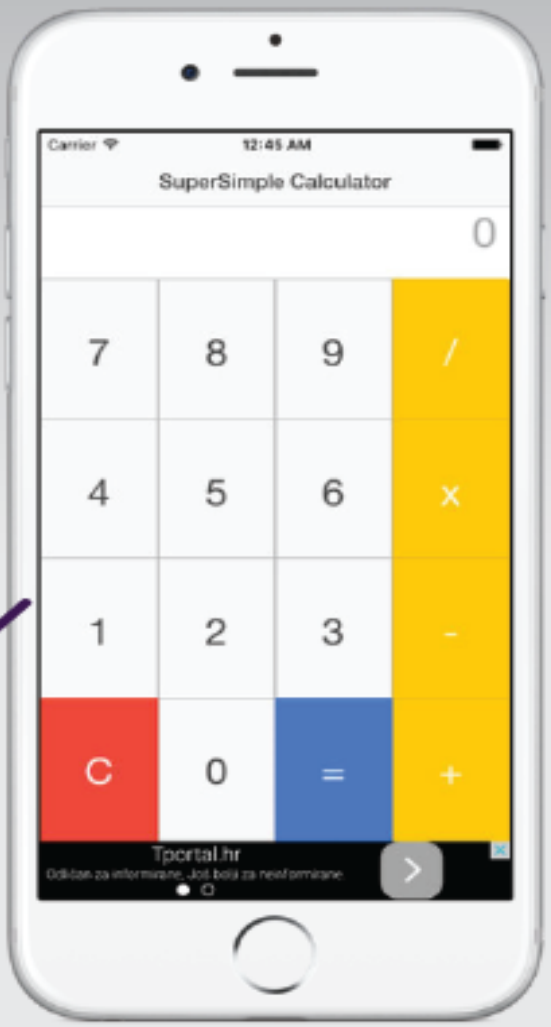
IONIC FRAMEWORK

ionic



Step by step from idea through prototyping to the app stores

Ionic framework

step by step from idea through prototyping to the app stores

Nikola Brežnjak

This book is for sale at http://leanpub.com/ionic-framework

This version was published on 2015-12-15

This is a Leanpub book. Leanpub empowers authors and publishers with the Lean Publishing process. Lean Publishing is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

Contents

How to get started with Ionic framework on Mac and Windows

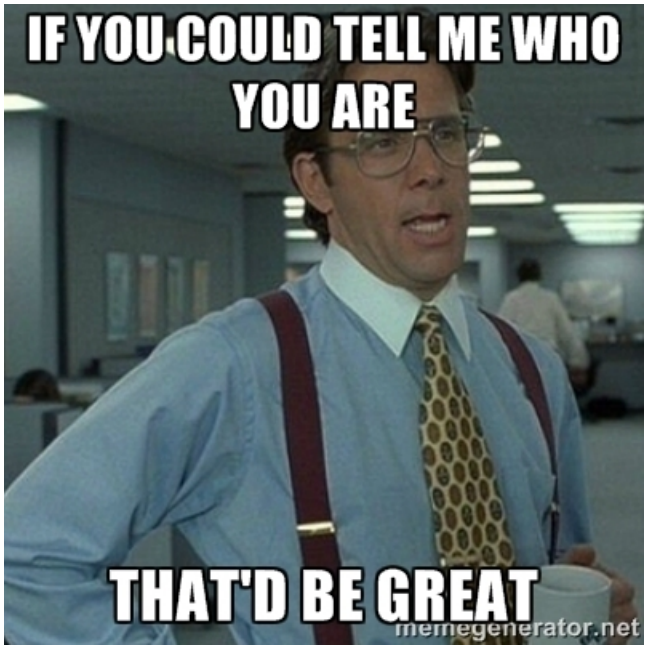
This is the first post in a series of posts which will teach you how to take advantage of your web development knowledge in building hybrid applications for iOS and Android.

This first post explains:

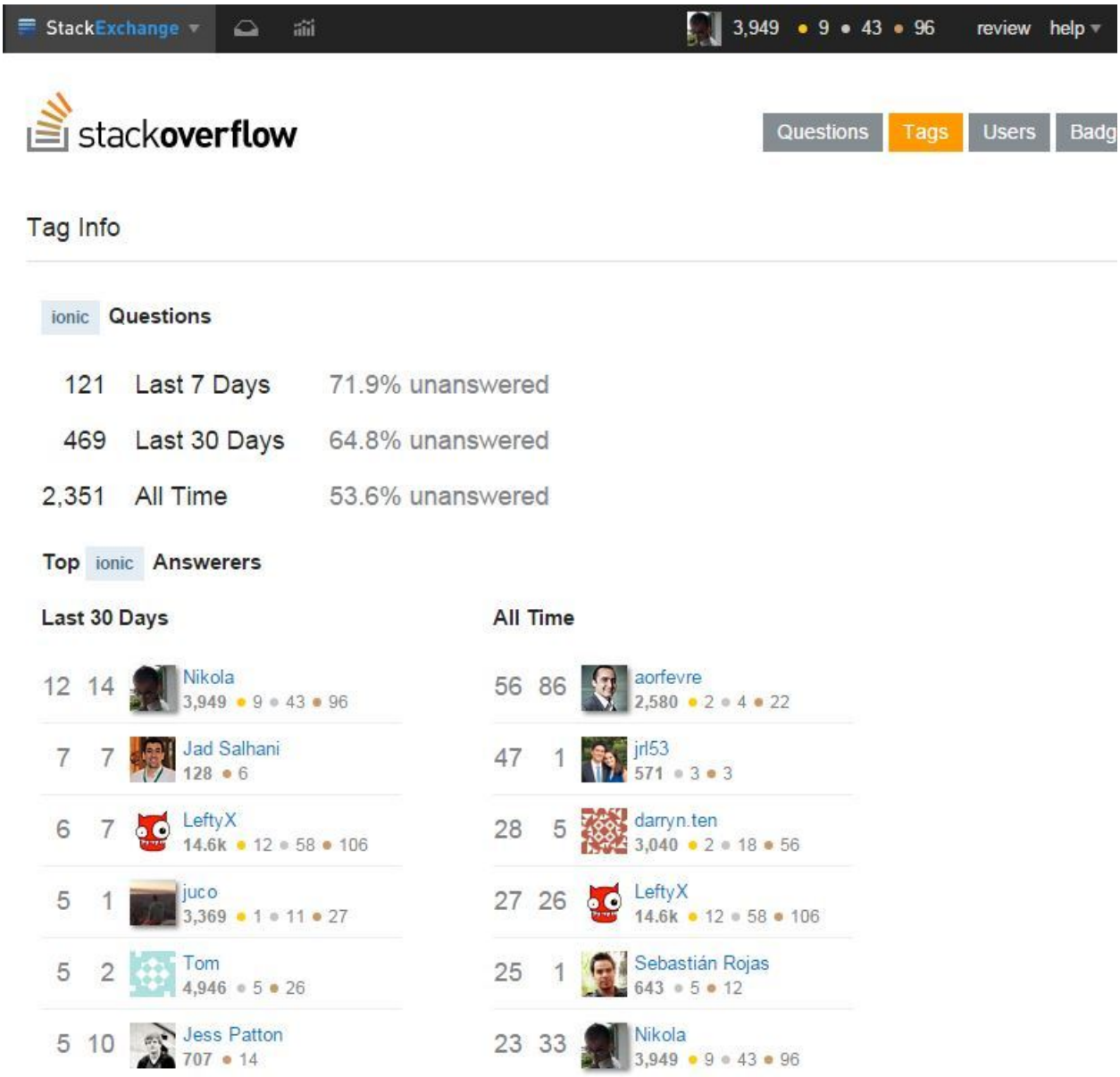
- How to make an app these days
- What actually is Ionic framework
- How to install Ionic on both Mac and Windows
- How to use Ionic CLI to start an Ionic project
- How to run an Ionic application

Why should you listen to me?

If you're like



Then let me share with you my current (YMMV[1] based on the time of your visit) StackOverflow profile[2] image:

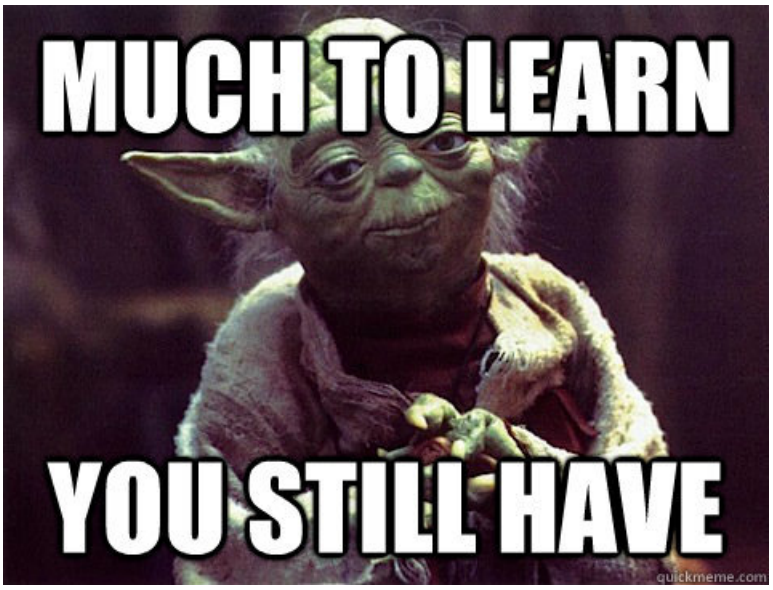


You can clearly see that I'm the top answerer in the `ionic` tag for the last month and in the top 10 answerers of all time. If you want to see what those answers actually are, you can take a look at my StackOverflow profile[3]. Also, I'm the guy behind the MEAN stack series[4] here on HackHands,

[1] http://www.urbandictionary.com/define.php?term=ymmv
[2] http://stackoverflow.com/users/534755/nikola?tab=profile
[3] http://stackoverflow.com/users/534755/nikola?tab=answers
[4] https://hackhands.com/how-to-get-started-on-the-mean-stack/

with a self-published (pay what you want) Getting MEAN with MEMEs[5] blog2book at Leanpub. If that's not enough, I'm currently a technical reviewer for the book "Getting started with Ionic" by PacktPub, and also a technical reviewer for the video "Rapid Ionic", also by PacktPub. Anyways, if nothing else, give it a shot with my unique "humorly" approach, and you just might like it and learn something.

Ok, enough about me, let's teach you some Ionic, since



Introduction

> 85% of mobile time is spent in apps

You're bombarded with reports all[6] over[7] the[8] web[9] that users tend to spend way more time on their

[5] https://leanpub.com/meantodo
[6] http://www.theguardian.com/technology/appsblog/2014/apr/02/apps-more-popular-than-the-mobile-web-data-shows
[7] http://www.smartinsights.com/mobile-marketing/mobile-marketing-analytics/mobile-marketing-statistics/
[8] http://venturebeat.com/2013/04/03/the-mobile-war-is-over-and-the-app-has-won-80-of-mobile-time-spent-in-apps/
[9] http://techcrunch.com/2015/06/22/consumers-spend-85-of-time-on-smartphones-in-apps-but-only-5-apps-see-heavy-use/

phones and especially in apps (rather than surfing the web using their phones) and you decided that **it's time to learn how to make an app**.

If you're a web developer, you have a decent knowledge of HTML, CSS, and JavaScript; also, you're most likely using one of the ever so slightly popular frameworks these days like AngularJS, Ember or React, just to name a few. If you're even proficient enough with the MEAN stack, you are a well-rounded full stack developer, and you basically have it all. Well, except the apps part, right?

> If you want to see what's all that fuss about the MEAN stack, you can check out the free four part tutorial series here on HackHands, starting with the first post on How to get started on the MEAN stack[10].

But, where to start with making an app? **Could you use some of your existing skills?** Up until fairly recently, if you wanted to make an app for (currently) two most popular mobile operating systems (iOS and Android) your only bet was to make the, so-called, **native application** by using the SDKs of the intended platforms.

This, of course, meant that you needed to make two versions of your application; one for iOS and one for Android. If you are a solo developer, chances that you're proficient in both are not so great.

Therefore, for some time, developers were opting for either iOS or Android, whereas big firms had two developing departments, one for each platform (worth mentioning Windows phone here as well).

Nowadays, luckily, with the Ionic Framework (and few others like PhoneGap[11], OnsenUI[12], Famo.us[13]) you can create one application by using the skills you, as a web developer, already have (others, don't fret - this isn't rocket science to be afraid of it) and then deploy this **one codebase** as an app to both iOS and Android stores. How cool is that, right? You can see the comparison review between the noted frameworks here[14].

[10] https://hackhands.com/how-to-get-started-on-the-mean-stack/
[11] http://phonegap.com/
[12] http://onsen.io/
[13] http://famous.org/
[14] https://www.airpair.com/ionic-framework/posts/hybrid-apps-ionic-famous-f7-onsen

Ways you can make an app these days

We've kind of touched all three in the Introduction section, but let's keep it nice and concise and list them here too. So, there are actually 3 ways that you can make an application for mobile devices these days:

- Native app
- Mobile website
- Hybrid app

Now, let's talk a bit more about the pros and cons of each of them.

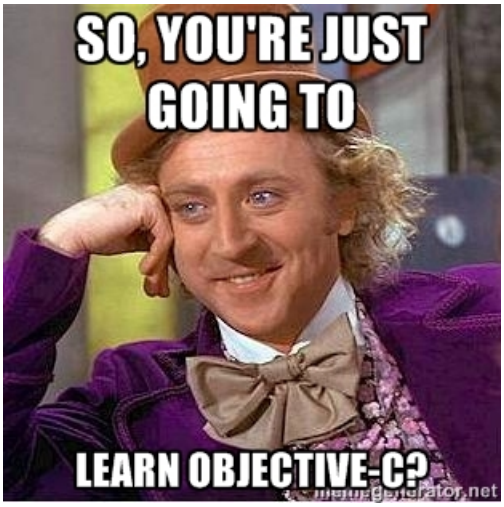
Native app

As mentioned previously, you can make an app specifically for iOS and Android by using their specific SDKs. If you want to build a native application for iOS you have to:

- have a Mac computer. *Sure, there are ways around it, but I honestly don't recommend them; for starters, a cheap Mac Mini will do just fine - at least it did so in my case*

- download Xcode[15] from the App Store (it's actually free)
- buy the Apple Developer license[16] that costs 99$ per year (if you want to publish to the App Store; and dooh!, you do)

You can write the apps by using the language Swift[17], or it's predecessor ObjectiveC[18].



If we're honest here, I think that Swift is a great step up from the clunky ObjectiveC, but that's just my own opinion (some people, of course, disagree[19]). Anyways, if you ever decide to go native just make sure you go with Swift as you'll get to know your way around it way sooner than with ObjectiveC, especially if you have a background in web development.

If you want to build a native application for Android you have to:

- have any computer
- download the appropriate SDKs (we'll cover this in the next section)
- buy the Google Developer license[20] which is 25$ per year

One of the pros of a native applications would be it's **speed** and direct access to a **native API** (you don't have to use any middleman wrappers, like in hybrid apps). A definite con of a native applications is that you need to build two (or more) applications, one for each desired platform.

[15] https://developer.apple.com/xcode/
[16] https://developer.apple.com/programs/
[17] https://developer.apple.com/swift/
[18] https://developer.apple.com/library/mac/documentation/Cocoa/Conceptual/ProgrammingWithObjectiveC/Introduction/Introduction.html
[19] http://www.infoworld.com/article/2968287/application-development/stop-the-funeral-apple-swift-versus-objective-c-alive-and-kicking.html
[20] https://play.google.com/apps/publish/signup/

Mobile website

Mobile website is actually a *"normal"* website (*yeah, don't go jumping because of the terminology; you're smart, you get the point*) that you visit with your browser on your phone, designed specifically to adapt to your phone's screen. As we've noted in the Introduction section, researches show that in todays world mobile websites tend to have a way lower engagement than they used to.

Developers used to make specific sites just for mobile browsers (on it's own domain; usually m.domain.com) but this proved to be hard to maintain. A practice called **responsive website design**[21] is used these days, where you basically have one HTML codebase, and you determine the look for specific devices (based on resolutions) by using the so-called **media queries**[22].

A great example of a mobile framework is jQuery mobile[23] that is soon coming out with its new 1.5 version, so we'll see if they bring something new to the table. From my personal experience with the framework from two years ago, I have all but good words for it; so, definitely use it if you're "only" making a mobile version of your web application.

A definite advantage of the mobile websites is that you can update them as you see fit, without waiting for the approval from Apple or Google. One of the disadvantages are definitely the fact that the mobile websites these days have way lower engagement than they used to, and that you can't basically use any of the additional phone features like for example camera or GPS.

Hybrid app

A hybrid app is a bassicaly a mobile application, written with the same languages that you use when building websites, with the addition that it contains an isolated browser instance, called **WebView**, which runs this web application inside of a native app. Hybrid apps can access the mobile device and use the additional phone features like for example camera or GPS.

Definite advantage of the hybrid apps is the fact that you can access the additional phone features via plugins and that you can do all the development with the same set of skills as you use when developing *"normal"* web applications. One of the disadvantages is the fact that, even though it's improving, the so-called Web View has it's limitations in terms of speed. So, it might not be best suited if you're on a quest to make the next best game with full blown 3D graphics.

What is Ionic and why it's so good

As I gave an answer on StackOverflow[24]:

[21] https://developers.google.com/web/fundamentals/layouts/rwd-fundamentals/

[22] https://developer.mozilla.org/en-US/docs/Web/Guide/CSS/Media_queries

[23] https://jquerymobile.com/

[24] http://stackoverflow.com/questions/31179211/use-ionic-or-cordova/31180666

Disclaimer: This will sound like advertisement, so I have to say I'm in no way affiliated with Ionic, I just happen to like it so much that I'm sharing the love for it.

Ionic is so much more than "just" an UI framework. Ionic allows you to:

- generate icons and splash screens for all devices and device sizes with a single command: `ionic resources`. This alone saves you at least a day of image preparing for various sizes.
- instantly update your apps with code changes, even when running directly on your device with `ionic run --livereload`
- build and test iOS and Android versions side-by-side and see changes instantly with `ionic serve --lab`
- share your Ionic apps with clients, customers, and testers all around the world without ever going through the App Store with `ionic share`
- easily access the full native functionality of the device using ngCordova (here you get to use any Cordova plugin) Also, Drifty[25] (the team behind the Ionic framework) is building a full-stack backend services and tools for your Ionic app like Deploy (for deploying a new version without going through Apple review process! - **this is huge!**), Analytics[26], Push notifications[27]. Ionic CLI (command line interface) uses Cordova in the backend and allows you to build (directly using Ionic CLI) apps for iOS and Android (just by doing `ionic build ios` or `ionic build android`). Ionic uses Angular as a frontend framework so if you're familiar with it, it will come as a bonus. They're working closely with the Angular 2.0 team[28] too. All in all, I personally think Ionic framework has a bright future, so if nothing else – give it a try I bet you'll like the ease of making an app with it.

Do I have your attention now? Great, let's install all the needed prerequisites in the next section and start using Ionic!

Installing prerequisites for both Mac and Windows

We need to have **Node.js** and **Git** installed in order to install both Ionic and Cordova. If you already have (and if you're a web developer chances are that you do) these tools installed, you can skip this section, and go straight to installing Ionic.

Installing Node.js

In order to download Node.js, visit http://nodejs.org/download/[29], where you'll see the following options:

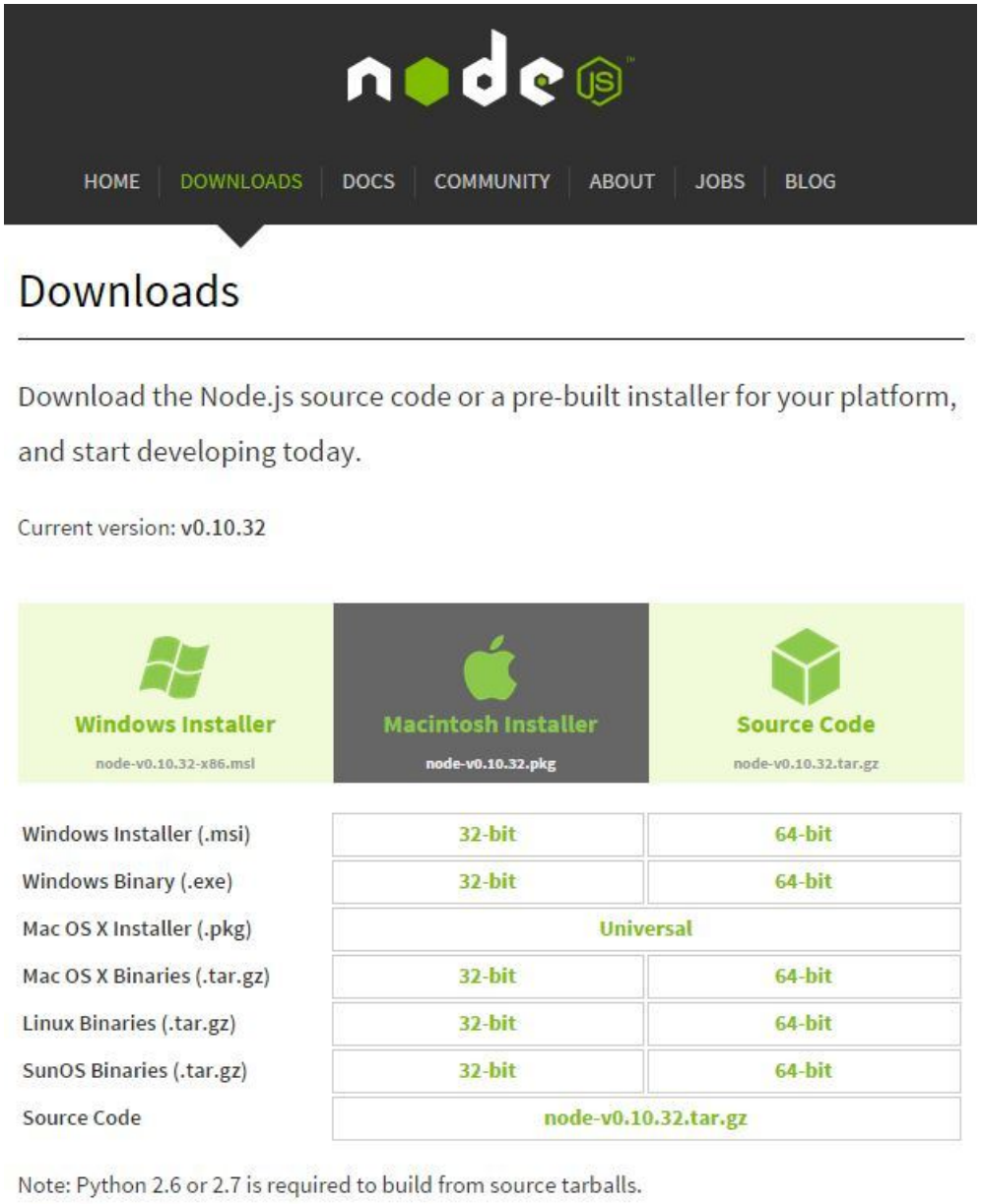
[25] http://drifty.com/

[26] http://www.nikola-breznjak.com/blog/ionic/ionic-analytics-alpha-lets-you/

[27] http://blog.ionic.io/announcing-ionic-push-alpha/

[28] http://blog.ionic.io/angular-2-ionic/

[29] http://nodejs.org/download/



Installation on Windows and Mac OS is simple as you just have to download and run the appropriate installer and follow the familiar instructions (next, next, next, sure I accept, next, finish).

If you have `brew` on your Mac then you can also install Node.js with:

```
brew install node
```

In both cases, `npm` (Node Package Manager - used to install other packages) will be installed along with Node.js.

To verify that you installed Node.js correctly on a **Windows** machine, run the following command

in your Command prompt (or, even better, use Console 2[30]):

```
node -v
```

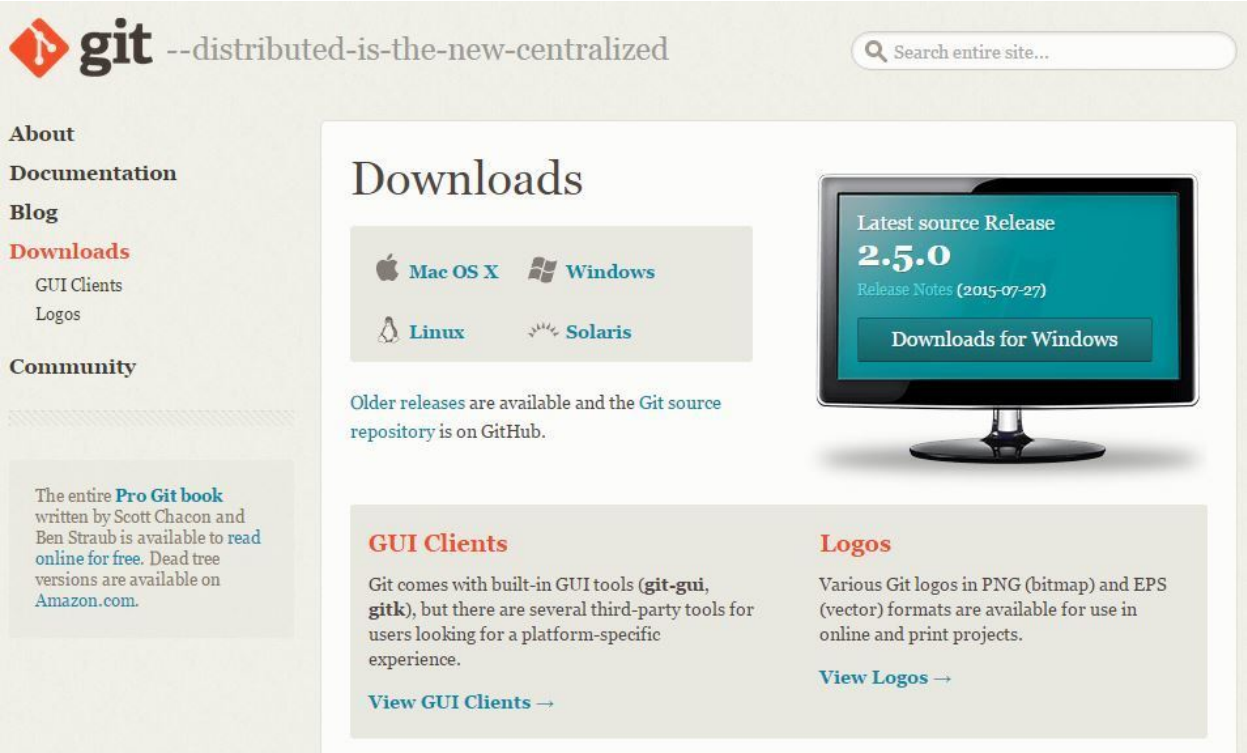
You should get an output similar to:

```
v0.12.7
```

To verify that you installed Node.js correctly on your **Mac**, run the same command as above in your Terminal (or, even better, use iTerm[31]), and you should get the same similar output as above.

Installing Git

In order to install Git, visit http://git-scm.com/download[32], where you'll see the following options:



Installation on Windows and Mac OS is as simple as for Node.js as you just have to download and run the appropriate installer and follow the, yet again, familiar instructions.

To verify that you installed Git correctly on your Windows/Mac machine, run the following command in your Command prompt/Terminal:

```
git
```

You should get an output similar to:

[30] http://www.nikola-breznjak.com/blog/quick-tips/customize-console-2-on-windows-machine/
[31] https://www.iterm2.com/
[32] http://git-scm.com/download

```
usage: git [--version] [--help] [-C <path>] [-c name=value]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p|--paginate|--no-pager] [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           <command> [<args>]

The most commonly used git commands are:
   add        Add file contents to the index
   bisect     Find by binary search the change that introduced a bug
   branch     List, create, or delete branches
   checkout   Checkout a branch or paths to the working tree
   clone      Clone a repository into a new directory
   commit     Record changes to the repository
   diff       Show changes between commits, commit and working tree, etc
   fetch      Download objects and refs from another repository
   grep       Print lines matching a pattern
   init       Create an empty Git repository or reinitialize an existing one
   log        Show commit logs
   merge      Join two or more development histories together
   mv         Move or rename a file, a directory, or a symlink
   pull       Fetch from and integrate with another repository or a local branch
   push       Update remote refs along with associated objects
   rebase     Forward-port local commits to the updated upstream head
   reset      Reset current HEAD to the specified state
   rm         Remove files from the working tree and from the index
   show       Show various types of objects
   status     Show the working tree status
   tag        Create, list, delete or verify a tag object signed with GPG

'git help -a' and 'git help -g' list available subcommands and some
concept guides. See 'git help <command>' or 'git help <concept>'
to read about a specific subcommand or concept.
```

You don't have to worry if you don't know how to use Git since you won't be needing it directly (npm uses it download packages). However, if you would like to learn (and there's no reason why you shouldn't expand your knowledge), you can check out this good interactive tutorial[33] made by CodeSchool.

[33] https://try.github.io/levels/1/challenges/1

Installing Ionic

If you've installed the needed prerequisites from the previous section, then in order to install Ionic (both on Mac and Windows) you just have to run the following command:

```
npm install ionic cordova -g
```

Ionic uses Cordova in the background, thus the need for it. We're using the **-g** flag here, in order to install the packages **ionic** and **cordova** globally[34].

To verify that you've installed Ionic correctly on your Windows/Mac machine, run the following command in your Command prompt/Terminal:

```
ionic -v
```

You should get an output similar to:

```
1.6.4
```

To verify that you've installed Cordova correctly on your Windows/Mac machine, run the following command in your Command prompt/Terminal:

```
cordova -v
```

You should get an output similar to:

```
4.2.0
```

If you had Node.js installed before (and haven't used it much since), you may get a notification like this:

```
1 ***************************************************
2  Upgrade warning - for the CLI to run correctly,
3  it is highly suggested to upgrade the following:
4
5  Please update your Node runtime to version  >=0.12.x
6 ***************************************************
```

To resolve this issue, just update Node.js; either by re-downloading it (on Windows) or by executing the following command in the Terminal (on a Mac):

```
brew update; brew upgrade node
```

> I would like to take a moment here and encourage you that you post a comment about any problems you may have (and from my experience from StackOverflow there tends to be quite of them) with these (seemingly simple) installation tasks. There always tends to be some older version of Node.js installed, or problems with cache and npm, blah blah blah... I will do my best to try to resolve your issue.

[34] https://docs.npmjs.com/getting-started/installing-npm-packages-globally

Using Ionic CLI

If you run `ionic` in your Terminal/Command prompt you will get an output similar to the one below:

```
> ionic
  _             _
 (_)           (_)
  _  ___  _ __  _  ___
 | |/ _ \| '_ \| |/ __|
 | | (_) | | | | | (__
 |_|\___/|_| |_|_|\___|  CLI v1.6.4

Usage: ionic task args

=======================

Available tasks: (use --help or -h for more info)

   start  ..........  Starts a new Ionic project in the specified PATH
   serve  ..........  Start a local development server for app dev/testing
   platform  .......  Add platform target for building an Ionic app
   run  ............  Run an Ionic project on a connected device
   emulate  ........  Emulate an Ionic project on a simulator or emulator
   build  ..........  Locally build an Ionic project for a given platform
   plugin  .........  Add a Cordova plugin
   resources  ......  Automatically create icon and splash screen resources (bet\
a)
            Put your images in the ./resources directory, named splash or icon.
            Accepted file types are .png, .ai, and .psd.
            Icons should be 192x192 px without rounded corners.
            Splashscreens should be 2208x2208 px, with the image centered in the\
 middle.

   upload  .........  Upload an app to your Ionic account
   share  ..........  Share an app with a client, co-worker, friend, or customer
   lib  ............  Gets Ionic library version or updates the Ionic library
   setup  ..........  Configure the project with a build tool (beta)
   io  .............  Integrate your app with the ionic.io platform services (al\
pha)
   push  ...........  Upload APNS and GCM credentials to Ionic Push (alpha)
   config  .........  Set configuration variables for your ionic app (alpha)
   browser  ........  Add another browser for a platform (beta)
```

```
    service  ........  Add an Ionic service package and install any required plug\
ins
    add  ............  Add an Ion, bower component, or addon to the project
    remove  .........  Remove an Ion, bower component, or addon from the project
    list  ...........  List Ions, bower components, or addons in the project
    ions  ...........  List available ions to add to your project
    templates  ......  List available Ionic starter templates
    info  ...........  List information about the users runtime environment
    help  ...........  Provides help for a certain command
    link  ...........  Sets your Ionic App ID for your project
    hooks  ..........  Manage your Ionic Cordova hooks
    state  ..........  Saves or restores state of your Ionic Application using th\
e package.json file
    docs  ...........  Opens up the documentation for Ionic
```

What you got is actually a nice summary of all the commands that you can run using the ionic CLI, along with their short descriptions. *Btw, if you're wondering what this CLI thing is (and you haven't Googled it yet[35]); it's actually an acronym from Command Line Interface, and in Ionic terms it's actually a tool that makes it easier to start, build, run, and emulate, (and a lot more), Ionic apps[36].*

In the following chapters we will cover most of these commands, but for now let's not burden you too much, and let's do a quick skin deep dive by making a simple project using `ionic start` command.

Starting a project with Ionic by using the existing templates

Cool, you've made it so far - I promise, you're going to see some code now!

Ionic CLI allows us to start and initialize your project by using the aforementioned `ionic start` command. If you take a look at the official documentation for the start command[37] you will see something like the following definition:

```
ionic start appname [template]
```

If you just run `ionic start appname` the Ionic CLI will make a bootstrap application with all the needed parts in the **appname** folder, with the so-called **blank** template.

There are three *named template starters*: **blank**, **sidemenu**, and **tabs**.

Additionally, you can use Github repo starters and Codepen URL starters. For a comprehensive list of (currently 20) starter apps check out the post from my friend Dragan over at Gajotres[38].

[35] https://en.wikipedia.org/wiki/Command-line_interface

[36] http://ionicframework.com/docs/cli/

[37] http://ionicframework.com/docs/cli/start.html

[38] http://www.gajotres.net/a-comprehensive-list-of-ionic-starter-apps/

In our example, we will use the **sidemenu** template, so execute the following command from your Terminal/Command prompt:

```
ionic start Ionic_1stTutorial sidemenu
```

You should see something similar to the following output:

```
C:\Users\Nikola\Desktop\IonicTesting>ionic start Ionic_1stTutorial sidemenu
Creating Ionic app in folder C:\Users\Nikola\Desktop\IonicTesting\Ionic_1stTutor\
ial based on sidemenu project
Downloading: https://github.com/driftyco/ionic-app-base/archive/master.zip
[============================]  100%  0.0s
Downloading: https://github.com/driftyco/ionic-starter-sidemenu/archive/master.z\
ip
[============================]  100%  0.0s
Updated the hooks directory to have execute permissions
Update Config.xml
Initializing cordova project

Your Ionic project is ready to go! Some quick tips:
 * cd into your project: $ cd Ionic_1stTutorial
 * Setup this project to use Sass: ionic setup sass
 * Develop in the browser with live reload: ionic serve
 * Add a platform (ios or Android): ionic platform add ios [android]
   Note: iOS development requires OS X currently
   See the Android Platform Guide for full Android installation instructions:
   https://cordova.apache.org/docs/en/edge/guide_platforms_android_index.md.html
 * Build your app: ionic build <PLATFORM>
 * Simulate your app: ionic emulate <PLATFORM>
 * Run your app on a device: ionic run <PLATFORM>
 * Package an app using Ionic package service: ionic package <MODE> <PLATFORM>

For more help use ionic --help or ionic docs
Visit the Ionic docs: http://ionicframework.com/docs

New! Add push notifications to your Ionic app with Ionic Push (alpha)!
https://apps.ionic.io/signup
+-----------------------------------------------------------+
+ New Ionic Updates for August 2015
+
+ The View App just landed. Preview your apps on any device
+ http://view.ionic.io
+
+ Invite anyone to preview and test your app
```

```
+ ionic share EMAIL
+
+ Generate splash screens and icons with ionic resource
+ http://ionicframework.com/blog/automating-icons-and-splash-screens/
+-----------------------------------------------------------+
```

Running the Ionic application

Now that we've initialized our Ionic application based on the **sidemenu** template, we have to run it in order to see what Ionic CLI generated for us.

First, change the directory to the name of the application you gave in the `ionic start` command. In our case, that is **Ionic_1stTutorial**:

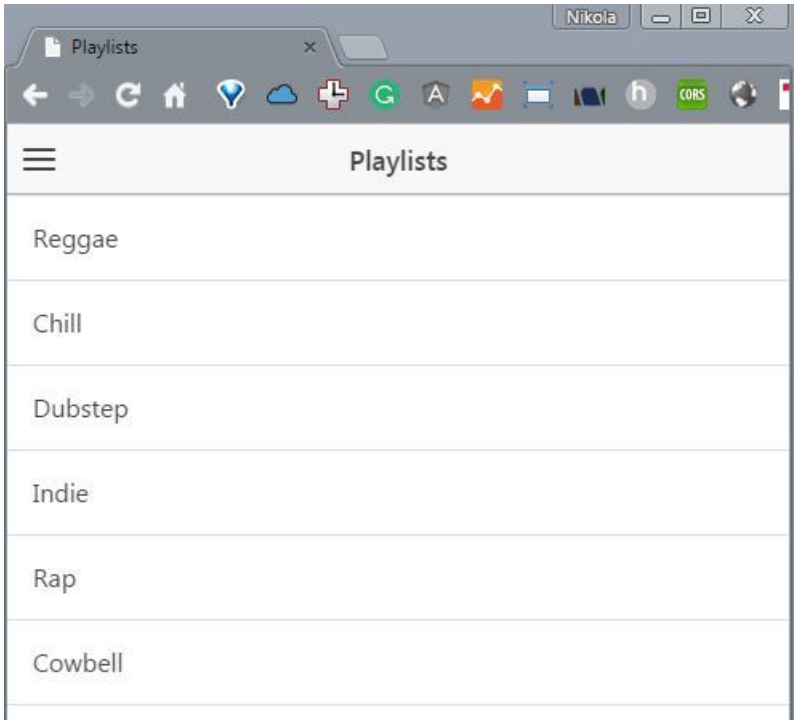
```
C:\Users\Nikola\Desktop\IonicTesting\Ionic_1stTutorial>```

If you open up the project in your editor (I use [Sublime Text 3](http://www.sublimetext.com/3)) you will see the following folder structure:

![](images/vKpZhByl.jpg)

In the following chapters, we will spend most of the time in the **www** folder.

> Since Ionic is based on [AngularJS](https://angularjs.org/) framework, you will need at least a basic understanding of it, and you can start exploring it with a [free interactive tutorial](http://angular.codeschool.com/) by CodeSchool. Also, you can take a look at my [fourth tutorial in the MEAN stack](https://hackhands.com/finishing-Angular-TODO-application-deploying-production/) series, which is all about AngularJS.

There are few ways in which you can get your Ionic application running:

+ `ionic serve` - starts the app in a local web browser
+ `ionic emulate android` - starts the app in an emulator (in this example android is used; you can also use **ios** if you're on a Mac and have all the prerequisites installed)
+ `ionic run android` - starts the app on the actual device that's plugged into your computer
+ `ionic build android` - creates an **.apk** file which you can physically copy to your Android device and run it (this scenario doesn't work for iOS devices in normal circumstances; you have to go through Xcode, as we'll describe in detail in the next chapter)
```

```
29
30 So, now run the following command in your Terminal/Command prompt:
31
32 `ionic serve`
33
34 You should see the following similar output:
```

C:UsersNikolaDesktopIonicTestingIonic_1stTutorial>ionic serve Running live reload server: http://localhost:35729 Running dev server: http://localhost:8100 Ionic server commands, enter: restart or r to restart the client app from the root goto or g and a url to have the app navigate to the given url consolelogs or c to enable/disable console log output serverlogs or s to enable/disable server log output quit or q to shutdown the server and exit

ionic $ “‘

Also, you should get your local browser started up automatically, pointing to the address **http://localhost:8100/#/app** with an output similar to the one on the image below (I resized the window for clarity - *if you’re using Chrome[39], you can get the use the Window Resizer plugin[40], or use the Chrome Dev Tools Emulate feature[41]*):



[39] https://www.google.com/chrome/

[40] https://chrome.google.com/webstore/detail/window-resizer/kkelicaakdanhinjdeammmilcgefonfh

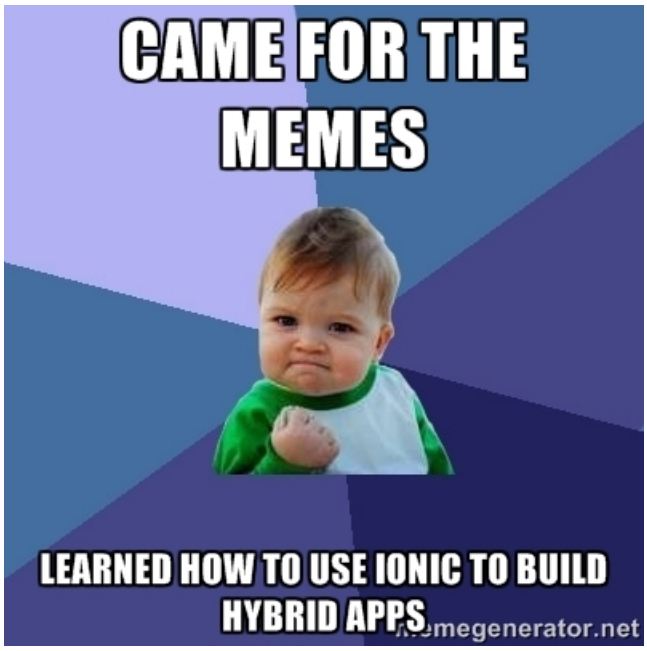
[41] https://developer.chrome.com/devtools/docs/device-mode

Awesome thing about this is that you have automatically set up **live reload** feature, which means that as soon as you change the code in your **www** folder, the application will reload automatically so that you don't have to keep pressing the F5 (+R) key on your Windows (Mac) machine.

If you like, you can get this project on Github[42].

Conclusion

In this chapter we've gone through the options that you have in making an app these days. Then we explained what actually is Ionic framework and how to install it on both Mac and Windows. With the use of the Ionic CLI we started a project based on the *sidemenu* template, and finally we ran it locally in the browser with the use of `ionic serve` command. In the next chapter I'll show you how to create your own calculator application by making use of Ionic Creator.



[42] https://github.com/Hitman666/Ionic_1stTutorial