Luồng Đọc và ghi trong Java (I/O)

1. Giới thiệu

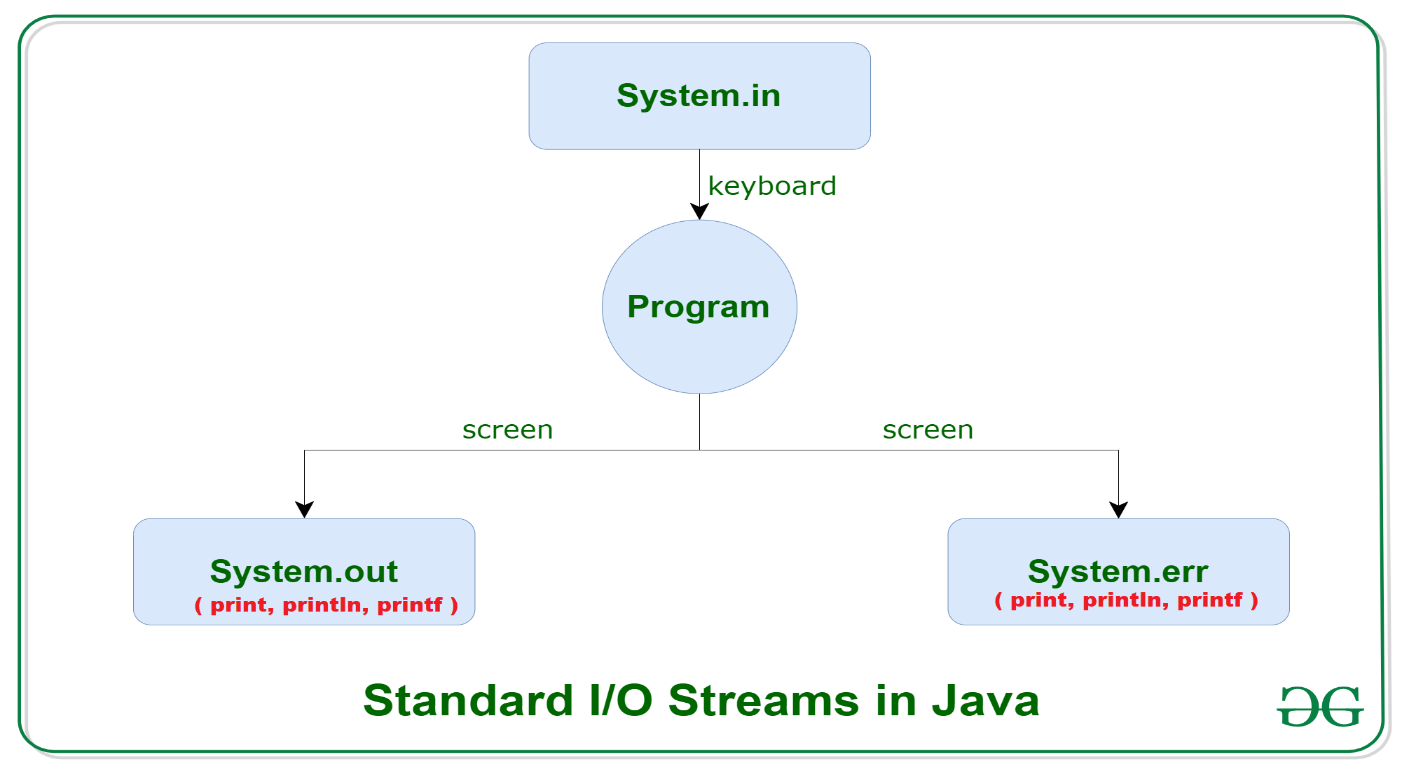
Java I/O hay Input/ Output trong Java là các hoạt động nhập ( đầu vào ) và xuất ( đầu ra ) trong Java. Gói java.io chứa tất cả lớp cần thiết cho hoạt động nhập, xuất.

Một Stream là luồng hay chuỗi dữ liệu/ thực thể ( như dòng nước chảy liên tục) liên tục để chuyển thông tin. Nó vận chuyển dữ liệu từ nguồn về đích hoặc ngược lại. Một stream bao gồm các byte dữ liệu.

Input (Đầu vào) hoặc Output (Đầu ra) có thể là: các thiết bị ngoại vi ( bàn phím, máy in, ổ đĩa …), HDD (file), Network Socket, hoặc các chương trình khác.

Trong Java, có 3 stream tiêu chuẩn được tạo, và gắn với console:

* System.in: stream luồng đọc tiêu chuẩn
* System.out: stream luồng ghi tiêu chuẩn
* System.err: stream luồng lỗi tiêu chuẩn



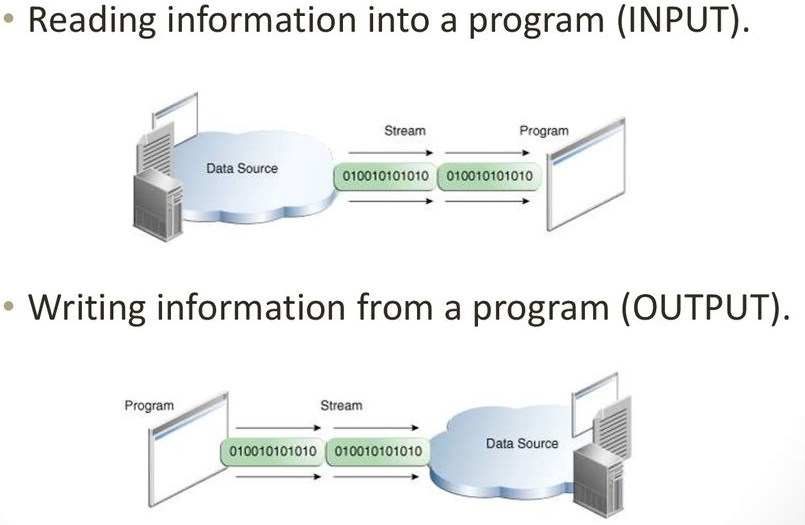
1. Các loại luồng stream

Trong Java, streams dùng để thực hiện tất cả các thao tác nhập/ ghi ( I/O ) .

Class về Stream giúp:

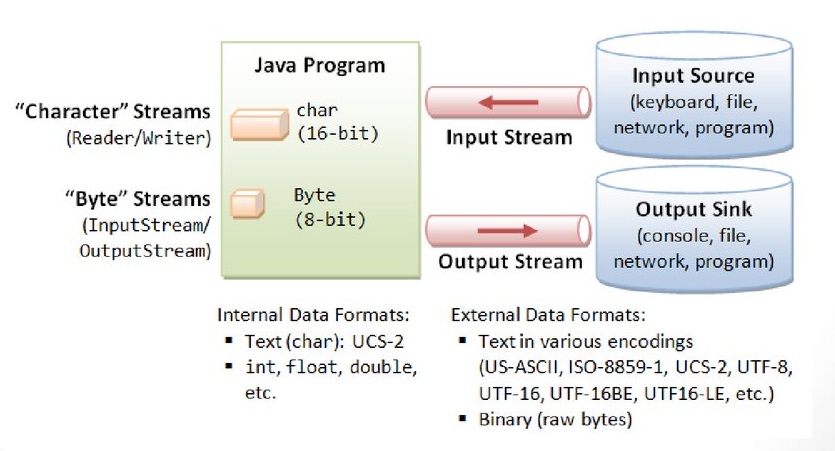
* Đọc input từ 1 stream
* Ghi output xuống 1 stream
* Quản lý các file ổ đĩa
* Chia sẻ dữ liệu trong mạng máy tính

Luồng đọc, ghi stream:



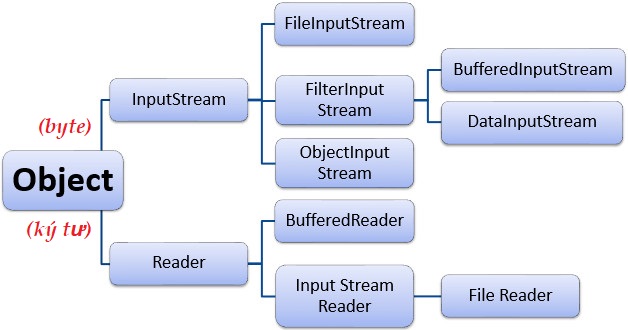
* Java cung cấp 2 kiểu luồng:

|  |  |
| --- | --- |
| Luồng Byte (ASCII) | Luồng Character ( ký tự) |
| * Hỗ trợ việc xuất, nhập dữ liệu trên byte 8 bit * Thường dùng khi đọc ghi dữ liệu nhị phân * Sử dụng trong trường hợp nhập xuất kiểu nguyên thủy (byte, short, int, long, char, Boolean, float, double), nhập xuất kiểu dữ liệu đối tượng ( object) * 2 class cha: InputStream ( đọc từ nguồn) và OutputStream ( ghi tới đích) | * Hỗ trợ việc xuất nhập kiểu dữ liệu ký tự (Unicode) * Đọc ghi dữ liệu Unicode 16 bit * Sử dụng trong trường hợp thao tác lưu trữ file chứa văn bản Unicode. * 2 class cha: Reader ( đọc ) và Writer ( Ghi ) |

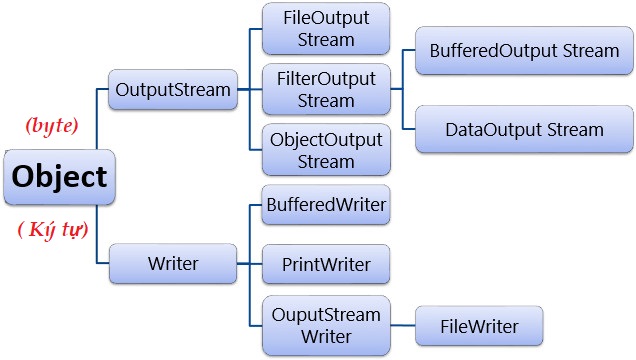


Xem chi tiết các interface, class tại: https://docs.oracle.com/javase/8/docs/api/java/io/package-tree.html

* Phân cấp luồng đọc/ luồng nhập dữ liệu / Input stream



* Phân cấp luồng ghi/ luồng xuất / Output Stream:



1. Thao tác xử lý dữ liệu đọc, ghi:

Luôn tuân thủ 3 bước:

* Bước 1: Tạo đối tượng stream và liên kết với nguồn dữ liệu.
* Bước 2: Thao tác dữ liệu ( Đọc hoặc ghi hoặc cả hai )
* Bước 3: Đóng luồng.

1. Lớp InputStream:

|  |  |
| --- | --- |
| **Phương thức** | **Mô tả** |
| public abstract int read() throws IOException | Đọc byte kế tiếp của dữ liệu từ input stream. Nó trả về -1 khi đọc đến vị trí cuối tập tin. |
| public abstract int read(byte b[]) throws IOException | Đọc tất cả các byte hiện có trong Stream vào mảng byte b[]. |
| public abstract int read(byte b[], int offset, int length) throws IOException | Đọc số lượng (**length**) byte  từ Stream hiện tại, lưu vào trong mảng byte (**b**) bắt đầu từ vị trí (**offset**) được chỉ định. |
| public int available()throws IOException | Trả về số lượng byte hiện có trong Stream. |
| public void close()throws IOException | Được sử dụng để đóng input stream hiện tại và giải phóng các tài nguyên đang liên kết với Stream. |
|

Một số lớp con InputStream:

* 1. ByteArrayInputStream

|  |  |
| --- | --- |
| ByteArrayInputStream được sử dụng để đọc mảng byte như là input stream. - Lớp ByteArrayInputStream trong java chứa một bộ đệm bên trong được sử dụng để đọc mảng byte dưới dạng luồng. - Bộ đệm của ByteArrayInputStream tự động tăng theo kích thước dữ liệu. | |
| **Constructor** | **Mô tả** |
| ByteArrayInputStream(byte[] ary) | Tạo ra một đối tượng ByteArrayInputStream mới sử dụng **ary** làm mảng đệm của nó. |
| ByteArrayInputStream(byte[] ary, int offset, int len) | Tạo ra một đối tượng ByteArrayInputStream mới sử dụng **ary** làm mảng đệm của nó có thể đọc lên **len quy định byte dữ liệu từ một mảng.** |
| **Methods** | **Description** |
| int available() | Nó được sử dụng để trả lại số byte còn lại có thể được đọc từ input stream. |
| int read() | Nó được sử dụng để đọc byte tiếp theo của dữ liệu từ input stream. |
| int read(byte[] ary, int off, int len) | Nó được sử dụng để đọc lên đến **len** byte dữ liệu từ một mảng các byte trong input stream. |
| boolean markSupported() | Nó được sử dụng để kiểm tra input stream cho các phương thức mark() và reset(). |
| long skip(long x) | Nó được sử dụng để bỏ qua x các byte của đầu vào từ input stream. |
| void mark(int readAheadLimit) | Nó được sử dụng để thiết lập vị trí đánh dấu hiện tại trong the stream. |
| void reset() | Nó được sử dụng để thiết lập lại bộ đệm của một mảng byte. |
| void close() | Nó được sử dụng để đóng ByteArrayInputStream. |

VD:

|  |
| --- |
| *try {  String s = "Hello word 12538 @#$";  is = new ByteArrayInputStream(s.getBytes());  int dem = 0;  while ((dem = is.read()) != -1){  System.out.print((char) dem);  } }catch (Exception e){  e.printStackTrace(); } finally {  try {  is.close();  } catch (IOException e) {  e.printStackTrace();  } }* |

* 1. FileInputStream

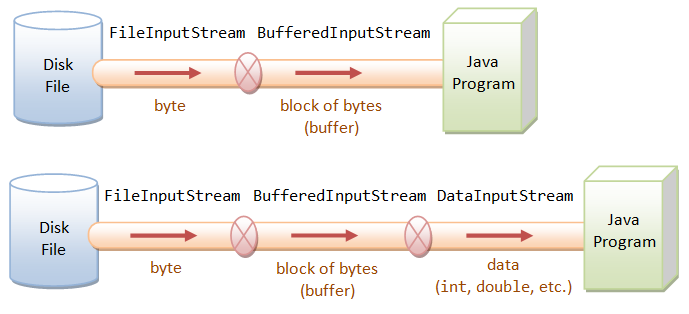
|  |  |
| --- | --- |
| Lớp FileInputStream: đọc được các byte từ một input file.  - Nó được sử dụng để đọc dữ liệu theo định dạng byte (các byte stream) như dữ liệu hình ảnh, âm thanh, video ....  - Có thể đọc các dữ liệu có định dạng ký tự. Tuy nhiên, để đọc các dòng ký tự (các character stream), sử dụng lớp FileReader | |
| **Method** | **Description** |
| int available() | Nó được sử dụng để trả về số byte ước tính có thể đọc được từ file input stream. |
| int read() | Nó được sử dụng để đọc byte dữ liệu từ file input stream. |
| int read(byte[] b) | Nó được sử dụng để đọc đến **b.length** byte dữ liệu từ file input stream. |
| int read(byte[] b, int off, int len) | Nó được sử dụng để đọc đến **len** byte dữ liệu từ ví trí **off** từ file input stream. |
| long skip(long x) | Nó được sử dụng để bỏ qua và loại bỏ x byte dữ liệu từ file input stream. |
| FileChannel getChannel() | Nó được sử dụng để trả về các đối tượng FileChannel duy nhất liên kết với file input stream. |
| FileDescriptor getFD() | Nó được sử dụng để trả về đối tượng FileDescriptor. |
| protected void finalize() | Nó được sử dụng để đảm bảo rằng phương thức close() được gọi khi không có tham chiếu đến file input stream. |
| void close() | Nó được sử dụng đẻ đóng stream. |

VD:

|  |
| --- |
| File file = **new** File(**"mau.txt"**); */\* Đọc file bằng FileInputStream \*/* InputStream is = **null**; FileInputStream fis = **null**; **try** {  fis = **new** FileInputStream(file);  **int** tmp = 0;  **while** ((tmp = fis.read()) != -1){  System.***out***.print((**char**) tmp);  } } **catch** (Exception e){  e.printStackTrace(); } **finally** {  **try** {  fis.close();  } **catch** (IOException e) {  e.printStackTrace();  } } |

* 1. FilterInputStream

|  |  |
| --- | --- |
| Lớp **FilterInputStream** cung cấp các lớp con khác nhau như BufferedInputStream, DataInputStream ... để cung cấp chức năng bổ sung, đi kèm với lớp khác | |
| **Phương thức** | **Mô tả** |
| int available() | Nó được sử dụng để trả về một số ước lượng của byte có thể được đọc từ input stream. |
| int read() | Nó được sử dụng để đọc byte tiếp theo của dữ liệu từ input stream. |
| int read(byte[] b) | Nó được sử dụng để đọc byte.length các byte dữ liệu từ input stream. |
| long skip(long n) | Nó được sử dụng để bỏ qua và loại bỏ n các byte dữ liệu từ input stream. |
| boolean markSupported() | Nó được sử dụng để kiểm tra xem các input stream có hỗ trợ các phương thức mark() và reset() không. |
| void mark(int readlimit) | Nó được sử dụng để đánh dấu vị trí hiện tại trong input stream. |
| void reset() | Nó được sử dụng để thiết lập lại input stream. |
| void close() | Nó được sử dụng để đóng input stream. |
| Lớp **BufferedInputStream:** sử dụng để đọc thông tin từ stream (buffered stream).  Lớp này sử dụng cơ chế đệm để làm cho hiệu suất đọc nhanh hơn. Chú ý:  - Khi các byte từ stream được bỏ qua hoặc đọc, bộ đệm nội bộ sẽ tự động nạp lại từ input steam chứa, nhiều byte tại một thời điểm.  - Khi một BufferedInputStream được tạo ra, một mảng đệm nội bộ sẽ được tạo ra.  Ưu điểm:   * Tăng hiệu quả khi đọc/ ghi | |
| **Constructor** | **Mô tả** |
| BufferedInputStream(InputStream IS) | Nó tạo ra đối tượng BufferedInputStream và lưu đối số của nó, input stream IS. |
| BufferedInputStream(InputStream IS, int size) | Nó tạo ra đối tượng BufferedInputStream với kích thước bộ đệm cụ thể và lưu đối số của nó, input stream IS, để sử dụng sau này. |
| **Phương thức** | **Mô tả** |
| int available() | Nó trả về một số ước lượng của byte có thể đọc được từ input stream mà không bị chặn bởi phương thức gọi tiếp theo cho input stream. |
| int read() | Nó đọc byte tiếp theo của dữ liệu từ input stream. |
| int read(byte[] b, int off, int ln) | Nó đọc các byte từ từ input stream được chỉ định vào một mảng byte được chỉ định, bắt đầu với số cho trước. |
| void close() | Nó đóng các input stream và giải phóng bất kỳ tàifd nguyên hệ thống nào liên quan đến stream. |
| void reset() | It repositions the stream at a position the mark method was last called on this input stream. |
| void mark(int readlimit) | Nó được xem như đính ước chung của phương thức đánh dấu cho input stream. |
| long skip(long x) | Nó bỏ qua và loại bỏ x byte dữ liệu từ input stream. |
| boolean markSupported() | Nó kiểm tra xem input stream có hỗ trợ các phương thức mark và reset không. |



VD:

|  |
| --- |
| *FilterInputStream fis = null; try {  is = new FileInputStream(file);  fis = new BufferedInputStream(is);  int dem = 0;  while ((dem = fis.read()) != -1){  System.out.print((char) dem);  } }catch (Exception e){  e.printStackTrace(); } finally {  try {  fis.close();  } catch (IOException e) {  e.printStackTrace();  }  try {  is.close();  } catch (IOException e) {  e.printStackTrace();  } }* |

4.4 ObjectInputStream

|  |  |
| --- | --- |
| Lớp **ObjectInputStream** sử dụng để đọc các đối tượng và dữ liệu nguyên thủy mà được ghi bằng việc sử dụng lớp ObjectOutputStream | |
| **Constructor** | **Mô tả** |
| public ObjectInputStream(InputStream in) | Tạo ra một ObjectInputStream đọc từ InputStream đã chỉ định. |
| **Phương thức** | **Mô tả** |
| public final Object readObject() | Đọc một đối tượng từ input stream. |
| public void close() | Đóng ObjectInputStream hiện tại. |

1. Lớp OutputStream

|  |  |
| --- | --- |
| **Phương thức** | **Mô tả** |
| public void write(int b) throws IOException | Được sử dụng để ghi một byte đến output stream hiện tại. |
| public void write(byte[] b) throws IOException | Được sử dụng để ghi một mảng các byte đến output stream hiện tại. |
| public void write(byte[] b, int offset, int length) throws IOException | Được sử dụng để ghi một mảng các byte có độ dài (**length**) xác định đến output stream hiện tại, bắt đầu từ vị trí (**offset**) được chỉ định. |
| public void flush() throws IOException | Đẩy các byte được lưu trong vùng đệm của Stream ra thiết bị ngoại vi. |
| public void close() throws IOException | Được sử dụng để đóng output stream hiện tại và giải phóng các tài nguyên đang liên kết với Stream |

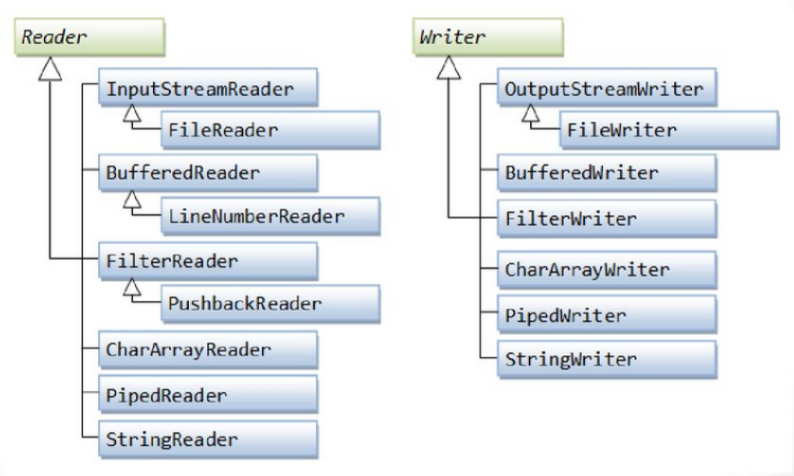
5.1 FileOutputStream

|  |  |
| --- | --- |
| **FileOutputStream** là một output stream được sử dụng để ghi dữ liệu vào một file theo định dạng byte (byte stream). - Sử dụng lớp FileOutputStream trong java khi phải ghi các giá trị nguyên thủy vào một file.  - Có thể ghi dữ liệu theo định dạng byte hoặc định dạng ký tự thông qua lớp FileOutputStream.  - Tuy nhiên, đối với các dữ liệu được ghi theo ký tự, sử dụng FileWriter thích hợp hơn FileOutStream. | |
| **Method** | **Description** |
| protected void finalize() | Nó được sử dụng để làm sạch kết nối với file output stream. |
| void write(byte[] ary) | Nó được sử dụng để ghi **ary.length** bytes từ mảng byte đến file output stream. |
| void write(byte[] ary, int off, int len) | Nó được sử dụng để viết **len** bytes từ mảng byte bắt đầu từ **off** tới file output stream. |
| void write(int b) | Nó được sử dụng để ghi byte cụ thể đến file output stream. |
| FileChannel getChannel() | Nó được sử dụng để trả lại đối tượng channel tập tin kết hợp với file output stream. |
| FileDescriptor getFD() | Nó được sử dụng để trả lại các mô tả của file liên quan đến stream. |
| void close() | Nó được sử dụng để đóng file output stream. |

5.2 BufferedOutputStream

|  |  |
| --- | --- |
| **BufferedOutputStream**: sử dụng để đệm một output stream.  Trong nội bộ của lớp này sử dụng bộ đệm để lưu trữ dữ liệu. Vì vậy, nó giúp hiệu suất ghi dữ liệu nhanh. | |
| **Constructor** | **Description** |
| BufferedOutputStream(OutputStream os) | Nó tạo ra đối tượng BufferedOutputStream được sử dụng để ghi các dữ liệu vào output stream cụ thể. |
| BufferedOutputStream(OutputStream os, int size) | Nó tạo ra đối tượng BufferedOutputStream được sử dụng để ghi các dữ liệu vào output stream cụ thể với kích thước bộ đệm cụ thể. |
| **Phương thức** | **Mô tả** |
| void write(int b) | Nó được sử dụng để ghi một byte cụ thể tới buffered output stream. |
| void write(byte[] b, int off, int len) | Nó ghi các byte từ byte-input stream được chỉ định vào mảng byte được chỉ định, bắt đầu bằng một giá trị cho trước **off** |
| void flush() | Xả những gì được lưu trong bộ đệm. |

1. Reader & Writer



* 1. BufferedReader

|  |  |
| --- | --- |
| **BufferedReader** sử dụng để đọc văn bản từ một input stream dựa trên các ký tự (character stream).  - Sử dụng để đọc dữ liệu theo dòng (line by line) bằng phương thức readLine().  - Nó giúp hiệu suất nhanh.  - Nó kế thừa lớp Reader. | |
| **Constructor** | **Mô tả** |
| BufferedReader(Reader rd) | Nó được sử dụng để tạo ra thể hiện của BufferedReader mà sử dụng kích thước mặc định cho một bộ đệm đầu vào. |
| BufferedReader(Reader rd, int size) | Nó được sử dụng để tạo ra thể hiện của BufferedReader có sử dụng kích thước quy định cho một bộ đệm đầu vào. |
| **Phương thức** | **Mô tả** |
| int read() | Nó được sử dụng để đọc ký tự vật duy nhất. |
| int read(char[] cbuf, int off, int len) | Nó được sử dụng để đọc các ký tự thành một phần của một mảng. |
| boolean markSupported() | Nó được sử dụng để kiểm tra input stream có hỗ trợ các phương thức mark() và reset() không. |
| String readLine() | Nó được sử dụng để đọc một dòng văn bản. |
| boolean ready() | Nó được sử dụng để kiểm tra liệu các input stream đã sẵn sàng để được đọc. |
| long skip(long n) | Nó được sử dụng để bỏ qua n ký tự. |
| void reset() | Nó định vị lại stream tại vị trí mà phương thức đánh dấu lần cuối được gọi vào input stream này. |
| void mark(int readAheadLimit) | Nó được sử dụng để đánh dấu vị trí hiện tại trong một stream. |
| void close() | Nó đóng các dòng đầu vào và giải phóng bất kỳ tài nguyên hệ thống nào liên kết đến stream. |

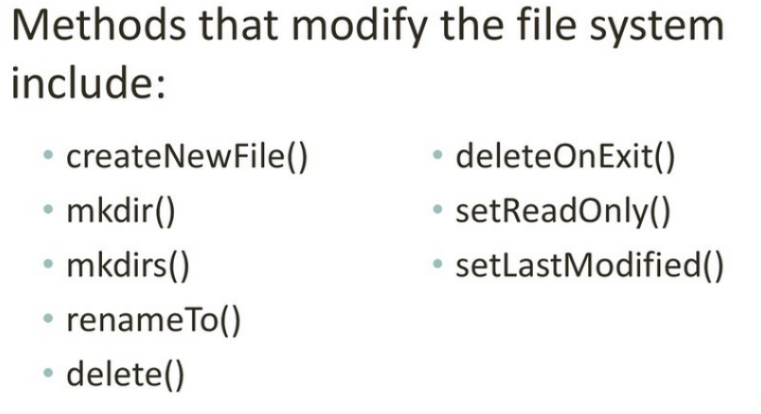
VD:

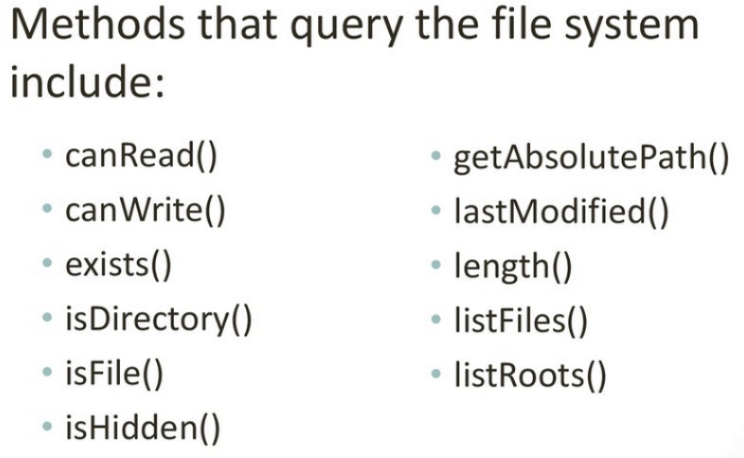
|  |
| --- |
| FileReader fr = **null**; BufferedReader br = **null**; **try**{  fr = **new** FileReader(**"D:\\mau.txt"**);  br = **new** BufferedReader(fr);  **int** i;  **while** ((i = br.read()) != -1) {  System.***out***.print((**char**) i);  } }**catch** (Exception e){  e.printStackTrace(); } **finally** {  **try** {  br.close();  } **catch** (IOException e) {  e.printStackTrace();  }  **try** {  fr.close();  } **catch** (IOException e) {  e.printStackTrace();  } } |

1. Lớp File

Đối tượng File biểu diễn file/thư mục thực sự trên đĩa. Lớp File trong Java biểu diễn các **pathname** của các file và thư mục theo phương thức trừu tượng. Lớp này được sử dụng để tạo các file và thư mục, để tìm kiếm các file, xóa các file, …

|  |  |
| --- | --- |
| File(File parent, String child); | Cú pháp sau tạo một đối tượng File mới từ một pathname trừu tượng cha và một chuỗi pathname con. |
| File(String pathname) | Cú pháp sau tạo một đối tượng File mới bởi việc biến đổi chuỗi pathname đã cho thành một pathname trừu tượng. |
| File(String parent, String child) | Cú pháp sau tạo một đối tượng File mới từ một chuỗi pathname cha và một chuỗi pathname con. |
| File(URI uri) | Cú pháp sau tạo một đối tượng File mới bởi việc biến đổi URI đã cho thành một pathname trừu tượng |





* Đường dẫn, file hiệu lực:

+ Trong Unix: “/u01/data/file.txt”

+ Trong Windows: C:\fpt\data.txt

* “C:\\fpt\\data.txt”