Java Input / Output

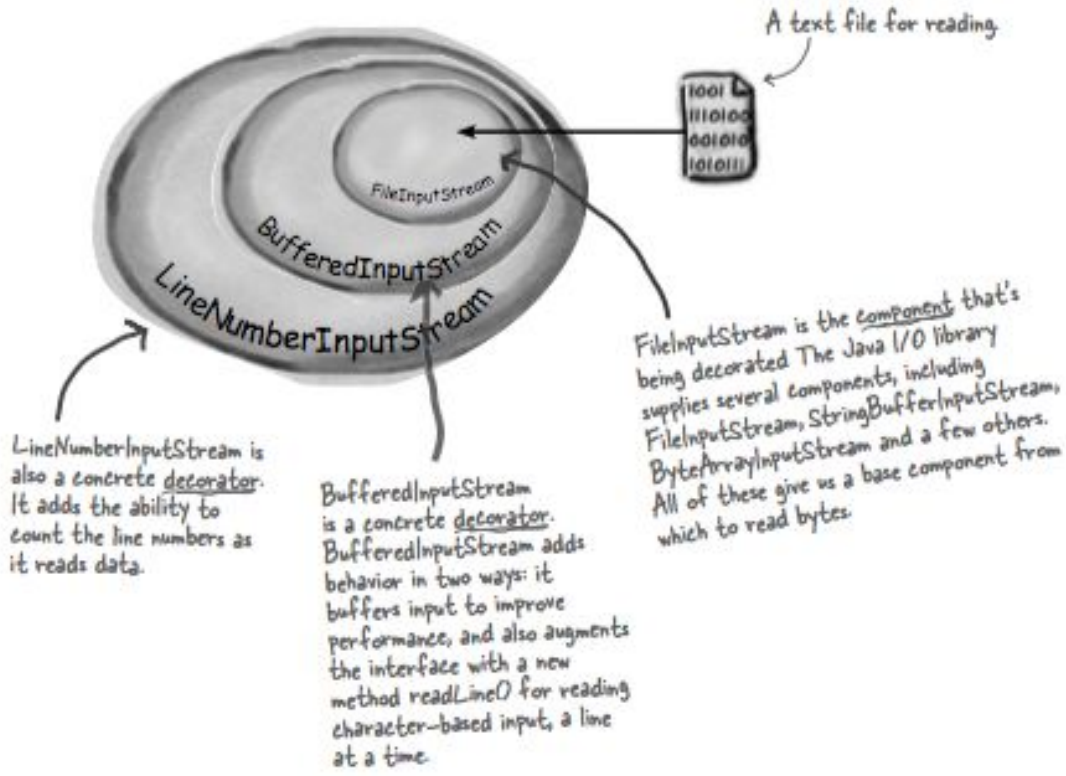
Java I/O

In Java I/O is built on streams (see https://docs.oracle.com/javase/tutorial/essential/io/streams.html)

Lower-level stream can be *decorated* with more functionality to provide higher abstraction level

InputStream (for example FileInputStream)	byte	
Character stream (FileReader)	character	
Line-oriented (BufferedReader)	lines	or
Scanner	flexible	

On decorator pattern



Head First Design Patterns, pages 79-107

Lab 05: Interpreter and history

Create Interpreter class.

```
Interpreter(InputStream in, PrintStream out, String name, History history) {
        this.in = in;
        this.out = out;
        this.name = name;
        this.history = history;
 }

public void run() {
          // loop until “:quit” is read from input stream and store all lines into history
          // store in history name (String) and inputText (String)
          // “:print” outputs whole history

          while (input not “:quit”) {
                    history.addEntry(new HistoryEntry(...));
                    if(input.equals(":print"))
                              out.println(history);
          }
}
```

Lab 05: Interpreter and history

Create Interpreter and **History classes. History should be singleton.**

```
private History() {
    private ArrayList<HistoryEntry> history;
    ...
}

public void addEntry(HistoryEntry e) {
        …
}

@Override
public String toString() {
        // output whole history as a String
}
```

Lab 05: Interpreter and history

Create Interpreter and History classes as well as **Main class.**

```
public static void main(String[] args) {
        Interpreter i = new Interpreter(System.in,
                                        System.out,
                                       "me",
                                       History.getInstance());
        i.run();
}
```