

Introduction to Java Applets

Outline

- 3.1 Introduction
- 3.2 Thinking About Objects
- 3.3 Sample Applet
- 3.4 A Simple Java Applet: Drawing a String
- 3.5 Two More Simple Applets: Drawing Strings and Lines
- 3.6 Another Java Applet: Adding Integers
- 3.7 Java Applet Internet and World Wide Web Resources

© 2000 Prentice Hall, Inc. All rights reserved.



1

3.1 Introduction

- Applet
 - Program that runs in
 - **appletviewer** (test utility for applets)
 - Web browser (IE, Communicator)
 - Executes when HTML (Hypertext Markup Language) document containing applet is opened
 - *Applications* run in command windows
- Notes
 - Mimic several features of Chapter 2 to reinforce them
 - Focus on fundamental programming concepts first
 - Some specific details will not be explained
 - Explanations will come later

© 2000 Prentice Hall, Inc. All rights reserved.



2

3

3.2 Thinking About Objects

- Java is an object-oriented language
 - However, Java has constructs from structured programming
 - In first seven chapters, focus on "conventional" parts of Java
 - Introduce object-oriented concepts as we encounter them
- Object orientation
 - Natural way to think about world and writing computer programs
 - Object-oriented programming models the real world
 - Attributes - properties of objects
 - Size, shape, color, weight, etc.
 - Behaviors - actions that objects can perform
 - A ball rolls, bounces, inflates and deflates

© 2000 Prentice Hall, Inc. All rights reserved.



4

3.2 Thinking About Objects

- Object orientation (continued)
 - Inheritance
 - New classes of objects absorb characteristics of existing classes
 - Information hiding
 - Objects usually do not know how other objects are implemented
 - We can drive cars without knowing how every part works internally
- Abstraction
 - View the big picture
 - See a photograph rather than a group of colored dots
 - Think in terms of houses, not bricks

© 2000 Prentice Hall, Inc. All rights reserved.



5

3.2 Thinking About Objects

- Class - unit of Java programming
 - Java focuses on nouns rather than verbs
 - C focuses on verbs and is action oriented
 - "blueprint" of the objects
 - Objects are created from the class
 - Built-in types (like **ints**) are variables
 - User-defined types are objects
 - Contain methods
 - Implement behaviors
 - Contain data
 - Implement attributes
 - Classes are reusable
 - Create standardized, interchangeable parts

© 2000 Prentice Hall, Inc. All rights reserved.



6

3.3 Sample Applets from the Java SE Java Development Kit

- Sample Applets
 - Provided in Java Development Kit 6(JDK 6)
 - Source code included (. **java** files)
 - Can study and mimic source code to learn new features
 - Remember, all programmers begin by mimicking existing programs
 - Located in **demo** directory of JDK install
 - Can download demos and JDK from
 - <http://download.oracle.com/javase/tutorial/deployment/applet/getStarted.html>
 - <http://www.oracle.com/technetwork/java/javase/downloads/index.html>

© 2000 Prentice Hall, Inc. All rights reserved.



7

3.3.1 Running applets

- In command prompt, change to subdirectory of applet
`cd appletDirectoryName`
- There will be an HTML file used to execute applet
- Type **appletviewer example1.html**
 - **appletviewer** loads the html file specified as its command-line argument
 - From the HTML file, determines which applet to load (more section 3.4)
- Applet will run, **Reload** and **Quit** commands under **Applet** menu

© 2000 Prentice Hall, Inc. All rights reserved.



8

3.4 A Simple Java Applet: Drawing a String

- Now, create applets of our own
 - Take a while before we can write applets like in the demos
 - Will cover many of same techniques
- Upcoming program
 - Create an applet to display
"Welcome to Java Programming!"
 - Show applet and HTML file, then discuss them line by line

© 2000 Prentice Hall, Inc. All rights reserved.



9

```


1 // Fig. 3.6: WelcomeApplet.java
2 // A first applet in Java
3 import javax.swing.JApplet; // import class JApplet
4 import java.awt.Graphics;    // import class Graphics
5
6 public class WelcomeApplet extends JApplet {
7     public void paint( Graphics g )
8     {
9         g.drawString( "Welcome to Java Programming!", 25, 25 );
10    }
11 }

```

```

1 <html>
2 <applet code="WelcomeApplet.class" width=300 height=30>
3 </applet>
4 </html>

```

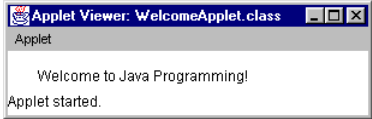


Outline

Java applet

HTML file

Program Output



© 2000 Prentice Hall, Inc. All rights reserved.

10

3.4 A Simple Java Applet: Drawing a String

```

1 // Fig. 3.6: WelcomeApplet.java
2 // A first applet in Java

```



- Lines that begin with `//` are comments
 - Gives name of source code and brief description of applet

```

3 import javax.swing.JApplet; // import class JApplet
4 import java.awt.Graphics;   // import class Graphics

```

- As stated in Chapter 2, Java has predefined classes grouped into packages
 - **import** statements tell compiler where to locate classes used
 - When you create applets, **import** the **JApplet** class (package **javax.swing**)
 - **import** the **Graphics** class (package **java.awt**) to draw graphics
 - Can draw lines, rectangles ovals, strings of characters
 - **import** specifies directory structure

© 2000 Prentice Hall, Inc. All rights reserved.  

11

3.4 A Simple Java Applet: Drawing a String

- Applets have at least once class definition (like applications)
 - Rarely create classes from scratch
 - Use pieces of existing class definitions
 - Java uses inheritance to create new classes from old ones

```
6 public class WelcomeApplet extends JApplet {
```

- Begins **class** definition for class **WelcomeApplet**
 - Keyword **class** then class name
- **extends** followed by class name
 - Indicates the class to inherit from (**JApplet**)
 - **JApplet** : superclass (base class)
 - **WelcomeApplet** : subclass (derived class)
 - **WelcomeApplet** now has methods and data of **JApplet**

© 2000 Prentice Hall, Inc. All rights reserved.



12

3.4 A Simple Java Applet: Drawing a String

```
6 public class WelcomeApplet extends JApplet {
```

- Someone else has defined "what it means to be an applet"
 - Class **JApplet** is defined for us
 - Applets require over 200 methods!
 - **extends JApplet** allows us to inherit methods
 - Do not have to define them all
 - Do not need to know every detail of class **JApplet**
- Class **WelcomeApplet** is a blueprint
 - Creates (instantiates) an object for use by program
 - **appletviewer** or browser creates an object of class **WelcomeApplet**
 - Keyword **public** required
 - File can only have one **public** class
 - **public** class name must be file name

© 2000 Prentice Hall, Inc. All rights reserved.



13

3.4 A Simple Java Applet: Drawing a String

```
7 public void paint( Graphics g )
```

- Our class inherits method **paint** from **JApplet**
 - By default, **paint** has an empty body
 - We override (redefine) **paint** in our class
- Methods **paint**, **init**, and **start**
 - Guaranteed to be called automatically for us
 - Our applet gets a "free" version of these by inheriting from **JApplet**
 - Free versions have an empty body (do nothing)
 - Every applet does not need all three - override only the ones you need

© 2000 Prentice Hall, Inc. All rights reserved.



14

3.4 A Simple Java Applet: Drawing a String

```
1 <html>
2 <applet code="WelcomeApplet.class" width=300 height=30>
3 </applet>
4 </html>
```

- Simple HTML file (**WelcomeApplet.html**)
 - Usually in same directory as **.class** file
 - Remember, **.class** file created after compilation
- HTML codes (tags)
 - Usually come in pairs
 - Begin with **<** and end with **>**
- Lines 1 and 4 - begin and end the HTML tags
- Line 2 - begins **<applet>** tag
 - Specifies code to use for applet
 - Specifies **width** and **height** of display area in pixels
- Line 3 - ends **<applet>** tag

© 2000 Prentice Hall, Inc. All rights reserved.



15

3.4 A Simple Java Applet: Drawing a String

```

1 <html>
2 <applet code="WelcomeApplet.class" width=300 height=30>
3 </applet>
4 </html>

```

- **appletviewer** only understands **<applet>** tags
 - Ignores everything else
 - Minimal browser
- Executing the applet
 - **appletviewer WelcomeApplet.html**
 - Perform in directory containing **.class** file

© 2000 Prentice Hall, Inc. All rights reserved.



16

```

1 // Fig. 3.6: WelcomeApplet.java
2 // A first applet in Java
3 import javax.swing.JApplet; // import class JApplet
4 import java.awt.Graphics;   // import class Graphics
5
6 public class WelcomeApplet extends JApplet {
7     public void paint( Graphics g )
8     {
9         g.drawString( "Welcome to Java Programming!", 25, 25 );
10    }
11 }

```

import allows us to use predefined classes (allowing us to use applets and graphics, in this case).

2. Class
WelcomeApplet

extends allows us to inherit the capabilities of class **JApplet**.

3. paint

Method **paint** is guaranteed to be called in all applets. Its first line must be defined as above.

```

1 <html>
2 <applet code="WelcomeApplet.class" width=300 height=30>
3 </applet>
4 </html>

```

HTML file



Program Output

© 2000 Prentice Hall, Inc. All rights reserved.

3.5 Two More Simple Applets: Drawing Strings and Lines

17

- More applets
 - First example
 - Display two lines of text
 - Use **drawString** to simulate a new line
 - We will actually use two **drawString** statements
 - Second example
 - Method **drawLine(x1, y1, x2, y2)**
 - Draws a line from (x1,y1) to (x2,y2)
 - Remember that (0, 0) is upper left
 - Use **drawLine** to draw a line beneath and above a string

© 2000 Prentice Hall, Inc. All rights reserved.



```

1 // Fig. 3.8: WelcomeApplet2.java
2 // Displaying multiple strings
3 import javax.swing.JApplet; // import class JApplet
4 import java.awt.Graphics;   // import class Graphics
5
6 public class WelcomeApplet2 extends JApplet {
7     public void paint( Graphics g )
8     {
9         g.drawString( "Welcome to", 25, 25 );
10        g.drawString( "Java Programming!", 25, 40 );
11    }
12 }

```

```

1 <html>
2 <applet code="WelcomeApplet2.class" width=300 height=45>
3 </applet>
4 </html>

```



The two **drawString** statements simulate a newline. In fact, the concept of lines of text does not exist when drawing strings.



Outline

18

1. import

2. Class
WelcomeApplet2
(extends JApplet)

3. paint

HTML file

Program Output

© 2000 Prentice Hall, Inc. All rights reserved.

19

```

1 // Displaying text and lines
2 import javax.swing.JApplet; // import class JApplet
3 import java.awt.Graphics; // import class Graphics
4
5 public class WelcomeLines extends JApplet {
6     public void paint( Graphics g )
7     {
8         g.drawLine( 15, 10, 210, 10 );
9         g.drawLine( 15, 30, 210, 30 );
10        g.drawString( "Welcome to Java Programming!", 25, 25 );
11    }
12 }

```

```

1 <html>
2 <applet code="WelcomeLines.class" width=300 height=40>
3 </applet>
4 </html>

```

Outline

1. import

2. Class
WelcomeLines
(extends JApplet)

3. paint

3.2 drawLine

3.3 drawString

HTML file

Program Output

Draw horizontal lines with **drawLine** (endpoints have same y coordinate).

© 2000 Prentice Hall, Inc. All rights reserved.