

PRACTICAL: 1

NOTEPAD

AIM: To implement Notepad Using Java

PLATFORM: JAVA

PROBLEM STATEMENT:

Create a Notepad Using JAVA. It should provide the following functionality:

1. Font :It changes the font of text component
2. Font Style :It provide various font styles such as Regular,Bold,Italic,etc
3. Font Size: It provide list of various font size available and on clicking on it according changes the size of text in text box.

DESCRIPTION: The java.awt.Font class is used to create Font objects to set the font for drawing text, labels, text fields, buttons, etc.

Syntax: **Font f = new Font(name, style, size);**

String <i>name</i>	int <i>style</i>	int <i>size</i>
"serif"	Font.PLAIN	Integer point size -- typically in range 10-48.
"sansserif"	Font.BOLD	
"monospaced"	Font.ITALIC	
or a system font.	Font.BOLD+Font.ITALIC	

Available System Fonts : For maximum portability, use the *generic font names*, but you can use any font installed in the system. It is suggested to use a font family *name*, and create the font from that, but you can also use the fonts directly. You can get an array of all available font family names or all fonts. // Font info is obtained from the current graphics environment.

```
GraphicsEnvironment ge = GraphicsEnvironment.getLocalGraphicsEnvironment();
```

```
//--- Get an array of font names (smaller than the number of fonts)
```

```
String[] fontNames = ge.getAvailableFontFamilyNames();
```

```
//--- Get an array of fonts. It's preferable to use the names above.
```

```
Font[] allFonts = ge.getAllFonts();
```

```
// setFont method is used to change the font of any component's text.
```

CONCLUSION: The notepad has been implemented.

PRACTICAL : 2
CCITT GROUP3 2D COMPRESSION METHOD

AIM : To implement compression Technique – CCITT Group3 2D

PLATFORM : JAVA

THEORY:

The CCITT Group3 2D Compression Scheme is also sometimes known as modified runlength encoding .It combines one dimensional coding scheme with a two dimensional coding scheme 2D encoding offers higher compression because statistically many lines differ very little from the lines above or the lines below. The CCITT Group3 2D scheme uses 'k' factor where the image is divided into several groups of k lines. The first line of every groups of k lines is encoded using the CCITT Group3 1D method , this using the CCITT Ghroup3 1D method,this line becomes the reference line for next line and 2 D scheme is used along with 1D scheme to encode the rest of the scanline in the group of k lines. The @D scheme uses a combination of additional codes called vertical code,pass code and horizontal code to encode every line in the group of k lines.

Pass Code:

There is only one type of pass code with the value of 0001.This is the case when the runlength in the reference line(b_1, b_2) is to the left of the next runlength in the coding line (a_1, a_2) that is b_2 is to the left of a_1 . The runlength $b_1 b_2$ is coded using makeup & terminating code.

			b1		b2						
Reference Line	0	0	1	1	0	0	0	0	0	0	0
Coding Line	0	0	0	0	0	0	0	1	1	0	0
	<i>a0</i>				<i>b1</i>	<i>b2</i>		<i>a1</i>		<i>a2</i>	

1. runlength $b_1 b_2$ coded
2. new a_0 becomes old b_2

Vertical Code:

This is the case when the runlength in reference line($b_1 b_2$) overlaps the next RL in coding line ($a_1 a_2$) by a maximum of +ive or -ive 3 pixels. The two different $a_0 a_1$ & $a_1 b_1$ runlength are coded

								b1		b2	
Reference Line	0	0	0	0	0	0	0	1	1	0	0
Coding Line	0	0	0	0	0	1	1	1	1	1	0
	<i>a0</i>					<i>a1</i>					<i>a2</i>

- runlength a_1, b_1 is encoded
- new a_0 becomes old a_1

Horizontal Code:

This is the case when the runlength in the ref line ($b_1 b_2$) overlaps the runlength ($a_1 a_2$) by more than plus or minus 3 pixels.The two runlength $a_0 a_1$ and $a_1 a_2$ are coded.

							b1		b2		
Reference Line	0	0	0	0	0	0	1	1	0	0	0
Coding Line	0	0	0	1	1	1	1	1	1	0	0
	<i>a0</i>			<i>a1</i>						<i>a2</i>	

runlength $a_0 a_1$ & $a_1 a_2$ are coded

new a0 becomes old a2

CONCLUSION: Thus CCITT Group3 2D Compression scheme is successfully implemented.

PRACTICAL : 3

AUDIO PLAYER

AIM : To implement a basic audio player using Java.

PLATFORM : JAVA

PROBLEM STATEMENT:

Create a Java applet which simulates an audio player. It should provide the following functionality:

1. Play Button : This button plays the specified audio file.
2. Loop Button : This button Loops the currently playing audio track.
3. Stop Button : Stops the sound file which is playing.

DESCRIPTION : The applet uses a sound in a GUI through which the user can stop/play/loop an audio file. The Java package contains a special interface java applet.-AudioClip. The audio clip interface provides the required functionality if > 1 sound is played,the output is mixed.

getAudioClip(url,string) method returns a state object,parameter being:

getCodebase():which return absolute path of java class file.

A string which is a path relative to absolute path.

The 3 buttons implement ActionListener for click event.

CONCLUSION : The audio player has been implemented.

PRACTICAL : 4

AUDIO MIXER

AIM : To implement Audio Mixer Using Java

PLATFORM : JAVA

PROBLEM STATEMENT:

Create a Java applet which simulates an audio player. It should provide the following functionality:

1. Play Button : This button plays the specified audio file.
2. Mix Button : This button mixes the currently playing audio file with another.
3. Stop Button : Stops the sound file which is playing.

DESCRIPTION : The applet uses a sound in a GUI through which the user can stop/mix/play an audio file. The Java package contains a special interface java applet.-AudioClip. The audio clip interface provides the required functionality if 'Start' clip1 is played,the output is mixed.

getAudioClip(url,string) method returns a state object,parameter being:

getCodebase():which return absolute path of java class file.

A string which is a path relative to absolute path.

The 3 buttons implement ActionListener for click event.

CONCLUSION : The audio mixer has been implemented.

PRACTICAL : 5

MEDIA PLAYER

AIM : To implement a basic media player using Java.

PLATFORM : JAVA

THEORY:

Media player can be created by instantiating the player class provided by java. For this, we have a Manager class for creating and utilizing various multimedia objects. Various interface provided that can create a media player, remove any previous events that may take place. The component class provides us with features that help us add or remove the visual area and control area of the player. The container class help us to access the visual area & the control area of the player in case of an event. Media player can be created that runs on a variety of platforms and can support various audio and video file formats

ALGORITHM:

1. Create a media player object that contains a player and file format component.
2. File opening mode allows user to browse through folders to select the required file.
3. Create a player by using the manager class and add a listener to listen to vents.
4. On clicking of any control on media player call the suitable operation and set the visual area components content accordingly.

CONCLUSION : The media player has been implemented.

PRACTICAL : 6

CALCULATOR WITH AUDIO

AIM : To design a Calculator with audio and with simple operation.

PLATFORM : JAVA

PROBLEM STATEMENT:

Create an object which can do the following:

1. Input the 2 nos and select an arithmetic operation such as add/subtract/divide/multiply.
2. Play an audio that describe the operand operation and result.

Example: $3 + 3 = 6$ should play "Three Plus Three equals six"

DESCRIPTION :

Audio must be created each number operation & stick words like "equals". These files are loaded in the applet using `getAudioClip()` applet method.

Control for the calculator are created using following classes:

`java.awt.Button`:for add & subtract buttons.

`java.awt.TextFiled`s:for inputing numbers & displaying result.

When an operation is performed ,the corresponding records are played serially.

CONCLUSION : Calculator with audio has been created.

PRACTICAL : 7

ANIMATION

AIM : To implement an animated applet

PLATFORM : JAVA

THEORY: Animation can be implemented via applets calling the repaint function whenever the contents of the applet need to be redrawn . However ,the repaint function leads to a flickering effect due to repainting of the applet. To have a smooth transition in motion ;the update function does not remove the previous contents of the applet.

Once the applet is started , the paint function is executed first. Later calls done by repaint indirectly call paint function with new parameter In update the previous calls to paint are still maintained

ALGORITHM :

1. Create an applet and initialize it.
2. Create an image file and display it in the applet at any location.
3. Add translation parameter to the image position and call repaint function to draw the translated image.

CONCLUSION: Animation was implemented in Java.

PRACTICAL: 8

FLASH ANIMATION

AIM: To create macromedia flash animation.

SOFTWARE: Macromedia Flash 8

DESCRIPTION:

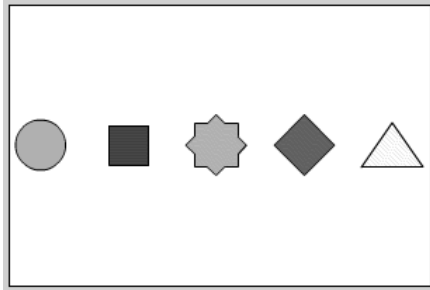
Macromedia Flash MX is one of the hottest technologies on the Web today. Leading corporate Web sites use its streamlined graphics to communicate their brands; major motion picture studios promote theatrical releases with Flash animations; and online gaming and educational sites provide rich user experiences with Flash interactivity.

As a vector-based animation and authoring application, Flash is ideal for creating high-impact, low-bandwidth Web sites incorporating animation, text, video, and sound. With robust support for complex interactivity and server-side communication, Flash is increasingly the solution for developing Internet applications as well. From designer to programmer, Flash has become the tool of choice for delivering dynamic content across various browsers and platforms.

To create a continuous scrolling graphic:

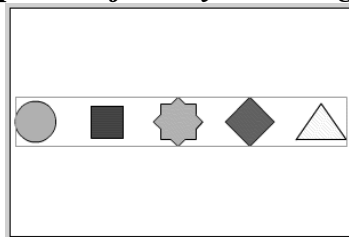
1. Create the necessary elements that will scroll across the Stage, and place them as they would appear at any given moment (Figure 1.1).

Figure 1.1. Five objects placed across the Stage as they would appear when they begin scrolling across from right to left. The objects could be buttons or simple graphics.



2. Select all the elements, and choose Modify > Group (Figure 1.2).

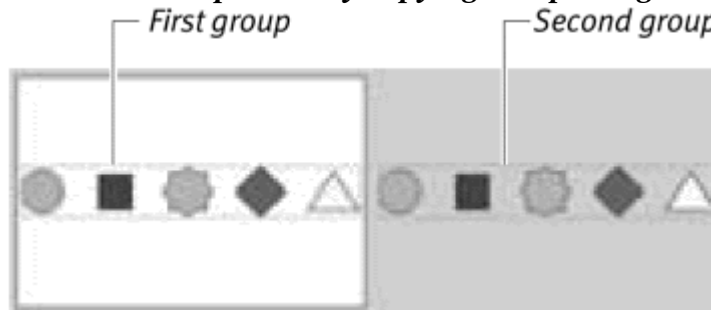
Figure 1.2. Group the objects by choosing Modify > Group.



3. Copy the group, and paste the copy next to the original group to create a long band of repeating elements.

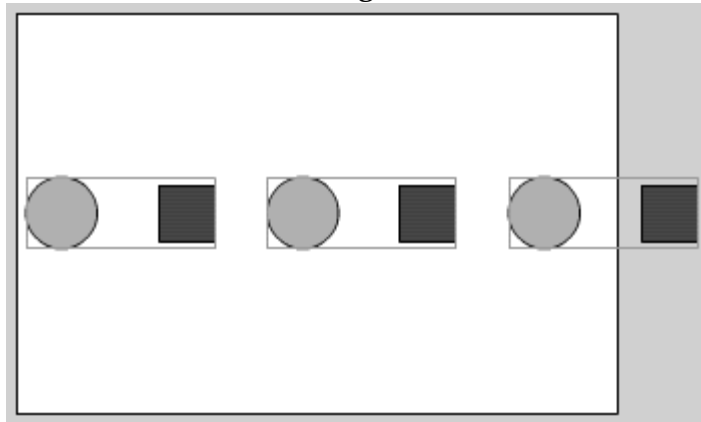
If your elements scroll from right to left, for example, place the second group to the right of the first group (Figure 1.3).

Figure 1.3. Create a pattern by copying and pasting the group.



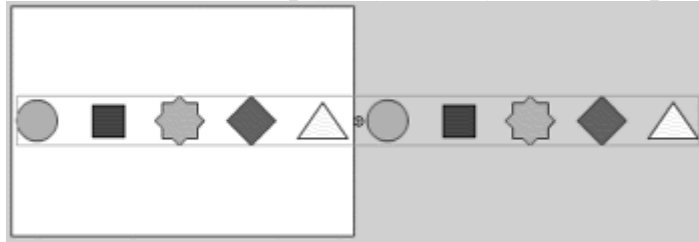
Your scrolling elements usually will be larger than the Stage, but if your first group is smaller, you'll need to duplicate it more than once to create a repeating pattern that extends beyond the Stage ([Figure 1.4](#)).

Figure 1.4. This group has only two objects. Repeat the groups to extend well past the Stage.



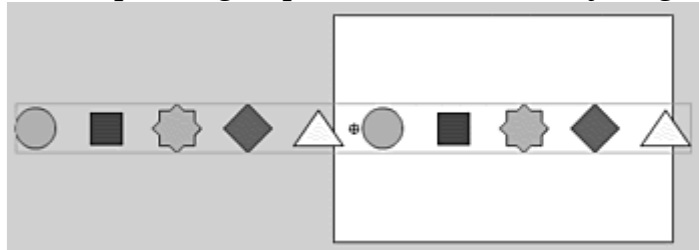
4. Select all your groups, and convert your selection to a graphic symbol ([Figure 1.5](#)). An instance of the symbol remains on the Stage, allowing you to apply a motion tween.

Figure 1.5. Create a graphic symbol of the entire pattern.



5. Create a keyframe at a later point in the Timeline.
6. Select the instance in the last keyframe, and move it so that the second repeated group of elements aligns with the first. When you move your instance, use its outlines to match its previous position ([Figure 1.6](#)).

Figure 1.6. The second repeated group is moved where the first group was originally.



7. Apply a motion tween between the keyframes.
8. Insert a new keyframe just before the last keyframe, and remove the last frame ([Figure 1.7](#)).

Figure 1.7. Create a new keyframe (top), and delete the last keyframe (bottom).



This technique makes the animation not have to play two identical frames (the first and the last) and creates a smooth loop.

CONCLUION : The flash file has been created.

PRACTICAL : 9

WEB PAGE

AIM : To create a HTML page for Online Shopping which include various flash, gif, sound and video files

SOFTWARE : Macromedia Flash 8, Notepad and Mozilla Firefox.

DESCRIPTION :

Flash is commonly used to create animation, advertisements, various web page components, to integrate video into web pages, and more recently, to develop rich Internet applications.

Embedding a Flash File

```
<html >  
<head>
```

```
<title>marrow</title>
```

```
</head>
```

```
<body bgcolor="#ffffff">
```

```
<embedsrc="marrow.swf" loop="false" quality="high" bgcolor="#ffffff" width="120" height="600"  
name="marrow" align="middle" allowScriptAccess="sameDomain" type="application/x-shockwave-  
flash" pluginspage="http://www.macromedia.com/go/getflashplayer" />
```

```
</body>
```

```
</html>
```

Embedding a Sound

Sound can be included in background to an Internet Explorer page using the bgsound element:

```
<bgsoundsrc="bigsound s.wav">
```

Or, in Netscape, the embed element:

```
<embedsrc="bigsound s.wav" autostart="true">
```

Embedding a Video File

To link to a video file, simply include it in an a element, as in the following example:

```
<a href="myVideo.mpg">This is a movie link.</a>
```

When a user clicks the link and has the supporting software, the video will play in the user's default media player.

Embedding a GIF File

The animated GIF can simply be included in an img element, like so:

```
</html>
```

CONCLUION: The Web page has been developed.

Creating animated GIFs

AIM: To create ANIMATED GIF

SOFTWARE: Macromedia Flash 8

DESCRIPTION:

Including an animated GIF file is the easiest of all the multimedia tasks. All Animated GIF creation programs are different, but their essence is the same. You start off with one frame, and when you want your image to change, you create another frame to represent that change. The frames can be on a complicated timeline, or as simple as the interface shown in Figures 2 and 3, which demonstrate an animated advertising banner being created using Macromedia's Fireworks.

The animation creation software then generates an animated GIF file consisting of as many frames as you indicate. You can also set the amount of time between the frame changes, as shown in Figure 4. Figures 5 and 6 show the transition between one frame of a completed animated advertising banner and another.

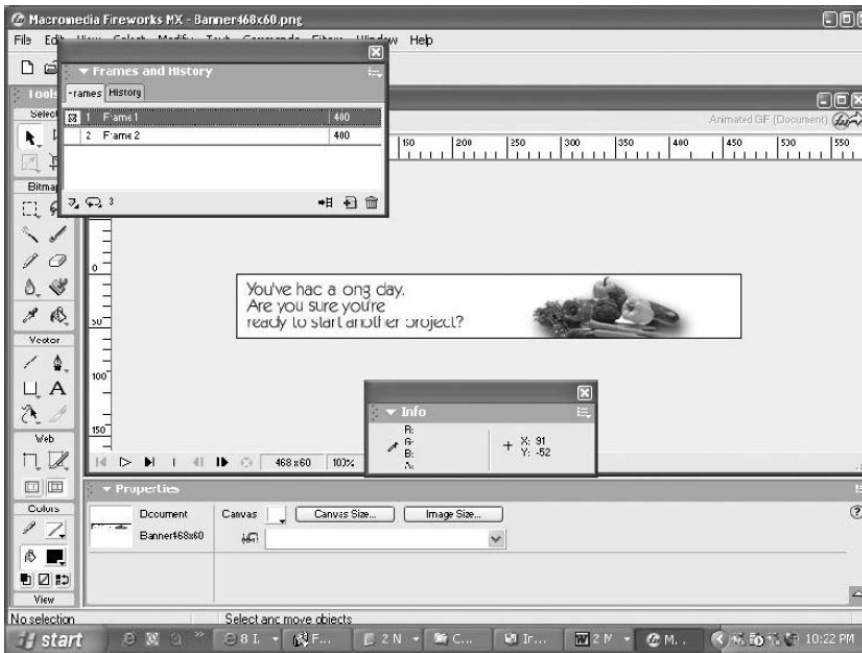


Figure 2: The first frame of an animation built in Fireworks.



Figure 2: The first frame of an animation built in Fireworks.

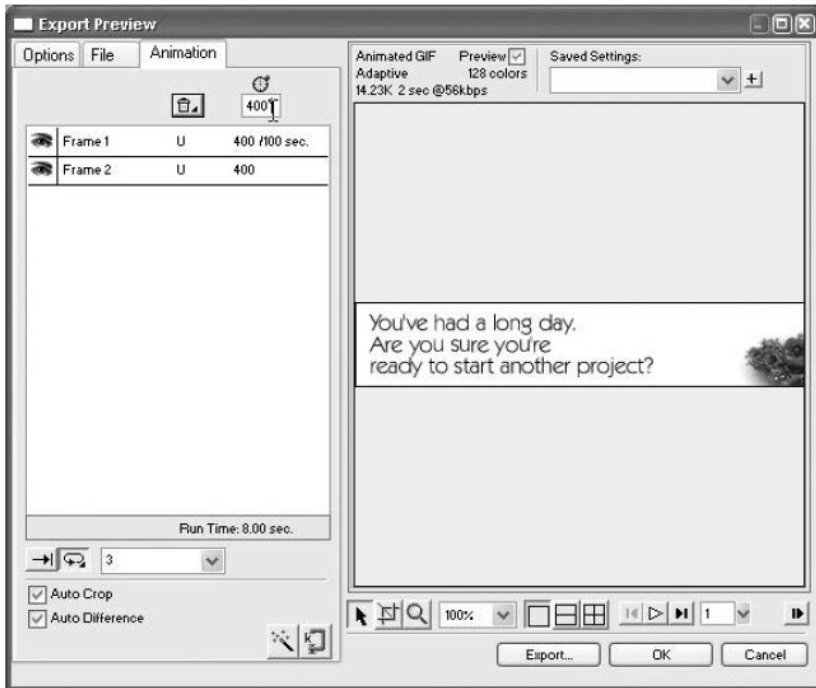


Figure 4: Changing the time interval between frame changes.



Figure 5: Transitioning between one frame of a completed animated advertising banner and another, part one.

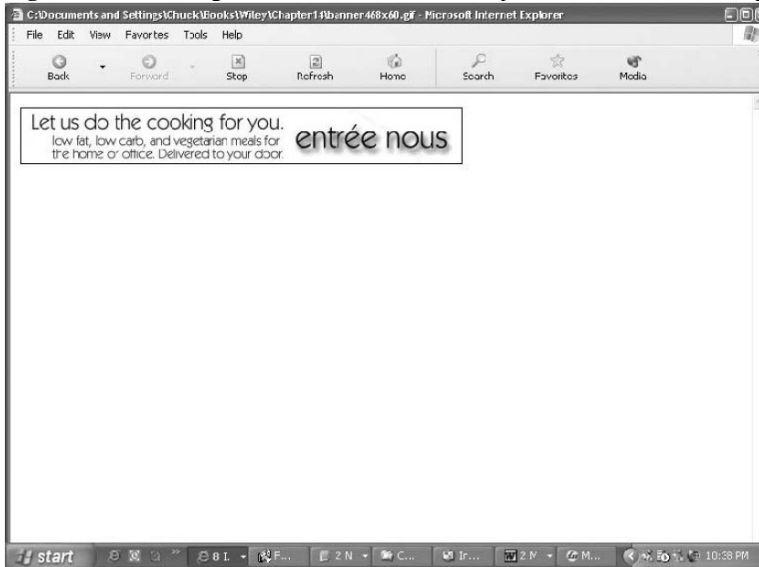


Figure 6: Transitioning between one frame of a completed animated advertising banner and another, part two.

CONCLUION: The Animated GIF has been developed.