

# GUI (gooey)

## Lecture 1(22/8/2024):

- Implement via swing package
  - Import javax.swing.[ClassName];
  - Import java.awt[ClassName];
- Event-driven programming:
  - Programming style based on signal-and-response approach.
  - (E.g. mouse clicks, key presses)
  - Event: the change in the state of an object or behavior by performing actions
  - Examples:
    - User presses a key, mouse click, close window, press button
- How:
  - GUI components send events to *listeners*.
  - Event: object that acts as a signal to a listener.
  -
- Listener: an object that performs some action in response to an event
  - A component may have several listeners.
- Event handler:
  - Methods of the listener object that specify what happens when events are received.
  - [ E.g. checkUsername() and checkPassword() will be the event handlers, when the button “login” is pressed ]

A diagram of a login form. It consists of two light gray rectangular input fields. The top field is labeled 'username' and the bottom field is labeled 'password'. Below these fields is a solid green rectangular button with the word 'LOGIN' in white capital letters. Underneath the button is a link that says 'Not registered? Create an account' in a smaller, green font. A thin green line is positioned to the left of the input fields and button, extending from the top of the form down to the 'Examples' section.

- 
- Examples:
  - Click button --> play sound

- Click button --> show image
  - Click button --> close window
- Common classes:
  - Import javax.swing.<ClassName>;
    - JFrame (window)
      - JFrame window = new JFrame("Demo Program for JFrame");
    - JButton (button)
    - JLabel (text)
      - JLabel label1 = new JLabel("Hello World")
- Important methods:
  - Window.setSize(x,y) [ sets the window size in pixels ] (e.g. 1280x720, 1920x1080)
  - Window.setVisible(true) --> This line shows the window.
    - The reason why some are hidden, is so that you don't overload users with multiple windows at once.
    - E.g. when you start the application, you open only the "Main Menu"
    - Then when you click a button, it opens a new window, and "hides"/closes the main menu.

## **Lecture 2(29/8/2024):**

- Today: more GUI components (text fields, menu) & How to arrange GUI components

GUI components:

- Text field: allows user to enter single line of text
  - `TextField in1 = new TextField("Enter name");`
  - `TextField in2 = new TextField(20);`
- Menu (JMenu)
  - A choice on menu = menu item --> (JMenuItem)
  - Menu bar: container for menus, typically placed near top of window interface
- Components learned:
  - `JButton`; `TextField`; `JMenu`; `JMenuItem`

## **Layout Managers:**

- Positions components inside the containers
- Component vs Container:
  - Component refers to basic elements of GUI
- *BorderLayout* manager: places components into 5 regions (North, South, East, West, Center)
  - One component per region
  - Center region expands to take up unused space
- *FlowLayout* manager: simplest manager
  - Arranges components one after another; from left to right in the order that components are added
- *GridLayout* arranges components in 2D grid, with rows & columns
  - If you see: `GridLayout(rows,0)`
    - Grid with specified number of rows, and as many columns as required
  - `GridLayout(0,3)`, then add 6 components:

- 2 rows, 3 columns ( $2 \times 3 = 6$ )
- GridLayout(2,4), then add 2 components:
  - 2 rows, 1 column (we only need 1 column to fit components)
- GridLayout(2,0), then add 5 components:
  - 2 rows, 3 columns ( $2 \times 2 = 4$  won't fit, but  $2 \times 3 = 6$  will)

### **Action commands:**

- When user clicks a button or item, an event is fired that goes to one or more action listeners.
- Action event includes a string instance
- Component vs Container:
  - Component refers to basic elements of GUI

## Lecture 3 (12/9/2024):

- Today: colours, fonts
- Demo: combining everything so far

### Colors:

- An object of the class Color (from java.awt package)
- JFrame can't be colored directly
  - The content pane will be customized
    - `Frame.getContentPane().setBackground(Color.BLUE);`
  - Components:
    - `Button.setBackground(Color.PINK);`
- RGB color system
- **Only** Integers and Floats can be used for the constructors of Color. No Doubles!!
- Integers must be in the range 0–255 inclusive
- float values must be in the range 0.0–1.0 inclusive
  - WRONG:
    - `new Color(0.0, 1.0, 2.0);`
  - CORRECT:
    - `new Color(0,155,255);`
    - `new Color(0.0f, 0.5f, 1.0f);`
  - Float = 32 bits || Double = 64 bits
  - Therefore, double can represent larger numbers

### Fonts:

- Object of the class Font (from java.awt)
- Constructor creates a font:
  - `Font font1 = new Font("SansSerif", Font.PLAIN, SIZE);`
    - Size: the font size
- Serifs are small lines that finish off the ends of the lines in letter
- To set the font for a Swing component, use its `setFont()` method.
-

## **Lecture 4(19/9/2024):**

- Today: Window Listeners, icons
- Window listener: handles events fired by the window manager
  - Opening, Closing, Minimizing, Maximizing, Deactivating, Activating
  - 
  - Defined: *public class ClassName implements WindowListener*
  - All seven methods need to be defined, but you can define their bodies as empty
- Icon: small images
  - Object of the ImageIcon class
  - *ImageIcon wavingIcon = new ImageIcon("waving.gif");*
  - 
  -

## **Lecture 5(26/9/2024):**

- Recap: Inheritance
  - Base/Parent/Super class
  - Derived classes inherit methods, variables from the base class
    - They can:
      - add additional instances, methods
      - Override methods
- Graphics class (found in java.awt\*)
  - Every component, element drawn on screen has a graphics object
  - Java uses a coord system (x,y axis); where (0,0) is the top left corner
    - X increases as you go right across screen    --> = + ; <-- = -
    - Y increases as you go down across screen    V = + ; ^ = -
    - ~
  - The *Paint* method draws elements on screen.
    - *Public void paint(Graphics g)*

- Repaint: changes graphics content of a window
- *setColor*: changes the colour of the pen
-