MOBA1

MOBILE WEB:

COMPONENTS

OVERVIEW

- Modern JavaScript
- Components in Web Development
- Hybrid Apps: PhoneGap/Cordova
- PWAs – Progressive Web Apps
- Pushing the Web Forward?

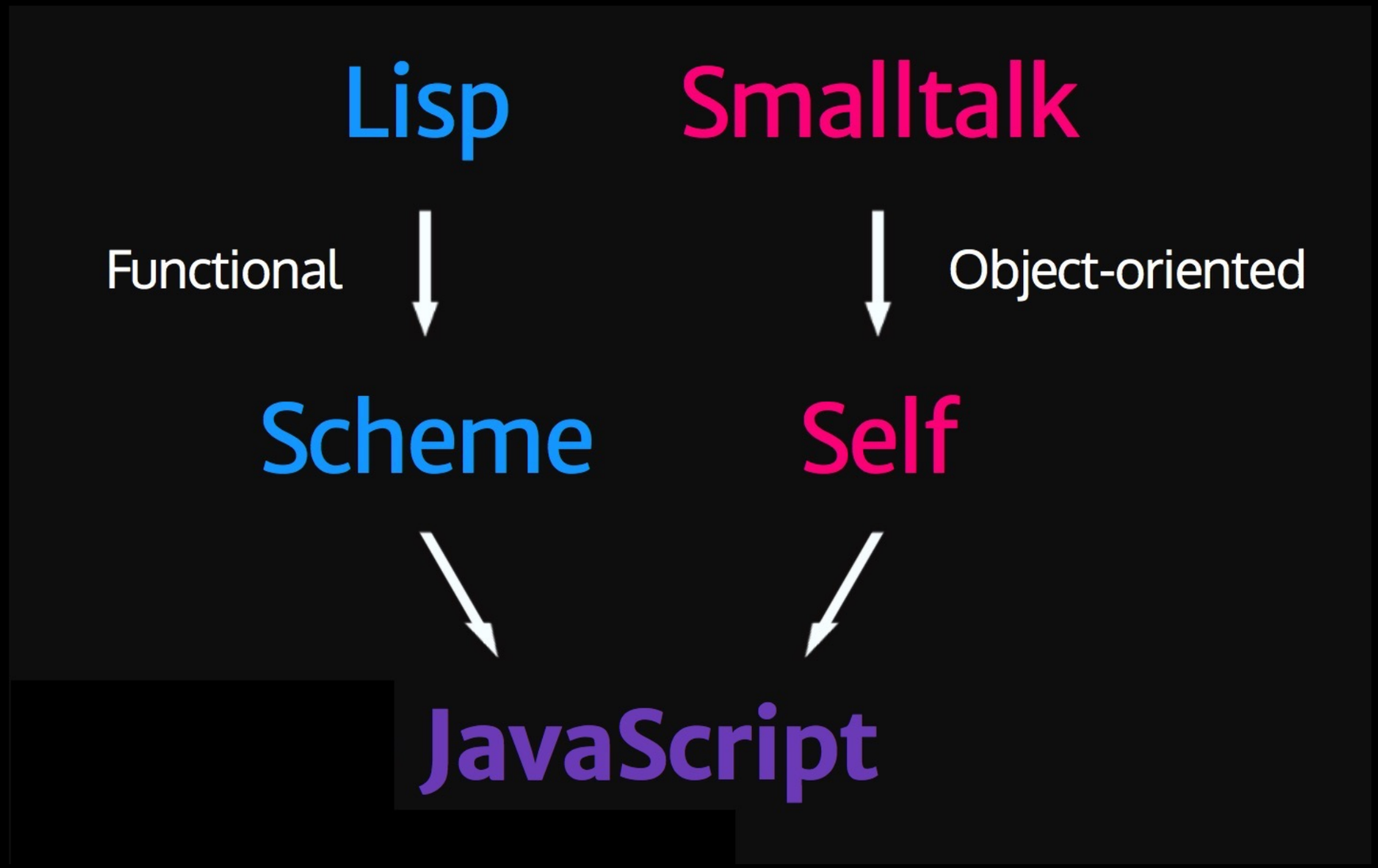
OVERVIEW

- Modern JavaScript
- Components in Web Development
- Hybrid Apps: PhoneGap/Cordova
- PWAs – Progressive Web Apps
- Pushing the Web Forward?

JAVASCRIPT BASICS: WEB2, ...

- JavaScript basics: statements, expressions, variables
- Values and types: numbers, strings, boolean, undefined, ...
- Functions, closures, this, scoping, strict mode
- Objects, methods, constructors, prototypes
- Arrays, Math-object, regular expressions
- If necessary consult:
  - WEB2 slides and related material
  - Speaking JavaScript, Chapter 1: Basic JavaScript

JAVASCRIPT


Lisp
Smalltalk
Functional
Object-oriented
Scheme
Self
JavaScript

ECMA-262 6TH EDITION

Also called:
ECMAScript 6, ES6, ECMAScript 2015, JavaScript 2015

- Major improvement (?) over ES5
- Language spec has almost 600 pages (ES 5.1: 245)
- Much needed features such as modules and classes
- Useful features like Maps, Sets, Promises or Generators

ECMASCRIPT 2016...2020

Version	Features
ES 2016	exponentiation operator `x ** y`
	Method `includes` for arrays
ES 2017	Async functions
	Shared array buffers
	More object and string methods
ES 2018	Asynchronous Iteration
	Rest/Spread Properties
ES 2019	More object and string methods
ES 2020	BigInt – arbitrary precision integers
	Optional chaining
	Nullish coalescing Operator
ES 2021	WeakRef and Finalizers
	Underscores as Numeric Seperator

MODERN JAVASCRIPT

- We will use quite a few of modern JavaScript features in MOBA2 (React and React Native)
- For those who have not attended WBE, a selection of these features will be presented in MOBA2
- Some of the new WBE slides can be found in Moodle

OVERVIEW

- Modern JavaScript
- Components in Web Development
- Hybrid Apps: PhoneGap/Cordova
- PWAs – Progressive Web Apps
- Pushing the Web Forward?

COMPONENTS AND FRAMEWORKS

- Web Components
  - Stencil, Polymer, ...
- Client side UI logic and components
  - React, Vue, ...
- Presentation layer frameworks
  - Ionic, jQuery Mobile, ...
- Native Components
  - React Native, NativeScript, ...

WEB DEVELOPMENT

> *In many instances you're either copying huge chunks of HTML out of some doc and then pasting that into your app ...*

A Guide to Web Components

HTML should be ...

- ... expressive enough to create complex UI widgets
- ... extensible to fill in any gaps with our own tags

This is eventually possible with Web Components

WEB COMPONENTS

- Bundle markup and styles into custom HTML elements
- Fully encapsulate all of their HTML and CSS
- Introduced by Alex Russell at Fronteers Conference 2011

EXAMPLE: IMAGE SLIDER

```
<div id="slider">
  <input type="radio" name="slider" id="slide1" selected="false" checked>
  <input type="radio" name="slider" id="slide2" selected="false"> ...
  <div id="slides">
    <div id="overflow">
      <div class="inner">
        <img src="images//rock.jpg">
        <img src="images/grooves.jpg">...
      </div>
    </div>
  </div>
  <label for="slide1"></label>
  <label for="slide2"></label>...
</div>
```

codepen.io/robdodson/pen/rCGvJ

EXAMPLE: BETTER IMAGE SLIDER

```
<img-slider>
  <img src="images/sunset.jpg" alt="a dramatic sunset">
  <img src="images/arch.jpg" alt="a rock arch">
  <img src="images/grooves.jpg" alt="some neat grooves">
  <img src="images/rock.jpg" alt="an interesting rock">
</img-slider>
```

THE VIDEO ELEMENT

```
<video src="./foo.webm" controls></video>
```

- There’s a play button, a scrubber, timecodes, a volume slider
- A way to build the *video* element from these parts was needed
- Browser makers created a secret place: the *Shadow DOM*

You can activate *Show user agent shadow DOM* in the browser’s DevTools

THE VIDEO ELEMENT

nts | Network | Sources | Timeline | Profiles | Resources | Audits | Console

```
<video id="video" controls preload="none" poster="http://media.w3.org/2010/05/sintel/
poster.png">
  #shadow-root (user-agent)
    <div>
      <div>
        <div>
          <input type="button">
          <input type="range" step="any" max="0">
          <div style="display: none;">0:00</div>
          <div>0:00</div>
          <input type="button">
          <input type="range" step="any" max="1" style="display: none;">
          <input type="button" style="display: none;">
          <input type="button" style="display: none;">
```

TEMPLATES

- New `template` element
- Not rendered on the page until it is activated using JavaScript

```
<template>
  <h1>Hello there!</h1>
  <p>This content is top secret :)</p>
</template>
```

SHADOW DOM

Select an element and call its *attachShadow* method

```
<!-- HTML -->
<div class="container"></div>
```

```
// JavaScript
var host = document.querySelector('.container')
var root = host.attachShadow({mode: 'open'})
root.innerHTML = '<p>How <em>you</em> doin?</p>'
```

SHADOW HOST AND SHADOW ROOT

- Shadow Host
  - Element that *attachShadow* is called on
  - The only piece visible to the user
  - The place where the element is supplied with content
  - Example: the *video* element is the shadow host
- Shadow Root
  - Document fragment returned by *attachShadow*
  - It and its descendants are hidden from the user
  - But they’re what the browser will actually render

CUSTOM ELEMENT

```
class ImageSlider extends HTMLElement {
  constructor() {
    super()
    const shadowRoot = this.attachShadow({mode: 'closed'})
    shadowRoot.innerHTML = `
      <style>
        ...
      </style>
      <div class="slider">
        ...
      </div>
    `
  }
}
customElements.define('image-slider', ImageSlider);
```

WEB COMPONENTS SUMMARY

Based on these pieces:

- Shadow DOM
- Custom Elements
- HTML Templates
- CSS additions

GitHub Repository w3c/webcomponents
MDN Web Components

BROWSER SUPPORT

- Web Comonents were introduced in 2011
- By now, Web Components should be everywhere
- Browser support: good
  caniuse.com/#search=Web%20Components
- Reason for slow progress: vendors couldn’t agree
- Web Components were a Google effort

WEB COMPONENT LIBRARIES

- Stencil: Web Component compiler
  https://stenciljs.com
- Lit (Successor of Polymer)
  https://lit.dev
- X-Tag: Mozilla’s alternative
  www.x-tags.org

COMPONENTS AND FRAMEWORKS

- Web Components
  - Stencil, Polymer, ...
- Client side UI logic and components
  - React, Vue, ...
- Presentation layer frameworks
  - Ionic, jQuery Mobile, ...
- Native Components
  - React Native, NativeScript, ...

REACT

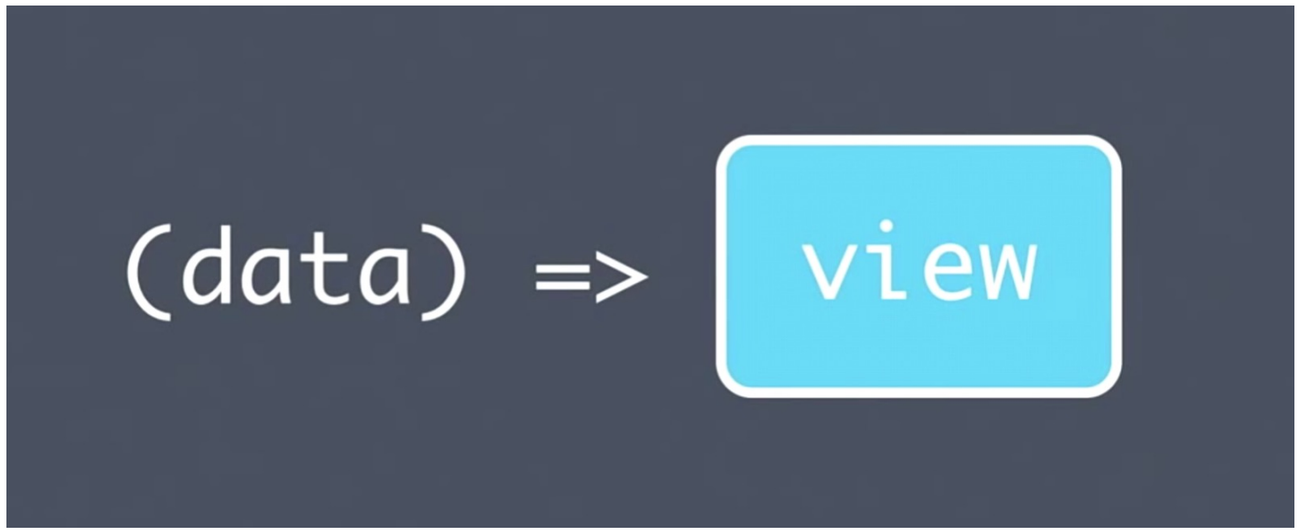
```
function Welcome(props) {
  return <h1>Hello, {props.name}</h1>
}

ReactDOM.render(
  <Welcome name="Taylor" />,
  mountNode
)
```

https://reactjs.org

REACT

React wraps an imperative API with a declarative one



REACT?

- React is the "View" in the application
- It is not (only) a framework, it is mainly a concept
- Helps you organize your templates in components

Short: We have components and fast rendering

More on React in MOBA2

VUE.JS

- Progressive framework for building user interfaces
- Core library is focused on the view layer only
- Capable of powering sophisticated Single-Page Applications

https://vuejs.org

VUE.JS

```
<div id="app-4">
  <ol>
    <li v-for="todo in todos"> {{ todo.text }} </li>
  </ol>
</div>
```

```
var app4 = new Vue({
  el: '#app-4',
  data: {
    todos: [
      { text: 'Learn JavaScript' },
      { text: 'Learn Vue' },
      { text: 'Build something awesome' }
    ]
  }
})
```

SVELTEJS

- Framework for building UIs, like Vue or React
- Svelte is a compiler, unlike React or Vue
- No virtual DOM, code compiled to vanilla JS
- Truly reactive framework, no complex state management libraries

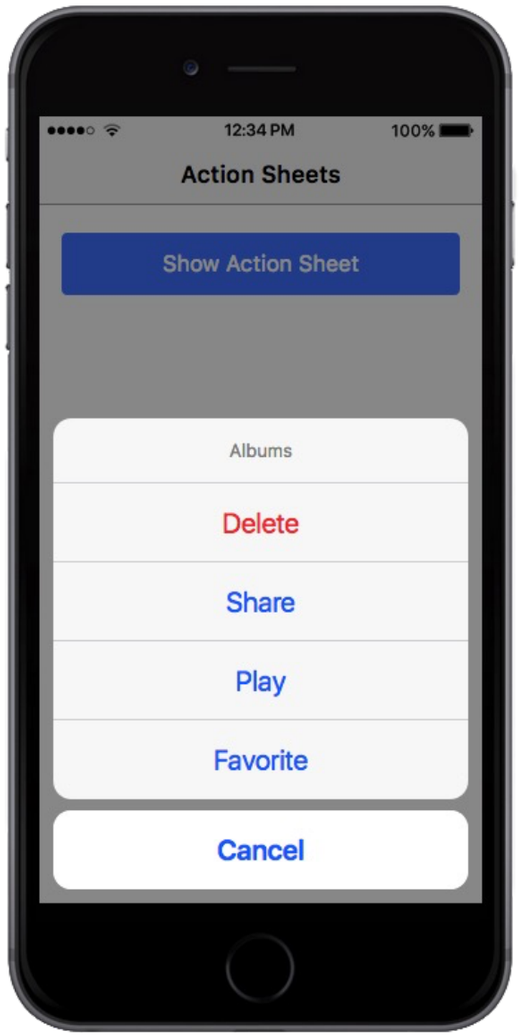
https://svelte.dev

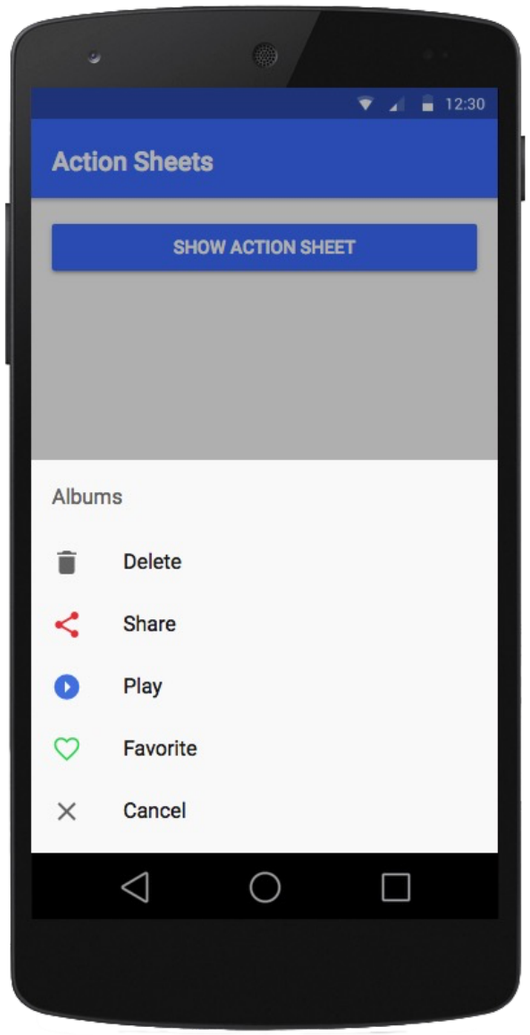
IONIC

- Open source UI toolkit for building mobile and desktop apps using web technologies
- Originally (2015) built on top of Angular and Apache Cordova
- Now official support for React and Vue, too

https://ionicframework.com

IONIC





Extensive Documentation

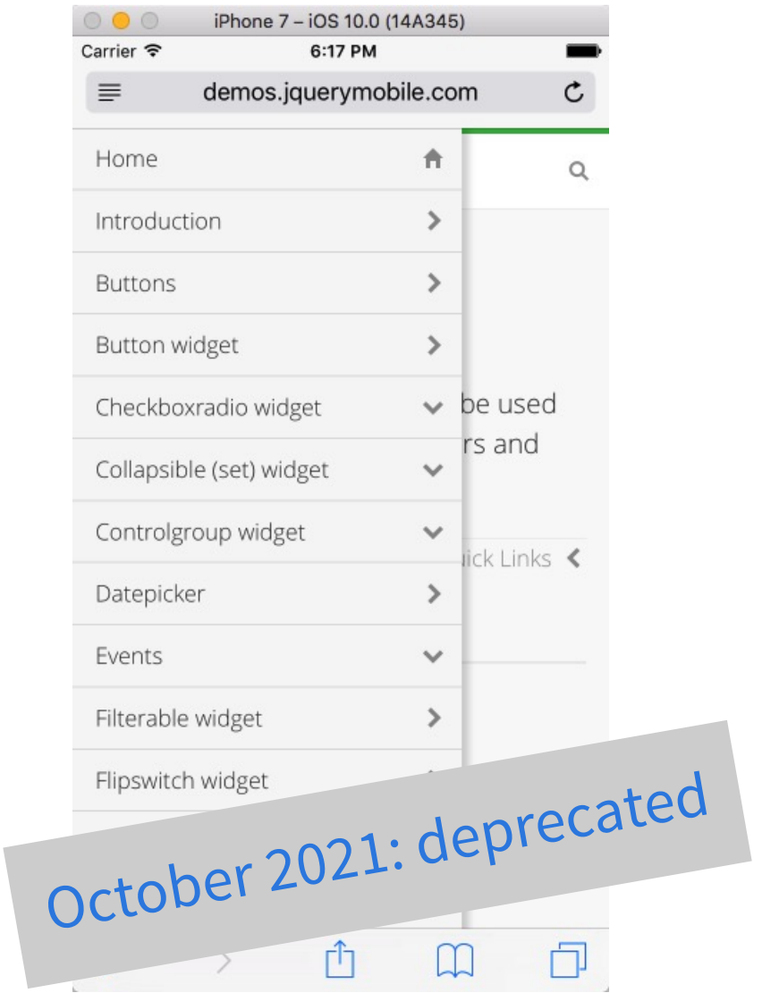
Docs: Components

FRAMEWORK 7

- Develop mobile, desktop, web apps with native look and feel
- Set of ready to use UI elements and widgets
- Focused on iOS and Google Material design
- Comes with Vue.js & React components
- Also a prototyping apps tool

https://framework7.io

WHAT ABOUT JQUERY MOBILE?



- Touch-optimized web framework
- Built on top of jQuery core
- Theming framework
- HTML5-driven for laying out pages with minimal scripting
- AJAX-powered navigation with animated page transitions
- Latest release: 1.4.5 (Oct 14)
- http://jquerymobile.com

OTHER OPTIONS

- React Native
  - Bild native iOS and Android apps with one codebase
  - Facebook, language: JavaScript, based on React.js
  - Used by Facebook, Walmart, Tesla, Airbnb, and others
- NativeScript
  - JavaScript or any language that transpiles to it
  - Fully native mobile apps
  - Supports Angular and Vue
- Flutter
  - Google’s UI toolkit for mobile, web, and desktop
  - Uses Dart, compiles to native
  - Uses neither WebView nor OEM widgets, custom rendering engine

OTHER OPTIONS

- Xamarin
  - Build one app that works on multiple platforms
  - Microsoft, language: C#
  - Some prebuilt apps to get a quick start
  - Used by Slack, Pinterest, Honeywell, and others
- Titanium
  - Apps are written in Javascript but produce a native app
  - Own set of APIs to bridge native APIs to Javascript

OVERVIEW

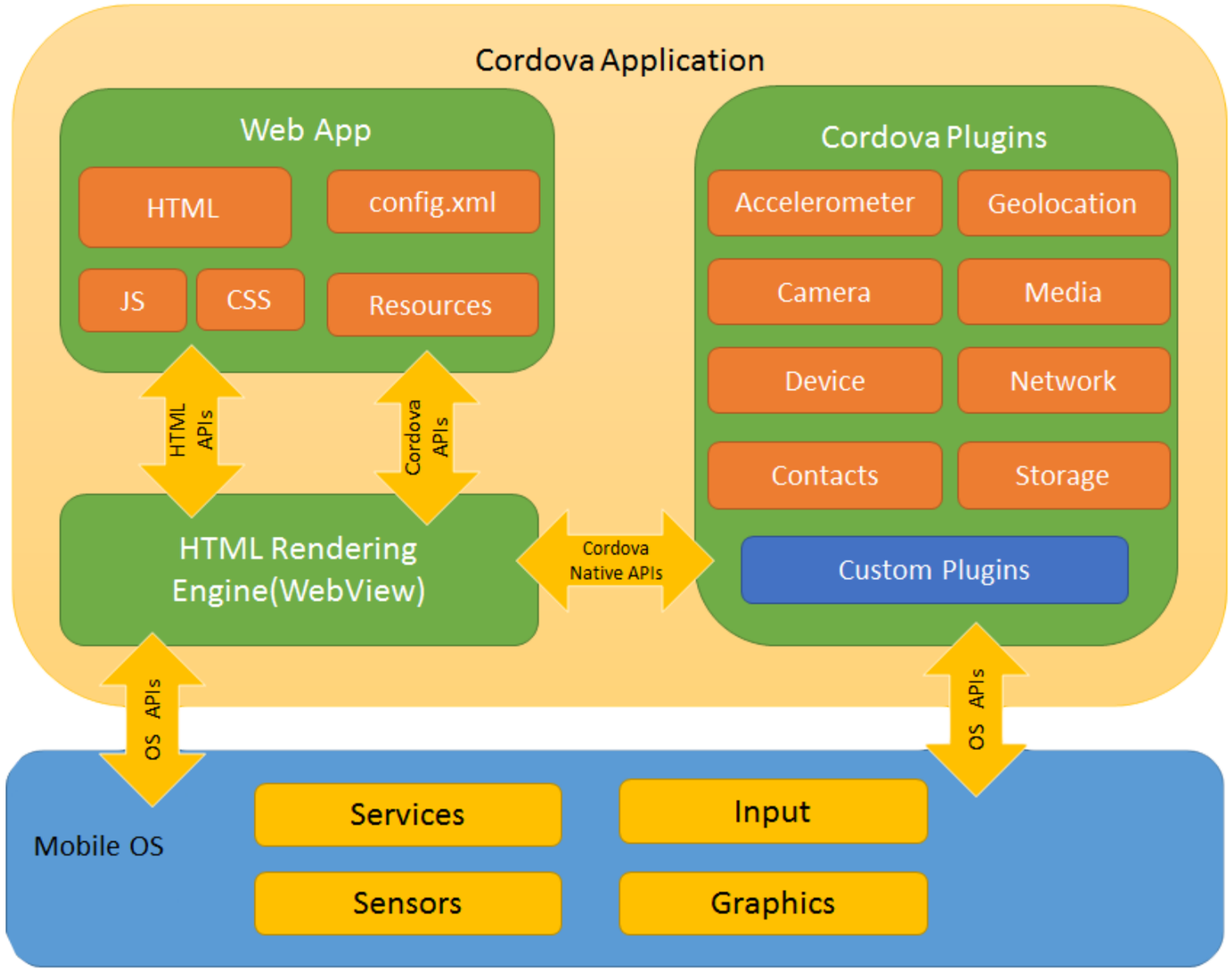
- Modern JavaScript
- Components in Web Development
- Hybrid Apps: PhoneGap/Cordova
- PWAs – Progressive Web Apps
- Pushing the Web Forward?

CORDOVA

- Open-source mobile development framework
- Apache Project, cross-platform
- Package web applications in native containers
  - Use standard web technologies - HTML, CSS, JS
  - Similar concept (Ionic Team): Capacitor
  - Similar concept on Desktop: Electron, NW.js

http://cordova.apache.org

CORDOVA – ARCHITECTURE



CORDOVA – WEBVIEW

- May provide the application with its entire UI
- By default a local file named *index.html*
- References CSS, JavaScript, images, media files
- Also possible: mix with native application components
- *config.xml* provides information about the app

CORDOVA – PLUGINS

- Integral part of the cordova ecosystem
- Provide an interface for Cordova and native components
- Bindings to standard device APIs
- Core Plugins provide access to device capabilities

https://cordova.apache.org/docs/en/latest/guide/support/

CORDOVA – CORE PLUGIN APIS

Platform:	Android	iOS	OS X	Windows 8.1, Phone 8.1, 10	Electron
CLI shorthand:	android	ios	osx	windows	electron
BatteryStatus	✔	✔	✘	✔ Windows Phone 8.1 only	Tests Pending
Camera	✔	✔	✘	✔	✔
Capture	✔	✔	✘	✔	Tests Pending
Connection	✔	✔	✘	✔	Tests Pending
Device	✔	✔	✔	✔	Tests Pending
Events	✔	✔	✘	✔	Tests Pending
File	✔	✔	✔	✔	Tests Pending
Geolocation	✔	✔	✘	✔	Tests Pending
Globalization	✔	✔	✘	✔	Tests Pending
InAppBrowser	✔	✔	✘	uses iframe	Tests Pending
Media	✔	✔		✔	Tests Pending
Notification	✔	✔	✘	✔	Tests Pending
Splashscreen	✔	✔	✘	✔	Tests Pending
Status Bar	✔	✔	✘	✔ WP 8.1 only	Tests Pending
Storage	✔	✔	✘	✔ localStorage & indexedDB	Tests Pending
Vibration	✔	✔	✘	✔ WP 8.1 only	✘

CORDOVA – PLUGINS

- Third-party plugins with additional bindings
- Not necessarily available on all platforms
- You can also develop your own plugins

https://www.npmjs.com/search?q=ecosystem%3Acordova

DEVELOPMENT PATHS

- Cross-platform (CLI) workflow
  - If you want your app to run on different platforms
  - Little need for platform-specific development
  - Centers around the Cordova CLI
  - Provides a common interface to apply plugins
- Platform-centered workflow
  - Focus on building an app for a single platform
  - Modifications at a lower level possible
  - Mix custom native components with web-based components

INSTALLATION (CLI)

```
# OS X and Linux
$ sudo npm install -g cordova

# Windows
C:\>npm install -g cordova
```

You should now be able to run cordova on the command line with no arguments and it should print help text

CREATE APP AND ADD PLATFORMS

```
$ cordova create hello com.example.hello HelloWorld

$ cd hello
$ cordova platform add ios
$ cordova platform add android

$ cordova platform ls
```

- Creates the required directory structure for the app
- By default, generates a skeletal web-based application
- Home page is the project's *www/index.html* file

https://cordova.apache.org/docs/en/latest/guide/cli/index.html

INSTALL PRE-REQUISITES FOR BUILDING

```
$ cordova requirements
Requirements check results for android:
Java JDK: installed .
Android SDK: installed
Android target: installed android-19,android-21,android-22,...
Gradle: installed

Requirements check results for ios:
Apple OS X: not installed
Cordova tooling for iOS requires Apple OS X
Error: Some of requirements check failed
```

https://cordova.apache.org/docs/en/latest/guide/platforms/android/
https://cordova.apache.org/docs/en/latest/guide/platforms/ios

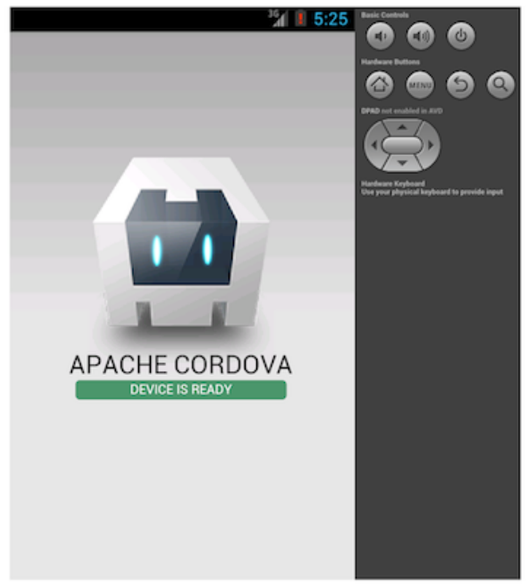
BUILD AND TEST THE APP

```
# build for all platforms
$ cordova build

# build for a specific platform
$ cordova build ios

# use platform specific emulator
$ cordova emulate android

# or plug the mobile into
# your computer and test
$ cordova run android
```



ADD PLUGINS

- Add plugins in order to access device-level features
- A plugin exposes a Javascript API for native SDK functionality

```
$ cordova plugin search camera
...
$ cordova plugin add cordova-plugin-camera
Fetching plugin "cordova-plugin-camera@~2.1.0" via npm
Installing "cordova-plugin-camera" for android
Installing "cordova-plugin-camera" for ios

$ cordova plugin ls
...
```

PHONEGAP (ADOBE)

PhoneGap FAQ

Q: What Is The Difference Between Phonegap And Cordova?

A: In October 2011, PhoneGap was donated to the Apache Software Foundation (ASF) under the name Apache Cordova. Through the ASF, future PhoneGap development will ensure open stewardship of the project. It will remain free and open source under the Apache License, Version 2.0. PhoneGap is an open source distribution of Cordova. Think about Cordova's relationship to PhoneGap like WebKit's relationship to Safari or Chrome.

For more details, read the blog post PhoneGap, Cordova, and what's in a name?

Adobe has discontinued PhoneGap Build and ended investment in PhoneGap and Apache Cordova. http://phonegap.com/

OVERVIEW

- Modern JavaScript
- Components in Web Development
- Hybrid Apps: PhoneGap/Cordova
- PWAs – Progressive Web Apps
- Pushing the Web Forward?

PROGRESSIVE WEB APPS

Combine the best of web and mobile apps. Think of it as a website built using web technologies but that acts and feels like an app. Main characteristics:

- Discoverable: find through search engines
- Installable: available on home screen
- Linkable: share via URL
- Network independent: works offline
- Progressive: usable on older browsers
- Re-engageable: notifications
- Responsive: usable on various device types
- Safe: secure connection

BROWSER SUPPORT: SERVICE WORKERS

- PWAs don’t depend on a single API
- Combination of various technologies
- Key ingredient: service worker support
- Now supported on all major browsers on desktop and mobile

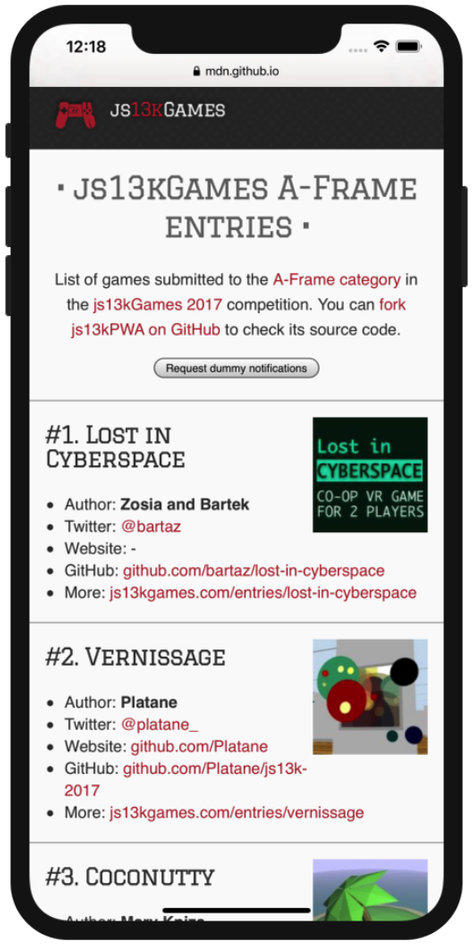
Service Worker API
https://developer.mozilla.org/en-US/docs/Web/API/Service_Worker_API
https://caniuse.com/#search=service

BROWSER SUPPORT: OTHER FEATURES

- Web App Manifest
- Push API
- Notifications API
- Add to Home Screen

Partial browser support (limited support in Safari)

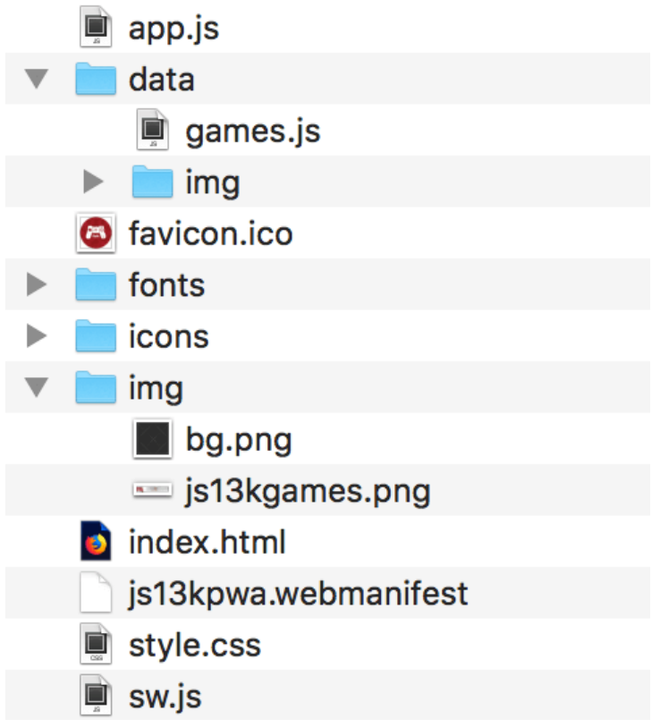
SAMPLE APP



Information about games submitted to the A-Frame category in the js13kGames 2017 competition

https://mdn.github.io/pwa-examples/js13kpwa/
https://github.com/mdn/pwa-examples/tree/master/js13kpwa

SAMPLE APP FILES



- HTML file index.html
- Basic CSS styling
- Images, scripts, fonts

SERVICE WORKERS

- Virtual proxy between the browser and the network
  - Cache the assets of a website
  - Make them available when the user's device is offline
- Run on a separate thread from the main JavaScript
- They don't have any access to the DOM structure
- Non-blocking API
- Can only be executed in secure contexts (https)

REGISTERING THE SERVICE WORKER

```
if('serviceWorker' in navigator) {
    navigator.serviceWorker.register('./pwa-examples/js13kpwa/sw.js')
}
```

- Checks if API is supported in the browser
- Registers service worker *sw.js*
- Service Worker-specific code is in *sw.js*
- Service Worker is automatically downloaded, installed, and activated

PREPARE CACHE DATA

```
var cacheName = 'js13kPWA-v1'

var appShellFiles = [
  '/pwa-examples/js13kpwa/',
  '/pwa-examples/js13kpwa/index.html',
  '/pwa-examples/js13kpwa/app.js',
  '/pwa-examples/js13kpwa/style.css',
  '/pwa-examples/js13kpwa/fonts/graduate.eot',
  ... ]

var gamesImages = []

for (let game of games) {
    gamesImages.push('data/img/' + game.slug + '.jpg')
}

var contentToCache = appShellFiles.concat(gamesImages)
```

INITIALIZE CACHING

```
self.addEventListener('install', (e) => {
  console.log('[Service Worker] Install')
  e.waitUntil(
    caches.open(cacheName).then((cache) => {
      console.log('[Service Worker] Caching all: app shell and content')
      return cache.addAll(contentToCache)
    })
  )
})
```

- Uses a special CacheStorage object `caches`
- Saving to web storage won’t work
- Cache is opened and files are added

RESPONDING TO FETCHES

```
self.addEventListener('fetch', (e) => {
  e.respondWith(
    caches.match(e.request).then((r) => {
      return r ||
        fetch(e.request).then((response) => {
          return caches.open(cacheName).then((cache) => {
            cache.put(e.request, response.clone())
            return response
          })
        })
    })
  )
})
```

- The `fetch` event fires at each HTTP request from our app
- Find the resource in the cache and return it if it’s there
- If not, fetch it from the network and store it in the cache

MAKE PWAS INSTALLABLE

Needed:

- A web manifest, with the correct fields filled in
- Website to be served from a secure (HTTPS) domain
- Icon to represent the app on the device
- A registered service worker (Chrome for Android)

THE MANIFEST FILE

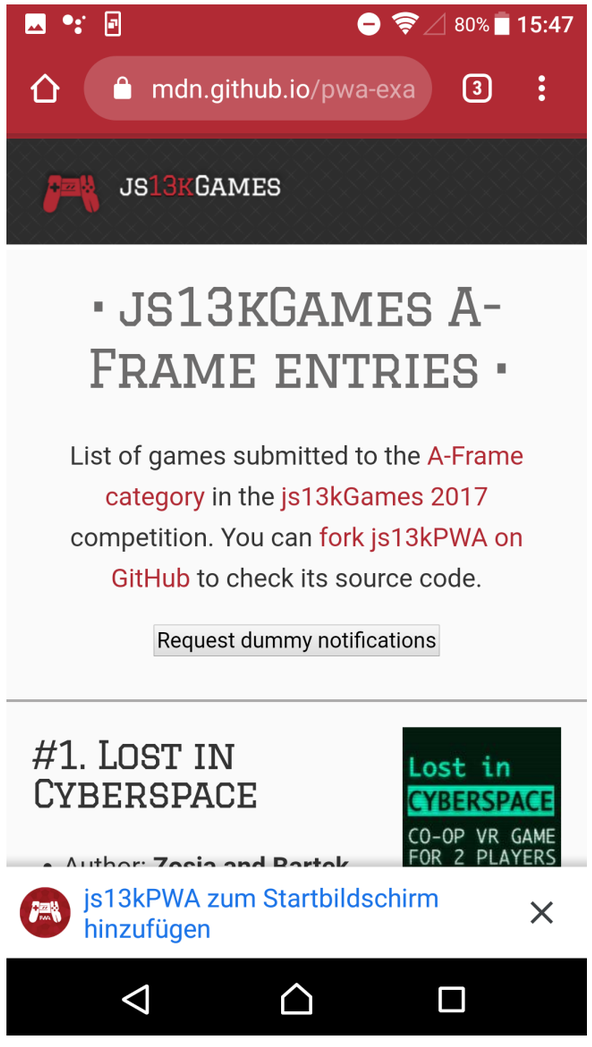
- Lists all the information about the website (JSON)
- Usually resides in the root folder of a web app
- Contains the title, paths to different-sized icons, ...
- Load the manifest file in the head of the HTML

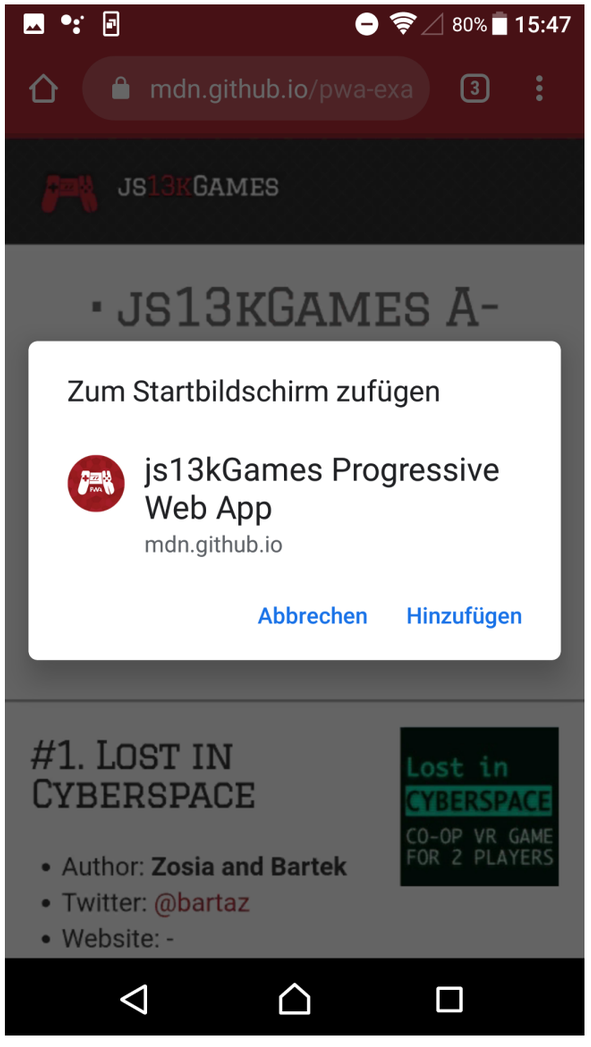
```
<link rel="manifest" href="js13kpwa.webmanifest">
```

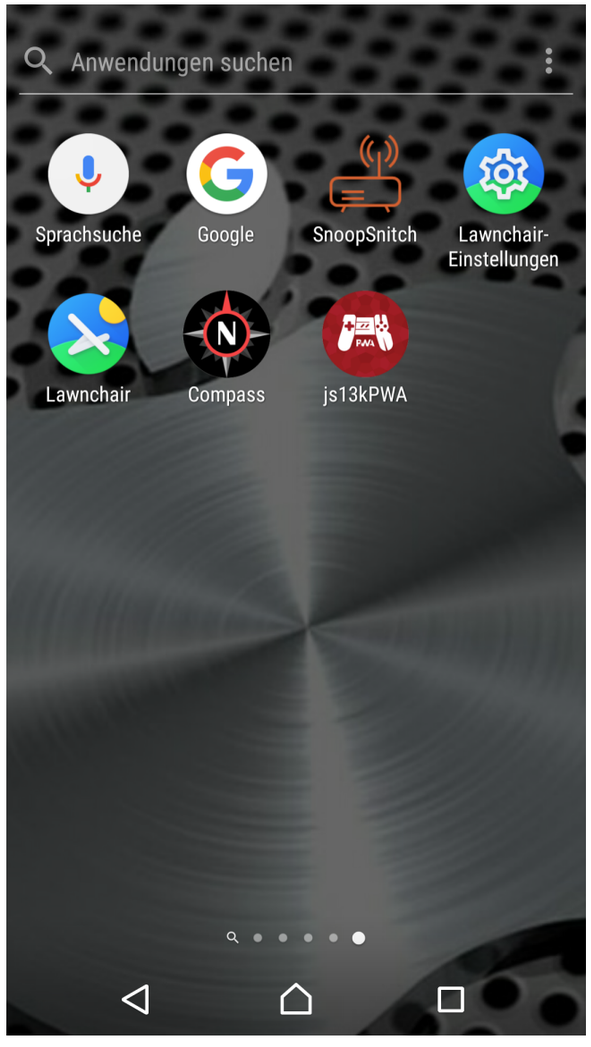
ADD TO HOME SCREEN

- Banner indicates that the app is installable as a PWA
- If the user agrees, an install banner is shown
- Some browsers also create a splash screen for the app

ADD TO HOME SCREEN







INSTALL BANNER CRITERIA (GOOGLE)

- You have a web app manifest file, which defines how your app appears on the user's system and how it should be launched
- The manifest must have a `short_name`, a `name` for display in the banner,
- A start URL (e.g. `/` or `index.html`) which must be loadable,
- *At least* an `192x192` PNG icon
- Your icon declarations should include a mime type of `image/png`
- You have a service worker registered on your site. We recommend a simple custom offline page service worker
- Your site is served over HTTPS (service worker requires HTTPS for security)
- The user has visited your site at least twice, with at least five minutes between visits.

★ **Note:** The criteria will change over time. For more information read the FAQ.

NOTIFICATIONS AND PUSH

- Notifications API

  https://developer.mozilla.org/en-US/docs/Web/API/Notifications_API

- Push API

  https://developer.mozilla.org/en-US/docs/Web/API/Push_API

Only partial support in iOS Safari...

IMAGES

- Reduce file sizes if possible
  (JPEG compression, PNG optimziers)
- Check screen sizes and resolution and load most suitable image (responsive images)
- Other strategies:
  - Placeholder images
  - Load on demand

PLACEHOLDER IMAGES

```
<img src='data/img/placeholder.png' data-src='data/img/SLUG.jpg' alt='NAME'>
```

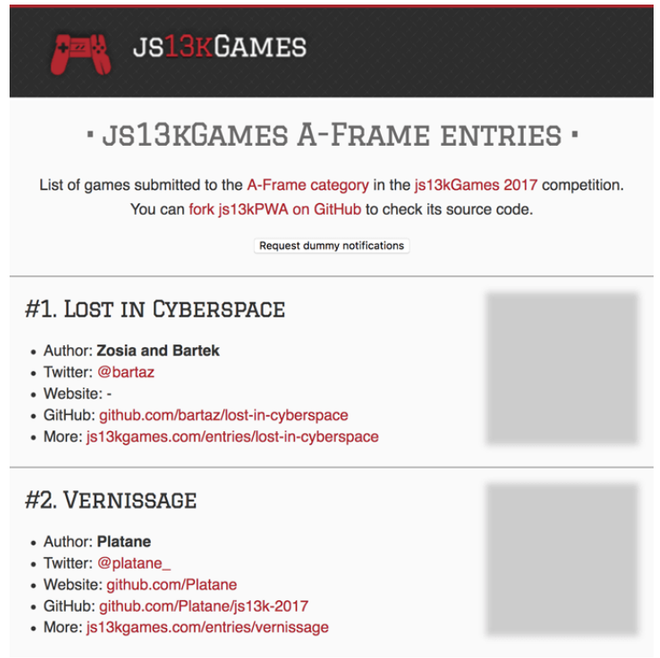
```
let imagesToLoad = document.querySelectorAll('img[data-src]')

const loadImage = (image) => {
  image.setAttribute('src', image.getAttribute('data-src'))
  image.onload = () => {
    image.removeAttribute('data-src')
  }
}

imagesToLoad.forEach((img) => {
  loadImage(img)
})
```

BLUR IN CSS

- Placeholder image can be blurred in CSS
- CSS transition to remove the blur effect




```
article img[data-src] {
  filter: blur(0.2em);
}

article img {
  filter: blur(0em);
  transition: filter 0.5s;
}
```

LOADING ON DEMAND

- Loading images that are not currently shown in the viewport can be delayed
- For that purpose, use the Intersection Observer API
- When an image is visible, we load and stop observing it
- Progressive enhancement to the previously working example
- If Intersection Observer is not supported, just load the images
- Could be extended to load list items conditionally, too (infinite scrolling)

https://developer.mozilla.org/en-US/docs/Web/API/Intersection_Observer_API

LOADING ON DEMAND

```
if ('IntersectionObserver' in window) {
  const observer = new IntersectionObserver((items, observer) => {
    items.forEach((item) => {
      if (item.isIntersecting) {
        loadImage(item.target)
        observer.unobserve(item.target)
      }
    })
  })
  imagesToLoad.forEach((img) => {
    observer.observe(img)
  })
} else {
  imagesToLoad.forEach((img) => {
    loadImage(img)
  })
}
```

PROGRESSIVE WEB APPS

Front-End Developer Handbook 2018:
https://frontendmasters.com/books/front-end-handbook/2018/

In 2018 Expect...

„Progressive Web Applications hopefully will catch fire. If they don’t, I fear they never will. At least not in their current form.“

PWAs not mentioned in the 2019 Handbook recap of 2018...

OVERVIEW

- Modern JavaScript
- Components in Web Development
- Hybrid Apps: PhoneGap/Cordova
- PWAs – Progressive Web Apps
- Pushing the Web Forward?

JAVASCRIPT CHANGES

Front-End Developer Handbook 2019:
https://frontendmasters.com/books/front-end-handbook/2019/

Recap of Front-end Development in 2018:

„The reality of too much JavaScript change too fast is realized and people start talking about what you need to know before you can even learn something like React. The fight is real.“

PUSHING THE WEB FORWARD

Pushing the web forward currently means cramming in more copies of native functionality at breakneck speed — interesting stuff, mind you, but there's just too much of it.

Quick, name all the new features browsers shipped in [...]! You see? You can't. That's the problem.

We get ever more features that become ever more complex and need ever more polyfills and other tools to function — tools that are part of the problem, and not of the solution.

(Peter-Paul Koch 2015: Stop pushing the web forward)

PUSHING THE WEB FORWARD

- New web standards and APIs appear at an amazing rate
- Trying to catch up with native features?

> *Web Applications traditionally assume that the network is reachable. This assumption pervades the platform. HTML documents are loaded over HTTP and traditionally fetch all of their sub-resources via subsequent HTTP requests. This places web content at a disadvantage versus other technology stacks.*

Source: Service Workers, W3C Draft

PUSHING THE WEB FORWARD

- Apple is criticized for its slow adoption of new web standards
- Nolan Lawson: Safari is the new IE
- Or: Does Apple has a different vision of progress?

> *If you're a MacBook user, you're losing an average of 1 hour of total battery life by using Chrome. Firefox is a little better, but Safari is the clear winner. You'll want to use Safari if you want to get the most battery out of your laptop.*

(BatteryBox Blog, post no longer available)

BREAKING THE WEB FORWARD

> *On the other hand, we have no process for countering Google's reverse embrace, extend, and extinguish strategy, since a section of web devs will be enthusiastic about whatever the newest API is.*

(Peter-Paul Koch 2021: Breaking the web forward

> *So we end up with convoluted specs like Service Worker that you need a PhD to understand, and yet we still don't have a working*

(Nolan Lawson)

THINK ABOUT / DISCUSS ...

Peter-Paul Koch 2015: Stop pushing the web forward

Peter-Paul Koch 2021: Breaking the web forward

READING MATERIAL, SOURCES

GENERAL

- Front-End Developer Handbook 2019
  https://frontendmasters.com/books/front-end-handbook/2019/
- Chapter *Mobile Web* in: Mobile Developer's Guide To The Galaxy, Open XChange
  https://www.open-xchange.com/resources/mobile-developers-guide-to-the-galaxy/

MODERN JAVASCRIPT

- JavaScript. The Core: 2nd Edition
  dmitrysoshnikov.com/ecmascript/javascript-the-core-2nd-edition/
- Eloquent JavaScript, 3rd Edition
  eloquentjavascript.net
- Current ECMAScript Language Specification
  ecma-international.org/ecma-262/

WEB COMPONENTS

- Web Components: Introduction
  www.webcomponents.org/introduction
- CSS-Tricks: A Guide to Web Components (old API)
  css-tricks.com/modular-future-web-components/
- MDN: Web Components
  developer.mozilla.org/en-US/docs/Web/Web_Components
- W3C: Web Components Specifications
  github.com/w3c/webcomponents

HYBRID APPS AND PWAS

- Cordova
  cordova.apache.org
- MDN: Progressive Web Apps
  developer.mozilla.org/en-US/docs/Web/Progressive_web_apps
- Progressive Web Apps are here. What's the big deal?
  blog.mozilla.org/firefox/progressive-web-apps-whats-big-deal/
- 2018 State of Progressive Web Apps
  medium.com/progressive-web-apps/f7517d43ba70