

#### Ficheiros de texto



- Ficheiros são Estruturas de Dados para **armazenamento** permanente da informação em dispositivos de memória secundária.
- Em Java há dois tipos de ficheiros: **texto** e binários.
- Embora em ambos, a informação seja armazenada como sequências de bits, um ficheiro de texto é processado como uma sequência de caracteres e um ficheiro binário como uma sequência de bytes.
- Por exemplo o inteiro 12345 pode ser armazenado:
  - em formato de texto como uma sequência de 5 caracteres "1". "2". "3". "4" e "5"
  - Ou em formato binário como uma sequência de bytes neste caso porque é um int 4 bytes que são

00000000 00000000 00110000 00111001 0

48

porque 12345=48\*256+57











#### Streams



- Como escrever programas em Java que interatuam com ficheiros em disco e outras fontes de bytes ou caracteres?
- As operações de entrada e saída (I/O) em Java são baseadas no uso de streams.
- Stream é uma fonte genérica de leitura de dados ou um destino genérico de escrita de dados.
- Um stream é independente de dispositivos físicos.
- Stream é uma abstração que terá que ser associado a uma entidade física.
- Há streams de caracteres (texto) e streams de bytes (binários).











#### Streams



- Quando se inicia a execução de uma aplicação Java são criados automaticamente três objetos do tipo stream:
  - System.in
    - Stream de input associado ao periférico standard de input, geralmente o teclado
  - System.out
    - Stream de output associado ao periférico standard de output, geralmente o ecrã
  - System.err
    - Stream de output associado ao periférico standard de visualização de erros, geralmente o ecrã
- Cada um destes streams pode ser redirecionado











#### Classe File



- Esta classe não é uma classe de stream, ela representa os usuais ficheiros e diretórios de um sistema de gestão de ficheiros.
- Exemplo de alguns métodos desta classe

```
File f = new File("c:\\jdk\\teste1.java");
// Ou separador só com uma / (mesmo em Windows)
      File f= new File("c:/jdk/teste1.java");
      // f é o nome lógico do ficheiro físico
      String nome=f.getName();
      String path=f.getPath();
      int comp=f.length();
      if(f.isDirectory())
             String [] files=f.list();
      else f.delete();
```











# Criação de streams associados a ficheiros



- Ler dados de um ficheiro texto
  - Usando a classe Scanner
    - Scanner in=new Scanner(new File("Texto.txt"));
      - Se Texto.txt n\u00e3o existe lan\u00e3a
         FileNotFoundException que tem que ser verific\u00e1vel
    - o int x=in.nextInt();
- Escrever dados em ficheiro texto
  - Usar a classe Formatter
    - Formatter out=new Formatter(new File("output.txt"));
      - Se output.txt não pode ser criado lança
         FileNotFoundException que tem que ser verificável
    - out.format("%d%n",x);











#### Classe Scanner



- Esta classe do package java.util (import java.util.Scanner;) é usada para ler informação do teclado assim como de qualquer ficheiro texto.
- Exemplo1: leitura do teclado

```
Scanner input=new Scanner(System.in);
int n=input.nextInt();
double d=input.nextDouble();
String s=input.next();
```









# **Classe Scanner**



```
import java.io.File;
import java.util.Scanner;
public class ficheiroLeitura {
  public static void main(String [] args) throws FileNotFoundException {
     //Abrir o ficheiro Dados.txt para leitura
     Scanner input=new Scanner(new File("Dados.txt"));
     //Ler do ficheiro
     while(input.hasNext()) {//verifica se há mais conteúdo a ler
          System.out.println(input.next());
          System.out.println(input.nextInt());
     //Fechar ficheiro
     input.close();
```











# **Classe Formatter**



- Esta classe do package java.util (import java.util.Formatter;) é usada para escrever dados formatados no ecrã, assim como em qualquer ficheiro texto, recorrendo ao método format(...), que é semelhante à função printf do C.
- Exemplo: Escrita no ecrã Formatter output=new Formatter(System.out); output.format("%s%n%d%s%.2f%c", "A Joana tem",21,"anos e mede",1.665,'m');
- Resultado escrito no ecrã:

A Joana tem 21 anos e mede 1,67 m











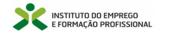
# **Classe Formatter**



- Com a classe Formatter não é possível acrescentar linhas (append) a um ficheiro de texto já existente.
- A alternativa será usar a classe FileWriter











# **Classe Formatter**



```
import java.io.File;
import java.util.Formatter;
public class ficheiroEscrita {
public static void main(String [] args) throws FileNotFoundException
  //Abrir o ficheiro Resultados.txt para escrita
  Formatter output=new Formatter(new File("Resultados.txt"));
  //Escrever no ficheiro
  output.format("%s%n%d%n%.2f%n","Ricardo",33,1.823);
  output.format("%s%n%d%n%.2f%n","Tiago",22,1.794);
  //Fechar ficheiro
  output.close();
                                                 Resutados.txt
                                                Ricardo
                                                33
                                                1.82
                                                Tiago
                                                22
                                                1,79
```











# Exemplo com Scanner e Formatter



```
public static void main(String [] args) throws FileNotFoundException {
   String []vec=new String[10];
   Scanner in=new Scanner(new File("C:/Res/textoIn.txt"));
   Formatter out = new Formatter(new File("C:/Res/textoOut.txt"));
   int nEl=0;
   while (in.hasNext()) {
        vec[nEl]=in.next(); nEl++;}
   String x;
   for(int i=0;i< nEl-1;i++)
        for(int j=i+1;j< nEl;j++)
            if(vec[i].compareTo(vec[j])>0)
                 x=vec[i]; vec[i]=vec[i]; vec[i]=x;
    out.format("%s","Listagem Ordenada");
    for(int i=0;i< nEl;i++)
       out.format("%n%s", vec[i]);
    out.close();
```





