INTRODUCTION

210301301

CORE JAVA

210301305

PRACTICAL ON CORE JAVA

Unit – 5
Applets in Java

Index

- Applets
  - Applet Class
  - Applet Life Cycle
  - Applet Structure
  - Common Methods used with Applet
  - Applet using Graphics Class
  - Basic Controls in Japplet

Applet in Java

- 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.
- In other words, we can say that Applets are small Java applications that can be accessed on an Internet server, transported over Internet, and can be automatically installed and run as apart of a web document.
- After a user receives an applet, the applet can produce a graphical user interface.
- It has limited access to resources so that it can run complex computations without introducing the risk of viruses or breaching data integrity.

Applet in Java

- **Advantage of Applet**

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

  - It works at client side so less response time.
  - Secured

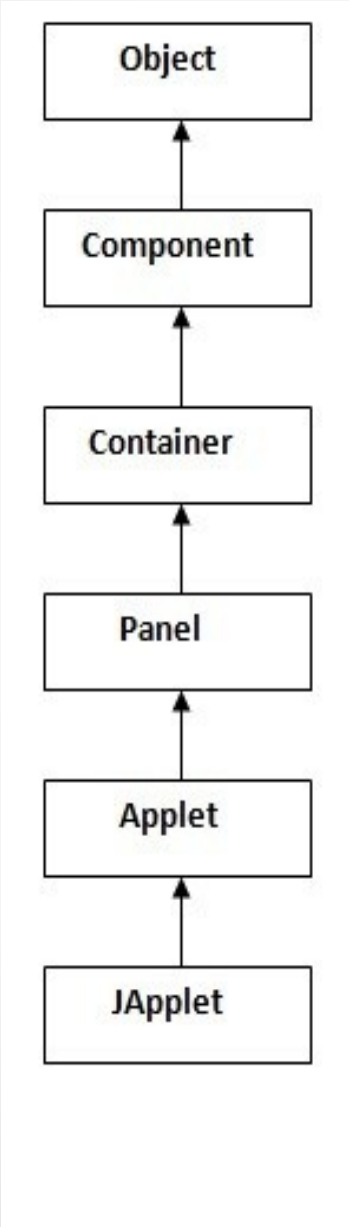
- It can be executed by browsers running under many platforms, including Linux, Windows, Mac Os etc.

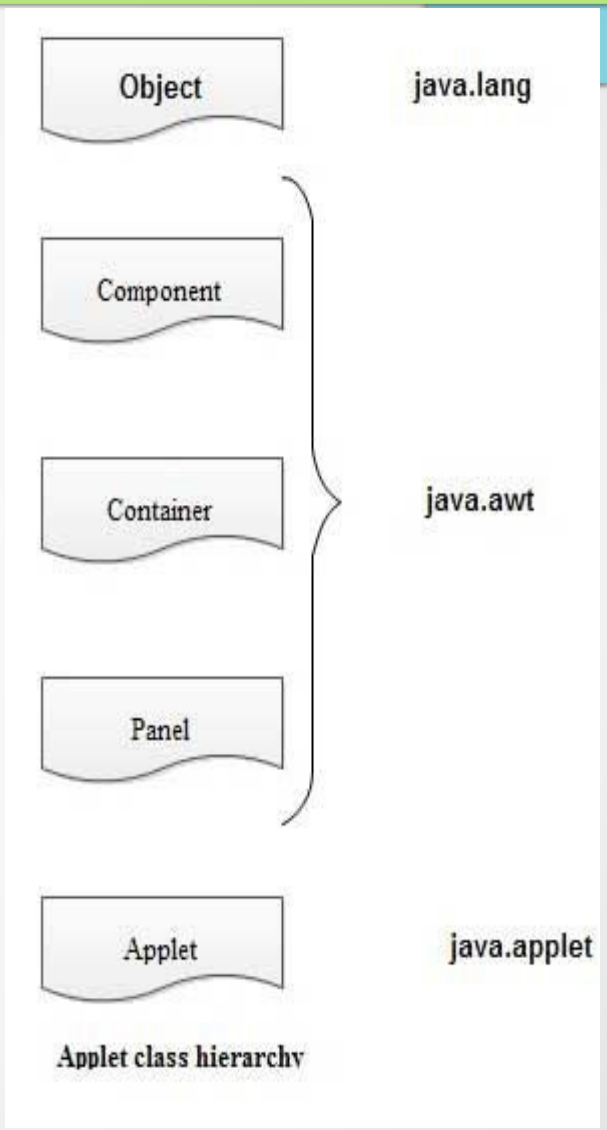
- **Drawback of Applet**
- Plugin is required at client browser to execute applet.

Difference between Applications & Applet

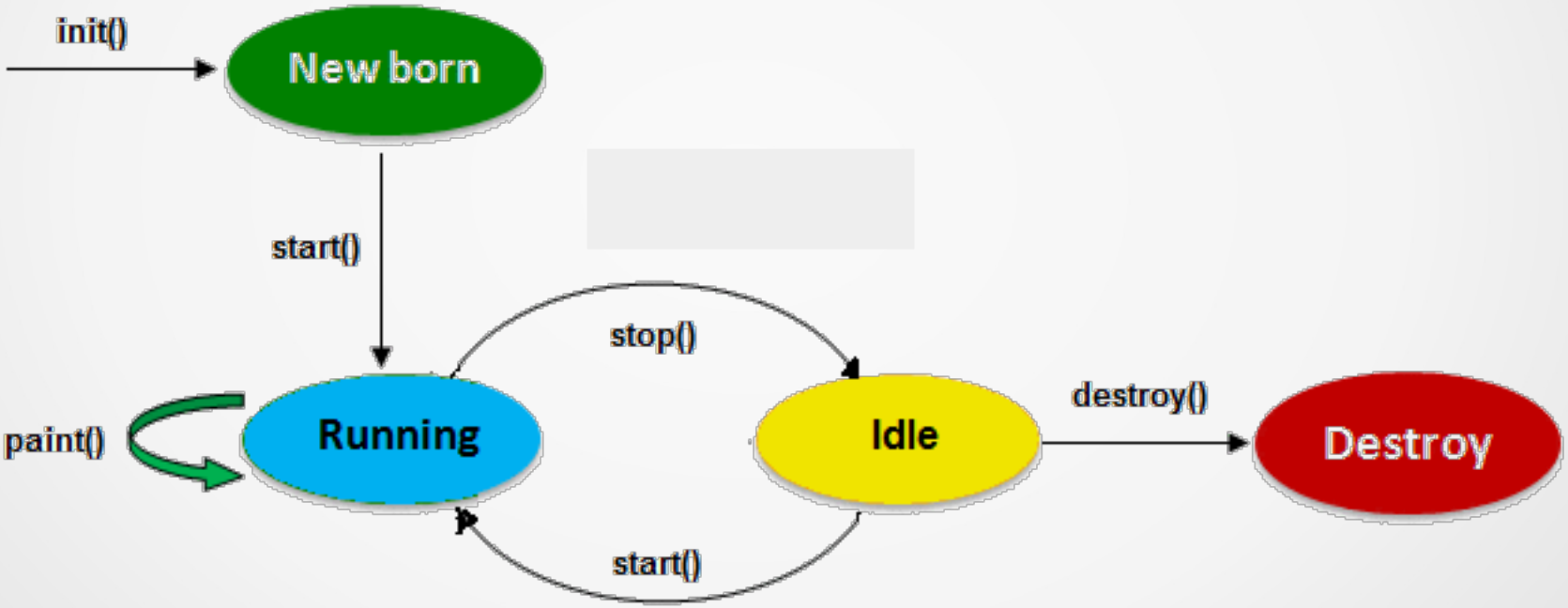
Applications	Applet
The execution of an application program starts from main().	The execution of the applet does not start from main() method.
These can run on their own. In order to get executed, they need not be embedded inside any web page.	Applets cannot run on their own. They have to be embedded inside a web page to get executed.
Applications are executed at command line.	Applets can be executed inside a web browser or appletviewer.
Applications have no inherent security restrictions.	Applets execute under strict security limitations that disallow certain operations.
Applications have their own life cycle.	Applets have their own life cycle. Init -> start -> paint -> stop -> destroy

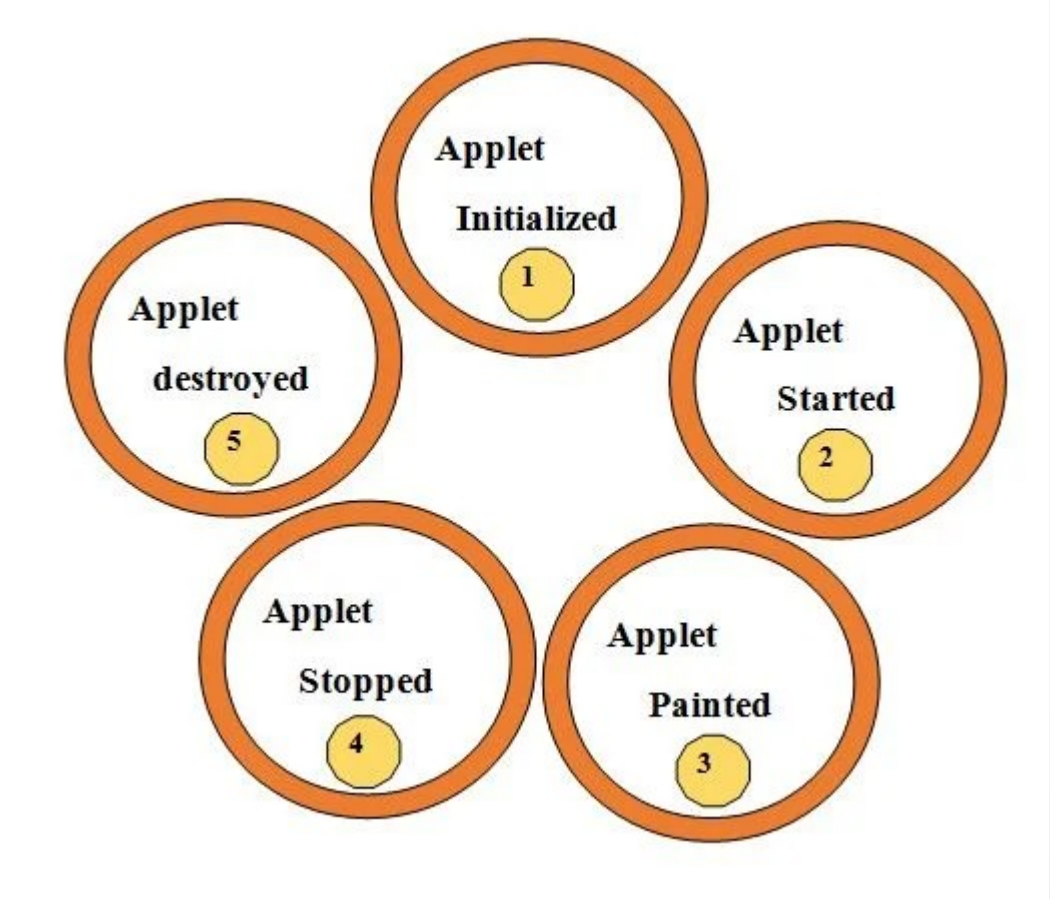
Hierarchy of Applet





Lifecycle of Java Applet




Applet
Initialized
1
Applet
Started
2
Applet
Painted
3
Applet
Stopped
4
Applet
destroyed
5

Lifecycle of Java Applet

- Applet is initialized.
- Applet is started.
- Applet is painted.
- Applet is stopped.
- Applet is destroyed.

Lifecycle methods for Applet

- The java.applet.Applet class 4 life cycle methods and java.awt. Component class provides 1 life cycle methods for an applet.

java.applet.Applet class

- 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.

Lifecycle methods for Applet

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.
- Java Plug-in software is responsible to manage the life cycle of an applet.

How to run an Applet?

- There are two ways to run an applet
  - By html file.
  - By appletViewer tool

Applet1.java

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.

Applet2.java

- public void drawRect(int x, int y, int width, int height)

  draws a rectangle with the specified width and height.

Applet5.java

Applet5_1.java

- public void fillRect(int x, int y, int width, int height)

  is used to fill rectangle with the default color and specified width and height.

Applet5_2.java

- public void drawOval(int x, int y, int width, int height)

  is used to draw oval with the specified width and height.

Applet8.java

- public void fillOval(int x, int y, int width, int height)

  is used to fill oval with the default color and specified width and height.

Applet9.java

Applet4.java

Displaying Graphics in Applet

- public void drawRoundRect(int x,int y,int width,int height,int arcWidth,int arcHeight)

  is used to draw the rounded rectangle.

- public void fillRoundRect(int x, int y, int width, int height,int arcWidth,int arcHeight)

  is used to fill rounded rectangle with the default color and specified width and height.

- public void setColor(Color c)

  is used to set the graphics current color to the specified color.

- public void setFont(Font font)

  is used to set the graphics current font to the specified font.

- public void drawLine(int x1, int y1, int x2, int y2)

  is used to draw line between the points(x1, y1) and (x2, y2).

- public boolean drawImage(Image img, int x, int y, ImageObserver observer)

  is used draw the specified image.

- public void drawArc(int x, int y, int width, int height, int startAngle, int arcAngle)

  is used draw a circular or elliptical arc.

- public void fillArc(int x, int y, int width, int height, int startAngle, int arcAngle)

  is used to fill a circular or elliptical arc.

setBackground and setForeground

- To set the color of the background of an applet window, setBackground () method is used.
- The general form of the setBackground () method is

  void setBackground(mycolor)

  Applet3.java

- Similarly, to set the foreground color to a specific color, that is, the color of text, setForeground () method is used.
- The general form of the setForeground () method is

  void setForeground(mycolor)

  Applet3_1.java

- mycolor is one of the color constants or the new color created by the user
- The list of color constants is given below:

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

  Applet1.java

Image in Applet

- **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 boolean drawImage(Image img, int x, int y, ImageObserver observer)

  is used draw the specified image.

  Applet6.java

- **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.

GetDocumentbase() and codeBase()

- In most of the applets, it is required to load text and images explicitly.
- Java enables loading data from two directories. The first one is the directory which contains the HTML file that started the applet (known as the **document base**).
- The other one is the directory from which the class file of the applet is loaded (known as the **code base**).
- These directories can be obtained as URL objects by using getDocumentBase ()and getCodeBase ()methods respectively.
- You can concatenate these URL objects with the string representing the name of the file that is to be loaded.

Applet7.java

repaint(), paint() and update()

- paint(): paint() is where you place code for drawing, writing etc. Initial placement of buttons and other widgets should not be done in paint(). Whenever a rendering is thought to be "damaged" (e.g, partially covered by another window), paint() is called to re-draw the scene.
- update(): update is called when the window is re-sized. The default implementation of update(): first clears the background; then calls paint().
- repaint(): The repaint() is intended to allow various methods to call for a re-rendering of the component. No graphics context is needed for repaint(). A call to repaint() calls update().

Graphics Class

- AppletFont.java
- Banner.java
- Doraemon.java
- DrawHouse.java
- GfgApplet.java
- House.java
- Household.java

Basic Controls on Japplet

- Jlable
- Jbutton
- JtextArea
- JtextField
- JradioButton
- JcheckBox
- JComboBox

Jlabel Constuctors

- JLabel ()
- JLabel (Icon image)
- JLabel (Icon image, int horizontalAlignment)
- JLabel (String text)
- JLabel (String text, Icon icon, int horizontalAlignment)
- JLabel (String text, int horizontalAlignment)

Jlabel Align

- The Jlabel has the follwing int type constants that indicate the alignment of the label content:
  - JLabel.CENTER
  - JLabel.LEFT
  - JLabel.RIGHT
  - JLabel.TOP
  - JLabel.BOTTOM

JLabel

- Jlabel is a built in Java Swing class that holds text you can display within an applet.
- Jlabel class is concrete subclass of **Jcomponent.**
- Java.lang.Object
  - Java.awt.Component
    - Java.awt.Container
      - Javax.swing.Jcomponent
        - Javax.swing.JLabel

Swing - JButton

- The JButton is a concrete subclass of abstract Button which is a sub class of Jcomponent.
- The JButton class is used to mouse press and mouse release events can be precessed separately.
- Constructors
  - JButton()
  - JButton(String label)
  - JButton(Icon i)
  - Jbutton(String label, Icon i)

JTextField

- Swing's **text component deals with two types of text :**
  - Simple text of one font and one color of text
  - Styled text with multiple fonts and multiple colors.
- Simple **type texts are deal by:**
  - *JTextFiled*
  - *JPasswordField*
  - *JTextArea*
- The styled texts are handled in :
  - *JEditorPane*
  - *JTextPane*

JTextField

- Simple type texts are deal by **JtextField, JpasswordField and JtextArea classes**
- **JtextField** is a subclass of **JTextComponent,** which is a subclass of **JComponent.**
- **JtextField can display one line of editable text** of one font and color at a time.
- The object of a JTextField class is a text component that allows the editing of a single line text.

JTextField

- Alignment can set using:
  - *JTextField.LEFT*
  - *JTextField.CENTER*
  - *JTextField.RIGHT*

JTextField

- A **JTextField** Constructors:
  - *JTextField()*
  - *JtextField(String s)*
  - *JtextField(int c)*
  - *JtextField(String s ,int c)*

Applet_control.java

JRadioButton

- Radio Buttons are like check boxes
- selection is displayed in a round graphics
- out of several options, only one will be in selected state and all the remaining are in deselected state

JRadioButton

- **Constructor:**
  - JRadioButton()
  - JRadioButton(Icon icon)
  - JRadioButton(Icon icon, boolean selected)
  - JRadioButton(String text)
  - JRadioButton(String text, boolean selected)
  - JRadioButton(String text, Icon icon)
  - JRadioButton(String text, Icon icon, boolean selected)

Applet_RB.java

JCheckBox

- In JcheckBox user is given an option to select any number of options out of several options given.
- **Constructor:**
  - JcheckBox()
  - JCheckBox(Icon icon)
  - JCheckBox(Icon icon, boolean selected)
  - JCheckBox(String text)
  - JCheckBox(String text, boolean selected)
  - JCheckBox(String text, Icon icon)
  - JCheckBox(String text, Icon icon, boolean selected)

Applet_CB.java

JTextArea

- JTextArea is a **subclass of JTextComponent.**
- JTextArea **component displays multiple lines** of text in one color and with one font.
- There is **no scroll bar to view the text.**
- If the text is large, then a **JscrollPane has to be created using the text area component.**

Applet_TA.java

JTextArea

- A **JTextArea** Constructors:
  - *Public JTextArea()*
  - *public JTextArea(int row, int column)*
  - *public JTextArea(String text, int row, int column)*

Applet_TA.java

JComboBox

- JComboBox is a **subclass of JComponent.**
- It is a **combination of JList & JtextField.**
- In JcomboBox, only **one item is visible at a time.**
- In Combox, **the items can be edited by setting the JComboBox editable.**
- JcomboBox **has a model view structure.**
- It has **DefaultComboBoxModel class**, which can be add more flexible methods to JComboBox.

UNIT – 4 JComboBox

- Constructors:
  - *JComboBox()*
  - *JComboBox(ComboBoxModel model)*
  - *JComboBox(Object[] arry)*
  - *JcomboBox(Vector v)*

Applet_combo.java