Object-Oriented Programming I

Interactivity

Slides by Magdin Stoica
Updates by Georg Feil

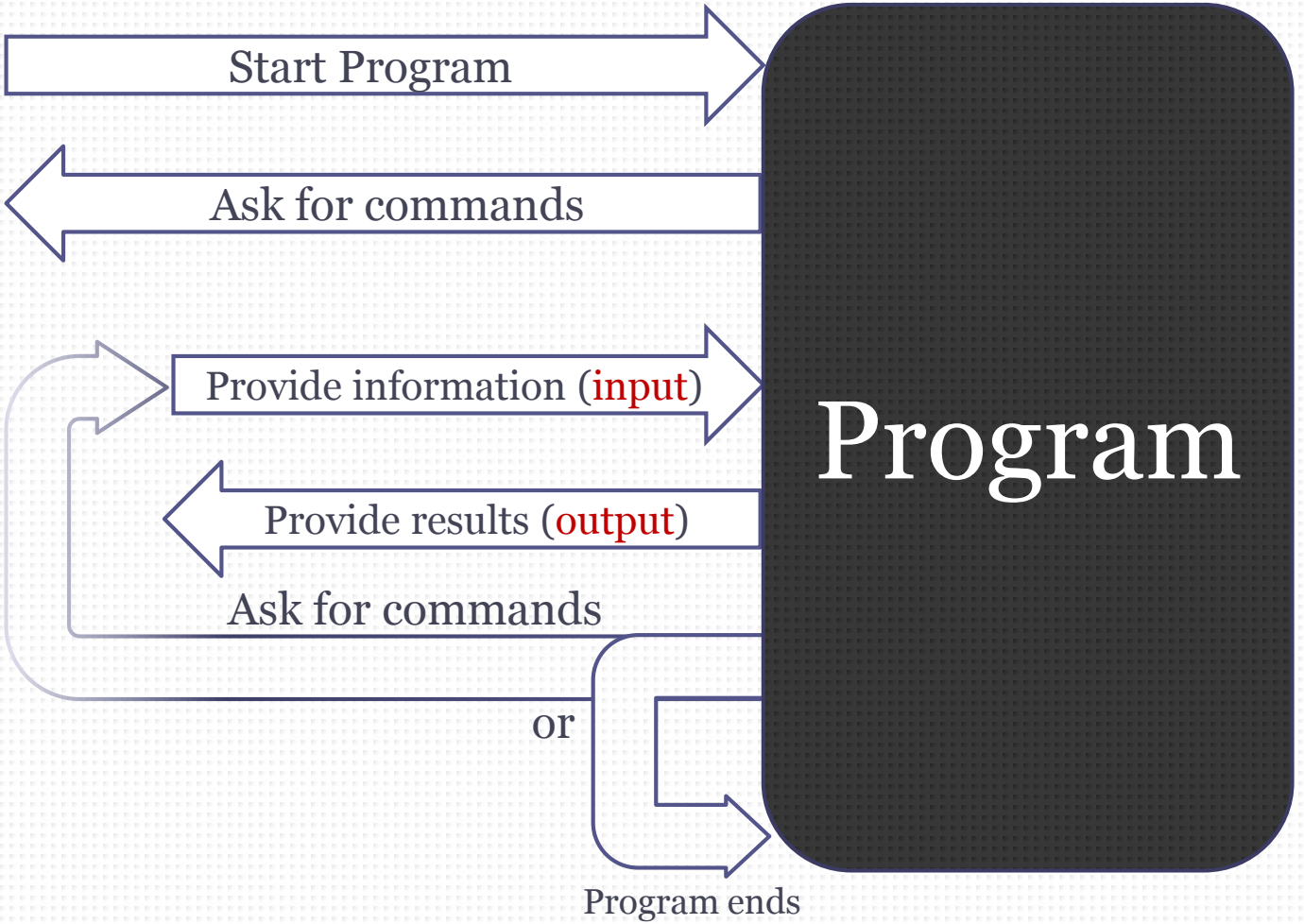
Interactivity - Learning Outcomes

1. Characterize interactivity and its role in computer programs
2. Define the purpose and the different types of user interfaces
3. Identify types of programs based on their interactivity aspects
4. Using predefined Java classes for reading input from the Console and writing output to the Console
5. Define the syntax of the import statement
6. User the import statement to import classes from another package

Reading Assignments

- Introduction to Java Programming (required)
  - Chapter 2: Elementary Programming
    - Section 2.3 only

Users and Programs



User Interface (UI)

- The set of "tools" used by the user to communicate with a program (application) is called "user interface" or UI
- The user interface of an application has two roles
  - Allows the user to provide information to the application (input)
  - Allows the application to provide results to the user (output)
- Input devices are computer devices used to provide input to the application: keyboard, mouse, touch
- Output devices are computer devices used to observe the output provided by the application: screen, printer
- The user interface may be part of your program or may be part of the computer's operating system (or both).

Types of Programs

- Programs can be categorized in terms of the type of user interface they provide:
  - Console programs use text-based user interfaces. The text is displayed on the computer's console or command prompt window
  - Graphical User Interface (GUI) Programs uses graphics like windows, menus, buttons, list-boxes, tables, check-boxes etc.
  - Web Programs have their user interfaces defined in HTML and displayed in a web browser
- Programs can be categorized in terms of their location relative to the user
  - Client programs run on the same computer the user is operating
  - Server programs run on a different computer the user is operating

Programming Journey in Context

Term 1

- Console Programs
  - Text-based user interface displayed inside the standard computer console
  - Are client programs, run on the same machine as the user
  - Only the keyboard is used as an input device

Term 2

- GUI Programs
  - Graphical User Interface: windows, buttons, check-boxes, drop-downs, tables, menus etc.
  - Also client programs, run on the same machine as the user
  - Programs can write any type of information anywhere on the screen: text, images, colours, shapes etc.

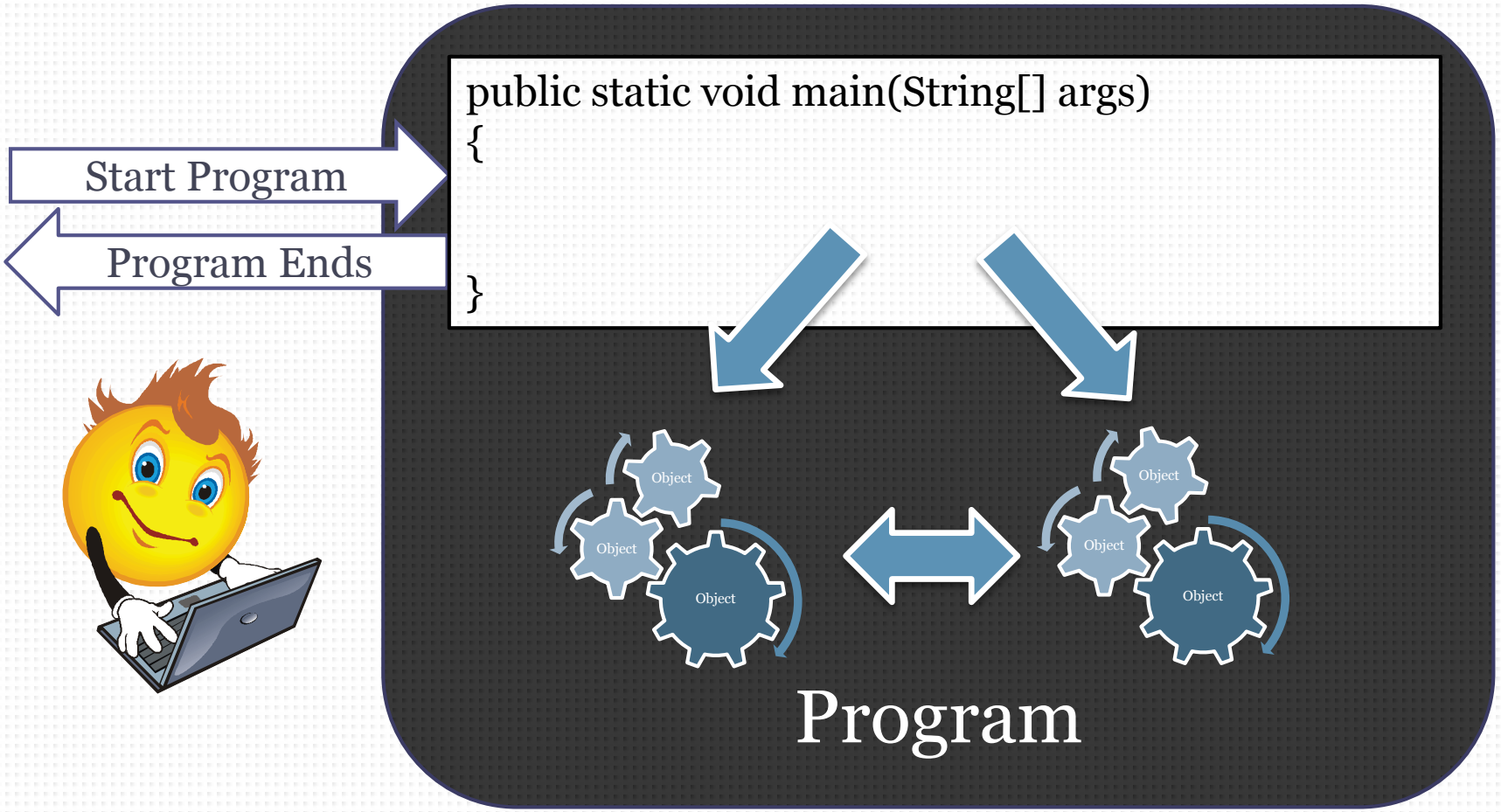
Term 3

- Web Programs
  - The user operates the program using a web browser
  - The UI is written in HTML
  - The actual program runs on a separate computer called a server

Console Programs

- Are client programs, run on the same machine as the user
- Text-based user interface displayed on the standard computer console
- Text flows sequentially from left to right, top to bottom.
- Program can only output new text sequentially
  - Program cannot go “back” and change something that has already been presented to the user
  - Program can only “print” or “output” new information
- Only the keyboard is used as an input device
  - User types inside the console text
  - Program can read the text and transform it and remember it in variables

Users and Programs



Reading input in a console program

- Reading the input entered by the user using the keyboard is done by using predefined classes from the Java library
  - Scanner: allows programs to scan the input and obtain typed information (String, int, double, etc)
  - System: provides general functionality like access to standard input and output, time, program properties etc.
  - InputStream: allows programs to read input information from a variety of sources, including the standard console
- Creating a scanner object involves declaring a variable of type Scanner and initializing the variable using the "new" operator

```
Scanner input = new Scanner(System.in);
```

- Once the scanner object is created and initialized the program can call methods to read the input provided by the user

Exercise 1 – Know your Java… Doc

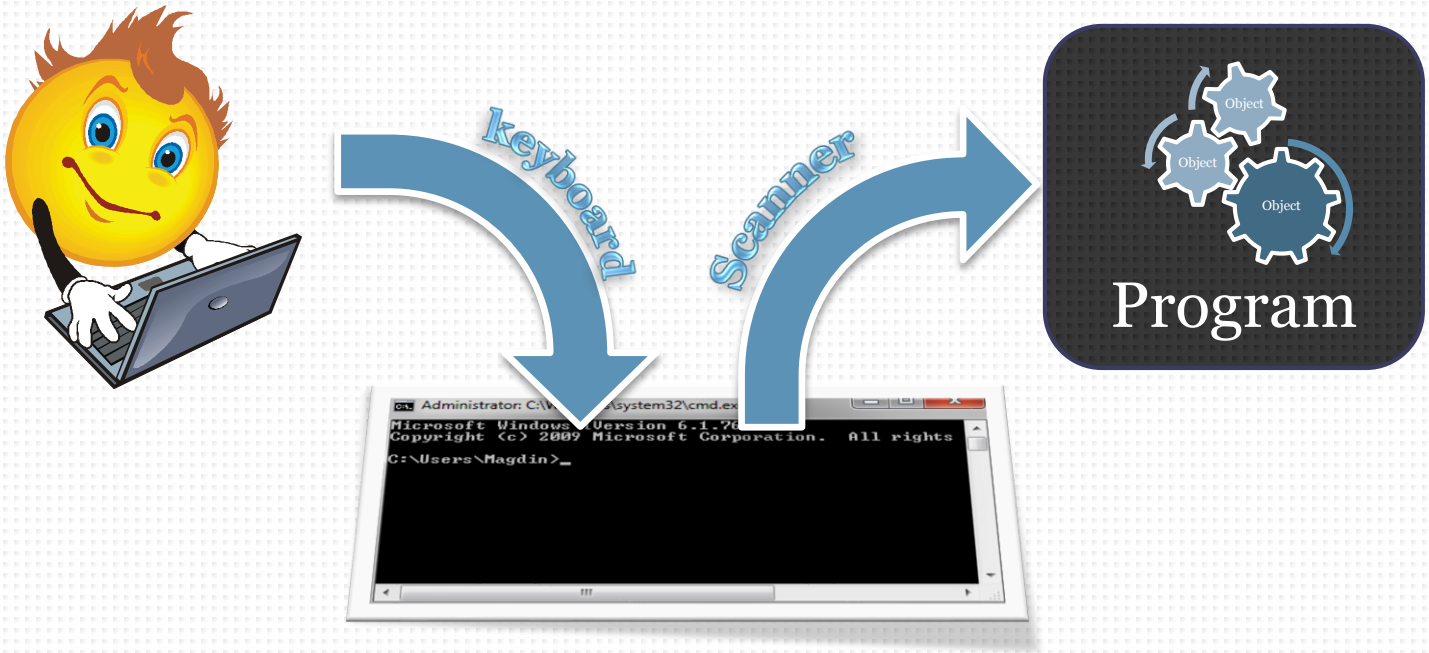
- The Java Language and all the JRE class libraries are documented using JavaDocs
  - HTML web sites providing full descriptions, examples and reference information for all methods and variables of any given predefined class.
- For the latest version of Java (version 7) go here:
  http://docs.oracle.com/javase/7/docs/api
- For a convenient way to search go here:
  http://javadocs.org

Exercise 1 – Know your Java… Doc

- Using your browser inspect the documentation for the 3 pre-defined classes involved in reading the input provided by the user
  - Search for "<name of the class>" where <name of the class> is:
    - System
    - InputStream
    - Scanner
  - If search doesn't work try adding "Java 7" or use directory/index
- Identify some methods you can call on a scanner object that would help you read the radius of a circle
- Identify the package the Scanner class is part of

Scanner Object

- The user types the input using the keyboard, the application reads the input using a Scanner object



Standard Operating System Console

Scanning the input

- Once the Scanner object is declared and initialized, it provides reading functionality via its methods
  - nextInt(), nextDouble() , next<type of data>(), nextLine()
  - Normally use nextLine() to read a String (whole line)

- Scanner's "next…" methods will
  - Read the input if already in the Console
  - Wait for the input to be provided by user into the Console. The program will not advance to the next statement until input is provided

Example: Reading the Circle's Radius

```
// Prompt the user for input
System.out.println("Please enter the radius of the circle: ");

// Read the input using a Scanner object
Scanner input = new Scanner(System.in);

double radiusFromUser = input.nextDouble();

// Use the input value to calculate area of the circle
double circleArea = 3.1415927 * radiusFromUser * radiusFromUser;

// Present the output to the user
System.out.println("The area is " + circleArea);
```

Exercise 2: Interactive Shape Fun

- Create a class called ShapeFun with a main method that contains the code on the previous slide
- Declare the class to be part of the sheridan package and save the .java file in a folder called sheridan
- Compile your program
- What happens?

import Statement

- The Scanner class is part of a different package than our Program class
  - Scanner is part of the "java.util" package
  - Our class is part of the "sheridan" package
- To make classes defined in other packages available to a program, we must use the import statement

import <full class name that includes package>;

or

import <full package name>.*;

Exercise 3: Interactive Shape Fun

- Fix the compilation error caused by the use of the Scanner class without an import statement
  - Add the statement: import java.util.Scanner;
- Compile and run
- Test your program to be sure that it works
- When printing the prompt try using print instead of println
  - What does this do differently?
  - Does it make things look a little nicer?

Exercise 4: Even more ShapeFun

- Extend the Program from exercise 3 so that the area calculation [`double circleArea =` ...] happens in its own method named calculateArea
- The method should accept one parameter, the radius, and return the calculated area
- Call the calculateArea method from your main method

Recommended Exercises *(use Scanner)*

- Exercise 2.1: Fahrenheit Convertor
- Exercise 2.3: Meter Convertor
- Exercise 2.6: Digit Sum Calculator
- Exercise 2.11: Payroll Application
- Exercise 2.14: BMI Calculator