



# Introdução à Programação Android



Guilherme Antonio Borges

[guilhermeborges.pf@gmail.com](mailto:guilhermeborges.pf@gmail.com)

# Apresentação

---

- ▶ Graduado em 2012 no Curso de Tecnologia em Sistemas para Internet no IFSUL campus Passo Fundo.
- ▶ Atualmente, estudante de Mestrado em Computação na UFRGS;
  - ▶ Grande área: Sistemas Distribuídos;
  - ▶ Interesse em: Computação Móvel e Ubíqua, Sistemas autoadaptativos e Sensoriamento Urbano;
- ▶ Experiência com Android:
  - ▶ 2010 – Cursado minicursos sobre o S.O Android;
  - ▶ 2011/2 e 2012/1 – Projeto de inovação tecnológica, cardápios eletrônicos utilizando S.O. Android;
  - ▶ 2011 e 2012 – 2 Minicursos ministrados;
  - ▶ 2012 – Cadeira de 60 horas sobre Programação para Dispositivos Móveis;
  - ▶ 2013 – Trabalhando Projeto de Pesquisa na UFRGS em sistemas distribuídos;
  - ▶ 2013 e 2014 – Ministrado Minicursos – Senai
- ▶ Contato:
  - ▶ Google Plus: <https://plus.google.com/+GuilhermeAntonioBorges>
  - ▶ E-mail: [guilhermeborges.pf@gmail.com](mailto:guilhermeborges.pf@gmail.com)

# Sumário

---

- ▶ Introdução
  - ▶ Plataforma Android
  - ▶ Referências/Fontes
- ▶ Ambiente de Desenvolvimento
- ▶ Criando Aplicações
- ▶ Implementação
- ▶ Apêndice
  - ▶ Exercícios Extras
  - ▶ Tutorial de Instalação e configuração
  - ▶ Tutorial de importação de projetos
  - ▶ Dicas

# Links

---

- ▶ Códigos Fonte exemplo da Apresentação:

- ▶ <https://drive.google.com/file/d/0B746pjy4jQVAV0o3ajhSaTBVZjA/view?usp=sharing>

# Introdução

# Computação Móvel

---



# Utilidade

---

- ▶ Monitoramento de Saúde;
- ▶ Monitoramento Veicular;
- ▶ Interação Social;
- ▶ Comércio;
- ▶ Jogos;
- ▶ Agenda;
- ▶ Trabalhar;
- ▶ Gerenciadores de conteúdo; e
- ▶ Gerir informações pessoais.

# Plataformas Móveis

---

- ▶ Distribuição de dispositivos por segmento
  - ▶ PC (Desk-Based and Notebook)
  - ▶ Ultramobile
  - ▶ Tablet
  - ▶ Mobile Phone (Smartphones)
- ▶ Sistemas Operacionais
  - ▶ Android
  - ▶ Windows
  - ▶ iOS/macOS
  - ▶ RIM (BlackBerry)
  - ▶ ...



# Plataforma Android

---

## ▶ Android

- ▶ Sistema operacional móvel que roda sobre o núcleo Linux
  - ▶ Projeto inicial da Google
  - ▶ Depois Open Handset Alliance (OHA)
  - ▶ Open Source
  - ▶ Lançado oficialmente em 2007
- 
- ▶ SDK possui um conjunto de bibliotecas e API's de simples acesso e fácil entendimento.

# Plataforma Android

---

## ▶ Open Handset Alliance (OHA)

- ▶ Aliança entre as empresas: Google, HTC, Dell, Intel, Motorola, Qualcomm, Texas Instruments, Samsung, LG, T-Mobile e Nvidia.
- ▶ Objetivo de Criar padrões abertos para telefonia móvel.
- ▶ Plataforma open source Android – SDK (Nov, 2007).
- ▶ <http://www.openhandsetalliance.com>

# SDK Manager

---

- ▶ É necessário utilizar o Android SDK, pois ele possui:
  - ▶ Bibliotecas da API Android
  - ▶ Ferramentas necessárias para criar, testar e depurar apps para Android
  - ▶ Criação de Emuladores
- ▶ Link:
  - ▶ <http://developer.android.com/sdk/index.html>

# Versões (Curiosidade)

---

► Nomes das versões do Android são doces:



**Cupcake**  
Android 1.5



**Donut**  
Android 1.6



**Eclair**  
Android 2.0/2.1



**Froyo**  
Android 2.2



**Gingerbread**  
Android 2.3



**Honeycomb**  
Android 3.0



**Ice Cream Sandwich**  
Android 4.0



**Jelly Bean**  
Android 4.1/4.3

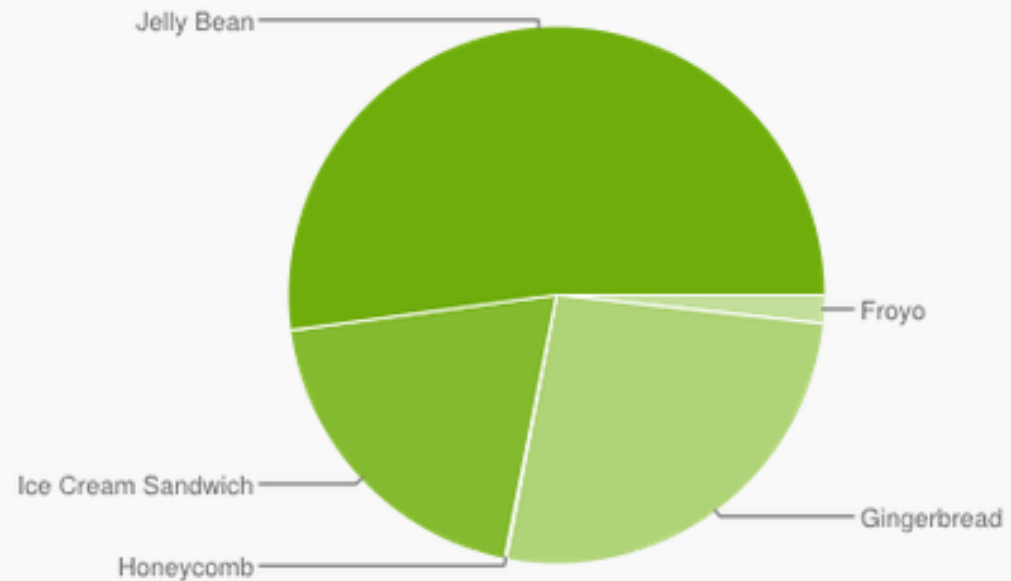


**KitKat**  
Android 4.4

# Versões

---

Version	Codename	API	Distribution
2.2	Froyo	8	1.7%
2.3.3 - 2.3.7	Gingerbread	10	26.3%
3.2	Honeycomb	13	0.1%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	19.8%
4.1.x	Jelly Bean	16	37.3%
4.2.x		17	12.5%
4.3		18	2.3%



*Data collected during a 7-day period ending on November 1, 2013.  
Any versions with less than 0.1% distribution are not shown.*

► <http://developer.android.com/about/dashboards/index.html>



# API Level

---

- ▶ Cada versão do Android possui uma API Level
  - ▶ É importante para referenciar as versões do Android
- ▶ A API Level é um valor inteiro que identifica uma versão do Android.
- ▶ A plataforma Android fornece uma estrutura de API que os aplicativos podem usar para interagir com o Sistema Android

## A estrutura API consiste em:

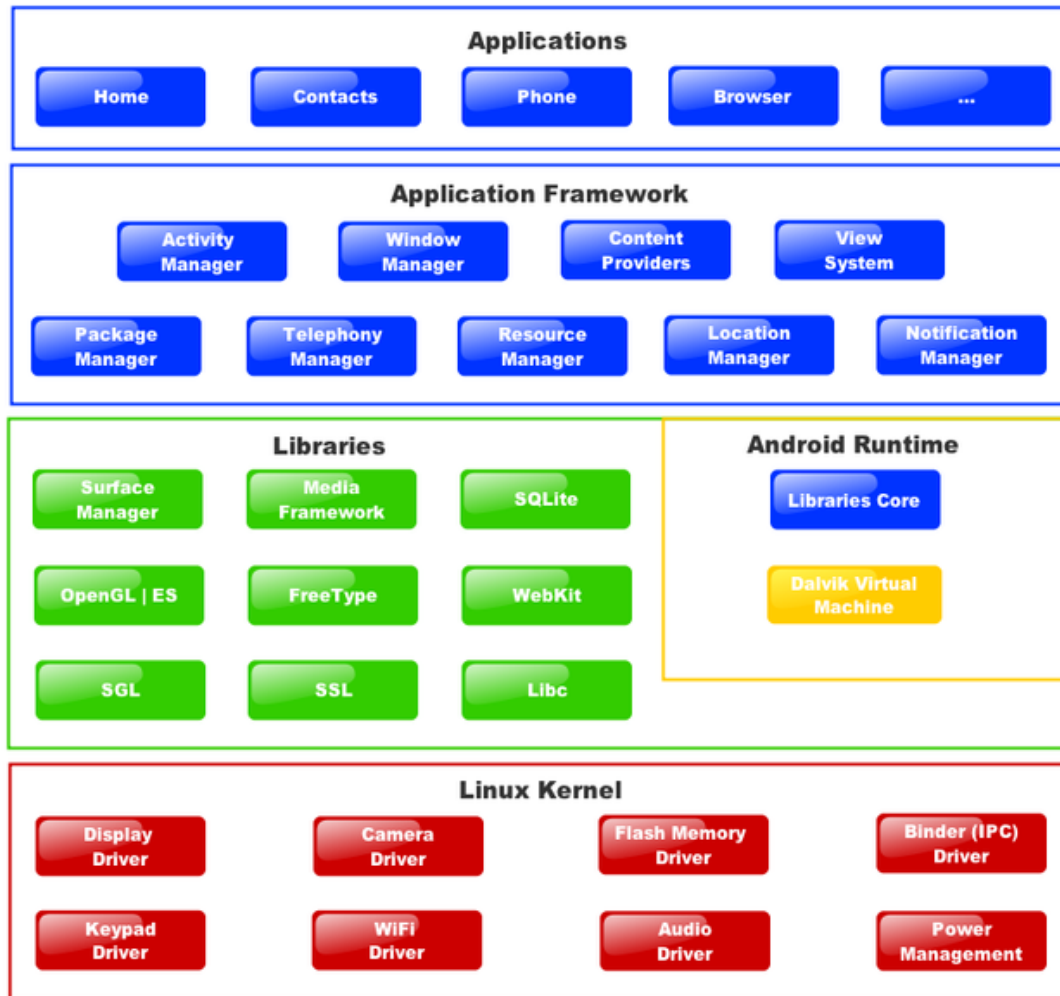
---

- ▶ Conjunto de pacotes e classes
- ▶ Conjunto de elementos e atributos XML para declarar um manifest file (arquivo de configuração da aplicação no Android)
- ▶ Conjunto de elementos e atributos XML para declarar e acessar recursos
- ▶ Conjunto de intenções
- ▶ Conjunto de permissões que os aplicativos podem solicitar, bem como a autorização incluída no sistema
- ▶ Cada versão sucessiva da plataforma Android pode incluir atualizações e com a API Level é possível identificar sua utilização no mesmo



# Arquitetura

---





# Máquina Virtual

---

- ▶ Aplicações escritas em Java são compiladas em bytecodes Dalvik e executadas usando a Máquina Virtual Dalvik.
- ▶ Máquina Virtual Dalvik
  - ▶ Máquina virtual especializada desenvolvida para uso em dispositivos móveis
  - ▶ Baseada em registradores
  - ▶ Otimizada para utilizar pouca memória
  - ▶ Permite que múltiplas instâncias da MV rodem ao mesmo tempo, deixando para o SO o isolamento de processos, o gerenciamento de memória e o suporte a threading.

# Principaux Classes

---

- ▶ Activity
- ▶ Intent
- ▶ View

# Outras Classes/Funcionalidades

---

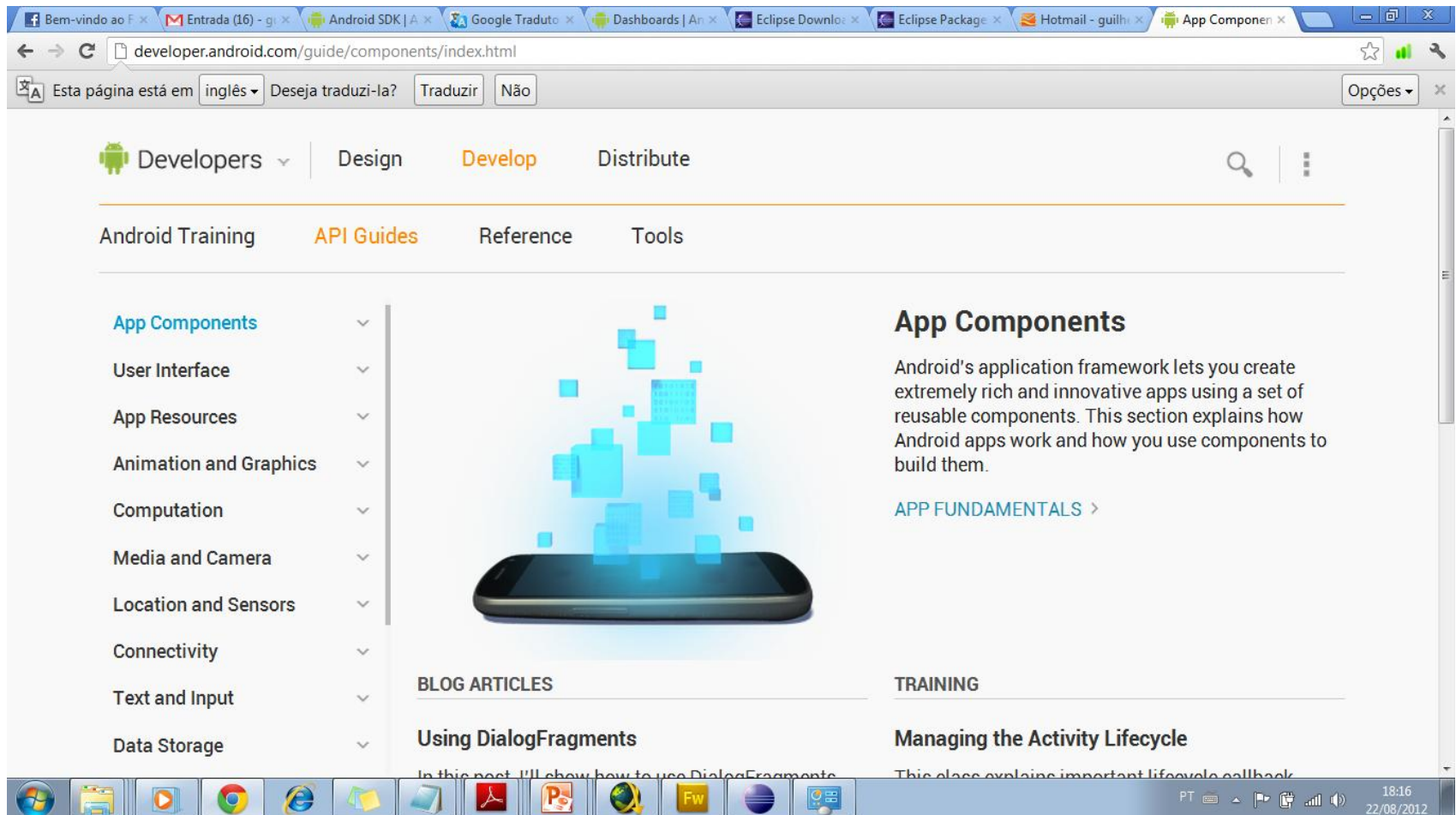
- ▶ BroadcastReceiver
- ▶ Notification
- ▶ Service
- ▶ AlarmManager
- ▶ Handler, Threads e AsyncTask
- ▶ Banco de dados (SQLite)
- ▶ Content Provider
- ▶ Mapas e GPS
- ▶ Http, sockets e Web Services
- ▶ SMS
- ▶ Google Cloud Messaging
- ▶ Reconhecimento de Gestos
- ▶ Sensores
- ▶ Bluetooth, OpenGL, NDK ...



# Referências

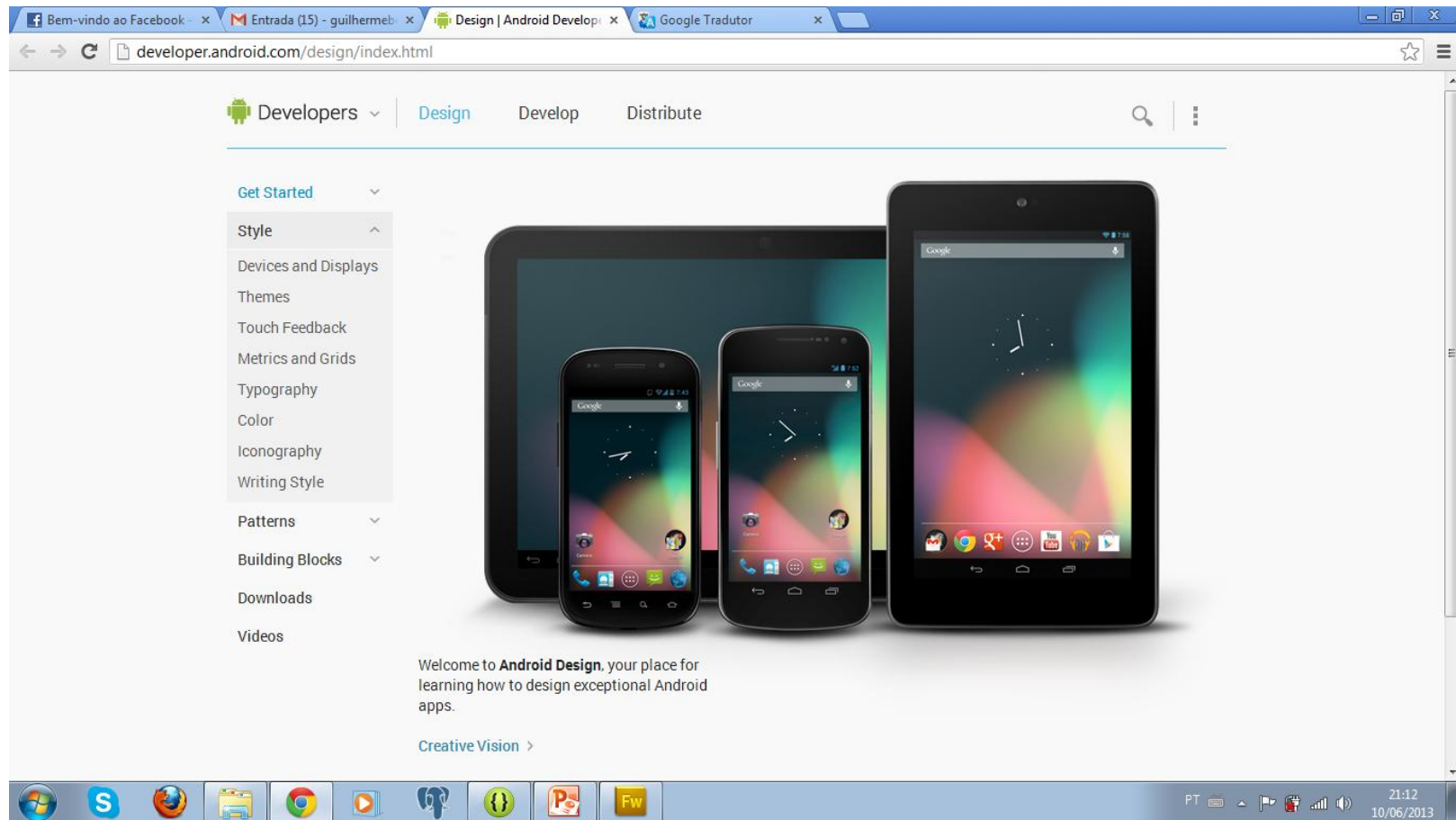
# Referência On-line

► <http://developer.android.com/guide/>



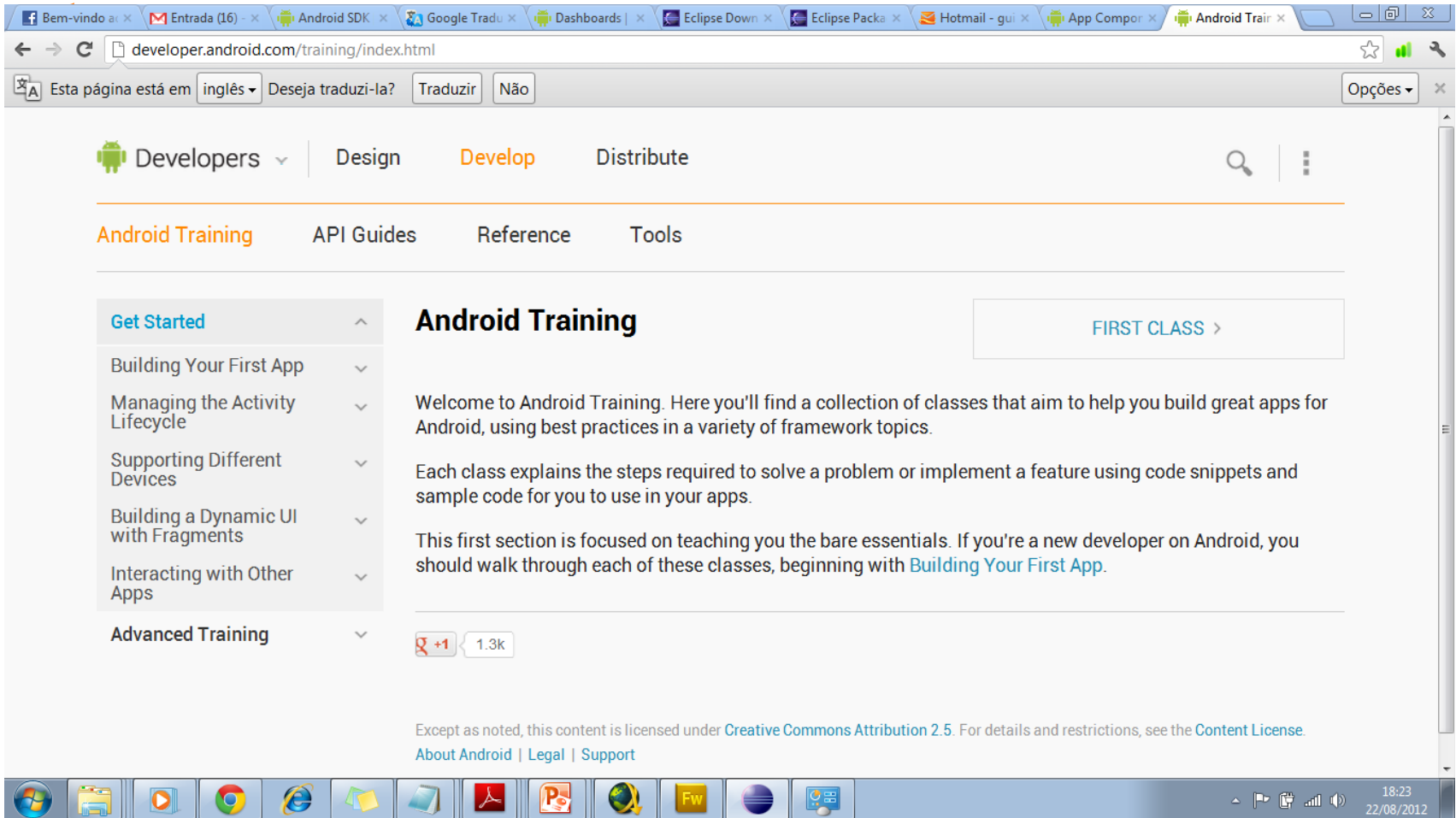
# Referência para Estilo

- ▶ <http://developer.android.com/design/index.html>



# Tutoriais

► <http://developer.android.com/training/index.html>



The screenshot shows a web browser window displaying the Android Developer Training page. The browser's address bar shows the URL `developer.android.com/training/index.html`. The page features a navigation bar with links for **Design**, **Develop** (highlighted), and **Distribute**. Below this, there are links for **Android Training**, **API Guides**, **Reference**, and **Tools**. The main content area is titled **Android Training** and includes a **FIRST CLASS >** button. The page describes the training content, stating that it aims to help developers build great apps for Android using best practices. It mentions that each class explains the steps required to solve a problem or implement a feature using code snippets and sample code. The first section is focused on teaching the bare essentials, and it suggests starting with [Building Your First App](#). A sidebar on the left lists various topics under **Get Started** and **Advanced Training**. At the bottom, there is a Creative Commons Attribution 2.5 license notice and links for [About Android](#), [Legal](#), and [Support](#).

Bem-vindo a... | Entrada (16) - x | Android SDK x | Google Tradu x | Dashboards | x | Eclipse Down x | Eclipse Packa x | Hotmail - gui x | App Compor x | Android Train x

← → ↻ `developer.android.com/training/index.html` ☆ 📶 🔍

Esta página está em **inglês** | Deseja traduzi-la? Traduzir Não Opções x

**Developers** ▾ | **Design** | **Develop** | **Distribute** 🔍 ⋮

**Android Training** | API Guides | Reference | Tools

**Get Started** ▴

- Building Your First App ▾
- Managing the Activity Lifecycle ▾
- Supporting Different Devices ▾
- Building a Dynamic UI with Fragments ▾
- Interacting with Other Apps ▾


**Advanced Training** ▾

**Android Training** **FIRST CLASS >**

Welcome to Android Training. Here you'll find a collection of classes that aim to help you build great apps for Android, using best practices in a variety of framework topics.

Each class explains the steps required to solve a problem or implement a feature using code snippets and sample code for you to use in your apps.

This first section is focused on teaching you the bare essentials. If you're a new developer on Android, you should walk through each of these classes, beginning with [Building Your First App](#).

 +1 1.3k

Except as noted, this content is licensed under [Creative Commons Attribution 2.5](#). For details and restrictions, see the [Content License](#).

[About Android](#) | [Legal](#) | [Support](#)

Windows taskbar: [Icons for various applications] 18:23 22/08/2012

# Livros

---

Google Android para Tablets 2ª Ed. (2012)

<http://www.novateceditora.com.br/livros/android-tablets/>



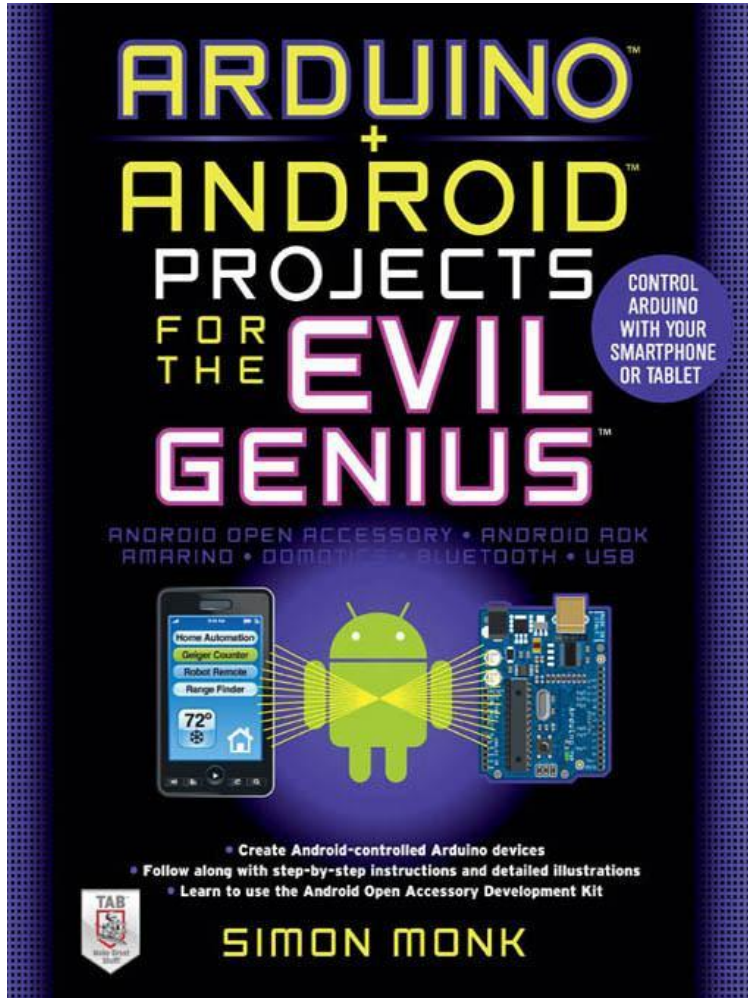
Google Android 3ª Ed. (2013)

<http://novatec.com.br/livros/googleand3/>



# Livro Interessante

---



- ▶ Livro interessante para quem quer “brincar” com alguns projetos entre Arduino e Android.
- ▶ É necessário conhecimento prévio em Android.
- ▶ Livro disponível somente em inglês.

# Referências

---

- ▶ <https://sites.google.com/site/rossettopf/pdm-6m1>
- ▶ <http://www.slideshare.net/joseberardo>
- ▶ <http://www.k19.com.br/downloads/apostilas/java/k19-k41-desenvolvimento-mobile-com-android>

# Ambiente de Desenvolvimento

# Ambiente de Desenvolvimento

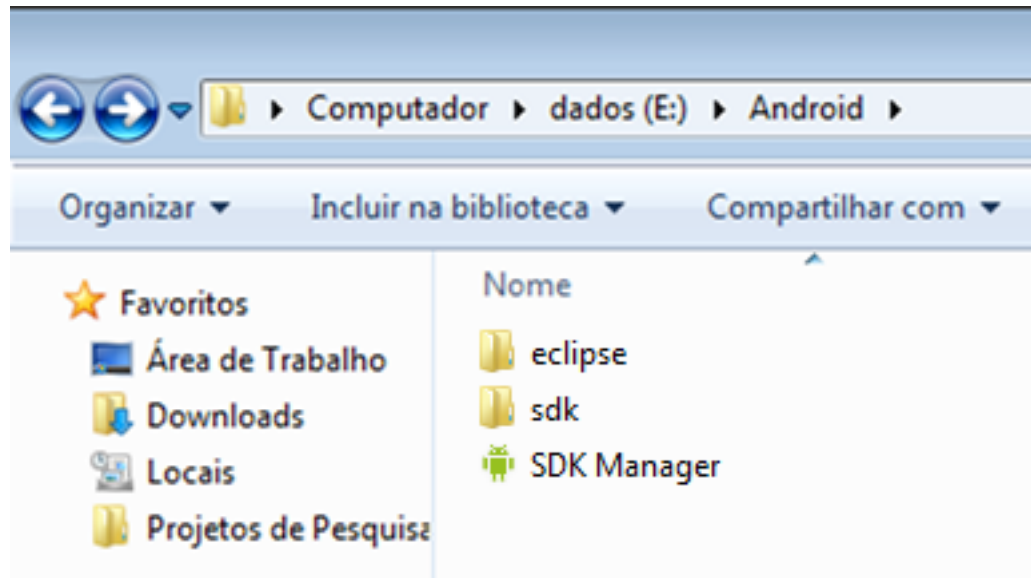
---

- ▶ SDK (Software Development Kit) do Android
  - ▶ API 20 – Android 4.4W – KitKat Wear
- ▶ Eclipse IDE
  - ▶ 4.2 – Juno – <http://www.eclipse.org/juno/>
  - ▶ Android Developer Tools –  
<http://developer.android.com/sdk/index.html>
- ▶ Android Development Tools Plugin (ADT)
  - ▶ Plugin para o Eclipse IDE – Versão atual: 22.3
  - ▶ <http://developer.android.com/tools/sdk/eclipse-adt.html>

# Sugestão (Para Android Developer Tools)

---

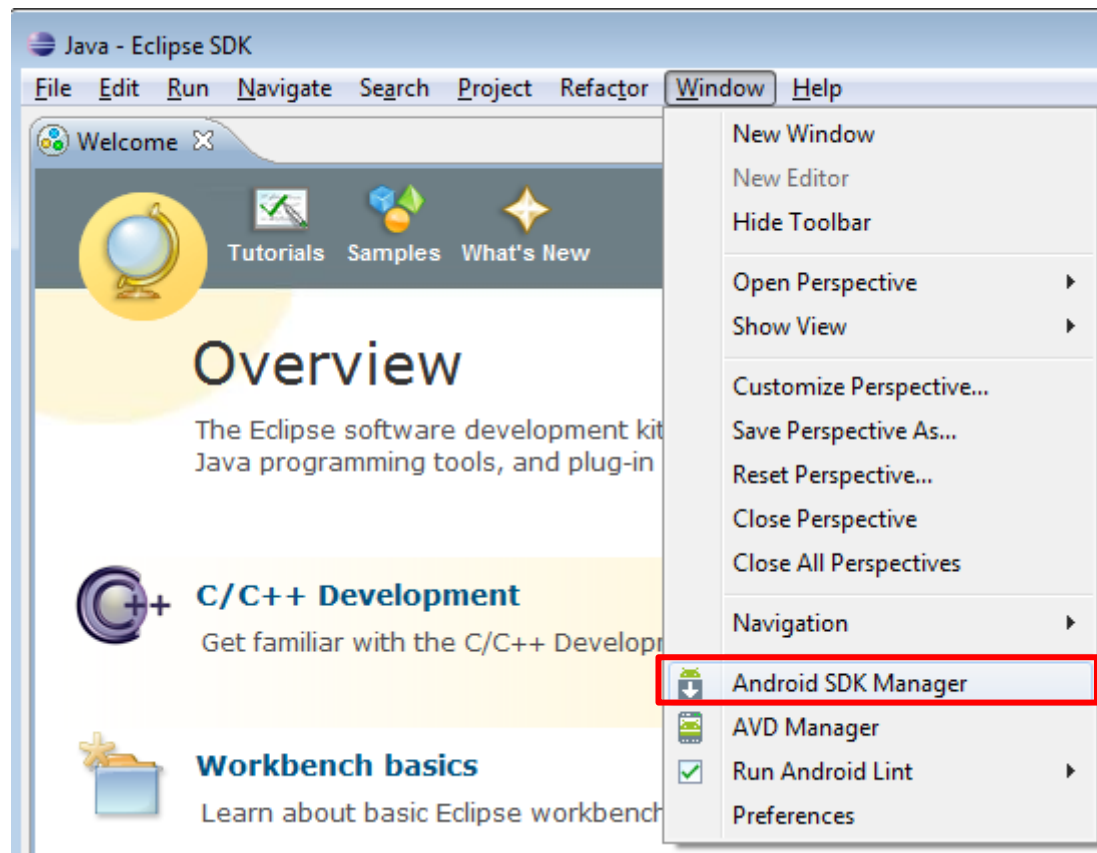
- ▶ Criar pasta C:\android
- ▶ Descompactar o Android Developer Tools



# Configuração

---

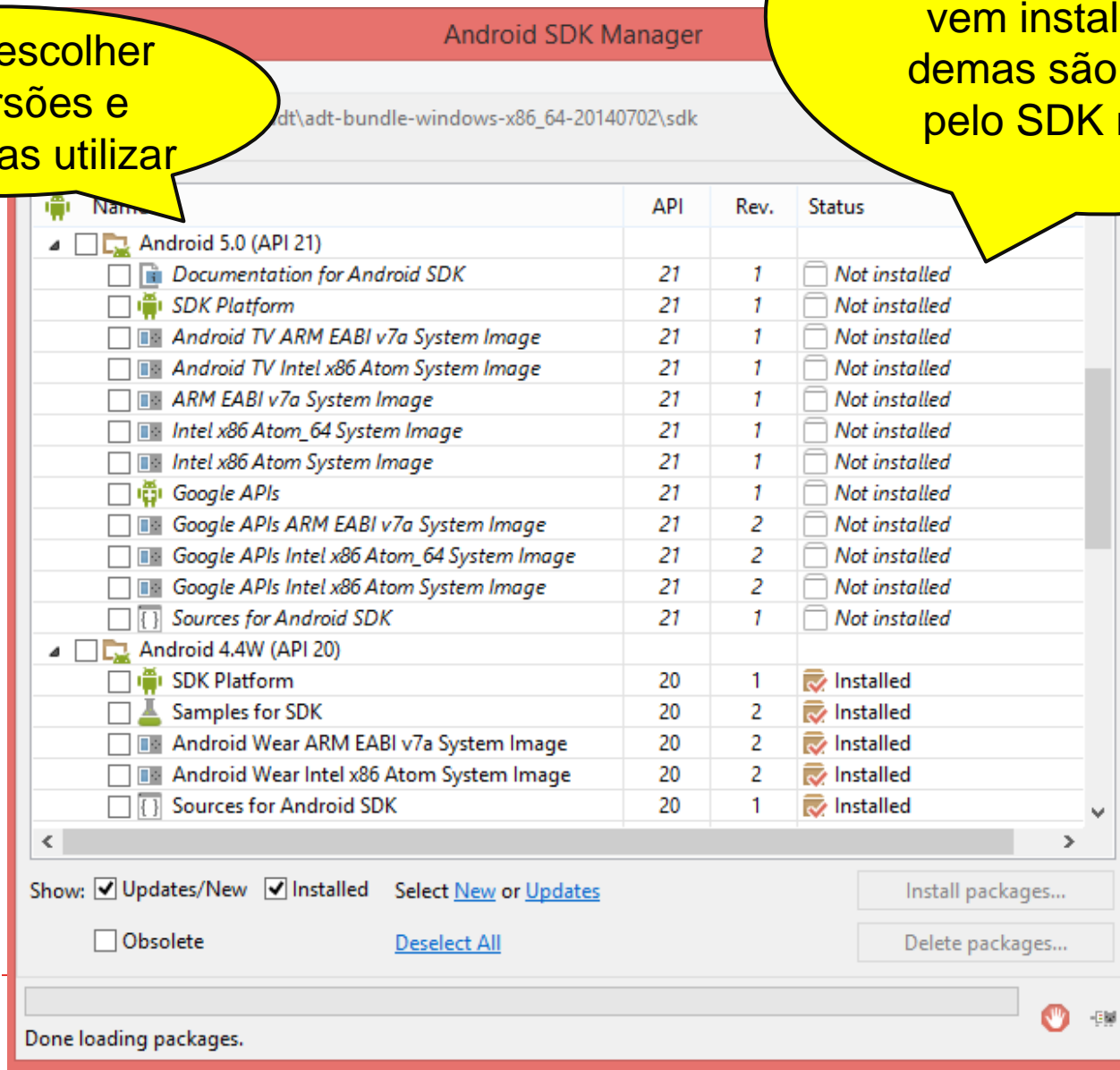
- Depois de Instalar o ADT Plugin, Executar pelo eclipse o Android SDK Manager



# Android SDK Manager

Permite escolher  
que versões e  
ferramentas utilizar

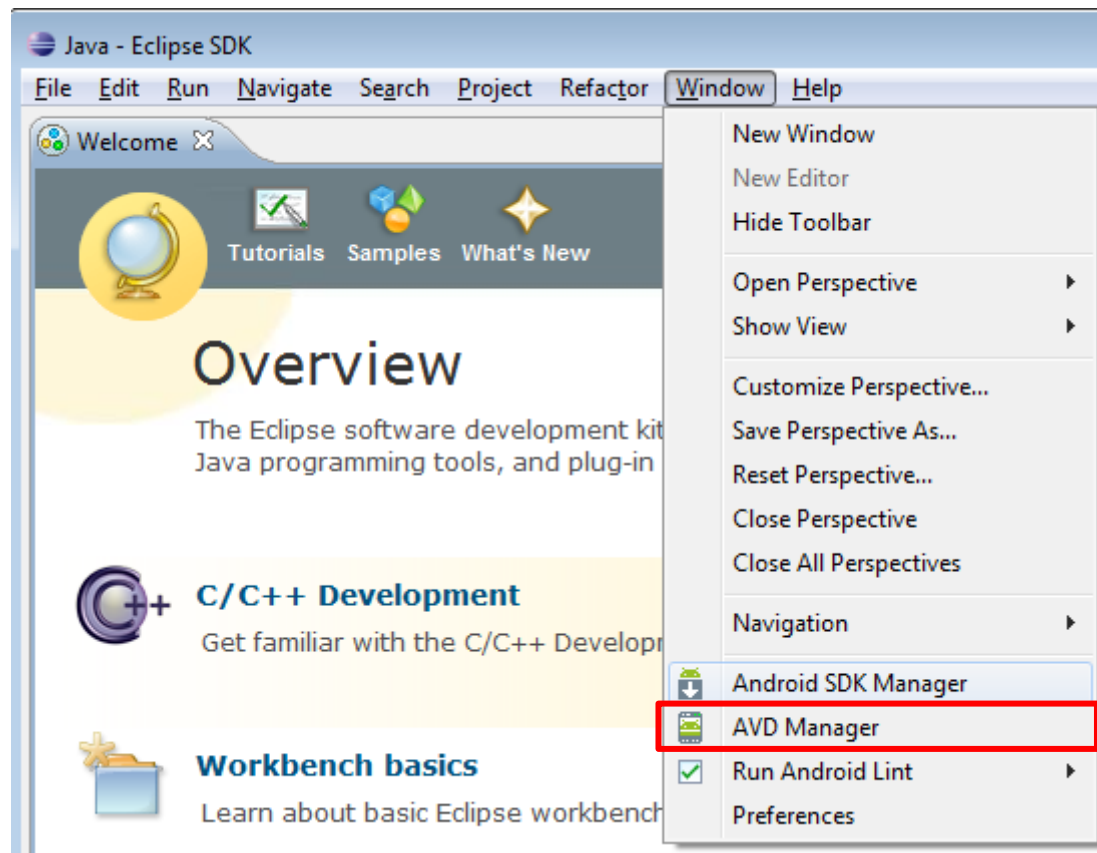
Originalmente somente  
as ferramentas básicas  
vem instaladas, as  
demais são baixadas  
pelo SDK manager



# Configuração

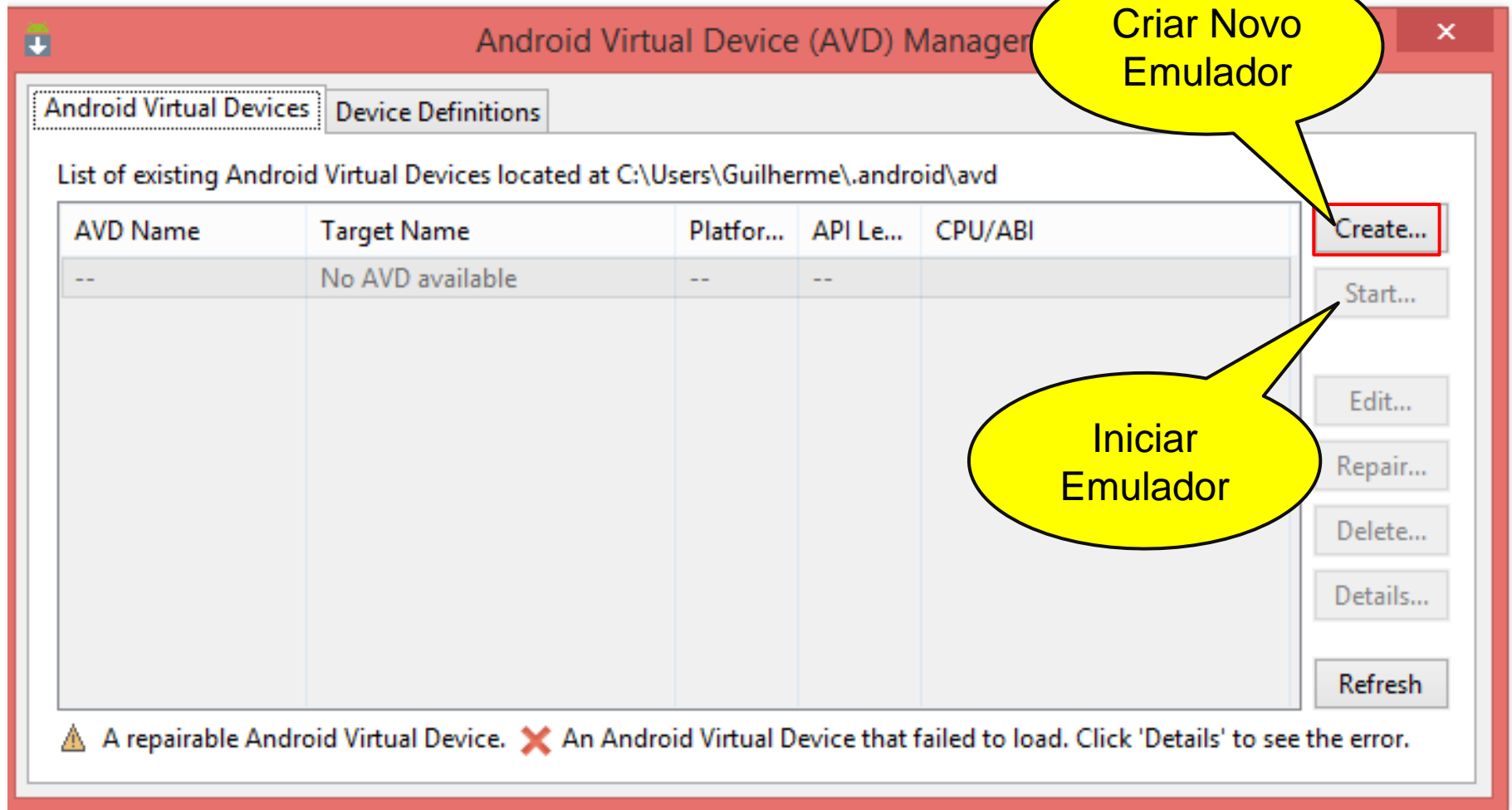
---

- Acessar o AVD SDK Manager



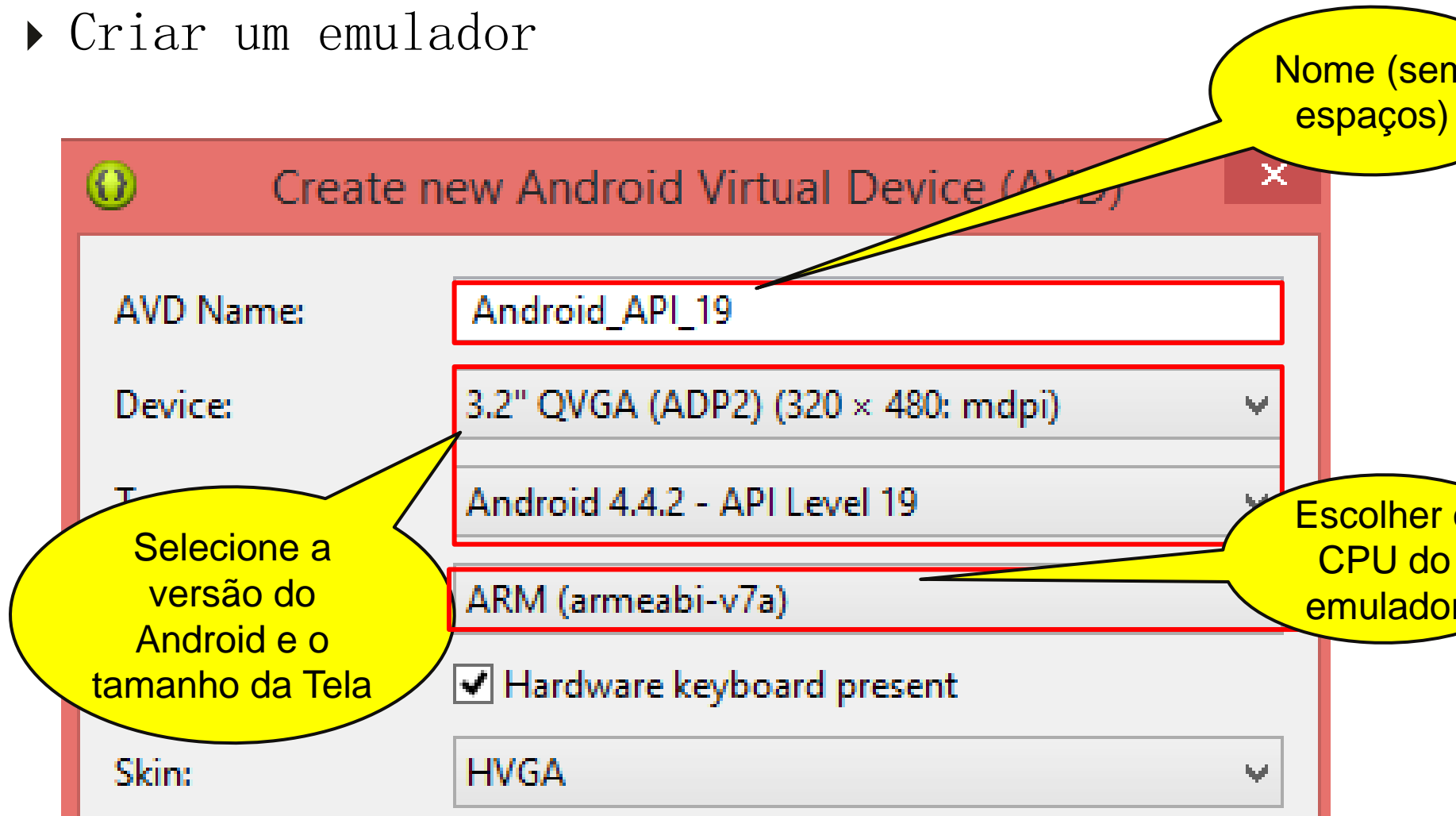


# Emulador



# Emulador

## ► Criar um emulador



# Emulador

Front Camera:

Back Camera:

Memory Options: RAM:  VM Heap:

Internal Storage:

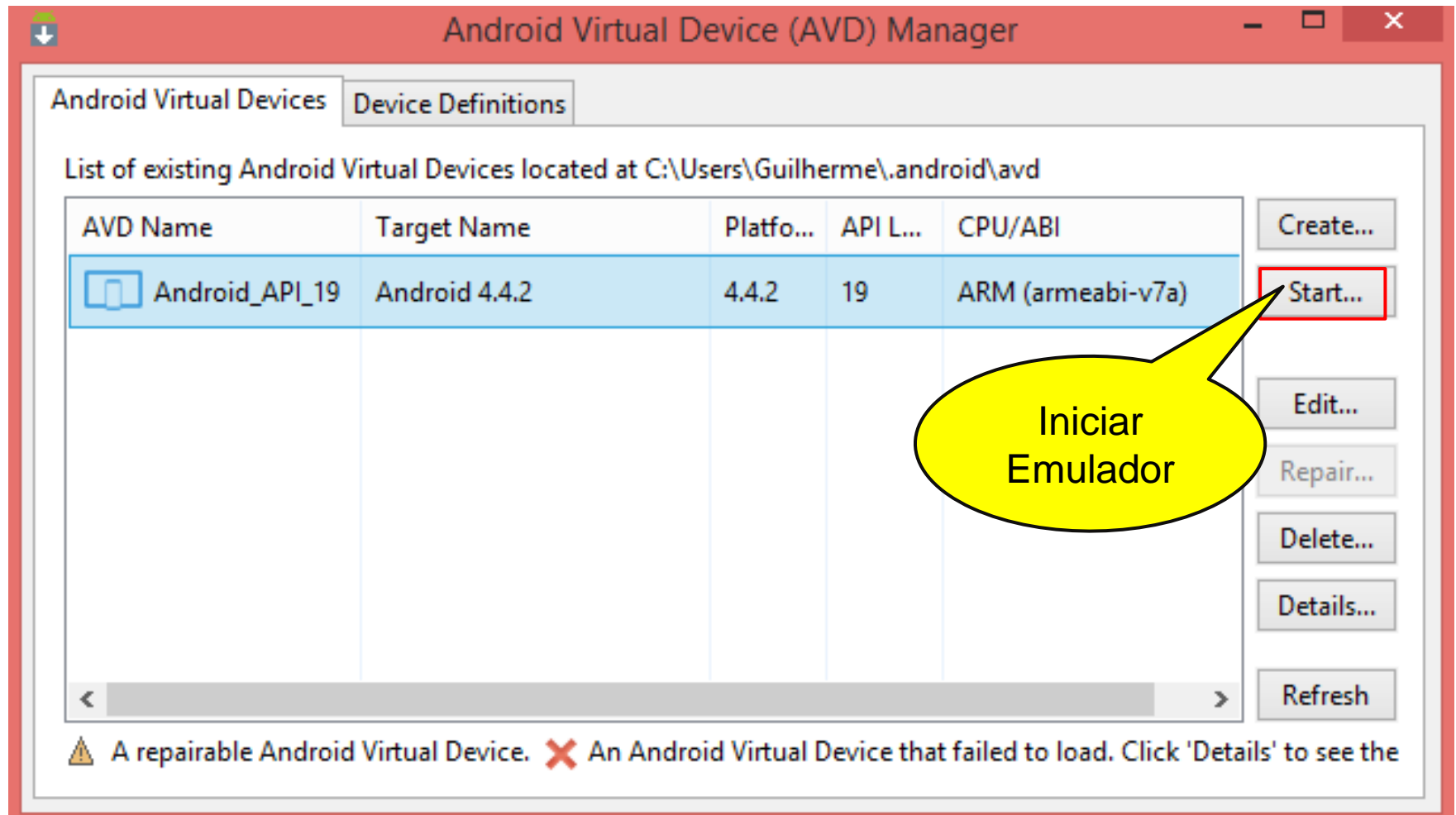
SD Card:

☒ Size:

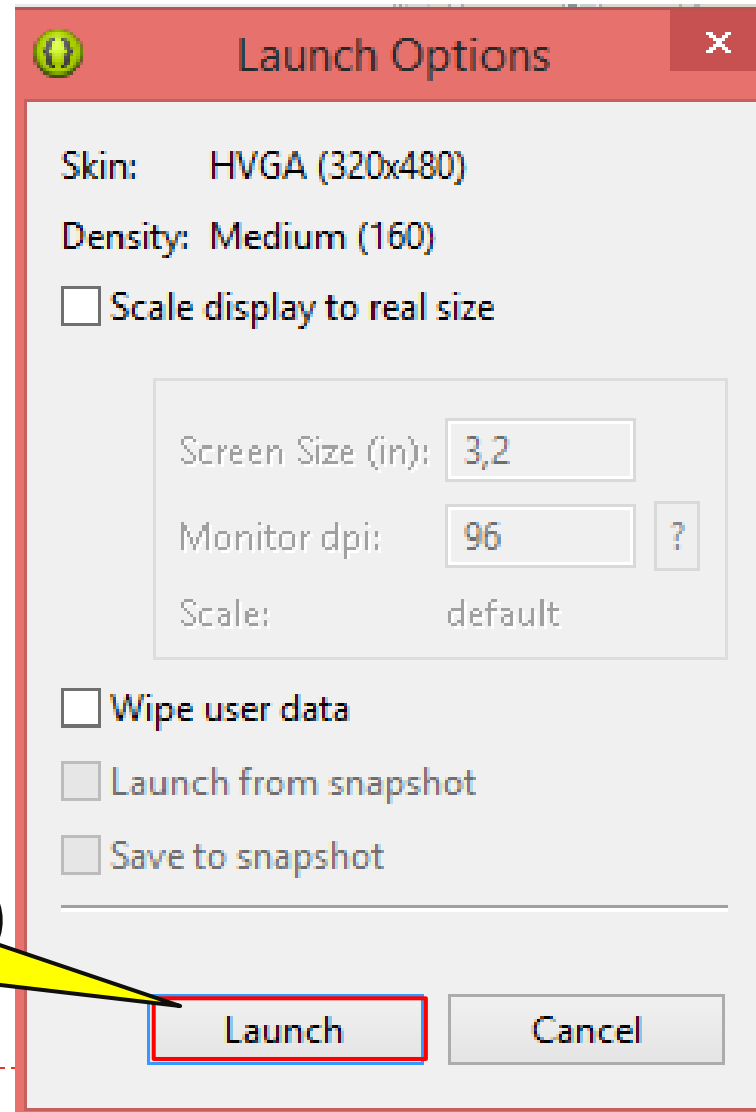
☐ File:

Este recursos  
servem para  
salvar  
informações no  
emulador.

# Emulador



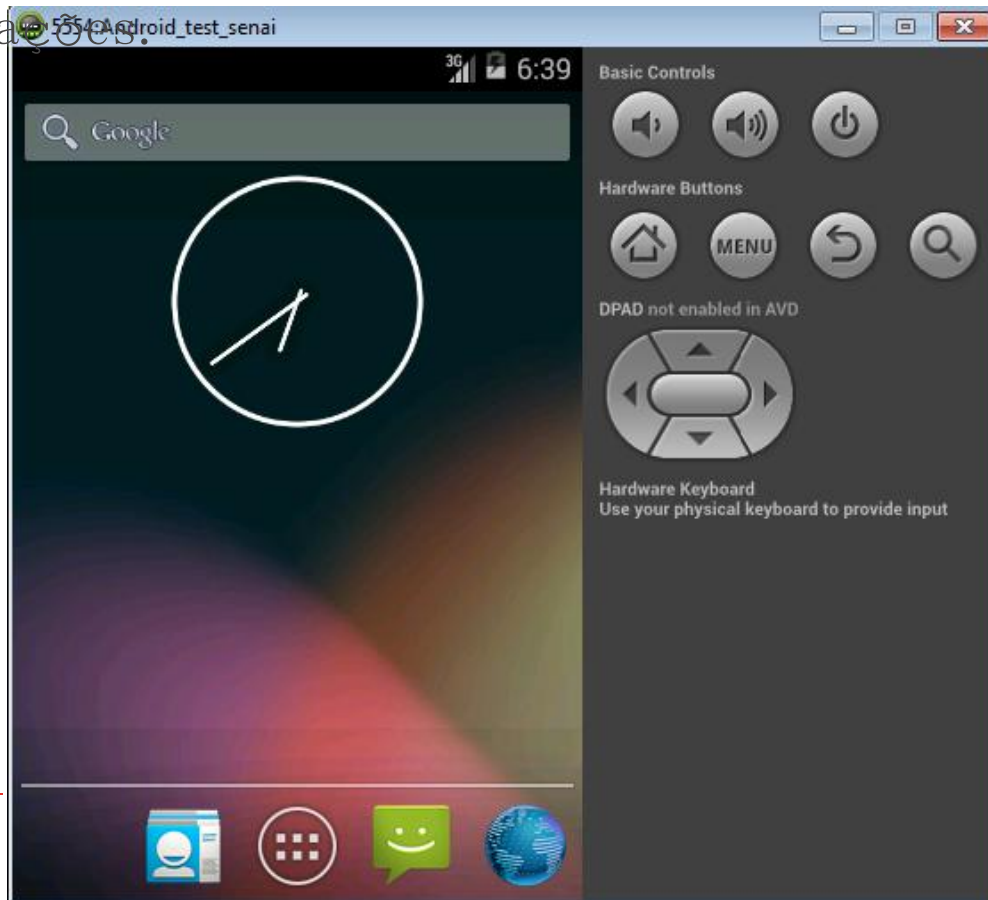
# Emulador



Iniciar  
Emulador

# Emulador

- ▶ Android Virtual Devices (AVD)
  - ▶ Usado para testar as aplicações
  - ▶ Simula um dispositivo real
  - ▶ É possível criar várias configurações de AVD para testar as aplicações.



# Criando Aplicações

# Resumo

---

- ▶ Criar uma aplicação
- ▶ Estrutura dos Arquivos
- ▶ Layouts
  - ▶ Elementos/Componentes XML
- ▶ Classe R
  - ▶ Integração XML com Activity
- ▶ Classe Activity e Classe View
  - ▶ Manipulando componentes da classe View pela Activity
  - ▶ Eventos
- ▶ Classe Intent
  - ▶ Chamadas com e sem retorno



# Exemplos

- ▶ Exemplos serão desenvolvidos durante a explicação dos componentes:

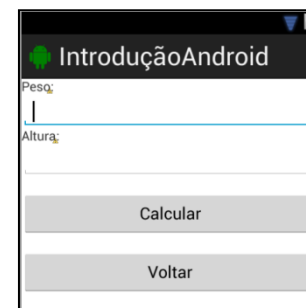
- ▶ Menu



- ▶ Abrir URL de navegador



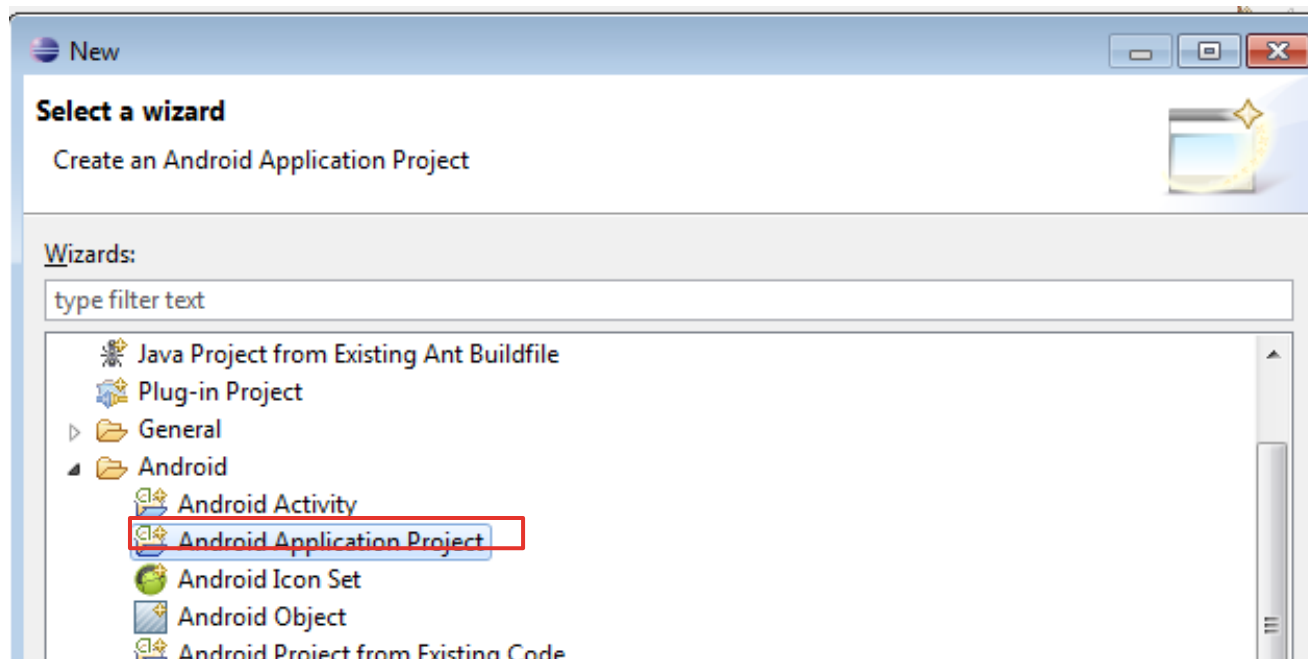
- ▶ Cálculo IMC



# Criando uma aplicação

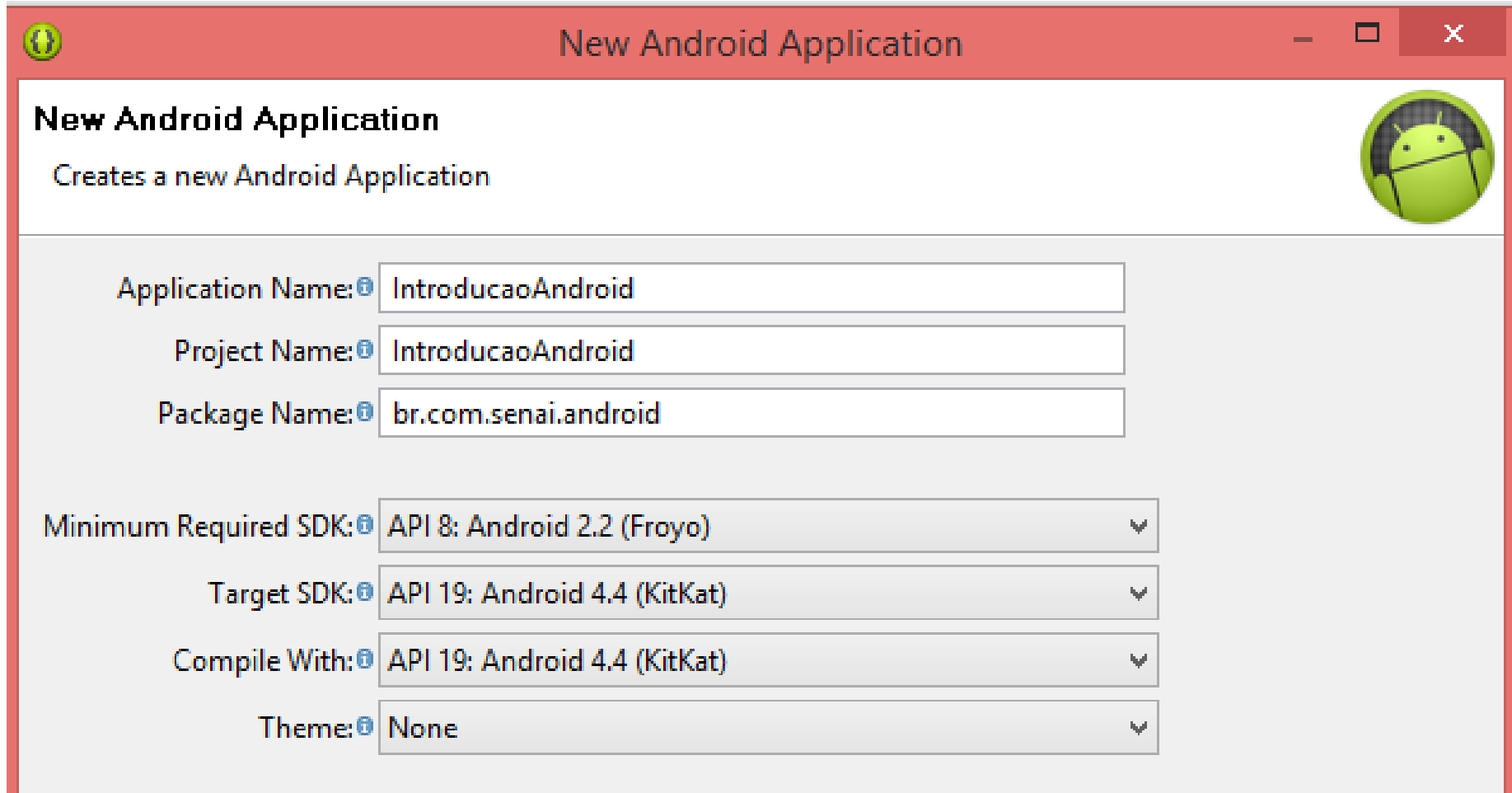
---

- ▶ Abrir a IDE eclipse
- ▶ Menu File → New → Other ou Menu File → New
- ▶ Selecionar Android Application Project



# Criando uma aplicação

- Preencha as informações e clique em **Next** >.



**New Android Application**

Creates a new Android Application

Application Name:

Project Name:

Package Name:

Minimum Required SDK:

Target SDK:

Compile With:

Theme:

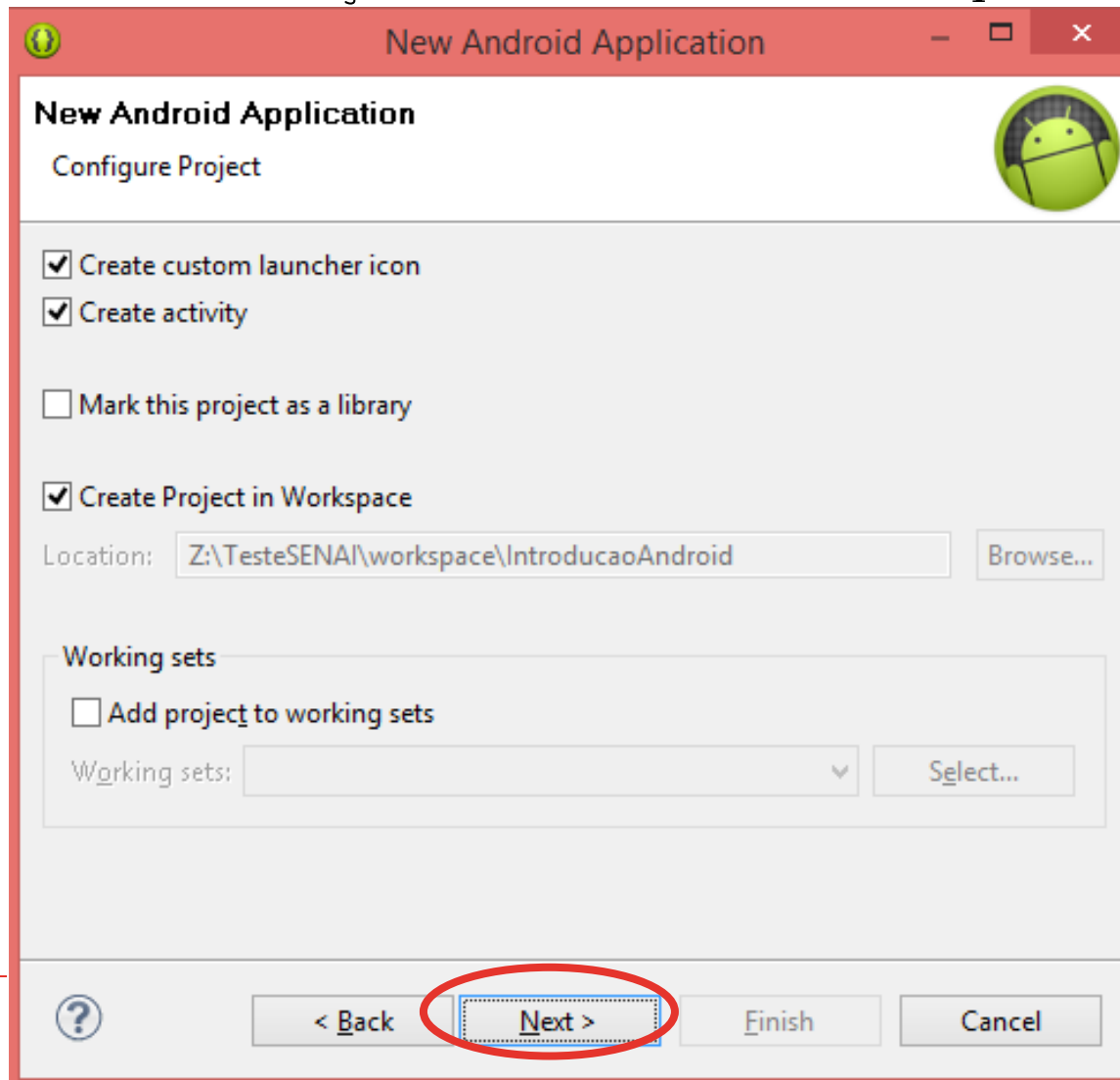
# Informações da tela de criação

---

- ▶ Application Name: nome da sua aplicação
- ▶ Project Name: nome do seu projeto
- ▶ Package Name: Nome do seu pacote
- ▶ Minimum Required SDK: Versão mínima requerida pela Aplicação
- ▶ Target SDK: Versão alvo da aplicação

# Criando uma Aplicação

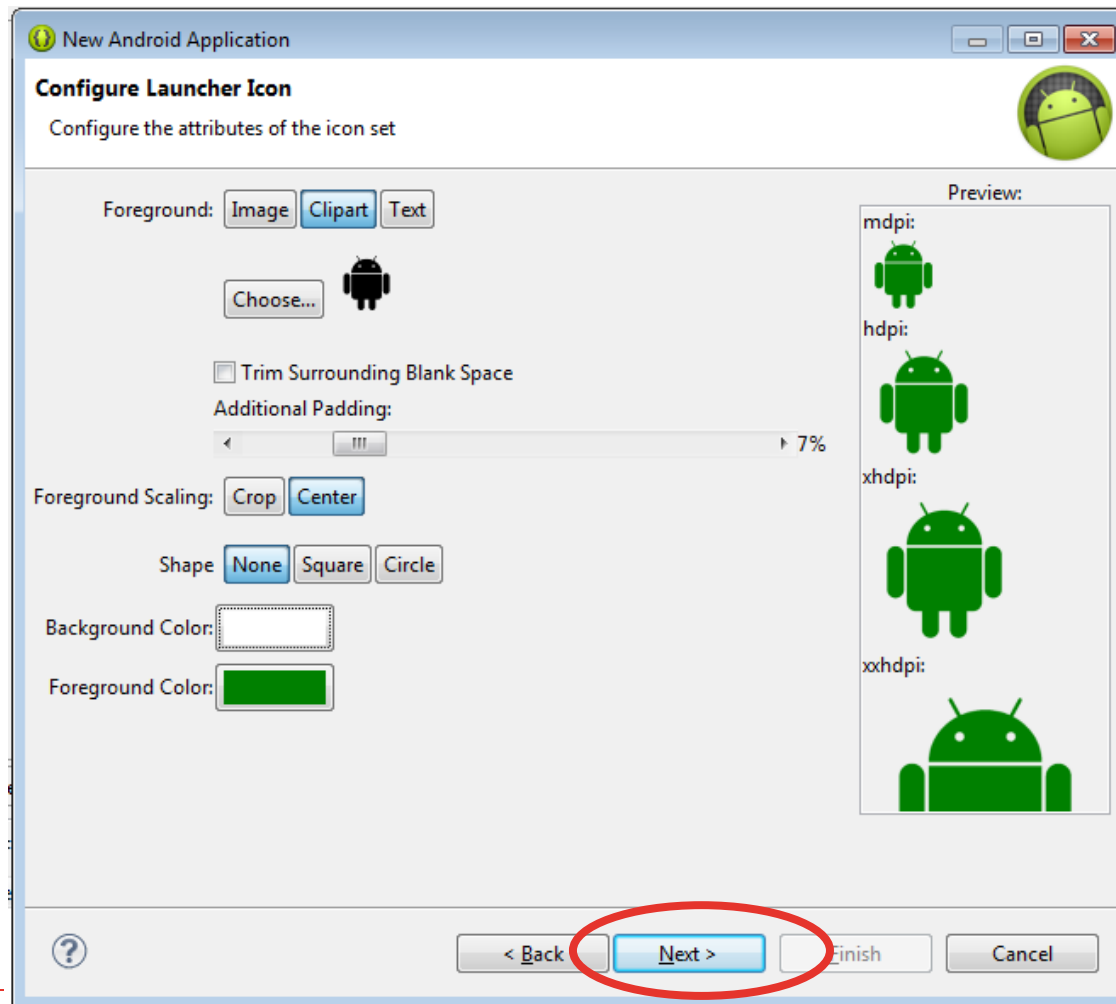
- ▶ Deixe as informações Default e clique em **Next**  
>



# Criando uma Aplicação

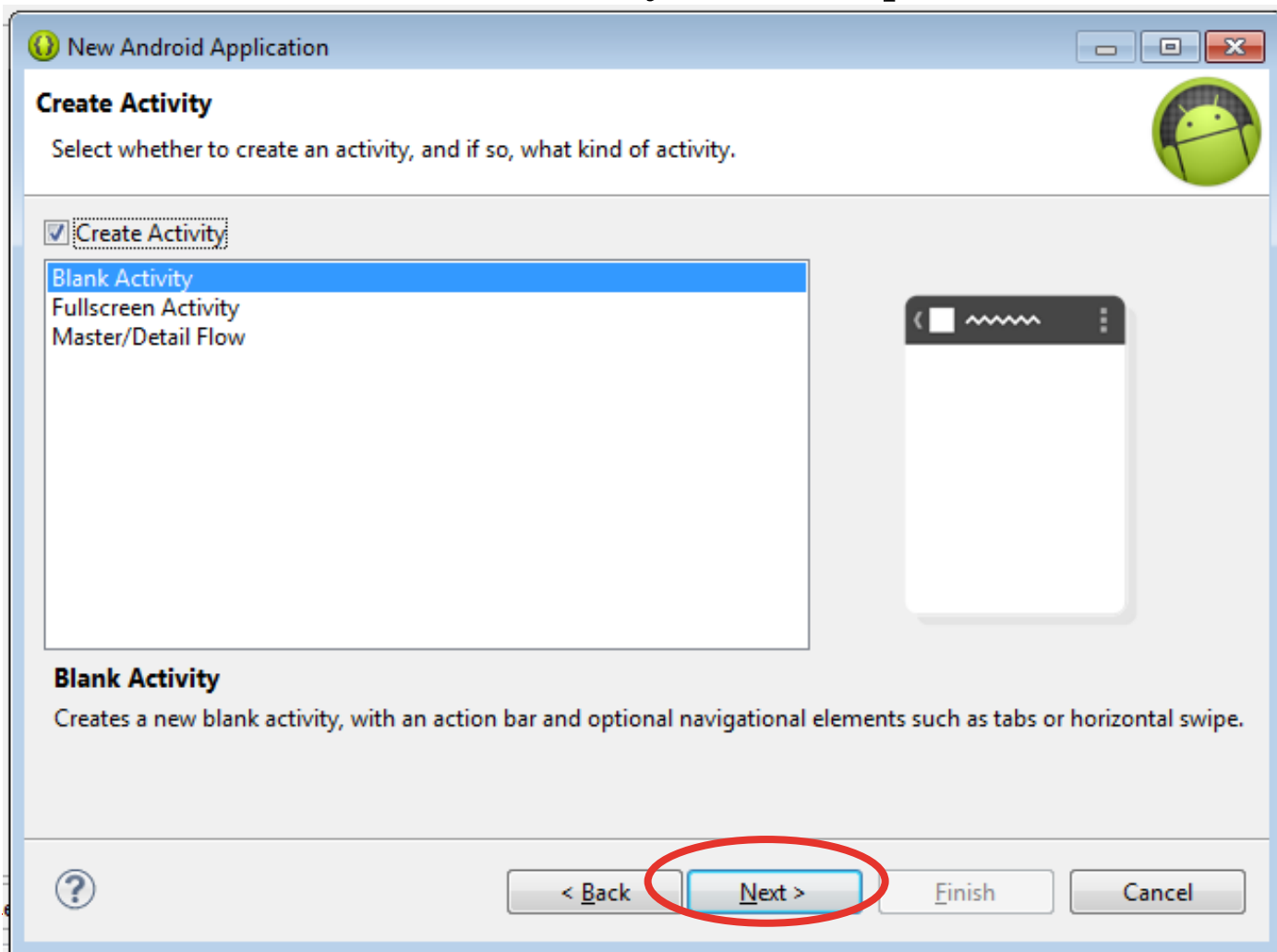
- Customize o ícone como quiser e clique em **Next**

>



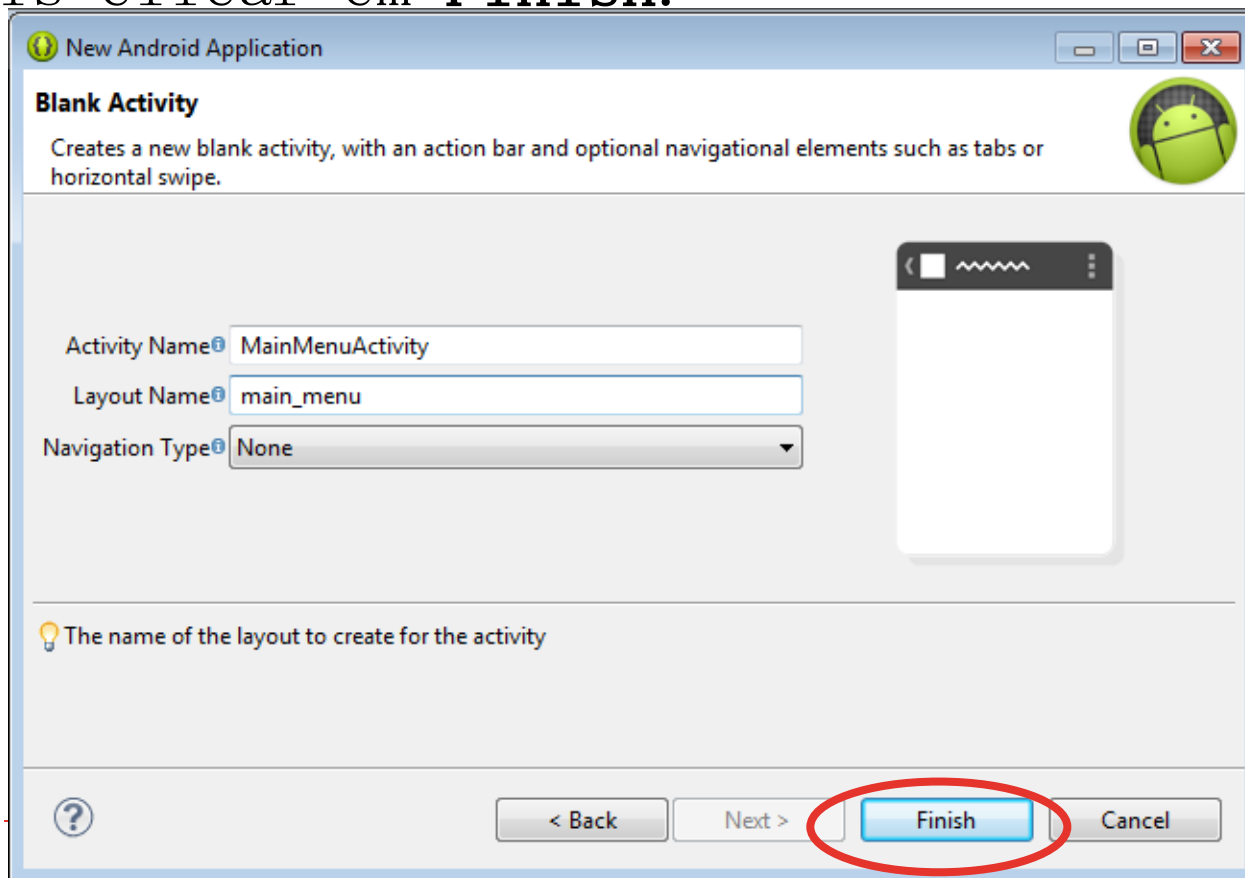
# Criando uma Aplicação

- Selecione Blank Activity e clique em Next >



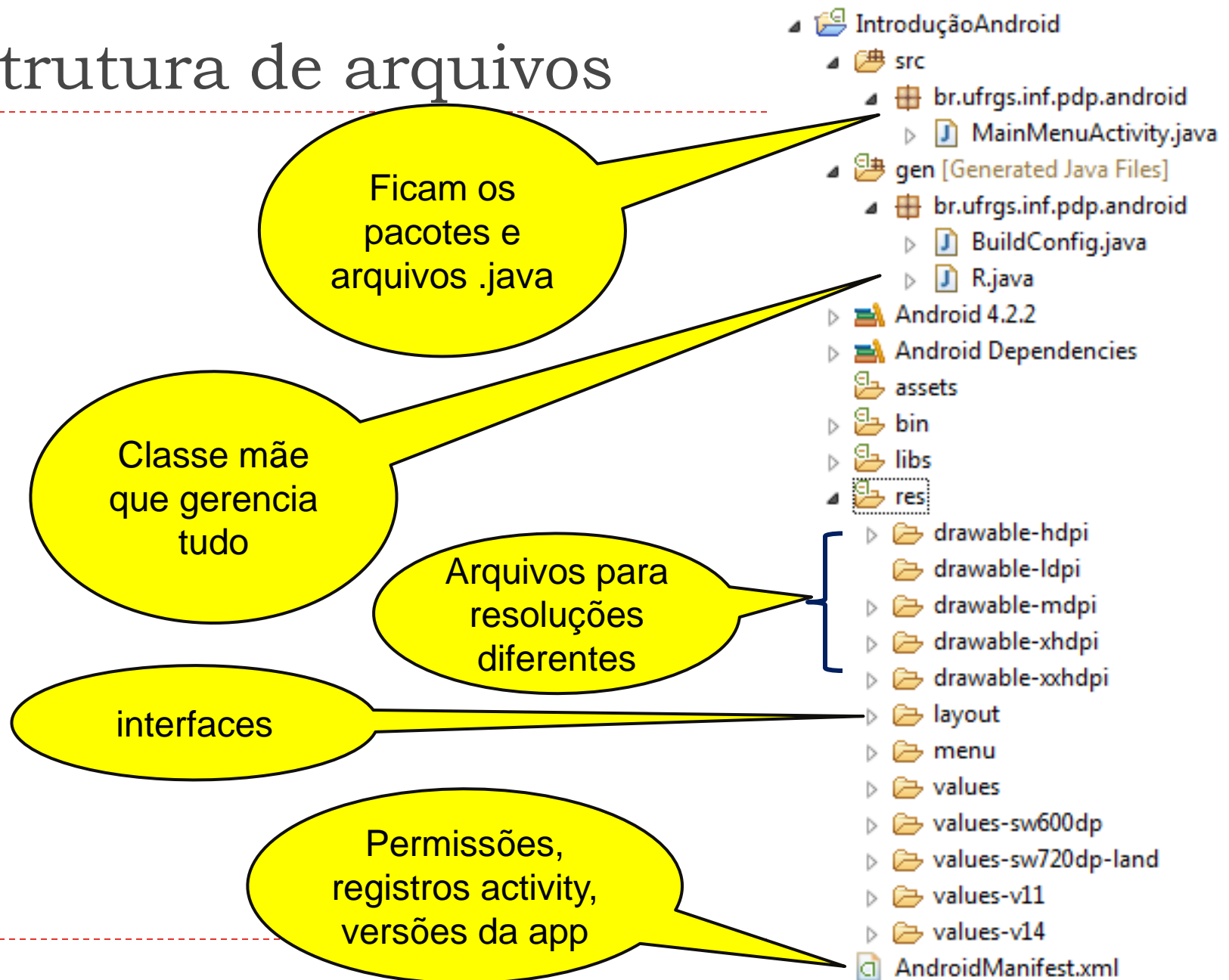
# Criando uma aplicação

- ▶ Activity Name: MainMenuActivity  
Layout Name: main\_menu, Navigation Type: None
- ▶ Depois clicar em **Finish**.

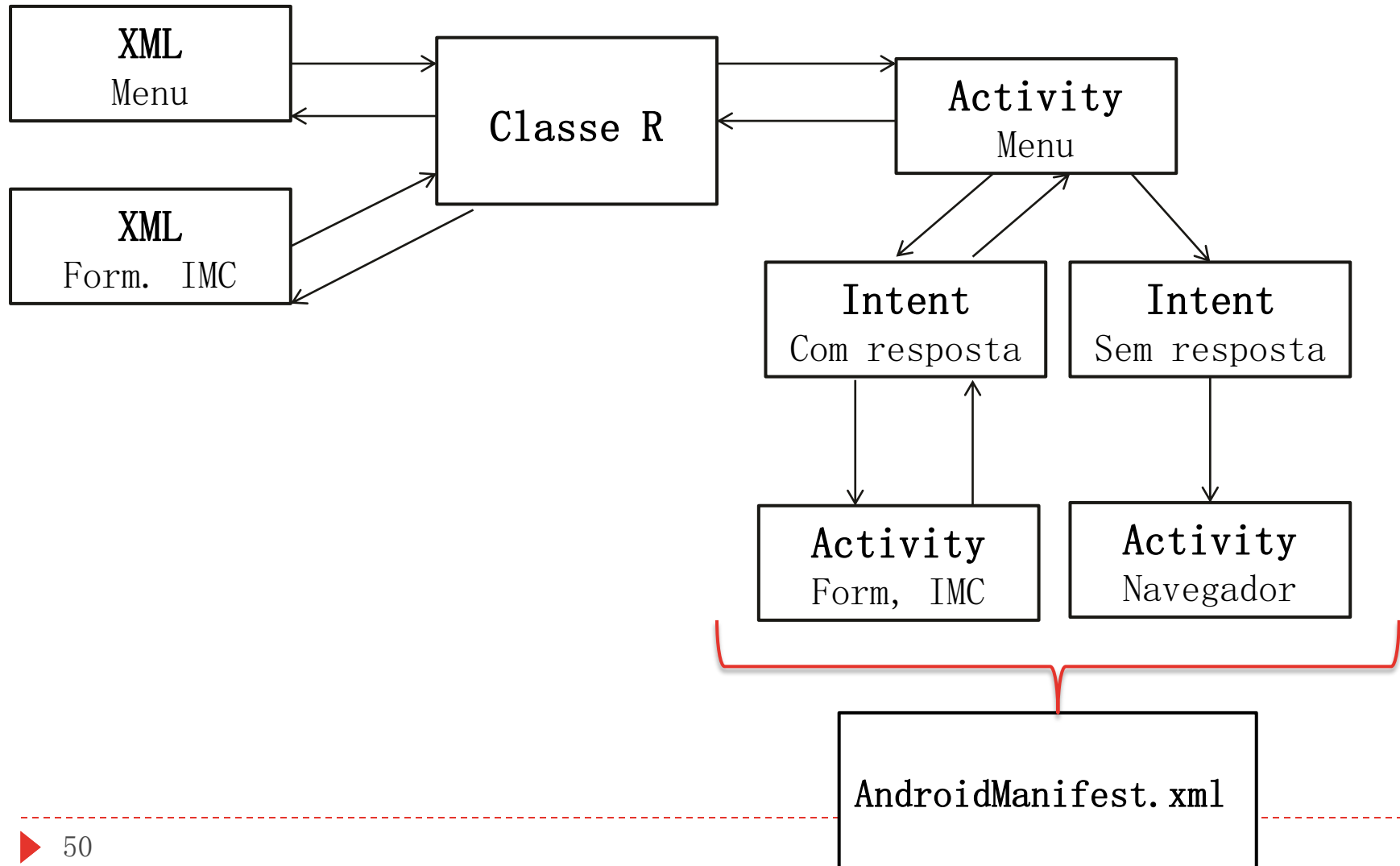




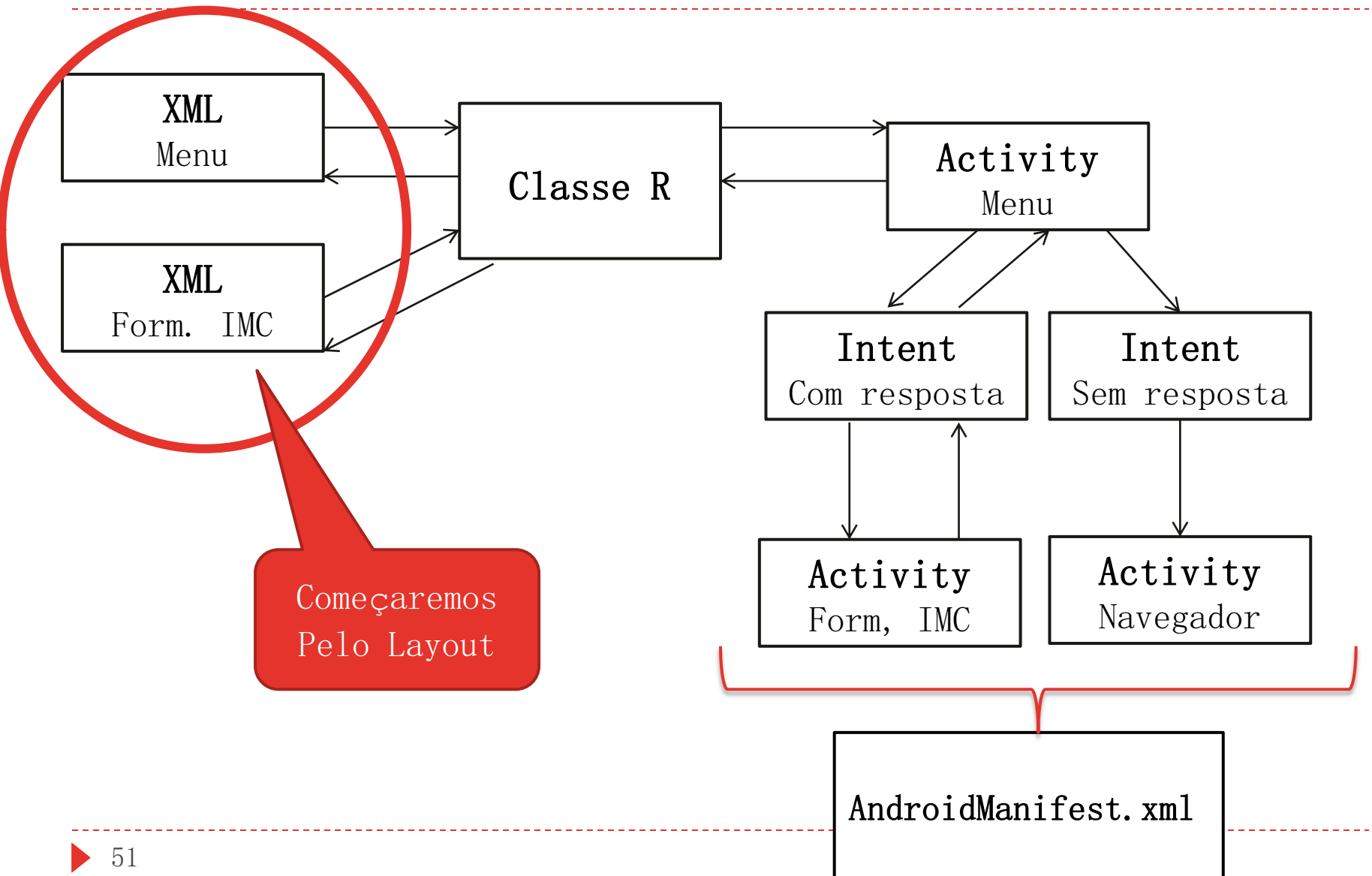
# Estrutura de arquivos



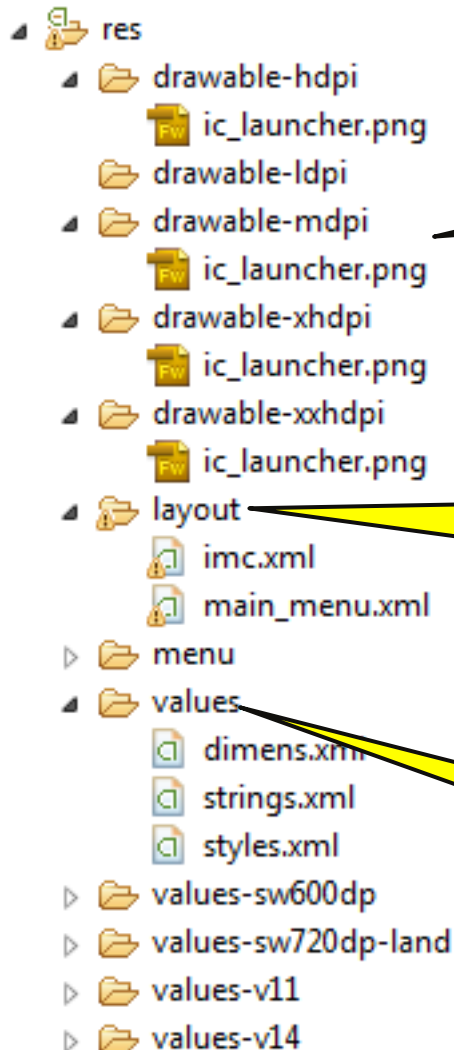
# Estrutura Conceitual



# Estrutura Conceitual



# Estrutura de arquivos Layout



Arquivos diferentes  
para resoluções  
diferentes  
(high, middle, lower)

XML com definições dos  
controles de tela  
Camada view  
Pode ter outra pasta  
layout-land com XML para  
quando gira o dispositivo

Variáveis  
Internacionalização  
(values-es, values-em)

# Layouts

---

- ▶ Os layouts XML são arquivos responsáveis por manter o Layout da nossa tela. É nele que definimos os componentes gráficos que a Activity irá utilizar além de especificar as posições em que eles serão disponibilizados.
- ▶ Quando nos referimos a componentes, estamos falando de elementos gráficos que serão exibidos para o usuário, como botões, caixas de texto, rótulos (label), etc.
- ▶ Tratam-se de objetos da classe `android.view.View`, ou filhas dessa classe

# Layouts

---

- ▶ `LinearLayout`
  - ▶ Define os controles na forma vertical e/ou horizontal
- ▶ `AbsoluteLayout`
  - ▶ Colocar os controles com coordenadas X e Y
- ▶ `Table Layout`
  - ▶ Similar ao `LinearLayout`, mas organiza os dados em forma de tabela
- ▶ `Relative Layout`
  - ▶ Os componentes são ajustados através de relacionamentos entre si ou ao seu pai
- ▶ `FrameLayout`
  - ▶ Arranja seus filhos de acordo com uma pilha de componentes que são adicionados, sendo que o topo da pilha contém o objeto que foi adicionado por último
- ▶ OBS. : Layouts podem ser Horizontais ou Verticais

# Controles

---

## ▶ Atributo Layouts

- ▶ Orientation = define se o layout é *"horizontal"* ou *"vertical"*

## ▶ Atributos Widgets

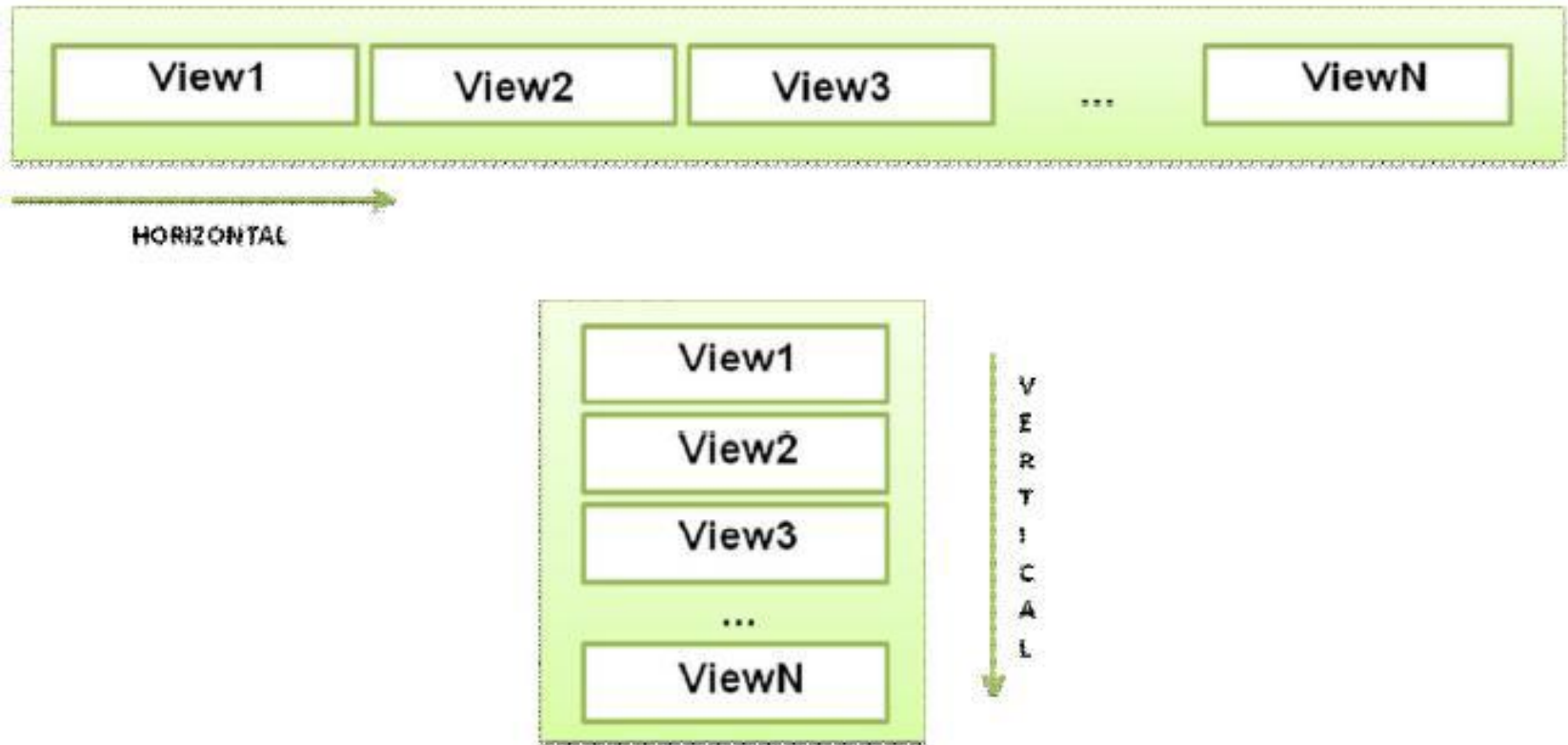
- ▶ Id = define o nome do controle
- ▶ Layout\_width = define a largura do controle
- ▶ Layout\_height = define a altura do controle
- ▶ Text = define um texto para o controle

## ▶ Valores para Layout\_width e Layout\_height

- ▶ Match\_parent = preenche o layout para toda a tela
- ▶ Wrap\_content = ocupar o tamanho necessário na tela

# LinearLayout

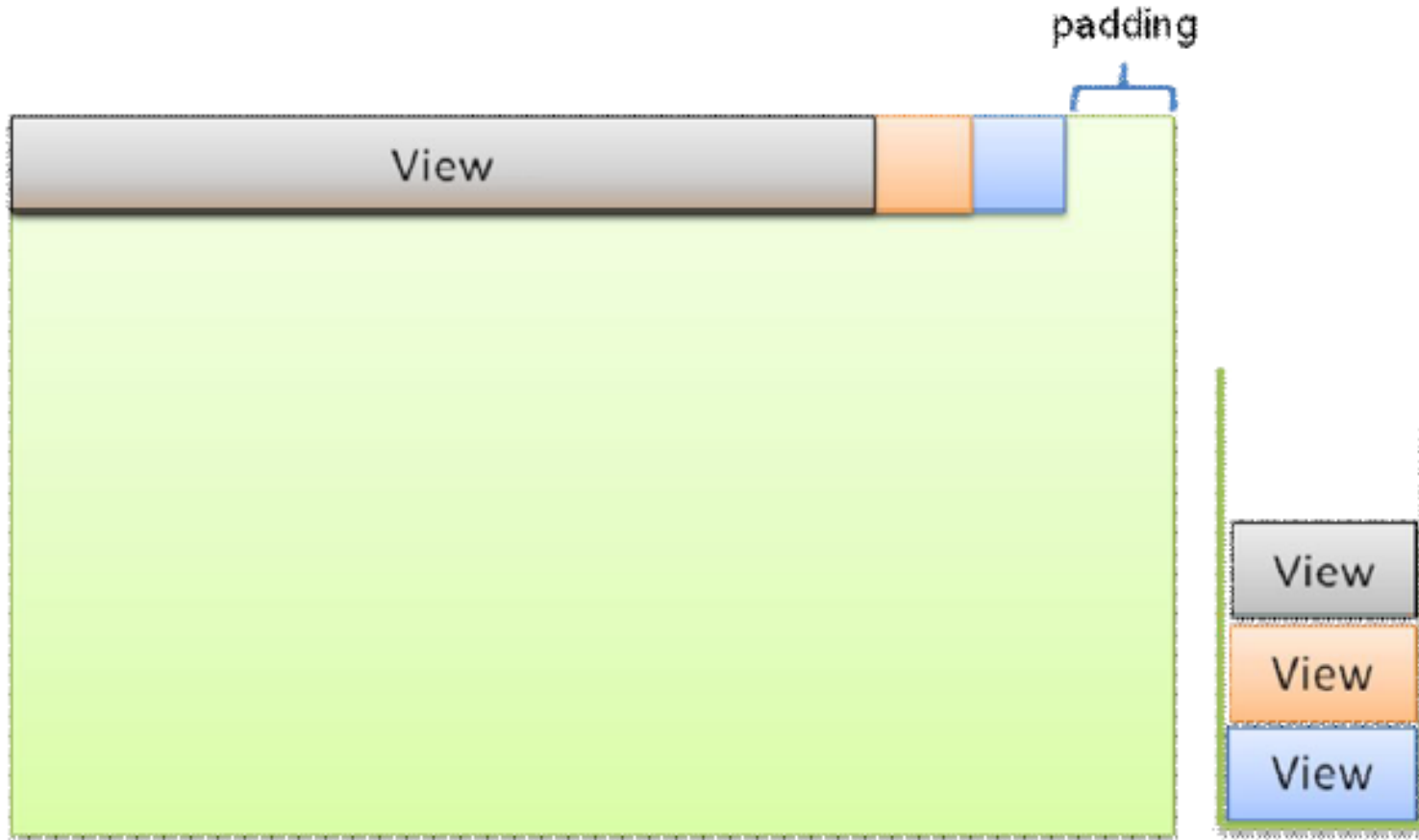
---



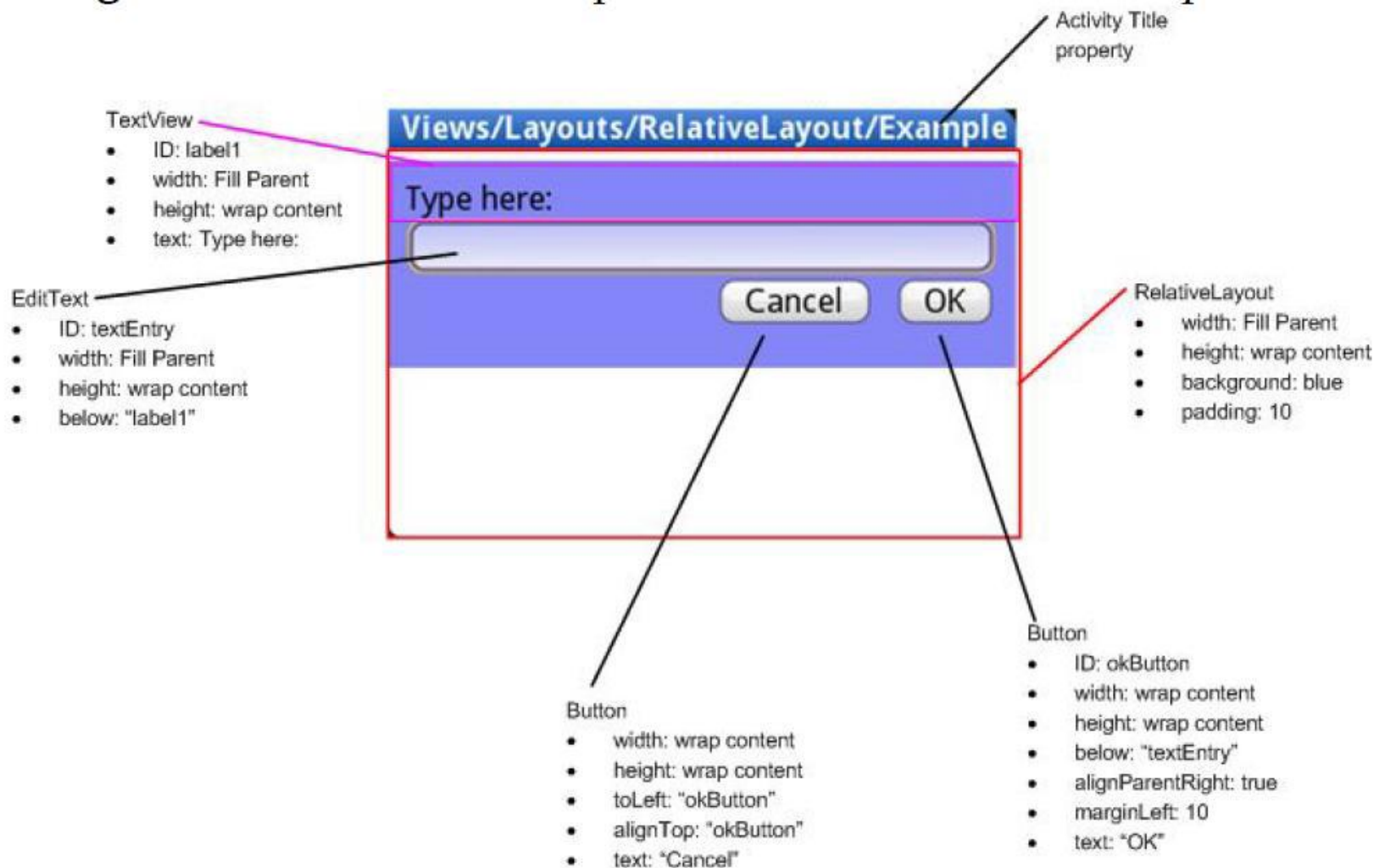


# FrameLayout

---

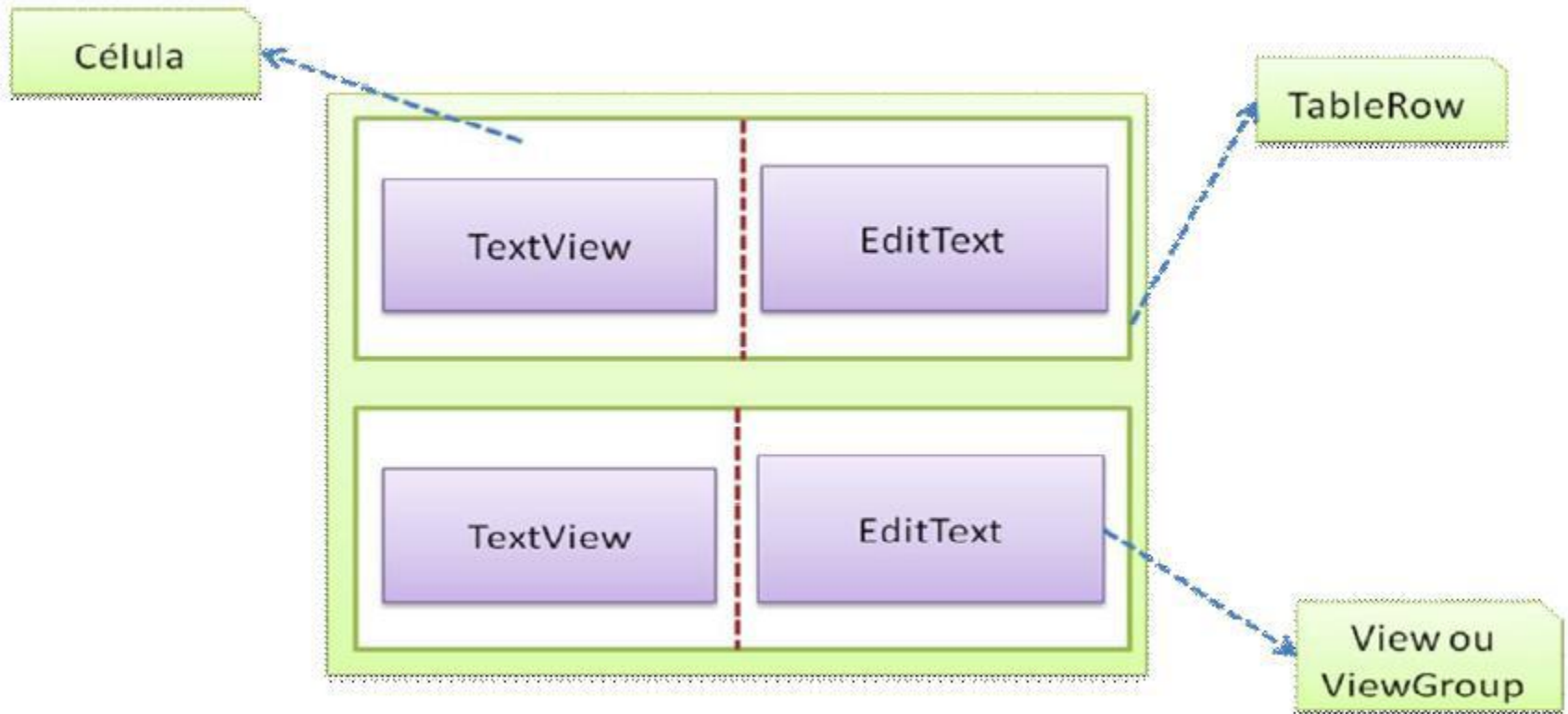


# RelativeLayout



# TableLayout

---



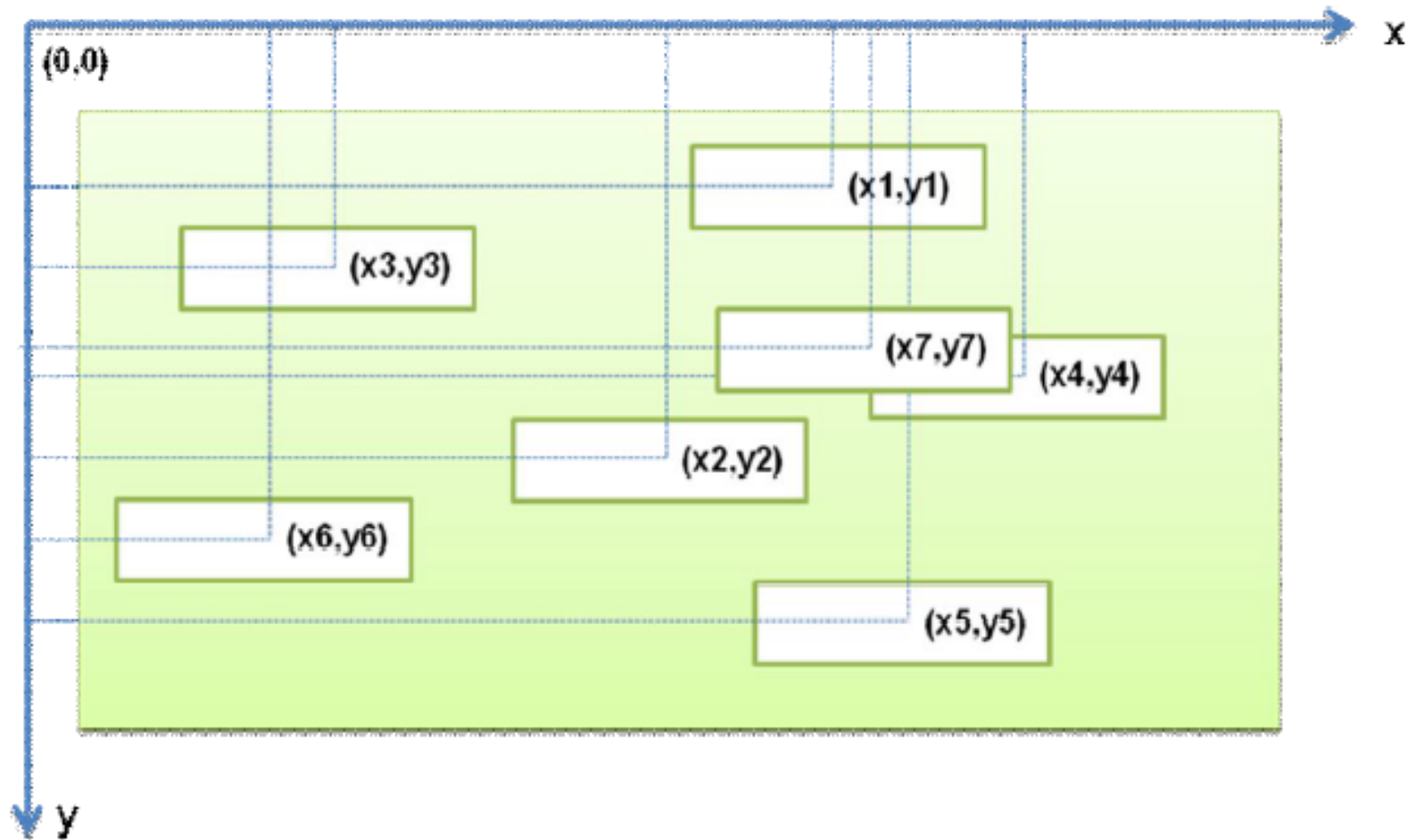
# TableLayout

---

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="wrap_content"
    android:layout_height="fill_parent">
    <TableRow>
        <TextView android:text="Nome:" />
        <EditText android:text="Ramon Rabello" />
    </TableRow>
    <TableRow>
        <TextView android:text="Data Nasc.:" />
        <EditText android:text="21/03/1986" />
    </TableRow>
    <TableRow>
        <Button android:text="Cadastrar" />
    </TableRow>
</TableLayout>
```



# AbsoluteLayout



# AbsoluteLayout

---

```
<?xml version="1.0" encoding="utf-8"?>
<AbsoluteLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="wrap_content"
    android:layout_height="fill_parent">
    <EditText android:layout_width="wrap_content"
        android:layout_height="wrap_content" android:text="Text01"
        android:layout_x="45px"
        android:layout_y="87px" />

    <EditText android:layout_width="wrap_content"
        android:layout_height="wrap_content" android:text="Text02"
        android:layout_x="90px"
        android:layout_y="12px" />

    <EditText android:layout_width="wrap_content"
        android:layout_height="wrap_content" android:text="Text03"
        android:layout_x="90px"
        android:layout_y="250px" />
</AbsoluteLayout>
```



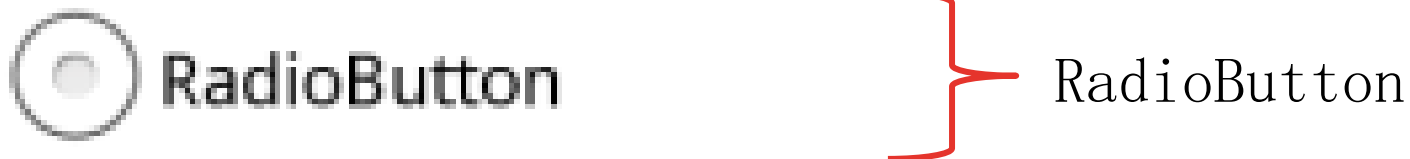
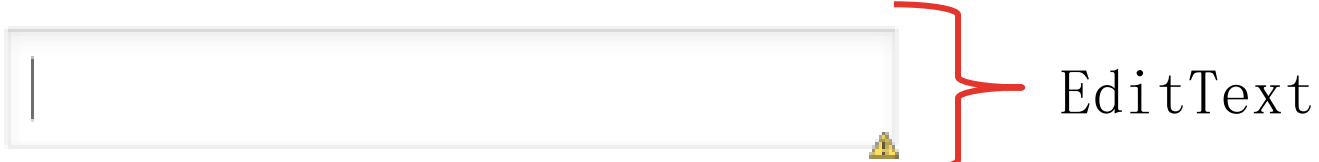
# Principais componentes

---

<b>TabHost</b>	Componente utilizado para separar em abas algumas telas. É muito adotado em telas de cadastro onde existem muitos campos para serem preenchidos. Ao invés de optar por fazer uma tela enorme cheio de elementos, é mais interessante criar várias activities e cada uma fica em uma aba. É empregado na aplicação de Contatos do próprio Android.
<b>ScrollView</b>	Trata-se de uma barra de rolagem. É utilizada em telas com grande quantidade de componentes. Consegue envolver apenas uma View, portanto geralmente está aninhado a um Layout como o LinearLayout, que por sua vez consegue adicionar outras Views, que também serão contempladas pelo ScrollView.
<b>ListView</b>	Componente que representa uma lista. Usada geralmente para exibir uma coleção de dados do mesmo tipo.
<b>GridView</b>	Assim como o componente Gallery, geralmente é utilizado para exibir imagens. No entanto, apresenta as figuras em uma espécie de tabela, onde o usuário tem contato com todas as imagens com um tamanho minúsculo.
<b>Button e ImageButton</b>	Trata-se de um botão para executar determinada ação. Esse botão pode receber um texto para que possa ser identificado. Caso o desenvolvedor prefira usar uma imagem ao invés de texto, o componente ImageButton pode ser empregado.
<b>Gallery</b>	Geralmente utilizado para exibir imagens. Apresenta as figuras em uma lista horizontal com barra de rolagem. O item selecionado fica em foco no centro da tela.
<b>EditText</b>	Elemento gráfico responsável por receber informações digitadas pelo usuário. Podemos fazer uma comparação com o objeto TextField da API Swing.
<b>CheckBox</b>	São caixas de seleção, utilizadas quando o usuário necessita selecionar múltiplas opções.
<b>RadioButton</b>	São caixas de seleção, utilizadas quando o usuário necessita selecionar apenas uma opção dentre varias.
<b>Spinner</b>	São semelhantes aos combo boxes. Trata-se de um componente que exibe alguma informação pré-selecionada, e ao ser clicado exibe uma lista com outras opções.
<b>TextView</b>	Componente responsável por exibir algum texto para o usuário. Semelhante ao objeto Label da API Swing.

# Principais componentes

---





# Principais componentes

---



# Exemplo de uso dos componentes

---

Hello world!

# Layout XML

---

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <TextView
        android:id="@+id/id_text_view"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/hello_world" />

    <EditText
        android:id="@+id/id_edit_text"
        android:text=""
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>

    <Button
        android:id="@+id/id_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="@string/executar" />

</LinearLayout>
```



**Layout Linear** com orientação vertical. Os componentes dessa tela serão organizados um em cima do outro.

**match\_parent:** Permite que o componente possa utilizar toda a área disponibilizada para ele. Pode ser encontrado em alguns códigos a constante `fill_parent` que produz o mesmo efeito do `match_parent`, porém `fill_parent` foi recentemente depreciada;

**wrap\_content:** Permite que o componente utilize o mínimo de espaço necessário para a sua criação.

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <TextView
        android:id="@+id/id_text_view"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/hello_world" />

    <EditText
        android:id="@+id/id_edit_text"
        android:text=""
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>

    <Button
        android:id="@+id/id_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="@string/executar" />

</LinearLayout>
```

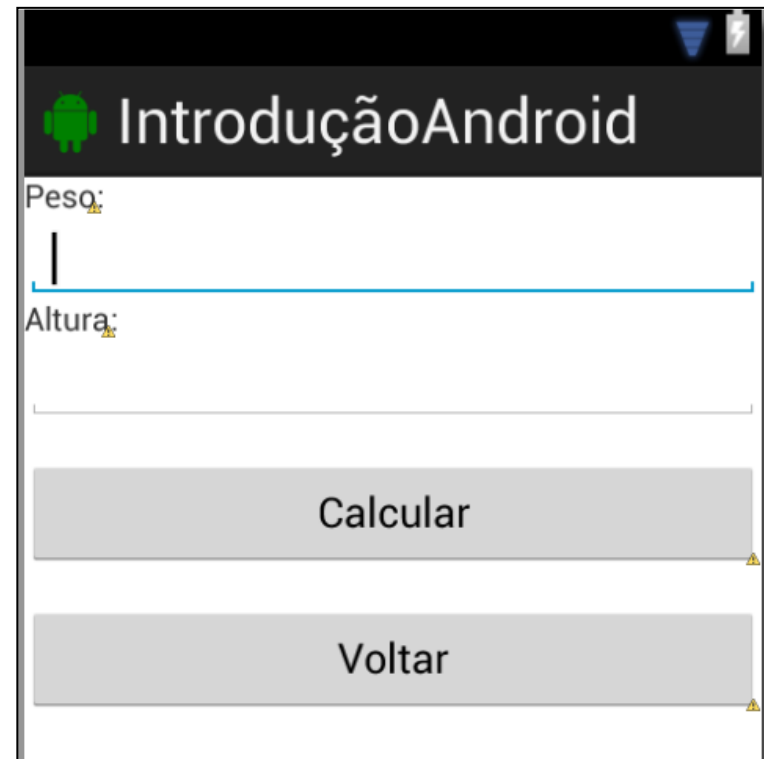
# Exemplos

Construir as View:

1) menu principal;



2) Calculo de IMC.



# Construir Tela Menu

Elemento Root: LinearLayout



Id: btnNavegador

Id: btnIMC

Id: btnSair

Elemento Horizontal: LinearLayout

Sem ID

Id: txtUltimoResultadoDeIMC

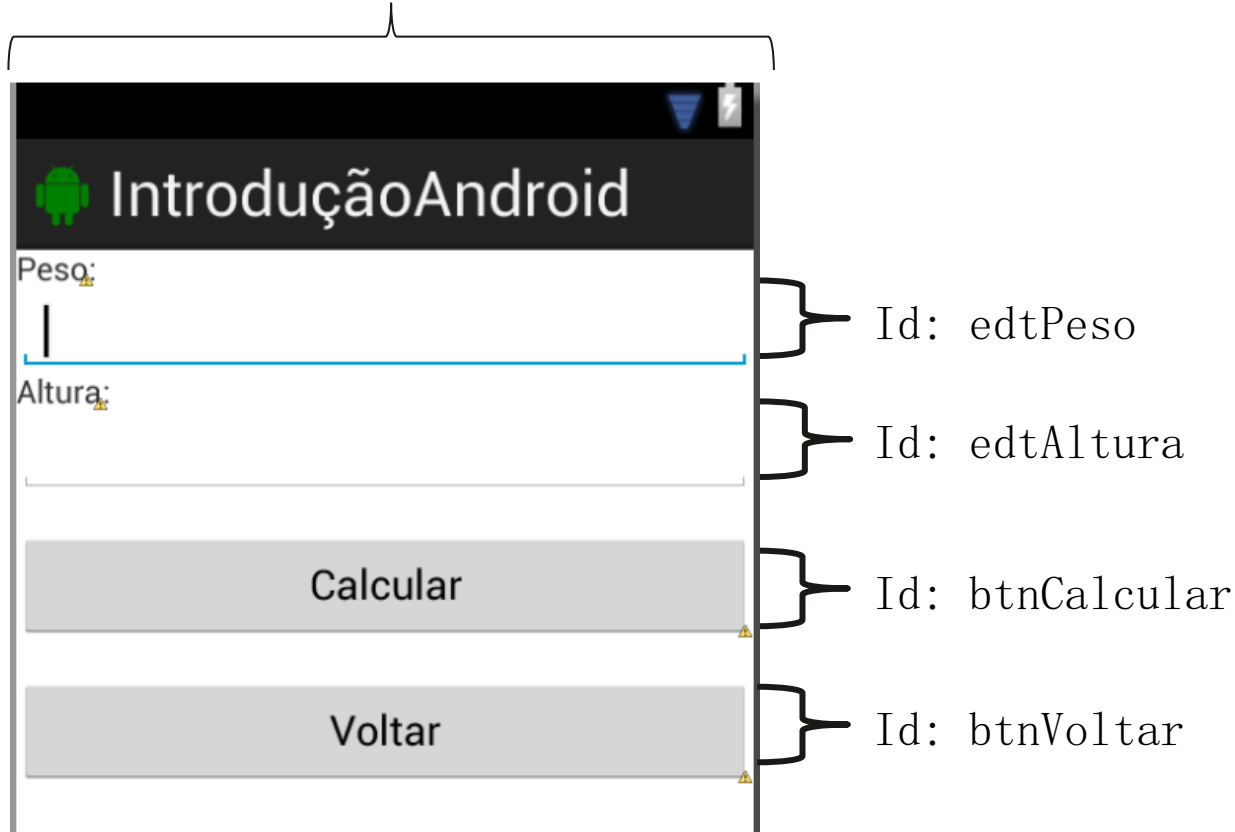
# XML do menu

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <Button
        android:id="@+id/btnNavegador"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Abrir Navegador" />
    <Button
        android:id="@+id/btnIMC"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="IMC" />
    <Button
        android:id="@+id/btnSair"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Sair" />
    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Último Resultado de IMC: " />
        <TextView
            android:id="@+id/txtUltimoResultadoDeIMC"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="0" />
    </LinearLayout>
</LinearLayout>
```

# Construir: Tela do Formulário de IMC

---

Elemento Root: LinearLayout



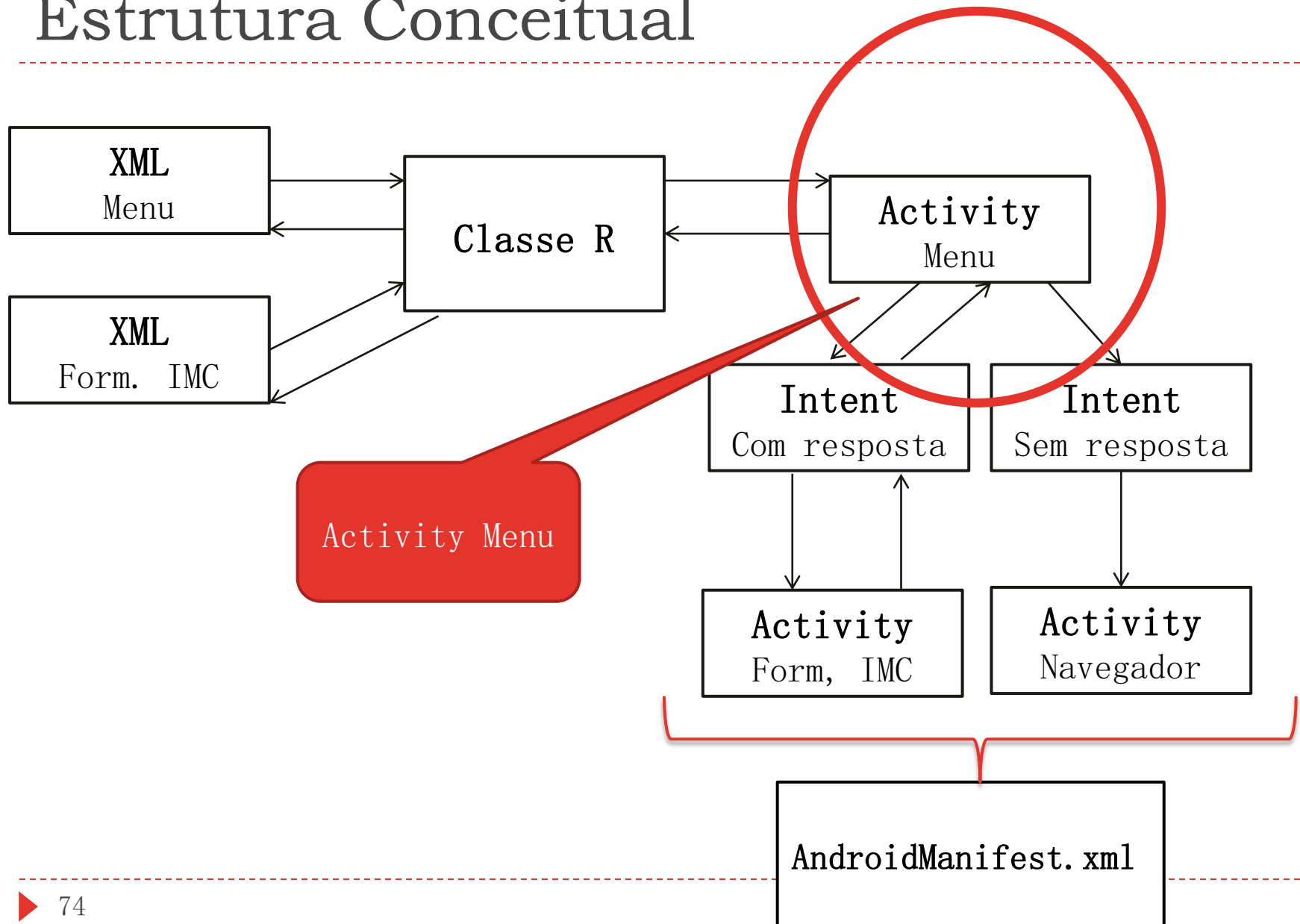


# XML do Formulário de IMC

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Peso:" />
    <EditText
        android:id="@+id/edtPeso"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="numberDecimal" >
        <requestFocus />
    </EditText>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Altura:" />
    <EditText
        android:id="@+id/edtAltura"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="numberDecimal" />
    <Button
        android:id="@+id/btnCalcular"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Calcular" android:layout_marginTop="15dp"/>
    <Button
        android:id="@+id/btnVoltar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Voltar" android:layout_marginTop="15dp"/>
</LinearLayout>
```



# Estrutura Conceitual



# Classe Activity

---

- ▶ Toda janela Android é uma Activity
- ▶ Um aplicativo pode ter várias Activity (herdam da classe `android.app.Activity`)
- ▶ Deve implementar o método

```
public void onCreate(Bundle  
    savedInstanceState) {...}
```

que inicializa os elementos da tela.

- ▶ Activities utilizam a classe `R` para acessar os elementos do layout
- ▶ A integração de um Layout XML com uma Activity é feita através do método `onCreate()` da Activity.



# A integração - Layout XML com Activity

---

- ▶ Deve-se referenciar o arquivo de XML em uma Activity.
- ▶ Esse procedimento é feito através do método *setContentView(int id)* da Activity, que deve ser utilizado dentro do método *onCreate()* do seu ciclo de vida.
- ▶ O método *setContentView()* recebe como parâmetro um *int*, que funciona como um identificador para o Layout. Esses identificadores podem ser recuperados através da classe R.

# A integração - Layout XML com Activity

---

- ▶ A classe `R` separa os dados que podem ser recuperadas em camadas.
- ▶ Por exemplo, para recuperar algum layout:
  - ▶ `R.layout.layout_desejado`
- ▶ Para recuperarmos uma `String`:
  - ▶ `R.string.texto_desejado`
- ▶ para recuperarmos algum componente gráfico:
  - ▶ `R.id.componente_desejado`
- ▶ O método `findViewById()` retorna um objeto do tipo `View`.
  - ▶ Deve-se fazer um cast para a classe do componente desejado

# Método onCreate MainActivity

- Adicionar os Objetos para integrar e manipular os elementos/componentes do layout

```
package br.com.ulbra.android.helloworld;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.widget.Button;
import android.widget.EditText;

public class MainActivity extends Activity {

    private Button btnExecutar;
    private EditText edtNome;

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

        btnExecutar = (Button) findViewById(R.id.btnExecutar);
        edtNome = (EditText) findViewById(R.id.edtNome);
    }
}
```

Pacote da classe

Importação das bibliotecas

Atributos da Classe que serão utilizados

Layout da Activity

Vincular Visões do Layout com Objetos Java

# Método onCreate MainActivity

---

- Adicionar os Objetos para integrar e manipular os elementos/componentes do layout

```
public class MainMenuActivity extends Activity {  
  
    private Integer IMC_RC = 1;  
    private Button btnNavegador;  
    private Button btnIMC;  
    private Button btnSair;  
    private TextView txtUltimoResultadoDeIMC;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.main_menu);  
  
        btnNavegador = (Button) findViewById(R.id.btnNavegador);  
        btnIMC = (Button) findViewById(R.id.btnIMC);  
        btnSair = (Button) findViewById(R.id.btnSair);  
        txtUltimoResultadoDeIMC = (TextView) findViewById(R.id.txtUltimoResultadoDeIMC);  
  
    }  
}
```

# Método onCreate IMCActivity

---

- Criar Activity e adicionar os Objetos para integrar e manipular os componentes do layout

```
imc public class IMCActivity extends Activity{  
  
    private Double resultadoCalculo;  
    private Button btnCalcular;  
    private Button btnSair;  
    private EditText edtPeso;  
    private EditText edtAltura;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.imc);  
  
        this.resultadoCalculo = 0.0;  
        this.btnCalcular = (Button) findViewById(R.id.btnCalcular);  
        this.btnSair = (Button) findViewById(R.id.btnVoltar);  
        this.edtAltura = (EditText) findViewById(R.id.edtAltura);  
        this.edtPeso = (EditText) findViewById(R.id.edtPeso);  
  
    }  
}
```



# Classe R

---

- ▶ É um dos arquivos mais importantes de um projeto Android;
- ▶ Todos os elementos são referenciados pela classe R;
- ▶ NÃO DEVE SER ALTERADA;
- ▶ Todo elementos de layout, strings, imagens... Incluídos no projeto são mapeados nesta classe;
  - ▶ Desta forma é possível acessar os elementos por suas referências

# Activity

---

- ▶ O Android trata as activities como se estivessem em uma pilha, chamada de activity stack (pilha de atividades).
- ▶ A tela que está interagindo com o usuário é a que está no topo dessa pilha, podendo existir várias outras activities abaixo dela em estado parado ou pausado.
- ▶ Uma Activity que for chamada e ficar em contato com o usuário ocupará o topo da pilha e a Activity que estava interagindo anteriormente ficará logo abaixo da nova.

# Activity - Métodos

---

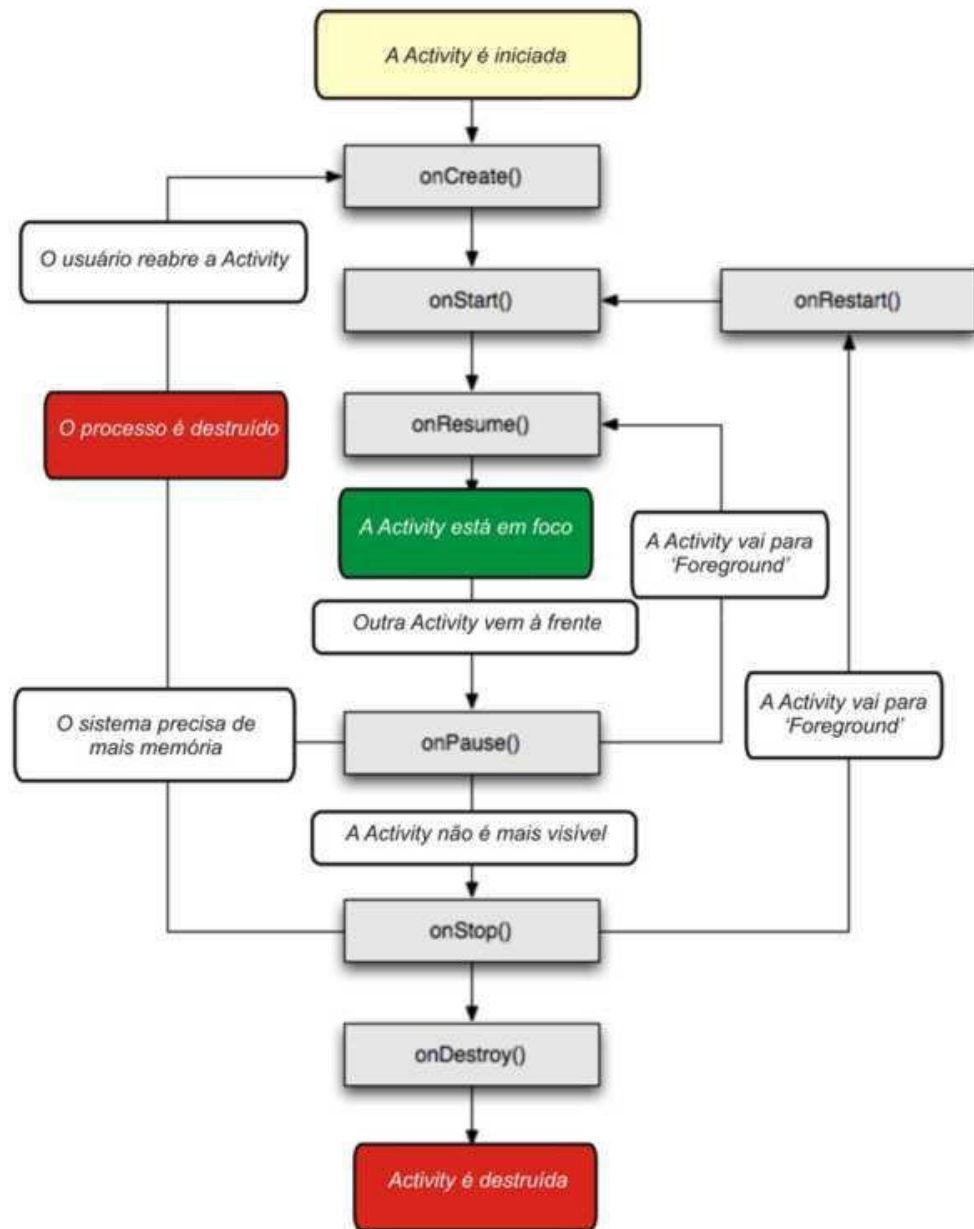
- ▶ O desenvolvedor consegue ter o controle das fases através de métodos.
- ▶ São eles que definem ações que serão executadas quando acontecer alguma fase da Activity, como por exemplo, persistir dados quando a tela for interrompida.

# Activity - fases

---

- ▶ Parada: a Activity não chegou a ser executada. A tela ainda não foi exibida;
- ▶ Ativa: a Activity está executando, ou seja, a tela foi criada e está sendo exibida para o usuário;
- ▶ Interrompida: a Activity foi interrompida por outra (recebe uma ligação)
- ▶ Destruída: a Activity foi finalizada. Isto indica que a tela foi destruída, que seu ciclo de vida chegou ao fim.

# Ciclo de Vida



# Funções dos Eventos Click Listener

---

- ▶ Eventos são utilizados para realizar ações. Esse eventos devem ser explicitamente programados para que funcionem.
- ▶ Funções de evento utilizada:
  - ▶ `onClick`: Quando o objeto (View) for clicado é acionado um evento. Para que isso funcione é necessário implementar um escutador do evento (`onClickListener`).

# Adicionar Evento de Click nos botões

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.main_menu);  
  
    btnNavegador = (Button) findViewById(R.id.btnNavegador);  
    btnIMC = (Button) findViewById(R.id.btnIMC);  
    btnSair = (Button) findViewById(R.id.btnSair);  
    txtUltimoResultadoDeIMC = (TextView) findViewById(R.id.txtUltimoResultadoDeIMC);  
  
    btnNavegador.setOnClickListener(new View.OnClickListener() {  
        @Override  
        public void onClick(View v) {  
            abrirNavegador();  
        }  
    });  
    btnIMC.setOnClickListener(new View.OnClickListener() {  
        @Override  
        public void onClick(View arg0) {  
            acessarIMC();  
        }  
    });  
    btnSair.setOnClickListener(new View.OnClickListener() {  
        @Override  
        public void onClick(View arg0) {  
            finish();  
        }  
    });  
}
```

Quando o botão  
Abrir Navegador for  
clicado o código dentro  
de onClick(View v) é  
executado

# Código IMCActivity

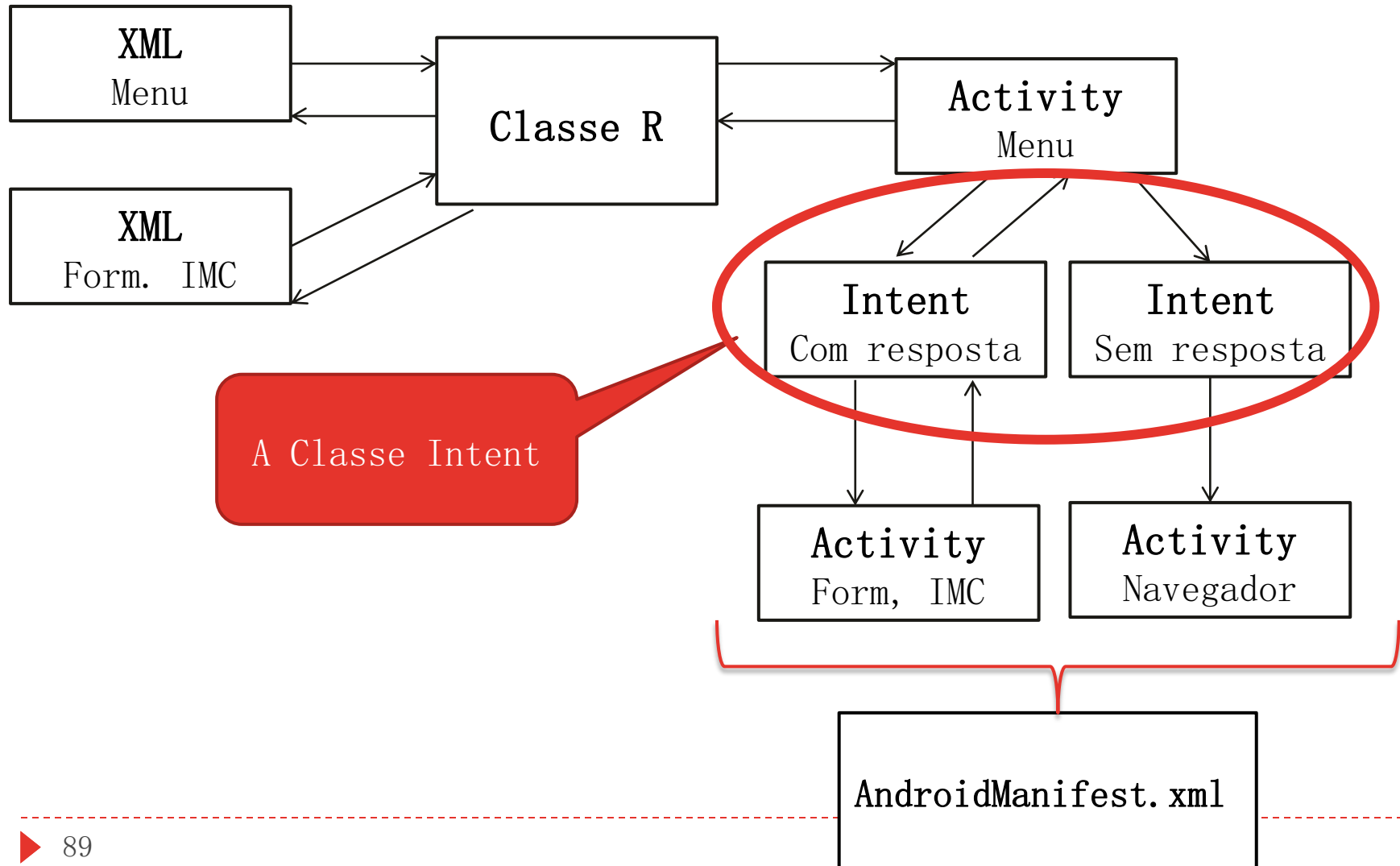
---

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.imc);

    this.resultadoCalculo = 0.0;
    this.btnCalcular = (Button) findViewById(R.id.btnCalcular);
    this.btnSair = (Button) findViewById(R.id.btnVoltar);
    this.edtAltura = (EditText) findViewById(R.id.edtAltura);
    this.edtPeso = (EditText) findViewById(R.id.edtPeso);
    btnCalcular.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Double altura = Double.parseDouble(edtAltura.getText().toString());
            Double peso = Double.parseDouble(edtPeso.getText().toString());
            resultadoCalculo = peso/(altura * altura);
            Toast.makeText(IMCActivity.this,
                "Seu imc é: "+String.valueOf(resultadoCalculo), Toast.LENGTH_LONG).show();
        }
    });
    btnSair.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Intent it = new Intent();
            it.putExtra("imc", resultadoCalculo);
            setResult(RESULT_OK,it);
            finish();
        }
    });
}
```



# Estrutura Conceitual



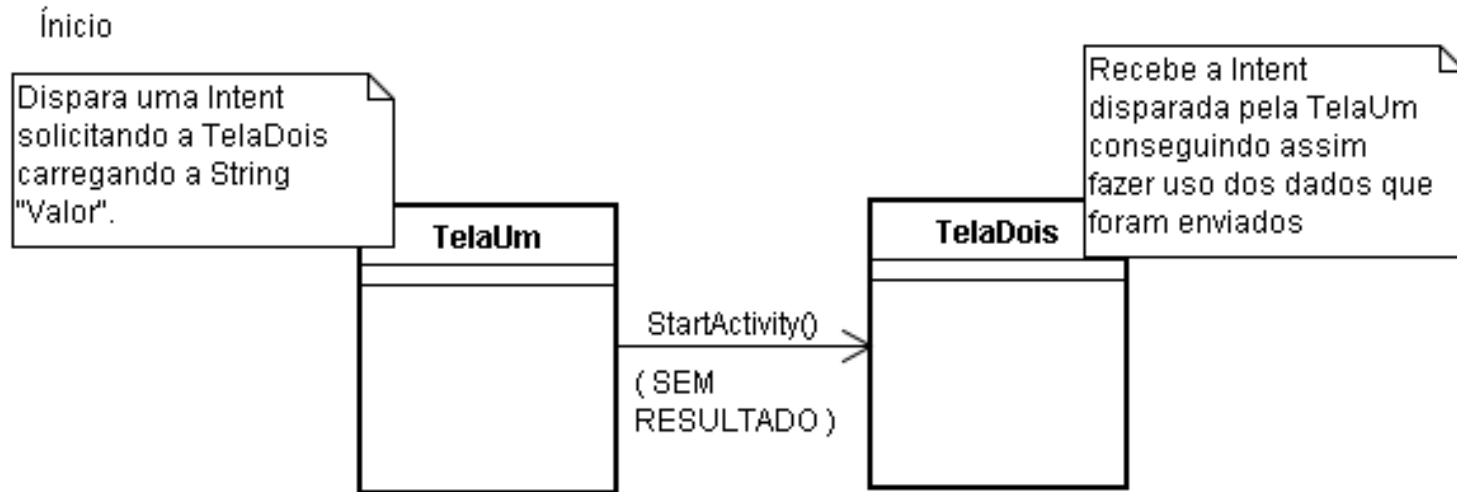
# Intent

---

- ▶ Indica que há a intenção de executar outra Activity
  - ▶ `startActivity()`
- ▶ Pode executar uma Activity esperando sua resposta
  - ▶ `startActivityForResult()`

# Trafegando dados usando o método `startActivity()`

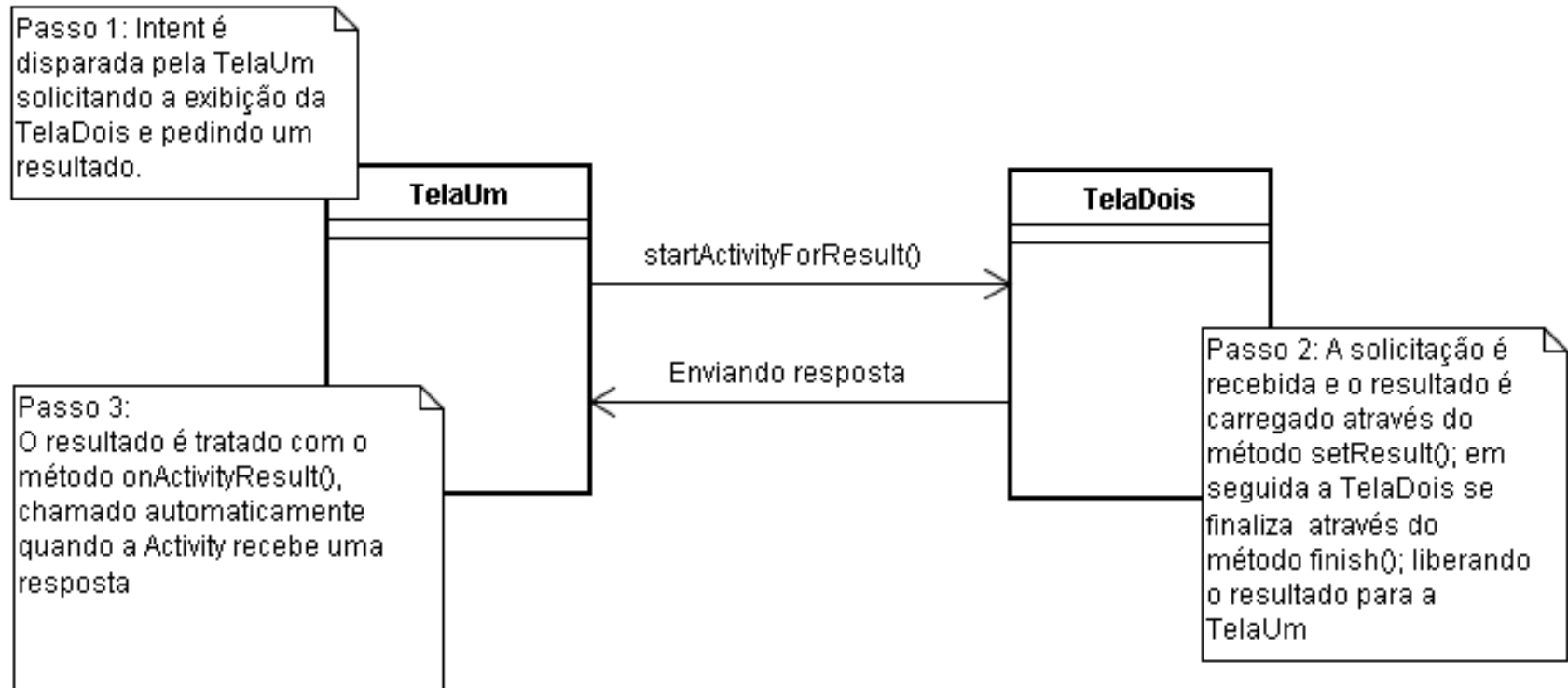
---



- ▶ Quando se usa o `startActivity()`, o desenvolvedor está dizendo que a Intent vai ser disparada, mas que a Activity que a disparou não tem a necessidade de receber nenhum resultado da próxima tela que será exibida.

# Trafegando dados usando o método `startActivityForResult()`.

---



# Código de função de retorno

## ► Activity

```
public void acessarIMC() {
    Intent intent = new Intent(getBaseContext(), IMCActivity.class);
    startActivityForResult(intent, IMC_RC);
}

protected void onActivityResult(int requestCode, int resultCode,
    Intent intentRetornada) {
    if (requestCode == IMC_RC) {
        if (resultCode == RESULT_OK) {
            Double imc = intentRetornada.getDoubleExtra("imc", 0);
            txtUltimoResultadoDeIMC.setText(imc.toString());
            Log.d("Marcador res", imc.toString());
        }
    }
}
```

## ► Activity

```
btnSair.setOnClickListener(new View.OnClickListener() {

    @Override
    public void onClick(View v) {
        Intent it = new Intent();
        it.putExtra("imc", resultadoCalculo);
        setResult(RESULT_OK, it);
        finish();
    }
});
```

# Navegando pelo Android - A Classe Intent

---

- ▶ Cenário: duas activities: uma para buscar dados e outra para listá-los.
- ▶ Na tela 1 uma série de componentes EditText para que o usuário possa fornecer as informações para a consulta:
- ▶ Os dados recuperados apenas poderão ser passados para a tela de lista caso uma **Intent** tenha os carregado através do método **putExtra()** e tenha sido disparada através do método **startActivity()** ou **startActivityForResult()**.

# Classe Intent

---

- ▶ Após a utilização dos métodos `setResult()` e `finish()`, a tela que disparou a `Intent` através de `startActivityForResult()`, além de voltar a ficar no topo da pilha de `activities`, recebe o resultado da `Activity` que foi finalizada.
- ▶ Para tratar esses resultados deve-se sobrescrever o método `onActivityResult(int requestCode, int resultCode, Intent intent)` da `Activity`.
- ▶ Esse método funciona como um `Listener` (Padrão de Projeto `Observer`), e é acionado quando a `Activity` que disparou uma `Intent` solicitando um resultado recebe a resposta desejada.

# A Classe Intent

---

- ▶ O método `onActivityResult()` possui três parâmetros:
  - ▶ `int requestCode`: Responsável por definir o identificador da requisição. Esse valor tem que coincidir com o `int` que foi passado no `startActivityForResult()`;
  - ▶ `int resultCode`: Responsável por definir o status da resposta. Indica se a resposta veio com sucesso ou falha;
  - ▶ `Intent intent`: Representa a Intent que é retornada como resposta. É nela que os dados podem estar armazenados.

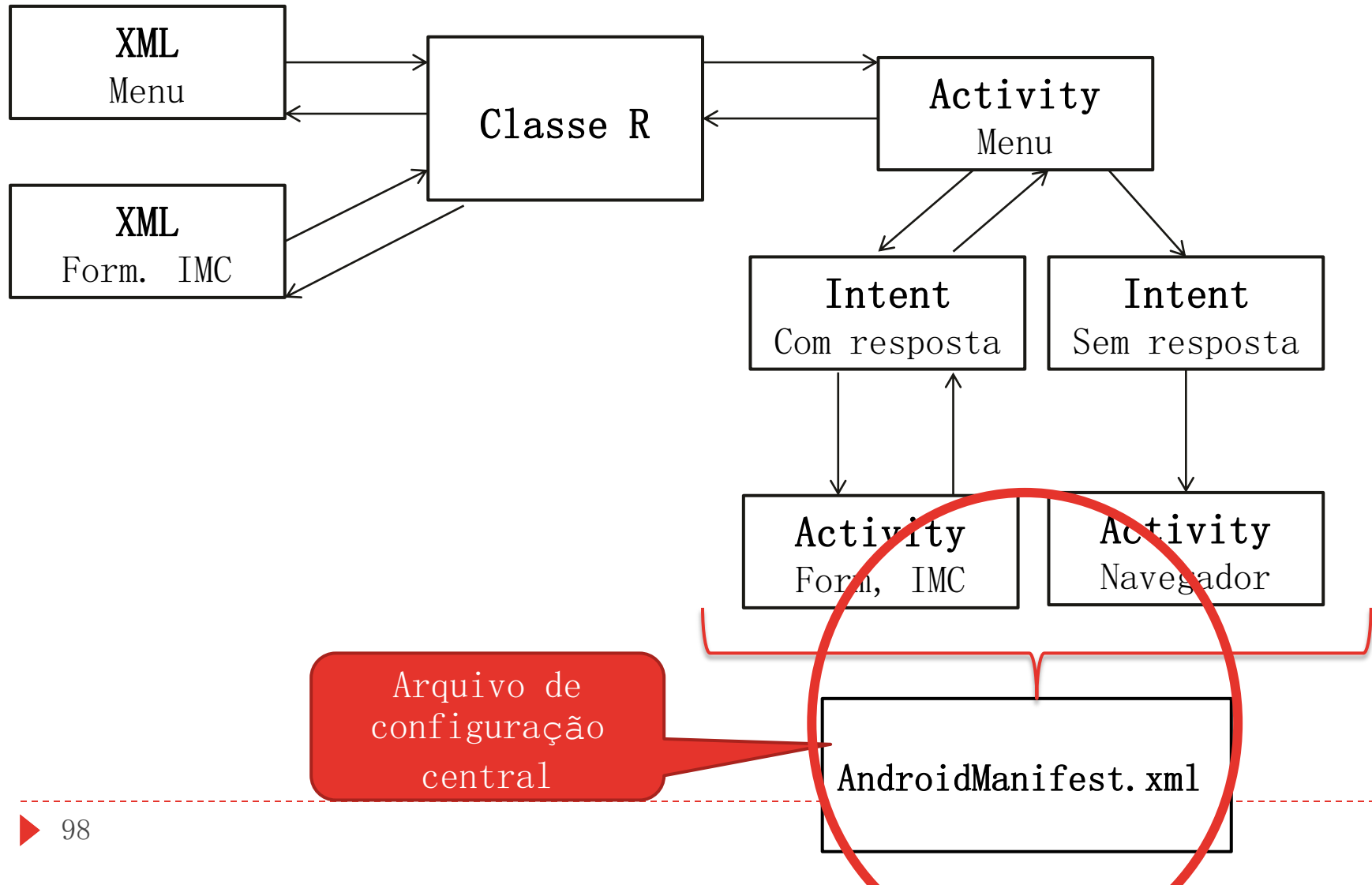


# A Classe Intent

---

- ▶ O método `setResult()` possui duas assinaturas diferentes, são elas:
- ▶ `setResult(int resultCode)`:
  - ▶ Retorna um `int`, que pode indicar, por exemplo, um retorno positivo ou negativo através das constantes `RESULT_OK` ou `RESULT_CANCELED`.
  - ▶ Esse método é útil quando não precisamos de qualquer informação extra, isto é, desejamos apenas saber se a solicitação realizada pela `Intent` foi um sucesso ou não;
- ▶ `setResult(int resultCode, Intent data)`:
  - ▶ Além do retorno do `int` referente ao código de resultado, será retornado também um objeto `Intent`.

# Estrutura Conceitual



# AndroidManifest.xml

---

- ▶ Arquivo de configuração onde deve-se informar todas as telas que o sistema vai exibir (as activities), serviços, permissões, filtros para intents (que serão abordados posteriormente), e assim por diante.
- ▶ No momento da instalação de uma aplicação no dispositivo, o Android exibe uma mensagem mostrando determinadas funcionalidades que o aplicativo fará uso e questiona ao usuário se ele ainda quer instalar esse sistema no seu aparelho.
- ▶ Trata-se de uma medida preventiva, para impedir que algum desenvolvedor malicioso desenvolva um software que, por exemplo, capture localizações GPS e as envie para um servidor web sem que o usuário perceba o que está acontecendo.

# AndroidManifest.xml

---

- ▶ Determinadas funções disponíveis para o desenvolvedor devem ser informadas no AndroidManifest.xml na tag `uses-permission`.
- ▶ Se o desenvolvedor não implementar esse procedimento, no momento em que a funcionalidade for acionada, uma exceção do tipo `SecurityException` será lançada.
- ▶ As responsabilidades desse arquivo vão muito além de permissões. É através dele que especificamos algumas bibliotecas “especiais”, que necessitam de algum controle maior por parte do fabricante, como é caso da biblioteca do Google Maps;

# AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="br.ufrgs.inf.pdp.android"
    android:versionCode="1"
    android:versionName="1.0" >
    <uses-sdk
        android:minSdkVersion="9"
        android:targetSdkVersion="17" />
    <uses-permission android:name="android.permission.INTERNET"/>
    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="br.ufrgs.inf.pdp.android.MainMenuActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name="br.ufrgs.inf.pdp.android.IMCActivity"></activity>
    </application>
</manifest>
```

Adicionar  
Permissão  
de uso da  
Internet

Adicionar a  
nova Activity  
criada



Dicas

# Dicas

---

- ▶ Deixar o emulador sempre aberto (lento para carregar)
- ▶ Para virar tela
  - ▶ Desligar num lock e usar teclas 7 e 9
  - ▶ CTRL F11 / CTRL F12
- ▶ Para trabalhar com Google Maps
  - ▶ Criar projeto como Google API (não Android....)
  - ▶ No caso não possua, baixar pelo SDK Manager pela API level que será utilizada.

# Tutorial de Instalação Eclipse IDE



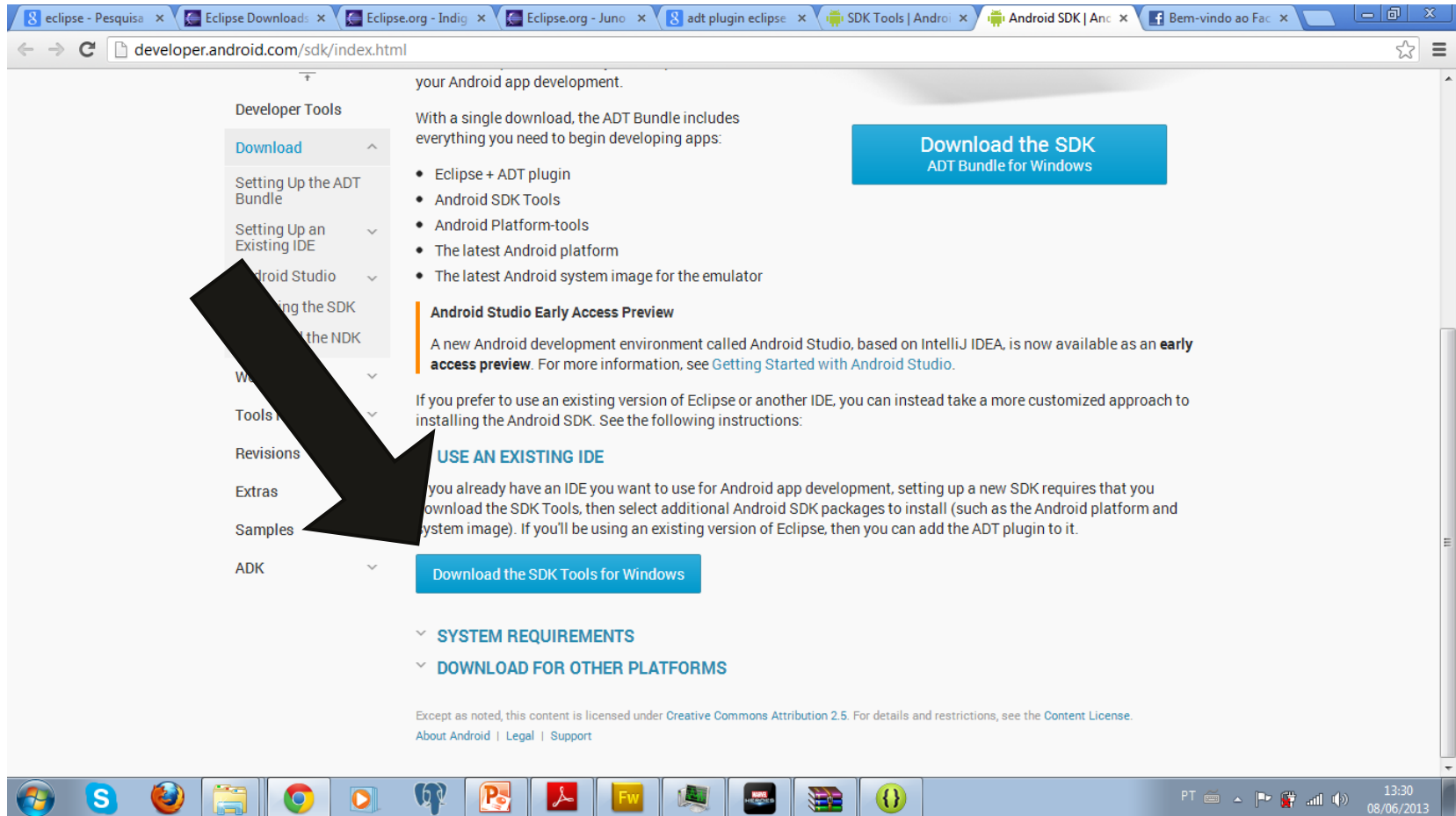
# Ambiente de Desenvolvimento

---

- ▶ SDK (Software Development Kit) do Android
- ▶ Eclipse IDE
- ▶ Android Development Tools (ADT) ( plugin para o Eclipse IDE)

# Download do SDK

► <http://developer.android.com/sdk/index.html>



The screenshot shows the Android SDK download page. A large black arrow points from the 'Samples' link in the left sidebar to the 'Download the SDK Tools for Windows' button. The page content includes:

- Developer Tools** sidebar with links: Download, Setting Up the ADT Bundle, Setting Up an Existing IDE, Android Studio, Installing the SDK, Installing the NDK, Windows, Tools, Revisions, Extras, Samples, ADK.
- your Android app development.**
- With a single download, the ADT Bundle includes everything you need to begin developing apps:**
  - Eclipse + ADT plugin
  - Android SDK Tools
  - Android Platform-tools
  - The latest Android platform
  - The latest Android system image for the emulator
- Download the SDK ADT Bundle for Windows** button.
- Android Studio Early Access Preview**

A new Android development environment called Android Studio, based on IntelliJ IDEA, is now available as an **early access preview**. For more information, see [Getting Started with Android Studio](#).
- If you prefer to use an existing version of Eclipse or another IDE, you can instead take a more customized approach to installing the Android SDK. See the following instructions:**
- USE AN EXISTING IDE**

If you already have an IDE you want to use for Android app development, setting up a new SDK requires that you download the SDK Tools, then select additional Android SDK packages to install (such as the Android platform and system image). If you'll be using an existing version of Eclipse, then you can add the ADT plugin to it.
- Download the SDK Tools for Windows** button.
- SYSTEM REQUIREMENTS**
- DOWNLOAD FOR OTHER PLATFORMS**
- Except as noted, this content is licensed under [Creative Commons Attribution 2.5](#). For details and restrictions, see the [Content License](#).  
[About Android](#) | [Legal](#) | [Support](#)

# Eclipse 4.2 - Juno

- ▶ <http://www.eclipse.org/downloads/packages/release/juno/sr2>

[Home](#) [Downloads](#) [Users](#) [Members](#) [Committers](#) [Resources](#) [Projects](#) [About Us](#)

Downloads Home ▾

- 🔗 [Juno Packages](#)
- 🔗 [Indigo Packages](#)
- 🔗 [Helios Packages](#)
- 🔗 [Galileo Packages](#)
- 🔗 [Ganymede Packages](#)
- 🔗 [Europa Packages](#)

## Eclipse Indigo Sr2 Packages

 **Eclipse IDE for Java EE Developers**, (212 MB)  
Downloaded 4,402,670 Times [Details](#)

 **Eclipse IDE for Java Developers**, (128 MB)  
Downloaded 1,662,174 Times [Details](#)

 **Eclipse IDE for C/C++ Developers (includes Incubating components)**, (108 MB)  
Downloaded 647,012 Times [Details](#)

 **Eclipse Modeling Tools**, (272 MB)  
Downloaded 179,483 Times [Details](#)

 **Eclipse IDE for JavaScript Web Developers**, (110 MB)  
Downloaded 176,729 Times [Details](#)

 **Eclipse IDE for Java and Report Developers**, (243 MB)  
Downloaded 137,530 Times [Details](#)

Windows [32-bit](#) [64-bit](#)

Mac Cocoa [32-bit](#) [64-bit](#)

Linux [32-bit](#) [64-bit](#)

Windows [32-bit](#) [64-bit](#)

Mac Cocoa [32-bit](#) [64-bit](#)

Linux [32-bit](#) [64-bit](#)

Windows [32-bit](#) [64-bit](#)

Mac Cocoa [32-bit](#) [64-bit](#)

Linux [32-bit](#) [64-bit](#)

Windows [32-bit](#) [64-bit](#)

Mac Cocoa [32-bit](#) [64-bit](#)

Linux [32-bit](#) [64-bit](#)

Windows [32-bit](#) [64-bit](#)

Mac Cocoa [32-bit](#) [64-bit](#)

Linux [32-bit](#) [64-bit](#)

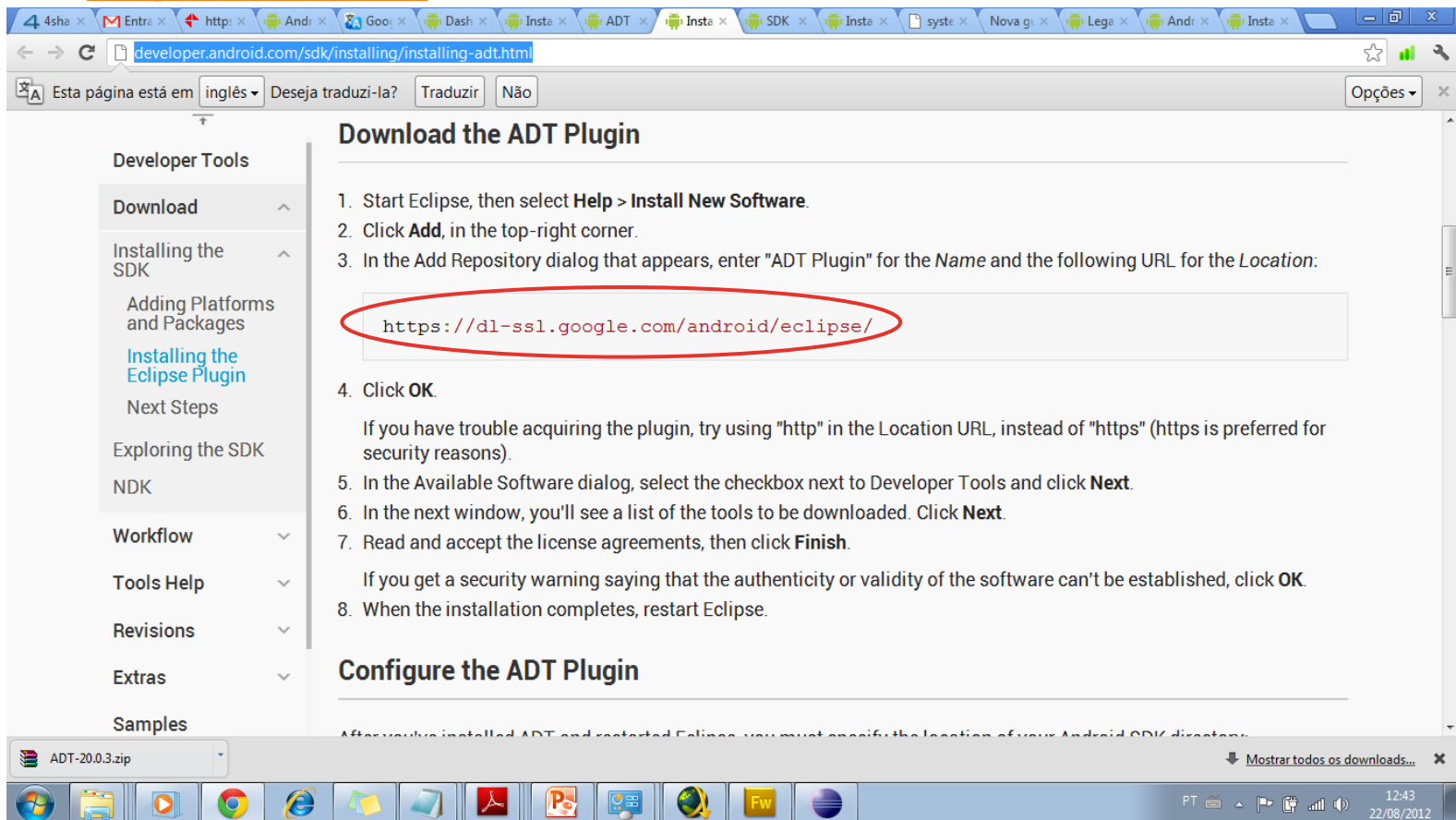
Windows [32-bit](#) [64-bit](#)

Mac Cocoa [32-bit](#) [64-bit](#)

Linux [32-bit](#) [64-bit](#)

# ADT Plugin

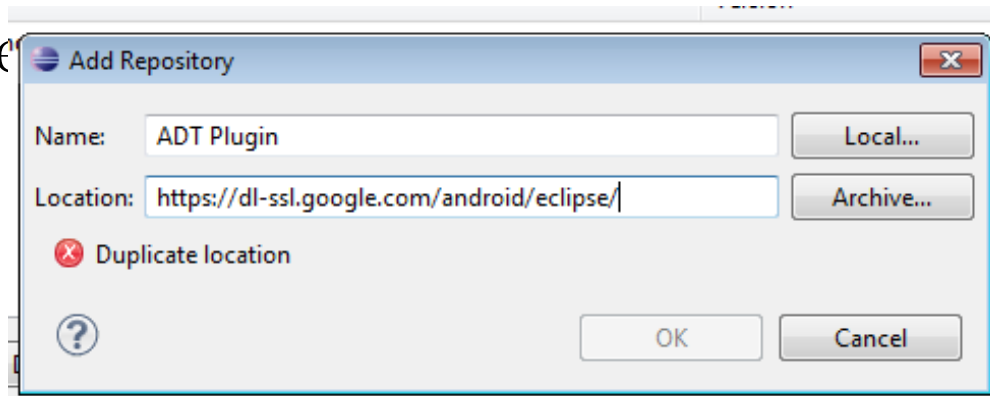
- ▶ <http://developer.android.com/sdk/installing/installing-adt.html>



# Instalando o ADT Plugin

---

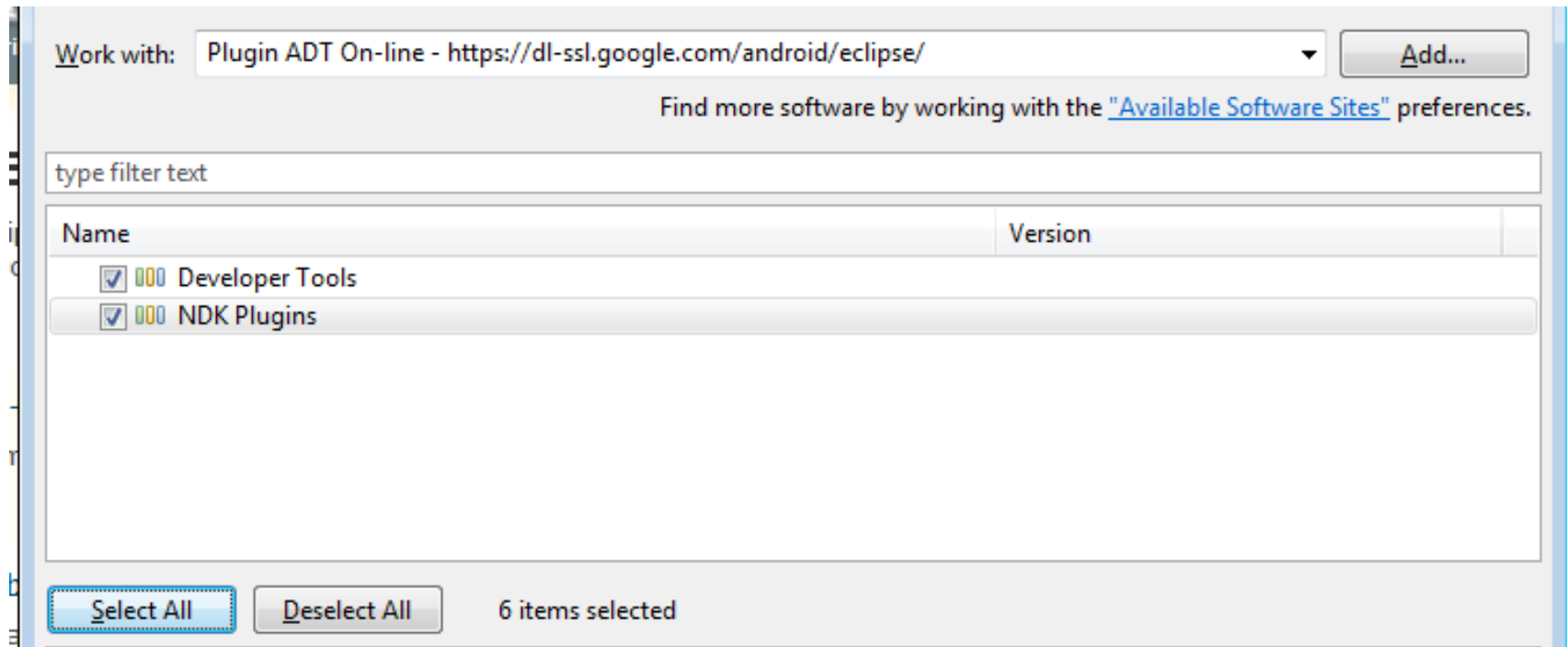
- ▶ Iniciar o Eclipse
- ▶ Entrar em **Help** > **Install New Software**.
- ▶ Clicar em **Add**, para adicionar um novo repositório
- ▶ Colocar as informações
  - ▶ Nome: ADT Plugin
  - ▶ URL: <https://dl-ssl.google.com/android/eclipse/>
- ▶ Clicar em **Add**



# Instalação do ADT Plugin

---

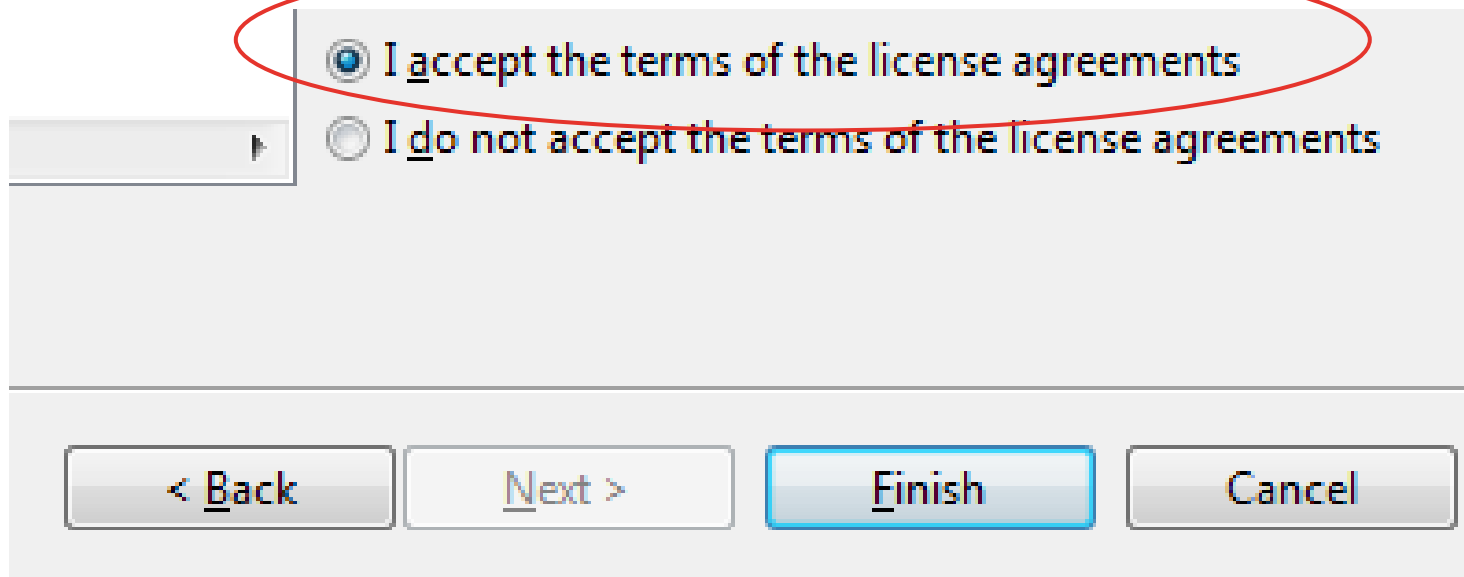
- ▶ Selecionar o repositório ADT Plugin
- ▶ Selecionar **todos** os itens e clicar em **Next>**



# Instalação do ADT Plugin

---

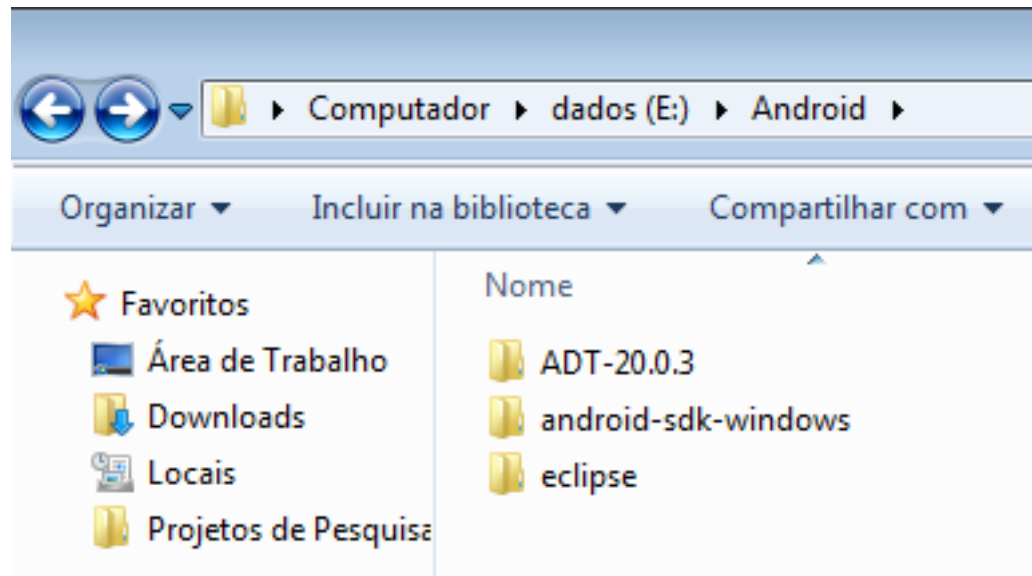
- ▶ Na próxima tela clicar em **Next>**
- ▶ Depois de clicar em **Next>**, aceitar os termos da licença e clicar em finalizar para instalar o Plugin.



# Sugestão (Para IDEs Eclipse)

---

- ▶ Criar pasta C:\android
- ▶ Descompactar SDK na pasta
- ▶ Descompactar Eclipse na pasta
- ▶ Descompactar Plugin ADT na pasta

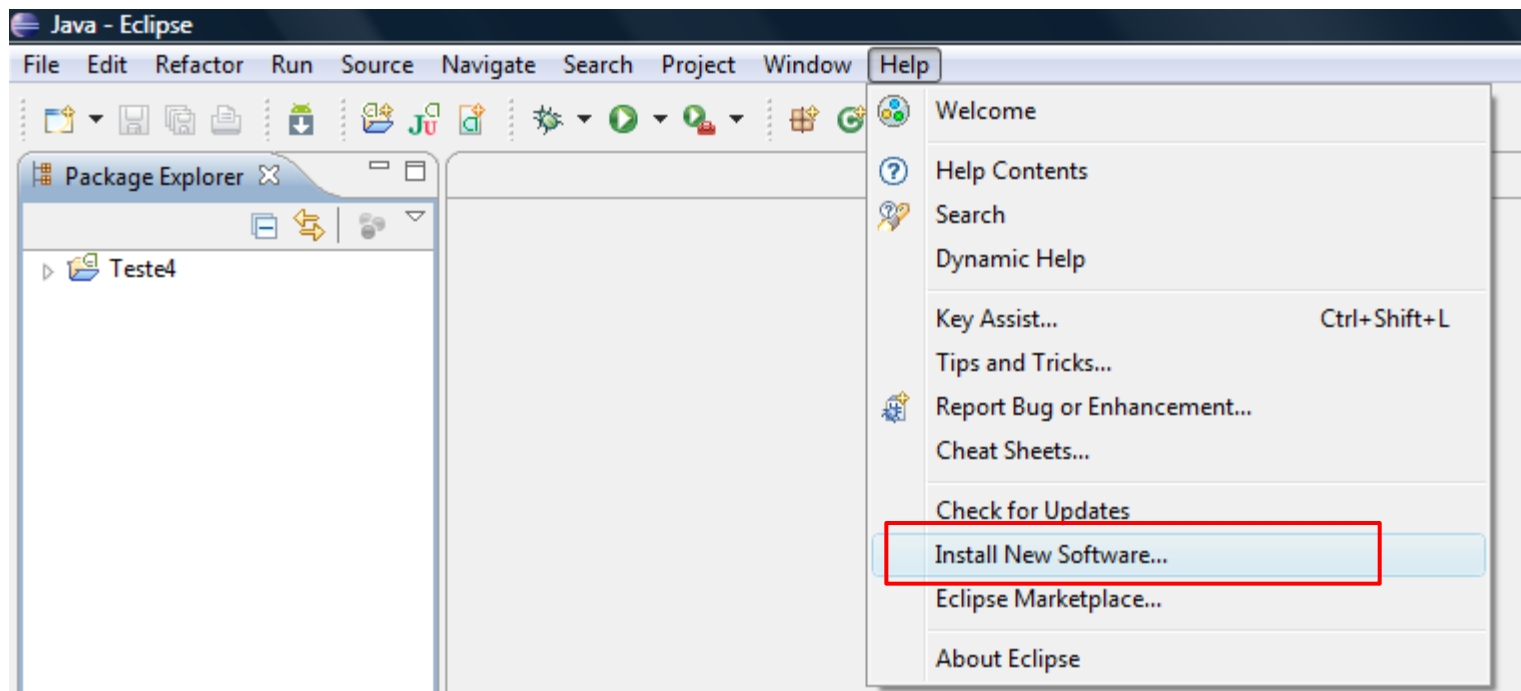




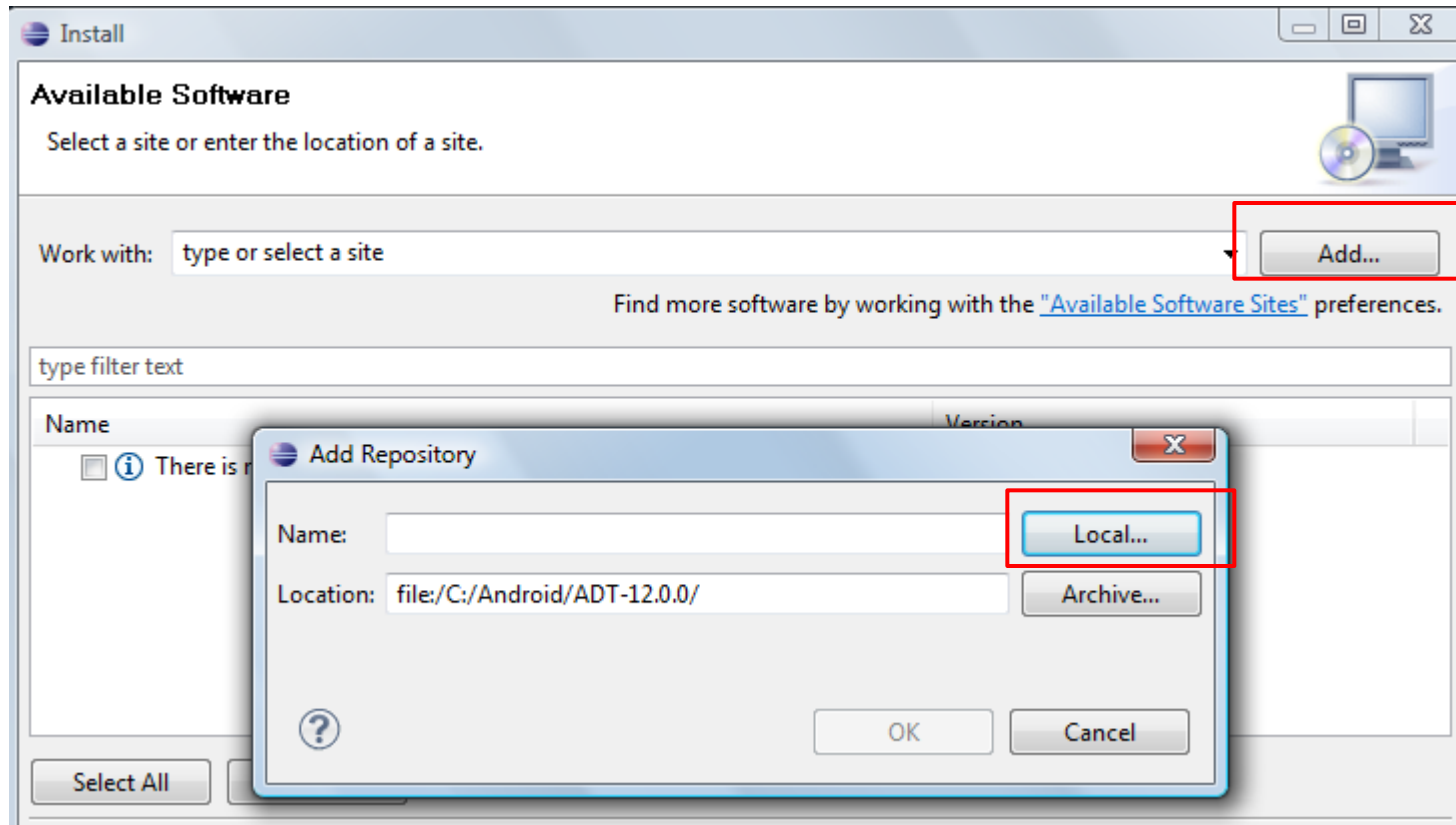
# Configuração Eclipse

---

► Menu Help → Install New Software



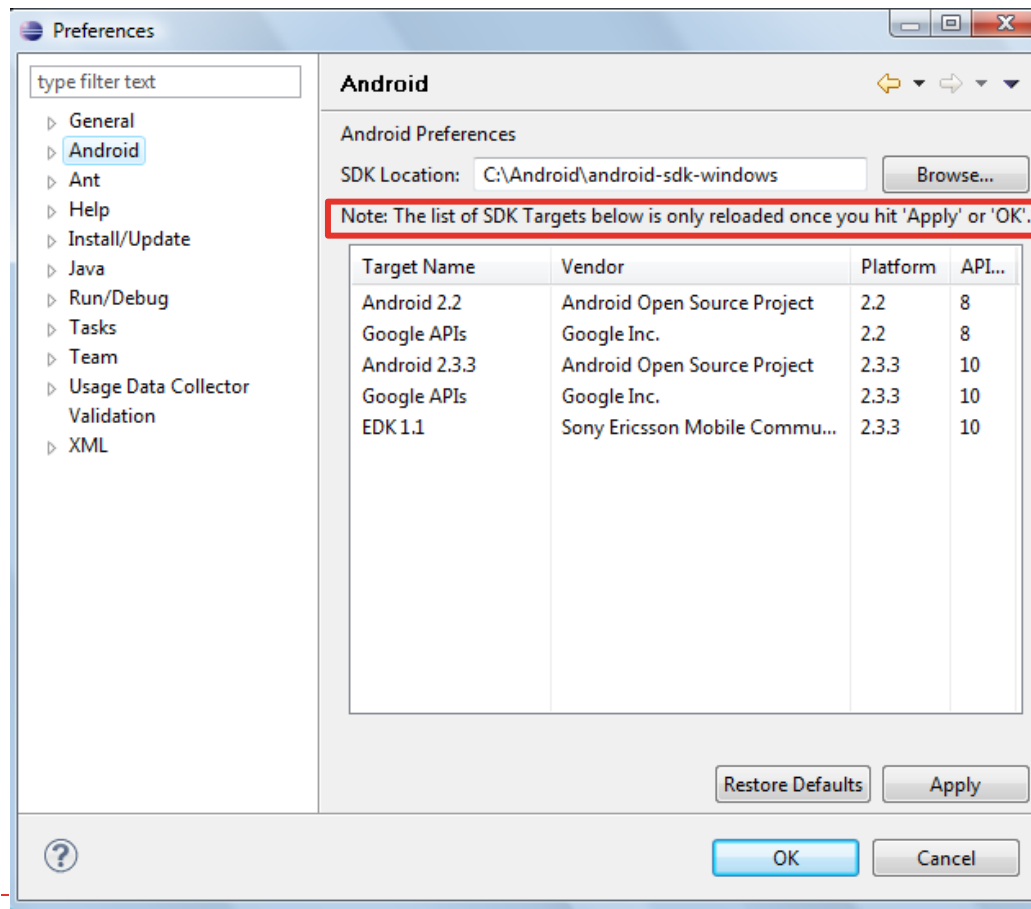
# Configuração Eclipse



- ▶ Depois de selecionar o local do ADT, clicar em Ok,
- ▶ Select All e Next (vai demorar um pouco)

# Configuração Eclipse

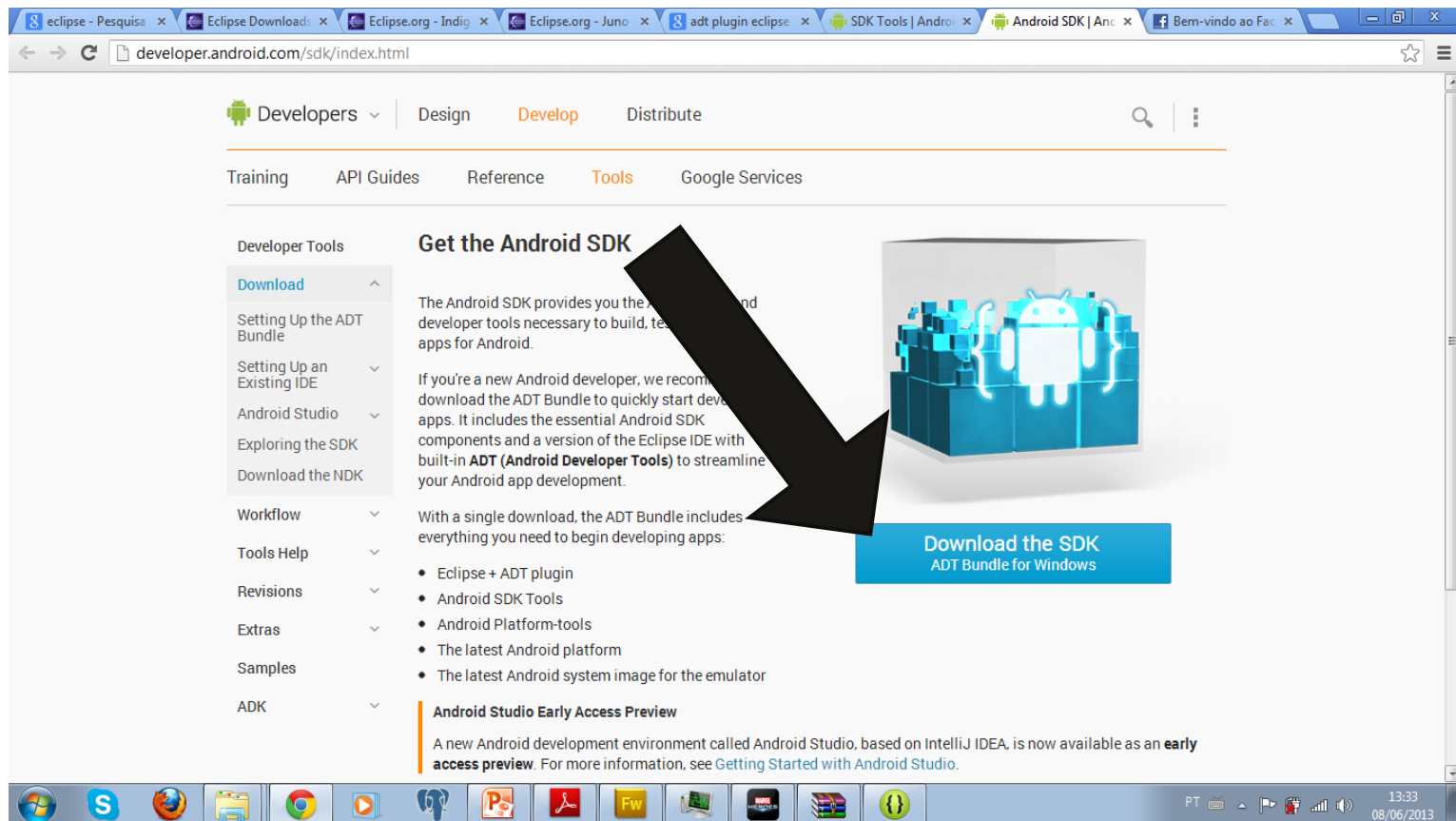
- ▶ Configurar caminho SDK no Eclipse
- ▶ Menu Window → Preferences



# Tutorial de Instalação Android Developer Tools

# Download do SDK

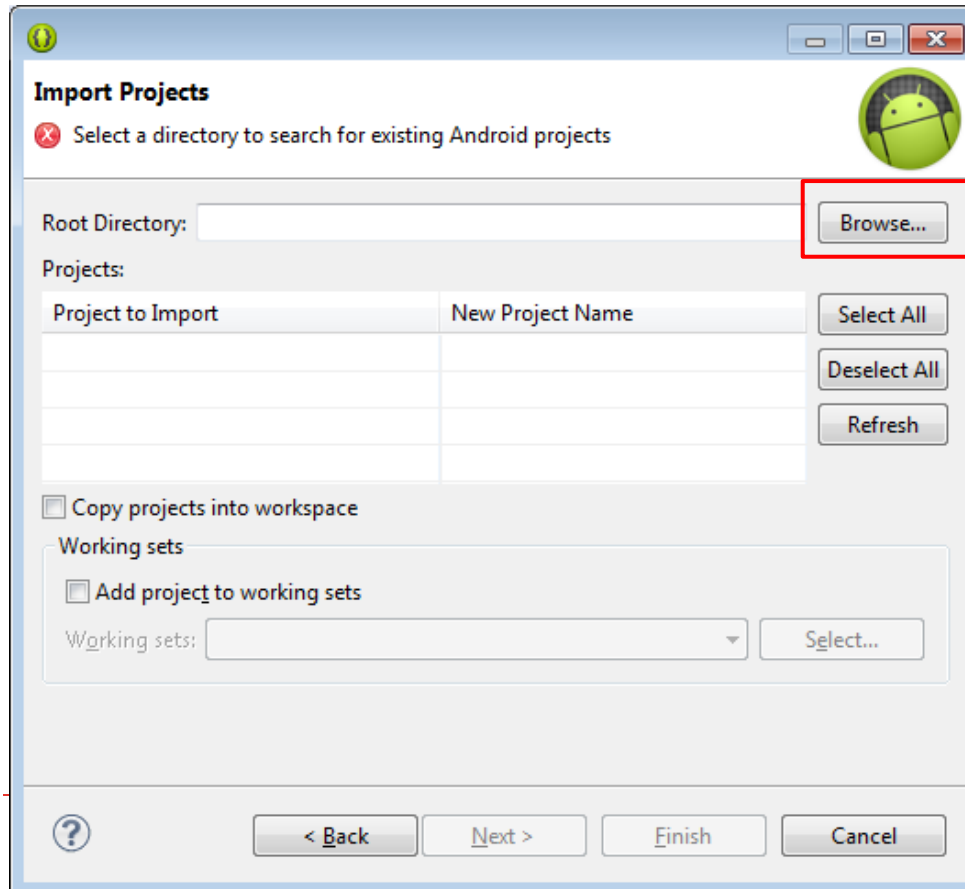
- ▶ <http://developer.android.com/sdk/index.html>
- ▶ Resumo: Baixar, Extrair e Executar.



# Tutorial de Importação de Projetos para a IDE

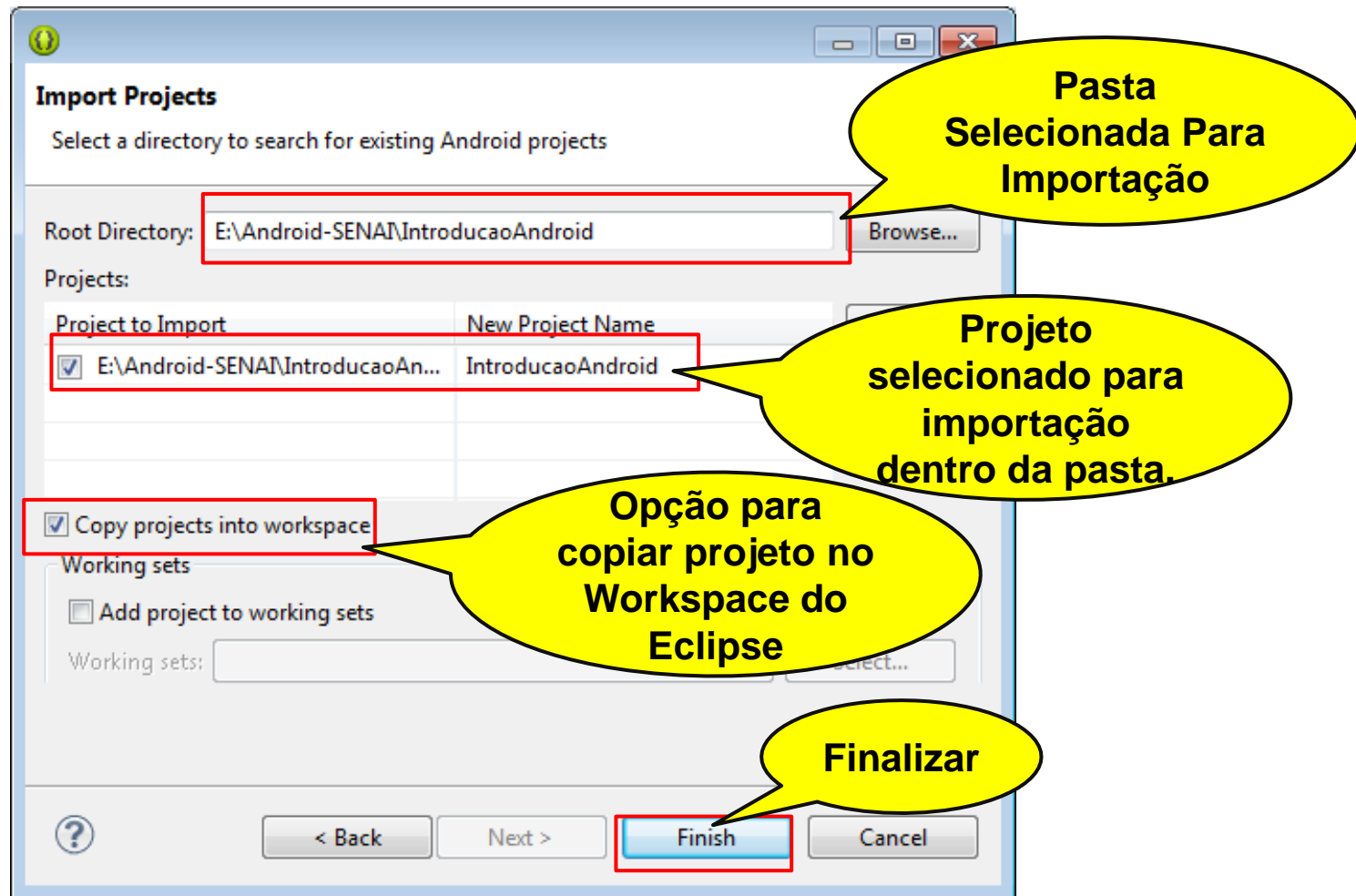
# Acessar opção de importação de projeto

- ▶ Acessar: Menu File -> Import -> Android -> Existing Android Code Into Workspace
- ▶ A seguinte tela deverá ser visível:



**Procurar a pasta do projeto a ser importado. OBS.: Fora do Workspace**

# Escolher projeto para importar





# Problemas que podem ocorrer

---

- ▶ Unable to resolve target 'android-17':
  - ▶ Indica que o a versão da importação é diferente da suportada pela IDE. Para resolver, deve entrar no arquivo AndroidManifest.xml e alterar o número na tag `android:targetSdkVersion="17"` *na versão sendo utilizada pela IDE, no caso caso da oficina "19".*
- ▶ Diretório SRC indicando erro:
  - ▶ Neste caso você deve limpar o projeto e após refatorar ele trocando o nome do projeto.



Dicas

# Dicas

---

- ▶ Deixar o emulador sempre aberto (lento para carregar)
- ▶ Para virar tela
  - ▶ Desligar num lock e usar teclas 7 e 9
  - ▶ CTRL F11 / CTRL F12
- ▶ Para trabalhar com Google Maps
  - ▶ Criar projeto como Google API (não Android....)
  - ▶ No caso não possua, baixar pelo SDK Manager pela API level que será utilizada.



# Introdução à Programação Android



Guilherme Antonio Borges

[guilhermeborges.pf@gmail.com](mailto:guilhermeborges.pf@gmail.com)