Programação para Dispositivos Móveis

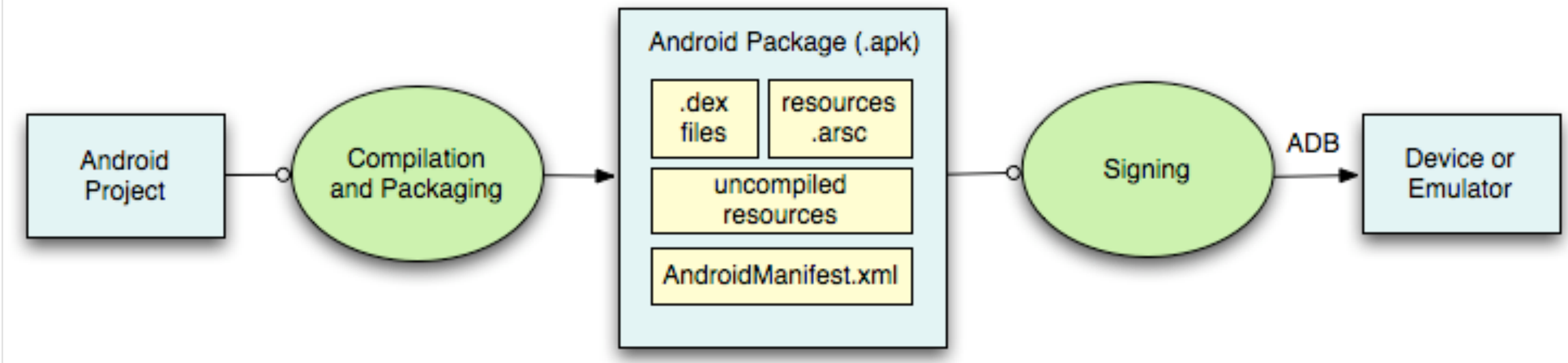
Leopoldo Teixeira
lmt@cin.ufpe.br | @leopoldomt

Componentes fundamentais de uma aplicação Android

Componentes

- Activity
- Service
- ContentProvider
- BroadcastReceiver

Processo de construção de apps Android



Criando um app

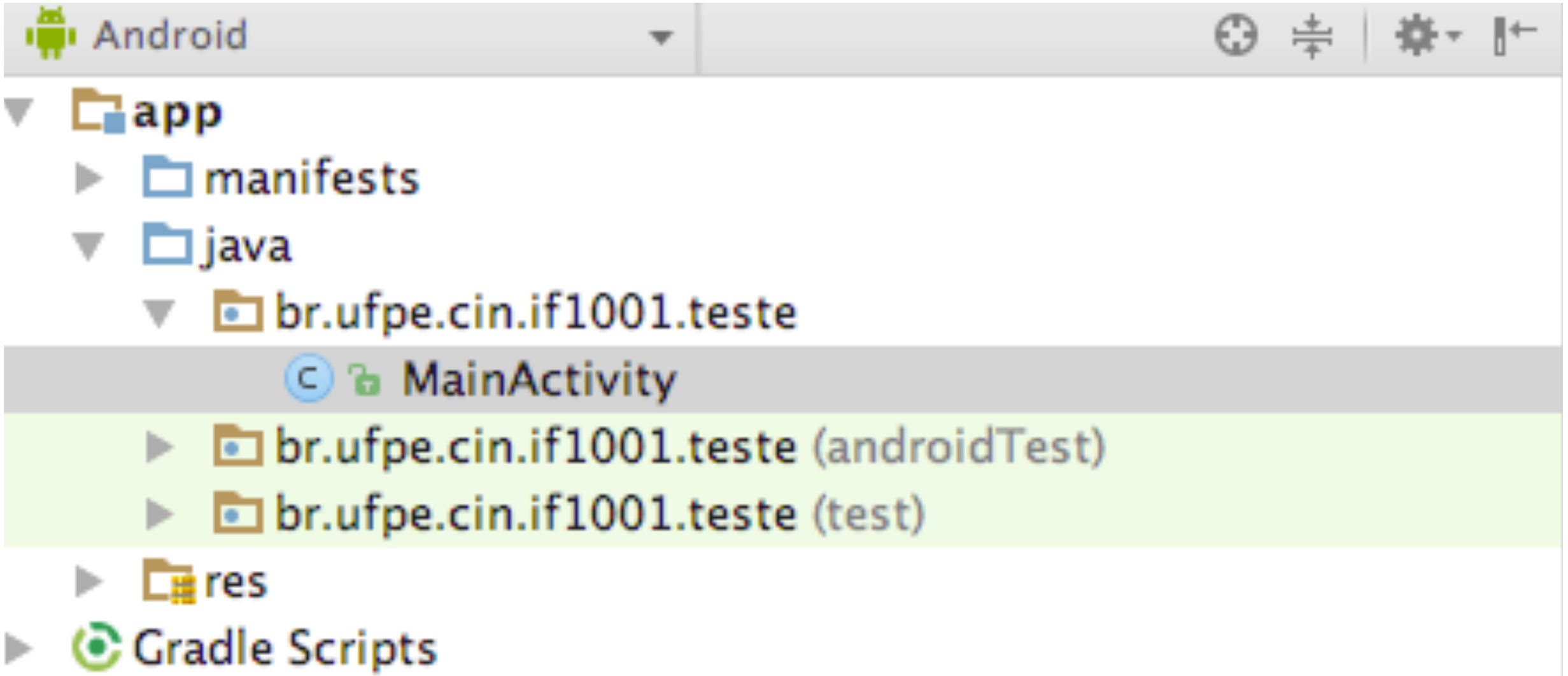
- Definir recursos
- Implementar classes da aplicação
- Empacotar o app
- Instalar e rodar

Hello World

- Não precisa escrever uma linha de código… (na prática)
- Vamos ver como se parece, na IDE…
- Podemos interagir com o emulador, por meio de telnet (https://developer.android.com/studio/run/emulator-commandline.html)
- Também é possível ‘debugar' a aplicação

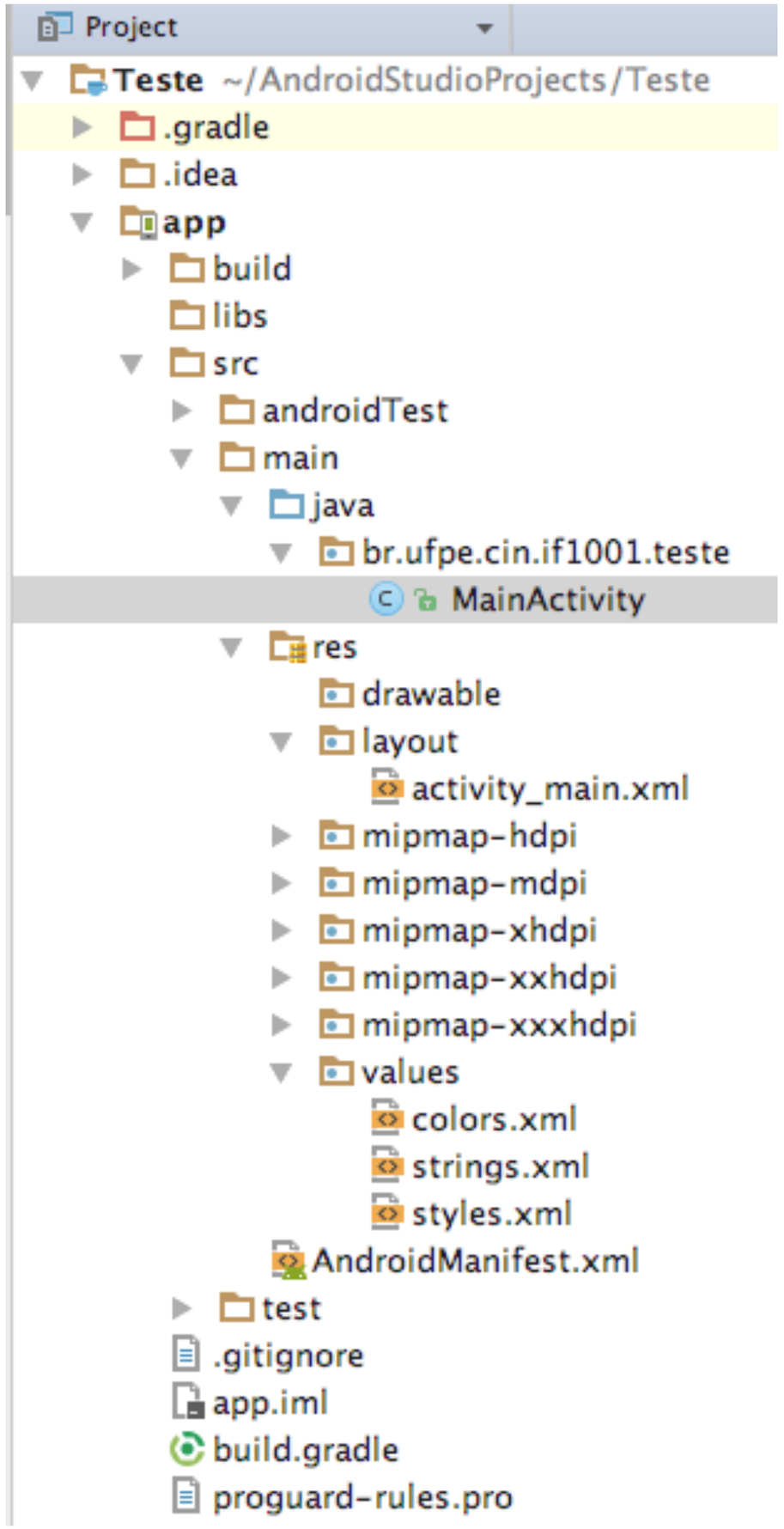
Projetos

- Application/Project/Package Name
- Minimum required SDK
- Target SDK



Project view

Classic Project view



Instant Run

- Alguns dispositivos vão permitir o modo “Instant Run”
- Ao invés de fazer o build do app inteiro, a IDE faz o hot swap de código e recursos no app rodando no dispositivo ou emulador

Código Java

- Geralmente há pelo menos uma Activity, no diretório relativo ao package name
- Há código gerado em vários outros diretórios
- Como, por exemplo, a classe R.java

Definindo Recursos

- Diretório res/

- Entidades que não são código-fonte

- Muitos tipos diferentes, como layouts, strings, imagens, menus, animação, etc.

- Permitem que apps sejam personalizados para diferentes dispositivos e usuários

- http://developer.android.com/guide/topics/resources/index.html

Exemplos

- **`res/drawable`** para imagens (PNG, JPEG…)
- **`res/layout`** para especificações de layout em XML
- **`res/menu`** para especificações de menu em XML
- **`res/raw`** para arquivos em geral
- **`res/values`** para strings, dimensões…
- **`res/xml`** para arquivos XML de propósito geral

Instruções de build

- **`build.gradle`**
  - Um ou mais arquivos utilizados

- Em projetos Eclipse, estava espalhado em arquivos como **`project.properties`** e **`.classpath`**

Gradle

- Build system, build automation (make, ant, maven)
- Essas ferramentas permitem especificar como utilizar compiladores, linkers, packagers, etc.
- DSL implementada em Groovy (embedded)
  - similar a uma definição XML

`build.gradle`

- Dois arquivos normalmente
- Project-level
  - controla configuração para todos os módulos
  - configuração do processo de build
  - dependências que precisam ser satisfeitas
- Module-level

build.gradle

- Dois arquivos normalmente
- Project-level
- Module-level
  - dependências específicas de código
  - informações de configuração específicas de Android - diretamente relacionado ao manifest

AndroidManifest.xml

- Base de qualquer aplicação
- Geralmente em **`src/main`**
- O que a sua aplicação tem, activities, services, etc
- Como estas peças se ligam com o sistema
  - que activity deve aparecer no main menu (launcher)

Elementos em comum AndroidManifest.xml e build.gradle

- Package name / Application ID
- minSdkVersion e targetSdkVersion
- Version Code e Version Name

build.gradle

- compileSdkVersion
    - API level a ser utilizada para compilação
- buildToolsVersion
    - versão da Android SDK build tools

AndroidManifest.xml

- Nome da aplicação
- Componentes
  - Activities, Services
- Intent Filters
- Entre outros: permissões, API mínima, etc.

Strings

- Vários tipos: String, String Array, Plurals
- Armazenadas em res/values/*.xml, ex.:
  <string name=“hello”>Hello World!</string>
- Pode incluir formatação e estilos
- Podem ser acessados por outros recursos (@string/string_name) e em código Java (R.string.string_name)

Layouts

- Especificam interface com o usuário
- Arquivos XML, mas ferramentas permitem editar visualmente
- Armazenados em res/layout/*.xml
- Também podem ser acessados tanto por outros recursos como em código Java
  (notação similar à de strings)

Usando Múltiplos Layouts

- Assim como feito para Strings, podemos criar múltiplos layouts

- Por conta de orientação da tela, tamanho de tela, etc.

R.java

- Em tempo de compilação, os recursos são usados para gerar a classe R.java

- Contém código Java usado para acessar recursos nas classes da aplicação

Criando um app

- Definir recursos
- Implementar classes da aplicação
- Empacotar o app
- Instalar e rodar

Implementando Classes

- Envolve a criação de pelo menos uma Activity

- Código de inicialização geralmente colocado no método onCreate()

Empacotar aplicação

- Juntar componentes e recursos da aplicação em um arquivo .apk (‘executável’ android)

- Desenvolvedores especificam informação necessária para aplicação em um arquivo chamado AndroidManifest.xml

Instalar e rodar aplicação

- Rodar em emulador ou dispositivo
- Também é possível rodar a partir da linha de comando (adb)
- É necessário habilitar USB debugging no celular

Programação para Dispositivos Móveis

Leopoldo Teixeira
lmt@cin.ufpe.br | @leopoldomt