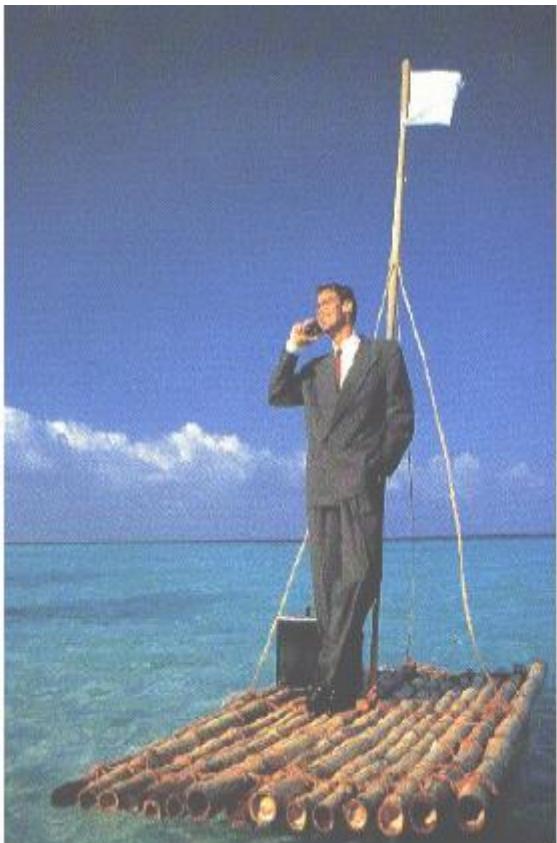


COMPUTAÇÃO MÓVEL



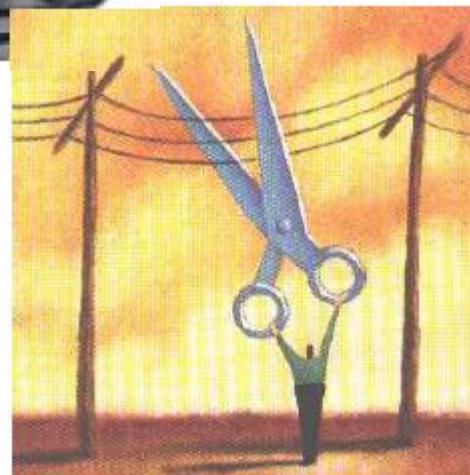
O que é computação Móvel

- Acesso à informação a qualquer lugar, a qualquer momento.



O que é computação Móvel

- Tecnicamente:
 - Processamento
 - Mobilidade
 - Comunicação sem fio



Dispositivos móveis

- O tamanho é importante



Dispositivos

- Notebooks



Ousborne 1981, 11 Kg



IBM 1986: 5,5 Kg



Apple 1991: ~3 Kg



Dell 2005: ~3 Kg



ClassMate 2007: 1 Kg



MacAir 2008: 0,5 cm

Outros Dispositivos

- PDAs
- Celulares
- Navegadores
- Robôs
- Sensores



Características comuns

- Interface limitada
 - Processamento
 - Comunicação
 - Energia

X	Interface	Proc.	Com.	Energ.
Notebook	Alta	Alta	Alta	Média
PDA/SmartPhone	Média	Média	Alta	Média
Embarcados	-	Baixa	Baixa	Baixa

Limitações

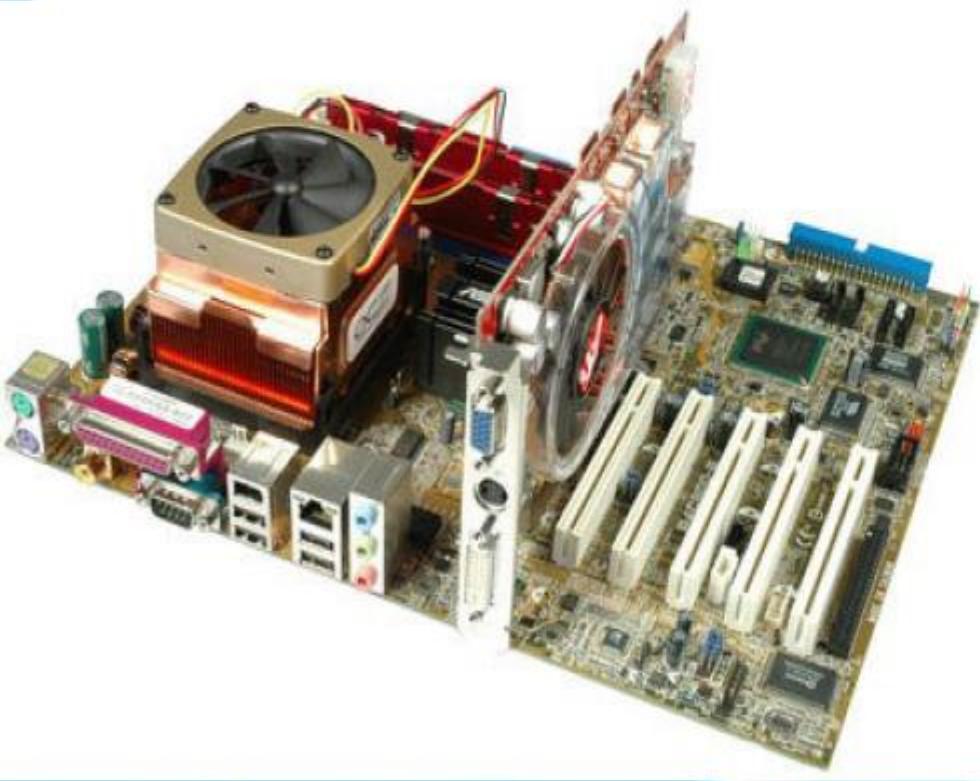
Inferfaces

- Monitor 19" => Tela 3"
- Teclado + Mouse => Reconhecimento de escrita e fala



Limitações

- Processamento
 - Processadores de alta capacidade
 - velozes
 - grandes
 - alta dissipação de energia!



Limitações

- Largura de banda



- Taxa de erros

Bits
Errados

10^{-12} a 10^{-15}

ou

\$0.40

10^{-3} a 10^{-6}

ou

\$40 000 000

Rede Fixa

Rede Móvel

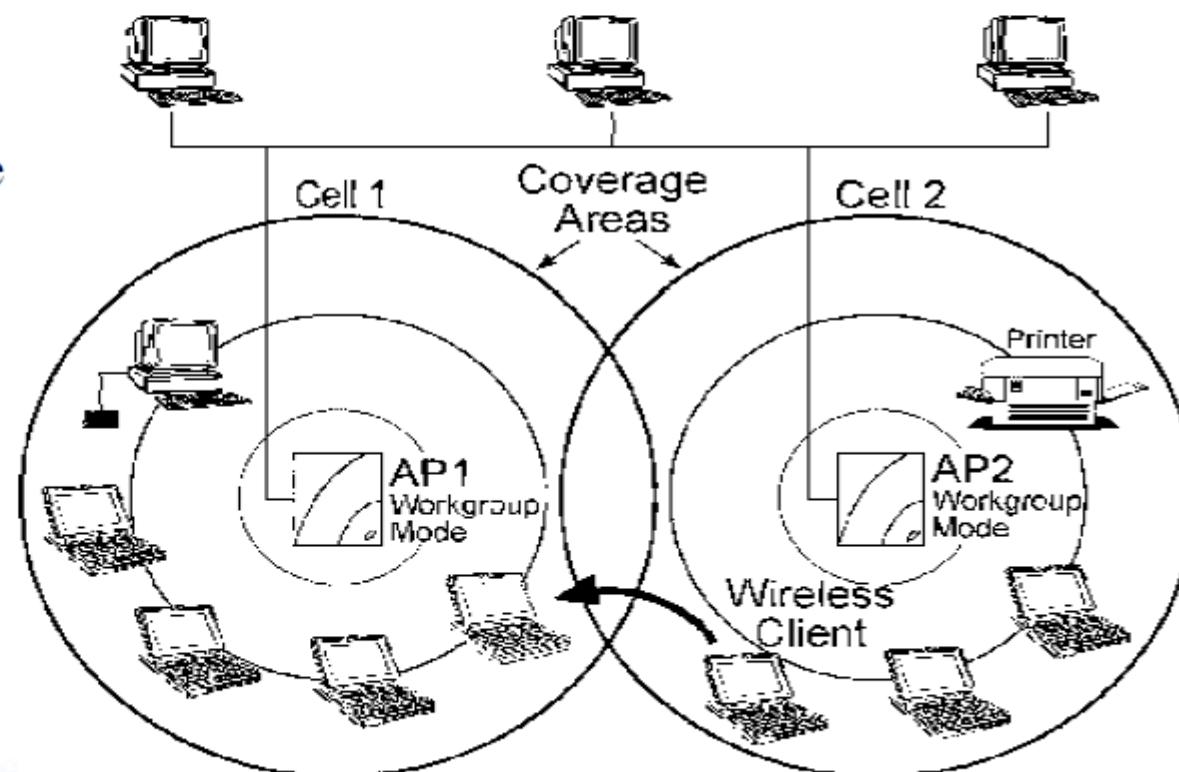
Energia

- Baterias têm duração limitada
- Perspectivas limitadas para aumento da capacidade da bateria
 - Não segue lei de Moore
- Soluções para preservar energia devem estar em toda a arquitetura
 - Hardware
 - Software
 - Protocolos de rede

Infra-estrutura

Interna

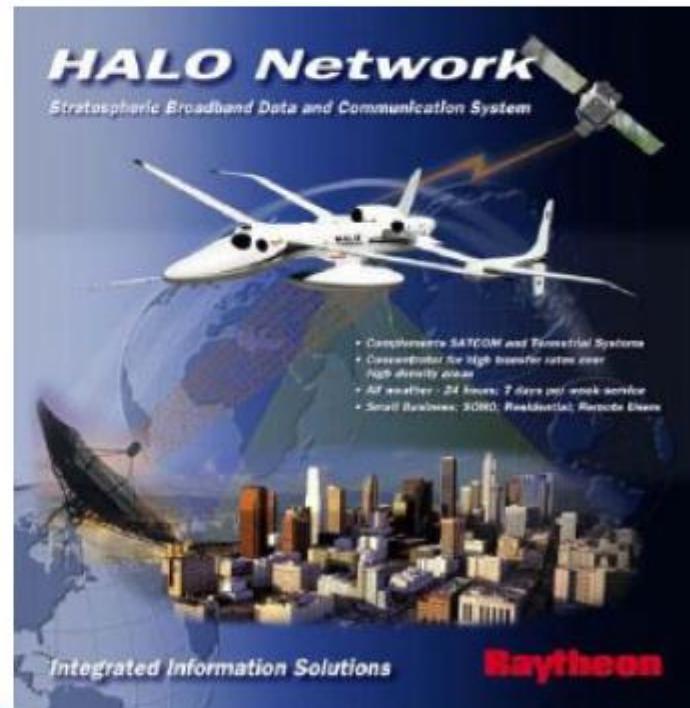
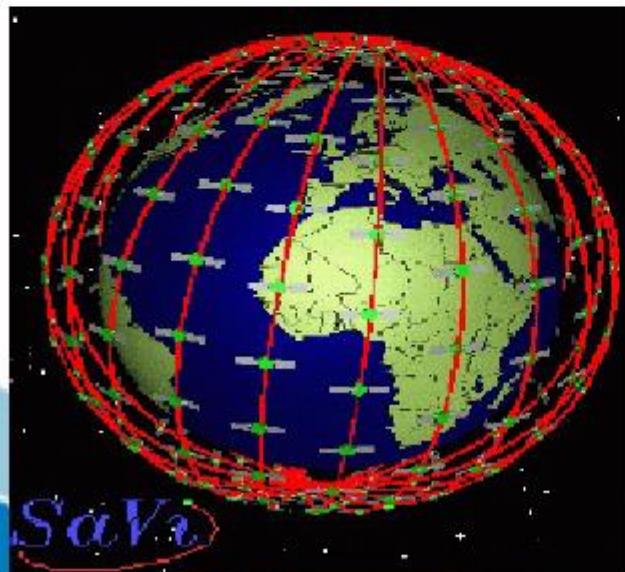
- Áreas Locais
- Tecnologias
 - 802.11 (Wi-fi)
 - Infra-vermelho
 - HomeRF
 - Bluetooth e ZigBee



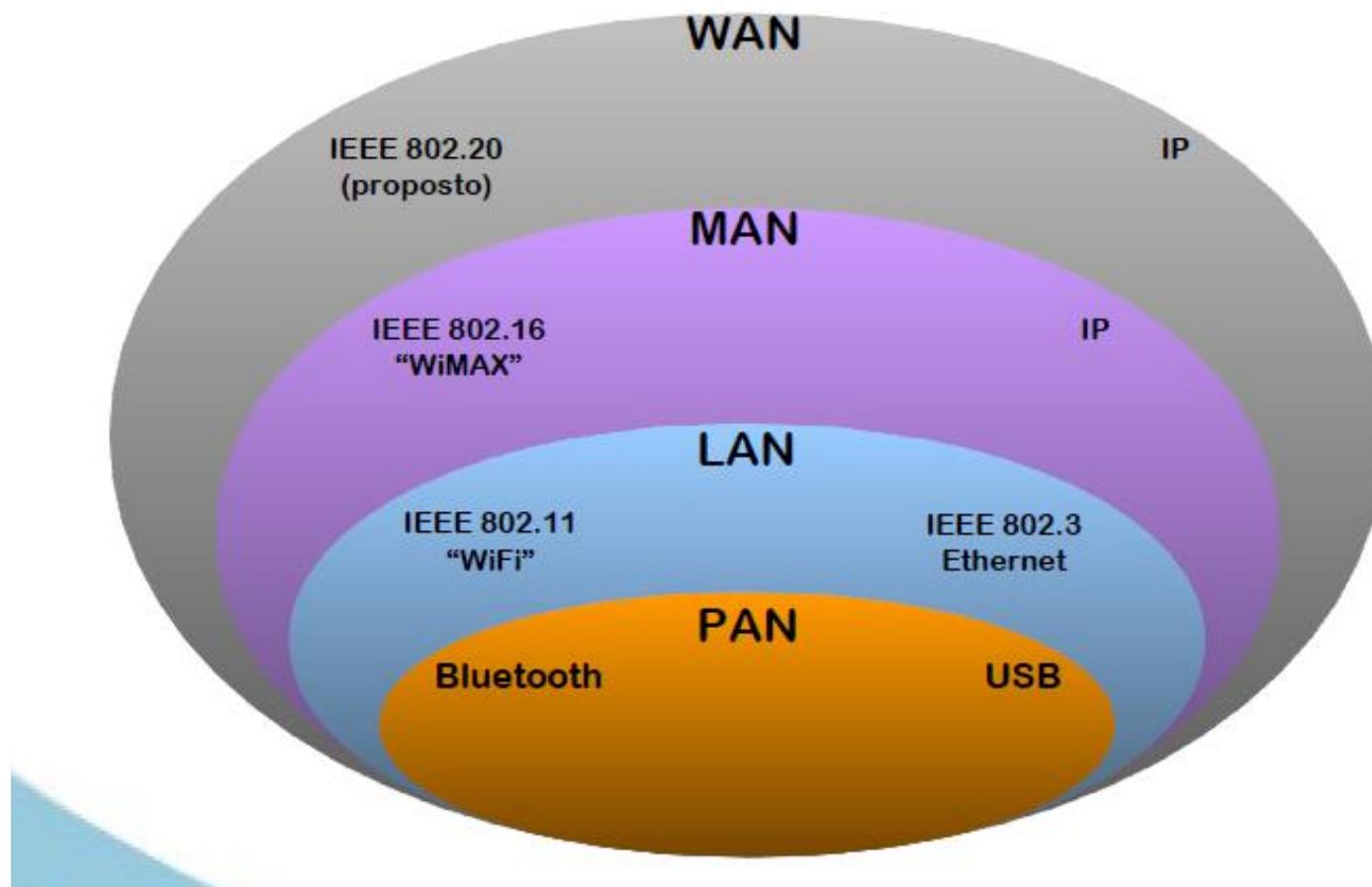
Infra-estrutura

Externa

- Áreas Metropolitanas ou Globais
- Tecnologias
 - Redes celulares
 - Wi-Max
 - Satélites
 - Aviões



Diversas Tecnologias



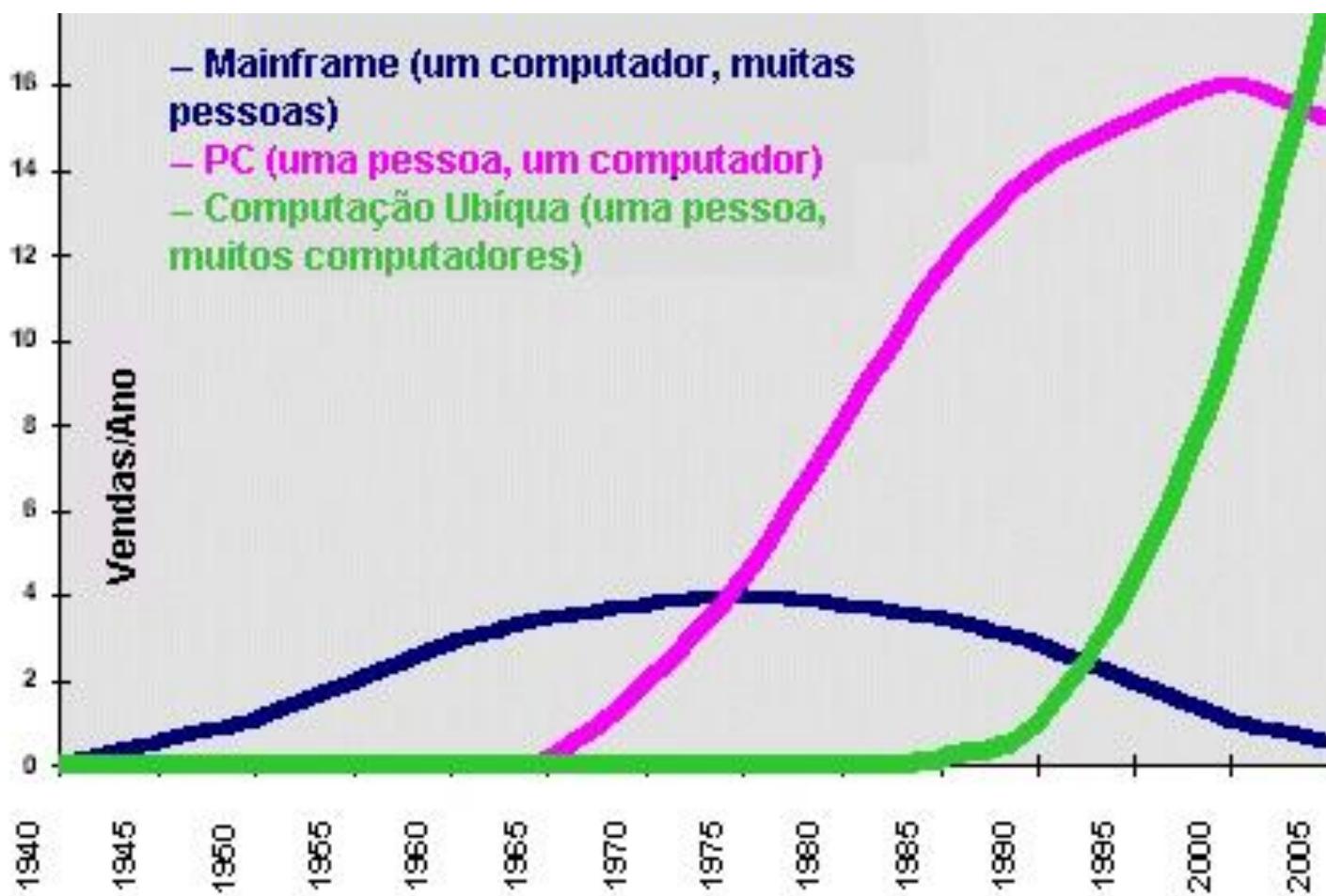
COMPUTAÇÃO UBÍQUA



O que é Computação Ubíqua?

- Ubíquo
 - adjetivo
 - 1 que está ou existe ao mesmo tempo em toda parte; onipresente
 - 2 que se difundiu extensamente; geral, universal
 - A ideia básica da computação ubíqua é que a computação move-se para fora das estações de trabalho e computadores pessoais e torna-se pervasiva na nossa vida quotidiana aonde quer que estejamos.
 - Computação Móvel + Computação Pervasiva

Tendência da Computação



Origem da Computação Ubíqua

- Idealizada por **Mark Weiser** que imaginou ambientes impregnados de computação, nos quais os dispositivos estão totalmente adaptados ao cotidiano.
- Ambientes: espaços físicos quaisquer - salas de aula, escritórios, edifícios.

Principais características da Computação Ubíqua

- Diversidade
- Descentralização
- Conectividade
- Onipresença
- Mudança na relação homem – máquina
 - (o papel do homem passa a ser mais passivo
 - x
 - computador deixa de ser o foco das atenções)
- Calm Technology
 - a integração é tranqüila e até imperceptível (computação invisível)

Tecnologias envolvidas

Hardware

- dispositivos de redes e de computação móvel

Software

- sistemas distribuídos e ferramentas de desenvolvimento

Modelagem de contexto

- sensoriamento e processamento de imagens

Interação

- interfaces hands-free e adaptação de interfaces a dispositivos de hardware

Aplicações

- projeto de novas aplicações e aspectos sociais da computação ubíqua

Tecnologias envolvidas

Computação Móvel

- Dispositivos pequenos: podem ser facilmente carregados, enquanto o usuário se movimenta livremente.
 - **Computadores móveis:** computadores pequenos; provisão e gerenciamento de energia elétrica; interfaces amigáveis, mas adaptadas ao tamanho do dispositivo; criação de novos dispositivos de interface
 - **Computadores Wearable:** projetados para o uso sem necessidade das mãos, podendo usar sensores (câmeras e microfones), e formas convenientes de teclados
 - **Conexão Wireless:** conexão wireless contínua à rede, mantendo o serviço funcionando mesmo os dispositivos em movimento

Tecnologias envolvidas

Computação Pervasiva

- Dispositivos operam a distância: o usuário não precisa estar fisicamente próximo a eles
 - **Interfaces Hands-Free:** reconhecimento de voz, liveboards, e outras interfaces, que juntas permitem que o usuário interaja, mesmo fisicamente distante dos dispositivos.
 - **Consciência de Contexto:** sensores que detectam o que está acontecendo e o que as pessoas estão fazendo no ambiente de forma geral. Informação representada de algum modo e disponibilizada para consulta por aplicativos, que têm uma idéia de o que está acontecendo ao redor do usuário.
 - **Ambiente Inteligente:** comportamentos automáticos ativados por determinados acontecimentos, sem nenhuma instrução explícita do usuário - Computação Invisível

Tecnologias envolvidas

Computação Ubíqua

- Computação Móvel + Computação Pervasiva
 - **Computação Desagregada:** reconfiguração dinâmica dos dispositivos de interface. Exemplo: a possibilidade de fazer sua apresentação mover-se para qualquer tela da sala. O "computador" é um grupo de diversos dispositivos conectados, que estão na verdade unidos a diferentes computadores na rede.
 - **Computação Sensível a Posição:** interação com os computadores muda, enquanto as pessoas se movem. Exemplo: guia automático de excursão em um museu; automaticamente mover seu desktop para o display mais próximo, enquanto você anda pela sala.
 - **Realidade Aumentada:** computadores wearables são combinados com a informação dos sensores de posição, a informação relevante ao usuário pode ser sobreposta à sua visão do mundo, vista através de um head-mounted display.
 - **Interfaces Sensíveis a Objetos:** associar objetos físicos a alguma informação. Exemplo: associar um objeto à webpage de seu fabricante.

Computação Ubíqua e Sistemas Operacionais

- Grande variedade de dispositivos computacionais gera necessidade da criação de SOs específicos para cada dispositivo
 - A especialização do dispositivo é um dos aspectos que determina o projeto do sistema operacional do dispositivo
- Exemplos de SO:
 - Palm OS (PDA), EPOC (celular), *Java Card* e *W/Smart Card para Smart Cards*, QXN, VxWorks etc

Desafios da Computação Ubíqua

Privacidade

- A proliferação de sensores e modelos de contexto irá armazenar grandes quantidades de informação a respeito das atividades dos usuários

Complexidade

- Quanto mais coisas acontecem automaticamente, mais confuso o sistema pode tornar-se para o usuário

Expansibilidade

- Sistemas de computação ubíqua serão feitos de muitas partes de hardware e de software, de muitas procedências

Segurança

- Se tudo está conectado, como prevenir e limitar ataques de programas ou hardware não-autorizados?

Desafios da Computação Ubíqua

Custo

- A quantidade de dispositivos computacionais é grande e consome energia.

Complexidade

- Quanto mais coisas acontecem automaticamente, mais confuso o sistema pode tornar-se para o usuário

Expansibilidade e Interoperabilidade

- Sistemas de computação ubíqua serão feitos de muitas partes de hardware e de software, de muitas procedências

Tolerância a falhas

Mobilidade



Introdução

- Atualmente, a **Computação Móvel** vem surgindo como um novo paradigma computacional.
- As redes que suportam a computação móvel são as Redes Móveis, que trazem novos requisitos e desafios não encontrados em redes de computadores tradicionais.

Introdução (cont.)

- A **Mobilidade** é a principal característica das Redes Móveis. Ela traz problemas e desafios que até então, não víamos, ou ignorávamos em ambientes fixos.
- A mobilidade impõe requisitos e gera problemas:
 - roteamento;
 - velocidade do canal;
 - interferências do ambiente;
 - localização da estação móvel;
 - duração da energia da bateria da estação parada e em movimento;
 - entre outros.

Portabilidade

- É a capacidade de um terminal móvel operar a partir de diferentes pontos de conexão, mas perde o contato durante o tempo de mudança do ponto de acesso.
 - ao se mover, as conexões são encerradas e reinicializadas no novo ponto de conexão. Ex: WLANs (IEEE 802.11).

Mobilidade

- É a capacidade de um terminal móvel continuar em contato contínuo com os recursos da rede.
 - nem o sistema, nem as aplicações precisam ser encerrados e reinicializados;
 - modo de acesso a rede: interface sem-fio ;
 - redes móveis.

Redes Móveis

São redes de computadores sem fio que possuem nós móveis:

- Redes Infra-Estruturadas:
 - Rede de telefonia celular;
 - Wireless LANs (IEEE 802.11, HIPERLAN);
 - Wireless ATM;
 - Redes via satélite.
- Redes Sem Infra-Estrutura:
 - Redes Móveis *Ad-hoc* (MANET);
 - WPAN – IEEE802.15(*Bluetooth*);
 - Redes de Sensores;
 - Redes Tolerantes a Atraso (DTN)

Estrutura das Redes Móveis

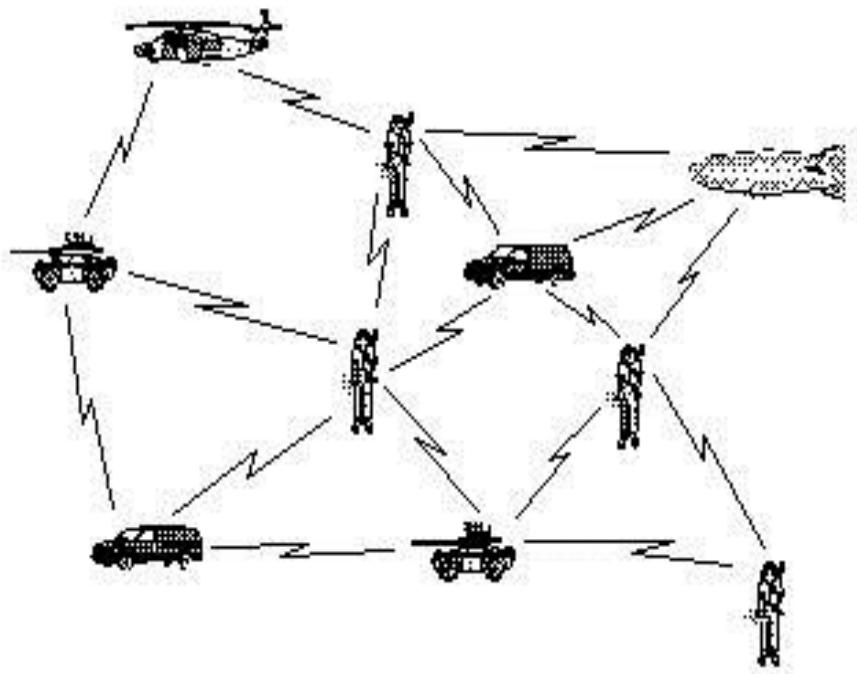
- Parte Fixa (redes de computadores tradicionais):
 - ERB- Estação Rádio Base;
 - ESM - Estação de suporte à mobilidade;
 - Estações Fixas (servidores, roteadores).
- Parte Móvel (equipamentos móveis):
 - Estações Móveis (*notebook*, celular, *palmtop*, PDA, sensores).
- Existem pesquisas propondo redes totalmente móveis:
 - Ex : Rede Móvel *Ad hoc*.

Problemas em Redes Móveis

- Mobilidade do usuário;
- Instabilidade (variação das condições do canal de comunicação sem fio);
- Baixa largura de banda (bandwidth);
- Alta taxa de erros (10^{-5} bits errados);
- Gerenciamento do consumo de energia da estação móvel;
- Suporte à QoS;
- Segurança.

Redes Móveis *Ad hoc*

- São redes, onde os dispositivos computacionais trocam informações diretamente entre si.
- IETF criou grupo de trabalho em MANET (*Mobile Ad-hoc NETwork*) - RFC 2501, RFC .



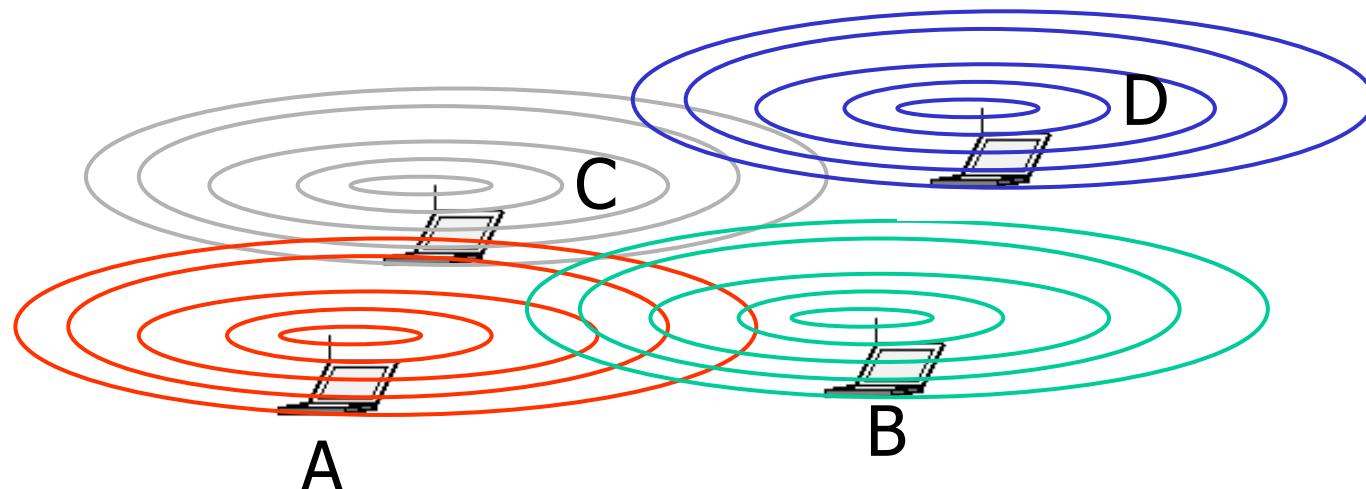
Indicadas para situações onde não se pode, ou não faz sentido, instalar uma rede fixa.

Vantagens

- Rápida instalação:
 - Excelente para cenários de desastre, campos de batalha ou conferências onde não existe uma estrutura prévia ou esta não está disponível.
- Tolerância à falhas:
 - Vários caminhos podem ser criados.

Vantagens

- Conectividade:
 - Os nós dentro da área de alcance podem trocar informações diretamente.



Desvantagens e dificuldades

- Localização:
 - Encontrar o nó móvel.
- Movimentação dos nós:
 - Nós não necessariamente seguem algum padrão de movimentação.
- Desligamento sem aviso dos nós:
 - O nó pode passar por períodos sem contato com a rede, ou mesmo desligados, e reaparecer em algum lugar imprevisto.

Desvantagens e dificuldades

- Qualidade do canal:
 - Canal sujeito a variações na qualidade.
- Baixa banda passante.
- Consumo de energia:
 - Tráfego de mensagens que não dizem respeito diretamente ao nó.
- Nós de capacidades e características diferentes.

Aplicações

- Fins militares;
- Cenários de catástrofes:
 - Furacões;
 - Terremotos;
 - Enchentes.
- Busca e salvamento;
- Conferências;
- Controle de tráfego;
- Qualquer outro cenário de troca de informações direta entre nós móveis que possa ser imaginado.

Diferentes pontos de vista

- Comunidade Militar:
 - Redes pequenas;
 - Mensagens pequenas, normalmente de controle;
 - Principal problema é encontrar os nós de forma eficiente e no menor espaço de tempo possível;
 - Não tem muita preocupação com a eficiência da rede ou com economia de energia.

Diferentes pontos de vista

- Comunidade Internet:
 - Redes grandes;
 - Mensagens grandes;
 - Grande fluxo;
 - Atraso, em alguns casos, não é um grande problema;
 - Principais pontos: eficiência e economia de energia;
 - Capacidade da implementação de múltiplos caminhos (*Multipath*);

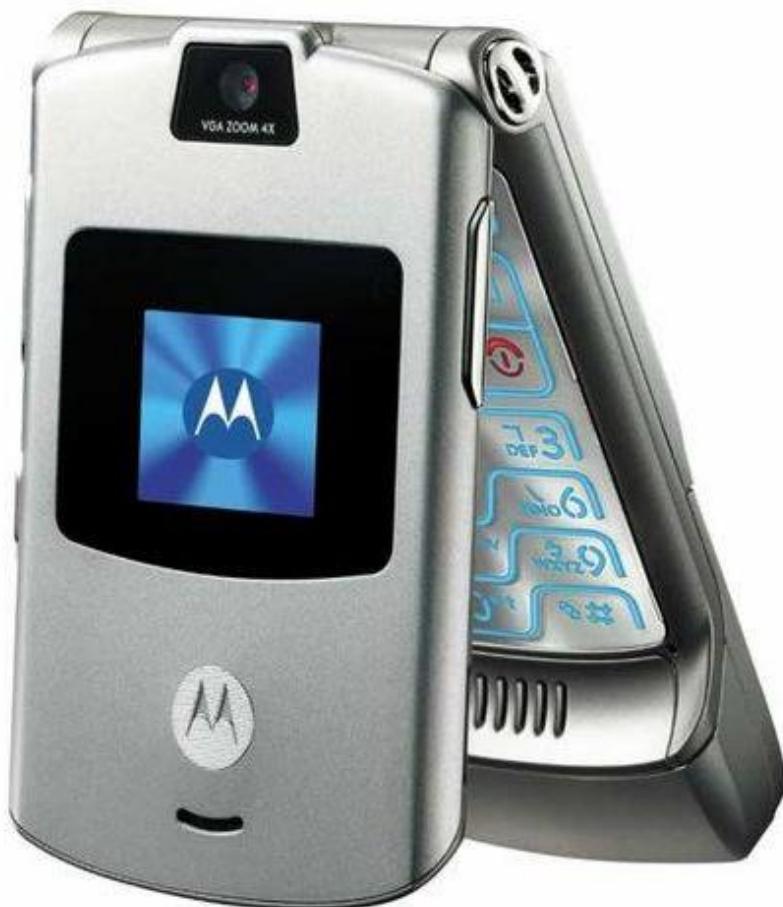
PLATAFORMA MÓVEL DIGITAL EMERGENTE

Tipos de tecnologias “mobile” - tablets, smartphones, etc. e seus impactos.

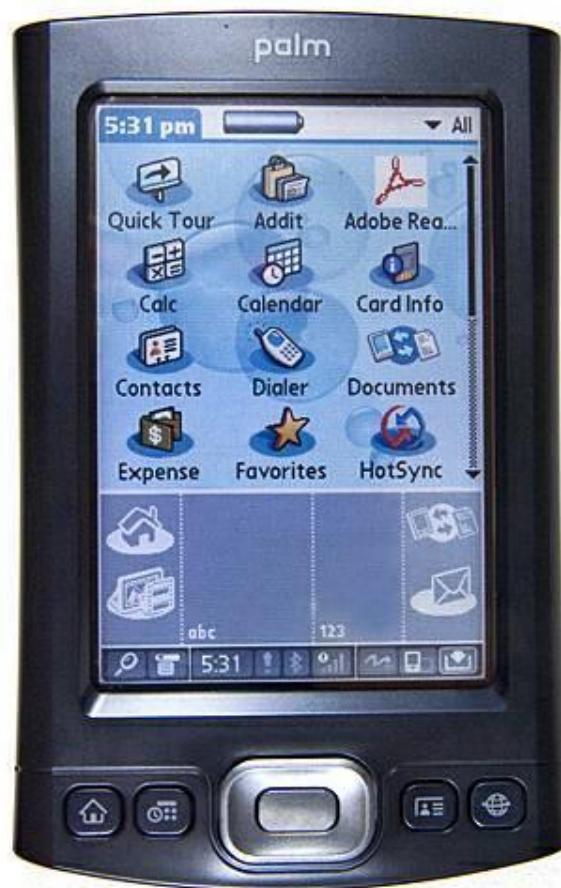


Você já teve um desses!

Celular



PDA (Assistente Pessoal Digital)



Você tem um desses

- Uma definição bastante comum para smartphone é “telefone celular com capacidade de PDA”;
- Na época do seu surgimento, a principal diferenciação em relação aos celulares era a possibilidade de acessar a internet (envio de e-mails);
- Atualmente, os smartphones são reprodutores de mp3, câmeras, entre outros.



BlackBerry Bold
9900



História

- A Research in Motion (RIM) foi fundada em 1984 – o primeiro BlackBerry chegou ao mercado em 1999;
- O produto era distribuído para executivos em grandes eventos e o uso era gratuito por um dia. Os executivos gostavam e acabavam adquirindo o aparelho;
- A ideia inicial do BlackBerry foi substituir o Palm – fazia tudo que a tecnologia PDA oferecia, porém, contava com uma novidade: a “tecnologia push”.

História

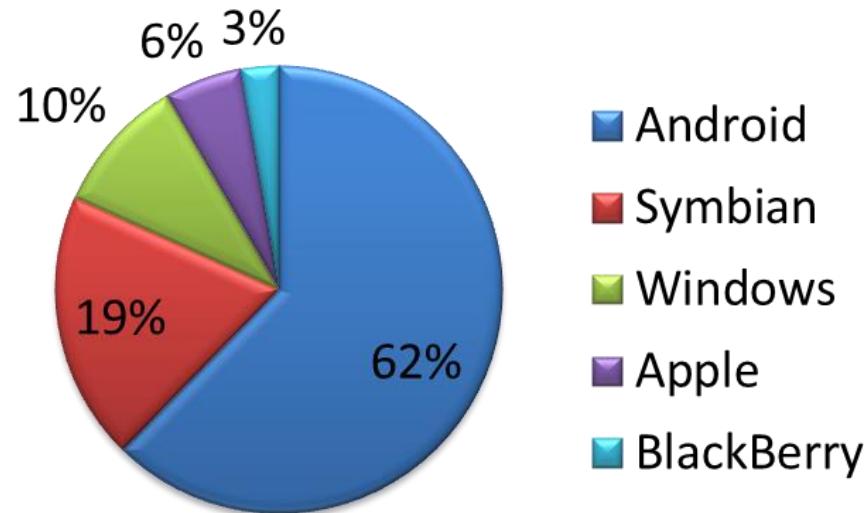
- Não há consenso sobre qual foi o primeiro smartphone fabricado, porém, o iPhone foi o primeiro smartphone a ser um sucesso de vendas;





Números

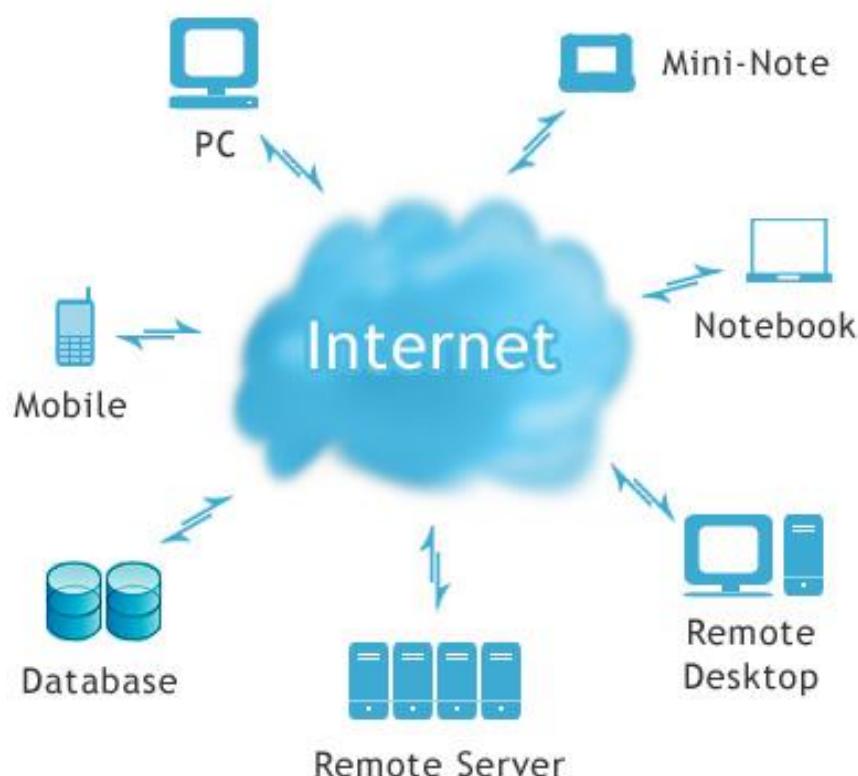
Crescimento das vendas de Smartphones (2011)	
Mundo	61% (média)
Brasil	179%



A venda de smartphones representa 7,5% da venda total de celulares

Queda de 33% no preço médio dos aparelhos

Cloud Computing



- Pode ser considerado como uma evolução da tecnologia push: é possível acessar seus dados (e não apenas e-mails ou contatos) de todos os tipos de hardwares que uma pessoa possui;
- Dropbox, iCloud, Google Docs, LogMe In,

Tendências (MWC 2012)

Processadores quad-core

Grandes telas HD

NFC (comunicação por proximidade)

4G (LTE)

Redução de preços



Sony Xperia P



Nokia Lumia 900



LG Optimus Vu



Caso Prático - HTC

- Demanda por smartphones deve chegar a 70%;
- Concorrência nos softwares – Android (Google) X Windows Phone (Microsoft);
- Crescimento da demanda – troca dos aparelhos antigos por mais modernos;



Mobilidade

- Crescimento da demanda - 134%;
 - Fabricantes retomam mercado - europeus e americanos X asiaticos;
 - 24,5% do mercado – Motorola
 - 16% do mercado
 - 16% do mercado
- 
-
- 
-
- 

Segurança

- Smartphone não estão alheios a problemas de segurança e vírus
- “Estamos ainda em uma época de adaptação, onde as pessoas não são tão desconfiadas como perante um computador”
 - Nume, da empresa CTS, é um hardware de encriptação, cuja incumbência é impossibilitar a intercepção das comunicações procedentes de telefones smartphones, tablets e computadores.

Desenvolvimento de aplicativos para dispositivos móveis usando Android



Plataforma Android

- Ambiente de software para dispositivos móveis
 - Não é uma plataforma de hardware
- Inclui
 - Sistema operacional baseado em kernel de Linux
 - UI rica
 - Aplicações de usuário
 - Bibliotecas de código
 - Frameworks de aplicação
 - Suporte a multimídia
 - Funcionalidades de telefonia
 - etc

Introdução a Android

- Plataforma de software
- Esforço principal da Google
 - Colaboração com a Open Handset Alliance
 - Quase 50 organizações
 - Comprometida com uma plataforma móvel melhor e mais aberta
- Considerada apenas uma novidade por muitos a princípio
 - Tornou-se um divisor de águas no mercado móvel
 - Baseado em Linux e na máquina virtual Dalvik

Plataforma

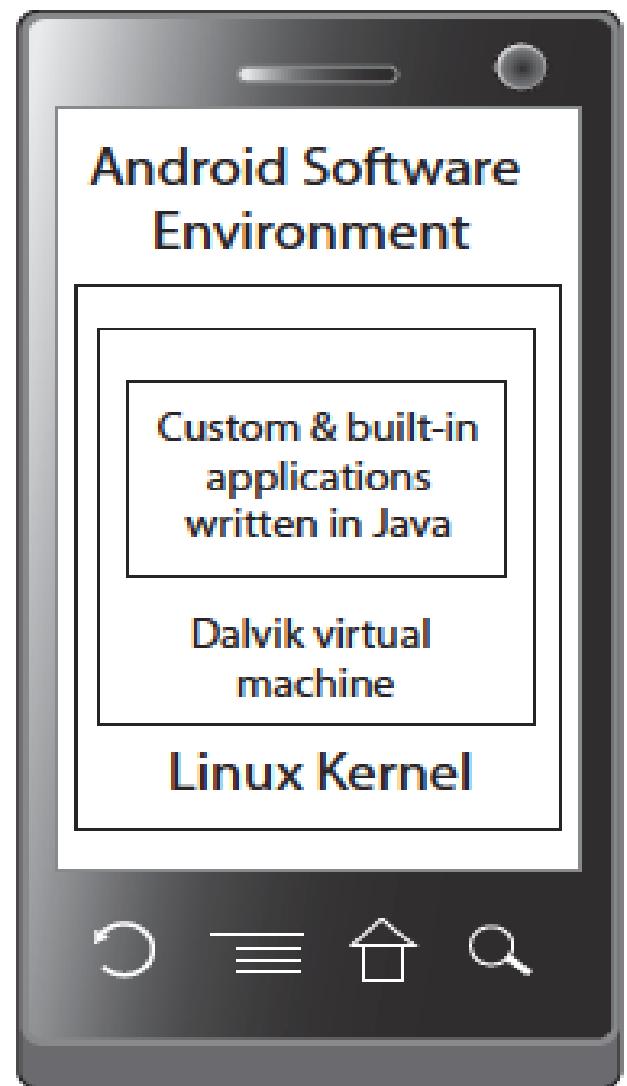
- Android é um Sistema Operacional que utiliza em seu núcleo a versão 2.6 do kernel do Linux. É um sistema leve porém poderoso;
- Atualmente o Android equipa Celulares, Tablets PC, SmartPhones, etc;
- Utiliza a Máquina Virtual Dalvik (desenvolvida pela Google).

Características

- Desenvolvido para tirar o maior proveito do que os dispositivos podem oferecer
- Construído em Java
- Suporta um Subset da linguagem Java

Plataforma Android

- Componentes do SO estão em C ou C++
- Aplicações são desenvolvidas em Java
 - Na sua imensa maioria
 - Aplicações do sistema estão em Java também
 - Nenhuma diferença entre aplicações do sistema e aplicações desenvolvidas usando o SDK
- Open-source



Características

- Dalvik – Máquina virtual
- Possui um conjunto de bibliotecas C/C++
- SQLite – Banco de dados relacional
- Redes e comunidades disponíveis pelo Google
- Android market

Tecnologias Utilizadas

- JDK J2SE
- Eclipse 3.6.2
- Android SDK

Mercado

- Operadoras de telefonia
 - AT & T, Verizon
 - Serviços de dados
 - Mercado premium e com larga margem de lucro
- Android X “feature phones”
 - Mercado desejado para Android
 - Feature phones
 - Consumidor “só quero falar e mandar mensagens”
 - Celulares gratuitos em operadoras
 - Atualmente com câmeras, GPS, etc

Mercado

- Android X Android
 - Mercado open-source é faca de dois gumes
 - Modelo de licenciamento
 - Concorrência acirrada entre fabricantes
 - Fenômeno “mee too”
 - Alterações em interface e código
 - Aparentemente em funcionalidades

Camadas

- Kernel linux
 - Drivers de hardware
- Bibliotecas
 - WebKit, SQLite, gráficos 2D, 3D, mídia, etc
- Managers
 - Telefonia, Activities (views), janelas, recursos, etc
- Android runtime
 - Core java
 - Dalvik VM

User applications: Contacts, phone, browser, etc.

Application managers: windows, content, activities, telephony, location, notifications, etc.

Android runtime: Java via Dalvik VM

Libraries: graphics, media, database, communications, browser engine, etc.

Linux kernel, including device drivers

Hardware device with specific capabilities such as GPS, camera, Bluetooth, etc.

Camadas

- Linux kernel
 - Base de um kernel linux e JVM otimizada
 - Duas tecnologias são cruciais para ambiente
- Porque linux?
 - Capacidades e poder de programação
 - Open-source
 - Cobre mudança rápida de produtos
 - Plataforma comprovadamente estável
 - Confiabilidade é mais importante que desempenho em mercado móvel
 - Voz X características de programação
 - Camada de abstração de hardware

Camadas

- Dalvik VM
 - Criado por Dan Bornstein;
 - Nome de uma cidade em Iceland
 - Diminui o tamanho do arquivo executável;
 - Formato .dex (Dalvik Executable);
 - Não é baseado em byte code Java mas sobre instâncias de arquivos .dex;
 - Neste caso para instanciar uma classe Java é necessário convertê-la para .dex;

Camadas

- Dalvik VM
 - Não é JME
 - Não é Java (??)
 - Aplicações são escritas em Java, são compiladas para bytecode Java, mas traduzidas para uma representação similar, porém diferente, chamada “arquivos dex”
 - Logicamente equivalentes a bytecodes java, mas permite a aplicações Android rodar em sua própria VM (livre de direitos)
 - Do ponto de vista do programador, Android é Java
 - Mas o runtime não é estritamente uma Java VM
 - Código legado (bibliotecas, por exemplo) são quase compatíveis em fonte (nearly source compatible)
 - Tente recompilar em Android, mas não espere resultados corretos todo o tempo

Introdução a Android

- Palavras chave para o programador:
 - Activity
 - Intent *
 - Resource *
 - Service
 - ContentProvider *
 - BroadcastReceiver

Programação Android - Intents

- Intents
 - O que se quer fazer
 - Buscar um registro
 - Lançar um website
 - Facilitam navegação de forma inovadora
 - É uma declaração de necessidade
 - Desejo de ação ou serviço
- IntentFilter
 - Declaração de capacidade e interesse em oferecer assistência para uma necessidade
 - Pode ser genérico ou específico em relação a qual Intent se oferece o serviço



Intents

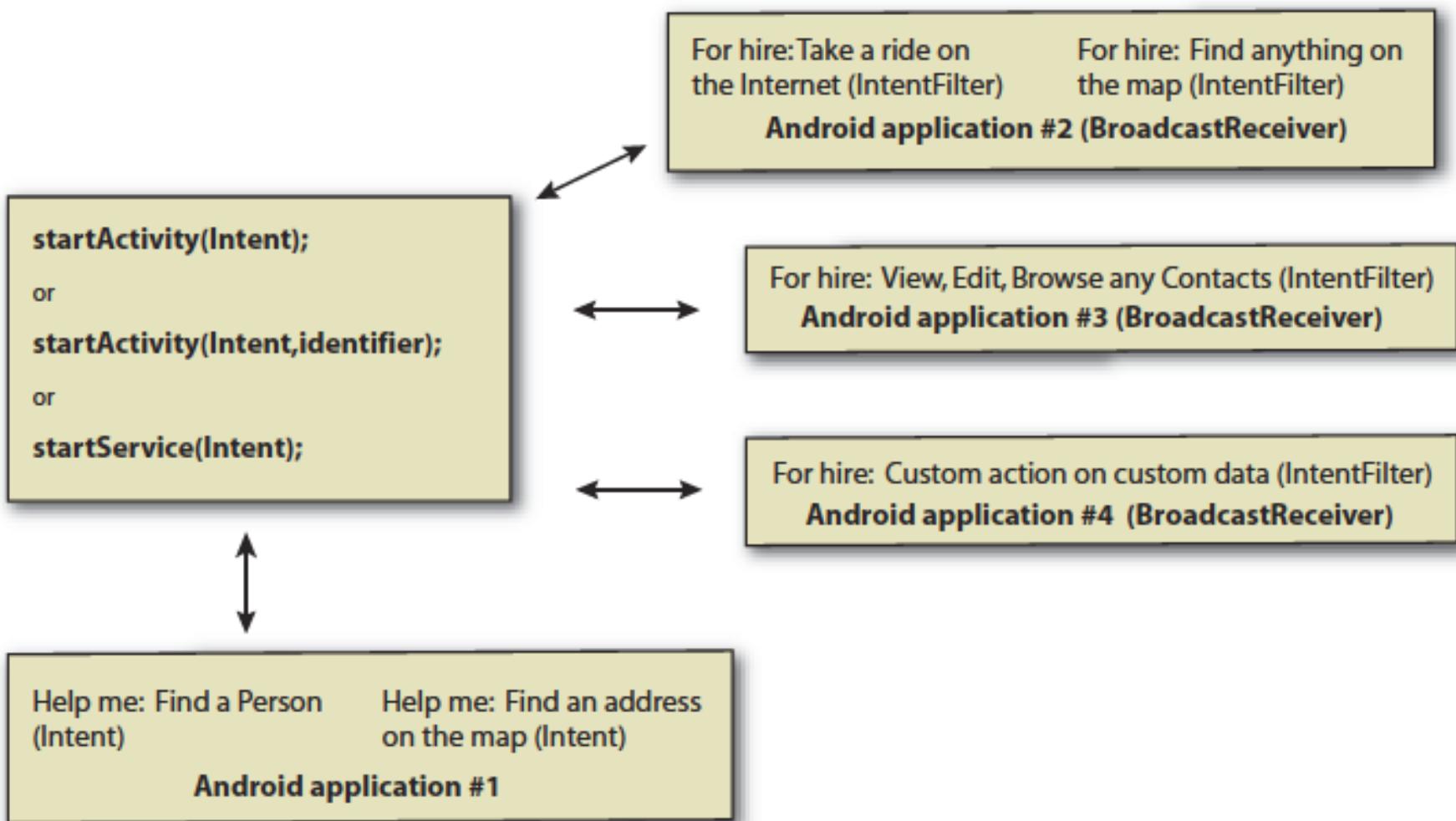
- O atributo de ação de um Intent é tipicamente um verbo
 - VIEW, PICK, EDIT
- Várias ações de Intents existentes são definidos como membros da classe Intent
 - Programadores podem criar novas ações
- Para ver uma informação, o Intent seguinte é usado:
 - `android.content.Intent.ACTION_VIEW`
- O componente de dados de um Intent é expresso na forma de uma URI e pode ser qualquer informação

Intents - URIs comuns

- Contact lookup
 - content://contacts/people
- Map lookup/search
 - Gei:0,0?q=23+Route+206+Stanhope+NJ
- Website em navegador
 - <http://www.google.com>



Intents



Intents - Exemplo

- Selecionar um registro em contatos
 - Intent pickIntent = new Intent(Intent.ACTION_PICK,
Uri.parse("content://contacts/people"));
 - startActivityForResult(pickIntent);
- Este Intent será avaliado e passado para o handler mais apropriado
 - Provavelmente uma Activity do sistema chamada com.google.android.phone.Dialer
 - Mas pode ser qualquer outra Activity que receba esse tipo de Intent

Intents

- IntentFilter
 - Define a relação entre o Intent e a aplicação
- Quando Intent é lançado, o sistema avalia as Activities, Services e BroadcastReceivers disponíveis
 - Envia o Intent para o receptor mais apropriado
 - Ver registro de contatos e ligar, enviar um SMS, buscar um endereço, ver um vídeo, ver um website, etc
- IntentFilters são definidos geralmente no arquivo de manifesto
 - AndroidManifest.xml

Componentes Android - Activity

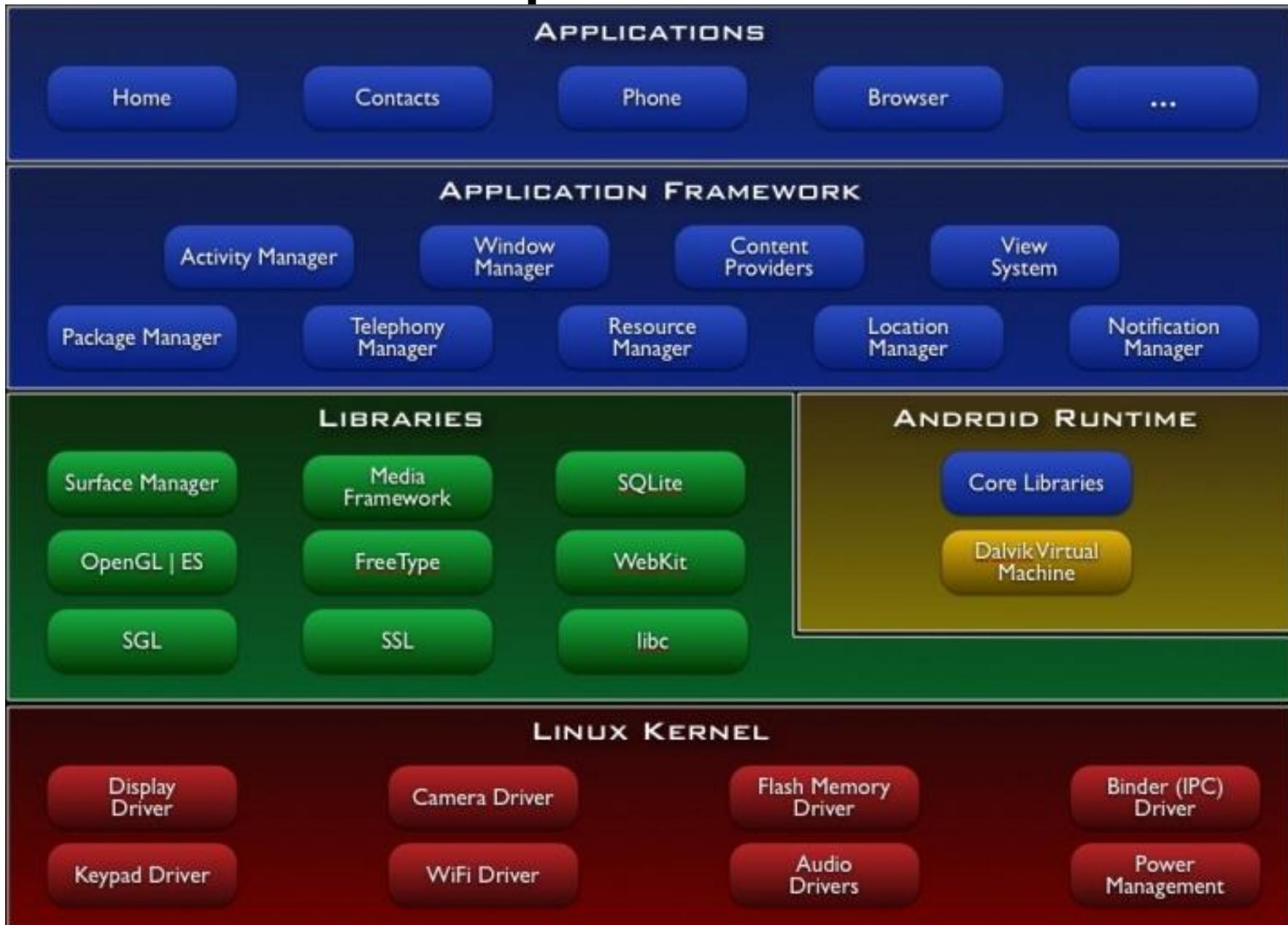
- É uma tela visível, relação quase de um para um
 - Controller do MVC
- Se uma aplicação precisar de uma interface, terá ao menos uma Activity
 - Aplicações Android geralmente contém mais de uma
- Cada Activity responde a eventos do usuário e do sistema
 - Emprega uma ou mais Views para apresentar os elementos de UI para o usuário
- Classe herdada pelo programador

ARQUITETURA





Arquitetura





Aplicações

APPLICATIONS

Home

Contacts

Phone

Browser

...

- Cliente de e-mail
- SMS
- Calendário
- Mapas
- Navegador
- Contatos e outros.



Framework



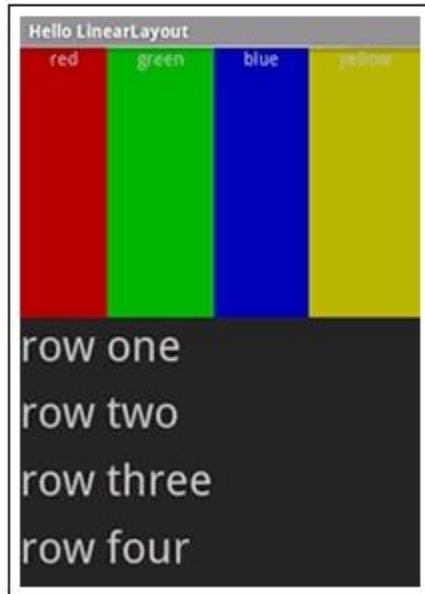
- Fornece uma plataforma de desenvolvimento aberta, o que oferece aos desenvolvedores a capacidade de construir aplicações ricas e inovadoras
- Desenvolvedores tem pleno acesso às APIs
- A arquitetura do aplicativo é projetado para simplificar a reutilização de componentes

Possui um conjunto de serviços e sistemas

- Conjunto de **Views** que podem ser usados para construir uma aplicação

Layouts

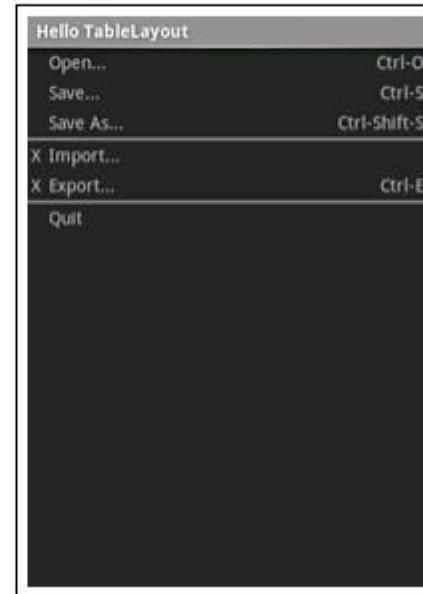
Linear Layout



Relative Layout



Table Layout



Grid View





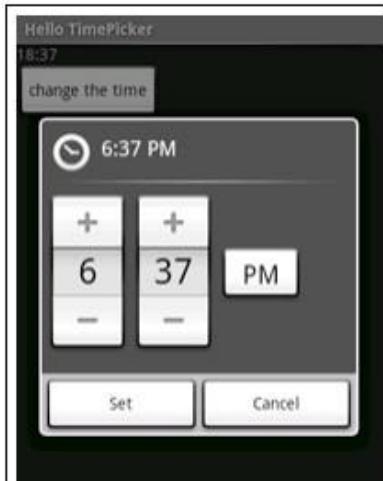
Views

Widgets & Other Views

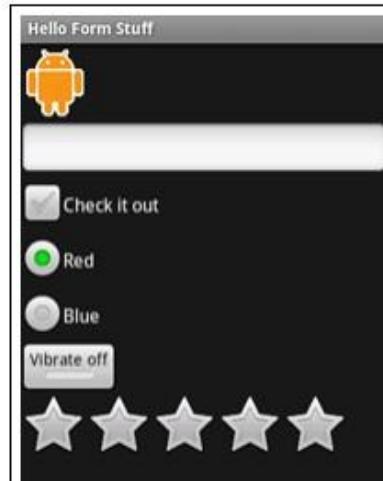
Date Picker



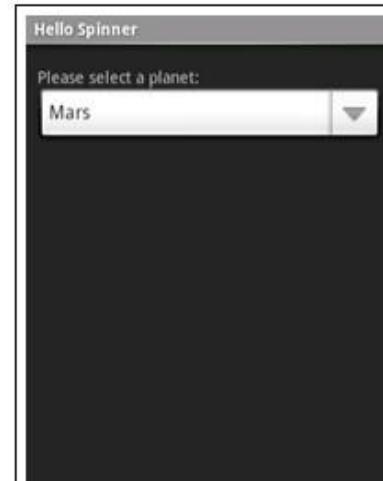
Time Picker



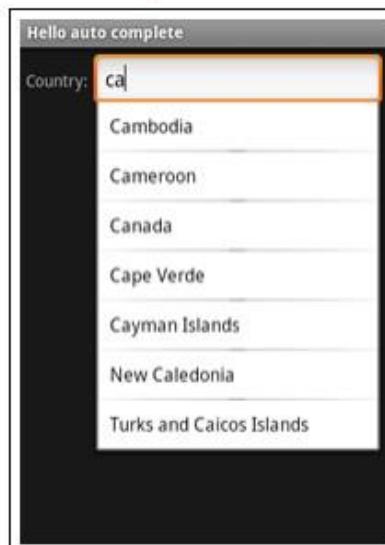
Form Stuff



Spinner



Auto Complete



Gallery



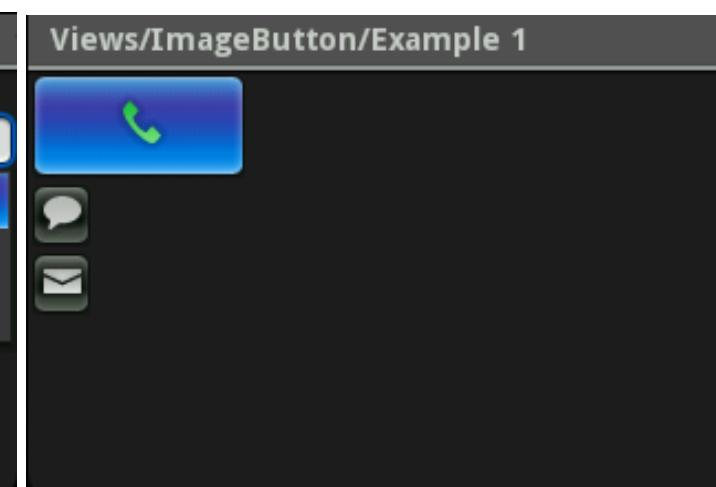
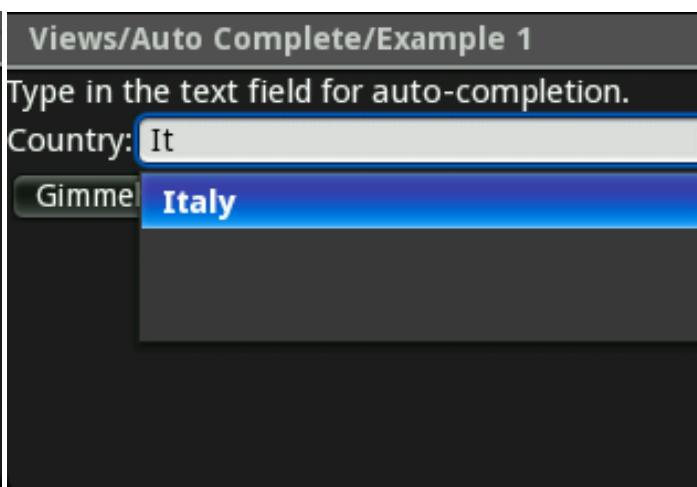
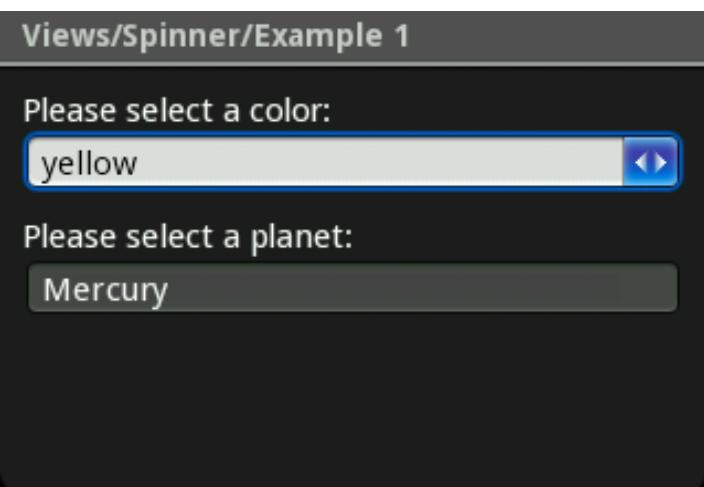
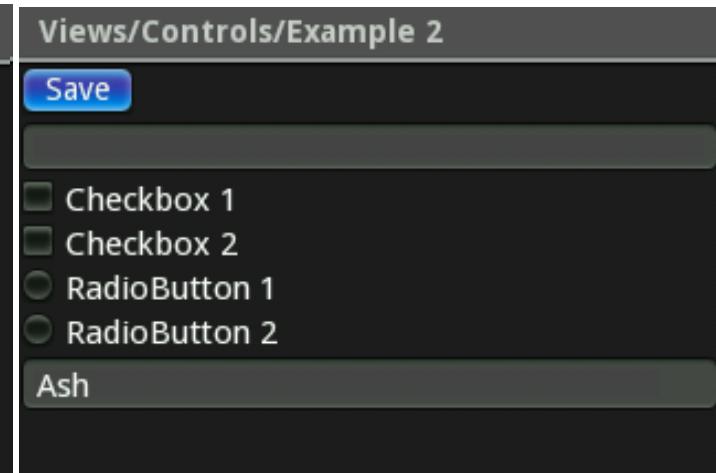
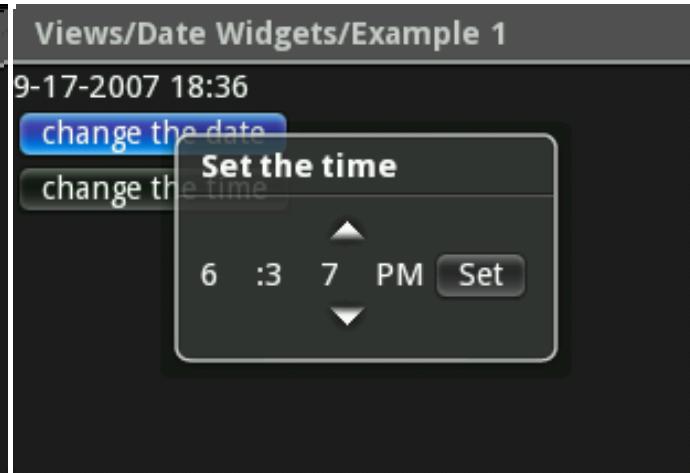
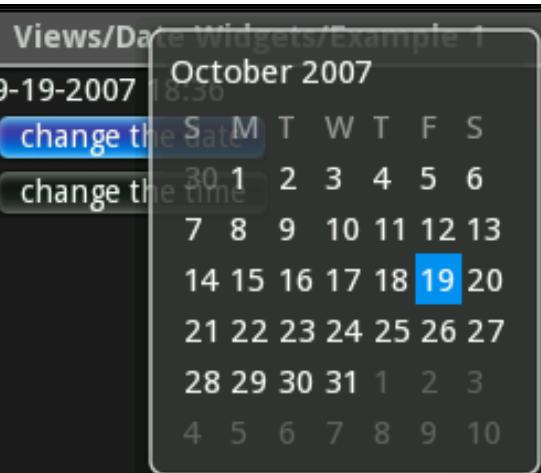
Google Map View



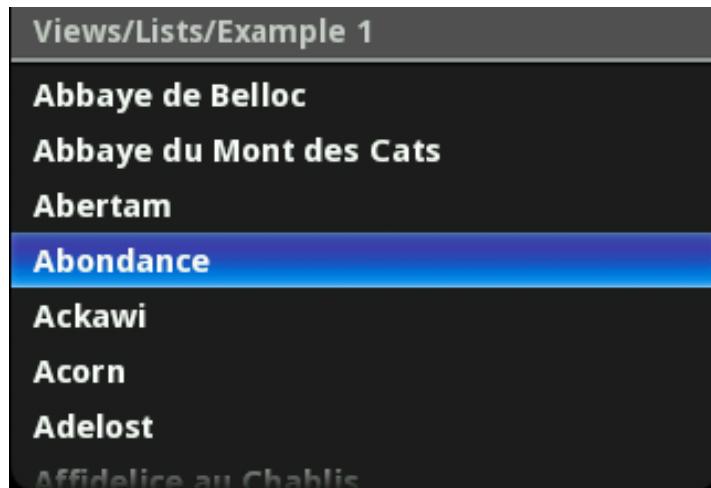
Web View



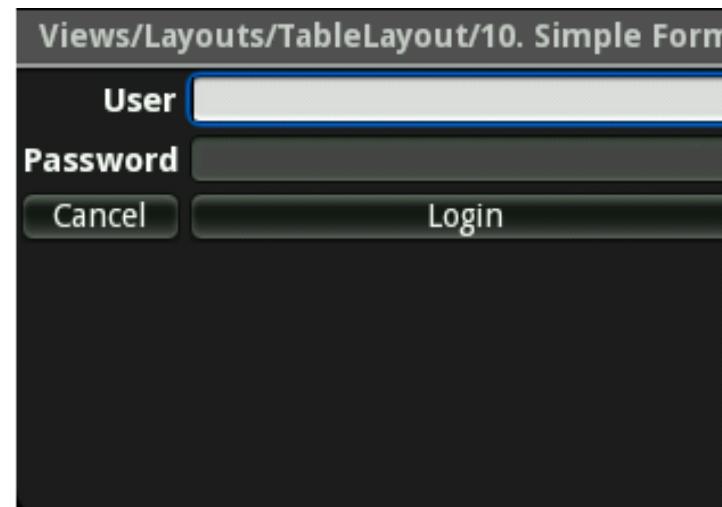
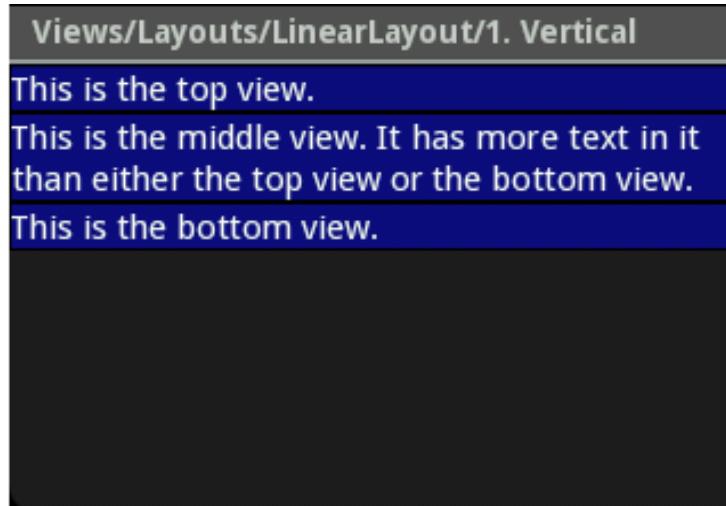
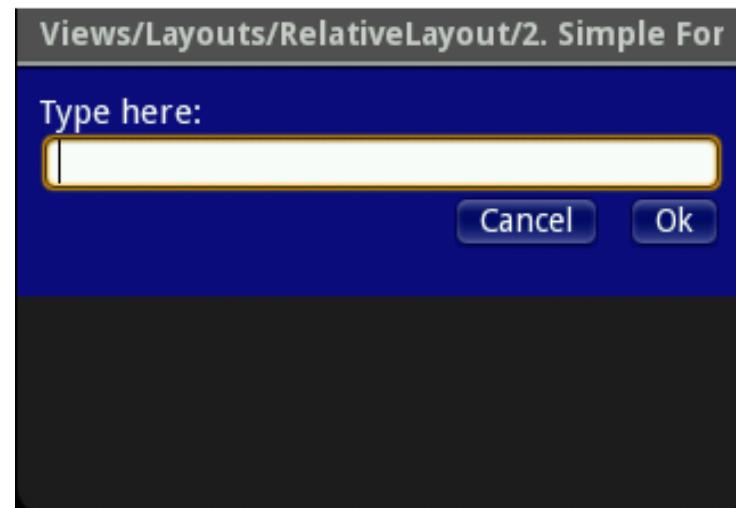
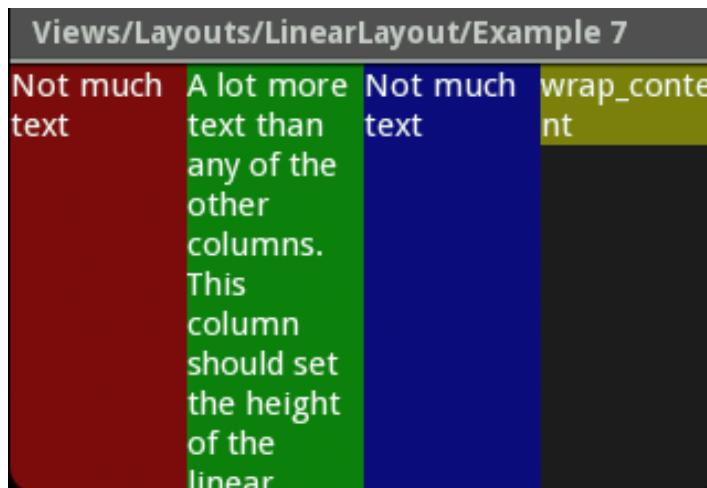
Views (1/3)



Views (2/3)

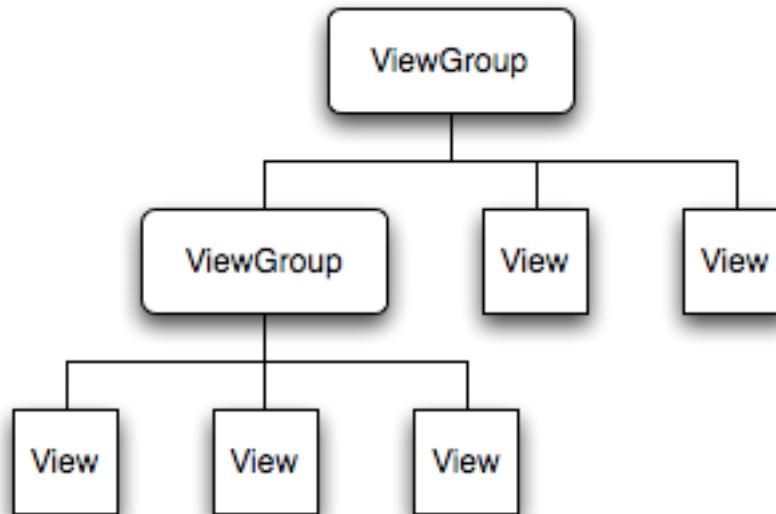


Views (3/3)



Interface

- Usa-se objetos View e ViewGroup.
- ViewGroup - estrutura de dados cujas propriedades do layout são guardadas para uma área retangular específica da tela.
- Para vincular a árvore view à tela para ser renderizada, sua Activity precisa chamar o método `setContentView(View view)`.



Componentes Android -

Activity

- Parte do package android.app
- Mostra elementos de UI, implementados por Views, e definidos em arquivos de layout XML
- Sair de uma Activity para outra é feito pelo método startActivity() ou startActivityForResult()
- Componente visível da aplicação
- Mais empregado na programação Android



Componentes Android -

Activity

- Uma atividade exibe uma UI e pode receber eventos do sistema e iniciados pelo usuário.
- Atividades podem interagir com outras atividades,
- Uma atividade pode empregar uma ou mais views para apresentar um UI
- Para se criar uma atividade estende-se a classe Activity.



Componentes Android - Activity

```
package com.msi.manning.chapter1;
import android.app.Activity;
import android.os.Bundle;
public class Activity1 extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState){
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
}
```

Componentes Android - Service

- Aplicações com ciclo de vida longo ou tarefas repetitivas
 - Sincronizações, ações cíclicas, etc
 - Lançados de forma periódica, quando necessários ou disparados por um alarme do sistema
 - Iniciados por startService(Intent)
 - Método da classe Context
- Herda da classe Service
- Não possui interface
 - Utiliza logs do sistema ou mensagens



Componentes Android - Service

- Tarefa que é rodada ao fundo, sem bloquear a interação do usuário.
 - uma aplicação que toca música enquanto o usuário faz outras atividades.
- Para criar um Service, estende-se a classe Service.



Componentes Android - Service

```
package com.msi.manning.chapter1;
import android.app.Service;
import android.content.Intent;
import android.os.IBinder;
import android.util.Log;
public class Service1 extends Service implements
Runnable {
    public static final String tag = "service1";
    private int counter = 0;
    @Override
    protected void onCreate() {
        super.onCreate();
        Thread aThread = new Thread (this);
        aThread.start();
    }
}
```



Componentes Android - Service

```
public void run() {  
    while (true) {  
        try {  
            Log.i(tag,"service1 firing : # " + counter++);  
            Thread.sleep(10000);  
        } catch(Exception ee) {  
            Log.e(tag,ee.getMessage());  
        }  
    }  
}  
@Override  
public IBinder onBind(Intent intent) {  
    return null;  
}  
}
```

Componentes Android - BroadcastReceiver

- Aplicações que recebem e respondem a eventos globais
 - Chamada telefônica, SMS, etc
 - Se registra para receber um determinado Intent
- Aplicação implementa um <receiver> no arquivo de manifesto ou se registra em runtime
- Não tem interface, nem devem executar por muito tempo
 - Se o ciclo da aplicação é maior, é sugerido iniciar um Service



Componentes Android - BroadcastReceiver

```
package com.msi.manning.unlockingandroid;
import android.content.Context;
import android.content.Intent;
import android.util.Log;
import android.content.BroadcastReceiver
public class MySMSMailBox extends BroadcastReceiver {
    public static final String tag = "MySMSMailBox";
    @Override
    public void onReceive(Context context, Intent intent) {
        Log.i(tag,"onReceive");
        if (intent.getAction().equals
            ("android.provider.Telephony.SMS_RECEIVED")) {
            Log.i(tag,"Found our Event!");
        }
    }
}
```

Componentes Android - ContentProvider

- Expõe dados para outras aplicações
 - Se uma Activity precisa de dados de outra aplicação, usa seu ContentProvider
- Implementa métodos padronizados para acessar dados
 - Acesso controlado (R, W, R/W)
- Maneira padronizada de troca de dados entre aplicações
 - Aplicações podem compartilhar arquivos ou bancos de dados SQLite, mas não é recomendado

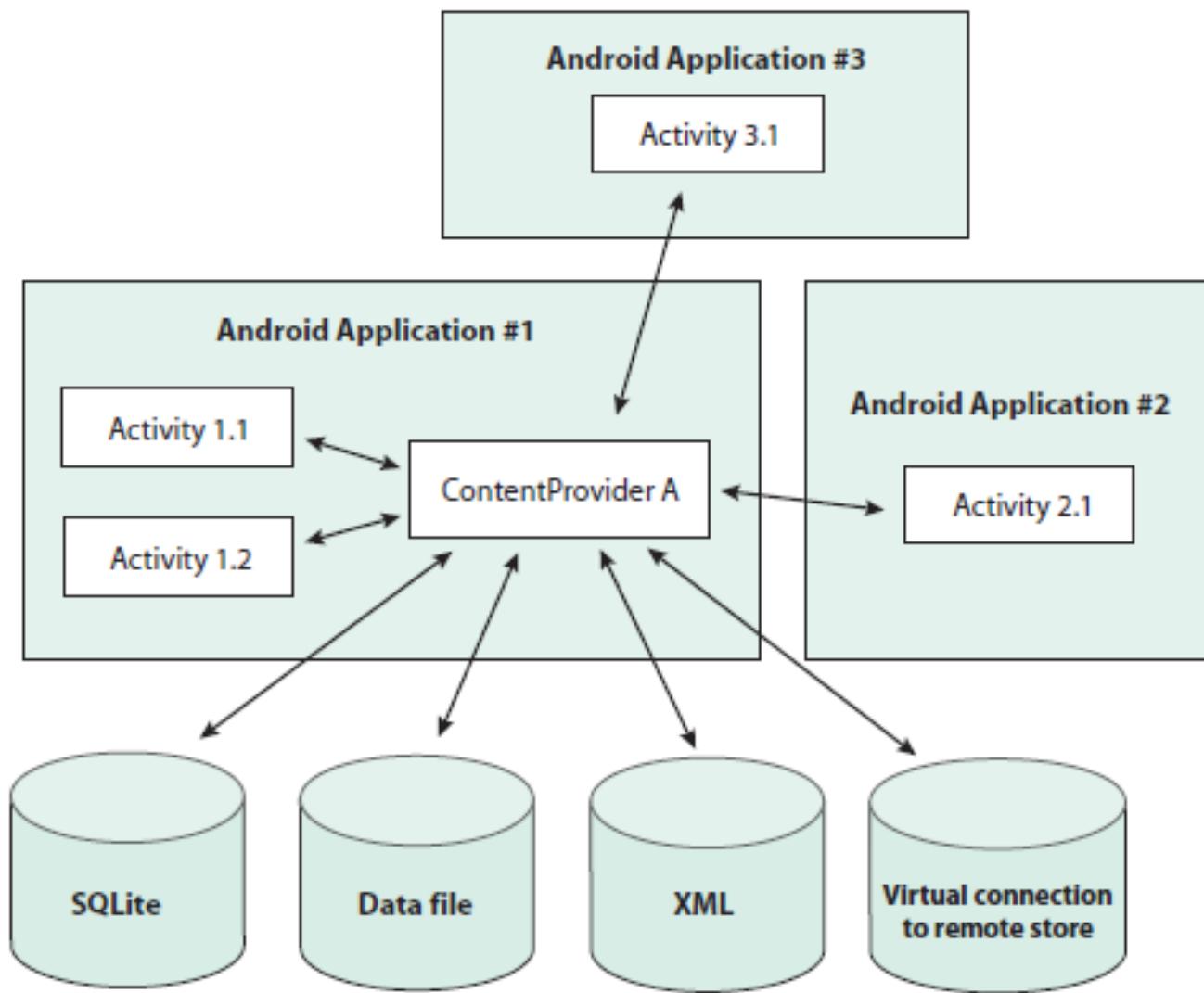


Componentes Android -

Content Provider

- gerencia os dados da aplicação. Você pode salvar dados em um sistema de arquivos, num banco SQLite, na internet ou em qualquer outra forma de persistência que sua aplicação conseguir acessar.
- Por meio do Content Provider outras aplicações podem acessar ou até modificar dados da sua aplicação (se tiverem a devida permissão).

Uso do ContentProvider



AndroidManifest.xml

- Para o sistema identificar sua aplicação, ele precisa ler o arquivo AndroidManifest.xml.
- Além de declarar os componentes da aplicação o arquivo manifesto faz também outras coisas:
 - Identifica permissões do usuário que a aplicação necessita. Ex: conexão com internet;
 - Declara componentes de hardware e software que a aplicação usa. Ex: câmera e serviço de bluetooth;
 - Entre outros.

Arquivo de manifesto

- `AndroidManifest.xml`
 - Informações sobre execução de uma aplicação Android
 - Contém ao menos uma entrada `Activity`, `Service`, `BroadcastReceiver` ou `ContentProvider`
 - Deployment descriptor

Exemplo de Manifesto

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.msi.manning.unlockingandroid">
    <application android:icon="@drawable/icon">
        <activity android:name=".Activity1" android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
    </application>
</manifest>
```



Layout

- O método mais comum para se fazer o layout é usar XML. Cada elemento do XML é um objeto View.

```
<?xml version="1.0" encoding="utf-8"?>
    <LinearLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:orientation="vertical" >
        <TextView android:id="@+id/text"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a TextView" />
        <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button" />
    </LinearLayout>
```



Exemplo: Hello World

```
package com.example.helloandroid;

import android.app.Activity;
import android.os.Bundle;
import android.widget.TextView;

public class HelloAndroid extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        TextView tv = new TextView(this);
        tv.setText("Hello, Android");
        setContentView(tv);
    }
}
```



Exemplo: Hello World

```
package com.example.helloworld;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;

public class Main extends Activity {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }

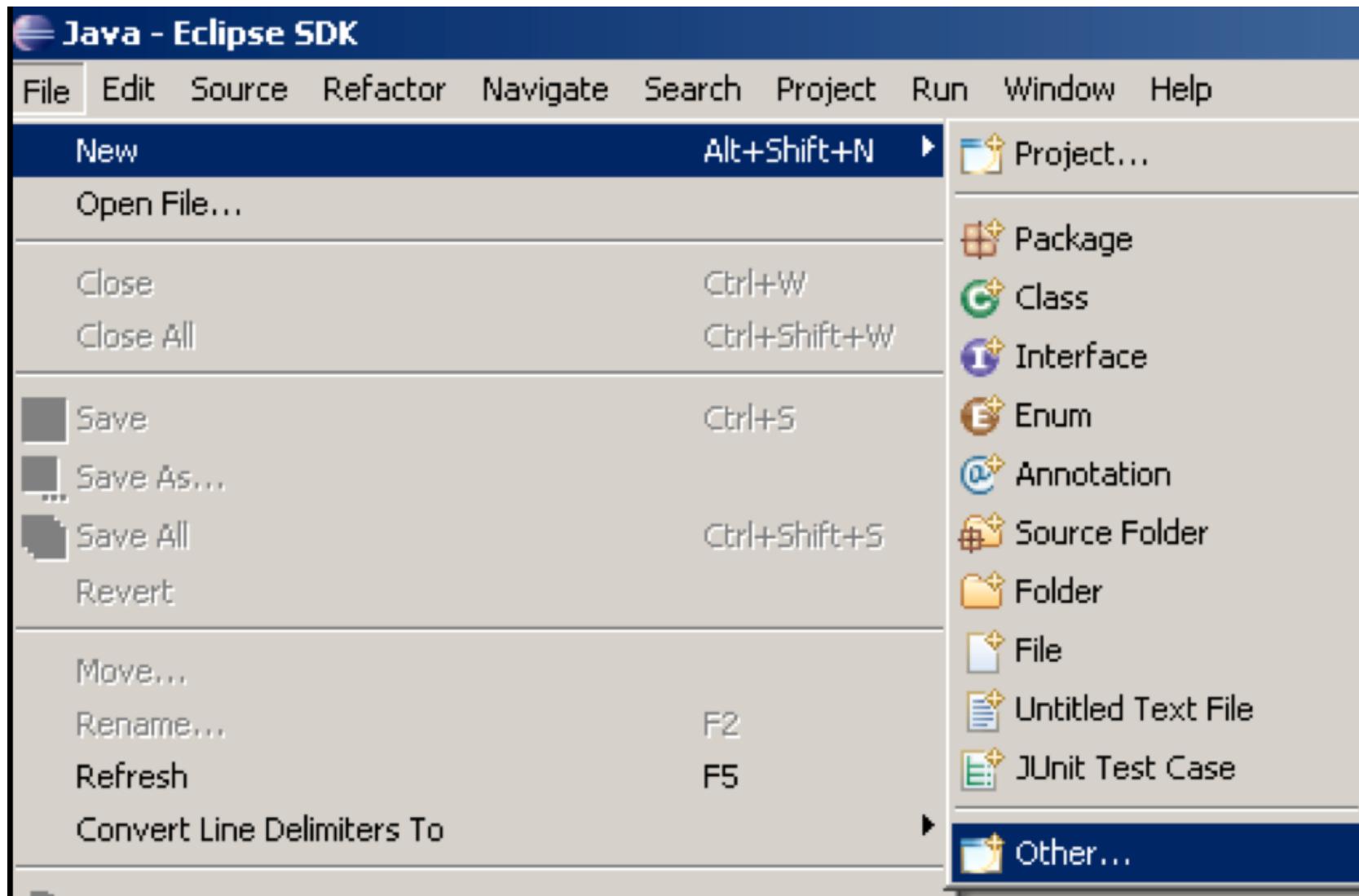
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```

CRIANDO O PRIMEIRO PROJETO...

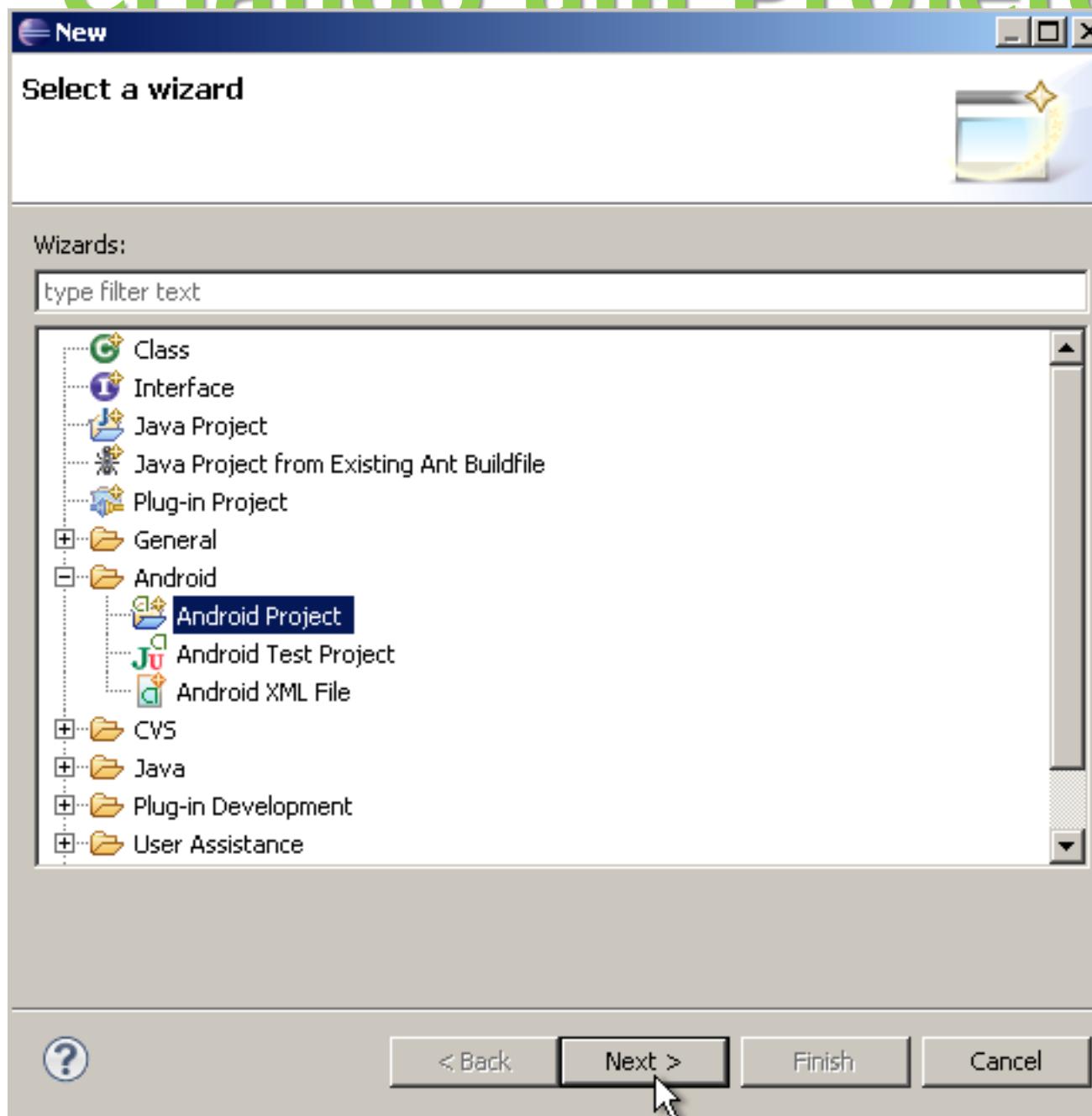
HELLO WORLD!



CRIANDO UM PROJETO



Criando um Projeto



Criando um Projeto

New Android App

New Android Application

 The prefix 'com.example.' is meant as a placeholder and should not be used

Application Name: Project Name: Package Name:

Build SDK: Choose... Minimum Required SDK:

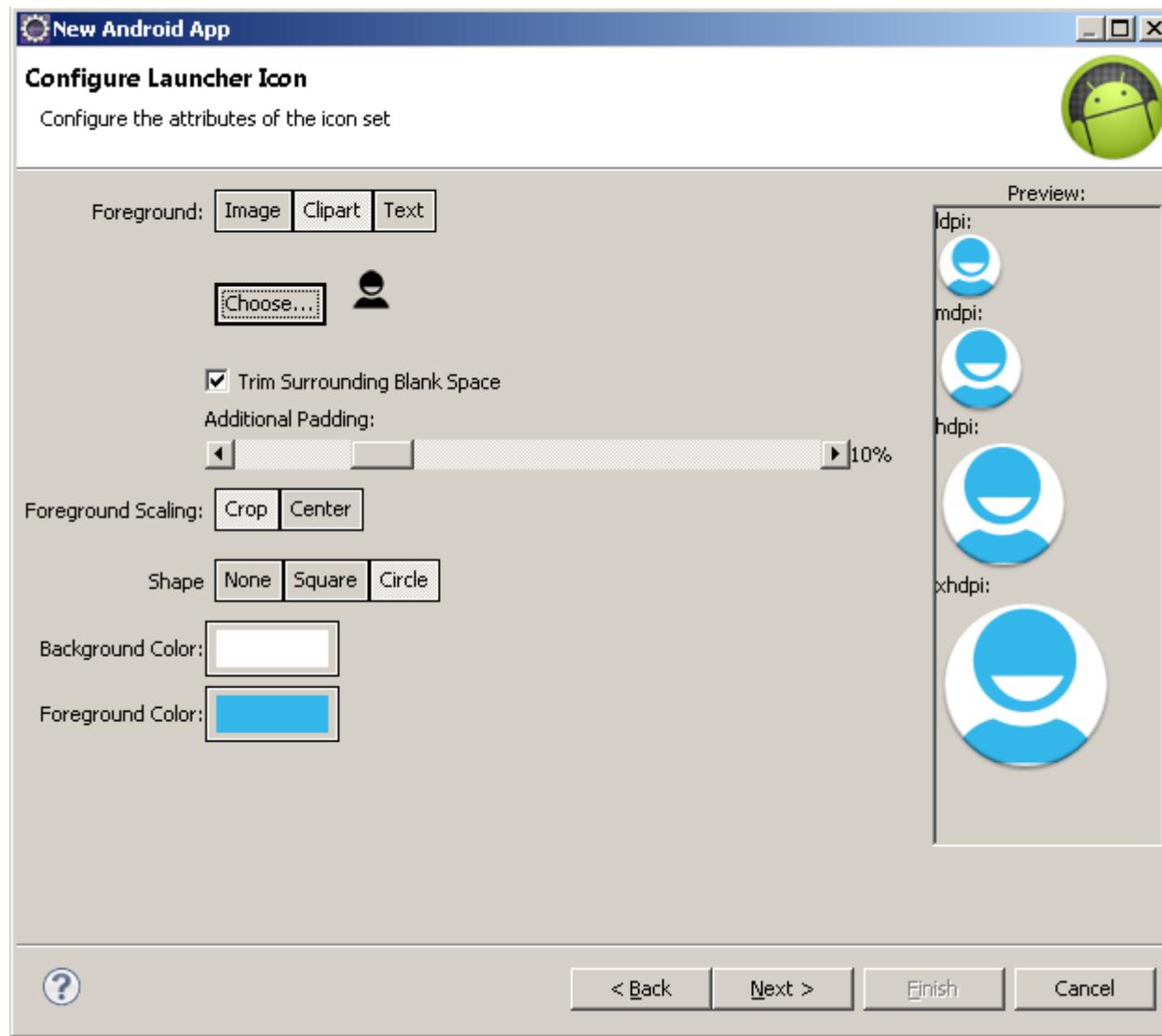
Create custom launcher icon
 Mark this project as a library
 Create Project in Workspace

Location: Browse...

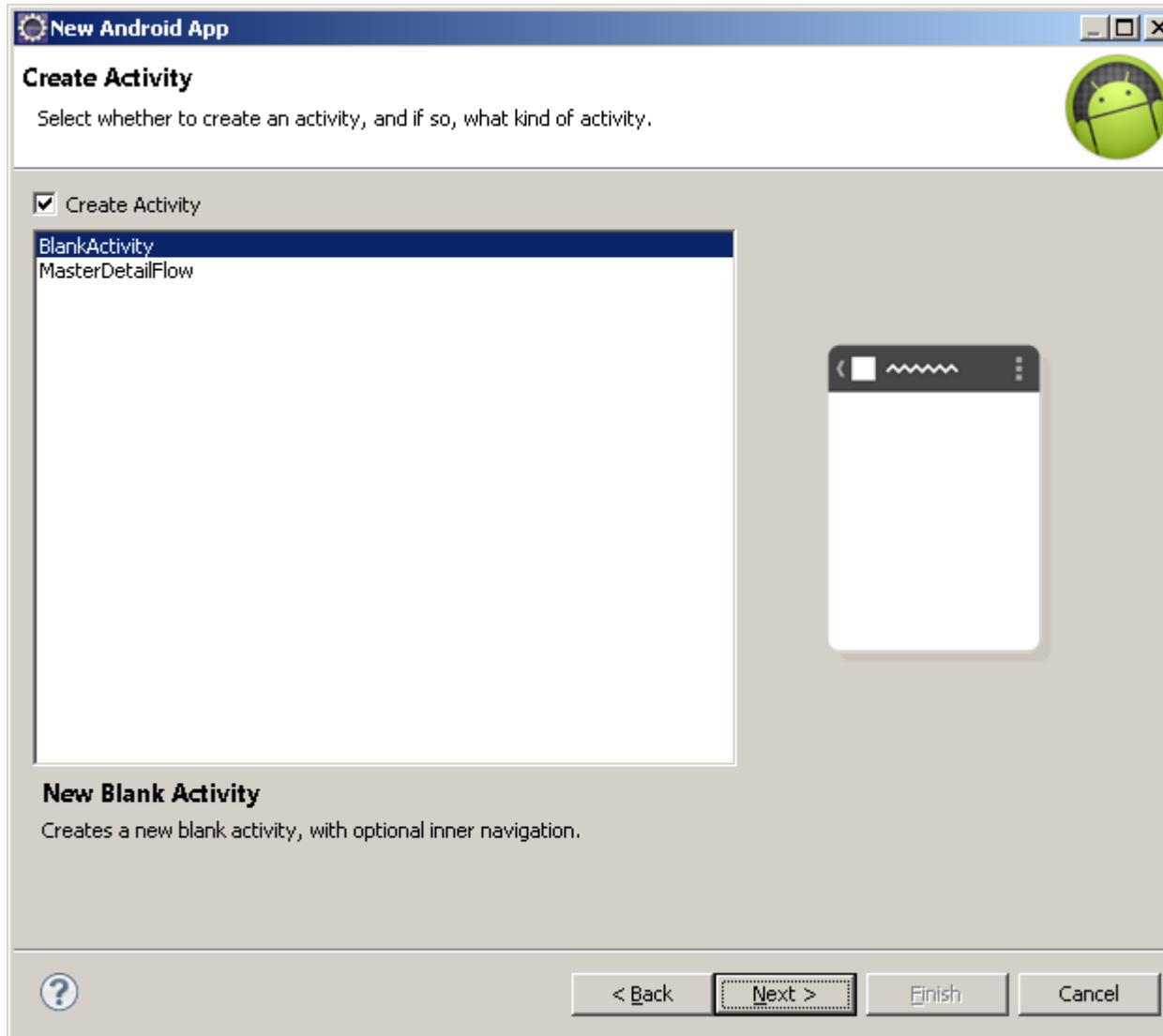
 Choose a target API to compile your code against. This is typically the most recent version, or the first version that supports all the APIs you want to directly access

< Back [Next >](#) [Finish](#) [Cancel](#)

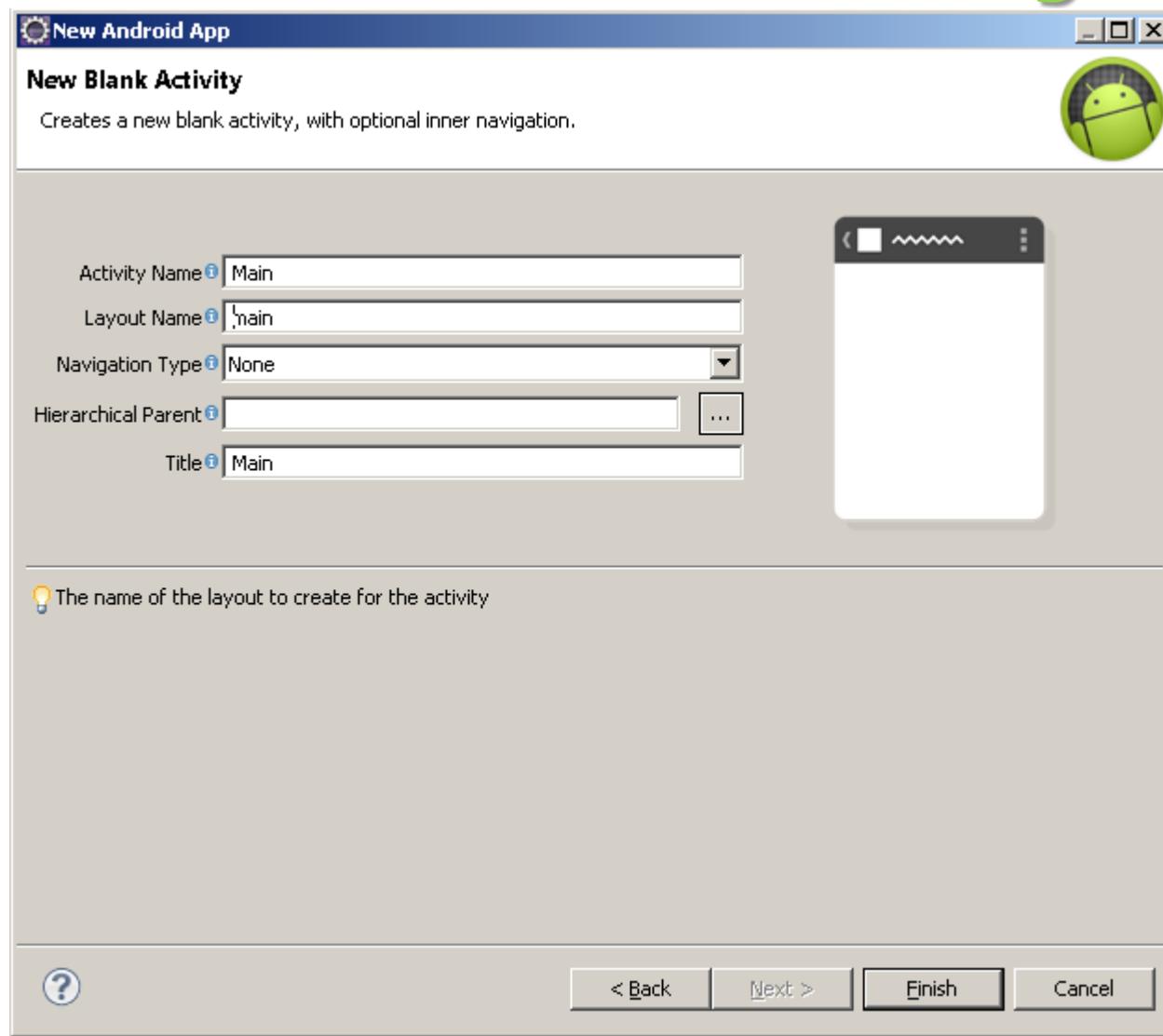
Criando um Projeto



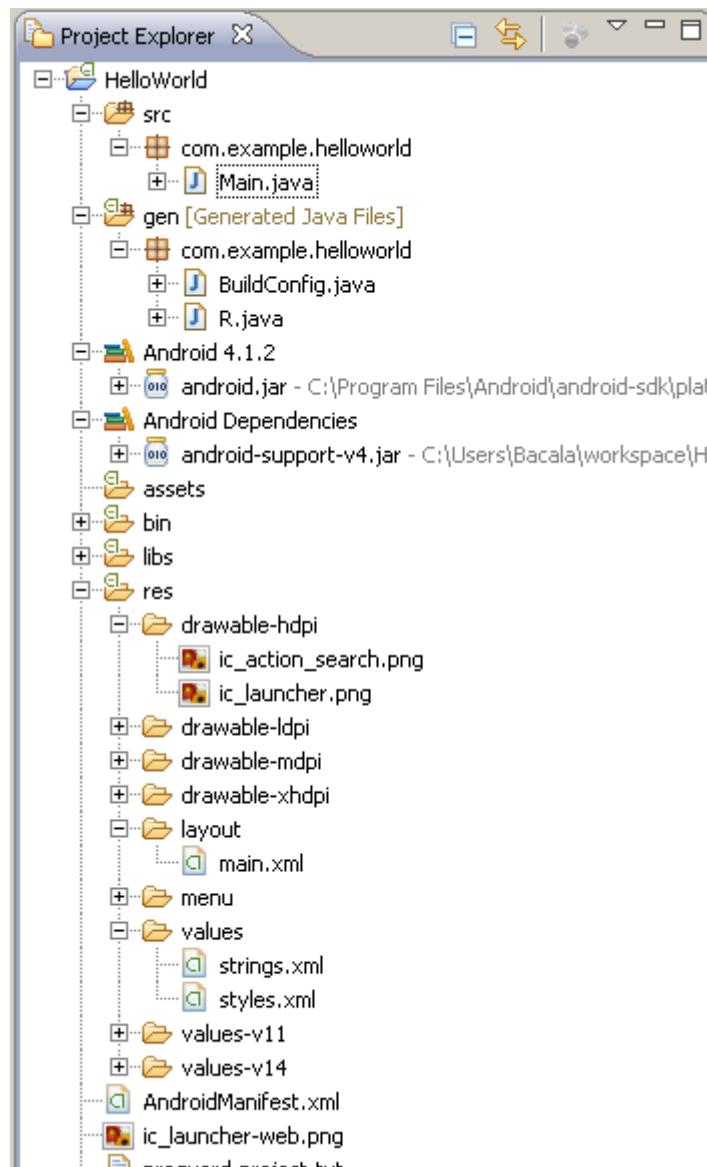
Criando um Projeto



Criando um Projeto



Entendendo a Estrutura do Projeto





Project Explorer X main.xml Main.java HelloWorld Manifest X

>HelloWorld

src

com.example.helloworld

Main.java

gen [Generated Java Files]

com.example.helloworld

BuildConfig.java

R.java

Android 4.1.2

android.jar - C:\Program Files\Android\android-sdk\platforms\android-16\android.jar

Android Dependencies

android-support-v4.jar - C:\Users\Bacala\workspace\HelloWorld\libs\android-support-v4.jar

assets

bin

libs

res

AndroidManifest.xml

ic_launcher-web.png

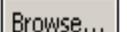
proguard-project.txt

project.properties

Android Manifest

Manifest General Attributes

Defines general information about the AndroidManifest.xml

Package com.example.helloworld 

Version code 1 

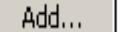
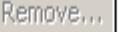
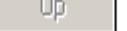
Version name 1.0 

Shared user id 

Shared user label 

Install location 

Manifest Extras 

Uses Sdk    

Manifest Application Permissions Instrumentation AndroidManifest.xml

A large blue arrow points from the Project Explorer to the manifest editor.

A large blue arrow points from the AndroidManifest.xml file in the Project Explorer to the manifest editor.



Project Explorer main.xml Main.java HelloWorld Manifest

HelloWorld

- src
 - com.example.helloworld
 - Main.java
- gen [Generated Java Files]
 - com.example.helloworld
 - BuildConfig.java
 - R.java
- Android 4.1.2
 - android.jar - C:\Program Files\Android\android-sdk\plat
- Android Dependencies
 - android-support-v4.jar - C:\Users\Bacala\workspace\H
- assets
- bin
- libs
- res
 - AndroidManifest.xml
 - ic_launcher-web.png
 - proguard-project.txt
 - project.properties

main.xml

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.helloworld"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="15" />

    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".Main"
            android:label="@string/title_activity_main" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Manifest Application Permissions Instrumentation AndroidManifest.xml

Project Explorer X main.xml Main.java HelloWorld Manifest X

Application Attributes
Defines the attributes specific to the application

Name	Browse...	Hardware accelerated
Theme	@style/AppTheme	Browse... Manage space activity
Label	@string/app_name	Browse... Allow clear user data
Icon	@drawable/ic_launcher	Browse... Test only
Logo	Browse...	Backup agent
Description	Browse...	Allow backup
Permission		Kill after restore
Process	Browse...	Restore needs application
Task affinity	Browse...	Restore any version
Allow task reparenting		Never encrypt
Has code		Large heap
Persistent		Can't save state
Enabled		Ui options
Debuggable		Supports rtl
Vm safe mode		

Application Nodes S P A G R M U Az

- Main (Activity)
 - Intent Filter
 - android.intent.action.MAIN (Action)
 - android.intent.category.LAUNCHER (Category)

Add... Remove... Up Down

AndroidManifest.xml

Manifest Application Permissions Instrumentation

The screenshot shows the Eclipse IDE interface for an Android project named 'HelloWorld'. The Project Explorer on the left lists files like 'Main.java', 'BuildConfig.java', and 'R.java'. The main window displays the 'HelloWorld Manifest' tab, which contains two sections: 'Application Attributes' and 'Application Nodes'. The 'Application Attributes' section lists various application-specific settings such as theme, icon, and permissions. The 'Application Nodes' section shows the manifest structure with an 'Activity' node and its associated intent filters. Blue arrows highlight the relationship between these two sections.



Project Explorer main.xml Main.java

The screenshot shows the Android Studio interface. On the left, the Project Explorer displays the project structure for "HelloWorld". It includes the "src" folder containing the package "com.example.helloworld" with a file "Main.java". The "gen" folder contains generated files for the same package. The "Android 4.1.2" folder lists "android.jar" from the SDK. "Android Dependencies" include "android-support-v4.jar". The "assets", "bin", "libs", and "res" folders are also shown. The "res" folder is expanded, revealing "drawable-hdpi" and "drawable-ldpi" subfolders with icons like "ic_launcher.png" and "ic_action_search.png", along with "layout", "menu", and "values" folders. On the right, the code editor shows the "Main.java" file. The code defines a class "Main" that extends "Activity". It overrides the "onCreate" and "onCreateOptionsMenu" methods. The "onCreate" method calls the superclass's "onCreate" and sets the content view to "R.layout.main". The "onCreateOptionsMenu" method inflates the menu from "R.menu.main" and returns true. The code editor has syntax highlighting for Java and XML.

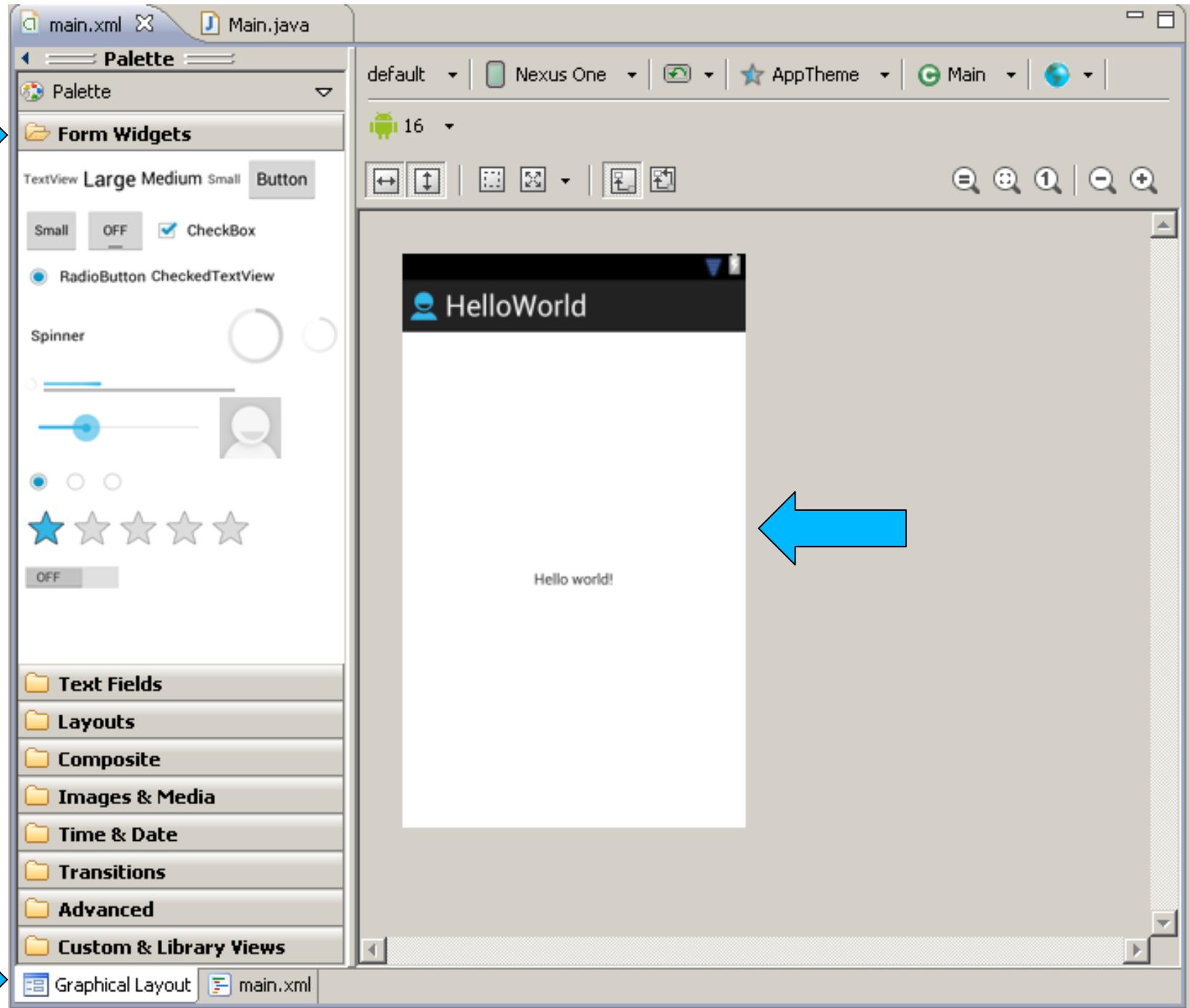
```
package com.example.helloworld;

import android.os.Bundle;

public class Main extends Activity {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```



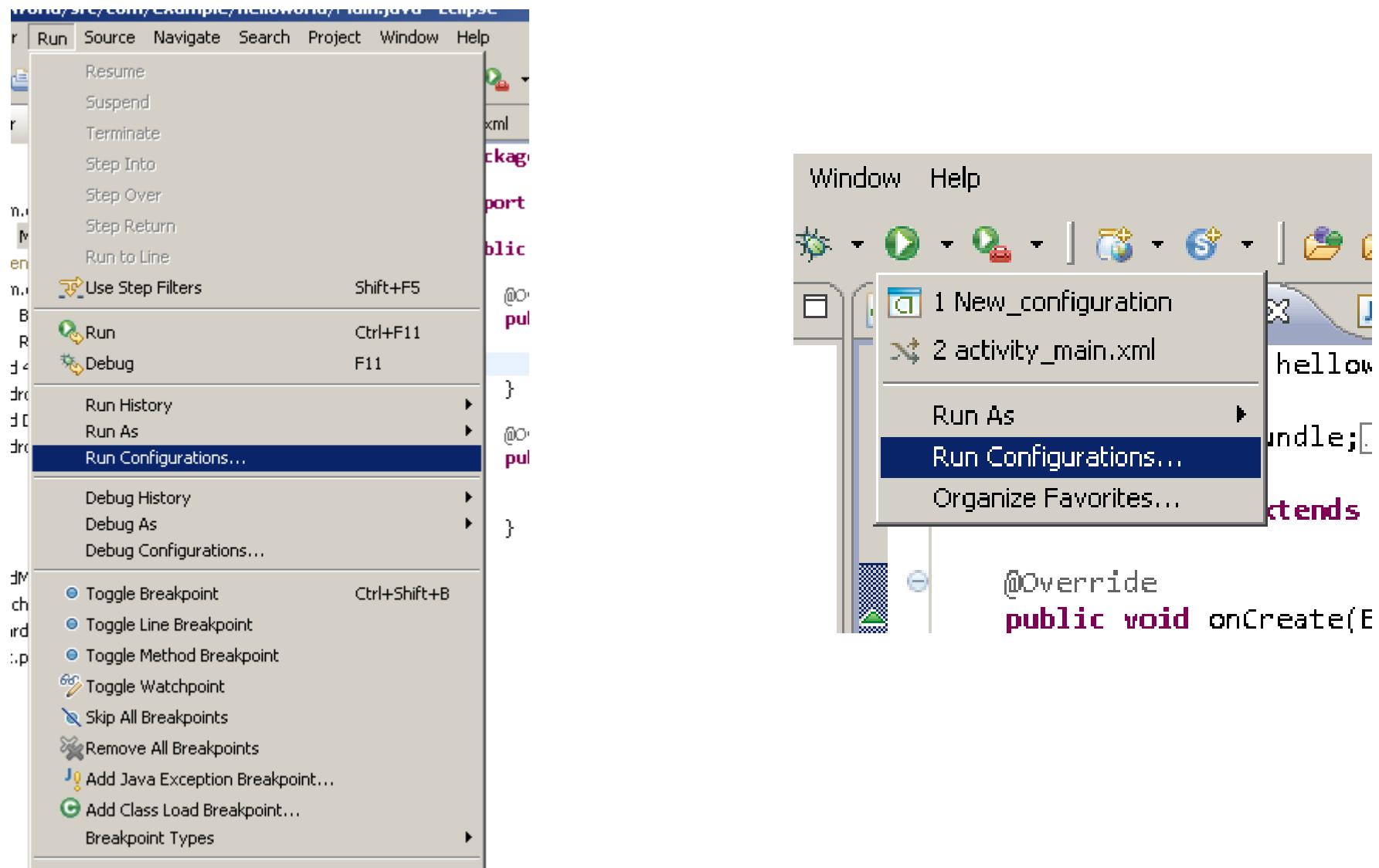


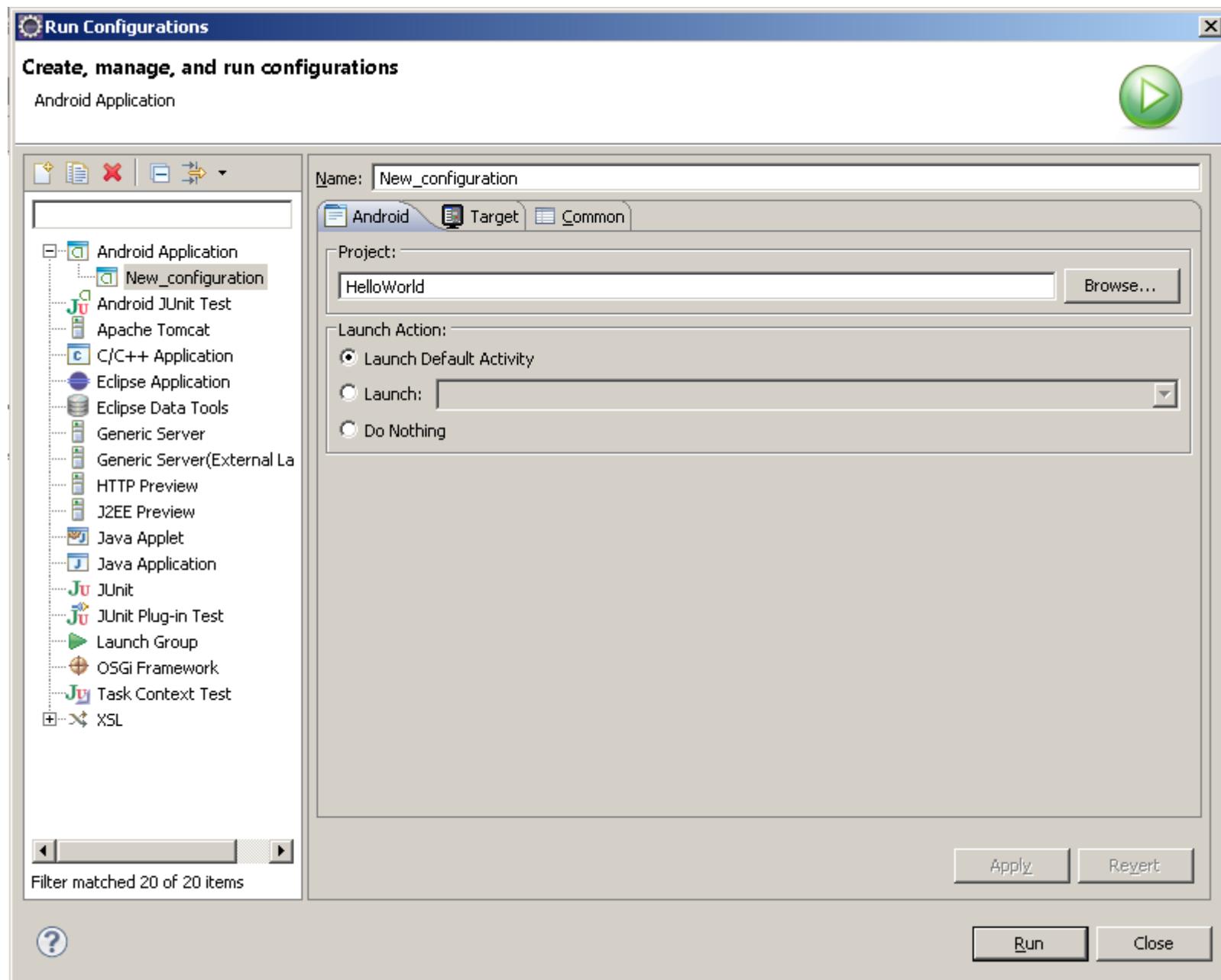
```
main.xml Main.java R.java X
+/* AUTO-GENERATED FILE. DO NOT MODIFY.

package com.example.helloworld;

public final class R {
    public static final class attr {
    }
    public static final class drawable {
        public static final int ic_action_search=0x7f020000;
        public static final int ic_launcher=0x7f020001;
    }
    public static final class id {
        public static final int menu_settings=0x7f070000;
    }
    public static final class layout {
        public static final int main=0x7f030000;
    }
    public static final class menu {
        public static final int main=0x7f060000;
    }
    public static final class string {
        public static final int app_name=0x7f040000;
        public static final int hello_world=0x7f040001;
        public static final int menu_settings=0x7f040002;
        public static final int title_activity_main=0x7f040003;
    }
    public static final class style {
        public static final int AppTheme=0x7f050000;
    }
}
```

Executando





Run Configurations

Create, manage, and run configurations

Android Application



Target

Name: New_configuration

Deployment Target Selection Mode

- Always prompt to pick device
- Launch on all compatible devices/AVD's
 - Active devices and AVD's
- Automatically pick compatible device: Always uses preferred AVD if set below, launches on compatible device/AVD's

Select a preferred Android Virtual Device for deployment:

AVD Name	Target Name	Platform	API Level	CPU/ABI
android4	Android 4.1.2	4.1.2	16	ARM (armeabi-v7a)

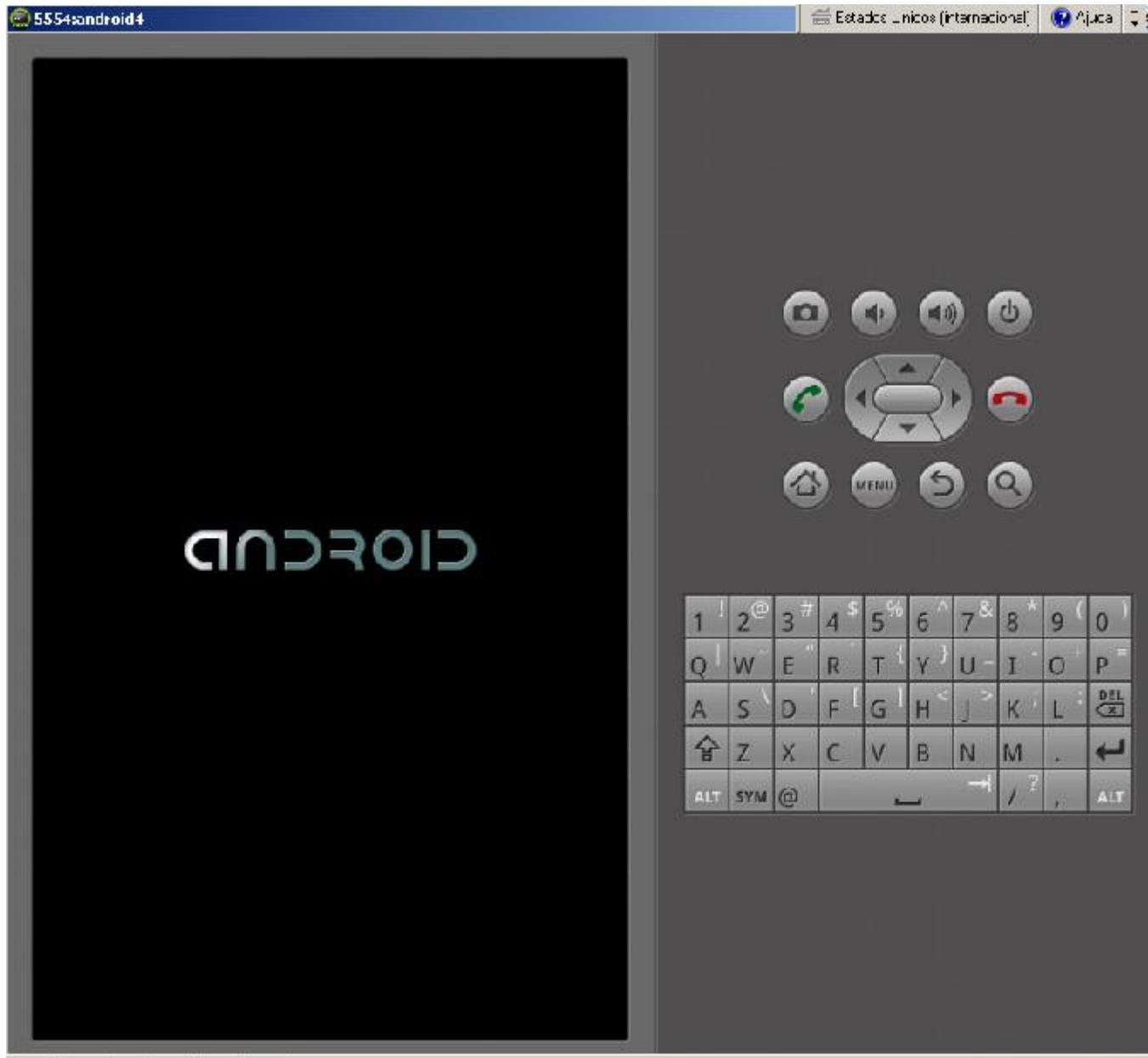
Emulator launch parameters:

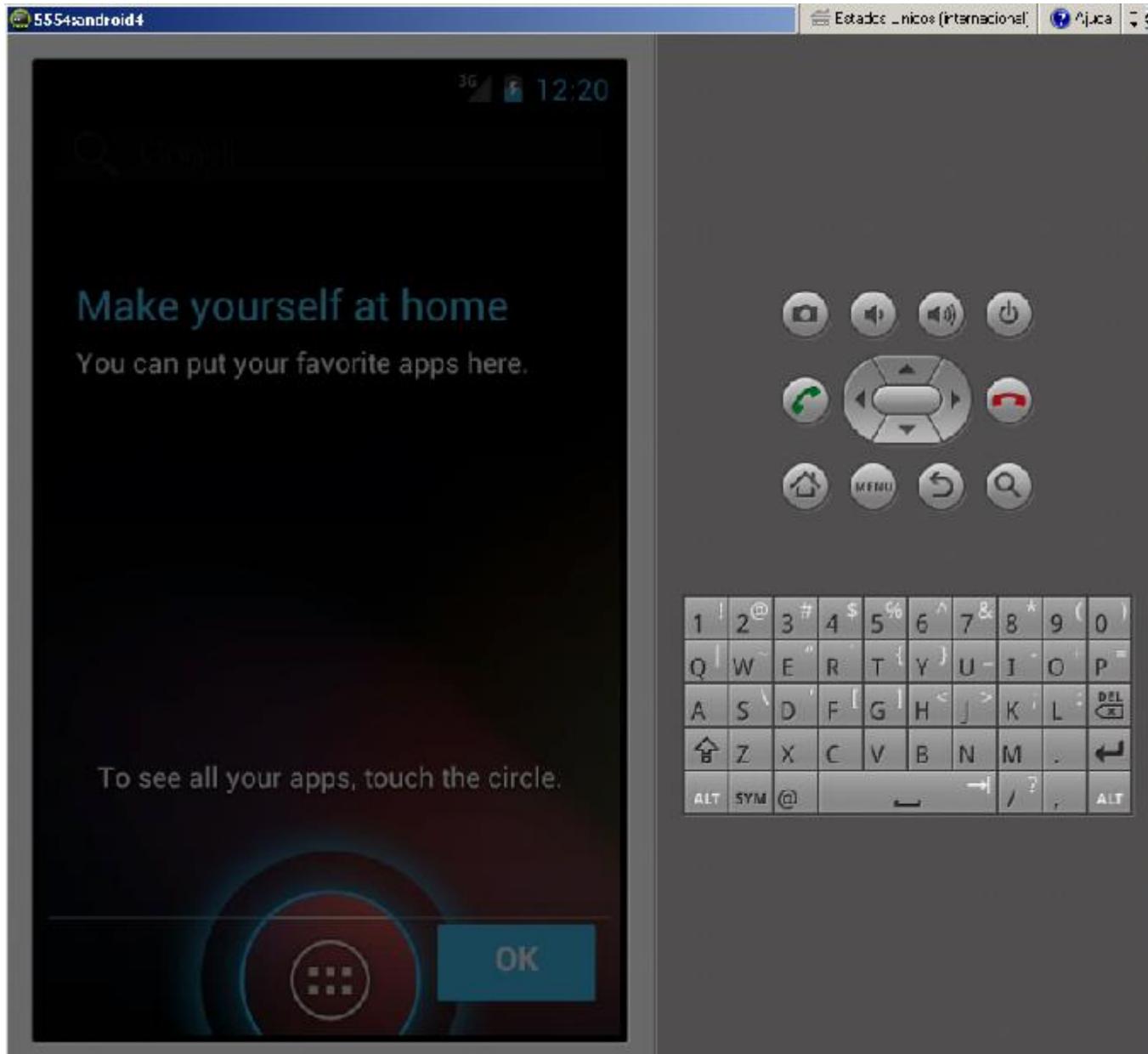
If no compatible and active devices or AVD's are found, then an AVD might be launched. Provide options for the AVD.

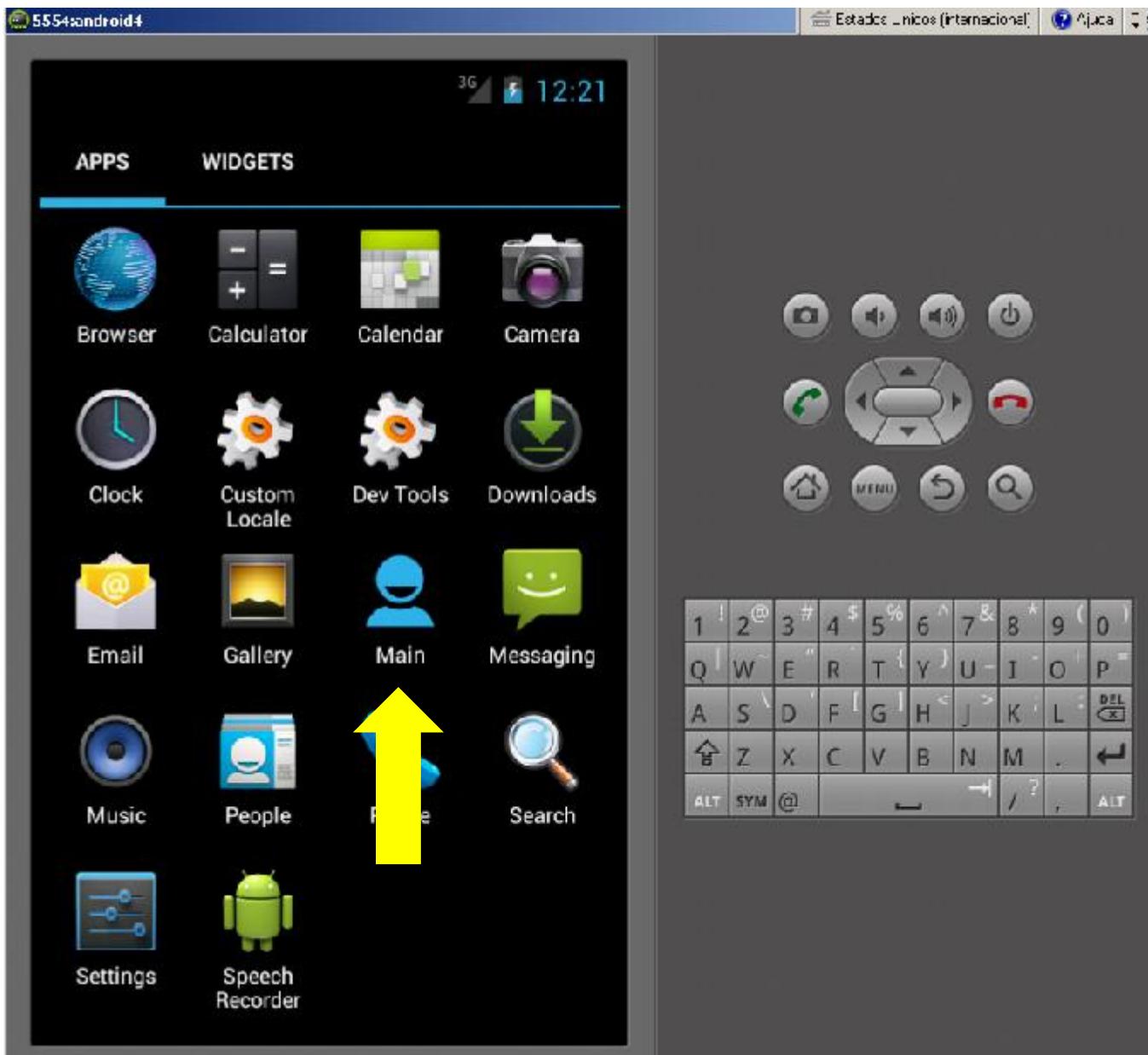
Network Speed:

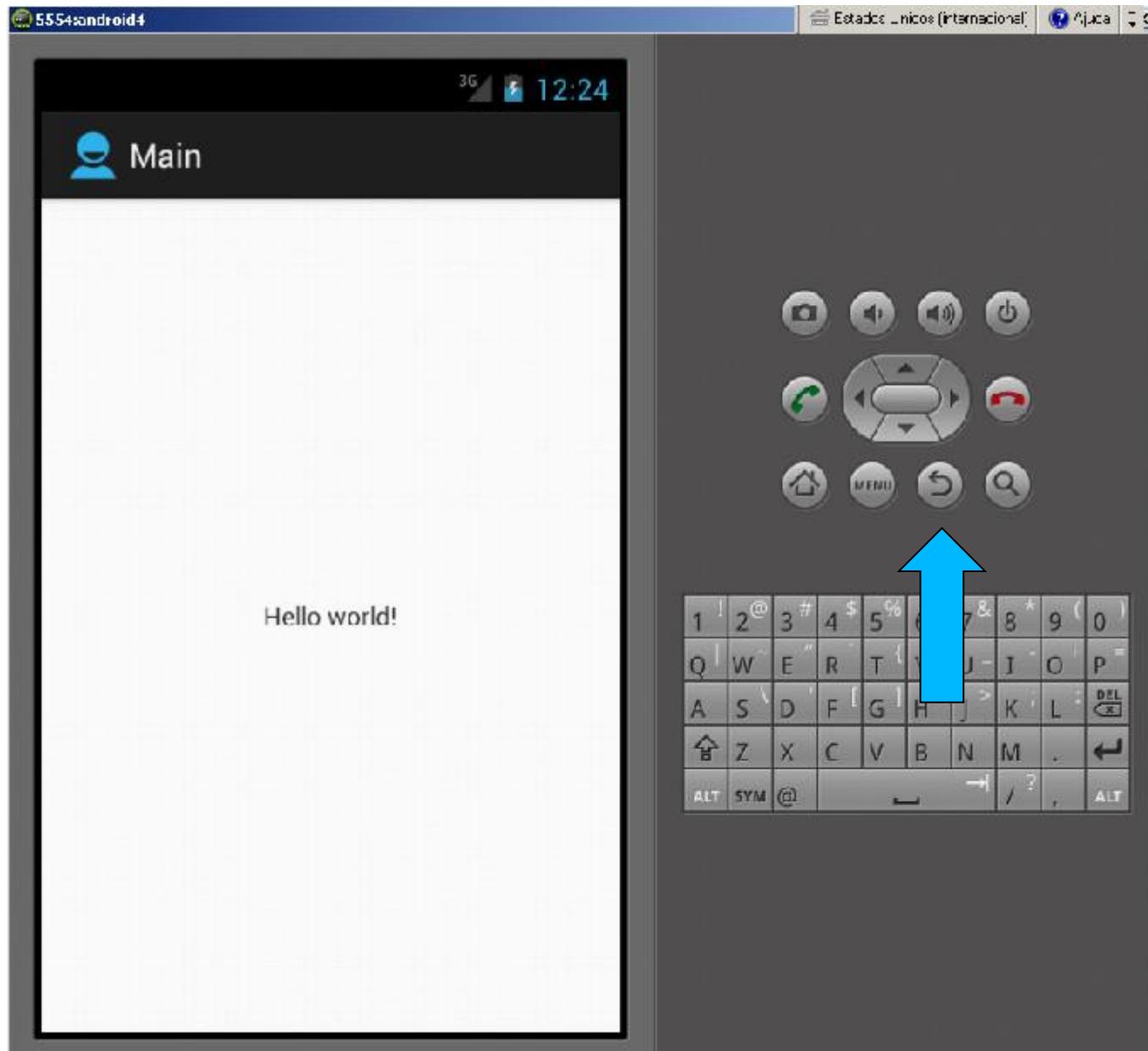
Filter matched 20 of 20 items







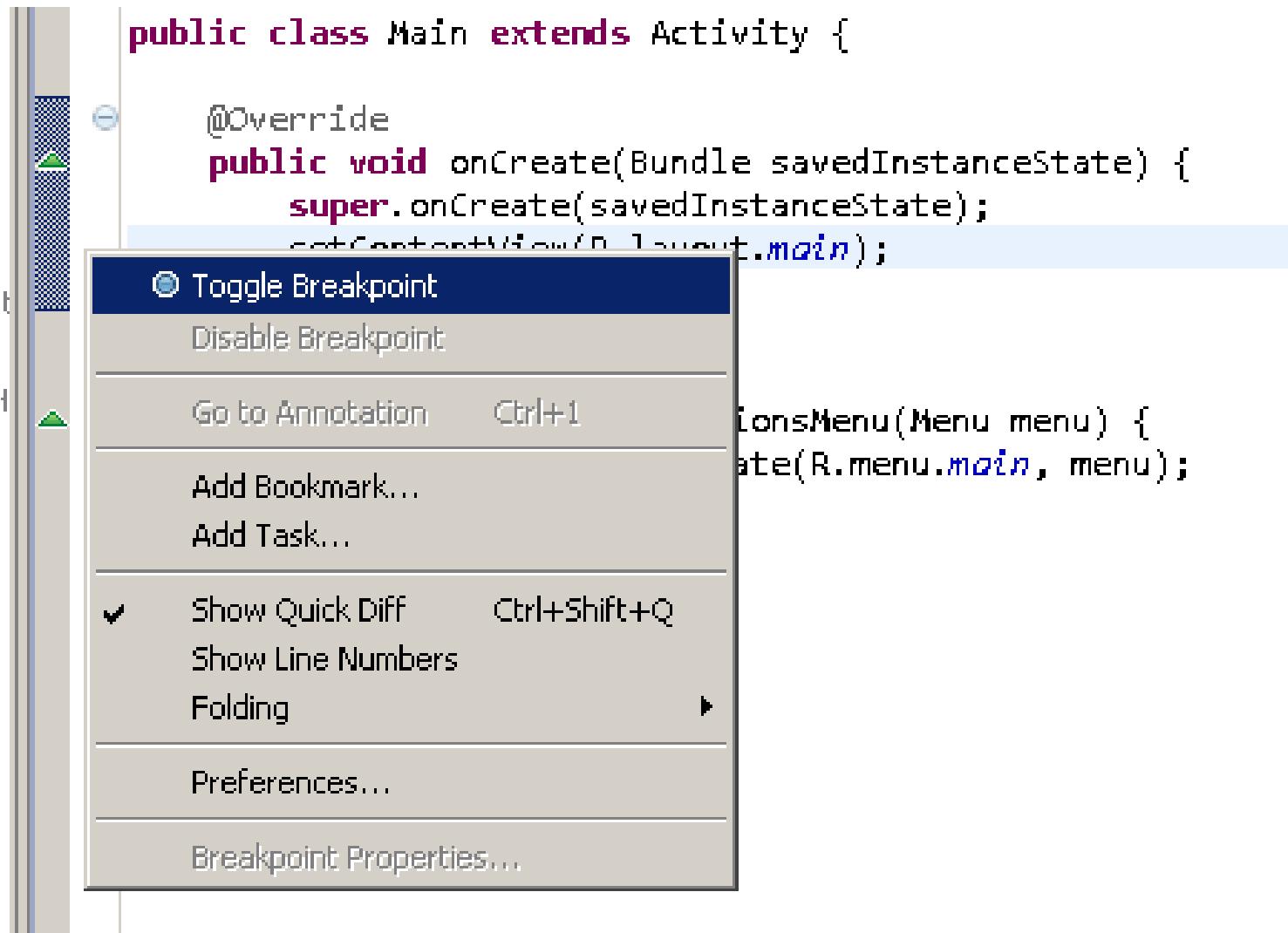




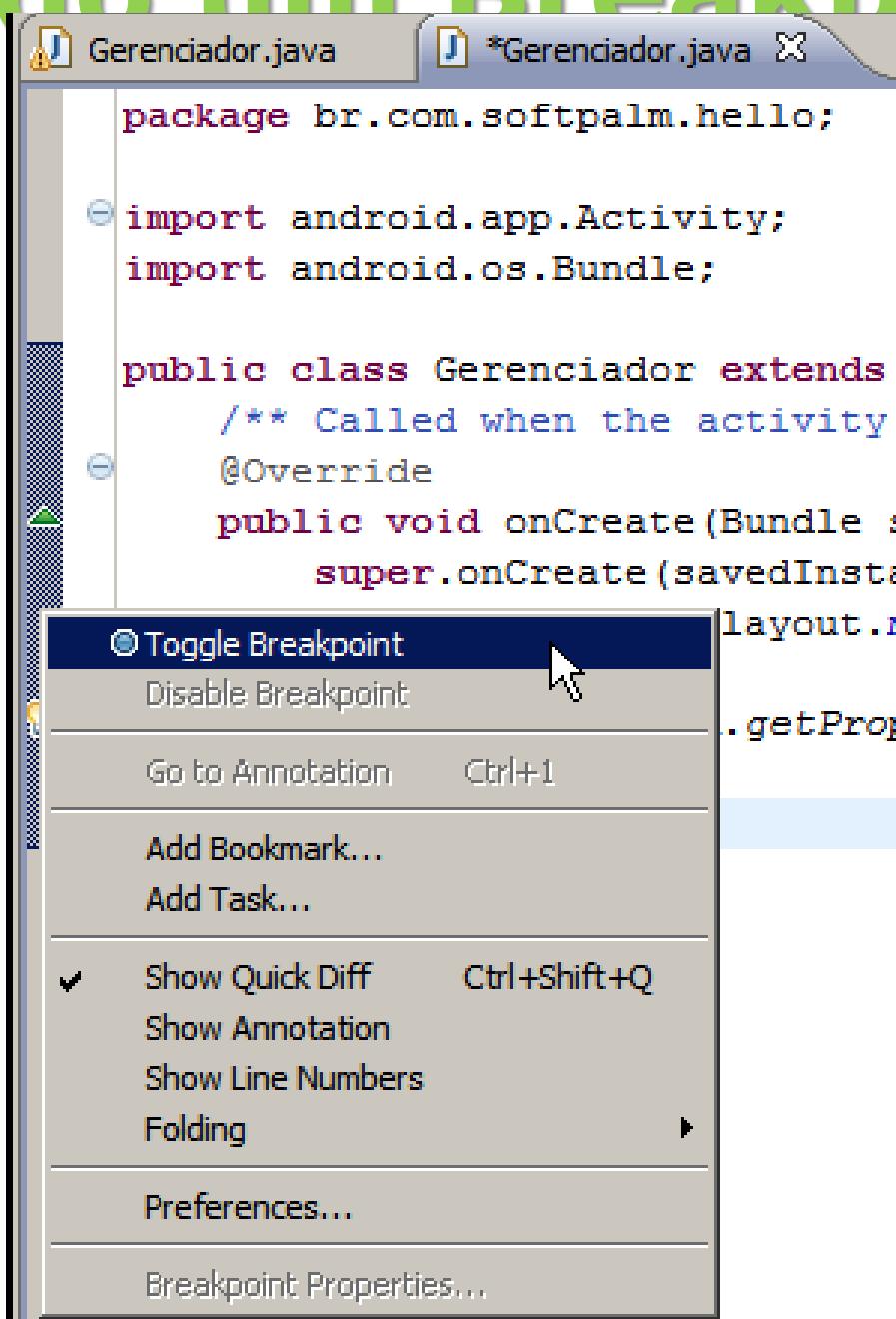
DEPURANDO E DISTRIBUINDO



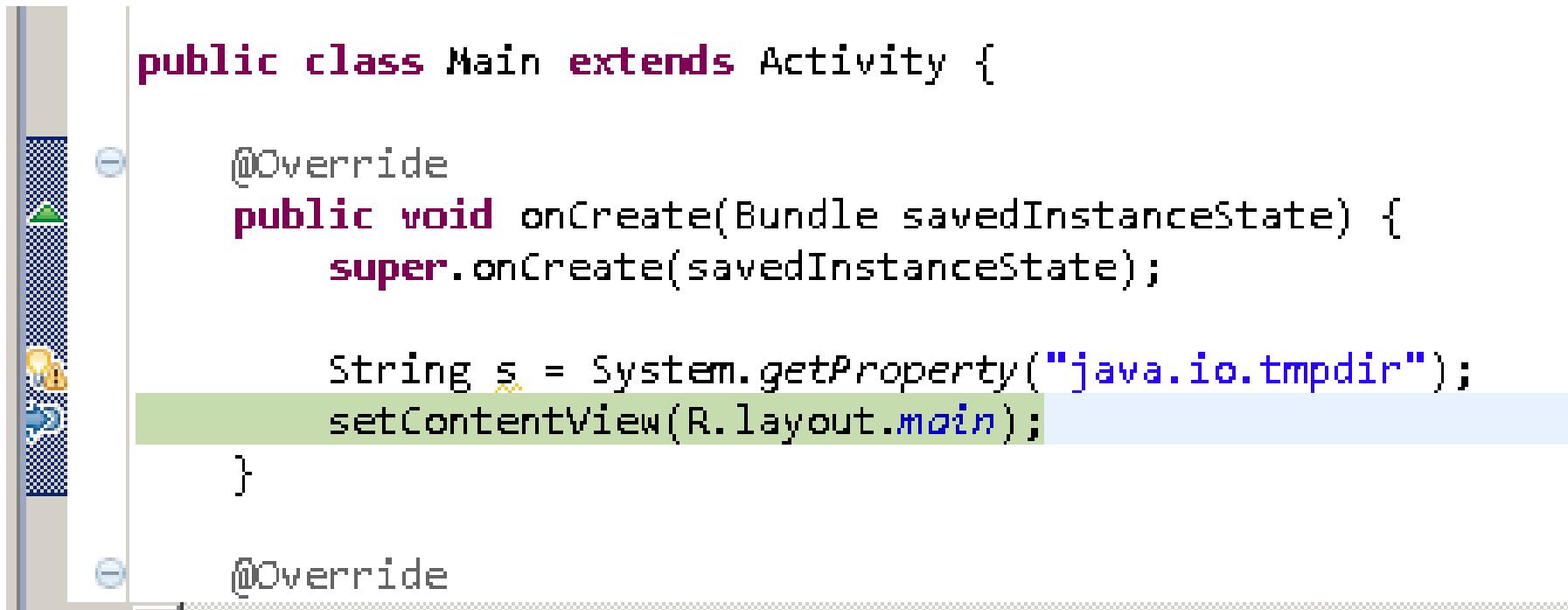
Criando um Breakpoint



Criando um Breakpoint



Criando um Breakpoint



The screenshot shows the Java code for a Main activity. A green rectangular highlight covers the line `setContentView(R.layout.main);`. To the left of the code, there is a vertical toolbar with several icons: a blue square at the top, a green triangle pointing up, a yellow lightbulb, a blue double-headed arrow, and a red circle with a minus sign.

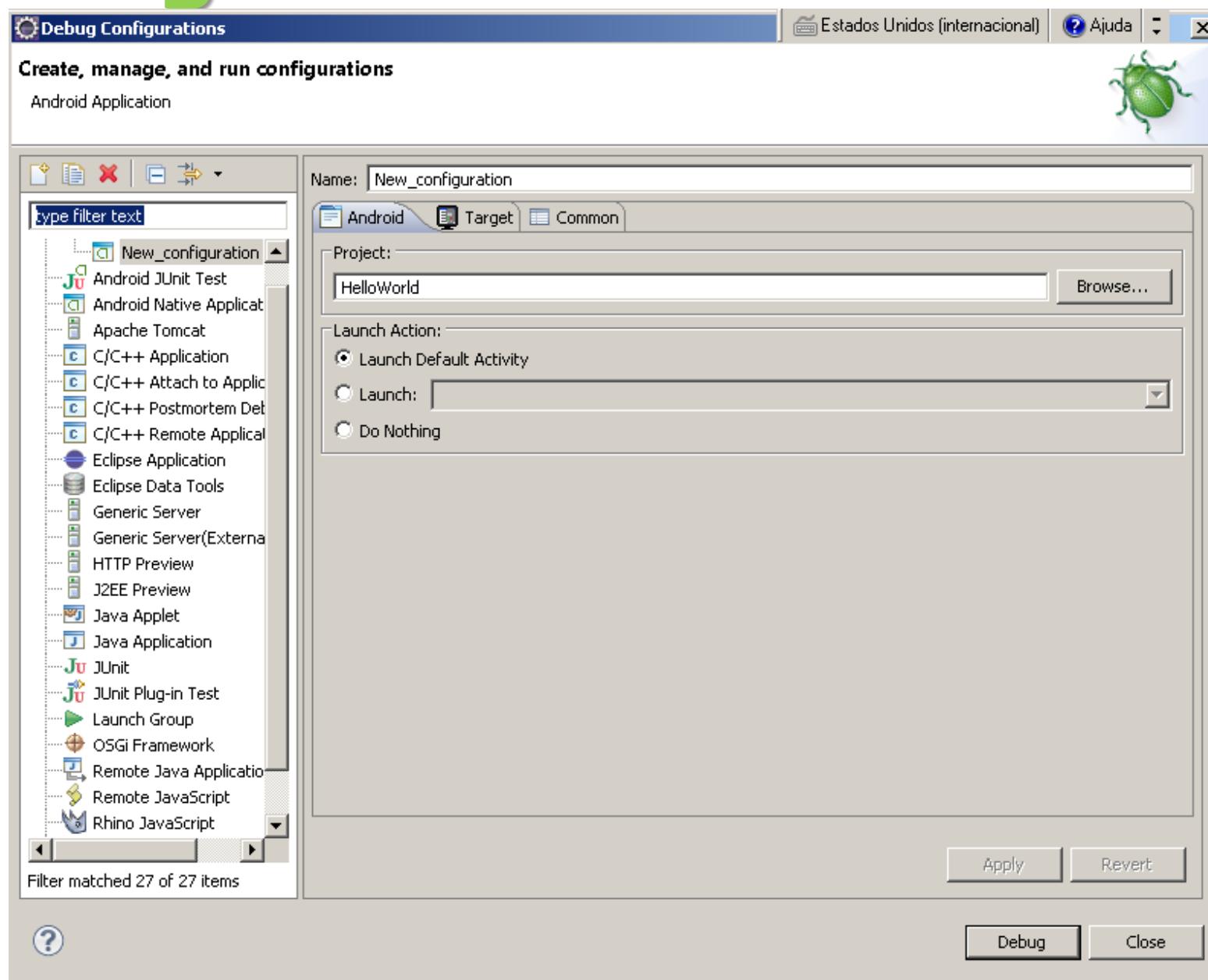
```
public class Main extends Activity {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        String s = System.getProperty("java.io.tmpdir");
        setContentView(R.layout.main);
    }

    @Override
}
```

Configurando o Modo Debug



Configurando o Modo Debug

Debug Configurations Estados Unidos (internacional) Ajuda X

Create, manage, and run configurations

Android Application

New_configuration

Deployment Target Selection Mode

- Always prompt to pick device
- Launch on all compatible devices/AVD's
- Automatically pick compatible device: Always uses preferred AVD if set below, launches on compatible device/AVD otherwise.

Select a preferred Android Virtual Device for deployment:

AVD Name	Target Name	Platform	API Level	CPU/ABI
android4	Android 4.1.2	4.1.2	16	ARM (armeabi-v...

Details... Start... Refresh Manager...

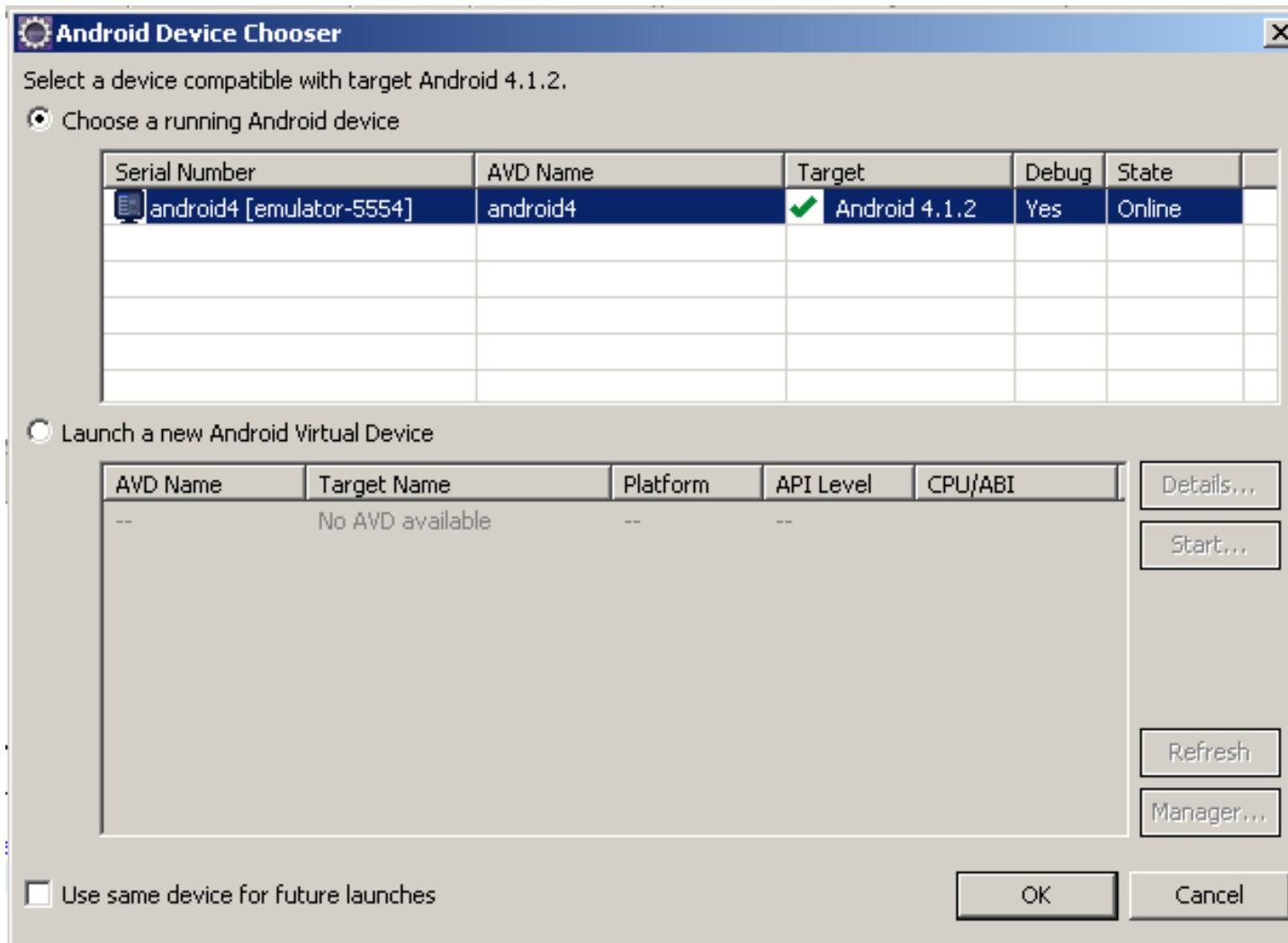
Emulator launch parameters:
If no compatible and active devices or AVD's are found, then an AVD might be launched. Provide options for the AVD launch below.

Network Speed: Full

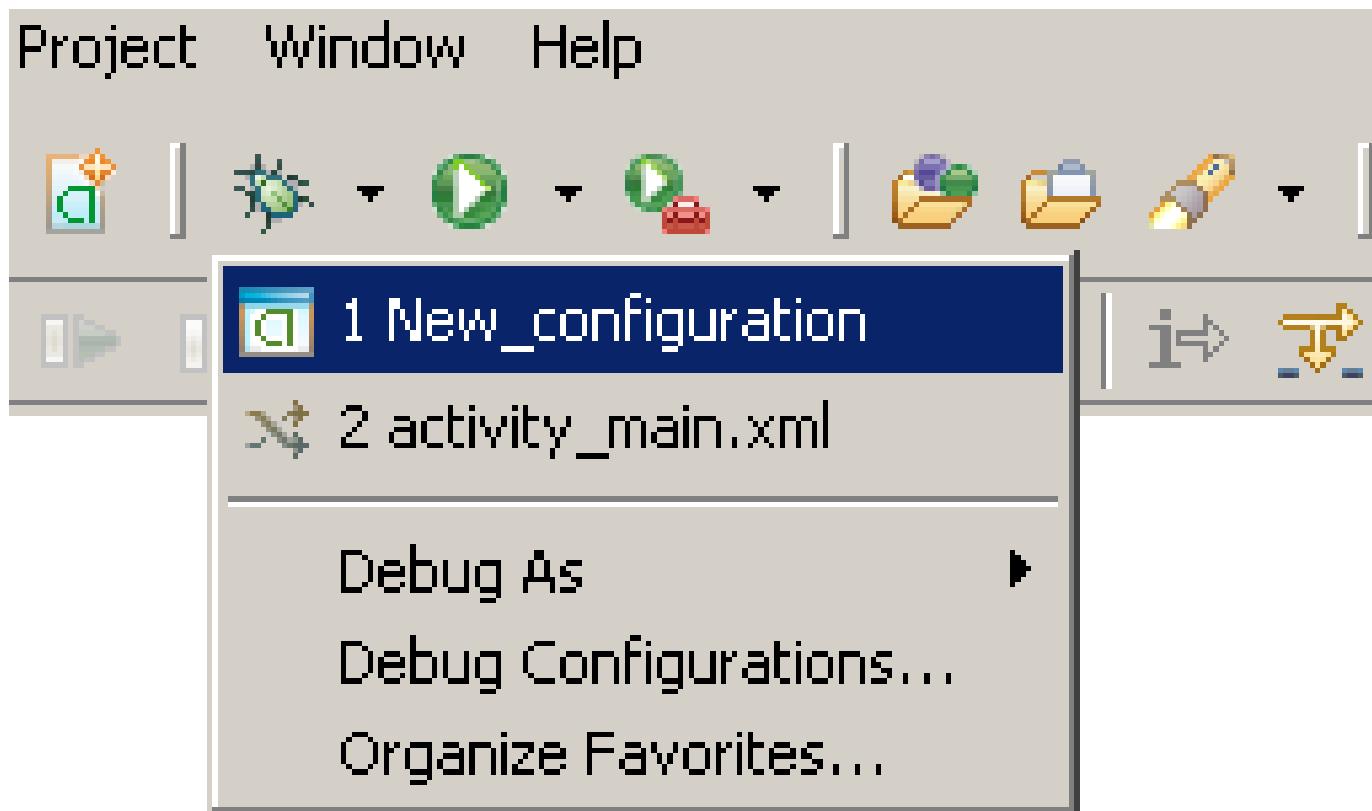
Apply Revert Debug Close

Filter matched 27 of 27 items

Configurando o Modo Debug seleção manual



Executando no Modo Debug





Debug - HelloWorld/src/com/example/helloworld/Main.java - Eclipse

File Edit Refactor Run Source Navigate Project Window Help

Variables Breakpoints

Name	Value
this	Main (id=830021221408)
s	/data/data/com.example.helloworld/cache (id=8300212017...)
savedInstanceState	null

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    String s = System.getProperty("java.io.tmpdir");
    setContentView(R.layout.main);
}
```

com.example.helloworld
import declarations
Main
onCreate(Bundle) : void
onCreateOptionsMenu(Menu) : boolean

Console Tasks Problems Executables

LogCat

L...	Time	PID	TID	Application	Tag
I	11-09 12:34:1...	822	822	com.example.h...	System.out
I	11-09 12:34:1...	822	822	com.example.h...	System.out

Iniciar 10:35

DEPLOY



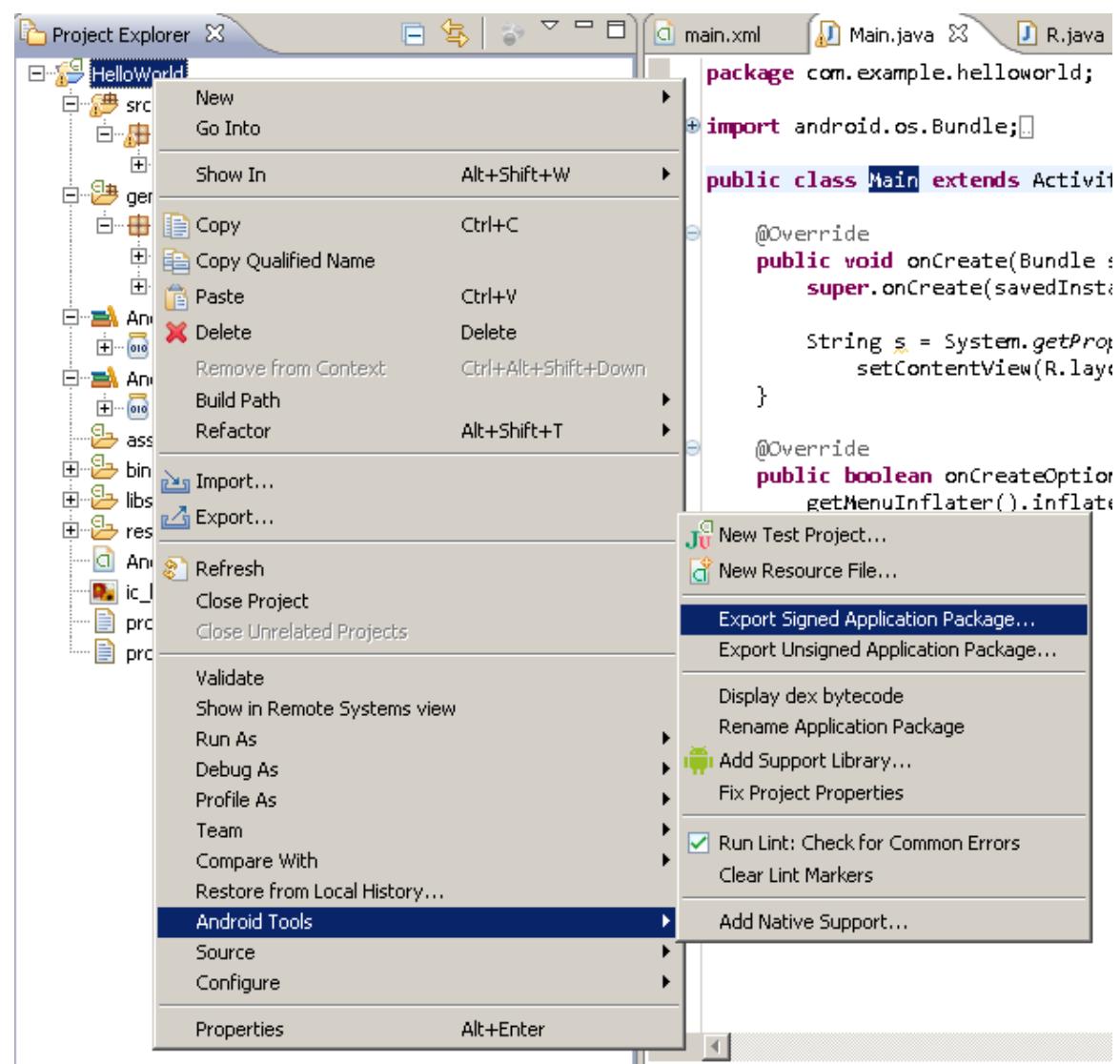
Deploy

- Processo de geração de um arquivo de instalação chamado .apk
- O Android exige que todas as aplicações instaladas tenham certificados digitais assinados com uma chave privada.
- Este certificado é usado para identificar o autor da aplicação e manter uma certa relação de confiança.



Deploy

- As ferramentas do Android implementadas no Eclipse ajudarão na assinatura



Deploy

 **Export Android Application**

Project Checks

Performs a set of checks to make sure the application can be exported.



Select the project to export:

Project:

No errors found. Click Next.

< Back Finish Cancel

Deploy



Deploy

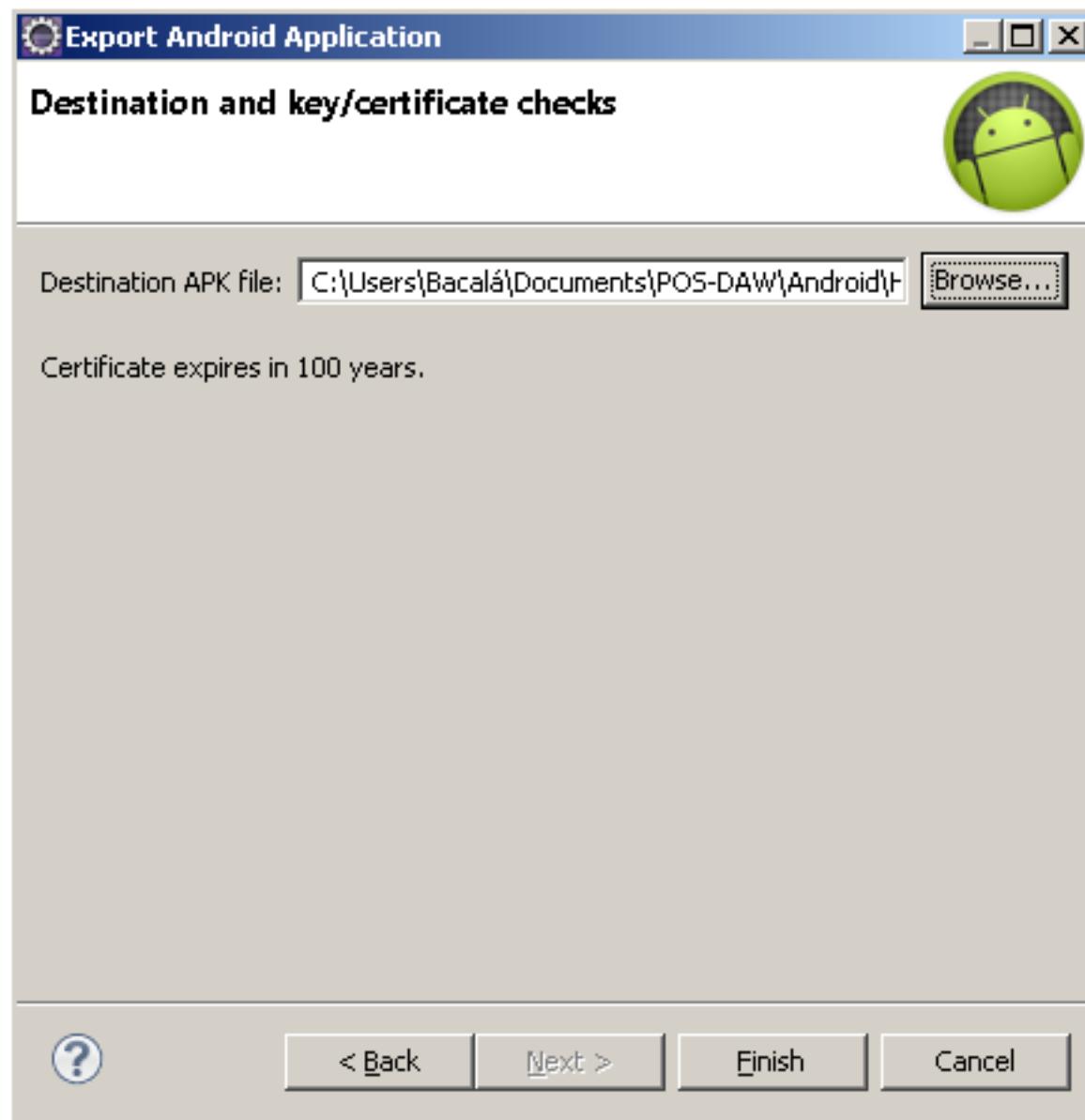
 Export Android Application

 Key Creation

Alias:	Bacala
Password:	*****
Confirm:	*****
Validity (years):	100
First and Last Name:	silvio Bacala Jr
Organizational Unit:	
Organization:	
City or Locality:	Uberlandia
State or Province:	MG
Country Code (XX):	

 < Back  Next >  Finish  Cancel

Deploy



Deploy

Nome	Data de modificação	Tipo	Tamanho
 HelloWorld.apk	09/11/2012 11:14	Arquivo APK	145 KB
 ~\$001-Android.pptx	09/11/2012 09:56	Apresentação do Microsoft Pow...	1 KB
 001-Android.pptx	09/11/2012 09:56	Apresentação do Microsoft Pow...	3.462 KB
 apostila-120909201917-phpapp02.pdf	08/11/2012 11:57	Adobe Acrobat Document	5.255 KB
 installer_r20.0.3-windows.exe	08/11/2012 11:38	Aplicativo	68.844 KB
 eclipse.zip	08/11/2012 11:12	Arquivo ZIP do WinRAR	214.442 KB
 android-sdk-windows.rar	08/11/2012 11:09	Arquivo do WinRAR	168.802 KB
 jdk-6u25-windows-i586.exe	08/11/2012 10:57	Aplicativo	78.498 KB