IIT KHARAGPUR

NPTEL ONLINE CERTIFICATION COURSES

OBJECT ORIENTED PROGRAMMING WITH JAVA

Java Applet Programming - I

Debasis Samanta
Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur

What is an applet?

Java programs are available in two flavors.

Applet

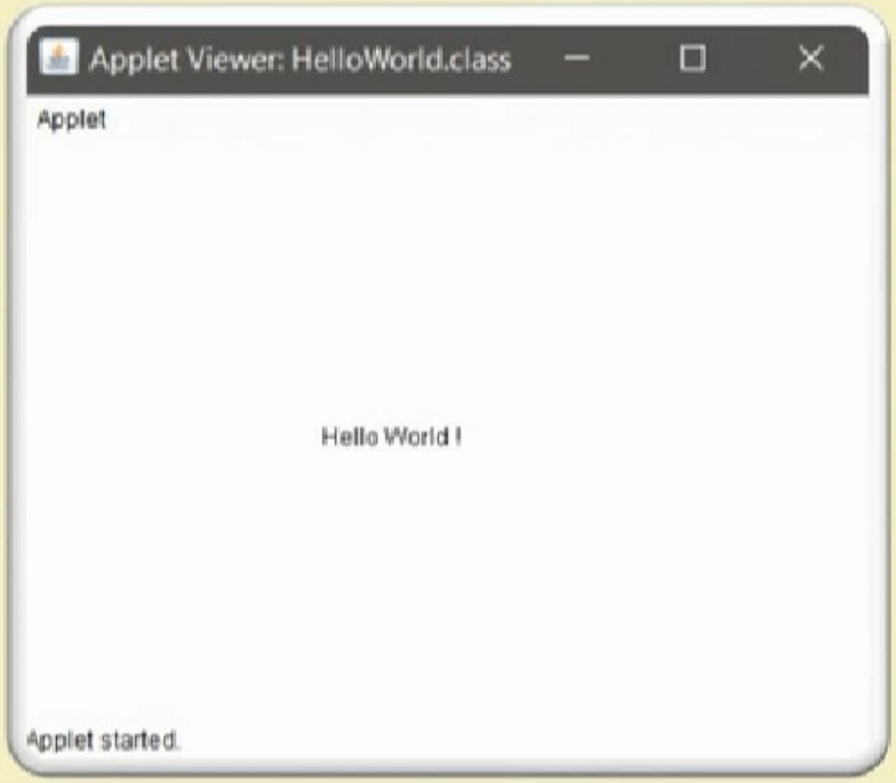
- A Java applet is a program that appears embedded in a web document and applet come into effect when the browser browse the web page.

Application

- It is similar to all other kind of programs like in C, C++, Pascal, etc. to solve a problem.

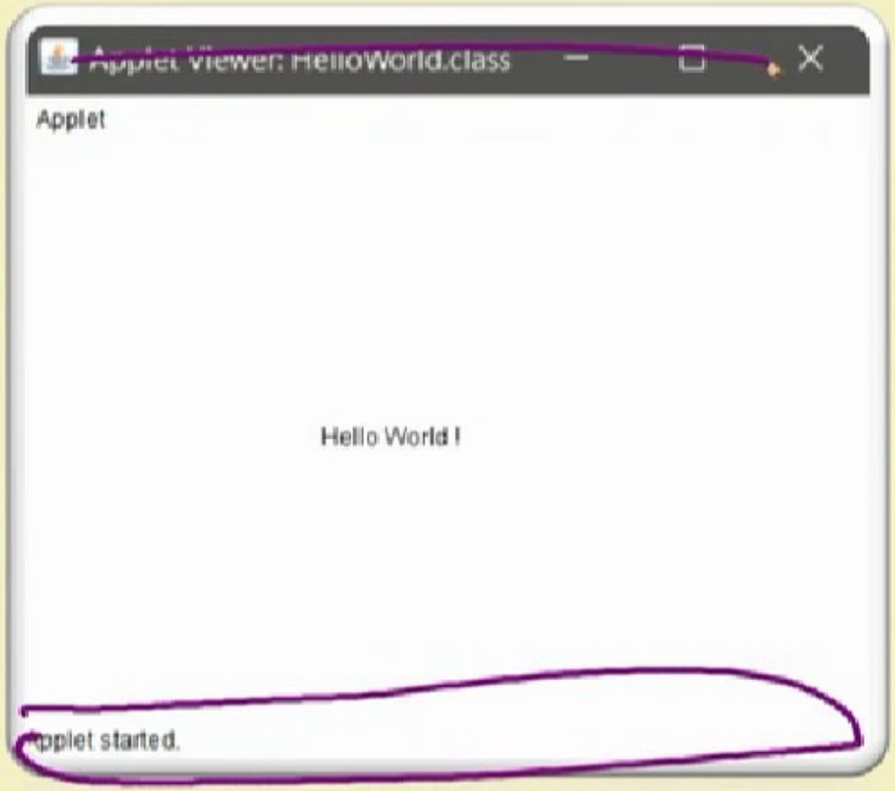
How an applet looks like?

Typically, a Java applet looks as below.



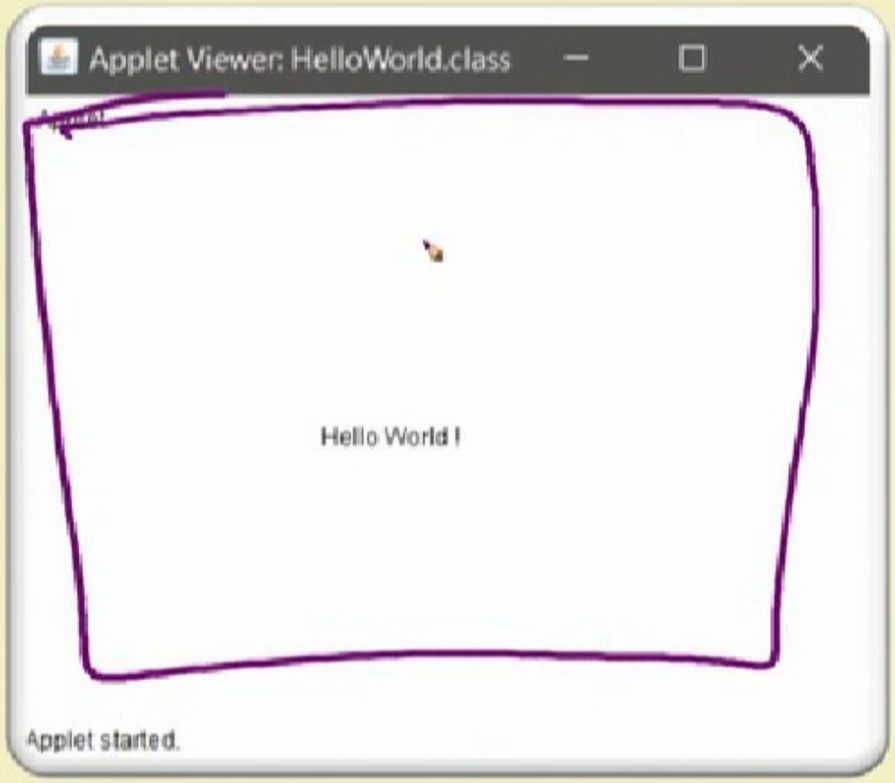
How an applet looks like?

Typically, a Java applet looks as below.



How an applet looks like?

Typically, a Java applet looks as below.



Applet program writing

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

public class HelloWorld extends Applet{
        public void paint(Graphics g){
                g.drawString("Hello World !",150,150);
        }
}
```

Building an applet

Edit → Save → Compile

- Edit the code in the same fashion as an application.

- The name of the applet will be same as the public class, defining the applet, for example, here

`HelloWorld.java`

Building an applet

Edit → Save → Compile

The program can be compiled in the same fashion as a Java application is compiled. That is,

```
javac HelloWorld.java
```

Building an applet

Edit → Save → Compile

After the successful compilation, the `javac` will produce a file named

`HelloWorld.class`

Hosting an applet with HTML

Edit an HTML file to host the applet just created. The HTML file will look like as:

```
<html>
   <body>
      <applet width="300" height="300" codebase="." code="HelloWorld.class">
      </applet>
   </body>
</html>
```

Hosting an applet with HTML

```
<html>
   <body>
      <applet width="300" height="300" codebase="." code="HelloWorld.class">
      </applet>
   </body>
</html>
```

- Write this program using any editor, for example, Notepad, Wordpad, etc.
- Save this to file giving a file name `HelloJava.html`

  Note: The name of the file is not necessarily be the same as the name of the class; but extension should be .html
- Now, the applet is ready for its execution!

Running an applet

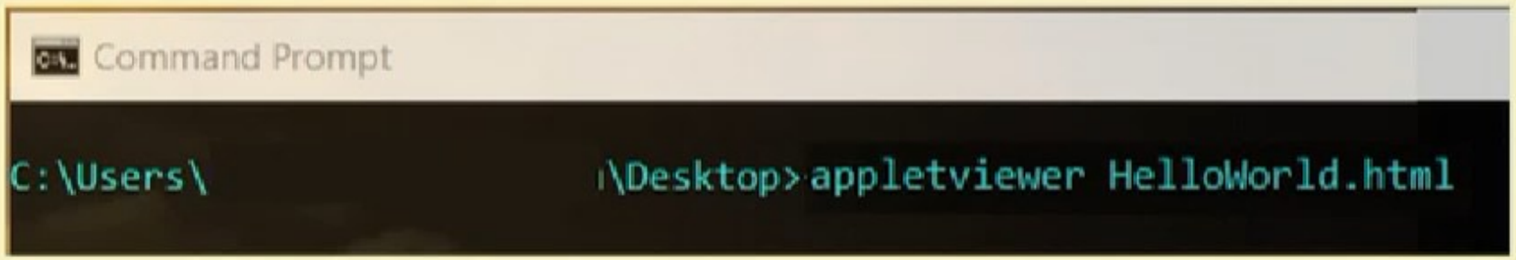
Execution

- A .html file can be run with a browser such as Internet Explorer, Netscape Navigator, etc.
- Java has its own program called **appletviewer** to run an applet.

Running an applet

Execution

- Open the command prompt and go to the program directory
- Run applet by typing appletviewer <file name>.class



- Alternatively, double click on the .html file so that the default brow your machine can run the .html file

Running an applet

Execution

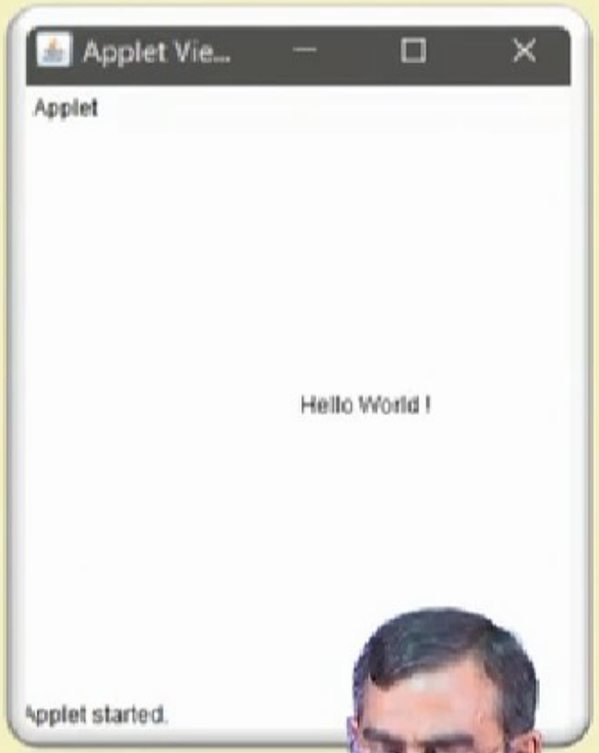
Structure of an Applet

Applet revisited

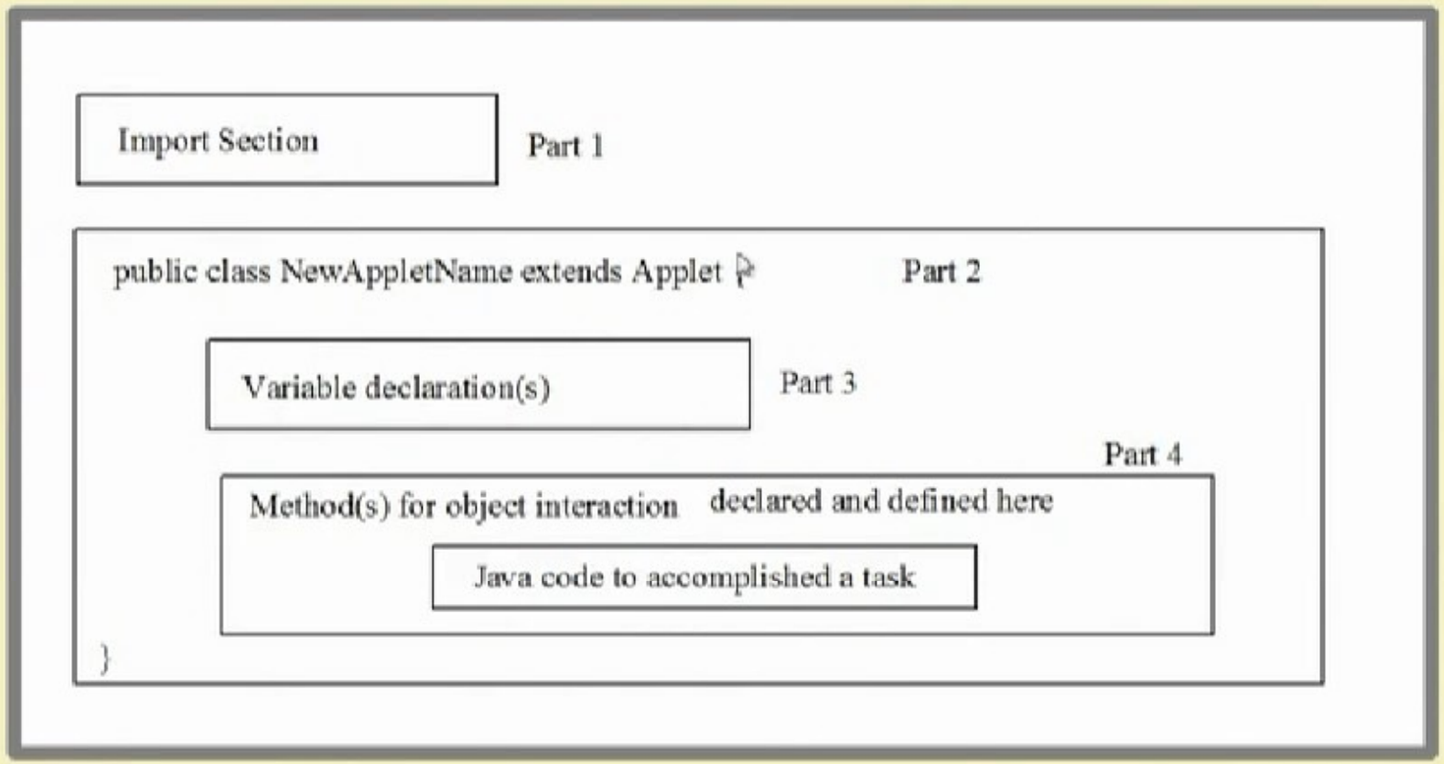
```
import java.applet.Applet;
import java.awt.Graphics;

public class HelloWorld extends Applet{
    public void paint(Graphics g){
            g.drawString("Hello World!",150,150);
    }
}
```

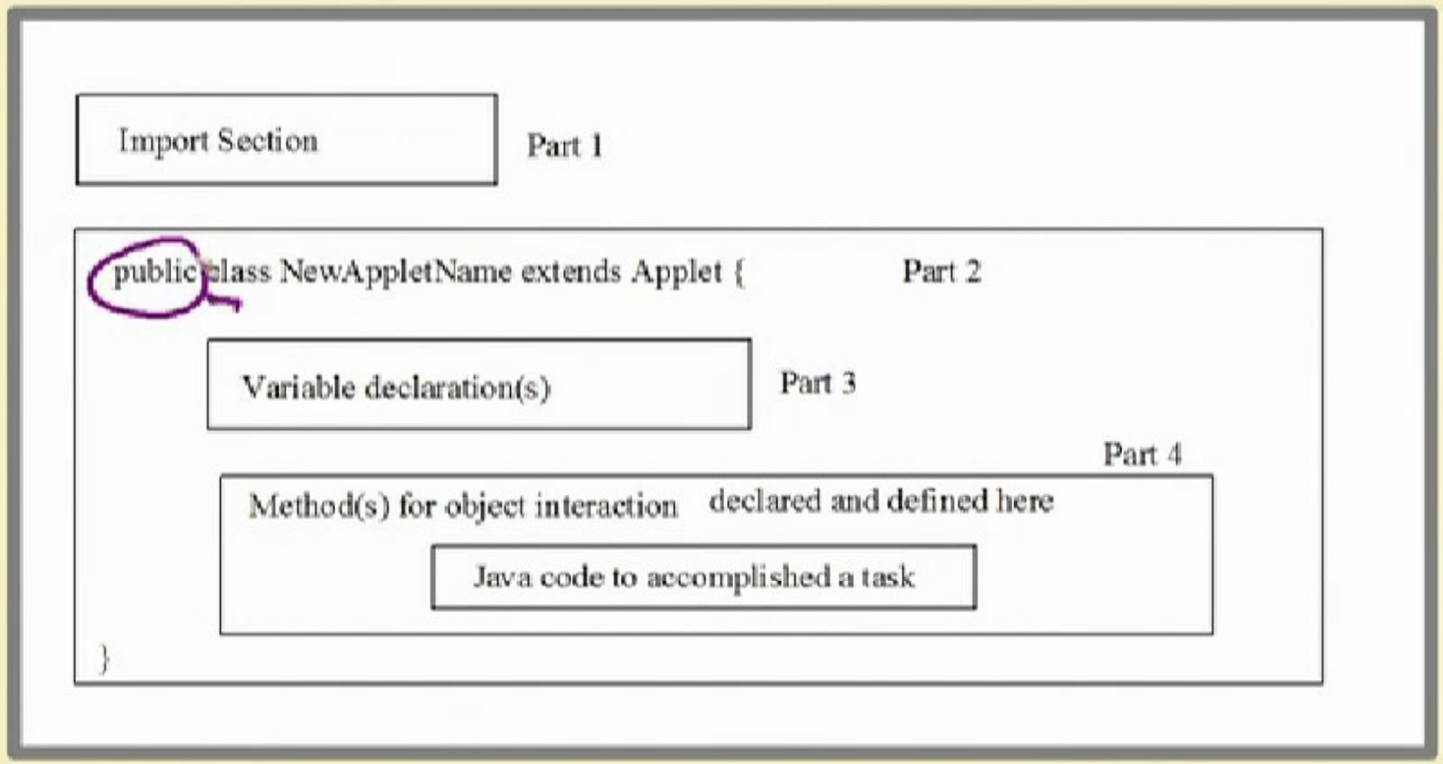
```
<html>
   <body>
      <applet width="300" height="300" code="HelloWorld.class">
      </applet>
   </body>
</html>
```



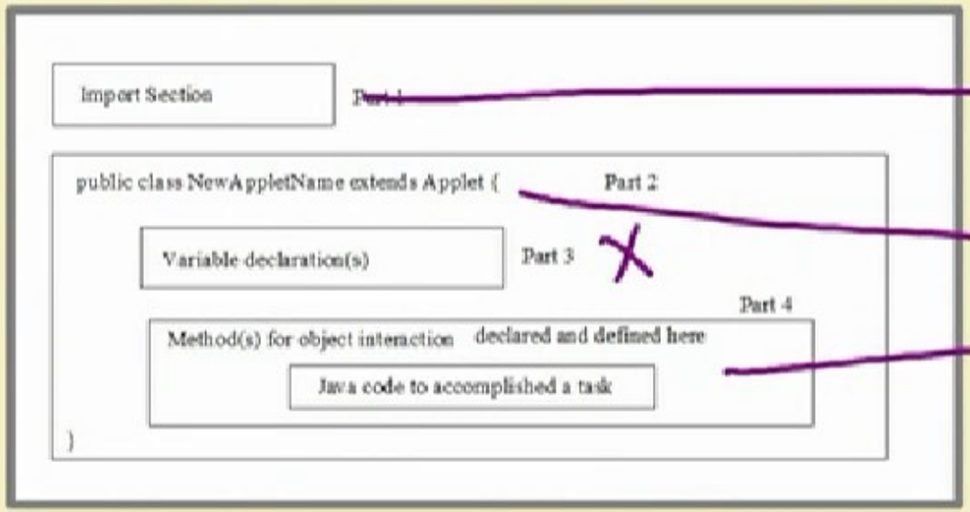
Basic structure of an applet



Basic structure of an applet



Basic structure of an applet




```
import java.applet.Applet;
import java.awt.Graphics;

public class HelloWorld extends Applet{
    public void paint(Graphics g){
        g.drawString("Hello World!",150,150);
    }
}
```

Basic methods in applet

- `public void init( )`
  - To initialize or pass input to an applet.
- `public void start( )`
  - The `start( )` method is called after the `init( )` method, it starts an applet.
- `public void stop( )`
  - To stop a running applet.

Basic methods in Applet

- `public void paint (Graphics g)`
  - To draw something within an applet.
- `public void destroy( )`
  - To remove an applet from memory completely.

```
import java.applet.Applet;
import java.awt.Graphics;

public class HelloWorld extends Applet{
    public void paint(Graphics g){
            g.drawString("Hello World!",150,150);
    }
}
```

Basic methods in applet

```
// Use of init( ) method in an applet //

import java.applet.Applet;
import java.awt.Graphics;

public class HelloWorld extends Applet{
   public void init( ) {
         resize(200,200);
    }

    public void paint(Graphics g){
          g.drawString("Hello World!",150,150);
    }
}
```

Basic methods in applet

```
// Use of init( ) method in an applet //

import java.applet.Applet;
import java.awt.Graphics;

public class HelloWorld extends Applet{
   public void init( ) {
        resize(200,200);
   }

    public void paint(Graphics g){
              g.drawString("Hello World!",150,150);
    }
}
```

```
<html>
   <body>
      <applet width="300" height="300" code="HelloWorld.class">
      </applet>
   </body>
</html>
```

Input to an Applet

Input passing to an applet

```
// Use of  init( ) to pass value through HTML to applet //

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

public class RectangleTest extends Applet {
            int x, y, w, h;
            public void init (  ) {
               x  = Integer.parseInt(getParameter (" xValue" ));
               y  = Integer.parseInt(getParameter (" yValue" ));
               w =  Integer.parseInt(getParameter (" wValue" ));
               h  = Integer.parseInt(getParameter (" hValue" ));
            }

            public void paint ( Graphics g ) {
               g.drawRect (x, y, w, h );
         }
    }
```

Input passing to an applet

```
// Use of  init( ) to pass value through HTML to applet //

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

public class RectangleTest extends Applet {
        int x, y, w, h;
        public void init (  ) {
          x = Integer.parseInt(getParameter ("xValue" ));
          y = Integer.parseInt(getParameter ("yValue" ));
          w = Integer.parseInt(getParameter ("wValue" ));
          h = Integer.parseInt(getParameter ("hValue" ));
        }

        public void paint ( Graphics g ) {
           g.drawRect (x, y, w, h );
      }
  }
```

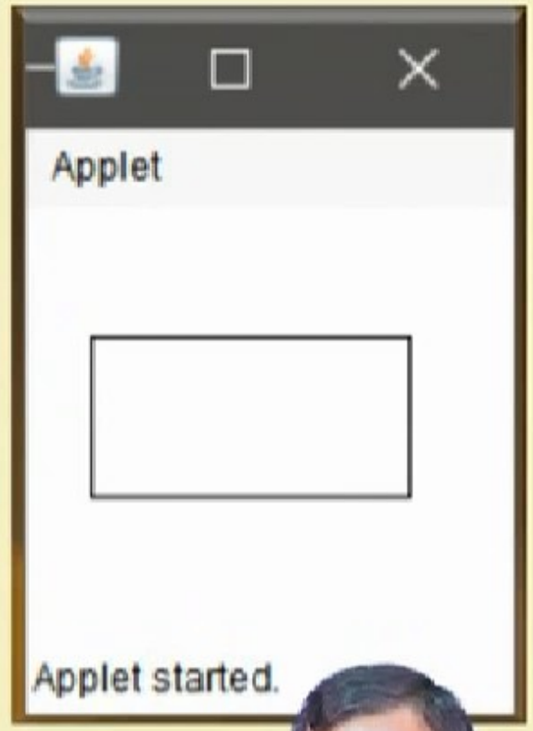
Input passing to an applet

Corresponding HTML document containing this applet and providing parameter values will be :

```
<applet code = " RectangleTest"  width = 150  height = 100 >
    < param name = xValue  value = 20 >
    < param name = yValue  value = 40 >
    < param name = wValue  value = 100>
    < param name = hValue  value = 50 >
</applet >
```

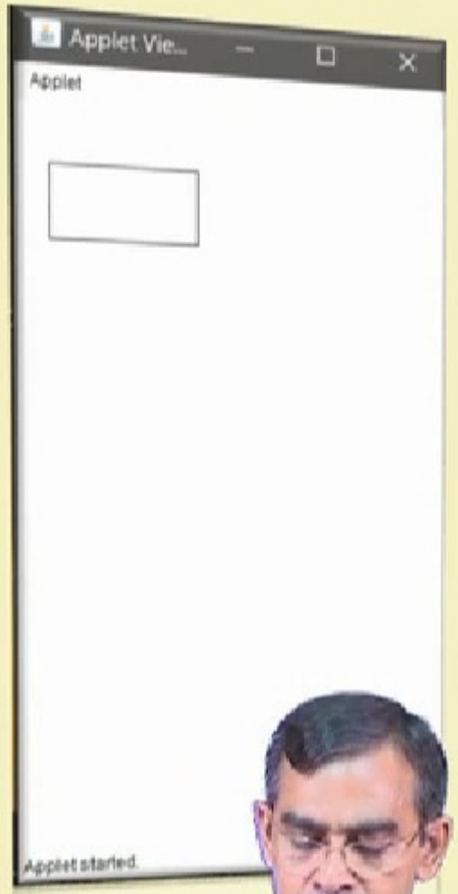
Basic methods in Applet - with resize

```
<applet code = " RectangleTest"  width = 150  height = 100 >
    < param name = xValue  value = 20 >
    < param name = yValue  value = 40 >
    < param name = wValue  value = 100>
    < param name = hValue  value = 50 >
</applet >
```



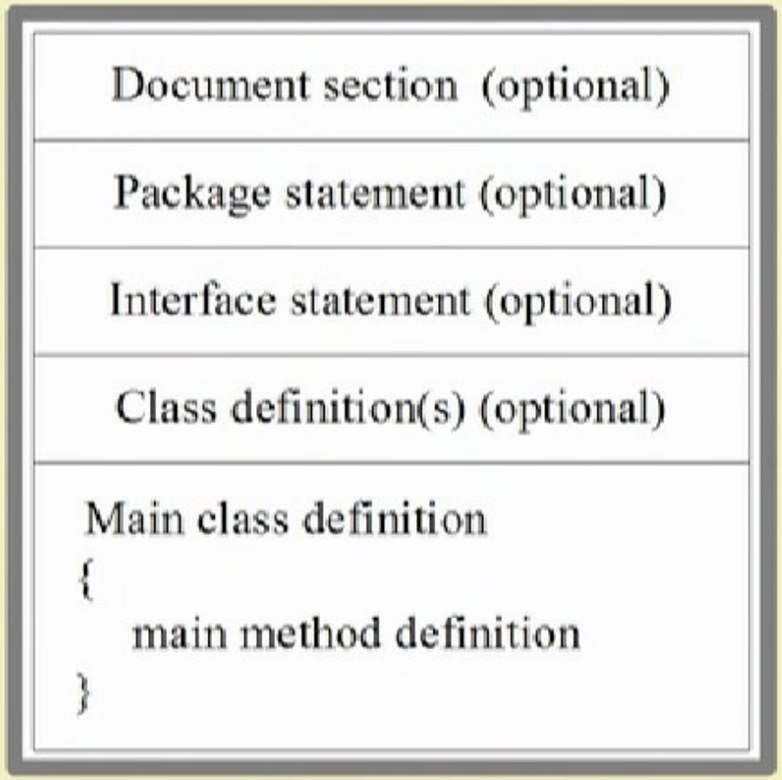
Basic methods in Applet - without resize

```
<applet code = " RectangleTest"  width = 150  height = 100 >
    < param name = xValue  value = 30 >
    < param name = yValue  value = 30 >
    < param name = wValue  value = 30>
    < param name = hValue  value = 30 >
</applet >
```

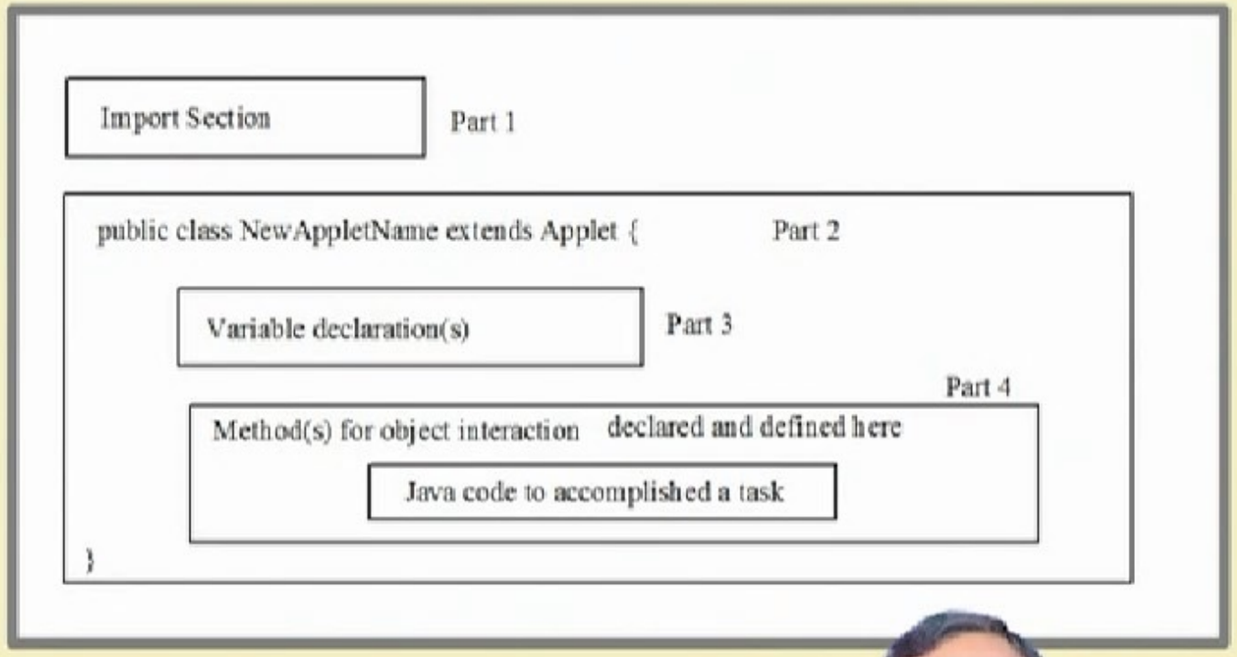


Application versus Applet

Application versus applet



Basic structure of an Application



Basic structure of an Ap

Application versus applet

- Applets do not use main() method for initiating the execution of code. Applets, when loaded, automatically call certain methods of Applet class to start and execute the code in applets
- Unlike application (stand alone), applets cannot be run independently. They are to be embedded in HTML pages as applet code, which browser can run
- Applet cannot read from or write to the file in the local computers
- Applet cannot communicate with other severs in the networks
- Applet cannot run any program from local computers
- Applets are restricted from using libraries from other languages, such as, C, C++, etc.

Questions to think...

- How more sophisticated applets can be designed in different applications?
- What is the latest technology known for applet?

Lecture 35
Applet Programming - II

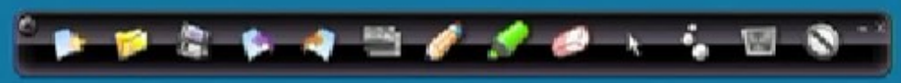
IIT KHARAGPUR

NPTEL ONLINE
CERTIFICATION COURSES

OBJECT ORIENTED PROGRAMMING WITH JAVA

Java Applet Programming – II

Debasis Samanta
Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur



Applet Basics


Biography
Research
Publications
Courses
Projects
Students
Others
Classic Data Structures
View Details
OBJECT-ORIENTED PROGRAMMING WITH C++ AND JAVA
D. SAMANTA
View Details
View Details
Research & Development
Multimodal Interaction
Text Entry System

Applet concept

- All applets are sub-class of **`Applet`** (defined in `java.applet` package in JDK. Hence, all applets must **`import`** `java.applet.*` package.
- Also, all applets must **`import`** `java.awt package`. AWT stands for *Abstract Window Toolkit*. As applets run in a window, so it is necessary to support for that.
- Applets are not executed by console-based Java runtime interpreter.
- In fact, applets are executed by a web browser (such as, Hot Java, Netscape Navigator, Internet Explorer 4.0, etc.)
- Also, applets can be executed by the standard executable program called **`appletviewer`** provided by Java$_2$™ SDK.

Execution of an applet

Execution of an applet is controlled with entirely different mechanism.

- As such, there is no `main()` method in an applet program, hence applet does not begin at `main()`.
- Output of an applet is not performed using `System.out.print()` function. Rather, it is handled with various AWT methods, such as `drawString()`, which outputs a string to a specified `(x, y)` coordinates.
- Input is also handled differently than a normal Java application.

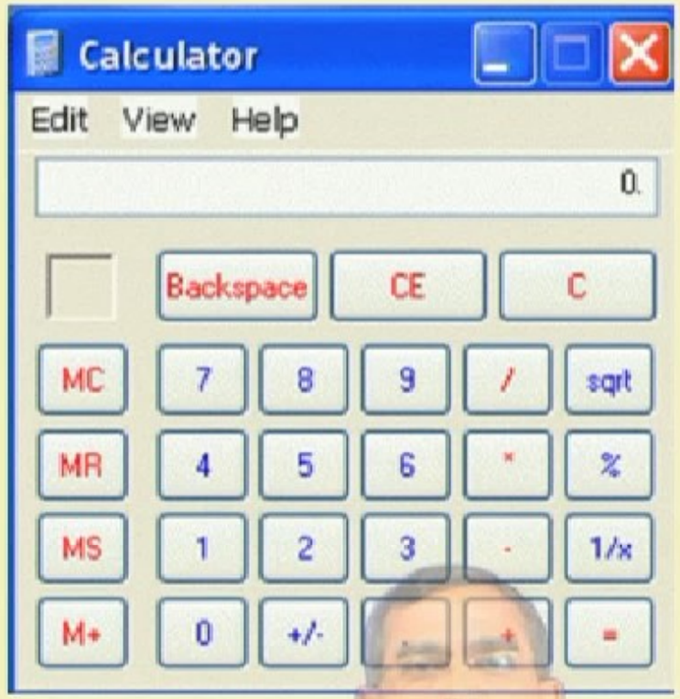
Applet window

An applet is a window-based program.

- So called normal, console-based programs, we call them as application (also called Java core programs) are different than applets.
- As in window programs, applets are event driven. An applet waits until an event occurs. The AWT notifies the applet about an event by calling an event handler that has been provided by the applet. Once this happens, the applet must take appropriate action and then quickly return control to the AWT.
- Here, an event execution implies the start of a thread.
- User should initiate interaction with an applet. It is unlike a non-windowed program, where program prompts for an input, then read it and execute (either in single or multiple threads).

Applet events

- In applets, events are triggered by a key space, mouse click, mouse drag, etc.
- More precisely, applet contains various controls, such as buttons, text fields, checkboxes, labels, scroll bars, etc.
- User can interact with these controls to generate events.



Applet Skeleton and Syntax

DEBASIS SAMANTA
IIT KHARAGPUR

Basic applet Structure

Import Statements

init() method

start() method

stop() method

paint() method

destroy() method

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

public class <APPLET-NAME> extends Applet{
  public void init(){
    //Code for applet initialization.
  }
  public void start(){
    //This method is called after init()
    //Also called whenever applet is restarted.
    //This method contains code to start or resume execution.
  }
  public void stop(){
    //This method is called when the applet is stopped.
    //This method contains code to suspend the execution of applet.
  }
  public void paint(){
    //This method contains code to paint the window of an applet.
  }
  public void destroy(){
    //This method is called when the applet is terminated.
    // This is the last method in the execution of an applet.
    /*
    This method includes the code to perform shutdown activities.
    Usually, the codes are the methods defined in:
    java.awt package
    */
  }
}
```

Class declaration

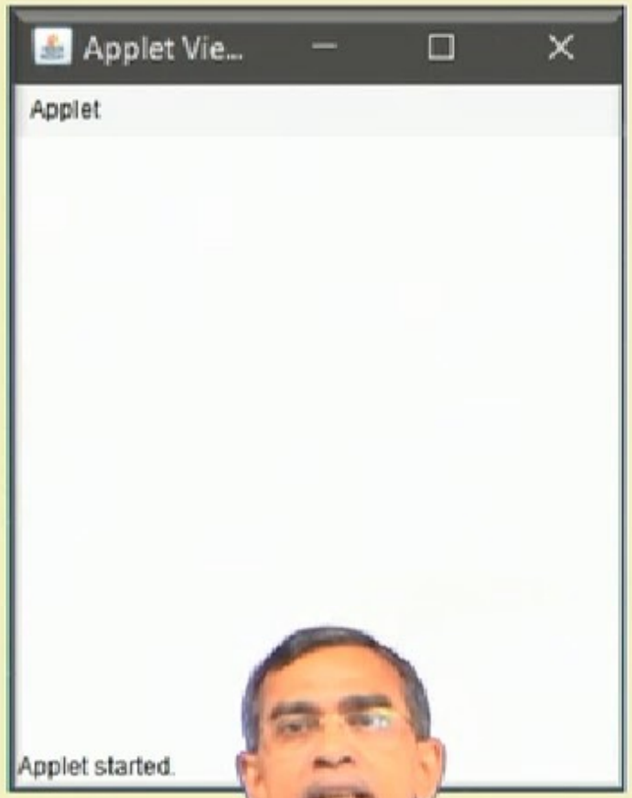
Designing an Applet

Applet with blank methods

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

public class AppletSkeleton extends Applet {
  public void init() {  }
  public void start() {  }
  public void stop() {  }
  public void destroy() {  }
  public void paint(Graphics g) {  }
}
```

```
<html>
   <body>
      <applet width="300" height="300" code="AppletSkeleton.class">
      </applet>
   </body>
</html>
```



Applet with blank methods

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

public class AppletSkeleton extends Applet {
  public void init() {  }
  public void start() {  }
  public void stop() {  }
  public void destroy() {  }
  public void paint(Graphics g) {  }
}
```

```
<html>
    <body>
        <applet width="300" height="300" code="AppletSkeleton.class">
        </applet>
    </body>
</html>
```

Order of invocation of Applet methods

- These are abstract methods defined in abstract class Applet and they need to be overridden in applet programs. It is important to note the order in which the various methods are called.

- When an applet begins, the AWT calls the method in the sequence

`init()` → `start()` → `paint()`

- When an applet is terminated, the following sequence of methods call takes place

`stop()` → `destroy()`

- Note: All these methods are optional.

Applet initialization and start : Methods

init( )

The `init()` method is the first method to be called. This is where you should initialize an applet. This method is called only once during the run time of your applet.

start( )

The `start()` method is called after `init()`. It is also called to restart an applet after it has been stopped. Whereas `init()` is called once—the first time an applet is loaded—`start()` is called each time an applet's HTML document is displayed onscreen. So, if a user leaves a web page and comes back, the applet resumes execution at `start()`.

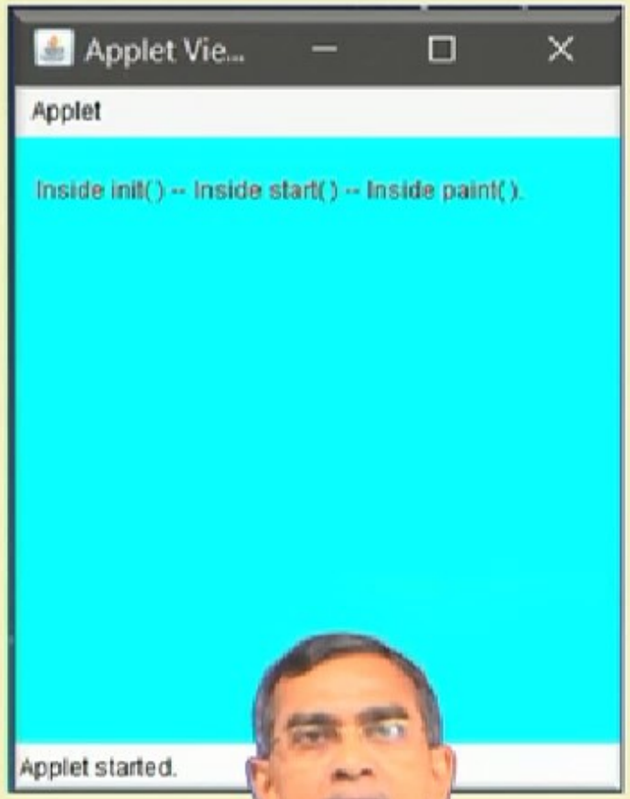
Applet paint method

paint( )

The `paint()` method is called each time the applet's output to be redrawn. This situation can occur for several reasons. For example, the window in which the applet is running may be overwritten by another window and then uncovered. Or the applet window may be minimized and then restored. `paint()` is also called when the applet begins execution. Whatever the cause, whenever the applet must redraw its output, `paint()` is called. The `paint()` method has one parameter of type `Graphics`. This parameter will contain the graphics context, which describes the graphics environment in which the applet is running. This context is used whenever output to the applet is required.

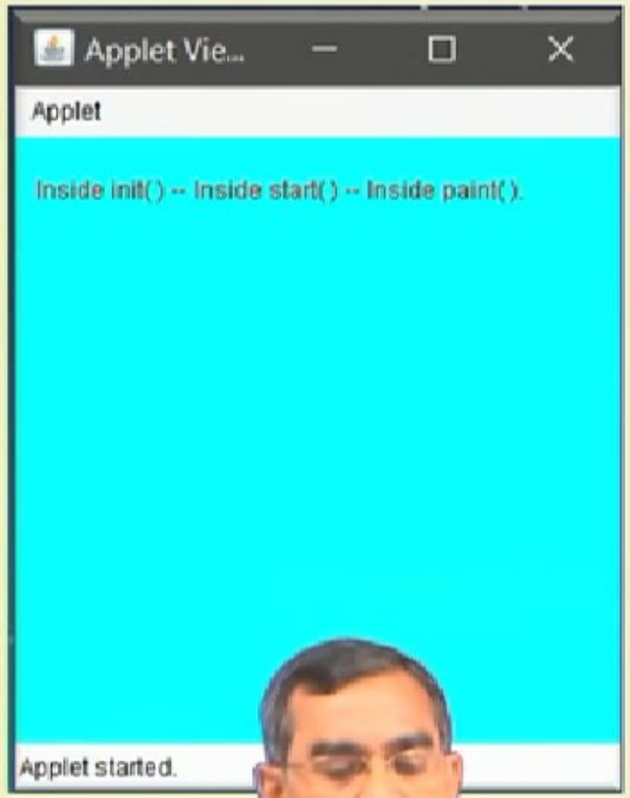
Order of execution of methods : An example

```
import java.awt.*;
import java.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);
  }
}
```



Order of execution of methods : An example

```
import java.awt.*;
import java.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);
  }
}
```



Applet termination : Methods

stop( )

The `stop()` method is called when a web browser leaves the HTML document containing the applet—when it goes to another page, for example. When `stop()` is called, the applet is probably running. You should use `stop()` to suspend threads that do not need to run when the applet is not visible. One can restart them when `stop()` is called if the user returns to the page.

destroy( )

The `destroy()` method is called when the environment determines that your applet needs to be removed completely from memory. At this point, you should free up any resources the applet may be using. The `stop()` method is always called before `destroy()`.

Use of stop(): An example

```
//  Create a child frame window from within an applet.
import java.awt.*;
import java.awt.event.*;
import java.applet.*;

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

// Create a subclass of Frame.
class SampleFrame extends Frame {
     SampleFrame(String title) {
            super(title);
            // create an object to handle window events
            MyWindowAdapter adapter = new MyWindowAdapter(this); // register it to receive those events
            addWindowListener(adapter);
     }
     public void paint(Graphics g) {
            g.drawString("This is in frame window", 10, 40);
     }
}
```

Use of stop(): An example

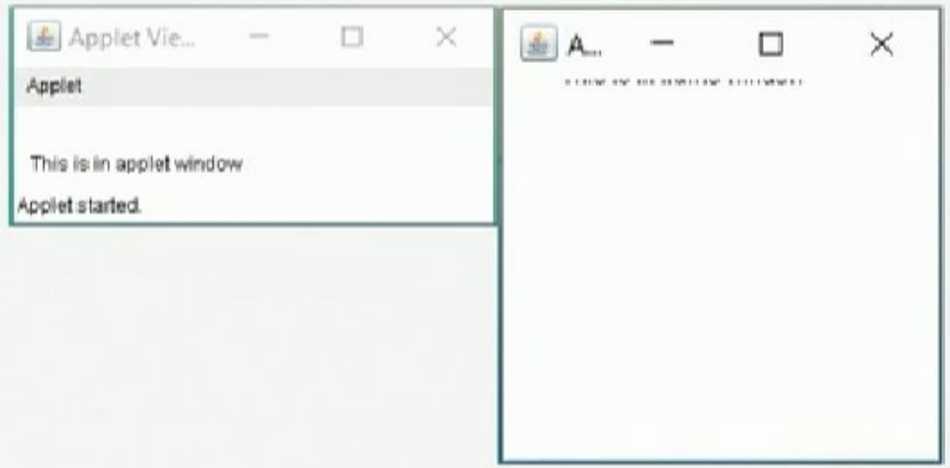
```
//  Create a child frame window from within an applet.
import java.awt.*;
import java.awt.event.*;
import java.applet.*;

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

// Create a subclass of Frame.
class SampleFrame extends Frame {
    SampleFrame(String title) {
        super(title);
        // create an object to handle window events
        MyWindowAdapter adapter = new MyWindowAdapter(this); // register it to receive those events
        addWindowListener(adapter);
    }
    public void paint(Graphics g) {
        g.drawString("This is in frame window", 10, 40);
    }
}
```

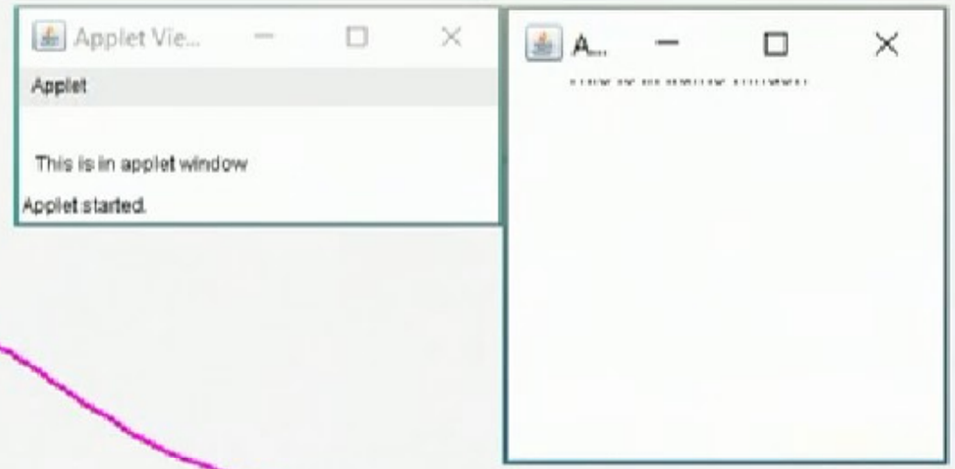
Use of stop(): An example

```
class MyWindowAdapter extends WindowAdapter {
    SampleFrame sampleFrame;
    public MyWindowAdapter(SampleFrame sampleFrame) {
        this.sampleFrame = sampleFrame;
    }
    public void windowClosing(WindowEvent we) {
        sampleFrame.setVisible(false);
    }
}
// Create frame window.
public class AppletFrame extends Applet {
    Frame f;
    public void init() {
        f = new SampleFrame("A Frame Window");
        f.setSize(250, 250);
        f.setVisible(true);
    }
    public void start() { f.setVisible(true); }
    public void stop() { f.setVisible(false); }
    public void paint(Graphics g) { g.drawString("This is in applet window", 10, 20); }
}
```



Use of stop(): An example

```
class MyWindowAdapter extends WindowAdapter {
    SampleFrame sampleFrame;
    public MyWindowAdapter(SampleFrame sampleFrame) {
        this.sampleFrame = sampleFrame;
    }
    public void windowClosing(WindowEvent we) {
        sampleFrame.setVisible(false);
    }
}
// Create frame window.
public class AppletFrame extends Applet {
    Frame f;
    public void init() {
        f = new SampleFrame("A Frame Window");
        f.setSize(250, 250);
        f.setVisible(true);
    }
    public void start() { f.setVisible(true); }
    public void stop() { f.setVisible(false); }
    public void paint(Graphics g) { g.drawString("This is in applet window", 10, 20); }
}
```



Use of destroy(): An example

```
import java.applet.Applet;
import java.awt.Graphics;

public class AppletLifecycle extends Applet{
   StringBuffer strBuffer;
   public void init() {
      strBuffer = new StringBuffer();
      addItem("initializing the apple ");
   }
   public void start() {
      addItem("starting the applet ");
   }
   public void stop() {
      addItem("stopping the applet ");
   }
   public void destroy() {
      addItem("unloading the applet");
   }
```

```
void addItem(String word) {
      System.out.println(word);
      strBuffer.append(word);
      repaint();
   }
   public void paint(Graphics g) {
      // Draw a Rectangle around the applet's display area.
      g.drawRect(0, 0,getWidth() - 1,getHeight() - 1);
      // display the string inside the rectangle.
      g.drawString(strBuffer.toString(), 10, 20); }
}
//<applet code="AppletLifecycle.class" width="300" height="200"></applet>
```

Use of destroy(): An example

```
import java.applet.Applet;
import java.awt.Graphics;

public class AppletLifecycle extends Applet{
   StringBuffer strBuffer;
   public void init() {
      strBuffer = new StringBuffer();
      addItem("initializing the apple ");
   }
   public void start() {
      addItem("starting the applet ");
   }
   public void stop() {
      addItem("stopping the a
   }
   public void destroy() {
      addItem("unloading the
   }
```

```
void addItem(String word) {
      System.out.println(word);
      strBuffer.append(word);
      repaint();
   }
   public void paint(Graphics g) {
      // Draw a Rectangle around the applet's display area.
      g.drawRect(0, 0,getWidth() - 1,getHeight() - 1);
                                                  he rectangle.
                                                  g(), 10, 20); }

                                                  ss" width="300" height="200"></applet>
```

```
appletviewer AppletLifecycle.java
initializing the apple
starting the applet
stopping the applet
unloading the applet
```

Applet update() Methods

- The **update()** method is defined in Applet class. By default, it fills an applet with default background color and then calls **paint()** method.
- Programmer can override the **update()** method, so that it performs all the intended activities, then in **paint()** to call **update()**.

```
public void update(Graphics g) {
    setBackground(Color.black);
    setForeground(Color.whilte);
    g.drawString("I am from update", 30,40);
    // redisplay your window, here.
}

public void paint(Graphics g) {
  update(g);
  g.drawString("I am from paint", 35,40);
}
```

Applet repaint() method

- Typically, an applet writes to its window using `update()` or `paint()`.
- **Note:** One cannot create a loop in `paint()` to change the information content in an applet.
- Using the `repaint()` method this can be done.
- The `repaint()` causes the AWT runtime system to execute a call to your applet's `update()`, which in turn calls `paint()`.

Applet repaint() methods

This version causes the entire window to be repainted and is the simplest version

```
void repaint( ) {…. }
```

This version specifies a region that will be repainted

```
void repaint(int left, int top, int width, int height) {..};
```

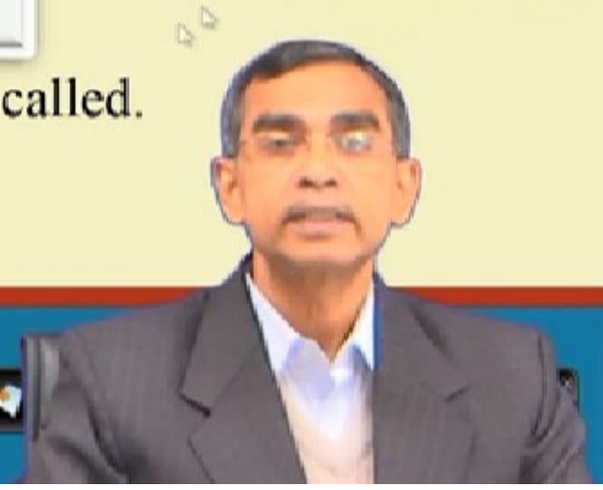
Applet repaint() Methods

```
void repaint(long maxDelay);
```

Here, **maxDelay** specifies the maximum number of milliseconds that can elapse before **update()** is called.

```
void repaint(long maxDelay, int x, int y, int width, int height);
```

Note: If the time elapses before **update()** can be called, it isn't called. There's no return value or exception thrown, so you must be careful.



Simple Banner applet : An example

```
/* 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.*;

public class SimpleBanner extends Applet implements Runnable {
  String msg = " A Simple Moving Banner.";
  Thread t = null;
  int state;
  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;
```

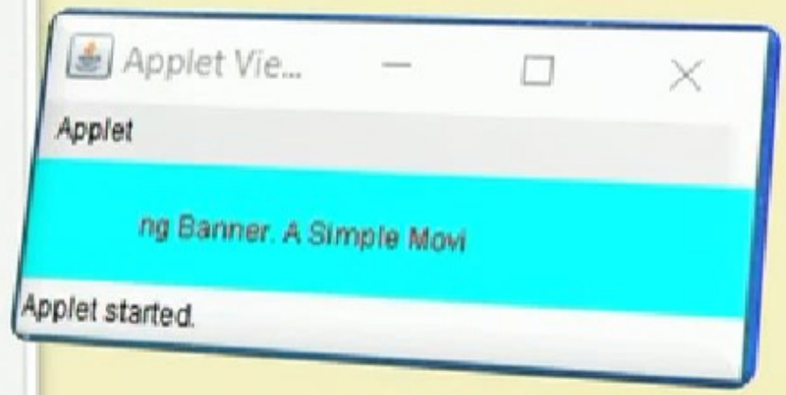
Simple Banner applet : An example

```
/* 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.*;

public class SimpleBanner extends Applet implements Runnable {
  String msg = " A Simple Moving Banner.";
  Thread t = null;
  int state;
  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;
```

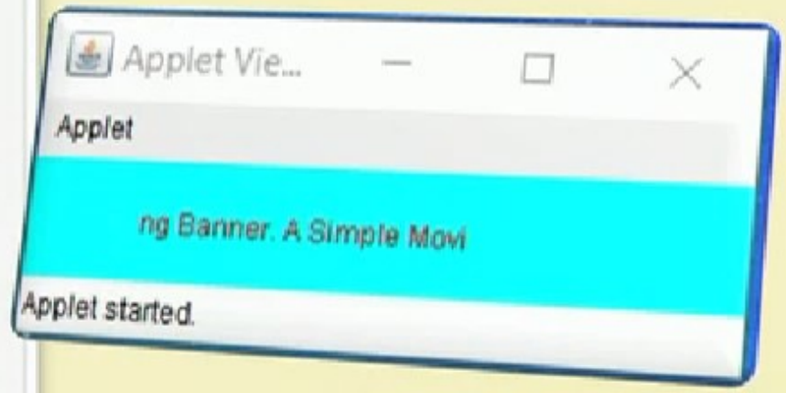
Simple Banner applet : An example

```
 t.start();
}
// Entry point for the thread that runs the banner.
public void run() {
  char ch;
  // Display banner
  for( ; ; ) {
    try {
      repaint();
      Thread.sleep(250);
      ch = msg.charAt(0);
      msg = msg.substring(1, msg.length());
      msg += ch;
      if(stopFlag)
        break;
      }
    catch(InterruptedException e) {  }
  }
}
// Pause the banner.
public void stop() {
  stopFlag = true;
```



Simple Banner applet : An example

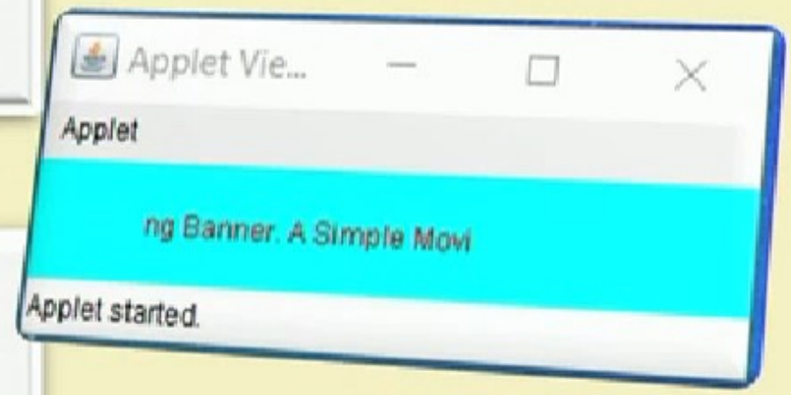
```
    t.start();
  }
  // Entry point for the thread that runs the banner.
  public void run() {
    char ch;
    // Display banner
    for( ; ; ) {
      try {
        repaint();
        Thread.sleep(250);
        ch = msg.charAt(0);
        msg = msg.substring(1, msg.length());
        msg += ch;
        if(stopFlag)
          break;
        }
      catch(InterruptedException e) {  }
    }
  }
  // Pause the banner.
  public void stop() {
    stopFlag = true;
```



Simple Banner Applet : An example

```
  t = null;
  }
  // Display the banner.
  public void paint(Graphics g) {
    g.drawString(msg, 50, 30);
  }
}
```

```
<html>
    <body>
        <applet width="300" height="50" code="SimpleBanner.class">
        </applet>
    </body>
</html>
```



Thank You