Programming in JAVA

lecture 4

Introduction to Java Swing GUI programming,
generating MessageBoxes and simple dialogs

GUI programming with NetBeans

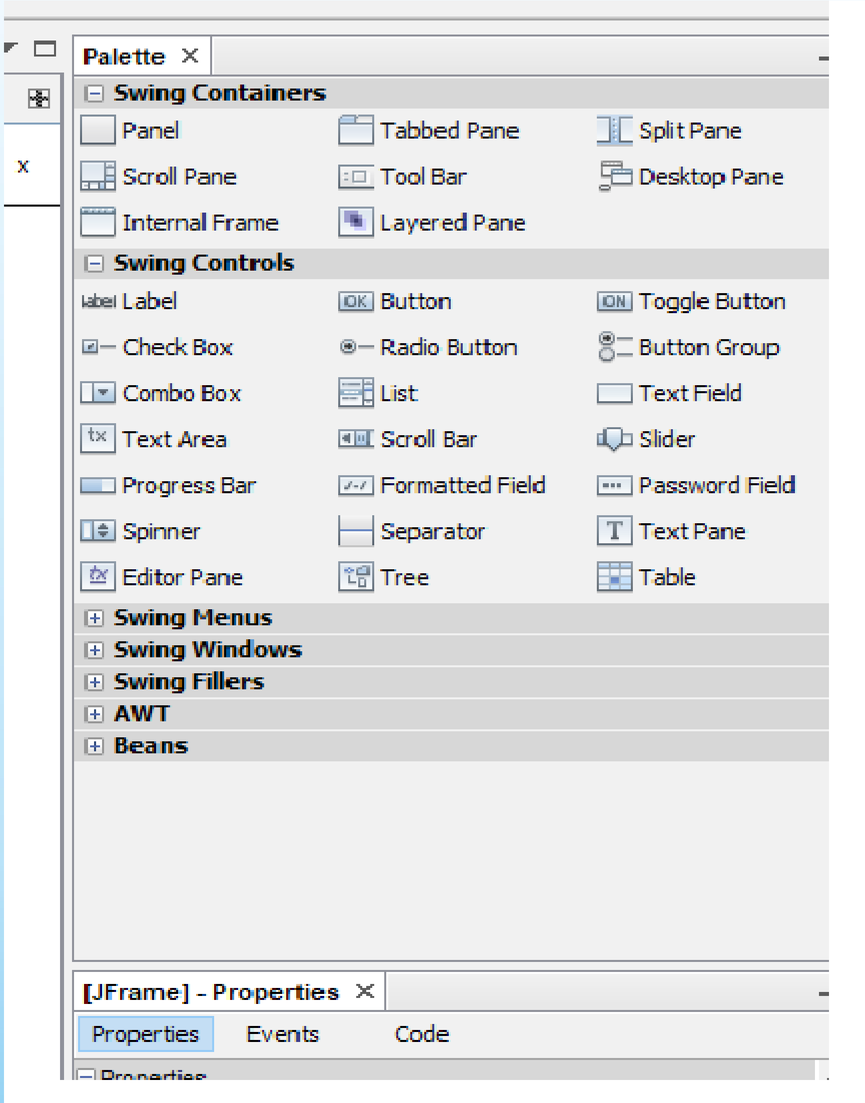
To program GUI with Netbeans you need to create a new project by:

1. Choose File→ New Project
2. In the window Java with Ant → Java Aplication (for Netbeans 11.3) or choose Java Application (for NetBeans 9.0)
3. Click next
4. In the window unclick option Create Main Class
5. Click Finish button
6. When the tree of the new project appears, expand it by clicking on it.
7. You will see <default package> subtree. Right click on it and choose from the popup menu New→**`JFrame`** form.
8. In the window you can leave the default name and click finish.

Note! Do do not choose JPanel form in step 7. JPanel cannot be the main window. If you choose JPanel in this step delete this class by right click on it.

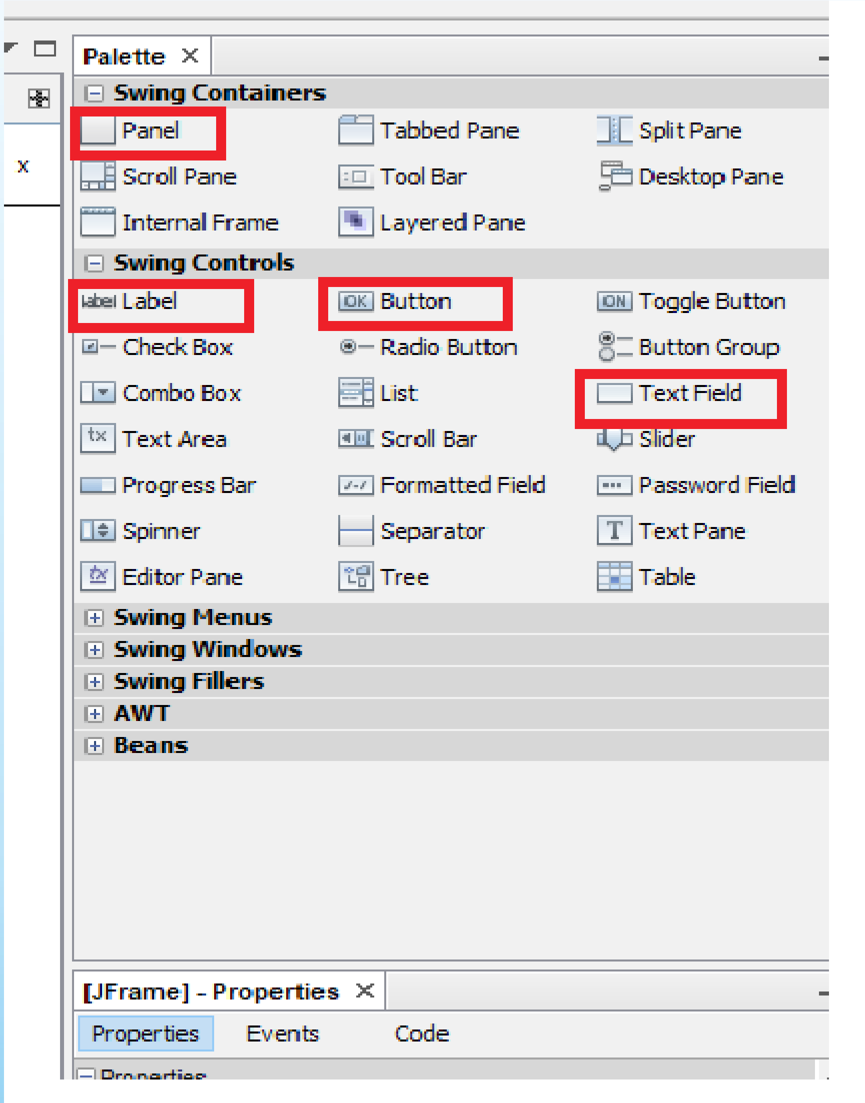
Component window

You will see on the top right Palette window containg components



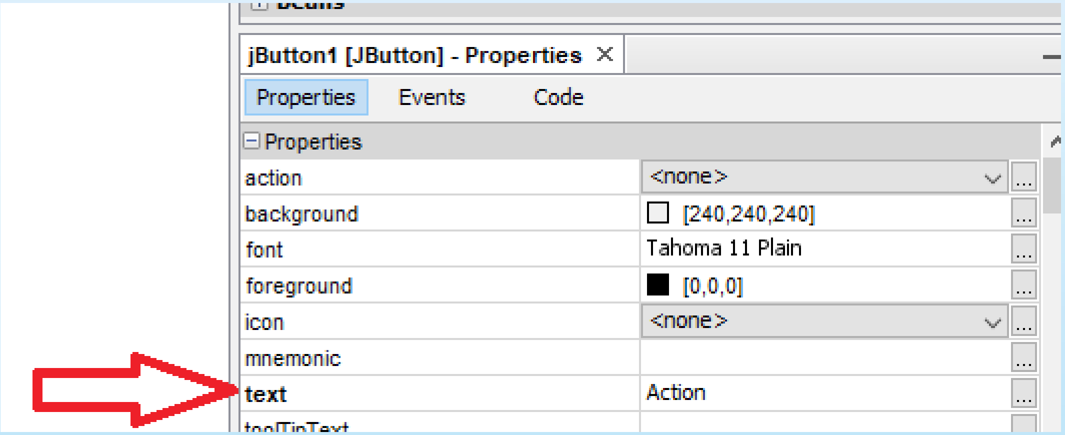
Component window

Here are the components to use in the example: Panel, Label, Button and TextField



Properties window

On the bootom right you will find the Properties window which allows you to set different properties.
Use the text property for Button, Label and TextField.

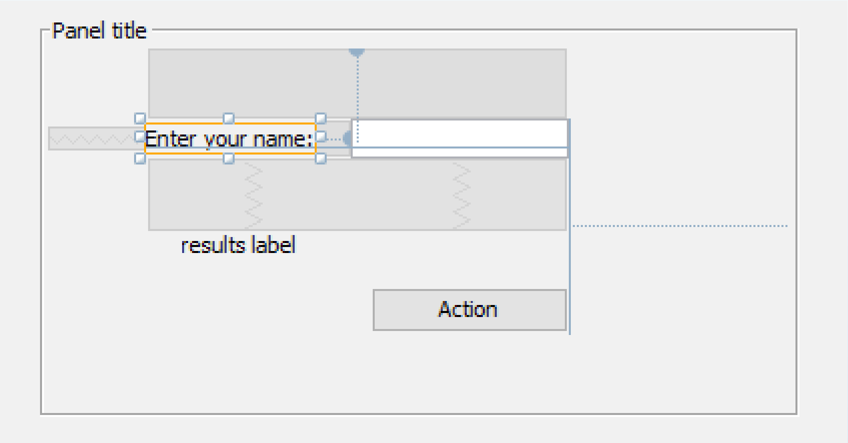


Properties

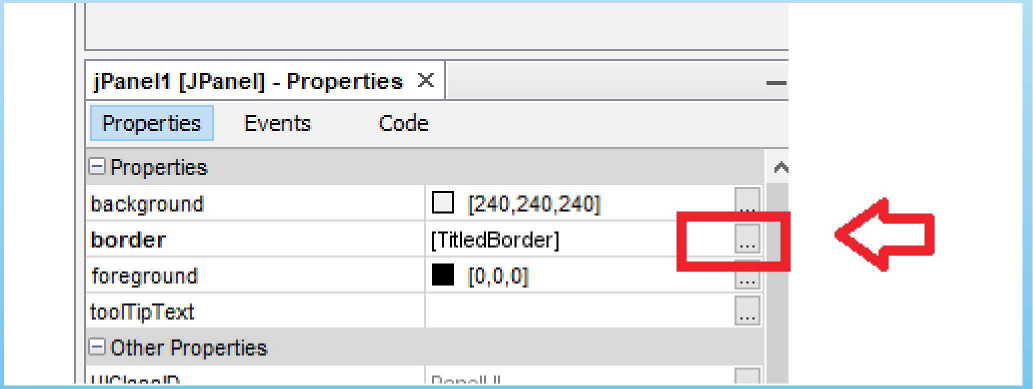
action	<none>
background	□ [240,240,240]
font	Tahoma 11 Plain
foreground	■ [0,0,0]
icon	<none>
mnemonic	
text	Action
toolTipText	

Example

Create a form that looks like the example below:



Start the creation by drag and drop of the panel component. This component is invisible by default. Set the border property of the Panel component. Keep the Panel component selected and choose border property (the window on the bottom right, under the component window). Then choose TitledBorder and enter the title in the popup window.



⊟ Properties

background	□ [240,240,240]
border	[TitledBorder]
foreground	■ [0,0,0]
toolTipText	

⊟ Other Properties

UIClassID	PanelUI

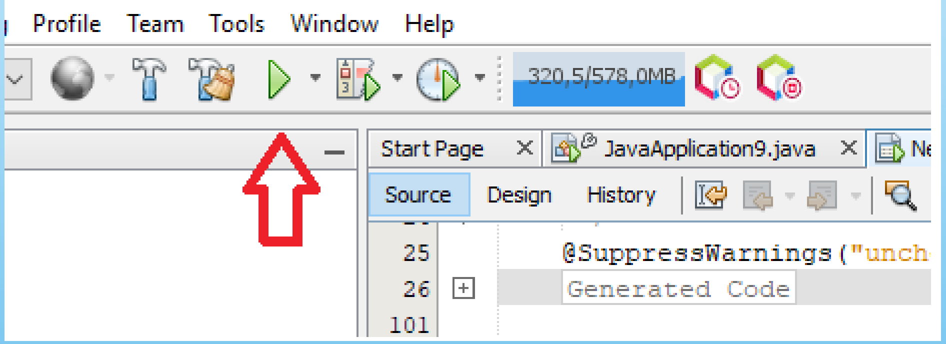
Code for the example

Double click on the Action button. This will switch the view into source code. The cursor will be in the position to enter action code for this button. Do not move the cursor. Type one line of code :

```
jLabel2.setText( "Hello " + jTextField1.getText());
```

Please note that if you changed components numbers, you will have to adjust numbers in the line above.

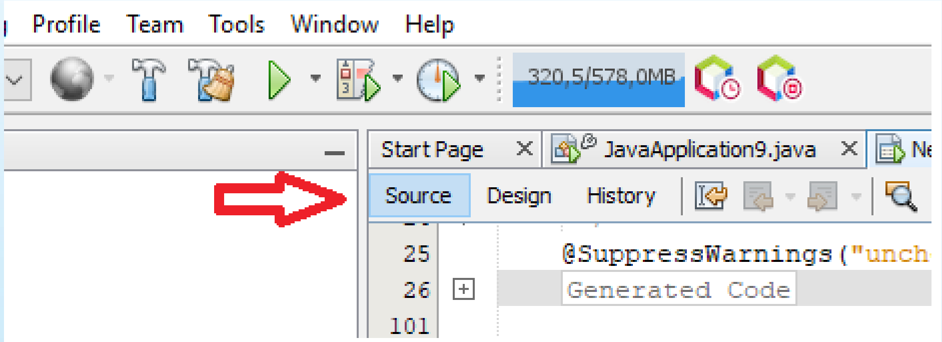
Start your program by clicking the green play button.



```
25    @SuppressWarnings("unch
26    Generated Code
101
```

Switching between source and design

You can switch between source and design by pressing Source or Design buttons



```
25        @SuppressWarnings("unch
26   ⊞    Generated Code
101
```

Executing your project without Netbeans

You can execute your project without Netbeans if you build the distribution version. To do this, choose Run → Build project (F11). Then go to the project folder (usually NetBeans Projects folder in My documents). Search for the DIST catalog.

You will find the *.jar file there. You can execute the file directly in Windows 10. In Linux OS you have to create a shortcut with the simple call:

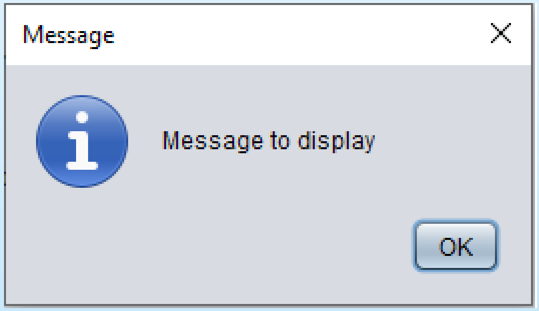
```
java -jar  yourApp.jar
```

Generating MessageBoxes and simple dialogs in JAVA Swing

Simple MessageBox

A simple MessageBox can be generated as follows:

```
JOptionPane.showMessageDialog(this,
                              "Message to display");
```



Message to display

It is important to add a reference to the JOptionPane class. It can be achieved by adding a import line to the top of the source file:

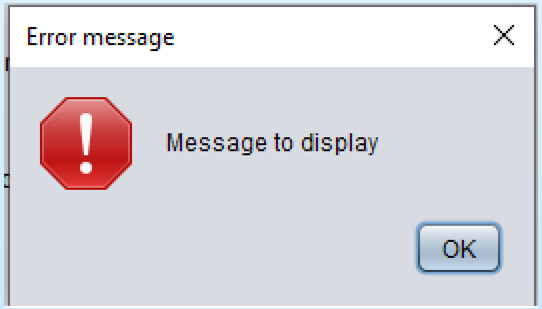
```
import javax.swing.JOptionPane;
```

Or you simply right click on the source code window and choose Fix Imports i from the popup menu. This works after you add code that uses a library class.

MessageBox with error message

An error message:

```
JOptionPane.showMessageDialog(this,
       "Message to display", "Error Title",
        JOptionPane.ERROR_MESSAGE);
```



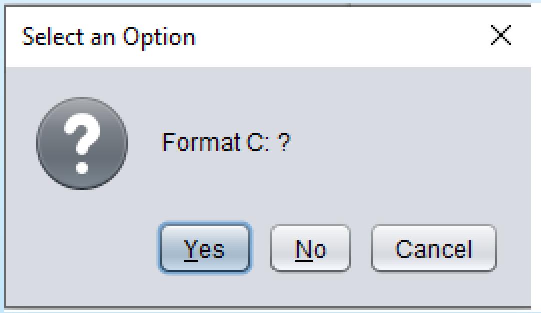
Message to display

- The first parameter of the function is the reference for the main window. This will be explained in one of the next lecture.
- The second parameter is the text to display.
- The third parameter is the text to display as the MessageBox title.
- The last one is the type of the icon displayed (`INFORMATION_MESSAGE`, `ERROR_MESSAGE or PLAIN_MESSAGE`)

ConfirmDialog

A confirm dialog:

```
int choice;
choice = JOptionPane.showConfirmDialog(this,
          "Format C: ?");
jLabel1.setText( String.valueOf(choice) );
```

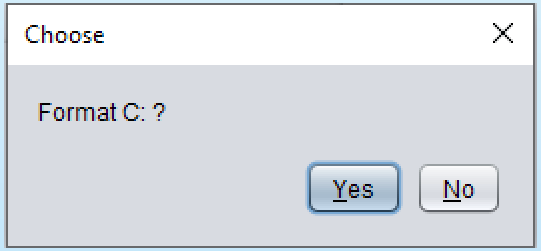


Format C: ?

ConfirmDialog with more parameters

A confirm dialog with more parameters:

```
int choice;
choice = JOptionPane.showConfirmDialog(null,
          "Format C: ?","Choose",
          JOptionPane.YES_NO_OPTION,
          JOptionPane.PLAIN_MESSAGE);
jLabel1.setText( String.valueOf(choice) );
```


Choose

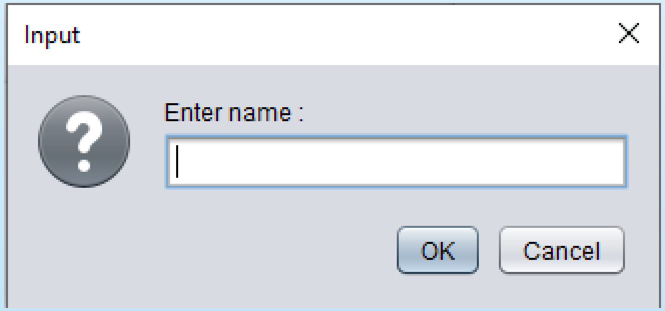
Format C: ?

- The last but one parameter is the buttons type set.
  You can choose from : `JOptionPane.YES_NO_OPTION,`
  `JOptionPane.YES_NO_CANCEL_OPTION,`
  `JOptionPane.OK_CANCEL_OPTION`
- The last parameter is the icon type (same as in MessageBox).

InputDialog

An input dialog:

```
String text;
text = JOptionPane.showInputDialog(this,
                                  "Enter name :");
jLabel1.setText( text );
```



Input
Enter name :
OK
Cancel