

## Our Audience

Our audience with this tutorial is the “Hybrid Developer” - someone who has developed websites in the past, but also also knows one or more other languages. As Diane puts it:

“the best way to think of hybrid developers are that they were web developers pre-apple, android and facebook app popularity, then for many reasons, they decided to learn objective-c, java and learned how to make a successful facebook app.

hybrid devs are those devs that know both HTML and one or more of the other languages, so they definitely know how HTML, JS and CSS work together.

but they have not kept up with the latest HTML5 syntax etc, so they are not necessarily bleeding edge web devs anymore, because they've had to pick up other languages because that is what their client base demanded of them.”

### ***What we can assume they know***

- A possibly dated understanding of basic HTML, CSS, and JS, and what roles those technology play in a website.
- Basic information about app development for at least one other platform - iOS, Android, or Facebook.

## Learning Objectives

Learning objectives describe what you want the learner to *be able to do* after they finish the tutorial. They are phrased in terms of demonstrable action verbs. *Know* and *understand* are not demonstrable verbs, but *list*, *describe*, *identify*, etc. are all ways of demonstrating knowledge.

Learning objectives will be achieved via in-depth explanations with code samples, downloadable example code, diagrams, and links off to reference documentation. Each item here represents one thing we would like to ensure that the The “Further Reading” heading contains links to other sites for advanced topics.

### ***What’s an Open Web App?***

Objectives: Learners will be able to:

- Describe possible capabilities of Open Web Apps. (*app gallery!*)
- List key differences between OWA and Traditional Apps.
- Map technologies to the roles they play in an OWA.

Further Reading:

- Apps on MDN links.

### ***Design it!***

Objectives:

- Identify key distinguishing features of apps & how to prepare for these in code
- Incorporate existing OWA UI Patterns into their apps (*later: use our UI library in their code*)
- Tips on apps format different expectations & how to address these

#### Dev Questions:

- Q: How do I create app-esque experiences with web technologies?
  - A: Layout, etc: jQuery mobile, Pastry Kit
  - A: UI Patterns: Bootstrap, jQuery UI
  - A: Touch interaction: jGestures
  - A: Animations: CSS Animations
- Q: How do I create apps that work across multiple platforms?
  - A: Client-side: Responsive Design or Adaptive Design
  - A: Server-side: Device Detection and separate templates
  - A: Some combination of the two
- Q: How do I know which tactic to take?
  - A:

#### Further reading:

- <http://mobile-patterns.com/>

### **Code it!**

#### Objectives: Learners will be able to:

- Set up a local dev environment.
- Identify the purpose of the files in an example project. (*download the sample code + any bootstrap libraries we have*)
- Locate more information on specific technology topics
  - Recent HTML5 & Device APIs
  - CSS3
  - JS Libraries & Tools
- Code for different screen sizes and different interaction metaphors (touch vs. mouse).
  - Phone portrait
  - Phone landscape
  - Tablet portrait
  - Tablet landscape
  - Desktop

#### Further Reading:

- HTML, CSS, JS Reference
- Localization

### **Test it!**

#### Objectives: Learners will be able to:

- Select platforms to test on
- List criteria for determining target platforms.
- Run tests on at least one platform. (*on the sample code downloaded in step one, if possible - mention tools like LocalTunnel*)

- Create tests for:
  - Different Screen Sizes
  - Device APIs

Further Reading:

- Links to useful testing tools.

### ***Publish it!***

Objectives: Learners will be able to:

- List options for deploying an OWA.
- List options for publishing an OWA.
- Recognize steps for publishing an OWA on Mozilla Marketplace.

Further Reading:

- Mozilla Marketplace information
- Self-publishing API Reference Link

### ***Manage it***

Objectives: Learners will be able to:

- Describe how updates work
- Recognize the steps for doing updates.
- Identify ways to engage users.
- List ways to make money from your OWA.
  - Mozilla Marketplace
  - In-App Payment
  - Other monetization methods, third-party payment providers

Further Reading:

- Mozilla Marketplace Sign-Up Link
- Apps API Reference
- Forums? Anything like that?
- Our IRC Channels?

## **What's an Open Web App?**

### ***Examples***

Content: “Checkout what’s possible with web apps by installing one of these featured, free apps from the Mozilla Marketplace.”

- a curated list of links to free apps in the Marketplace (when it goes live)
- examples are currently scarce or have not been created by the team, will ask directly if there is a public repo that contains more than the single public app example ([MozillaBall](#))

## Compare to other platforms

**Content:** “Web apps offer developers the full flexibility and resources of the open web, and are capable of being installed on all of today’s major smartphone platforms.”

- iOS
  - primary language is Objective C
  - Apple enforces interaction and style requirements on developers - *with web apps, you have the same level of developer control and creativity the web offers*
  - Apple is the sole gatekeeper for any app that wants wide-scale distribution on its devices - *web app distribution is not chained to a single marketplace.*
  - iOS apps aren’t available to the majority smartphone owners - *web apps can be installed on any device*
  - Apple charges a fee to begin developing for their platform - *just checked: yeah, developing for the web is still free*
- Android
  - primary language is Java
  - Android requires you install a large SDK and an emulator in order to test your application - *web apps can be installed instantly in your browser*
  - Android lacks the controls and UI frameworks available on other platforms - *web apps can use any of the great JavaScript library UI frameworks in creating compelling user interfaces*

## Technologies Used

**Content:** “Unlike other platforms like iOS and Android, the first-tier languages of web apps are HTML, CSS, and JavaScript. You can choose the libraries and frameworks you are already familiar with, and if you need something new, just remember, there’s a world wide web of resources at your fingertips.”

## Code it!

### Bootstrap

- App Manifest: <https://developer.mozilla.org/en/Apps/Manifest>
  - Online manifest validator: <http://appmanifest.org/>
- App Cache: [https://developer.mozilla.org/en/Using\\_Application\\_Cache](https://developer.mozilla.org/en/Using_Application_Cache)
- Install your app: [https://developer.mozilla.org/en/Apps/Platform-specific\\_details](https://developer.mozilla.org/en/Apps/Platform-specific_details)

### HTML5 / CSS / JS

- HTML5 APIs
  - Use this for gathering richer content about major APIs we want to highlight: <http://www.netmagazine.com/features/developer-s-guide-html5-apis>
  - This is more a list of all modern web APIs, some are not actually from the HTML5 spec, but are closely related to app development: <http://platform.html5.org/>
- JavaScript
  - The language that has been taking the developer world by storm, now available for writing apps that run on any platform

- JavaScript language introduction: [https://developer.mozilla.org/en/JavaScript/About\\_JavaScript](https://developer.mozilla.org/en/JavaScript/About_JavaScript)
- JavaScript reference: <https://developer.mozilla.org/en/JavaScript/Reference>
- List of major JavaScript libraries/frameworks: <http://www.everyjs.com/>
- List of micro libraries for targeted use-case solutions: <http://microjs.com/>
- CSS
  - The standard declarative styling language of the web, CSS has rapidly evolved beyond simple layout shuffling and into the realm of animations, 3D transforms, and complex graphics.
  - CSS introduction: [https://developer.mozilla.org/en/CSS/Getting\\_Started](https://developer.mozilla.org/en/CSS/Getting_Started)
  - CSS properties reference: [https://developer.mozilla.org/en/CSS/CSS\\_Reference](https://developer.mozilla.org/en/CSS/CSS_Reference)
  - Advanced CSS Demos: <https://developer.mozilla.org/en-US/demos/tag/tech%3Acss3/>

## ***Going Mobile***

- An introduction to responsive design: <http://coding.smashingmagazine.com/2011/01/12/guidelines-for-responsive-web-design/>
- Responsive design techniques, tips, and tricks: <http://www.smashingmagazine.com/2011/07/22/responsive-web-design-techniques-tools-and-design-strategies/>
- Examples of well-executed responsive designs: <http://designmodo.com/responsive-design-examples/>

## ***Localization***

- Best practices: [https://developer.mozilla.org/en/Web\\_Localizability/Creating\\_localizable\\_web\\_applications](https://developer.mozilla.org/en/Web_Localizability/Creating_localizable_web_applications)