Chapter 17

Applets

Introduction

- When Java 1.0 was first released in 1995 programmer were really excited to try this new language because it has a new technology called applets.
- Applets allow a programmer to write a piece of software and embed it in a web page.
- This works on the principle that a compiled Java program can be moved to any JVM and be guaranteed to run.

How applets work

- Applet code is stored on the web server with the web page.
- When a user click on the page, the applet is downloaded to the users local JVM (if they have one)
- The applet is then started and runs locally within the web page.
- When the user closes the browser, the applet is cleared from the JVM

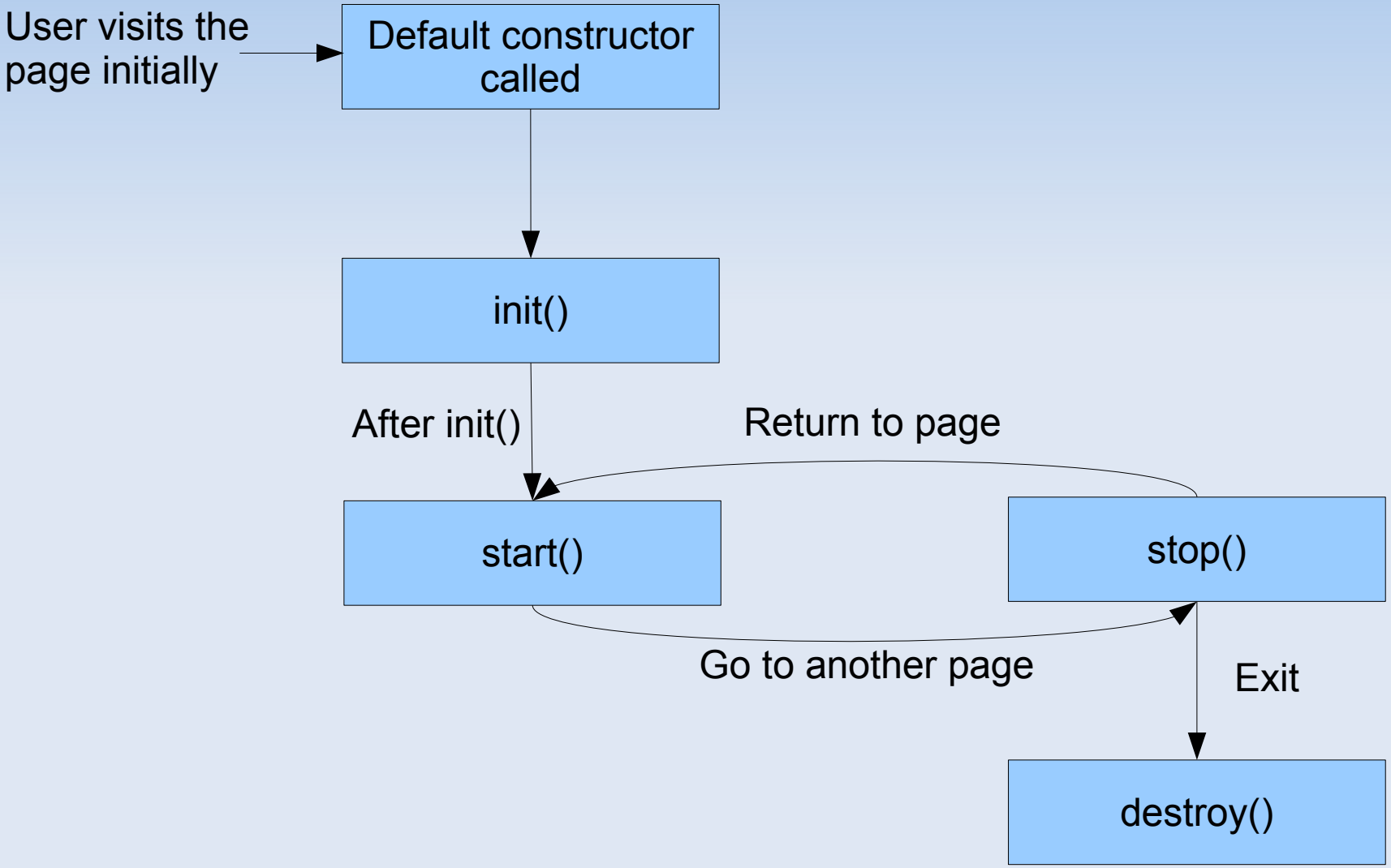
The Applet Class

- When we make a GUI application, we extend JFrame.
- When we make an Applet we extend JApplet.
- The JApplet class contains 4 methods that can and should be overridden to make the applet function.
- Applets have no main method! This is because they are loaded by a special instance of the JVM.

The Applet Class

```
import javax.swing.*;

public class FirstApplet extends JApplet{

    //The no-arg constructor is called first
    public FirstApplet(){
    }

    //The init method is called after the constructor
    public void init(){
    }

    //The start method is called after the init() method
    //and each time the page becomes active
    public void start(){
    }

    //The stop method is called each time the page
    //becomes inactive
    public void stop(){
    }

    //Called when the browser exits
    public void destroy(){
    }
}
```

The Applet Class



The Applet Class

- init() - The init method is invoked after the applet is created. This is where you do your initialization work like setting up the user interface, load resources like images and audio, and get string parameters from the applet tag.
- start() - The start method is called after the init and every time the page becomes active. Here you should put code that starts timer and threads needed for animations and the like.

The Applet Class

- Stop() - This method is the opposite of start. It is called every time a user navigates away from the page. Here you should put code that stops any timers or threads for animations.
- Destroy() - This method is called when the browser is closing and the applet is being removed from the JVM. Here you should put code that explicitly releases or closes any resources that the applet has created.

The Applet Class

- NOTE: You do NOT need to override all of these methods if you don't need them!
- You may never need to override the destroy method unless you are working with trusted applets.
- The JApplet class will act as our frame. We will add elements onto the applet using add(component) just like we do with JFrame

The Applet Class

```
import javax.swing.*;

public class FirstApplet extends JApplet{
    private JLabel jlblHello = new JLabel("Hello, World!",
                                          JLabel.CENTER);

    public FirstApplet(){
        add(jlblHello);
    }
}
```

We can add our label to the Content Pane of the JApplet in the constructor, or in the init() method. Its up to you!

The Applet Class

- Now that we have our class, and we have compiled it, how to we make it so other people can use it through a web page?
- You can embed it in a web page using the applet HTML tag.
- You simply create a new blank file called <appletName>.html and use the applet tag to point to the applet you want people to use.

The Applet Class

```
<html>
   <head>
      <title>Hello World Applet</title>
   </head>

   <body>
      <applet code="FirstApplet.class"
              width="350"
              height="200" >
      </applet>
   </body>
</html>
```

The Applet HTML Tag

- The applet tag can take more parameters then the ones show in the last slide

```
<applet
    codebase="applet_url"
    code="class_file_name.class"
    width="viewing_width"
    height="viewing_height"
    archive="jar_file_name.jar"
    vspace="vertical_padding"
    hspace="horizontal_padding"
    align="applet_alignment"
    alt="alternative_text" >

    <param name="name" value="parameter value">
    <param name="name2" value="parameter value2">

</applet>
```

The Applet HTML Tag

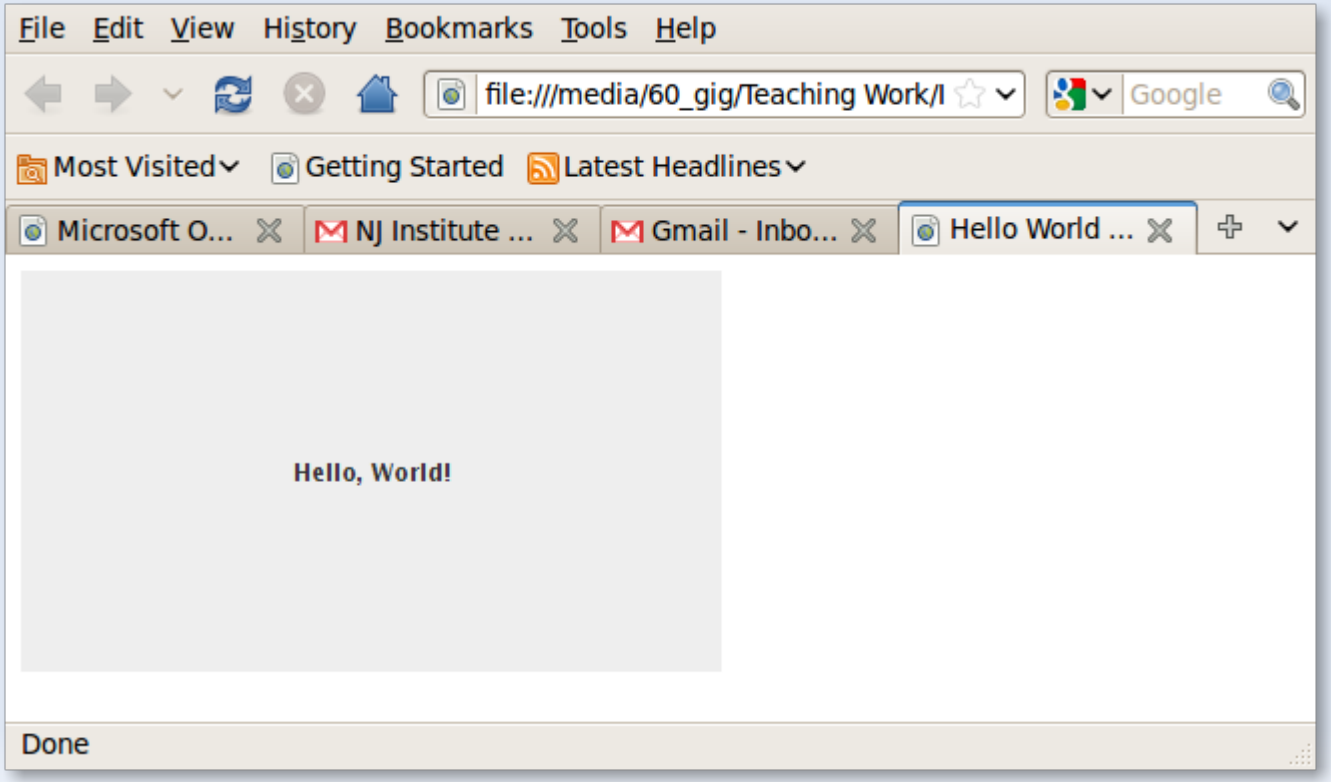
- codebase – This specifies the base location of where your applet code is located. If you don't specify this attribute then the browser will load the applet from the same directory as the HTML page. This tag takes a URL and can be used to load an applet from anywhere on the Internet.
- archive – This attribute specifies that the applet code and all required classes are contained in a single jar file. This can be useful in reducing load time.

The Applet HTML tag

- vspace and hspace – These attributes specify the size, in pixels of the blank space to put around the applet.
- align – This attribute tells the browser how to align the applet on the page. The possible values are left, right, top, texttop, middle, absmiddle, baseline, bottom, or absbottom.
- alt – This attribute specifies the text to be displayed in case the user doesn't have a Java JVM installed. Or if the browser cannot run Java.

Viewing the applet

- Once you have your applet written, and your HTML page complete. You can simply open the HTML file from your favorite browser and see the result.



Enabling applets to run as applications

- Despite some differences, The JFrame class and the JApplet class have a lot in common.
- They both are subclasses of the Container class, and all of there user interface components, layout managers, and event handling are identical.
- Applications however are invoked by the main method, and Applets are invoked using the init() and start() methods.

Enabling applet to run as applications

- There are some security differences as well
  1) Applets are NOT to allowed to read from or write to the file system of the local computer of which they are running.
  2) Applets are not allowed to invoke local programs
  3) Applets are not allowed to connect to any other system other then the one that hosts in original code. This prevents the applet from connecting the users computer to another system on the Internet.

Enabling applets to run as applications

- NOTE: You can allow an applet to have those capabilities if you create a trusted applet and the user allows the applet to bypass those restrictions.
- See http://www.developer.com/java/ent/article.php/3303561

Enabling applets to run as applications

- An applet can be allowed to run as a local application without any loss of functionality.
- You can implement a main method inside of an applet.
- If you are running the applet as an applet it will ignore the main method
- If you are running the applet as a program, the main method will execute.

Enabling applets to run as applications

```
import javax.swing.*;

public class FirstApplet extends JApplet{
    private JLabel jlblHello = new JLabel("Hello, World!",
                                          JLabel.CENTER);

    public void init(){
        add(jlblHello);
    }

    public static void main(String[] args){
        JFrame frame = new JFrame();
        FirstApplet fa = new FirstApplet();

        fa.init();
        fa.start();

        frame.add(fa);
        frame.setTitle("Hello World");
        frame.setSize(400,300);
        frame.setLocationRelativeTo(null);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setVisible(true);
    }
}
```

Enabling applets to run as applications

- Now the program will run both as an applet and as a local application

Passing strings to applets

- Remember those param HTML tags located between the <applet> and </applet> tags?
- Those allow you to pass data from the web page into the applet.
- In later chapters, we will learn how to dynamically generate those param tags so that the user has the ability to choose what gets passed into the applet if necessary.

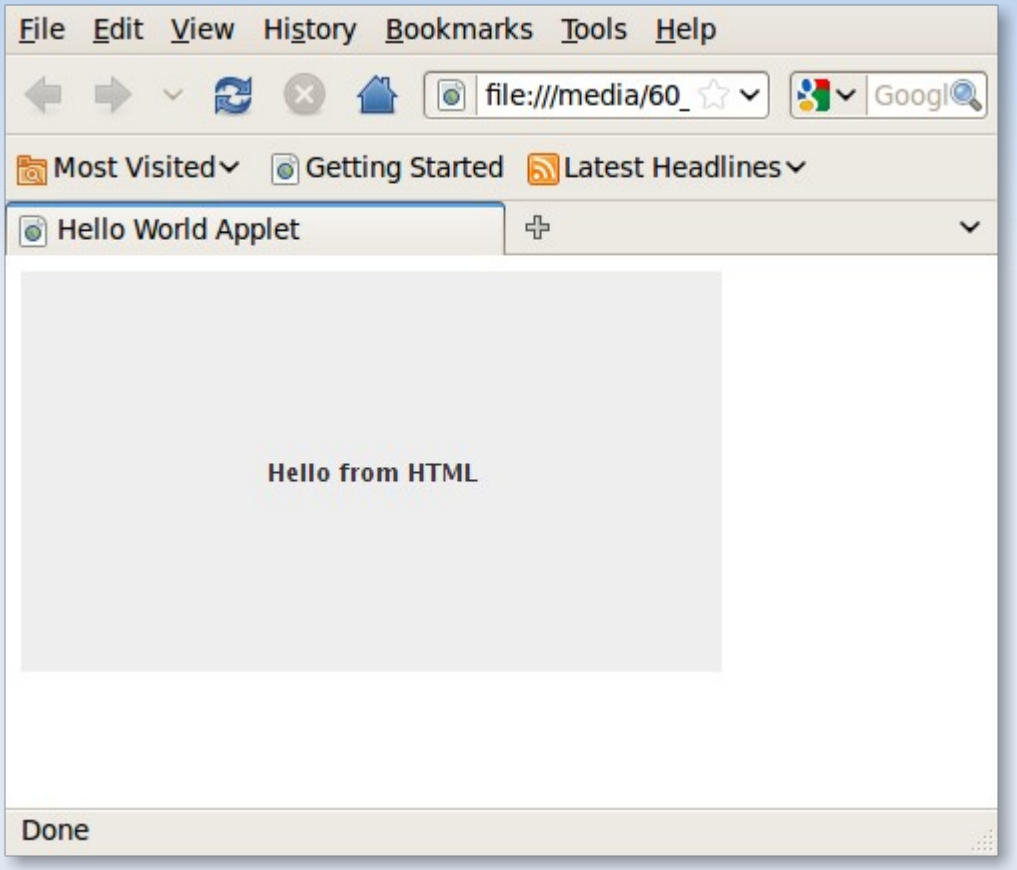
Passing strings to applets

```
import javax.swing.*;

public class DisplayTextFromParam extends JApplet{
    private JLabel message;

    public DisplayTextFromParam(){
    }

    public void init(){
        message = new JLabel(getParameter("message"), JLabel.CENTER);
        add(message);
    }
}
```

Passing strings to applets

```
<html>
    <head>
        <title>Hello World Applet</title>
    </head>

    <body>
        <applet code="DisplayTextFromParam.class"
                width="350"
                height="200" >
                <param name="message" value="Hello from HTML" />
        </applet>
    </body>
<html>
```

Passing strings to applets



Locating resources with a URL

- In normal applications, any image you want to use in your program can simply be created with a string that represents the location of the image on your hard drive

```
ImageIcon icon = new ImageIcon(“images/cat.png”);
```

- If you want to use an image in an applet, it is not so simple. You need to use the URL class to create a Resource Locater for the resource you need.

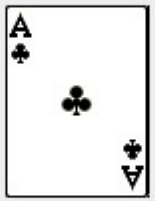
Locating resources with a URL

- Getting the URL for a resource is really easy
- You can simply use the static getClass().getResource() method call

```
URL iconURL = this.getClass().getResource(“images/icon.jpg”);
ImageIcon icon = new ImageIcon(iconURL);
```

Locating resources with a URL

```
import javax.swing.*;
import java.net.*;

public class FirstApplet extends JApplet{

	public FirstApplet(){
		URL imageURL = getClass().getResource("images/ace.png");
		ImageIcon icon = new ImageIcon(imageURL);

		add(new JLabel(icon));
	}
}
```



Playing audio in any program

- Originally applets were the only programs able to play audio. This quickly changed as people needed the ability to play audio in any program that they wrote.
- The audio objects and controls are still contained inside of the applet class.
- There is a simple interface to control audio clips once they are created.

Playing audio in any program

```
import javax.swing.*;
import java.net.URL;
import java.applet.*;

public class PlayAudio extends JApplet {
  private AudioClip audioClip;

  public PlayAudio() {
    URL urlForImage = getClass().getResource("images/mario.jpg");
    add(new JLabel(new ImageIcon(urlForImage)));

    URL urlForAudio = getClass().getResource("audio/mario.mid");
    audioClip = Applet.newAudioClip(urlForAudio);
    audioClip.loop();
  }

  public void start() {
    if (audioClip != null) audioClip.loop();
  }

  public void stop() {
    if (audioClip != null) audioClip.stop();
  }
```

Playing audio in any program

```
  public static void main(String[] args) {

    JFrame frame = new JFrame("Super Mario Brothers");

    PlayAudio applet = new PlayAudio();
    applet.init();

    frame.add(applet, java.awt.BorderLayout.CENTER);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.pack();
    frame.setVisible(true);
  }
}
```

Lab Assignment

- Page # 587 Problem 17.4*

Homework

- Page # 590 Problem 17.12**

Advanced HW

- Page # 591 Problem 17.16***

Acknowledgments

Introduction to Java Programming by Y. Daniel Liang