HackMe 2.0 - XperienZ

HKME-17 - Project setup Guidelines

Version 1.0

Published on 12/02/2016

**Document Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 12/02/2016 | 1.0 | First Draft | Saurabh Nutan Sohoni |

**References**

|  |  |
| --- | --- |
| **No.** | **Artefact Name** |
|  | <https://developers.google.com/web/updates/2015/10/media-devices> |
|  | <https://mobiforge.com/design-development/html5-mobile-web-device-orientation-events> |
|  | <https://github.com/ajfisher/deviceapi-normaliser> |
|  | <https://www.html5rocks.com/en/tutorials/webrtc/basics/> |
|  | <https://bitbucket.org/webrtc/codelab> |
|  | <https://developers.google.com/web/updates/2015/07/interact-with-ble-devices-on-the-web> |

Contents

[1. Introduction 4](#_Toc468461699)

[1.1 Purpose 4](#_Toc468461700)

[1.2 Scope 4](#_Toc468461701)

[1.3 Hardware requirements 4](#_Toc468461702)

[1.4 Software requirements 4](#_Toc468461703)

[2. PROJECT SETUP AND API USAGE GUIDELINES 5](#_Toc468461704)

[2.1 Control of Compass, Accelerometer and Gyroscope using HTML5 Device Orientation API 5](#_Toc468461705)

[2.2 Normalization of raw orientation values using Device API normalizer 5](#_Toc468461707)

[2.3 Audio/Video recording and playback using MediaStream API 6](#_Toc468461708)

[2.4 Video and Data streaming using WebRTC 6](#_Toc468461709)

[2.5 Conversion of 2D video stream to stereoscopic 3D using Three.js 7](#_Toc468461710)

[2.6 Accessing smartphone Bluetooth module through Web app 7](#_Toc468461711)

[2.7 Binding all pieces together 8](#_Toc468461712)

# Introduction

# Purpose

The purpose of this document is to guide developers in setting up a project using Web APIs to control sensors such as Compass, Accelerometer, Gyroscope, Camera, Speaker, Microphone and Bluetooth in-built in a smartphone. The project aims at controlling an integrated system such as a drone via Virtual Reality (VR) headset and provide first person view to the user as a part of feedback. Human Computer Interaction (HCI) is achieved when users perform specific gestures in order interact with the drone and change its position and orientation.

# Scope

HCI system mentioned in the introduction can be realized with the help of graphic shown below:



----WebRTC----

Drone (Autonomous Car) Communication First person view through VR headset

Channel

# Hardware requirements

Since this project aims at extending the capabilities of a smartphone, the major hardware required will be a smartphone with working sensors and some microcontrollers and headsets to simulate the experience. Detailed requirement is given below:

1. A Smartphone preferably running on Android or iOS with sensors viz. compass, accelerometer, gyroscope, camera, speaker, microphone and Bluetooth in working condition. Current state of sensor can be easily tested using hardware diagnostic tools. For e.g.: Motorola Device Help Android App.
2. VR headset. Any VR headset with 3D viewing capabilities is sufficient. This project uses Google Cardboard VR headset
3. Microcontroller to control the motion of drone. This project uses Arduino
4. Drone chassis, stepper motor, wheels, etc. for building and controlling drone. This project uses car similar to Lego cars

# Software requirements

This project aims at creating a simple web app. Hence the software requirements are minimal and are given below:

1. Web Browser: Google Chrome (version >=7.0), Firefox (version >= 6.0)
2. Mobile Browser: Chrome for android (version >= 3.0), Firefox (version >= 6.0), Safari Mobile (version >= 4.2)
3. Text editor software such as Notepad++, Sublime Text, etc.

# PROJECT SETUP AND API USAGE GUIDELINES

This section provides a step by step guide to setup a web application for accessing sensors and providing controller and feedback based on the gesture command. Results will be displayed in a web browser. Every section given below is roughly divided into two areas viz. brief about the API and setup and usage guidelines (marked by star shaped graphic).

# Control of Compass, Accelerometer and Gyroscope using HTML5 Device Orientation API

# HTML5 Device Orientation API provides information about the orientation and movement of a device. Information comes from the positional sensors such as compasses, gyroscopes and accelerometers. Via this API, a web app can access and make use of information about how a device is physically oriented in space.

The HTML5 Device Orientation API specifies three events, which are outlined below:

1. deviceorientation – Fired when significant orientation occurs
2. compassneedscalibration – Fired when compass needs calibration and calibration will provide more accurate data
3. devicemotion – Fired regularly with information regarding the motion of device

This API returns raw values of different angles viz. alpha, beta and gamma defined below and illustrated in the figure:

1. Alpha - The direction, the device is facing with rotation allowed around Z axis.
2. Beta – The direction, the device is facing when it is tilted from front to back
3. Gamma – The direction, the device is facing when the device is tilted left to right



This API is available out of the box with HTML5 DocType specification while creating html webpage.

# Normalization of raw orientation values using Device API normalizer

Device API Normalizer is a library that can be used to normalize the data coming out of the HTML5 device orientation API code across browsers. Most devices now support at least part of the spec with motion being the most weakly supported in terms of the values being exposed but orientation being most wacky due to varying implementations. This library creates two helper functions designed to parse device orientation and device motion events. It returns objects that you can query as per the spec at the W3C with as much consistency as possible provided.

More documentation on Device API library is available at following GitHub repository:

<https://github.com/ajfisher/deviceapi-normaliser>

To make use of this library, user has to include deviceapi-normaliser.js in html file using script tag.

# Audio/Video recording and playback using MediaStream API

For creating a first person view experience to the user wearing VR headset, we will be recording a video by using in built camera module from a smartphone. This smartphone will be docked on the drone chassis. The resolution of video recording can be controlled based on the available hardware.

mediaDevices is available on navigator object present on global window object in modern browsers (listed in the software requirements section above). It has following methods which will be used in the context of this project:

1. enumerateDevices - Returns an array of available media devices including camera, speaker, microphone, etc. with primary device selected by default
2. getUserMedia – Takes a configuration object which specifies audio/video quality and resolution parameters and returns live stream of data

Stream of data returned by getUserMedia can be as a source to html <video> tag and playback is handled via in-built speakers and display on a smartphone.

As this API is available through JavaScript on global window object, users need not to install any plugin. The only requirement is the web app must be hosted on a secure server (HTTPS) or on localhost.

# Video and Data streaming using WebRTC

WebRTC stands for Web Real Time Communication. This API is aimed at facilitating plug-in free video and data communication between client machines (most of the time without the interference of server using STUN protocol).

WebRTC makes use of MediaStream API and implements following:

1. Media capture through MediaStream API
2. RTCPeerConnection API for real time media communication between clients with facilities of data encryption and bandwidth management
3. RTCDataChannel API for real time generic data communication between clients

In this project, live video captured from drone will be streamed to smartphone with user. In this way user can experience remote events by sitting on a couch and not going anywhere else!

For establishing real time channel using RTC, peers first need to handshake by using signaling protocol. This is usually established by using XHR or web socket based communication. Once connection is established, 86% communication is directly between peers using STUN servers. While the remaining 14% is achieved by using TURN servers which fallback to conventional routing methods.

All guidelines are mentioned at the following webrtc codelab. This includes establishing basic connection, use of socket.io for handshake and implementing live stream through STUN and TURN servers via public network.

<https://bitbucket.org/webrtc/codelab>

# Conversion of 2D video stream to stereoscopic 3D using Three.js

Three.js makes use of WebGL render engine to create stereoscopic images (and videos if generate enough frames per second) for creating an experience optimized for VR. For every animation created using Three.js, following basic components need to be created:

1. A Scene – This is like a canvas
2. A renderer – This is analogous to a painter
3. A Camera – These are all viewers observing the painting

Apart from just conversion, one can add various graphic elements such as cubes, spheres, etc. for enhancing user experience by adding more objects to interact.

For using Three.js, following JavaScript files needs to be imported into HTML of web app:

1. three.js
2. StereoEffect.js
3. DeviceOrientationControls.js
4. OrbitControls.js

# Accessing smartphone Bluetooth module through Web app

Till now, the web app has recognized all of the gestures performed by the user for controlling the drone. Now, in order to move the drone, the web app needs to interact with hardware via Arduino microcontroller. This involves two steps viz. communication of instructions from web app to Arduino and communication from Arduino to motors installed on drone. This section gives more information on communication from web app to Arduino.

Very similar to MediaStream API, Bluetooth API is available for use on modern browsers (listed in the software requirements section above). With the help of requestDevice method on window.navigator.bluetooth object, user can access available Bluetooth devices and communication can be achieved. In this project, communication will take place from Bluetooth module present on a smartphone to Bluetooth Low Energy (BLE) module present on Arduino.

All guidelines for establishing a connection, data transfer and disconnecting from device can be found at: <https://developers.google.com/web/updates/2015/07/interact-with-ble-devices-on-the-web> . Again, to access the Bluetooth module on a smartphone, a web app must be hosted on secure (HTTPS) server or on localhost.

# Binding all pieces together

There you go, till now, this project has every other module in place for realizing the concept of target application. Import all the code into single index.html file and request it through controller smartphone at user’s end and recorder smartphone present on the drone. To facilitate debugging, let us summarize what all we have created:

* Live video and audio capture through camera module on smartphone installed on drone
* Live video and audio streaming through webRTC channel from drone to user
* Stereoscopic display of video capture inside Google cardboard VR headset using Three.js
* Gesture recognition through head movements of user using accelerometer, compass and gyroscope
* Controlling motion of drone w.r.t. user command via Bluetooth communication to Arduino