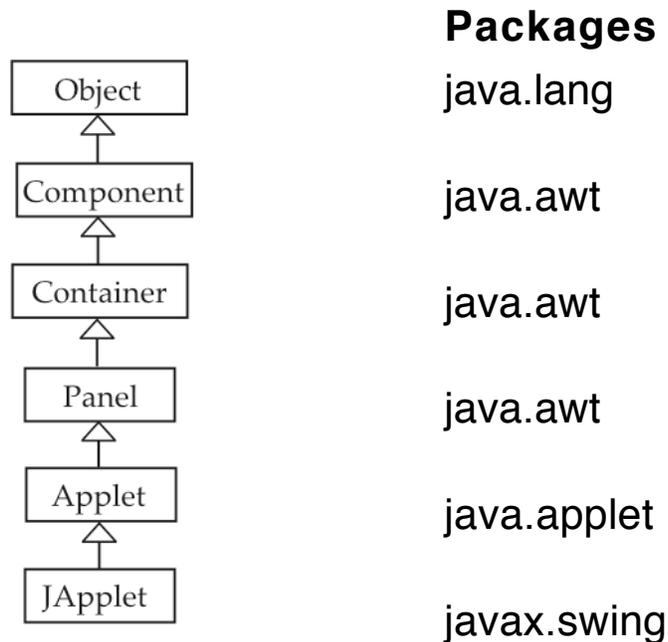


Applets

Applets are defined as subclasses of the class

- Applet in the package java.applet or
- JApplet in the package javax.swing.

Class Hierarchy



An applet object is instantiated and controlled by a web browser, such as Firefox, Safari, Netscape Navigator, Mozilla, or Internet Explorer, or by appletviewer, a program provided in the Java SDK.

Main Methods

- May be overridden to define behavior.
- Most are not called explicitly; a browser or appletviewer calls them.
- Includes methods that allows us to draw on the surface of the applet.

public void init()

- Called when the applet is first loaded.
- Put initialization code here (what you would normally do in a constructor in an application).

public void stop()

- Called when browser leaves the page containing the applet.
- Override to “suspend” threads running in the applet.

public void start()

- Called when browser visits or re-visits the page.
- Override to “resume” suspended threads.

public void destroy()

- Called when applet is unloaded.
- Override to free resources.
- Always called.

Two Ways to Draw on an Applet

Old Way

Override the *paint* method from Component to describe the surface of the applet.

public void paint(Graphics g)

- A Component method called when applet starts and whenever the applet needs to be redisplayed (forced by calling the method *repaint*).
- This method determines the appearance of the applet, which is a panel.

Note: *repaint* calls the method *update*, which paints the background color on the surface and then calls *paint*, passing the current graphics context object *g*.

New Way

Put a JPanel on the JApplet's contentPane and override *paintComponent* to draw on the panel.

public void paintComponent(Graphics g)

- A JComponent method called when applet starts and whenever the applet needs to be redisplayed (forced by calling the method *repaint*).
- This method determines the appearance of the panel on the applet.

Next two methods *are* called explicitly

public void showStatus(String message)

- Print a message at bottom of applet in the status window.
- Useful for debugging.

public boolean isActive()

- Tells whether applet is currently running.

HTML: Applet Tags

html = HyperText Markup Language

html (xhtml) Code

```
<object classid="java:DCS" data="DCS.class"
        codetype="application/java"
        width="680" height="280">
    Your browser cannot handle a Swing applet.
</object>
```

Other Attributes Inside applet Tag

```
codebase="DCSwing"
vspace, hspace, align, name
```

Parameters: Between <object> and </object>

```
<param name="first" value="cat and the hat"/>
<param name="second" value="222"/>
<param name="color" value="blue"/>
```

Method in Applet

```
public String getParameter(String name)
```

Example: A Digital Clock

This applet acts as a digital clock, showing the current time with the format *hh:mm:ss*. The applet is a thread that sleeps for a second and then reports the time obtained from a Calendar object.

When the web page containing the applet is replaced by another page, the *stop* method is called and it sets a variable to **null** so that the thread ends its run method and dies.

The *start* method creates a new thread and makes it runnable.

```
import java.awt.*;
import javax.swing.*;
import java.util.Calendar;

public class DigitalClock extends JApplet implements Runnable
{
    private Thread clockThread = null;
    private Font font = new Font("Monospaced", Font.BOLD, 132);
    private Color color = Color.green;
    private ClockPanel clockPanel;

    public void init()
    {
        clockPanel = new ClockPanel();
        getContentPane().add(clockPanel);
        clockPanel.setBackground(Color.cyan);
        String param = getParameter("color");
```

```
    if ("red".equals(param)) color = Color.red;
    else if ("blue".equals(param)) color = Color.blue;
    else if ("yellow".equals(param)) color = Color.yellow;
    else if ("orange".equals(param)) color = Color.orange;
    else color = Color.green;
}
```

```
public void start()
{
    if (clockThread == null)
    {
        clockThread = new Thread(this);
        clockThread.start();
    }
}
```

```
public void stop()
{
    clockThread = null;
}
```

```
public void run()
{
    while (Thread.currentThread() == clockThread)
    {
        clockPanel.repaint();
        try
        { Thread.sleep(1000);
        }
        catch (InterruptedException e) { break; }
    }
}
```

```

class ClockPanel extends JPanel
{
    public void paintComponent(Graphics g)
    {
        super.paintComponent(g);
        Calendar calendar = Calendar.getInstance();
        // calendar is an instance of java.util.GregorianCalendar
        int hour = calendar.get(Calendar.HOUR_OF_DAY);
        int minute = calendar.get(Calendar.MINUTE);
        int second = calendar.get(Calendar.SECOND);
        g.setFont(font);
        g.setColor(color);
        String time = hour + ":" + minute/10 + minute%10 +
                    ":" + second/10 + second%10;
        g.drawString(time, 15, 160);
        showStatus(time);
    }
}

```

When the stop method of DigitalClock is called, it changes the instance variable *clockThread* to **null** so that when the **while** loop in the run method tests

Thread.currentThread() == clockThread,
the loop is completed and the run method terminates putting the thread into the “dead” state.

HTML for DigitalClock

Source DigitalClock.java

HTML Code clock.html

```
<?xml version="1.0" encoding="UTF-8"?>
<html>                    <!-- clock.html -->
  <head>
    <title> Digital Clock Applet </title>
  </head>
  <body style="background-color:cyan">
    <h1> The Swing Digital Clock Applet</h1>
    <p>
      <object classid="java:DigitalClock "
              data="DigitalClock.class"
              codebase="DigitalClock"
              codetype="application/java"
              width="680" height="280">
        Your browser cannot handle a Swing applet.
      <param name="color" value="blue"/>
    </object>
    </p>
  </body>
</html>
```

Put *clock.html* and a directory DigitalClock in the same directory and place DigitalClock.class in the folder DigitalClock.

Execute applet in the Java SDK using

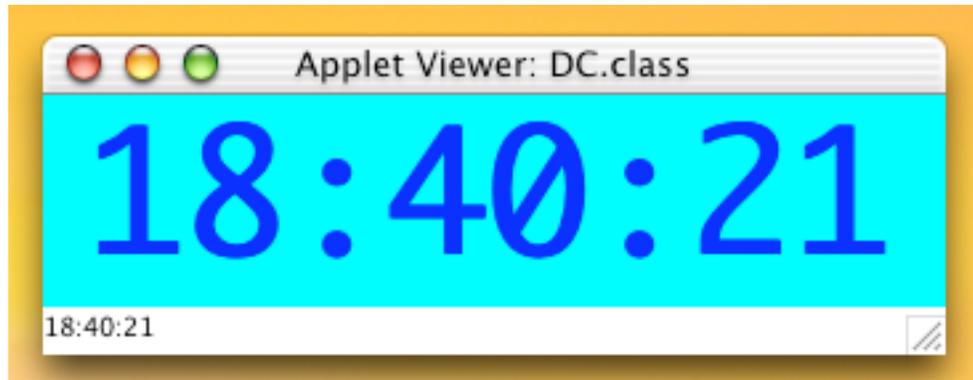
```
% appletviewer clock.html
```

The HTML code may have to be altered for the applet viewer.

Alternatively

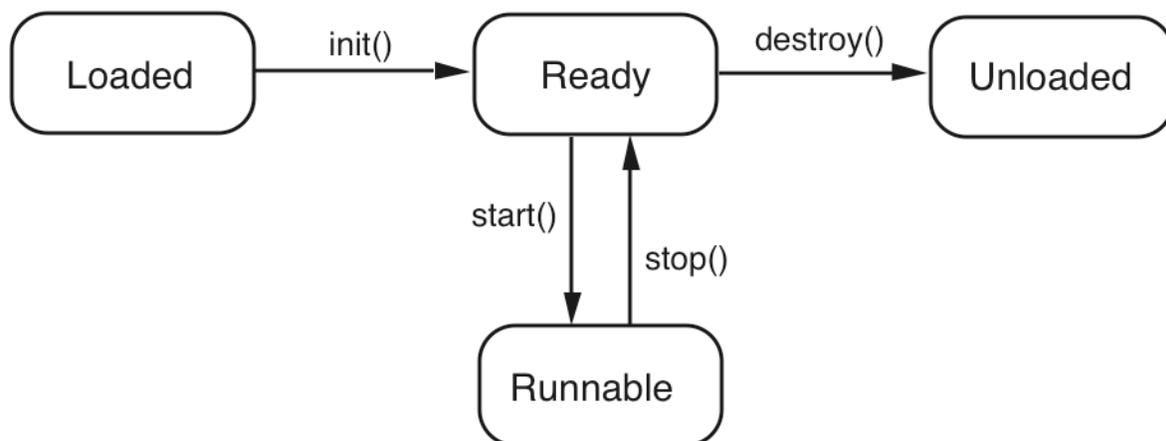
You may open the html file using a web browser such as Opera, Internet Explorer, or Mozilla.

Snapshot



Note the time printed by `showStatus` at the bottom of window.

Applet Lifecycle



Security Restrictions

Applets are forbidden certain capabilities, depending their origin and what software is executing them.

| Action | Browser from Net | Browser from local disk | Applet-viewer | Application |
|-----------------------|-------------------------|--------------------------------|----------------------|--------------------|
| Read local file | No | No | Yes | Yes |
| Write local file | No | No | Yes | Yes |
| Delete file | No | No | No | Yes |
| Read user.name | No | Yes | Yes | Yes |
| Connect to server | Yes | Yes | Yes | Yes |
| Connect to other host | No | Yes | Yes | Yes |
| Load Java library | No | Yes | Yes | Yes |
| Call exit | No | No | Yes | Yes |
| Create pop-up window | Yes | Yes | Yes | Yes |