Neste capítulo, será apresentada a Plataforma Android, especificando suas principais características, assim como sua infra-estrutura, arquitetura e principais recursos.

O que é o Android

“O Android é uma plataforma de desenvolvimento para aplicativos móveis como smartphones e contêm um sistema operacional baseado em Linux, uma interface visual rica, GPS, diversas aplicações já instaladas e ainda um ambiente de desenvolvimento bastante poderoso, inovador e flexível”. [Ricardo Lecheta].

Principais Características

O Android é formado por um conjunto de ferramentas que estão incluídas no SDK (Software Development Kit) e uma infra-estrutura que possibilita o desenvolvimento de aplicações para os dispositivos móveis. O SDK é o responsável por controlar o ciclo de vida das aplicações e a utilização dos recursos do dispositivo, como por exemplo, no gerenciamento de memória e consumo da bateria.

As aplicações para o Android são criadas através da linguagem de programação JAVA.

Arquitetura

Recursos Android

API

Aplicações

Componentes Básicos

Ciclo de vida

Entender como funciona o ciclo de vida de uma Activity (tela ou atividade) é fundamental para criarmos aplicações mais robustas. Todo o ciclo de vida de uma Activity é gerenciado pelo Sistema Operacional Android (SO). O SO quando necessita de mais recursos acaba interrompendo a execução de alguma Activity, liberando assim os recursos que antes estavam sendo utilizados pela mesma. Essa atitude do SO é justificável já que estamos nós referindo a um dispositivo móvel, onde, os recursos disponíveis são limitados como, capacidade de armazenamento (memória) e poder de processamento, e que para manter um bom funcionamento da aplicação esses recursos têm que ser bem gerenciado.

A figura xx mostra todo o ciclo de vida de uma Activity e todos os métodos que são chamados por cada faze do ciclo.

Os principais métodos do ciclo de vida de uma Activity são: onCreat, onStart, onResume, onPause, onStop, onRestart e onDestroy.

O método onCreat é o responsável por criar a Activity e é executada somente uma única vez, ele pode ser utilizado para abrir arquivos que serão utilizados pela Activity, por exemplo. No momento da chamada do método onStart, já deve existir a Activity e seus componentes visuais, então ele tem a função de exibir a Activity ao usuário, porem sem oferecer a interação com a mesma. A partir do método onResume que a Activity pode interagir com o usuário, ou seja, nesta faze dizemos que a Activity está em execução.

Já o método onPause, é chamado sempre que a Activity corrente for interrompida por uma outra Activity ficando temporariamente suspensa e salvando o seu estado atual, para que, quando esta voltar a executar, possa recuperar tudo, se necessário, nó método onResume. O método onStop é chamado quando a Activity corrente, está sendo executada em segundo plano ou está sendo eliminada porque não será mais utilizada. Podendo nessa faze a Activity ser reiniciada com a chamada do método onRestart ou finalizada com a chamada do método onDestroy.

É importante notar que o SO pode, nas fazes “Parada” e “Interrompida” do ciclo, finalizar alguma Activity que está muito tempo sem ser executada, para liberar recursos. E que também, no ciclo da Activity, os únicos métodos que não são executados mais de uma vez são os métodos onCreat e o onDestroy.

Layout

Classe R

É uma classe JAVA, que ao criar um novo projeto, é gerada automaticamente pela ferramenta de desenvolvimento (plugin ADT ou MOTODEV) e gerenciada pela mesma. A classe R nunca deve ser alterada manualmente. Sua atualização acontece quando um novo arquivo, não importando o seu tipo, é adicionado em alguma das seguintes pastas: drawable, layout e values. Essas pastas contêm respectivamente, imagens, arquivos XML que representam as telas da aplicação e arquivos XML que contém as mensagens da aplicação. Então, ao adicionar um novo recurso em uma dessas pastas, é gerada automaticamente uma nova constante na classe R.

Como podemos perceber, a classe R serve de acesso aos recursos da aplicação, pois, nela são guardadas constantes, do tipo int e de valor único dentro do projeto, que são utilizadas para referenciar algum recurso do projeto, que pode ser: uma imagem, botão ou um arquivo XML que define alguma tela da aplicação.

O acesso aos recursos do projeto usando a classe R funciona da seguinte maneira. Por padrão a nova constante recebe o nome da pasta que contém o recurso, mais o nome do arquivo sem sua extensão. Nesse caso teríamos que, para acessar a imagem “android.png” que está na pasta drawable, faríamos R. drawable.android. Só mais um detalhe, na hora de acessar um recurso usando a classe R a pasta e o arquivo deve estar em minúsculo e não deve conter espaços.

Arquivo AndroidManifest

O Arquivo AndroidManifest é a base de toda aplicação Android. Ele é um arquivo de configuração, obrigatório em todo o projeto Android, contendo permissões de acesso ao Hardware, como, sensores e câmera, assim como o acesso a internet. Além disso, é necessário que toda Activity do projeto esteja declarada nesse arquivo, caso contrário não será possível utilizá-la.

Na figura XX é mostrado um exemplo de Arquivo AndroidManifest.xml, onde é pedido a permissão para a aplicação ter acesso a internet.

Como podemos perceber na figura XXX, a tag <manifest> tem como atributo o nome do pacote (package), código da versão (android:versionCode) e nome da versão (versionName) e dentro dela fica toda a configuração da aplicação. Na linha 7, temos a permissão para acessar a localização via GPS. Dentro da tag <application> irá ficar todas as Activitys que fazem parte da aplicação.

Lembrando que, o conteúdo do arquivo AndroidManifest.xml, de penderá de cada aplicação Android.

Ferramentas

As ferramentas de programação ou IDEs, são úteis para auxiliar o desenvolvedor em suas tarefas, proporcionando aumento da produtividade, reduzindo o tempo de desenvolvimento.

Para o desenvolvimento de aplicações para a o ambiente Android tem-se o plugin ADT. Este é instalado na IDE Eclipse e toda sua configuração é feita manualmente.

Outra alternativa seria, o MOTODEV Studio. O MOTODEV Studio é uma versão modificada do Eclipse criada pela Motorola. Já vem com tudo configurado e com mais recursos, pronto para o desenvolvedor começar a criar suas aplicações sem se preocupar com a configuração. Além disso, o MOTODEV tem mecanismo de detectar problemas na aplicação como é o caso do validador de aplicativo, que tem o papel de detectar problemas na aplicação para garantir que a aplicação irá funcionar como deveria. Por exemplo, fazendo buscas no código por condições impróprias, permissões faltantes ou layouts impróprios.