Simple Java Programs

Outline

- **Category of Java program**
- Basic steps to develop a Java program
- Basic input/output
- Java development environment

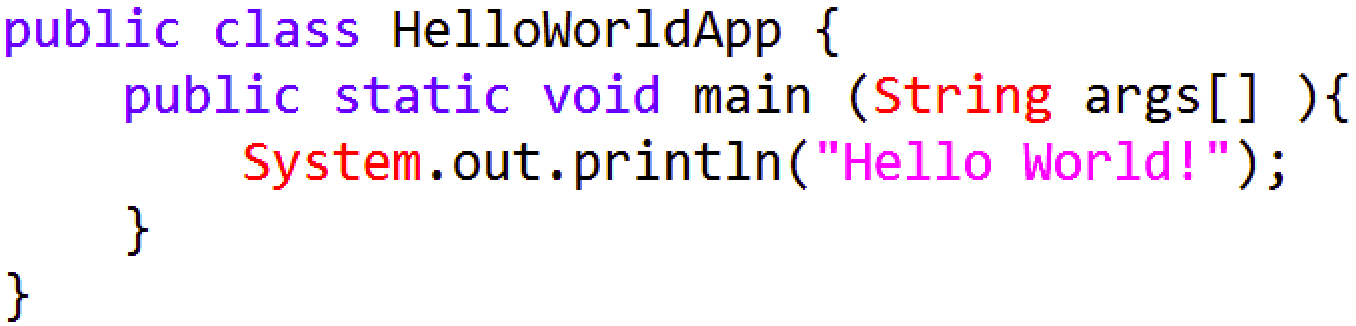
Java program type

- Application & Applet
- Difference between structure and environment
- Application
  - Standalone and executed in JVM
- Applet
  - Embedded in HTML pages
  - Run via appletViewer
  - Or Web browser (call JVM)

Application

- HelloWorldApp.java

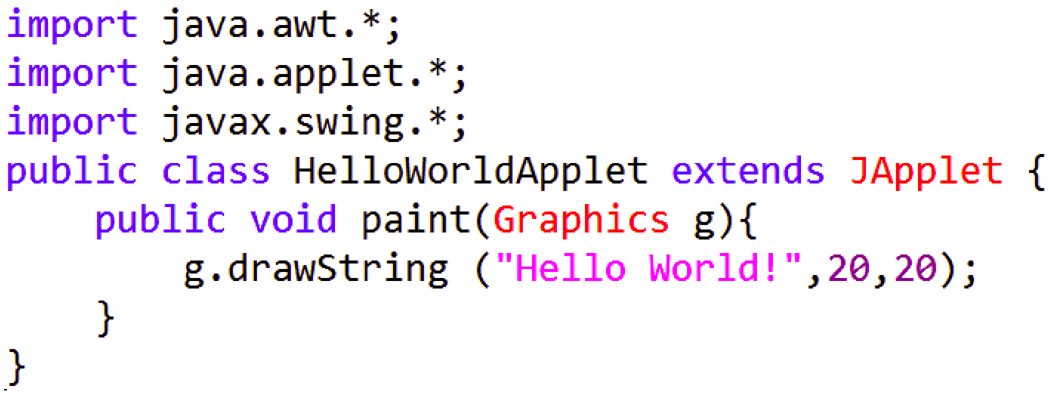
- Key points
  - Class is main body
  - Public class name and file name are identical
  - main() method
  - System.out.print & println & printf



```
public class HelloWorldApp {
    public static void main (String args[] ){
        System.out.println("Hello World!");
    }
}
```

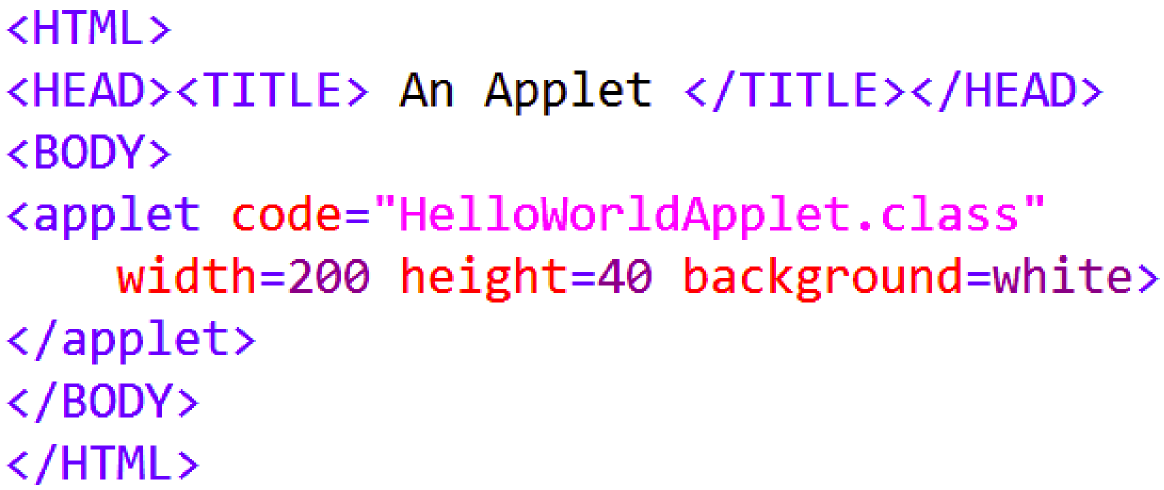
Applet

- **HelloWorldApplet.java**
  - ***import***
  - ***extends* JApplet**
    - **Applet or JAapplet**
  - **paint() method to draw something**
  - **No main() method**



```
import java.awt.*;
import java.applet.*;
import javax.swing.*;
public class HelloWorldApplet extends JApplet {
    public void paint(Graphics g){
        g.drawString ("Hello World!",20,20);
    }
}
```

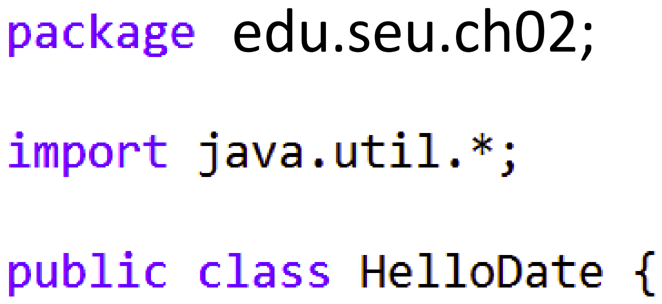
- **HelloWorldApplet.html**



```
<HTML>
<HEAD><TITLE> An Applet </TITLE></HEAD>
<BODY>
<applet code="HelloWorldApplet.class"
    width=200 height=40 background=white>
</applet>
</BODY>
</HTML>
```

Basic Construction

- ❑ **HelloData.java**
  - ➢ ***package*** **(0 or 1)**
  - ➢ ***import*** **(0 or more)**
    - ● **Import Libs**
  - ➢ **Class definition (1 or more)**
    - ● **One file only owns one public class**
  - ➢ **Class = class header + class body**
  - ➢ **Class members = field + method**
    - ● **Field (variables)**
    - ● **Method (function)**
  - ➢ **Method = method header + method body**



```
package  edu.seu.ch02;

import java.util.*;

public class HelloDate {
```

Outline

- Category of Java program
- **Basic steps to develop a Java program**
- Basic input/output
- Java development environment

Edit, compile, and run

- ❑ Edit ASCII source code
  - ➢ Any editor tool, e.g., Notepad, Sublime Text
- ❑ Compile Java source code
  - ➢ Use the tool javac from JDK
- ❑ Run a Java program
  - ➢ Use the tool java

Java basic tool: JDK

- Basic tool for Java is JDK
- Download it from http://java.sun.com
  - Download JavaSE or Netbeans
- Install JDK
  - Bin: tool files
  - Jre: Java runtime environment files
  - Demo: Examples
  - Include: C header files
  - Lib: store libraries
  - Db: database

Application: edit, compile, and run

- Edit the program
  - Public class name and file name are identical
  - Case sensitive
- Compile the code to get a .class file
  - .class file contains JVM instructions
  - Compile the code using javac.exe
  - E.g., javac Hello.java
- Execute the .class file
  - E.g., java Hello
  - java Hello.class (Use class name instead of file name)

Set path and classpath

- ❑Path: Path of the command (javac&java)
- ❑Classpath: Path of the classes
- ❑Use command line to set the path
  - ➢Set path=.;c:\jdk\bin;…
- ❑Use the system environment
  - ➢Computer->Properties->Advanced system settings->Advanced->Environment Variables
- ❑Use option –classpath (-cp) to set class
  - ➢javac –cp libxx.jar className.java
  - ➢java –cp libxx.jar className

Compile files that use *package*

- ❑File and path
- ❑The program include *package* sentence
- ❑Use *import*

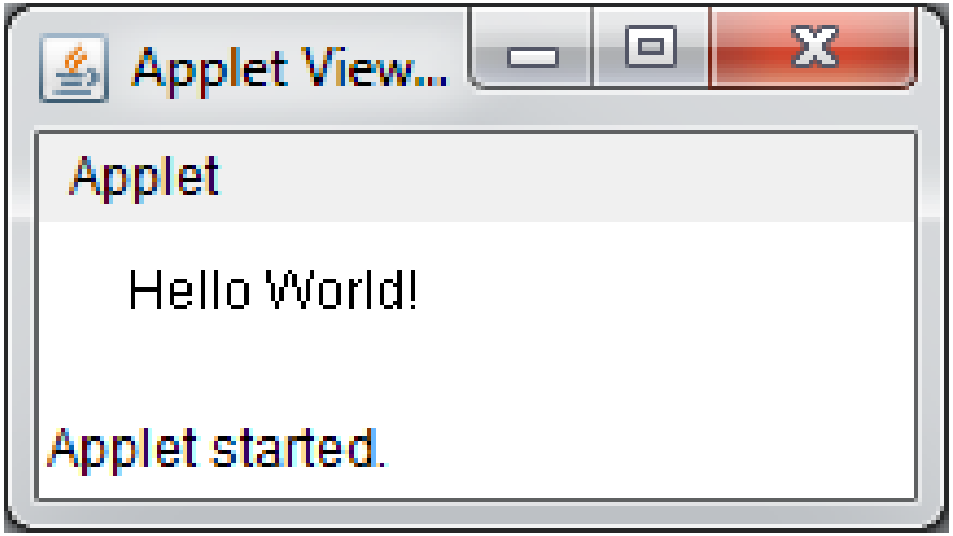
- ❑Compile and run
  - ➢javac –d classes src\edu\ch02\*.java
  - ➢java –cp classes edu.seu.ch02.HelloDate

Applet: edit, compile, and run

- Java Applet should be embedded into HTML
- Java Applet introduce dynamic interactive content for WWW
  - Edit and compile the source code
  - Embed Applet into HTML file
    - Use <applet> tag
    - <applet code=“HelloWorldApplet.class” Width=200 height=40 background=white>

Use appletviewer to run Applet

- ***appletviewer*** **HelloWorldApplet.html**



Hello World!

Applet started.

Use browser to run Applet

- ❑Download and install JRE
- ❑Enable Java in the browser
- ❑Put .class and .html file to the Web server, and then use a browser to access it
- ❑Replacement techniques for Applet
    - ➢Flash, SilverLight, etc.

Other tools

- Main tools
  - javac: compile
  - java: execute a Java program in cmd
  - javaw: execute a Java GUI program
  - appletViewer: execute an applet program

- Other tools
  - jar: packing tool
  - javadoc: documentation generator
  - javap: disassembles one or more class files

Usage of jar

- Compiling: javac A.java
- Packing: jar cvfm A.jar A.man A.class
  - c: create, v: verbose, f: filename, m: manifest
- Executing: java -jar A.jar

- A.man is a manifest file including
  - Manifest-Version: 1.0
  - Class-Path: .
  - Main-Class: A

Usage of JavaDoc

- Javadoc –d direcotryname xxx.java
- /** */ Tag

@author	classes and interfaces only
@version	classes and interfaces only
@see	field, method, and class
@param	methods and constructors only
@return	methods only
@exception	

Java API documentation

- Online documentation
  - http://docs.oracle.com/javase/8/docs/api/index.html
- Download Web documentation
  - jdk-8u60-docs-all.zip
  - http://www.oracle.com/technetwork/java/javase/documentation/jdk8-doc-downloads-2133158.html

Usage of javap

- ❑Use javap to class information
  - ➢javap className

- ❑Use javap to decompile class
  - ➢javap –c className

Outline

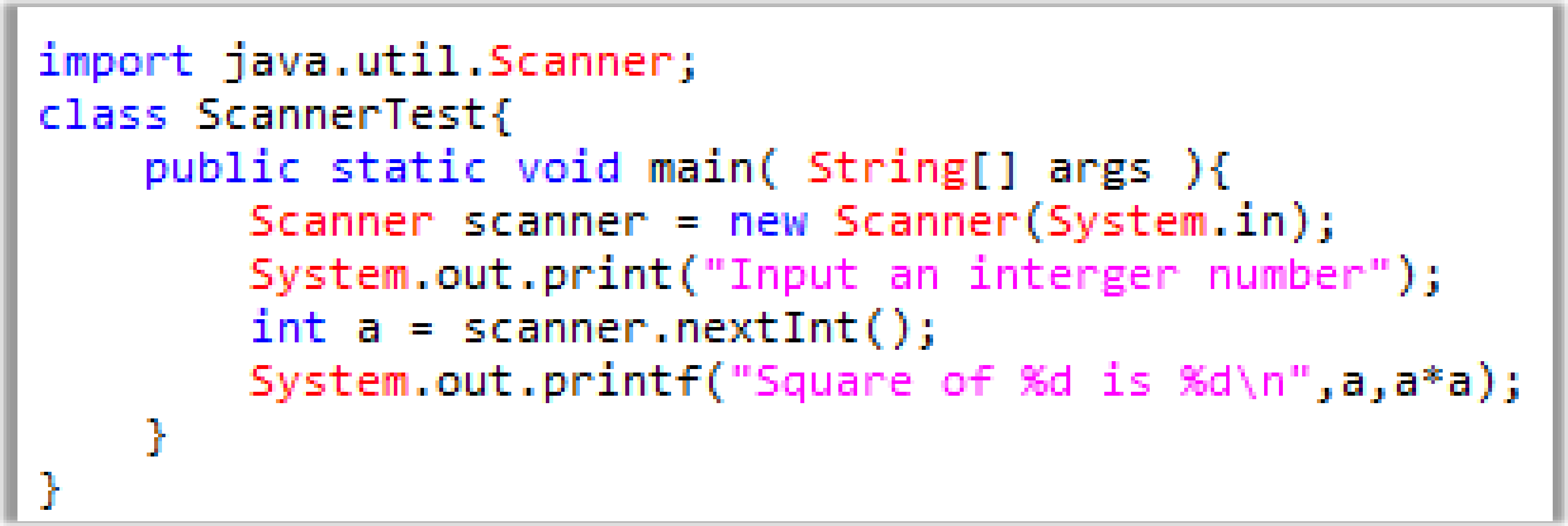
- Category of Java program
- Basic steps to develop a Java program
- **Basic input/output**
- Java development environment

Input and output

- Java Application
  - Text User Interface
  - Graphic User Interface (GUI)
- Java Applet
  - GUI
- Each type of UI has its input and output

TUI: Usage of Scanner

- ❑Use **java.util.Scanner** (ScannerTest.java)
  - ➢Use method ***nextInt()***
  - ➢***nextDouble()***
  - ➢Use ***next()*** to get a word
  - ➢Since JDK 1.5

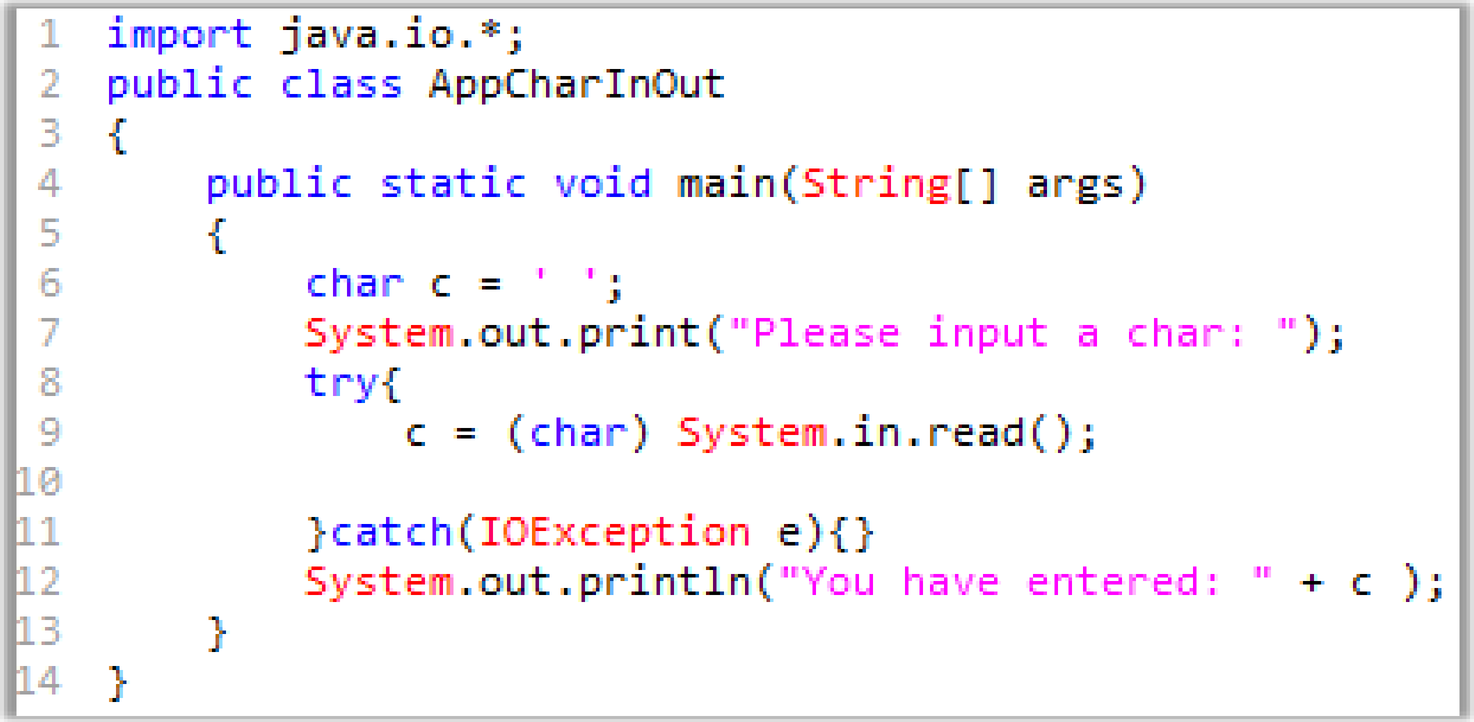


```java
import java.util.Scanner;
class ScannerTest{
    public static void main( String[] args ){
        Scanner scanner = new Scanner(System.in);
        System.out.print("Input an interger number");
        int a = scanner.nextInt();
        System.out.printf("Square of %d is %d\n",a,a*a);
    }
}
```

TUI: Usage of in & out

❑ AppCharInout.java

- ➢ ***java.io*** **package**
- ➢ ***System.in.read()***
- ➢ ***System.out.print()*** **&** ***println*****,** ***printf***
- ➢ **Note: try{}catch**



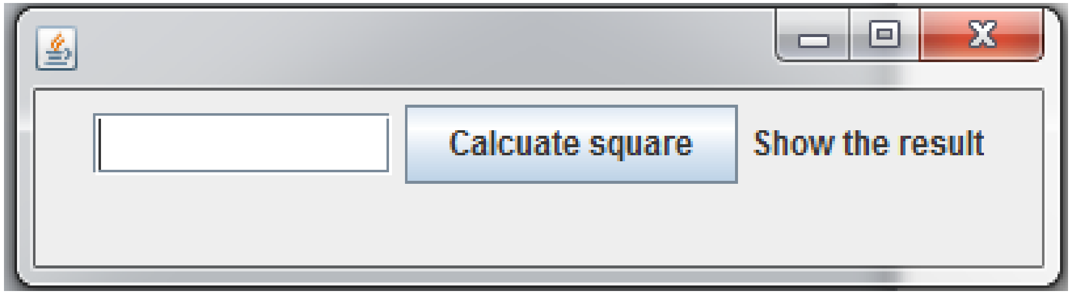
```java
import java.io.*;
public class AppCharInOut
{
        public static void main(String[] args)
        {
                char c = ' ';
                System.out.print("Please input a char: ");
                try{
                        c = (char) System.in.read();

                }catch(IOException e){}
                System.out.println("You have entered: " + c );
        }
}
```

GUI: input & output

- GUI
  - Use TextField object to obtain user’s input data
  - Use Label object or TextField object to output the data
  - Use Button to execute a command

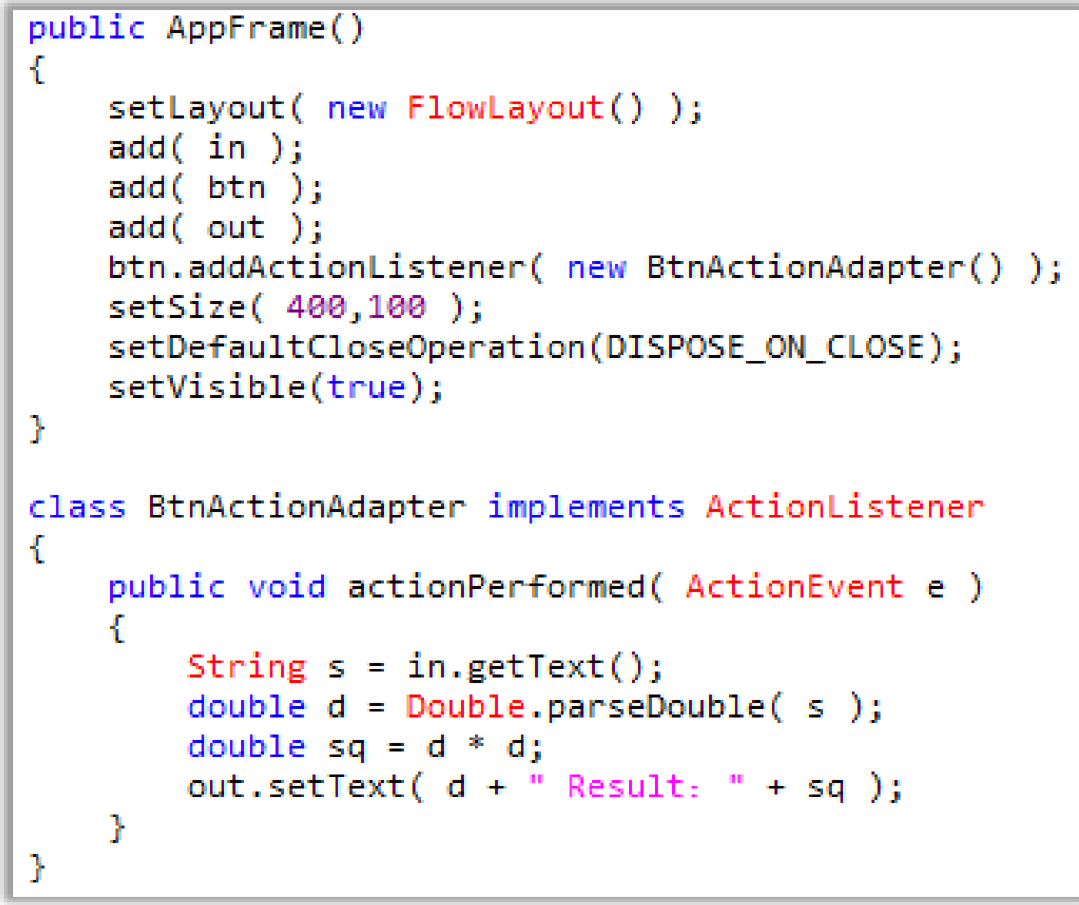
Java Application: GUI input & output

- Java Application creates user interface
- AppGraphInOut.java
- Create a Frame to build its user interface
  - Set Frame size in AppFrame
  - ***setVisible(true)***



Example

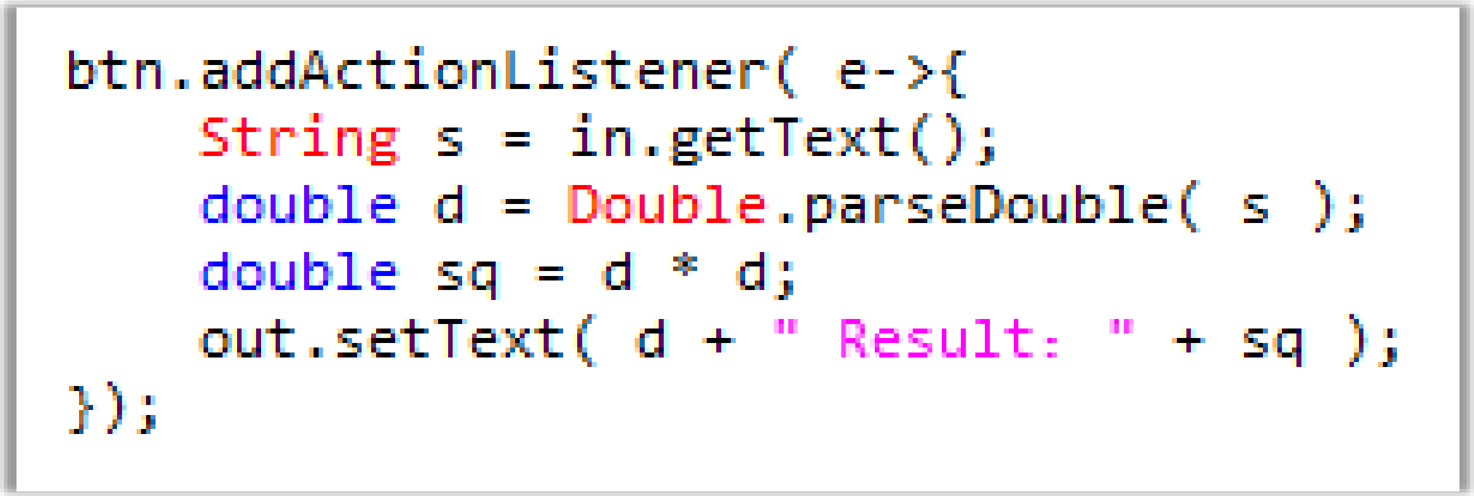
- **AppGraphInOut.java**
- **Add(xxx)**
- ***Btn.addActionListener***
  - **Process event**
- ***actionPerformed()*** **method**
  - **Process specific event**



```
public AppFrame()
{
    setLayout( new FlowLayout() );
    add( in );
    add( btn );
    add( out );
    btn.addActionListener( new BtnActionAdapter() );
    setSize( 400,100 );
    setDefaultCloseOperation(DISPOSE_ON_CLOSE);
    setVisible(true);
}

class BtnActionAdapter implements ActionListener
{
    public void actionPerformed( ActionEvent e )
    {
        String s = in.getText();
        double d = Double.parseDouble( s );
        double sq = d * d;
        out.setText( d + " Result: " + sq );
    }
}
```

Process event

- E->{…} in Java8
  - AppGraphInOut8.java



```
btn.addActionListener( e->{
    String s = in.getText();
    double d = Double.parseDouble( s );
    double sq = d * d;
    out.setText( d + " Result: " + sq );
});
```

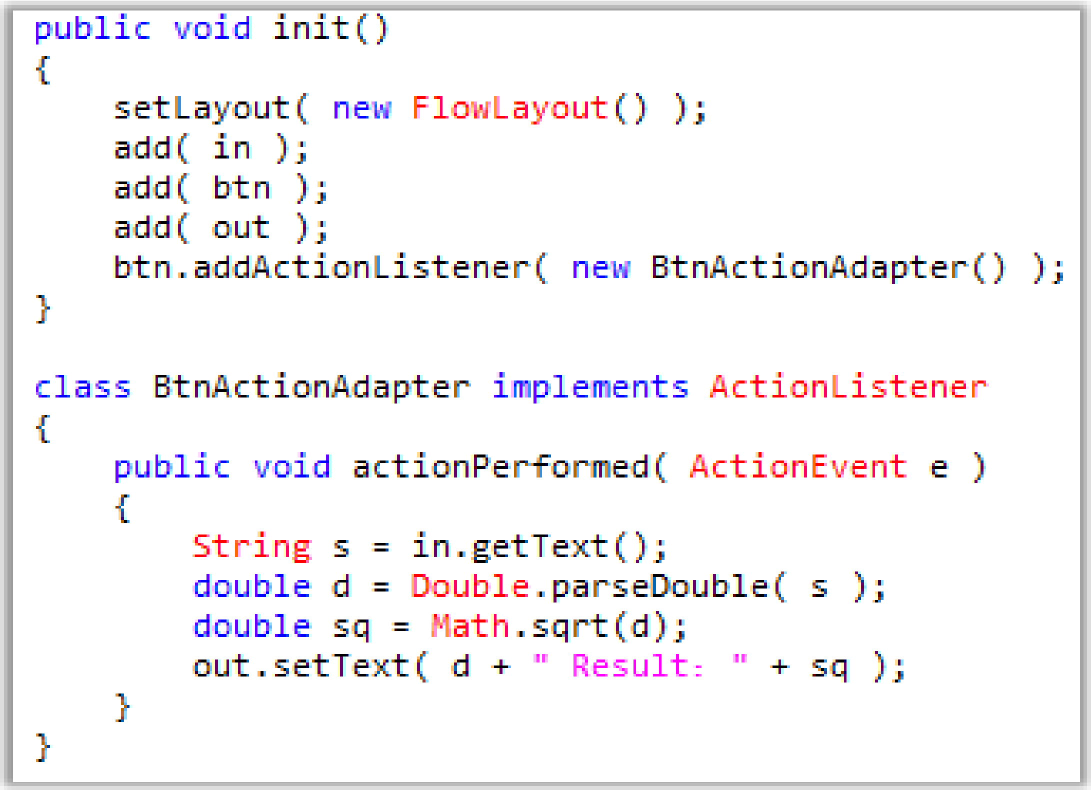
Applet Input & Output

❑ AppletInOut.java (AppletInOut.html)

- ➢ Init()
  - ● add(xxxx)
  - ● Btn.addActionListener

    Process event

- ➢ actionPerformed()
- ➢ Demo
  - ● appletViewer AppletInOut.html



```
public void init()
{
    setLayout( new FlowLayout() );
    add( in );
    add( btn );
    add( out );
    btn.addActionListener( new BtnActionAdapter() );
}

class BtnActionAdapter implements ActionListener
{
    public void actionPerformed( ActionEvent e )
    {
        String s = in.getText();
        double d = Double.parseDouble( s );
        double sq = Math.sqrt(d);
        out.setText( d + " Result: " + sq );
    }
}
```

Application & Applet

- AppAppletInOut.java

- Satisfy three conditions
  - Extends Applet
  - Contains main()
  - Create a UI and add Applet

Outline

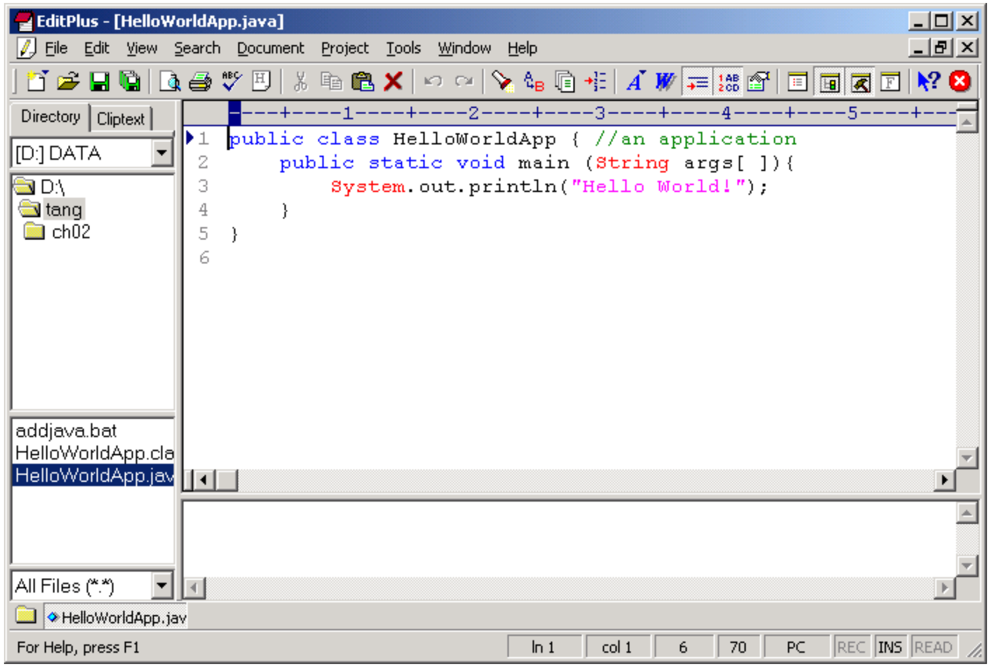
- Category of Java program
- Basic steps to develop a Java program
- Basic input/output
- **Java development environment**

Development environment

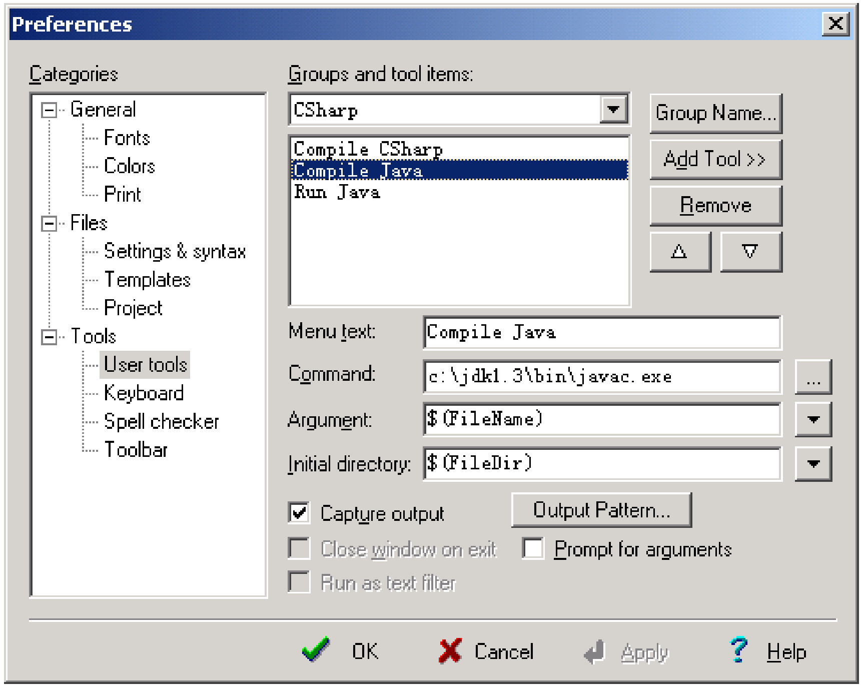
- ❑ Three types of development environment
- ❑ Directly use JDK
  - ➢ javac, java, appletViewer
- ❑ Text editor+JDK command
  - ➢ SublimeText, Notepad++, UltraEdit, EditPlus
  - ➢ Jcreator, Kawa, Freejava
- ❑ IDE (Integrated Development Environment)
  - ➢ IntelliJ (popular), Eclipse, NetBean
  - ➢ Jdeveloper(Oracle), VisualAge for Java(IBM)
  - ➢ BlueJ (for education)

EditPlus

❑ Tools-Preference-Tools-UserTools



```
public class HelloWorldApp { //an application
    public static void main (String args[ ]){
        System.out.println("Hello World!");
    }
}
```



CSharp

Compile CSharp
Compile Java
Run Java

Menu text: Compile Java

Command: c:\jdk1.3\bin\javac.exe

Argument: $(FileName)

Initial directory: $(FileDir)

☑ Capture output

☐ Close window on exit ☐ Prompt for arguments

☐ Run as text filter

Configuration of EditPlus

	Compile	Run
Menu text	Compile Java	Run Java
Command	c:\jdk\bin\javac.exe	c:\jdk\bin\java.exe
Argument	$(FileName)	–classpath . $(FileNameNoExt)
Initial directory	$(FileDir)	$(FileDir)
Capture output	Checked (Optional)	Checked (Optional)

IDE

- ❑ Open Source Eclipse (free)
  - ➢ Download link:
    - ● https://eclipse.org/downloads/
- ❑ Oracle NetBeans (free)
  - ➢ Download link: http://java.sun.com
- ❑ IntelliJ IDEA
  - ➢ Download link: www.jetbrains.com/idea/
- ❑ Android studio (free)
  - ➢ https://developer.android.com/sdk/

Eclipse

- Similar to NetBeans
- Use various plugin
- Programming
  - Create a new project
  - Add a Class
  - Write a method main()
  - Run As …
  - Packing: File-Export-java-Runable jar
  - Java doc, Project-Generate Javadoc

NetBeans

- ❑ Project Manager
- ❑ Source Editor
- ❑ UI Construction too
- ❑ Build Management tool
- ❑ Debug

- ❑ Create a project
- ❑ Write a method main()
- ❑ Run

Code Template in Eclipse

- Code Template
- Eclipse
  - Input main, press Clt+\
  - Input sysout, press Ctl+\, get system.out.println(“”);
  - Window-Perferences-Java-Editor-Templates
  - Window-Preferences-General-Keys-Content Assist

Code Template in NetBean

- Code Template
- NetBean
  - Input psvm, press Tab
    - main()
  - Input sout, press Tab
    - system.out.println(“”);
  - Tool-Options-Editor-Code Templates

Thank you

zhenling@seu.edu.cn