



# ゼロから始めるモバイルアプリ開発

## AWS Mobile Hubハンズオン

### 【事前準備編】

2016年01月16日 JAWS-UG沖縄

西島 幸一郎 @k\_nishijima

ゼロから始めるモバイルアプリ開発 /  
**AWS Mobile Hubハンズオン**  
にご参加（表明）いただき  
ありがとうございます！



**JAWS-UG**

AWS User Group - Japan

**OKINAWA**

<http://jaws-ug.jp/>

GitHub repositories <https://github.com/jaws-ug/>

こちらの資料は「事前準備編」です。  
色々と時間がかかることが多いので、  
出来ればイベント前に自宅で  
やってきてください！



勿論分からぬところや  
詰まった所があれば、  
前日まで or 当日に  
お気軽にご質問ください！



JAWS-UG

AWS User Group - Japan

OKINAWA

<http://jaws-ug.jp/>

Github repositories <https://github.com/jaws-ug/>

本資料は

<http://bit.ly/handson201601-setup>  
よりダウンロード可能です。

GitHubリポジトリはこちら。

[https://github.com/jaws-ug/hands-on/  
tree/master/Mobile-Hub](https://github.com/jaws-ug/hands-on/tree/master/Mobile-Hub)

# あんた誰？

西島 幸一郎 / にしじま こういちろう

アルスリーインスティテュート ソリューションアーキテクト

<https://www.r3it.com>

JAWS-UG沖縄のコアメンバー

AWSサムライ2013/2014 2年連続拝命

当資料について、ご質問などあればFacebook/Twitterなどで  
お気軽にお問い合わせください！



@k\_nishijima



nishijima.koichiro



# アールスリーインスティテュート

大阪の業務系システム開発会社

AWSとkintoneで「ハイスピードSI」

西島は沖縄から100%リモートワーク

【コミュニティにフルコミットする】と  
宣言している珍しい会社



We are hiring!



JAWS-UG

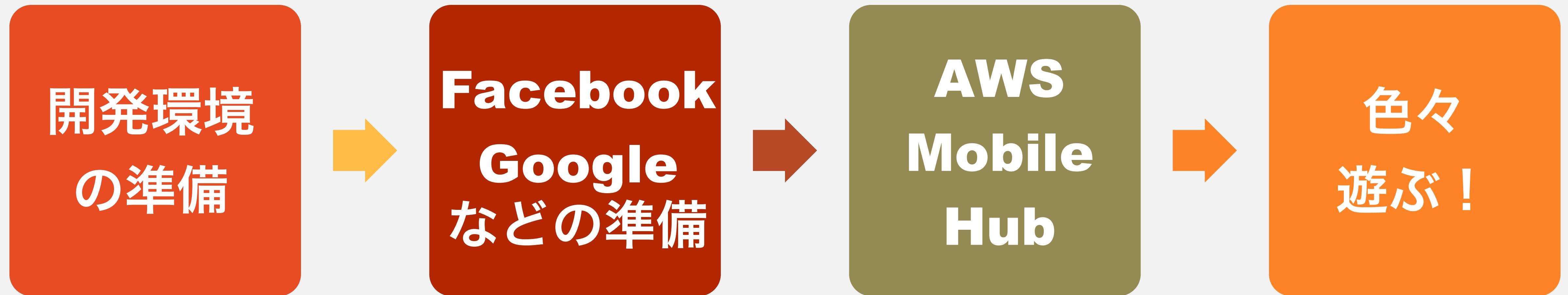
AWS User Group - Japan

OKINAWA

<http://jaws-ug.jp/>

GitHub repositories <https://github.com/jaws-ug/>

# ハンズオンの流れ



# ハンズオンの流れ



この辺りが結構時間がかかるので、事前準備編が誕生しました(^^;)



**JAWS-UG**

AWS User Group - Japan

**OKINAWA**

<http://jaws-ug.jp/>

GitHub repositories <https://github.com/jaws-ug/>



# ご注意！

- ◆ 当資料は**Android**向けかつ**Mac**で作成されました
- ◆ ただし**Windows**でもほぼ同じ、**iOS**向けもかなり似たような感じだと思います。
- ◆ **MobileHub**自体は**iOS**および**Windows**での開発もサポートしています。
- ◆ 今回は **iOS**向けではなく**Android**向け資料ですが、きっと将来拡張される予定です（プルリクエストお待ちしております！）
- ◆ **Windows**環境向けのサポートも同じく・・・  
(当方は窓環境が全く無いのでサポートできません・・・)



# 開発環境の準備一覧

- ◆ **Android Studioの準備**
- ◆ **Androidエミュレーターの設定**
- ◆ **Android実機の接続**



**JAWS-UG**

AWS User Group - Japan

**OKINAWA**

<http://jaws-ug.jp/>

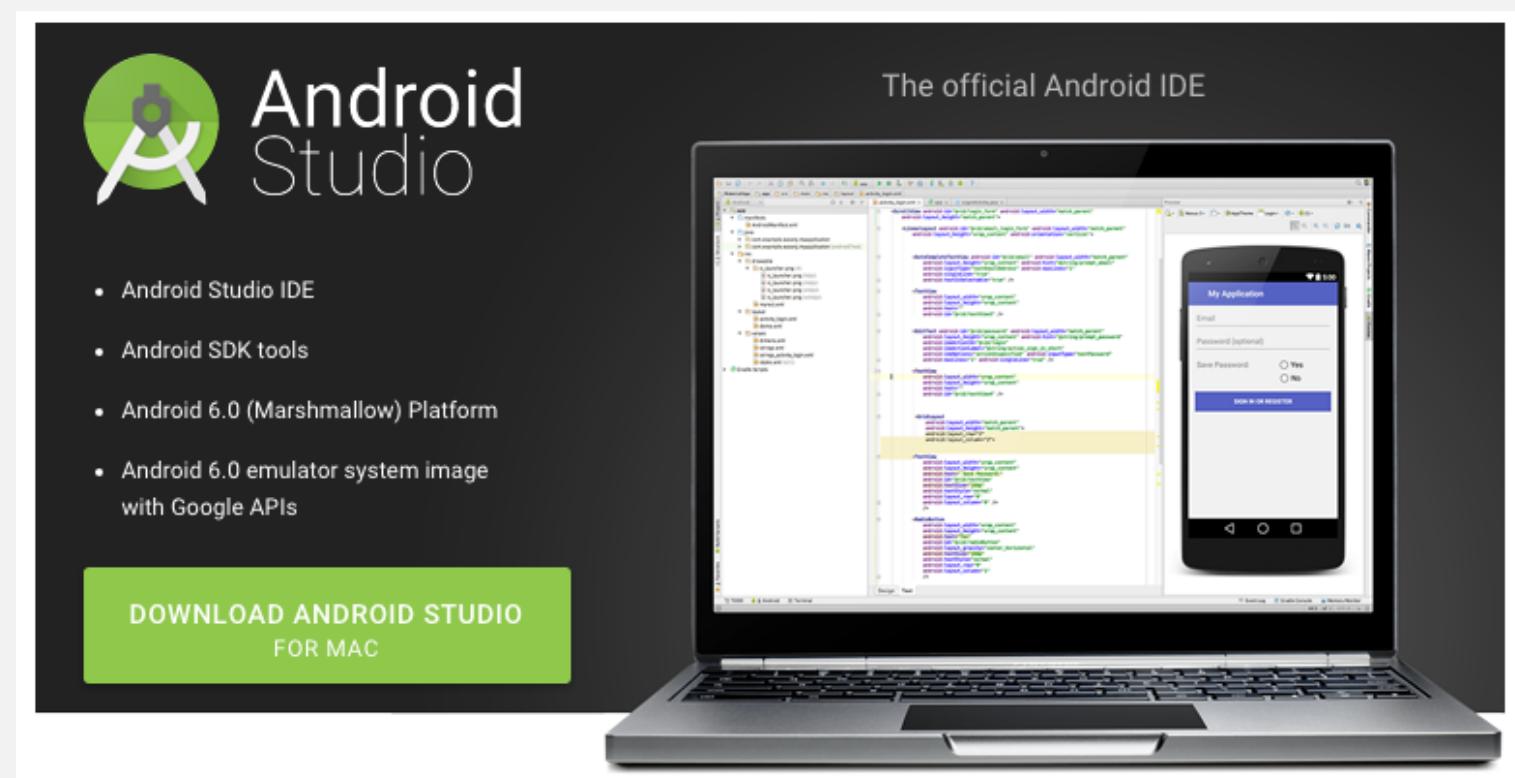
GitHub repositories <https://github.com/jaws-ug/>

# Android Studioの準備

◆ ダウンロードはこちらです。

<http://developer.android.com/sdk/>

◆ ダウンロードが終わったら、  
そのファイルをダブルクリックして  
インストール後、起動してください。



JAWS-UG

AWS User Group - Japan

OKINAWA

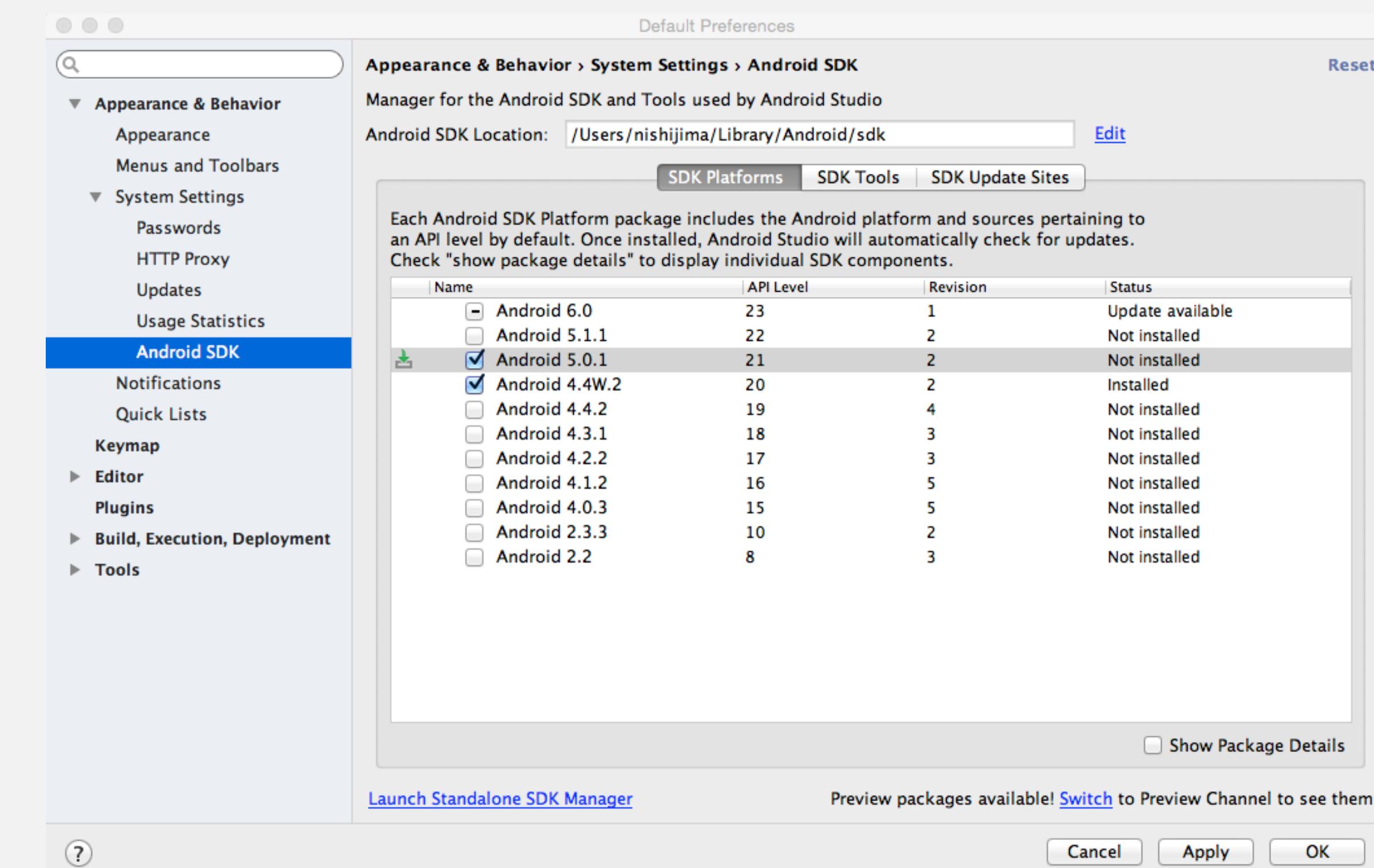
<http://jaws-ug.jp/>

Github repositories <https://github.com/jaws-ug/>

# Android SDKインストール

**メニューのTools -> Android -> SDK Manager -> Appearance & Behavior -> System Settings -> Android SDK と選択**

- ◆ **Android SDK**は、ご自分の持ってる端末のバージョンを確認して、それにチェックを入れます。
- ◆ 分からない・持っていないければひとまず**6.0**をチェック。
- ◆ 右下の**OK**ボタンでインストールが始まります。



# Android SDKインストール

♦ 待ち時間は結構長いので、焦らずにのんびり待ちましょう。

## Installing Requested Components

SDK Path: /Users/nishijima/Library/Android/sdk

Loading SDK information...

Installing Archives:

Preparing to install archives

Installing SDK Platform Android 5.0.1, API 21, revision 2

97%, 6885 KiB/s, 0 seconds left



JAWS-UG

AWS User Group - Japan

OKINAWA

<http://jaws-ug.jp/>

GitHub repositories <https://github.com/jaws-ug/>

# Android Virtual Device(AVD)の準備

- ◆ **Android**アプリの開発では、色々なスペックを持つ端末を1台の開発マシンで出来るよう  
に、**Android Virtual Device(AVD)**という仕組みがあります。
- ◆ 実端末がなくても、どんなスペックの端末でも仮想的に作り出せて、便利♪・・・と思う  
のですが、このエミュレーターが滅茶苦茶起動が遅くて（大  
体5分位？）涙が出てきます。
- ◆ 一旦起動すれば我慢できますが・・・



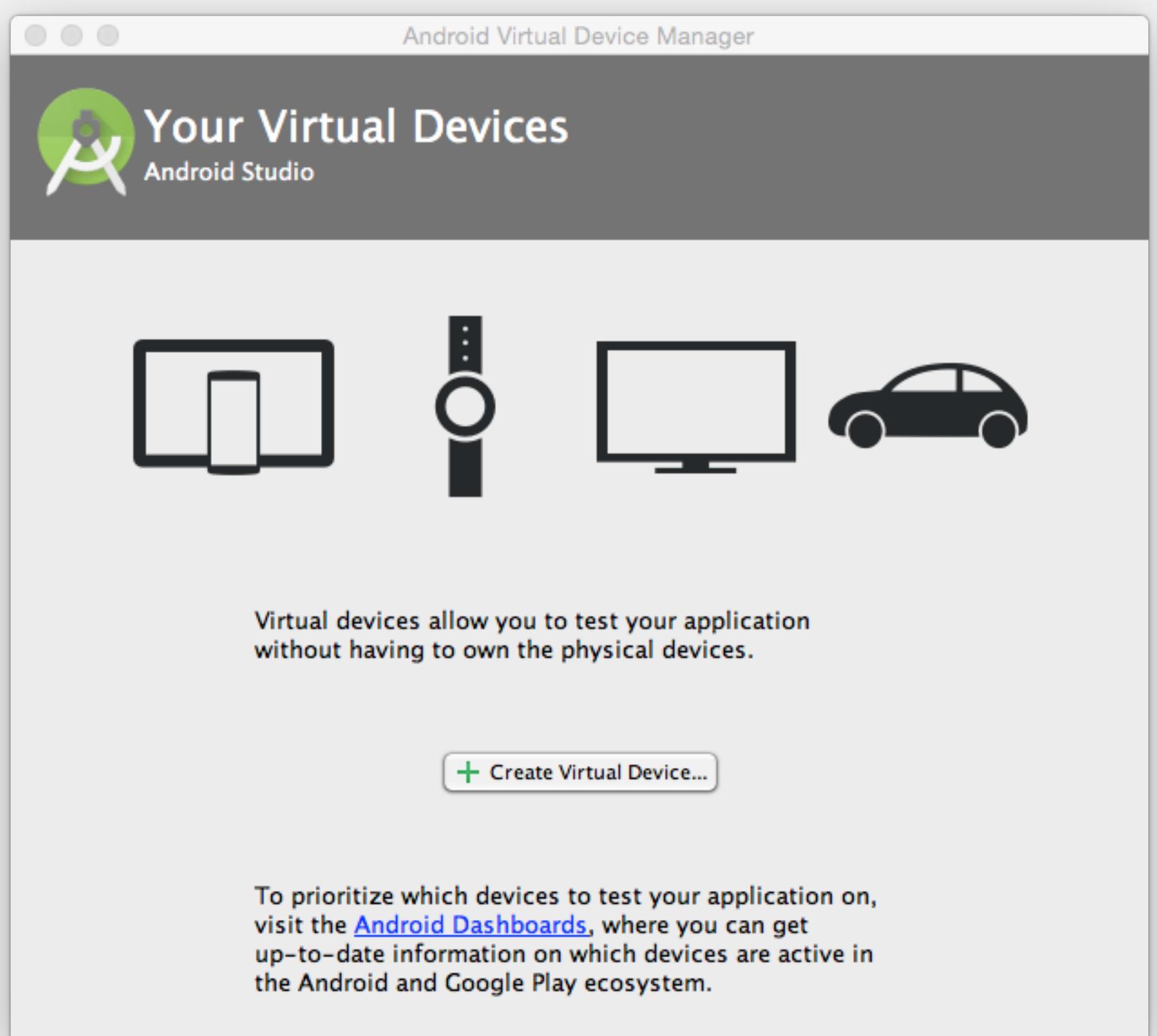
# Android Virtual Device(AVD)の準備

- ♦ 実機がある人は、実機をそのまま開発に利用すれば、無理してAVDを使う必要はありません（デバッグは辛いですが・・・）。
- ♦ 自分の所有端末を超えた多端末対応するには、当然必要となってきます。
- ♦ 設定するのが面倒そうだな、と思った方はこの項目はスキップして構いません。
- ♦ 起動を少しだけ早くする方法も載せておきますので、AVDを使う方は是非この方法も設定しておいてください。

## Android Virtual Device(AVD)の準備

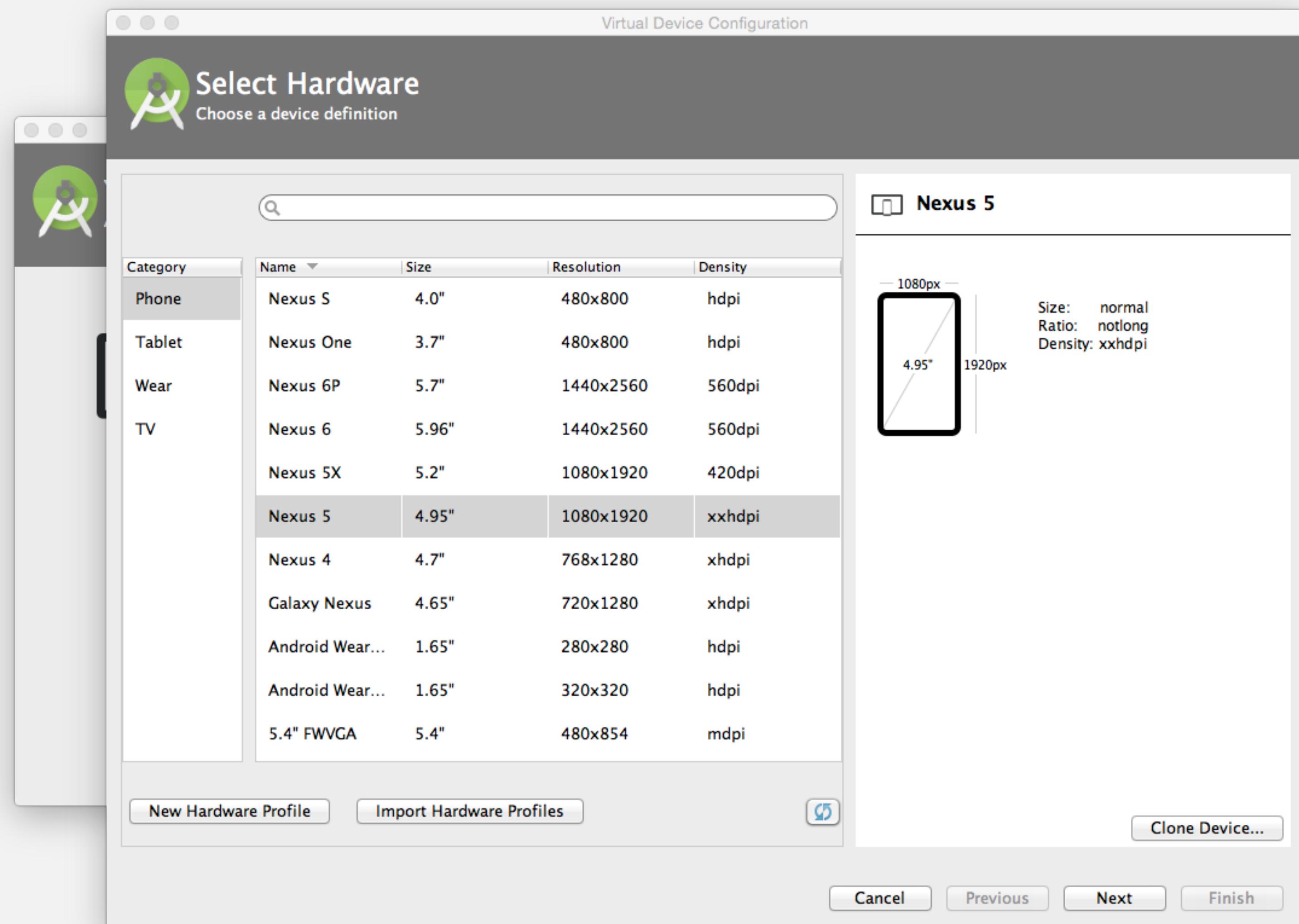
メニューのTools -> Android -> AVD Managerと選択

- ◆ タブレットから時計、車まで物理デバイスがなく  
てもテストできるぜ！って、はい、そのとおりで  
ございます。
- ◆ **Create Virtual Device** をクリックします。



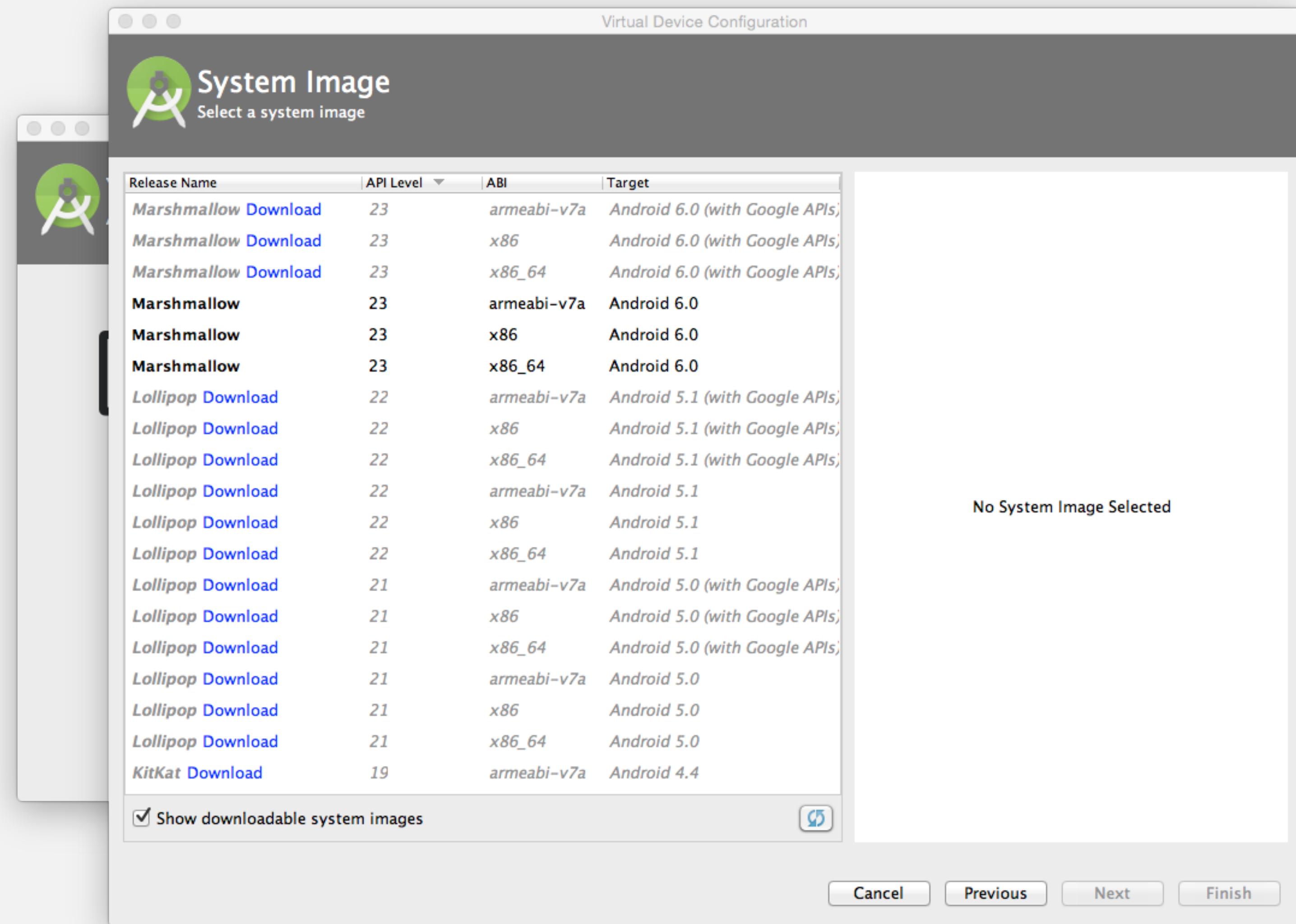
# Android Virtual Device(AVD)の準備

- ◆ 自分が持っている端末を選択して **Next**で次に進みます。
- ◆ 分からなければ、スクリーン解像度で適当に選ぶか、**Nexus 5**辺りを選んでおいてください。



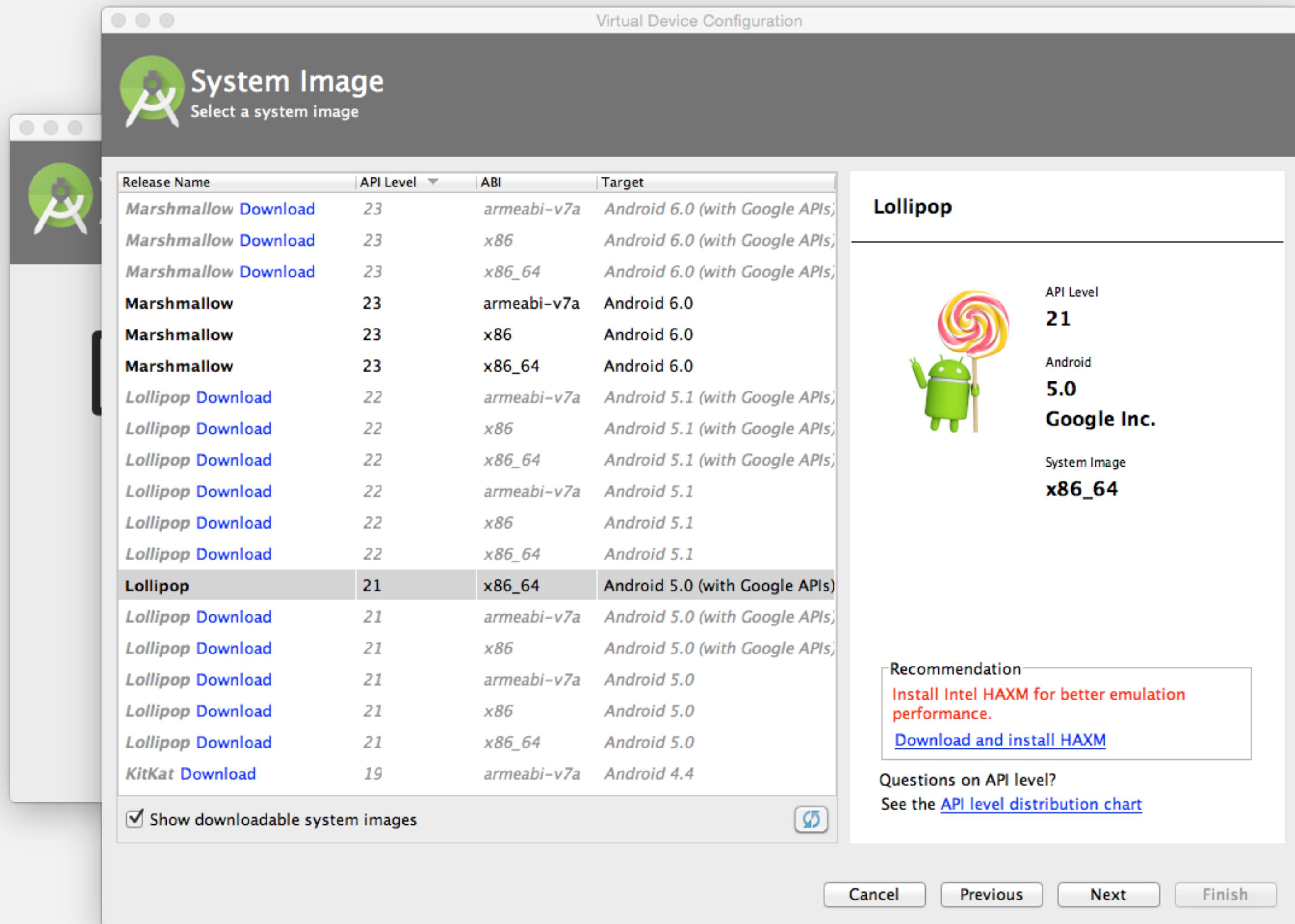
# Android Virtual Device(AVD)の準備

- ◆ 次はシステムイメージの選択画面になります。
- ◆ **Androidは1端末に複数OSバージョンがインストールできるので、こうなっています。**
- ◆ 自分の持ってるバージョンがない場合、下の「**Show downloadable system images**」にチェックを入れると出できます。



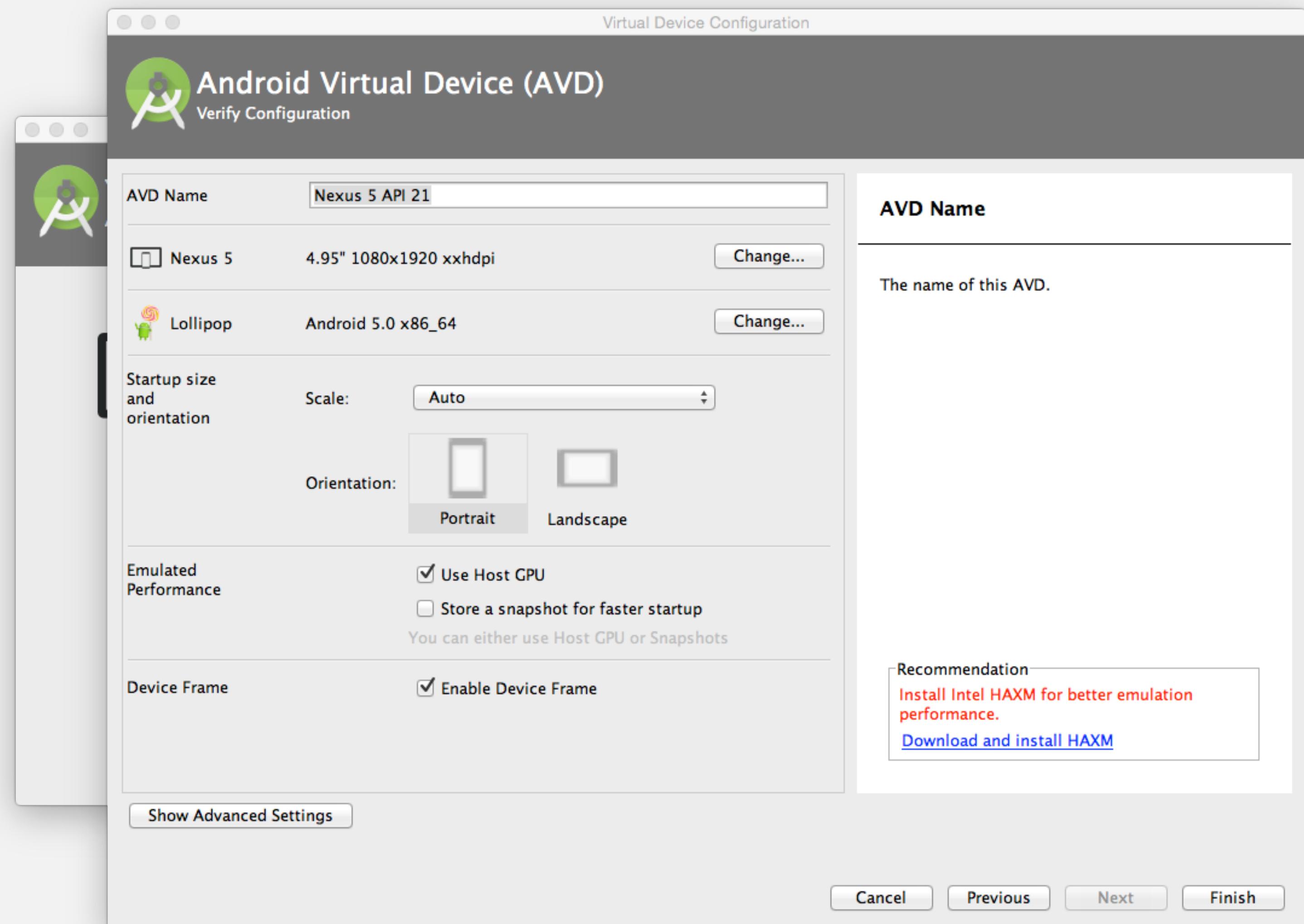
# Android Virtual Device(AVD)の準備

- ◆ 名前の横の**Download**をクリックでダウンロードされます。これも長いので気長に待ちましょう。
- ◆ APIレベルはOSのメジャーバージョンとほぼ同義です。
- ◆ ABI列は最近のOSなら**x86\_64**を選べば良いと思います。
- ◆ インストールが終わったら、**Next**で次に進みます。



# Android Virtual Device(AVD)の準備

- ❖ 最後に名前を入れたり色々設定できますが、デフォルトのまま「Finish」をクリックで大丈夫だと思います。



JAWS-UG

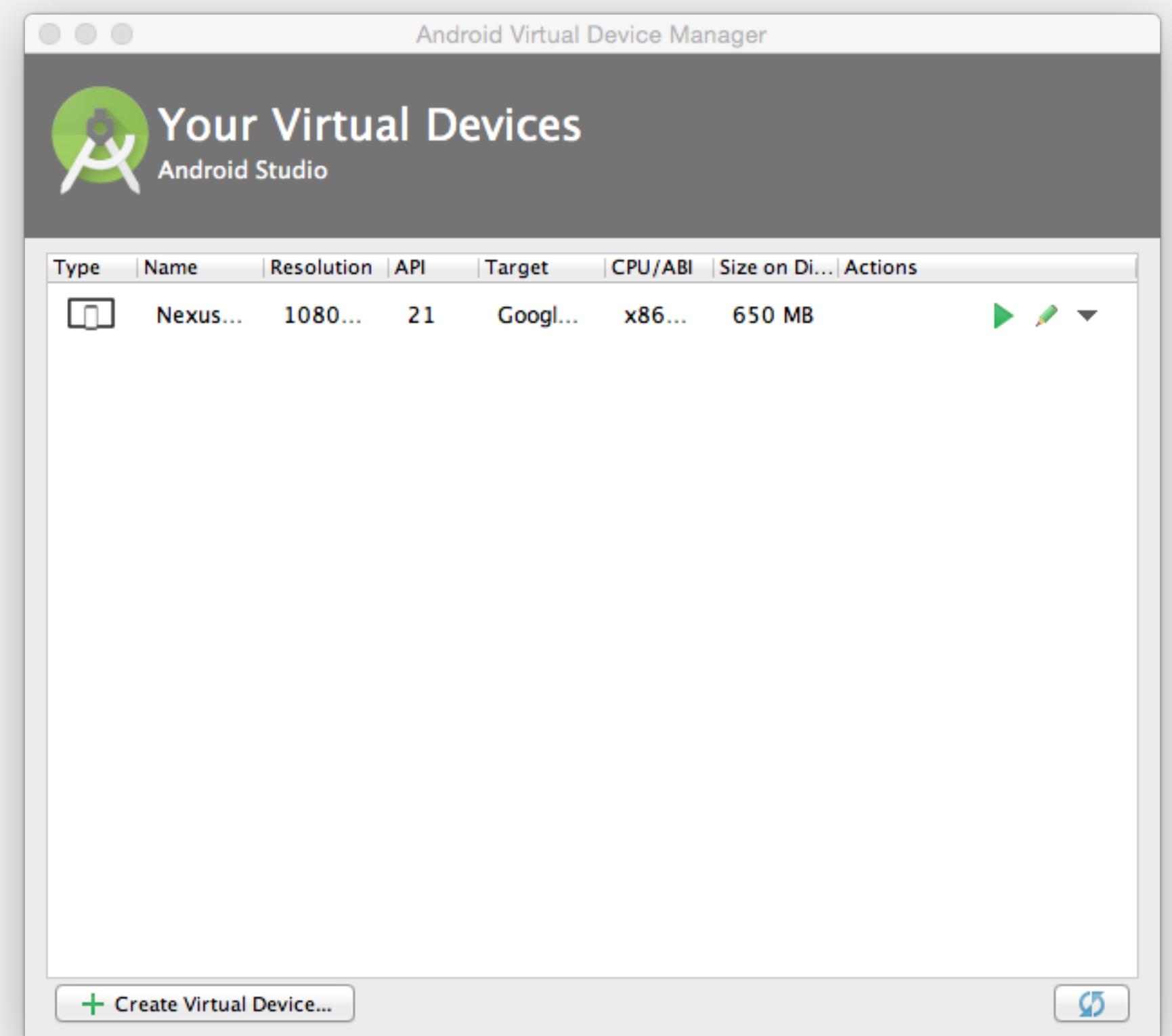
AWS User Group - Japan

OKINAWA

<http://jaws-ug.jp/>  
GitHub repositories <https://github.com/jaws-ug/>

# Android Virtual Device(AVD)の準備

- ◆ 一覧に出てくるようになります。
- ◆ 次から、この仮想デバイスを使って動作テストが出来るようになります。

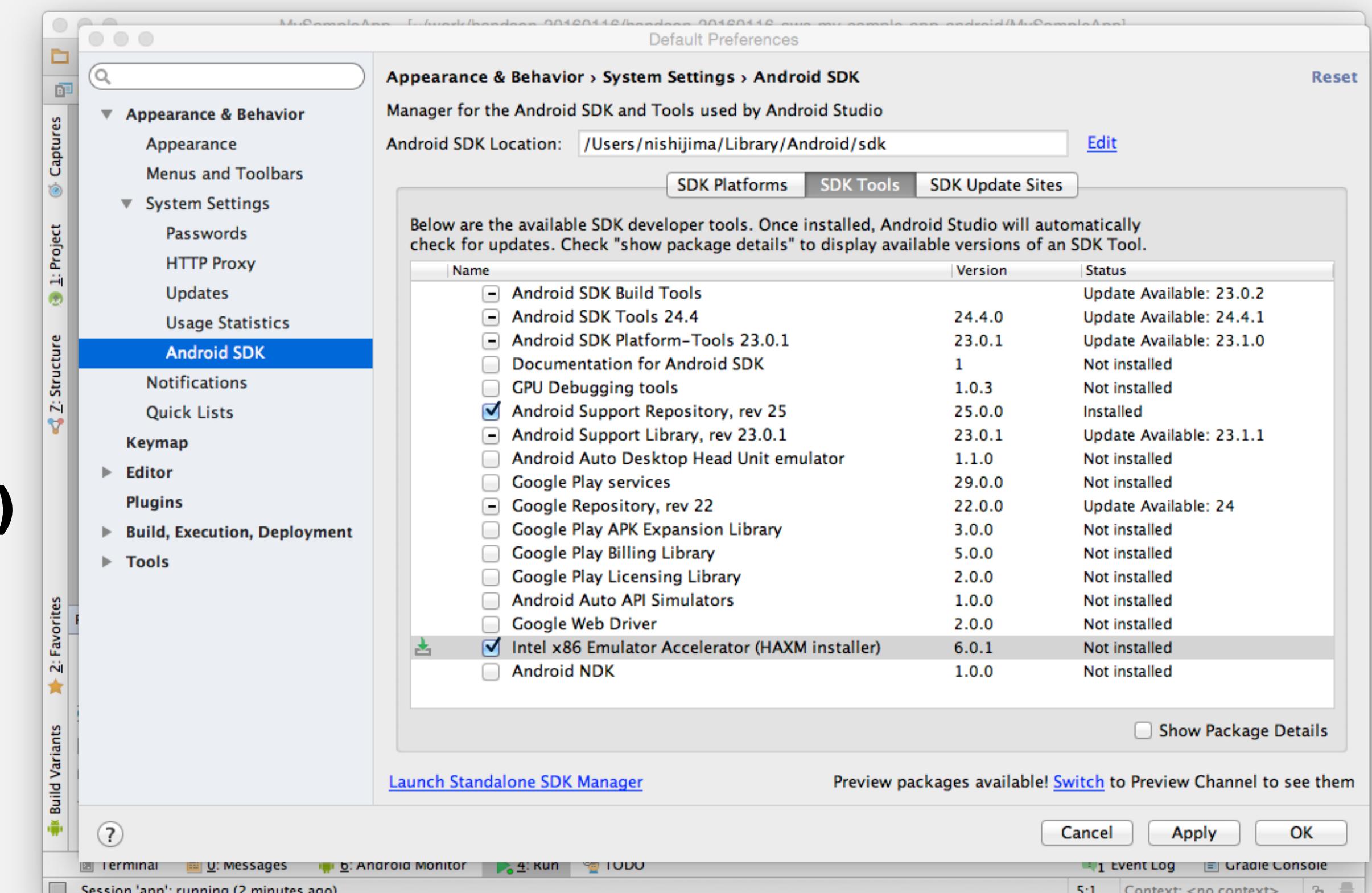


# Android Virtual Device(AVD)の準備

おまけ（と言うか事実上必須？）

## HAXM (Intel® Hardware Accelerated Execution Manager)

- ◆ HAXMというのを入れると、エミュレーターの起動が少し早くなります。
- ◆ Tools -> Android -> SDK Manager -> Appearance & Behavior -> System Settings -> Android SDKと選択して、SDK Toolsのタブの中から Intel x86 Emulator Accelerator (HAXM installer) にチェックを入れて、OKをクリックします。
- ◆ インストーラーのみのダウンロードなので、すぐ終わるはずです。



# Android Virtual Device(AVD)の準備

おまけ（と言うか事実上必須？）

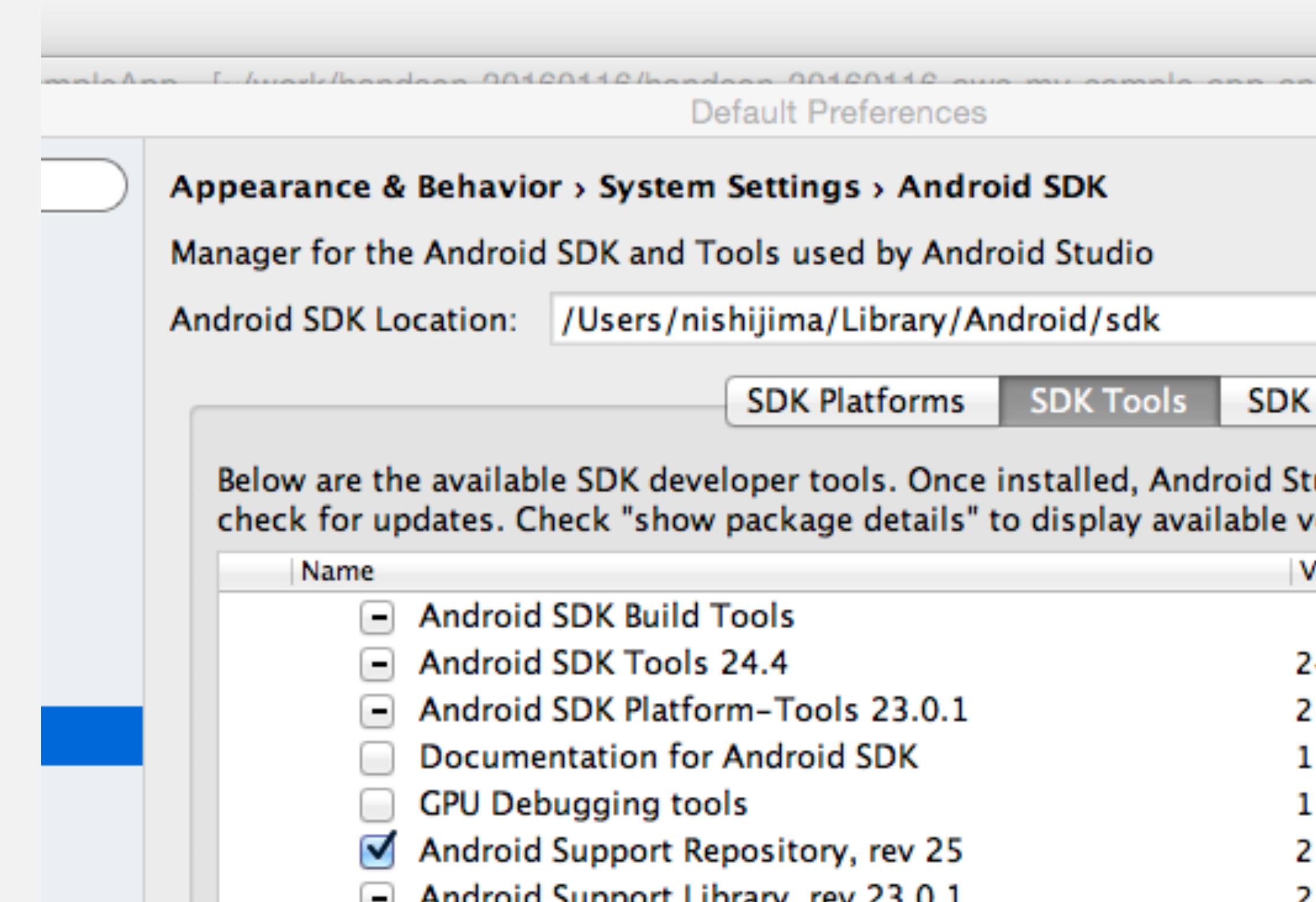
## HAXM (Intel® Hardware Accelerated Execution Manager)

- ◆ ダウンロードされたファイルは、先の画面にあった  
**「Android SDK Location」** の下になります。

- ◆ Macならコンソールから以下のようなコマンド

で.dmgをマウントできます。

```
open ~/Library/Android/sdk/extras/intel/
Hardware_Accelerated_Execution_Manager/
IntelHAXM_6.0.1.dmg
```



**JAWS-UG**

AWS User Group - Japan

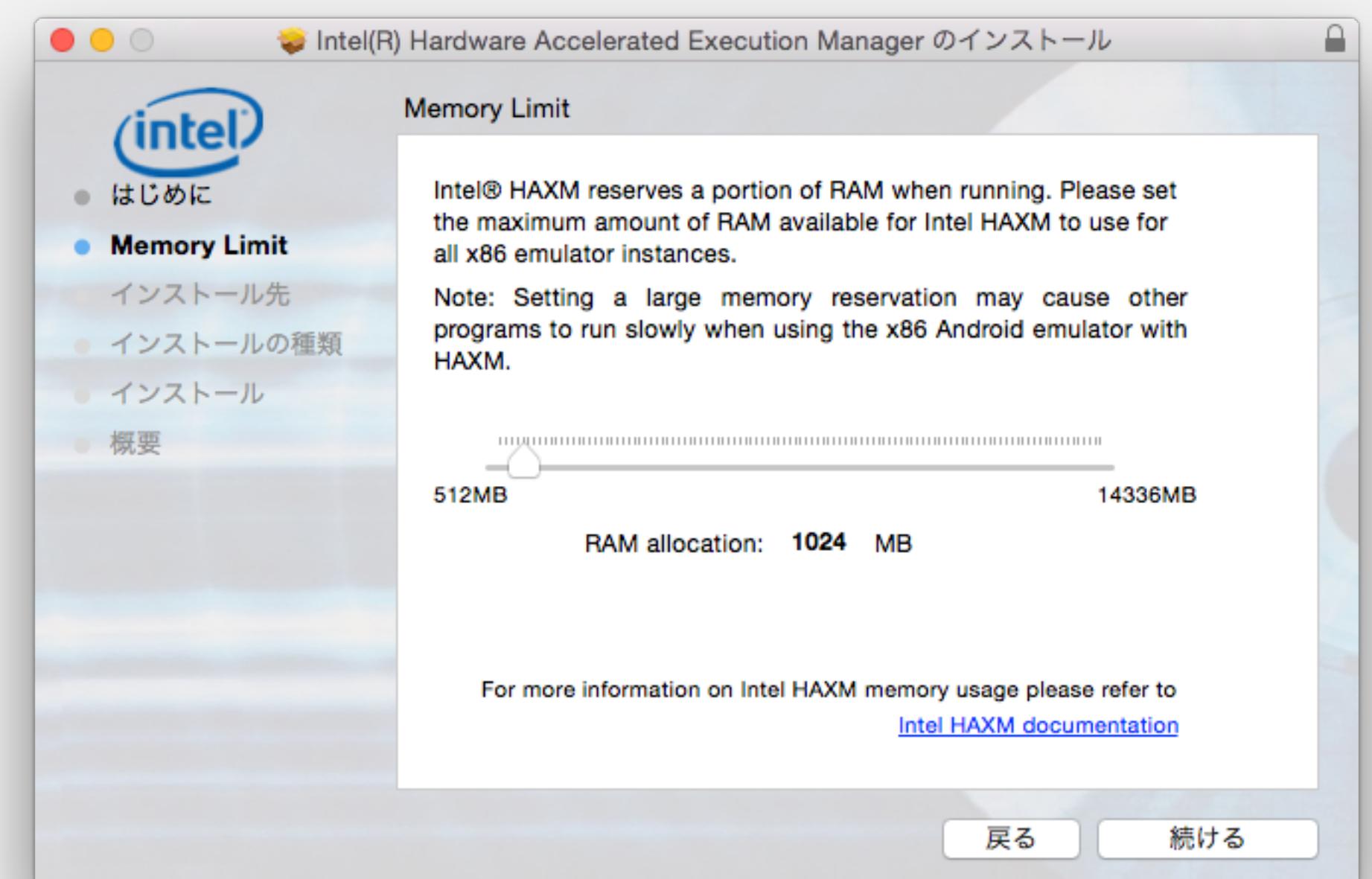
**OKINAWA**

<http://jaws-ug.jp/>  
GitHub repositories <https://github.com/jaws-ug/>

# Android Virtual Device(AVD)の準備 おまけ（と言うか事実上必須？）

## HAXM (Intel® Hardware Accelerated Execution Manager)

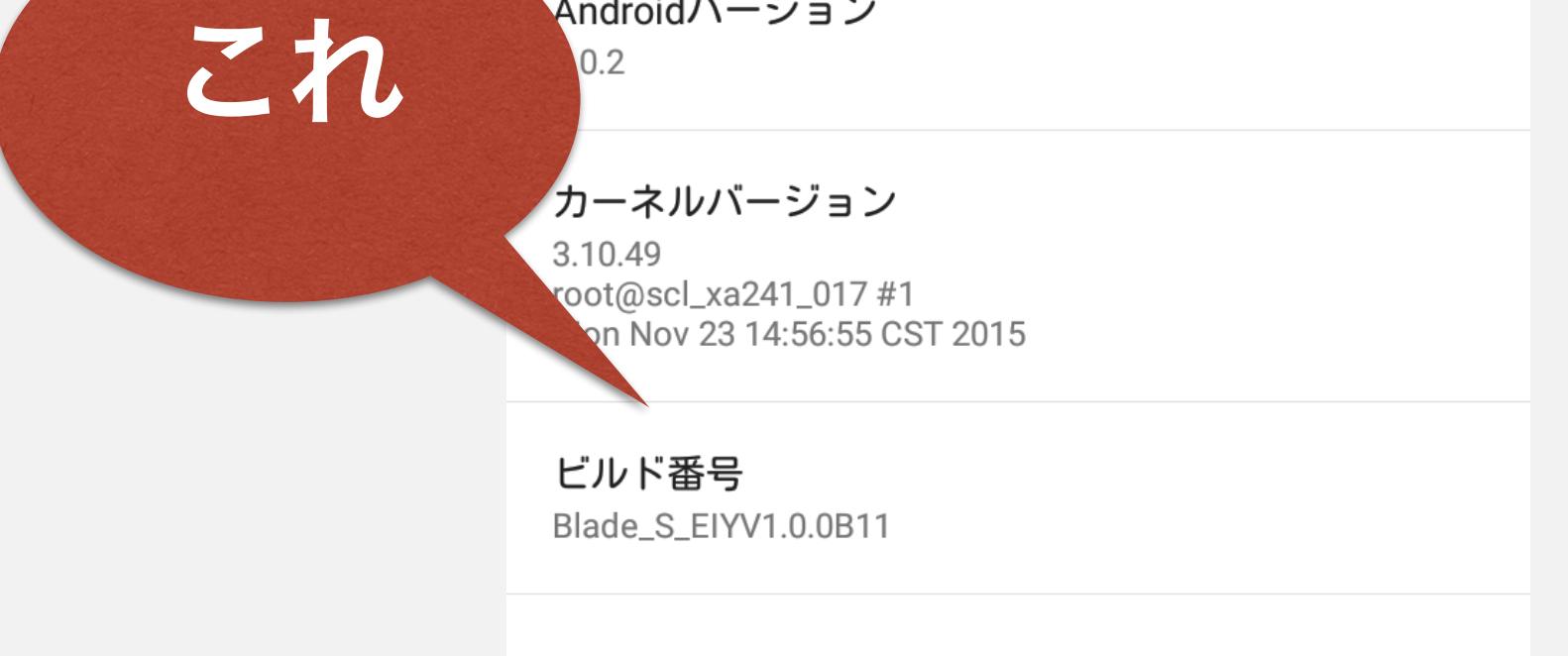
- ◆ 出てきた.**.mpkg**をダブルクリックしてインストールします。
- ◆ 途中の設定で**MemoryLimit**を設定する項目がありますが、これは**1024MB**程度では少なすぎて上手く動かないようです・・・。積んでいるマシンの総メモリ量によりますが、**2048MB**程度を目安にして設定してください。
- ◆ 設定画面などがないので、もしエラーで**HAXM**が起動してくれなかった場合、再度このインストーラーを起動して設定する必要があります。



# Android実機の接続

- ♦ 使いのAndroid端末の「設定画面」 →  
「端末情報」と選択し、「ビルド番号」  
の項目を7回タップします。

(MGSVのクワイエット復活とは何の関係もありません)



**JAWS-UG**  
AWS User Group - Japan

**OKINAWA**

<http://jaws-ug.jp/>  
GitHub repositories <https://github.com/jaws-ug/>

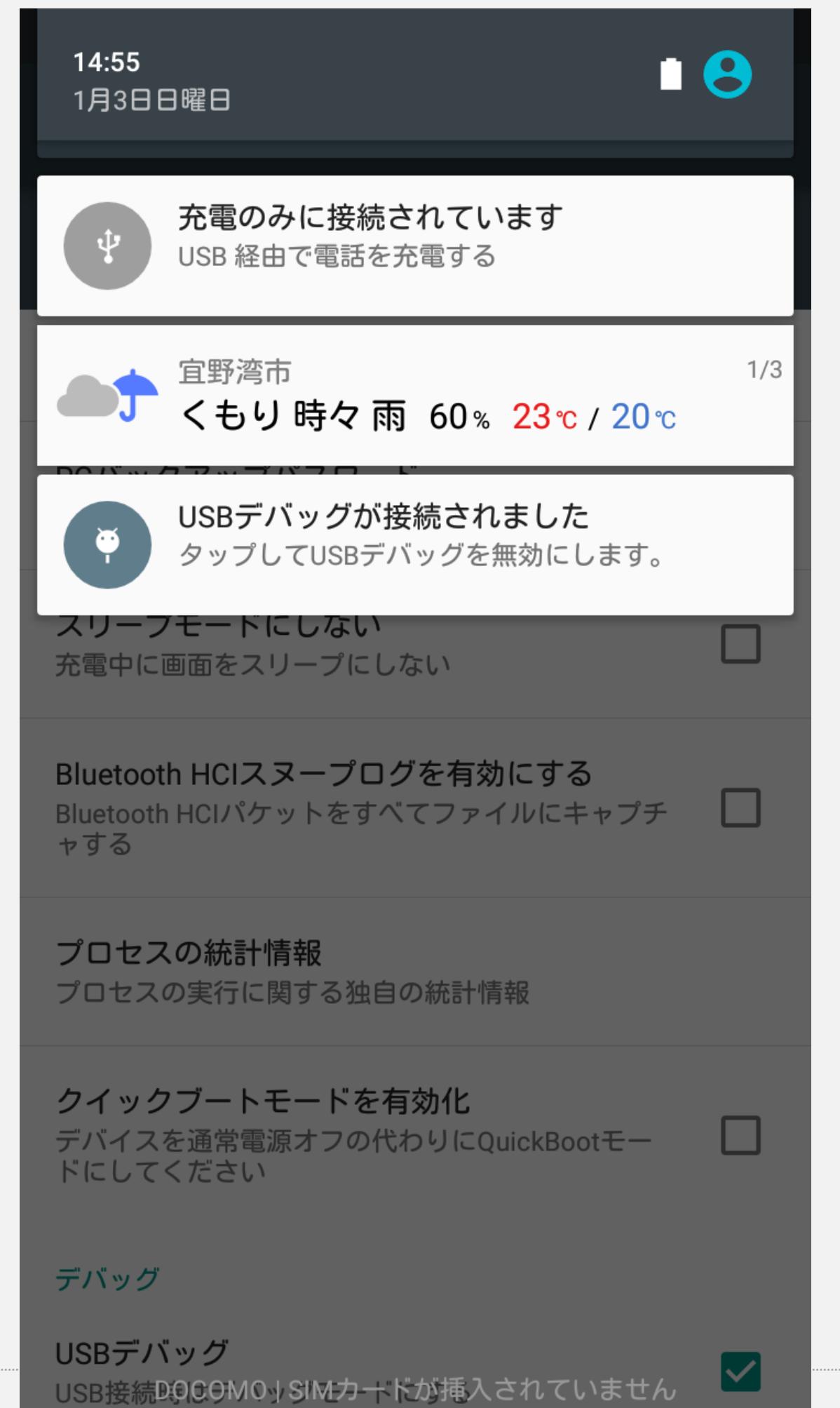
# Android実機の接続

- ◆ 設定の中に「開発者向けオプション」という項目が追加されるので選択し、その中の「USBデバッグ」という項目にチェックを入れます。
- ◆ 質問にはOKと答えてください。



# Android実機の接続

- ❖ 次にPCにUSBケーブルで繋いだ時には、右のように「USBデバッグが接続されました」と表示されるはずです。



この辺で  
ひとやすみ



# クラウド環境の準備一覧

- ◆ AWSアカウントを持ってない人は取得する
- ◆ Facebook AppIDの取得
- ◆ Google Cloud Messagingの設定



JAWS-UG

AWS User Group - Japan

OKINAWA

<http://jaws-ug.jp/>

GitHub repositories <https://github.com/jaws-ug/>

# AWSアカウントの取得

◆ クレジットカードと携帯電話を握りしめて、  
こちらの手順でお願いします！

**<https://aws.amazon.com/jp/register-flow/>**



JAWS-UG

AWS User Group - Japan

OKINAWA

<http://jaws-ug.jp/>

GitHub repositories <https://github.com/jaws-ug/>

# Facebook AppIdの取得

- ♦ **MobileHub**のアプリでのユーザ認証を、**Facebook**アカウントを利用することが可能です。
- ♦ ユーザ認証を組み込むと、アプリ側の表示に利用できたり、色々出来ることが広がります。
- ♦ アプリでユーザ認証を利用するつもりがない場合、設定する必要はありません。
- ♦ **Facebook**のアカウントを持っていない方は、  
まず <https://www.facebook.com> よりアカウントを取得してください。

# Facebook AppIdの取得

- ◆ <https://developers.facebook.com/> の右上のMyApps -> Add a New App を選択
- ◆ ログインしていなければ、右上のボタンは LogIn になっていると思います。
- ◆ Androidのアイコンを選択します。



JAWS-UG

AWS User Group - Japan

OKINAWA

<http://jaws-ug.jp/>  
GitHub repositories <https://github.com/jaws-ug/>

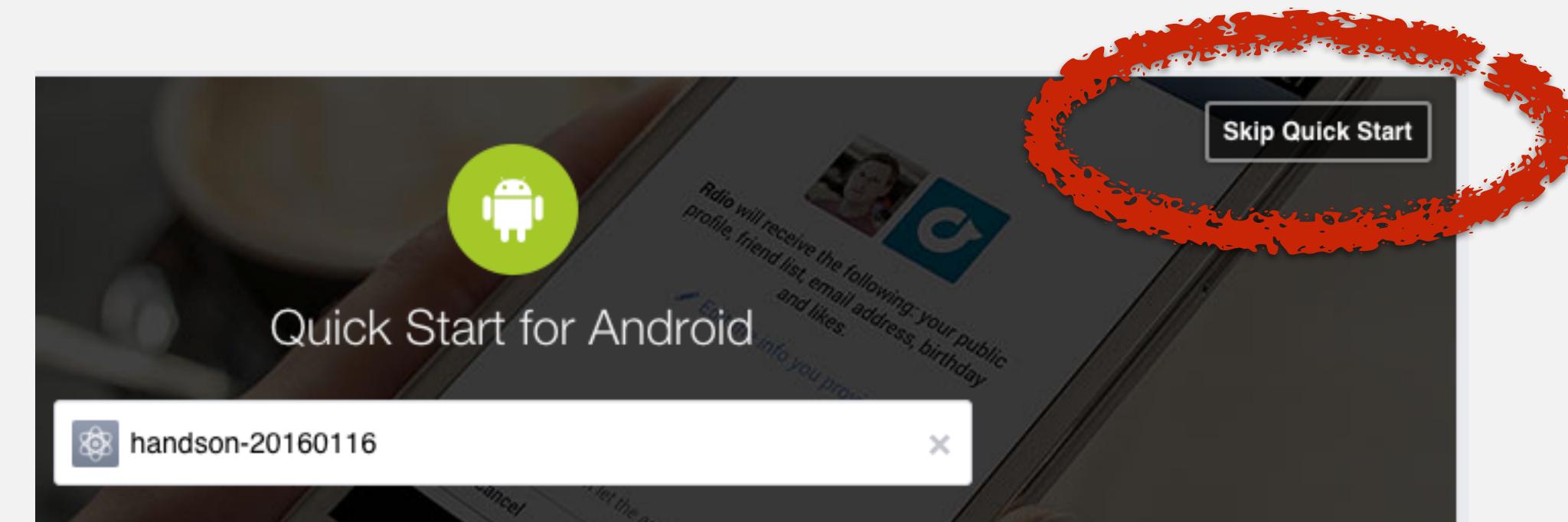
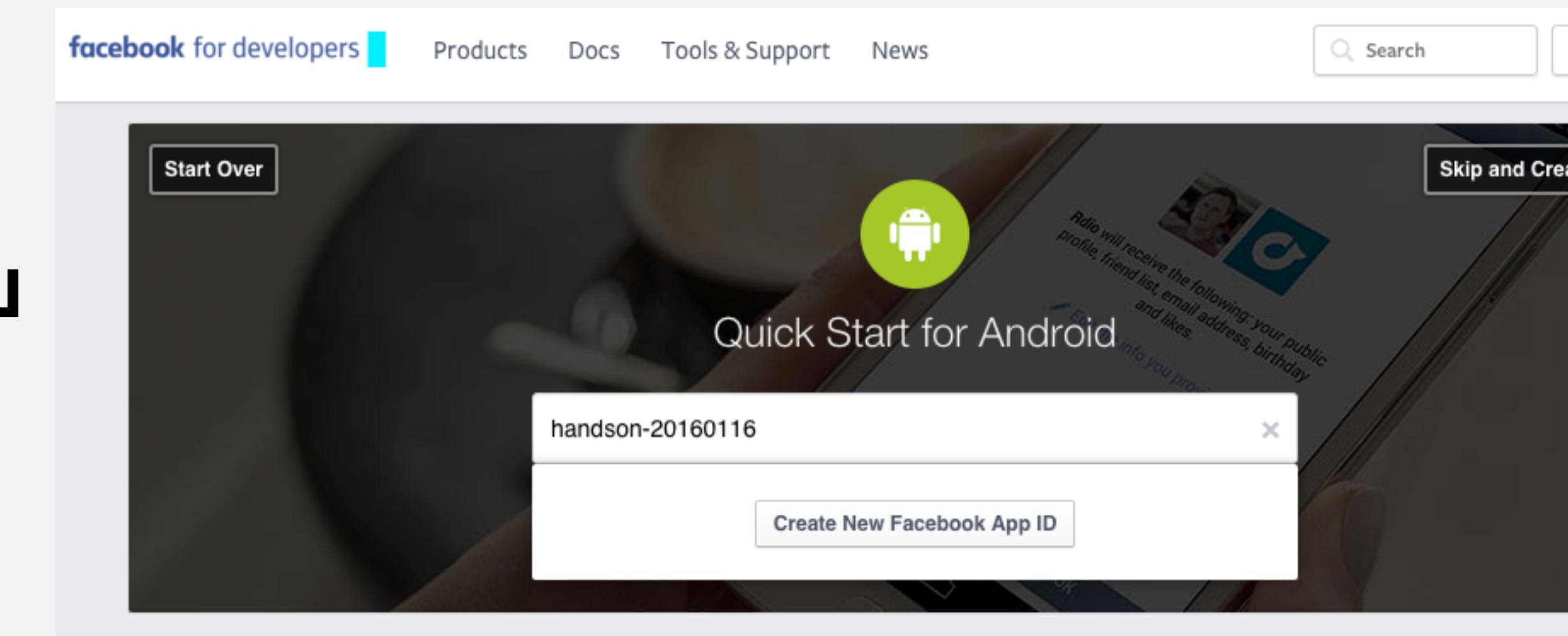
# Facebook App IDの取得

♦ アプリ名を適当に入力し

「Create New Facebook App ID」  
をクリックします。

♦ Quick Startが始まりますが、

右上のSkip Quick Startをクリックしてスキップしてしまいましょう。



# Facebook AppIDの取得

- ◆ カテゴリを選択して、「アプリIDを作成」をクリックします。
- ◆ ダッシュボードのAppIDの項目が今回利用するアプリのAppIDになります。
- ◆ メモをとるか、また後で開けるようにログイン方法などをお忘れなく。



handson-20160116

Dashboard

handson-20160116

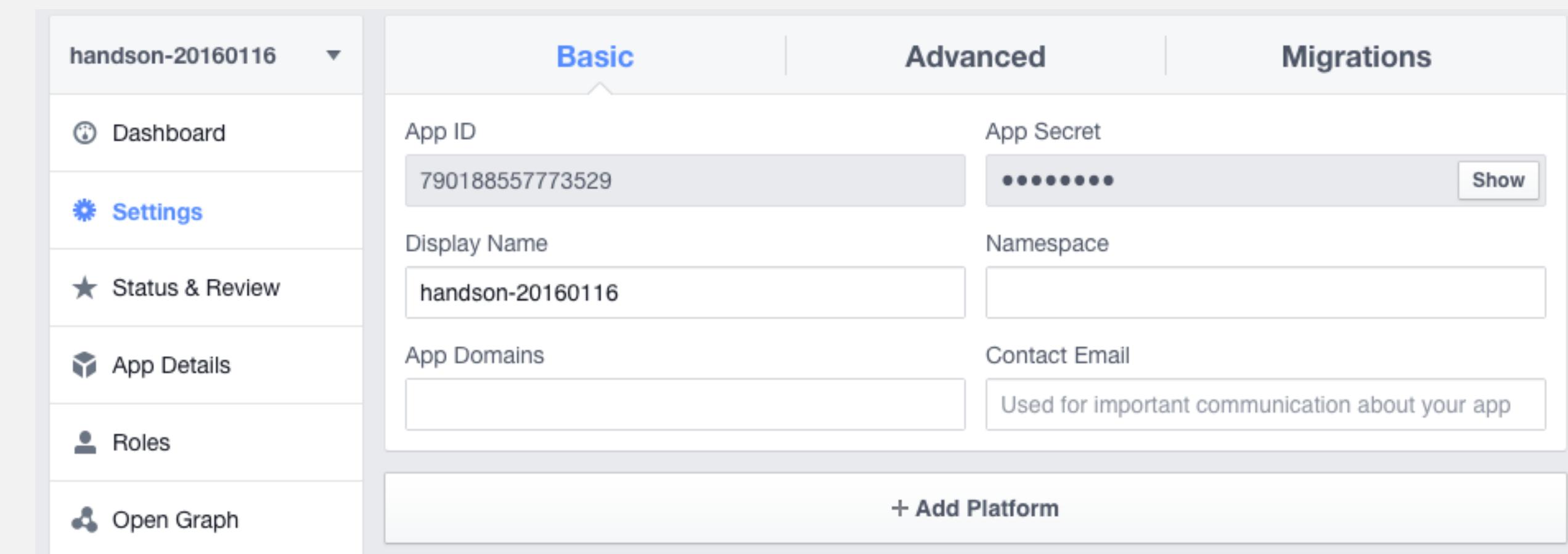
This app is in development mode and can only be used by app admins, developers, and testers.

App ID: 790188557773529 API Version [?] v2.5 App Secret: [REDACTED]

Basic	Advanced	Migrations
App ID: 790188557773529	App Secret: [REDACTED]	

# Facebook Appの設定

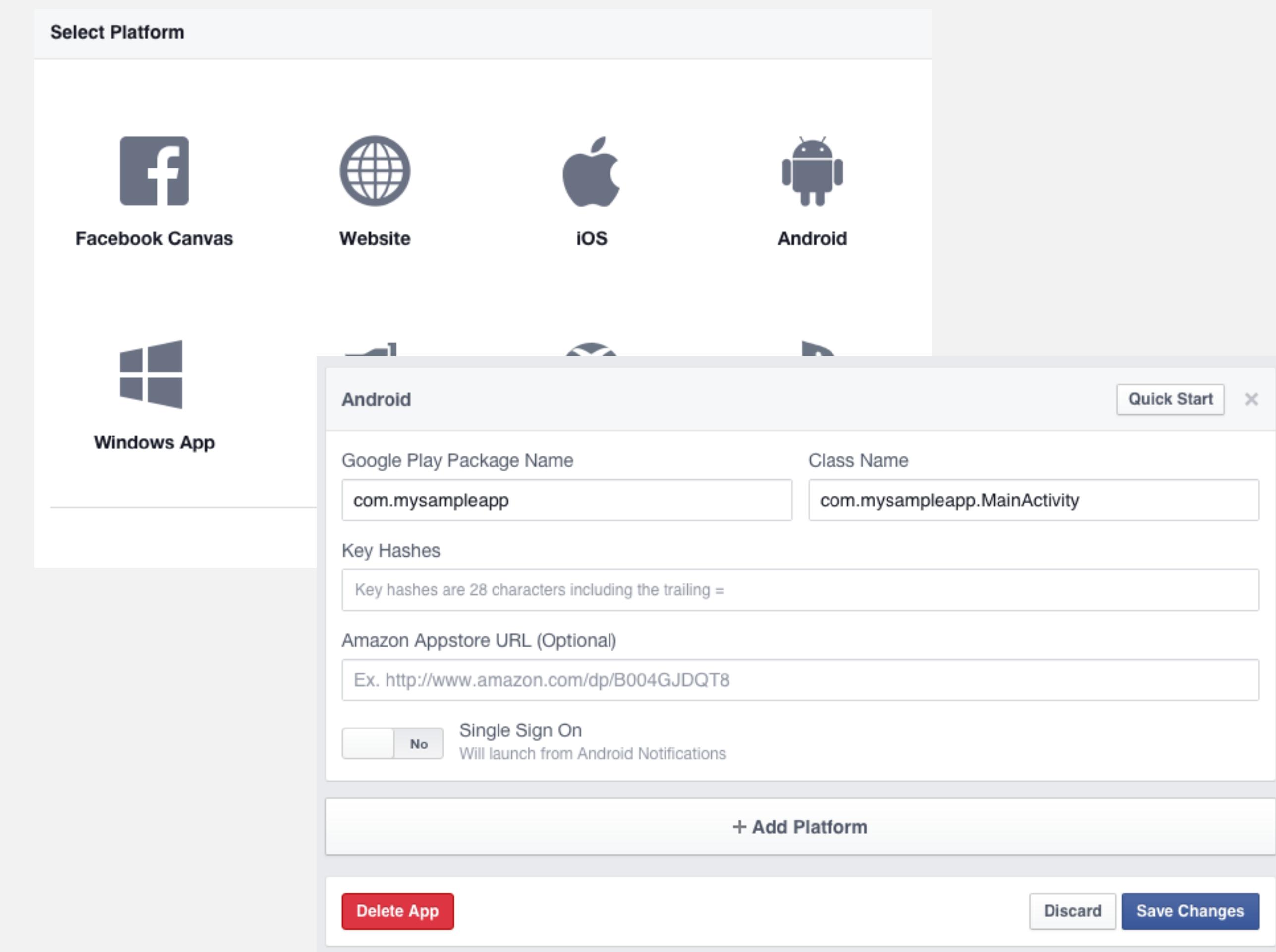
- ◆ Dashboardの下の「Settings」を選択し、「+ Add Platform」をクリックします。



The screenshot shows the AWS Mobile Hub interface for a Facebook app named "handson-20160116". The "Basic" tab is selected. On the left sidebar, there are links for Dashboard, Settings (which is the active tab), Status & Review, App Details, Roles, and Open Graph. The main area contains fields for App ID (790188557773529), App Secret (redacted), Display Name (handson-20160116), Namespace (empty), App Domains (empty), Contact Email (Used for important communication about your app), and a "+ Add Platform" button.

# Facebook Appの設定

- ◆ 「Android」を選択、
- ◆ 「Google Play Package Name」の項目には「**com.mysampleapp**」
- ◆ 「Class Name」の項目には「**com.mysampleapp.MainActivity**」を入力して「**Save Changes**」をクリックします。

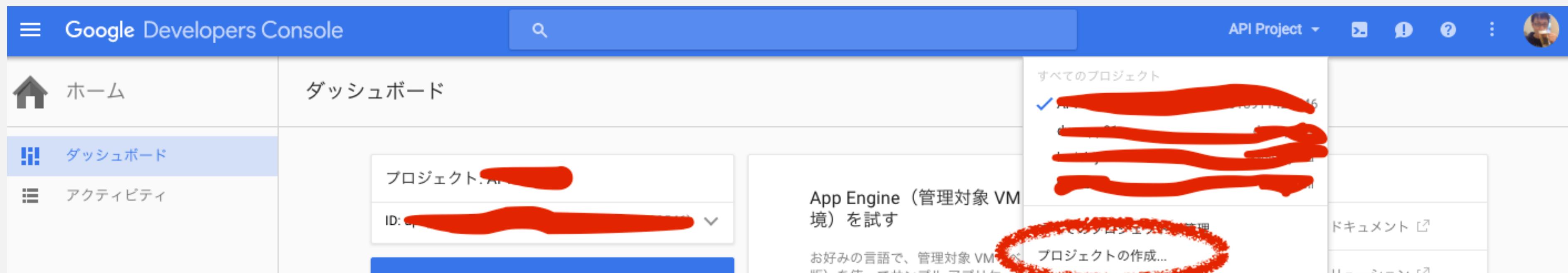


# Google Cloud Messagingの設定

- ◆ MobileHubのアプリで通知を利用する際、Androidの場合は**Google Cloud Messaging**を、iOSの場合は**Apple Push Notification Service(APNs)**を利用しますので、それぞれ事前の設定が必要になります。
- ◆ アプリで通知を利用するつもりがない場合、設定する必要はありません。
- ◆ Googleアカウントを持っていない方は、  
まず <https://accounts.google.com/SignUp?hl=ja> よりアカウントを取得してください。

# Google Cloud Messagingの設定

- ♦ <https://console.developers.google.com/> にアクセスし、右上のメニューから「プロジェクトの作成」をクリックします。



- ♦ 名前を適当に入れて作成をクリック。



# Google Cloud Messagingの設定

- ◆ 作成したプロジェクトがダッシュボードに現れる。
- ◆ この「プロジェクト番号」は後に利用するので、メモしておくと良いと思います。
- ◆ 「**APIを有効にして~**」をクリック。



プロジェクト: handson-20160116

プロジェクトの詳細

プロジェクト ID  
handson-20160116

プロジェクト番号  
1087253399742

Google API を利用する

API を有効にし、認証情報を作成して、  
使用状況を追跡します

API API を有効にして管理しましょう

# Google Cloud Messaging の設定

# ◆ 「Cloud Messaging for Android」を探してクリック

API API Manager

## 概要

概要 認証情報

Google API 有効な API (7)

100 件以上のすべての API を検索

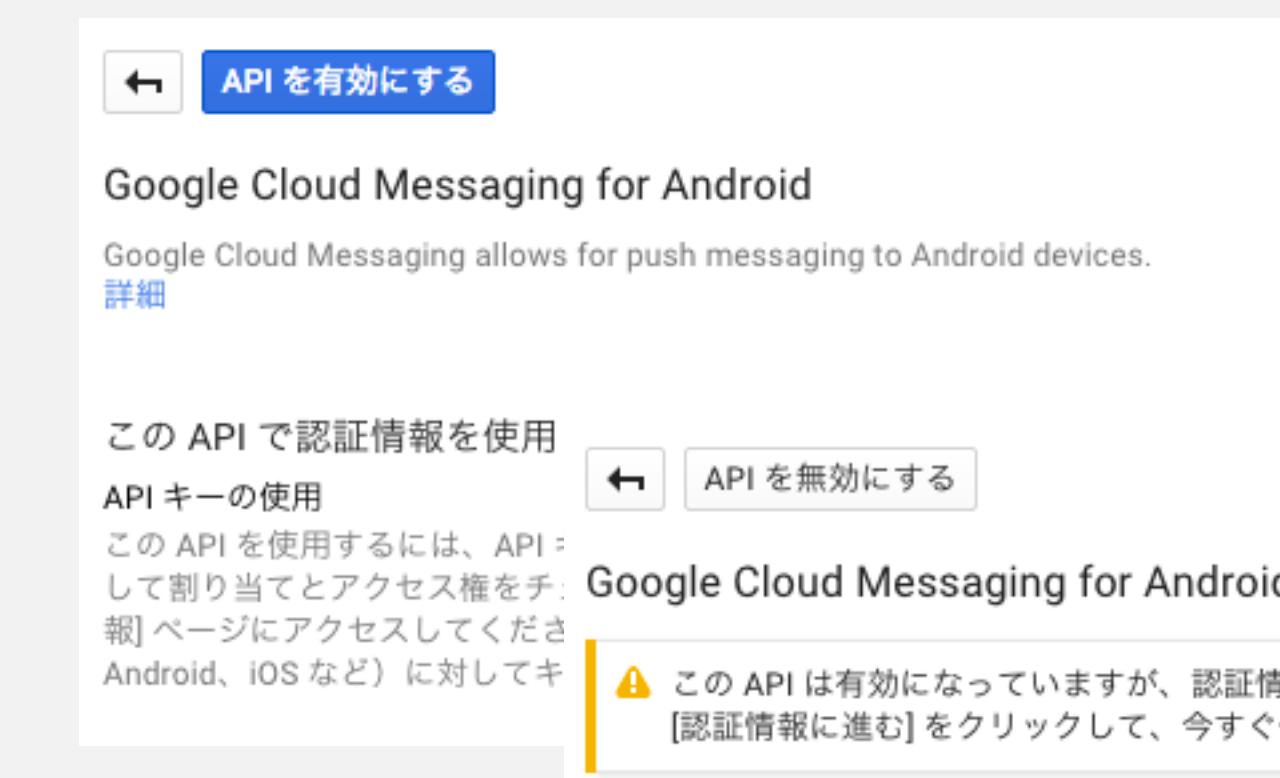
### よく使われている API

 Google Cloud API Compute Engine API BigQuery API Cloud Storage Service Cloud Datastore API Cloud Deployment Manager API Cloud DNS API ▽ その他	 Google Maps API Google Maps Android API Google Maps SDK for iOS Google Maps JavaScript API Google Places API for Android Google Places API for iOS Google Maps Roads API ▽ その他	 Google Apps API Drive API Calendar API Gmail API Google Apps Marketplace Admin SDK Contacts API CalDAV API
 Mobile API Cloud Messaging for Android Google Play Games API Google Play Developer API Google Places API for Android	 Social API Google+ API Blogger API Google+ Pages API Google+ Domains API	 YouTube API YouTube Data API YouTube Analytics API YouTube Reporting API
 Advertising API AdSense Management API DCM/DFA Reporting And Trafficking API Ad Exchange Seller API Ad Exchange Buyer API DoubleClick Search API Analytics API DoubleClick Bid Manager API	 その他の一般的な API Translate API Custom Search API URL Shortener API PageSpeed Insights API Fusion Tables API Web Fonts Developer API	

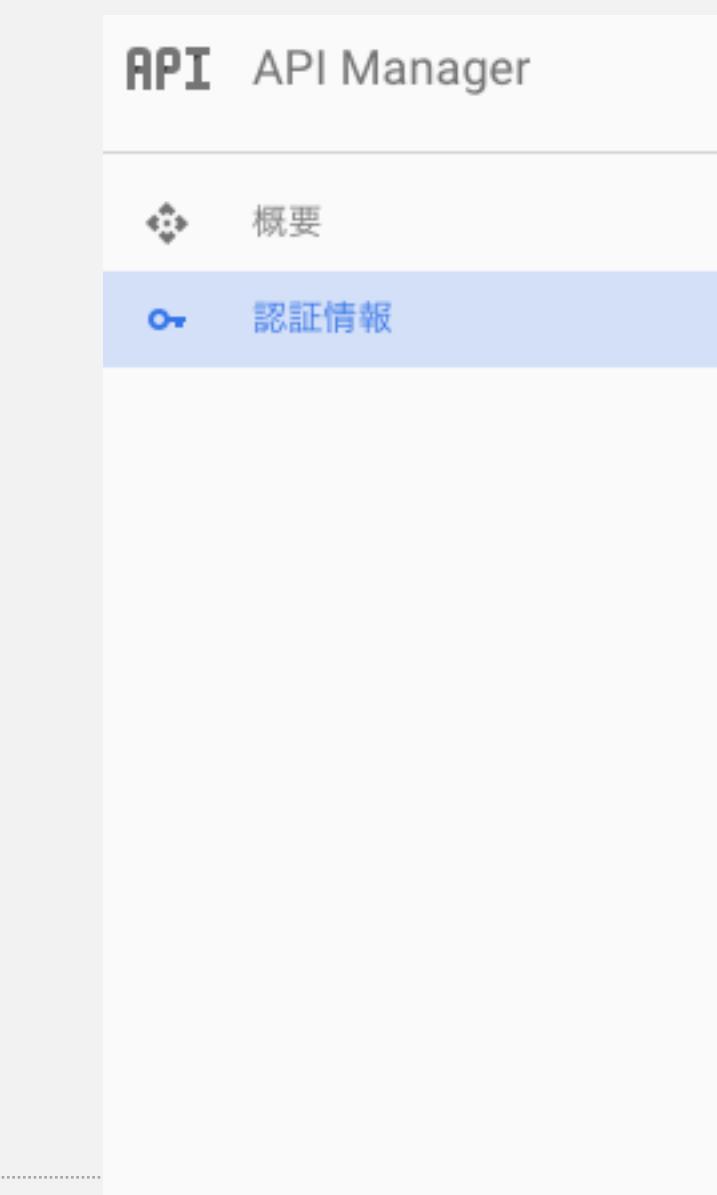


# Google Cloud Messagingの設定

- ◆ 「APIを有効にする」をクリック
- ◆ 「強く推奨」と言われても、青いボタンをクリックしても上手く先に進めなかつたので無視。
- ◆ 左メニューの認証情報→「新しい認証情報」→「APIキー」と選択する。



The screenshot shows the AWS API Manager interface for enabling the Google Cloud Messaging for Android API. It includes a warning message about creating authentication information before use.



The screenshot shows the AWS API Manager interface with the '認証情報' (Authentication Information) tab selected under the 'API Manager' menu. It includes a note about enabling the API and creating authentication information.

# Google Cloud Messagingの設定

- ◆ 4種類の鍵のうち、「Androidキー」を選択。
- ◆ 適当に名前を入れて、「作成」をクリック。
- ◆ APIキーの文字列は作成直後のほか、左メニューの認証情報からも確認できる。
- ◆ 後で利用するので見方を忘れないようにしておいてください。



新しいキーの作成

特定の Google API の呼び出しには API キーが必要です。API キーによりプロジェクトが識別されます。また、割り当ての実施や請求の処理にも使用されるため、A

サーバーキー ブラウザキー Android キー iOS キー

認証情報

Android API キーの作成

名前

handson-20160116-key

Android アプリに使用を制限 (省略可)

Android 搭載端末は API リクエストを直接 Google に送信します。Google では、各リクエストが登録済みのパッケージ名と SHA-1 署名フィンガープリント名に一致する Android アプリから送信されたものであるかどうかを確認します。パッケージ名は AndroidManifest.xml ファイルから取得します。フィンガープリントは以下のコマンドを使用して取得します。

詳細

```
keytool -list -v -keystore mystore.keystore
```

+ パッケージ名とフィンガープリントを追加

Note: It may take up to 5 minutes for settings to take effect

作成 キャンセル



**JAWS-UG**

AWS User Group - Japan

**OKINAWA**

<http://jaws-ug.jp/>

GitHub repositories <https://github.com/jaws-ug/>

事前準備は以上で終了です。

本編でお会いしましょう。



**JAWS-UG**

AWS User Group - Japan

**OKINAWA**

<http://jaws-ug.jp/>

GitHub repositories <https://github.com/jaws-ug/>