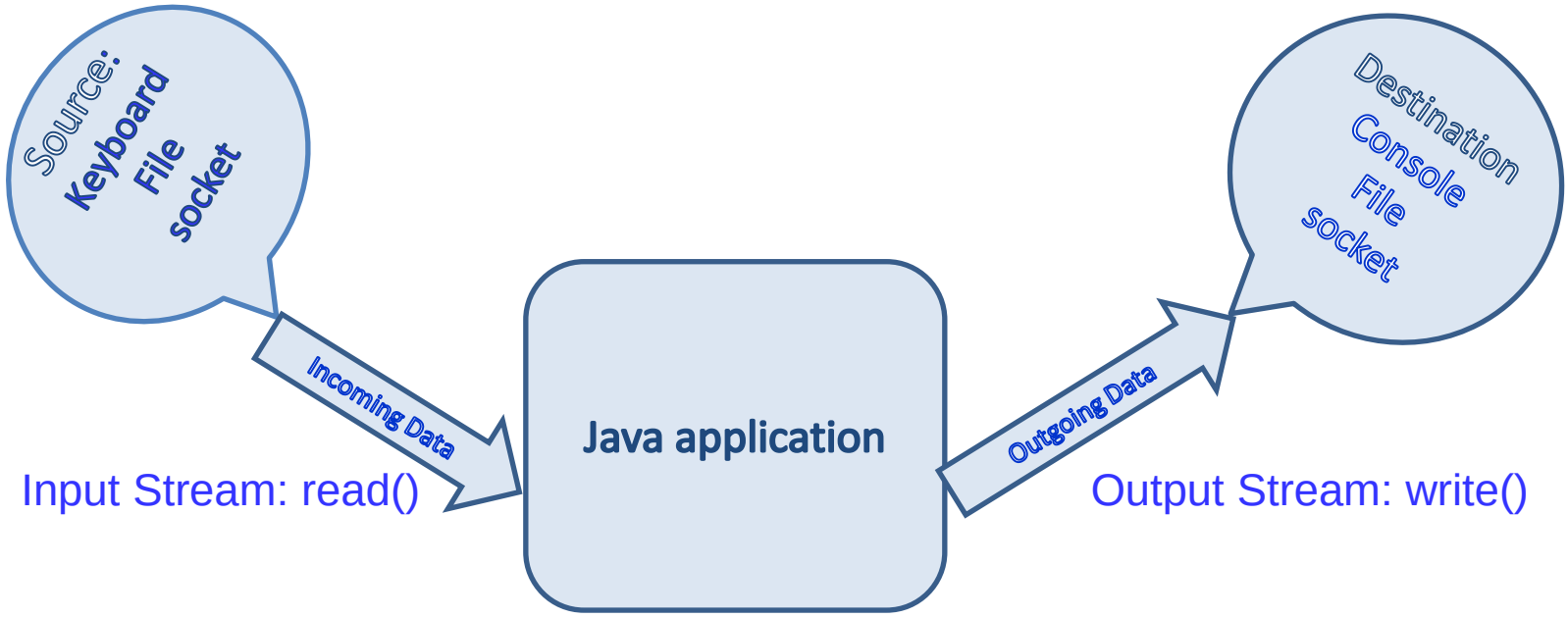
Agenda

- I/O Fundamentals
- Stream Classes
- Input and Output Stream
- File I/O
- File Writer
- File Stream
- Scanner
- Console Input - Output

Basic IO operations



Flow of data is depicted as Stream

Input Stream : Flow of data from a source to java application

Output Stream: Flow of data from application to some destination

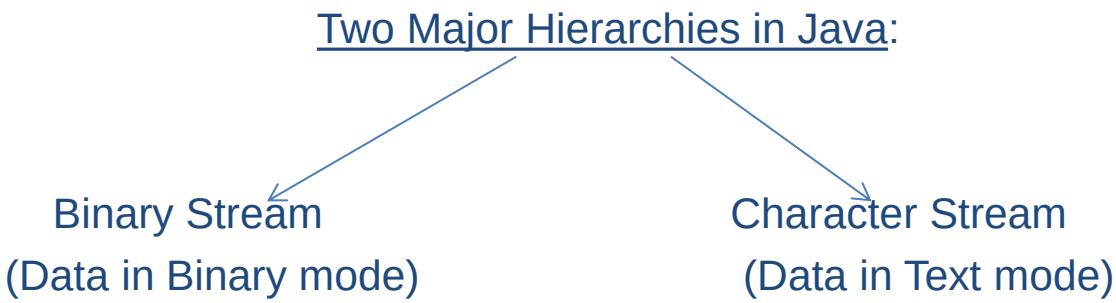
I/O Fundamentals

I/O Fundamentals

- A stream can be thought of as a flow of data from a source or to a sink.
- A source stream initiates the flow of data, also called an input stream.
- A sink stream terminates the flow of data, also called an output stream.

- Sources and sinks are both node streams.

IO



- Java technology supports two types of streams: *character and byte*.
- Input and output of *character* data is handled by *readers and writers*.
- Input and output of byte data is handled by *Input streams and output streams*:
- Normally, the term stream refers to a byte stream.
- The terms reader and writer refer to character streams.

Stream Classes

Fundamental Stream Classes

InputStreamReader
OutputStreamWriter

The InputStream Methods

The InputStream Methods

- The three basic read methods are:

int read()

int read(byte[] buffer)

int read(byte[] buffer, int offset, int length)

The OutputStream Methods

The OutputStream Methods

- The three basic write methods are:

void write(int c)

void write(byte[] buffer)

void write(byte[] buffer, int offset, int length)

- Other methods include:

void close()

void flush()

The Reader Methods

The Reader Methods

- The three basic read methods are:

int read()

int read(char[] cbuf)

int read(char[] cbuf, int offset, int length)

The Reader class is an abstract class for reading character streams which is available in the java.io package.

An InputStreamReader is a bridge from byte streams to character streams i.e. it reads bytes and decodes them into Unicode characters according to a particular platform

The BufferedReader class is the subclass of the Reader class. It reads character-input stream data from a memory area.

The Writer Methods

The Writer Methods

- The basic write methods are:

void write(int c)

void write(char[] cbuf)

void write(char[] cbuf, int offset, int length)

void write(String string)

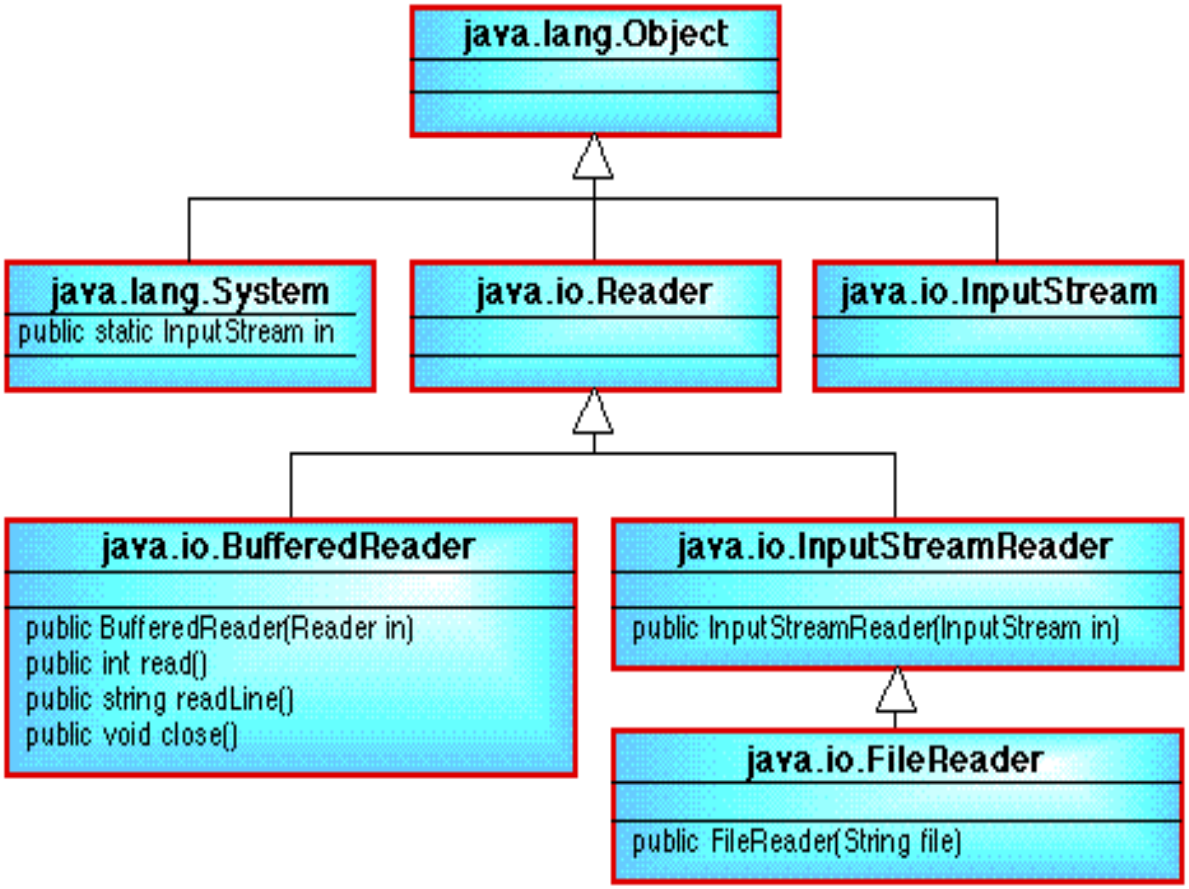
void write(String string, int offset, int length)

- Other methods include:

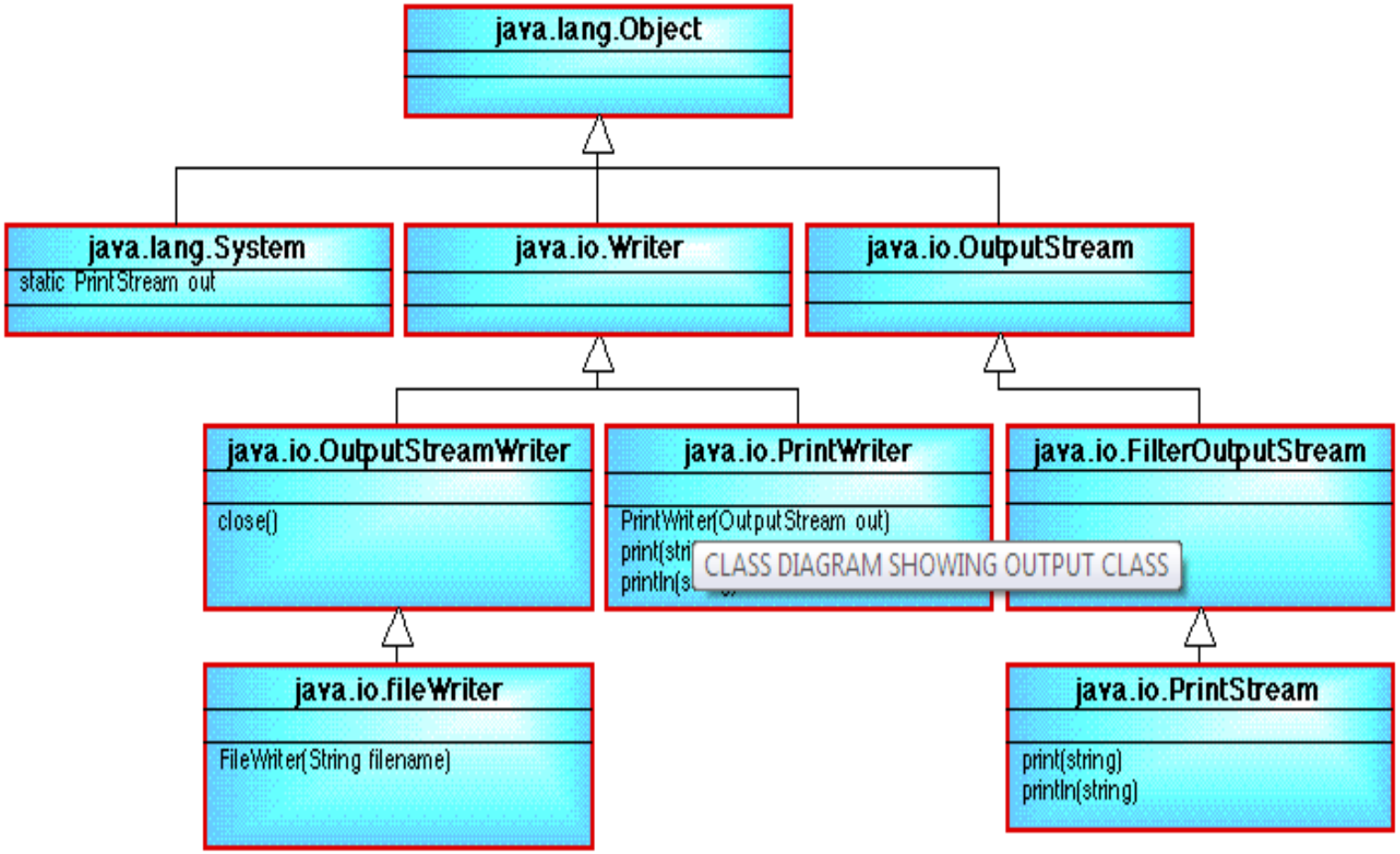
void close()

void flush()

Reader Classes



Writer Classes



File Stream I/O

File Stream I/O

- For file input:
  - Use the FileReader class to read characters.
  - Use the BufferedReader class to use the readLine method.
- For file output:
  - Use the FileWriter class to write characters.
  - Use the PrintWriter class to use the print and println methods.

Files and File I/O

The java.io package enables you to do the following:

- Create File objects
- Manipulate File objects
- Read and write to file streams

Files and File I/O

Write to file

At this point we have a writer object and we can send real content to the file. You can do this using the write() method, which has more variant but the most commonly used requires a string as input parameter.

Calling the write() method doesn't mean that it immediately writes the data into the file. The output is maybe cached so if you want to send your data immediately to the file you need to call the flush() method.

As last step you should close the file with the close() method and you are done.

FileWriter

Open a file

To open a file for writing use the FileWriter class and create an instance from it.
The file name is passed in the constructor like this:

```
FileWriter writer = new FileWriter(fileName);
```

This code opens the file in overwrite mode. If you want to append to the file then you need to use an other constructor like this:

```
FileWriter writer = new FileWriter(fileName,true);
```

Besides this the constructor can throw an IOException so we put all of the code inside a try-catch block.

FileWriter :

FileWriter is a subclass of OutputStreamWriter class that is used to write text (as opposed to binary data) to a file.

The FileWriter class creates an internal FileOutputStream to write bytes to the specified file

BufferedWriter :

The BufferedWriter class is used to write text to a character-output stream

```
FileWriter fstream = new FileWriter(file_name);
BufferedWriter out = new BufferedWriter(fstream);
```

Standard Streams

Standard Streams : feature provided by many operating systems.

- By default, they read input from the *keyboard*
- write output to display.
- They also support I/O operations on files.

- *System.in* : used to read input from the keyboard.
- *System. Out* : used to write output to display.
- *System. Err* : used to write error output

Console I/O

Console I/O

The variable System.out enables you to write to standard output.

It is an object of type PrintStream.

The variable System.in enables you to read from standard input.

It is an object of type InputStream.

The variable System.err enables you to write to standard error.

It is an object of type PrintStream.

System.in is a byte stream that has no character stream features.

To use Standard Input as a character stream, wrap System.in within the InputStreamReader as an argument.

```
InputStreamReader inp = new InputStreamReader(system.in);
```

Scanner

- Scanner : a final class introduced in Java 5.0
- The scanner API provides basic input functionality for reading data from the system console or any data stream
- java.util.Scanner class can be used to accomplish the same thing but with less code
- This is an enhanced input functionality, as it makes easy to read primitive data types from the keyboard
- A simple text scanner which can parse primitive types and strings using regular expressions
- A Scanner breaks its input into tokens using a delimiter pattern.
  (default: whitespace)

Java 6 Features….. Console Class

Interaction with the user in the command line environment :

1**. through the Standard Streams**

- they read input from the keyboard and write output to the display
- this feature is controlled by the command line interpreter, not the program
- The Java platform supports three Standard streams as System.in ,System.out ,Sysytm.err

2**. through the Console (New Addition)**

- A more advanced alternative to the Standard Streams
- This is a single, predefined object of type Console
- The Console also provides input and output streams that are true character streams

Console Class continued..

- The Console is particularly useful for secure password entry through its readPassword method
  - First, it suppresses echoing, so the password is not visible on the user's screen.
  - Second, readPassword returns a character array, not a String, so the password can be overwritten, removing it from memory as soon as it is no longer needed.
- Before a program can use the Console, it must attempt to retrieve the Console object by invoking System.console()
- System.console will return null if console operations are not permitted
  - either because the OS doesn't support them

    or
  - the program was launched in a noninteractive environment.

Questions ?

Assignments