# GetDocumentBase and getCodeBase methods

***getDocumentBase():*** Gets the URL of the document in which this applet is embedded.

For example, suppose an applet is contained within the document:

*http://java.sun.com/products/jdk/1.2/index.html*

*The document base is:*

*http://java.sun.com/products/jdk/1.2/index.html*

***getCodeBase():*** Gets the base URL. This is the URL of the directory which contains this applet.

*Sample code:*

*Applet:*

```
URL url = getDocumentBase();
AudioClip audioClip = getAudioClip(url, "music/JButton.wav");
```

- In most of the applets, it is required to load text and images explicitly. Java enables loading data from two directories.
  i. The first one is the directory which contains the HTML file that started the applet (known as the **document base).**
  ii. 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.

```
/*
   <applet code="GetDocumentBaseExample" width=350 height=250>
   </applet>
 */

  import java.applet.Applet;
   import java.awt.Graphics;
   import java.net.URL;
   import java.awt.*;
   import java.awt.event.*;

   public class GetDocumentBaseExample extends Applet{

         public void paint(Graphics g){
         String message;

      //getCodeBase() method gets the base URL of the directory in which
contains this applet.
        URL appletCodeDir=getCodeBase();
         message = "Code Base : "+appletCodeDir.toString();
         g.drawString(message,10,90);

        // getDocumentBase() Returns an absolute URL of the Document
         URL appletDocDir = getDocumentBase();
         message="Document Base : "+appletDocDir.toString();
         g.drawString(message,10,120);
         g.drawString("http://ecomputernotes.com", 200, 250);


         }
   }
```

# JAVA Event Handling

**Event and Listener:** Changing the state of an object is known as an event. For example, click on button, dragging mouse etc. The java.awt.event package provides many event classes and Listener interfaces for event handling.

Any program that uses GUI (graphical user interface) such as Java application written for windows, is event driven. Event describes the change of state of any object.

**Example:** Pressing a button, Entering a character in Textbox.

**Components of Event Handling:**

Event handling has three main components,

- **Events:** An event is a change of state of an object.
- **Events Source:** Event source is an object that generates an event.
- **Listeners:** A listener is an object that listens to the event. A listener gets notified when an event occurs.

**How Events are handled?**

A source generates an Event and send it to one or more listeners registered with the source. Once event is received by the listener, they processes the event and then return. Events are supported by a number of Java packages, like **java.util**, **java.awt** and **java.awt.event**.

# IMPORTANT EVENT CLASSES AND INTERFACE

| EVENT CLASSES | DESCRIPTION | LISTENER INTERFACE |
|---|---|---|
| **ActionEvent** | generated when button is pressed, menu-item is selected, list-item is double clicked | ActionListener |
| **MouseEvent** | generated when mouse is dragged, moved, clicked, pressed or released also when the enters or exit a component | MouseListener |
| **KeyEvent** | generated when input is received from keyboard | KeyListener |
| **ItemEvent** | generated when check-box or list item is clicked | ItemListener |
| **TextEvent** | generated when value of text area or text field is changed | TextListener |
| **MouseWheelEvent** | generated when mouse wheel is moved | MouseWheelListener |
| **WindowEvent** | generated when window is activated, deactivated, deiconified, iconified, opened or closed | WindowListener |
| **ComponentEvent** | generated when component is hidden, moved, resized or set visible | ComponentEventListener |
| **ContainerEvent** | generated when component is added or removed from container | ContainerListener |
| **AdjustmentEvent** | generated when scroll bar is manipulated | AdjustmentListener |

| | | |
|---|---|---|
| **FocusEvent** | generated when component gains or loses keyboard focus | FocusListener |

***Example of Event Handling***

```
import java.awt.*;
import java.awt.event.*;
import java.applet.*;
import java.applet.*;
import java.awt.event.*;
import java.awt.*;

public class Test extends Applet implements KeyListener{
 String msg="";
 public void init() {
  addKeyListener(this);
 }
 public void keyPressed(KeyEvent k) {
  showStatus("KeyPressed");
 }
 public void keyReleased(KeyEvent k) {
  showStatus("KeyRealesed");
 }
 public void keyTyped(KeyEvent k) {
  msg = msg+k.getKeyChar();
  repaint();
 }
 public void paint(Graphics g) {
  g.drawString(msg, 20, 40);
 }
}
```

**HTML code:**

```
< applet code="Test.class" width=300, height=100 >
```