From Idea to Virtual Reality
An Intro to WebVR

Andrea Hawksley, eleVR, HARC
Who am I?
eleVR
eleVR.com

An SF-based research group exploring immersive media, particularly virtual and augmented reality, as a tool to reveal new ways to understand our world and express ourselves within it.

Vi Hart, Emily Eifler, Andrea Hawksley, Evelyn Eastmond, Elijah Butterfield
A Gentle Introduction to WebVR
Why Make WebVR?

Pros

- **Hardware Agnostic**: Same code easily accessible across VR devices
- **Easy for Users**: Does not require downloading and installing specialized software per experience
- **Open, Accessible, and Linked**

Cons

- **Lag**: Performance may be worse than native application
- **Development Stack**: Forced to use Javascript and WebGL
- **Under Development**: Not yet supported by all browsers; API subject to change

[https://iswebvrready.org/](https://iswebvrready.org/)
Why Use ThreeJS?

- Many standard building blocks already built out for you
- Fewer things to learn (need to learn JS anyways for web dev)
- Straight WebGL is much harder to debug
- Very large, well-documented, and thorough open source library
Examples of WebVR Projects
Hypernom

hypernom.com

A 4D “Pacman” style game

Vi Hart, Andrea Hawksley, Henry Segerman, and Marc ten Bosch
eleVR Player

player.elevr.com

First web player for spherical videos in VR (both mono and stereo)

Andrea Hawksley and Andrew Lutomirski
Float
elevr.com/float

A puzzle game where you visit islands to make them come alive. Written with the HTC Vive in mind.

Elijah Butterfield, Emily Eifler, Vi Hart, and Andrea Hawksley
Draw


Really basic drawing tool intended to be evocative of KidPix and early MSPaint

Vi Hart and Andrea Hawksley
Marco Polo

https://github.com/hawksley/marcopolo

VR Marco Polo. A visual variant for multiple players with one headset and controller setup (HTC Vive).

Andrea Hawksley
Plane Sight

www.math.miami.edu/~kenken/planesight.html

Three VR Plane Fields created by a participant in my intro webVR workshop earlier this year at ICERM “Illustrating Mathematics” conference at Brown

Ken Baker
Getting Started
Downloads

WebVR-enabled browser (for headset development)

- Chromium for VR: https://webvr.info/get-chrome/
- Firefox Nightly: https://mozvr.com/#developers

Code/text editor

- Sublime Text (recommended): https://www.sublimetext.com/

WebVR Boilerplate

- Recommended three.js boilerplate from Boris Smus: https://github.com/borismus/webvr-boilerplate

WebVR API Emulator from Jaume Sanchez (for emulating a headset in Chrome): https://github.com/spite/WebVR-Extension
Workshop Specific Resources

Including these slides!

http://web.mit.edu/hawksley/Public/IntroToWebVR
Resources

General WebVR Info: https://webvr.info/

WebVR 1.0 API: https://w3c.github.io/webvr/

WebVR API Info (not yet fully updated for 1.0): https://developer.mozilla.org/en-US/docs/Web/API/WebVR_API

Three.js Resources: https://threejs.org/

WebGL, WebAudio, Controller APIS... And other JS resources can be found around the web/physical world
Making our VR World
Setup Steps

1. Pull out JS from index.html file in boilerplate
2. Name your project: <title>Weird Reality Demo</title>
3. Setup localhost
   a. If you have python installed:
      % python -m SimpleHTTPServer 8000
4. Remove boilerplate objects that you aren’t using
5. Make sure you haven’t broken everything
// Create 3D objects.
var tetGeometry = new THREE.TetrahedronGeometry(1);
var tetMaterial = new THREE.MeshNormalMaterial();
var tet = new THREE.Mesh(tetGeometry, tetMaterial);

// Position mesh
tet.position.set(2, controls.userHeight, -1);

// Add mesh to your three.js scene
scene.add(tet);
Don’t Forget to Add Lighting

var light = new THREE.AmbientLight( 0x404040 );
scene.add( light );

var light2 = new THREE.PointLight( 0xffffff, 1, 100 );
light2.intensity = 1;
scene.add( light2 );

var light3 = new THREE.DirectionalLight( 0xffffff );
light3.position.set( 0, 1, 1 ).normalize();
scene.add(light3);
Use Loops and Arrays

// create Five Intersecting Tetrahedra:
var tetGeometry = new THREE.TetrahedronGeometry(1);
var tet = new Array(5);
var fit = new THREE.Object3D();

var t = ((1 + Math.sqrt(5))/2);
var fturn = 6.283/5;
var axis = new THREE.Vector3( t, 1, 0 );
axis.normalize();

for(var i = 0; i < 5; i++) {
    tet[i] = new THREE.Mesh(tetGeometry, new THREE.MeshLambertMaterial());
    tet[i].rotateOnAxis(axis, i*fturn);
    fit.add(tet[i]);
}

fit.position.set(-2, controls.userHeight+1, -2);
scene.add(fit);
Maybe Some Images?

// Tip: Use a spherical camera or your phone to capture a spherical photo
// Add an equirectangular panorama
var pano;
var loader = new THREE.TextureLoader();
loader.load('Greeley_pan_small_stars.jpg', onTextureLoaded);

function onTextureLoaded(texture) {
    var geometry = new THREE.SphereGeometry(1000, 32, 32);
    var material = new THREE.MeshBasicMaterial({
        map: texture,
        side: THREE.BackSide
    });

    pano = new THREE.Mesh(geometry, material);
    pano.position.y = 190;
    scene.add(pano);
}
Animate Your Reality

var lastRender = 0;
function animate(timestamp) {
    var delta = Math.min(timestamp - lastRender, 500);
    lastRender = timestamp;

    // Apply rotation
    fit.rotation.y += delta * 0.0003;
    fit.rotation.x += delta * 0.00005;

    // Update VR headset position and apply to camera.
    controls.update();

    // Render the scene through the manager
    manager.render(scene, camera, timestamp);
    requestAnimationFrame(animate);
}
window.addEventListener("keydown", onkey, true);
function onkey(event) {
    event.preventDefault();
    if (event.keyCode == 32) { // space bar
        isRotation = !isRotation;
    }
}

document.body.addEventListener( 'click', onClick);
function onClick(event) {
    isRotation = !isRotation;
}
Interact with Your Reality

```javascript
var lastRender = 0;
var isRotation;
function animate(timestamp) {
  ...
  if (isRotation) {
    // Apply rotation
    fit.rotation.y += delta * 0.0003;
    fit.rotation.x += delta * 0.00005;
  }
  ...
}
```
Useful to know about this boilerplate

- The “camera” is located where the headset is:
  - camera.position => the headset position
  - camera.rotation => the headset rotation
- Does not currently have controller support
- Does not have manual positioning for camera
- You need relatively few of its files, and should clean it up
  - But this can be tricky…
- Defaults to camera not at (0,0,0), but at userHeight of 1.6m
  - Set controls.standing = false to change this
Gotchas!
Problems (you’re likely to run into)

- Typos!
- Forgot to add object to scene
- Forgot to add lights to scene
- Cross Origin (CORS) Issues
  - Run from localhost while developing
  - Make sure CORS is enabled (in headers) when you use cross origin resources
- Check if your scale and/or positioning if way off
- Are you inside an object? Point of View
  - Did you remember to position the object?
Debugging

- Use the Developer Console
  - First, check for errors in the console
  - Add Breakpoints to see status of variables and narrow down when your code breaks
- Use the WebVR API Emulation extension (see earlier download rec)
- Don’t leave print lines everywhere
- Create a Debug Flag
  - Put Console.log, Console.time functions inside of if statements that turn on and off depending on Debug Flag status
- Google is your friend!
That was too hard...
A-Frame

https://aframe.io/

Use markup to create webVR experiences.

Include aframe.js

HTML like:
...
<body>
  <a-scene>
    <a-sphere ...>
      <a-sphere ...>
    </a-sphere>
  </a-scene>
</body>

MozVR (Mozilla)
React VR

https://facebook.github.io/react/

Also use markup to create webVR experiences.

```html
<View>
  <VrGlobe src="hello_world.jpg" />
  <View style={{position: 'absolute', transform: [{position: [0, 0, -3]}]}}>
    <Image style={{width: 1, height: 1}} src="react.jpg" />
    <Text fontSize={0.1}>
      Hello from React VR!
    </Text>
  </View>
</View>
```

React
(Javascript Library from Facebook)
Vizor

http://vizor.io/

A platform for exploring, creating, and publishing VR on the web.

React

(JS Library from Facebook)
ThreeJS Editor

http://threejs.org/editor

WYSIWIG editor for ThreeJS development

Ricardo Cabello (mrdoob)
Describe scenes in natural language to create VR stories

My first scene will last 500 seconds and display an image located at https://s3.amazonaws.com/gurivr/logo.png along with a text saying: "Guri is cool!" to my left and a panorama located at https://s3.amazonaws.com/gurivr/pano.jpg

Mi segunda escena dura 5 segundos, has a skyblue background y muestra un texto que dice "La ciencia ha eliminado las distancias."

Dan Zajdband
SocialVR

http://artfab.art.cmu.edu/projects/socialvr/
http://cmuartfab.github.io/social-vr/

Browser-based interactive design editor for personalized VR experiences

CMU Art Fab
Thanks!