Unit -1
Java Applets

Topics to be covered…

1 .1 Concept of Applet Programming

- Local and Remote applets
- Difference between applet and application
- Preparing to write applets
- Building applet code
- Applet life cycle
- Creating an executable applet

1.2 Designing a web page:

- Applet tag
- Adding applet to HTML file
- Running the applet
- Passing parameter to applet

1.1 Concept of Applet Programming

- Java supports two types of programming :

1) Application program:

application is a program that runs on your computer under the operating system of that Computer.

2) Applet program:

An **applet** is an application designed to be transmitted over the Internet and executed by a Java-compatible Web browser.

APPLET

- **Applet** is a small program that are primarily used in internet computing, they can be transported over the internet from one computer to another and run using applet viewer or java compatible web browser.
- Java applet is a java class that you embed in an HTML page and is downloaded and executed by a web browser.
- Applet can’t be executed directly .
- For running an applet ,HTML file must be created which tells the browser what to load and how to run it.

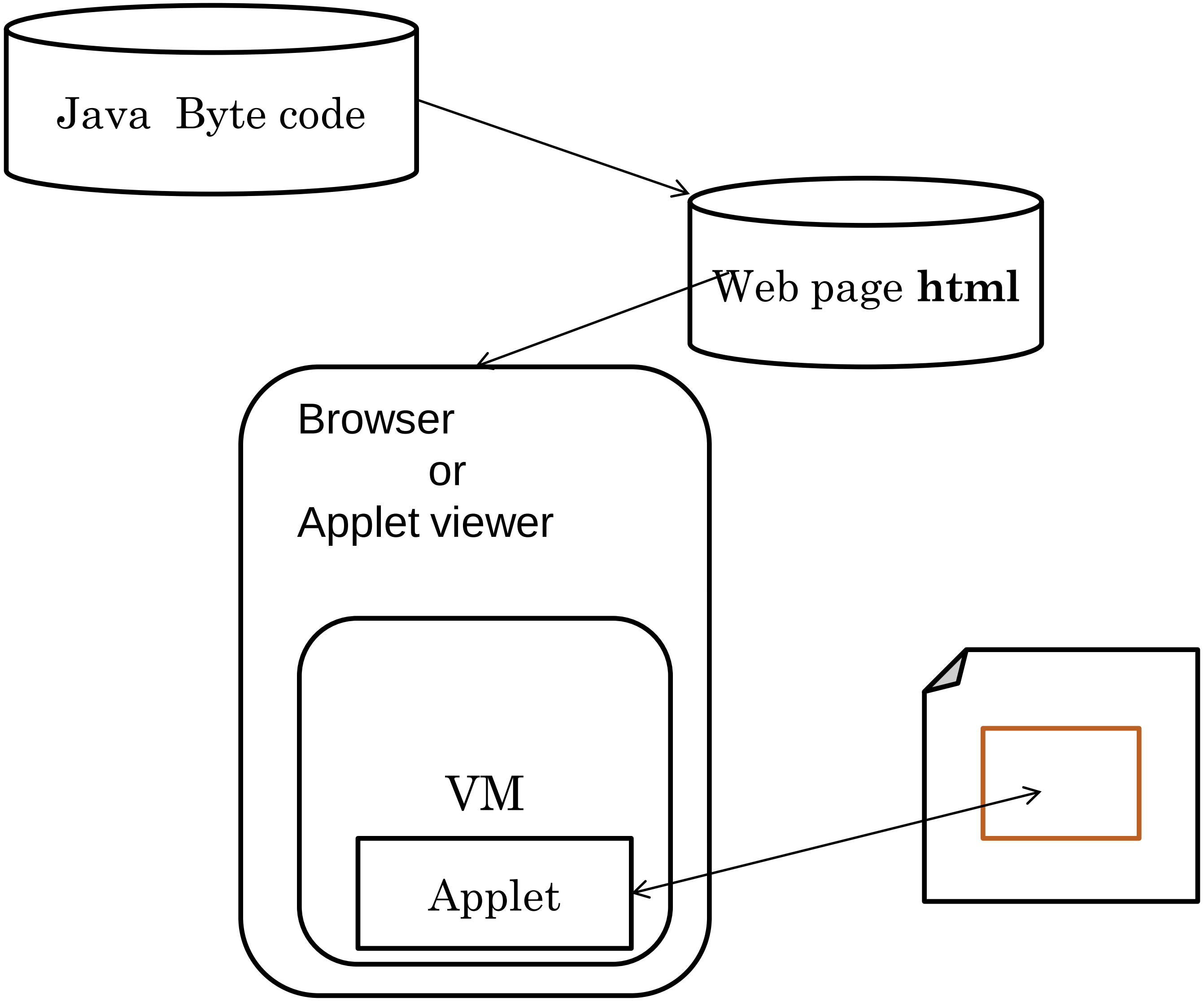
Applet

- Applet begins execution via loading of a HTML page "containing it" after that java enabled web browser or "applet viewer" is required to run an applet
- Now ,Web pages not only contain static text or simple image but it can also perform arithmetic operation ,displays graphics ,play sounds and moving Images.
- We can embed applets into web pages in two ways:

1) We can write our own applets and embed them into web pages.

2) We can download an applet from a remote computer system and then embed it into a web page.

LOADING OR DISPLAYING APPLET



LOCAL APPLET

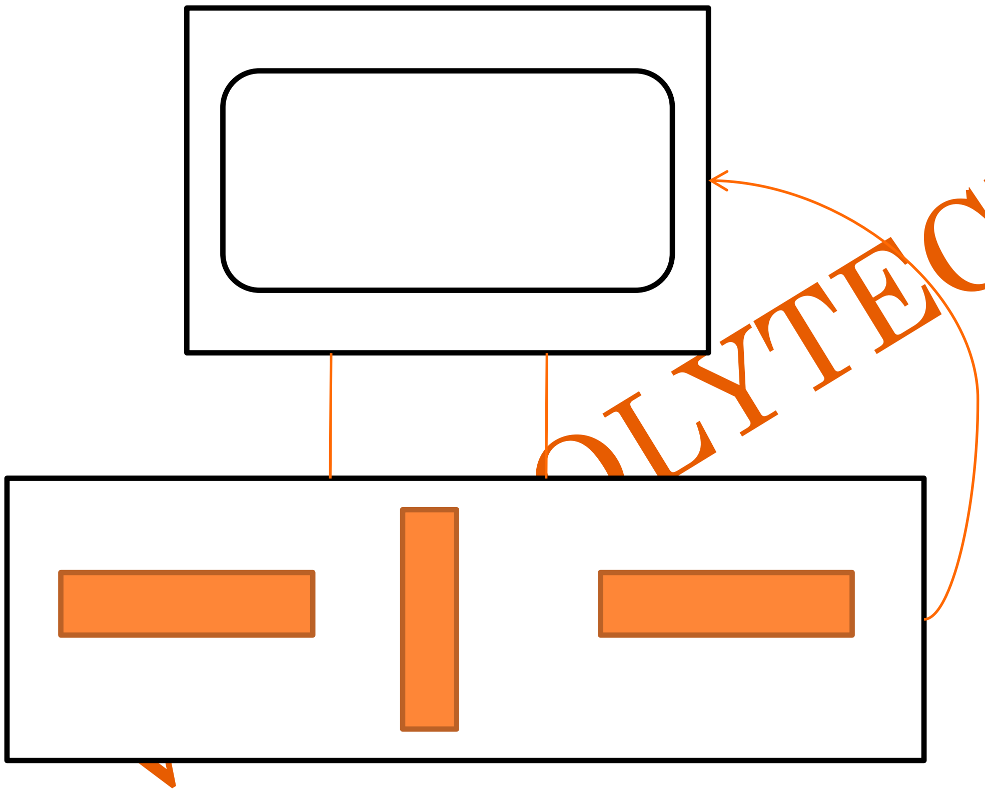
- An applet developed locally and stored in a local system is known as local applet.

- When a web page is trying to find a local applet, it does not need to use the Internet and therefore the local system does not require the Internet Connection.

- It simply searches the directories in the local system and locates and loads the specified applet.

- User can write his own applets and embed them into web pages.

Local applet



Remote applet

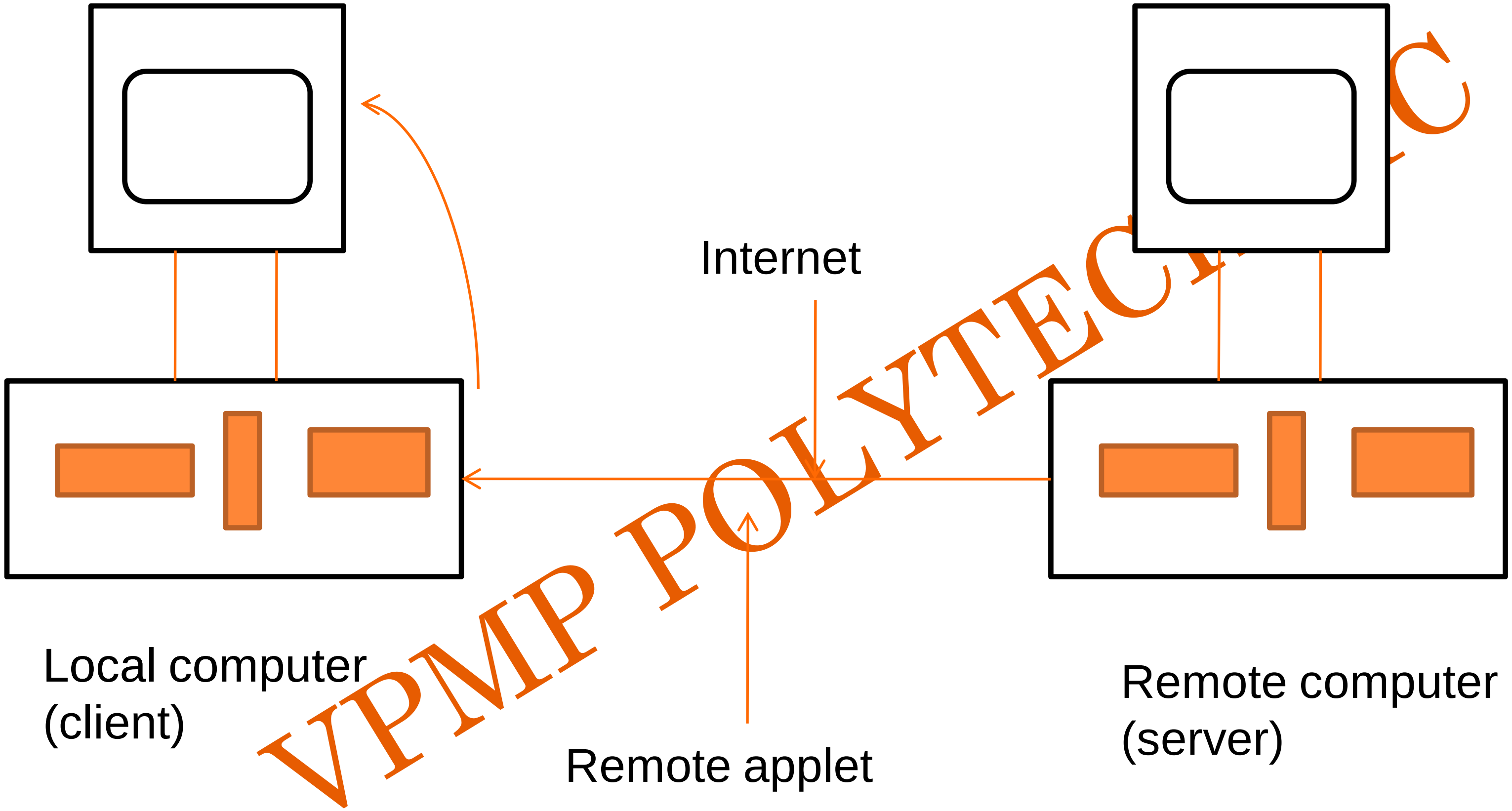
- An applet which is developed by someone else and stored on a remote computer connected to the Internet is known as remote Applet.
- If our system is connected to the Internet, we can download the remote applet onto our system via Internet and then run it.
- User can thus download an applet from a remote computer system.
- In order to locate and load a remote applet ,we must know the applet's address on web. This address is known as **URL(Uniform Resource Locator).**
- **URL** and must be specified in the applet's HTML document as the value of the **CODEBASE** attribute.

REMOTE APPLET

CODEBASE= http: // www. Netserve.com /applets

- In case of local applets ,CODEBSAE may be absent or may specify a local directory.

REMOTE APPLET



Loading a remote applet

LIMITATIONS OF APPLET

- All the restriction and limitations are placed in the interest of security of systems. These ensure that an applet can not do any damage to local system.
- Applets allow neither to execute any application nor to load any DLL s on the local system.
- Applet do not need a main method.
- Applet runs under an applet viewer or a java compatible web browser.
- Applets are restricted from using libraries from other languages such as c ,c++.
- Applets cant read or write files on the web user's disk. If information must be saved to disk as an applet is executing ,the storage of information must be done on the disk from which the web page is served.

LIMITATIONS OF APPLET

- Applet can not make network connection to a computer other than the one from which the web page is served, expect to direct the browser to a new location.
- Applet can not run any programs on the web user's system, including browser plug-in, ActiveX controls or other browser related items.
- Some of Java's functionality (like removal of pointers ,verification of byte code and restricted remote and local file access) blocked for applets because of security concerns

When to use Applet?

- When we need something dynamic to be included in the display of a web page.
- When we require some flash outputs. for example ,applets that produce sounds, animations or some special effects would be used when displaying certain pages.
- When we want to create a program and make it available on the internet for us by others on their computers.

Difference : Applet and application

Applet

- Applet is dynamic program which is run on browser .
- Applet do not use the main() method for initiating the execution of the code.
- Applets can not be run independently. They must be run under an applet viewer or a java compatible web browser.

Application

- Application is static program which run using java interpreter.
- Application uses main() method for execution of the code.
- Application runs independently using javac compiler.

DIFFERENCE : APPLET AND APPLICATION

Applet	Application
Applets can not read or write files on the web user's disk.	Application can read write files on the web user's disk.
Applet can not communicate with other server on the network.	Application can communicate with other servers on the network.
Applet can not run any program from the local computer.	Application program can run from local computer.
Applets are restricted from using libraries from other languages such as c,c++.	Application program can use methods of c, c++ libraries.

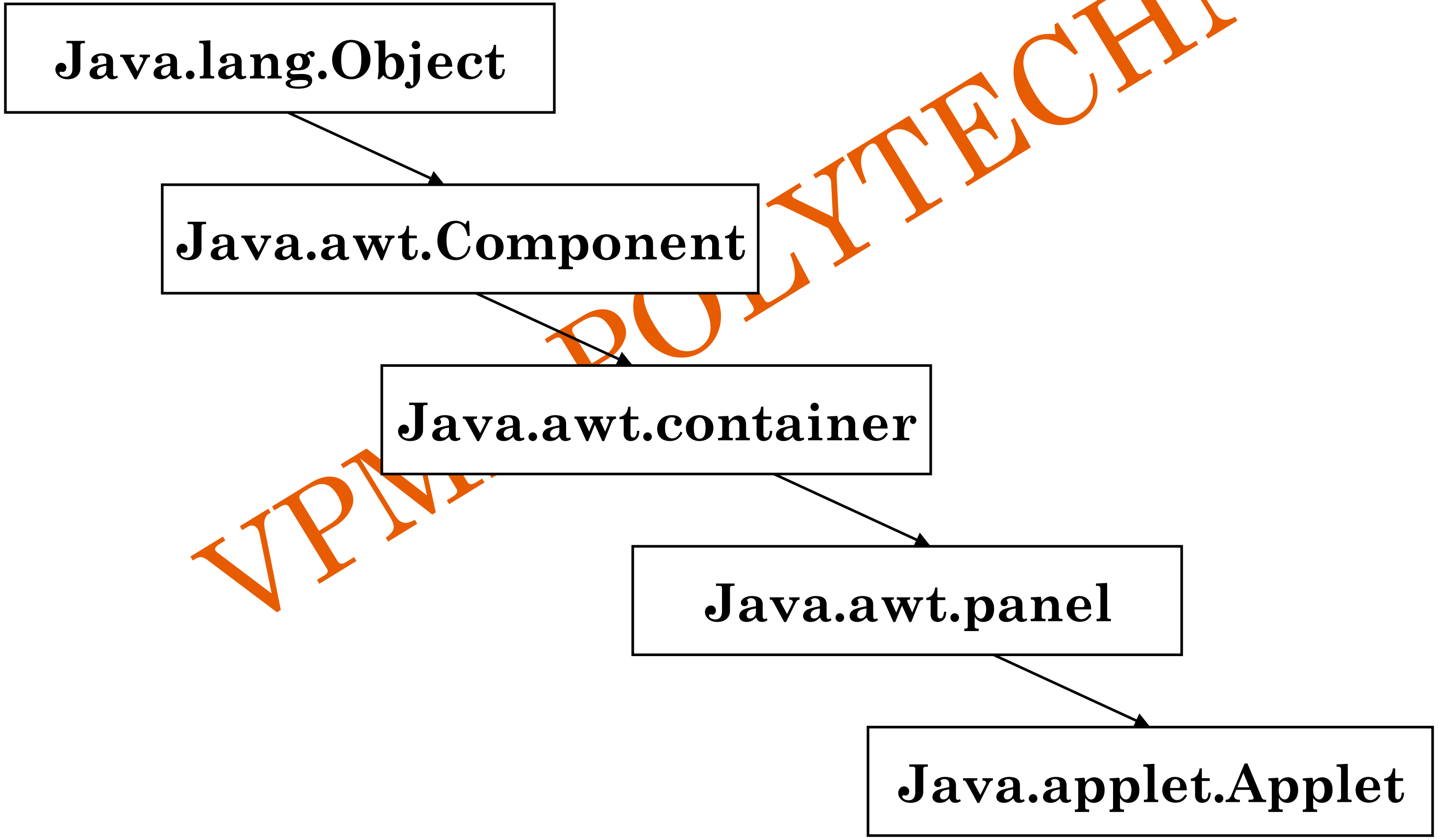
Applet class

- **Java.applet.Applet** class is actually a subclass of **java.awt.panel.**
- Java. applet is **smallest** package in Java API.
- It consists of **a single class (applet)** and **three** interfaces: **AppletContext ,AppletStub** and **Audioclip**.
- Applet class contains only a single default parameter less constructor ,which is not used generally.
- Applets are constructed by the run time environment when they are loaded ,they do not have to be explicitly constructed.

APPLET CLASS

- **Java.applet.Applet** class is actually a subclass of **java.awt.panel.**
- Java. applet is **smallest** package in Java API.
- It consists of **a single class (applet)** and **three** interfaces: **AppletContext ,AppletStub** and **Audioclip**.
- Applet class contains only a single default parameter less constructor ,which is not used generally.
- Applets are constructed by the run time environment when they are loaded ,they do not have to be explicitly constructed.

- Applet class contains 21 methods that are used to display images, play audio files ,respond to events and obtain information about applet’s execution environment ,referred as applet’s context.



Java.lang.Object

Java.awt.Component

Java.awt.container

Java.awt.panel

Java.applet.Applet

Method	Description
Image getImage(URL url):	Used to retrieve an Image identified by URL.
AudioClip getAudioClip(URL url) :	Used to retrieve Audioclip object that is identified by URL.
void play(URL url)	If an audio clip is found then play method is used to play an audio clip.
void init(), void start(), void stop(), void destroy()	Used to implement each of four life cycles stages of an applet.
boolean isActive() :	Return **true if** applet has been started else return **false** if it has been stopped

Method	Description
AppletContext getAppletContext() :	Used to obtain AppletContextObject associated with an applet.
String getAppletInfo():	Returns a string object that provides information about applet. this includes version, copyright and ownership data as well as applet specific data.
URL getCodeBase() :	Returns base URL specifying the applet's location.
URL getDocumentBase():	Returns the URL of the HTML document in which the applet is contained.
void setStub(AppletStub *stubObj*):	Used to set the AppletStub associated with the applet. It should not be used unless you are constructing your own custom applet viewer.

Method	Description
String getParameter(String paramname):	Used to obtain parameter data that is passed to an applet in an HTML file.returns **null** if parameters not found.
String [ ] [ ] getParameterInfo( )	Returns array that describe all the parameters used by an object
void resize(Dimension *dim*):	Used to resize an applet according to the dimensions specified by *dim*.
void showStatus(String *str*) :	Used to display a status message in the status window of the browser or appletviewer.

Simple Applet display Method

Method	Description
void drawString (String ***message*****,int x,int y)** ***e.g. drawString("hi",10,100)***	To output a string to an applet.
void setBackground(Color *colorname*) ***e.g. setBackground(Color.red)***	To set the back ground of a applet window
void setForeground(Color *colorname*) ***e.g. setForeground(Color.pink)***	To set the foreground color of an applet window.
Color getBackground()	To obtain the current settings for the background color.

Simple Applet display Method

Method	Description
Color getForeground()	To obtain the current settings for the foreground.
Applet getApplet(String *name*)	To obtain the applet specified by given name from the current applet context.
void showStatus(String status)	To dispaly the status message in the status bar of applet window.
URL getDocumentBase()	To obtain the directory of the current browsers page.
URL getCodeBase()	To obtain the directory from which the applet's class file was loaded.

Pre-defined color in Java

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

- For example this sets background color to green and text color to red:

  setBackground(Color.yellow);

  setForeground(Color.red);

java

Color constructor:

Color obj= new **Color (**int **red** ,int **green,** int **blue);**

E.g:

Color c=new **Color**(255,100,100);

g.**setColor**(c);

Color Obtained	**Red Value**	**Green Value**	**Blue Value**
white	**255**	**255**	**255**
Black	0	0	0
Lightgray	**192**	**192**	**192**
darkgray	**128**	**128**	**128**
Red	**255**	0	0
Green	0	**255**	0
blue	0	0	**255**
Yellow	**255**	**255**	0
purple	**255**	0	**255**

Steps to developing and testing an applet:

1) Building an applet code(.java file)
2) Creating an executable applet (.class file)
3) Designing a web page using HTML tags
4) Preparing <APPLET> tag
5) Incorporating <APPLET> tag into the web page
6) Creating HTML file
7) Testing the applet code

HTML TAGS

<HTML>...........</HTML>

<HEAD>...........</HEAD>

<TITLE>...........</TITLE>

<BODY>...........</BODY>

<H1>........</H1>.......<H6>.........</H6>

<CENTER>...........</CENTER>

<APPLET...>

<APPLET...>...........</APPLET>

HTML TAGS

<PARAM….>

<B>……….</B>

<BR>

<P>

<IMG……….>

<HR>

<A…..></A>

<FONT>……….</FONT>

<!………………….>

BUILDING APPLET CODE

- Applet code uses the services of two classes, namely **Applet** and **Graphics** from java class.
- Applet class is contained in **java.applet** package ,which provides life and behavior to the applet through its methods such as **init() ,start()** and **paint().**
- Applet class maintains the *life cycle* of an applet.
- When an applet is loaded ,java automatically calls a series of Applet class methods for the starting,running,stopping the applet code.
- When **paint()** method is called ,it will actually display the result of the applet code on the screen.The output may be text,graphics or sound.

BUILDING APPLET CODE

- The **paint()** method requires a **Graphics** object as an argument.

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

- We have to import java.awt package that contains **Graphics** class.
- All output operation of an applet are performed using the methods defined in the **Graphics** class.
- The *appletclassname* is the main class for the applet. When the applet is loaded ,java creates an instance of this class and then create Applet class methods are called on that instance to execute the code.
- *appletclassname* should be declared **public** because it is main applet class.

General format of applet code:

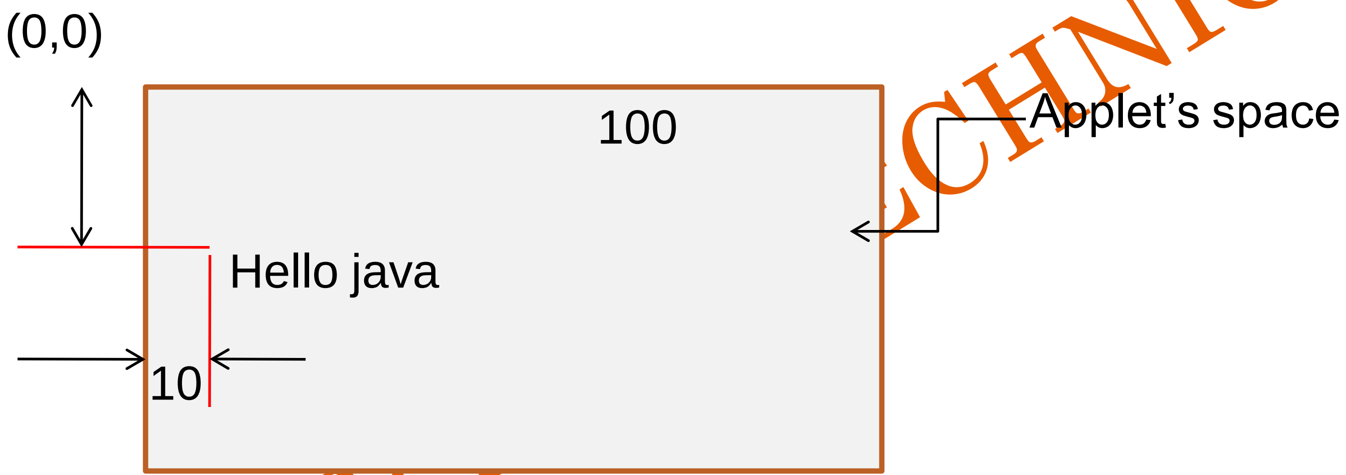
```
import  java.awt.*;
import  java.applet.*;
………
………
public  class appletclassname extends Applet
{
   …………
   …………
     public void paint(Graphics g)
   {
          //Applet operation code
   }
    …………
     …………
}
```

General format of applet code:

```
//save file with name Hellojava.class in a java
   subdirectory


import  java.awt.*;
import  java.applet.*;
public class Hellojava extends Applet
{
public void paint(Graphics g)
   {
          g.drawString( “Hello java” ,10,100);
   }
}
```

OUTPUT:



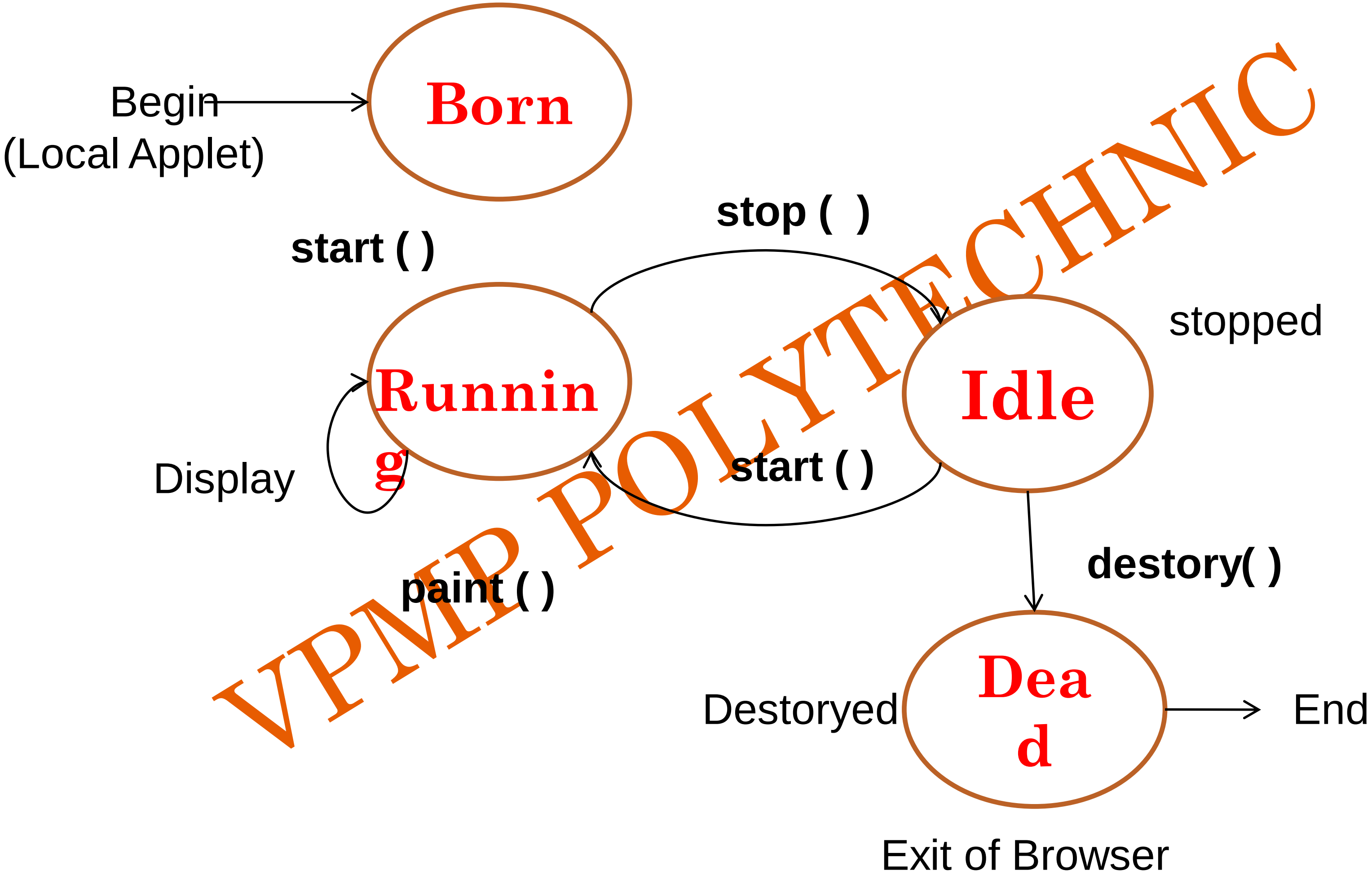
Applet life cycle

- Every java applet inherits a set of default behaviours from the Applet class.so when an applet is loaded ,it undergoes a series of changes in its state.

- **The applet states are:**

1) Born or Initialization state
2) Running state
3) Idle state
4) Dead or destroyed state

Applet life cycle



INITIALIZATION STATE

- Applet enters the initialization state when it is first loaded. This is achieved by calling the init() method of Applet class. The applet is born.
- Initialization state occurs only once in the applet's life cycle.
- We do following at this stage,

  - create objects needed by the applet

  - set up initial values,initailize variables

  - Load images or fonts

  - set up colors

Initialization state

- To provide any of behaviour mentioned above, we must override **init() method:**

```
public void init()
        {
        ............(Action)
        }
```

Running state

- Applet enters the ***running*** state when the system calls **start( )** method of Applet class.
- This occurs automatically after applet is initialized.
- Starting of applet can also occur if the applet is already in "stopped" (idle) state.
- Unlike **init() method,** the **start( )** method may be called more than once.
- We may override the **start( )** method to create a thread to control the applet.

```
public void start( )
{
............(Action)
............
}
```

Idle or Stopped State

- An applet becomes *idle* when it is stopped from running.
- Stopping occurs automatically when we leave the page containing the currently running applet.
- We can also stop applet by calling the **stop( )** method explicitly.
- If we use thread to run the applet then we must use **stop( )** method to terminate the thread.
- To do this override **stop() method:**

```
public void stop()
{
..................(Action)
.................. }
```

DEAD STATE

- An applet is said to be dead when it is removed from memory .this occurs automatically by invoking the **destroy()** method when we quit the browser.
- Destroying stage occurs only once in the applet's life cycle.
- If the applet has created any resources, like threads we may override **destroy( )** method to clean up these resources .

```
public void destroy()
{
...............
................
}
```

Display state

- Applets moves to the display state ,whenever it has to perform some output operations on the screen.
- This happens immediately after the applet enters into the running state.
- The **paint()** method is called to accomplish this task.
- Almost every applet will have **paint()** method .default version of **paint()** method does nothing.
- We must have to override **paint()** if we want anything to be displayed on the screen.
- Display state is not considered as a part of the applet's life cycle.

Display state

- The **paint()** method is defined in the applet class.it is inherited from the **Component** class ,a super class of Applet.

```
public void paint()
{
………….(Display statements)
………….
}
```

1.2 Designing web page using HTML tag

Designing web page using HTML tag

- Java programs resides on web pages.to run a java applet ,it is necessary to have a web page that reference that applet.
- A web page is made up of text and HTML tags that can be interpreted by a web browser or applet viewer.
- A web page is also known as *HTML page* or *HTML document.*
- Web pages are stored using a file extension **.html**
- HTML files should be stored in the same directory as the compiled code of the applets.

Creating an executable applet

- Executable applet is **.class** file of the applet which is obtained by compiling the source code of the applet.
- Compiling an applet is same as compiling an application.

  Ex. Hellojava.java

1) Move to directory containing the source code and type command:

   javac *Hellojava*.**java**

2) The compiled output file called Hellojava.class is placed in same directory as the source .
3) If any error message is received ,then we must check for errors ,correct themand compile the applet again.

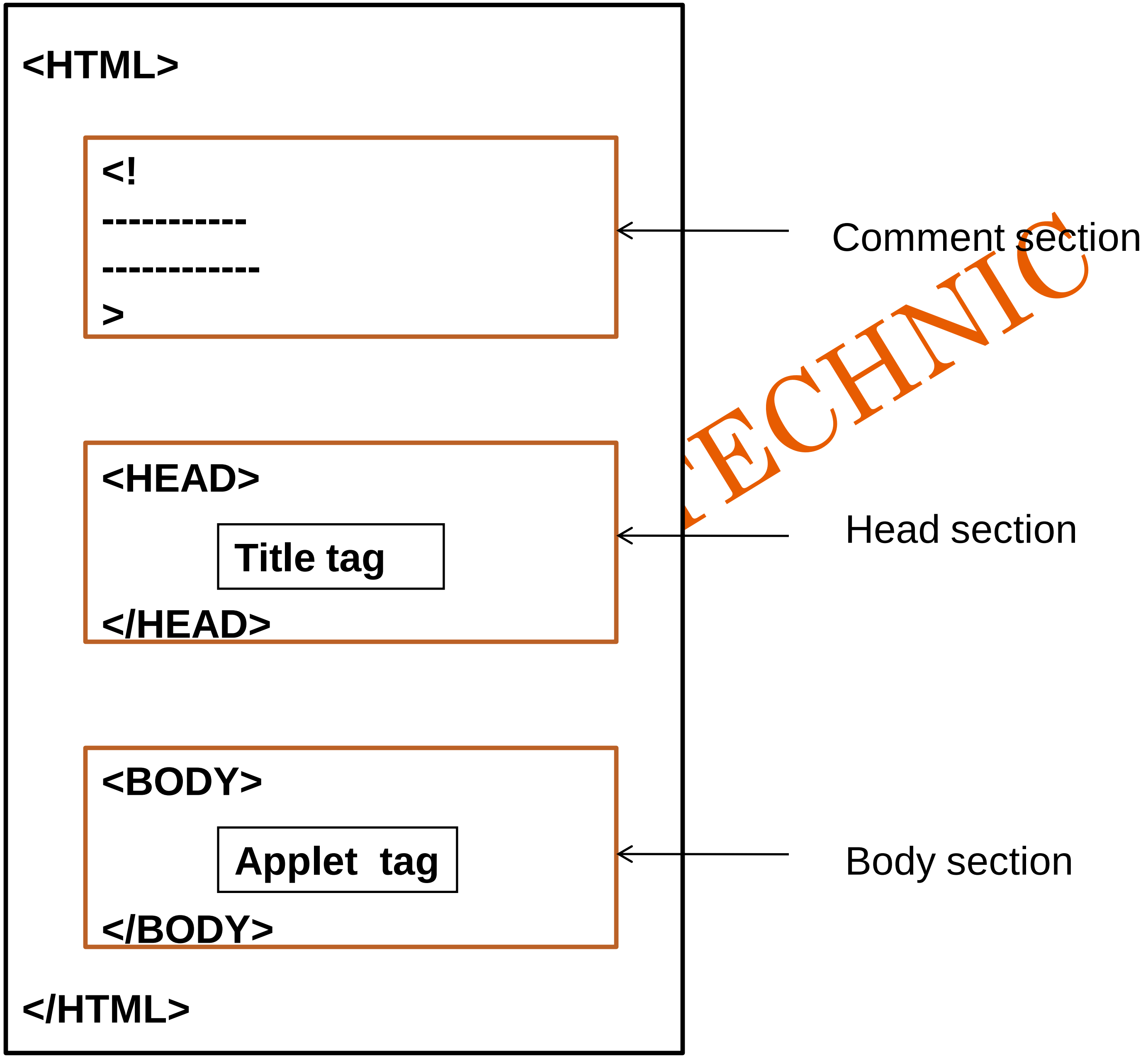
Creating an executable applet

- Web page is divided in to three major sections:

1) Comment section(optional)

2) Head Section(optional)

3) Body section



<HTML>

<!

>

<HEAD>

Title tag

</HEAD>

<BODY>

Applet tag

</BODY>

</HTML>

Comment section

Head section

Body section

HEAD SECTION:

<HEAD>

<TITLE> Welcome to Java Applets **</TITLE>**

</HEAD>

Here, text enclosed in <TITLE> and </TITLE> will appear in the title bar of the web browser when it displays the page.

Body Section:

After Head section comes the body section. This section contains the entire information about the web page and its behavior.

```
<BODY>
        <CENTER>
                <H1> WELCOME </H1>
        </CENTER>
        <APPLET .......>


        </APPLET>
</BODY>
```

APPLET TAG

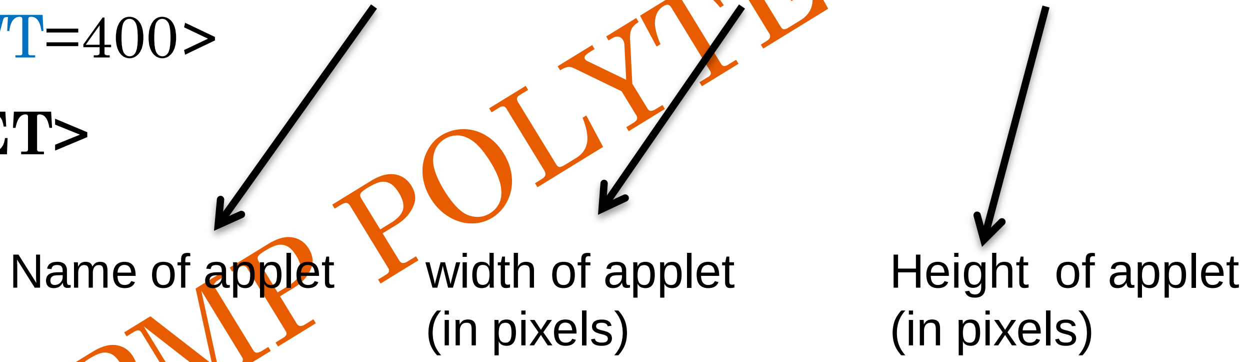
- **<APPLET ….>** tag supplies the name of applet to be loaded and tells the browser how much space the applet requires.
- The ellipsis in the tag <APPLET ….> indicates that it contains certain attributes that must be specified.
- This HMTL code tells the browser to load the compiled java applet Hellojava.class ,which is in the same directory as this HTML file .and also specify display area for the applet output.

APPLET TAG

Minimum requirement of <APPLET> TAG:

<APPLET *CODE*=Hellojava.class *WIDTH*=200
*HEIGH*T=400>

</APPLET>



Name of applet

width of applet
(in pixels)

Height of applet
(in pixels)

Attributes of Applet tag

```
<APPLET
        [ CODEBASE= codebase_URL]
                CODE=AppletfileName.class
        [ALT = Alternate_text  ]
        [Name = applet_instance_name ]
                WIDTH = Pixels
                HEIGHT = Pixels
        [ ALIGN= Alignment]
        [ VSPACE = Pixels ]
        [ HSPACE = Pixles]                    >
  [ <PARAM NAME=name1 VALUE= value1 >]
  [ <PARAM NAME=name2 VALUE= value2 >]
</APPLET>
```

Adding applet to HTML file

```
<HTML>
        <! This page includes a welcome title in title bar
  and also display a          welcome message.  >
<HEAD>
        <TITLE> Welcome to Java Applets </TITLE>
</HEAD>
<BODY>
        <CENTER>          <H1> WELCOME </H1>
        </CENTER>
< APPLET  CODE=Hellojava.class  WIDTH=200
  HEIGHT=400 >
</APPLET>
</BODY>
</HTML>
```

RUNNING THE APPLET

- We must have following files in our current directory:

1) Hellojava.java
2) Hellojava.class
3) Hellojava.html

- To run an applet ,we require one of following tools:

1) **Java enabled web browser** :if we use it,we will be able to see the entire web page containing the applet.

2) **Java appletviewer:**

   - if we use it,we will only see the applet output.
   - appletviewer is not full fleged web browser and therfore it ignores all of the HTML tags except the part which runs applet.

Syntax: appletviewer Hellojava.html

STEPS TO ADDING AN APPLET TO HTML PAGE

1) Insert an **<APPLET>** tag at an appropriate place in web page.

2) Specify the name of the applet 's **.class** file.

3) If the **.class** file is not in the current directory ,uses the codebase parameter to specify

- The relative path if file is on the local system
- The URL of the directory containing the file if it is on a remote computer.

4) Specify the space required for display of he applet in terms of width and height in pixels.

STEPS TO ADDING AN APPLET TO HTML PAGE

5) Add any user defined parameters using <PARAM> tags.

6) Add alternate HTML text to be displayed when a non java browser is used.

7) Close the applet declaration with the </APPLET> tag.

PASSING PARAMETER TO APPLETS

- <PARAM ...> tag is used to supply user defined parameters to applet.
- Passing parameter to an appplet code using <PARAM> tag is similar to passiiing parameters to the **main()** method using command line argument.
- To pass and handles parameter ,do following:

1) Include appropriate <PARAM...> tag in HTML document.
2) Provide code in the applet to parse these parameters.

- Parameters are passed to an applet when applet is loaded.
- We can define **init( )** method in applet to get hold of the parameters defined in <PARAM> tags.

PASSING PARAMETER TO APPLETS

- getParameter() method ,takes one string argument representing the name of program.
- Each <PARAM…> tag has a *name* attribute (color) and value attribute (red).
- Inside the applet code ,the applet can refer to that parameter by name to find its value.
- We can also change text to be displayed by an applet by supplying new text to the applet through <PARAM….> tag.
- E.g. : <APPLET …..>

  <**PARAM** *name*=color *value*=“red”>

  <**PARAM** *name*=text *value*=“I like java!”>

  </APPLET>

Example of <PARAM> tag

```
import     java.applet.*;
import     java.awt.*;
public class Helloparam extends Applet
{          String str;
           public void init()
           {
           str=getParameter("string");
           if(str==null)
                      str="java";
                      str="hello" + str;
           }
```

Example of <PARAM> tag

```
          public void paint(Graphics g)
          {
                    g.drawString(str,10,100);
          }
}
```

//save file in path “D:\javapro\Helloparam.java”

HTML FILE FOR <PARAM> TAG

```
<html>
   <! parameterized HTML file>
<HEAD>
         <TITLE>Welcome to java Applets</TITLE>
</HEAD>
<BODY>
   <APPLET CODE=Helloparam.class width=400
   height=400 >
   <PARAM NAME="string" VALUE= "Applet !">
   </APPLET>
</BODY>
</html>
//save file in path “D:\javapro\Helloparam.html”
```

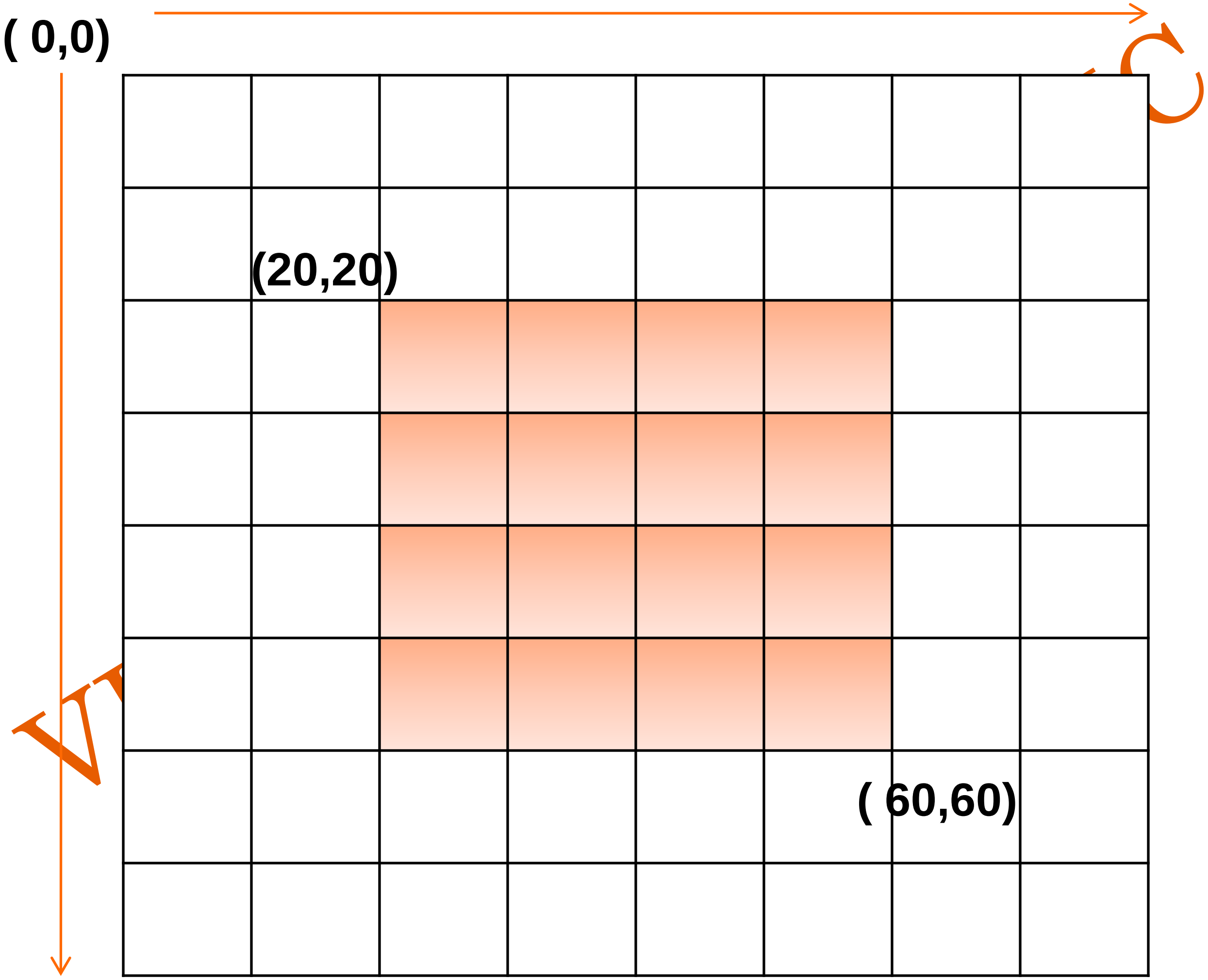
Graphics class

- The Graphics class is part of AWT.
- It is contained in **java.awt.Graphics** ,and it includes methods for drawing types of shapes or text in variety of fonts.
- Every applet has its own area of the screen known as **canvas. Where** it creates its display.
- Java applets draws graphical images inside its space using the coordinate system. Java's coordinate system has the origin (0, 0) in the upper-left corner.
- **Positive x** values are to the **right** and **positive y** values are to the **bottom. The** values of coordinates x and y are in pixels.

Graphics class

- Using drawing methods of class we can draw a shape on the screen.
- All the drawing methods have arguments representing end points, corners or starting location of shape as values in applet's coordinate system.

Coordinate system of Java



Methods of Graphics class

Methods	**Description**
clearRect( )	Erases a rectangular area of canvas
copyArea( )	Copies a rectangular area of the canvas to other area.
drawArc( )	Draws a hollow arc.
drawLine( )	Draws a straight line.
drawOval( )	Draws a hollow oval
drawPolygon( )	Draws a hollow polygon.
drawRect( )	Draws a hollow rectangle.
drawRoundRect( )	Draws hollow rectangle with rounded corners.

METHODS OF GRAPHICS CLASS

Methods	Description
drawString( )	Displays a text string
fillArc( )	Draws a filled arc.
fillOval( )	Draws a filled oval.
fillPolygon( )	Draws a filled polygon.
fillRect( )	Draws a filled rectangle.
fillR0undRect( )	Draws a filled rectangle with rounded corners
getColor( )	Retrieve the current drawing color
getFontMetrics( )	Retrieves information about the current font.
setColor( )	Sets the drawing color.
setFont( )	Sets the font.

Thank you