ESTRITAMENTE **CONFIDENCIAL**

ELABORATA
INFORMÁTICA

LINGUAGEM JAVA
Módulo 2

Entrada/Saída

Tópicos

- Introdução ao pacote IO Java
- Fluxos de bytes
- Fluxos de caracteres
- Fluxos pré-definidos
- Ler e gravar arquivos

Apresentação do Instrutor

O Instrutor

- Nome
- Formação acadêmica
- Experiências profissionais
- Principais projetos

Apresentação do Curso

O Curso

- Objetivo
- Conteúdo
- Divisão
- Metodologia

ELABORATA INFORMÁTICA

Introdução ao pacote IO Java

Sistema IO Java

- O sistema de I/O Java é bem grande, contendo muitas classes, interfaces e métodos. Parte da razão de seu tamanho é que Java define dois sistemas de I/O completos: um para I/O de bytes e outro para I/O de caracteres.
- Serão apresentados os recursos mais usados e importantes. Felizmente os sistema de I/O Java é coeso e coerente; uma vez que você entenda os aspectos básicos, o resto será fácil de dominar.

Sistema IO é baseado em fluxos

- Os programas Java executam I/O por intermédio de fluxos. Uma fluxo é uma abstração que produz ou consome informações. Ele é vinculado a um dispositivo físico pelo sistema I/O de Java.
- Todos os fluxos se comportam igualmente, mesmo que os dispositivos físicos aos quais estejam vinculados sejam diferentes.
- Os mesmos métodos usados para a gravação no console também podem ser usados na gravação em um arquivo em disco.

ELABORATA INFORMÁTICA

Fluxos

Fluxos de bytes e Fluxos de caracteres

- Versões modernas de Java definem dois tipos de fluxos: de bytes e de caracteres.
- Os fluxos de bytes fornecem um meio conveniente para o tratamento de entrada e saída de bytes. Eles são usados, por exemplo, na leitura ou gravação de dados binários. São especialmente úteis no trabalho com arquivos.
- Os fluxos de caracteres foram projetados para o tratamento da entrada e saída de caracteres. Eles usam o Unicode e, portanto podem ser internacionalizados.
- Em alguns casos, os fluxos de caracteres são mais eficientes do que os fluxos de bytes.

Fluxos de bytes e Fluxos de caracteres

- O fato de Java definir dois tipos de fluxos diferentes aumenta e muito o sistema de I/O, porque dois conjuntos de hierarquias de classes separados são necessários.

- O grande número de classes pode fazer o sistema de I/O parecer mais assustados do que realmente é.

- No nível mais baixo, todo o I/O continua orientado a bytes. Os fluxos baseados em caracteres apenas fornecem um meio conveniente e eficiente de tratamento de caracteres.

ELABORATA
INFORMÁTICA

Fluxos de Bytes

Classes de fluxos de bytes

- Os fluxos de bytes são definidos com uso de duas hierarquias de classes. No topo delas estão duas classes abstratas: **InputStream** e **OutputStream**.

- **InputStream** define as características comuns a fluxos de entrada de bytes

- **OutputStream** descreve o comportamento dos fluxos de saída de bytes.

Classes de fluxos de bytes

- A partir do **InputStream** e **OutputStream**, são criadas muitas subclasses concretas que oferecem funcionalidade variada e tratam os detalhes de leitura e gravação em vários dispositivos, como arquivos em disco.

ELABORATA
INFORMÁTICA

Fluxos de caracteres

Classes de fluxos de caracteres

- Os fluxos de caracteres são definidos com uso de duas hierarquias de classes encabeçadas pelas seguintes duas classes abstratas: **Reader** e **Writer**.

- **Reader** é usada para entrada.

- **Writer** para saída.

Classes de fluxos de caracteres

- As classes concretas derivadas de **Reader** e **Writer** operam com fluxos de caracteres Unicode.

- De **Reader** e **Writer** são derivadas muitas subclasses concretas que tratam várias situações de I/O.

- Em geral, as classes baseadas em **caracteres** são equivalentes às classes baseadas em **bytes**.

Fluxos Pré-definidos

Classe System

- Os programas Java importam automaticamente o pacote **java.lang**.
- Esse pacote define uma classe chamada **System**, que encapsula vários aspectos do ambiente de tempo de execução.
- Entre outras coisas, ela contém três variáveis de fluxos predefinidas, chamadas **in**, **out** e **err**.
- Esses campos são declarados como **public**, **final** e **static** dentro de **System**, ou seja, podem ser usados por qualquer parte do programa e sem referência a um objeto **System** específico.

Classe System

- **System.out** é o fluxo de saída básico por padrão, ele usa o console.

- **System.in** é a entrada básica, que por padrão é o teclado.

- **System.err** é o fluxo de erro básico, que por padrão também usa o console.

Classe System

- No entanto, esses fluxos podem ser redirecionados para qualquer dispositivo de I/O compatível.

- **System.in** é um objeto de tipo **InputStream.**

- **System.out** e **System.err** são objetos de tipo **PrintStream**.

- São fluxos de bytes, mesmo que normalmente sejam usados na leitura e gravação de caracteres no console.

ELABORATA
INFORMÁTICA

Arquivos texto

Arquivos texto

- Arquivos textos são arquivos armazenados num dispositivo de armazenamento e podem ser lidos facilmente por seres humanos.

Tipos

- Texto puro
- Texto delimitado por caractere

Criando arquivos texto

Criando arquivo texto

- Para criar um arquivo texto contendo texto puro usando a Linguagem Java podemos utilizar as classes **FileWriter** e **BufferedWriter**.

Criando arquivo texto

```
package br.com.elaborata.io.escrita;

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

/**
 * @author professor
 *
 */
public class ArquivoTextoEscrita {
public static void main(String[] args) {

FileWriter fw = null;
BufferedWriter out = null;
...
```

Criando arquivo texto

```
...
try {
        fw = new FileWriter("dadoscurso.txt");
        out = new BufferedWriter(fw);

        out.write("Curso Linguagem Java Avançado\n");
        out.write("60 Horas\n");
        out.write("Período noturno");
        out.flush();
} catch (IOException e) {
        e.printStackTrace();
}
...
```

Criando arquivo texto

```
...
finally {
        try {
                out.close();
                fw.close();
        } catch (IOException e) {
                e.printStackTrace();
        }
    }
  }
}
```

Criando arquivo texto CSV

ELABORATA INFORMÁTICA

```
package br.com.elaborata.io.escrita;

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

/**
 * @author professor
 *
 */
public class ArquivoTextoEscrita {
public static void main(String[] args) {

FileWriter fw = null;
BufferedWriter out = null;
...
```

Criando arquivo texto CSV

```
try {
        fw = new FileWriter("dadoscurso.csv");
        out = new BufferedWriter(fw);

        StringBuilder sb = new StringBuilder();
        sb.append("Curso Aplicações Linguagem Java");
        sb.append(",");
        sb.append("60 Horas");
        sb.append(",");
        sb.append("Período noturno");
        out.write(sb.toString());

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

Criando arquivo texto CSV

```
...
finally {
        try {
                out.close();
                fw.close();
        } catch (IOException e) {
                e.printStackTrace();
        }
    }
  }
}
```

ELABORATA INFORMÁTICA

Lendo arquivos texto

Lendo arquivo texto

- Para ler o conteúdo de um arquivo texto contendo texto puro usando a Linguagem Java é necessário utilizar as classes **FileReader** e **BufferedReader**.

Lendo arquivo texto

```
package br.com.elaborata.io.leitura;

import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;

/**
 * @author professor
 *
 */
public class ArquivoTextoLeitura {
public static void main(String[] args) {

FileReader fr = null;
BufferedReader in = null;
...
```

Lendo arquivo texto

```
try {
        fr = new FileReader("dadoscurso.txt");
        in = new BufferedReader(fr);
        String linha;
        do {
                linha = in.readLine();
                if (linha != null) {
                        System.out.println(linha);
                }
        } while (linha != null);
} catch (FileNotFoundException e) {
        e.printStackTrace();
} catch (IOException e) {
        e.printStackTrace();
}
```

Lendo arquivo texto

```
...
finally {
        try {
                out.close();
                fw.close();
        } catch (IOException e) {
                e.printStackTrace();
        }
    }
  }
}
```

Lendo arquivo texto CSV

```
package br.com.elaborata.io.leitura;

import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;

/**
 * @author professor
 *
 */
public class ArquivoTextoLeitura {
public static void main(String[] args) {

FileReader fr = null;
BufferedReader in = null;
...
```

Lendo arquivo texto CSV

```
try {
        fr = new FileReader("dadoscurso.csv");
        in = new BufferedReader(fr);
        String linha;
        String dados[];
        do {
                linha = in.readLine();
                if (linha != null) {
                        dados = linha.split(",");
                        System.out.println(dados[0]);
                        System.out.println(dados[1]);
                        System.out.println(dados[2]);
                }
        } while (linha != null);
} catch (FileNotFoundException e) {
        e.printStackTrace();
} catch (IOException e) {
        e.printStackTrace();
}
```

Lendo arquivo texto CSV

```
...
finally {
        try {
                out.close();
                fw.close();
        } catch (IOException e) {
                e.printStackTrace();
        }
    }
  }
}
```

Dúvidas

OBRIGADO!

ESTRITAMENTE **CONFIDENCIAL**

ELABORATA
INFORMÁTICA

www.**elaborata**.com.br

Horário de Atendimento Comercial
Segunda à sexta – das 9:00h às 19:30h e
Sábado - das 8:00h às 15:00h.

Rua Monsenhor Celso, 256 - 1º Andar
Centro - Curitiba - PR

41.3324.0015
41.99828.2468

cursos@elaborata.com.br