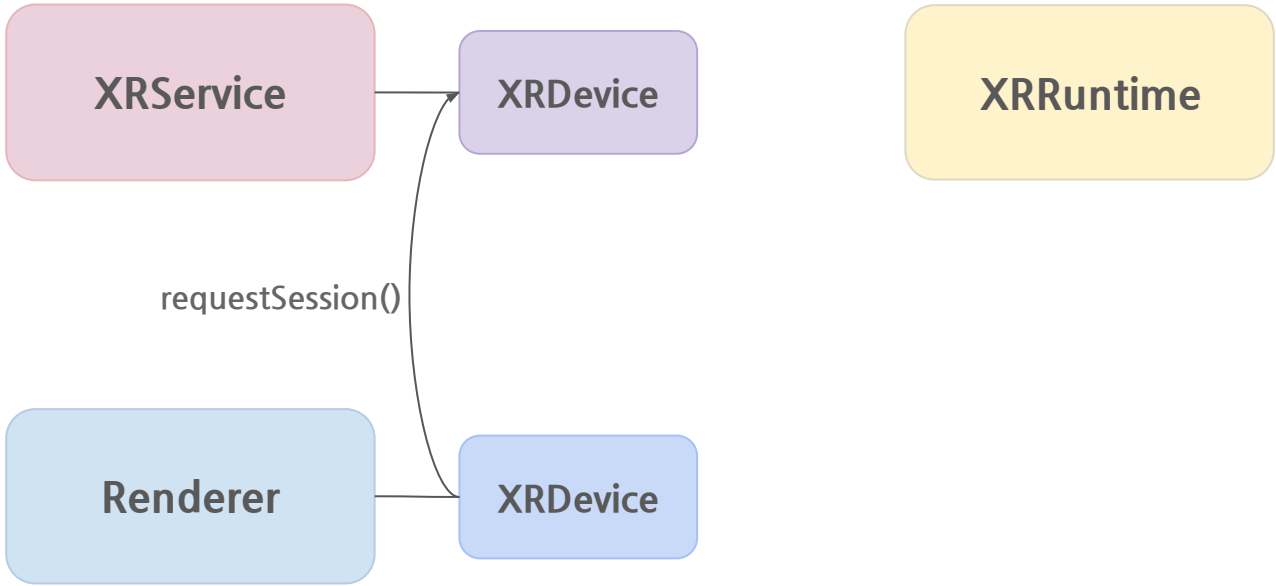

Create XRDevice
XRService
XRDevice
XRRuntime
requestDevice()
callback
Mojo IPC
Renderer
XRDevice
Create XRDevice

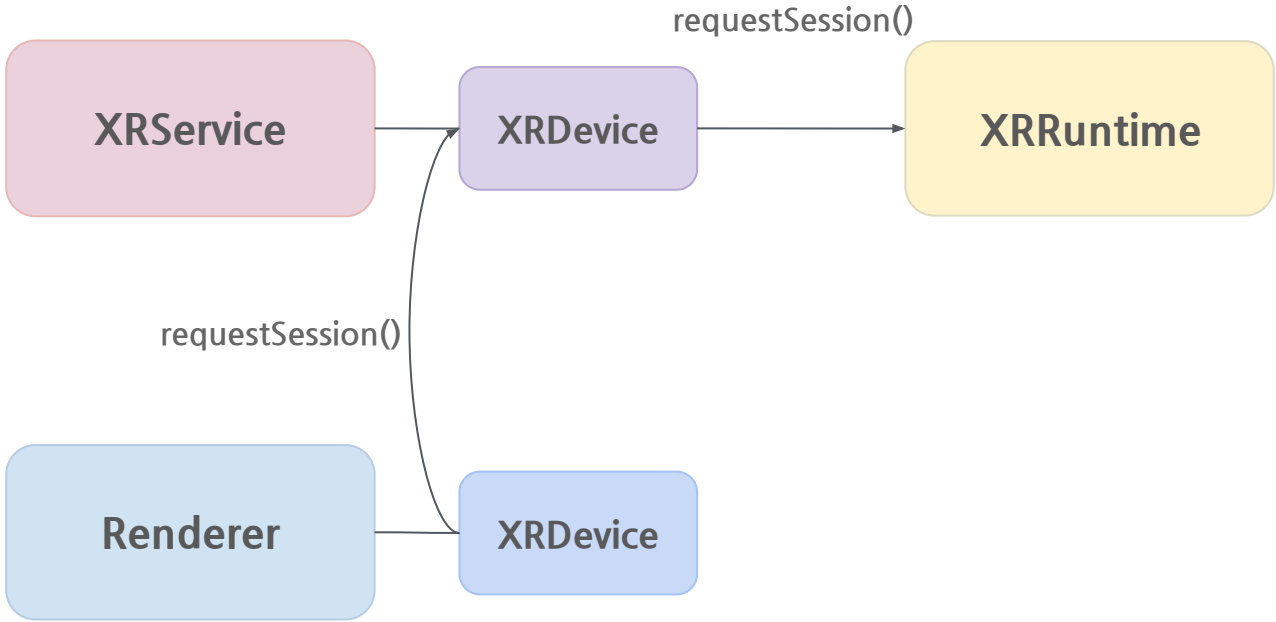
STEP2: Open Session

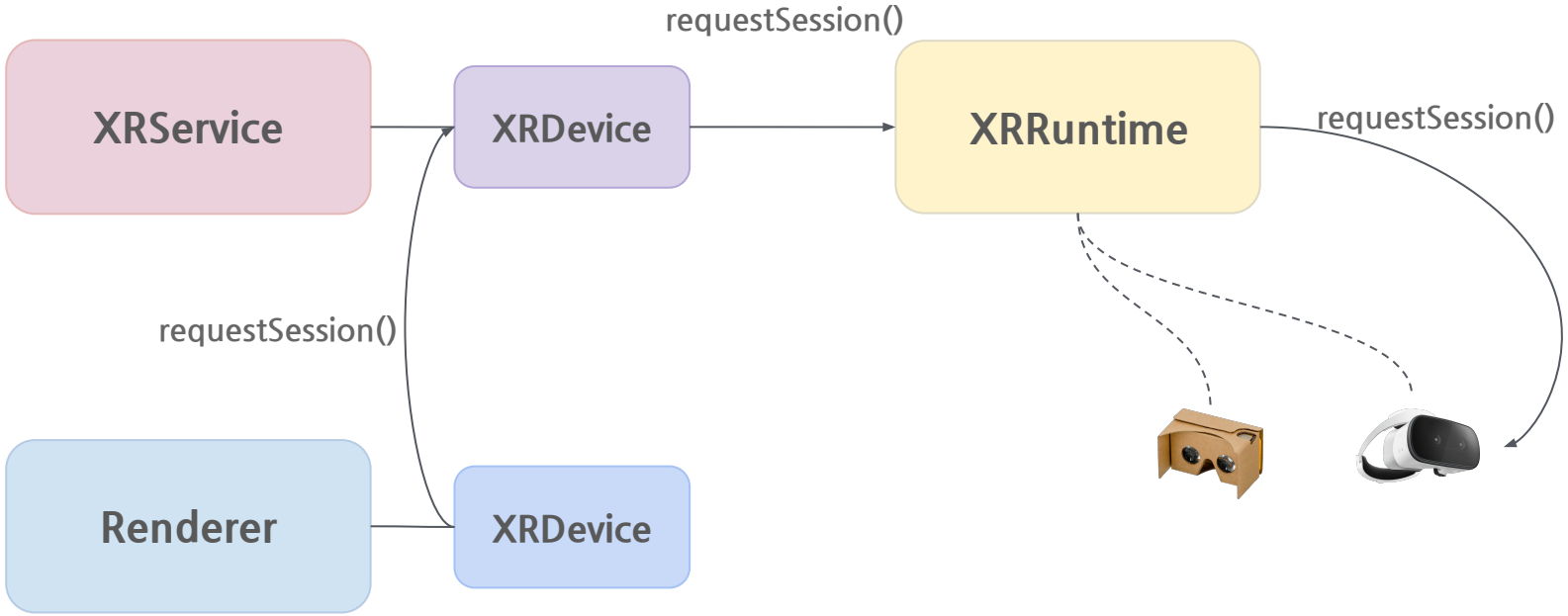


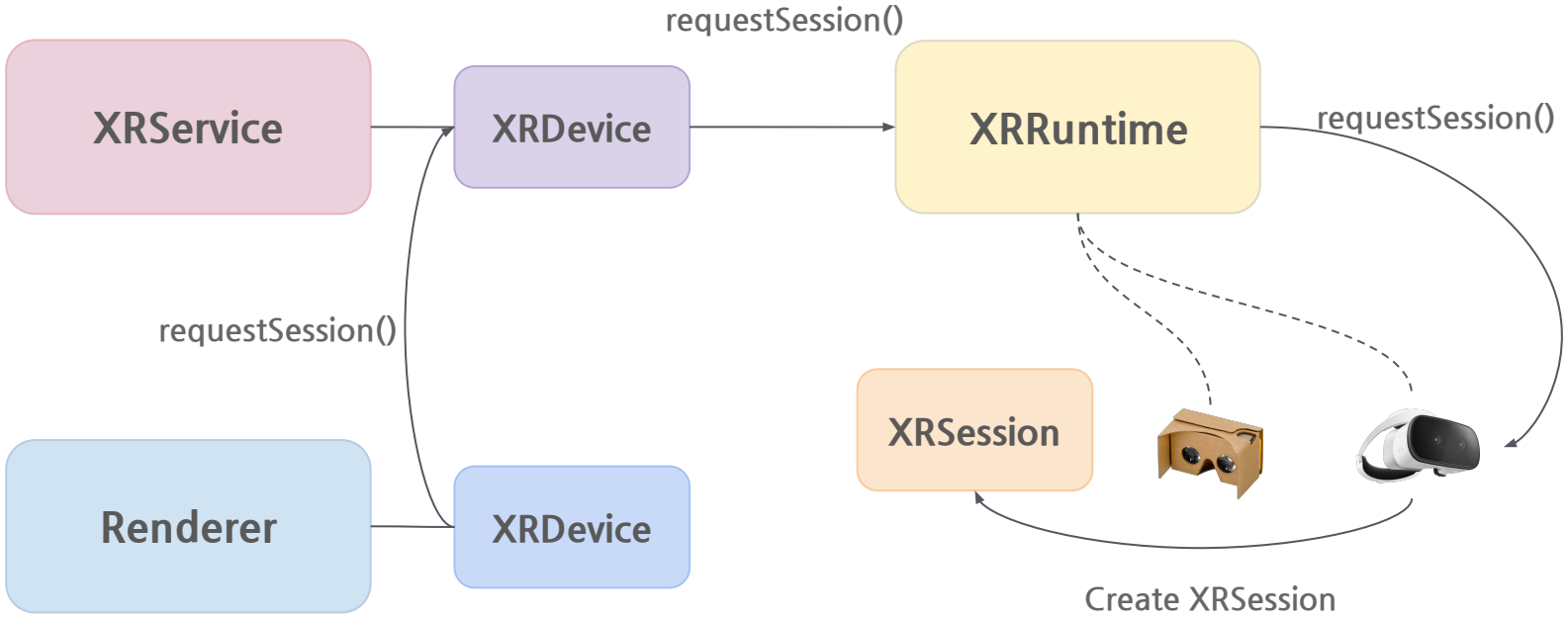
```
device.requestSession();
```

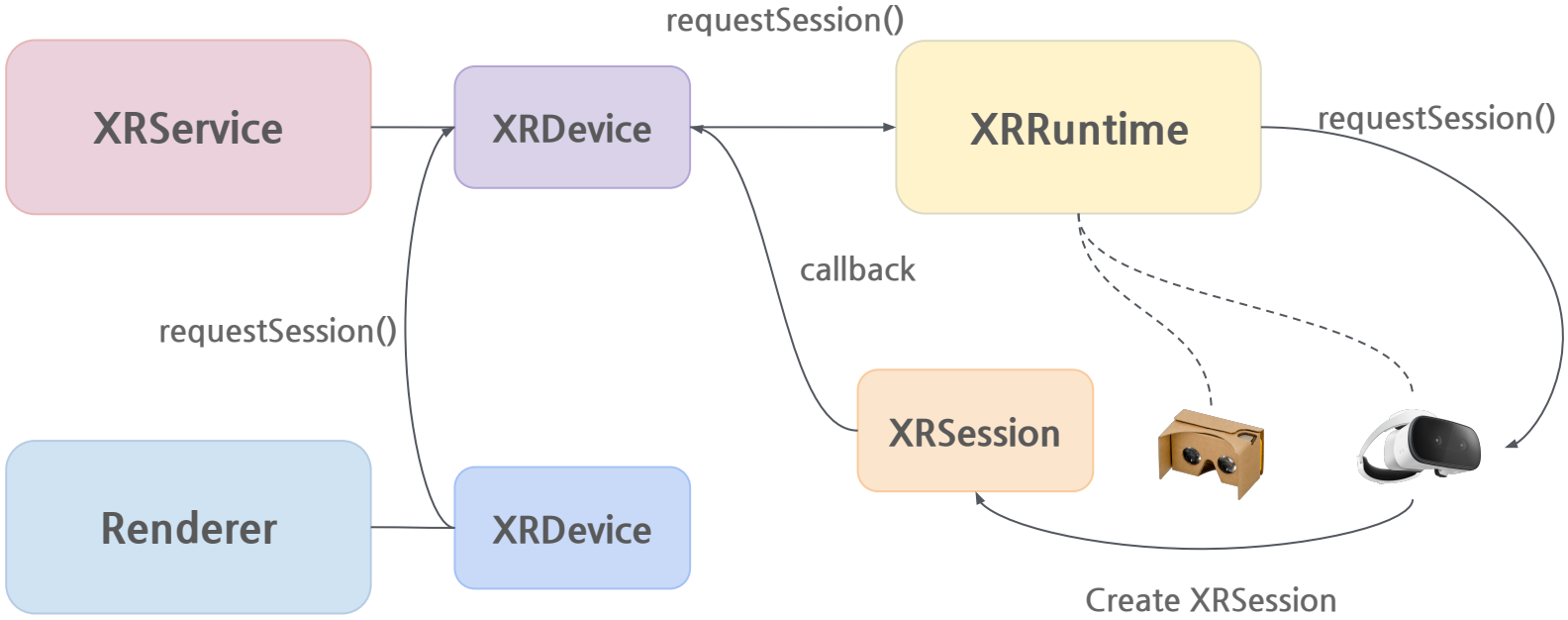

XRService
XRDevice
XRRuntime
Renderer
XRDevice

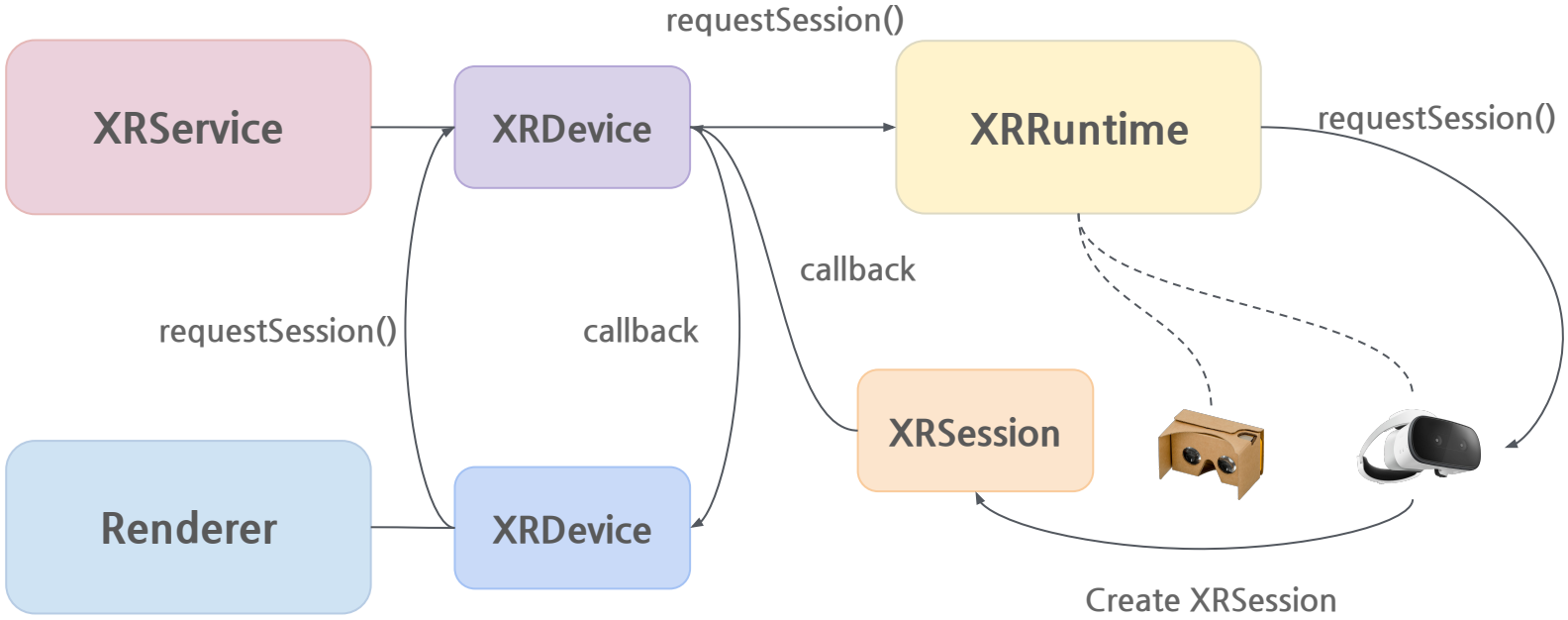

XRService
XRDevice
XRRuntime
requestSession()
Renderer
XRDevice

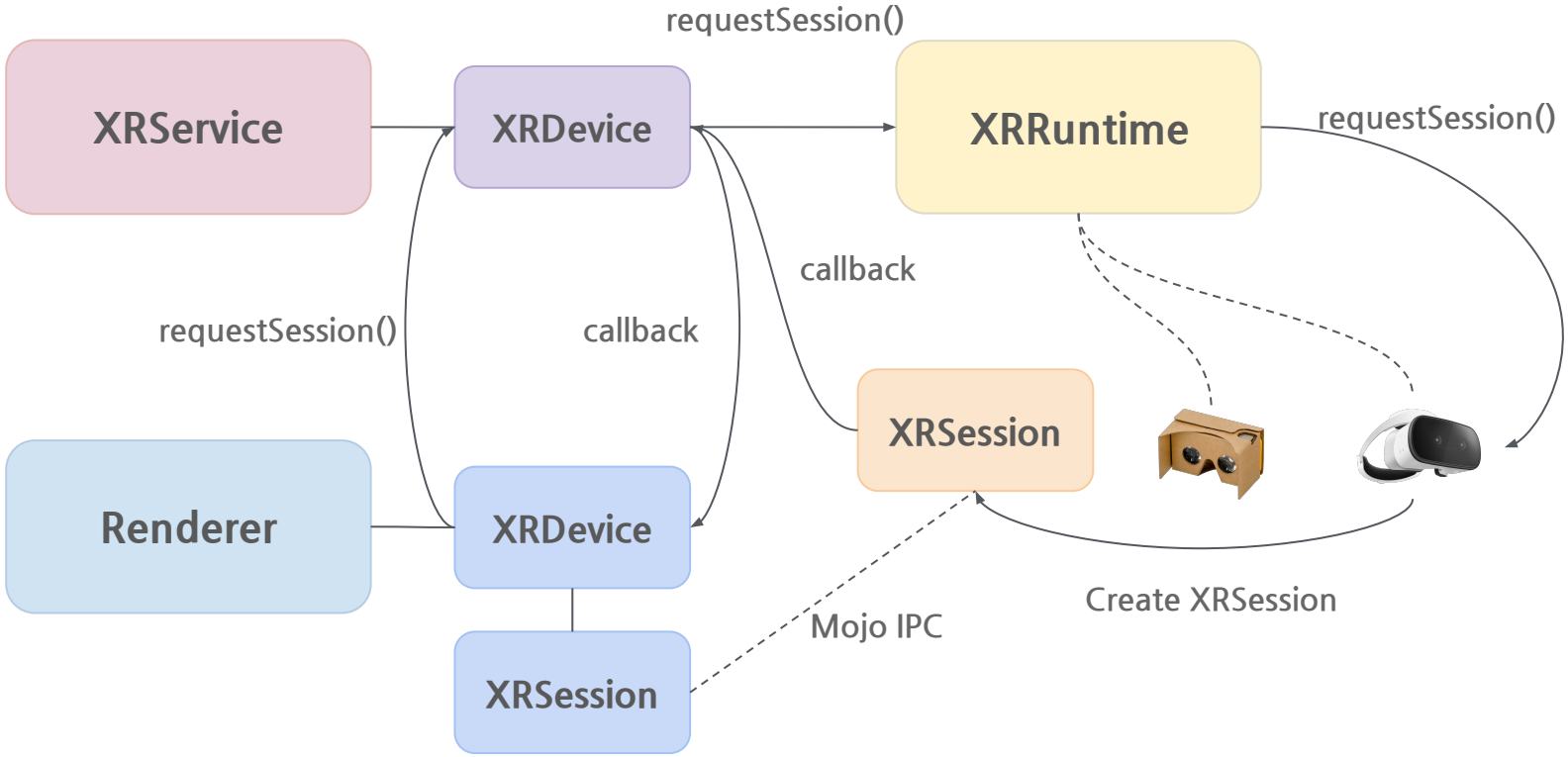

requestSession()
XRService
XRDevice
XRRuntime
requestSession()
Renderer
XRDevice


requestSession()
XRService
XRDevice
XRRuntime
requestSession()
requestSession()
Renderer
XRDevice


requestSession()
XRService
XRDevice
XRRuntime
requestSession()
requestSession()
XRSession
Renderer
XRDevice
Create XRSession

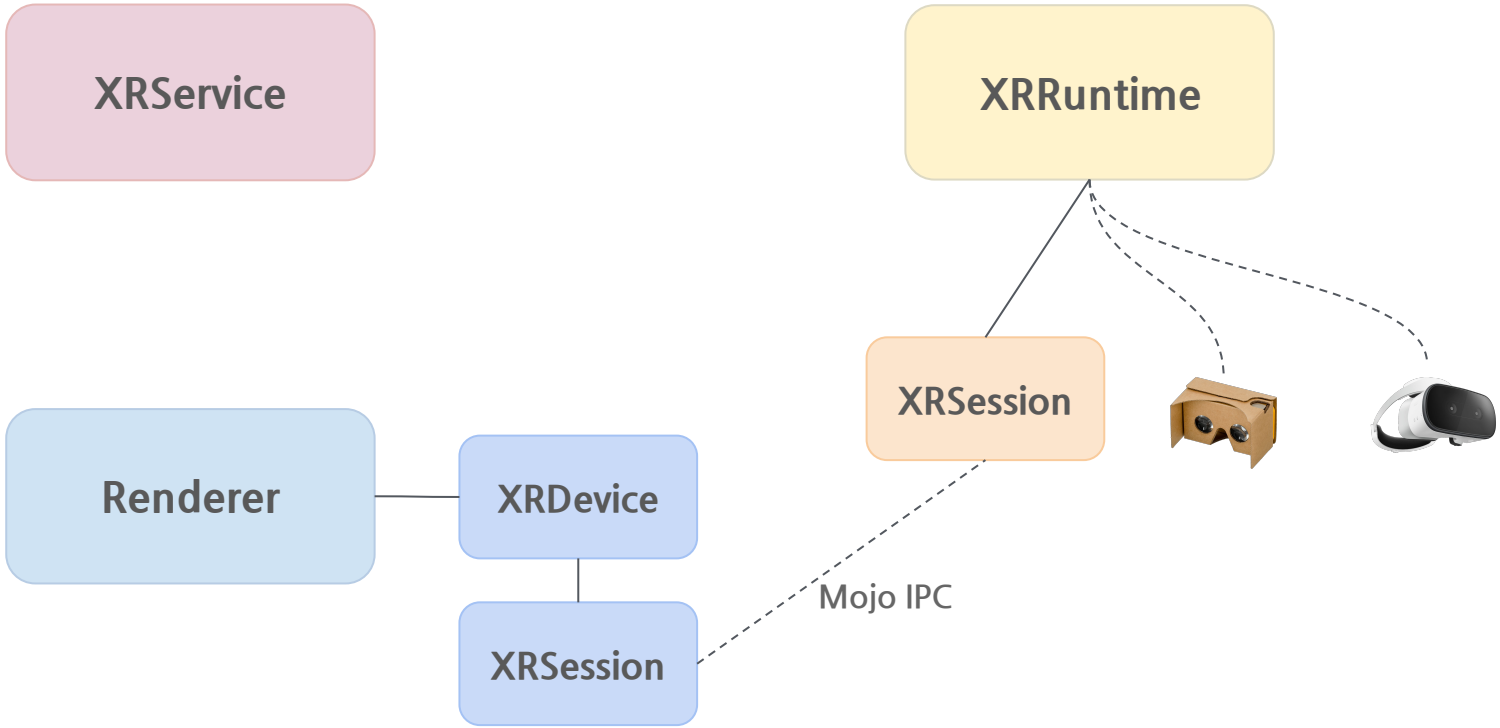

requestSession()
XRService
XRDevice
XRRuntime
requestSession()
callback
requestSession()
XRSession
Renderer
XRDevice
Create XRSession

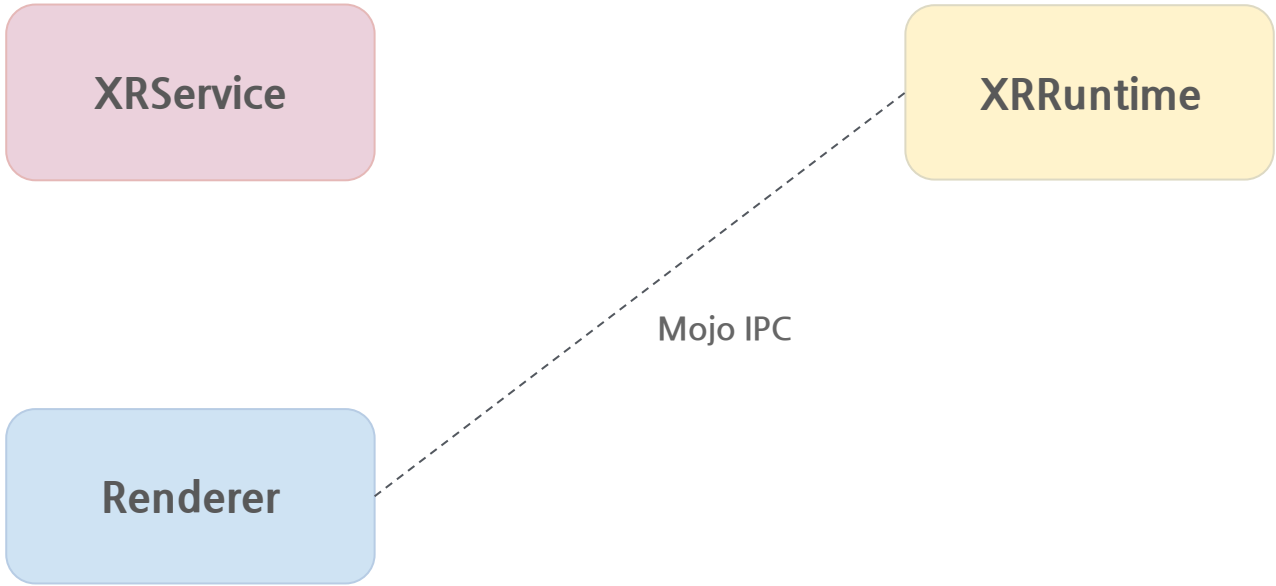

requestSession()
XRService
XRDevice
XRRuntime
requestSession()
callback
requestSession()
callback
XRSession
Renderer
XRDevice
Create XRSession


requestSession()
XRService
XRDevice
XRRuntime
requestSession()
callback
requestSession()
callback
XRSession
Renderer
XRDevice
Create XRSession
Mojo IPC
XRSession

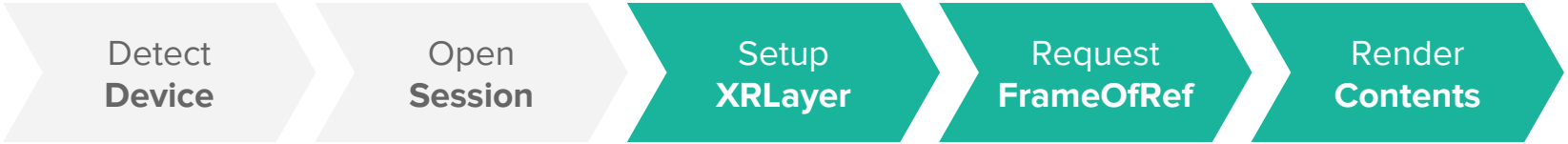
XRSession이 생성되고나면,

Renderer와 Device는 IPC로 연결된다

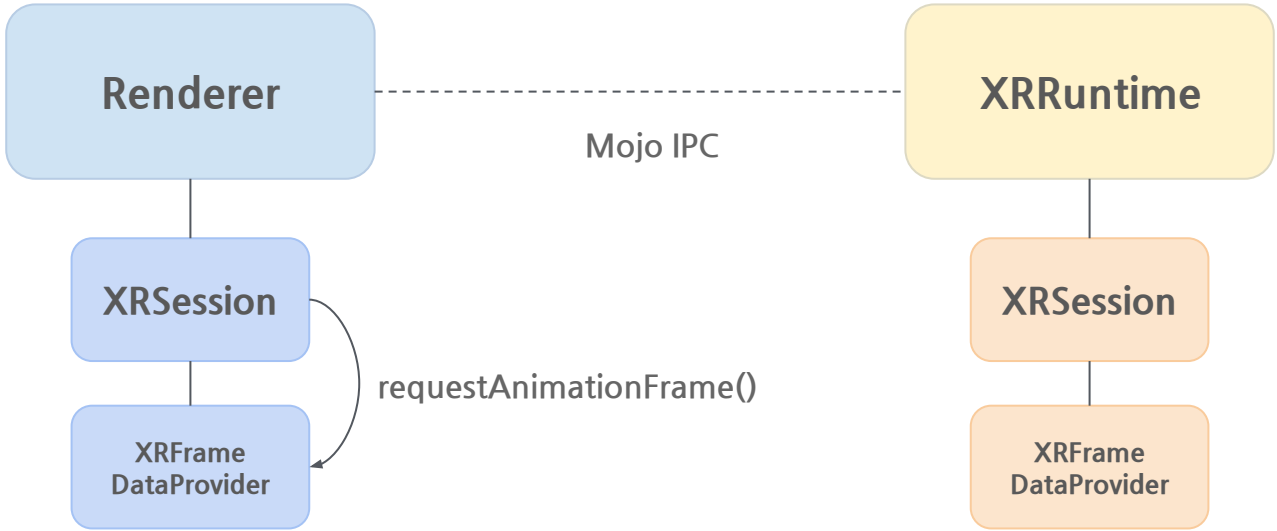

XRService
XRRuntime
XRSession
Renderer
XRDevice
XRSession
Mojo IPC

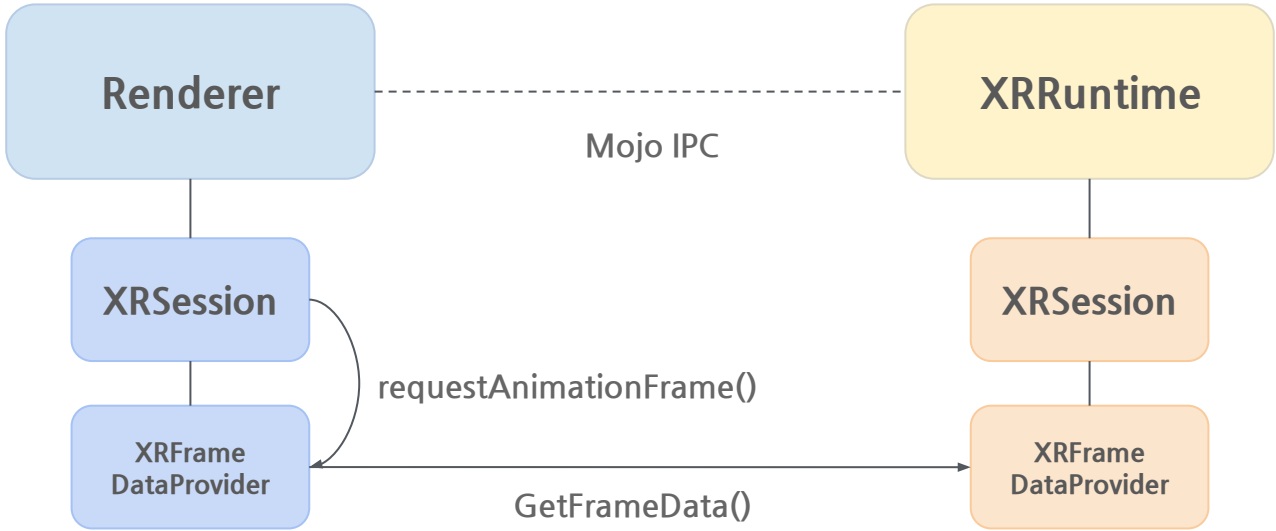

XRService
XRRuntime
Mojo IPC
Renderer

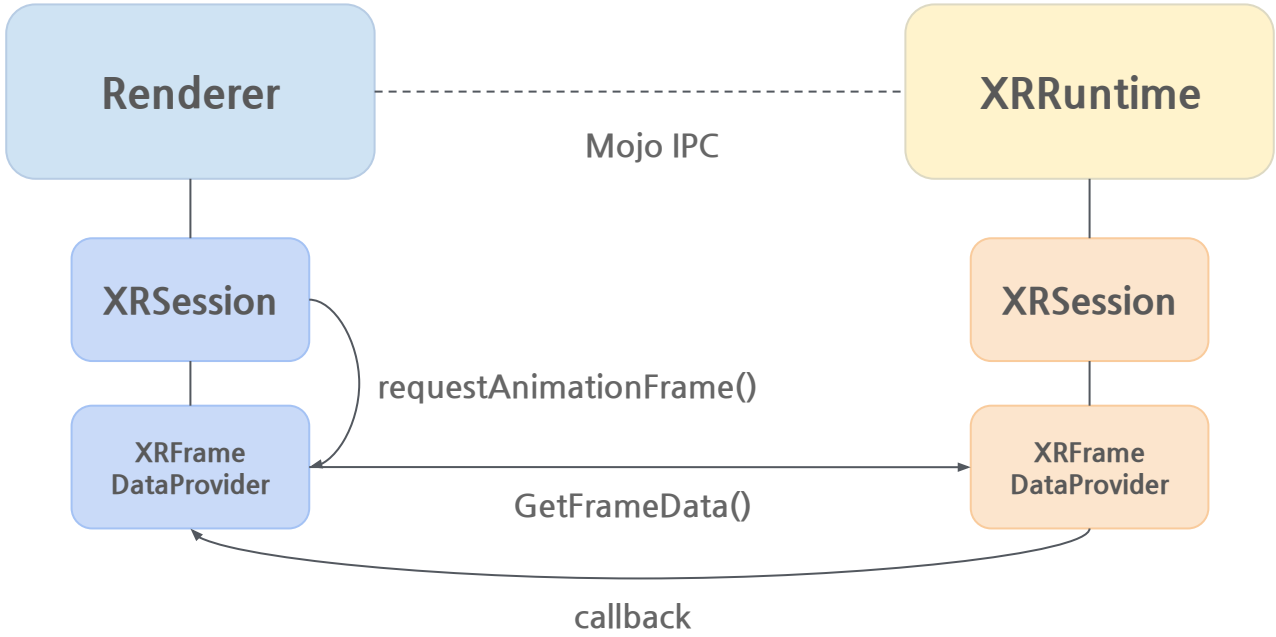
STEP3,4,5: Render Contents

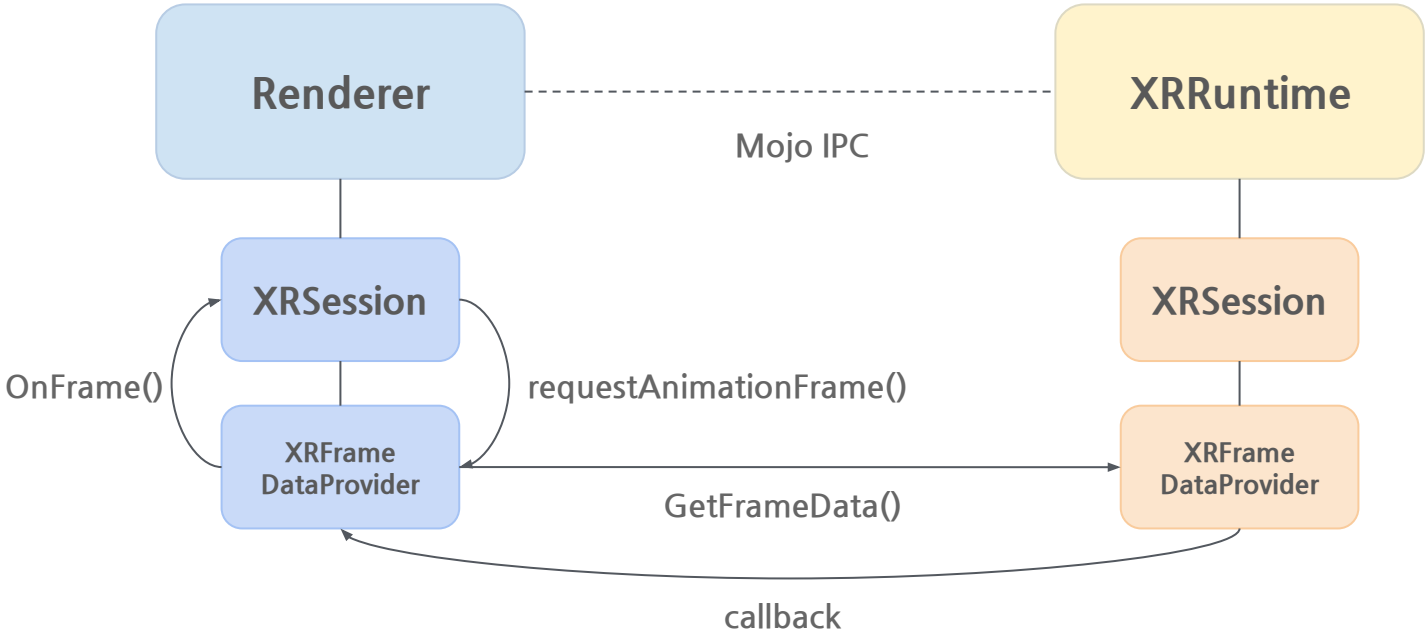


```
session.requestAnimationFrame();
```

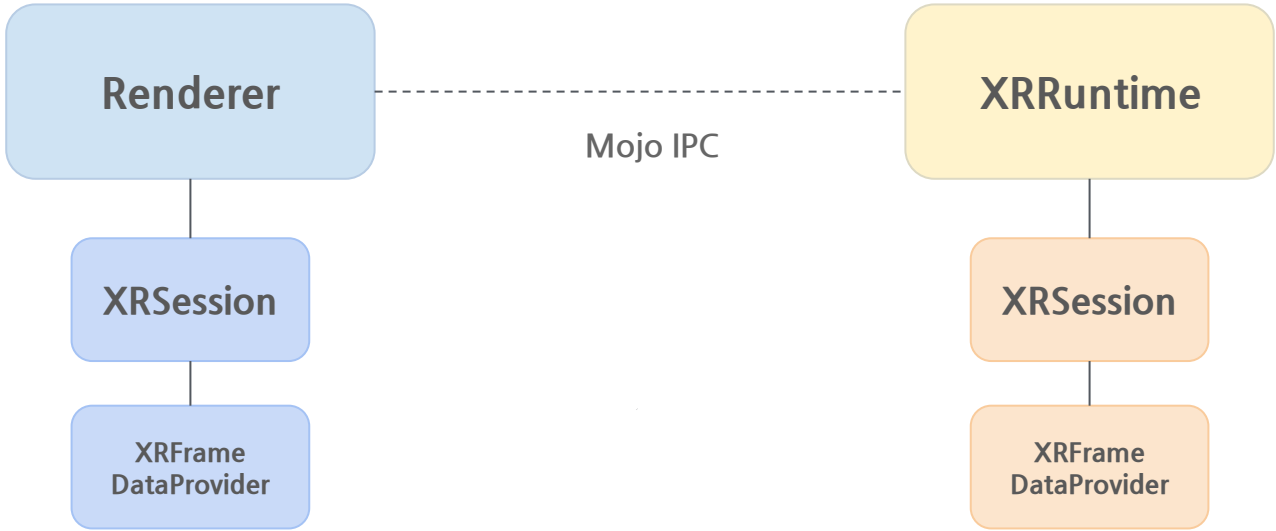

Renderer
XRRuntime
Mojo IPC
XRSession
XRSession
requestAnimationFrame()
XRFrame
DataProvider
XRFrame
DataProvider

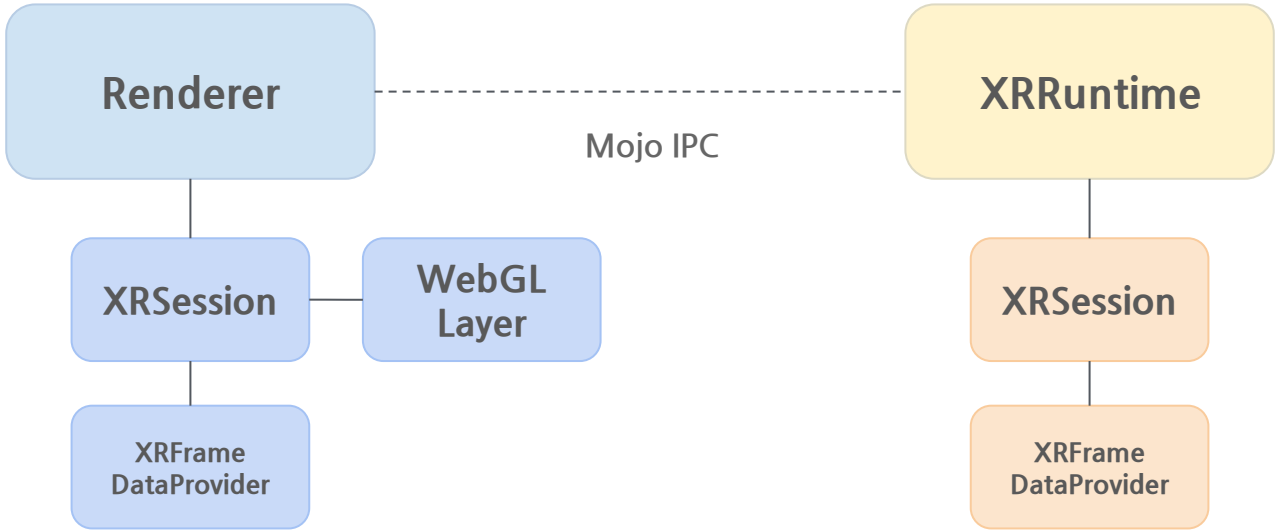

Renderer
XRRuntime
Mojo IPC
XRSession
XRSession
requestAnimationFrame()
XRFrame DataProvider
XRFrame DataProvider
GetFrameData()

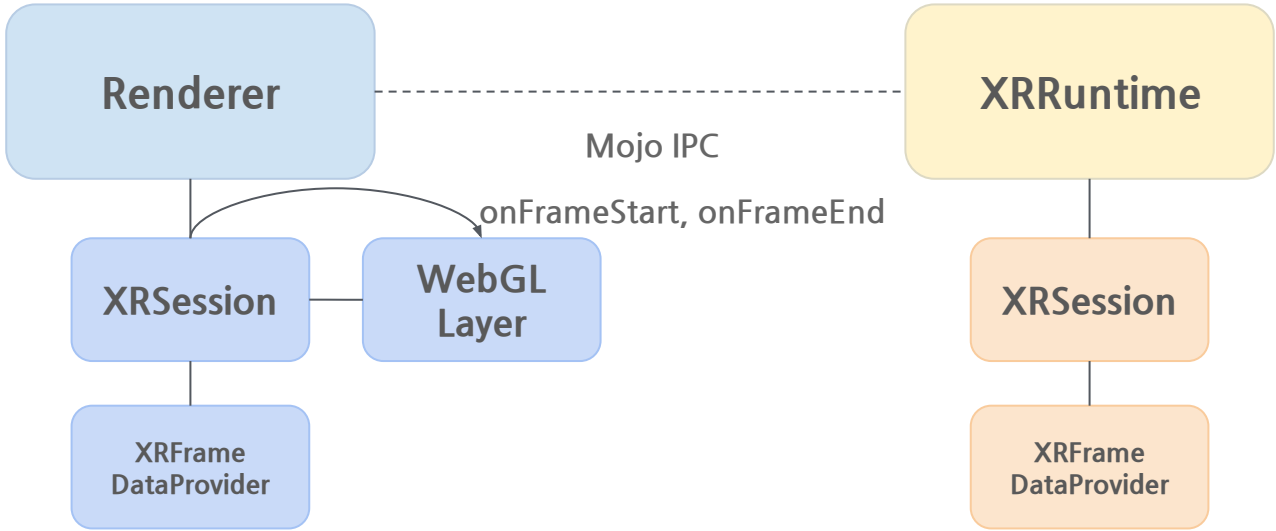

Renderer
XRRuntime
Mojo IPC
XRSession
XRSession
requestAnimationFrame()
XRFrame
DataProvider
XRFrame
DataProvider
GetFrameData()
callback

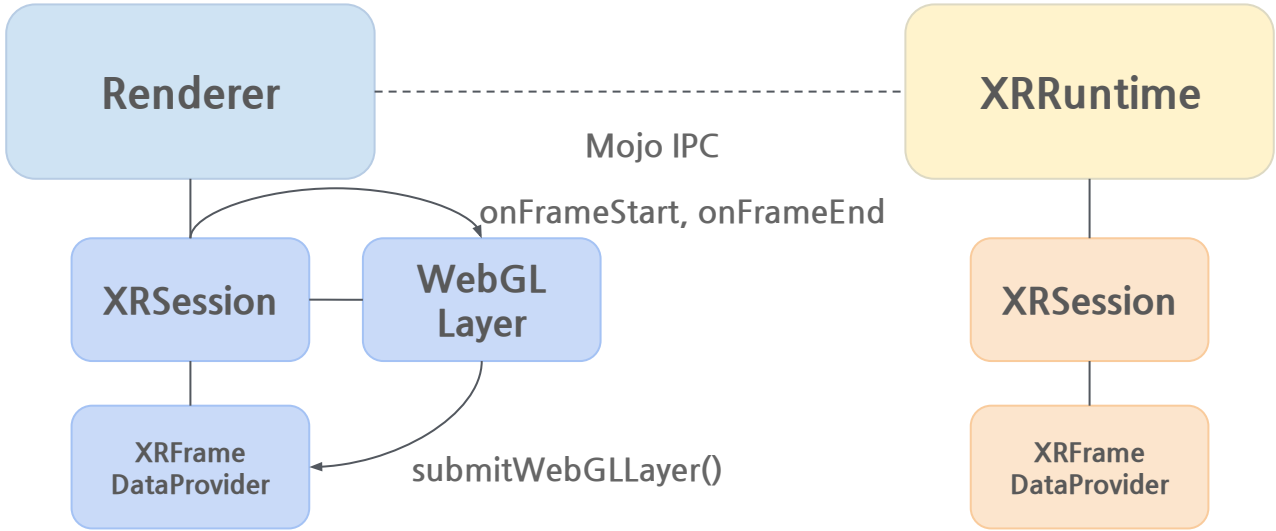

Renderer
Mojo IPC
XRRuntime
XRSession
XRSession
OnFrame()
requestAnimationFrame()
XRFrame DataProvider
XRFrame DataProvider
GetFrameData()
callback

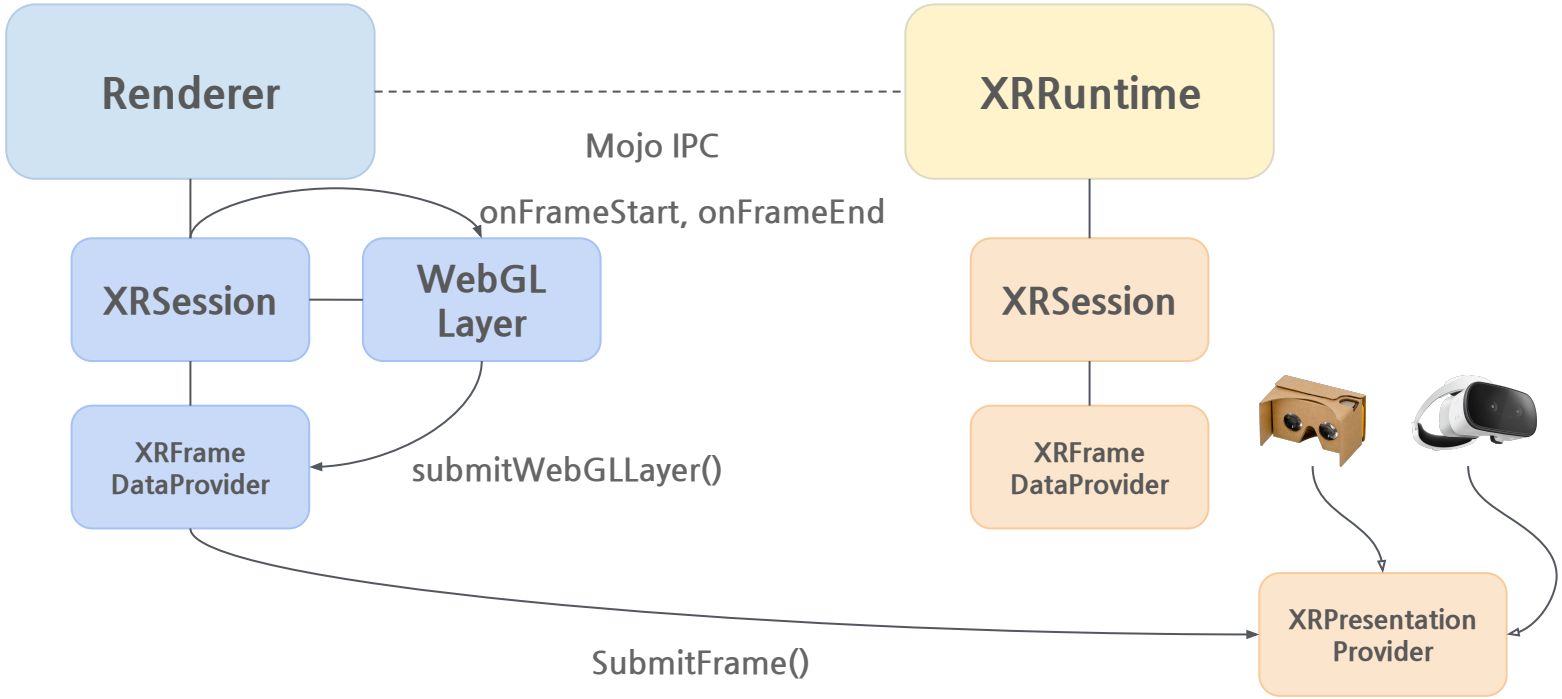
WebGLLayer, FrameOfReference는
Renderer 측에서 생성된다


Renderer
Mojo IPC
XRRuntime
XRSession
XRSession
XRFrame
DataProvider
XRFrame
DataProvider


Renderer
Mojo IPC
XRRuntime
XRSession
WebGL Layer
XRSession
XRFrame DataProvider
XRFrame DataProvider


Renderer
XRRuntime
Mojo IPC
onFrameStart, onFrameEnd
XRSession
WebGL Layer
XRSession
XRFrame DataProvider
XRFrame DataProvider


Renderer
XRRuntime
Mojo IPC
onFrameStart, onFrameEnd
XRSession
WebGL Layer
XRSession
XRFrame DataProvider
submitWebGLLayer()
XRFrame DataProvider


Renderer
XRRuntime
Mojo IPC
onFrameStart, onFrameEnd
XRSession
WebGL Layer
XRSession
XRFrame DataProvider
submitWebGLLayer()
XRFrame DataProvider
XRPresentation Provider
SubmitFrame()

Q & A

Thank you