CS CAPSTONE REQUIREMENTS DOCUMENT

APRIL 17, 2019

CREATING IMMERSIVE EXPERIENCES ON THE WEB USING VR AND AR

PREPARED FOR

INTEL

ALEXIS MENARD _____ _____
Signature *Date*

PREPARED BY

GROUP 47- WEBPHYSICSVR

JONATHAN JONES _____ _____
Signature *Date*

EVAN BRASS _____ _____
Signature *Date*

BROOKS MIKKELSEN _____ _____
Signature *Date*

TIM FORSYTH _____ _____
Signature *Date*

BRANDON MEI _____ _____
Signature *Date*

Abstract

This document is a Software Requirements Specification (SRS) that outlines the purpose, scope, and technical requirements of the WebPhysicsVR immersive web experience hosted by the Intel 01.org open source portal. This document describes the website functionality and provides a tentative timetable for the fulfillment of project goals and requirements.

CONTENTS

Section	Original	New

1.3.3	• WebXR API
	• WebGL 2.0
	• (Some physics API)
	• If needed:
	  – Web Workers
	  – Web Assembly (for physics / processing intensive code or porting non-JavaScript libraries)
		• Add mention of Cannon.js, Three.js
		• Remove use of web workers and web assembly

1.4	• XR : Family of hardware devices capable of Virtual Reality and Augmented Reality. • API : Application Program Interface. • HMD : Head mounted display. There is an optic lens for each eye which runs at 90Hz. • WebXR : Open source web API for accessing virtual reality (VR) and augmented reality (AR) devices, including sensors and head-mounted displays. • WebGL : Open graphics API. • Web Worker : API for performing and communicating between parallel computation on the Web.	• Remove reference to web workers, since we did not use them.

2.1	• Users will be able to interact with objects in the environment through their controller (or simulated/virtual controller) – Translate objects – Scale objects – Rotate objects – Pick up/grasp objects – Drop/throw objects	• Removed the requirement to scale objects. • Giving users the ability to scale objects worked against a good user experience as scaling could go wrong in many ways. • User-scaled objects often got in the way of UI elements and in some cases removed the ability to scale objects down entirely. • Without a second button on the daydream controller, scaling had to be done using in scene UI elements.
2.1	• User can ”pause/play”. – During pause mode, objects are not animated and are not affected by the physics engine. – During play mode, objects are animated and are affected by physics. • User can ”freeze” objects, stopping all kinematic forces that the object is experiencing	• As with the scaling issue above, we were severely limited by the single button functionality that the controllers could provide. • Without a dedicated GUI for this sort of project, we decided that it was more important to place our focus on polishing the VR components rather than struggling with out-of-scope additions.
2.5	• Website will be compatible with (Enter list of devices here)	• Added a list of compatible devices.
2.5	• Website will run on multiple devices and operating systems (Windows, Android, Mac)	• WebXR is not compatible with iOS devices.

1 INTRODUCTION

1.1 Purpose

The purpose of this project is to provide a technical demonstration of the WebXR API capabilities. It will be an example of how WebXR can be used to create an entertaining, interactive VR experience within a web browser. It will have a focus on education and act as a virtual physics lab.

1.2 Scope

WebPhysicsVR will be an interactive physics simulation that utilizes the WebXR API. Its primary use is as tech demo and a platform for education.

As a tech demo, our project will incorporate all the major features of WebXR. We'll want to faithfully present the physical concepts in an educational and engaging fashion while also focusing on designing a quality WebXR experience.

At a minimum, users should be able to visualize the effects of gravity. If we have time to do so, we could visualize other forces like adding friction between objects or giving certain objects magnetism as well as mass.

We intend to support a spectrum of VR devices though we will only test on a few selected devices that we have access to.

1.3 Product overview

1.3.1 User Interfaces

Users with compatible hardware will be able to experience an immersive/enhanced version of the website right when they access it, with a click of a button.

Within the VR experience, we will have interfaces for the user to manipulate the objects and physical constants. These interfaces will be displayed in the most contextually meaningful location in the environment. This might mean different floating icons around the 3d objects, floating interfaces or control objects that are within the environment.

1.3.2 Hardware Interfaces

WebXR is compatible with a range of XR devices, web browsers and operating systems. This includes mobile devices. For the full intended experience, an HMD with body and controller tracking devices should be used.

1.3.3 Software Interfaces

- WebXR API
- WebGL 2.0 (via Three.js)
- Cannon.js

1.3.4 Functions

The main functions of this website are:

- VR Experience accessible without need to download client side software
- Interactive physics environment

1.4 Definitions

- XR : Family of hardware devices capable of Virtual Reality and Augmented Reality.
- API : Application Program Interface.
- HMD : Head mounted display. There is an optic lens for each eye which runs at 90Hz.
- WebXR : Open source web API for accessing virtual reality (VR) and augmented reality (AR) devices, including sensors and head-mounted displays.
- WebGL : Open graphics API.

2 SPECIFIC REQUIREMENTS

2.1 External interfaces

- Users may interface with the website using:
  - Mobile devices with positional tracking
  - Head mounted displays, whether they are opaque, transparent, or utilize video pass-through
  - Fixed displays with head tracking capabilities
- Users will be able to interact with objects in the environment through their controller (or simulated/virtual controller)
  - Translate objects
  - Rotate objects
  - Pick up/grasp objects
  - Drop/throw objects
- Users must be able to change physical constants. Potential mechanisms for doing that:
  - Physical Constants Dashboard with levers/switches/dials to change the environment
  - Floating user interfaces around the objects with icons/bars and other traditional UI features
- User can spawn in a multitude of objects stored on the website through a spawn menu

2.2 Functions

- Utilize all core features of WebXR including:
  - Headset location and orientation

  - Controller location and orientation
  - Multi-platform compatibility
  - Parsing input from XR devices

- The website will load the correct VR environment depending on the hardware that the user has.
- Website will pause when a controller has been disconnected and resume when the connection is reestablished.
- Website will notify user if web browser/hardware is not compatible with WebXR
- Render environment and physics in real time

2.3 Usability requirements

Offer a way of manipulating objects in the simulation for all supported platforms. On a phone this might mean having a virtual manipulator at a fixed location from the user's face. On devices like the Oculus Rift, this would probably mean the hand controllers.

We want to pair a non-technical-user friendly physics demonstration with a technical-user friendly code base.

2.4 Performance requirements

- Latency must not exceed 11 milliseconds within the simulation.
- Re-factor to distribute to multiple web workers if needed to achieve performance requirements.
- Meet an average 60 frames per second, appropriately degrading the content to meet that budget on mobile phones and other low powered devices.
- HMDs should be running at 90 fps

2.5 Software system attributes

- Landing page will be compatible with all evergreen browsers
- Website will be compatible with browsers that support WebXR
- Website will be compatible with

  - Magic Window compatible mobile devices using compatible browsers.
  - Google Daydream
  - HTC Vive

- Website will run on multiple devices and operating systems (Windows, Android, Mac)

3 GANTT CHART