Red Hat Mobile Application Platform Hosted 3

Getting Started

For Red Hat Mobile Application Platform Hosted 3

Last Updated: 2017-12-21

Red Hat Mobile Application Platform Hosted 3 Getting Started

For Red Hat Mobile Application Platform Hosted 3

Legal Notice

Abstract

This tutorial guides you through the core features of Red Hat Mobile Application Platform Hosted 3.

Table of Contents

PREFACE

Overview

To get started with Red Hat Mobile Application Platform Hosted (RHMAP) quickly, the first step is to understand the project life cycle. This involves creating a project, creating a client and a Cloud App from templates, deploying the Cloud App to the MBaaS, building the Client App, and deploying it to a mobile device.

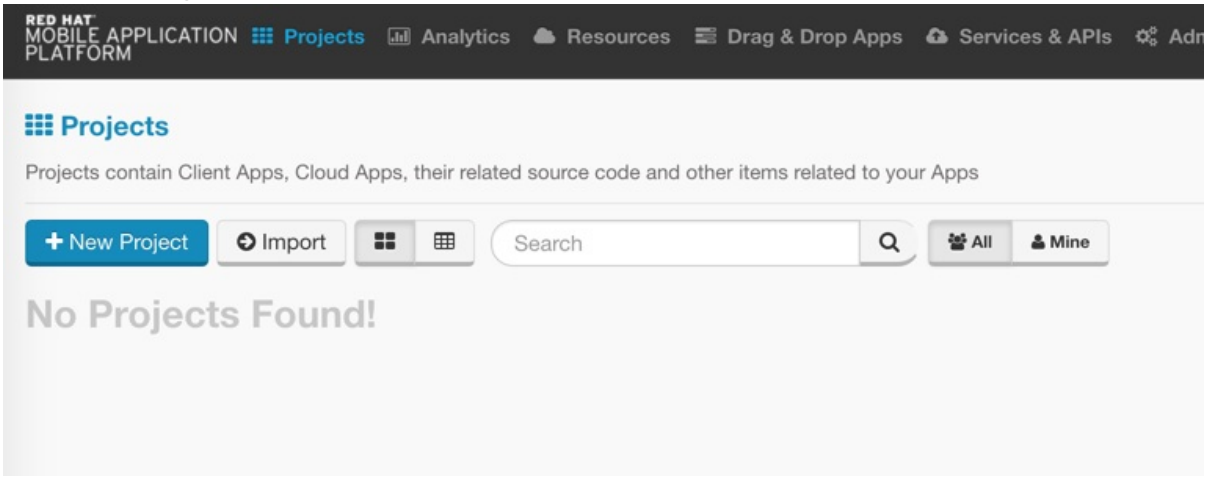
This guide uses the Studio — the web interface of RHMAP — for all operations. However, you can also use the FHC command line tool to access most functions of RHMAP.

CHAPTER 1. CREATE A PROJECT

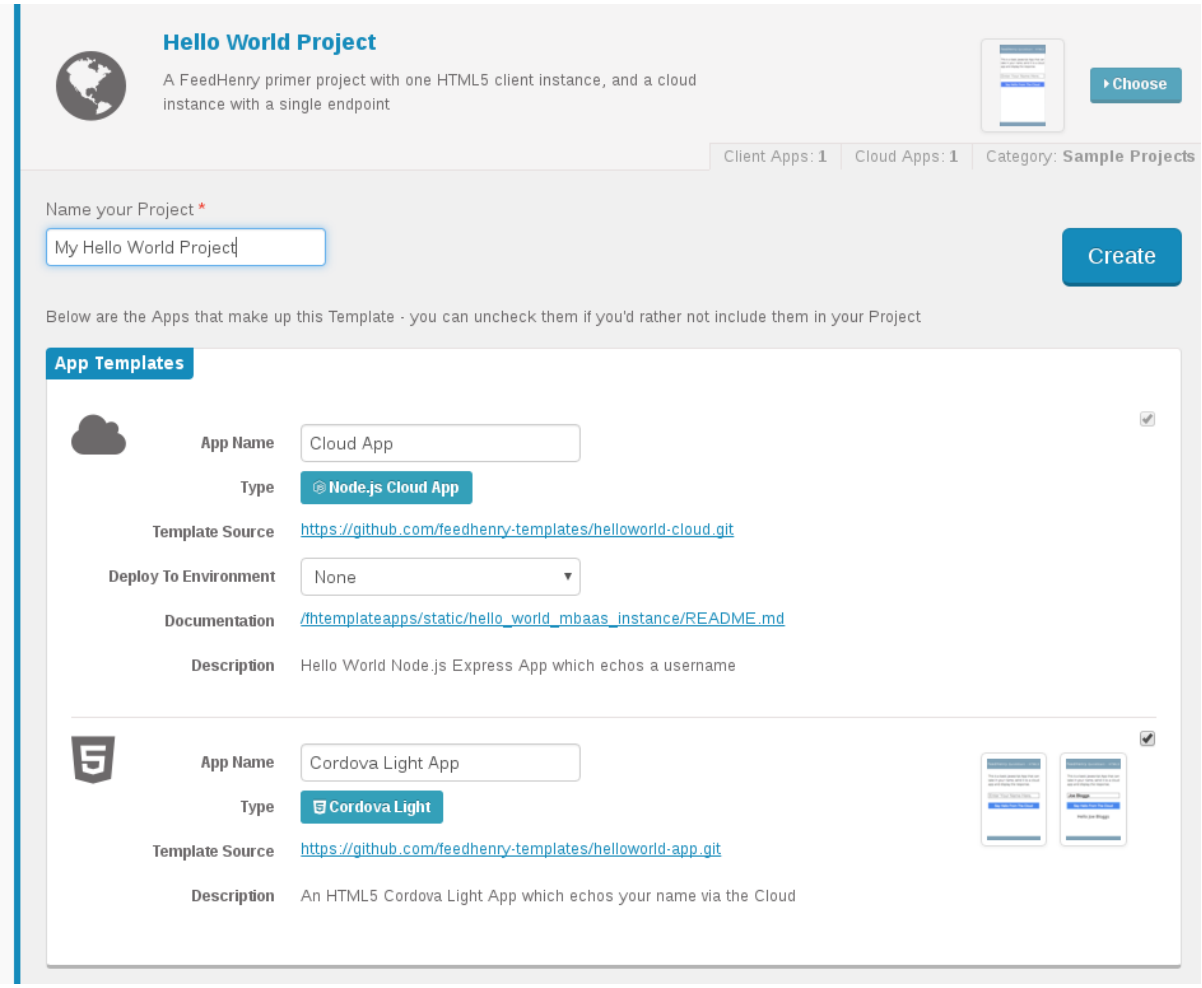
Projects help you group all code bases related to a single mobile application in one place. Projects contain Client Apps, Cloud Apps, MBaaS services, and any data and configurations associated with them.

Create a new project from an existing template. RHMAP offers many project templates to choose from out of the box, with basic *Hello World* examples for all supported platforms.

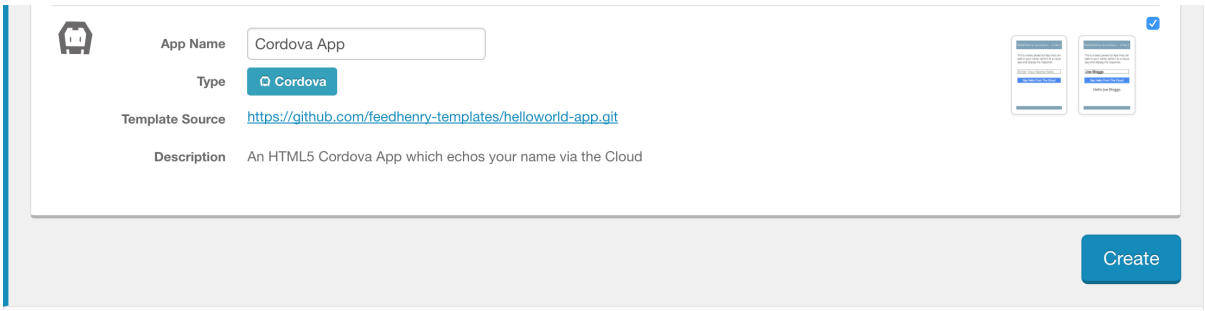
1. Log in to the Studio and navigate to the **Projects** area.
2. Click **New Project**.



3. Select the **Hello World Project** template by clicking **Choose** on the right side next to it.
4. Enter a name for the project in the "App Name" field.



5. Scroll down to find the Cordova App option. Enter a name for the project in the "App Name" field and also ensure the checkbox is selected.



6. Click **Create**.

7. The progress bar turns green when the project is successfully created. Click **Finish**.

1.1. EXPLORE THE PROJECT

After creating a project, you can see the **Apps, Cloud Apps & Services** section. This displays the Client and Cloud Apps, and also the MBaaS services associated with the project.

- **Client Apps**: applications deployed on mobile devices used by the end users.
- **Cloud Apps**: applications deployed in the MBaaS that handle requests from Client Apps and communicate with other internal or external systems.
- **MBaaS Services**: reusable services used by Cloud Apps and shared across multiple projects.

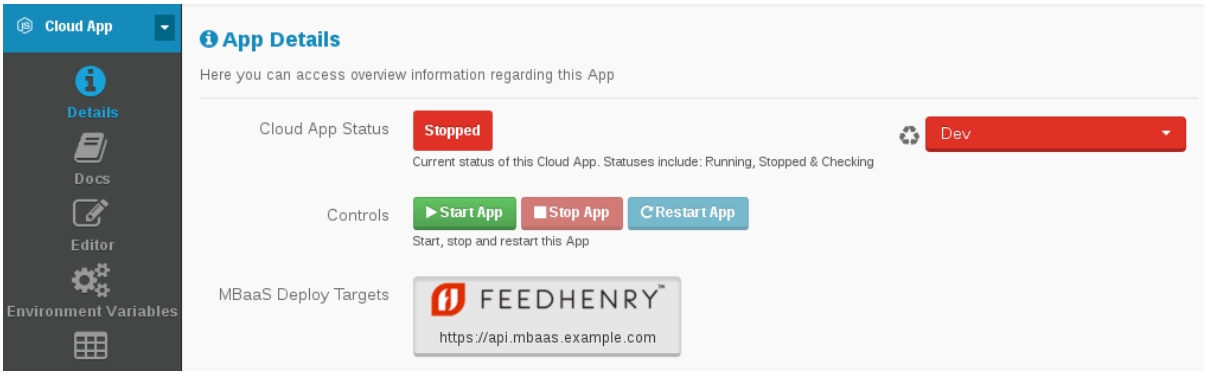
The newly created *My Hello World Project* contains one Client App (Cordova technology) and one Cloud App (Node.js technology) with a single HTTP endpoint. You can add more Client and Cloud Apps, and also MBaaS services to the project by clicking the **+** symbol in each corresponding box.

CHAPTER 2. DEPLOY THE CLOUD APP

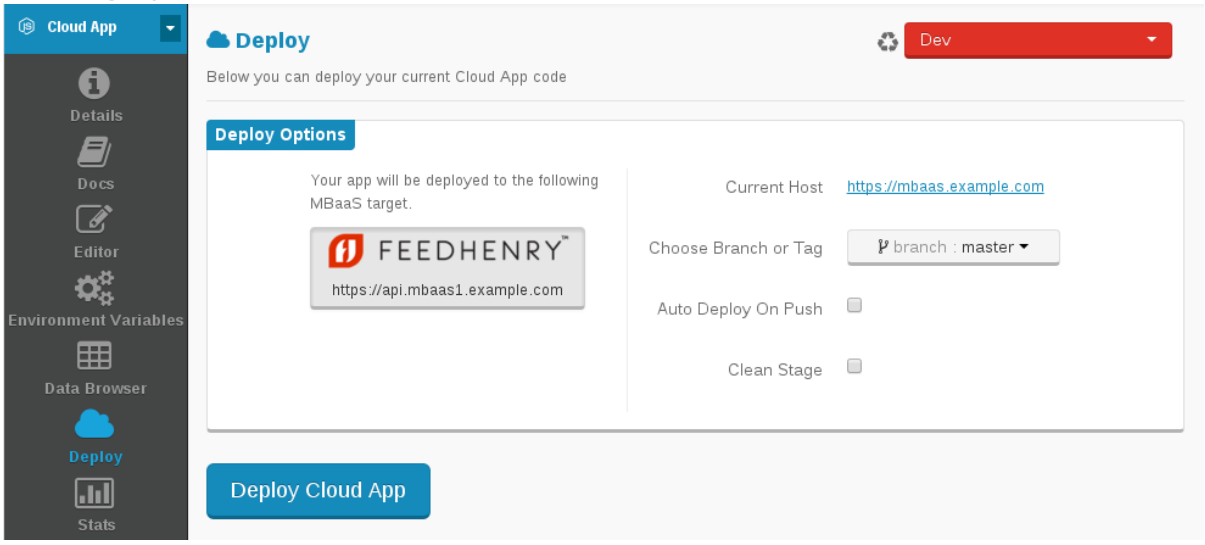
Depending on the setup of your cluster, the Cloud App may need to be deployed manually after creating the project.

1. In the **Apps, Cloud Apps & Services** section of the screen, click on the *Cloud App* - this will display the *App Details* screen.

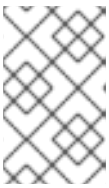
2. In the *App Details* section of the screen, see the value of the **Cloud App Status** field.
   If the status is *Stopped*, you must deploy the Cloud App. If it is *Running*, skip to section *3. Preview the Client App*.



3. Click **Deploy** on the sidebar on the left.



4. Click **Deploy Cloud App**



NOTE

By default, the Cloud App deployment reuses existing docker images built from the same commit hash and for the same Node.JS runtime. To force RHMAP to create a new docker image, select the **Clean Stage** option.

Once the deployment is finished, the progress bar turns green and the Cloud App is deployed.

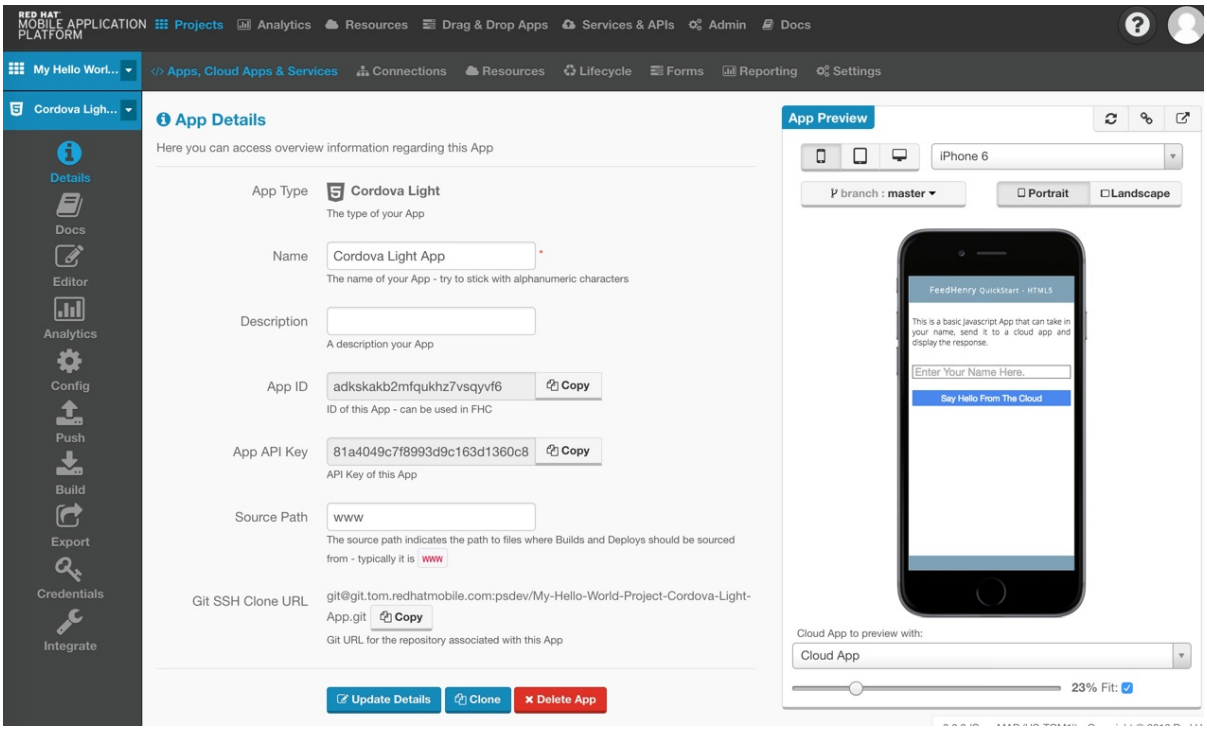
5. Click **Details** on the sidebar on the left.

6. Verify that the **Cloud App Status** is now *Running*.

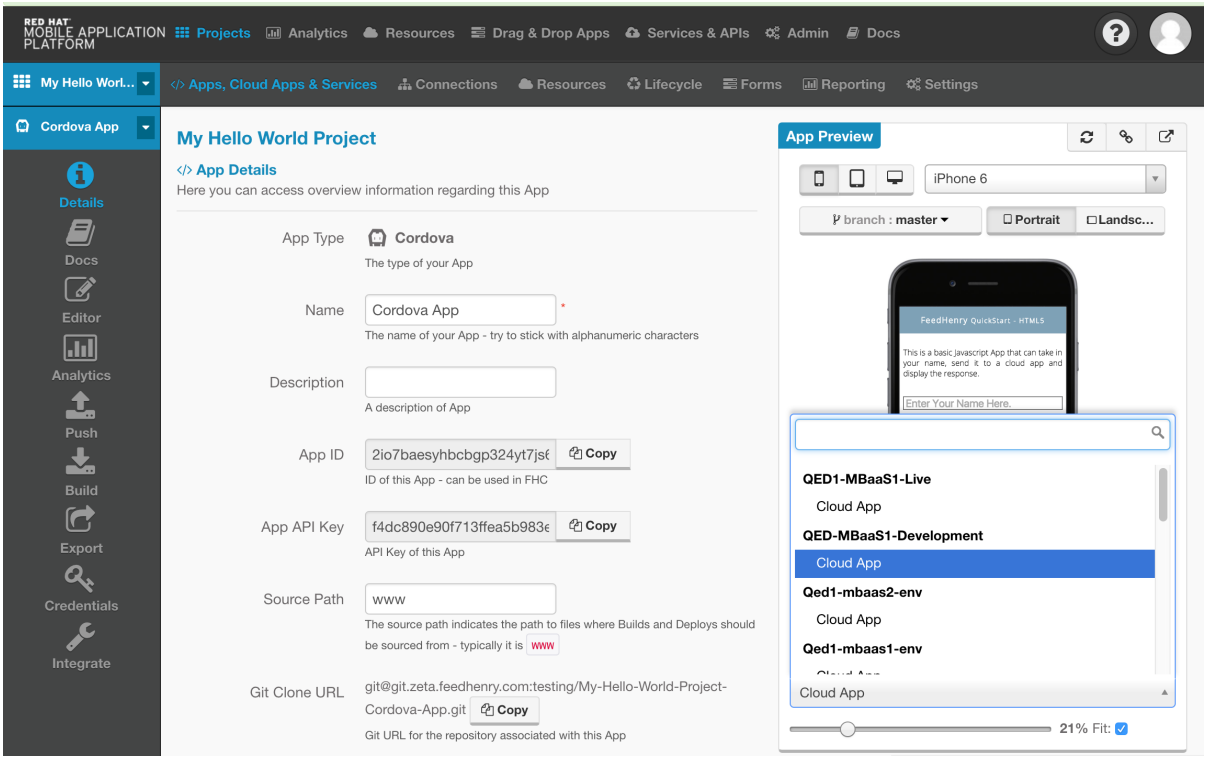
CHAPTER 3. PREVIEW THE CLIENT APP

With the project created and the Cloud App deployed, you can now test the Client App.

1. In the **Apps, Cloud Apps & Services** screen, click on the *Cordova App* in the **Apps** box. This opens the **App Details** page. On the right, you can interact with a running preview of your app. On the left is metadata, such as your app's name, ID, and git repository URL.



2. Ensure that the environment where the Cloud App is deployed to is selected in the "App Preview" section.



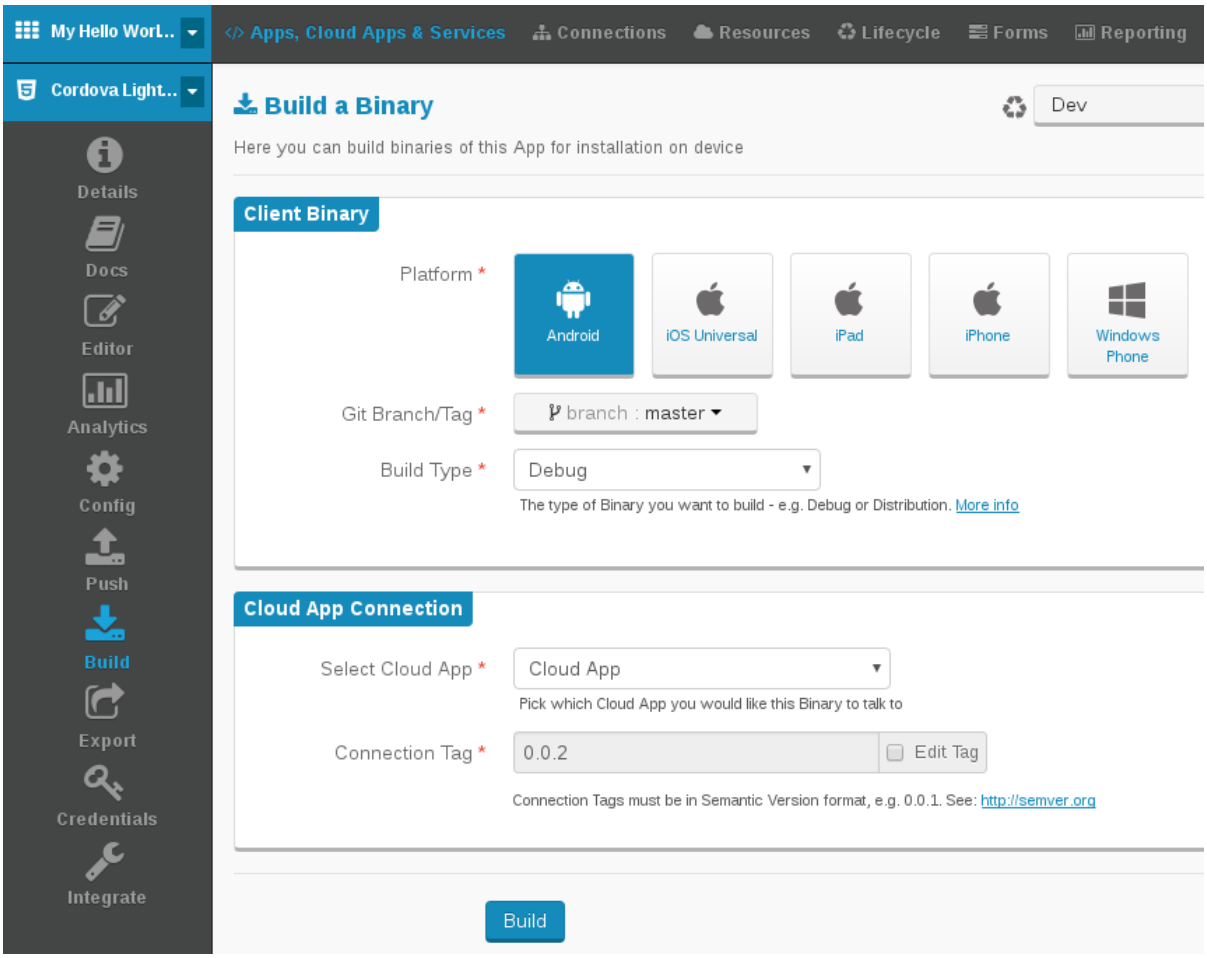
3. In the preview, enter your name in the provided box.

4. Click **Say Hello From The Cloud**.

The Client App makes a call to the Cloud App in this project and shows a response in the area beneath the button.

CHAPTER 4. RUN THE CLIENT APP ON A MOBILE DEVICE

1. Click **Build** on the sidebar on the left.
2. In the **Client Binary** section, select **Android** as the target platform.
3. Click **Build**.



After some time, the binary gets built and you are presented with a URL and a QR code.

4. Install the Client App binary on your mobile device, in one of the following ways:
   - Download the Client App binary to your computer and transfer it to the device manually.
   - On the mobile device, open the download URL in a browser.
   - On the mobile device, scan the QR code with a QR code reader. The QR code contains the download URL.

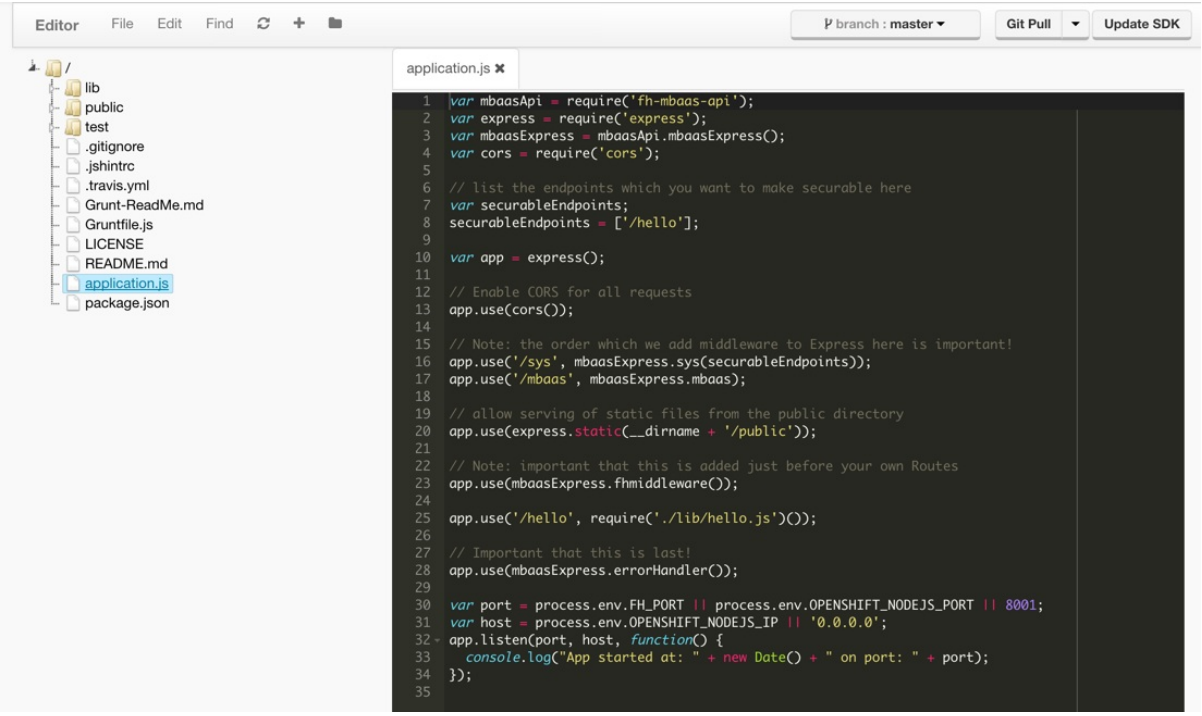
NOTE

On your Android mobile device, you must enable the option to install apps from unknown sources. See the section User Opt-In for Apps from Unknown Sources in the *Alternative Distribution Options* guide in Android documentation for more information. The Client App built in this example is considered by Android as coming from an *unknown source* since the Client App binary is not signed with a developer's certificate.

CHAPTER 5. CUSTOMIZE THE CLOUD APP

To better understand how the Cloud App works, make a minor modification to the code. Add a **`timestamp`** field with the value of the current UNIX time stamp to the server response. In the next section, you will modify the Client App to display the time stamp.

1. Navigate to the **Projects** area using the navigation bar at the top.
2. Open the *My Hello World Project* project.
3. Open the *Cloud App*.
4. Click **Editor** on the sidebar on the left.
   This area lets you edit the source code of any file in the Git repository of the Cloud App. The Cloud App in this project is a Node.js web application framework called *Express*.



5. Open the **`application.js`** file.
   `application.js` handles all requests to the Cloud App. The Client App sends requests to the **`/hello`** endpoint and the **`application.js`** file routes those requests to another file called **`hello.js`**.

```
app.use('/hello', require('./lib/hello.js')());
```

   To learn more about routing in Express, take a look at the Express Router documentation.

Change the **`lib/hello.js`** file to return a timestamp in the response.

1. Open **`lib/hello.js`**.
2. Add a **`timestamp`** property to the POST response object, with the value of the current UNIX time stamp.
   Find this line in the POST handler:

```
res.json({msg: 'Hello ' + world});
```

Change that line to the following:

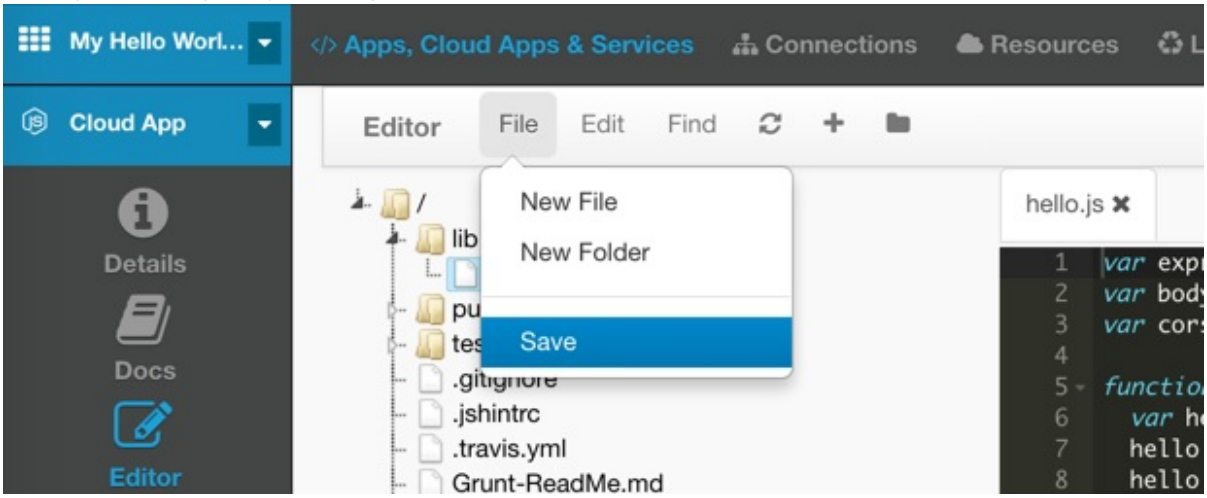
```
res.json({msg: 'Hello ' + world, timestamp: new Date().getTime() });
```

The POST handler now looks like this:

```
hello.post('/', function(req, res) {
  console.log(new Date(), 'In hello route POST / req.body=',
req.body);
  var world = req.body && req.body.hello ? req.body.hello : 'World';

  // see http://expressjs.com/4x/api.html#res.json
  res.json({msg: 'Hello ' + world, timestamp: new Date().getTime()
});
});
```

3. Save your changes by clicking *File > Save* in the editor.



The changes are saved to the Git repository of the Cloud App. To propagate the changes to the running instance, you must re-deploy the Cloud App.

4. Click **Deploy** on the sidebar on the left.

5. Click **Deploy Cloud App**.

CHAPTER 6. MODIFY THE CLIENT APP

Change the Client App to also show the **`timestamp`** property from the received server response.

First, create a placeholder for the response.

1. Navigate to the **Apps, Cloud Apps & Services** page.
2. Open the **Cordova App** Client App.
3. Open the **Editor**.
4. Open the **`www/index.html`** file.
5. Add a new **`<div>`** that will show the received **`timestamp`**.
   This element acts as a placeholder for the received value.

   Find this line:

   ```
   <div id="cloudResponse" class="cloudResponse"></div>
   ```

   Replace it with the following:

   ```
   <div id="cloudResponse" class="cloudResponse"></div>
   <div id="timestamp" class="cloudResponse"></div>
   ```

6. Save the changes using *File > Save*, or using the *Ctrl + S* keyboard shortcut (Windows) or *cmd + S* keyboard shortcut (Mac).

Modify the handler of the **Say Hello From The Cloud** button to use the received **`timestamp`** value to populate the placholder.

1. Open the **`www/js/hello.js`** file in the editor.
   This file contains a click handler for the **Say Hello From The Cloud** button, which uses the **`$fh.cloud`** API to call the **`/hello`** endpoint of the Cloud App and populates the placeholder **`<div id="timestamp">`** element.
2. Set the placeholder to the received **`timestamp`** value.
   Find the following code:

   ```
   document.getElementById('cloudResponse').innerHTML = "<p>" + res.msg
   + "</p>";
   ```

   Replace it with the following:

   ```
   document.getElementById('cloudResponse').innerHTML = "<p>" + res.msg
   + "</p>";
   document.getElementById('timestamp').innerHTML = "<p>" +
   res.timestamp + "</p>";
   ```

3. Save your changes.
   The preview will update automatically.
4. Click **Say Hello From The Cloud** in the preview.

The area below the button now also contains a long string of numbers, which represent the time stamp. If it does not work, try refreshing the page.

CHAPTER 7. CONCLUSION

You should now have an understanding of basic concepts, such as:

- Projects, Client Apps, and Cloud Apps and how these are related.
- Building an app binary for Android and trying it on a mobile device.
- Making changes to Client Apps and Cloud Apps.

You can find detailed guides and explanations in the RHMAP documentation.

For a more detailed walkthrough, which includes some insight into local development, you can watch the Red Hat Mobile Application Platform Overview Demo.