**Applet**

An **applet** is a Java program that runs in a Web browser. It is embedded in the webpage to generate the dynamic content and works at client side. An applet can be a fully functional Java application because it has the entire Java API at its disposal.

Java applets are classes written in java that are not intended to run as stand-alone programs (like applications) but as subprograms of a browser that is already managing a window

Applets are not trusted by default, so they have several restrictions in running on the client machine

Applets should not have a main() method. Instead they have inti(), start(), paint() etc. for displaying on the browser window

-no printing or file I/O

-cannot connect through the network to any machine but its own server

-any new windows created by the applet have warning label

**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.

There are some important differences between an applet and a standalone Java application, including the following −

* An applet is a Java class that extends the java.applet.Applet class.
* A main() method is not invoked on an applet, and an applet class will not define main().
* Applets are designed to be embedded within an HTML page.
* When a user views an HTML page that contains an applet, the code for the applet is downloaded to the user's machine.
* A JVM is required to view an applet. The JVM can be either a plug-in of the Web browser or a separate runtime environment.
* The JVM on the user's machine creates an instance of the applet class and invokes various methods during the applet's lifetime.
* Applets have strict security rules that are enforced by the Web browser. The security of an applet is often referred to as sandbox security, comparing the applet to a child playing in a sandbox with various rules that must be followed.
* Other classes that the applet needs can be downloaded in a single Java Archive (JAR) file.

**Life Cycle of an Applet**

Four methods in the Applet class gives you the framework on which you build any serious applet

* **init** − This method is intended for whatever initialization is needed for the applet. It is called after the param tags inside the applet tag have been processed.
* **start** − This method is automatically called after the browser calls the init method. It is also called whenever the user returns to the page containing the applet after having gone off to other pages.
* **stop** − This method is automatically called when the user moves off the page on which the applet sits. It can, therefore, be called repeatedly in the same applet.
* **destroy** − This method is only called when the browser shuts down normally. Because applets are meant to live on an HTML page, you should not normally leave resources behind after a user leaves the page that contains the applet.
* **paint** − Invoked immediately after the start() method, and also any time the applet needs to repaint itself in the browser. The paint() method is actually inherited from the java.awt.

Simple Applet

import java.applet.\*;

import java.awt.\*;

public class HelloWorldApplet extends Applet {

public void paint (Graphics g) {

g.drawString ("Hello World", 25, 50);

}

}

These import statements bring the classes into the scope of our applet class

* java.applet.Applet
* java.awt.Graphics

**The Applet Class**

Every applet is an extension of the java.applet.Applet class. The base Applet class provides methods that a derived Applet class may call to obtain information and services from the browser context.

**Invoking an Applet**

An applet may be invoked by embedding directives in an HTML file and viewing the file through an applet viewer or Java-enabled browser.

The **<applet>** tag is the basis for embedding an applet in an HTML file. Following is an example that invokes the "Hello, World" applet –

<html>

<title>The Hello, World Applet</title>

<hr>

<applet code = "HelloWorldApplet.class" width = "320" height = "120">

If your browser was Java-enabled, a "Hello, World"

message would appear here.

</applet>

<hr>

</html>

**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(). The values may be passed for the parameters by adding <param> tags between <applet> and </applet>. The browser ignores text and other tags between the applet tags.

Syntax:

public String getParameter(String parameterName)

**Example of using parameter in Applet:**

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);

}

}

**myapplet.html**

<html>

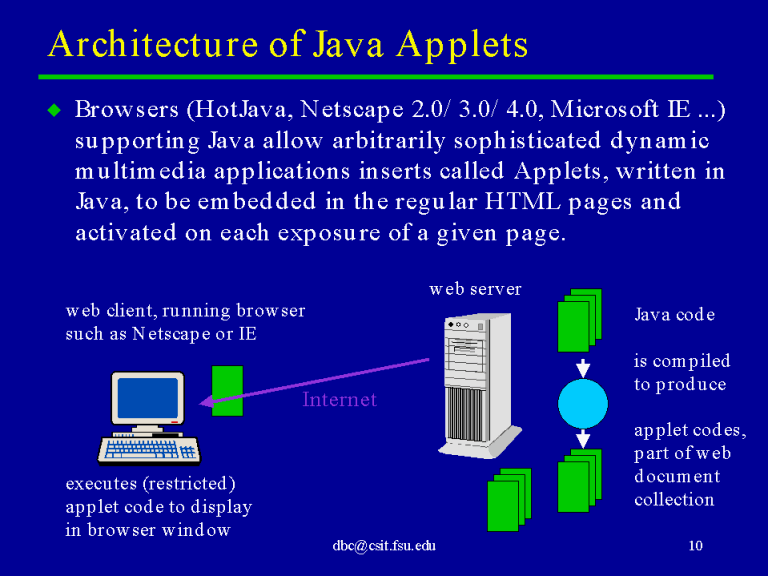
<body>

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

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

</applet> </body>

</html>



**Event driven**

* + An applet waits until an event occurs.
  + The AWT notifies the applet about an event by calling event handler that has been provided by the applet, The applet takes appropriate action and then quickly return control to AWT.
  + All Swing components descend from the AWT Container class

**User initiates interaction with an Applet**

These interactions are sent to the applet as events to which the applet must respond

**Example**: when user click a mouse inside the applet's window, a mouse-click event is generated.

If user presses a key in the applet's window has input focus, a key press event is generated.

When the user interacts with one of these controls (buttons and check boxes) , an event is generated.

**Applet Security Policies**

**Policy-based**

Security policy limits the resources a program can use

java.policy

**Permissions**

Actions that are allowed

**Applet Security Basics**

A Java applet downloaded from the Web runs in either a Java-enabled Web browser or a Java appletviewer. From a security standpoint, Java applets downloaded from the Internet or from any remote sources are restricted from reading and writing files and making network connections on client host systems. They are also restricted from starting other programs, loading libraries, or making native calls on the client host system.

Applets loaded over the network are usually considered to be untrusted code.

In general, applets downloaded from a network or remote sources are considered untrusted. An applet can be considered trusted, based on the following factors:

1. Applets installed on a local filesystem or executed on a localhost.
2. Signed applets provide a way to verify that the applet is downloaded from a reliable source and can be trusted to run with the permissions granted in the policy file.

The only way to be sure that an untrusted applet cannot perform any malicious actions is to run it in a tightly controlled environment.

* All unsigned applets are run under the standard applet security model.
* If usePolicy is not defined in the java.policy file, then a signed applet has the AllPermission permission only if Java Plug-in can verify the signers, and the user agrees to granting the AllPermission permission when prompted.
* If usePolicy is defined, then a signed applet has only the permissions defined in java.policy and no prompting occurs.