

- [SDKのインストール](#)
  - [システムとソフトウェアの要件](#)
  - [SDKをインストールする](#)
  - [ADT Plugin for Eclipseのインストール](#)
  - [ADT Pluginのアップデート](#)
- [EclipseでのAndroidアプリケーション開発](#)
  - [Androidプロジェクトの作成](#)
  - [起動設定の作成](#)
  - [アプリケーションの実行とデバッグ](#)
- [その他のIDEやツールを使ったAndroidアプリケーション開発](#)
  - [Androidプロジェクトの作成](#)
  - [Androidアプリケーションのビルド](#)
  - [Androidアプリケーションの実行](#)
  - [アプリケーションにデバッガをアタッチする](#)
  - [IDEでデバッグ用のポートを設定する方法](#)

## SDKのインストール

このページは、Android SDKをインストールして、あなたの開発環境をセットアップする方法を記載します。以下のリンクからSDKをダウンロードすることができます。

[SDKのダウンロード](#)

## システムとソフトウェアの要件

Android SDKでコードとツールを使用しているAndroidアプリケーションを開発するために、下記のような開発環境を必要とします。

Windows XP or Vista  
 開発用コンピュータ Mac OS X 10.4.8 またはそれ以降 (x86のみ)  
 Tested on Linux Ubuntu Dapper Drake

開発環境 Eclipse IDE for Java Developers [Eclipse](#) 3.2, 3.3 (Europa)  
[Android Development Tools](#) plugin (optional)  
 Java SE Development Kit 5 or 6 (JRE alone is not sufficient)  
 Any other environment or IDE Not compatible with Gnu Compiler for Java (gcj)  
[Apache Ant](#) 1.6.5 or later for Linux and Mac, 1.7 or later for Windows

## SDKをインストールする

SDKをダウンロードした後に、あなたのPCの適当な場所にzipファイルを解凍してください。デフォルトでは android\_sdk\_<platform>\_<release>\_<build> のディレクトリに解凍されます。ディレクトリには tools/, samples/, その他のディレクトリがあります。解凍したディレクトリの場所はAndroidのプラグインの導入やSDKのツールを使用する際に必要になるので覚えておいてください。

以後、\$SDK\_ROOTはSDKをインストールしたディレクトリとして扱います。

以下のように、あなたは"\$SDK\_ROOT/tools"を実行パスに加えることができます。

- Linuxの場合
  - ~/.bash\_profileまたは~/.bashrcファイルを編集してください。PATH環境変数をセットする行を探して、そこに \$SDK\_ROOT/toolsを加えてください。環境変数の設定行を探すのが面倒くさければ、以下の一行を足すことで設定できます。

```
export PATH=${PATH}:<path to your $SDK_ROOT/tools>
```

- Macの場合
  - ホームディレクトリの.bash\_profileをLinuxの場合と同じように編集してください。 .bash\_profileがなければ作成します。
- Windowsの場合
  - マイコンピュータのプロパティから詳細設定を開き、環境変数ボタンを押して、システム環境変数のPathを編集して \$SDK\_ROOT/toolsを追加します。

\$SDK\_ROOT/toolsを実行パスに加えることで、ツールディレクトリのフルパスを入力することなくAndroid Debug Bridge(adb)と他のコマンドラインツールを使用することが出来るようになります。  
なお、SDKをアップデートして\$SDK\_ROOTが変更された場合、実行パスの更新を忘れないように注意してください。

## ADT Plugin for Eclipseのインストール

あなたがAndroidアプリケーションを開発するためにEclipse IDEを使用しているなら、Android Development Tools(ADT)と呼ばれているカスタムプラグインをインストールすることができます。ADTプラグインはAndroidプロジェクトとツールの統合したサポートを提供します。

ADTプラグインは、Androidアプリケーションの作成、実行、デバッグをより速くより簡単にする様々な強力な拡張を含んでいます。

あなたがEclipse IDEを使用しないなら、ADTプラグインをインストールする必要はありません。

ADTプラグインのダウンロード、インストール方法は下記の通りです。Eclipseのリモートサイト更新機能を利用してください。

1. **Eclipse** を起動、**Help > Software Updates > Find and Install**
2. ダイアログが出るので、**Search for new features to install** を選んでNextを押す
3. **New Remote Site** を押す
4. ダイアログが出るので、名前(例えばAndroid Plugin)と以下のURLを入力してOKを押す  
<https://dl-ssl.google.com/android/eclipse/>
5. サーチリストに加えて、チェックし **Finish** を押す
6. サーチ結果のダイアログの **Android Plugin** のチェックボックス > **Eclipse Integration > Android Development Tools** を選択、そしてNextを押す
7. ライセンス契約を読み、問題なければライセンス契約に同意を選び、Nextを押す
8. **Finish** を押す
9. ADTプラグインは署名されていません。 **Install All** を押すことでインストールを受け入れることができます
10. Eclipseを再起動
11. Eclipseの再起動後にpreferencesのSDK root directoryを適切な場所に更新します
  1. **Window > Preferences** を選択しPreferences画面を表示(Mac OS X: **Eclipse > Preferences**)
  2. 左のペインからAndroidを選択
  3. SDK Locationを **Browse** で参照してSDKのルートディレクトリに設定
  4. **Apply** を押して **OK** を押す

## ADT Pluginのアップデート

ADTプラグインを最新版にアップデートするために、以下のステップに従ってください。

1. **Help > Software Updates > Find and Install** を選択
2. **Search for updates of the currently installed features** を選択し、**Finish** を押す
3. ADTのアップデートが存在すればインストールする

あるいは

1. **Help > Software Updates > Manage Configuration** を選択
2. ツリーを展開して、**Android Development Tools xx.xx.xx**を選択 xx.xx.xxはバージョン番号
3. 右ペインで **scan for Updates** を選択

## EclipseでのAndroidアプリケーション開発

Eclipse IDEでAndroidアプリケーションを開発し始めるためには、最初にAndroidプロジェクトを作り、起動設定を行う必要があります。

その後、あなたは思う存分あなたのアプリケーションを作りこんだり、走らせたり、デバッグすることができるでしょう。

以下のセクションでは、Eclipse環境でADTプラグインをインストールしていると仮定して説明を行います。  
まだADTプラグインをインストールしていなければ、先にADTプラグインをインストールしてください。

## Androidプロジェクトの作成

ADTプラグインは新規、あるいは既存のコードのために素早くEclipseプロジェクトを作るために使うことができる新しいプロジェクトウィザードを提供します。プロジェクトをつくるために、以下のステップに従ってください。

1. **File > New > Project** を選択

Android > Android Project を選択し Next を押す

2.

3. 作成するプロジェクトの内容によって以下の中から選んでください(図 )

1. 新規のソースコードでプロジェクトを作成する

1. **Create new project in workspace** を選び、プロジェクト名、ベースパッケージ名、.javaファイルとして作成するスタブのActivityクラスの名前とアプリケーションの名前を入力してください。

2. 既存のソースコードを使用してプロジェクトを作成する

1. **Create project from existing source** を選択します。SDKに含まれるサンプルアプリケーションのいずれかをビルドして実行したい場合にこのオプションを使用してください。

サンプルアプリケーションは、SDKのsampleディレクトリにあります。

2. 既存のソースコードを含んでいるディレクトリを参照して、「OK」をクリックしてください。ディレクトリが有効なAndroidマニフェストファイルを含んでいれば、ADTプラグインはパッケージ、Activity、アプリケーション名を自動的に入力します。

4. **Finish** を押す

ADTプラグインは、プロジェクトに必要な以下のフォルダ、ファイルを作成します。

**src/**

.java Activityファイルのスタブを含むフォルダ

**res/**

リソースのためのフォルダ

**AndroidManifest.xml**

あなたのプロジェクトのマニフェストファイル

## 起動設定の作成

Eclipseでアプリケーションをデバッグする前に、起動設定を作成しなければいけません。

起動設定では、起動するプロジェクト、Activity、エミュレーターオプション、その他を指定します。

起動設定を作成するために以下の手順に従ってください。

1. **Run > Open Run Dialog ...** あるいは **Run > Open Debug Dialog ...** の適切なほうを選ぶ(実行かデバッグか)
2. 左側のプロジェクトタイプのリストのAndroid Applicationを右クリックしNewを選択
3. 今回作成する構成の名称を入力
4. Androidタブから、プロジェクトと開始するActivityを参照
5. Emulatorタブから希望の画面とネットワークプロパティを設定。これらは[Emulatorの起動オプション?](#)でも設定可能
6. Commonタブから追加のオプションを選ぶことも可能
7. 起動設定を保存するために **Apply** を押す、あるいは **Run** か **Debug** の適切なほうを選ぶ

## アプリケーションの実行とデバッグ

一旦アプリケーションのためにプロジェクトと起動設定を準備すれば、下記のようにアプリケーションを実行したりデバッグすることができます。

アクティブな起動設定から、起動するかデバッグを行うために、Eclipseのメインメニューから、**Run > Run** あるいは **Run > Debug** から適切な方を選びます。

Run configuration managerのアクティブな起動設定は最も最近実行された起動設定であることに注意してください。Eclipse Navigationペインでアプリケーションが選択されていたとしても、必ずしも一致しているわけではありません。

アクティブな起動設定をセット、あるいは変更するにはRun configuration managerを使います。

**Run > Open Run Dialog ...** あるいは **Run > Open Debug Dialog** から設定することができます。

アプリケーションを実行、デバッグすると以下のことが行われます。

- エミュレータが起動していなければ、エミュレータが起動します
- 最後のビルドから変更があれば、プロジェクトをコンパイルし、アプリケーションをエミュレータにインストールします。
- **Run** アプリケーションを実行します。
- **Debug** EclipseのJavaデバッガをアプリケーションにアタッチして"Wait for debugger"のデバッグモードで起動します。

## その他のIDEやツールを使ったAndroidアプリケーション開発

Androidアプリケーションを開発するにはEclipseでAndroidプラグインを使うことをお勧めします。このプラグインは機能的にIDEと融合した編集、ビルド、デバッグ機能を提供します。しかし、SDKにはIntelliJ(またはプラグインなしでEclipseを使いたいならば)を含む他のIDEで開発することができるためのツールを含んでいます。

### Androidプロジェクトの作成

Android SDKは、ビルドファイルとともに、activityCreator(プロジェクトのためにいくつかのスタブファイルを生み出すプログラム)を含んでいます。

あなたは新しいコード、または、既存のコード(例えばSDKに含まれるサンプルアプリケーション)からAndroidプロジェクトをつくるプログラムを使うことができます。

LinuxとMacのために、SDKはactivityCreator.py、パイソンスクリプトと、Windowsのために、activityCreator.bat(バッチスクリプト)を提供しています。プラットフォームに関係なく、同様にactivityCreatorを使うことができます。

Androidプロジェクトを作成するために、activityCreatorを実行して、以下のステップに従ってください。

1. コマンドラインで、SDKのtoolsディレクトリに移動し、あなたのプロジェクトファイルのために新しいディレクトリを作ってください。
2. activityCreatorを実行します。コマンドの引数には完全なクラス名を指定しなければなりません。新しいコードのためにプロジェクトを作っているのであれば、クラスはスクリプトが作成するスタブクラスの名前を意味します。既存のコードからプロジェクトを作っているのであれば、パッケージで1つのActivityクラスの名前を指定しなければなりません。activityCreatorスクリプトのコマンドオプションは以下の通りです。
  1. --out <folder>  
出力ディレクトリを設定します。デフォルトはカレントディレクトリです。あなたのプロジェクトファイルのために新しいディレクトリを作成したのであれば、それを指定するために、このオプションを使用してください。
  2. --ide intellij  
新しいプロジェクトでIntelliJ IDEAプロジェクトファイルを生成するかどうかのオプションです。

例を示します。

```
~/android_linux_sdk/tools$ ./activityCreator.py --out myproject your.package.name.ActivityName
package: your.package.name
out_dir: myproject
activity_name: ActivityName
~/android_linux_sdk/tools $
```

activityCreatorスクリプトは、以下のファイルとディレクトリを生成します(既存のものには上書きしません)

- AndroidManifest.xml  
指定されたActivityクラスと同期したアプリケーションのマニフェストファイル
- build.xml  
アプリケーションをビルドしたりパッケージするためのAnt用ファイル
- src/your/package/name/ActivityName.java  
生成時に指定したActivityクラスファイル
- your\_activity.iml, your\_activity.ipr, your\_activity.iws  
-ide intellijフラグを指定したときだけに生成されるIntelliJ用のプロジェクトファイル
- res/  
リソース用のディレクトリ
- src/  
ソースディレクトリ
- bin/  
ビルドスクリプトのアウトプットディレクトリ

生成してからはディレクトリはどこに移動しても構いませんが、SDKのtoolsフォルダが実行パスに設定されていることが必要です。

また、SDKディレクトリを変更するとビルドスクリプトを手動で変更する必要があるため、SDKディレクトリの変更は控えたほうがよいでしょう。

### Androidアプリケーションのビルド

アプリケーションをビルドにはactivityCreatorによって生成されるbuild.xmlファイルをAntに対して使用します。

1. もしまだAntを手に入れていなければ、[Apache Ant home page](#) から手に入れることができます。インストールし、実行パス

に加えてください。

2. Antを使う前に、環境変数[JAVA\_HOME]にJava JDKのインストールパスを指定しておく必要があります。  
メモ: JDKをWindowsにインストールする際に、"Program Files"ディレクトリがデフォルトになっています。このパスにはスペースが含まれているので、Antの実行が失敗します。この問題を解決するには環境変数[JAVA\_HOME]に"JAVA\_HOME=c:\Progra~1\Java\"のように設定します。もっと簡単な方法はJDKをスペースを含まない、"c:\java\jdk1.6.0\_02"のようなディレクトリにインストールすることです。
3. まだプロジェクトを作成していなければ、上記のAndroidプロジェクトの作成に従って、先にプロジェクトを作成しておいてください。
4. あなたはbuild.xmlがあるディレクトリと同じパスで、antとタイプするだけで、Antのビルドファイルを実行することができます。しかしソースファイルまたはリソースを変更するたびにAntを実行しなければいけません。Antを実行しておかないと、修正がアプリケーションに反映されません。

## Androidアプリケーションの実行

コンパイルされたアプリケーションを実行するには、以下に示すadbツールを使用して、.apkファイルをエミュレータの/data/appディレクトリにアップロードします。

1. エミュレータを起動します(<あなたのSDKディレクトリ>/tools/emulator をコマンドラインから起動)。
2. エミュレータをホームスクリーン(待受け画面)に移動してください(あなたがエミュレータでアプリケーションを再インストールするとき、ホームキーを押して該当のアプリケーションを動作させないようにしておきましょう)。
3. 実行可能ファイルをアップロードしてmyproject/bin/(<appname>).apkをインストールするために、adbを実行します。
4. 例えば、Lunar Landerのサンプルをインストールするには、コマンドラインで<あなたのsdk\_dir>/sample/LunarLanderディレクトリへ移動し、./../tools/adb install /LunarLander.apk と入力します。
5. エミュレータ上で利用可能なアプリケーションのリストを開き、画面をスクロールさせて、あなたのアプリケーションを選択して起動してください。

メモ:あなたが初めてActivityをインストールするとき、アプリケーションランチャーに現れる前、あるいは、他のアプリケーションから呼ばれる前に、エミュレータを再起動する必要があるかもしれません。なぜなら、パッケージマネージャは通常エミュレータ起動時のみ完全にマニフェストを調べるからです。

## アプリケーションにデバグガをアタッチする

このセクションはエミュレータで実行アプリケーションをデバグするためにIDEを中継する方法とデバグ情報(例えばCPU使用率)をスクリーンに表示する方法を解説します。

デバグガのアタッチはEclipseプラグインを使えば自動的に行われますが、他のIDEでもデバグポートを監視してデバグ情報を受け取ることができるように設定することができます。

1. IDEとエミュレータの間のポートフォワーディングの役割を果たす、[Dalvikデバグモニターサーバー\(DDMS\)?](#)を起動してください。
2. アクティビティにデバグガがアタッチされるまでアプリケーションの起動を待つようにエミュレータのデバグオプション設定をしてください。これらの大量のデバグオプションはエミュレータ上でCPU使用率やリフレッシュレートを表示するようなことではDDMSを使用しなくてもよいことを覚えておきましょう。
3. デバグするためにIDEでポート番号の8700を割当ててください。プロジェクトをデバグするためにEclipseのセットアップをどうするかは以下で述べます。

## IDEでデバグ用のポートを設定する方法

DDMSはエミュレータの上の全てのVMに特定のデバグポートを割り当てます。