

Androidアプリケーション解体新書

Androidアプリケーションには以下の4つの構成ブロックがあります。

- Activity(アクティビティ)
- Intent Receiver(インテントレシーバー)
- Service(サービス)
- Content Provider(コンテンツプロバイダ)

すべてのアプリケーションが上記の4つ全てが必要なわけではなく、いくつかを組合せてアプリケーションを作成します。

まず、あなたのアプリケーションにどんなコンポーネントが必要かを決めて、AndroidManifest.xmlと呼ばれるファイルに記述する必要があります。

これは、あなたのアプリケーションの能力と必要とするコンポーネントを宣言するXMLファイルです。詳細は[AndroidManifest.xml?](#)を見てください。

Activity(アクティビティ)

アクティビティは、4つのAndroidの構成ブロックの中で最も一般的です。アクティビティは、通常あなたのアプリケーションの一つの画面を意味します。各々のアクティビティは、[Activity?](#)ベースクラスを拡張した単独のクラスとして実装されます。あなたのクラスは[Views?](#)から成るユーザインタフェースを表示することや、イベントに対応しています。大部分のアプリケーションは、複数の画面から構成されています。

たとえば、テキストメッセージングアプリケーションは、送信先のコンタクトリスト画面、メッセージ作成画面、メッセージ閲覧画面、設定変更画面を持っているかもしれません。これらのスクリーンの各々は、アクティビティとして実行されます。

新しいアクティビティの起動によって別のスクリーンに移る、アクティビティは前のアクティビティに値を返す場合があります。たとえば、ユーザーに写真を選ばせるアクティビティはコール元に戻された写真を返します。新しいスクリーンが開くとき、前画面はヒストリースタックの上に中断されて、置かれます。

ユーザーは、ヒストリーで直前にオープンされたスクリーンによって後ろに操縦することができます。スクリーンは、彼らが残ることが不適当であるヒストリースタックから取り外されるほうを選ぶこともできます。アンドロイドは、ホームスクリーンから起動されるアプリケーションごとに、ヒストリースタックを保持します。

Intent and Intent Filters(インテントとインテントフィルター)

アンドロイドは、スクリーンからスクリーンへ移るインテントと呼ばれる特別なクラスを使用します。インテントは、アプリケーションが何がされることを望むかについて述べます。インテントデータ構造の2つの最も重要な部分は、作用する行動とデータです。行動のための典型的な値はMAIN(アクティビティの前のドア)、VIEW、PICK、EDIT、その他です。そして、データはURIとして表されます。たとえば、人への連絡先を見るために、あなたはその人を代表しているURIに、VIEWアクションとデータセットでインテントをつくるでしょう。

IntentFilterと呼ばれる関連したクラスが、あります。インテントが効果的に何かする要請である間、インテントフィルタはインテントアクティビティ(または、インテントレシーバ、下記参照)が取り扱うことができるものの説明です。人への連絡先を示すことができるアクティビティは、人を代表しているデータに適用されると、それがアクションVIEWを取り扱う方法を知っていると、言ったインテントフィルタを発行します。アクティビティは、彼らのIntentFiltersをAndroidManifest.xmlファイルで発表します。

映るためにスクリーンから操縦することは、インテントを解決することによって達成されます。前方へ操縦するために、アクティビティはstartActivity(myIntent)を呼びます。システムは、それからすべてのインストールされたアプリケーションのためにインテントフィルタを見て、試合が最もインテントフィルタをmyIntentする活動を選択します。新しいアクティビティは意図を知らされます。それはそれがラウンチされる原因になります。startActivityが呼ばれるとき、インテントを分解するプロセスは実行時で起こります。そして、それは2つの鍵となるベネフィットを提供します：

- アクティビティは、単にインテントのリクエストフォームの中に作ることによって、他の構成要素から機能を再利用することができます
- アクティビティは、等しいインテントフィルターでいつでも新しいアクティビティと取り替えられることができます

Intent Receiver(インテントレシーバー)

例えば、電話が着信した時、ネットワークが利用可能になった時、真夜中になった時といった外部のイベントに反応するアプリケーションを作成したい時には、IntentReceiverを使えます。インテントレシーバーはUIを表示しません。ただし、注意を引くことが発生したらユーザーに知らせるために、インテントレシーバーはNotificationManagerを使うでしょう。インテントレシーバーは、

AndroidManifest.xmlに登録されますが、Context.registerReceiver()を使うことによって、コード中で登録することもできます。アプリケーションは、コールされる時に、そのアプリケーションのインテントレシーバーが動いている必要はありません; システムは、必要であればそのアプリケーションを起動し、インテントレシーバーが開始されます。アプリケーションは、Context.broadcastIntent()を使うことによって、そのアプリケーション独自のインテントを他のインテントレシーバーにブロードキャストすることもできます。

Service(サービス)

サービスは、永続的に、UIなしで動くコードです。サービスのよい例として、プレイリストから曲を再生するメディアプレイヤーがあります。メディアプレイヤーアプリケーションでは、多分、ユーザーに曲を選択させたり、選択された曲の再生を開始させたりするための、1個かそれ以上のアクティビティがあるでしょう。しかし、ユーザーは、新たな画面に移動した後も、音楽を再生し続けることを期待するでしょうから、音楽再生自体がアクティビティによって制御されるべきではありません。このケースでは、メディアプレイヤーアクティビティはバックグラウンドで音楽を再生し続けるために、Context.startService()を使うことによって、サービスを起動します。システムは音楽が終わるまで再生サービスを走らせつづけます。(システムでサービスにプライオリティを与える方法についての詳細は、[Androidアプリケーションのライフサイクル?](#)を参照してください。)Context.bindService()メソッドでサービスに接続できる(そしてサービスがまだ走っていないならば、起動できる)ことに注意してください。サービスに接続中は、そのサービスによって公開されたインタフェースを介して、あなたはそれと通信できます。音楽サービスにおいてこのことは、あなたに音楽の一時停止や巻き戻し等の余地を与えるかもしれないことを示します。

Content Provider(コンテンツプロバイダー)

アプリケーションは自身のデータをファイル、SQLiteデータベース、その他の筋が通ったメカニズムに格納できます。また一方、コンテンツプロバイダーは、あなたのアプリケーションのデータを他のアプリケーションと共有したい場合に有効です。コンテンツプロバイダーは、他のアプリケーションが、そのコンテンツプロバイダーによって操作されるデータの型の格納と取り出しができるようにするためのメソッドの標準セットを実装するクラスです。

コンテンツプロバイダーについての更なる詳細は、[Accessing Content Providers?](#)を参照してください。