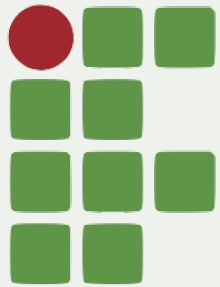
Empacotamento em arquivos JAR e uso de bibliotecas

POO29004 – Engenharia de Telecomunicações

Prof. Emerson Ribeiro de Mello

http://docente.ifsc.edu.br/mello/poo

01 de abril de 2020



INSTITUTO FEDERAL
Santa Catarina
Câmpus São José

Arquivo JAR

- `Java ARchive` (JAR) é um **arquivo compactado com um conjunto de classes** e outros recursos (imagens, áudio, etc) e usado para distribuir aplicativos ou bibliotecas Java

- **Para executar um aplicativo** no formato JAR e que possua um arquivo manifesto[a] indicando qual classe tem o método `main`

```
1 java -jar aplicativo.jar
```

- Indicando a classe (quando não tem manifesto)

```
2 java -cp:aplicativo.jar poo.Principal
```

[a] https://docs.oracle.com/javase/tutorial/deployment/jar/manifestindex.html

Criando arquivo JAR executável usando o gradle

- Considere que criou um projeto Java (chamado `agenda`) usando o gradle e a classe que contém o método `main` chama-se `poo.Principal`

- No arquivo `build.gradle` é necessário inserir o seguinte bloco

```
3  // Linhas abaixo devem ser inseridas após o bloco "dependencies"
4  jar {
5      manifest {
6          // Classe principal da aplicação
7          attributes "Main-Class": "poo.Principal"
8      }
9  }
```

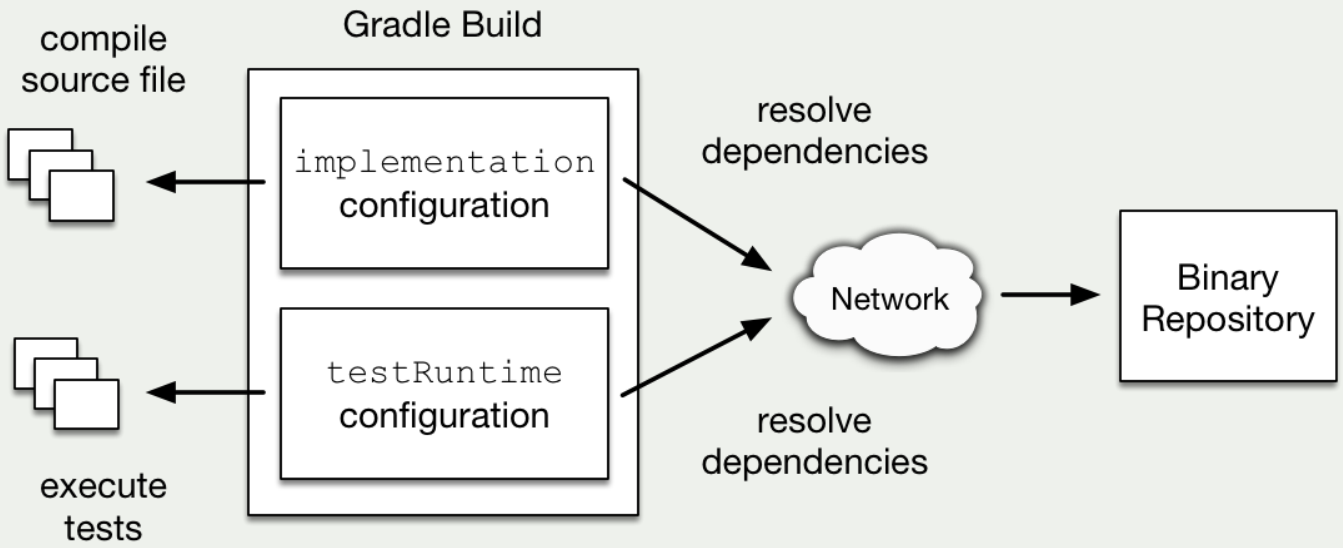
- Para gerar o Jar e executar a aplicação

```
10 ./gradlew jar
11
12 java -jar build/libs/agenda.jar
```

Dependência de bibliotecas com o Gradle

Dependências de bibliotecas com o Gradle

- Gradle permite informar dependências para compilar o projeto ou para executar testes
- As bibliotecas poderão estar em arquivos JAR no diretório do projeto
- Ou podem ser obtidas, de forma transparente, de repositórios na Internet, como `jcenter()` ou `mavenCentral()`
  - Procure por bibliotecas em https://mvnrepository.com



Dependência de bibliotecas com arquivos JAR locais

1. Crie um projeto gradle com o IntelliJ ou parta do esqueleto de projeto disponível aqui

2. Crie um diretório `libs` na raiz do projeto e coloque ali os arquivos `.jar` das bibliotecas que fará uso
   - Se estiver fazendo uso do git, inclua a linha `!libs/*.jar` no arquivo `.gitignore` para garantir que os arquivos `jar` das bibliotecas sejam colocados no git

3. Inclua a dependências dessas bibliotecas dentro da bloco `dependencies` no arquivo **build.gradle**. Abaixo um exemplo para incluir o arquivo `biblioteca.jar`

```
13 dependencies {
14     // importando arquivo JAR que está dentro do diretório libs
15     implementation files('libs/biblioteca.jar')
16 }
```


Estrutura de arquivos do projeto Gradle

```
.
|-- build.gradle
|-- gradle
|   `-- wrapper
|       |-- gradle-wrapper.jar
|       `-- gradle-wrapper.properties
|-- gradlew
|-- gradlew.bat
|-- libs
|   `-- biblioteca.jar
|-- settings.gradle
`-- src
    |-- main
    |   |-- java
    |       `-- poo
    |           `-- Principal.java
    `-- test
        |-- java
            `-- poo
                `-- AppTest.java
```


Dependência de bibliotecas que serão baixadas da internet

- Suponha que você queira usar a biblioteca do projeto Google ZXing, sendo assim foi no JCenter para procurar pelas linhas que deve colocar no arquivo `build.gradle`
- Abaixo é listado um bloco `dependecies` com adição de bibliotecas ZXing que serão baixadas da Internet e de uma biblioteca que está em um arquivo JAR local

```
37 dependencies {
38   testImplementation 'junit:junit:4.12'
39
40   // https://bintray.com/bintray/jcenter/com.google.zxing%3Acore
41   implementation 'com.google.zxing:core:3.4.0'
42
43   // https://bintray.com/bintray/jcenter/com.google.zxing%3Ajavase
44   implementation 'com.google.zxing:javase:3.4.0'
45
46   // importando arquivo JAR que está dentro do diretório libs
47   implementation files('libs/biblioteca.jar')
48 }
```

Empacotando aplicação que contém dependências

- Para empacotar a aplicação em um único arquivo JAR que contenha todas as biblioteca (tudo aquilo que foi declarado no bloco `dependecies`) será necessário fazer uso do plugin Shadow
- Adicione a linha referente ao Shadow dentro do bloco `plugins` do arquivo `build.gradle`

```
49 plugins {
50   id 'com.github.johnrengelman.shadow' version '5.2.0'
51   id 'java'
52 }
```

- E para gerar o JAR, basta executar a tarefa `shadowJar` do Gradle.

```
53 ./gradlew shadowJar
```

  - Isso também pode ser feito pelo painel Gradle da IDE IntelliJ

Exercícios

Exercícios – entrega via Github Classroom: 07/04/2020

1. **Empacotamento de aplicação com dependências**
   - Baixe o exemplo em https://github.com/poo29004/java-qrcode-barcode-jar, estude os conteúdos dos arquivos `Principal.java` e `build.gradle`

2. **Aplicação Java distribuída em arquivo JAR**
   - Modifique o projeto **Agenda Telefônica** da Lista 5 de formar a permitir gerar um arquivo `JAR` executável
   - Faça um *push* com as modificações para o repositório

3. **Aplicação Java distribuída em arquivo JAR**
   - Modifique o projeto **Drone** da Lista 05 de forma que faça uso de coordenadas GPS (latitude e longitude) (faça *push* pro repositório)
   - Classe `GeodeticCalculator.java` da biblioteca GeoTools provê métodos para calcular a distância entre duas coordenadas GPS
   - A biblioteca GeoCalc também provê funcionalidades para cálculos com coordenadas geográficas