



# Red Hat Mobile Application Platform

## ホスト型 3

## スタートガイド

---

Red Hat Mobile Application Platform ホスト型 3 向け

Red Hat Customer Content  
Services





## 法律上の通知

Copyright © 2017 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat Software Collections is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

## 概要

本チュートリアルでは、Red Hat Mobile Application Platform ホスト型 3 のコアとなる機能を解説します。

---

## 目次

前書き .....	3
第1章 プロジェクトの作成 .....	4
1.1. プロジェクトについて調べる .....	5
第2章 クラウドアプリのデプロイ .....	7
第3章 クライアントアプリのプレビュー .....	8
第4章 モバイルデバイスでのクライアントアプリの実行 .....	10
第5章 クラウドアプリのカスタマイズ .....	12
第6章 クライアントアプリの変更 .....	14
第7章 まとめ .....	16



---

# 前書き

## 概要

Red Hat Mobile Application Platform ホスト型 (RHMAP) をすばやく開始するには、プロジェクトのライフサイクルを理解することが重要です。これにはプロジェクトの作成、テンプレートからのクライアントとクラウドアプリの作成、MBaaS へのクラウドアプリのデプロイ、クラウドアプリの構築、クラウドアプリのモバイルデバイスへのデプロイが関わってきます。

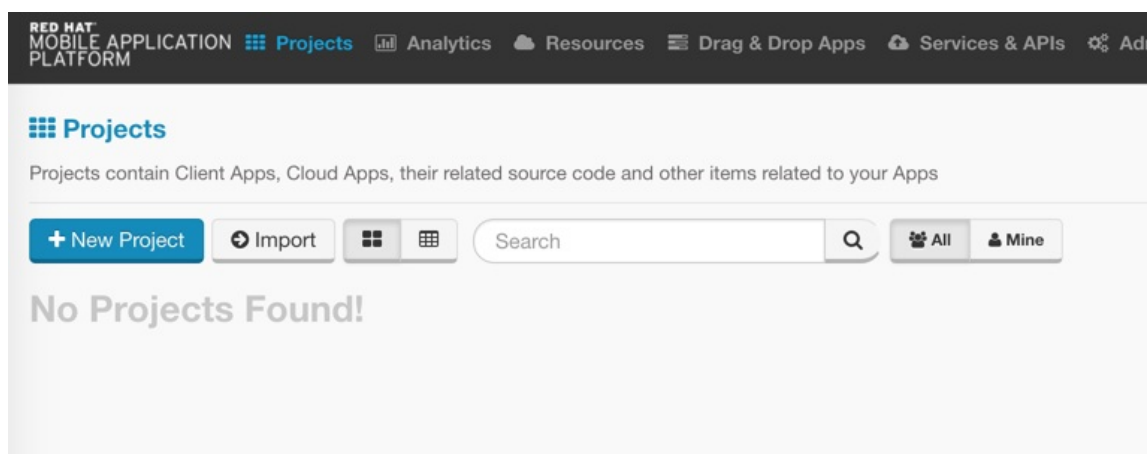
本ガイドの全操作では、RHMAP のウェブインターフェースである Studio を使用しています。ただし、[FHC コマンドラインツール](#) を使っても、RHMAP のほとんどの機能にアクセスすることができます。

## 第1章 プロジェクトの作成

プロジェクトを使うと、単一のモバイルアプリケーションに関連したコードベースをすべて 1 つの場所にまとめることができます。プロジェクトには、クライアントアプリ、クラウドアプリ、MBaaS サービス、およびこれらに関連するすべてのデータと設定が格納されます。

既存のテンプレートから新規プロジェクトを作成します。RHMAP には最初から多くのプロジェクトテンプレートが含まれており、基本的な **Hello World** の例はサポート対象の全プラットフォームで使用できます。

1. Studio にログインし、**プロジェクト** エリアに移動します。
2. **新規プロジェクト** をクリックします。



3. **Hello World Project** テンプレートの右側にある **選択** をクリックしてこれを選択します。
4. プロジェクトの名前を「アプリ名」フィールドに入力します。



**Hello World Project**  
A FeedHenry primer project with one HTML5 client instance, and a cloud instance with a single endpoint

Client Apps: 1 | Cloud Apps: 1 | Category: Sample Projects

Name your Project \*  
My Hello World Project

Create

Below are the Apps that make up this Template - you can uncheck them if you'd rather not include them in your Project

**App Templates**

**Cloud App**  
Type: Node.js Cloud App  
Template Source: <https://github.com/feedhenry-templates/helloworld-cloud.git>  
Deploy To Environment: None  
Documentation: [/htemplateapps/static/hello\\_world\\_mbaas\\_instance/README.md](/htemplateapps/static/hello_world_mbaas_instance/README.md)  
Description: Hello World Node.js Express App which echos a username

**Cordova Light App**  
Type: Cordova Light  
Template Source: <https://github.com/feedhenry-templates/helloworld-app.git>  
Description: An HTML5 Cordova Light App which echos your name via the Cloud

5. Cordova App のオプションまでスクロールダウンします。プロジェクト名を「アプリ名」フィールドに記入し、チェックボックスにチェックを入れます。

https://github.com/feedhenry-templates/helloworld-app.git. The 'Description' is 'An HTML5 Cordova App which echos your name via the Cloud'. The 'Create' button is at the bottom right."/>

**Cordova App**  
Type: Cordova  
Template Source: <https://github.com/feedhenry-templates/helloworld-app.git>  
Description: An HTML5 Cordova App which echos your name via the Cloud

Create

6. 作成 をクリックします。
7. プロジェクト作成に成功すると、進捗バーが緑色になります。終了 をクリックします。

## 1.1. プロジェクトについて調べる

プロジェクトを作成したら、**アプリ**、**クラウドアプリ** & **サービス** セクションが表示されます。ここではプロジェクトに関連するクライアントアプリ、クラウドアプリ、および MBaaS サービスが表示されます。

- ※ **クライアントアプリ**: エンドユーザーが使用するモバイルデバイスにデプロイされたアプリケーション。
- ※ **クラウドアプリ**: クライアントアプリからのリクエストを処理し、他の内部または外部システムと通信する、MBaaS にデプロイされたアプリケーション。

- ※ **MBaaS サービス**: クラウドアプリが使用し、複数のプロジェクトで共有される再利用可能なサービス。

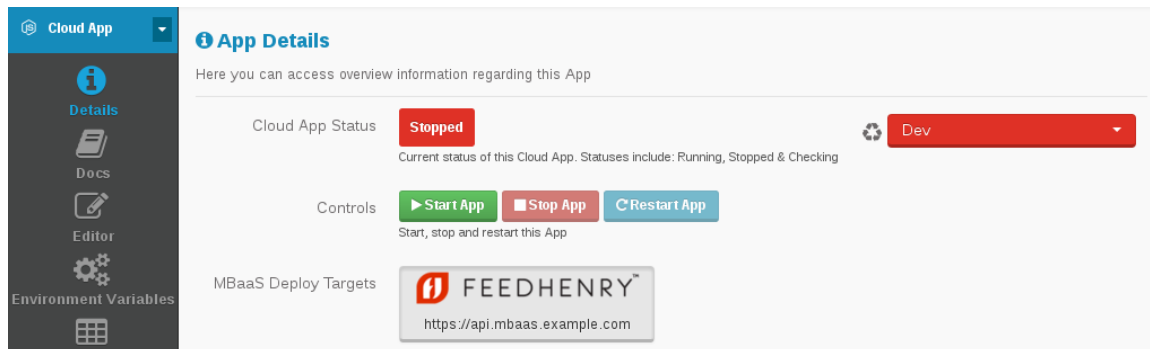
新規に作成された **My Hello World Project** には、Cordova アプリ (Cordova テクノロジー) 1 つと HTTP エンドポイントがあるクラウドアプリ (Node.js テクノロジー) 1 つが含まれます。対応する各ボックスの + 記号をクリックすると、クライアントアプリ、クラウドアプリ、MBaaS サービスを追加することができます。

## 第2章 クラウドアプリのデプロイ

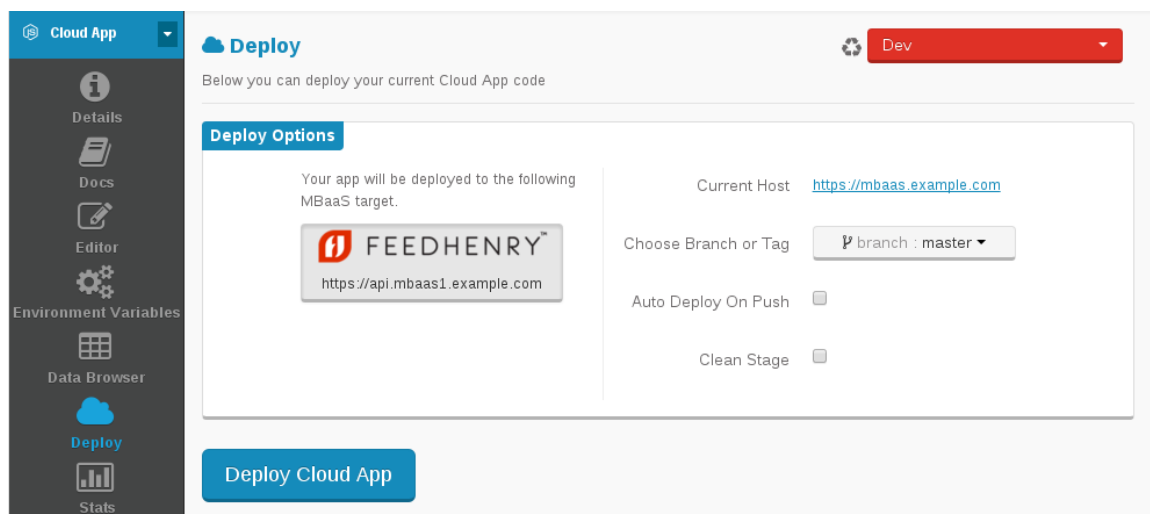
お使いのクラスターの設定によっては、プロジェクト作成後にクラウドアプリを手動でデプロイする必要がある場合があります。

1. 画面の **アプリ、クラウドアプリ & サービス** セクションで、**クラウドアプリ** をクリックします。これで **アプリの詳細** 画面が表示されます。
2. **アプリの詳細** 画面で **クラウドアプリのステータス** フィールドの値を確認します。

ステータスが **Stopped** の場合は、クラウドアプリをデプロイする必要があります。**Running** の場合は、[3. クライアントアプリのプレビュー](#) セクションに進みます。



3. 左側のサイドバーにある **デプロイ** をクリックします。



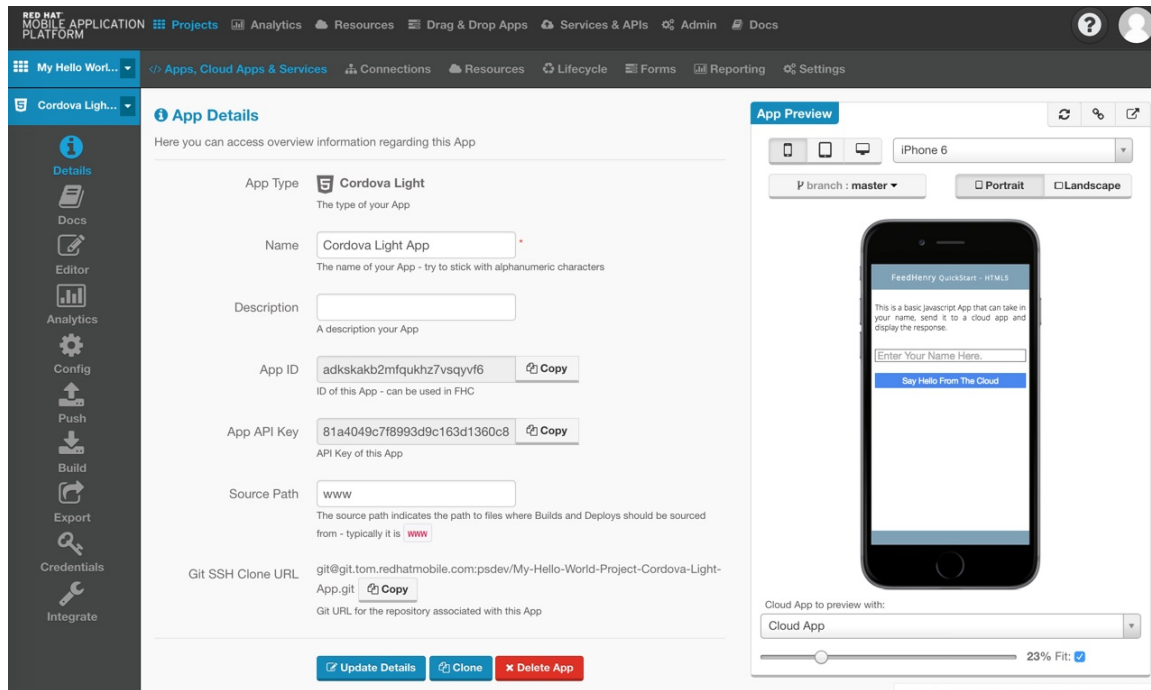
4. **クラウドアプリをデプロイ** をクリックします。デプロイが完了したら進捗バーが緑色になり、クラウドアプリがデプロイされます。
5. 左側のサイドバーにある **詳細** をクリックします。
6. **クラウドアプリのステータス** が **Running** になっていることを確認します。

## 第3章 クライアントアプリのプレビュー

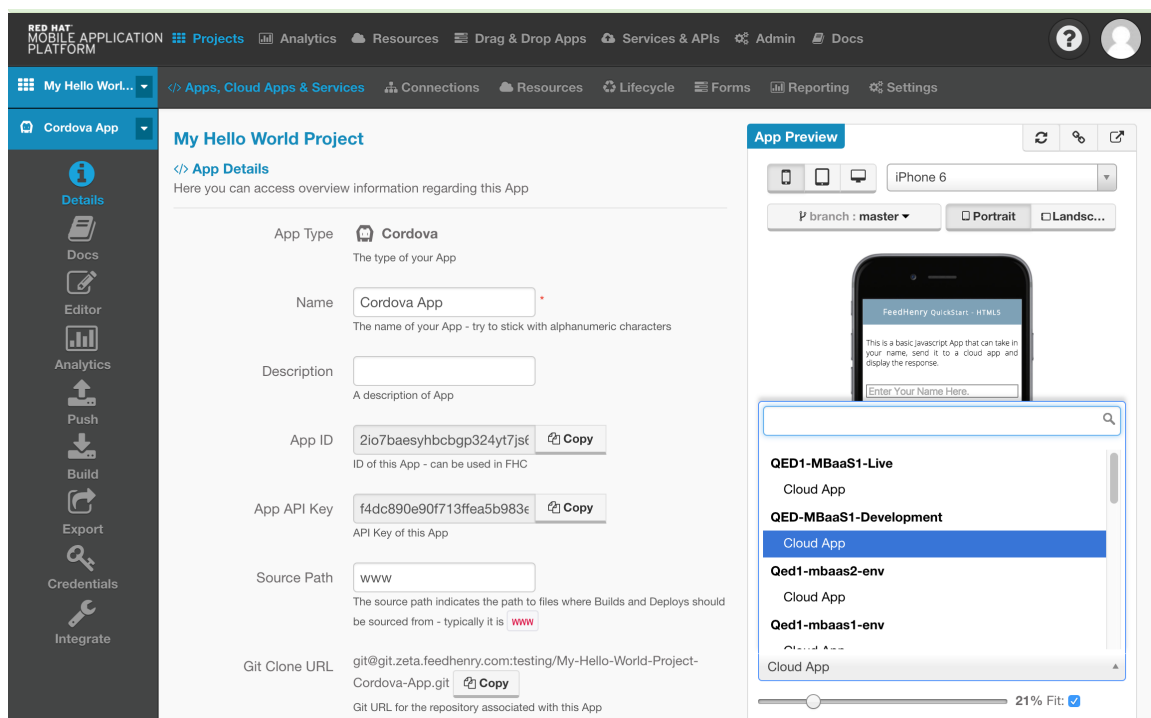
プロジェクトが作成され、クラウドアプリがデプロイされたら、クライアントアプリをテストできます。

1. アプリ、クラウドアプリ & サービス 画面の **アプリ** ボックスで **Cordova App** をクリックします。

これで **アプリ詳細** ページが開きます。右側で実行中のアプリのプレビューを操作することができ、左側にはアプリの名前、ID、git リポジトリ URL などのメタデータが表示されます。



2. プレビューするクラウドアプリが「アプリのプレビュー」セクションで選択されてることを確認します。



3. プレビューのボックスに名前を記入します。
4. **Say Hello From The Cloud** をクリックします。

クライアントアプリがプロジェクトのクラウドアプリを呼び出し、ボタンの下に応答が表示されます。

## 第4章 モバイルデバイスでのクライアントアプリの実行

1. 左側のサイドバーにある **ビルド** をクリックします。
2. クライアントバイナリー セクションで、ターゲットプラットフォームに **Android** を選択します。
3. **ビルド** をクリックします。

The screenshot shows the 'Build a Binary' page in the Red Hat Mobile Application Platform. The left sidebar contains navigation options: Details, Docs, Editor, Analytics, Config, Push, Build (highlighted), Export, Credentials, and Integrate. The main content area is titled 'Build a Binary' and includes a 'Dev' button. Below the title, it says 'Here you can build binaries of this App for installation on device'. The 'Client Binary' section has the following fields: Platform (with buttons for Android, iOS Universal, iPad, iPhone, and Windows Phone; Android is selected), Git Branch/Tag (set to 'branch : master'), and Build Type (set to 'Debug'). A note below Build Type says 'The type of Binary you want to build - e.g. Debug or Distribution. [More info](#)'. The 'Cloud App Connection' section has 'Select Cloud App' (set to 'Cloud App') and 'Connection Tag' (set to '0.0.2'). A note below says 'Connection Tags must be in Semantic Version format, e.g. 0.0.1. See: <http://semver.org>'. At the bottom is a 'Build' button.

しばらくするとバイナリーが構築され、URL と QR コードが表示されます。

4. モバイルデバイスにクライアントアプリのバイナリーをインストールするには、以下のいずれかの方法に従います。
  - ✳ クライアントアプリのバイナリーをコンピューターにダウンロードし、手動でデバイスに転送する。
  - ✳ モバイルデバイス上で、ブラウザーを使ってダウンロード URL を開く。
  - ✳ モバイルデバイスの QR コードリーダーで QR コードをスキャンする。QR コードにはダウンロード URL が含まれています。



#### 注記

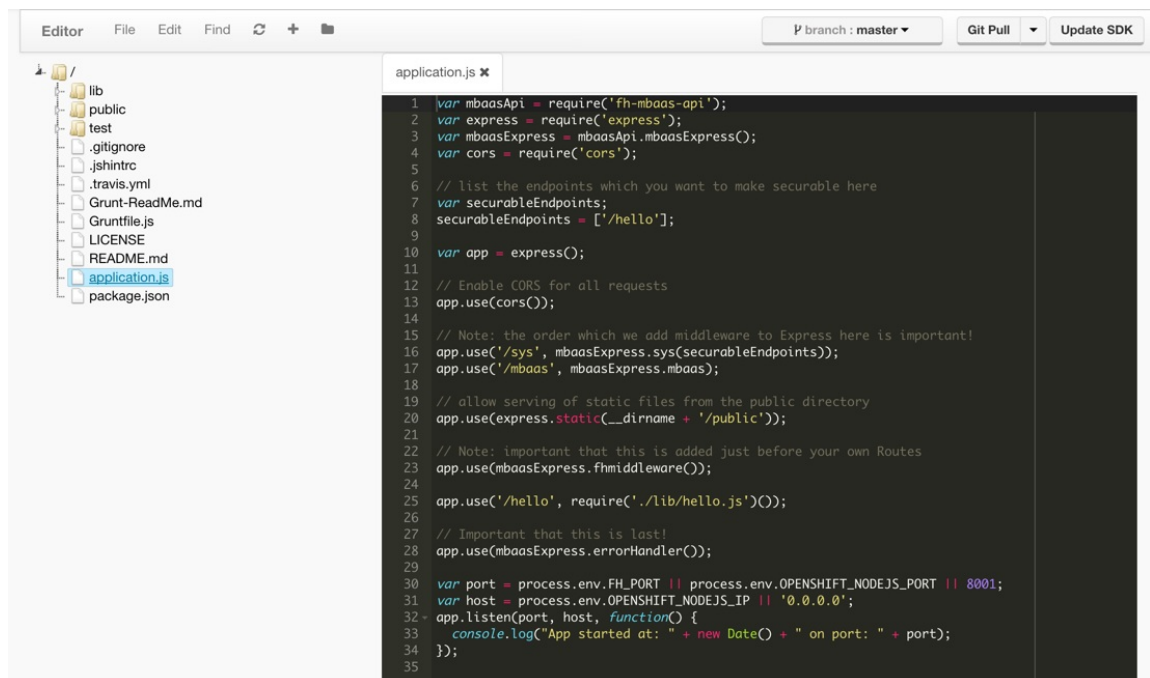
Androidモバイルデバイスでは、不明なソースからアプリをインストールするオプションを有効にしておく必要があります。Android用ドキュメントの **Alternative Distribution Options** ガイドにある [User Opt-In for Apps from Unknown Sources](#) セクションを参照してください。この例でビルドされたクライアントアプリのバイナリーは開発者の署名がないので、Androidでは **不明なソース** からのものとして認識されます。

## 第5章 クラウドアプリのカスタマイズ

クラウドアプリの機能について理解を深めるために、コードにマイナーな修正を加えてみましょう。サーバーからの応答に、現在の UNIX タイムスタンプの値を持つ **timestamp** フィールドを追加します。以下のセクションでは、クライアントアプリを修正してタイムスタンプを表示するようにします。

1. 上部のナビゲーションバーで **プロジェクト** エリアに移動します。
2. **My Hello World Project** プロジェクトを開きます。
3. **クラウドアプリ** を開きます。
4. 左側のサイドバーにある **エディター** をクリックします。

このエリアでは、クラウドアプリの Git リポジトリにあるファイルのソースコードを編集できます。このプロジェクトのクラウドアプリは、**Express** と呼ばれる Node.js web アプリケーションです。



5. **application.js** ファイルを開きます。

**application.js** は、クラウドアプリへのリクエストすべてを処理します。クライアントアプリがリクエストを **/hello** エンドポイントに送信し、**application.js** ファイルがこれらのリクエストを **hello.js** と呼ばれる別のファイルに転送します。

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

Express での転送についての詳細は、[Express Router documentation](#) を参照してください。

**lib/hello.js** ファイルを修正して、応答でタイムスタンプを返すようにします。

1. **lib/hello.js** を開きます。



2. **timestamp** 属性を POST 応答オブジェクトに追加し、値は現在の UNIX タイムスタンプにします。

POST ハンドラーで以下の行を探します。

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

この行を以下のように変更します。

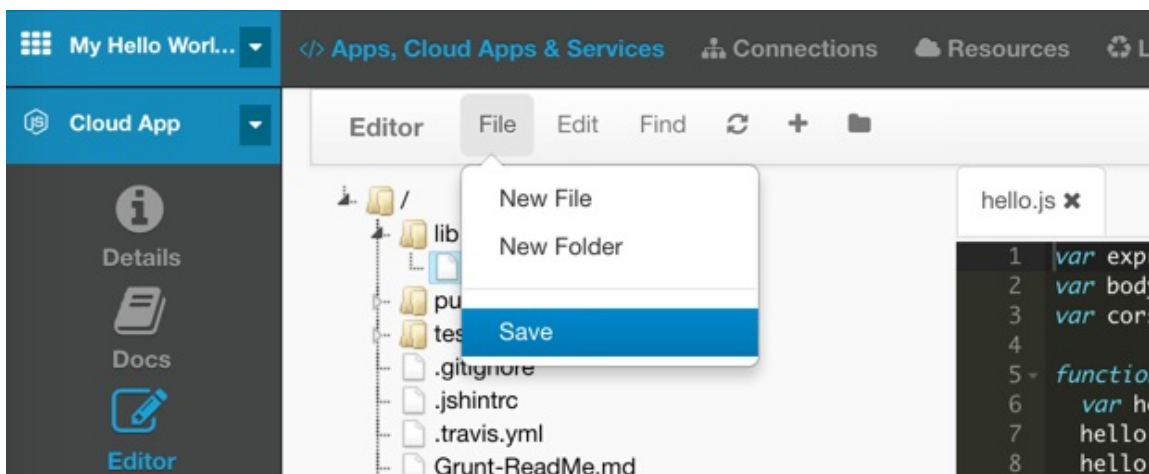
```
res.json({msg: 'Hello ' + world, timestamp: new Date().getTime()});
```

POST ハンドラーは以下のようになります。

```
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. エディターで **ファイル > 保存** をクリックして変更を保存します。



変更はクラウドアプリの Git リポジトリに保存されます。実行中のインスタンスに変更を適用するには、クラウドアプリを再度デプロイする必要があります。

4. 左側のサイドバーにある **デプロイ** をクリックします。
5. **クラウドアプリをデプロイ** をクリックします。

## 第6章 クライアントアプリの変更

クライアントアプリを変更して、受け取ったサーバー応答から **timestamp** 属性を表示するようにします。

まず、応答のプレースホルダーを作成します。

1. **アプリ、クラウドアプリ & サービス** ページに移動します。
2. **Cordova App** クライアントアプリを開きます。
3. **エディター** を開きます。
4. **www/index.html** ファイルを開きます。
5. 受信した **timestamp** を表示する **<div>** を新たに追加します。

この要素は、受信した値のプレースホルダーとして機能します。

以下の行を見つけます。

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

これを以下のコードで置き換えます。

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

6. **ファイル > 保存** を使用するか、**Ctrl + S** (Windows の場合) もしくは **cmd + S** (Mac の場合) のショートカットを使用して変更を保存します。

**Say Hello From The Cloud** ボタンのハンドラーを編集して、受信した **timestamp** の値をプレースホルダーに表示するようにします。

1. エディターで **www/js/hello.js** ファイルを開きます。

このファイルには、**Say Hello From The Cloud** ボタンのクリックハンドラーが含まれています。これは **\$fh.cloud** API を使用してクラウドアプリの **/hello** エンドポイントを呼び出し、プレースホルダー **<div id="timestamp">** 要素を作成します。

2. プレースホルダーに受信した **timestamp** の値を設定します。

以下のコードを見つけます。

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

これを以下のコードで置き換えます。

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

3. 変更を保存します。

プレビューが自動的に更新されます。

4. **Say Hello From The Cloud** をプレビューでクリックします。

ボタンの下のエリアには、タイムスタンプを表す数字が表示されるようになります。これ  
が表示されていない場合には、ページを更新します。



## 第7章 まとめ

以下の基本的なコンセプトが理解できていることを確認してください。

- 🔗 プロジェクト、クライアントアプリ、クラウドアプリ、およびそれらの関係。
- 🔗 アンドロイド用にアプリのバイナリーを構築し、モバイルデバイスでこれを試す。
- 🔗 クライアントアプリとクラウドアプリでの変更。

[RHMAP documentation](#) には詳細なガイドと説明が用意されています。

詳細 (ローカル開発の概要を含む) については、[Red Hat Mobile Application Platform Overview Demo](#) を視聴してください。