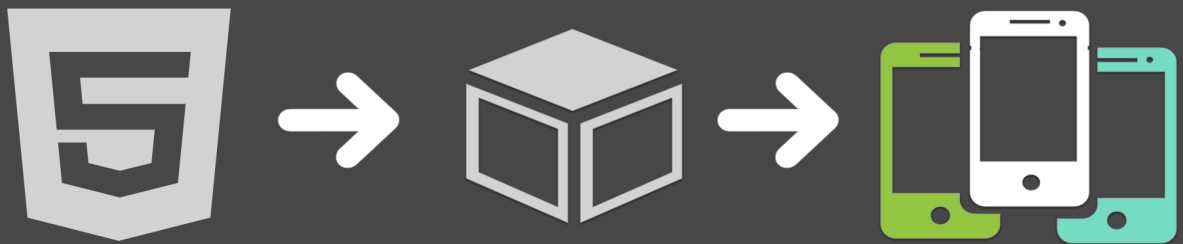


PhoneGap Build

Starter



Painless Mobile Apps
Development

Zainul Setyo Pamungkas

PhoneGap Build Starter

Painless Mobile Apps Development

Zainul Setyo Pamungkas

This book is for sale at <http://leanpub.com/phonegapbuild>

This version was published on 2015-05-26



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.

©2014 - 2015 Zainul Setyo Pamungkas

For Almighty Allah, Prophet Muhammad SAW, My Beloved Parents

Contents

So, what is PhoneGap Build?	1
Step by Step	3
Prepare Development Environment	3
Import Signing Keys	3
Preparing Assets	5
Configuring Application	5
Zip Upload and Git Deployment	8

So, what is PhoneGap Build?

PhoneGap Build is a cloud service that allow developers to compile HTML5 application into native mobile application. It compile not only for Android and iOS, but also for WebOS, Windows Mobile, Symbian, and Bada. It designed to make mobile application development faster and easier by using existing technology, HTML5. It works like magic. We don't have to know how application compilation done in server. All we have to do is just uploading assets file and we will get market-ready application in different platforms.

PhoneGap Build let us develop for several platforms without even setting up SDK. We don't need to setup ADT and Eclipse for developing Android application, or even download and setup Xcode for developing iOS application. It take care the work in the cloud, we don't need to worry about setting up and maintaining native SDKs. It build applications using latest SDK for targeted platforms. We don't have to manually install and set up new version of SDKs.

PhoneGap Build makes HTML5 application really shine. HTML5 has good cross compatibility across mobile platforms already. And Phonegap Build leverage its capabilities in to the next level by making it native applications that can be installed on real devices. It can be done even without any changes to support multiple platforms using single code base.

PhoneGap Build makes mobile application development cost-effective. Reducing cost for each different platform that want to be supported. Developers can also rapidly prototype app using existing tools to develop web application. They don't need to learn another programming language, juts use existing skill from web development.

PhoneGap Build support 'classic upload' and Git (GitHub) for deploying application. Classic upload can be done by uploading web assets in *.zip file. While Git deployment provide us the power of version control, especially using GitHub service. Developers can commit changes to GitHub repositories then pull it from PhoneGap Build to be compiled. This deployment method will significantly increase the speed of application development. One of GitHub power's that can be used is collaboration development. This is great if developers are working on team.

PhoneGap Build gives extra capabilities on application by using plugins. Plugins are used to extend native functionality exposed by the PhoneGap native-app container. They are including child browser, generic push (push notification), barcode scanner, analytics, and FacebookConnect. Developers can include original PhoneGap's plugin to use with Phonegap Build and extend native capabilities of application.

PhoneGap Build allow developers to configure application through single file named `config.xml`. It expand options that can be set up to build application using PhoneGap Build. They are including describing application, configuring device orientation, setting up specific icon to match targeted platform, configuring splash screen, and many more will be described in next section.

Currently, PhoneGap Build offer two different plans, free and paid. Both plan are open source friendly. Which mean we have no limitation to create open source application and how many collaborators are collaborating in the project. The only difference is just how many private application you can build. Free plan allow developers to create only one private application, while paid plan allow 25 applications to be created in private.

Step by Step

In 5 easy steps, you can use PhoneGap Build and get it set up.

Prepare Development Environment

Before using PhoneGap Build service, you will need to check that you have all of the required elements, as listed below:

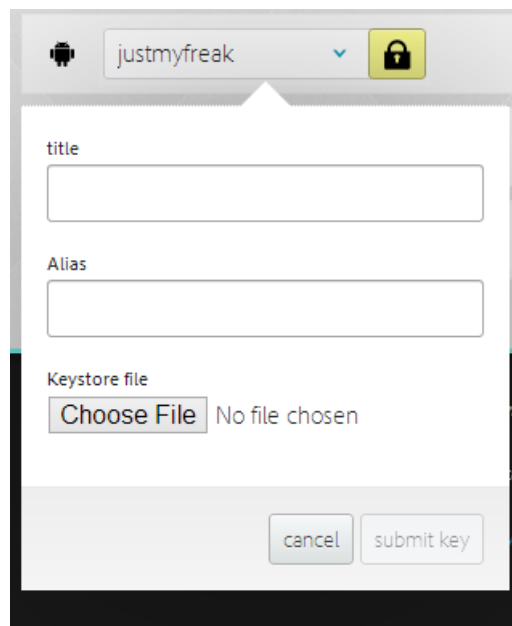
- PC or Mac. This is needed to get certificate for signing application.
- Text editor : Preferably with xml highlighting such as Notepad++ or Sublime Text. You will need this to write `config.xml`.
- Register apple developer program. You will need to enroll on it if you plan to build for iOS devices.
- Google Chrome or other browser. need web browser to use PhoneGap Build.
- GitHub account. You will need it if you want to deploy via Git

Import Signing Keys

Importing signing keys is very important especially if you plan to deploy and publish application on iOS, Android, and Blackberry. For testing purpose, Android and Blackberry require no sign key. But for iOS, you will need to have signing for both development and publishing application.

For Android signing, we first need to generate a signing keystore file. Full details are available in <http://developer.android.com/guide/publishing/app-signing.html>¹. Please record alias as well as keystore password and key password that you set for your keystore. Then go to your account, click add key, and fill all your details.

¹<http://developer.android.com/guide/publishing/app-signing.html>



Importing Signing Key

For Blackberry signing, we first need to register on RIM's site at <https://www.blackberry.com/SignedKeys/>², then follow the installation process. Once installed, we must find SDK directory when we installed key.

- 1 C:\SDK_PATH\bin\signtool.csk
- 2 C:\SDK_PATH\bin\signtool.db

Both files are required to sign and build Blackberry application. `signtool.csk` contain database private key and salt and `signtool.db` contain credentials for connecting on RIM's server. Then we need to add both file in BlackBerry key section like we did on Android.

For iOS signing, we have to register Apple's developer program on <http://developer.apple.com>³ and pay \$99/year for enrolling iOS developer program. We need two files `*.p12` and `*.mobileprovision` to sign iOS application. There are two ways to retrieve those 2 files, first is using standard Apple method documented on <http://developer.apple.com>⁴ and second is using Windows http://community.phonegap.com/nitobi/topics/detailed_guide_for_setting_up_building_ios_apps_without_a_mac⁵

Once we get `*.p12` and `*.mobileprovision`, we can add it to PhoneGap Build sign like we did on Android. Please note that there is 2 version of `*.p12` certificate, first is for development and second is for production (distribute on App Store). For testing purpose, we can use development certificate and add our iDevice's UUID in `*.mobileprovision`.

²<https://www.blackberry.com/SignedKeys/>

³<http://developer.apple.com>

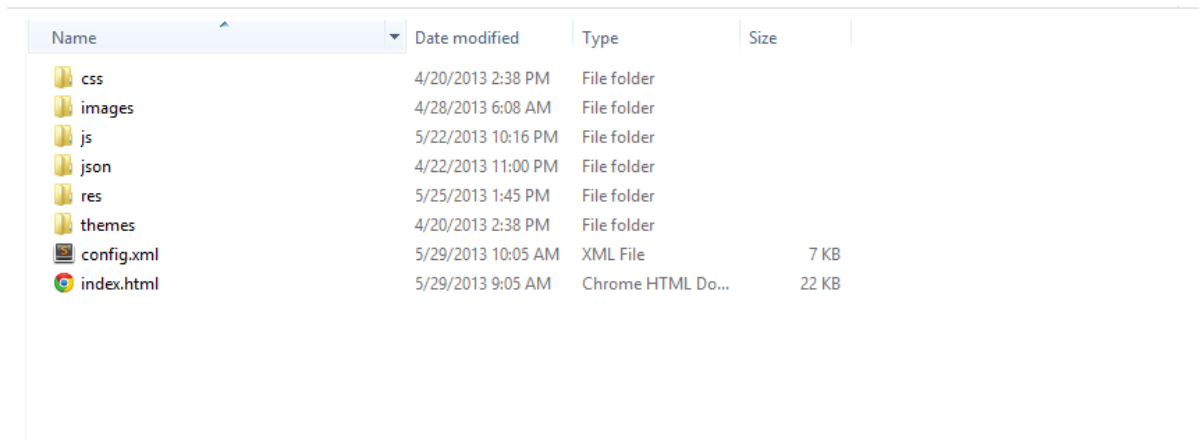
⁴<http://developer.apple.com>

⁵http://community.phonegap.com/nitobi/topics/detailed_guide_for_setting_up_building_ios_apps_without_a_mac

Preparing Assets

Assets are HTML5 application including html, js, css, and image resources. Actually, Phonegap Build need only one `index.html` and place other resources on servers. But it will not wise for mobile application. Static files such as html, js, css, images, or even json data should be cached offline. It will increase application performance and load time as well as reducing internet connection in application.

When creating application for PhoneGap Build, you have to leave `index.html` at root level of your project directory. Make sure there is no other `*.html` file other than index on root level. This will cause build failed on certain Phonegap version. If you want to add another html file, make sure you add it inside your folder. Here is suggested project folder of Phonegap Build application :



Name	Date modified	Type	Size
css	4/20/2013 2:38 PM	File folder	
images	4/28/2013 6:08 AM	File folder	
js	5/22/2013 10:16 PM	File folder	
json	4/22/2013 11:00 PM	File folder	
res	5/25/2013 1:45 PM	File folder	
themes	4/20/2013 2:38 PM	File folder	
config.xml	5/29/2013 10:05 AM	XML File	7 KB
index.html	5/29/2013 9:05 AM	Chrome HTML Do...	22 KB

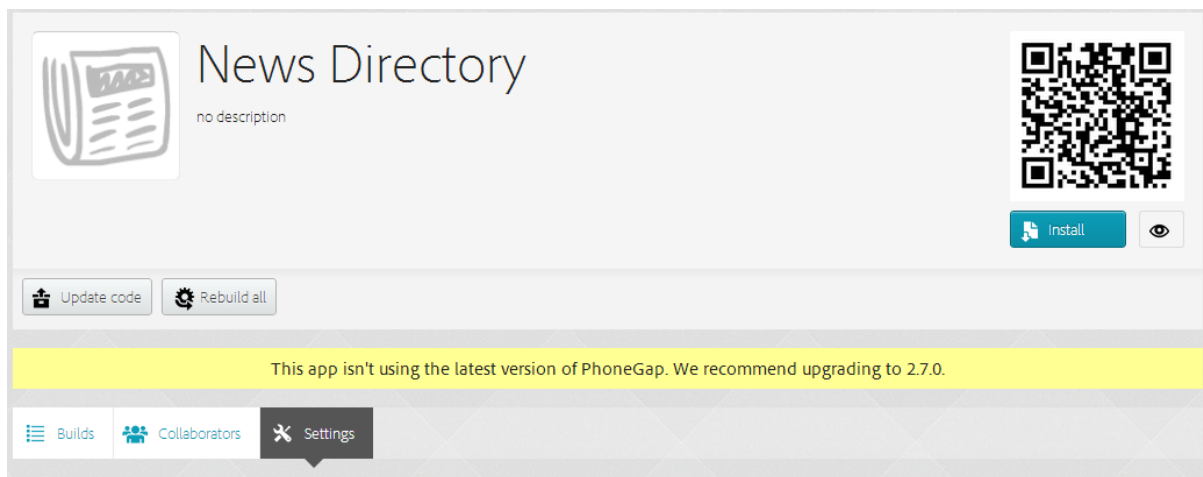
Project Directory Example

Configuring Application

PhoneGap Build allows us to make configuration for building application. Application can be configured through web interface inside PhoneGap Build account or by using `config.xml`. The `config.xml` file is specified using W3C widget specification (<http://www.w3.org/TR/widgets/>⁶), which allows developer to specify certain configuration easily through single xml file.

Web configuration can be found under Settings tab of application. By using web configuration we can specify application's icon, name, package, version number, PhoneGap version, as well as its description. Package name must be reverse-domain style (e.g. `com.yourcompany.yourapp`). As for version number, you can use major/minor/patch style version such as `1.0.1`

⁶<http://www.w3.org/TR/widgets/>



Another way to configuring application is by using `config.xml` file. This file should be placed under root level of application, same location with `index.html`. The `config.xml` gives us more than we can do in web configuration. Below is example of initial `config.xml`.

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <widget xmlns      = "http://www.w3.org/ns/widgets"
3        xmlns:gap    = "http://phonegap.com/ns/1.0"
4        id            = "com.yourcompany.yourapp"
5        version       = "1.0.0">
6
7        <name>yourapp</name>
8        <description>Your application description</description>
9  </widget>

```

Mandatory Properties

- `<widget>` : The `<widget>` element must be root of your `config.xml` document. Attributes. It contain following attributes :
 - `xmlns.XML's xmlns` with value `http://www.w3.org/ns/widgets`
 - `xmlns:gap` with value `http://phonegap.com/ns/1.0`
 - `id`. Application package name.
 - `version`. Application version number
- `<name>` : The name of application
- `<description>` : The description of application

Along with mandatory properties, we can add configuration for :

1. Application Icon

Application icon can be configured by using `<icon>` element. You can have zero or more of this element. If you don't add `<icon>` element, PhoneGap Build will use default PhoneGap Build icon. The default icon must be named `icon.png` and placed in root of your application.

```
1  `<icon src="icon.png"/>`
```

For iOS icon, PhoneGap Build support classic, retina, and iPad (and retina iPad starting from PhoneGap 2.5.0). The following will define icon for specific screen type of iDevice.

```
1  <icon src="icons/ios/icon.png" gap:platform="ios" width="57" height="57" />
2  <icon src="icons/ios/icon-72.png" gap:platform="ios" width="72" height="72" />
3  <icon src="icons/ios/icon_at_2x.png" gap:platform="ios" width="114" height="114\"
4  " />
5
6  <!-- retina iPad support: PhoneGap 2.5.0+ only -->
7  <icon src="icons/ios/icon-72_at_2x.png" gap:platform="ios" width="144" height="\
8  144" />
```

As for Android icon, PhoneGap Build support ldpi, mdpi, hdpi, and xhdpi display resolution. The following elements will define icon for each specific screen size.

```
1  <icon src="icons/android/ldpi.png" gap:platform="android" gap:density="ldpi" />
2  <icon src="icons/android/mdpi.png" gap:platform="android" gap:density="mdpi" />
3  <icon src="icons/android/hdpi.png" gap:platform="android" gap:density="hdpi" />
4  <icon src="icons/android/xhdpi.png" gap:platform="android" gap:density="xhdpi" \
5  />
```

2. Application Splash Screens

You can have zero or more of `<gap:splash>` elements. If you don't add this element, your application will not have splash screen. PhoneGap Build support classic, retina, and iPad (and retina iPad starting from Phonegap 2.5.0) for splash screen resolution.

```
1  <gap:splash src="splash/ios/Default.png" gap:platform="ios" width="320" height=\
2  "480" />
3  <gap:splash src="splash/ios/Default_at_2x.png" gap:platform="ios" width="640" h\
4  eight="960" />
5  <gap:splash src="splash/ios/Default_iphone5.png" gap:platform="ios" width="640"\
6  height="1136" />
7  <gap:splash src="splash/ios/Default-Landscape.png" gap:platform="ios" width="10\
8  24" height="768" />
9  <gap:splash src="splash/ios/Default-Portrait.png" gap:platform="ios" width="768\
10 " height="1024" />
11 <!-- retina iPad support: PhoneGap 2.5.0+ only -->
12 <gap:splash src="splash/ios/Default-Landscape_at_2x.png" gap:platform="ios" wid\
13 th="2048" height="1496" />
14 <gap:splash src="splash/ios/Default-Portrait_at_2x.png" gap:platform="ios" width\
15 h="1536" height="2008" />
```

As for Android, PhoneGap Build support ldpi, mdpi, hdpi, and xhdpi display resolution for splash screen. The following elements will define icon for each specific screen size.

```

1  <gap:splash src="splash/android/ldpi.png" gap:platform="android" gap:density="l\
2  dpi" />
3  <gap:splash src="splash/android/mdpi.png" gap:platform="android" gap:density="m\
4  dpi" />
5  <gap:splash src="splash/android/hdpi.png" gap:platform="android" gap:density="h\
6  dpi" />
7  <gap:splash src="splash/android/xhdpi.png" gap:platform="android" gap:density="\
8  xhdpi" />

```

3. PhoneGap Version Preference

PhoneGap version can be configured through `<preference>` element. For example `<preference name="phonegap-version" value="2.7.0">`. Phonegap Build support Phonegap versions 2.0.0, 2.1.0, 2.2.0, 2.3.0, 2.5.0, and 3.2.0 (default). If you do not specify version, your application will use default version of PhoneGap.

4. Device Orientation Preference

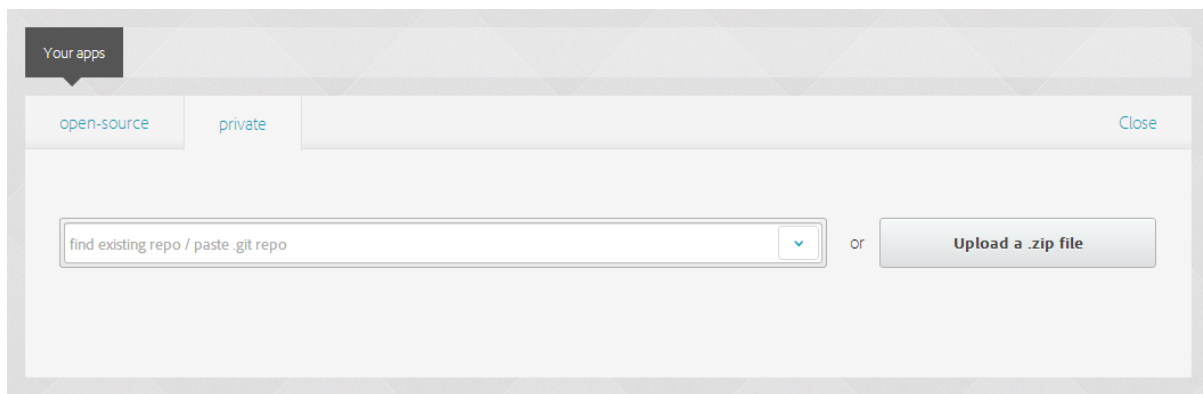
Device orientation can be configured through `<preference>` element. For example `<preference name="orientation" value="default" />`. Value has three options, they are default, landscape, and portrait. default means both landscape and portrait orientation are supported.

5. Targeting Specific Device Preference

Target specific preference works for iOS only. You can specify your application support handset, tablet, or universal. Example `<preference name="target-device" value="universal" />`

Zip Upload and Git Deployment

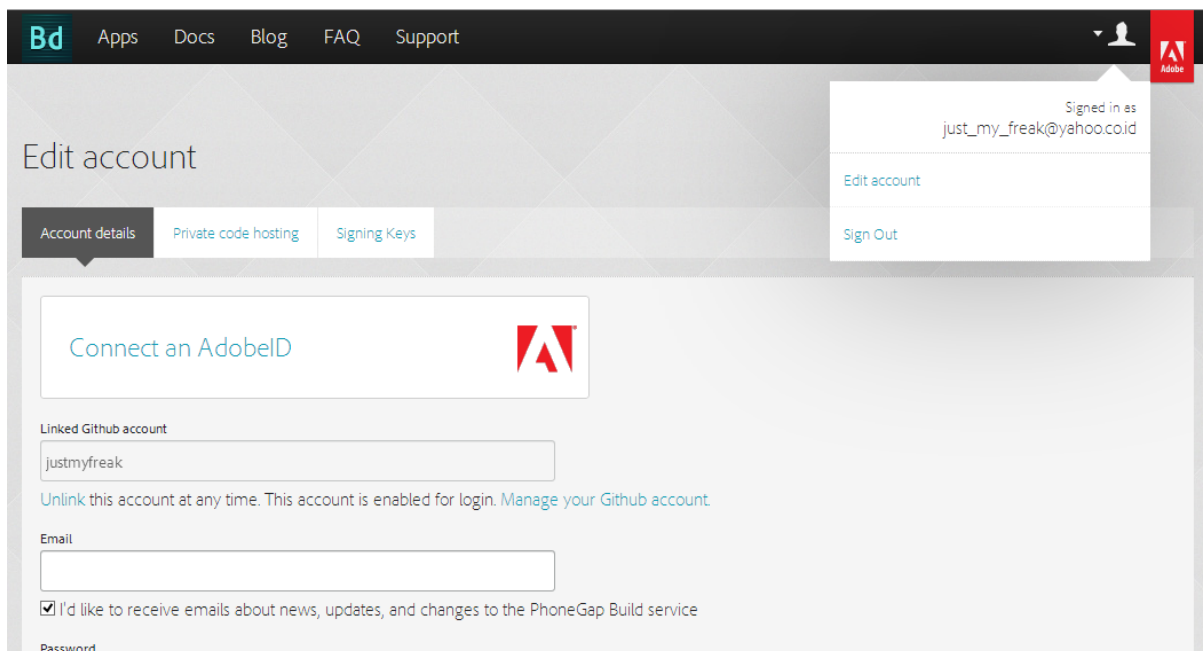
Uploading HTML5 application can be done by using two methods, uploading *.zip of your source code or by selecting existing repository in GitHub account. Zip upload only works for private application, while Git works for both private and open source application.



PhoneGap Deployment

Zip uploading is self explanatory. Click on new app then choose 'Upload a .zip file'. Then you are prompted with dialog in your browser to choose zipped assets of your HTML5 application. Your application will be build once it finished uploading.

To enable GitHub deployment, first you must link your GitHub to PhoneGap Build. To do it go to account, then choose edit account. And then link your GitHub account. If you want to build private application using existing GitHub private repository, you must provide your ssh key. To add your ssh key, simply click on 'Private code hosting' tab and add paste ssh key. Your existing application on GitHub now will appear in your 'new app' when creating new PhoneGap Build application.



GitHub Deployment

And that's it!!

By this point, you should be ready to use PhoneGap Build and are free to play around and discover more about it.