GrodnoVR:

My first and last experience with ReactVR

Artur Basak

GROCON

GRODNO 2018

Few words about me

Artur Basak

*instinctools EE Labs, Lead UI Developer

Developing, implementing and testing user interfaces more than 8 years

/artur.basak

/arturbasak

/@arturbasak

GrodnoVR demo app - https://grodno-vr.github.io

Terms

VR-headset - Head-mounted Device

- Oculus Rift
- Oculus Go
- HTC Vive
- Google Daydream
- Google Cardboard
- Samsung Gear VR
- PlayStation VR

Frame from film «Jhonny Mnemonik»

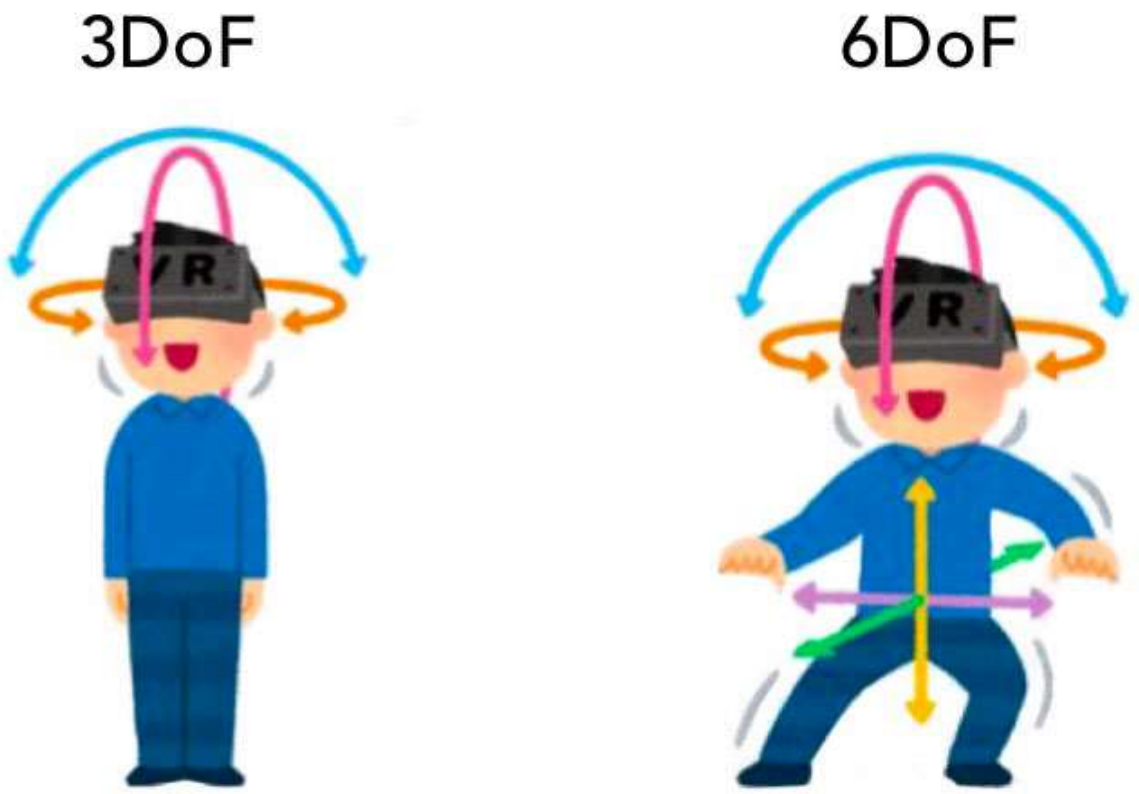
Frame from film «Jhonny Mnemonik»

VR vs AR vs AV vs MR/XR

- VR - Virtual Reality
- AR - Augmented Reality
- AV - Augmented Virtuality
- MR / XR - Mixed / Extended Reality

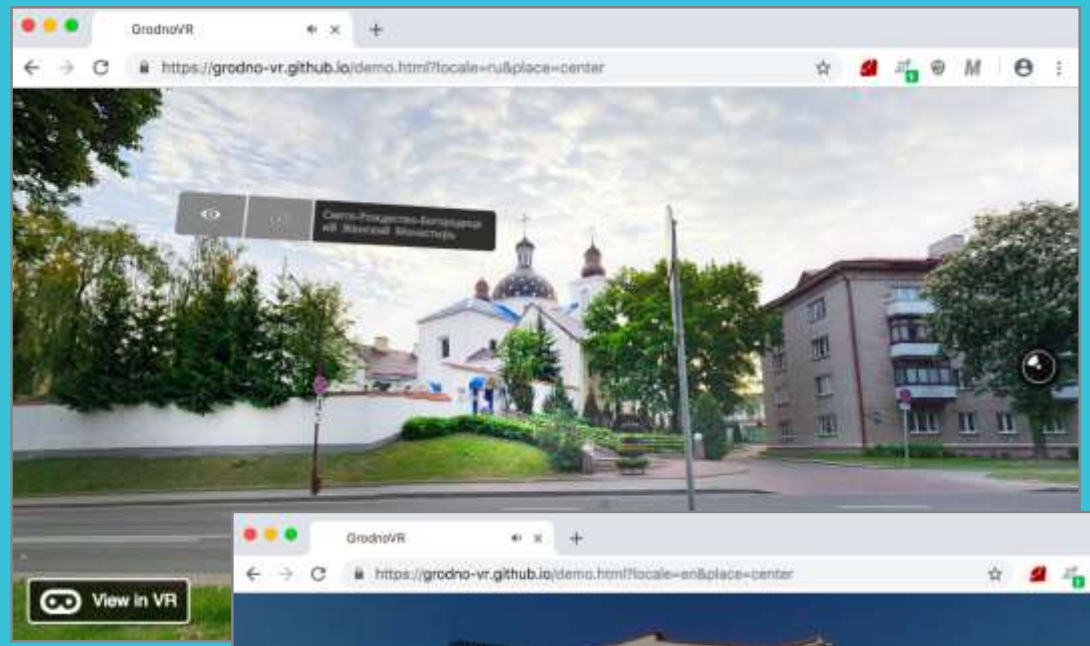
Picture from uxplanet.org

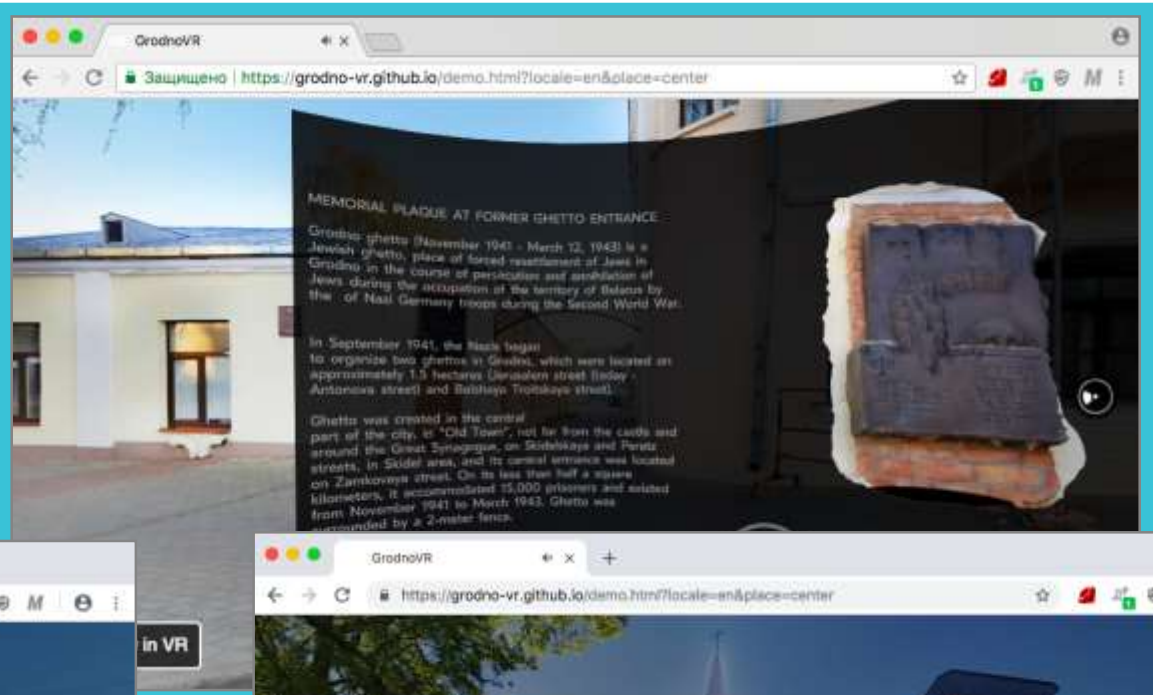
Degrees of Freedom

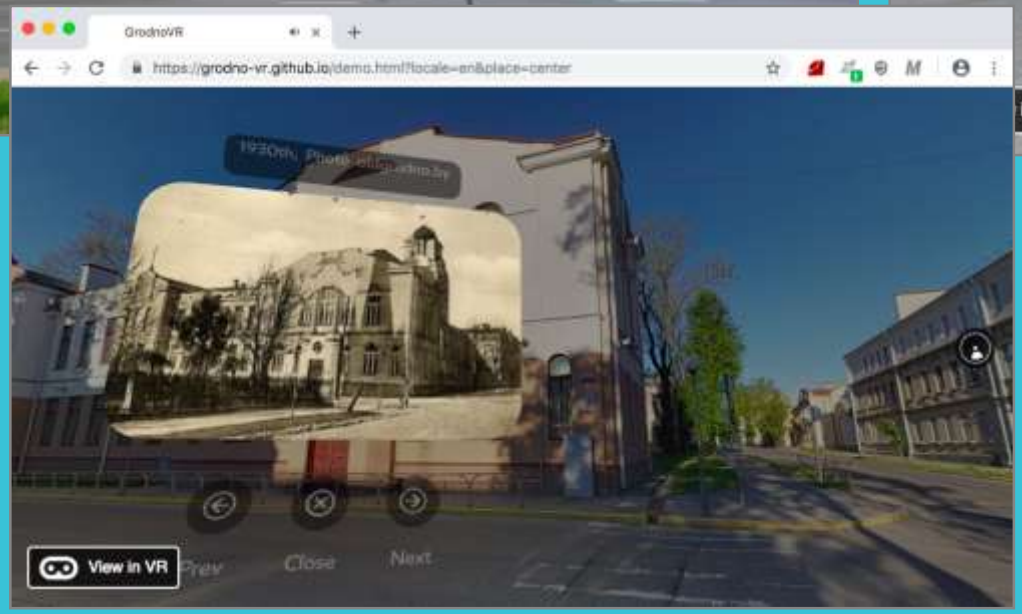


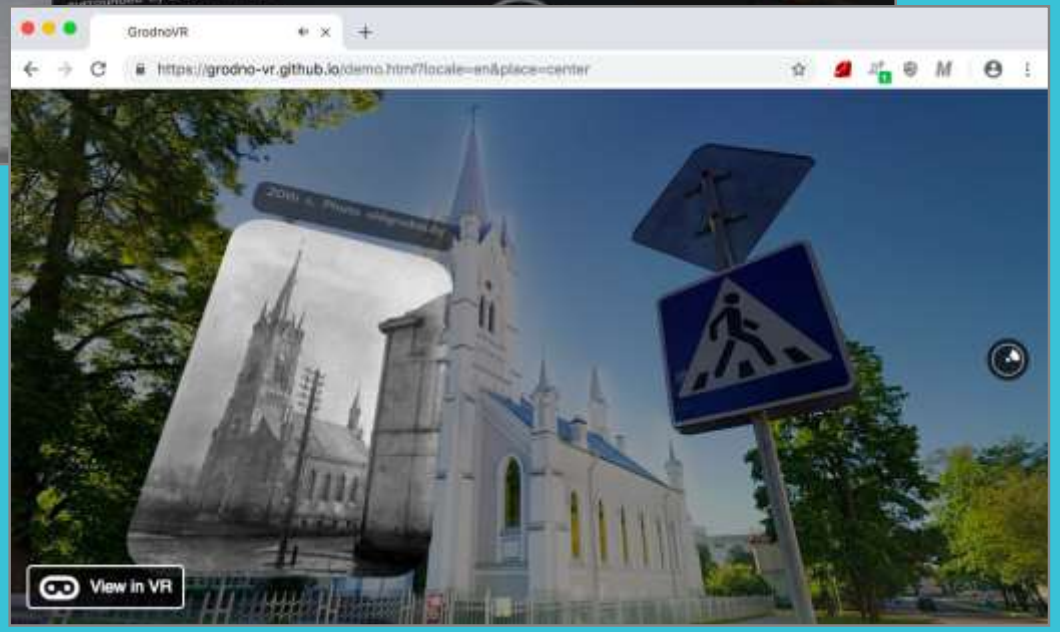
Picture from ichip.ru, source: Unity

GrodnoVR demo app - https://grodno-vr.github.io





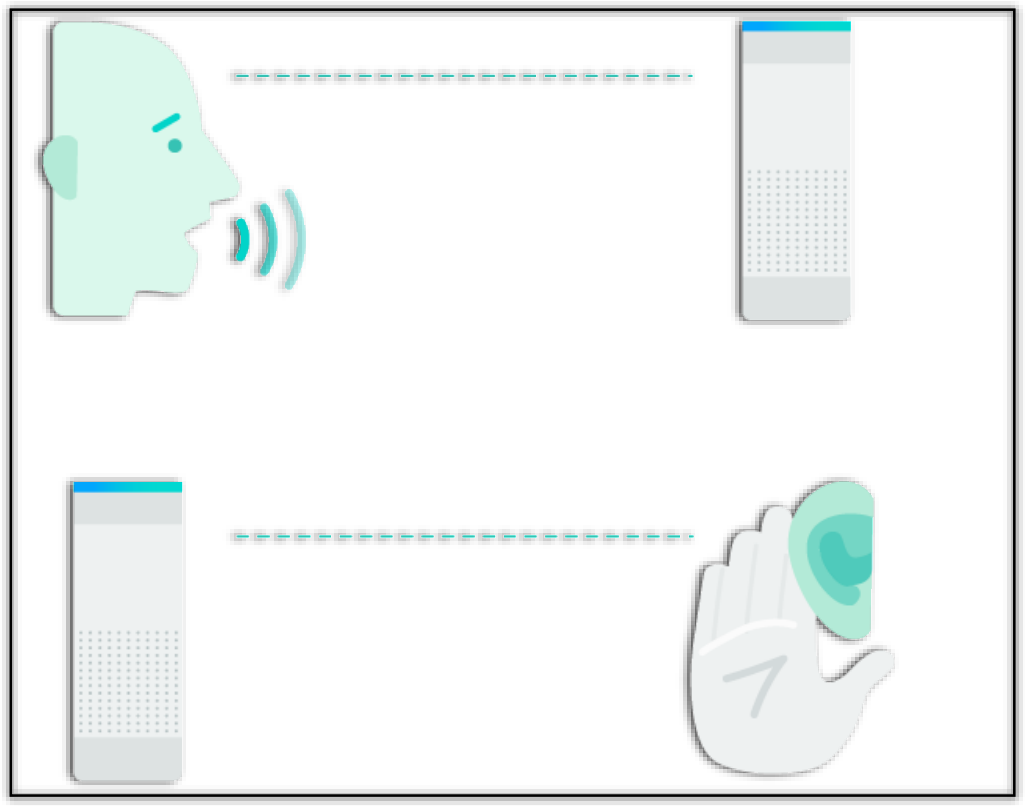




Idea & Motivation

The future of UI - VR & VUI

Picture from mixedreality.mozilla.org



Picture from developer.amazon.com/alexa-skills-kit/vui

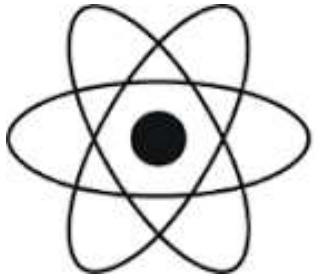
Why ReactVR?

https://immersive-web.github.io/webvr/spec/1.1

Platform compatibility:
https://webvr.rocks
https://webvr.info

Why ReactVR?

React Native

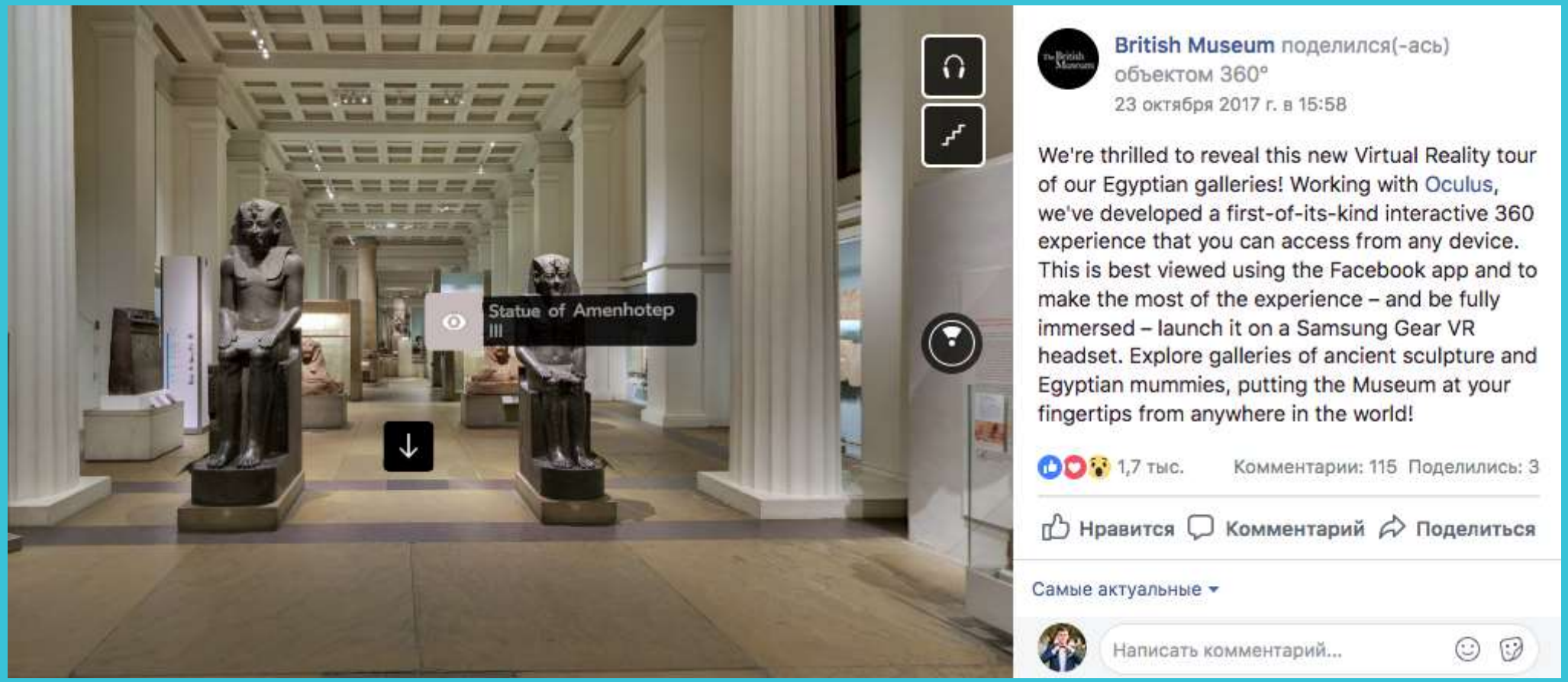


React VR

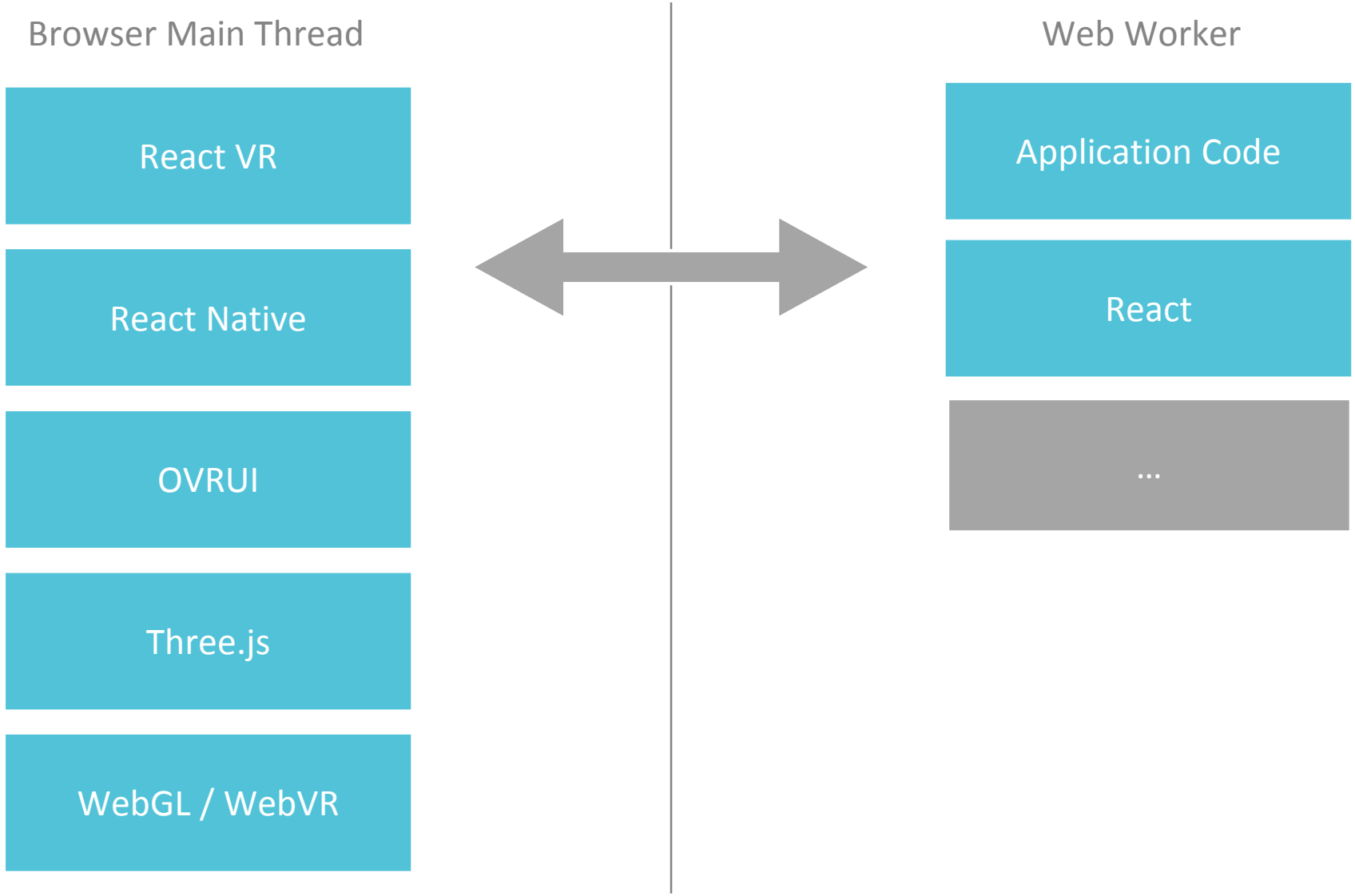
oculus

Idea & Motivation

THE BRITISH MUSEUM (BY OCULUS) - [HTTP://VR.BRITISHMUSEUM.ORG](http://vr.britishmuseum.org)



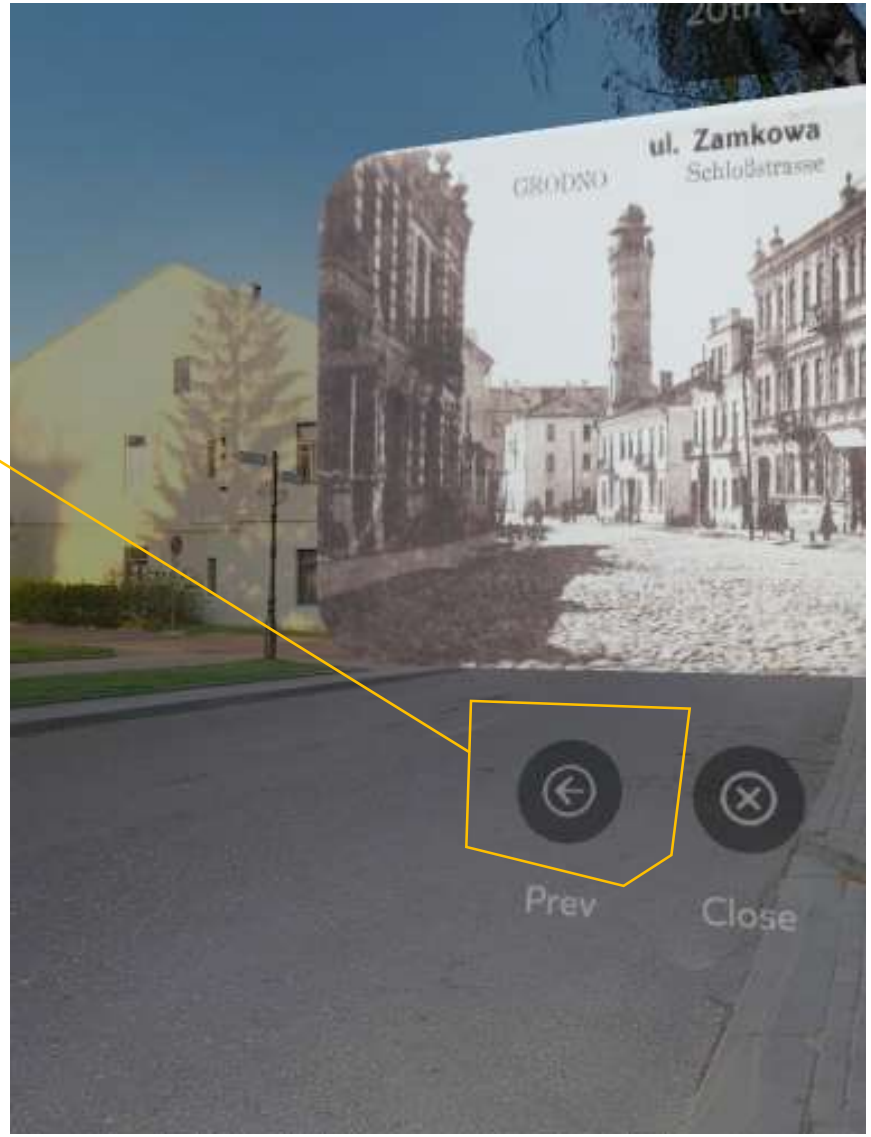
React VR is Built on React Native



React Components

Arrow.js

```
class Arrow extends React.Component {

    constructor(props) {
        super(props);
        this.state = {...};
    }

    mouseIn() {...}

    mouseOut() {...}

    render() {
        const { onClick, direction, text } = this.props;
        const { opacity, scale } = this.state;

        return (
            <View style={styles.centerView}>
                <GazeButton
                    onClick={() => onClick()}
                    onEnter={() => this.mouseIn()}
                    onExit={() => this.mouseOut()}
                    onEnterSound={asset('audio/hover-1.wav')}
                    style={[...]}
                >
                    <Image/>
                </GazeButton>
                <Text style={styles.tooltipButton}>
                    {text}
                </Text>
            </View>
        );
    }
}

export default Arrow;
```

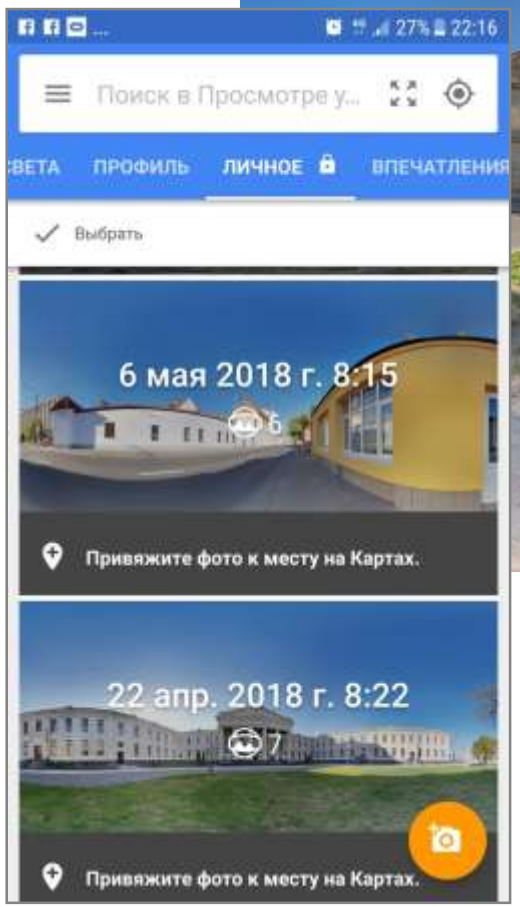


New Tools

Tools

360-DEGREE PHOTO

- Google Street View
- Photoshop



Place Component

index.js

```
import React from 'react'
import { asset, Pano, Sound, Animated } from 'react-vr';

import { Portal, Label, Gallery, VRInformation } from '../.';
import { lightMixin, cameraMixin, loadingMixin } from './mixins';

import styles from './styles';

const SuperClass = loadingMixin(cameraMixin(lightMixin(React.Component)));

const AnimatedPano = Animated.createAnimatedComponent(Pano);

class Place extends SuperClass {

    constructor(props) {
        super(props);
        this.state = {...};
    }

    componentDidMount() {
        this.startLoading();
    }
```

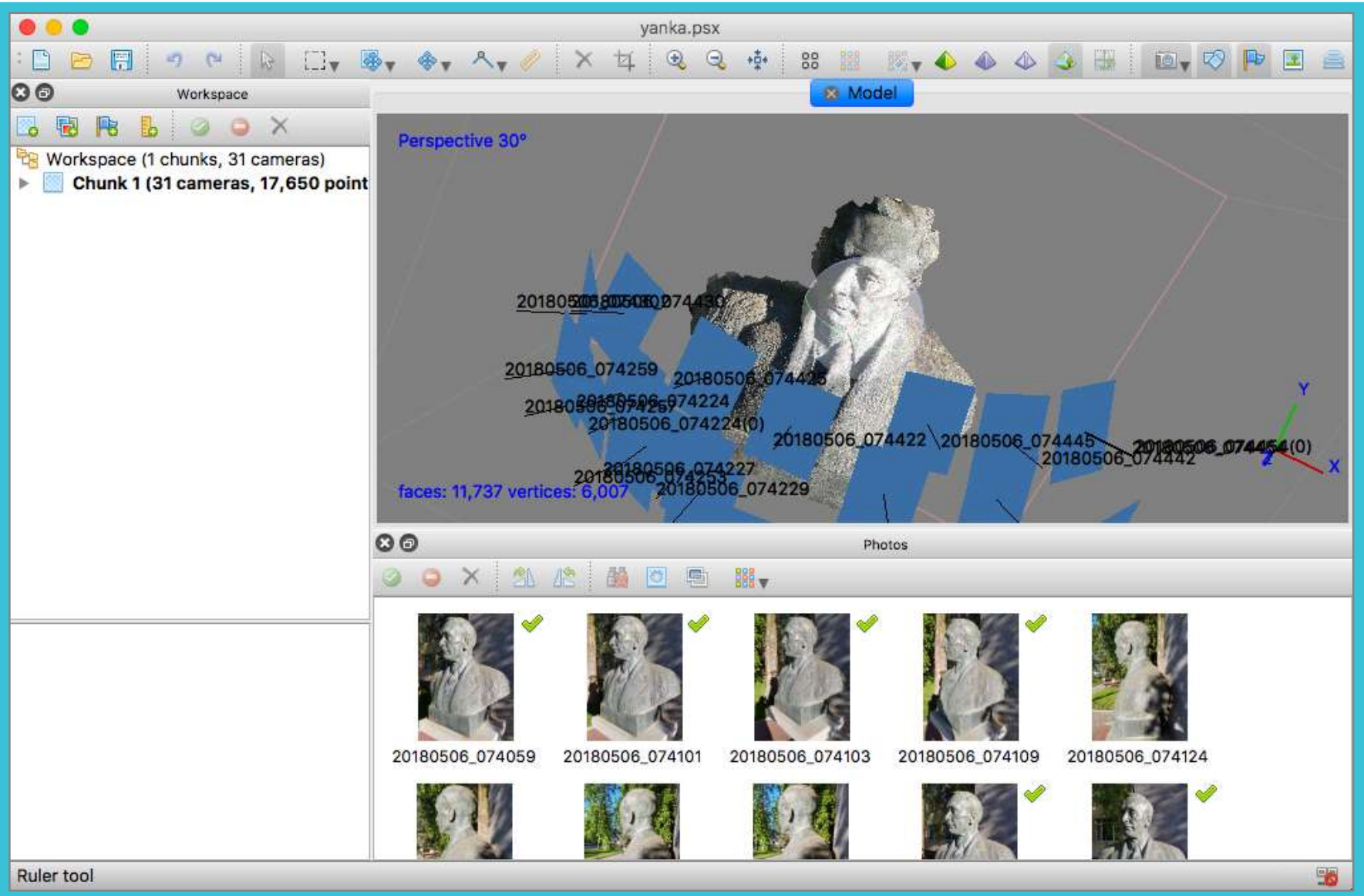
index.js

```
    render() {
        const { place = {}, style = {} } = this.props;
        const { loading, showOldImages, showInfo, light } = this.state;

        return (
            <Animated.View style={[...]}>
                { place.sound && <Sound loop={true} source={asset(place.sound)} /> }

                { this.renderLoading() }

                <AnimatedPano
                    onLoad={() => this.onLight(() => this.stopLoading())}
                    source={asset(`/places/${place.name}/background.jpg`)}
                    stereo={'TOP_BOTTOM_3D'}
                    style={{
                        ...style,
                        position:'absolute',
                        opacity: light,
                        tintColor: loading ? 'grey' : 'white'
                    }}
                />
```

Tools

PHOTOGRAMMETRY

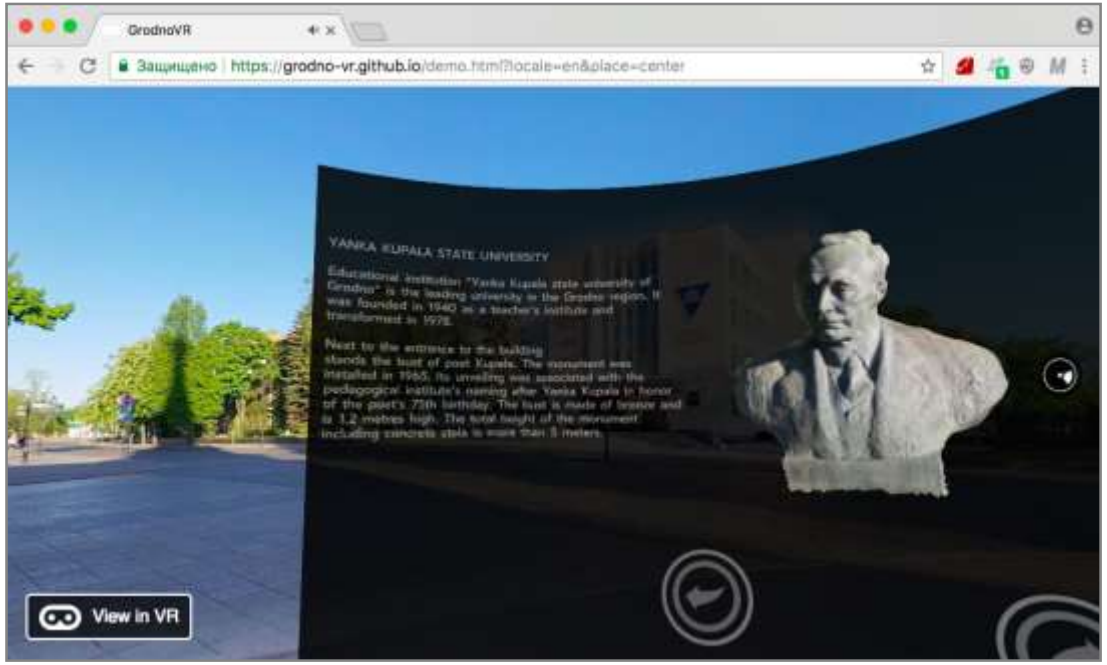
- Agisoft PhotoScan Pro (MacOS)
- Autodesk ReCap Photo (Win)



Model Component

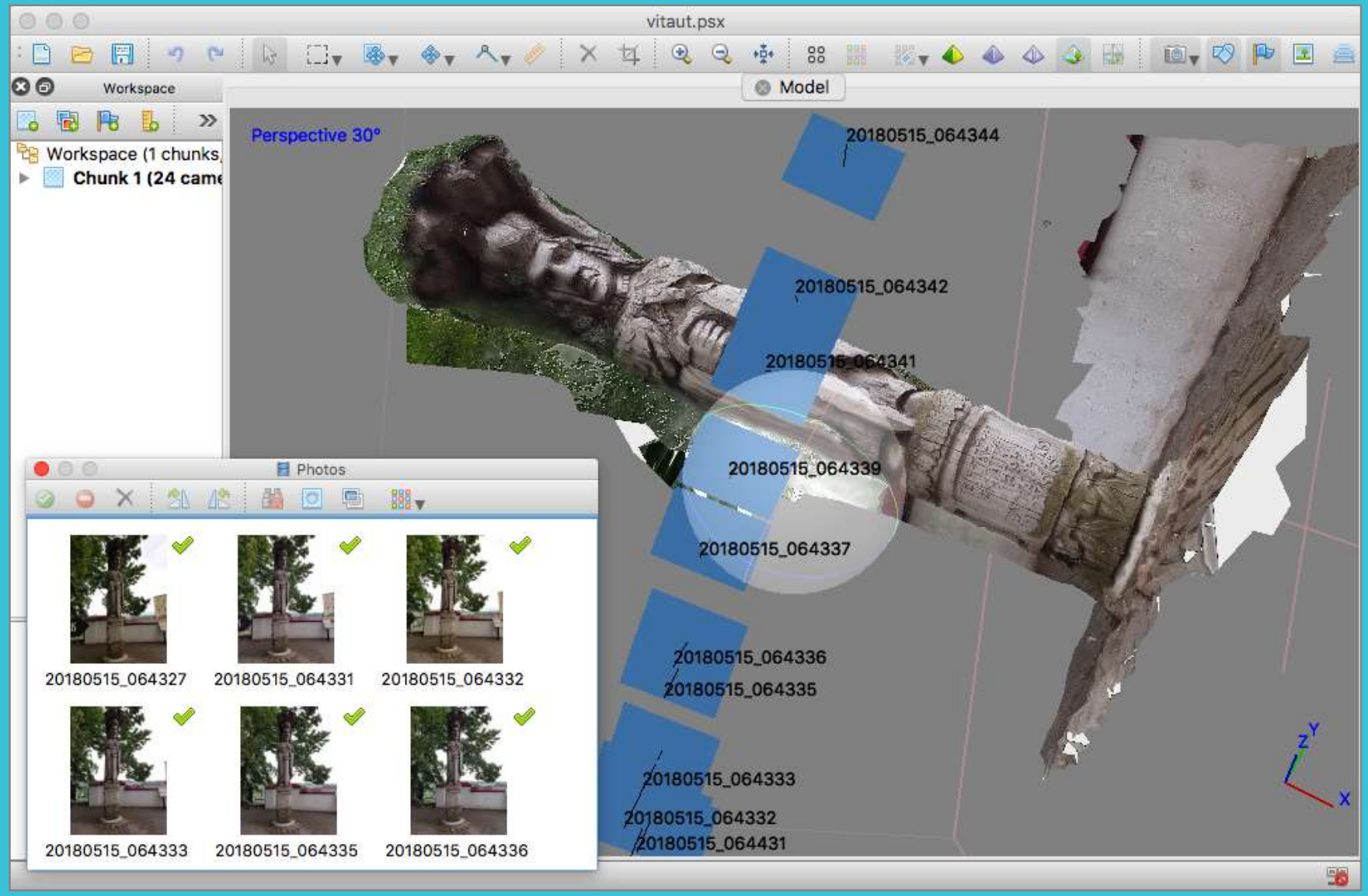
index.js

```
import React from 'react';
import { asset, Model, Animated } from 'react-vr';

const AnimatedModel = Animated.createAnimatedComponent(Model);
const Easing = require('Easing');

class Model3D extends React.Component {

    constructor(props) {
        super(props);
        this.state = { rotation: new Animated.Value(50), scale: new Animated.Value(1) };
    }

    componentWillReceiveProps(nextProps) {...}

    startModelScaling() {...}

    stopModelScaling() {...}

    render() {
        const { disabled, style, obj, mtl, rotationAxis } = this.props.details || {};
        const transform = [...style.transform, { scale: this.state.scale } ];

        if (rotationAxis) {...}

        return (
            <AnimatedModel
                onEnter={() => !disabled && this.startModelScaling()}
                onExit={() => !disabled && this.stopModelScaling()}
                style={{transform: transform...}}
                lit={true}
                source={{
                    obj: asset(obj),
                    mtl: asset(mtl)
                }}
            />
        );
    }
```



Tools

PHOTOGRAMMETRY FAILS

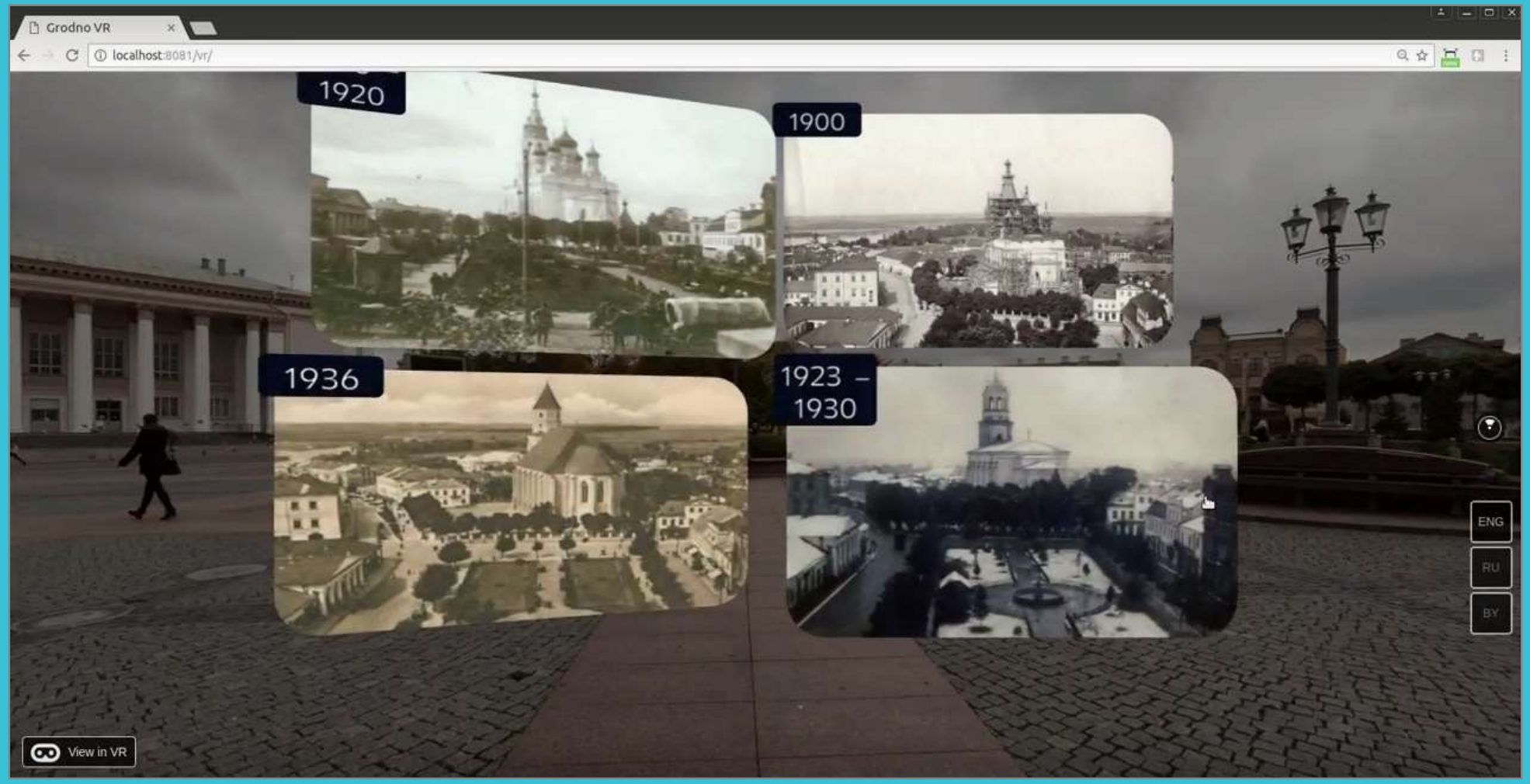


VR UI: Controls

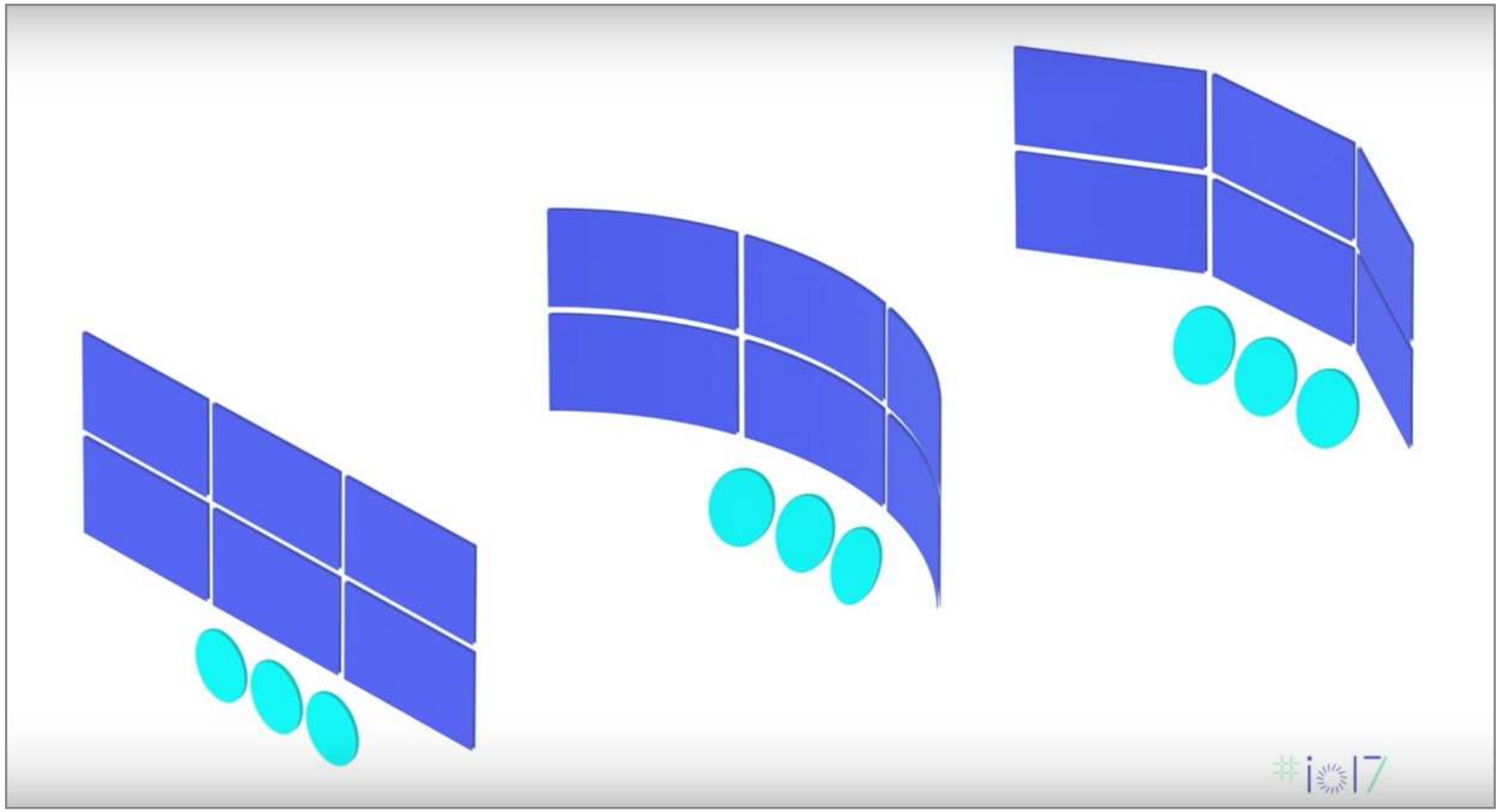
GROCON

GRODNO 2018

VR UI: Panels


Grodno VR
localhost:8081/vr/
1920
1900
1936
1923 – 1930
ENG
RU
BY
View in VR

VR UI: Cylindrical Panels

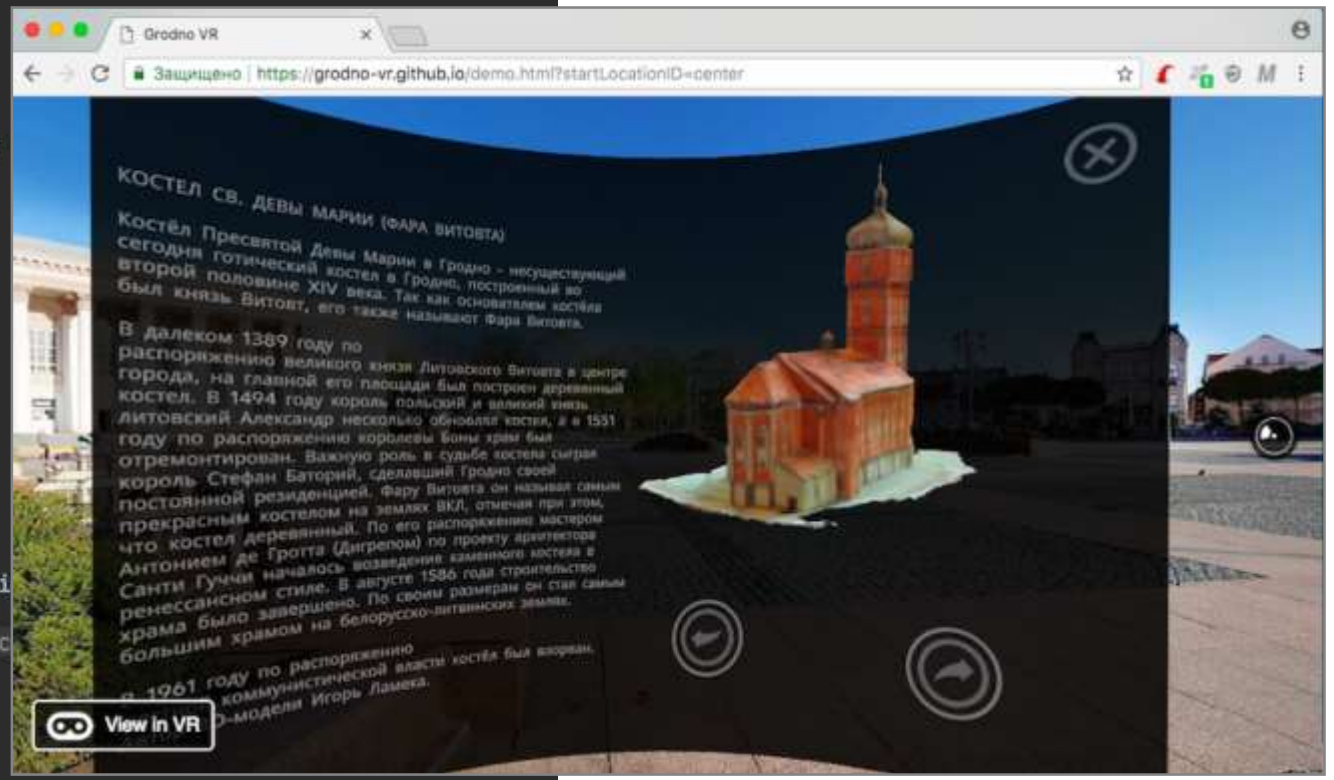


Frame from «Designing Screen Interfaces for VR (Google I/O ’17)» Chris McKenzie

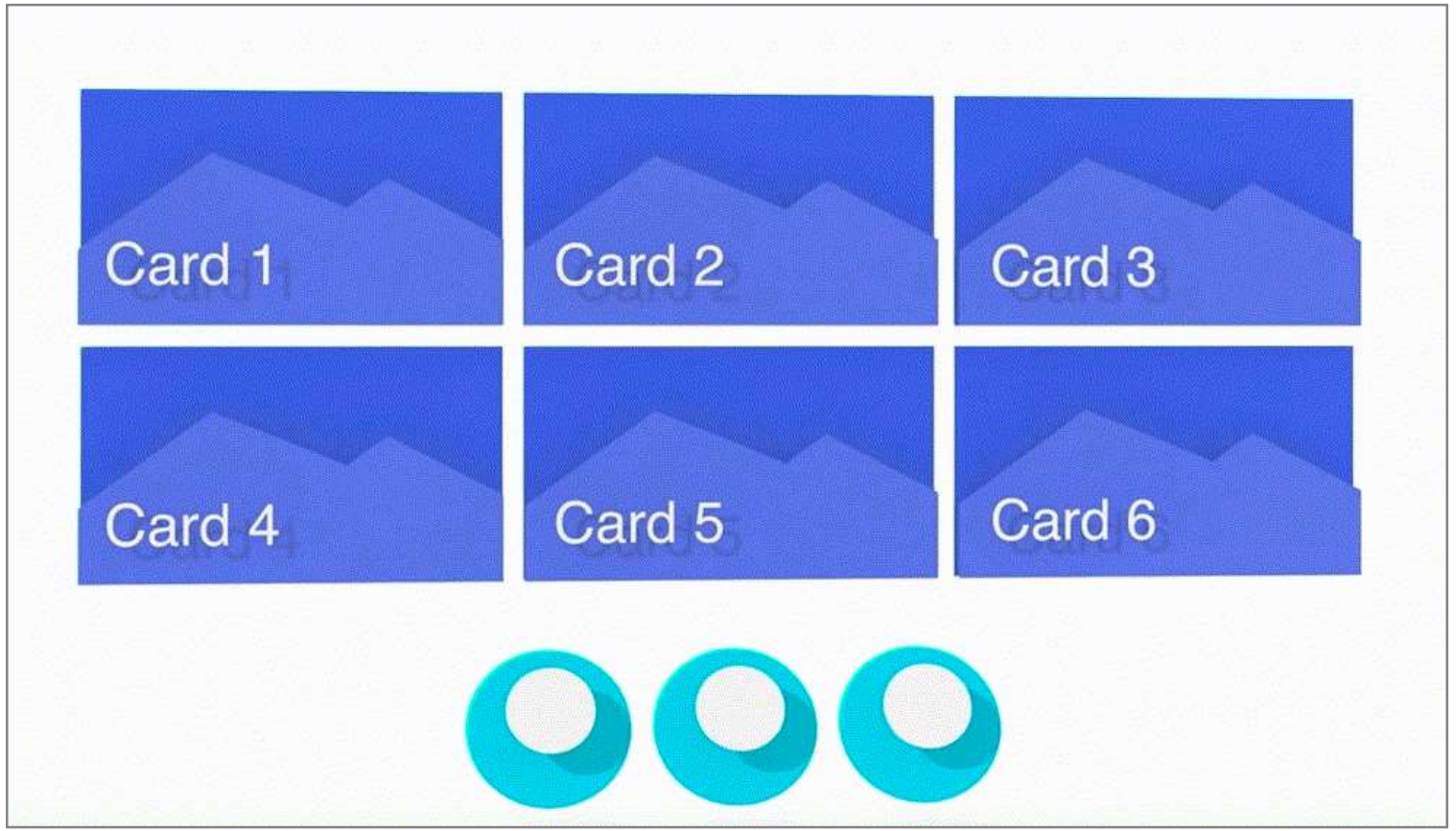
VR UI: Cylindrical Panels

VRInformation.js

```
render() {
    const { title, description, model, translateX, width, height, onClose } = this.props;
    const { opacity, scale, closeOpacity, closeScale } = this.state;

    return (
        <AmbientLight intensity={1}>
            <CylindricalPanel
                layer={{ width: 4096, height: 820, radius: 700 }}
                style={{ transform: [ translateY: 5 }], position:
            >
                <Animated.View
                    style={{opacity: opacity...}}
                >
                    <View/>
                    <View style={{flexDirection: 'row'...}}>
                        <Text
                            style={{
                                margin: 20,
                                fontSize: 24,
                                fontWeight: '300',
                                width: model ? 600 : 900
                            }}
                        >
                            {`${title.toUpperCase()}\n\n${descripti
                        </Text>
                        { model && <View style={{justifyContent: 'c
                        </View> }
                    </View>
                </Animated.View>
            </CylindricalPanel>

            { model && <Model3D rotation={this.state.modelRotation} details={model} /> }
            { model && this.renderModelControls() }

        </AmbientLight>
    );
}
```



VR UI: Swiveling Panels



Frame from «Designing Screen Interfaces for VR (Google I/O '17)» Chris McKenzie

VR UI: Swiveling Panels

VR UI: Swiveling Panels

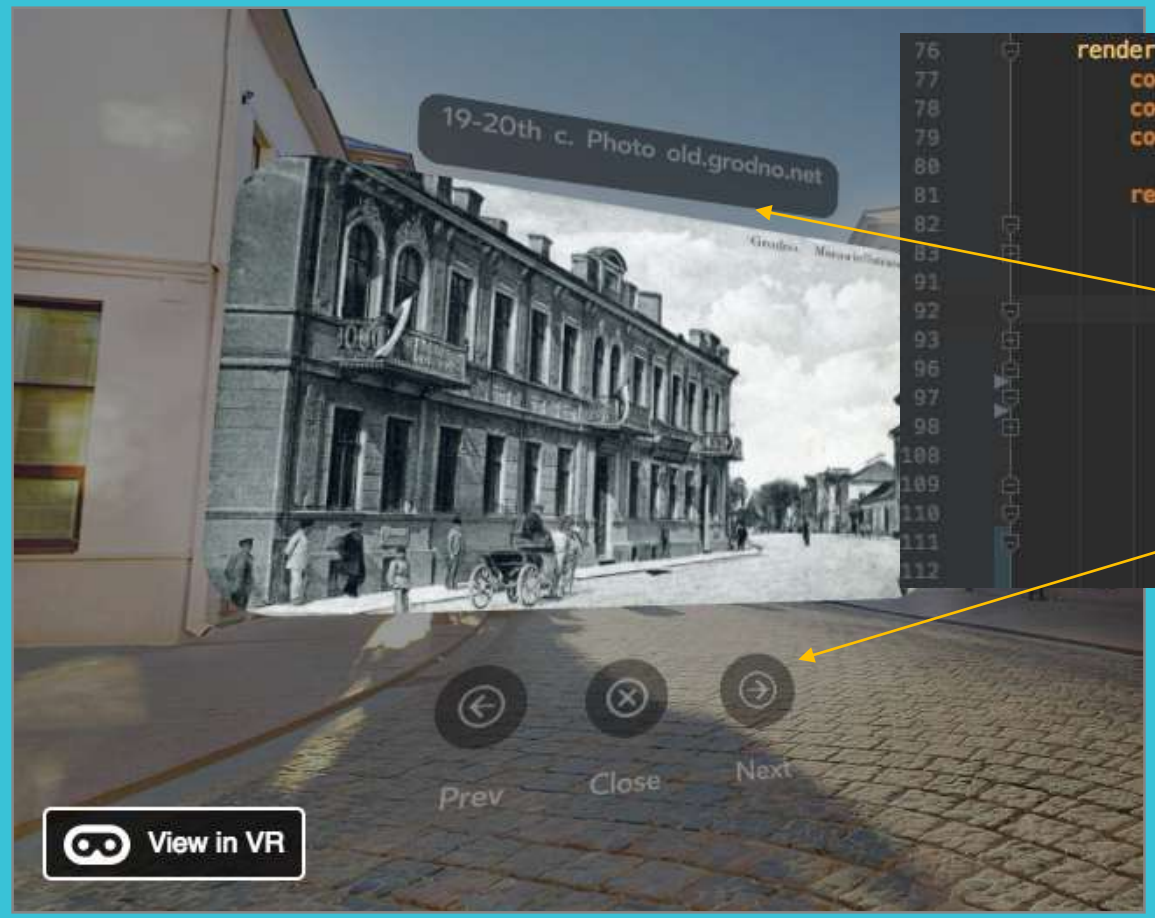
index.js

```
class Gallery extends React.Component {

    constructor(props) {...}

    componentDidMount() {
        this.rotateOnce();
    }

    rotateOnce() {
        this._initRotate = this._initRotate === -5 ? 5 : -5;
        Animated.timing(this.state.rotateY, {
            toValue: this._initRotate,
            duration: 1600
        }).start(() => this.rotateOnce());
    }

    nextImage(step) {...}

    imageLoadHandler() {...}

    render() {
        const { selectedImage, scale, opacity } = this.state;
        const { images = [], locale = 'en', style, onClose } = this.props;
        const currentImage = images[selectedImage] || {};

        return (
            <Animated.View
                style={[...]}
            >
                <View/>
                <View style={styles.imageView}>
                    <Animated.Image
                        onLoad={() => this.imageLoadHandler()}
                        source={asset(currentImage.source)}
                        style={[ styles.image, {...currentImage.style, transform: [{
                    />
```

styles.js

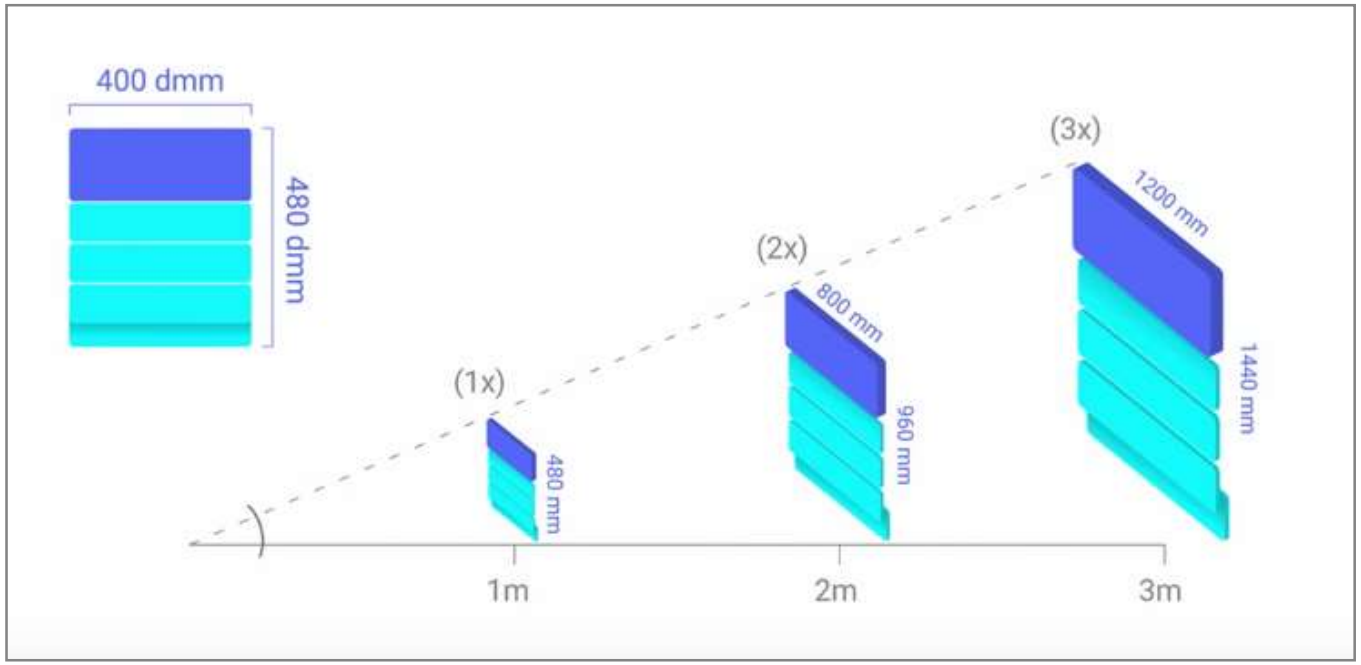
```
import { StyleSheet } from 'react-vr';

const styles = StyleSheet.create({
    galleryView: {
        height: 6,
        position: 'absolute'
    },
    imageView: {
        flexDirection: 'row',
        alignItems: 'center',
        justifyContent: 'center'
    },
    image: {...},
    centerView: {alignItems: 'center'...},
    yearLabel: {
        backgroundColor: 'black',
        borderRadius: 0.2,
        padding: 0.2,
        marginBottom: 0.2,
        textAlign: 'center',
        fontSize: 0.3,
        opacity: 0.5,
        height: 0.7,
        color: 'white'
    },
    tooltipButton: {
        opacity: 0.5,
```

VR UI: Angles & Sizes




```
render() {
    const { selectedImage, scale, opacity } = this.state;
    const { images = [], locale = 'en', style, onClose } = this.props;
    const currentImage = images[selectedImage] || {};

    return (
        <Animated.View
            style={[...]}
        >
            <View style={[styles.centerView, { transform: [{ rotateX: 6 }] }]}>
                <Text/>
            </View>
            <View style={styles.imageView}>
                <Animated.Image/>

            </View>
            <View style={[styles.centerView, { flexDirection: 'row', transform: [{ rotateX: -27 }] }]}>
                <Arrow direction="left"
                       onClick={() => this.nextImage(-1)}
```

VR UI: Angles & Sizes



Text size

Headline	Regular 40dmm
Title	**Medium 32dmm**
Subheading	Regular 28dmm
Body 2	**Medium 24dmm**
Body 1	Regular 24dmm
Caption	Regular 20dmm
BUTTON	**MEDIUM 24dmm**

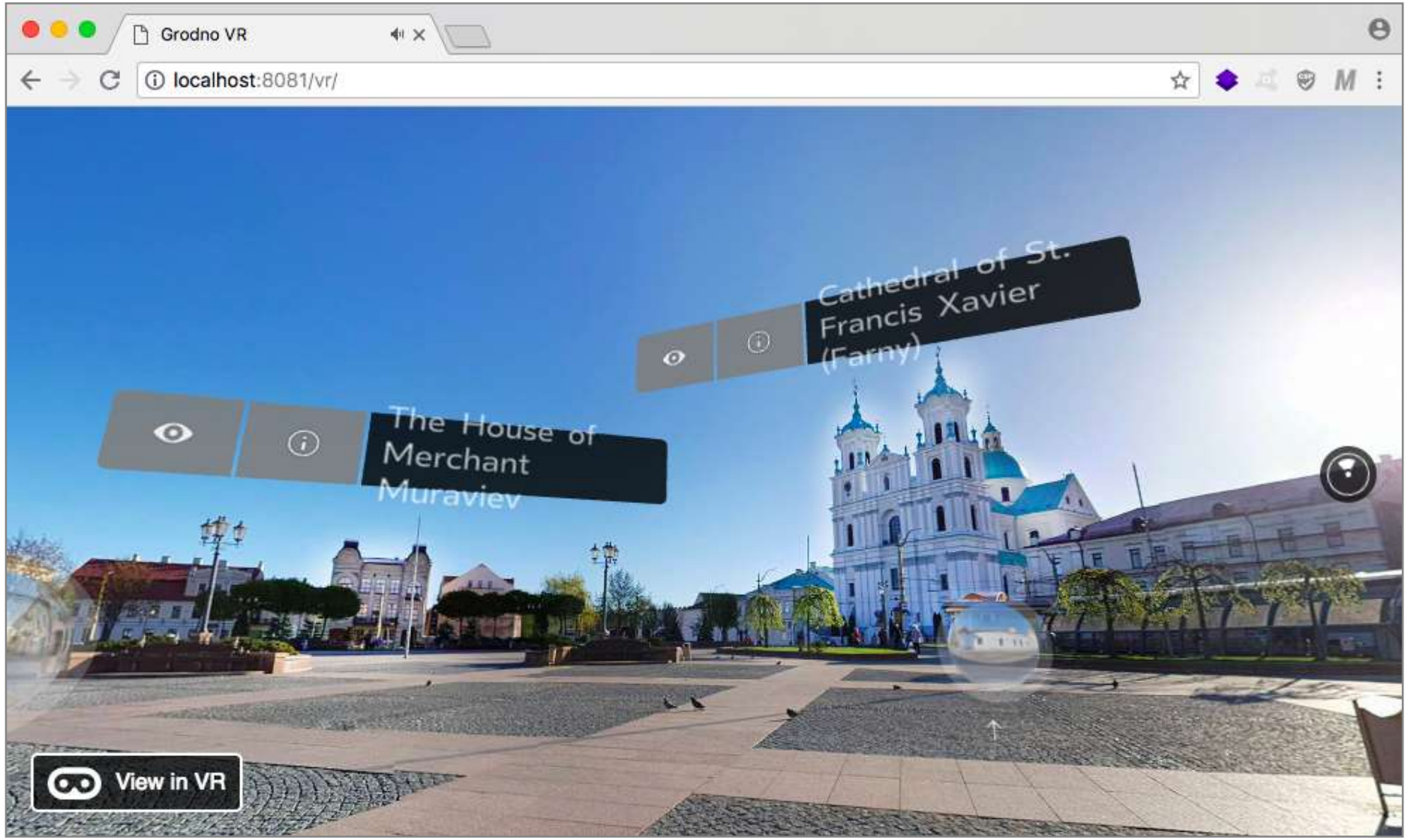
Hit size

Minimum
64x64dmm + 16dmm padding

Comfortable
96x96dmm + 16dmm padding

Frame from «Designing Screen Interfaces for VR (Google I/O ’17)» Adam Glazier

VR UI: Angles & Sizes



VR UI: GazeButton

VR UI: GazeButton

Index.js

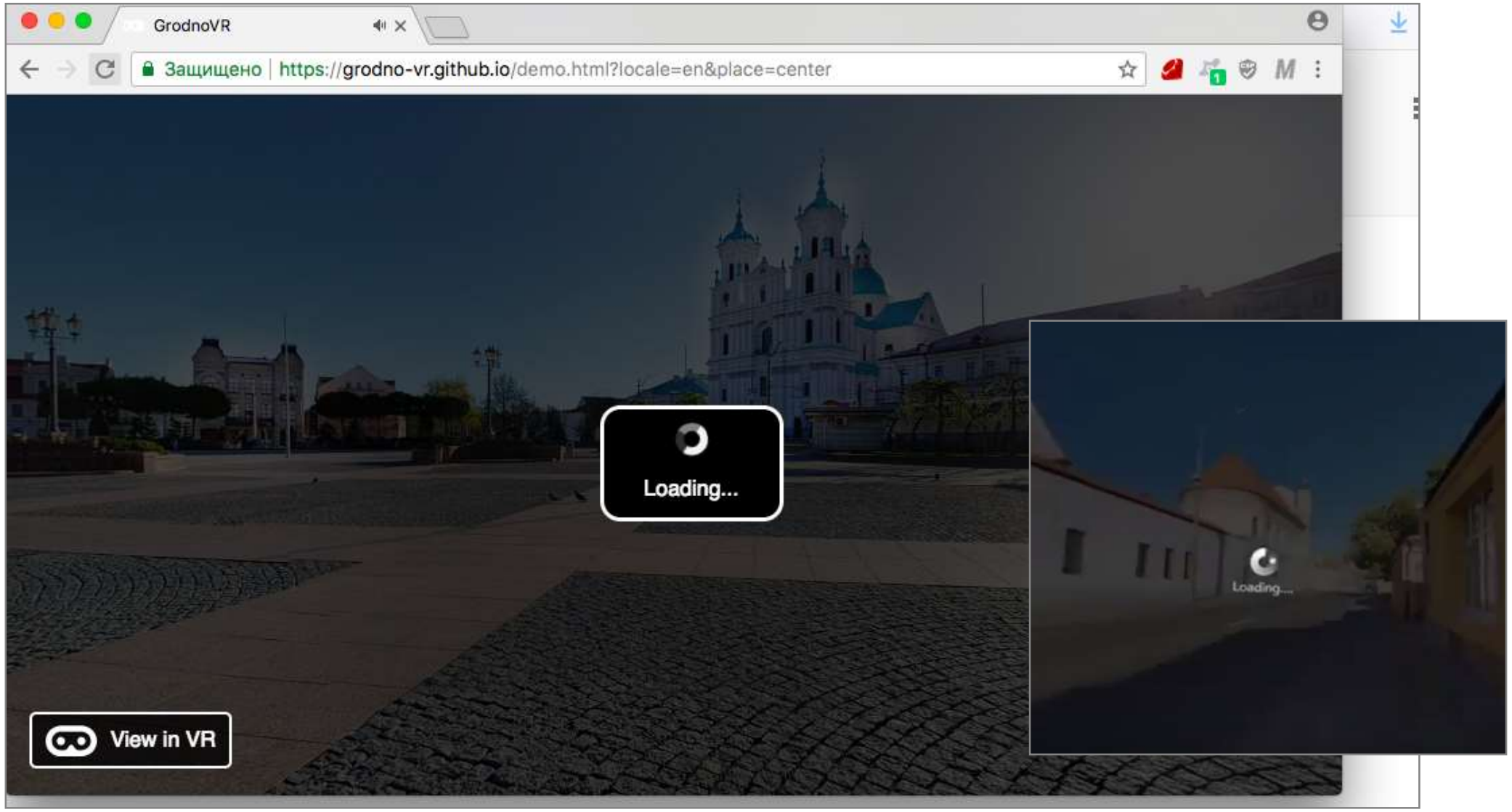
```
const GAZE_TIMEOUT = 1500; // 1.5s
const AnimatedButton = Animated.createAnimatedComponent(VrButton);

class GazeButton extends React.Component {

    constructor(props) {
        super(props);
        this.lastTimeoutId = 0;
        this.state = { hasFocus: false };
    }

    componentWillUnmount() {
        if (this.lastTimeoutId) {
            clearTimeout(this.lastTimeoutId);
        }
    }

    render() {
        const { onClick, onExit, onEnter, children, ...others } = this.props;
        return (
            <AnimatedButton
                {...others}
                onClick={() => onClick()}
                onEnter={() => {
                    onEnter && onEnter();
                    this.setState({ hasFocus: true });
                    this.lastTimeoutId = setTimeout(() => {
                        this.setState({ hasFocus: false });
                        VrSoundEffects.play(asset('audio/click.wav'));
                        onClick();
                    }, GAZE_TIMEOUT);
                }}
                onExit={() => {
                      onExit && onExit();
                      this.setState({ hasFocus: false });
                      clearTimeout(this.lastTimeoutId);
                      this.lastTimeoutId = 0;
                }}
```

Cross-Platform

GROCON

GRODNO 2018

VR & Web Components



VR & Web Components

loadingMixin.js

```
import React from 'react';
import { VrHeadModel, NativeModules } from 'react-vr';
import { VRLoading } from '../../../components'

const { DomOverlayModule } = NativeModules;

const loadingMixin = Base => class extends Base {

    constructor(props) {
        super(props);
        this.state = { ...this.state, loading: false };
    }

    loadingText() {...}

    renderLoading() {
        return this.state.loading
            && VrHeadModel.inVR()
            && <VRLoading text={this.loadingText()} />;
    }

    startLoading() {
        this.setState({ loading: true });
        if (!VrHeadModel.inVR()) {
            DomOverlayModule.loading({ text: this.loadingText() });
        }
    }

    stopLoading() {
        this.setState({ loading: false });
        if (!VrHeadModel.inVR()) {
            DomOverlayModule.closeOverlay();
        }
    }

};
```

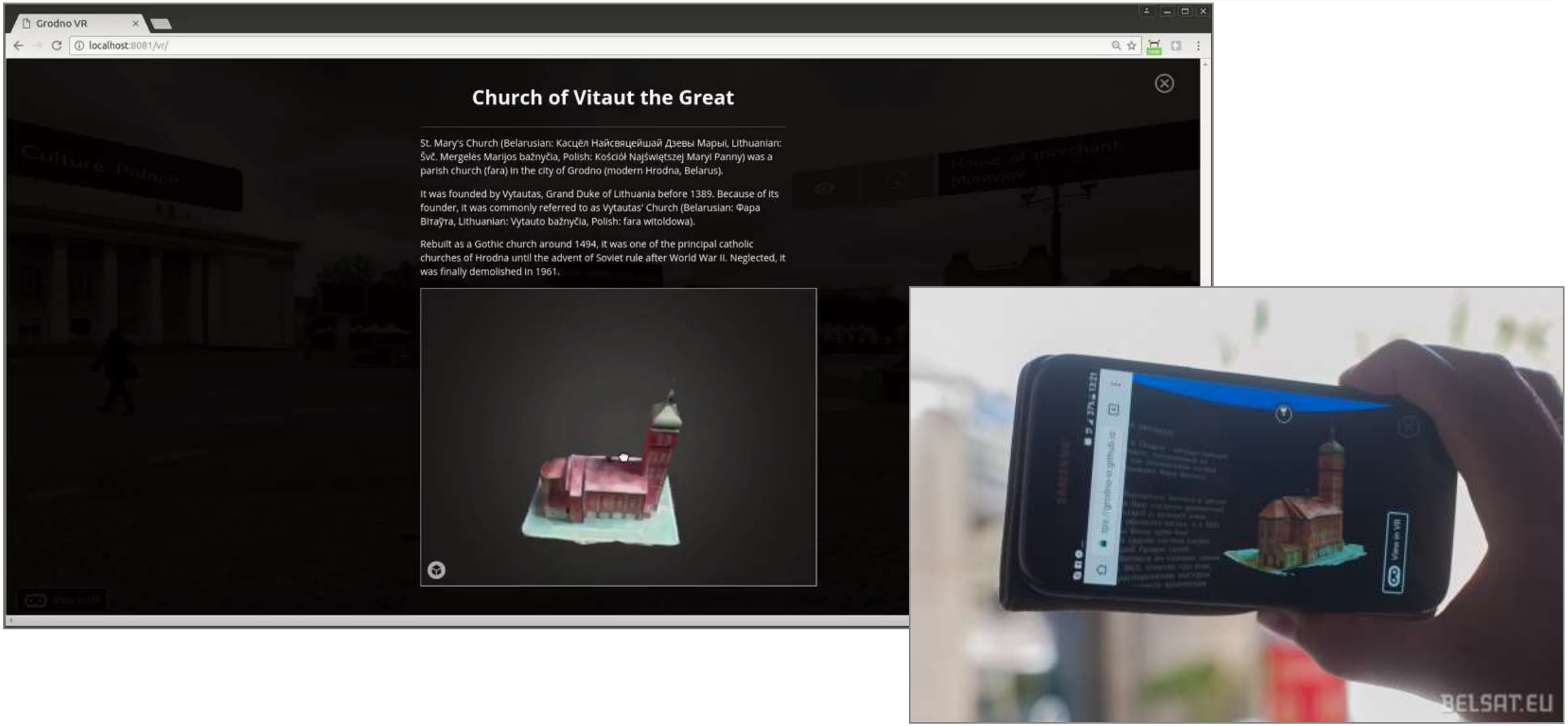
domOverlayModule.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import { Module } from 'react-vr-web';

import WebInformation from '../components/Information/WebInformation';
import WebLoading from '../components/Loading/WebLoading';

class DomOverlayModule extends Module {

    constructor(overlayContainer) {...}

    openInformation(props) {...}

    loading(props = {}) {
        ReactDOM.render(
            <WebLoading {...props} />,
            this._overlayContainer,
        );
    }

    closeOverlay() {...}

}

export default DomOverlay
```

WebLoading.js

```
import React from 'react';

const WebLoading = props => (
    <div className="loading">
        <div className="loader"></div>
        <p>{props.text || 'Loading...'}</p>
    </div>
);

export default WebLoading;
```

Web vs VR


Grodno VR
localhost:8081/vr/
Church of Vitaut the Great
St. Mary's Church (Belarusian: Касцёл Найсвяцейшай Дзевы Марыі, Lithuanian: Švč. Mergelės Marijos bažnyčia, Polish: Kościół Najświętszej Maryi Panny) was a parish church (fara) in the city of Grodno (modern Hrodna, Belarus).
It was founded by Vytautas, Grand Duke of Lithuania before 1389. Because of its founder, it was commonly referred to as Vytautas' Church (Belarusian: Фара Вітаўта, Lithuanian: Vytauto bažnyčia, Polish: fara witoldowa).
Rebuilt as a Gothic church around 1494, it was one of the principal catholic churches of Hrodna until the advent of Soviet rule after World War II. Neglected, it was finally demolished in 1961.
BELSAT.EU

RayCaster

rayCastersModule.js

```
class RayCastersModule {
    constructor(vr) {
        this._vrRef = vr;
        this._currentRayCasters = [this.createMouseRayCaster()];

        this.changeRayCasters(this._currentRayCasters);

        window.addEventListener('vrdisplaypresentchange', event => {
            this._currentRayCasters = [];

            // TODO: OVRUI promise to support multiple ray casters in future
            if (event && event.display && event.display.isPresenting) {
                this._currentRayCasters.push(this.createVrDotRayCater());
            } else {
                this._currentRayCasters.push(this.createMouseRayCaster());
            }

            this.changeRayCasters(this._currentRayCasters);
        });
    }

    createVrDotRayCater() {
        return {
            getType: () => 'simpleDot',
            getRayOrigin: () => [0, 0, 0],
            getRayDirection: () => [0, 0, -1],
            drawsCursor: () => true
            // getMaxLength: () => 20
        };
    }

    createMouseRayCaster() {
        return new OVRUI.MouseRayCaster();
    }

    turnOffRayCasters() {...}
```

Future of ReactVR

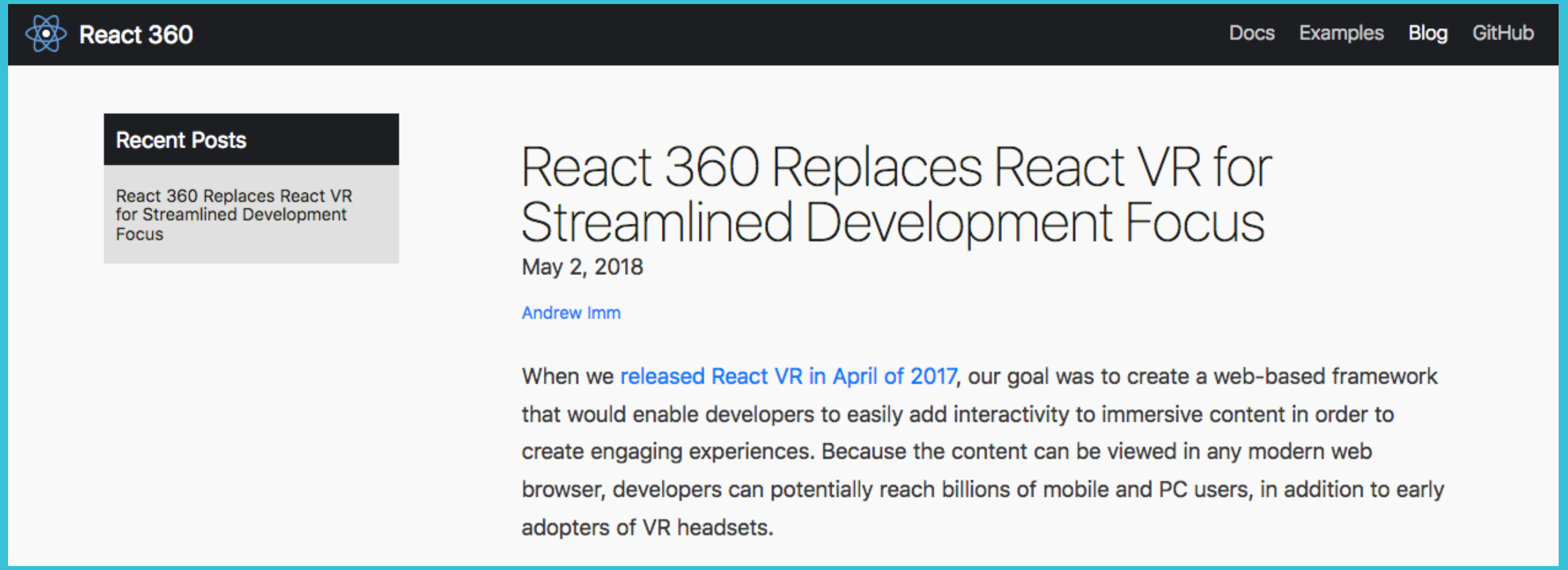
GROCON

GRODNO 2018

React 360 is Replacing React VR

Frame from film «Ready Player One»

React360


React 360
Docs
Examples
Blog
GitHub
Recent Posts
React 360 Replaces React VR for Streamlined Development Focus
React 360 Replaces React VR for Streamlined Development Focus
May 2, 2018
Andrew Imm
When we released React VR in April of 2017, our goal was to create a web-based framework that would enable developers to easily add interactivity to immersive content in order to create engaging experiences. Because the content can be viewed in any modern web browser, developers can potentially reach billions of mobile and PC users, in addition to early adopters of VR headsets.

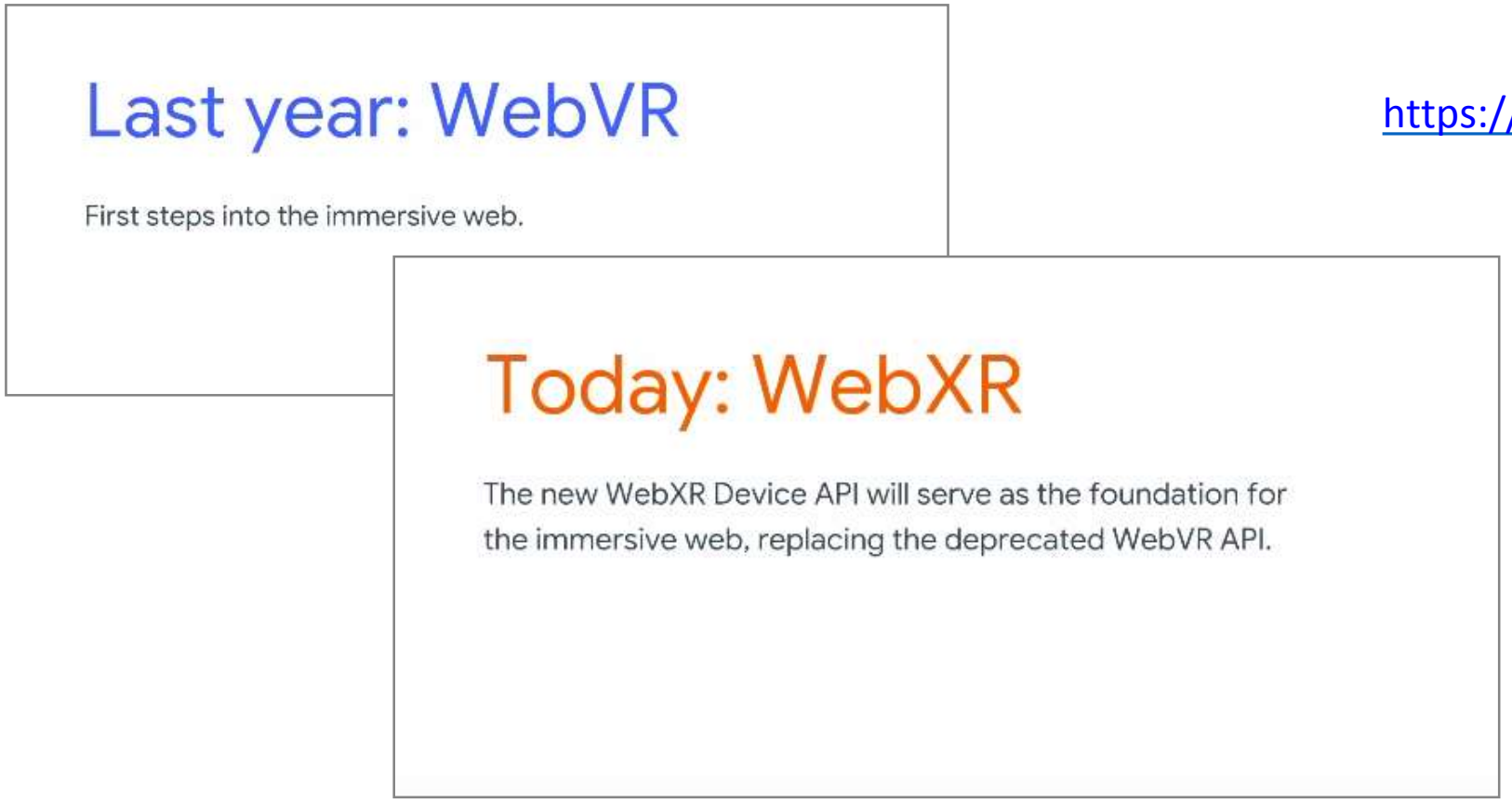
Other libraries

- A-Frame
- Babylon.js
- WebVR + Gamepad + Three.js

WebXR is Replacing WebVR

Frame from film «Ready Player One»

WebVR is deprecated



https://immersive-web.github.io/webxr

Frame from «The future of the web is immersive (Google I/O ’18)» Brandon Jones

- VR works in web
- Immersive content + photogrammetry
- Guidelines: reuse, but not all rules
- Try to support all devices & platforms
- Expect changes and updates
- To be a pioneer

Picture from marvel.wikia.com

Thank You
and Have fun!

Picture from vrscout.com