di
departamento de informática

dwdm
desenvolvimento para a web e dispositivos móveis

Técnicas Avançadas de Programação

Desenvolvimento para Web e Dispositivos Móveis

1º Ano, 2º Semestre

Joana Fialho
E-mail: jfialho@estgv.ipv.pt
Nuno Costa
E-mail: ncosta@estgv.ipv.pt
Carlos Simões
E-mail: csimoes@estgv.ipv.pt

Escola Superior de Tecnologia e Gestão de Viseu
2019-2020

Ficheiros

Durante a execução de um programa, os objetos e variáveis ficam na memória central do computador, pelo que, quando o programa termina, esses dados ficam inacessíveis. É necessário guardar informação em ficheiros.

Em JAVA usam-se streams (fluxos).

Fluxo de dados entre teclado e memória -> leitura de dados durante a execução do programa

Fluxo de dados entre a memória e o ecrã -> escrita no ecrã durante a execução do programa

Ficheiros

O fluxo de dados em ficheiros é semelhante:

em vez da leitura do teclado, faz-se a leitura do ficheiro;

em vez da escrita para o ecrã, faz-se a escrita para o ficheiro.

Existem várias classes, em JAVA, que permite manipular os fluxos de dados e essas classes estão na biblioteca java.io.

Ficheiros – classe File

Representa ficheiro em disco e cria representações lógicas de ficheiros ou pastas:

```
File f1=new File(nomeFich);
```

nomeFich representa o caminho (path) para o ficheiro em string. (Ex.: “c:/Exemplo/fich1.txt”).

Se o ficheiro estiver na pasta corrente ou se se pretender criar o ficheiro na pasta corrente, basta indicar, na string, o nome do ficheiro (“fich1.txt”).

Ficheiros – classe File

Método	Descrição
delete()	Apaga ficheiro ou pasta
exists()	Verifica se existe
length()	Devolve tamanho do ficheiro, em bytes
renameTo()	Altera nome do ficheiro ou pasta
SetReadOnly()	Marca o ficheiro ou pasta como só leitura
listFiles()	Devolve tabelas de objetos File com ficheiros da pasta
mkdir()	Cria subpasta

Ficheiros de texto

Classe para leitura de caracteres – `FileReader`

Classe para escrita de caracteres – `FileWriter`

Ambas as classes recebem, como parâmetro, objeto da classe File.

```
FileReader fin=new FileReader(new File(nomeFich));

FileWriter fout=new FileWriter(new File(nomeFich));
```

Ficheiros de texto

Classe para leitura de **linhas de caracteres** – `BufferedReader`

Classe para escrita de **linhas de caracteres** – `BufferedWriter`

Recebem, como parâmetro, objetos da classe FileReader e FileWriter, respetivamente.

```
BufferedReader fichIn=new BufferedReader(fin);

BufferedWriter fichOut=new BufferedWriter(fout);
```

Ficheiros de texto

Quando se abre um ficheiro para escrita, se ele não existir, é criado; se existir, apaga o que houver e escreve por cima. Se se tenta abrir um ficheiro de leitura, que não existe, há erro.

Os métodos que manipulam ficheiros devem ter, no seu cabeçalho, `throws IOException`

Método	Descrição	Classe Associada
readLine()	Lê linha de caracteres	BufferedReader
Integer.ParseInt(str)	Converte a string str para número (ex. “12” para 12)	Útil após o readLine da classe BufferedReader
write(linha,0,tam)	Escreve a string linha, desde o índice 0 ao índice tam (exclusive)	BufferedWriter
newLine()	Mudança de linha	BufferedWriter
String.ValueOf	Converte número para string	Útil para depois poder utilizar BufferedWriter
close()	Fecha ficheiros	BufferedReader BufferedWriter

Ficheiros de texto

```
ArrayList<Integer> nums=new ArrayList<Integer>();
BufferedReader f=new BufferedReader(new FileReader(new File( pathname: "numeros.txt")));
String linha=f.readLine();
while (linha!=null)
{
    nums.add(Integer.parseInt(linha));
    linha=f.readLine();
}
f.close();
```

Acima, encontra-se exemplo de utilização de leitura de ficheiros de texto. Atenção que são sempre lidas strings, pelo que, se se souber que são valores numéricos, há que fazer a devida conversão.

Ficheiros de texto

Veja-se, agora, exemplo de escrita num ficheiro de texto.

```
public void guardaPessoasTxt(String nf) throws IOException {
    FileWriter fich = new FileWriter(new File(nf));
    BufferedWriter f = new BufferedWriter(fich);
    for (Pessoa p : listagem) {
        f.write( str: p.getNome() + ";" + p.idade() + ";" + p.getNif() + ";" + p.getContacto());
        f.newLine();
    }
    f.close();
}
```

Ficheiros de texto

Pode usar-se a classe PrintWriter em vez da classe BufferedWriter. Esta classe pode ser iniciada diretamente da classe File, sem usar a classe FileWriter. PrintWriter tem acesso a métodos com mais opção de formatação, como printf, println, entre outros.

No entanto, **não existe** PrintReader.

```
PrintWriter f=new PrintWriter(nf);
for (Pessoa p:listagem) {
    f.println(p.getNome()+";"+p.idade()+";"+p.getNif()+";"+p.getContacto());
}
f.close();
```

Ficheiros

Notas:

- A classe System assegura a leitura do teclado e a escrita no ecrã através da divisão System.in e System.out, respetivamente.
- A leitura de dados, pelo teclado, é feita pela classe Scanner, que permite ler strings a partir de um ficheiro ou teclado. Sendo a partir do teclado:

```
Scanner sc=new Scanner(System.in);
```

Para ler inteiro, será `int i=sc.NextInt();`

Depois, lê doubles, floats, strings (`nextDouble()`, `nextFloat()`, `next()`, `nextLine()`)

Ficheiros de Objetos

Por vezes, para guardar a informação dos objetos que se manipulam, os ficheiros de texto podem não ser os mais adequados. Usam-se, assim, ficheiros de objetos que, não sendo legíveis, permitem o armazenamento dos objetos, tal como eles são.

Classe para estabelecer fluxos de dados para leitura – FileInputStream

Classe para estabelecer fluxos de dados para escrita – FileInputStream

Ambas as classes recebem, como parâmetro, objeto da classe File.

```
FileInputStream fin=new FileInputStream (new File(nomeFich));

FileOutputStream fout=new FileOutputStream(new File(nomeFich));
```

Ficheiros de Objetos

Classe para leitura de **objetos** – ObjectInputStream

Classe para escrita de **objetos** – ObjectoutputStream

Recebem, como parâmetro, objetos da classe FileInputStream e FileInputStream, respetivamente.

```
ObjectInputStream fichIn=new ObjectInputStream (fin);

ObjectoutputStream fichOut=new ObjectoutputStream (fout);
```

Ficheiros de Objetos

Abrindo, para escrita, um ficheiro existente, o seu conteúdo é apagado e substituído por nova informação (como nos ficheiros de texto).

Método de leitura de objeto da classe
ObjectInputStream -> readObject()

Método de escrita de objeto da classe
ObjectoutputStream -> writeObject()

Ficheiros de Objetos

Quando se pretende guardar objetos de uma classe definida pelo programador, essa classe deve conter, no cabeçalho, as palavras reservadas implements Serializable

(Ex.: `public class Pessoa implements Serializable`)

Este cabeçalho autoriza os seus objetos a serem enviados para o método writeObject(). De igual modo, o método readObject() pode ser usado para enviar a informação do ficheiro para a memória central. Para cada objeto, apenas são guardadas as variáveis de instância (próprias do objeto, os atributos).

Ficheiros de Objetos

Nos métodos que manipulam ficheiros de objetos, deve aparecer, no cabeçalho, a sequência reservada throws IOException (como nos ficheiros de texto). Além disso, na leitura de objetos, pode aparecer, também, ClassNotFoundException, no caso do ficheiro de objetos, que se está a ler, não contiver objetos.

Também nos ficheiros de objetos, é utilizado o método close() para os fechar.

Ficheiros de objetos

Escrita de uma lista, num ficheiro de objetos:

```
ArrayList<Integer> lista=new ArrayList<Integer>();
 lista.add(3);
 lista.add(5);
 FileOutputStream fich=new FileOutputStream(new File( pathname: "teste2"));
 ObjectOutputStream f=new ObjectOutputStream(fich);
 f.writeObject(lista);
 f.close();
```

Leitura do ficheiro de objetos (deve saber-se o que contém):

```
FileInputStream fich2=new FileInputStream(new File( pathname: "teste2"));
ObjectInputStream f2=new ObjectInputStream(fich2);
ArrayList<Integer> lista2=(ArrayList<Integer>) f2.readObject();
f.close();
```

Ficheiros de objetos

Note-se que o método `readObject` lê objetos da classe `Object`. Todas as classes descendem direta ou indiretamente de `Object`, pelo que, após leitura do objeto (invocação do método `readObject`), deve fazer-se a respetiva conversão (*cast*).