

Programming in Java

Lecture 19: The Applet Class

Mahesh Kumar

Assistant Professor (Adhoc)

Department of Computer Science
Acharya Narendra Dev College
University of Delhi

Course webpage

[<http://www.mkbhandari.com/mkwiki>]

Outline

- 1 Applet Basics
- 2 The Applet Class
- 3 Applet Architecture
- 4 Simple Applet Display Methods
- 5 A Simple Banner Applet

Applet Basics

- Applet is a special type of program that is embedded in the webpage to generate the dynamic content. It runs inside the browser and works at client side (Applets are not stand-alone programs)
- Two types of Applets (two varieties of applets based on **Applet** class)
 - ① Based directly on the **Applet** class
 - These applets use the Abstract Window Toolkit (AWT) to provide the graphical user interface (or use no GUI at all).
 - Available since Java was first created.
 - Used especially when only a very simple user interface is required.
 - ② Based on the **Swing** class **JApplet**, which inherits **Applet**
 - Swing applets use the Swing classes to provide the GUI.
 - Swing offers a richer and often easier-to-use user interface than the AWT. Thus, Swing-based applets are now the most popular.
 - Since **JApplet** inherits **Applet**, all the features of **Applet** are also available in **JApplet**.

Applet Basics

- There are many advantages of applet. They are as follows:
 - It works at client side so less response time.
 - Secured (*runs in a restricted environment*)
 - It can be executed by browsers running under many platforms, including **Linux, Windows, Mac OS** etc.
- Drawback of Applet is plugin is required at client browser to execute applet.
- How to run an Applet?
 - By **html** file.
 - By **appletViewer** tool (*for testing purpose*).
- **NOTE:** Latest versions of **Mozilla, Chrome, Safari, Opera** browsers do not support applets, so only using **Internet Explorer** you can test your applets.

Simple example of Applet by html file

- To execute the applet by html file, create an applet and compile it. After that create an html file and place the applet code in html file. Now click the html file.

//First.java

```
import java.applet.Applet;

import java.awt.Graphics;

public class First extends Applet{

    public void paint(Graphics g){

        g.drawString("welcome",150,150);

    }

}
```

```
<!-- myapplet.html -->
<html>
<body>
    <applet
        code ="First.class"
        width ="300"
        height ="300" >
    </applet>
</body>
</html>
```

- To execute the applet by html file

- 1) **javac First.java**
- 2) **open myapplet.html using Internet Explorer**

Simple example of Applet by appletviewer tool

- To execute the applet by appletviewer tool, create an applet that contains applet tag in comment and compile it. After that run it by: **appletviewer First.java**.

//First.java

```
import java.applet.Applet;
import java.awt.Graphics;
public class First extends Applet{
    public void paint(Graphics g){
        g.drawString("welcome",150,150);
    }
}
```

```
/*
<applet code="First.class" width="300" height="300">
</applet>
*/
```

- To execute the applet by appletviewer tool, write in command prompt:

javac First.java

appletviewer First.java

Simple example of Applet by appletviewer tool

```
Command Prompt - appletviewer First.java

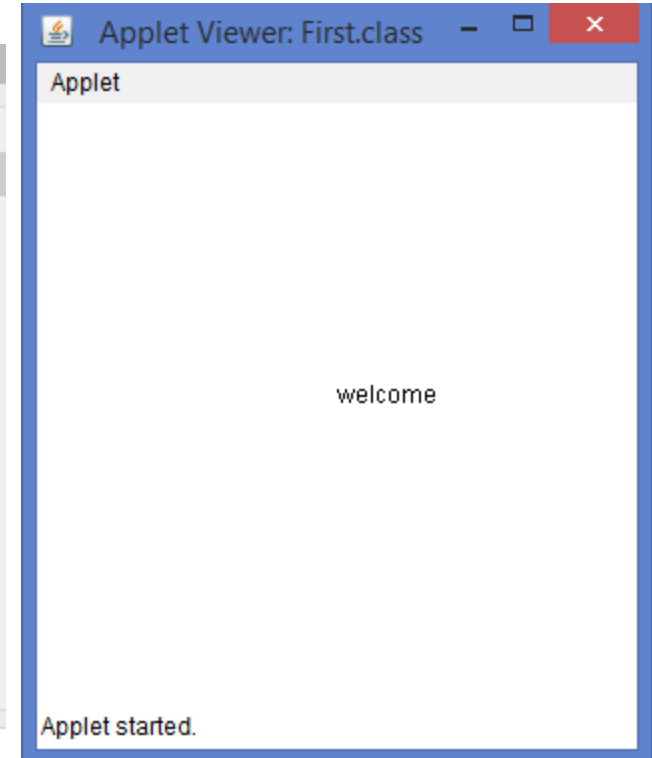
! error

C:\Users\M K Bhandari>cd Desktop
C:\Users\M K Bhandari\Desktop>cd JavaPrograms
C:\Users\M K Bhandari\Desktop\JavaPrograms>dir
Volume in drive C is OS
Volume Serial Number is 5E07-25B5

Directory of C:\Users\M K Bhandari\Desktop\JavaPrograms

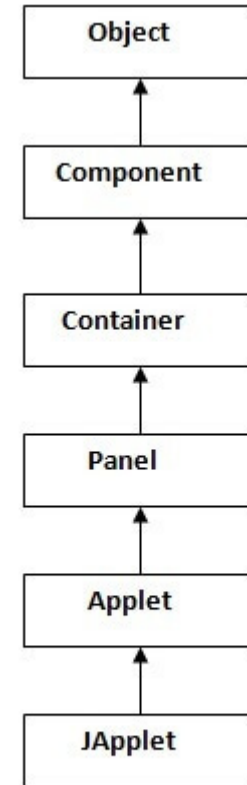
04/14/2020  06:07 PM    <DIR>          .
04/14/2020  06:07 PM    <DIR>          ..
04/15/2020  12:10 AM             359 First.class
04/15/2020  12:09 AM             278 First.java
04/14/2020  06:05 PM             107 My.java
04/14/2020  06:06 PM             107 myapplet.html
               4 File(s)              851 bytes
               2 Dir(s)  223,180,308,480 bytes free

C:\Users\M K Bhandari\Desktop\JavaPrograms>javac First.java
C:\Users\M K Bhandari\Desktop\JavaPrograms>appletviewer First.java
```



The Applet Class

- **Applet** provides all necessary support for applet execution, such as **starting and stopping**.
- It also provides methods that load and display images, and methods that load and play audio clips.
- **Applet** extends the AWT class **Panel**, **Panel** extends **Container**, which extends **Component**. As shown here ->
- These classes provide support for Java's window-based, graphical interface. Thus, **Applet** provides all of the necessary support for window-based activities.



The Methods Defined by Applet

Method	Description
<code>void destroy()</code>	Called by the browser just before an applet is terminated. Your applet will override this method if it needs to perform any cleanup prior to its destruction.
<code>AccessibleContext getAccessibleContext()</code>	Returns the accessibility context for the invoking object.
<code>AppletContext getAppletContext()</code>	Returns the context associated with the applet.
<code>String getAppletInfo()</code>	Overrides of this method should return a string that describes the applet. The default implementation returns null .
<code>AudioClip getAudioClip(URL url)</code>	Returns an AudioClip object that encapsulates the audio clip found at the location specified by <i>url</i> .
<code>AudioClip getAudioClip(URL url, String clipName)</code>	Returns an AudioClip object that encapsulates the audio clip found at the location specified by <i>url</i> and having the name specified by <i>clipName</i> .
<code>URL getCodeBase()</code>	Returns the URL associated with the invoking applet.
<code>URL getDocumentBase()</code>	Returns the URL of the HTML document that invokes the applet.
<code>Image getImage(URL url)</code>	Returns an Image object that encapsulates the image found at the location specified by <i>url</i> .
<code>Image getImage(URL url, String imageName)</code>	Returns an Image object that encapsulates the image found at the location specified by <i>url</i> and having the name specified by <i>imageName</i> .
<code>Locale getLocale()</code>	Returns a Locale object that is used by various locale-sensitive classes and methods.
<code>String getParameter(String paramName)</code>	Returns the parameter associated with <i>paramName</i> . null is returned if the specified parameter is not found.

The Methods Defined by Applet

Method	Description
<code>String[][] getParameterInfo()</code>	Overrides of this method should return a String table that describes the parameters recognized by the applet. Each entry in the table must consist of three strings that contain the name of the parameter, a description of its type and/or range, and an explanation of its purpose. The default implementation returns null .
<code>void init()</code>	Called when an applet begins execution. It is the first method called for any applet.
<code>boolean isActive()</code>	Returns true if the applet has been started. It returns false if the applet has been stopped.
<code>boolean isValidRoot()</code>	Returns true , which indicates that an applet is a validate root.
<code>static final AudioClip newAudioClip(URL url)</code>	Returns an AudioClip object that encapsulates the audio clip found at the location specified by <i>url</i> . This method is similar to <code>getAudioClip()</code> except that it is static and can be executed without the need for an Applet object.
<code>void play(URL url)</code>	If an audio clip is found at the location specified by <i>url</i> , the clip is played.
<code>void play(URL url, String clipName)</code>	If an audio clip is found at the location specified by <i>url</i> with the name specified by <i>clipName</i> , the clip is played.

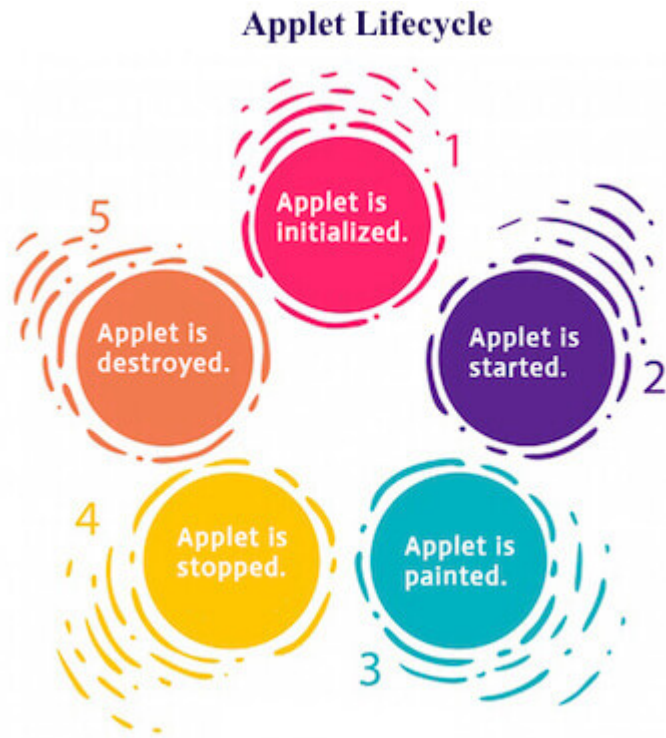
The Methods Defined by Applet

<code>void resize(Dimension <i>dim</i>)</code>	Resizes the applet according to the dimensions specified by <i>dim</i> . Dimension is a class stored inside java.awt . It contains two integer fields: width and height .
<code>void resize(int <i>width</i>, int <i>height</i>)</code>	Resizes the applet according to the dimensions specified by <i>width</i> and <i>height</i> .
<code>final void setStub(AppletStub <i>stubObj</i>)</code>	Makes <i>stubObj</i> the stub for the applet. This method is used by the run-time system and is not usually called by your applet. A <i>stub</i> is a small piece of code that provides the linkage between your applet and the browser.
<code>void showStatus(String <i>str</i>)</code>	Displays <i>str</i> in the status window of the browser or applet viewer. If the browser does not support a status window, then no action takes place.
<code>void start()</code>	Called by the browser when an applet should start (or resume) execution. It is automatically called after init() when an applet first begins.
<code>void stop()</code>	Called by the browser to suspend execution of the applet. Once stopped, an applet is restarted when the browser calls start() .

Applet Architecture

- An applet is a GUI-based program, its architecture is different from the console-based programs
 - ① Applets are event driven.
 - An applet waits until an **event** occurs.
 - The run-time system notifies the applet about an event by calling an **event handler** that has been provided by the applet.
 - It must perform specific actions in response to events and then return control to the run-time system.
 - ② The user initiates interaction with an applet—not the other way around
 - In a **console-based program**, when the program needs input, it will prompt the user and then call some input method, such as `readLine()`.
 - In applet, the user interacts with the applet as he or she wants, when he or she wants.
 - For example: **a mouse click, a keypress**
 - Applets can contain various controls, such as **push buttons** and **check boxes**. When the user interacts with one of these controls, **an event is generated**.

An Applet Skeleton (Lifecycle of an Applet)



- For creating any applet **java.applet.Applet** class must be inherited, provides 4 life cycle methods and **java.awt.Component** class provides 1 life cycle methods **paint()** for an applet.

- 1 **public void init()**: is used to **initialized** the Applet. **It is invoked only once.**
- 2 **public void start()**: is invoked after the **init()** method or browser is maximized. It is used to **start** the Applet.
- 3 **public void paint(Graphics g)**: is used to **paint** the Applet. **It provides Graphics class object that can be used for drawing oval, rectangle, arc etc.**
- 4 **public void stop()**: is used to **stop** the Applet. It is invoked when Applet is stop or browser is minimized.
- 5 **public void destroy()**: is used to **destroy** the Applet. **It is invoked only once.**

An Applet Skeleton (Lifecycle of an Applet)

// An Applet skeleton.

import java.awt.*;

import java.applet.*;

/*

<applet code="AppletSkel" width=300 height=100>

</applet>

*/

public class AppletSkel extends Applet {

// Called first.

public void init() {

// initialization

}

/* Called second, after init().

Also called whenever the applet is restarted. */

public void start() {

// start or resume execution

}

// Called when the applet is stopped.

public void stop() {

// suspends execution

}

/* Called when applet is terminated.

This is the last method executed. */

public void destroy() {

// perform shutdown activities

}

/* Called when an applet's window

must be restored. */

public void paint(Graphics g) {

// redisplay contents of window

}

}

An Applet Skeleton (Lifecycle of an Applet)

- Although this skeleton does not do anything, it can be compiled and run. When run, it generates the following empty window when viewed with appletviewer.



An Applet Skeleton (Lifecycle of an Applet)

- **Applet Initialization and Termination:** It is important to understand the order in which the various methods shown in the skeleton are called.
 - ① When an applet begins, the following methods are called, in this sequence:
 1. `init()`
 2. `start()`
 3. `paint()`
 - ② When an applet is terminated, the following sequence of method calls takes place:
 1. `stop()`
 2. `destroy()`
- **Overriding `update()`:** The default version of **`update()`** simply calls **`paint()`**. However, you can override the **`update()`** method so that it performs more subtle repainting.

Simple Applet Display Methods


- To output a string to an applet, use **drawString()**, which is a member of the **Graphics** class.
- Typically, it is called from within either **update()** or **paint()**. It has the following general form:


void drawString(String *message*, int x, int y)

- Here, *message* is the string to be output beginning at x,y. In a Java window, the upper-left corner is location 0,0.
- The drawString() method will not recognize newline characters. You must do so manually, specifying the precise X,Y location where you want the line to begin.

setBackground() and setForeground() methods

- To set the background color of an applet's window, use **setBackground()**. To set the foreground color use **setForeground()**. These methods are defined by **Component**, and they have the following general forms:

void setBackground(Color *newColor*) //setBackground(Color.green); 

void setForeground(Color *newColor*) //setForeground(Color.red); 

- Here, *newColor* specifies the new color. The class **Color** defines the constants shown here that can be used to specify colors:

Color.black	Color.magenta
Color.blue	Color.orange
Color.cyan	Color.pink
Color.darkGray	Color.red
Color.gray	Color.white
Color.green	Color.yellow
Color.lightGray	

getBackground() and getForeground() methods

- You can obtain the current settings for the background and foreground colors by calling **getBackground()** and **getForeground()**, respectively. They are also defined by **Component** and are shown here:

Color getBackground()

Color getForeground()

Example of setBackground() and setForeground() methods

```
/* A simple applet that sets the foreground and
background colors and outputs a string. */
import java.awt.*;
import java.applet.*;

/*
<applet code="Sample" width=300 height=50>
</applet>
*/

public class Sample extends Applet{
    String msg;
    // set the foreground and background colors.
    public void init( ) {
        setBackground(Color.cyan);
        setForeground(Color.red);
        msg = "Inside init( ) --";
    }
```

```
// Initialize the string to be displayed.
public void start( ) {
    msg += " Inside start( ) --";
}

// Display msg in applet window.
public void paint(Graphics g) {
    msg += " Inside paint( ).";
    g.drawString(msg, 10, 30);
}
```

A Simple Banner Applet

```
/* A simple banner applet. This applet creates a thread that  
scrolls the message contained in msg right to left across the  
applet's window.*/
```

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

```
/*
```

```
<applet code="SimpleBanner" width=300 height=50>  
</applet>
```

```
*/
```

```
public class SimpleBanner extends Applet  
    implements Runnable {
```

```
    String msg = " A Simple Moving Banner.";
```

```
    Thread t = null;
```

```
    int state;
```

```
    volatile boolean stopFlag;
```

```
    // Set colors and initialize thread.
```

```
    public void init( ) {
```

```
        setBackground(Color.cyan);
```

```
        setForeground(Color.red);
```

```
    }
```

```
    // Start thread
```

```
    public void start( ) {
```

```
        t = new Thread(this);
```

```
        stopFlag = false;
```

```
        t.start( );
```

```
    }
```

```
    // Entry point for the thread that runs the banner.
```

```
    public void run( ) {
```

```
        // Redisplay banner
```

```
        for( ; ; ) {
```

```
            try {
```

```
                repaint( );
```

```
                Thread.sleep(250);
```

```
                if(stopFlag)
```

```
                    break;
```

```
            } catch(InterruptedException e){ }
```

```
        }
```

```
    }
```

```
}
```

A Simple Banner Applet

```
// Pause the banner.  
public void stop() {  
    stopFlag = true;  
    t = null;  
}  
// Display the banner.  
public void paint(Graphics g) {  
    char ch;  
    ch = msg.charAt(0);  
    msg = msg.substring(1, msg.length());  
    msg += ch;  
    g.drawString(msg, 50, 30);  
}  
}
```

- **Execute the program and see how Banner is displayed?**

// Self Study from Page Nos. 759-764

- Using the Status Window
- The HTML APPLET Tag
- Passing Parameters to Applets

References

R Reference for this topic

- [Book: Java: The Complete Reference, Ninth Edition: Herbert Schildt]
<https://www.amazon.in/Java-Complete-Reference-Herbert-Schildt/dp/0071808558>
- [Web: GeeksforGeeks]
<https://www.geeksforgeeks.org/java/>
- [Web: Java T Point tutorial]
<https://www.javatpoint.com/java-tutorial>