Fundamentos de Desenvolvimento de Software
Prof. Bernardo Copstein

Roteiro: usando Gson com Maven

O objetivo deste roteiro é mostrar como configurar um projeto Java usando Maven. Para tanto vamos usar o sistema que consome o serviço de CEP desenvolvido anteriormente e usar a biblioteca Gson para converter strings JSON em instancias de objetos Java.

Parte 1: criando o projeto

De maneira complementar a esta "parte 1" considere olhar também este tutorial.

O Maven é uma ferramenta antiga. Suas formas de uso vêm evoluindo. Originalmente o próprio Maven era usado para criar a estrutura do projeto. Hoje em dia a maioria das tecnologias que usam o Maven possuem uma ferramenta para tanto. Neste primeiro projeto, entretanto, vamos usar a opção original. A consequência é que serão gerados alguns arquivos em formatos antigos que terão de ser substituídos por outros fornecidos pelo professor

Inicialmente siga os passos que seguem:

1) Abra o VSCode
2) No menu principal selecione "View|Command Pallet" e em seguida "Java: Create Java Project"
3) Selecione o tipo de projeto a ser criado: "maven"
4) Selecione o tipo de "arquétipo" que será usado: "maven-archetype-quickstart"
5) Selecione a versão do "arquétipo": 1.4
6) Indique o identificador da sua organização: exemplo -> "com.demo"
7) Indique o nome do artefato que será construído: exemplo -> "consulta_cep_json"
8) Informe a pasta onde deverá ser criada a pasta raiz do projeto.
9) A partir deste ponto o Maven começa a baixar os arquivos que necessita. Antes de encerrar a criação do projeto irá fazer mais algumas perguntas que podem ser respondidas apenas com a tecla "<ENTER>" (assumindo os valores default).
10) Quando a criação do projeto estiver encerrada, toda a estrutura de pastas do projeto estará criada a partir da pasta indicada no passo 8. Localize esta pasta no "explorador de arquivos" e analise a estrutura criada. Note que ele gera um arquivo de descrição do projeto que já prevê as dependências para o JUnit (ferramenta de teste unitário) pois atualmente não se concebe um projeto sem teste unitário.
11) Como o "maven-archetype-quickstart" é um pouco antigo, o arquivo "pom" criado e os exemplos de casos de teste que ele gera usam uma versão ultrapassada do JUnit. Então é necessário fazer algumas alterações no projeto criado antes de começarmos:
    a. Remova o arquivo "App.java" da pasta "src.main.java ... ".
    b. Remova o arquivo "AppTest.java" da pasta "src.test.java ...".
    c. Faça download do arquivo "pom" disponibilizado no Moodle pelo professor e copie por cima do arquivo "pom" gerado pelo Maven. Este arquivo é o que contém as dependências do projeto. Entre as dependências definidas nesse arquivo estão aquelas necessárias para o uso do JUnit, tema da próxima unidade da nossa disciplina.
    d. Altere o arquivo "pom" de acordo com as informações do seu projeto (nome da organização, nome do artefato), pois o arquivo fornecido pelo professor

certamente não usa os mesmos nomes que você havia escolhido. Altere os campos “groupId” e “artefactId” com os valores que foram indicados nos passos 6 e 7. Altere também a tag “<mainclass>” indicando o nome da classe que irá conter o método “main”. Exemplo: “com.demo.App”.

e. Normalmente quando modificamos o “pom” o VSCode pergunta se deve levar em conta as modificações. Responda “Always”.

12) Copie os arquivos do projeto que consome o serviço de CEP para a pasta “src/main/java/<sua organização>”. Por exemplo, se você chamou sua organização de “com.demo” então copie os arquivos para a pasta “src/main/java/com/demo” .
13) Após copiar os arquivos como indicado no passo 12, você deve notar que eles indicam erro na primeira linha. É que está faltando indicar o “package”. Insira na primeira linha de cada um dos arquivos o comando “package com.demo” (ou substitua com.demo pelo nome da sua organização).
14) Se tudo foi feito corretamente, neste momento todos os erros do projeto devem desaparecer.
15) Para compilar o programa abra o terminal e use o comando: “mvn package”. Observe que antes de iniciar a compilação o Maven irá baixar todas a dependências necessárias para o projeto. Dependendo da velocidade da internet esse processo pode ser um pouco demorado, mas só irá ocorrer uma vez para cada equipamento onde o projeto for ser compilado.
16) Após a compilação o arquivo compilado (.jar) estará disponível na pasta “target” com o nome de “<nome_do_artefato>CSV-1.0-SNAPSHOT.jar” (onde <nome_do_artefato> é o nome que você indicou para o artefato no “pom”. Para executar este programa digite o comando que segue:

Java -jar .\target\<nome_do_artefato>-1.0-SNAPSHOT.jar

17) Pronto, seu primeiro projeto “maven” compila e executa corretamente.
18) Se em algum momento você quiser eliminar todos os arquivos compilados anteriormente use o comando “mvn clean”. Para limpar e recompilar “mvn clean package”.

Parte 2: usando a biblioteca Gson

Garanta ter conseguido executar a parte 1 deste roteiro sem erro antes de seguir a diante.

Na primeira parte do roteiro, aparentemente, desenvolvemos uma maneira “complicada” de compilar e executar um projeto que já compilava e executava dentro do VSCode sem a necessidade de um arquivo POM. Ocorre que a partir de agora fica muito mais fácil de acrescentar novas dependência no projeto e de garantir que todos os integrantes de uma equipe tenham as mesmas versões de todas as dependências em qualquer máquina que queiram trabalhar.

Para instalar e poder usar a biblioteca GSon no seu projeto, basta acrescentar a dependência que segue no arquivo “POM”:

```
<dependencies>
    <!--  Gson: Java to JSON conversion -->
    <dependency>
      <groupId>com.google.code.gson</groupId>
      <artifactId>gson</artifactId>
      <version>2.10.1</version>
      <scope>compile</scope>
    </dependency>
```

```
</dependencies>
```

A primeira fez que for solicitado ao Maven que compile o projeto (mvn package) ele irá baixar e instalar o GSon. Para saber como usar a biblioteca Gson para converter Strings JSON em objetos Java e vice-versa consulte: gson/UserGuide.md at master · google/gson (github.com)

Lembre: o objetivo é converter o String JSON recebido do serviço de CEP em uma instancia de classe Java. Para criar uma classe Java correspondente a uma String JSON pode-se usar https://app.quicktype.io/. Altere o programa de maneira que o String JSON seja convertido em uma instancia de classe Java e exiba as informações na tela a partir do método “toString” da classe Java.