COMP6700/2140 Std. Lib., I/O

Alexei B Khorev and Joshua Milthorpe

Research School of Computer Science, ANU

February 2017

Topics

1. Standard Library — Java SE API
2. String, Character, package `java.lang`
3. Input and Output, package `java.io`
4. *Scanner* class
5. *Math* class
6. Package `java.util`, class *Random*

Importance of Standard Library

- **API is absolutely indispensable**
- *Oracle* vs *Google* case
- Java SE API documentation
  - Demo
  - Keep it under your electronic pillow (may be not ☺)

Package `java.lang`

It's always *there* (no import needed)

- Most important classes:
  - *String*
  - *Character*, *Integer* (and other wrapper classes for primitive types)

I/O business

- Simple ways
- Command-line arguments (not really an input)
- *Real* I/O

Getting input into program: command-line arguments

It is vital to control the data used and produced by a program, so called *input-output, I/O*. The simplest way to get your data into the program is to use the *command-line arguments*. They are picked up by those parameters args (the array of *String*s) which are passed to the main:

```
public class ArgsTest {
   public static void main( String[] args ) {
      System.out.println("first arg = " + args[0]);
      System.out.println("second arg = " + args[1]);
      System.out.println("third arg = " + args[2].toUpperCase());
   }
}
```

```
abx% java ArgsTest I love perignon
first arg = I
second arg = love
third arg = PERIGNON
```

args are *String*s, and if you intend to provide an integer (or float), the input string can be converted:

```
String s = "123"; int i = Integer.parseInt(s);
String s = "-123.45"; double d = Double.parseDouble(s);
```

Getting input into program: *Scanner*

The command-line arguments is a very rigid way to control input. It can be done only once, at the start of program execution.

Reading text from the terminal's standard input (`stdin`, or console) provides a better way: It allows you to control the moment of execution when the program will receive input data. One convenient way to provide such input channel into the program, is to create a *Scanner* object to read form `stdin` stream (`stdin` is the object *System.in*, and `stdout` is *System.out* — standard input and standard output). Details will be discussed in *J10*.

```java
import java.util.Scanner;

public class ConsoleTest {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      // the user is expected to type something and hit "return"
      System.out.print("Enter some text: ");
      String input = sc.next();
      System.out.println("You entered " + input +
                ", it has " + input.length() + " characters");
   }
}
```

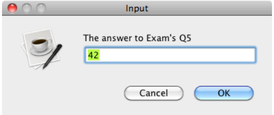
demo (the code is available in ConsoleTest.java)

Getting input into program: simple GUI

A simple input from the user can be alternatively provided using a *Graphical User Interface*:

```
import javax.swing.JOptionPane; // to make JOptionPane available

public class GUITest {
   public static void main( String[] args ) {
      String input = JOptionPane.showInputDialog("The answer to Exam's Q5");
      System.out.println( "Your answer to Q5 is " + input );
   }
}
```


Input

The answer to Exam's Q5

42

Running the program:

```
abx% java GUITest
Your answer to Q5 is 42
```

Math class

In every language, you need a collection of standard routines which every part of a program can easily call. For example — mathematical functions:

- sin, cos, tan and all other trigonometrical functions and their inverses (`asin` which is sin^{-1} *etc*)
- log, exp, power function x^y (denoted as pow(x,y)), $\sqrt{x}, \ldots$
- `min, max, floor, ceil, ...`

and others. Such “global” functions are implemented as `public static` methods inside a dedicated class, for mathematics case — `java.lang.Math`. If you need to use them by their usual ”short name without having to type `Math.sin(2*Math.PI*x)` all the time, include the *static import* at the beginning of your program:

```
import static java.lang.Math.sin;
import static java.lang.Math.PI;
```

Random class

Apart from java.lang and java.io there are *plenty* of other useful packages. One *extremely* important package is java.util, which contains *JFC* (Java Framework Collection, *ie* containers API), generic algorithms (like Sort and Search), functional interfaces and streams. *Scanner* — used above — is also from that package.

Another useful class from java.util is *Random*.

```java
import java.util.Random;

class RosencrantzAndGuildenstern {
    public static void main(String[] args) {
        int numberOfTosses = Integer.parseInt(args[0]);
        Random rand = new Random();
        for (int i = 0; i < numberOfTosses; i++)
            System.out
                  .print(rand.nextInt() > 0 ? "Heads " : "Tails ");
        System.out.println();
    }
}
```

```
abx% javac RosencrantzAndGuildenstern.java
abx% java RosencrantzAndGuildenstern 177
```

Where to look for this topic in the textbook?

- Hortsmann's *Core Java for the Impatient*, Ch. 1.6, 1.8.8, 1.7.2
- Also explore Java™ Platform, Standard Edition 8 API Specification