Chương 5: Luồng và tập tin

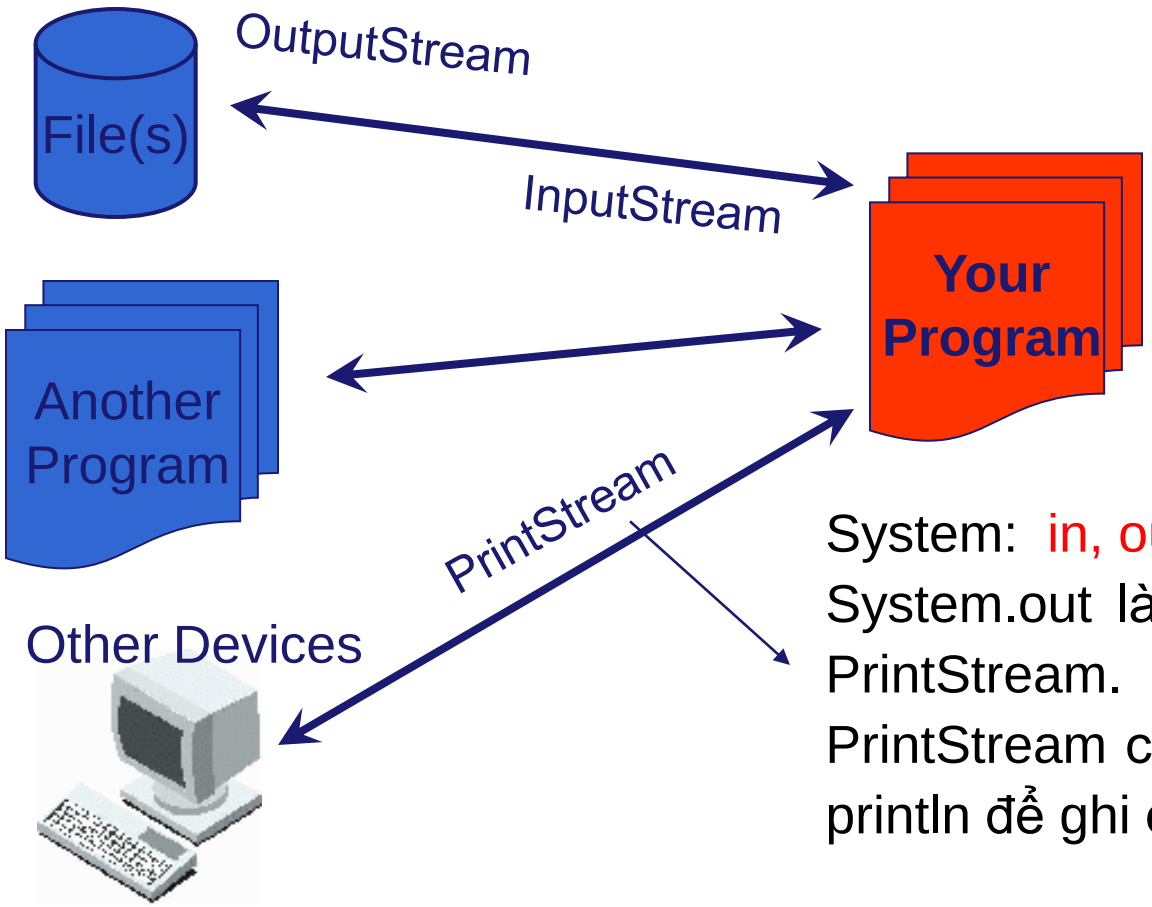
ThS. Trần Văn Hữu

Nội dung

- Nhập xuất dữ liệu
- Khái niệm về luồng dữ liệu
- Luồng và tệp
  - Luồng byte
  - Luồng ký tự
- Lớp File
- Truy cập tệp tuần tự
- Truy cập tệp ngẫu nhiên

Nhập/Xuất dữ liệu (1)

- Nhập xuất dữ liệu trong Java dựa trên mô hình luồng dữ liệu

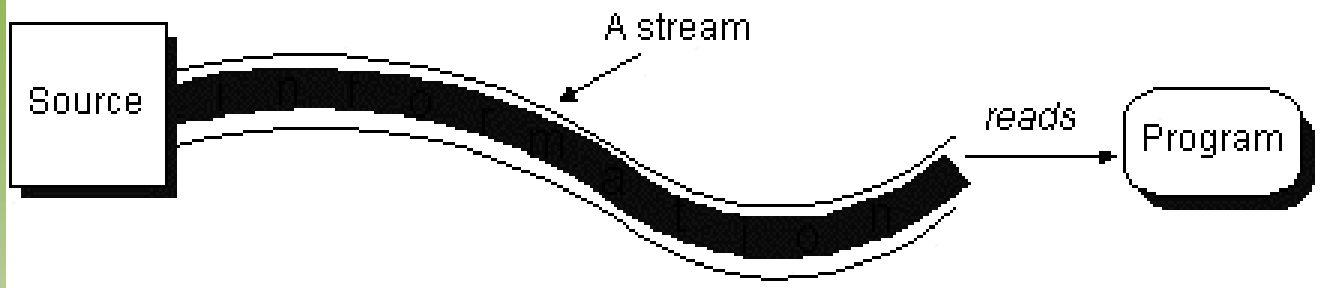


System: in, out

System.out là 1 thể hiện của lớp PrintStream.

PrintStream có phương thức print, println để ghi dữ liệu xuống luồng.

Nhập/Xuất dữ liệu (2)



Đọc dữ liệu

Open a Stream

While more Information

Read

Close the Stream

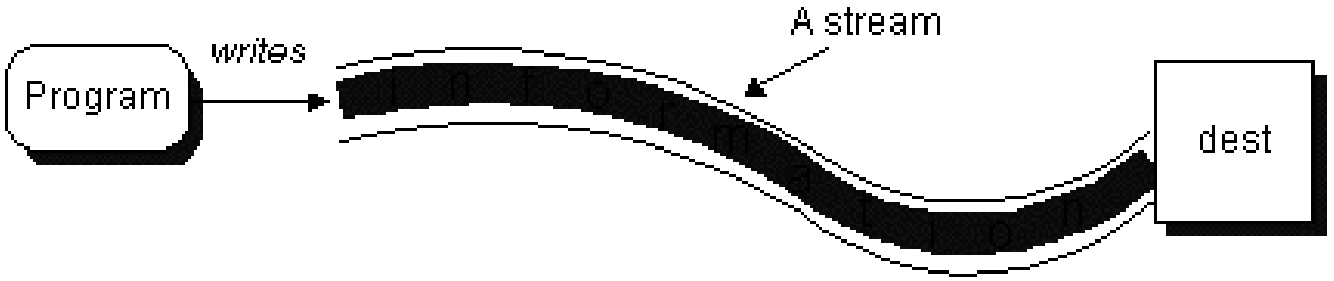
Ghi dữ liệu

Open a Stream

While more Information

Write

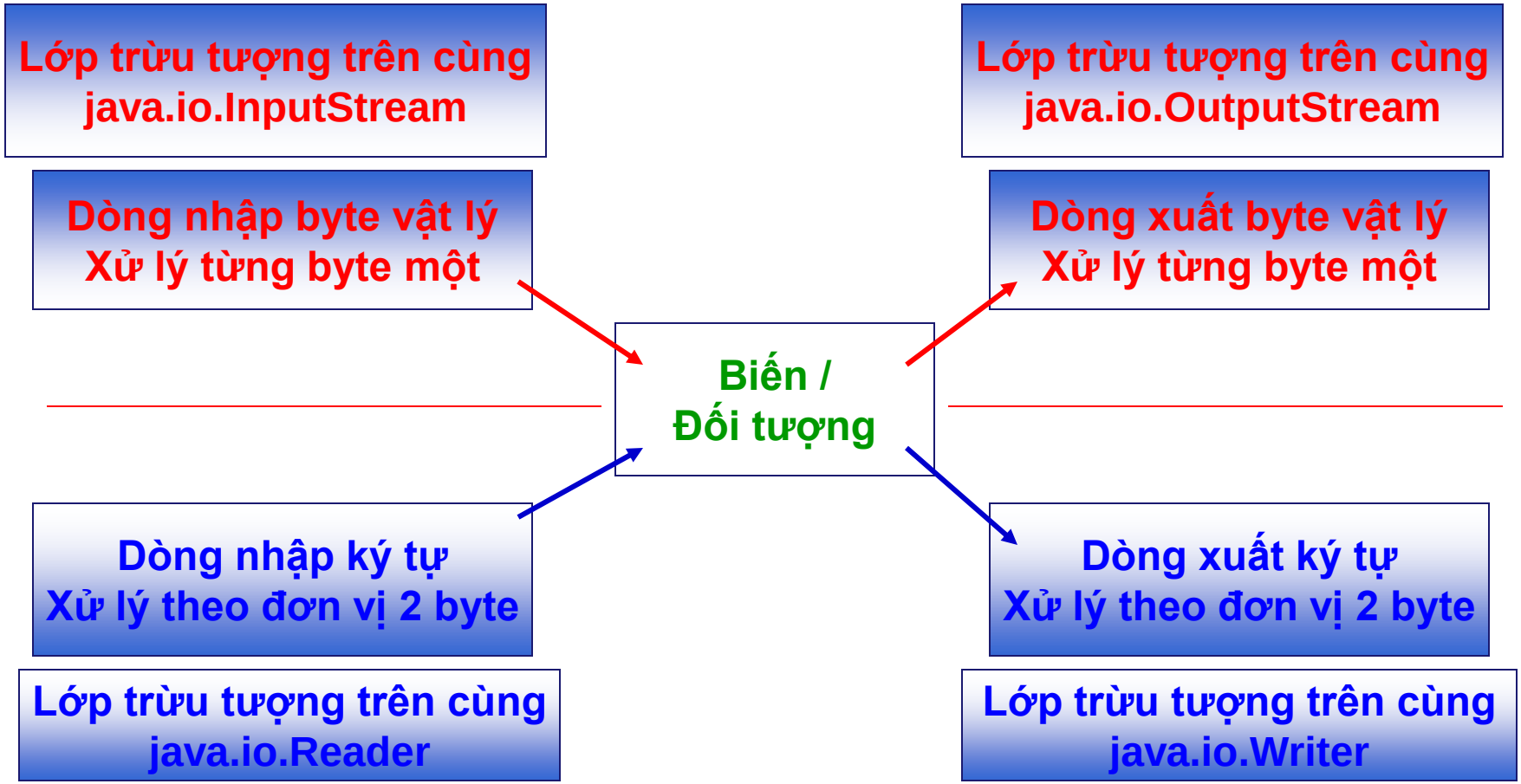
Close the Stream



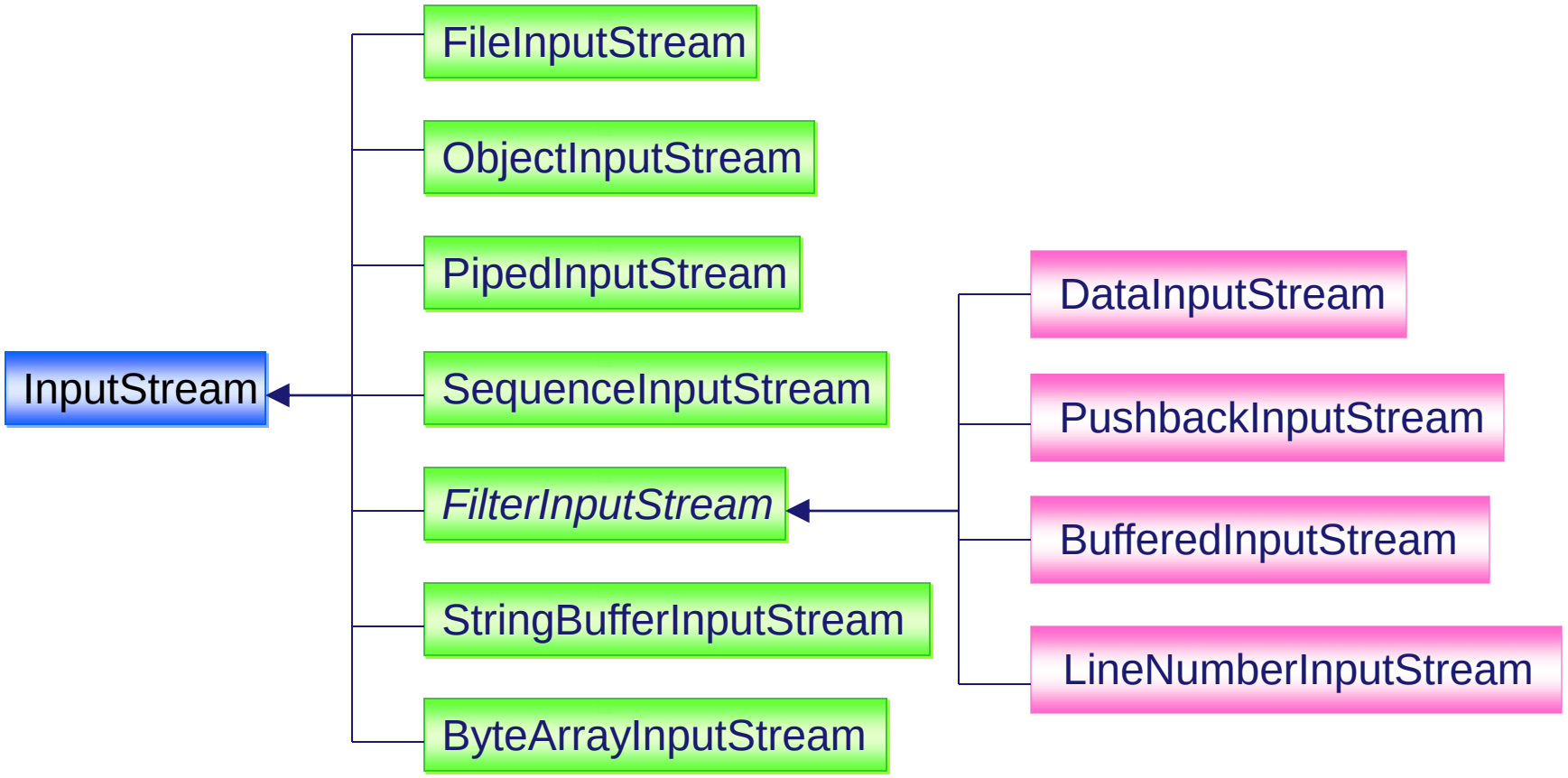
Luồng dữ liệu (data streams)

- Chương trình Java nhận và gửi dữ liệu thông qua các đối tượng thuộc một kiểu luồng dữ liệu nào đó.
- Luồng (stream) là một dòng dữ liệu đến từ một nguồn (source) hoặc đi đến một đích (dest)
  - Nguồn và đích có thể là tệp (file), bộ nhớ, một tiến trình (process), hay thiết bị (bàn phím, màn hình,…)
  - Luồng nhập & luồng xuất

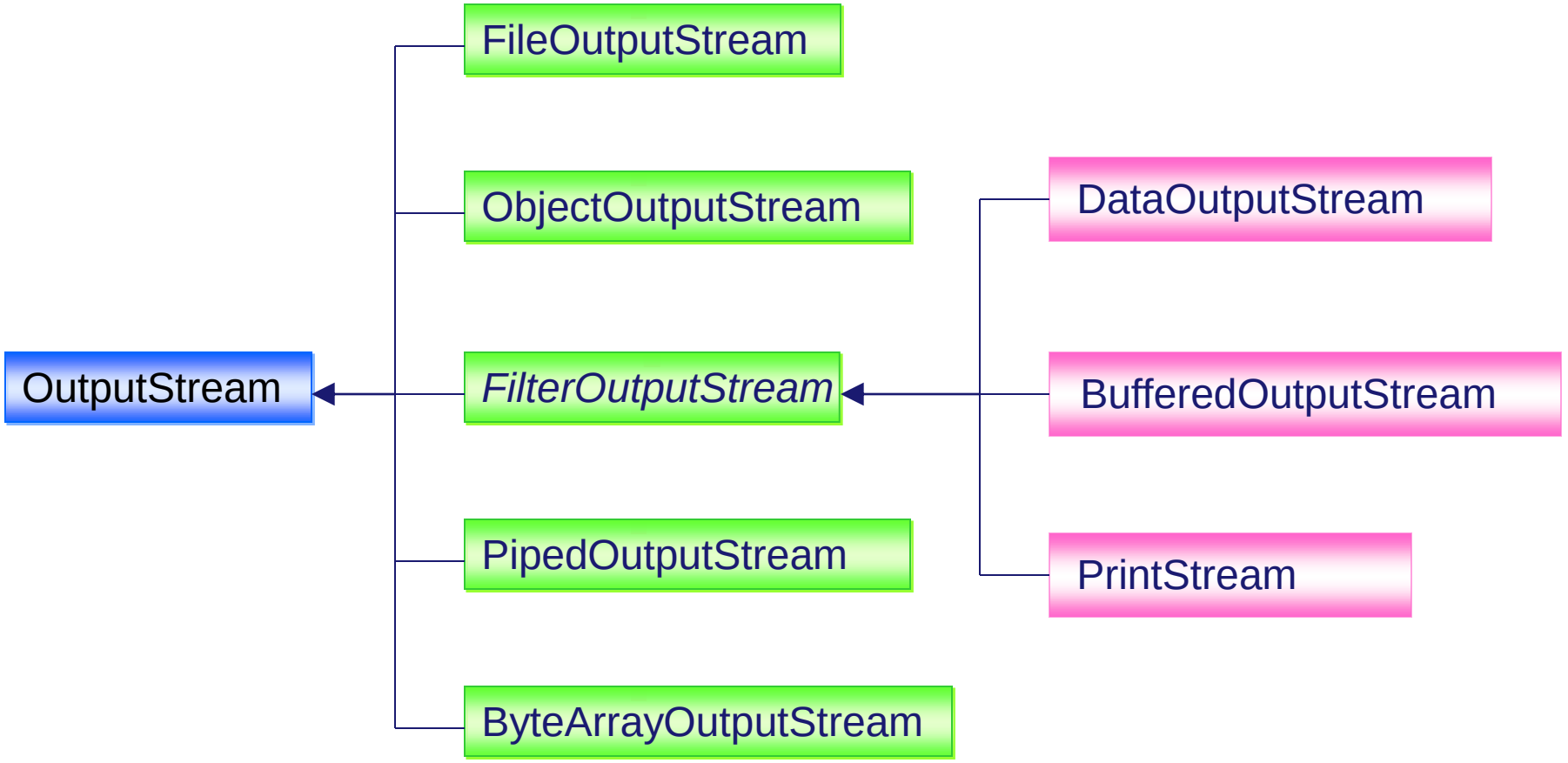
IO classes trong gói java.io



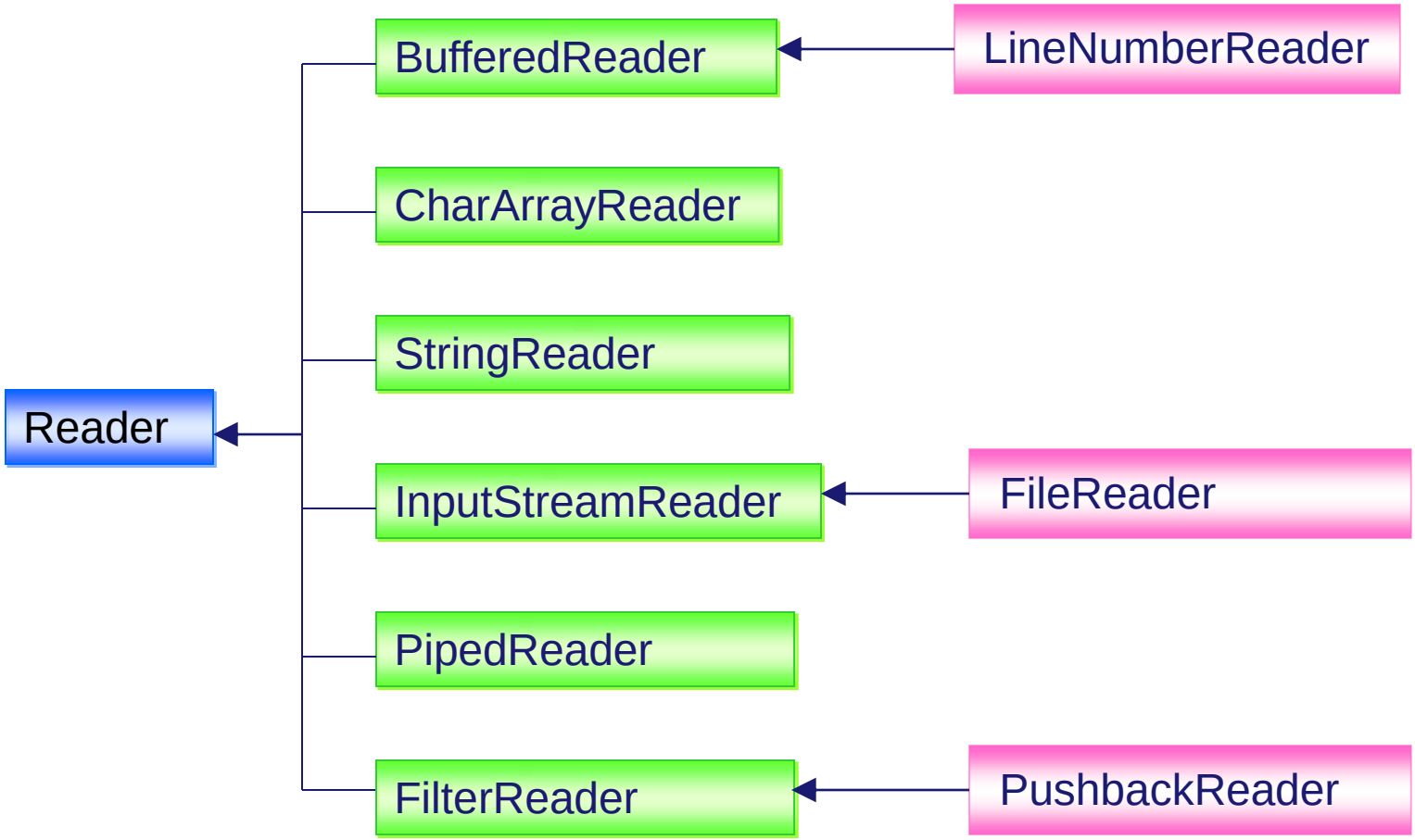
Luồng nhập – byte



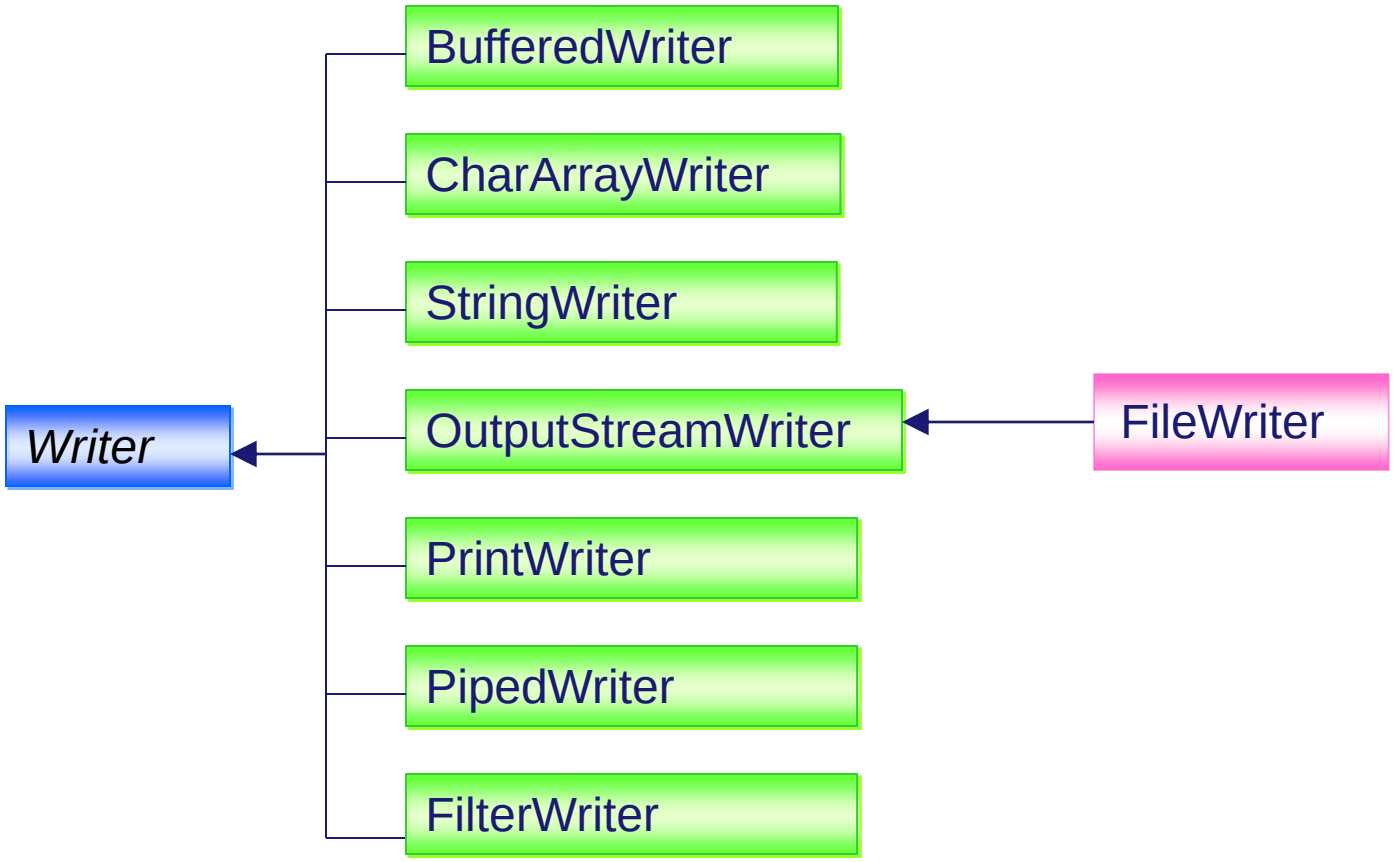
Luồng xuất – byte



Phân cấp các lớp nhập theo ký tự



Phân cấp các lớp xuất theo ký tự



InputStream/OutputStream

- Là hai lớp trừu tượng định nghĩa những thao tác truy xuất dữ liệu cơ bản (mức khái quát) theo từng byte vật lý mà không phân biệt nguồn dữ liệu là loại gì (file, chuỗi,…).
- Các lớp dẫn xuất từ hai lớp này nhằm cụ thể hóa các dòng nhập xuất byte vật lý tùy từng tình huống.

InputStream

- InputStream
  - int available()
  - void close()
  - void mark(int readlimit)
  - boolean markSupported()
  - int read()
  - int read(byte buf[])
  - int read(byte buf[], int offset, int len)
  - void reset()

OutputStream

- OutputStream
  - void close()
  - int write(int c)
  - int write(byte buf[])
  - int write(byte buf[], int offset, int len)

Lớp Reader & Writer

- Reader
  - int read()
  - int read(char buf[])
  - int read(char buf[], int offset, int len)
  - int read(CharBuffer target)
  - void close()
  - void mark(int readAheadLimit)
  - boolean markSupported()
  - boolean ready()

Lớp Reader & Writer

- Writer
  - int write(int c)
  - int write(char buf[])
  - int write(char buf[], int offset, int len)
  - void close()

Đối tượng vào/ra

- Để nhập hoặc xuất dữ liệu, chúng ta phải tạo ra đối tượng vào hoặc ra
- Đối tượng vào hoặc ra thuộc kiểu luồng tương ứng và phải được gắn với một nguồn dữ liệu hoặc một đích tiêu thụ dữ liệu

Sử dụng bộ đệm

- Bộ đệm → Kỹ thuật tăng tính hiệu quả của thao tác vào/ra
  - Đọc và ghi dữ liệu theo khối
  - Giảm số lần thao tác với thiết bị
- Thay vì ghi trực tiếp tới thiết bị → ghi lên bộ đệm
  - Khi bộ đệm đầy, dữ liệu được ghi ra thiết bị theo khối
  - Có thể ghi vào thời điểm bất kỳ bằng phương thức flush()
- Thay vì đọc trực tiếp từ thiết bị → đọc từ bộ đệm

Nhập xuất qua thiết bị chuẩn

- System.out cho phép in ra luồng ra chuẩn
  - Là đối tượng của lớp PrintStream
- System.err cho phép in ra luồng thông báo lỗi chuẩn
  - Là đối tượng của lớp PrintStream
- System.in cho phép đọc vào từ thiết bị vào chuẩn
  - Là đối tượng của lớp InputStream

Đọc dữ liệu từ luồng vào chuẩn

- System.in không sử dụng được trực tiếp
- Chúng ta muốn đọc một dòng ký tự
  1. Tạo đối tượng luồng ký tự (InputStreamReader)
  2. Tạo đối tượng luồng có bộ đệm (BufferedReader)

Ví dụ

```
InputStreamReader reader = new
                                    InputStreamReader(System.in);
BufferedReader in = new BufferedReader(reader);
. . .
String s;
try {
    s = in.readLine();
}
catch (Exception e) {
    ...
}
```

Đọc/Ghi kiểu dữ liệu nguyên thủy

- Để đọc/ghi các kiểu dữ liệu nguyên thủy, ta sử dụng luồng DataInputStream và DataOutputStream.
- Các phương thức được định nghĩa trong giao diện DataOutput:
  - void write(byte[] b)
  - void write(byte[] b, int off, int len)
  - void write(int b)
  - void writeBoolean(boolean v)
  - void writeBytes(String s)

Đọc/Ghi kiểu dữ liệu nguyên thủy

- Các phương thức được định nghĩa trong giao diện DataOutput: (tt)
  - void writeChar(int v)
  - void writeChars(String s)
  - void writeDouble(double v)
  - void writeFloat(float v)
  - void writeInt(int v)
  - void writeLong(long v)
  - void writeShort(int v)
  - void writeUTF(String str)

Đọc/Ghi kiểu dữ liệu nguyên thủy

- Các phương thức được định nghĩa trong giao diện DataInput:
  - boolean readBoolean()
  - byte readByte()
  - char readChar()
  - double readDouble()
  - float readFloat()
  - void readFully(byte[] b)
  - void readFully(byte[] b, int off, int len)
  - int readInt()

Đọc/Ghi kiểu dữ liệu nguyên thủy

- Các phương thức được định nghĩa trong giao diện DataInput: (tt)
  - String readLine()
  - long readLong()
  - short readShort()
  - int readUnsignedByte()
  - int readUnsignedShort()
  - String readUTF()
  - int skipBytes(int n)

Ví dụ - Đọc/Ghi dữ liệu nguyên thủy

```
import java.io.*;
public class DataIODemo {
    public static void main(String[] args) {
            try {
                    DataOutputStream out = new DataOutputStream(new
                                    FileOutputStream("D:/TestIO.txt"));
                    out.writeInt(10);
                    out.writeLong(123456789);
                    out.writeDouble(123.456789);
                    out.writeFloat(123.456789f);
                    out.writeBoolean(true);
                    out.writeUTF("Day la mot xau ki tu");
                    out.close();
            }
            catch (IOException e) {
                    System.out.println(e.getMessage());
            }
```

Ví dụ - Đọc/Ghi dữ liệu nguyên thủy

```
        //Đọc dữ liệu đã ghi từ file
        try {
                DataInputStream in = new DataInputStream(new
                                        FileInputStream("D:/TestIO.txt"));
                System.out.println("Gia tri nguyen " + in.readInt());
                System.out.println("Gia tri long " + in.readLong());
                System.out.println("Gia tri double " + in.readDouble());
                System.out.println("Gia tri float " + in.readFloat());
                System.out.println("Gia tri boolean " + in.readBoolean());
                System.out.println("Gia tri xau " + in.readUTF());
                in.close();
        }
        catch (IOException e) {
                System.out.println(e.getMessage());
        }
    }
}
```

Thao tác các kiểu đối tượng

- Luồng nhập/xuất đối tượng
  - Để lưu lại một đối tượng, ta có thể lưu lần lượt từng thuộc tính của nó. Khi đọc lại đối tượng ta phải tạo đối tượng mới từ các thuộc tính đã ghi.

    ➔ Không hiệu quả?
  - Java hỗ trợ đọc/ghi các đối tượng (object) một cách đơn giản hơn thông qua lớp ObjectInputStream và ObjectOutputStream.
  - Một đối tượng muốn có thể được đọc/ghi phải cài đặt giao tiếp java.io.Serializable.

Ví dụ - Đọc/Ghi đối tượng

❖ Ghi đối tượng ra file

```
try
{
    FileOutputStream f = new FileOutputStream("birthfile.dat");
    ObjectOutputStream oStream = new ObjectOutputStream(f);

    String babyName = "Briney Spears";
    Date today = new Date();
    oStream.writeObject(babyName);
    oStream.writeObject(today);

    oStream.close();
} catch (IOException e) {
    System.out.println("Error IO file");
}
```

Ví dụ - Đọc/Ghi đối tượng

❖ Đọc đối tượng từ file

```
try
{
    FileInputStream f = new FileInputStream("birthfile.dat");
    ObjectInputStream inStream = new ObjectInputStream(f);

    String name = (String) inStream.readObject();
    Date birthDate = (Date) inStream.readObject();

    System.out.println("Name of baby: " + name);
    System.out.println("Birth date: " + birthDate);

    inStream.close();
} catch (IOException e) {
    System.out.println("Error IO file");
}
```

Ví dụ - Đọc/Ghi đối tượng tự tạo

- class Student.java

```
import java.io.Serializable;
public class Student implements Serializable{
    private String name;
    private int age;
    public Student(String name, int age){
        this.name = name;
        this.age = age;
    }
    public String getInfo(){
        String ret="My name is "+name+"\nI am "+age+" years old.";
        return ret;
    }
}
```

Ví dụ - Đọc/Ghi đối tượng tự tạo

❖ Ghi đối tượng tự tạo

```
try {
    FileOutputStream f = new FileOutputStream("student.dat");
    ObjectOutputStream oStream = new ObjectOutputStream(f);
    Student x = new Student("Nguyen Thi Tam", 18);
    oStream.writeObject(x);
    Student y = new Student("Phan Dinh Tung", 21);
    oStream.writeObject(y);
    oStream.close();
}catch (Exception ex)
 {
    System.out.println (ex.getMessage());
}
```

Ví dụ - Đọc/Ghi đối tượng tự tạo

❖ Đọc đối tượng tự tạo

```
try {
    FileInputStream g = new FileInputStream("student.dat");
    ObjectInputStream inStream = new ObjectInputStream(g);
    Student a = (Student)inStream.readObject();
    System.out.println (a.getInfo());
    Student b = (Student)inStream.readObject();
    System.out.println (b.getInfo());
    inStream.close();
}catch(ClassNotFoundException ex){
    System.out.println (ex.getMessage());
}catch(Exception ex){
    System.out.println (ex.getMessage());
}
```

Phân cấp các lớp thao tác File

- java.lang.**Object**
  - java.io.**File** (implements java.lang.Comparable<T>, java.io.Serializable)
  - java.io.**FileDescriptor**
  - java.io.**RandomAccessFile** (implements java.io.Closeable, java.io.DataInput, java.io.DataOutput)

- Lớp File: Giúp truy xuất các thuộc tính của 1 file/thư mục.
- Lớp FileDescriptor: Giúp đồng bộ việc truy xuất file.
- Lớp RandomAccessFile: Giúp đọc/ghi file ngẫu nhiên

Lớp File

- Một trong các nguồn và đích dữ liệu thông thường là tập tin
- Lớp File cung cấp các chức năng cơ bản để thao tác với tập tin.
  - Nằm trong gói java.io
  - Tạo tập tin, mở tập tin, các thông tin về tập tin và thư mục
  - Cho phép lấy thông tin về file và thư mục

Tạo đối tượng File

- File myFile;
- myFile = new File(“data.txt”);
- myFile = new File(“myDocs”, “data.txt”);
- Thư mục cũng được coi như là một tệp
  - File myDir = new File(“myDocs”);
  - File myFile = new File(myDir, “data.txt”);
  - Có phương thức riêng để thao tác với thư mục

Các phương thức

- Tên tệp
  - String getName()
  - String getPath()
  - String getAbsolutePath()
  - String getParent()
  - boolean renameTo(File newName)
- Kiểm tra tệp
  - boolean exists()
  - boolean canWrite()
  - boolean canRead()
  - boolean isFile()
  - boolean isDirectory()

Các phương thức (tt)

- Nhận thông tin
  - long lastModified()
  - long length()
  - boolean delete()
- Thư mục
  - boolean mkdir()
  - Boolean mkdirs()
  - String[] list()

Ví dụ: Hiển thị thông tin file

```
public class FileInfo {
    public static void main(String[] args){
     File file = new File("d:\\DemoUnicode.java");
     if (file.exists()){
            System.out.println ("Path is: " + file.getAbsolutePath());
            System.out.println ("It's size is: " + file.length());
            Date dateModifier = new Date(file.lastModified());
            System.out.println ("Last update is: " +dateModifier);
     }
     else
            System.out.println ("The file does not exist");
    }
}
```

Bài tập

- Viết chương trình hiển thị thông tin của một thư mục (tương tự lệnh DIR của DOS)

Thao tác với tệp tuần tự (1)

- Thao tác với tệp tuần tự thông qua luồng Byte
- Đọc từ tệp
  - FileInputStream: Đọc dữ liệu từ tệp
  - public void FileInputStream (String FileName)
  - public void FileInputStream (File file)
- Ghi ra tệp
  - FileOutputStream: Ghi dữ liệu ra tệp
  - public void FileOutputStream (String FileName)
  - public void FileOutputStream (File file)

Thao tác với tệp tuần tự (2)

- Thao tác với tệp tuần tự thông qua luồng ký tự
- Đọc từ tệp
  - FileReader: Đọc ký tự từ tệp
  - BufferedReader: Đọc có bộ đệm (đọc từng dòng readLine())
- Ghi ra tệp
  - FileWriter: Ghi ký tự ra tệp
  - PrintWriter: Ghi theo dòng (print() và println())

Ví dụ – Copy File

- Viết chương trình file copy, thực hiện việc copy một tệp sử dụng cả 2 luồng hướng byte và hướng kí tự

Thao tác với tệp ngẫu nhiên

- Dùng lớp RandomAccessFile trong gói java.io
- Lớp này triển khai giao diện InputData và OutputData, nên chúng có tất cả các phương thức của cả 2 lớp này, ngoài ra chúng còn có các phương thức sau:
  - public void seek(long pos)
  - public long getFilePointer()
  - public long length()
  - public void writeChar(int v)
  - public final void writeChars(String s)

RandomAccessFile

- Tương tự C/C++ khi mở một tệp truy cập ngẫu nhiên bạn phải chỉ rõ chế độ làm việc.
  - Đọc: ‘r’
  - Ghi: ‘w’
  - Đọc ghi đồng thời: ‘rw’.
  - Ví dụ muốn mở tệp a.txt theo chế độ đọc ghi đồng thời:

```
RandomAccessFile = new RandomAccessFile(“C:/ a.txt”, “rw”)
```

RandomAccessFile – Ví dụ

```
import java.io.RandomAccessFile;
public class rndexam {
    public static void main(String[] args) {
        try {
                RandomAccessFile rf;
                rf = new RandomAccessFile("D:/BTJava/rdFile.txt", "rw");
                rf.writeBoolean(true);
                rf.writeInt(67868);
                rf.writeChars("J");
                rf.writeDouble(678.68);
                rf.seek(1);
```

RandomAccessFile – Ví dụ

```
                System.out.println(rf.readInt());
                System.out.println(rf.readChar());
                System.out.println(rf.readDouble());
                rf.seek(0);
                System.out.println(rf.readBoolean());
                rf.close();
        }
        catch (Exception ex) {
                System.out.print(ex.getMessage());
        }
    }
}
```

Bài tập

1. Viết chương trình cho phép đọc các thông số của một tập tin và hiển thị ra màn hình
2. Viết chương trình cho phép tạo ngẫu nhiên n số nguyên, mỗi số có giá trị từ 0→999 và ghi vào tập tin. Sau đó đọc và xuất ra màn hình
3. Viết chương trình hỗ trợ chức năng nén tập tin.

Hỏi & đáp