Java Applet

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.

Advantage of Applet

There are many advantages of applet. They are as follows:

It works at client side so less response time.

Secured

It can be executed by browsers running under many plateforms, including Linux, Windows, Mac Os etc.

Drawback of Applet

Plugin is required at client browser to execute applet.

Lifecycle of Java Applet

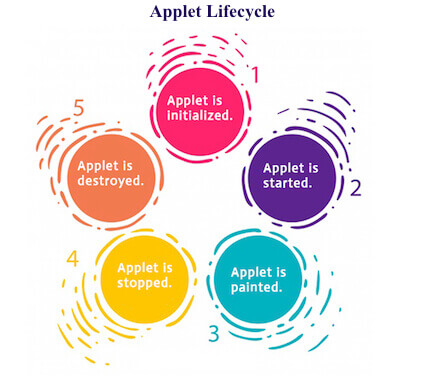
Applet is initialized.

Applet is started.

Applet is painted.

Applet is stopped.

Applet is destroyed.



For creating any applet java.applet.Applet class must be inherited. It provides 4 life cycle methods of applet.

**public void init():** is used to initialized the Applet. It is invoked only once.

**public void start():** is invoked after the init() method or browser is maximized. It is used to start the Applet.

**public void stop():** is used to stop the Applet. It is invoked when Applet is stop or browser is minimized.

**public void destroy():** is used to destroy the Applet. It is invoked only once.

java.awt.Component class

The Component class provides 1 life cycle method of applet.

**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.

Who is responsible to manage the life cycle of an applet?

Java Plug-in software.

How to run an Applet?

There are two ways to run an applet

By html file.

By appletViewer tool (for testing purpose).

//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);

}

}

#### Note: class must be public because its object is created by Java Plugin software that resides on the browser.

### myapplet.html

<html>

<body>

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

</applet>

</body>

</html>

**c:\>**javac First.java

**c:\>**appletviewer myaaplet.html

# Displaying Graphics in Applet

java.awt.Graphics class provides many methods for graphics programming.

## Commonly used methods of Graphics class:

**public abstract void drawString(String str, int x, int y):** is used to draw the specified string.

**public void drawRect(int x, int y, int width, int height):** draws a rectangle with the specified width and height.

**public abstract void fillRect(int x, int y, int width, int height):** is used to fill rectangle with the default color and specified width and height.

**public abstract void drawOval(int x, int y, int width, int height):** is used to draw oval with the specified width and height.

**public abstract void fillOval(int x, int y, int width, int height):** is used to fill oval with the default color and specified width and height.

**public abstract void drawLine(int x1, int y1, int x2, int y2):** is used to draw line between the points(x1, y1) and (x2, y2).

**public abstract booleandrawImage(Image img, int x, int y, ImageObserver observer):** is used draw the specified image.

**public abstract void drawArc(int x, int y, int width, int height, intstartAngle, intarcAngle):** is used draw a circular or elliptical arc.

**public abstract void fillArc(int x, int y, int width, int height, intstartAngle, intarcAngle):** is used to fill a circular or elliptical arc.

**public abstract void setColor(Color c):** is used to set the graphics current color to the specified color.

**public abstract void setFont(Font font):** is used to set the graphics current font to the specified font.

**import** java.applet.Applet;

**import** java.awt.\*;

**public** **class** GraphicsDemo **extends** Applet{

**public** **void** paint(Graphics g){

g.setColor(Color.red);

g.drawString("Welcome",50, 50);

g.drawLine(20,30,20,300);

g.drawRect(70,100,30,30);

g.fillRect(170,100,30,30);

g.drawOval(70,200,30,30);

g.setColor(Color.pink);

g.fillOval(170,200,30,30);

g.drawArc(90,150,30,30,30,270);

g.fillArc(270,150,30,30,0,180);

}

}

# Displaying Image in Applet

Applet is mostly used in games and animation. For this purpose image is required to be displayed. The java.awt.Graphics class provide a method drawImage() to display the image.

## Syntax of drawImage() method:

**public abstract booleandrawImage(Image img, int x, int y, ImageObserver observer):** is used draw the specified image.

## How to get the object of Image:

|  |
| --- |
| The java.applet.Applet class provides getImage() method that returns the object of Image. Syntax: |

**public** Image getImage(URL u, String image){}

## Other required methods of Applet class to display image:

|  |
| --- |
| **public URL getDocumentBase():** is used to return the URL of the document in which applet is embedded.  **public URL getCodeBase():** is used to return the base URL. |

**import** java.awt.\*;

**import** java.applet.\*;

**public** **class** DisplayImage **extends** Applet {

Image picture;

**public** **void** init() {

picture = getImage(getDocumentBase(),"sonoo.jpg");

}

**public** **void** paint(Graphics g) {

g.drawImage(picture, 30,30, **this**);

}

}

# Animation in Applet

|  |
| --- |
| Applet is mostly used in games and animation. For this purpose image is required to be moved. |

**import** java.awt.\*;

**import** java.applet.\*;

**public** **class** AnimationExample **extends** Applet {

Image picture;

**public** **void** init() {

picture =getImage(getDocumentBase(),"bike\_1.gif");

}

**public** **void** paint(Graphics g) {

**for**(**int** i=0;i<500;i++){

g.drawImage(picture, i,30, **this**);

**try**{Thread.sleep(100);}**catch**(Exception e){}

}

}

# EventHandling in Applet

## As we perform event handling in AWT or Swing, we can perform it in applet also. Let's see the simple example of event handling in applet that prints a message by click on the button.

## Example of EventHandling in applet:

**import** java.applet.\*;

**import** java.awt.\*;

**import** java.awt.event.\*;

**public** **class** EventApplet **extends** Applet **implements** ActionListener{

Button b;

TextField tf;

**public** **void** init(){

tf=**new** TextField();

tf.setBounds(30,40,150,20);

b=**new** Button("Click");

b.setBounds(80,150,60,50);

add(b);add(tf);

b.addActionListener(**this**);

setLayout(**null**);

}

**public** **void** actionPerformed(ActionEvent e){

tf.setText("Welcome");

}

}

# Parameter in Applet

We can get any information from the HTML file as a parameter. For this purpose, Applet class provides a method named getParameter(). Syntax:

**public** String getParameter(String parameterName)

**import** java.applet.Applet;

**import** java.awt.Graphics;

**public** **class** UseParam **extends** Applet{

**public** **void** paint(Graphics g){

String str=getParameter("msg");

g.drawString(str,50, 50);

}

}

<html>

<body>

<applet code="UseParam.class" width="300" height="300">

<param name="msg" value="Welcome to applet">

</applet>

</body>

</html>

# Applet Communication

java.applet.AppletContext class provides the facility of communication between applets. We provide the name of applet through the HTML file. It provides getApplet() method that returns the object of Applet. Syntax:

**pu**

**import** java.applet.\*;

**import** java.awt.\*;

**import** java.awt.event.\*;

**public** **class** ContextApplet **extends** Applet **implements** ActionListener{

Button b;

**public** **void** init(){

b=**new** Button("Click");

b.setBounds(50,50,60,50);

add(b);

b.addActionListener(**this**);

}

**public** **void** actionPerformed(ActionEvent e){

AppletContext ctx=getAppletContext();

Applet a=ctx.getApplet("app2");

a.setBackground(Color.yellow);

}

}

<html>

<body>

<applet code="ContextApplet.class" width="150" height="150" name="app1">

</applet>

<applet code="First.class" width="150" height="150" name="app2">

</applet>

</body>

</html>

# JApplet class in Applet

As we prefer Swing to AWT. Now we can use JApplet that can have all the controls of swing. The JApplet class extends the Applet class.

**import** java.applet.\*;

**import** javax.swing.\*;

**import** java.awt.event.\*;

**public** **class** EventJApplet **extends** JApplet **implements** ActionListener{

JButton b;

JTextField tf;

**public** **void** init(){

tf=**new** JTextField();

tf.setBounds(30,40,150,20);

b=**new** JButton("Click");

b.setBounds(80,150,70,40);

add(b);add(tf);

b.addActionListener(**this**);

setLayout(**null**);

}

**public** **void** actionPerformed(ActionEvent e){

tf.setText("Welcome");

}

}

# Painting in Applet

|  |
| --- |
| We can perform painting operation in applet by the mouseDragged() method of MouseMotionListener. |

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** java.applet.\*;

**public** **class** MouseDrag **extends** Applet **implements** MouseMotionListener{

**public** **void** init(){

addMouseMotionListener(**this**);

setBackground(Color.red);

}

**public** **void** mouseDragged(MouseEvent me){

Graphics g=getGraphics();

g.setColor(Color.white);

g.fillOval(me.getX(),me.getY(),5,5);

}

**public** **void** mouseMoved(MouseEvent me){}

}