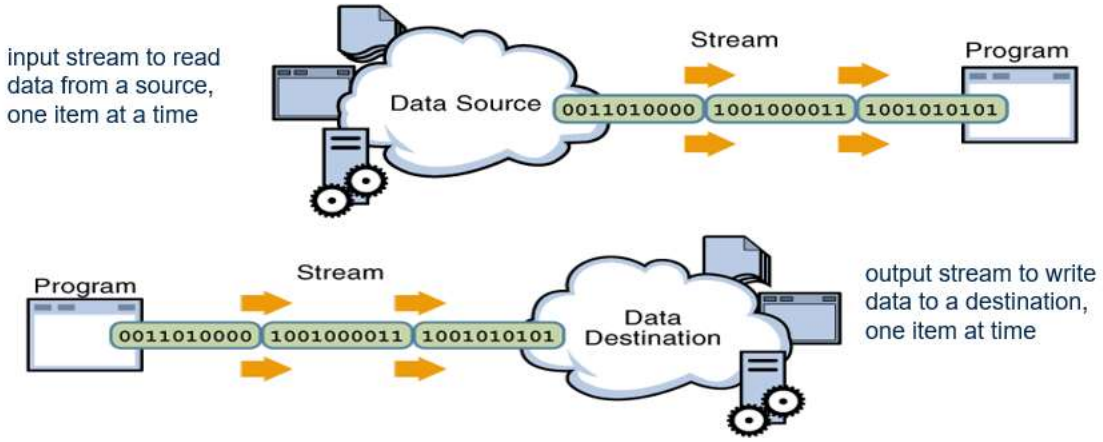
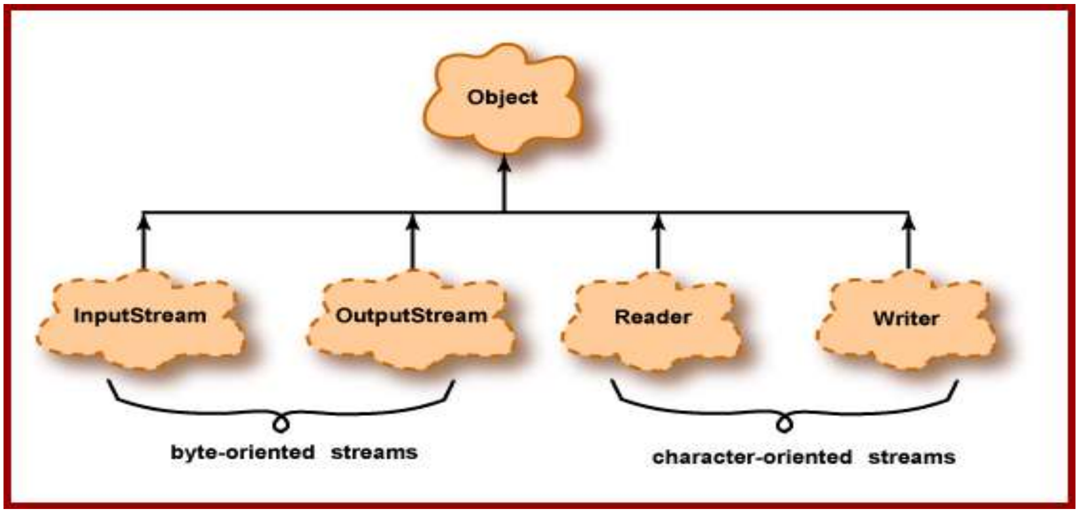
Contents

File IO

I/O streams


input stream to read data from a source, one item at a time
Data Source
Stream
Program
0011010000
1001000011
1001010101
Program
Stream
0011010000
1001000011
1001010101
Data Destination
output stream to write data to a destination, one item at time

Types of Streams



- Byte Streams: Handle I/O of raw binary data.
- Character Streams: Handle I/O of character data.
- Buffered Streams: Optimize input and output with reduced number of calls to the native API.
- Data Streams: Handle binary I/O of primitive data type and String values.
- Object Streams: Handle binary I/O of objects.
- Scanning & Formatting: Allows program to read and write formatted text.

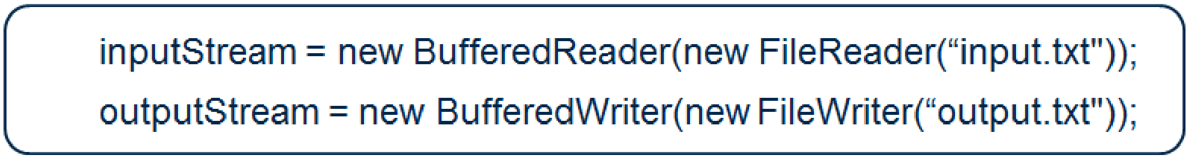
Methods of I/O Stream

Methods of Input Stream	**Methods of Output Stream**
close, read	close, flush, write,

Predefined Streams

- System.out :refers to the standard output stream
- System.err :refers to standard error stream
- System.in : refers to standard input

Buffered Streams



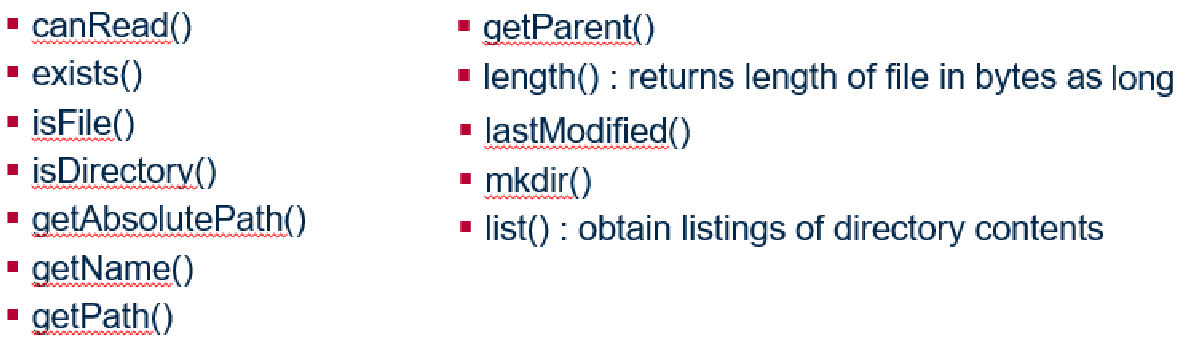
```
inputStream = new BufferedReader(new FileReader(“input.txt"));
outputStream = new BufferedWriter(new FileWriter(“output.txt"));
```

File Class

File class doesn’t operate on streams

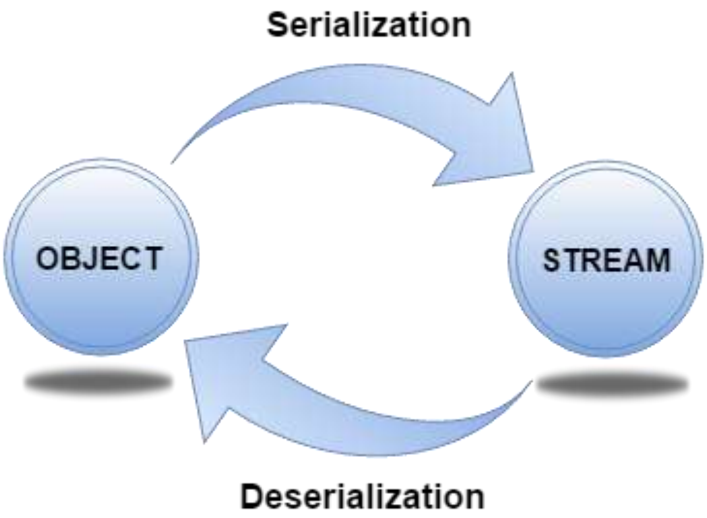
Represents the pathname of a file or directory in the host file system

Methods of File Class :



- canRead()
- exists()
- isFile()
- isDirectory()
- getAbsolutePath()
- getName()
- getPath()
- getParent()
- length() : returns length of file in bytes as long
- lastModified()
- mkdir()
- list() : obtain listings of directory contents

Java.io.Serializable



- Serialization is a process to read and write objects.
- It provides ability to read or write a whole object to and from a raw byte stream.
- Use object serialization in the following ways:
    - Remote Method Invocation (RMI): Communication between objects via sockets.
    - Lightweight persistence: Archival of an object for use in a later invocation of the same program.