# INF01043 - Tutorial Android

## Instalando o ambiente de programação

1) Baixe e instale o Java SE Development Kit.

http://www.oracle.com/technetwork/java/javase/downloads/index.html

## Instalando as ferramentas de desenvolvimento

Baixe e descompacte o Android SDK Tools.

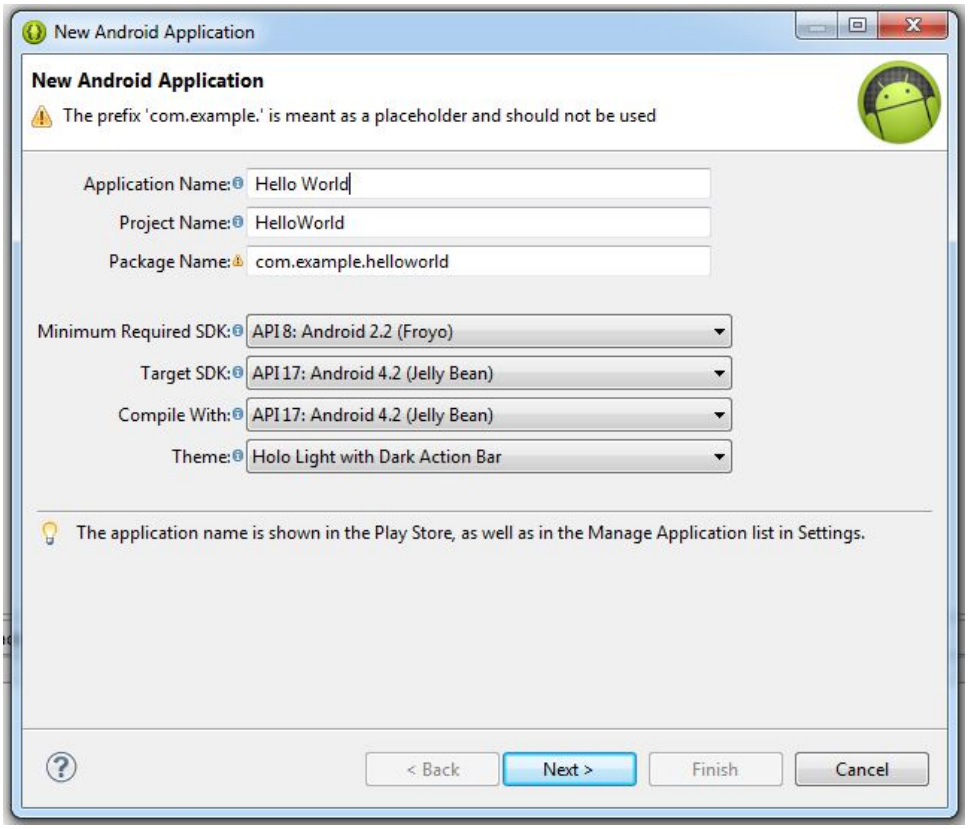
http://developer.android.com/sdk/index.html

Va até a pasta aonde você descompactou os arquivos e inicie o eclipse.

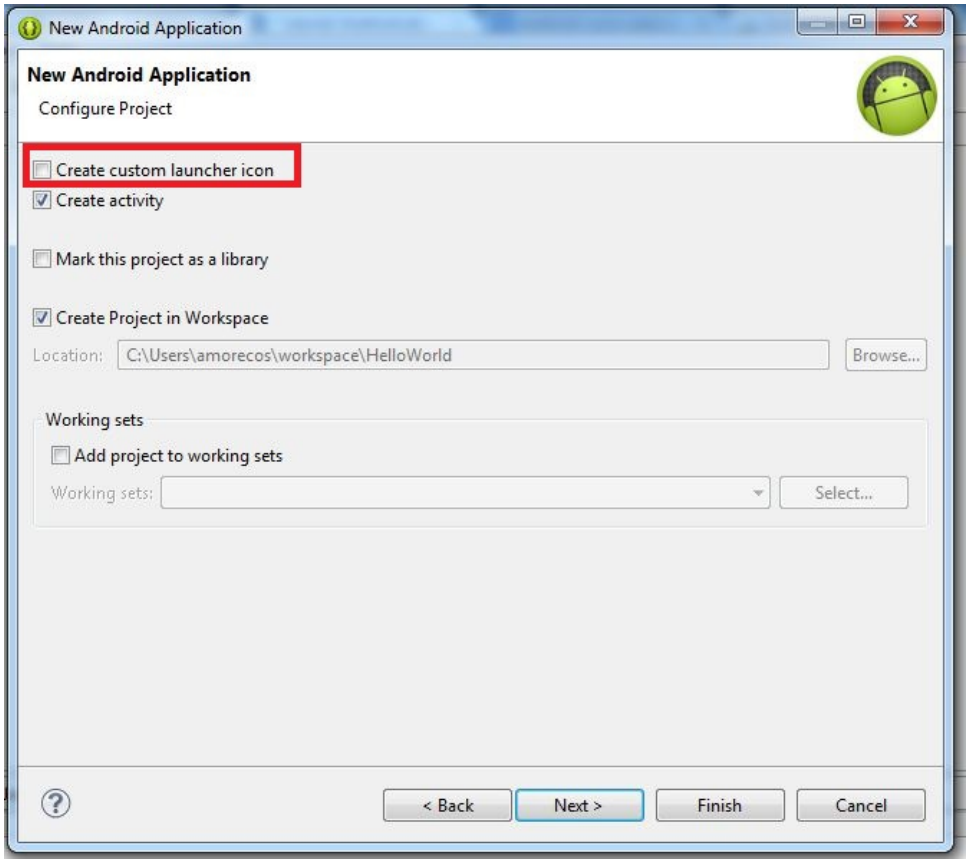
## Criando um projeto

Para a criação de um projeto vá em File>New>Android Application Project

Escolha um nome para a aplicação e mude o nome do pacote e clique em next.

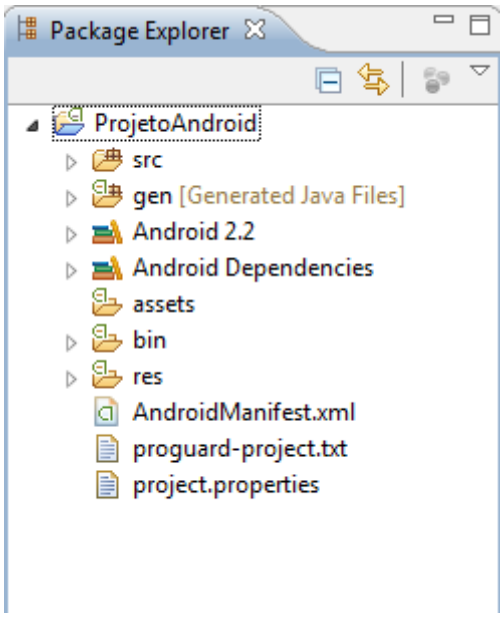



Desmarque a opção de criação de ícone personalizado e clique em next.

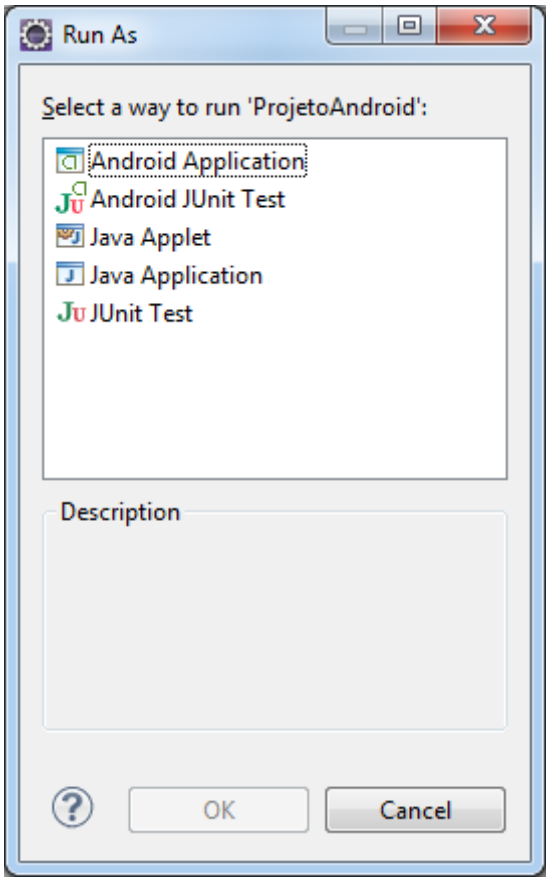



Nas próximas duas telas apenas clique em next e depois finalize a criação do projeto.

O projeto estará organizado da seguinte forma:



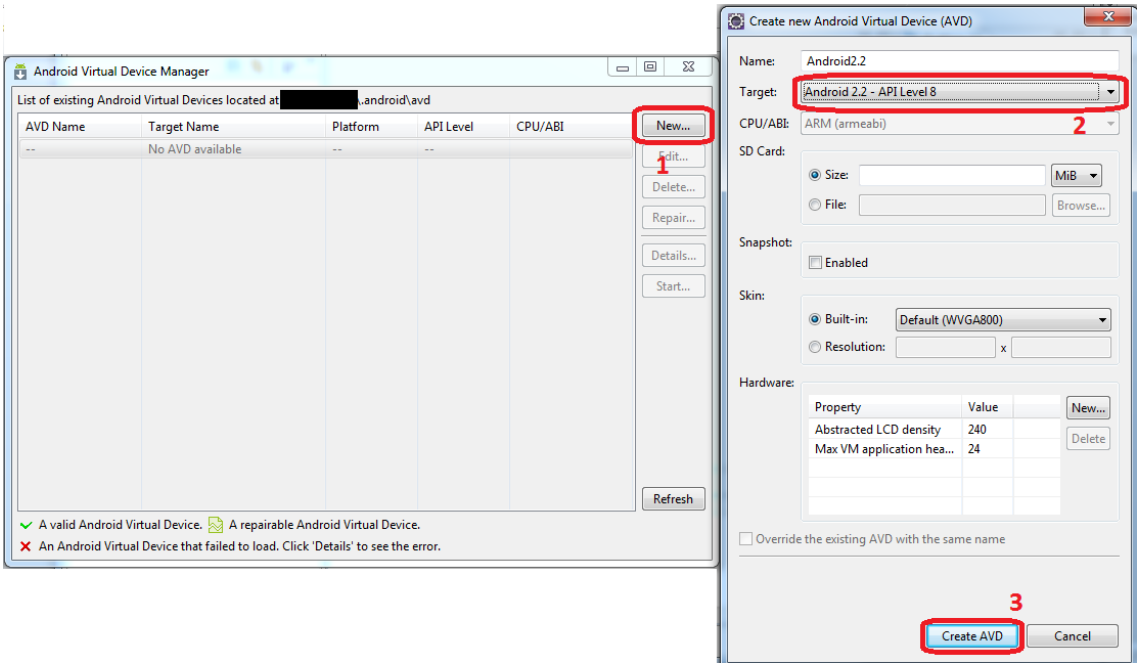
Para testar o projeto clique em *'Run'*. A primeira execução do projeto abrirá uma janela para escolher a forma de execução da aplicação. Selecione *'Android Application'*.




Em seguida serão listados os dispositivos/emuladores criados no gerenciador. Os tópicos **Android Virtual** e **Android Device** mostrarão como criar e configurar o emulador/dispositivo android.

## Android Vitual

Em *Window>AVD Manager* abra o gerenciador de dispositivos virtual android e adicione um novo dispositivo no botão *'New'*. Na nova janela adicione o nome para o seu dispositivo e a versão android instalada.

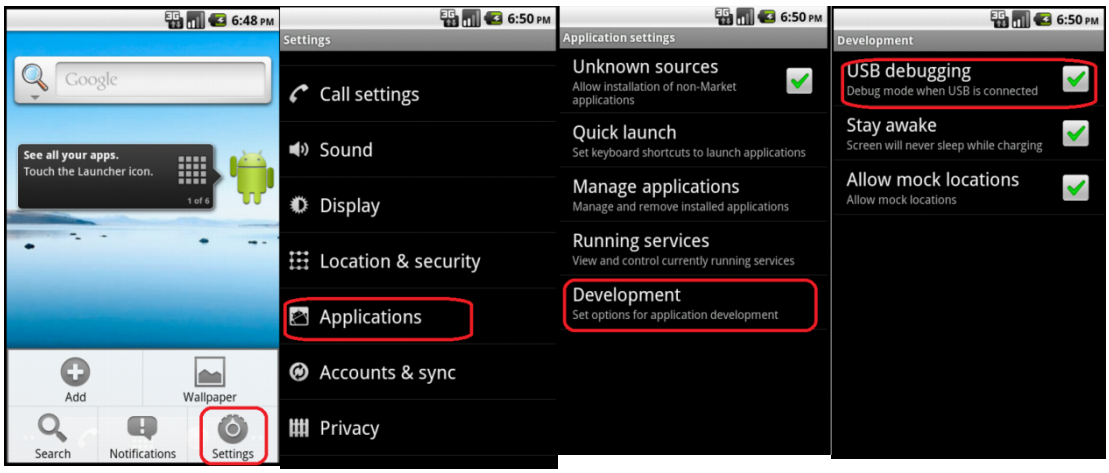



Para abrir o dispositivo selecione-o em seguida clique em *'Start'*.

## Android Device

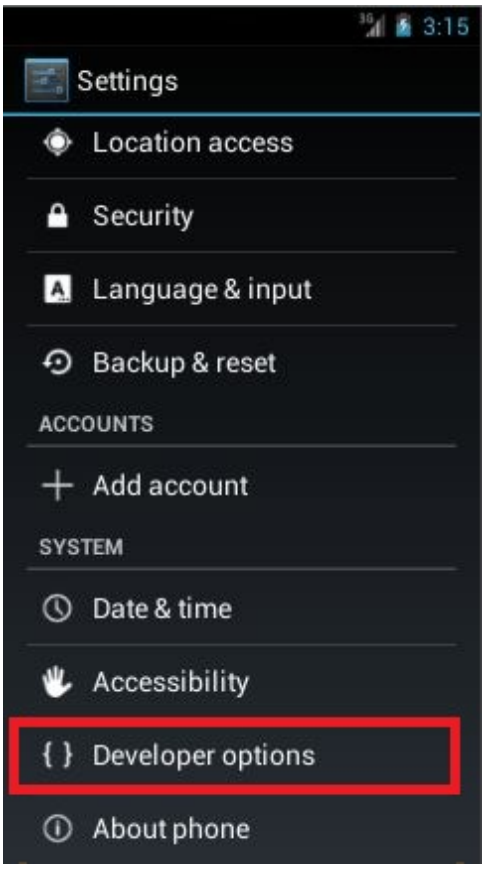
## Para as versões de Android anterioes a 4.0

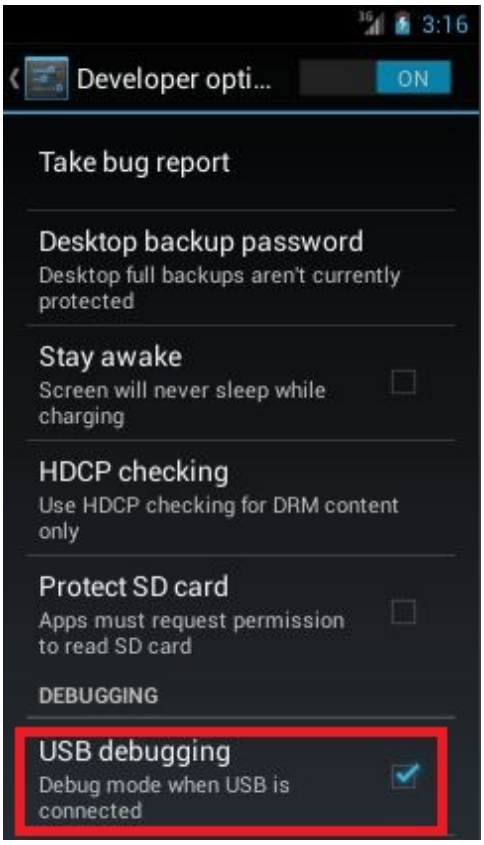
Para a execução da aplicação no dispositivo android é necessário habilitar a USB em modo de depuração. Conecte o dispositivo na porta USB e siga o caminho segundo seu dispositivo. *Menu>Configurações>Aplicativos>Desenvolvimento* e deixe marcada a opção de Depuração USB.



## Para as versões de Android a partir da 4.0 até a 4.1.2

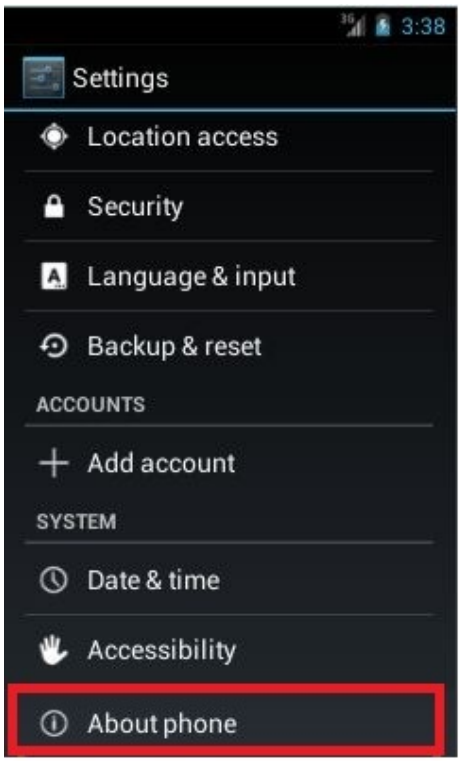
Para a execução da aplicação no dispositivo android é necessário habilitar a USB em modo de depuração. Conecte o dispositivo na porta USB e siga o caminho segundo seu dispositivo. *Menu>Configurações>Opções do Desenvolvedor* e deixe marcada a opção de Depuração USB.

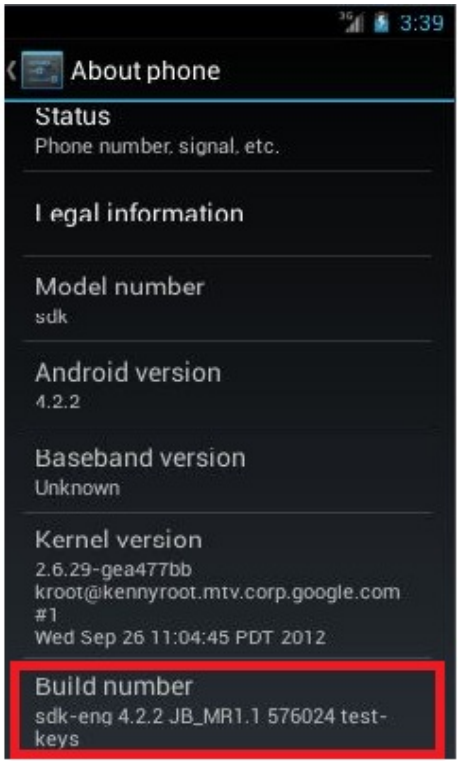


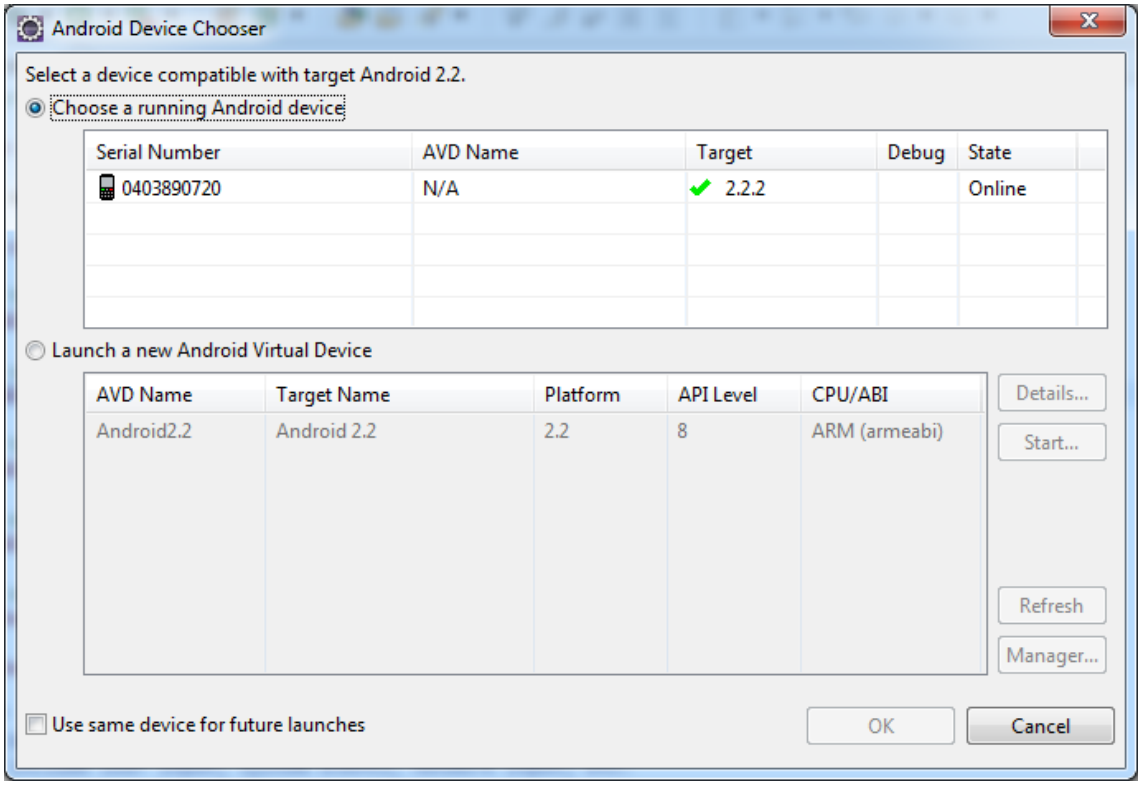

## Para as versões de Android a partir da 4.2

A partir da versão 4.2 do Android, o menu de opções de desenvolvimento vem oculto, para exibir este menu é necessário entrar em *Configurações>Sobre o telefone,* depois rolar a tela até o fim e clicar 7 vezes sobre build number. Após seguir os passos da versão 4.0 para habilitar a execução no dispositivo.




Volte ao eclipse e clique em *'Run'* para executar o projeto. Agora aparecerá o dispositivo conectado no computador além dos emuladores criados. Selecione o dispositivo para carregar a aplicação.




**Como criar uma activity**

**Como interagir com um elemento da interface**

Primeiro devemos ir no arquivo xml do Layout aonde está o componente que nos interessa, e verificarmos qual o id deste elemento.

```
<Button
    android:id="@+id/button1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@+id/textView1"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="113dp"
    android:text="Button" />
```

Na activity, criamos uma variável do mesmo tipo do componente, e utilizamos a função findViewById para associamor a nossa variável com o componente.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    Button b1 = (Button) findViewById(R.id.button1);
}
```

**Como adicionar um callback para capturar um evento de um elemento da interface**

Em primeiro lugar, criamos uma referência para o elemento que desejamos adicionar um callback seguindo os passos descritos acima. Depois setamos o evento que desejamos capturar e sobrescrevemos a função que trata este evento.

```
b1 = (Button) findViewById(R.id.button1);
b1.setOnClickListener(new OnClickListener()
{
    @Override
    public void onClick(View arg0) {
        // TODO Auto-generated method stub
    }
});
```

**Como chamar uma outra activity**

Para chamarmos uma outra activity, é necessário criar uma intent com o tipo da activity que desejamos chamar, e utilizamos o comando startActiviy passando esta intent.

```
Intent t = new Intent(MainActivity.this, SecondActivity.class);
startActivity(t);
```

**Como passar dados para outra activity**

Para passarmos dados para outra activiy, seguimos os passos anteriores e criamos uma intent com o tipo da activity que queremos passar a mensagem.

A intent tem uma estrutura do tipo chave-valor, o qual podemos utilizar para passar a mensagem. Para adicionarmos a mensagem usamos o comando putExra, passando como primeiro parâmetro a chave e o segundo a mensagem.

```
Intent t = new Intent(MainActivity.this, SecondActivity.class);
t.putExtra("msg", ed1.getText().toString());
```

**Como obter a intent que foi utilizado para chamar uma activity**

Na função OnCreate da activity, utilizamos a função getIntent para obtermos a intent que foi utilizada para iniciar a activity.

Para obter os dados enviado pela outra activity, utilizamos a função get?Extra, substituindo a ? pelo tipo do dado que foi enviado, passando como parâmetro para a função a chave do valor que desejamos obter.

```
Intent intent = getIntent();
String text = intent.getStringExtra("msg");
```

**Como obter os dados dos sensores de movimento**

Primeiro é necessário fazermos a nossa activity implementar a classe SensorEventListener. Quando implementamos esta classe, obrigatoriamente devemos incluir dois métodos na activity:onSensorChanged e onAccuracyChanged.

O primeiro método será chamado toda vez que houver alguma alteração nos valores dos sensores que estamos monitorando, o segundo método é chamado quando ocorre alguma mudança na precisão dos valores obtidos, e não nos interessa por enquanto.

Para a nossa activity começar a escutar os sensores, é necessário registrar um listener com os sensores que desejamos escutar e com qual frequência desejamos receber os eventos.

```
public class SecondActivity extends Activity implements SensorEventListener{

    SensorManager sensorManager = (SensorManager) getSystemService(SENSOR_SERVICE);

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_second);
        sensorManager.registerListener(this, sensorManager.getDefaultSensor(Sensor.TYPE_GYROSCOPE),
                sensorManager.SENSOR_DELAY_NORMAL);
    }

    @Override
    public void onSensorChanged(SensorEvent event) {
        // TODO Auto-generated method stub

        if(event.sensor.getType() == Sensor.TYPE_GYROSCOPE)
        {

            float sensorX = event.values[0];
            float sensorY = event.values[1];
            float sensorZ = event.values[2];
        }
    }
```