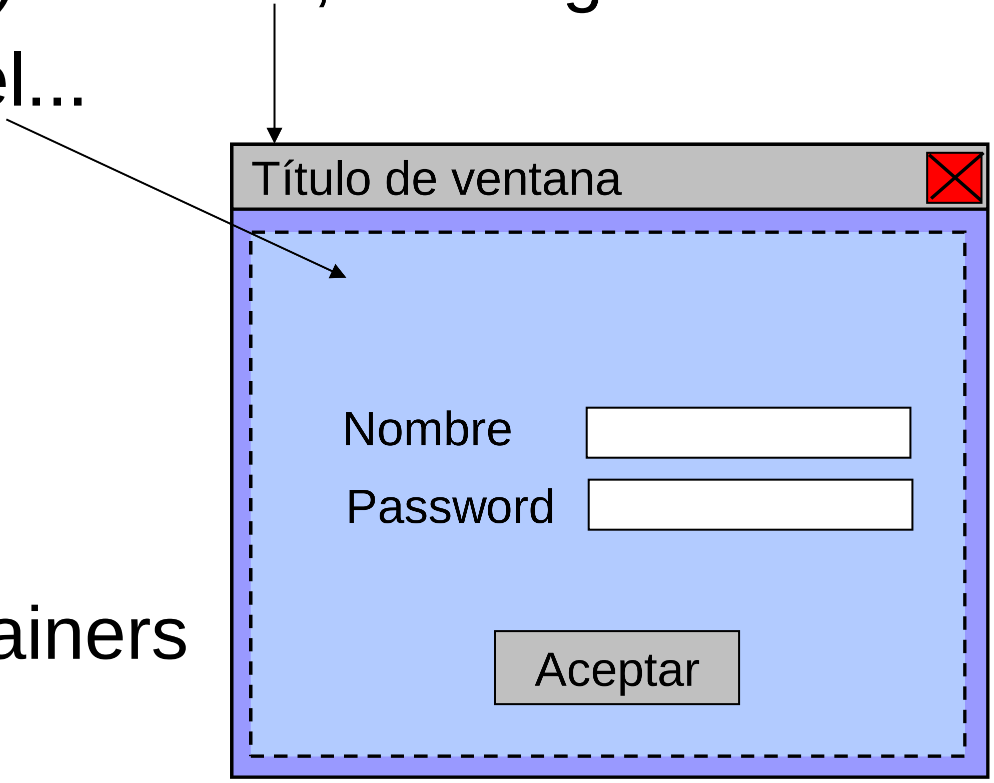
Unit 4
Basic Swing tutorial

Software Analysis and Design Project

Universidad Autónoma de Madrid

Main elements

- Containers:
  - First level (windows): JFrame, JDialog...
  - Intermediate: JPanel...
- Components:
  - Labels, text fields, buttons, ...
  - Located inside containers



Steps for GUI creation

1. **Create window (JFrame)**
2. Get the window container and assign a *layout* to it
3. Create components
   1. Define actions associated to components, for example when a button is pressed
4. Add components to the container
5. Show window

```
// create window
JFrame window = new JFrame("My GUI");

// get container, assign layout
Container container = window.getContentPane();
container.setLayout(new FlowLayout());

// create components
JLabel label = new JLabel(“Name");
final JTextField field = new JTextField(10);
JButton button   = new JButton(“Click here");

// associate actions to components
button.addActionListener(
  new ActionListener() {
    public void actionPerformed(ActionEvent e) {
      JOptionPane.showMessageDialog(null, field.getText());
    }
  }
);

// add components to container
container.add(label);
container.add(field);
container.add(button);

// show window
window.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
window.setSize(100,140);
window.setVisible(true);
```

Steps for GUI creation

1. Create window (JFrame)
2. **Get the window container and assign a *layout* to it**
3. Create components
    1. Define actions associated to components, for example when a button is pressed
4. Add components to the container
5. Show window

```java
// create window
JFrame window = new JFrame("My GUI");

// get container, assign layout
Container container = window.getContentPane();
container.setLayout(new FlowLayout());

// create components
JLabel label = new JLabel("Name");
final JTextField field = new JTextField(10);
JButton button   = new JButton("Click here");

// associate actions to components
button.addActionListener(
  new ActionListener() {
    public void actionPerformed(ActionEvent e) {
      JOptionPane.showMessageDialog(null, field.getText());
    }
  }
);

// add components to container
container.add(label);
container.add(field);
container.add(button);

// show window
window.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
window.setSize(100,140);
window.setVisible(true);
```

Steps for GUI creation

1. Create window (JFrame)
2. Get the window container and assign a *layout* to it
3. **Create components**
   1. Define actions associated to components, for example when a button is pressed
4. Add components to the container
5. Show window

```
// create window
JFrame window = new JFrame("My GUI");

// get container, assign layout
Container container = window.getContentPane();
container.setLayout(new FlowLayout());

// create components
JLabel label = new JLabel("Name");
final JTextField field = new JTextField(10);
JButton button    = new JButton("Click here");

// associate actions to components
button.addActionListener(
  new ActionListener() {
    public void actionPerformed(ActionEvent e) {
      JOptionPane.showMessageDialog(null, field.getText());
    }
  }
);

// add components to container
container.add(label);
container.add(field);
container.add(button);

// show window
window.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
window.setSize(100,140);
window.setVisible(true);
```

Steps for GUI creation

1. Create window (JFrame)
2. Get the window container and assign a *layout* to it
3. Create components
    1. **Define actions associated to components, for example when a button is pressed**
4. Add components to the container
5. Show window

```
// create window
JFrame window = new JFrame("My GUI");

// get container, assign layout
Container container = window.getContentPane();
container.setLayout(new FlowLayout());

// create components
JLabel label = new JLabel("Name");
final JTextField field = new JTextF
JButton button   = new JButton("Cli

// associate actions to components
button.addActionListener(
  new ActionListener() {
    public void actionPerformed(ActionEvent e) {
      JOptionPane.showMessageDialog(null, field.getText());
    }
  }
);

// add components to container
container.add(label);
container.add(field);
container.add(button);

// show window
window.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
window.setSize(100,140);
window.setVisible(true);
```

The code within the *actionPerformed* method will be executed when the button is clicked. In this case, a message will be shown in the screen

Steps for GUI creation

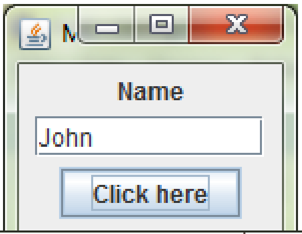
1. Create window (JFrame)
2. Get the window container and assign a *layout* to it
3. Create components
    1. Define actions associated to components, for example when a button is pressed
4. **Add components to the container**
5. Show window

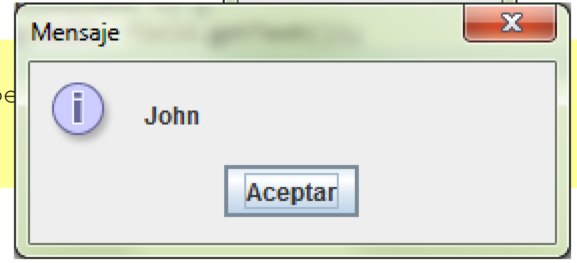
```
// create window
JFrame window = new JFrame("My GUI");

// get container, assign layout
Container container = window.getContentPane();
container.setLayout(new FlowLayout());

// create components
JLabel label = new JLabel("Name");
final JTextField field = new JTextField(10);
JButton button    = new JButton("Click here");

// associate actions to components
button.addActionListener(
  new ActionListener() {
    public void actionPerformed(ActionEvent e) {
      JOptionPane.showMessageDialog(null, field.getText());
    }
  }
);

// add components to container
container.add(label);
container.add(field);
container.add(button);

// show window
window.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
window.setSize(100,140);
window.setVisible(true);
```

Steps for GUI creation

1. Create window (JFrame)
2. Get the window container and assign a *layout* to it
3. Create components
    1. Define actions associated to components, for example when a button is pressed
4. Add components to the container
5. **Show window**

```
// create window
JFrame window = new JFrame("My GUI");

// get container, assign layout
Container container = window.getContentPane();
container.setLayout(new FlowLayout());

// create components
JLabel label = new JLabel("Name");
final JTextField field = new JTextField(10);
JButton button    = new JButton("Click here");

// associate actions to components
button.addActionListener(
  new ActionListener() {
    public void actionPerformed(ActionEvent e) {
      JOptionPane.showMessageDialog(null, field.getText());
    }
  }
);

// add components to container
container.add(label);
container.add(field);
container.add(button);

// show window
window.setDefaultCloseOpe
window.setSize(100,140);
window.setVisible(true);
```



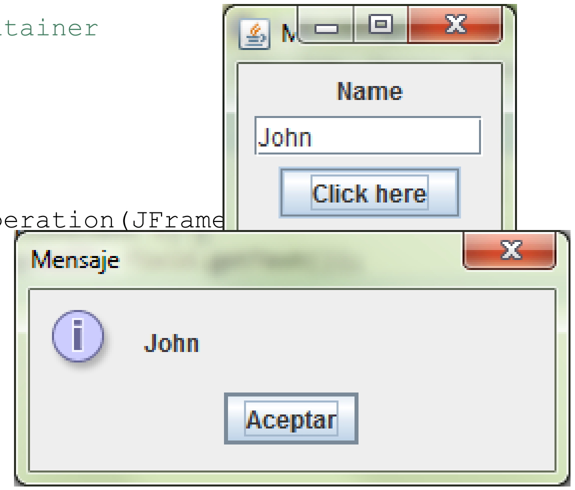


Steps for GUI creation

1. Create window (JFrame)
2. Get the window container and assign a *layout* to it
3. Create components
    1. Define actions associated to components, for example when a button is pressed
4. Add components to the container
5. Show window

```
// create window
JFrame window = new JFrame("My GUI");

// get container, assign layout
Container container = window.getContentPane();
container.setLayout(new FlowLayout());

// create components
JLabel label = new JLabel("Name");
final JTextField field = new JTextField(10);
JButton button   = new JButton("Cli

// associate actions to components
button.addActionListener(
  e ->  JOptionPane.showMessageDialog(null, field.getText())
);

// add components to container
container.add(label);
container.add(field);
container.add(button);

// show window
window.setDefaultCloseOperation(JFrame
window.setSize(100,140);
window.setVisible(true);
```

We can use a lambda expression instead of an anonymous class



GUI with several windows

1. Create window (JFrame)
2. Get the window container and assign a *layout* to it
3. **Create a JPanel for each window**
   1. The first window has a button that takes us to the second window
4. Add windows to the container
5. Show window

```
// create window
JFrame window = new JFrame("My GUI");

// get container, assign layout
Container container = window.getContentPane();
container.setLayout(new FlowLayout());

// create window 1
final JPanel window1 = new JPanel();
JButton button = new JButton("next");
window1.add(button);
window1.setVisible(true);

// create window 2
final JPanel window2 = new JPanel();
JLabel label = new JLabel("second window");
window2.add(label);
window2.setVisible(false);

// a click on the button hides window #1, shows #2
button.addActionListener(
  e -> { window1.setVisible(false);
         window2.setVisible(true); }
);

// add windows to container
container.add(window1);
container.add(window2);
```

The first window, which contains a button, is visible initially

GUI with several windows

1. Create window (JFrame)
2. Get the window container and assign a *layout* to it
3. **Create a JPanel for each window**
   1. The first window has a button that takes us to the second window
4. Add windows to the container
5. Show window

```
// create window
JFrame window = new JFrame("My GUI");

// get container, assign layout
Container container = window.getContentPane();
container.setLayout(new FlowLayout());

// create window 1
final JPanel window1 = new JPanel();
JButton button = new JButton("next");
window1.add(button);
window1.setVisible(true);

// create window 2
final JPanel window2 = new JPanel();
JLabel label = new JLabel("second window");
window2.add(label);
window2.setVisible(false);

// a click on the button hides window #1, shows #2
button.addActionListener(
  e -> { window1.setVisible(false);
       window2.setVisible(true); }
);

// add windows to container
container.add(window1);
container.add(window2);
```

The second window is hidden initially

GUI with several windows

1. Create window (JFrame)
2. Get the window container and assign a *layout* to it
3. Create a JPanel for each window
    1. **The first window has a button that takes us to the second window**
4. Add windows to the container
5. Show window

```
// create window
JFrame window = new JFrame("My GUI");

// get container, assign layout
Container container = window.getContentPane();
container.setLayout(new FlowLayout());

// create window 1
final JPanel window1 = new JPanel();
JButton button = new JButton("next");
window1.add(button);
window1.setVisible(true);

// create window 2
final JPanel window2 = new JPanel();
JLabel label = new JLabel("second window");
window2.add(label);
window2.setVisible(false);

// a click on the button hides window #
button.addActionListener(
  e -> { window1.setVisible(false);
         window2.setVisible(true); }
);

// add windows to container
container.add(window1);
container.add(window2);
```

When the button is clicked window #1 is shown and window #2 is hidden

GUI with several windows

1. Create window (JFrame)
2. Get the window container and assign a *layout* to it
3. Create a JPanel for each window
    1. The first window has a button that takes us to the second window
4. **Add windows to the container**
5. Show window

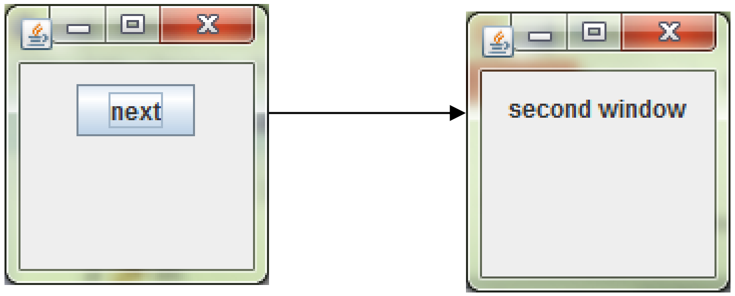
```
// create window
JFrame window = new JFrame("My GUI");

// get container, assign layout
Container container = window.getContentPane();
container.setLayout(new FlowLayout());

// create window 1
final JPanel window1 = new JPanel();
JButton button = new JButton("next");
window1.add(button);
window1.setVisible(true);

// create window 2
final JPanel window2 = new JPanel();
JLabel label = new JLabel("second window");
window2.add(label);
window2.setVisible(false);

// a click on the button hides window #1, shows #2
button.addActionListener(
  e -> { window1.setVisible(false);
         window2.setVisible(true); }
);

// add windows to container
container.add(window1);
container.add(window2);
```

A container can include other containers (window1 and window2 are containers)

GUI with several windows

1. Create window (JFrame)
2. Get the window container and assign a *layout* to it
3. Create a JPanel for each window
    1. The first window has a button that takes us to the second window
4. Add windows to the container
5. **Show window**

```
// create window
JFrame window = new JFrame("My GUI");

// get container, assign layout
Container container = window.getContentPane();
container.setLayout(new FlowLayout());

// create window 1
final JPanel window1 = new JPanel();
JButton button = new JButton("next");
window1.add(button);
window1.setVisible(true);

// create window 2
```



```
ows #2

        window2.setVisible(true); }
);

// add windows to container
container.add(window1);
container.add(window2);
```

Building GUI Components

We can create subclasses of Swing Components

```
public class MyPanel extends JPanel {

  MyPanel () {
    // assign layout
    this.setLayout(new FlowLayout());

    // create components
    JLabel     label = new JLabel("Name");
    final JTextField field    = new JTextField(10);
    JButton    button    = new JButton("click here");

    // associate actions to components
    button.addActionListener(
      new ActionListener() {
        public void actionPerformed(ActionEvent e) {
          JOptionPane.showMessageDialog(null,
            field.getText());
        }
      }
    );

    // add components
    // to contaiiner
    this.add(label);
    this.add(field);
    this.add(button);
  }
}
```

Panel components are created in the constructor. There can be constructors with different parameters.

Those subclasses can be used as Swing components

```
// This is the example in slide #3,
// using the new panel class we have created

// create window
JFrame window = new JFrame("My GUI");

// get container, assign layout
Container container = window.getContentPane();
container.setLayout(new FlowLayout());

// create components
JPanel panel = new MyPanel();

// add components to container
container.add(panel);

// show window
window.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
window.pack();
window.setVisible(true);
```

It is recommended to create GUI components instead of using a single class for the whole user interface.

References

- Swing Tutorial: http://docs.oracle.com/javase/tutorial/uiswing/
- Swing API (JavaDoc): http://download.oracle.com/javase/6/docs/api/javax/swing/package-summary.html
- Collection of Swing examples:

  http://download.oracle.com/javase/tutorial/uiswing/examples/components/index.html