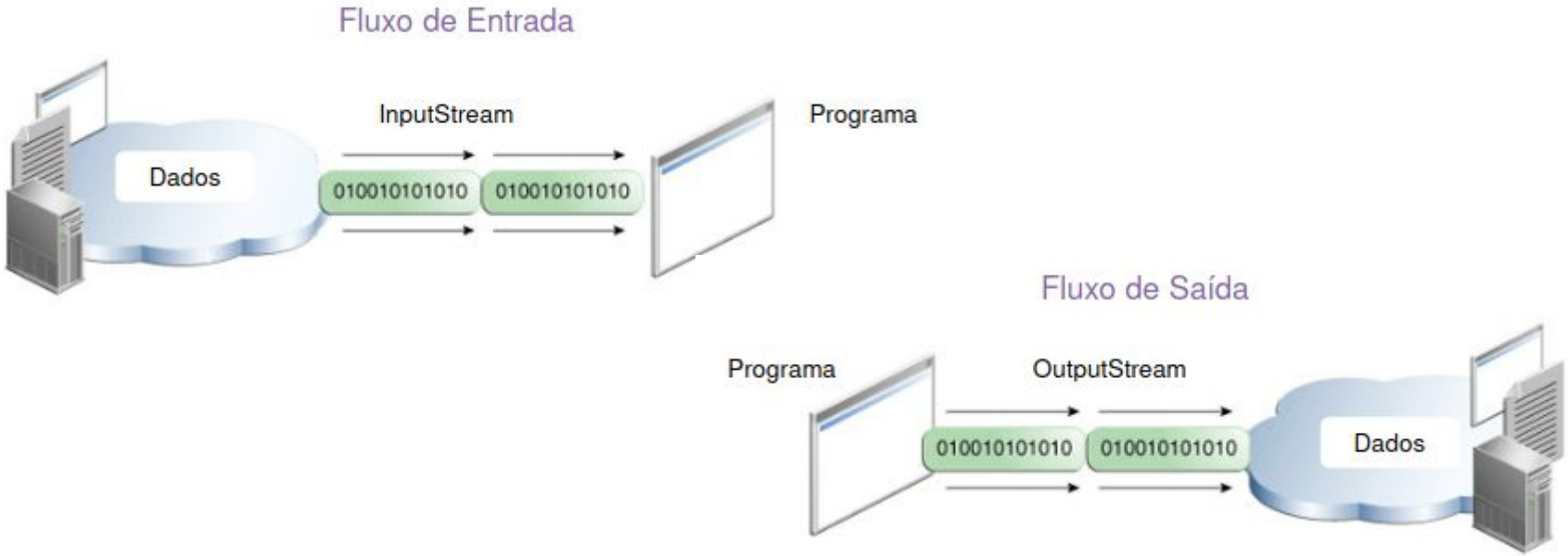
Arquivos

Fluxo de entrada e saída

I/O Streams representam uma entrada de dados ou uma saída de dados. Fluxos de dados podem ter arquivos como origem ou destino.



Fluxo de Byte

Leem e escrevem dados tendo o byte (8 bits) como unidade mínima de informação.

São tratados por subclasses de:

- InputStream, para leitura temos os metodos: read(), read(byte[]), skip(long), close(), etc
- OutputStream, para escrite temos os metodos: write(int), write(byte[]), flush(), close(), etc

Após leitura/escrita, o cursor avança dentro do arquivo. O fluxo de dados é sequencial. InputStream e OutputStream são classes abstratas.

Fluxo de Byte em Arquivos

As classes FileInputStream e FileOutputStream são especializações de InputStream e OutputStream que lidam com fluxo de dados. Para manipularmos arquivos, vamos utilizar as classes que estendem InputStream e OutputStream, pois essas classes são abstratas e uma classe abstrata não pode ser instanciada.

Fluxo de Caracteres em Arquivo Binário

As classes DataInputStream e DataOutputStream são especializações de FilterInputStream e FilterOutputStream.

Essas classes lidam com fluxos modificados de informações recebidas/enviadas através de um fluxo de dados existente. Utilizaremos as classes filhas para modificar o fluxo de um InputStream/OutputStream que retornam bytes para conseguirmos ler/escrever caracteres.

Sobrescrita x Escrita ao final do arquivo binário

Quando instanciamos a classe FilterOutputStream, podemos instancia-la de duas maneiras. A maneira (*) vai sobrescrever o conteúdo do arquivo de destino e a maneira (**) vai escrever ao final do arquivo mantendo o conteúdo existente no arquivo intacto.

(*) FilterOutputStream file = new FilterOutputStream(“filePath”);

(**) FilterOutputStream file = new FilterOutputStream(“filePath”, true);

Fluxo de Caracteres em Arquivo Texto

Existem duas classes que fazem a ponte entre fluxos de caracteres e fluxos de byte InputStreamReader e OutputStreamWriter. Essas classes são estendidas por FileReader e FileWriter.

Para lermos o texto de um fluxo de dados, vamos utilizar a classe BufferedReader. Esta classe lê o texto armazenando os caracteres em um buffer para fornecer uma leitura eficiente.

Para escrevermos um conjunto de caracteres em um arquivo, vamos utilizar a classe PrintWriter. Esta classe imprime representações formatadas de objetos em um fluxo de saída de texto.

Sobrescrita x Escrita ao final do arquivo texto

Quando instanciamos a classe FileWriter, podemos instancia-la de duas maneiras. A maneira (*) vai sobrescrever o conteúdo do arquivo de destino e a maneira (**) vai escrever ao final do arquivo mantendo o conteúdo existente no arquivo intacto.

(*) FileWriter file = new FileWriter(“filePath”);

(**) FileWriter file = new FileWriter(“filePath”, true);

Exceções Típicas

- IOException

  Superclasse de toda classe de exceção relacionada a entrada e saída através de arquivos

- FileNotFoundException

  Disparada quando tenta-se criar um objeto de fluxo para uma fonte que não pode ser localizada

- EOFException

  Disparada quando o final de arquivo é atingido inesperadamente enquanto os dados são lidos através de um fluxo de entrada