ACESSANDO ARQUIVOS LOCAIS EM JAVA

O PACOTE JAVA.IO

- import java.io.*
- Oferece um conjunto de classes para se trabalhar com arquivos, diretórios e seus dados.
- Também oferece recursos para manipulação de dados durante o processo de leitura e gravação.

CLASSE “FILE”

- Útil para recuperar informações sobre arquivos ou diretórios do disco.
- Objetos da classe File não abrem arquivos ou fornecem quaisquer capacidades de processamento.
- Objetos da classe File são usados com objetos de outras classes java.io para especificar arquivos ou diretórios que serão processados.
- Usada para representar o sistema de arquivos.
  - A existência de um objeto não significa a existência de um arquivo ou diretório.
  - Contém métodos para testar a existência de arquivos, apagar arquivos, criar diretórios, listar o conteúdo de diretórios, etc..

CLASSE “FILE”

- Fornece quatro construtores:
  - public File(String name)
  - public File(String pathToName, String name)
  - public File(File directory, String name)
  - public File(URI uri)

- Exemplos:
  - file:/C:/data.txt
  - file:/home/student/data.txt

CLASSE “FILE”

- Alguns métodos:
  - String getAbsolutePath()
  - String getParent()
  - long length()
  - long lastModified()
  - boolean exists()
  - boolean isFile()
  - boolean isDirectory()
  - boolean delete()
  - boolean mkdir()
  - String[] list()

CLASSE “FILE”

- Exemplos de utilização:
  - FileDemonstration.java
  - FileDemonstrationTest.java

- Tais arquivos estão disponíveis para download neste tópico.
- Os alunos devem efetuar o download e analisar tais codificações.

CLASSE “FILE”

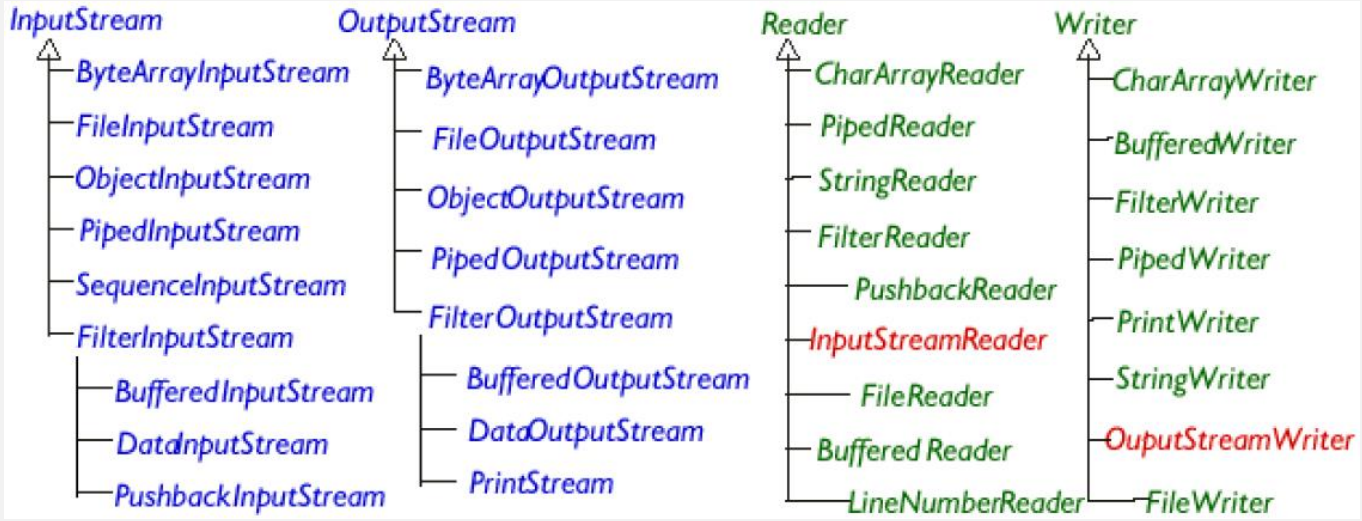
- Caractere Separador
  - No Windows, o separador é uma barra invertida (\)
  - No UNIX ou Linux, é um caractere barra (/)
  - Java processa ambos de forma idêntica
  - Usar \ como um separador em vez de \\ em uma string é um erro de lógica.
  - \ indica que o \ seguido pelo próximo caractere representa uma sequência de espaço. Use \\ para inserir um \ em uma string literal, exemplo: "C:\\temp\\2012\\File.txt”

FLUXOS DE ENTRADA E SAÍDA

- Existem várias fontes de leitura e gravação para onde se deseja gravar ou enviar dados:
  - Arquivos
  - Conexões em Rede
  - Console (teclado / vídeo)
  - Memória
- Há várias formas diferentes de ler/escrever dados:
  - Sequencialmente / aleatoriamente
  - Como bytes / como caracteres
  - Linha por linha / palavra por palavra
- O Pacote java.io oferecem objetos que abstraem fontes / destinos e fluxos de bytes e caracteres.

FLUXOS DE ENTRADA E SAÍDA

- Classes e interfaces para fluxos de E/S
  - Dois grupos:
    - E/S de bytes: InputStream e OutputStream
    - E/S de caracteres: Reader e Writer



FLUXOS DE ENTRADA E SAÍDA

- Leitura e Gravação de texto com buffer
  - A maneira mais eficiente de ler um arquivo texto é usar o FileReader juntamente com um BufferedReader.
  - Para gravar, use o FileWriter juntamente com um BufferedWriter.

- Exemplo:
  - https://www.devmedia.com.br/leitura-e-escrita-de-arquivos-de-texto-em-java/25529

- Artigo interessante para complementar os estudos:
  - https://www.devmedia.com.br/java-arquivos-e-fluxos-de-dados/22859

REFERÊNCIAS

- DEITEL, P.; DEITEL, H. Java – Como Programar. PRENTICE HALL, 2010.