**INSTITUTO FEDERAL DA PARAÍBA – CAMPUS CAJAZEIRAS**

**CURSO DE ANÁLISE E DESENVOLVIMENTO DE SISTEMAS**

**DISCIPLINA DE PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS**

**EXERCÍCIO DE FIXAÇÃO**

***ASSUNTO:*** *INTRODUÇÃO A DESENVOLVIMENTO PARA ANDROID*

1) **src** – Arquivos de código-fonte Java estarão disponíveis aqui;

**gen** – Contém arquivos gerados automaticamente, como o R.java que contém referências a certos recursos do projeto;

**res** – Recursos como imagens e certos arquivos de configuração XML estão aqui dentro e estes podem ser mantidos separados do código-fonte;

**/res/values** – Usado para definir *strings*, cores, dimensões e *arrays* estáticos de *strings* or *integers*;

**/res/animator** – Contém animações em XML;

**/res/layout** – Contém os layouts a serem usados na aplicação;

**/res/menu** – Contém recursos de menu a serem usados na aplicação;

**/res/raw** – Contém recursos *raw* que podem ser verificados pelos seus IDs;

**/res/drawable** – Fornece diferentes *drawables* de bitmap para telas de média, alta e extremamente alta densidade;

**libs** – Arquivos de biblioteca externa serão localizados nesta pasta;

**assets** – Contém hierarquia *raw* de arquivos e diretórios, sem outras capacidades;

**bin** – Área usada pelo compilador para preparar os arquivos para serem empacotados no APK, que pode-se encontrar neste diretório.

2) A classe R é aquela que contém as definições para todos os recursos de um pacote de aplicação particular. Ela está no *namespace* do pacote da aplicação. Ex: Se você disser no seu *manifest* que o nome do seu pacote é *com.foo.bar,* uma classe R é gerada com os símbolos de todos os seus recursos em *com.foo.bar.R*.

3) **manifest** - Possui um atributo *package* para definir o pacote do projeto;

**uses-permission** – Declara permissões que você determinou que sua aplicação precisa para operar de forma apropriada;

**permission** – Para definir uma restrição a qualquer componente da aplicação você utiliza essa tag, além de definir o atributo *android:permission*. Além disso nessa *tag* você pode definir o nível de acesso que será permitido, um *label*, e um recurso externo;

**application** – Esta *tag* usa atributos para especificar meta-dados para a sua aplicação *android*;

**activity** – Para cada atividade exibida na aplicação, uma *tag ‘activity’* é requerida. Para definir o nome da classe, usamos o atributo *android:name*, além disso essa *tag* dá suporte ao uso de *intent-filters* que especificam que *intent* inicializará a atividade;

**service –** Para cada serviço da classe, essa *tag* tem de ser criada. Também dá suporte aos *intent-filters*;

**provider** – Devem ser criadas para cada um dos *Content Providers* da aplicação; estes são utilizados para gerenciar o acesso à base de dados e compartilhamento dentro e entre aplicações;

**receiver** – Pode-se registrar um *Broadcast Receiver* acrescentando essa *tag* sem ter iniciado primeiro a aplicação;

4) **build** – Diretório de saída para os arquivos compilados e onde o APK será gerado;

**libs** – Arquivos de biblioteca externa serão localizados nesta pasta;

**src** – Arquivos de código-fonte Java estarão disponíveis aqui;

**/src/androidTest** – Aqui ficarão as classes de teste que serão executadas em um dispositivo Android de fato;

**/src/main** – Representa o diretório de código-fonte principal;

**/src/test** – Aqui ficarão as classes de teste que serão executadas localmente na máquina.

5) **Kernel Linux** – Camada de baixo nível do SO, fornece a funcionalidade básica do sistema tal como: gerenciamento de processos, de memória e de dispositivos, como também fornece uma gama de drivers de dispositivos que faz nossa tarefa ficar mais fácil quando constrói uma interface entre o Android e os dispositivos periféricos;

**Bibliotecas** – Camada acima do Kernel Linux, que fornece diferentes bibliotecas úteis para o bom funcionamento do SO. Tais bibliotecas usam Java, construídas especificamente para o Android;

***Android Runtime*** – Terceiro componente da arquitetura Android, fornece a parte mais importante do Android, a máquina virtual. Ela é similar a JVM, porém essa foi desenhada e otimizada para Android. Além disso ela utiliza funções de núcleo do Linux tais como: gerenciamento de memória e *multithreading* e permite que cada aplicação Android execute seu próprio processo.

**Framework de aplicação** – Localiza-se acima da *Android Runtime* na arquitetura. As aplicações Android interagem diretamente com essa camada. Ela gerencia as funções básicas do dispositivo Android tais como: gerenciamento de recursos, de chamadas de voz;

**Aplicações** – As aplicações criadas por terceiros (usuários ou desenvolvedores) serão instaladas nessa camada.