Ciência da Computação

Programação Orientada a Objetos Avançado

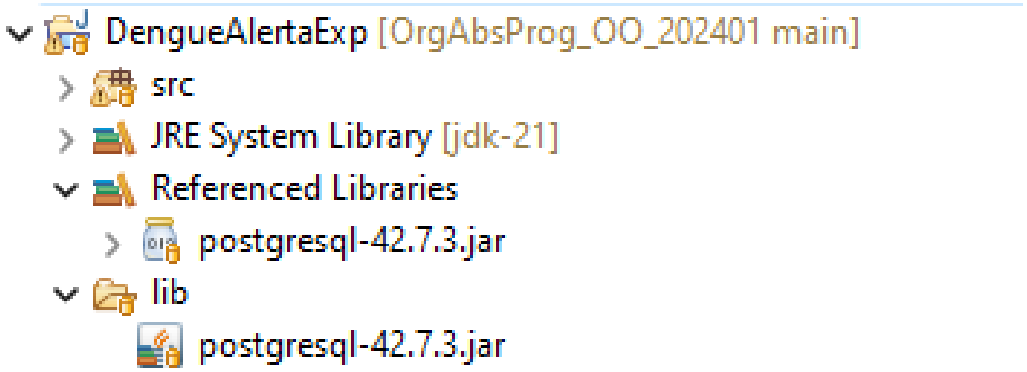
Prof. Luciano Rodrigo Ferretto

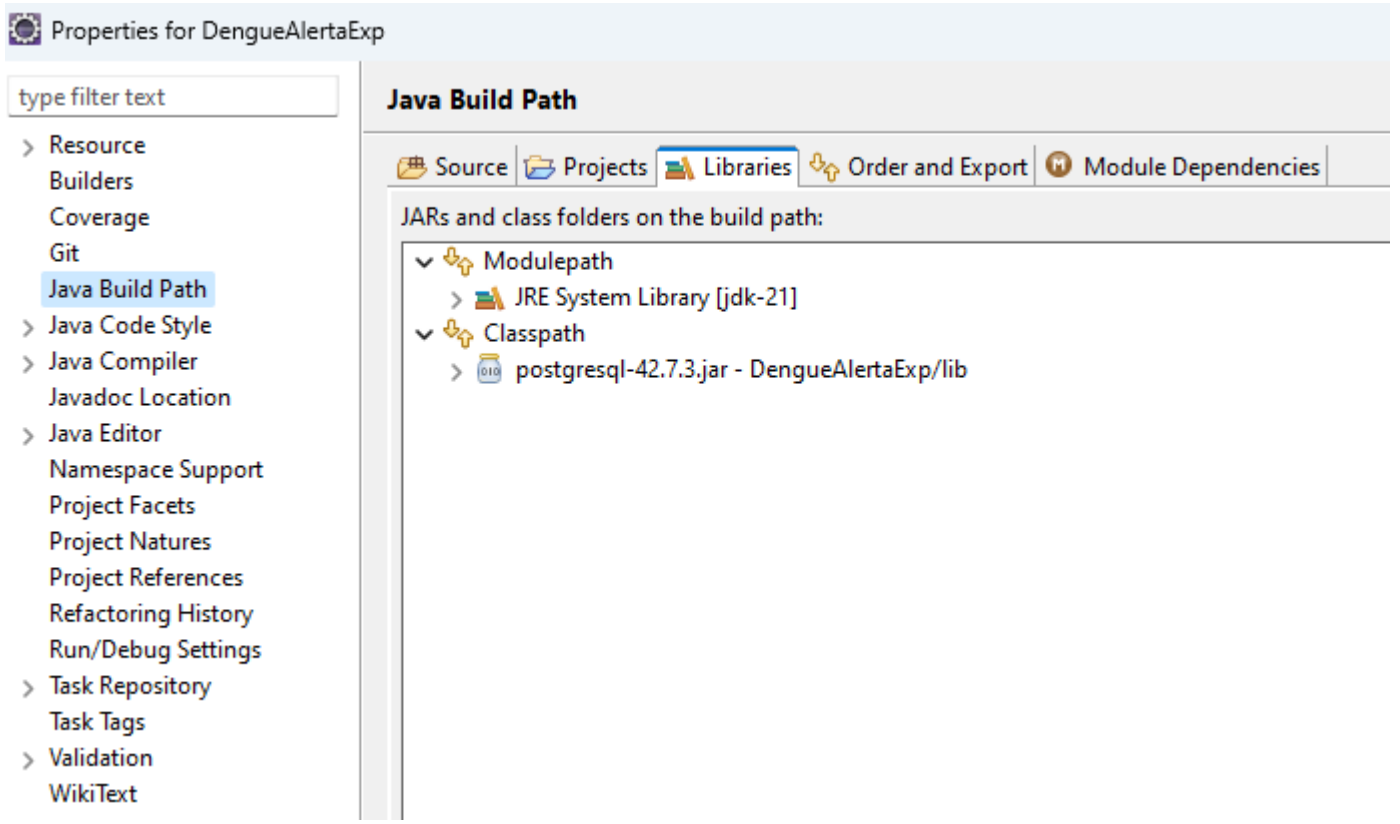
Maven no Java: Uma Visão Geral

Era Pré-Maven: Compilação, Empacotamento e Gerenciamento de Bibliotecas Manual

- O desenvolvedor precisava **gerenciar** manualmente as versões das dependências e garantir a compatibilidade entre elas.
- A **compilação** envolvia o uso do comando *javac* para compilar os arquivos *.java* em arquivos *.class*.
- Após a compilação, o desenvolvedor precisava **empacotar** manualmente as classes compiladas e as dependências em um JAR (Java Archive) executável.

Era Pré-Maven: Compilação e Gerenciamento de Bibliotecas Manual





Era Pré-Maven: Makefiles

- Para automatizar parte desse processo, alguns desenvolvedores começaram a usar ferramentas de construção baseadas em Makefiles, como o make do UNIX.
- No entanto, o make foi projetado para C/C++ e não estava bem adaptado às necessidades do Java, especialmente no que diz respeito ao gerenciamento de dependências.

Era Pré-Maven: Ferramentas de Build Ant

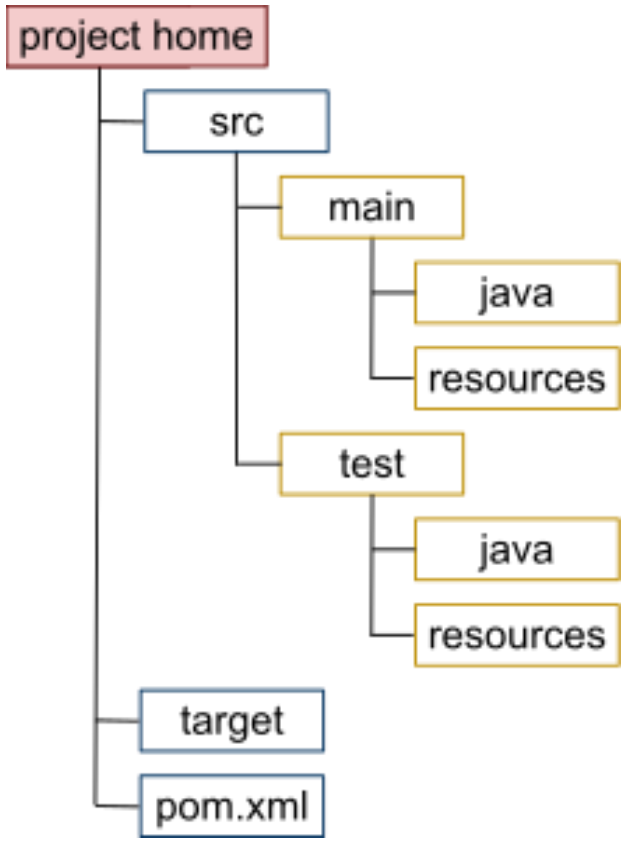
- A necessidade de uma solução melhor levou ao desenvolvimento do **Apache Ant** no final dos anos 1990.
- Lançado pela Apache Software Foundation, o Ant foi uma grande melhoria em relação ao uso de Makefiles.
- O Ant introduziu um sistema de build baseado em XML, onde as tarefas eram definidas de forma declarativa. Ele era mais flexível e adequado para Java do que os Makefiles.
- O Ant também permitia a execução de tarefas complexas, como **compilação**, **empacotamento** e **execução de testes**, por meio de uma configuração centralizada em um arquivo build.xml.

Surgimento do Maven

- O Maven é uma ferramenta de automação de build e gerenciamento de dependências amplamente utilizada no ecossistema Java.
- **Automação de Build**
  - Maven automatiza o processo de build de projetos Java. Isso inclui tarefas como compilação do código fonte, execução de testes, geração de relatórios, empacotamento e implantação de artefatos (como arquivos JAR, WAR, etc.).
- **Gerenciamento de Dependências**
  - O Maven simplifica o gerenciamento de dependências de bibliotecas externas. Você especifica as dependências no arquivo pom.xml (Project Object Model), e o Maven resolve e baixa essas dependências automaticamente de repositórios remotos como o Maven Central.

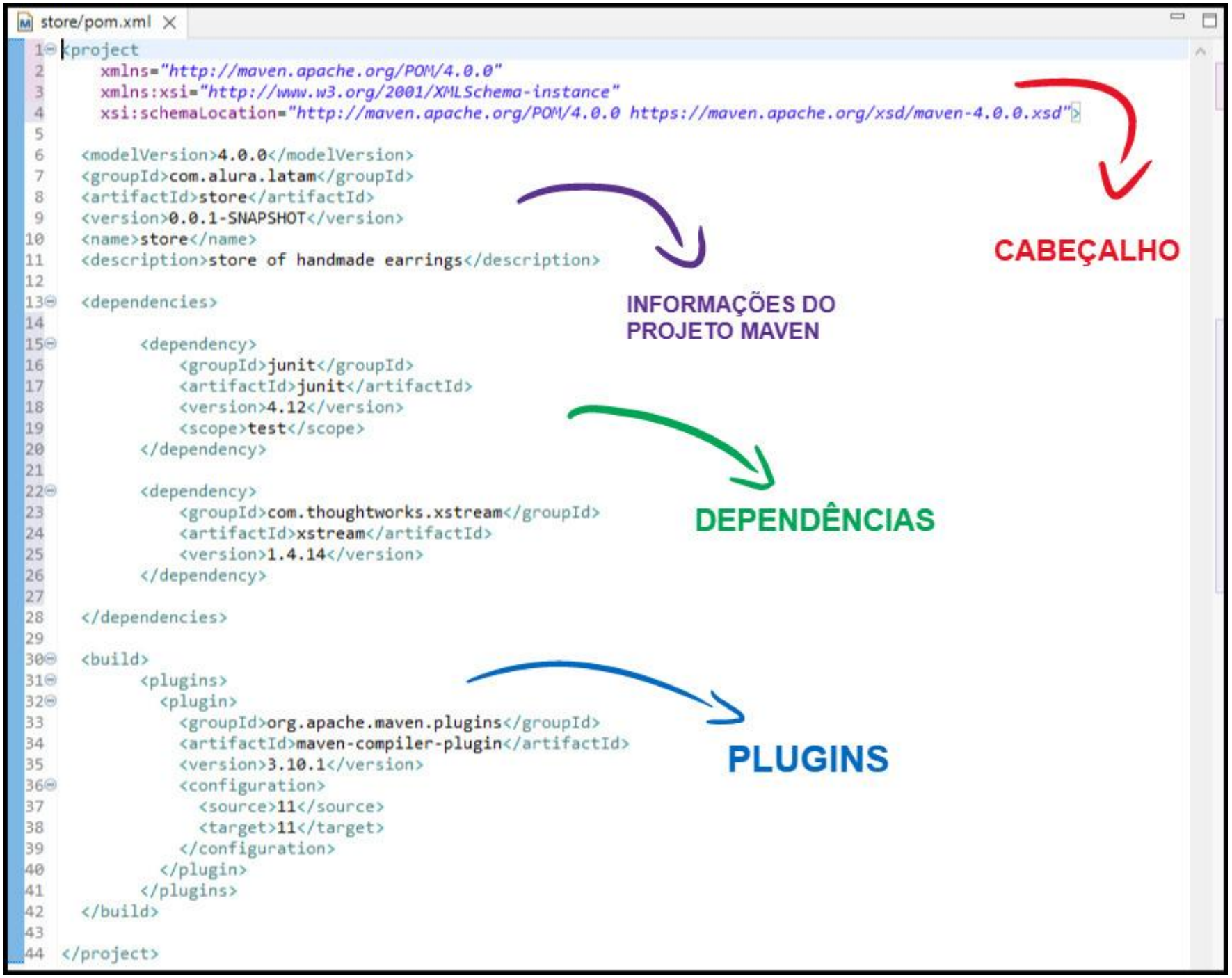
Surgimento do Maven

- **Estrutura de Diretórios Padrão**
  - O Maven adota uma estrutura de diretórios padrão para projetos Java, o que ajuda a padronizar e organizar o código fonte e os recursos do projeto. A estrutura típica de um projeto Maven é:



Surgimento do Maven

- **Arquivo POM (Project Object Model)**
  - O pom.xml é o arquivo central em um projeto Maven. Ele define o projeto, suas dependências, plugins e outras configurações de build. Um exemplo básico de um arquivo pom.xml:




```xml
<project
    xmlns="http://maven.apache.org/POM/4.0.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

  <modelVersion>4.0.0</modelVersion>
  <groupId>com.alura.latam</groupId>
  <artifactId>store</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <name>store</name>
  <description>store of handmade earrings</description>

  <dependencies>

      <dependency>
          <groupId>junit</groupId>
          <artifactId>junit</artifactId>
          <version>4.12</version>
          <scope>test</scope>
      </dependency>

      <dependency>
          <groupId>com.thoughtworks.xstream</groupId>
          <artifactId>xstream</artifactId>
          <version>1.4.14</version>
      </dependency>

  </dependencies>

  <build>
      <plugins>
        <plugin>
          <groupId>org.apache.maven.plugins</groupId>
          <artifactId>maven-compiler-plugin</artifactId>
          <version>3.10.1</version>
          <configuration>
            <source>11</source>
            <target>11</target>
          </configuration>
        </plugin>
      </plugins>
  </build>

</project>
```

Surgimento do Maven

- **Plugins**
  - Maven é extensível através de plugins. Existem muitos plugins disponíveis para diversas tarefas, como compilação, empacotamento, teste, geração de relatórios e muito mais. Você pode configurar e adicionar plugins no pom.xml.
- **Repositórios**
  - Maven utiliza repositórios para armazenar dependências e plugins. Existem dois tipos principais de repositórios:
  - **Repositório Local**: Localizado no sistema de arquivos do desenvolvedor, normalmente no diretório ~/.m2/repository.
  - **Repositório Remoto**: Repositórios públicos (como Maven Central) ou privados que armazenam artefatos compartilhados entre projetos.

Elementos Principais do POM.xm.

- O **groupId** é um identificador único para o grupo ou organização que está desenvolvendo o projeto. Ele é usado para agrupar projetos relacionados. O padrão de nomenclatura geralmente segue a convenção de nomes de domínio invertidos, similar aos pacotes Java, para garantir a unicidade.

```
<groupId>br.edu.atitus</groupId>
<artifactId>denguealerta</artifactId>
<version>0.0.1-SNAPSHOT</version>
<name>DengueAlerta</name>
<description>Demo project for Spring Boot</description>
```

Elementos Principais do POM.xm.

- O **artifactId** é o identificador único do projeto dentro do grupo. Ele representa o nome do artefato (por exemplo, um JAR ou WAR) que será gerado.

```
<groupId>br.edu.atitus</groupId>
<artifactId>denguealerta</artifactId>
<version>0.0.1-SNAPSHOT</version>
<name>DengueAlerta</name>
<description>Demo project for Spring Boot</description>
```