Week 11
Java Applets

Road Map

- Introduction to Java Applets
- Review applets that ship with JDK
- Make our own simple applets
    - Introduction the applet environment
- html needed for applets

Applet Viewer and Browser

- An applet is a program that runs inside an applet viewer on client-side.
- The applet viewer takes care of providing the environment and calling several of the applet's methods.
- Modern browsers come with applet capabilities.
- The JDK also comes with an applet viewer.
  - The JDK applet viewer is really just a minimum browser. It only understands the applet tag.

Simple Java Applet: Drawing a String

- Now, create applets of our own
- Upcoming program
  - Create an applet to display
    `"Welcome to Java!!"`

Applet Example

```
import java.awt.*;
import java.applet.*;


public class WelcomeApplet extends JApplet {
public void init() {
  }
public void paint(Graphics g) {
      g.drawString("Welcome to Java Programming!",  25, 25 );
  }
}
```

`extends` allows us to inherit the capabilities of class `Applet`.

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

Simple Java Applet: Drawing a String

- Methods `init`, `start` and `paint`.
  - Guaranteed to be called automatically
  - Our applet gets "free" version of these by inheriting from `Applet`
    - Free versions have empty body (do nothing)
    - Every applet does not need all three methods
      - Override the ones you need

- Our class inherits method `paint` from `Applet`
  - By default, `paint` has empty body
  - Override (redefine) `paint` in our class

- Applet container “draws itself” by calling method `paint`

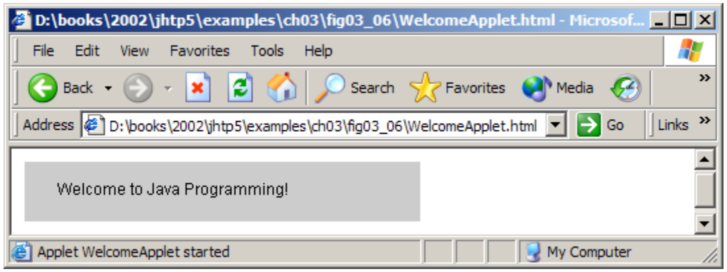
- Compile
  - `javac WelcomeApplet.java`
  - If no errors, bytecodes stored in `WelcomeApplet.class`

Simple Java Applet: Drawing a String

- Create an HTML file
  - Loads the applet into `appletviewer` or a browser
  - Ends in `.htm` or `.html`
- To execute an applet
  - Create an HTML file indicating which applet the browser (or `appletviewer`) should load and execute
- Executing the applet
  - `appletviewer WelcomeApplet.html`

    Perform in directory containing `.class` file

```
<html>
<applet code = "WelcomeApplet.class" width = "300" height = "45">
</applet>
</html>
```



D:\books\2002\jhtp5\examples\ch03\fig03_06\WelcomeApplet.html - Microsof...
File
Edit
View
Favorites
Tools
Help
Back
Search
Favorites
Media
Address
D:\books\2002\jhtp5\examples\ch03\fig03_06\WelcomeApplet.html
Go
Links
My Computer

Welcome to Java Programming!

Applet WelcomeApplet started

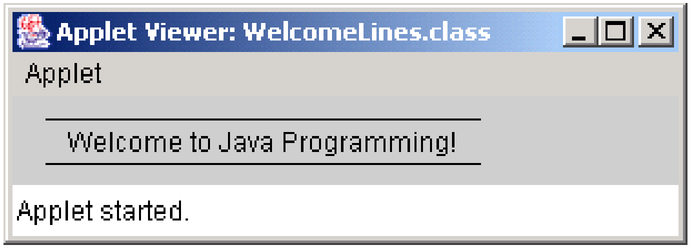
Simple Java Applet: Drawing a String

```
import java.awt.*;
import java.applet.*;

public class WelcomeApplet3 extends Applet {

		public void init() {
		}

		public void paint(Graphics g) {
				g.drawString("Welcome to Java Programming!", 25, 25 );
				g.drawLine (15, 10, 210, 10);
				g.drawLine (15, 30, 210, 30);
		}
}
```



```
Welcome to Java Programming!
```

```
Applet started.
```

JApplet Methods

Method	When the method is called and its purpose
public void `init()`	This method is called once by the applet container when an applet is loaded for execution. It performs initialization of an applet. Typical actions performed here are initializing fields, creating GUI components, loading sounds to play, loading images to display (see Chapter 19, Multimedia) and creating threads (see Chapter 16, Multithreading).
public void `start()`	This method is called after the `init` method completes execution. In addition, if the browser user visits another Web site and later returns to the HTML page on which the applet resides, method `start` is called again. The method performs any tasks that must be completed when the applet is loaded for the first time and that must be performed every time the HTML page on which the applet resides is revisited. Typical actions performed here include starting an animation (see Chapter 19) and starting other threads of execution (see Chapter 16).
public void `paint( Graphics g )`	This drawing method is called after the `init` method completes execution and the `start` method has started. It is also called every time the applet needs to be repainted. For example, if the user covers the applet with another open window on the screen and later uncovers the applet, the `paint` method is called. Typical actions performed here involve drawing with the `Graphics` object `g` that is passed to the `paint` method by the applet container.
public void `stop()`	This method is called when the applet should stop executing—normally, when the user of the browser leaves the HTML page on which the applet resides. The method performs any tasks that are required to suspend the applet's execution. Typical actions performed here are to stop execution of animations and threads.
public void `destroy()`	This method is called when the applet is being removed from memory—normally, when the user of the browser exits the browsing session (i.e., closes all browser windows). The method performs any tasks that are required to destroy resources allocated to the applet.

Fig. **`JApplet`** methods that the applet container calls during an applet's execution.

Servlets

- A Servlet is executed on Server-side.
- No interface is needed.
- Has these life-cycle methods: init( ), service( ), and destroy( )
- It processes HTTP requests (Get, Post, etc.) from the client and issues responses.
- All JEE app servers (Tomcat, WebLogic, Spring, etc.) include a servlet container
- JSPs also run on server

Simple Java Servlet

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.util.*;
public class Hello extends HttpServlet {
private static final String CONTENT_TYPE = "text/html";
/**Initialise global variables*/
public void init() throws ServletException {
}
/**Process the HTTP request Get*/
public void doGet (HttpServletRequest request,HttpServletResponse response) throws ServletException, IOException {
		response.setContentType(CONTENT_TYPE);
		PrintWriter out = response.getWriter();
		out.println("<p>The servlet <b>Hello</b> received a <b>"+ +"<font color=\"red\">GET</font></b>. </p>");
		out.println("<h1>Hello. Hello .class. says hello</h1>");
}
**Process the HTTP request Post*/
public void doPost (HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
		response.setContentType(CONTENT_TYPE);
		PrintWriter out = response.getWriter();
		out.println("<p>The servlet <b>Hello</b> received a<b>"+"<font color=\"red\">POST</font></b>.</p>");
		out.println("<h1>Hello. Hello.class. says hello</h1>");
}
public void destroy() { }
}
```