CSC207H: Software Design
Lecture 11

Wael Aboelsaadat

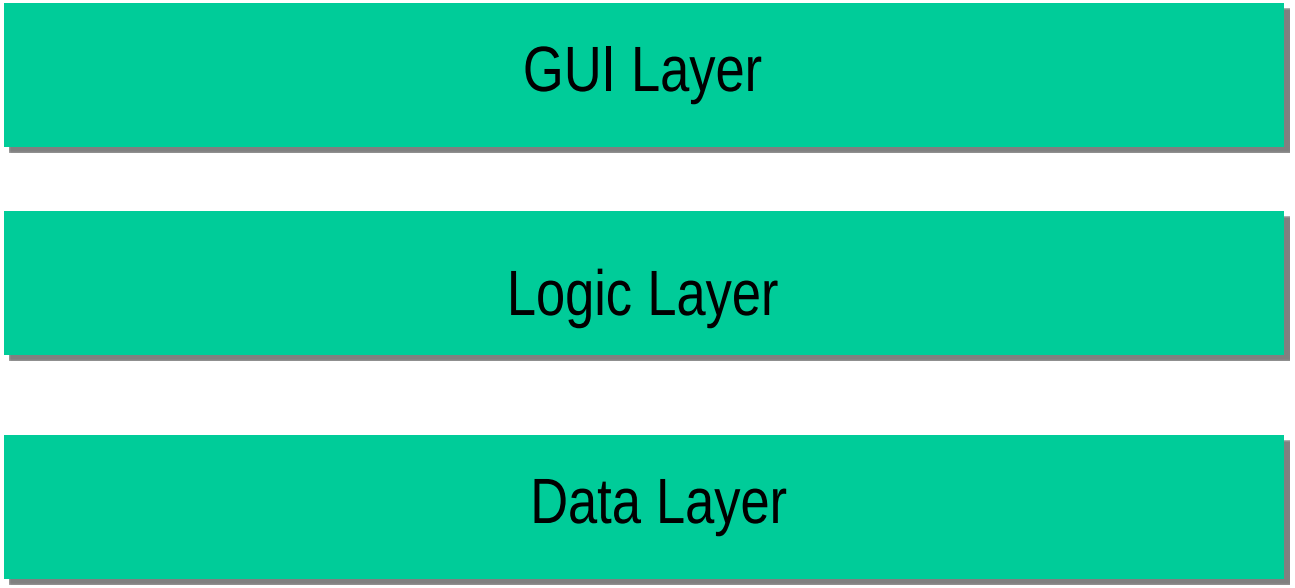
wael@cs.toronto.edu
http://ccnet.utoronto.ca/20075/csc207h1y/
Office: BA 4261
Office hours: R 5-7

Acknowledgement: These slides are based on material by Prof. Karen Reid (University of Toronto), Prof. Chris North (Virginia Tech), and Prof. Birchfield (Clemson Univ.)

Software house: what happens inside?

Understand the requirements	Project Managers Group
Design the program	Architects Group
Write the program	Development/Programmer Group
Test the program	QA Group
Write Documentation	Documentation Group
Package/Sell/Market program	Marketing/Sales Group

Program Architecture



- 3 separate teams
  - Different skill sets
  - Different quality criteria

GUI Layer

Typical command line program

- Non-interactive
- Linear execution

IBM 705

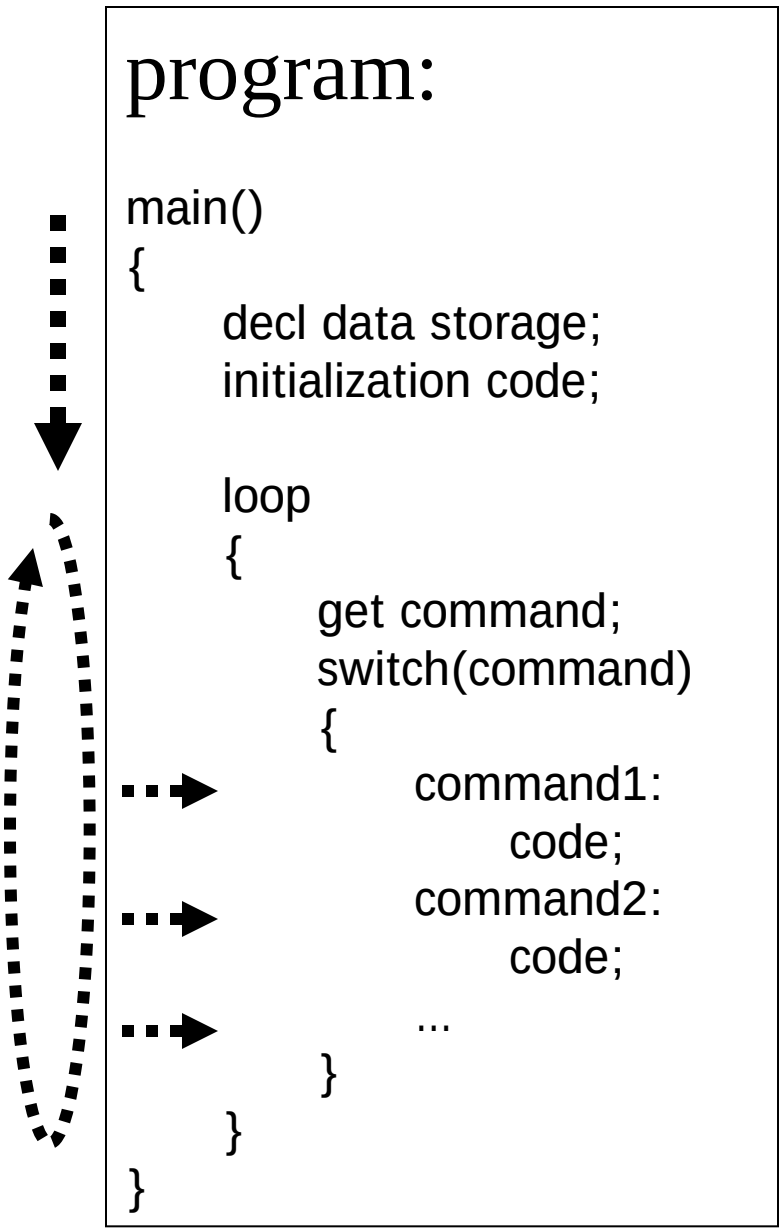
Univac 1956

program:

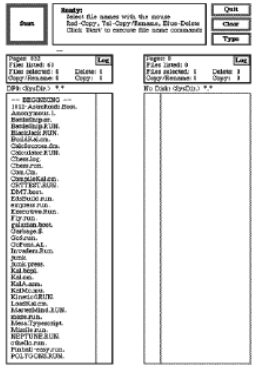
```
main()
{
        code;
        code;
        code;
        code;
        code;
        code;
        code;
        code;
        code;
        code;
        code;
        code;
}
```

Interactive command line program

- User input commands
- Non-linear execution
- Unpredictable order
- Much idle time



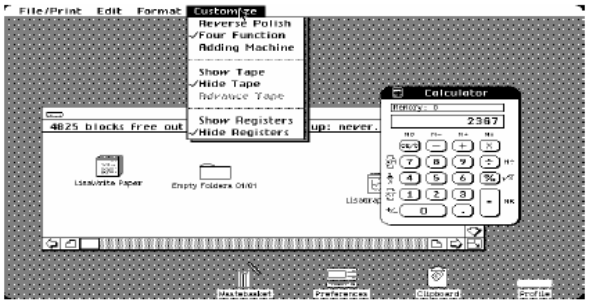
Interactive Graphical User Interface

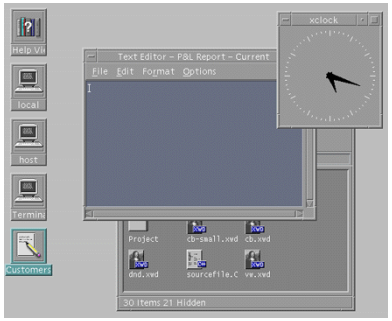


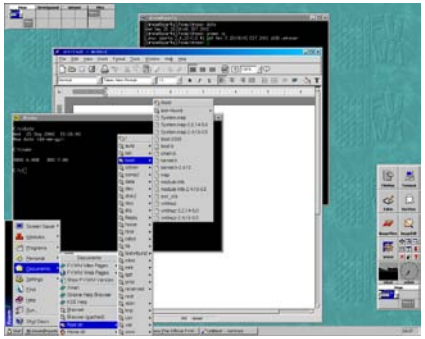
Xerox Alto

Xerox PARC, 1973

Lisa Interface

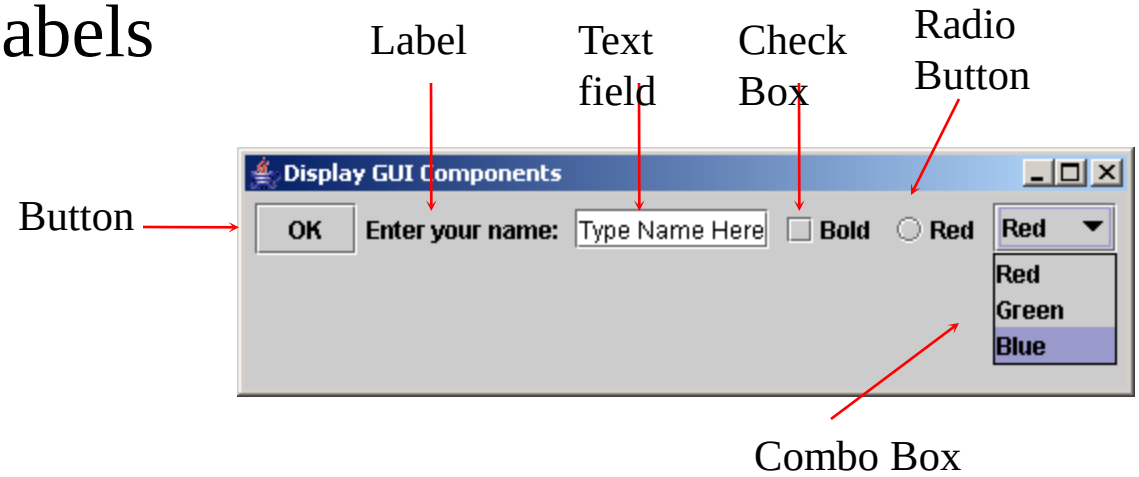




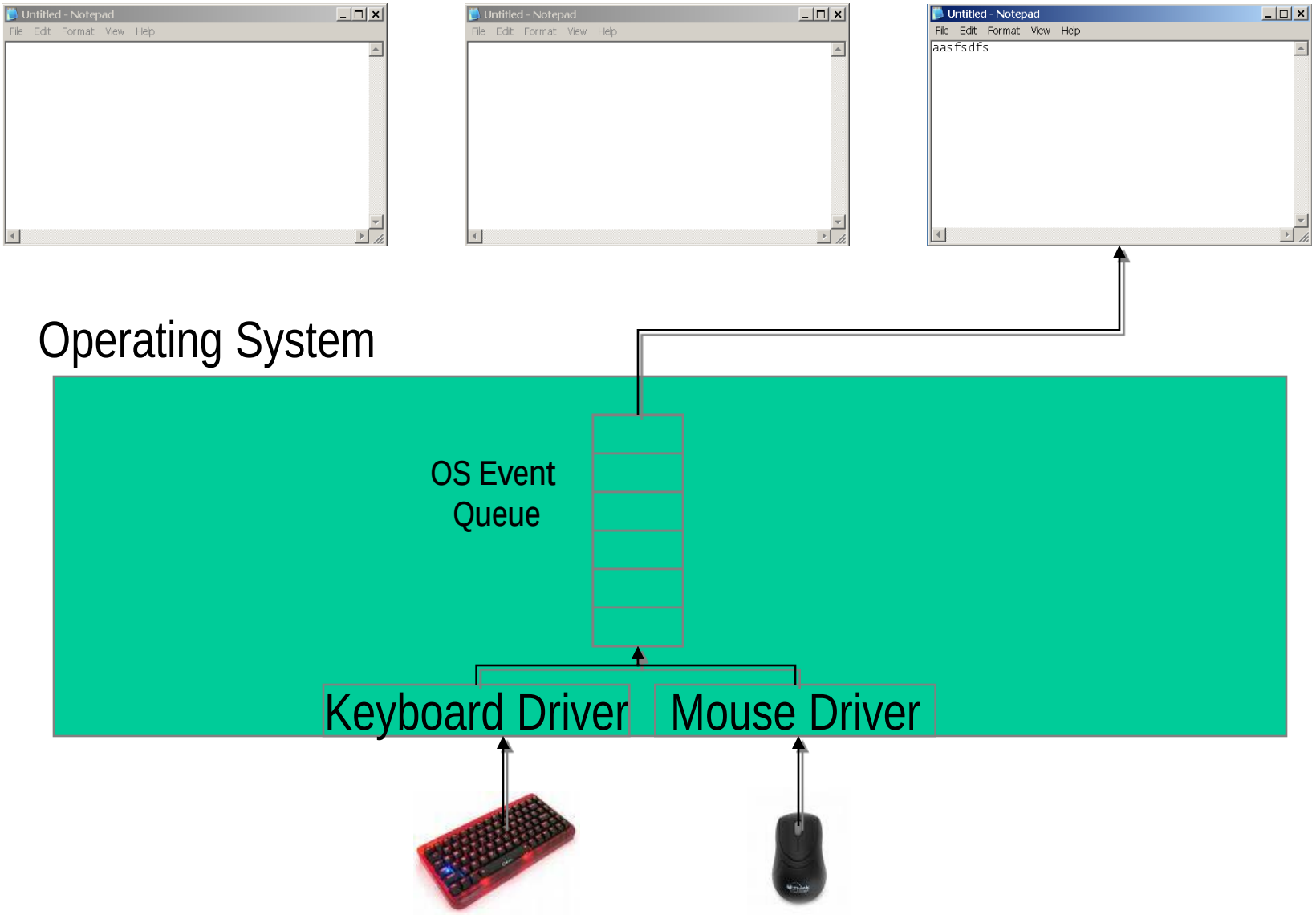


Interactive Graphical User Interface

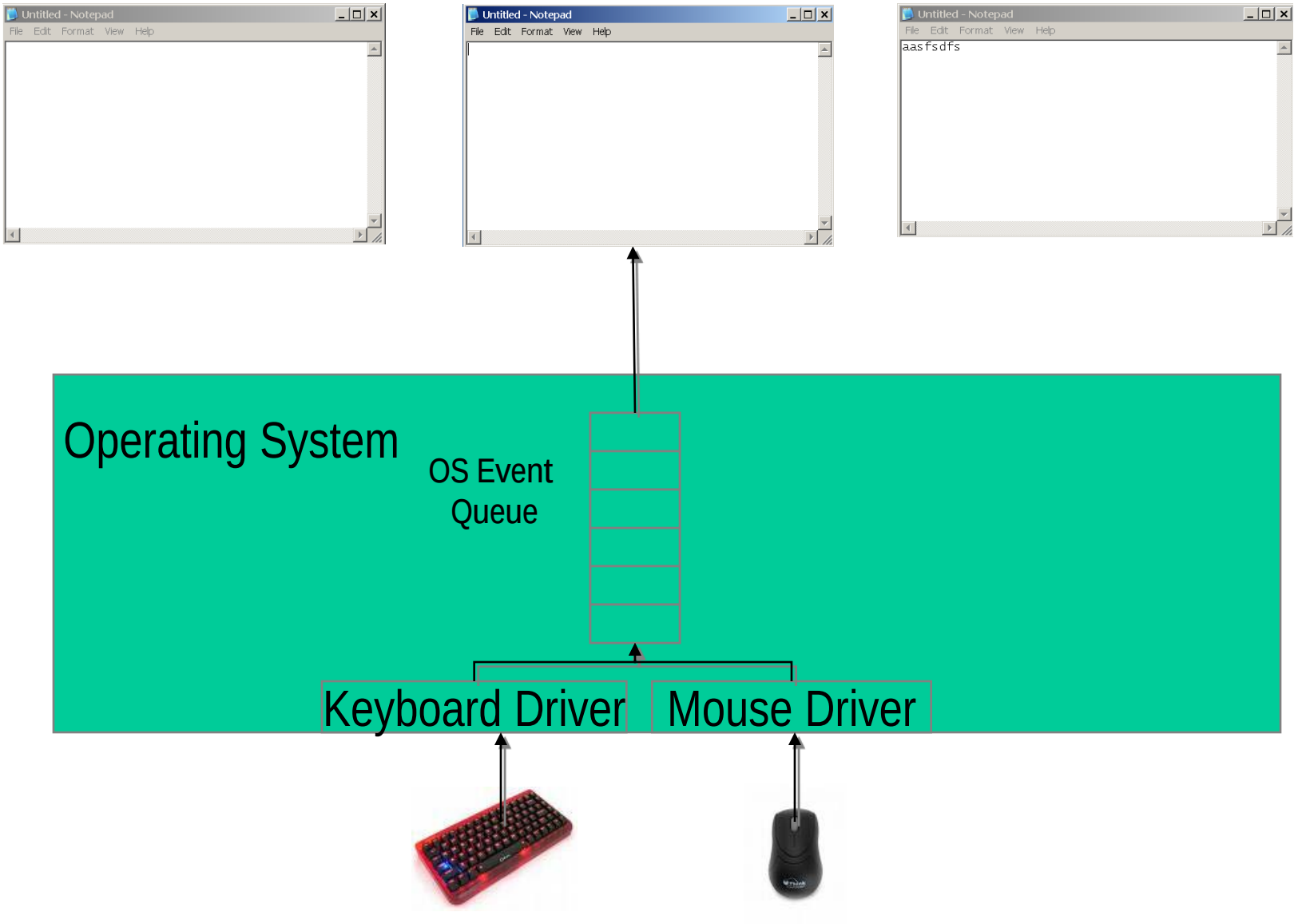
- What's make a GUI GUI?
  - Windows
  - Selection controls: drop-downs, radio-buttons, check boxes, menus,..
  - Activation controls: buttons, icons
  - Input controls: text fields, text areas
  - Structure information visually: lists, grids, trees, labels



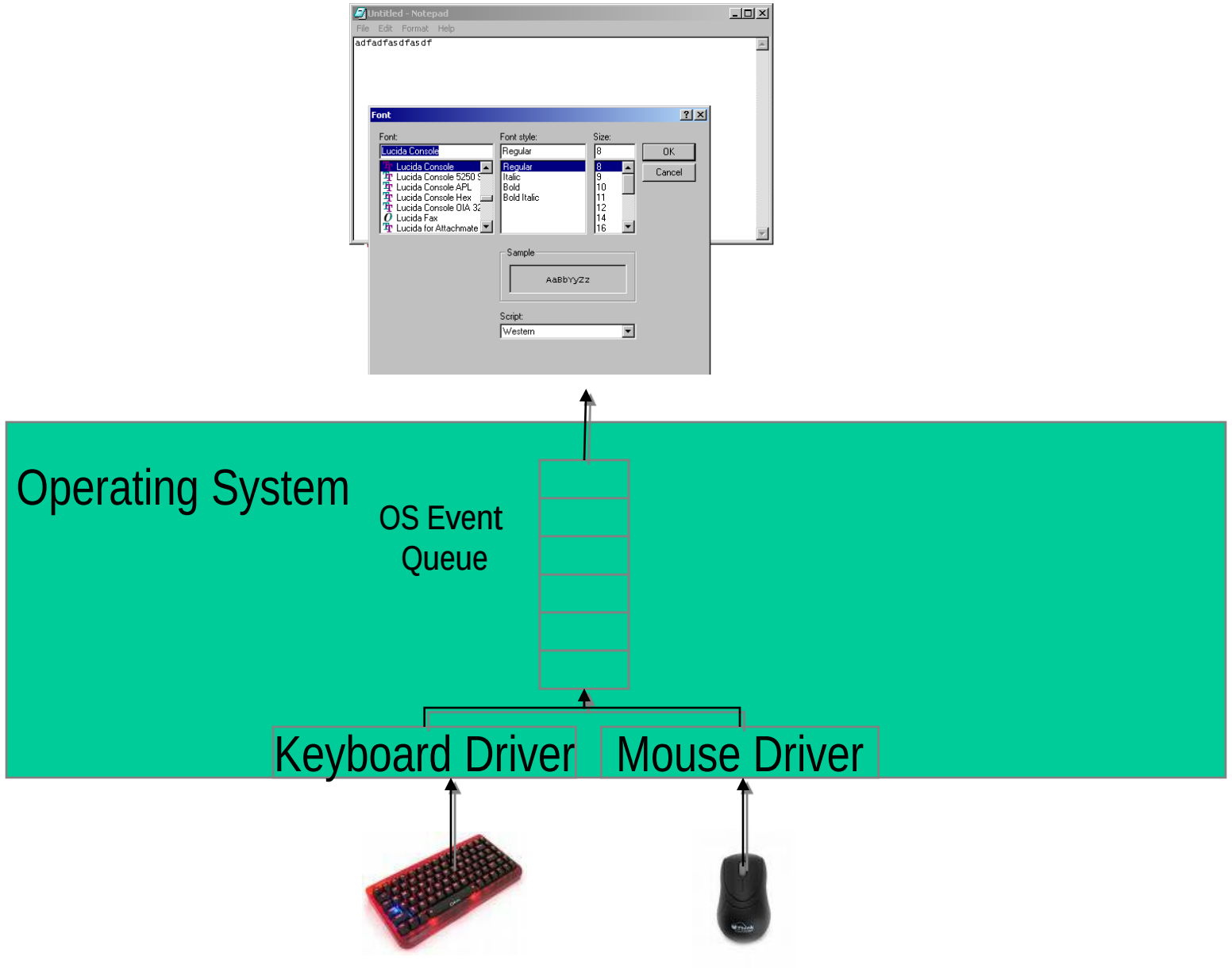
Input Events



Input Events

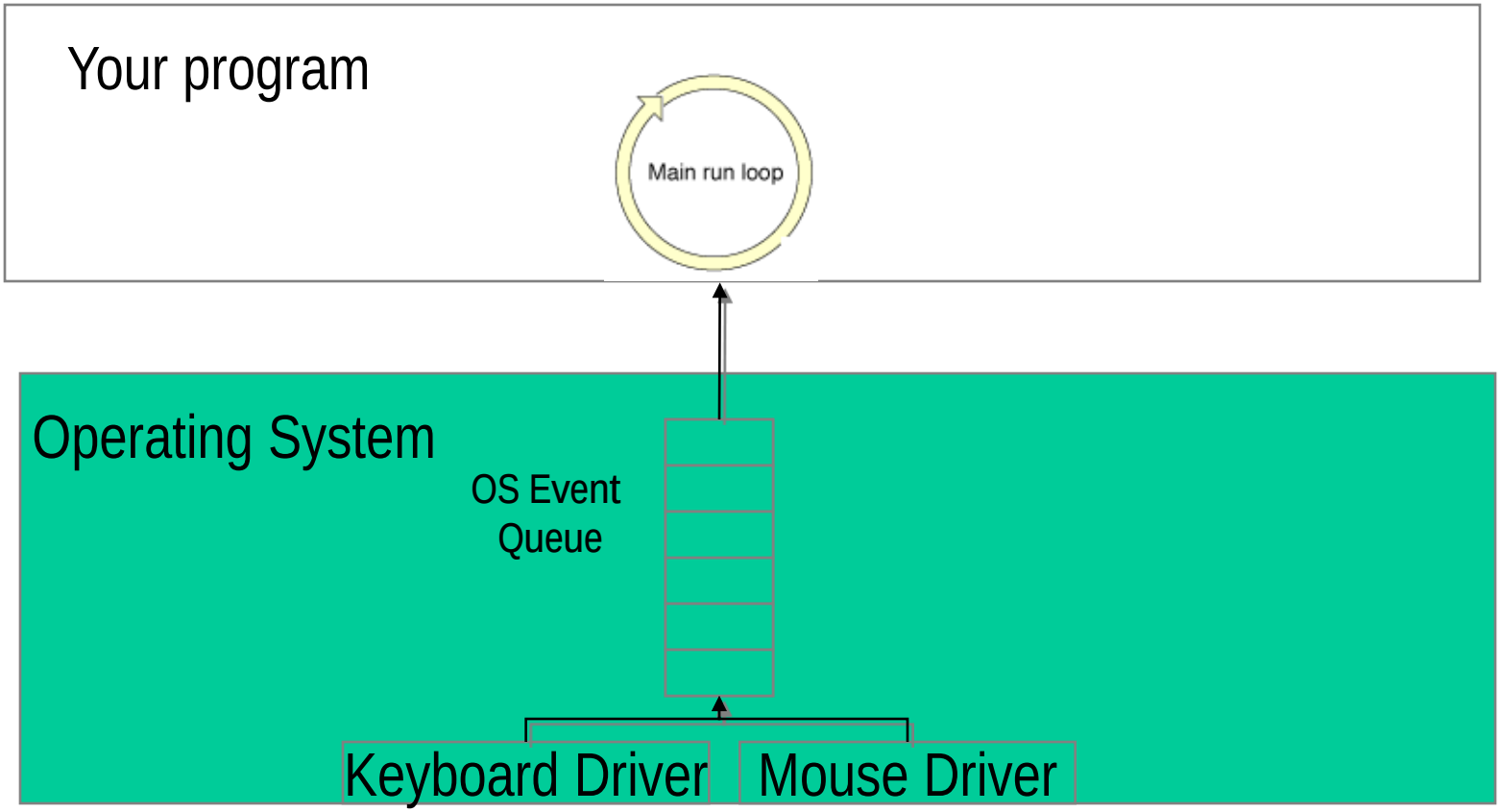


Input Events



Input Events: programming model

1. Use an infinite loop to keep checking the event queue
2. When you find the event you are interested in, execute the relevant code



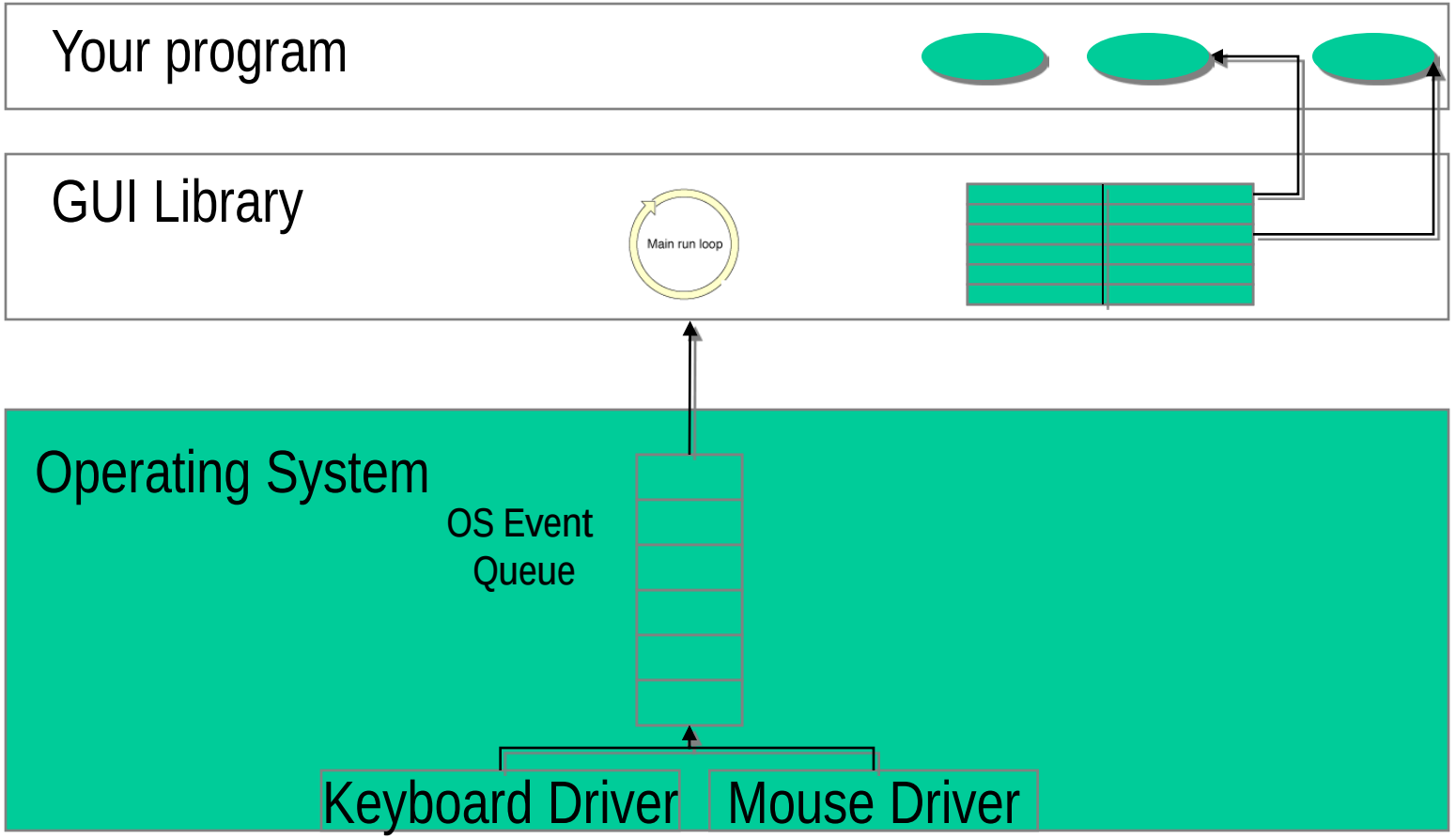
Event loop – pseudo code

```
int main() {
    return WinMain();
}

WinMain() {
  while (true) {  // loop forever, waiting for an event
    if (event_exists) {        //there is an event, figure out what to do
      if (event == keydown_a) display('user pressed the A key');
      else if (event == window_resize) display('window resized');
      else if (event == repaint) display('need to repaint window');
      else if (event == keydown_escape) exit_program();
    }
  }
}
```

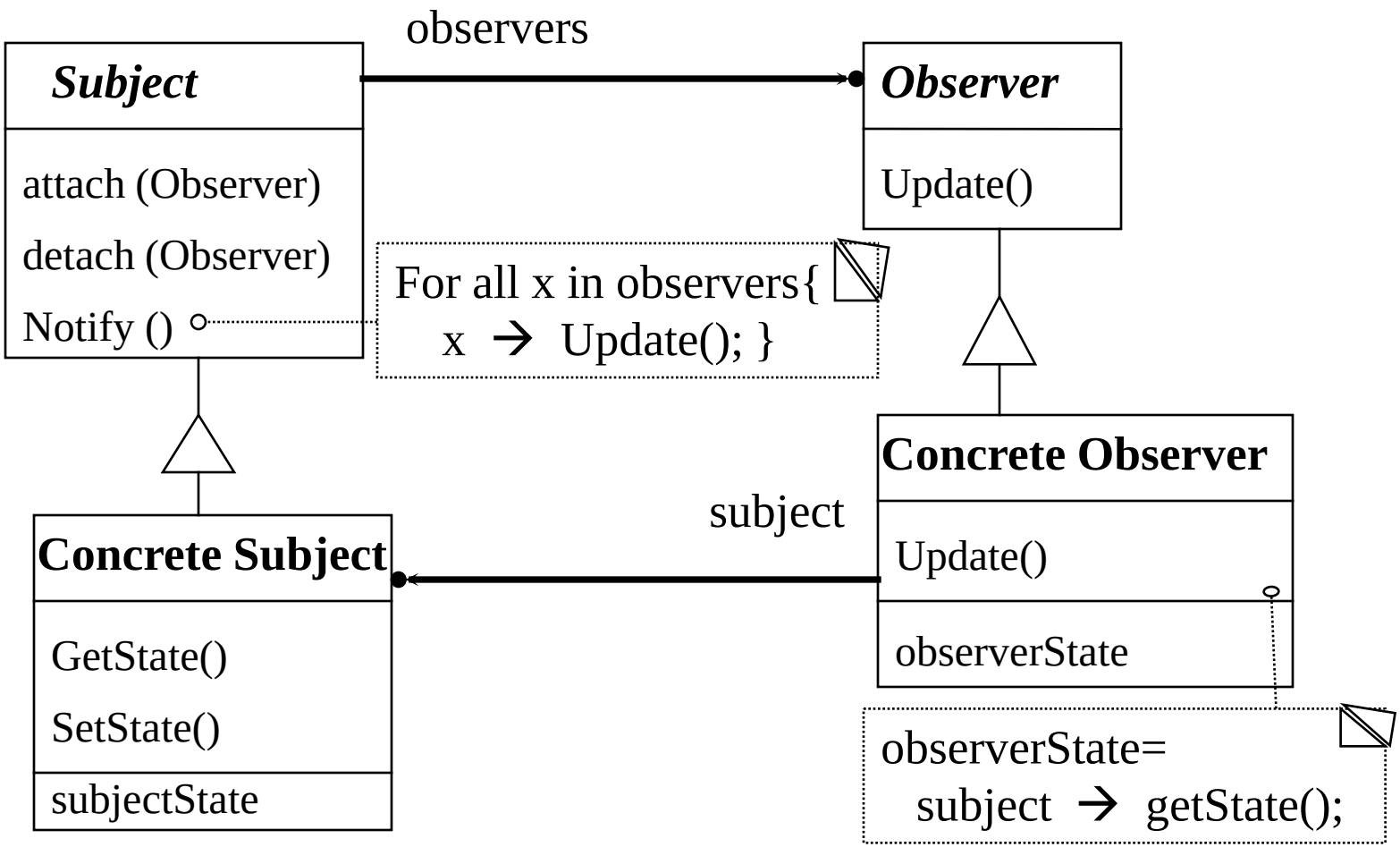
Input Events: programming model

Use an intermediate GUI library:

- specify specific events you are interested in.
- specify method/function in your code that should be called when an event you are interested in is received

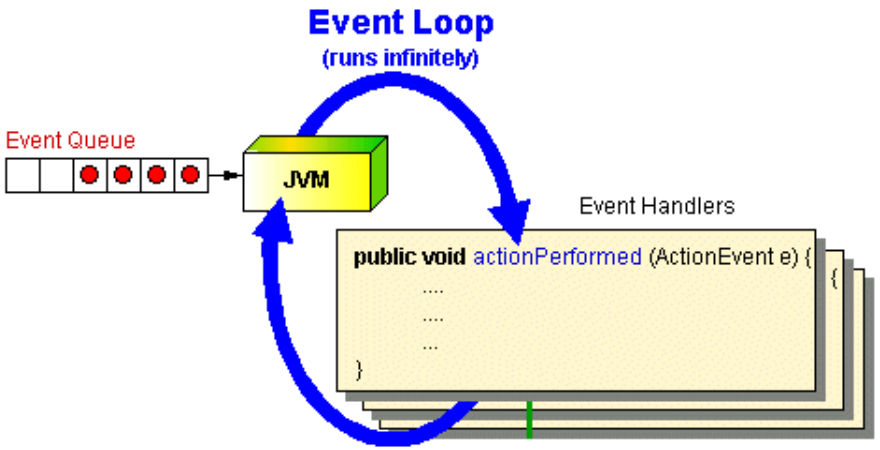


Observer Pattern



Java GUI program

- Event loop automatic in separate program

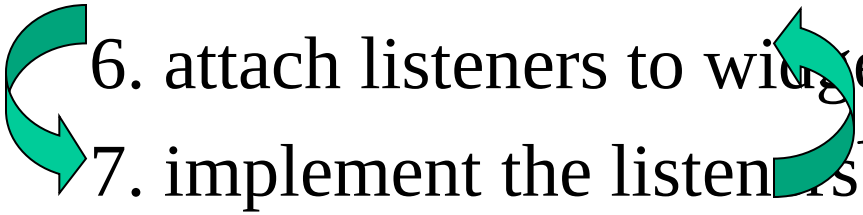


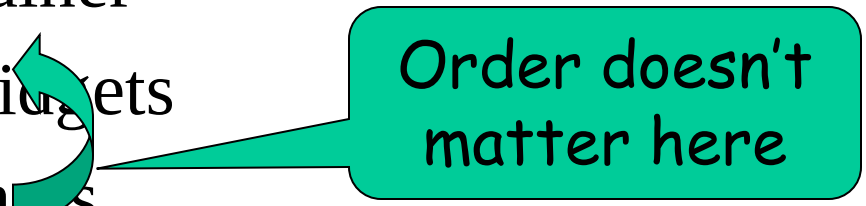
Java program:

```
Class{
main()
{
        decl data storage;
        initialization code;

        create GUI objects;
                register listeners;
}

listener1()
{       do stuff;
}
listener2()
{       do stuff;
}
...
```

Creating a GUI in Java

1. import Java libraries - swing, awt
2. construct a frame object
3. declare widgets - instance variables
4. choose, construct and set the layout in a container
5. add widgets to container
6. attach listeners to widgets
7. implement the listeners



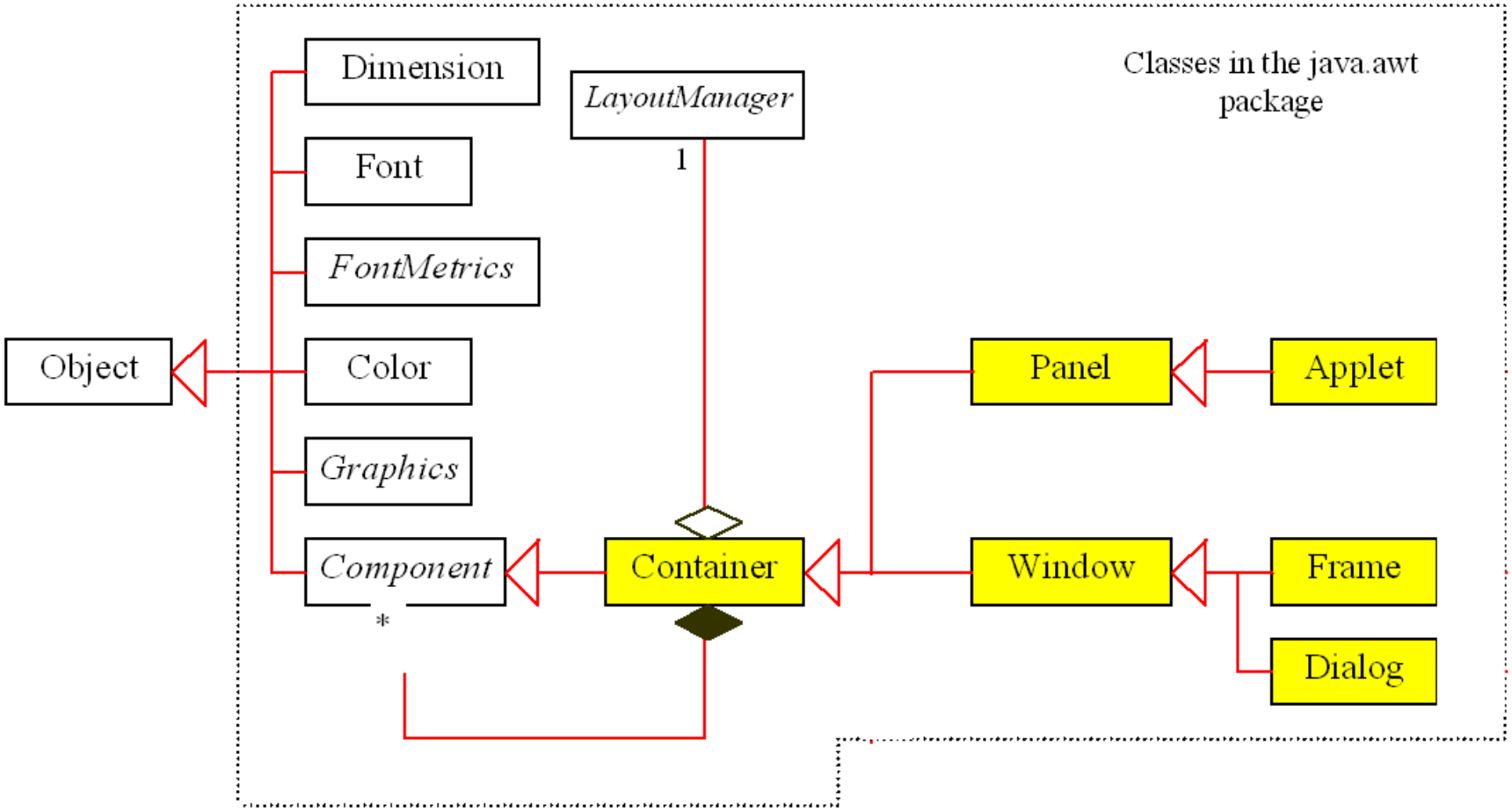


Frames vs. Dialogs

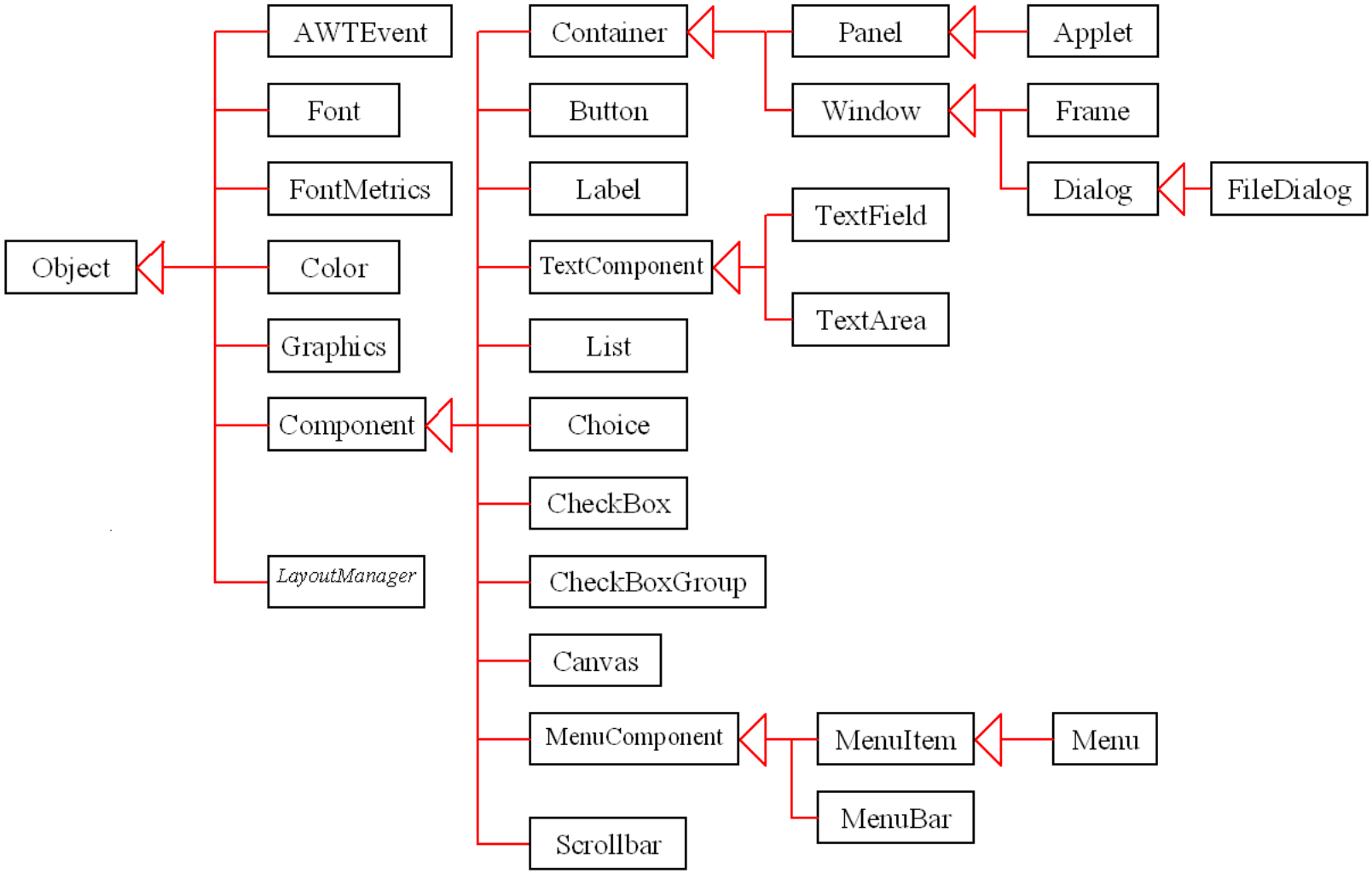
- Frame

- Dialog

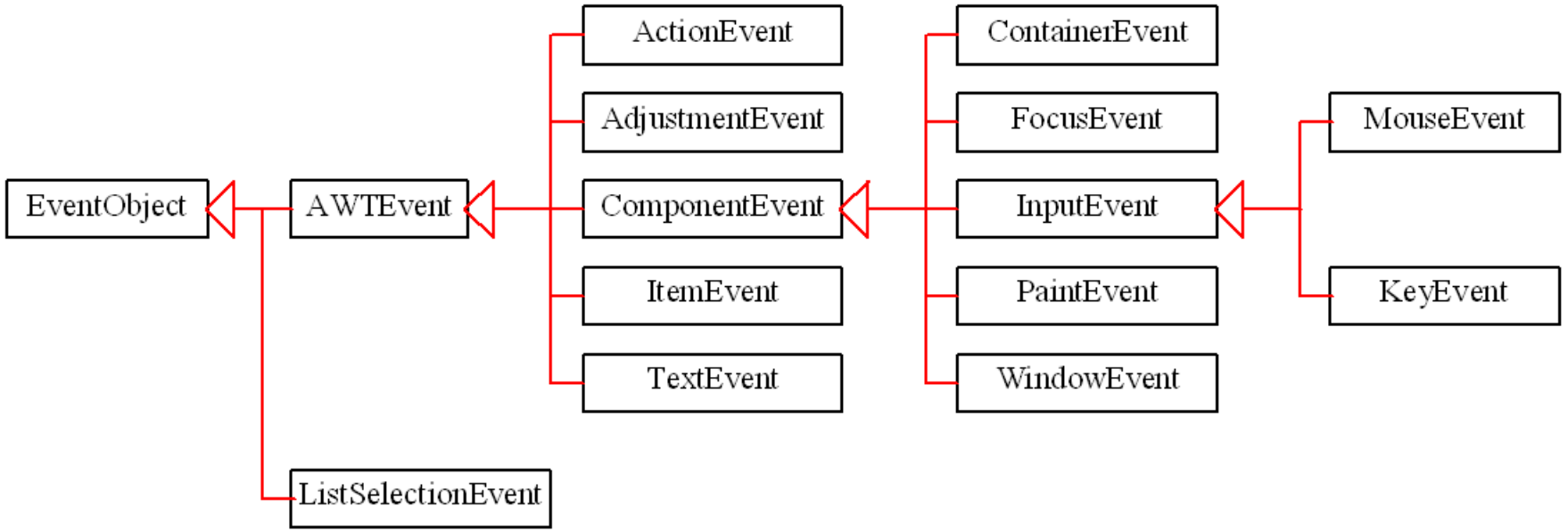
AWT Class Hierarchy: containers



AWT Class Hierarchy



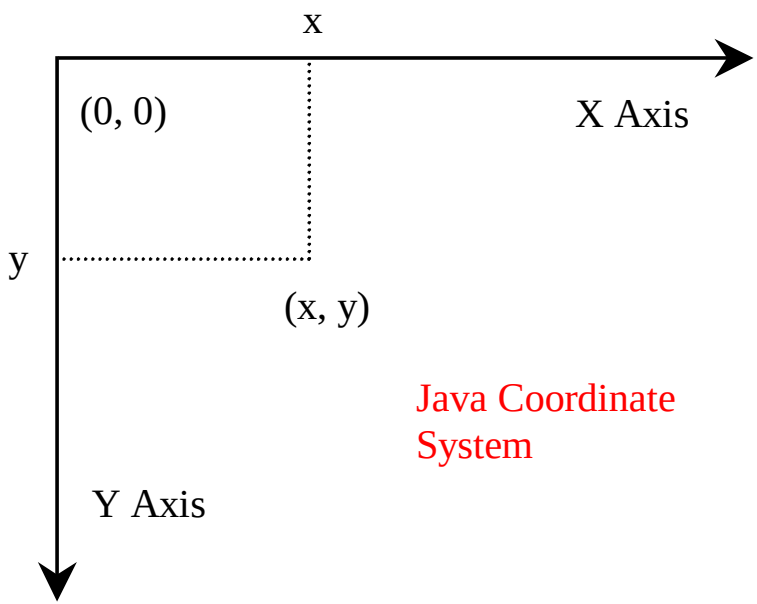
AWT Class Hierarchy: event classes



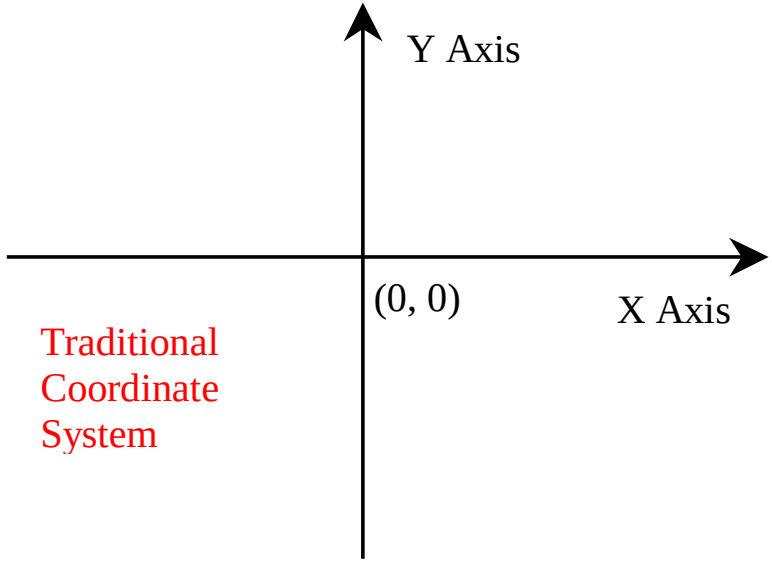
Event Types & Sources

User Action	Source Object	Event Type Generated
Click a button	`Button`	`ActionEvent`
Click a check box	`CheckBox/radio`	`ItemEvent, ActionEvent`
Press return on a text field	`TextField`	`ActionEvent`
Window opened, closed, etc.	`Window`	`WindowEvent`
Mouse pressed, released, etc.	`Component`	`MouseEvent`
Key released, pressed, etc.	`Component`	`KeyEvent`

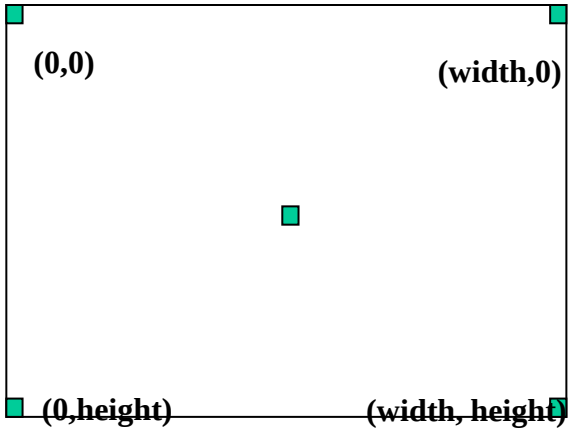
Java Coordinate System



Java Coordinate System

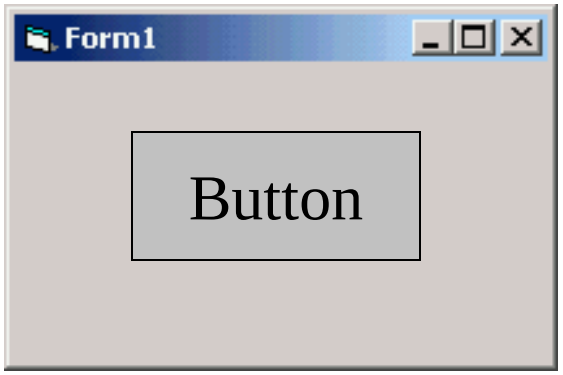


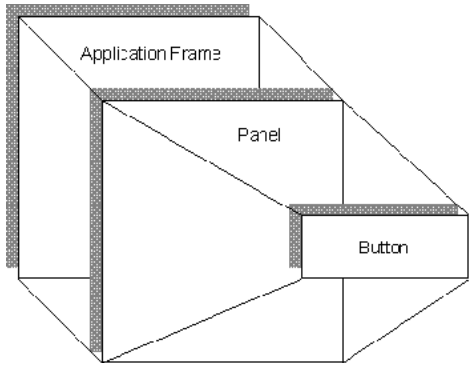
Traditional Coordinate System

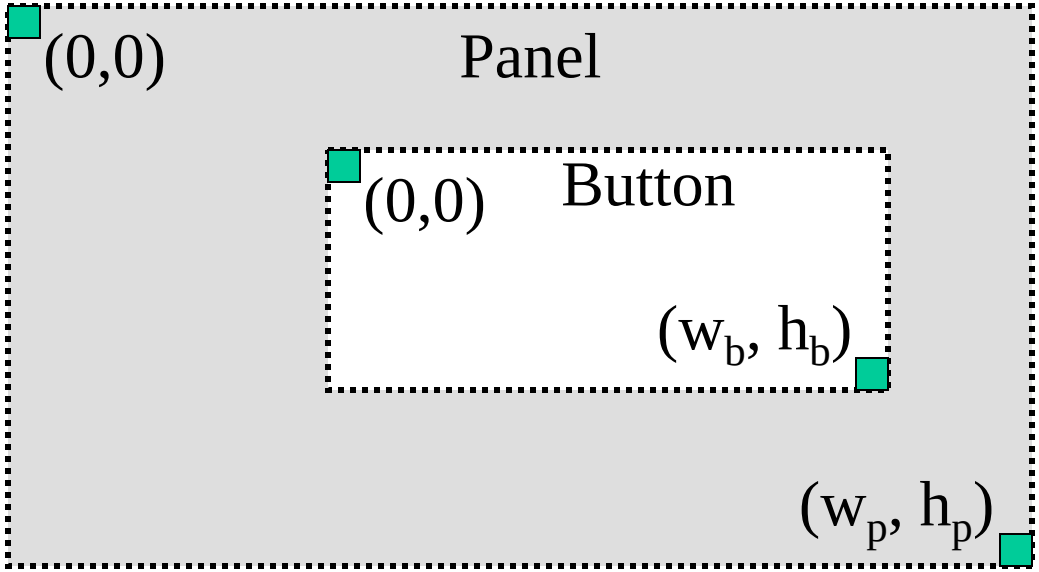


Component Hierarchy & Containers

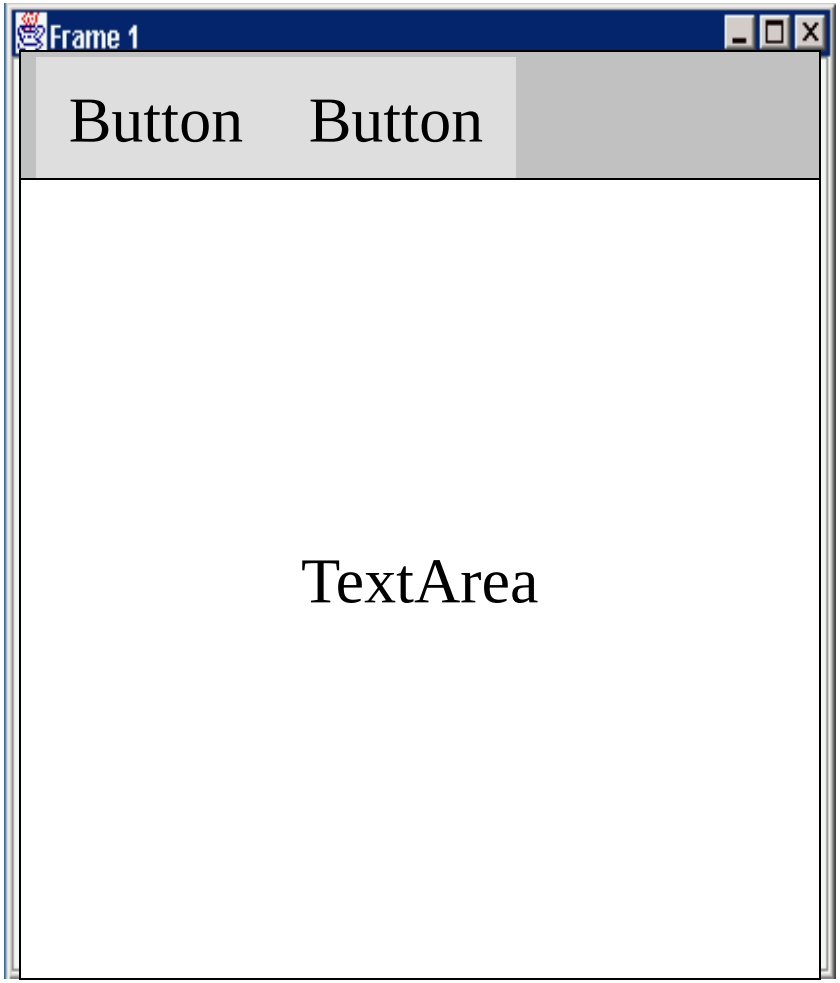
- You must add components to a container
- Each component has its own subwindow
  - Subwindow = rectangular area within parent component
  - Has own coordinate system





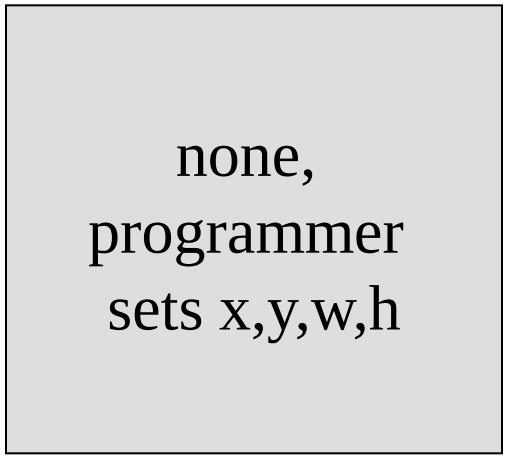


Example 1

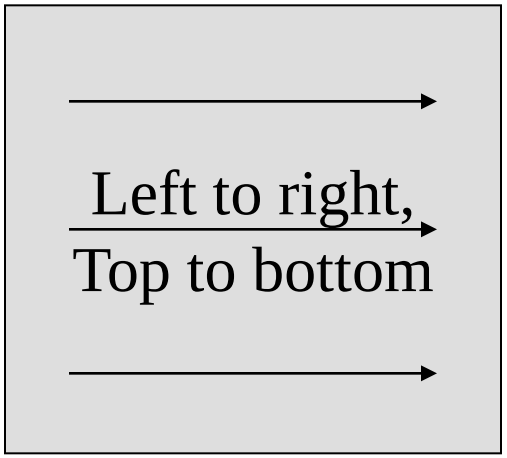


Layout Manager Heuristics

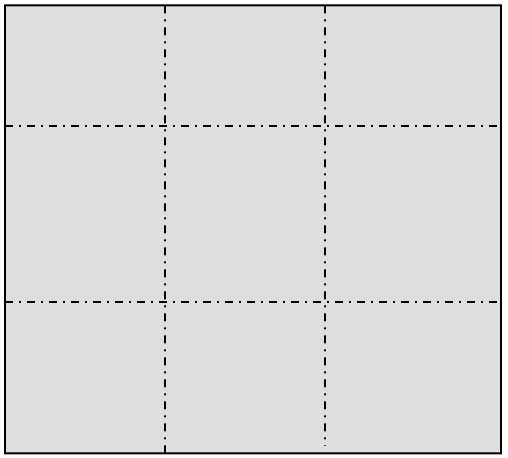
null



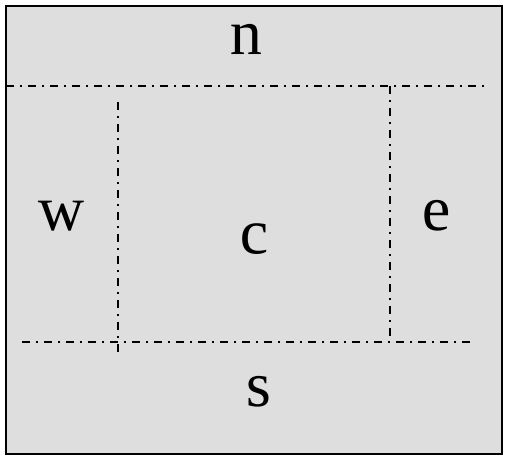
FlowLayout



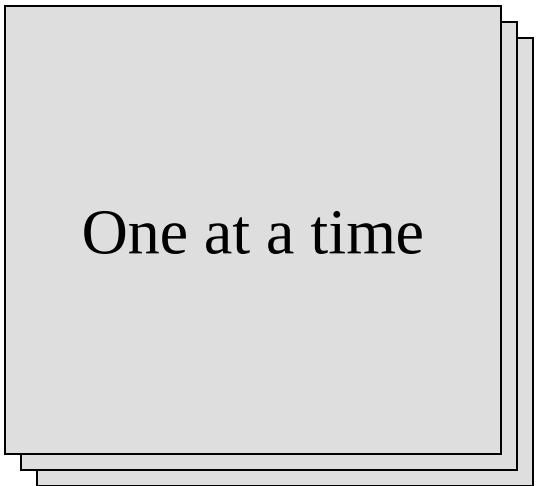
GridLayout



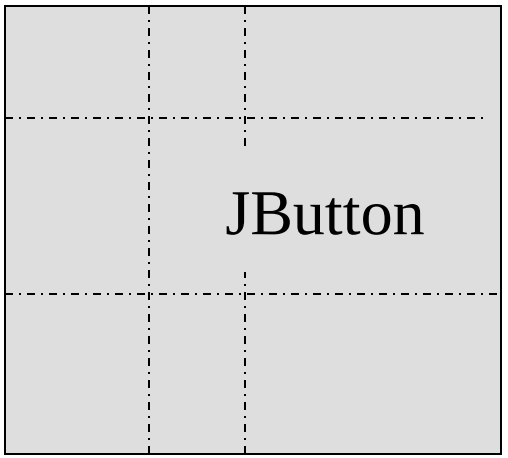
BorderLayout



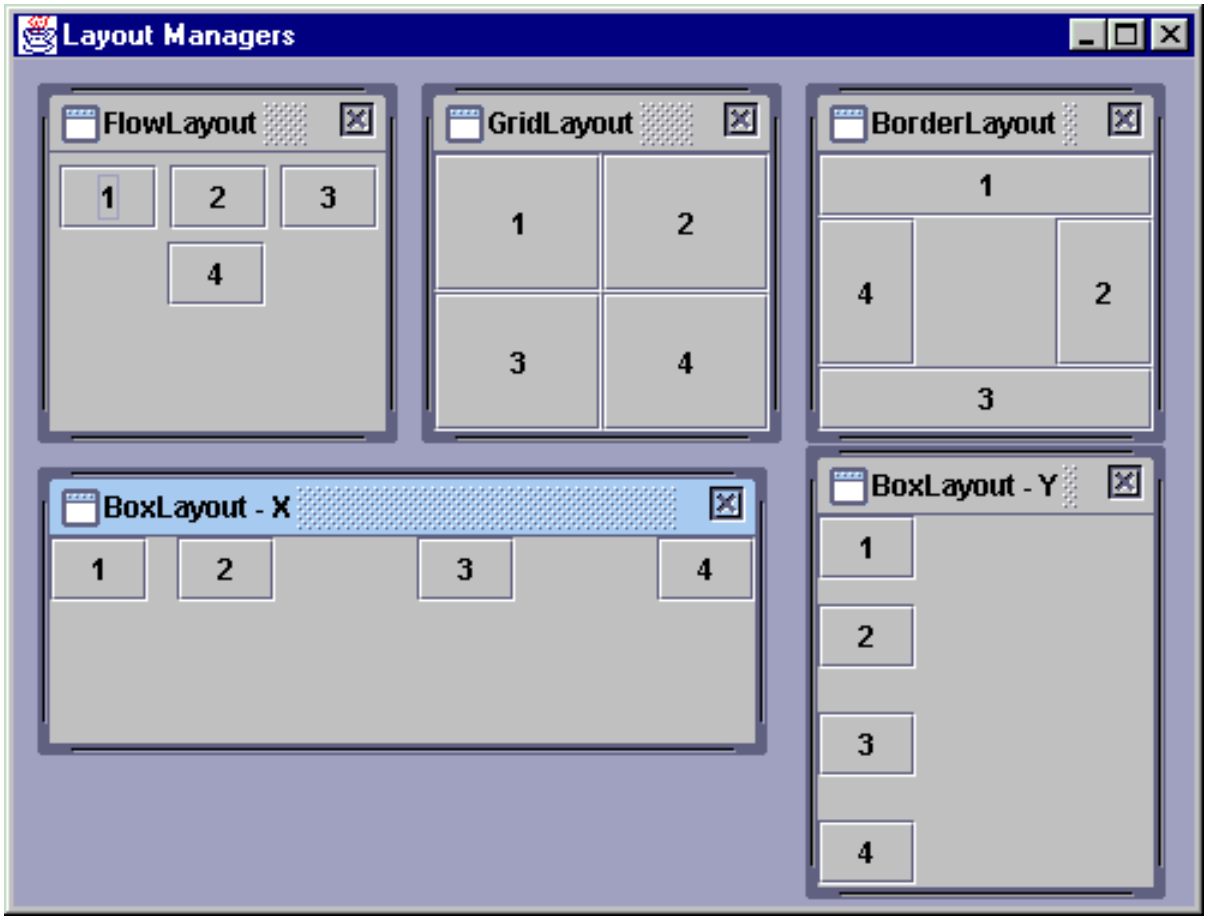
CardLayout



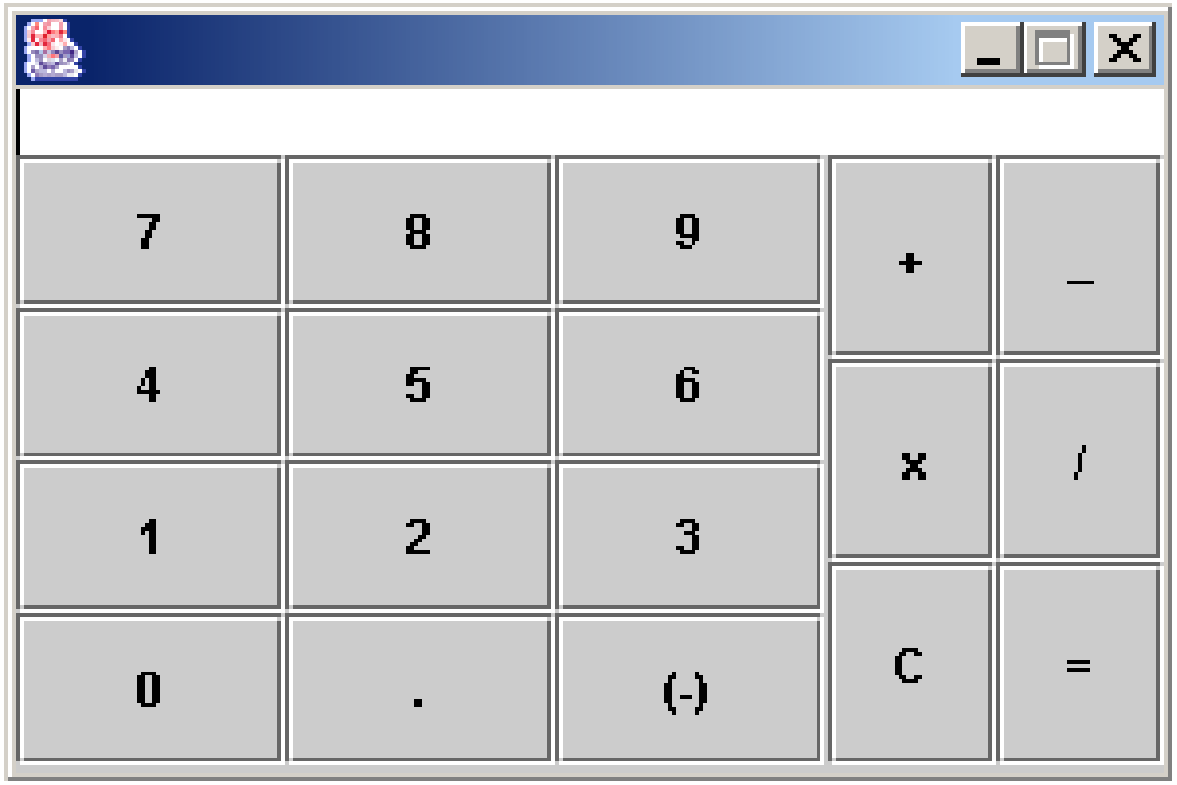
GridBagLayout



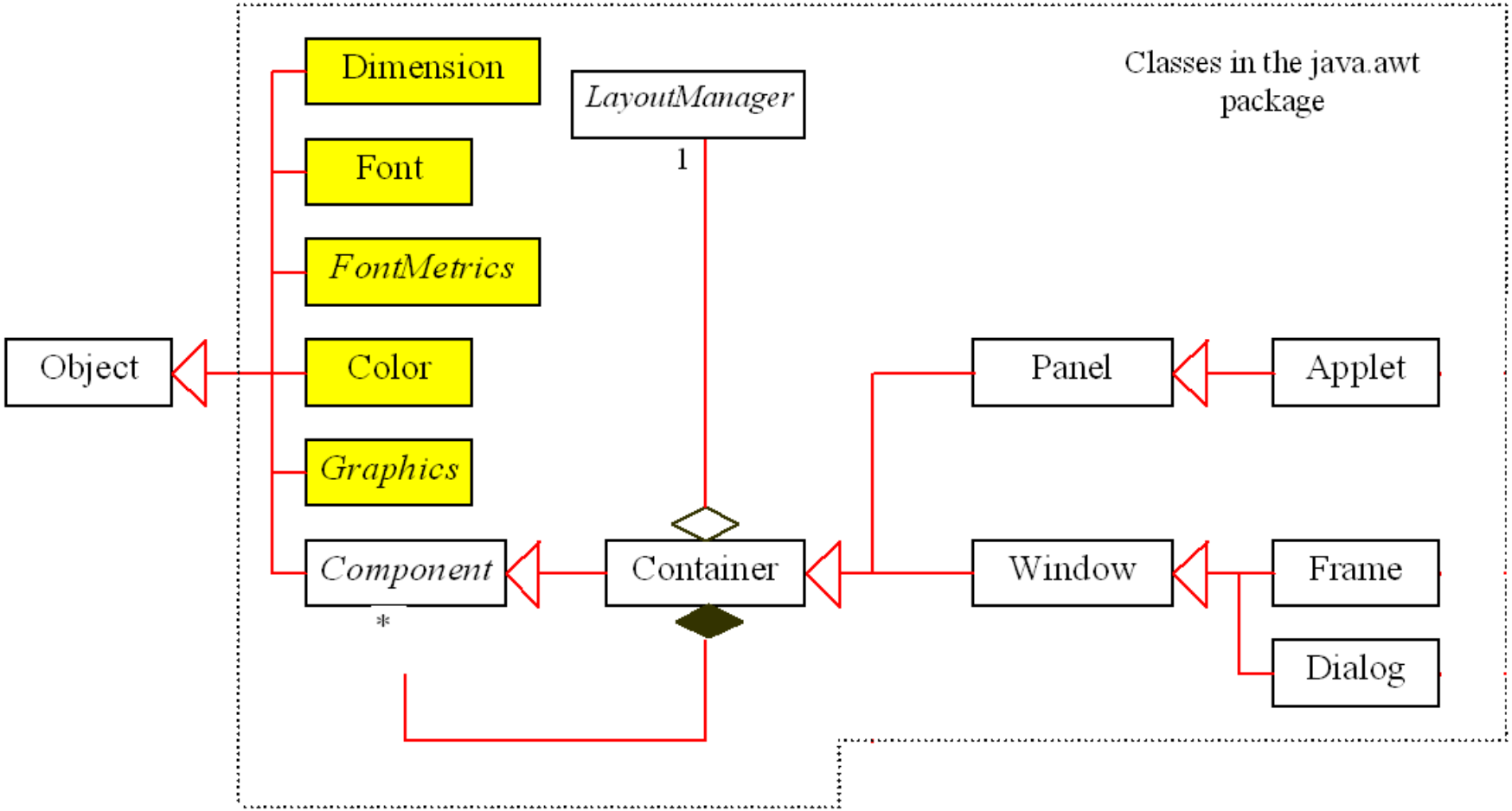
Example 2: Layout Manager



Example 3



AWT Class Hierarchy: helper classes

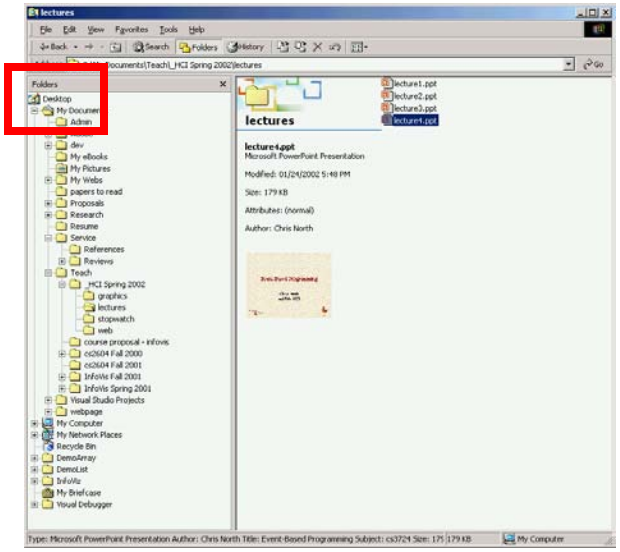


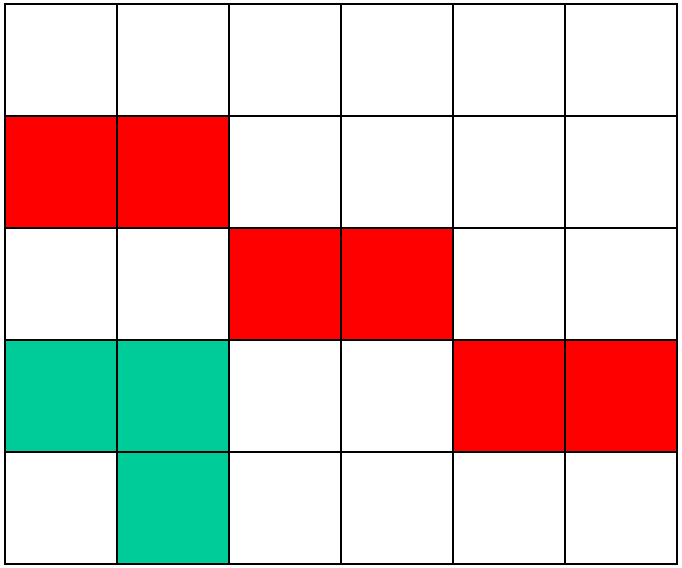
Graphics Handle

The `Graphics` object `g` is created automatically by the JVM for every visible GUI component.

This object controls how information is drawn. You can use various drawing methods defined in the Graphics class to draw strings and geometric figures.

Painting & Screen Pixels





Color

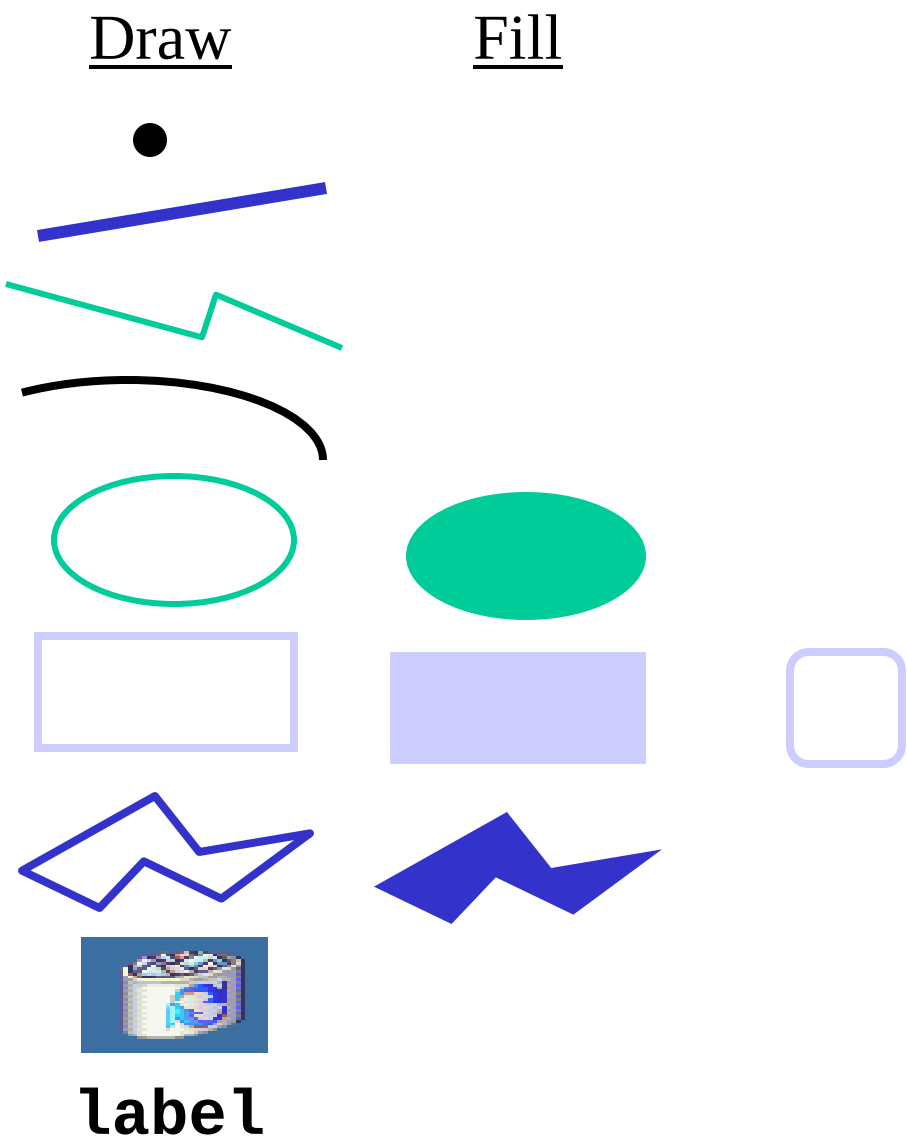
- All the components we have examined have methods to set the background and foreground colours using *setBackground(Color)* and *setForeground(Color)*
- The *Color* object allows various ways of specifying a colour
- There are a set of predefined colours Color.blue etc
- You can specify the RGB values for the colour

  Color c=new Color(10,10,10);

Red, green,blue

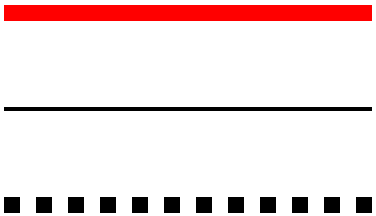
Graphics Primitives

- Point (x,y)
- Line (pt1,pt2)
- PolyLine (pt list)
- Arc
- Oval (pt, w,h)
- Rectangle (pt, w,h)
  - RoundRectangle
- Polygon (pt list)
- Image (file, x,y)
- Text (string, x,y)



Graphics Attributes

- Color
- Font
- Stroke attributes:
  - Line width, dash, end caps, joins, miter
- Paint attributes:
  - Color, gradient, texture
- Composite:
  - Blending
- Transforms:
  - Translate, rotate, flip, shear, scale



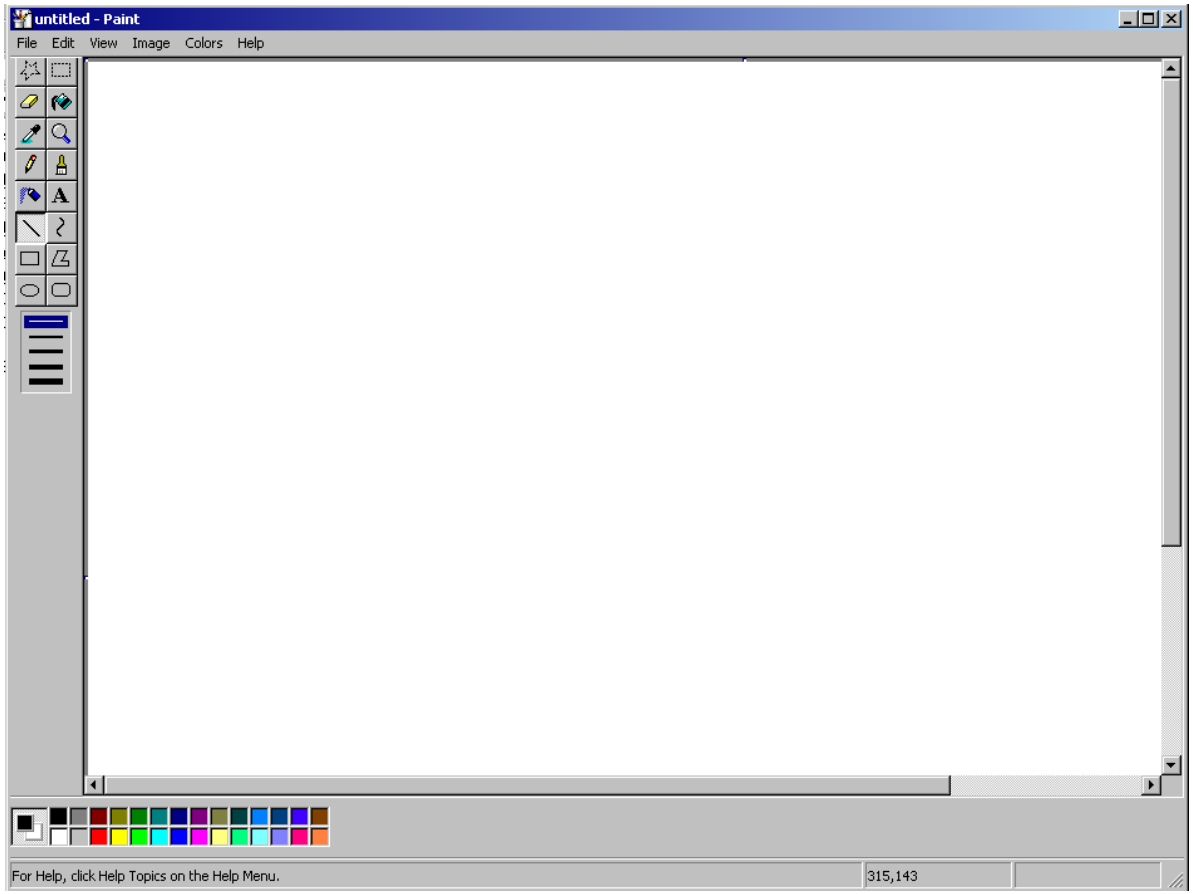
Painting in Java

```
import java.awt.Graphics

public paint (Graphics graphic )
{
   // put your painting code here

}
```

Drawing Geometric Figures and Strings

- Drawing Lines, `e.g. graphics.drawLine(50,50,100,200);`
- Drawing Rectangles, e.g. graphics.`drawRect(20,20,100,50)`
- Drawing Ovals, e.g. graphics.`drawOval(30,30,50,50);`
- Drawing Strings, e.g.
  `graphics.drawString(Hello”, 50,50);`

Example 4


untitled - Paint
File Edit View Image Colors Help
For Help, click Help Topics on the Help Menu.
315,143