

## Sumário de Hoje

- Modelagem de Programas Orientada a Objetos
- Introdução a Padrões de Projeto (Design Patterns)
- Introdução a Ambientes Integrados de Desenvolvimento

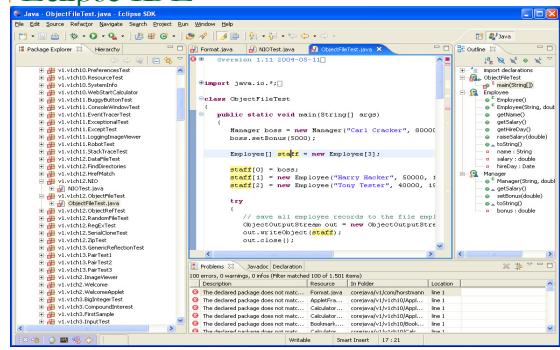
Paulo André Castro

POO

ITA - Stefanini

83

## Eclipse IDE



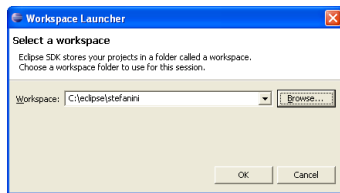
Paulo André Castro

POO

ITA - Stefanini

84

## Criando Workspace



Paulo André Castro

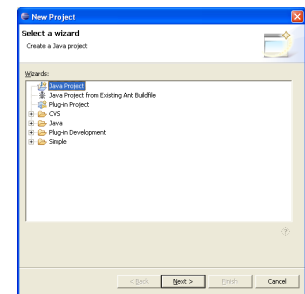
POO

ITA - Stefanini

85

## Criando projetos

- Menu File | New | Project
- Java Project
- A partir de código pré-existente arquivos ant
- CVS
- Java
  - Java Project
  - Java Project from ..Ant
- Plug-In Development



Paulo André Castro

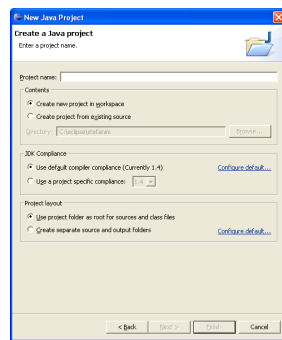
POO

ITA - Stefanini

86

## Criando Projetos – Passo 2

- Escolha
  - Nome do Projeto
  - Projeto vazio ou criado a partir de código pré-existente
  - JDK alvo
  - Project Layout



Paulo André Castro

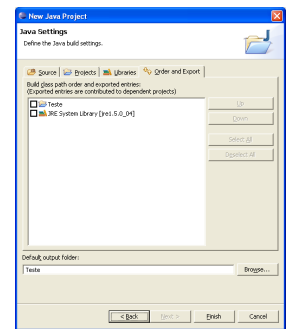
POO

ITA - Stefanini

87

## Configuração do Projeto

- Escolha dos diretórios com código-fonte
- Bibliotecas utilizadas
- Projetos requeridos
- Ordem de importação/exportação



Paulo André Castro

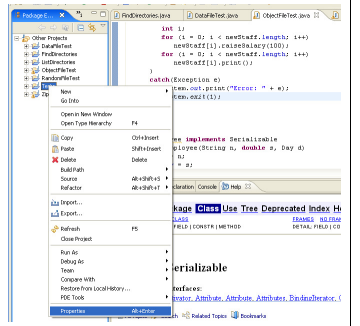
POO

ITA - Stefanini

88

## Opções de Projeto

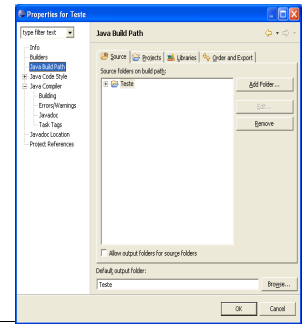
- Criação de elementos do Projeto
  - Classes, interfaces, etc.
- Refactoring
- Propriedades, etc.



Paulo André Castro POO ITA - Stefanini 89

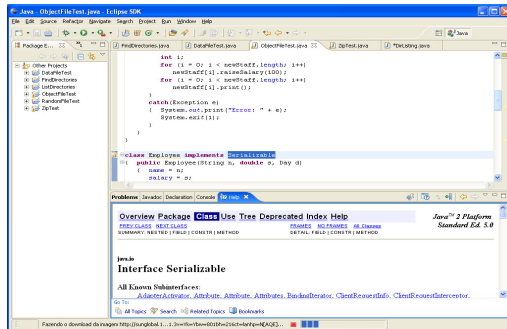
## Propriedades do Projeto

- Configuração do Classpath e acesso a bibliotecas
- Configuração de diretórios de destino e fonte
- Configuração de destino do Javadoc
- Referências a outros projetos, etc.



Paulo André Castro POO ITA - Stefanini 90

## Alguns Exemplos de Programas



Paulo André Castro POO ITA - Stefanini 91

## Alguns Exemplos de Programa

- Exemplo 1:
 

```
public class Hello {
    public static void main(String args[])
    { System.out.println("Hello World!");
    }
}
```
- Compile e Execute o programa acima através do Eclipse

Paulo André Castro POO ITA - Stefanini 92

## Exemplo 2

- File: ShowTwoArgs.java

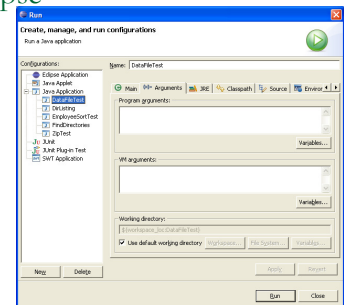
```
public class ShowTwoArgs {
    public static void main(String[] args) {
        System.out.println("First arg: " +
            args[0]);
        System.out.println("Second arg: " +
            args[1]);
    }
}
```

Compile e Execute o programa acima

Paulo André Castro POO ITA - Stefanini 93

## Inserindo argumentos para os programas através do Eclipse

- Menu Run | Run ...
- Tab Arguments



Paulo André Castro POO ITA - Stefanini 94

## Exemplo 3 – Usando Loops

```
public class ShowArgs {
    public static void main(String[] args) {
        for(int i=0; i<args.length; i++) {
            System.out.println("Arg " + i +
                " is " +
                args[i]);
        }
    }
}
```

Paulo André Castro

POO

ITA - Stefanini

95

## Exemplo 4 – Loops Aninhados

```
public class TriangleArray {
    public static void main(String[] args) {
        int[][] triangle = new int[10][1];
        for(int i=0; i<triangle.length; i++) {
            triangle[i] = new int[i+1];
        }
        for (int i=0; i<triangle.length; i++) {
            for(int j=0; j<triangle[i].length; j++) {
                System.out.print(triangle[i][j]);
            }
            System.out.println();
        }
    }
}
```

Paulo André Castro

POO

ITA - Stefanini

96

## Exemplo 5

```
class Ship3 {
    public double x, y, speed, direction;
    public String name;

    public Ship3(double x, double y,
        double speed, double direction,
        String name) {
        this.x = x; // "this" differentiates instance vars
        this.y = y; // from local vars.
        this.speed = speed;
        this.direction = direction;
        this.name = name;
    }

    private double degreesToRadians(double degrees) {
        return(degrees * Math.PI / 180.0);
    }
}
```

Paulo André Castro

POO

ITA - Stefanini

97

## Exemplo 5

```
public void move() {
    double angle = degreesToRadians(direction);
    x = x + speed * Math.cos(angle);
    y = y + speed * Math.sin(angle);
}

public void printLocation() {
    System.out.println(name + " is at ("
        + x + ", " + y + ").");
}

public class Test3 {
    public static void main(String[] args) {
        Ship3 s1 = new Ship3(0.0, 0.0, 1.0, 0.0, "Ship1");
        Ship3 s2 = new Ship3(0.0, 0.0, 2.0, 135.0, "Ship2");
        s1.move();
        s2.move();
        s1.printLocation();
        s2.printLocation();
    }
}
```

Paulo André Castro

POO

ITA - Stefanini

98

## Saída do Exemplo 5

- **Compiling and Running:**  
javac Test3.java  
java Test3
- **Output:**  
Ship1 is at (1,0).  
Ship2 is at (-1.41421,1.41421).

Paulo André Castro

POO

ITA - Stefanini

99

## Onde estão as classes ? CLASSPATH

- The **CLASSPATH** environment variable defines a list of directories in which to look for classes
  - Default = current directory and system libraries
  - Best practice is to not set this when first learning Java!
- **Setting the CLASSPATH**  
set CLASSPATH = .;C:\java\D:\cwp\echoserver.jar  
setenv CLASSPATH .:~/java:/home/cwp/classes/
  - The period indicates the current working directory
- **Supplying a CLASSPATH**  
javac -classpath .;D:\cwp WebClient.java  
java -classpath .;D:\cwp WebClient

Paulo André Castro

POO

ITA - Stefanini

100

## Pacotes...

- A **package** lets you group classes in subdirectories to **avoid accidental name conflicts**
  - To create a package:
    1. Create a subdirectory with the same name as the desired package and place the source files in that directory
    2. Add a package statement to each file

```
package packagename;
```
    3. Files in the main directory that want to use the package should include

```
import packagename.*;
```
- The package statement must be the **first statement** in the file
- If a package statement is omitted from a file, then the code is part of the default package that has no name

Paulo André Castro

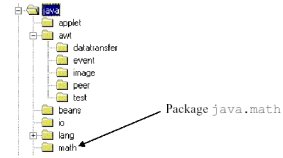
POO

ITA - Stefanini

101

## Pacotes...

- The package hierarchy reflects the file system directory structure



- The root of any package must be accessible through a Java system default directory or through the CLASSPATH environment variable

Paulo André Castro

POO

ITA - Stefanini

102

## Exercício

- Criar packages hello, triangle, showArgs e ship
- Mover programas correspondentes
- Definir variáveis de instância de Ship como private e criar métodos de acesso

Paulo André Castro

POO

ITA - Stefanini

103

## Mais sobre modificadores (métodos e variáveis)

- **public**: o método ou variável ao qual se refere é acessível de "qualquer lugar" no código
  - Uma classe deve ser declarada public para ser acessível por outras classes
  - Uma classe pública deve estar declarada num arquivo com o mesmo nome da classe. Ex. " public class Ship ..." deve estar no arquivo Ship.java
- **private**: O método ou variável ao qual se refere é acessível exclusivamente por métodos da mesma classe
  - Declarar uma variável private a faz acessível pelo resto do código apenas através de métodos públicos

Paulo André Castro

POO

ITA - Stefanini

104

## Mais sobre modificadores (métodos e variáveis)

- **protected**: Acessível apenas a variáveis e métodos da classe, das classes filhas (herdadas) e das classes que pertencem ao mesmo pacote
  - Variáveis e métodos protected são herdados mesmo por classes que não pertencem ao mesmo pacote
- [default]: Similar ao protected, exceto por:
  - Variáveis e métodos [default] NÃO são herdados por classes que não pertencem ao mesmo pacote
  - Em outras palavras: Variáveis e métodos [default] são herdadas APENAS por classes que pertencem ao mesmo pacote

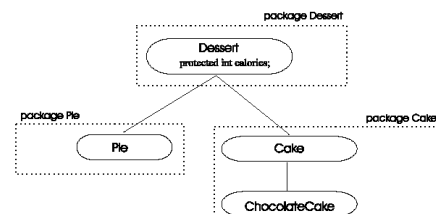
Paulo André Castro

POO

ITA - Stefanini

105

## Exemplo - protected



- Cake, ChocolateCake e Pie herdam o campo calories
- Entretanto, se o código na classe Cake tem uma referência ao objeto Pie, o campo calories de Pie não pode ser acessado em Cake.
- Campos protected de uma classe não podem ser acessados fora de um mesmo pacote, exceto se na mesma árvore de hierarquia

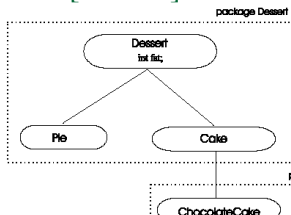
Paulo André Castro

POO

ITA - Stefanini

106

## Exemplo – [default]



- Even through inheritance, the fat data field cannot cross the package boundary
  - Thus, the fat data field is accessible through any Dessert, Pie, and Cake object within any code in the Dessert package
  - However, the ChocolateCake class does not have a fat data field, nor can the fat data field of a Dessert, Cake, or Pie object be accessed from code in the ChocolateCake class

Paulo André Castro

POO

ITA - Stefanini

107

## Sumário de modificadores de acesso

Data Fields and Methods	Modifiers			
	public	protected	default	private
Accessible from same class?	yes	yes	yes	yes
Accessible to classes (nonsubclass) from the same package?	yes	yes	yes	no
Accessible to subclass from the same package?	yes	yes	yes	no
Accessible to classes (nonsubclass) from different package?	yes	no	no	no
Accessible to subclasses from different package?	yes	no	no	no
Inherited by subclass in the same package?	yes	yes	yes	no
Inherited by subclass in different package?	yes	yes	no	no

Paulo André Castro

POO

ITA - Stefanini

108

## Outros modificadores

- **final**
  - For a class, indicates that it cannot be subclassed
  - For a method or variable, cannot be changed at runtime or overridden in subclasses
- **synchronized**
  - Sets a lock on a section of code or method
  - Only one thread can access the same synchronized code at any given time
- **transient**
  - Variables are not stored in serialized objects sent over the network or stored to disk
- **native**
  - Indicates that the method is implemented using C or C++

Paulo André Castro

POO

ITA - Stefanini

109

## Algumas Diretrizes para gerar bom código

- Uma classe deve o menor número possível de métodos públicos (mas deve ter pelo menos um!)
  - Isto diminui o acoplamento entre as classes do projeto, o que facilita a manutenção
- Deve-se evitar variáveis públicas. Crie métodos de acesso get/set. Exemplo:

```

class Ship {
    private double speed;
    public double getSpeed() { return speed; }
    public void setSpeed(double speed) {
        this.speed=speed;
    }
}
    
```

Paulo André Castro

POO

ITA - Stefanini

110

## Sumário de Hoje

- Introdução ao Ambiente Eclipse
  - Criando workspaces e projetos
  - Compilando e executando programas
- Desenvolvimento de Programas básicos (modo texto)
  - Primeiros Programas
  - Javadoc, Os conceitos de CLASSPATH, package e import
- O sistema de I/O Orientado a Objetos do Java
  - Acessando arquivos Texto
  - Acessando arquivos Binários
  - Serialização e armazenamento de Objetos

Paulo André Castro

POO

ITA - Stefanini

111

## O sistema de IO em Java

- A biblioteca java.io tem mais de 60 classes(stream) de input/output
- Dois grandes grupos
  - Classes baseadas em tráfego de bytes
    - DataStreams:
  - Classes baseados em tráfego de caracteres
    - Reader e Writer
- Em qualquer operação de IO pode ocorrer uma exceção do tipo IOException

Paulo André Castro

POO

ITA - Stefanini

112

## A classe File

- A **File** object can refer to either a file or a directory

```
File file1 = new File("data.txt");
File file1 = new File("C:\\java");
```

- To obtain the path to the current working directory use

```
System.getProperty("user.dir");
```

- To obtain the file or path separator use

```
System.getProperty("file.separator");
System.getProperty("path.separator");
```

or

```
File.separator()
File.pathSeparator()
```

Paulo André Castro

POO

ITA - Stefanini

113

## Métodos úteis em File

- **isFile/isDirectory**
- **canRead/canWrite**
- **length**
  - Length of the file in bytes (long) or 0 if nonexistent
- **list**
  - If the File object is a **directory**, returns a **String array** of all the files and directories contained in the directory; otherwise, **null**
- **mkdir**
  - Creates a new subdirectory
- **delete**
  - Deletes the directory and returns **true** if successful
- **toURL**
  - Converts the file path to a URL object

Paulo André Castro

POO

ITA - Stefanini

114

## Exemplo de File: Programa que lista o diretório do usuário

```
import java.io.*;

public class DirListing {
    public static void main(String[] args) {

        File dir = new File(System.getProperty("user.dir"));

        if(dir.isDirectory()){
            System.out.println("Directory of " + dir);
            String[] listing = dir.list();
            for(int i=0; i<listing.length; i++) {
                System.out.println("\t" + listing[i]);
            }
        }
    }
}
```

Paulo André Castro

POO

ITA - Stefanini

115

## Resultado

```
> java DirListing
Directory of C:\java\
DirListing.class
DirListing.java
test
TryCatchExample.class
TryCatchExample.java
XslTransformer.class
XslTransformer.java
```

Paulo André Castro

POO

ITA - Stefanini

116

## Exercício: Listar o conteúdo de um diretório

- Faça um programa em Java que liste o conteúdo de um diretório passado na linha de comando ou o diretório corrente, caso não seja solicitado nenhum
- Exemplos de uso:
  - C:>eclipse\java ListDir "c:\Arquivos de Programa"
    - Lista o conteúdo do diretório Arquivos de Programa
  - C:>eclipse\java ListDir
    - Lista o conteúdo do diretório eclipse:

Paulo André Castro

POO

ITA - Stefanini

117

## Solução

```
import java.io.*;

public class DirListing {
    public static void main(String []args) {

        if(args.length<1) {
            args=new String[2];
            args[0]=System.getProperty("user.dir");
        }
        File dir = new File(args[0]);
        if(dir.isDirectory()) {
            String []list= dir.list();
            for(int i=0;i<list.length;i++)
                System.out.println("\t"+list[i]);
        }
        else
            System.out.println(args[0]+" não é um diretorio");
    }
}
```

Paulo André Castro

POO

ITA - Stefanini

118

## Resultado

Paulo André Castro POO ITA - Stefanini 119

## Classes para escrever Texto

Desired ...	Methods	Construction
Character File Output	<b>FileWriter</b> write(int char) write(byte[] buffer) write(String str)	File file = new File("filename"); FileWriter fout = new FileWriter(file); or FileWriter fout = new FileWriter("filename");
Buffered Character File Output	<b>BufferedWriter</b> write(int char) write(char[] buffer) write(String str) newLine()	File file = new File("filename"); FileWriter fout = new FileWriter(file); BufferedWriter bout = new BufferedWriter(fout); or BufferedWriter bout = new BufferedWriter(new File("filename"));

Paulo André Castro POO ITA - Stefanini 120

## Classes para escrever Texto

Desired ...	Methods	Construction
Character Output	<b>PrintWriter</b> write(int char) write(char[] buffer) write(String str) print( ... ) println( ... )	FileWriter fout = new FileWriter("filename"); PrintWriter pout = new PrintWriter(fout); or PrintWriter pout = new PrintWriter(new FileWriter("filename")); or PrintWriter pout = new PrintWriter(new BufferedWriter(new FileWriter("filename")));

Paulo André Castro POO ITA - Stefanini 121

## Exemplo de Escrita de Arquivo de Texto

```
import java.io.*;

public class CharacterFileOutput {
    public static void main(String[] args) {
        FileWriter out = null;

        try {
            out = new FileWriter("book.txt");
            System.out.println("Encoding: " + out.getEncoding());
            out.write("Core Web Programming");
            out.close();
            out = null;
        } catch (IOException ioe) {
            System.out.println("IO problem: " + ioe);
            ioe.printStackTrace();
            try {
                if (out != null) {
                    out.close();
                }
            } catch (IOException ioe2) { }
        }
    }
}
```

Paulo André Castro POO ITA - Stefanini 122

## Codificação de caracteres

```
> java CharacterFileOutput
Encoding: Cp1252

> type book.txt
Core Web Programming
```

- **Note: Cp1252 is Windows Western Europe / Latin-1**
  - To change the system default encoding use `System.setProperty("file.encoding", "encoding");`
  - To specify the encoding when creating the output stream, use an `OutputStreamWriter`

```
OutputStreamWriter out =
    new OutputStreamWriter(
        new FileOutputStream("book.txt", "8859_1"));
```

Paulo André Castro POO ITA - Stefanini 123

## Classes de Leitura de Streams de Texto

Desired ...	Methods	Construction
Character File Input	<b>FileReader</b> read() read(char[] buffer)	File file = new File("filename"); FileReader fin = new FileReader(file); or FileReader fin = new FileReader("filename");
Buffered Character File Input	<b>BufferedReader</b> read() read(char[] buffer) readLine()	File file = new File("filename"); FileReader fin = new FileReader(file); BufferedReader bin = new BufferedReader(fin); or BufferedReader bin = new BufferedReader(new FileReader(new File("filename")));

Paulo André Castro POO ITA - Stefanini 124

## Exemplo - FileReader

```
import java.io.*;

public class CharacterFileInput {
    public static void main(String[] args) {
        File file = new File("book.txt");
        FileReader in = null;

        if(file.exists()) {
            try {
                in = new FileReader(file);
                System.out.println("Encoding: " + in.getEncoding());
                char[] buffer = new char[(int)file.length()];
                in.read(buffer);
                System.out.println(buffer);
                in.close();
            } catch (IOException ioe) {
                System.out.println("IO problem: " + ioe);
                ioe.printStackTrace();
            }
        }
    }
}
```

Paulo André Castro

POO

ITA - Stefanini

126

## Resultado - FileReader

```
> java CharacterFileInput
Encoding: Cp1252
Core Web Programming
```

- Alternatively, could read file one line at a time:

```
BufferedReader in =
    new BufferedReader(new FileReader(file));
String lineIn;
while ((lineIn = in.readLine()) != null) {
    System.out.println(lineIn);
}
```

Paulo André Castro

POO

ITA - Stefanini

127

## I/O em Arquivos (Streams) Binários

- Handle byte-based I/O using a **DataInputStream** or **DataOutputStream**

Data Type	DataInputStream	DataOutputStream
byte	readByte	writeByte
short	readShort	writeShort
int	readInt	writeInt
long	readLong	writeLong
float	readFloat	writeFloat
double	readDouble	writeDouble
boolean	readBoolean	writeBoolean
char	readChar	writeChar
String	readUTF	writeUTF
byte[]	readFully	

- The `readFully` method blocks until all bytes are read or an EOF occurs
- Values are written in big-endian fashion regardless of computer platform

Paulo André Castro

POO

ITA - Stefanini

128

## Classes para Escrita em Streams Binárias

Desired ...	Methods	Construction
Binary File Output bytes	<b>FileOutputStream</b> write(byte) write(byte[] buffer)	File file = new File("filename"); FileOutputStream fout = new FileOutputStream(file); or FileOutputStream fout = new FileOutputStream("filename");
Binary File Output byte short int long float double char boolean	<b>DataOutputStream</b> write(byte) write(short) write(int) write(long) write(float) write(double) write(char) write(boolean) writeUTF(string) writeBytes(string) writeChars(string)	File file = new File("filename"); FileOutputStream fout = new FileOutputStream(file); DataOutputStream dout = new DataOutputStream(fout); or DataOutputStream dout = new DataOutputStream( new FileOutputStream( new File("filename")));

Paulo André Castro

POO

ITA - Stefanini

129

## Classes para Escrita em Streams Binárias

Desired ...	Methods	Construction
Buffered Binary File Output	<b>BufferedOutputStream</b> flush() write(byte) write(byte[] buffer, int off, int len)	File file = new File("filename"); FileOutputStream fout = new FileOutputStream(file); BufferedOutputStream bout = new BufferedOutputStream(fout); DataOutputStream dout = new DataOutputStream(bout); or DataOutputStream dout = new DataOutputStream( new BufferedOutputStream( new FileOutputStream( new File("filename"))));

Paulo André Castro

POO

ITA - Stefanini

130

## Exemplo – Escrita em Arquivos Binários

```
import java.io.*;

public class BinaryFileOutput {

    public static void main(String[] args) {
        int[] primes = { 1, 2, 3, 5, 11, 17, 19, 23 };
        DataOutputStream out = null;

        try {
            out = new DataOutputStream(
                new FileOutputStream("primes.bin"));

            for (int i=0; i<primes.length; i++) {
                out.writeInt(primes[i]);
            }
            out.close();
        } catch (IOException ioe) {
            System.out.println("IO problem: " + ioe);
            ioe.printStackTrace();
        }
    }
}
```

Paulo André Castro

POO

ITA - Stefanini

131

## Classes para Leitura em Streams Binárias

Desired ...	Methods	Construction
Binary File Input bytes	<b>FileInputStream</b> read() read(byte[] buffer)	File file = new File("filename"); FileInputStream fn = new FileInputStream(file); or FileInputStream fn = new FileInputStream("filename");
Binary File Input byte short int long float double char boolean	<b>DataInputStream</b> readByte() readShort() readInt() readLong() readFloat() readDouble() readChar() readBoolean() readUTF() readFully(byte[] buffer)	File file = new File("filename"); FileInputStream fn = new FileInputStream(file); DataInputStream din = new DataInputStream(fn); or DataInputStream din = new DataInputStream( new FileInputStream( new File("filename")));

Paulo André Castro

POO

ITA - Stefanini

132

## Classes para Leitura em Streams Binárias

Desired ...	Methods	Construction
Buffered Binary File Input	<b>BufferedInputStream</b> read() read(byte[] buffer, int off, int len) skip(long)	File file = new File("filename"); FileInputStream fn = new FileInputStream(file); BufferedInputStream bin = new BufferedInputStream(fn); DataInputStream din = new DataInputStream(bin); or DataInputStream din = new DataInputStream( new BufferedInputStream( new FileInputStream( new File("filename"))));

Paulo André Castro

POO

ITA - Stefanini

133

## Exemplo – Leitura de Arquivos Binários

```
import java.io.*;

public class BinaryFileInput {
    public static void main(String[] args) {
        DataInputStream in = null;
        File file = new File("primes.bin");
        try {
            in = new DataInputStream(
                new FileInputStream(file));
            int prime;
            long size = file.length()/4; // 4 bytes per int
            for(long i=0; i<size; i++) {
                prime = in.readInt();
                System.out.println(prime);
            }
            in.close();
        } catch (IOException ioe) {
            System.out.println("IO problem: " + ioe);
            ioe.printStackTrace();
        }
    }
}
```

Paulo André Castro

POO

ITA - Stefanini

134

## Resumo

- Um objeto File pode referir-se tanto a um arquivo quanto a um diretório
- Use classes Reader/Writer para lidar com streams baseadas em caracteres
  - Para ler linhas use BufferedReader
  - Use classes de formatação para a formatação de texto (Ex. DecimalFormat, package java.text)
- Use classes DataStream para lidar com streams baseadas em bytes.
- Associe um FileOutputStream a um DataOutputStream para escrever em arquivos binários tipos básicos do Java
- Associe um FileInputStream a um DataInputStream para ler de arquivos binários usando tipos básicos do Java
- Para ler ou escrever objetos em streams deve-se fazer uso das classes ObjectOutputStream e ObjectInputStream.

Paulo André Castro

POO

ITA - Stefanini

135

## Exercício 1

- Crie um programa que armazene uma lista de Funcionários (name,salary,hireDay) criados "manualmente" em um arquivo texto
  - Leia este arquivo texto, atualizando a lista em memória e
  - Liste na tela os dados dos funcionários lidos
  - Use no mínimo duas classes: DataFileTest e Employee
  - Formato do registro:
    - [name] | [salary] | [year] | [month] | [day]
  - Utilizar StringTokenizer para separar campos
    - StringTokenizer t=new StringTokenizer(str,"|");
    - String s=t.nextToken();

Paulo André Castro

POO

ITA - Stefanini

136

## Exercício 1.1

- Crie um método em Employee que incrementa o salário de um funcionario pelo percentual passado como parâmetro. Após a leitura dos dados do arquivo aumente o salário de todos em 10%. Apresente os dados

Paulo André Castro

POO

ITA - Stefanini

137

## Solução – Exercício 1 – Classe DataFileTest

```
public class DataFileTest
{
    static void writeData(Employee[] e, PrintWriter
    os)
    throws IOException
    {
        os.println(e.length);
    }

    int i;
    for (i = 0; i < e.length; i++)
        e[i].writeData(os);
}

static Employee[] readData(BufferedReader is)
throws IOException
{
    int n = Integer.parseInt(is.readLine());
    Employee[] e = new Employee[n];
    int i;
    for (i = 0; i < n; i++)
    {
        e[i] = new Employee();
        e[i].readData(is);
    }
    return e;
}

public static void main(String[] args)
{
    Employee[] staff = new Employee[3];
    staff[0] = new Employee("Harry Hacker", 35500,
    Format.getDate(1989,10,1));
    staff[1] = new Employee("Carl Cracker", 75000,
    Format.getDate(1987,12,15));
    staff[2] = new Employee("Tony Tester", 38000,
    Format.getDate(1990,3,15));
}

try
{
    PrintWriter os = new PrintWriter(new
    FileWriter("employee.dat"));
    writeData(staff, os);
    os.close();
}
catch (IOException e)
{
    System.out.println("Error: " + e);
    System.exit(1);
}

try
{
    BufferedReader is = new BufferedReader(new
    FileReader("employee.dat"));
    Employee[] in = readData(is);
    for (i = 0; i < in.length; i++) in[i].print();
    is.close();
}
catch (IOException e)
{
    System.out.println("Error: " + e);
    System.exit(1);
}
}
```

Paulo André Castro

POO

ITA - Stefanini

138

```
class Employee
{
    public Employee(String n, double s, Date
    d)
    {
        name = n;
        salary = s;
        hireDay = d;
    }

    public Employee() {}
    public void print()
    {
        System.out.println(name + " " + salary
        + " " + hireYear());
    }

    public void raiseSalary(double byPercent)
    {
        salary *= 1 + byPercent / 100;
    }

    public int hireYear()
    {
        return Format.getYear(hireDay);
    }

    public void writeData(PrintWriter os) throws IOException
    {
        os.println(name + " ");
        os.println(salary + " ");
        os.println(Format.getYear(hireDay) + " ");
        os.println(Format.getMonth(hireDay) + " ");
        os.println(Format.getDayOfMonth(hireDay) + " ");
    }

    public void readData(BufferedReader is) throws IOException
    {
        String s = is.readLine();
        StringTokenizer t = new StringTokenizer(s, " ");
        name = t.nextToken();
        salary = Double.parseDouble(t.nextToken());
        int y = Integer.parseInt(t.nextToken());
        int m = Integer.parseInt(t.nextToken());
        int d = Integer.parseInt(t.nextToken());
        hireDay = Format.getDate(y,m,d);
    }

    private String name;
    private double salary;
    private Date hireDay;
}
```

Paulo André Castro

POO

ITA - Stefanini

139

## Solução 1.1

### Na classe DataFileTest

```
public void raiseSalary(Employee[] e) {
    int i;
    for (i = 0; i < staff.length; i++)
        staff[i].raiseSalary(10.0);
}
```

### Na classe Employee,

```
public void raiseSalary(double byPercent)
{
    salary *= 1 + byPercent / 100;
}
```

Paulo André Castro

POO

ITA - Stefanini

140

## Exercício 2 – Opcional/Homework

- Refaça o exercício 1, agora acessando arquivos binários através da classe RandomAccessFile:
- Crie um programa que armazene uma lista de Funcionários (name,salary,hireDay) criados "manualmente" e grave estes objetos em um arquivo Leia o arquivo de dados, atualizando a lista em memória e
  - Liste na tela os dados dos funcionários lidos
  - Use no mínimo duas classes: ObjectFileTest e Employee

Paulo André Castro

POO

ITA - Stefanini

141

## Exercício 3 Opcional/Homework

- Crie um programa que armazene uma lista de Funcionários (name,salary,hireDay) criados "manualmente" e grave estes objetos em um arquivo usando o recurso de serialização (implements Serializable)
  - Leia o arquivo de dados, atualizando a lista em memória e
  - Liste na tela os dados dos funcionários lidos
  - Use no mínimo duas classes: ObjectFileTest e Employee

Paulo André Castro

POO

ITA - Stefanini

142

## Exercício 4

- Crie um programa que liste as entradas de um arquivo .zip e o conteúdo de arquivos textos
  - Utilize as classes ZipInputStream e InputStreamReader

Paulo André Castro

POO

ITA - Stefanini

143

## Solução – Exercício 4

```
public class ZipTest extends CloseableFrame
implements ActionListener
{
    public ZipTest()
    {
        MenuBar mbar = new MenuBar();
        Menu m = new Menu("File");
        MenuItem m1 = new MenuItem("Open");
        m1.addActionListener(this);
        MenuItem m2 = new MenuItem("Exit");
        m2.addActionListener(this);
        mbar.add(m);
        setMenuBar(mbar);
        fileList.addActionListener(this);

        add(fileList, "North");
        add(fileText, "Center");
    }

    public void actionPerformed(ActionEvent evt)
    {
        String arg = evt.getActionCommand();
        if (evt.getSource() == fileList)
        {
            loadZipFile(arg);
        }
        else if (arg.equals("Open"))
        {
            FileDialog d = new FileDialog(this,
            "Open zip file", FileDialog.LOAD);
            d.setFile("*.zip");
            d.setDirectory(lastDir);
            d.setVisible(true);
            String f = d.getFile();
            lastDir = d.getDirectory();
            if (f != null)
            {
                zipname = lastDir + f;
                scanZipFile();
            }
        }
        else if (arg.equals("Exit")) System.exit(0);
    }
}
```

Paulo André Castro

POO

ITA - Stefanini

144

```
public void scanZipFile()
{
    fileList.removeAll();
    try
    {
        ZipInputStream zin = new ZipInputStream(new
        FileInputStream(zipname));
        ZipEntry entry;
        while ((entry = zin.getNextEntry()) != null)
        {
            fileList.add(entry.getName());
            zin.closeEntry();
        }
        zin.close();
    }
    catch (IOException e) {}
}

public void loadZipFile(String name)
{
    try
    {
        ZipInputStream zin = new ZipInputStream(new
        FileInputStream(zipname));
        ZipEntry entry;
        fileText.setText("");
        while ((entry = zin.getNextEntry()) != null)
        {
            if (entry.getName().equals(name))
            {
                BufferedReader in = new BufferedReader(new
                InputStreamReader(zin));
                String s;
                while ((s = in.readLine()) != null)
                {
                    fileText.append(s + "\n");
                    zin.closeEntry();
                }
                zin.close();
            }
        }
    }
    catch (IOException e) {}
}
```

Paulo André Castro

POO

ITA - Stefanini

145

## Solução – Exercício 4

```
public static void main(String args[])
{
    Frame f = new ZipTest();
    f.setVisible(true);
}

private List fileList = new List();
private TextArea fileText = new TextArea();
private String lastDir = "";
private String zipname;
}
```

Paulo André Castro

POO

ITA - Stefanini

146

## Exercício 5

- Crie um arquivo compactado a partir de uma lista de arquivos (mantendo a informação de caminho) contida em um arquivo texto, cujo nome é passado como parâmetro.
  - Utilize as classes `java.io.ZipFile`, `java.io.ZipInputStream`

Paulo André Castro

POO

ITA - Stefanini

147