VRDC

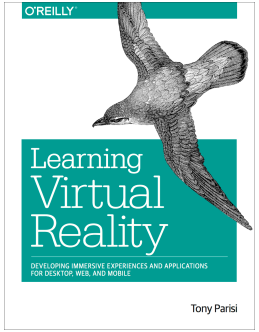
WebVR: Building for the Immersive Web

Tony Parisi
Head of VR/AR, Unity Technologies

About me

Co-creator, VRML, X3D, glTF
Head of VR/AR, Unity
tonyp@unity3d.com
Advisory
http://www.uploadvr.com
http://www.highfidelity.io
http://www.shiift.world/
Groups
http://www.meetup.com/WebGL-Developers-Meetup/
http://www.meetup.com/Web-VR/
Sites
www.tonyparisi.com
www.dopaminevr.com

Learning Virtual Reality
Programming 3D Applications with HTML and WebGL
WebGL: Up and Running
http://www.amazon.com/-/e/B008UZ888I



The Next Computing Platform

Billions invested

$14B to $120B market by 2020

A few million units have shipped

The industry is anticipating scaling to billions within five years

Q: How do we reach a billion headsets?

A: Not one app at a time.

The Next Wave: WebVR

Frictionless access
Frictionless discovery
Connected applications
Integrate the world's data

"If you haven't heard of WebVR yet, it's time to take notice."
-- VentureBeat, September 2016

The Web Goes 3D!

How it Works

Render 3D scene in WebGL - rendering standard runs on all browsers

3B computers and devices!

Use new browser APIs to track HMD pose

Experimental API - need development version of browsers

Browser API extensions for 6DOF input controllers

Supported Browsers and Devices

Browser	Devices	V1.0 API Available Date
Firefox (Nightly)	Oculus Rift, HTC Vive, OSVR	Spring 2016
Chromium (Development)	Oculus Rift, HTC Vive, OSVR	Spring 2016
Samsung Internet	Samsung Gear VR-capable phones	Spring 2016
Mobile Chromium (Beta)	Daydream-capable phones	Late 2016
Oculus "Carmel"	Samsun Gear VR-capable phones	Late 2016
Microsoft Edge	Hololens?	unannounced

A Tour of the WebVR API

1. Querying for VR displays
2. Presenting content to the VR display
3. Refreshing the VR display
4. Head tracking

Querying for VR Displays

```
myVRApp.prototype.queryVRDisplays = function() {
   if (navigator.getVRDisplays) {
      navigator.getVRDisplays().then(function (displays) {
         if (displays.length > 0) {
            vrDisplay = displays[0];
            // perform initialization using vrDisplay
       } else {
             console.log(
                  "WebVR supported, but no VRDisplays found.");
         }
      });
   } else if (navigator.getVRDevices) {
         console.log(
            "WebVR supported, but not the latest version.");
   } else {
         console.log("Your browser does not support WebVR.");
   }
}
```

displays returned in promise callback

Presenting Content to the VR Display

```
myVRApp.prototype.present = function(on) {
  if (on) {
    vrDisplay.requestPresent( { source: canvas } );
  }
  else {
    vrDisplay.exitPresent();
  }
}

// Set up VR button handling
var enterVRButton = document.querySelector( '.enterVR' );

enterVRButton.onclick = function() {
  app.present(true);
};
```

Refreshing the VR Display

```
function onAnimationFrame (t) {
  // request another animation frame for next time through
  vrDisplay.requestAnimationFrame(onAnimationFrame);
  // do the work: this is where the application
  // gets the latest HMD position/orientation,
  // updates the scene objects and animations,
  // and renders using WebGL;
  // ... code omitted ...
  // finally, submit the frame
  vrDisplay.submitFrame();
}

// start up the run loop
vrDisplay.requestAnimationFrame(onAnimationFrame);
```

set up callback to refresh display (90hz!)

copy the bits from the canvas

Head Tracking

```
function onAnimationFrame (t) {
  // request another animation frame for next time through
  vrDisplay.requestAnimationFrame(onAnimationFrame);
  // do the work: this is where the application
  // gets the latest HMD position/orientation:
  var pose = vrDisplay.getPose();
  // now we update scene objects and animations,
  // based on the pose value, time and other inputs;
  // and render using WebGL;
  // ... code omitted ...
  // finally, submit the frame
  vrDisplay.submitFrame(pose);
}

// start up the run loop
vrDisplay.requestAnimationFrame(onAnimationFrame);
```

get HMD position/orientation

The WebVR Ecosystem

Frameworks

JavaScript libraries: Three.js, Babylon.js
Markup systems: GLAM, SceneVR, A-Frame, ReactVR
WebVR Polyfill

```
<glam>
    <scene>
        <cube id="photocube”></cube>
    </scene>
</glam>
```

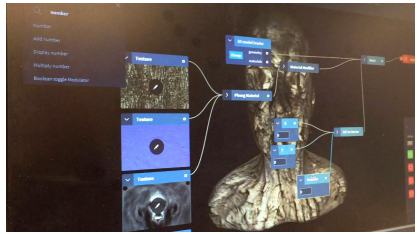
```
#photocube {
    image:url(../images/photo.png);
}
```

Formats

The “JPEG of 3D”

Tools

Unity, Unreal export
Browser-based VR creation



Responsive WebVR

Big challenges
in designing for
desktop x mobile
x 2D x 3D

WebVR Polyfill:
emulate API for existing
mobile browsers
for Cardboard VR

Development Status

Developer builds of Chrome, Firefox (desktop and mobile)
Samsung Internet browser for Gear VR
Oculus “Carmel” for Gear VR
Coming soon to Daydream VR
Coming soon for Microsoft Edge
WebVR 1.0 API - preliminary spec
http://mozvr.github.io/webvr-spec/
W3C Community Group
https://www.w3.org/community/webvr/
Slack Channel
https://webvr.slack.com/
Examples
https://webvr.info/samples/

https://webvr.info/

VRDC

WebVR: Building for the Immersive Web

Tony Parisi
Head of VR/AR, Unity Technologies