

Monaca Localkit β版 利用マニュアル

目次

1. Monaca Localkitとは
 - 1.1. 概要
 - 1.2. 動作環境
2. セットアップ
 - 2.1. Monaca Localkitの入手方法
 - 2.2. Monaca Localkitのインストール
 - 2.3. ファイアウォール設定 (windows8.1)
 - 2.4. ファイアウォール設定 (mac)
 - 2.5. Monaca Debuggerのインストール
3. 利用手順
 - 3.1. Localkit
 - Localkitの起動
 - ログイン
 - プロジェクト設定
 - プロジェクト配信
 - 3.2. Debugger
 - ログイン
 - サーバー検索
 - ペアリング
 - 3.3. ローカル開発
 - プロジェクトファイルの編集
 - リモートビルド
 - 3.4. 高度な開発
 - weinreを利用したデバッグ
 - Chrome dev toolsを利用したデバッグ (Windows)
 - Safari Webインスペクターを利用したデバッグ (Mac)
 - 3.5. リモートビルド
 - プロジェクトのアップロード
 - アプリ設定とビルド設定
 - リモートビルド
4. 注意事項
 - 4.1. 利用上の注意点
 - 4.2. お問い合わせ

1. Monaca Localkitとは

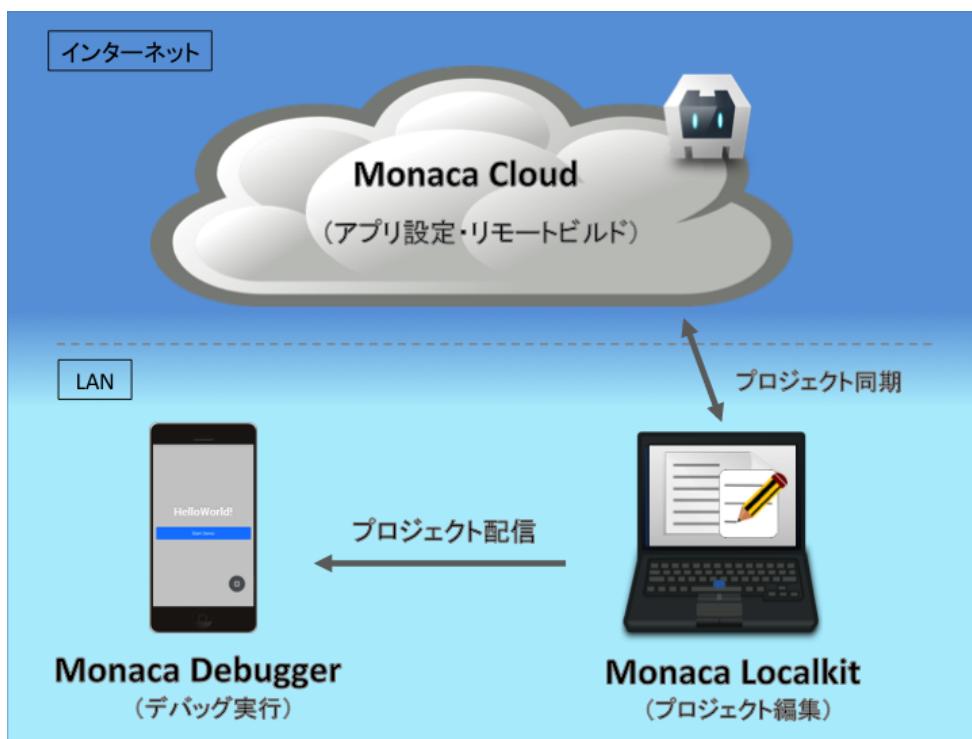
1.1 概要

Monaca Localkitは、アプリ開発プラットフォーム『[Monaca](#)』が提供するローカル環境開発支援ツールです。

従来のMonacaでは『[MonacaクラウドIDE](#)』や『[Monacaデバッガー](#)』など、インターネットを中心とした開発ツールを提供してきました。

Monaca Localkitは、ローカル環境下で様々な開発ツールとMonacaを組み合わせることで、よりセキュアで高速なアプリ開発を実現します。

システム構成イメージ



1.2 動作環境

Monaca Localkitを利用するためには以下の環境が必要です。

PC

OS	Windows7以上/ Mac OSX 10.9以上
必須ソフトウェア	Google Chrome

モバイル端末

OS	Android 2.3以上 / iOS6.0以上
必須ソフトウェア	Monacaデバッガー

2. セットアップ

ローカル開発環境のセットアップについて説明します。

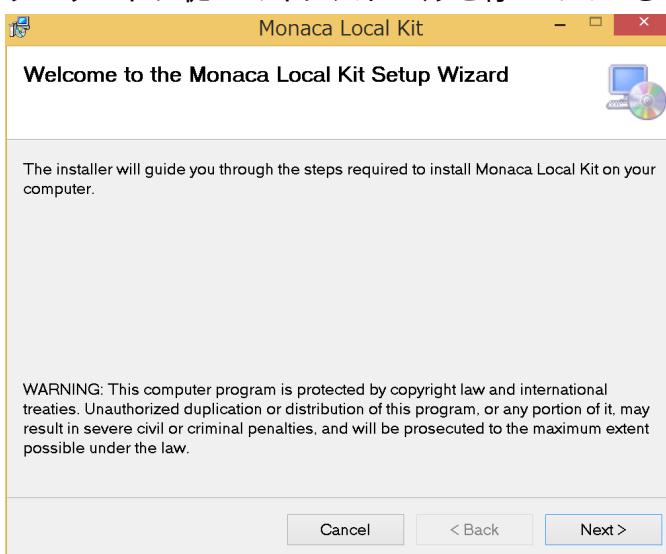
2.1 Monaca Localkitのインストール

Windows PCの場合

1. [こちら](#)からMonaca Localkitインストーラをダウンロードしてください
2. ダウンロードした Monaca LocalKit.msi を実行してください

名前	更新日時	種類	サイズ
Monaca Local Kit.msi	2014/10/28 9:01	Windows インストーラ パッケージ	2,161 KB

3. ウィザードに従ってインストールを行ってください



4. インストールが完了するとChromeウェブストアのMonaca Localkitのダウンロードページが表示されますので、Monaca LocalkitをGoogleChromeにインストールしてください

Macの場合

[Chromeウェブストア](#)にアクセスし、Monaca LocalkitをGoogleChromeにインストールしてください。

ChromeウェブストアのMonaca Localkit

<https://chrome.google.com/webstore/detail/monaca-localkit/igimoohpkianbofjknpbnfehmkecbegl>

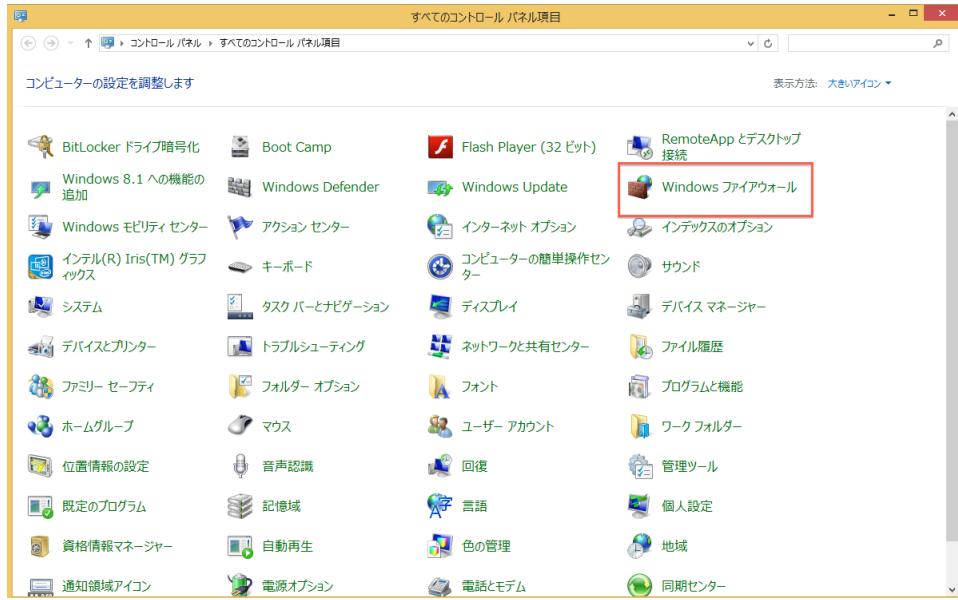
2.3 ファイアウォール設定

Windowsの場合

インストーラを利用してインストールを行った場合、ファイアウォール設定は自動的に行われるため、下記の作業は不要です。

Chromeウェブストアから直接Monaca Localkitをインストールされた方は、以下の手順に従ってファイアーウォール設定にGoogle Chromeを追加してください。

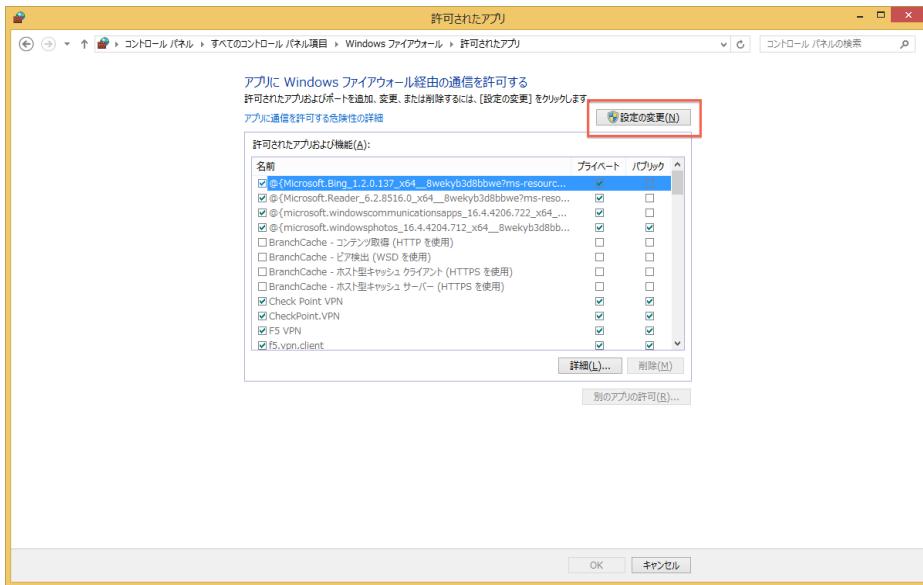
1. コントロールパネルを開き、[Windowsファイアーウォール]をクリックします



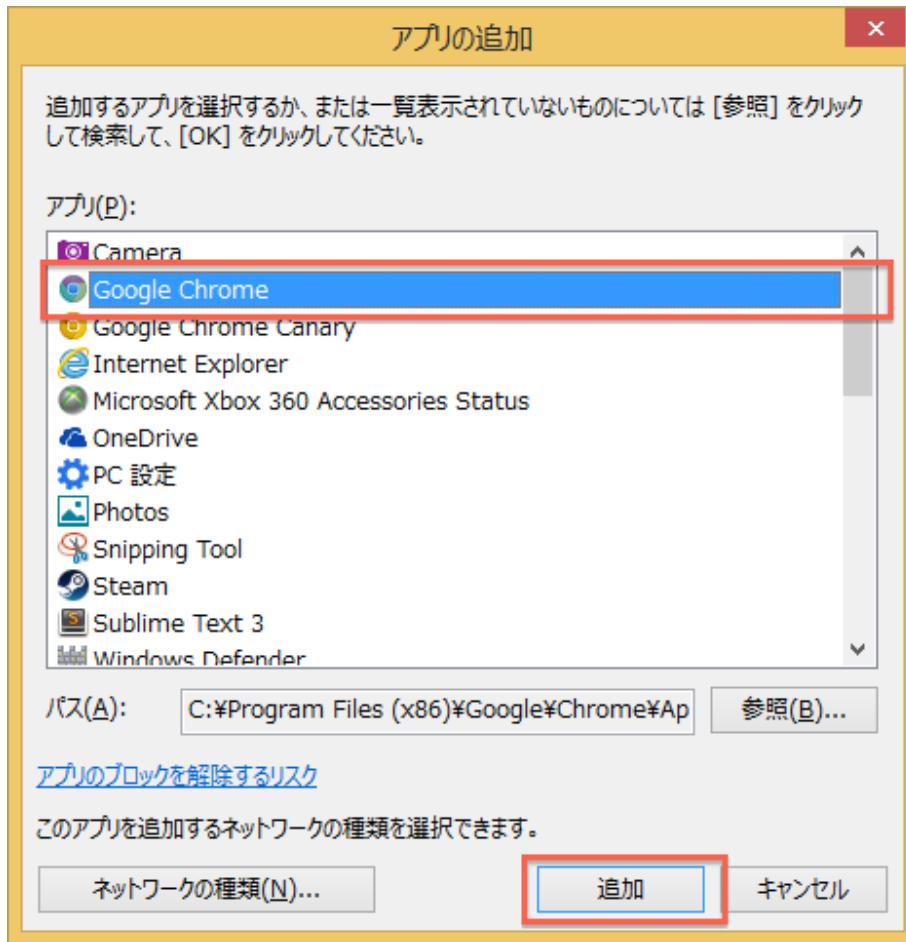
2. Windowsファイアーウォール設定画面で[Windowsファイアーウォールを介したアプリまたは機能を許可]をクリックします



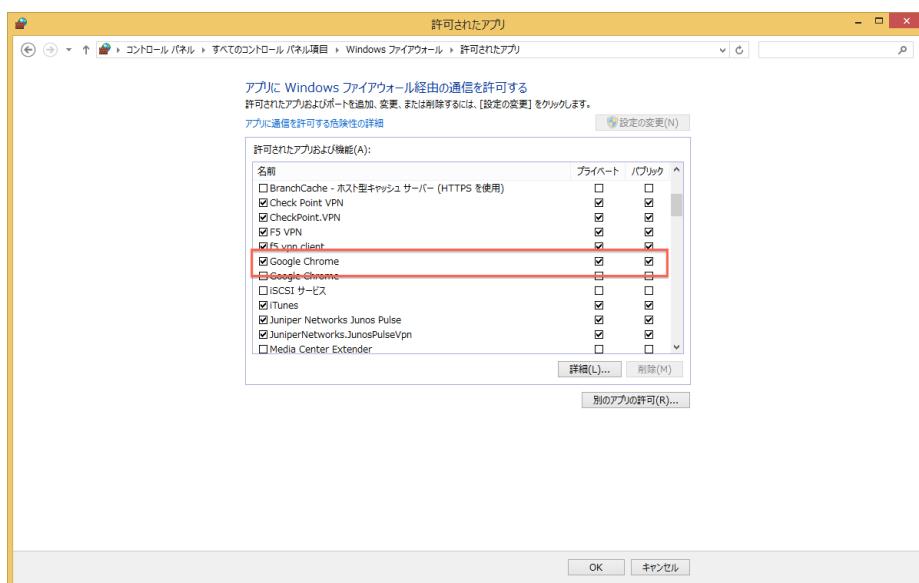
3. 許可されたアプリ画面で[設定の変更]をクリックします



4. [別のアプリの許可]が入力可能となるのでクリックします
5. アプリの追加ダイアログが表示されるのでGoogle Chromeを選択し、追加をクリックします



6. GoogleChomeが許可されていることと、Monaca Localkitとモバイル端末を接続しているネットワーク（プライベート・パブリックのどちらか）にチェックが付いていることを確認してください



Mac OS PCの場合

以下の手順に従ってファイアーウォール設定にGoogle Chromeを追加してください。

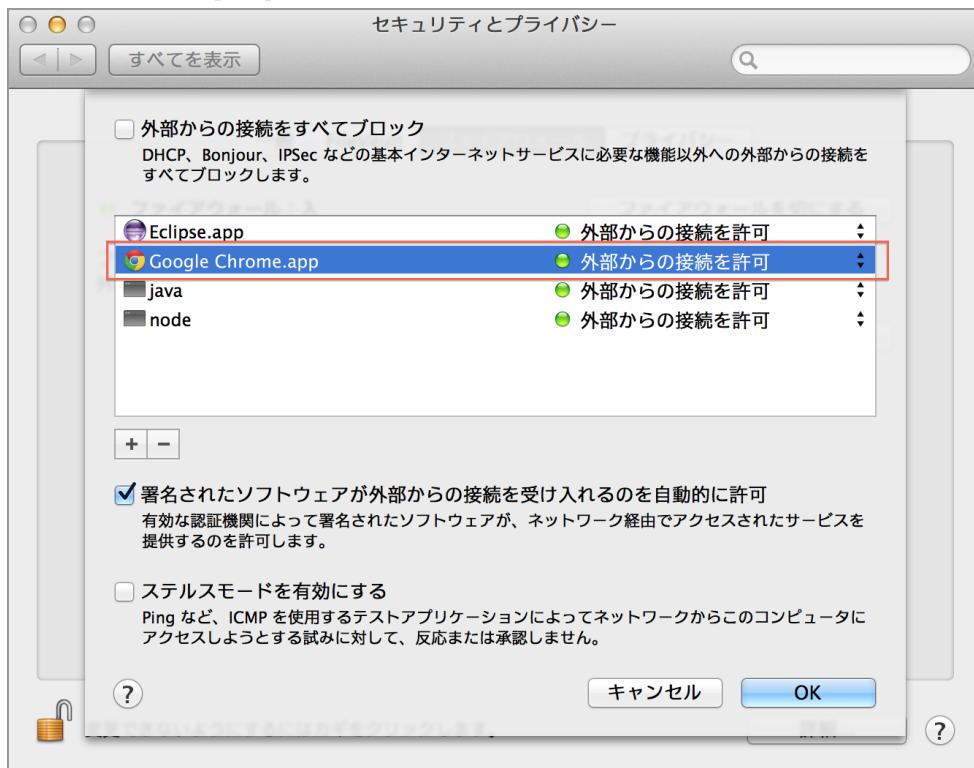
1. 「システム環境設定」を表示し[セキュリティとプライバシー]をクリックします



2. セキュリティとプライバシー画面で[ファイアーウォールオプション]をクリックします



3. ダイアログで[Google Chrome.app]を追加し、パラメータに「外部からの接続を許可」を設定し、[OK]をクリックします



2.4 Monaca Debuggerのインストール

ローカル開発で使用するiOS/Android端末にMonaca Debuggerをインストールしてください。

iOS端末の場合

- AppStore版デバッガー

AppStoreから『Monaca』をダウンロードしてください



- カスタムビルド版デバッガー

Monaca IDEで開発用のデバッガーをビルドして利用することができます。

カスタムビルド版デバッガーは、ユーザが独自のプラグインを組み込むことが可能で、safariのリモートデバッグ機能を利用することができます。

詳しくは[こちら](#)をご確認ください。

Android端末の場合

- GoolePlay/ハイパフォーマンス版

Crosswalkエンジンが組み込まれたデバッガーで、従来のAndroid WebViewより高性能なWebViewを利用できます。詳しくは[こちら](#)をご覧ください。

Google Playから『Monacaデバッガー（ハイパフォーマンス版）』をダウンロードしてください。



- GoolePlay通常WebView版

Google Playから『Monacaデバッガー』をダウンロードしてください。



3. 利用手順

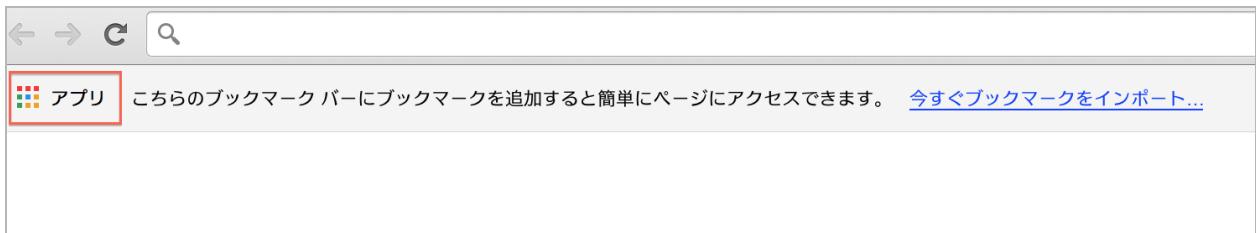
ローカル開発の手順を説明します。

3.1 Monaca Localkit

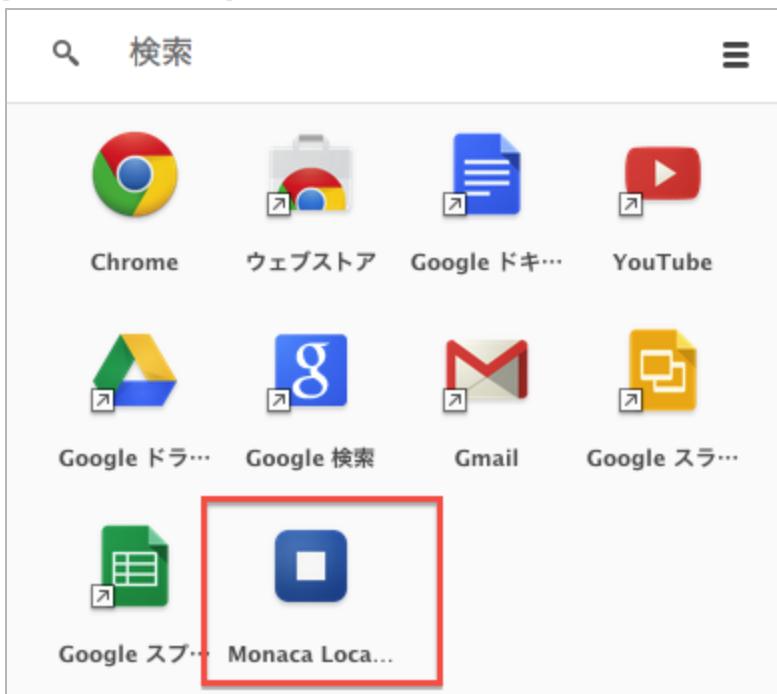
Monaca Localkitの起動

はじめに以下の手順でMonaca Localkitを起動します。

1. PCでGoogle Chromeを実行
2. Google Chromeのブックマークバーに表示される[アプリ]をクリック
(またはアドレスバーに chrome://apps/ を入力)



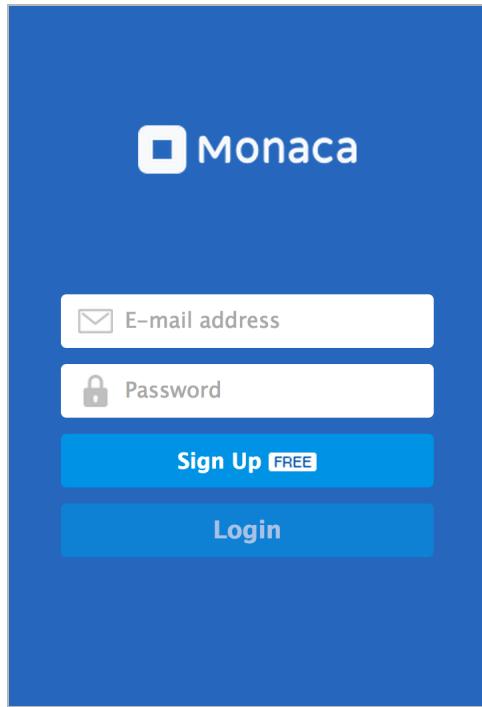
3. [Monaca Localkit]をクリック



ログイン

Monaca Localkitを起動するとログイン画面が表示されます。

※前回ログイン済みの場合は自動的に再ログインします



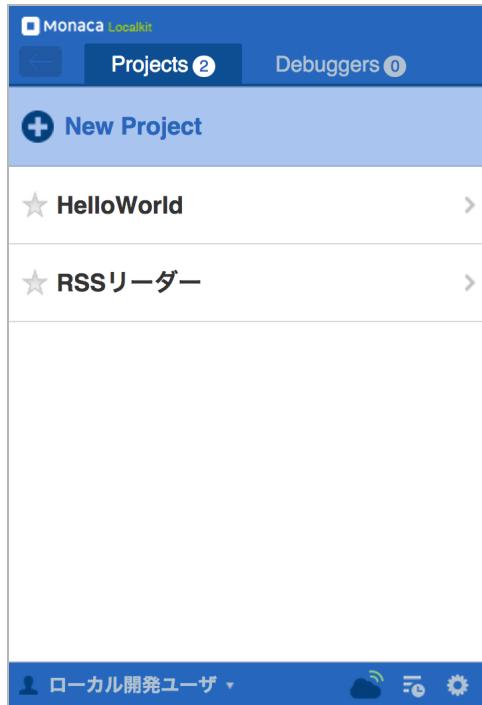
Monacaアカウントとして登録されたEメールアドレスとパスワードを入力し、ログインを行ってください。

Monacaアカウントをお持ちでない場合は、サインアップページでアカウントの新規登録を行ってください。

([Sign Upボタン] をクリックするとMonacaのサインアップページが表示されます)

プロジェクト設定

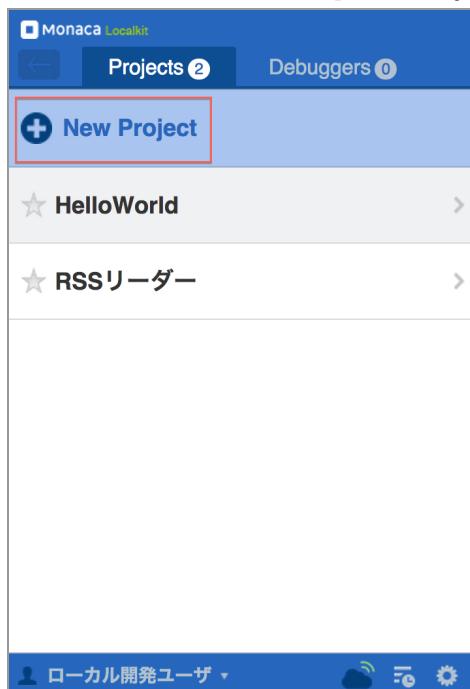
ログインするとプロジェクト一覧画面が表示されます。



- 新規プロジェクトを作成する場合

以下は新しいプロジェクトを作成する手順です。

1. プロジェクト一覧画面で[New Project]を選択



2. プロジェクトのテンプレートを選択

Add New Project

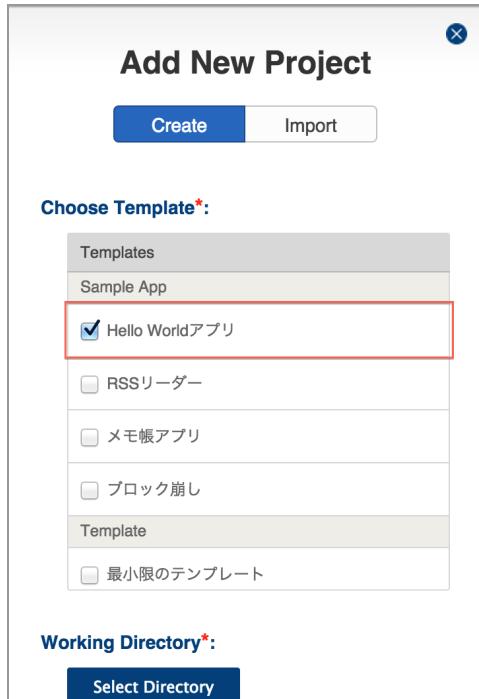
Create **Import**

Choose Template*:

Templates
Sample App
<input checked="" type="checkbox"/> Hello Worldアプリ
<input type="checkbox"/> RSSリーダー
<input type="checkbox"/> メモ帳アプリ
<input type="checkbox"/> ブロック崩し
Template
<input type="checkbox"/> 最小限のテンプレート

Working Directory*:

Select Directory



3. [Select Directory]をクリックし、作業ディレクトリを選択

Onsen UI Tabbar

Onsen UI Master-Detail

Working Directory*:

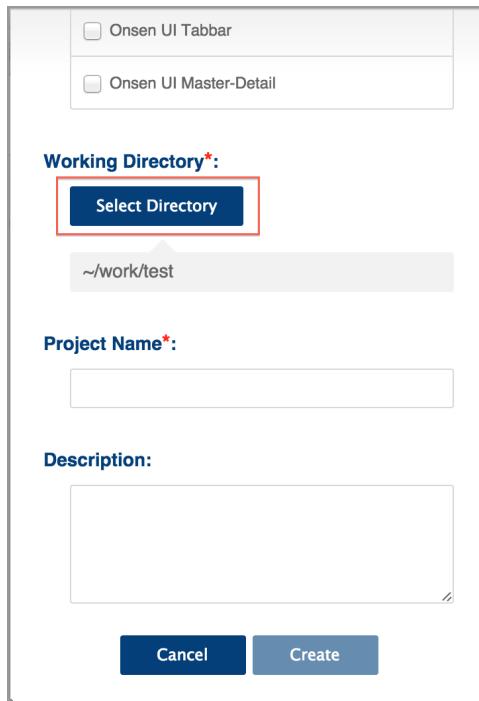
Select Directory

~/work/test

Project Name*:

Description:

Cancel **Create**



4. [Project Name]を入力し、[Create]をクリック

Working Directory*:
Select Directory
~/work/test

Project Name*:
Test Project

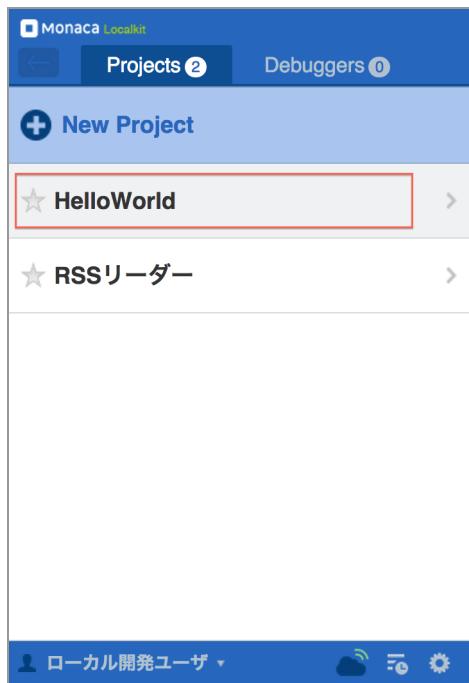
Description:

Cancel **Create**

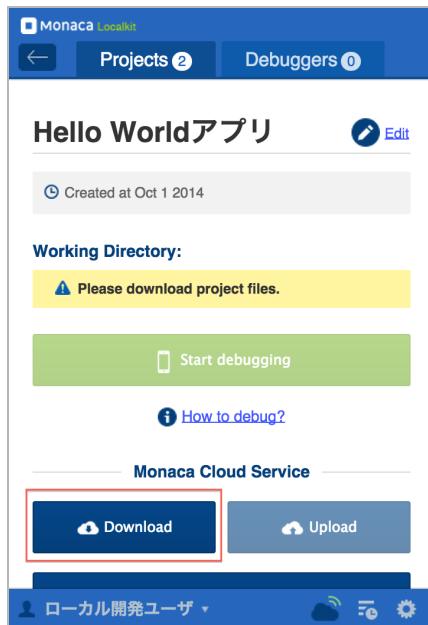
- Cloud上で作成した既存プロジェクトの場合

以下はCloud上で作成された既存のMonacaプロジェクトを開発対象として設定する手順です。

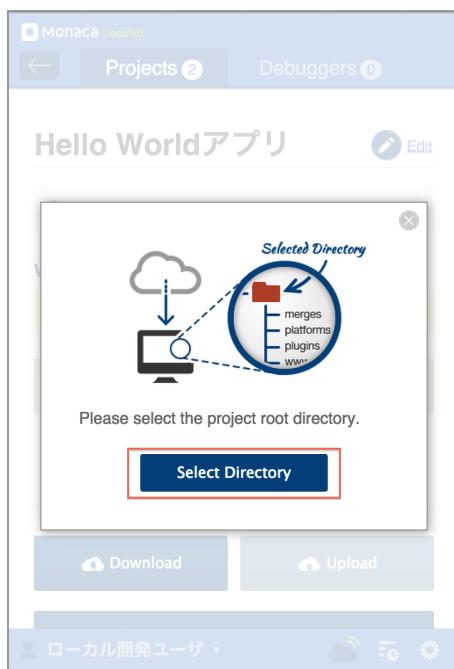
- プロジェクトを選択



2. プロジェクトの詳細画面で[Download]ボタンをクリックします



3. プロジェクトファイルのダウンロード先となるディレクトリを選択

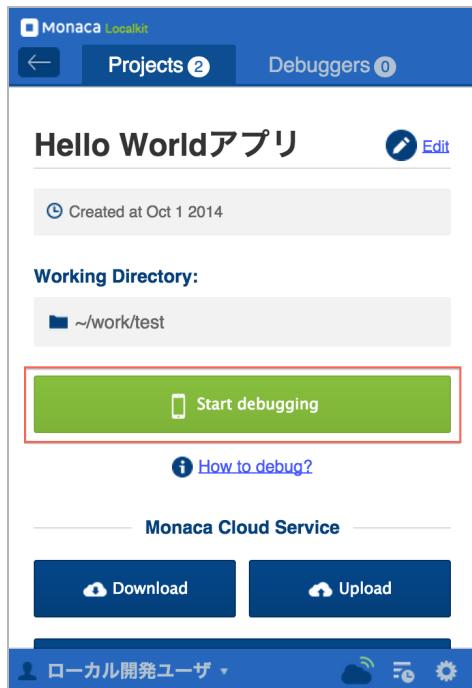


4. ディレクトリを選択すると、自動的にプロジェクトファイルがダウンロードされます

プロジェクト配信

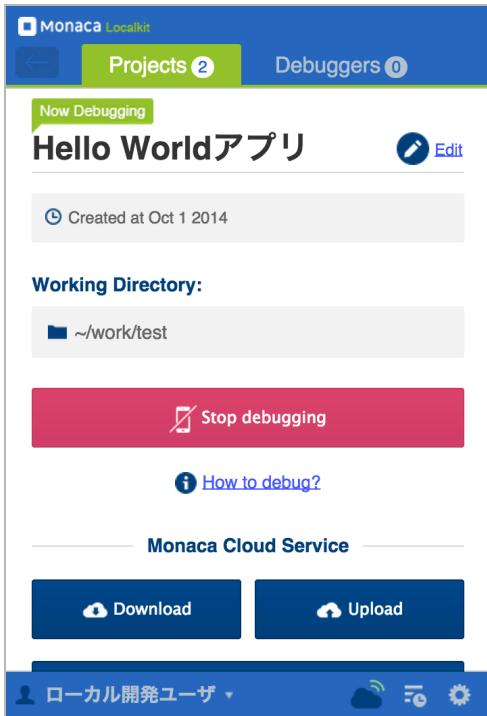
プロジェクトの詳細画面で[Start Debugging]ボタンをクリックすると、Monaca Localkitがプロジェクトの配信を開始します。

プロジェクト未配信状態



The screenshot shows the Monaca Localkit interface. At the top, there are tabs for 'Projects' (2) and 'Debuggers' (0). Below that, the project 'Hello World' is listed with a 'Edit' button. A message 'Created at Oct 1 2014' is shown. The 'Working Directory' is set to '~/work/test'. A prominent green button with the text 'Start debugging' is highlighted with a red box. Below it is a link 'How to debug?'. Under the heading 'Monaca Cloud Service', there are 'Download' and 'Upload' buttons. At the bottom, there is a user profile 'ローカル開発ユーザ' and other icons.

プロジェクト配信中



3.2 デバッガー

デバッグに使用するモバイル端末でMonacaデバッガーを起動してください。

ログイン

デバッガー起動するとログイン画面が表示されます。

※前回ログイン済みの場合は自動的に再ログインします

必ず**Monaca Localkitにログインしたユーザと同じユーザ**でログインを行ってください。

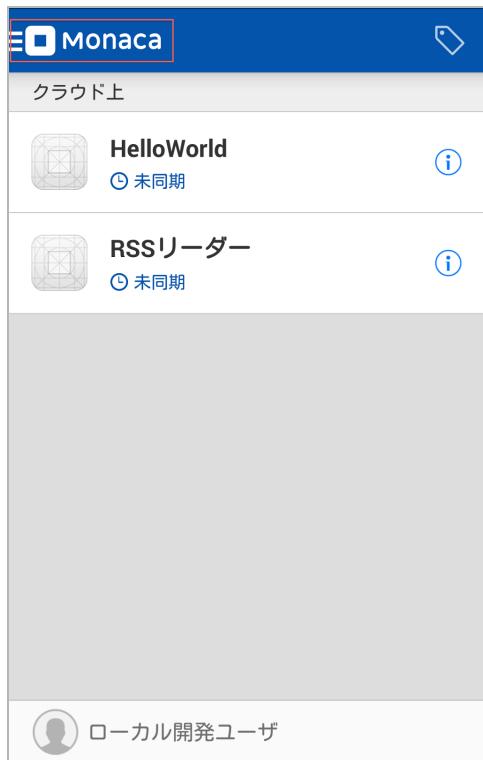


Monaca Localkitとデバッガーのペアリング

ログインするとプロジェクト一覧画面が表示されます。

以下の手順でMonaca Localkitとデバッガーのペアリングを行ってください。

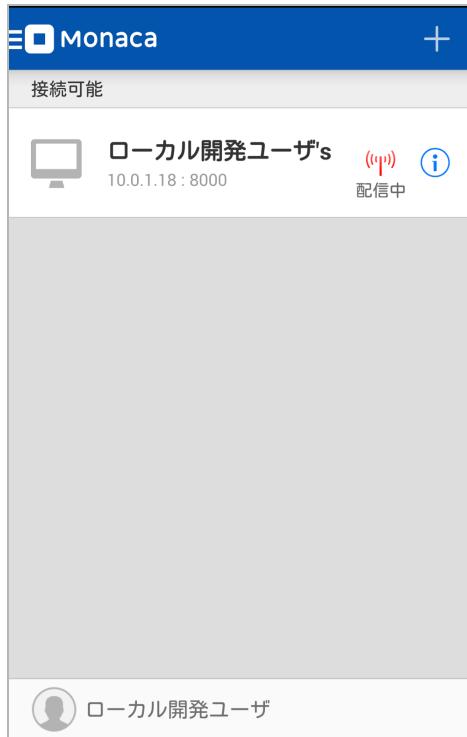
1. プロジェクト一覧画面で画面左上のボタンをクリックし、サイドメニューを表示してください



2. サイドメニューから[ローカルPCs]を選択します



3. 接続可能なサーバの一覧が表示されます



※リストに[配信中]が表示されない場合は、**3.1 Monaca Localkit**の「プロジェクト配信」を行ってください

4. リストから接続可能なPCを選択します

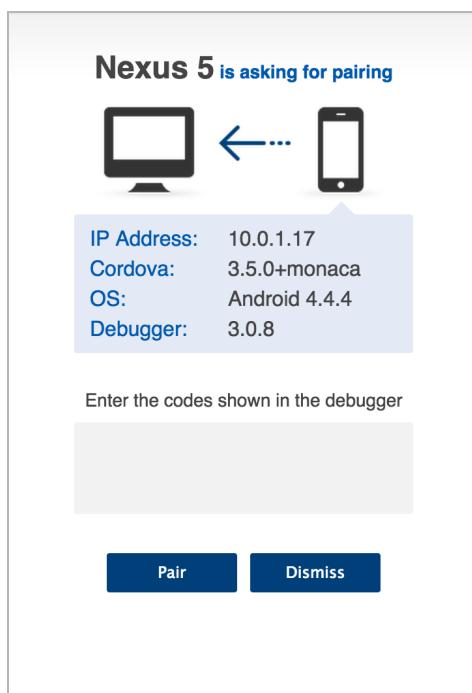


5. デバッガーでは6桁のペアリングコードが表示され、Monaca Localkitではペアリングコード入力画面が表示されます

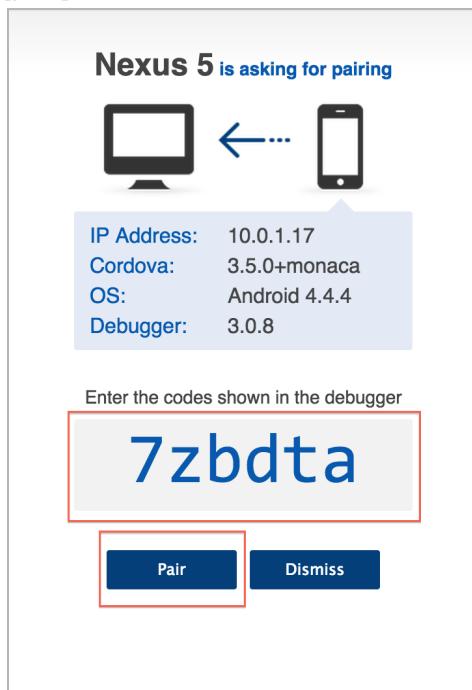
デバッガー



Monaca Localkit



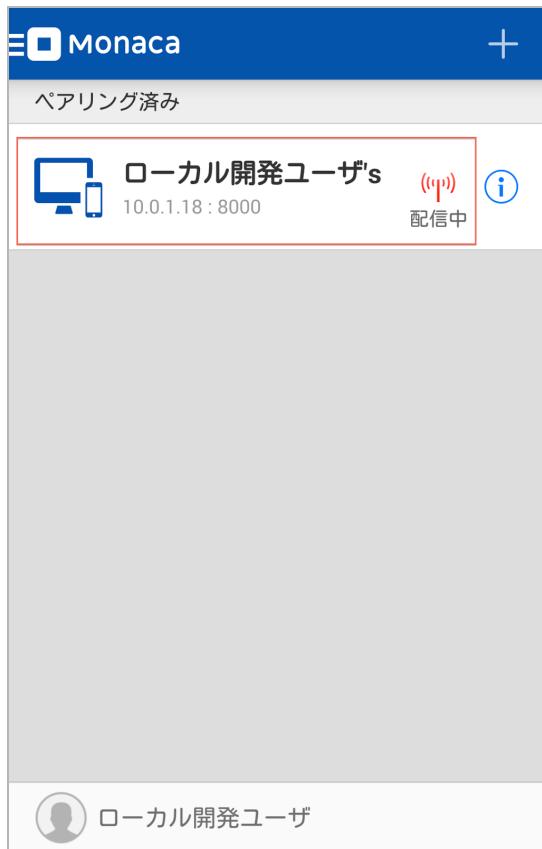
6. Monaca Localkitにデバッガーで表示されたペアリングコードを入力し、[pair]ボタンをクリック



以上でデバッガーとMonaca Localkitのペアリングが完了します。

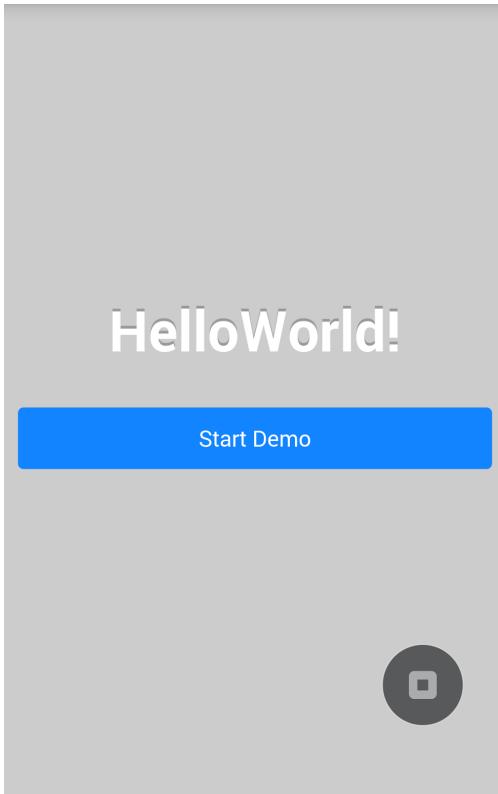
デバッガーでのプロジェクト実行

デバッガーでリストからペアリング済みプロジェクトを選択すると、プロジェクトファイルのダウンロードが行われます。



ダウンロードが完了するとプロジェクトが実行されます。

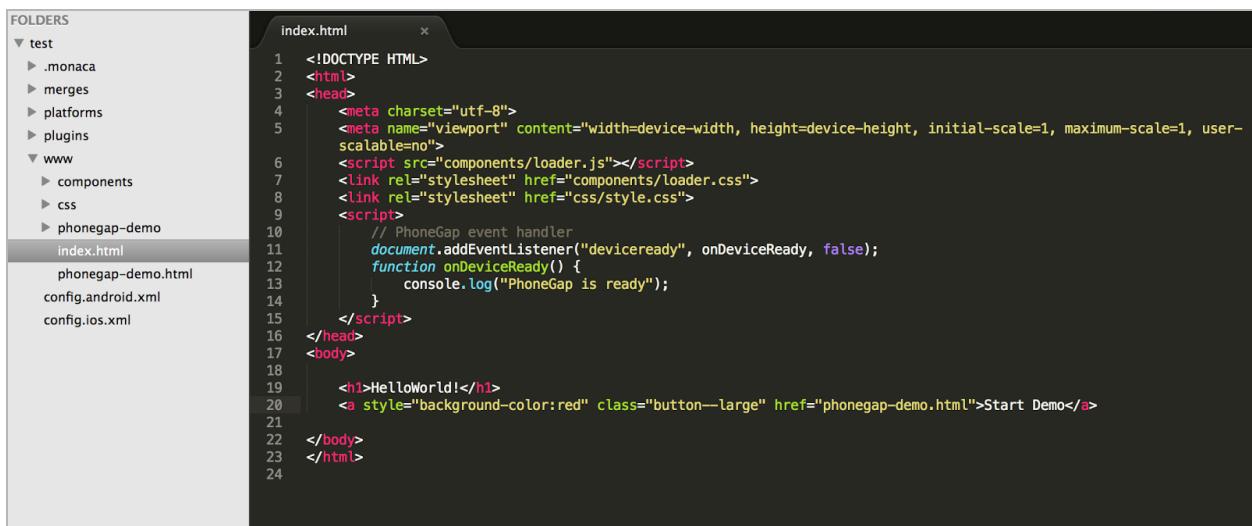
プロジェクト実行画面



3.3 ローカル開発

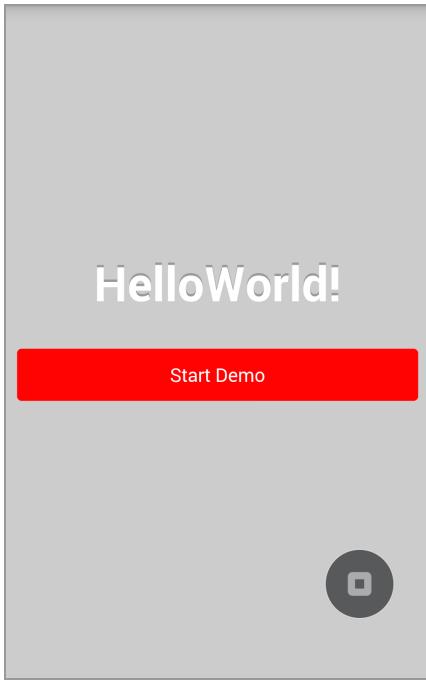
プロジェクトに設定した作業ディレクトリ以下のファイルを任意のツールで編集して保存すると、デバッガーに変更が即座に同期され、プロジェクトが再実行されます。

テキストエディタでボタンの背景色を赤色に変更して保存



```
index.html
1  <!DOCTYPE HTML>
2  <html>
3  <head>
4      <meta charset="utf-8">
5      <meta name="viewport" content="width=device-width, height=device-height, initial-scale=1, maximum-scale=1, user-scalable=no">
6      <script src="components/loader.js"></script>
7      <link rel="stylesheet" href="components/loader.css">
8      <link rel="stylesheet" href="css/style.css">
9      <script>
10         // PhoneGap event handler
11         document.addEventListener("deviceready", onDeviceReady, false);
12         function onDeviceReady() {
13             console.log("PhoneGap is ready");
14         }
15     </script>
16 </head>
17 <body>
18
19     <h1>HelloWorld!</h1>
20     <a style="background-color:red" class="button--large" href="phonegap-demo.html">Start Demo</a>
21
22 </body>
23 </html>
24
```

デバッガーでは変更されたファイルが同期され、プロジェクトが再実行される



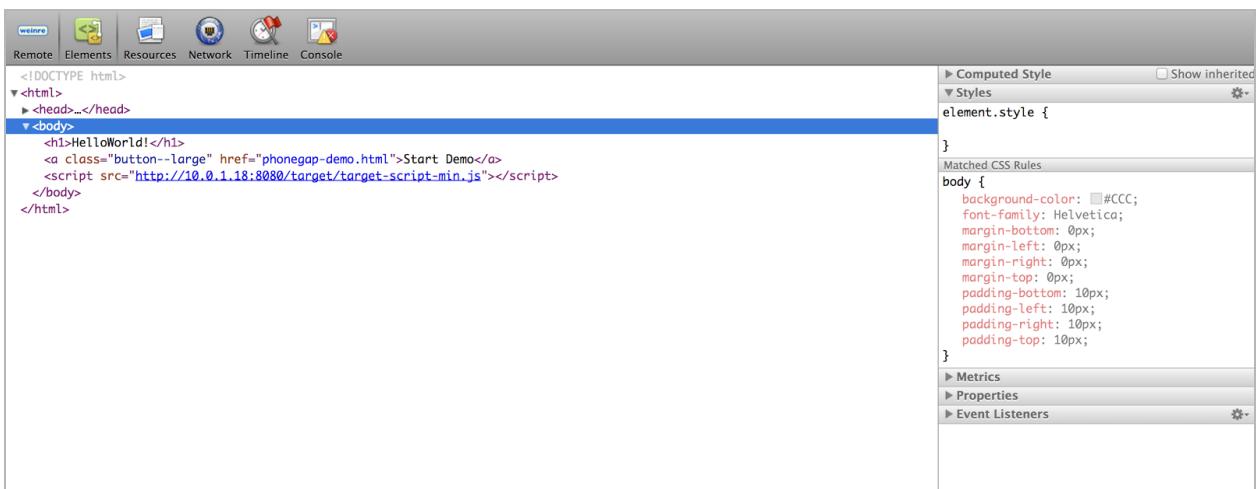
3.4 高度な開発

- **weinreを利用したデバッグ(Windows/Mac)**

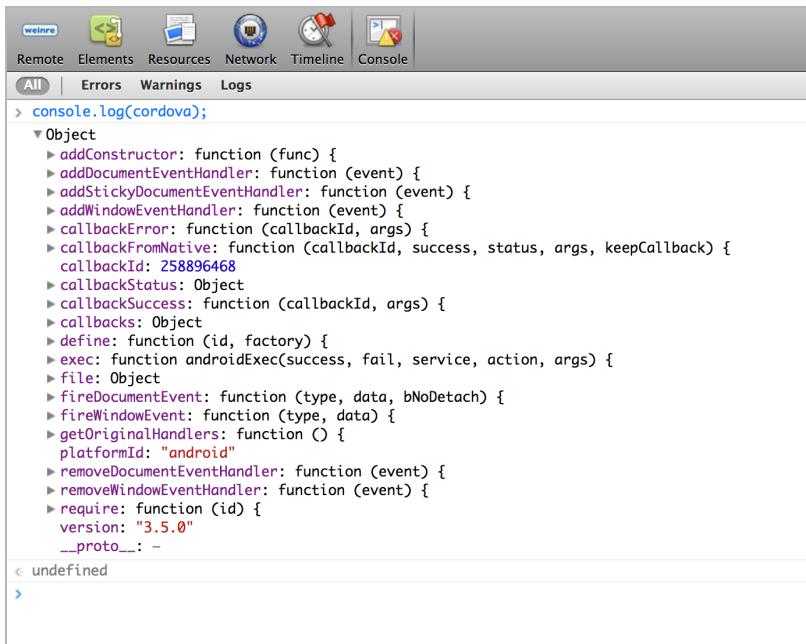
[weinre](#)とは「WEb INspector REmote」の略で、リモートデバッグ機能を提供しているツールです。

[weinre](#)を使うと、DOMインスペクタでCSSの適用を確認したり、コンソールでjavascriptを実行することができます。

DOMインスペクタ



コンソール



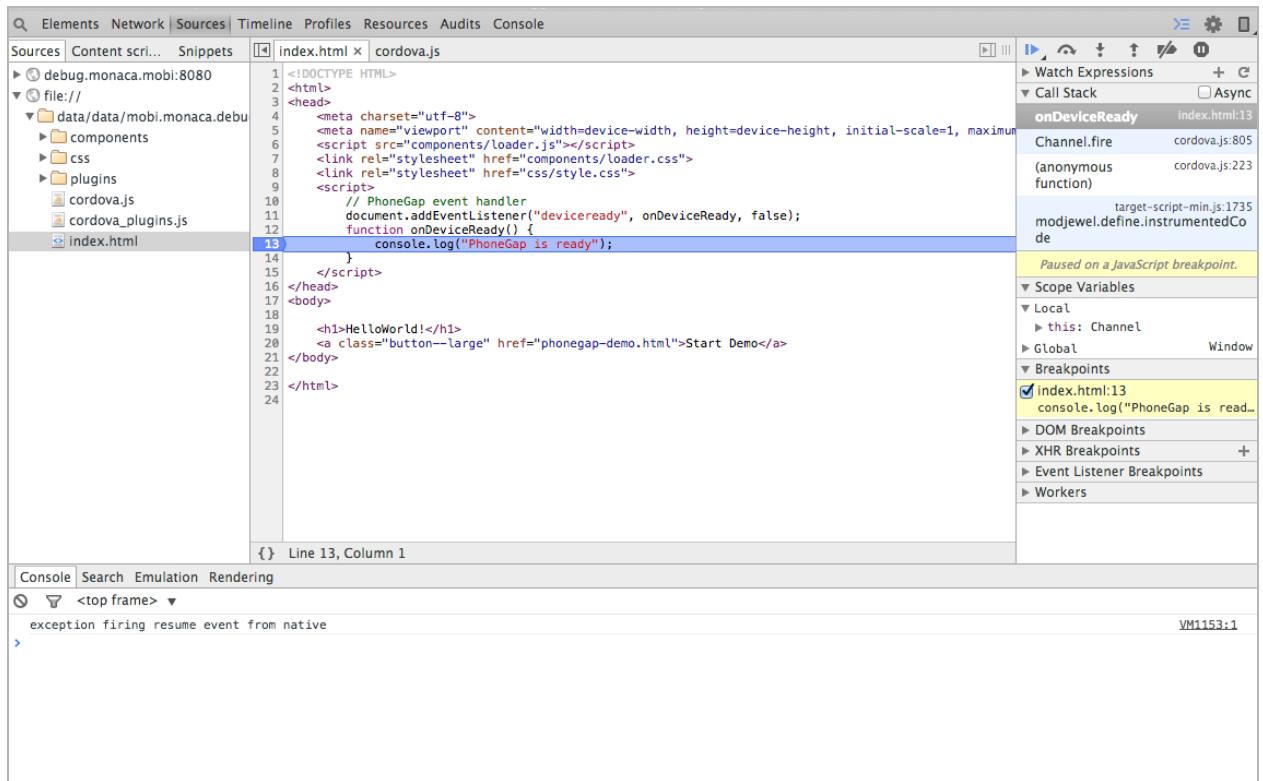
The screenshot shows the Chrome Dev Tools interface with the 'Console' tab selected. The output of the command `console.log(cordova);` is displayed, showing the structure of the `cordova` object. The object has properties like `addConstructor`, `addDocumentEventHandler`, `addStickyDocumentEventHandler`, `addWindowEventHandler`, `callbackError`, `callbackFromNative`, `callbackStatus`, `callbackSuccess`, `callbacks`, `define`, `exec`, `file`, `fireDocumentEvent`, `fireWindowEvent`, `getOriginalHandlers`, `platformId`, `removeDocumentEventHandler`, `removeWindowEventHandler`, `require`, and `version`. The `platformId` is set to "android". The `__proto__` property is also shown.

```
> console.log(cordova);
▼ Object
  ► addConstructor: function (func) {
  ► addDocumentEventHandler: function (event) {
  ► addStickyDocumentEventHandler: function (event) {
  ► addWindowEventHandler: function (event) {
  ► callbackError: function (callbackId, args) {
  ► callbackFromNative: function (callbackId, success, status, args, keepCallback) {
    callbackId: 258896468
  ► callbackStatus: Object
  ► callbackSuccess: function (callbackId, args) {
  ► callbacks: Object
  ► define: function (id, factory) {
  ► exec: function androidExec(success, fail, service, action, args) {
  ► file: Object
  ► fireDocumentEvent: function (type, data, bNoDetach) {
  ► fireWindowEvent: function (type, data) {
  ► getOriginalHandlers: function () {
    platformId: "android"
  ► removeDocumentEventHandler: function (event) {
  ► removeWindowEventHandler: function (event) {
  ► require: function (id) {
    version: "3.5.0"
  ► __proto__: -
```

- **Chrome Dev Toolsを利用したリモートデバッグ(Androidデバッガーのみ)**
Google ChromeをインストールしたコンピュータとAndroidデバッガーをインストールした端末をUSB接続することで、Chrome Dev Toolsのリモートデバッグ機能を利用することができます。

Chrome Dev Toolsのリモートデバッグ機能は、weinreと同じようにDOMインスペクタ、コンソール、そしてJavascriptのステップ実行が可能です。

Javascriptステップ実行の例:



Chrome Dev Toolsを利用するAndroid OSのバージョンは
 ハイパフォーマンス版デバッガーの場合 : Android4.0以上
 通常WebView版デバッガーの場合 : Android4.4以上
 となります。

利用方法については[こちら](#)をご確認ください。

- **Safariを利用したリモートデバッグ(Mac+ iOS)**

Macをご利用の方は、カスタムビルド版デバッガーをインストールしたiOS端末とPCをUSB接続することで、MacのSafariを利用したリモートデバッグを行うことが可能です。

利用方法については[こちら](#)をご確認ください。

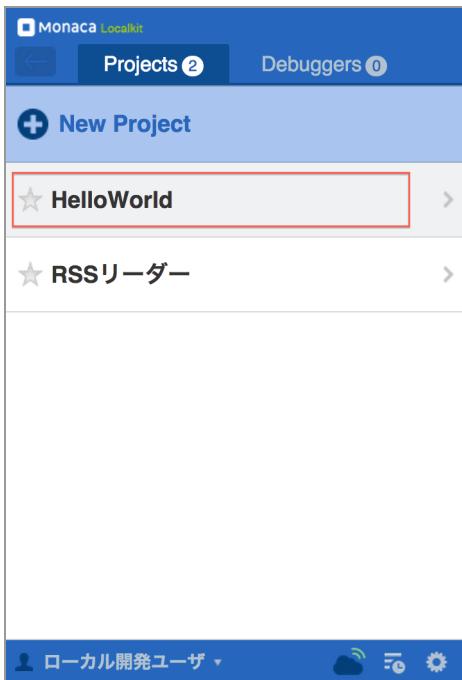
3.5 リモートビルト

Monacaでは、Cloud上のビルトサーバーを利用してiOS/Android/Windows8/ChromeAppsのアプリビルトを行うことができます。

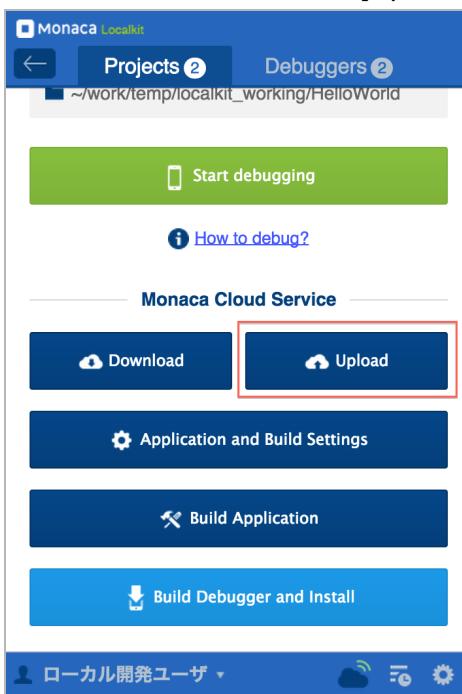
プロジェクトのアップロード

リモートビルトを行うために、ローカルで編集したプロジェクトをアップロードします。アップロードの手順は以下のとおりです。

- プロジェクト一覧からアップロードしたいプロジェクトを選択します



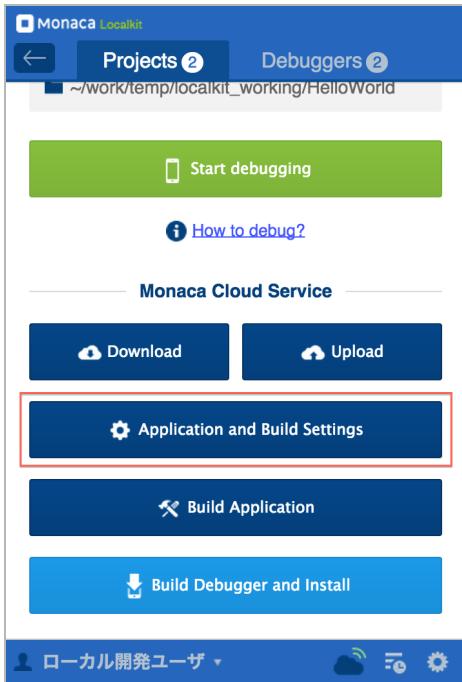
- プロジェクトの詳細画面で[Upload]ボタンをクリックします



アプリ設定とビルト設定

アップロードしたプロジェクトに対して、アプリの設定とビルトの設定 (iOS/Androidのみ) を行います。

- プロジェクトの詳細画面で[Application and Build Settings]をクリックします



2. 新しいウインドウにプロジェクトの設定画面が表示されます

The screenshot shows the 'Cordova Plugin Management' window. On the left, there are two tabs: 'Build' and 'Build Setting'. The 'Build Setting' tab is selected, showing a sidebar with 'Project' (selected) and 'CordovaPlugin' (highlighted). Below the sidebar, there are sections for 'Build' (Androidアプリ, Androidビルド, iOSアプリ, iOSビルド, Windows 8, Chrome Apps) and 'JS/CSSコンポーネント'. The main area is titled 'Cordovaプラグインの管理' and lists several plugins with their status: 'appCCloud' (mobi.monaca.appccloud.AppCCloudPlugin) is '有効' (Enabled); 'Battery' (org.apache.cordova.battery-status) is '有効' (Enabled); 'Camera' (org.apache.cordova.camera) is '有効' (Enabled); 'Console' (org.apache.cordova.console) is '有効' (Enabled); 'Contacts' (org.apache.cordova.contacts) is '有効' (Enabled); and 'Device' (org.apache.cordova.device) is '未登録' (Not Registered). A blue button 'Cordovaプラグインのインポート' is located at the top right of the main area.

3. アプリの設定 (Androidアプリの場合)

画面左側の一覧から[Androidアプリ]を選択するとAndroidアプリ用の設定が表示されます。

Androidアプリ設定については[こちら](#)をご確認ください。



(iOSアプリの場合)

画面左側の一覧から[iOSアプリ]を選択するとiOSアプリ用の設定が表示されます。

iOSアプリ設定については[こちら](#)をご確認ください。



(Windows8 アプリの場合)

画面左側の一覧から [Windows8] を選択すると Windows8 ストア アプリ用の設定が表示されます。

Windows8 ストア アプリ設定については [こちら](#) をご確認ください。



(Chrome Appsの場合)

画面左側の一覧から[Chrome Apps]を選択するとChromeアプリ用の設定が表示されます。

Chromeアプリ設定については[こちら](#)をご確認ください。

Build Setting

Build

Project

Build

Chrome Apps

Chrome Apps 設定

アプリ設定

名前:

略称:

説明:

バージョン:

オフラインでも動作可能:

許可する外部URL:

アイコン:

保存する

4. ビルドの設定

(Androidアプリの場合)

画面左側の一覧から[Androidビルド]を選択するとAndroidアプリ用のビルド設定が表示されます。

Androidビルド設定については[こちら](#)をご確認ください。



(iOSアプリの場合)

画面左側の一覧から[iOSビルド]を選択するとiOSアプリ用のビルド設定が表示されます。

iOSアプリのビルド設定を行うためには[iOS Developer Program](#)の登録が必要となりますのでご注意ください。

iOSビルド設定については[こちら](#)をご確認ください。

ビルド設定 ビルド

iOSビルド設定

Project
Cordovaプラグイン
JS/CSSコンポーネント

Build
Androidアプリ
Androidビルド
iOSアプリ
iOSビルド
Windows 8
Chrome Apps

iOSビルド設定
iOSアプリケーションをビルドするために必要な項目を設定します。このページの設定はユーザー アカウントごとに有効になります。
?

秘密鍵とCSR
Monacaの開発者情報をiOS Provisioning Portalに登録するための秘密鍵を生成し、CSRを発行します。ユーザー名とメールアドレスには、iOS Dev Centerの登録情報と完全に一致している必要があります。

秘密鍵:

秘密鍵の登録名: -

CSR情報:

証明書のアップロード
iOS Provisioning Portalで発行された証明書をアップロードします。デベロッパー証明書はデバッグビルドに、ディストリビューション証明書は配布ビルドとリリースビルドに必要となります。

リモートビルド

アップロードしたプロジェクトのビルドを行います。

1. ビルド対象プロジェクトの詳細画面で[Build Application]をクリックします

Monaca LocalKit

Projects 2 Debuggers 2
~/work/temp/localkit_working/HelloWorld

Start debugging

How to debug?

Monaca Cloud Service

Download Upload

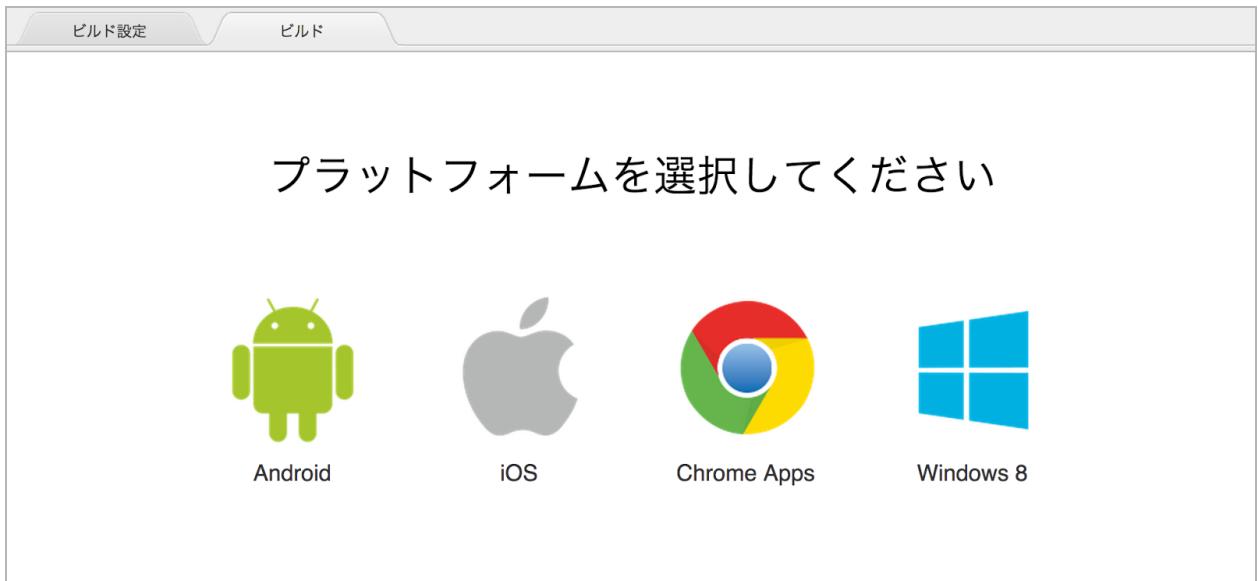
Application and Build Settings

Build Application (highlighted with a red box)

Build Debugger and Install

ローカル開発ユーザ

2. 新しいウインドウでビルド対象プラットフォームの選択画面が表示されます



3. 任意のプラットフォームを選択し、ウィザードに従ってビルドを実行してください



4. ビルドが完了すると、アプリのパッケージをダウンロードすることができます



4. 注意事項（免責事項）

- 利用上の注意点

本ソフトウェアはベータ版です。

正式配布版の前段階の評価版として関係者の方々に配布し、性能や機能、使い勝手などを評価して頂くバージョンとなっております。

従って、正式版の機能を一通り備えた完成品に近い状態ではありますが、ソフトウェアの不具合などにより正常に機能しない場合があります。注意事項をご理解の上、本ソフトウェアをご利用ください。

本ソフトウェアの使用において生ずる損害については、弊社は一切の責任を負うものではありません。

5. お問い合わせ

お問い合わせにつきましては[Monacaサポートサービス](#)をご利用ください。