

Lista Prática Sobre: Maven e Versionamento com Git

Orientações:

- Data de Entrega 27/02/2021 via Tarefa do Microsoft Teams
- Entregar apenas o link de um repositório git (GitHub/GitLab)

Objetivos da Lista:

- Aprimorar conhecimentos de versionamento com git
- Criar um repositório no GitHub/GitLab
- Familiarizar com o processo de automatização da *build* e gerência de dependências com Maven
- Familiarizar com recursos de *streams* do Java
- Conhecer as bibliotecas: OpenCSV e Lombok

Questão 1 Arquivos CSV (Comma-separated values) são arquivos muito utilizados para facilitar o envio, recebimento e armazenamento de um conjunto de dados. Nesses arquivos os dados ficam separados por “,” (por isso o nome). Várias bibliotecas que fazem análise estatística, usam arquivos CSV. Os arquivos CSV, normalmente, possuem a primeira linha como um cabeçalho dos dados armazenados.

Considere o exemplo de um CSV apresentado na Figura 1. É um arquivo com informações de alunos. A primeira linha é o cabeçalho com quais dados estão armazenados.

```
1 Nome,Idade,Curso
2 Capiroto,22,Eng Computacao
3 Capirota,23,Eng Software
4 Apolonia,18, Eng Producao
5 Solaire,26, Eng Biomedica
6 Astora,35, Eng Telecom
```

Figura 1:

Para converter esse arquivo em um objeto Java, podemos usar a biblioteca OpenCSV, que faz o mapeamento. Para isso precisamos criar uma classe modelo com os campos que desejamos fazer a leitura. A classe na Figura 2 é capaz de mapear o CSV mostrado na Figura 1.

Observe que existem membros da classe com o mesmo nome do cabeçalho do arquivo CSV. É necessário também criarmos *getters* e *setters*, mas a biblioteca Lombok cuida disso. Veja a anotação `@Data` na linha 5. Essa biblioteca precisa ser colocada como dependência para o Maven.

Para fazer a conversão usando o OpenCSV podemos usar o exemplo apresentado na Figura 3.

Para usar os recursos da biblioteca OpenCSV é necessário colocar como dependência para o Maven. Após a execução da linha 12 uma `List` será criada com objetos do tipo `Pessoa` com cada linha do CSV mapeado para um objeto. Com isso podemos manipular esses objetos e executar operações.

Tarefa:

Você deverá criar um programa em Java capaz de ler um arquivo CSV (vendas—games.csv, incluído na tarefa do teams) com informações de vendas de jogos digitais. E criar as seguintes funcionalidades:

```

1 package br.inatel.cdg.model;
2
3 import lombok.Data;
4
5 @Data
6 @AllArgsConstructor
7 public class Pessoa {
8
9     private String nome;
10    private int idade;
11    private String curso;
12
13 }

```

Figura 2: Classe para mapear um CSV

```

1 List<Pessoa> listaPessoas = new ArrayList<>();
2
3 //csvFilePath e o caminho onde se encontra o arquivo
4 //nesse exemplo e um instancia de Path do pacote java.nio
5 try {
6     Reader reader = Files.newBufferedReader(csvFilePath);
7     CsvToBean<Pessoa> csvToBean = new CsvToBeanBuilder(reader)
8         .withType(Pessoa.class)
9         .withIgnoreLeadingWhiteSpace(true)
10        .build();
11
12    listaPessoas = csvToBean.parse();
13 } catch (IOException e) {
14     e.printStackTrace();
15 }
16 return listaPessoas;

```

Figura 3: Exemplo de Conversão de um CSV para um Java Pojo (Plain old Java Objects).

1. Filtrar e criar um *ArrayList* (ou coleção equivalente) buscando jogos de plataformas específicas. Exemplo: Filtrar no CSV jogos para PS4, ou para PC.
2. Filtrar e cirar um *ArrayList* (ou coleção equivalente) buscando jogos de *publishers* específicas. Exemplo: Filtrar no CSV jogos publicados pela Activision, ou Sony.
3. Configurar o Maven para gerar o jar executável

Dicas:

- Pesquisar sobre as bibliotecas Lombok e OpenCSV
- Estudar a função *filter()* do Java
- Verificar como carregar arquivos da pasta *resources* presente no Maven (facilita empacotar arquivos de configuração)
- Criar enums para os tipos de Plataformas e Publishers