Trabalhando com arquivos em Java

Fabio Lubacheski
fabio.aglubacheski@sp.senac.br

Trabalhando com arquivos em Java

- Durante a execução de um programa, seus dados ficam na memória. Quando o programa termina, ou o computador é desligado, os dados na memória desaparecem.
- Para armazenar os dados permanentemente, você tem que colocá-los em um **arquivo**. Arquivos usualmente são guardados em um disco rígido (HD), num pendrive ou em um CD-ROM.
- Os arquivos são organizados em pasta (=*diretório*=*folder*), e dentro de uma pasta um arquivo é identificado por um nome único.

Trabalhando com arquivos em Java

- Trabalhar com arquivos é muito parecido com trabalhar com livros. Para utilizar um livro, você tem que **abri-lo**. Quando você termina, você tem que **fechá-lo**.

- Enquanto o livro estiver **aberto**, você pode tanto lê-lo quanto escrever nele. Na maioria das vezes, você lê o livro inteiro em sua ordem natural, uma página após a outra.

- Em um arquivo texto os dados são organizados como uma sequência de caracteres dividida em linhas terminadas por um caractere de fim de linha (**\n**). Em um programa em Java a manipulação do arquivo passa por três etapas: **abertura do arquivo**, **leitura/escrita dos dados** e **fechamento do arquivo**.

Manipulação de arquivos em Java

- Para abrir um arquivo em Java utilizamos a classe **`FileReader`**. Para utilizar a classe basta instanciar um objeto de leitura **`FileReader`** passando o nome do arquivo como parâmetro no construtor da classe, veja o exemplo:

```
FileReader arquivo;
arquivo = new FileReader("entrada.txt");
```

- **Importante:** O arquivo que será lido deve estar na mesma pasta raiz do projeto no **NetBeans.**

- Em seguida é necessário associar o objeto de leitura `arquivo` a um fluxo (stream) de entrada bufferizada baseado em **caracteres** através da classe **`BufferedReader`**.

Manipulação de arquivos em Java

- Isso é feito criando um objeto de leitura bufferizada através do construtor da classe **`BufferedReader`**.

```
FileReader arquivo;
arquivo = new FileReader("entrada.txt");

BufferedReader leBufferizado;
leBufferizado = new BufferedReader(arquivo);
```

Manipulação de arquivos em Java

- A classe **`BufferedReader`** disponibiliza a função **`readLine()`** que lê uma linha do arquivo de entrada, lembrando que uma linha no arquivo é finalizada pelo caracter `'\n'`, veja o exemplo:

```
String linha;
linha = leBufferizado.readLine();
System.out.println(linha);
```

- É possível ler várias linhas dos arquivo de entrada, quando as linhas do arquivo acabam a função `readLine()` retorna o valor `null` para linha.

Manipulação de arquivos em Java

- **Importante:** Ao final não esqueça de fechar o arquivo com função `close()`, a partir do objeto de leitura bufferizada.

```
// fecha arquivos
leBufferizado.close();
```

- A utilização das classes **`FileReader`** e **`BufferedReader`** geram algumas exceções, que são erros inesperado na manipulação dos arquivos, por exemplo a exceção `IOException` (erro de entrada e saída) ou exceção `FileNotFoundException`, que ocorre se o arquivo não for localizado. O correto seria tratar a exceção, mas repassamos a exceção adiante utilizando a claúsula `throws`.

Exercícios

1) Escreva um programa que abre um arquivo e lê todas as linhas do arquivo e imprime na tela. O nome do arquivo deve ser informado pelo usuário (teclado).
2) Escreva um programa que copia uma arquivo .java para um um novo arquivo. Os nomes dos arquivos devem ser informados pelo usuário.
3) Reescreva o programa que copia um arquivo para que toda linha que comece com // seja ignorada no arquivo destino.
4) Escreva um programa que conta o número de ocorrências de cada caractere em um arquivo. O programa imprime uma tabela que dá o número de ocorrências de cada caractere.
5) Faça um programa que resolve a **intersecção** entre dois vetores considerando que os vetores são lidos a partir de um arquivo.

Fim