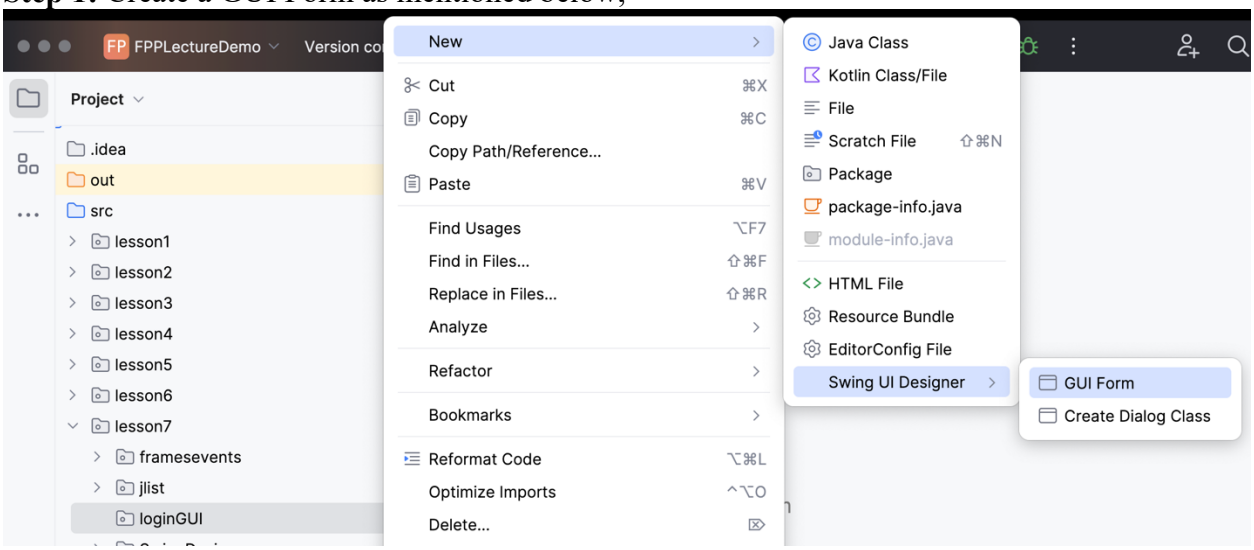
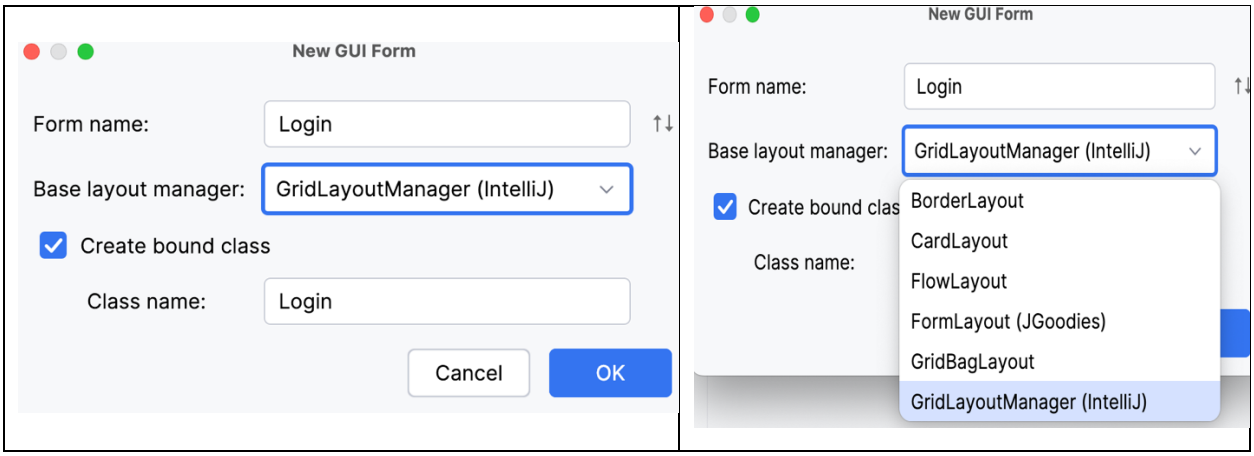
Lesson – 7 - Java Swing Designer GUI Form in IntelliJ

Reference: https://jetbrains.design/intellij/
https://www.jetbrains.com/help/idea/designing-gui-major-steps.html

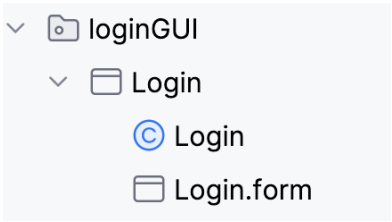
Step 1: Create a GUI Form as mentioned below,



Step 2: Name the form and class file. You can give the same name for both or can change it. I used name as Login. You can choose the layout that you want for your Application.



Step 3: You will see the Login.form and Login.java files.



Step 4: To execute your Application, you must add the following code to every Application.

You can see the below code from Login.java

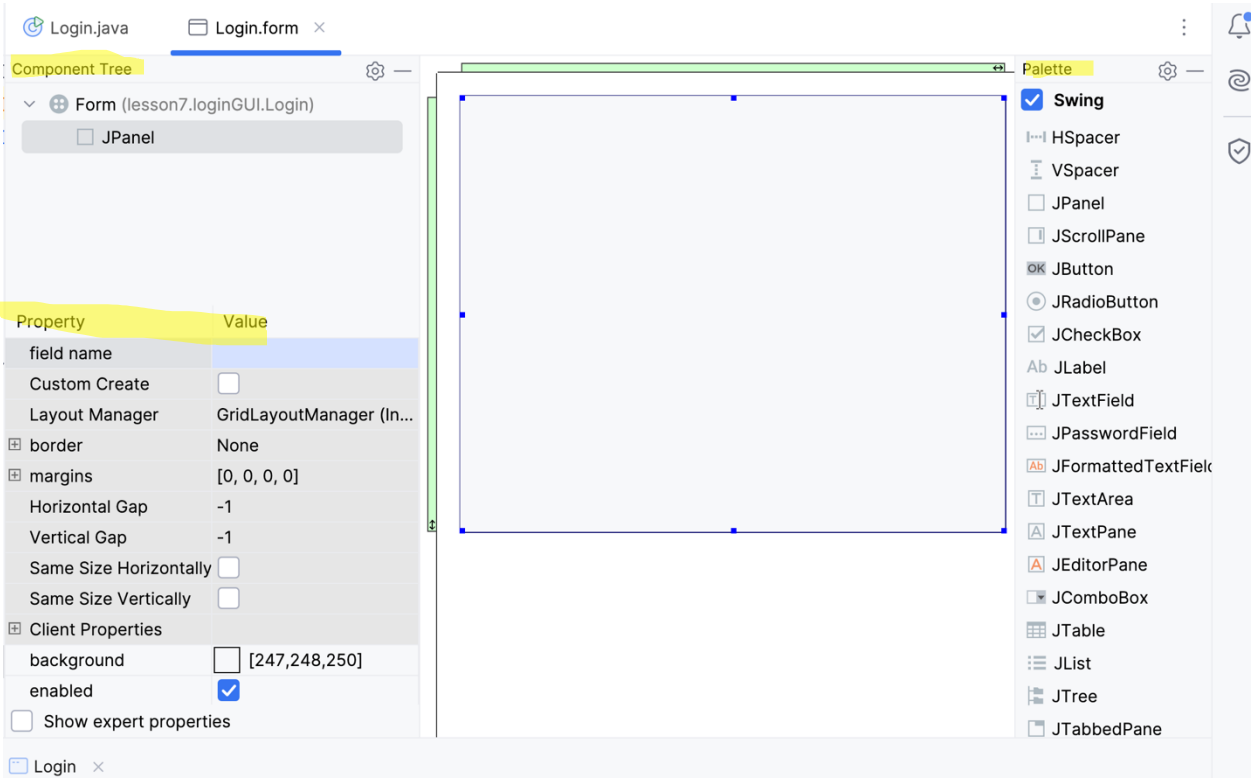
```
public class Login {
}
```

After adding the Mandatory code to get the JFrame. The same code you can add for all the programs using JFrame.

```
import javax.swing.*;
import java.awt.*;

public class Login extends JFrame {

  public Login() {
      // Default visibility is false. You have enabled visibility true
      setVisible(true);
      // Terminates the Application when the frame is closed.
      setDefaultCloseOperation(EXIT_ON_CLOSE);
      setTitle("Login Form");
      // Provide the frame width and height
      setSize(300, 400);
      // Make your screen center
      setLocationRelativeTo(null);
      setResizable(false);// If you wish

   }

 public static void main(String[] args) {
      /*
      While it is not mandatory to use EventQueue.invokeLater,
      it is a best practice for all Swing applications to ensure
      thread safety and avoid potential concurrency issues.
       */
      EventQueue.invokeLater(new Runnable() {
         public void run() {
            Login mf = new Login();
         }
      });
   }
}
```

Step 5: Now you can design the Login.form

JPanel



Left Side

Component Tree – Hold all the components added in the JPanel – Root Component

Property Window – Can modify the properties of the components to give a field name to use it code, change background color border, etc.,

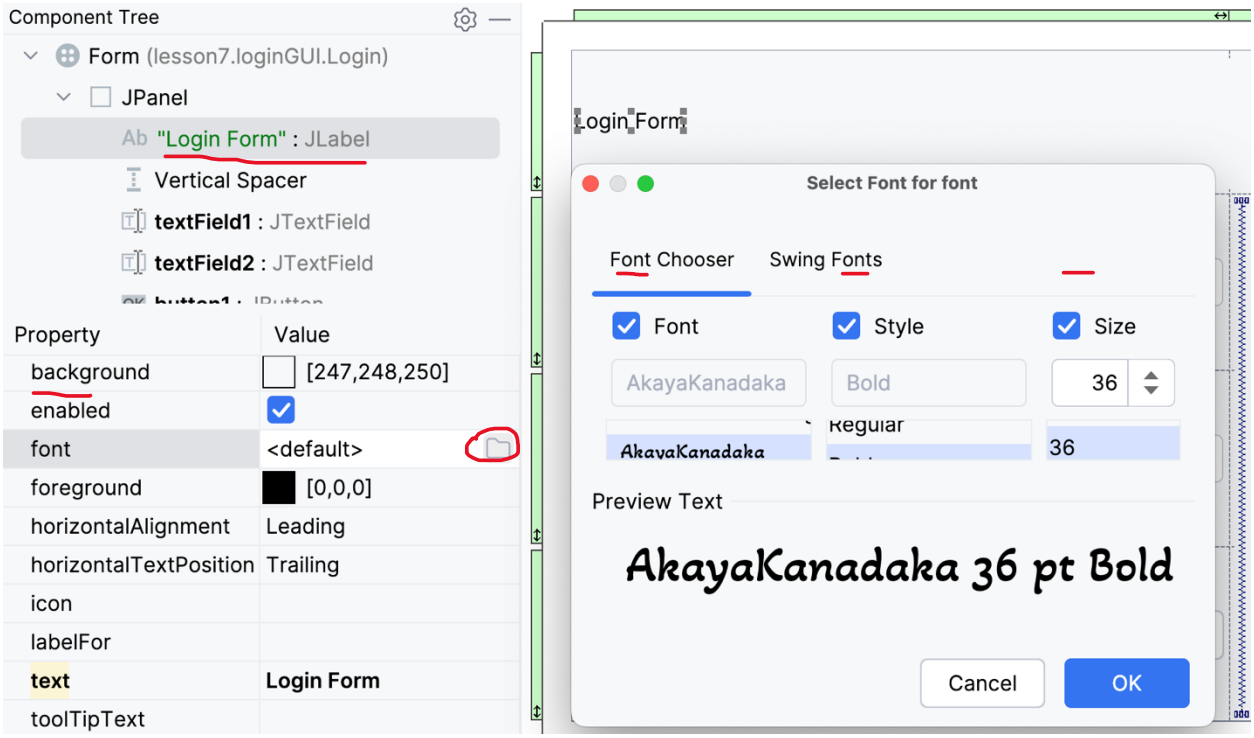
Middle

JPanel to add contents/components to your form

Right Side

Palette: Shows all the components you can add to your form. Simply select and drop it in the panel.

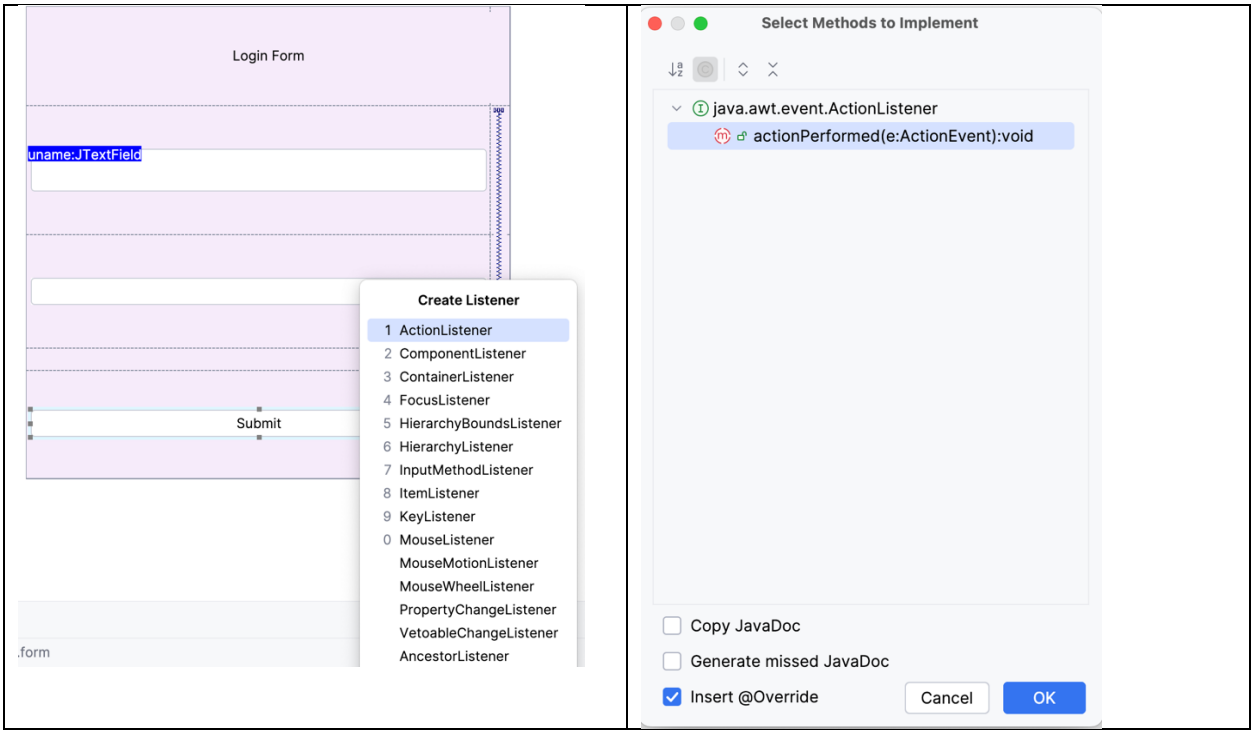
Step 6: Customizing
Label Customization is done with Font



Button Customization. Must provide field name, text

Property	Value
field name	**submit**
Custom Create	
Horizontal Size Policy	Can Shrink, Can Grow
Vertical Size Policy	Fixed
Horizontal Align	Fill
Vertical Align	Center
Indent	0
Minimum Size	[-1, -1]
Preferred Size	[-1, -1]
Maximum Size	[-1, -1]
Client Properties	
background	[223,245,255]
enabled	
font	**20 pt Bold**
foreground	[0,0,0]
hideActionText	
horizontalAlignment	Center
horizontalTextPosition	Trailing
icon	
text	**Submit**

Step 7: Adding Event Handling to the Button. Right click the component needed to perform action listener > Create Listener. Choose the type of Listener you prefer. Here, we will discuss ActionListener. Click OK. It will generate the Anonymous implementation inside your file.



This is the generated code.

```
submit.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {

    }
});
```

You have learned inner class in chapter 6. You can implement the listener as a member of the inner class, local inner class, or Lambda.

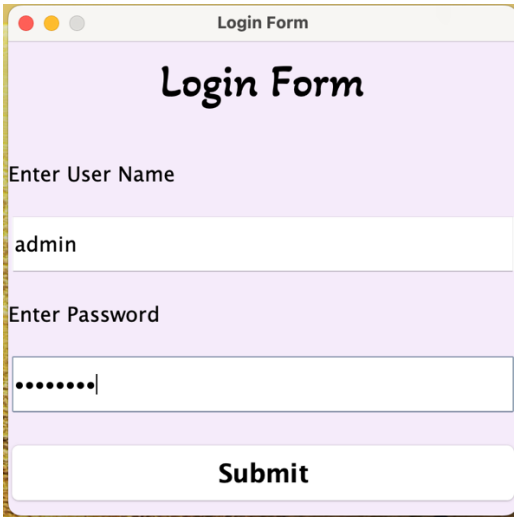
Give the field name to your panel. To get the Login.form design inside your JFrame add the below line inside your constructor.
`private JPanel mypanel;`

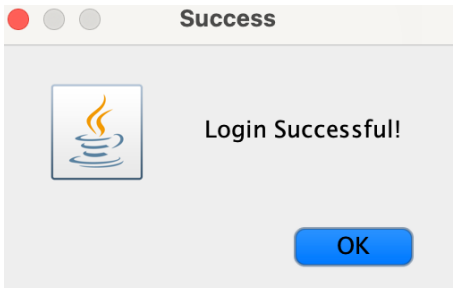
Inside constructor,
`setContentPane(mypanel);`

Write the below code inside the listener part to validate and verify the login credentials.

```
submit.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        String username = uname.getText();
        char[] pwd = password.getPassword();
        // Perform validation
        if (username.isEmpty() || pwd.length==0) {
            JOptionPane.showMessageDialog(null, "Please enter both username and password",
"Error", JOptionPane.ERROR_MESSAGE);
        } else if (username.equals("admin") && Arrays.equals(pwd,"admin123".toCharArray())) {
            JOptionPane.showMessageDialog(null, "Login Successful!", "Success",
JOptionPane.INFORMATION_MESSAGE);
        } else {
            JOptionPane.showMessageDialog(null, "Invalid username or password", "Error",
JOptionPane.ERROR_MESSAGE);
        }
    }
});
```

Run you Application





About Grid Layout

- GridLayout arranges components in a grid-like fashion with a specified number of rows and columns.
- All components added to a GridLayout are given equal size and are laid out in rows and columns based on the specified dimensions.
- Components are added to cells in the grid from left to right, top to bottom.
- Suitable for creating uniform layouts where components are arranged in a grid pattern.