Android アプリケーション基本

[Android]

- http://developer.android.com/guide/topics/fundamentals.html

基本

■ソースコードとパッケージ

- Android アプリケーションは、Java で書かれます。
- コンパイルされた Java コード (および必要な、データおよびリソースファイルも一緒に) は aapt ツールにより .apk という拡張子の Android パッケージとしてアーカイブされます。
- このファイルは、モバイルデバイスへアプリケーションを配布し、インストールするための乗り物となります。
- ユーザーは、デバイスへこのファイルをダウンロードします。
- すべてのコードは一つの .apk ファイルに含まれ、これが 1 つのアプリケーションと見なされます。

■アプリケーションはどのように動くのか

プロセスの起動とシャットダウン

- デフォルトでは、どのアプリケーションも、自分自身の Linux プロセスで起動されます。
- Android は、アプリケーションコードの実行が必要な時にプロセスを開始します。
- アプリケーションが、不要になり、他のアプリケーションがリソースを必要としたときにシャットダウンされます。

仮想マシン

- すべてのプロセスが独自に仮想マシン (VM) を持ちます。なので、アプリケーションコードは、他のアプリケーションから隔離されて実行されます。

ユーザーとパーミッション

- デフォルトでは、どのアプリケーションもユニークな Linux ユーザー ID に結びつけられます。
- パーミッションも設定されるため、アプリケーションのファイルは、結局ユーザーであるアプリケーション自身からのみ参照できます。
- しかし、パーミッションを他のアプリケーションにエクスポートする方法も存在します。

■ユーザー ID の共有

- 2 つのアプリケーションで同じユーザー ID を共有することが可能。
- この場合、お互いのファイルを参照することが可能になる。
- システムリソースを節約するために、同じ ID をもつアプリケーションを同じ Linux プロセスに配置して、同じ VM を供給することができる。

アプリケーション コンポーネント

■他のアプリケーションの利用

・Android の主要な機能は、あるアプリケーションが他のアプリケーションの要素を (許されていれば) 活用できること。
・たとえば、あなたのアプリケーションがイメージのスクロールリストを表示する必要があり、他のアプリケーションが適切に開発されたスクローラーをもっており、他から利用できるようにしている場合、あなたは、自分で開発するよりも、そのスクローラーを呼び出すことができる。
・アプリケーションは、他のアプリケーションやリンクとコードを合併させない。
・それよりは、他のアプリケーションの一部を必要に応じて、かなりシンプルに開始できる。

■コンポーネント

・システムは、アプリケーションの一部を必要に応じてプロセスを開始でき、Java オブジェクトをインスタンス化できる必要がある。
・なので、アプリケーションはその他多くのシステムと異なっている、Android アプリケーションは、ひとつのエントリーポイント (main() 関数のような) を持たない。
・そうではなく、それらは、次の 4 つの本質的なコンポーネントを持っており、システムは必要に応じてインスタンス化して起動する。

■ Activity

概要

・アクティビティ は ユーザーが行えることを、一箇所に集中することを試みるビジュアルユーザーインターフェース を表現する。
・たとえば、アクティビティは ユーザーが選択できるメニューアイテムのリストを表現すし、写真をキャプションとともに表示する。
・テキストメッセージングアプリケーションは、おそらく、メッセージを送信するために、リストの内容を表示する、ひとつのアクティビティを持つ。
・2 つめの選択された連絡先へメッセージを記入するアクティビティ、そして、他の古いメッセージをレビューする、設定を変更するアクティビティ。
・それぞれのアクティビティは、他から独立しているにもかかわらず、それらは凝集度の高いユーザーインターフェースとして一緒に働く。
・それらは Activity 基底クラスのサブクラスとして実装される。

アクティビティとは？

・あるアプリケーションはただひとつ、またはいくつかの Activity から構成される。
・アクティビティをいくつ使うべきかなどはもちろんアプリケーションの設計によるが、典型的には、アクティビティのうちひとつが最初のアクティビティとしてマークされ、アプリケーションが起動したときにユーザーに表示される。
・ひとつのアクティビティを他へ移動することは、現在のアクティビティを完了させ、次を起動させること。

Window について

・どのアクティビティも描画用のデフォルトのウィンドウをあたえられている。
・通常は、ウィンドウはスクリーンいっぱいにひろがっている。しかしスクリーンより小さく、ウィンドウの最前面にフロートさせることもできる。
・あるアクティビティは、追加のウィンドウを利用可能。例えば、ユーザーの応答を得るポップアップダイアログ、または、重要な情報をユーザーに伝えるウィンドウなど

Widget について

・ウィンドウのビジュアルな内容は、View クラスから派生したオブジェクトにより提供される。
・どのビューもウィンドウ内の特定の矩形スペースをコントロールする。
・親のビューは子供のレイアウトを含み、体系化している。
・末端のビュー (階層の最下層) は矩形にコントロールしユーザーのアクションに反応して描画を行う。
・ビューは、アクティビティとユーザの意思疎通が行われる場所
・例えば、ビューが小さなイメージを表示し、ユーザーがそのイメージをタップすることで、アクションを開始する。
・Android は数多くの既製のビューを持っている。ボタン、テキスト、スクロールバー、メニュー、チェックボックスなど

・ビューの階層は、アクティビティのウィンドウに、Activity.setContentView により置かれる。
・Content View は View オブジェクトのルート。さらなる情報

■ Service

サービスとは？

・サービスは、ビジュアルなユーザーインターフェースを持たず、バックグラウンドで不定期に起動される。
・例えば、ユーザーが何かを行っているときにバックグラウンドで音楽を再生する、ネットワーク越しのデータの取得やなんらかの計算、必要となったときに結果を提供できるようにしておく。
・すべてのサービスは、Service 基底クラスを継承している。

・上記例にて、メディアプレーヤーがプレイリストから曲を再生することをあげた。プレーヤーアプリケーションは、おそらく、ユーザーに曲を選択させてそれを再生するなど 1 つ以上のアクティビティ、を持つだろう。
・しかしながら、音楽がアクティビティにより音楽自身を再生することはできない。
・なぜなら、ユーザーはプレイヤーから去った後も音楽が再生され続け、何か別なことが始まるのを期待している。
・音楽を再生したままで、メディアアプレイヤーのアクティビティはサービスをバックグラウンドで走らせることができる。
・システムは音楽再生サービスをそれを開始したアクティビティが、スクリーンから消えても行い続ける。
・実行中のサービスに接続（バインド）できる。（そしてそのサービスが開始されていなければ、開始する）
・接続中、公開されているサービスインターフェースを通してサービスと通信できる
・この音楽サービスでは、インターフェースは一時停止、巻き戻し、停止、再開をユーザーに許すだろう
・アクティビティや他のコンポーネント、サービスはアプリケーションプロセスのメインスレッドで実行される
・なので、それらは、他のコンポーネント、ユーザーインターフェースをブロックすることはない。しばしば、他のスレッドを時間のかかるタスクのために呼び出す。

■ Broadcast Receiver

・ブロードキャストレシーバーは、ブロードキャストの受信と再送信以外なにもしないコンポーネント
・多くのブロードキャストはシステムコードから送信される。例えば、タイムゾーンの変更通知、バッテリーの低下、写真がとられた、ユーザーが言語設定を変えた。など
・アプリケーションも、ブロードキャストをはじめられる。例えば、なにがしかのデータのデバイスへのダウンロードが完了し、利用可能になったことを、他のアプリケーションに知らせる。
・アプリケーションは、重要と考えれば、いくつでもどんな通知にも対応するようブロード

キャストレシーバーをもつことができる。
・すべてのブロードキャストレシーバーは、BroadcastReciever 基底クラスを継承する。
・ブロードキャストレシーバーはユーザーインターフェースを表示しない。
・しかしながら、受け取った情報へのレスポンスとして、アクティビティを起動することがある。もしくは、NotificatonManager を使って、ユーザーに通知する。
・通知は、いろいろな方法でユーザーの注意をひくことができる。バックライトのフラッシュ、デバイスを振動させる、音をならす。など
・典型的には永続的なアイコンをユーザーがメッセージを受け取れるようにステータスバーに配置する

■ Content providers

・コンテントプロバイダーは他のアプリケーションから利用できるように特定のアプリケーションのデータを作成する。
・データは SQLite データベースなど、ファイルシステムに保存できる。
・コンテントプロバイダーは、ContentoProvider 基底クラスを継承し、他のアプリケーションが受け取り保存できるデータ型を制御する標準的なメソッドを実装する
・しかしながら、アプリケーションは、これらのメソッドを直接呼び出さない。
・その代わり、ContentResolver オブジェクトを 利用し変わりに呼び出す。
・ContentResolver はどんなコンテントプロバイダーにも通信できる。プロバイダーと協力してプロセス間のコミュニケーションを管理する。

コンポーネントを活性化する インテント

・コンテントリゾルバからのリクエスト対象とされたとき、コンテントプロバイダーは活性化され
・ほかの３つのコンポーネント ー アクティビティ、サービス、ブロードキャストレシーバー ー は インテント非同期メッセージにより活性化される。
・インテントは、Intent オブジェクトであり、メッセージ内容を保持する。
・アクティビティとサービスの為に、とりわけ、インテントはリクエストされたアクションを命名し、操作対象となるデータの URI を特定する。
・例えば、インテントはイメージをユーザーに提示またはユーザーにテキストを編集させるアクティビティへリクエストを運ぶ
・ブロードキャストレシーバーには、インテントオブジェクトは通知されたアクションに命名する。
・例えば、カメラボタンが押されたことを、関係者に通知する。

・それぞれのコンポーネントをアクティベートするために別々のメソッドがある

■アクティビティとインテント

・インテントオブジェクトを Context.startActivity、または、Activity.startActivityForResult() に渡すことによりアクティビティは起動される。
・応答アクティビティは、自身の getIntent() メソッドにより起動された初期インテントを見ることができる。
・Android は アクティビティの onNewIntent() メソッドをそれをつづくインテントに渡すために呼び出す。
・一つのアクティビティはしばしば次を開始する。アクティビティから結果が変えることが想定される場合、startActivity() の代わりに、startActivityForResult() により開始する。
・例えば、ユーザーが写真を選択することでアクティビティが開始され、選択された写真を結果として受け取ることが想定される場合。
・結果はアクティビティの onActivityResult() メソッドから、インテントオブジェクトとしてかえってくる。