Aula 19 - Arquivos

A linguagem JAVA nos permite manipular arquivos durante o nosso desenvolvimento. Essa manipulação inclui leitura, escrita e criação de novos arquivos.

1. Leitura de arquivo de forma simples

Para leitura de arquivos de forma simples, podemos usar a classe File.

Exemplo:

```
package AulaArquivo;

import java.io.File;
import java.io.IOException;
import java.util.Scanner;

public class Programa {

    public static void main(String[] args) {
        File arquivo = new File("C:\\temp\\arquivo01.txt");
        Scanner sc = null;

        try {
            sc = new Scanner(arquivo);
            while(sc.hasNextLine()) {
                System.out.println(sc.nextLine());
            }
        }
        catch(IOException e) {
            System.out.println("Erro ao ler o arquivo - " +
e.getMessage());
        }
        finally {
            if (sc != null)
                sc.close();
        }
    }
}
```

2. Leitura de arquivo de forma otimizada

Para leitura de arquivos de forma otimizada, devemos utilizar as classes **FileReader** e **BufferedReader**. Ambas são complementares e são utilizadas para leitura de arquivos utilizando o buffer da memória do computador.

Exemplo:

```
package AulaArquivo;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.util.Scanner;

public class Programa {

    public static void main(String[] args) {

        String path = "C:\\temp\\arquivo01.txt";

        FileReader fr = null;
        BufferedReader br = null;

        try {
            fr = new FileReader(path);
            br = new BufferedReader(fr);

            String line = br.readLine();

            while(line != null){
                System.out.println(line);
                line = br.readLine();
            }
        }
        catch(IOException e) {
            System.out.println("Erro ao ler o arquivo - " +
e.getMessage());
        }
        finally {
            try{
                if (br != null)
                    br.close();

                if (fr != null)
                    fr.close();
            }
            catch(IOException e) {
```

```
                System.out.println("Erro ao escrever no arquivo - " +
e.getMessage());
            }
        }
    }
}
```

É possível simplificar a nossa programação se utilizarmos o recurso **try with resources**.

Esse recurso deve ser utilizado nos ajuda na visibilidade do contexto dos objetos fr e br, pois ambos são instanciados no início do bloco try..catch.

Exemplo:

```
package AulaArquivo;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.util.Scanner;

public class Programa {

    public static void main(String[] args) {

        String path = "C:\\temp\\arquivo01.txt";

        try (BufferedReader br = new BufferedReader(new
FileReader(path))) {
            // Obtem a primeira linha do arquivo
            String line = br.readLine();

            // Percorre o arquivo
            while(line != null){
                System.out.println(line);
                line = br.readLine();
            }
        }
        catch(IOException e) {
            System.out.println("Erro ao escrever no arquivo - " +
e.getMessage());
        }
    }
```

```
}
```

3. Escrita de arquivo de forma otimizada

Para escrita de arquivos de forma otimizada, devemos utilizar as classes **FileWriter** e **BufferedWriter**. Ambas são complementares e são utilizadas para escrita de arquivos utilizando o buffer da memória do computador.

O FileWriter, possui duas formas de uso:

new FileWriter(path): Cria ou recria (zerado) o arquivo conforme o path;

new FileWriter(path, true): Escreve mais conteúdo em um arquivo existente;

Exemplo:

```
package AulaArquivo;

import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;

public class Programa {

    public static void main(String[] args) {

        String[] lines = new String[] {"Turma", "Java", "Top"};

        String path = "C:\\temp\\arquivo02.txt";

        // Acrescenta mais linhas no arquivo
        //try (BufferedWriter bw = new BufferedWriter(new
FileWriter(path, true))) {

        // Cria o recria o arquivo
        try (BufferedWriter bw = new BufferedWriter(new
FileWriter(path))) {
            for(String line: lines) {
                // escreve a linha no arquivo
                bw.write(line);
                // pula uma linha no arquivo
                bw.newLine();
            }
        }
```

```
        catch(IOException e) {
            e.printStackTrace();
        }
    }
}
```

Exercícios

Crie um programa que solicite ao usuário uma lista de compras. O programa deve solicitar 10 produtos, contendo nome, quantidade e preço unitário. Todas essas informações devem ser gravadas em um arquivo chamado lista.txt, separadas por vírgula.

Exemplo - Arquivo lista.txt:

Arroz,1,21.90
Feijao,3,8.50
Pizza,2,14.90

Criar tratamentos e validações no programa.

```
package ExListaComprasArquivo;
import java.util.ArrayList;
import java.util.Scanner;
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;

public class Programa {
   public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String nome, linhaProduto;
        double qtd, preco;
        String path = "C:\\temp\\lista.txt";

        ArrayList<String> listaProdutos = new ArrayList<>();

        for (int i=1; i<=3; i++) {
            System.out.print("Digite o nome do " + i + "° produto:
");
            nome = sc.next();

            System.out.print("Digite a quantidade do " + i + "°
produto: ");
            qtd = sc.nextDouble();
```

```
            System.out.print("Digite o preço do " + i + "° produto:
");
            preco = sc.nextDouble();

            linhaProduto = nome + "," + qtd + "," + preco;

            listaProdutos.add(linhaProduto);
        }

        try (BufferedWriter bw = new BufferedWriter(new
FileWriter(path))) {
            for (String produto : listaProdutos) {
                bw.write(produto);
                bw.newLine();
            }
        }
        catch(IOException e) {
            e.printStackTrace();
        }

        sc.close();
    }
}
```

A partir do programa acima, crie um programa que leia o arquivo lista.txt e calcule o preço total de cada item multiplicando quantidade e preço, e escreva o total em um novo arquivo chamado total.txt.

Exemplo:

Arquivo: lista.txt
Arroz,1,21.90
Feijao,3,8.50
Pizza,2,14.90

O arquivo total.txt deverá ter:
Arroz,21.90
Feijao,25.50
Pizza,29.80

Ambos os arquivos devem estar na mesma pasta.

```
package ExTotalComprasArquivo;
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.BufferedReader;
import java.io.FileReader;
```

```java
import java.io.IOException;
import java.util.ArrayList;

public class Programa {
   public static void main(String[] args) {
       String[] infosProduto;
       double total;
       String linhaTotal;
       String path = "C:\\temp\\lista.txt";
       String pathNovo = "C:\\temp\\total.txt";

       ArrayList<String> linhasTotal = new ArrayList<String>();

       try (BufferedReader br = new BufferedReader(new
FileReader(path))){

           String line = br.readLine();

           while(line != null){
               infosProduto = line.split(",");

               total = Double.parseDouble(infosProduto[1]) *
Double.parseDouble(infosProduto[2]);

               linhaTotal = infosProduto[0] + "," + total;

               linhasTotal.add(linhaTotal);

               line = br.readLine();
           }
       }
       catch(IOException e) {
           System.out.println("Erro ao ler o arquivo - " +
e.getMessage());
       }

       try (BufferedWriter bw = new BufferedWriter(new
FileWriter(pathNovo))) {
           for (String produto : linhasTotal) {
               bw.write(produto);
               bw.newLine();
           }
       }
       catch(IOException e) {
           e.printStackTrace();
       }
   }
}
```