Unit VII

Working with Graphics User Interface (GUI)

Tutor: Pema Galey

Learning Outcomes

In this session, you will learn about:

- Window Fundamentals
- Work with Abstract Window Toolkit (AWT) control components
- Identify various Graphics methods in Java
- Working with Frame windows
- Working with Font class

Introducing GUI in Java

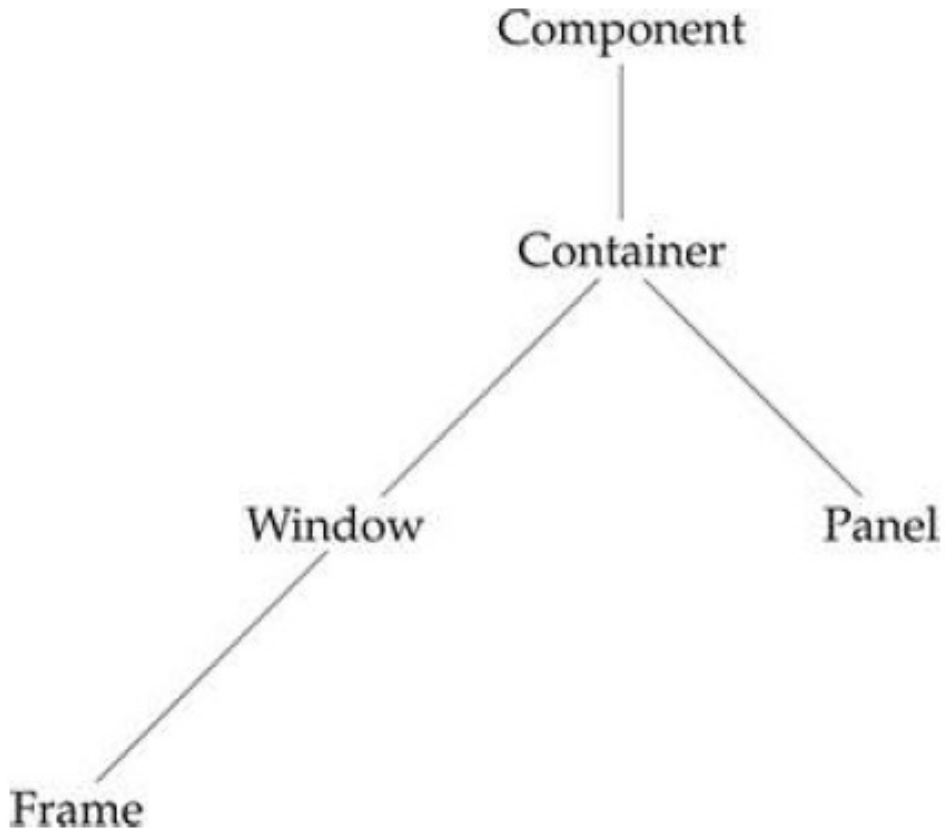
- The Abstract Window Toolkit (AWT) was Java's first GUI framework.
  - The AWT classes are contained in the **java.awt** package.

- New GUI frameworks are Swing and JavaFX

Note: *Applet (java.applet) was also GUI framework but now depreciated*

Windows Fundamentals

- Window fundamentals:
  1. Component
  2. Container
  3. Panel
  4. Window
  5. Frame
  6. Canvas



AWT Containers

Component :

- **Component** is an abstract class that encapsulates all of the attributes of a visual component.
- Except for menus, all user interface elements that are displayed on the screen and that interact with the user are subclasses of **Component**
- Responsible for managing events, such as mouse and keyboard input, positioning and sizing the window, and repainting
- A **Component** object is responsible for remembering the current foreground and background colors and the currently selected text font.

AWT Containers

Container:

- The Container is a component in AWT that can contain another components like buttons, textfields, labels etc.
- The classes that extends Container class are known as container such as Frame, Dialog and Panel.

Window:

- The window is the container that have no borders and menu bars. You must use frame, dialog or another window for creating a window.

Panel:

- The Panel is the container that doesn't contain title bar and menu bars. It can have other components like button, textfield etc.

Frame:

- The Frame is the container that contain title bar and can have menu bars. It can have other components like button, textfield etc.

Useful methods of the Component class

Method	Description
`public void add(Component c)`	inserts a component on this component.
`public void setSize(int width,int height)`	sets the size (width and height) of the component.
`public void setLayout(LayoutManager m)`	defines the layout manager for the component.
`public void setVisible(Boolean status)`	changes the visibility of the component, by default false.

Graphics Class

- AWT supports various graphics methods that enable you to draw shapes, such as lines, arcs, ellipses, circles and rectangles.
- To use the graphics methods, you must import the class Graphics which is inside the awt package:

```
import java.awt.Graphics;
```

Working with Frame Windows

- **Frame** constructors:
  - `Frame( ) throws HeadlessException`
  - `Frame(String title) throws HeadlessException`

- Some of the methods:
  - `void setSize(int newWidth, int newHeight)`
  - `void setSize(Dimension newSize)`
  - `void setVisible(boolean visibleFlag)`
  - `void setTitle(String newTitle)`

Working with Frame Windows

The `paint( )` Method

- The method is defined by **Component** and overridden by **Container** and **Window**. Thus, it is available to instances of **Frame**.
- The **paint( )** method is called each time an AWT-based application's output must be redrawn.
- The **paint( )** method is shown here:
  - `void paint(Graphics context)`

Working with Frame Windows

- **Displaying a String**
  - `void drawString(String message, int x, int y)`
- **Setting the Foreground and Background Colors**
  - `void setBackground(Color newColor)`
  - `void setForeground(Color newColor)`

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

Graphics Methods in Java

Drawing lines, rectangles, and polygons:

- The drawLine()method is used to draw lines in an applet.
- The following syntax shows how to define the drawLine()method:
  - `void drawLine(int x1, int y1, int x2, int y2)`
- The drawRect()is used to draw a rectangle in an applet.
- The following syntax shows how to define the drawRect()method:
  - `void drawRect(int x, int y, int width, int length)`
- You can also draw arbitrary shapes, such as polygons using the `drawPolygon()` and `fillPolygon()` methods.
- The following syntax shows how to use the drawPolygon()method:
  - `void drawPolygon(int x[], int y[], int num)`

Graphics Methods in Java

Drawing arcs, circles, and ellipses:

- The `drawArc()` method is used to draw an arc.
- The following syntax shows how to define the drawArc()method:
  - **`void drawArc(int x, int y, int width, int height, int startAngle, int sweepAngle)`**
- The `drawOval()` method is used for drawing circles and ellipses.
- The following syntax shows how to define the `drawOval()` method:
  - **`void drawOval(int x, int y, int width, int height)`**

Graphics Methods in Java

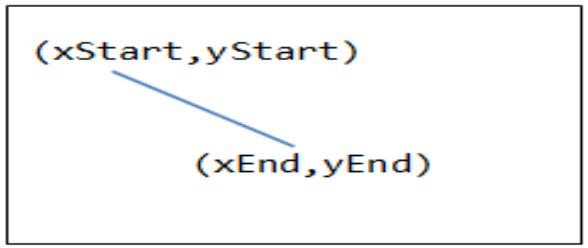
Painting various graphic objects:

- The `fillPolygon()` method is used to draw a filled polygon in an applet.
- The following syntax shows how to define the `fillPolygon()` method:
  - `void fillPolygon(int x[], int y[], int num)`
- The `fillOval()` method is used to draw filled circles and ellipses.
- The following syntax shows how to define the `fillOval()` method:
  - `void fillOval(int x, int y, int width, int height)`

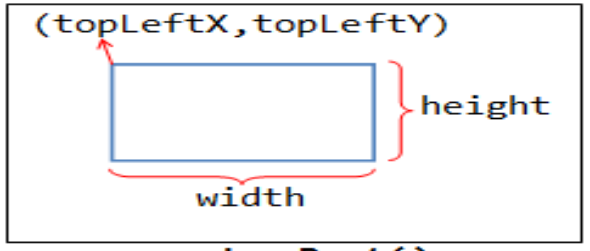
Graphics Methods in Java

- You can draw filled arcs using the `fillArc()` method.
- The following syntax shows how to define the `fillArc()` method:
  - **`void fillArc(int x, int y, int width, int height, int startAngle, int sweepAngle)`**
- The **`fillRect()`** method is used to draw a filled rectangle in an applet.
- The following syntax shows how to define the **`fillRect()`** method:
  - **`void fillOval(int x, int y, int width, int height)`**

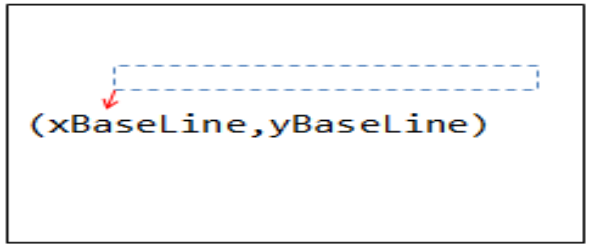
Method name	Function	Syntax
drawLine()	To draw a line	void drawLine(int x1, int y1, int x2, int y2)
drawRect()	To a draw a rectangle	void drawRect(int x, int y, int width, int length)
drawPolygon()	To draw a polygon	void drawPolygon(int x[], int y[], int num)
drawArc()	To draw an arc	void drawArc(int x, int y, int width, int height, int
drawOval()	To draw circles and ellipses	void drawOval(int x, int y, int width, int height)
fillPolygon()	To draw a polygon with filled color	void fillPolygon(int x[], int y[], int num)
fillOval()	To draw filled circles & ellipses	void fillOval(int x, int y, int width, int height)
fillArc()	To draw filled arcs	void fillArc(int x, int y, int width, int height, int
fillRect()	To draw fill rectangles	void fillRect(int x, int y, int width, int length)



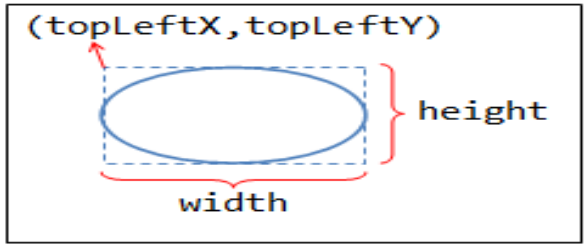
drawLine()



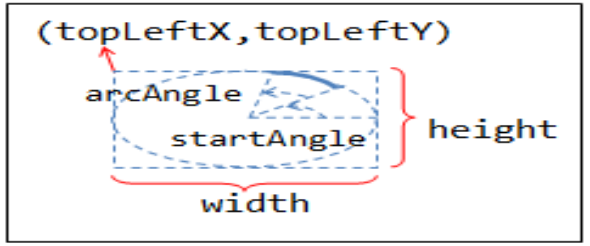
drawRect()



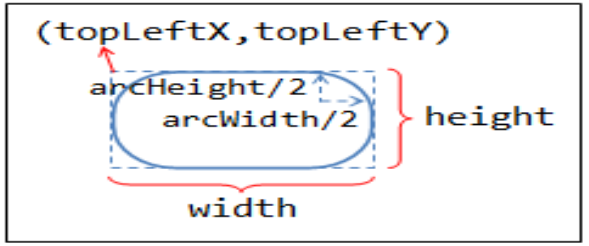
drawString()



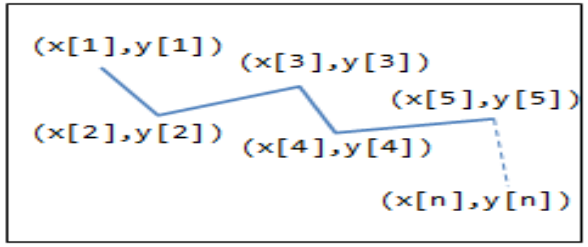
drawOval()



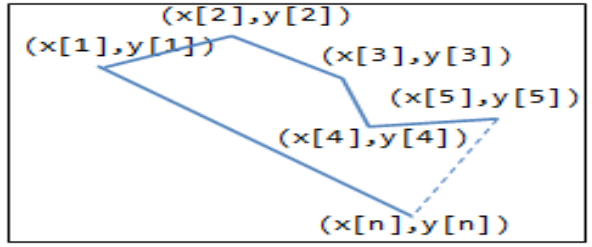
drawArc()



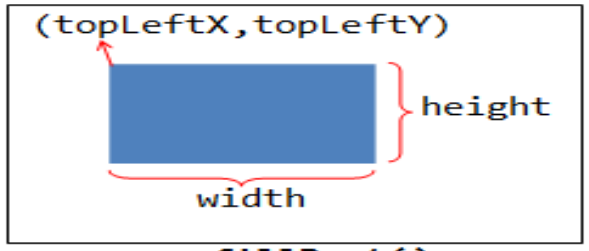
drawRoundRect()



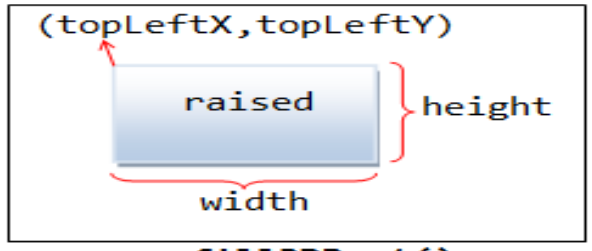
drawPolyline()



drawPolygon()



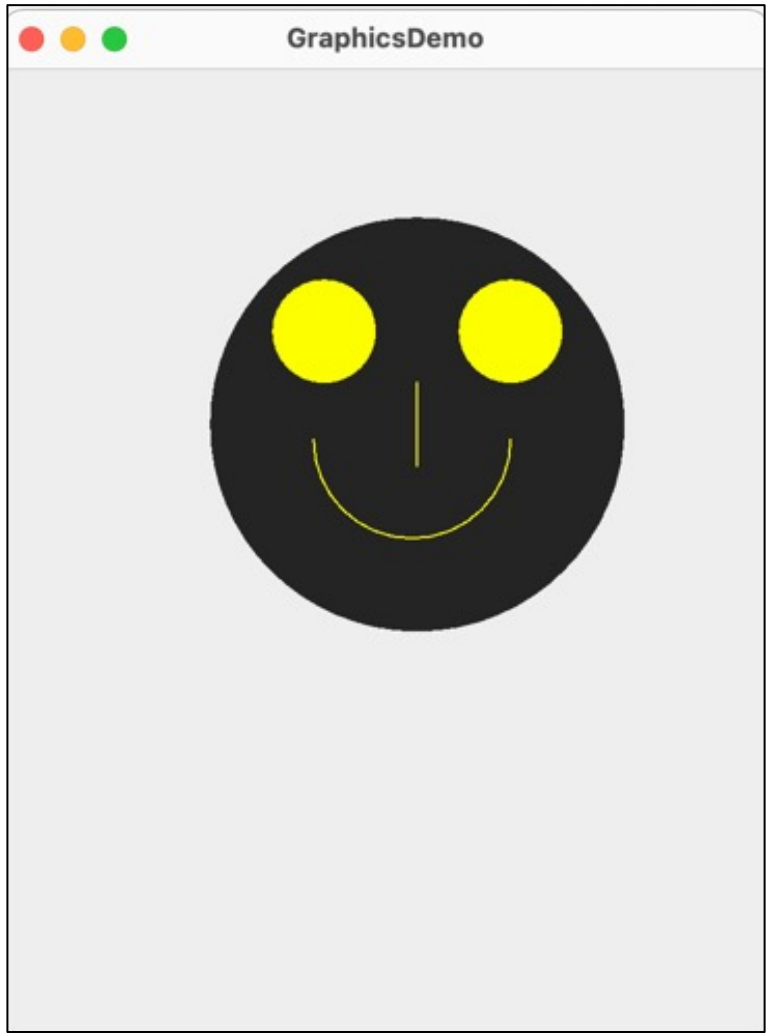
fillRect()



fill3DRect()

Example: Graphics

```
import java.awt.Color;
import java.awt.Dimension;
import java.awt.Frame;
import java.awt.Graphics;
public class FrameDemo extends Frame {
   public void paint(Graphics g) {
      g.fillOval(100, 100, 200, 200);
      g.setColor(Color.yellow);
      g.fillOval(130, 130, 50, 50);
      g.fillOval(220, 130, 50, 50);
      g.drawLine(200, 180, 200, 220);
      g.drawArc(150, 160, 95, 95, 0,-180);
   }
   public static void main(String []args){
      FrameDemo fam =new FrameDemo();
      fam.setSize(new Dimension(370,500));
      fam.setTitle("GraphicsDemo");
      fam.setVisible(true);
   }
}
```



Working with Fonts

- The AWT supports multiple type fonts
- Fonts have a family name, a logical font name, and a face name.
- Fonts are encapsulated by the **Font** class.
- **Font** class constructors:
  - `new Font(family, font_name, font_size);`
- Method:
  - `setFont(new Font());`

Home Work-1

- Design the student registration form using HTML. The following fields must be included:
  - Name
  - Student Number
  - Programme (Dropdown)
  - Year (Dropdown)
  - Semester (Dropdown)
  - Module(s) – use textarea to list multiple modules
  - Amount (Nu.) if any
  - Submit (Button)

Home Work - 2

1. Draw a house with the help of Graphics Methods and include your house description
2. Draw a Smiley face
3. Create your own design using different shapes

Thank you!