ΤΕΧΝΙΚΕΣ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΑΦΟΥΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Graphical User Interfaces (GUI)
SWING

Επισκόπηση

Swing

- Η Swing είναι η βιβλιοθήκη της Java για τον προγραμματισμό GUIs (Graphical User Interfaces) [και πιο γενικά για την χρήση γραφικών].
  - Η μετεξέλιξη του AWT (Abstract Window Toolkit) το οποίο ήταν το πρώτο αλλά όχι τόσο επιτυχημένο πακέτο της Java για GUI.

Swing

- Στην Swing βιβλιοθήκη ένα GUI αποτελείται από πολλά στοιχεία/συστατικά (components)
  - π.χ. παράθυρα, κουμπιά, μενού, κουτιά εισαγωγής κειμένου, κλπ.
- Τα components αυτά πυροδοτούν συμβάντα
  - Π.χ. το πάτημα ενός κουμπιού, η εισαγωγή κειμένου, η επιλογή σε ένα μενού, κλπ
- Τα συμβάντα αυτά τα χειρίζονται τα αντικείμενα-ακροατές, που έχουν ειδικές μεθόδους γι αυτά
  - Τι γίνεται όταν πατάμε ένα κουμπί, όταν κάνουμε μια επιλογή κλπ
- Όλο το πρόγραμμα κυλάει ως μια αλληλουχία από συμβάντα και τον χειρισμό των ακροατών.

Παράδειγμα

- Θα υλοποιήσουμε ένα πρόγραμμα που δημιουργεί ένα παράθυρο με ένα κουμπί, το οποίο αν πατήσουμε κλείνει το παράθυρο

- Με βάση το προηγούμενο μοντέλο:
  - Το component είναι το κουμπί
  - Πυροδοτείται το συμβάν όταν πατάμε το κουμπί
  - Ο ακροατής στο συμβάν αυτό είναι επιφορτισμένος με το καθήκον να κλείσει το παράθυρο.

```
import javax.swing.JFrame;
import javax.swing.JButton;


public class FirstWindow extends JFrame
{
    public static final int WIDTH = 300;
    public static final int HEIGHT = 200;

    public FirstWindow( )
    {
        super( );
        setSize(WIDTH, HEIGHT);
        setTitle("First Window Class");
        setDefaultCloseOperation(JFrame.DO_NOTHING_ON_CLOSE);

        JButton endButton = new JButton("Click to end program.");
        endButton.addActionListener(new EndingListener( ));

        add(endButton);
    }
}
```

Η δημιουργία του ActionListener γίνεται ως ανώνυμο αντικειμενο μιας και δεν θα το χρησιμοποιήσουμε ποτέ άμεσα

```
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;

public class EndingListener implements ActionListener
{
    public void actionPerformed(ActionEvent e)
    {
        System.exit(0);
    }
}
```

Ένας ακροατής πάντα υλοποιεί το interface ActionListener και πρέπει να υλοποιεί την μέθοδο actionPerformed(ActionEvent)

Όταν πατάμε το κουμπί στο GUI καλείται η μέθοδος actionPerfomed του ακροατή που έχουμε καταχωρίσει για το κουμπί

Η κλήση της actionPerformed από τον ActionListener γίνεται αυτόματα μέσω της βιβλιοθήκης Swing, δεν την κάνει ο προγραμματιστής

Η παράμετρος ActionEvent περιέχει πληροφορία σχετικά με το συμβάν που μπορεί να χρησιμοποιηθεί.

```
public class DemoButtonWindow
{
    public static void main(String[] args)
    {
        FirstWindow w = new FirstWindow( );
        w.setVisible(true);
    }
}
```

Εδώ δημιουργούμε το παράθυρο μας

Πολλά συστατικά

- Αν θέλουμε να βάλουμε πολλά components μέσα στο παράθυρο μας τότε θα πρέπει να προσδιορίσουμε που θα τοποθετηθούν αλλιώς θα μπούνε το ένα πάνω στο άλλο.
- Αυτό γίνεται με την εντολή setLayout που καθορίζει την τοποθέτηση μέσα στο παράθυρο
  - Αυτό μπορεί να γίνει με διαφορετικούς τρόπους

FlowLayout

- Απλά τοποθετεί τα components το ένα μετά το άλλο από τα αριστερά προς τα δεξιά
- Καλούμε την εντολή

```
setLayout(new FlowLayout());
```

(Πρέπει να έχουμε κάνει include java.awt.FlowLayout)

- Μετά προσθέτουμε κανονικά τα components με την add.

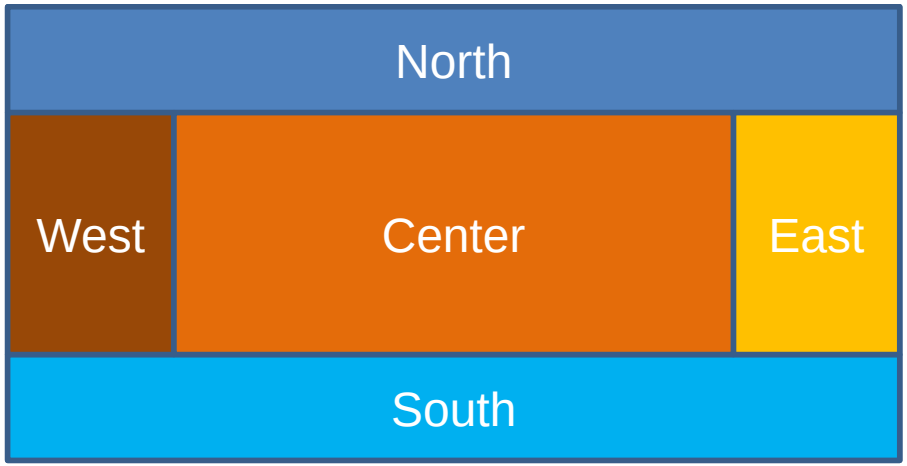
BorderLayout

- Στην περίπτωση αυτή ο χώρος χωρίζεται σε πέντε περιοχές: North, South, East, West Center
- Καλούμε την εντολή

```
setLayout(new BorderLayout());
```

(Πρεπει να έχουμε κάνει include java.awt.BorderLayout)

- Μετά όταν προσθέτουμε τα components με την add, προσδιορίζουμε την περιοχή στην οποία θα προστεθούν.
  - Π.χ., `add(label, BorderLayout.CENTER)`



GridLayout

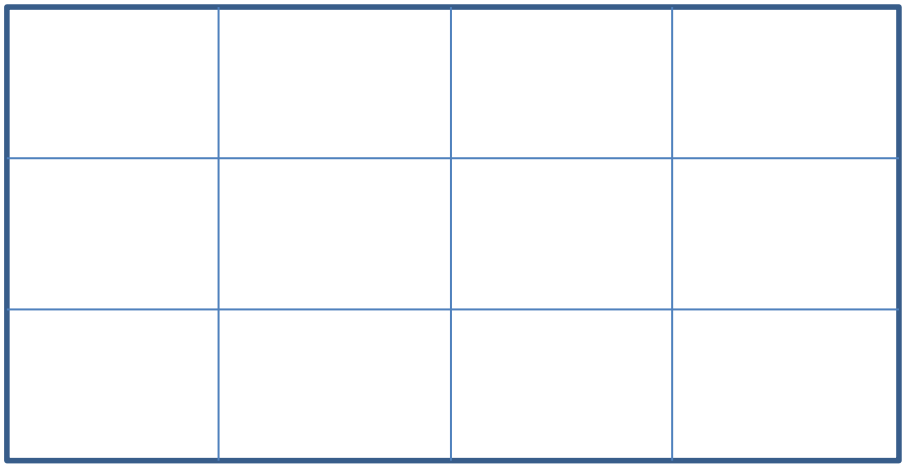
- Στην περίπτωση αυτή ορίζουμε ένα πλέγμα με n γραμμές και m στήλες και αυτό γεμίζει από τα αριστερά προς τα δεξιά και από πάνω προς τα κάτω
- Καλούμε την εντολή

```
setLayout(new GridLayout(n,m));
```

(Πρεπει να έχουμε κάνει include java.awt.GridLayout)

- Μετά προσθέτουμε κανονικά τα components με την add.

Grid 3x4



JPanel

- Το panel (τομέας) είναι ένας container
  - Μέσα σε ένα container μπορούμε να βάλουμε components και να ορίσουμε χειρισμό συμβάντων.
- Τα panels κατά μία έννοια ορίζουν ένα παράθυρο μέσα στο παράθυρο
  - Το panel έχει κι αυτό το δικό του layout και τοποθετούμε μέσα σε αυτό συστατικά.
  - Π.χ., ο παρακάτω κώδικας εκτελείται μέσα σε ένα JFrame.

```
setLayout(new BorderLayout());

JPanel buttonPanel = new JPanel();
buttonPanel.setLayout(new FlowLayout());

JButton button1 = new JButton("one");
buttonPanel.add(button1);

JButton button2 = new JButton("two");
buttonPanel.add(button2);

add(buttonPanel, BorderLayout.SOUTH);
```

Παράδειγμα

- Θα δημιουργήσουμε ένα παράθυρο με τρία panels το κάθε panel θα παίρνει διαφορετικό χρώμα με ένα διαφορετικό κουμπί.

```
import javax.swing.JFrame;
import javax.swing.JPanel;
import java.awt.BorderLayout;
import java.awt.GridLayout;
import java.awt.FlowLayout;
import java.awt.Color;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;

public class PanelDemo extends JFrame implements ActionListener
{
    public static final int WIDTH = 300;
    public static final int HEIGHT = 200;

    private JPanel redPanel;
    private JPanel whitePanel;
    private JPanel bluePanel;


    public PanelDemo( )
    {
        super("Panel Demonstration");
        setSize(WIDTH, HEIGHT);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout( ));
```

Η κλάση υλοποιεί τον ακροατή και την actionPerformed μεθοδο

Δηλώνουμε τα τρία πάνελ με τα τρία χρώματα

Ορίζουμε τα χαρακτηριστικά του βασικού παραθύρου

Συνέχεια στην επόμενη

Συνέχεια από
την προηγούμενη

Δημιουργούμε ένα μεγάλο πάνελ που θα κρατάει τα τρία χρωματιστά πάνελ

```
JPanel biggerPanel = new JPanel( );
biggerPanel.setLayout(new GridLayout(1, 3));

redPanel = new JPanel( );
redPanel.setBackground(Color.LIGHT_GRAY);
biggerPanel.add(redPanel);

whitePanel = new JPanel( );
whitePanel.setBackground(Color.LIGHT_GRAY);
biggerPanel.add(whitePanel);

bluePanel = new JPanel( );
bluePanel.setBackground(Color.LIGHT_GRAY);
biggerPanel.add(bluePanel);

add(biggerPanel, BorderLayout.CENTER);
```

Δημιουργούμε τα χρωματιστά πάνελ και τα προσθέτουμε στο μεγάλο πάνελ

Βάζουμε το μεγάλο πάνελ στο κέντρο του παραθύρου

Συνέχεια στην
επόμενη

Συνέχεια από την προηγούμενη

```
        JPanel buttonPanel = new JPanel( );
        buttonPanel.setBackground(Color.LIGHT_GRAY);
        buttonPanel.setLayout(new FlowLayout( ));

        JButton redButton = new JButton("Red");
        redButton.setBackground(Color.RED);
        redButton.addActionListener(this);
        buttonPanel.add(redButton);

        JButton whiteButton = new JButton("White");
        whiteButton.setBackground(Color.WHITE);
        whiteButton.addActionListener(this);
        buttonPanel.add(whiteButton);

        JButton blueButton = new JButton("Blue");
        blueButton.setBackground(Color.BLUE);
        blueButton.addActionListener(this);
        buttonPanel.add(blueButton);

        add(buttonPanel, BorderLayout.SOUTH);

    } // τέλος του constructor
```

Δημιουργούμε ένα πάνελ που θα κρατάει τα τρία κουμπιά

Δημιουργούμε τα τρία κουμπιά και τα προσθέτουμε στο πάνελ

Ο ακροατής των κουμπιών είναι το ίδιο το αντικείμενο

Βάζουμε το πάνελ με τα κουμπιά στον πάτο του παραθύρου

Συνέχεια στην επόμενη

Συνέχεια από την προηγούμενη

```
    public void actionPerformed(ActionEvent e)
    {
        String buttonString = e.getActionCommand( );

        if (buttonString.equals("Red"))
            redPanel.setBackground(Color.RED);
        else if (buttonString.equals("White"))
            whitePanel.setBackground(Color.WHITE);
        else if (buttonString.equals("Blue"))
            bluePanel.setBackground(Color.BLUE);
        else
            System.out.println("Unexpected error.");
    }

    public static void main(String[] args)
    {
        PanelDemo gui = new PanelDemo( );
        gui.setVisible(true);
    }

}
```

Η συνάρτηση actionPerformed που καλείται όταν πατηθούν τα κουμπιά (μιας και το αντικείμενο είναι και ακροατής)

Επιστρέφει το actionCommand String, το οποίο αν δεν το έχουμε αλλάξει είναι το όνομα του κουμπιού

Το αποτέλεσμα του κάθε διαφορετικού κουμπιού.

Δημιουργία του παραθύρου

actionCommand

- Ένα String πεδίο που κρατάει πληροφορία για το συμβάν
  - Αν δεν αλλάξουμε κάτι αυτό είναι το όνομα του κουμπιού
- Μπορούμε να διαβάσουμε το String με την εντολή getActionCommand.
- Μπορούμε να θέσουμε μια τιμή στο String με την εντολή setActionCommand(String)
- Π.χ.

```
redButton.setActionCommand("RedButtonClick");
```

Χρώματα

- Μπορούμε να ορίσουμε τα δικά μας χρώματα με την RGB σύμβαση
  - `Color myColor = new Color(200,100,4);`
  - Τα ορίσματα ειανι οι RGB (Red, Green, Blue) τιμές

Ακροατές

- Στο πρόγραμμα μας ορίσαμε την κλάση που δημιουργεί το παράθυρο (extends JFrame) να είναι και ο ακροατής (implements ActionListener) των συμβάντων μέσα στο παράθυρο.
  - Αυτό είναι μια βολική λύση γιατί όλος ο κώδικας είναι στο ίδιο σημείο
  - Έχει το πρόβλημα ότι έχουμε μία μόνο μέθοδο actionPerformed στην οποία θα πρέπει να ξεχωρίσουμε όλες τις περιπτώσεις.

- Πιο βολικό να έχουμε ένα διαφορετικό ActionListener για κάθε διαφορετικό συμβάν
  - Προβλήματα:
    - Θα πρέπει να ορίσουμε πολλαπλές κλάσεις ακροατών σε πολλαπλά αρχεία
    - Θα πρέπει να περνάμε σαν παράμετρους τα στοιχεία που θέλουμε να αλλάξουμε.

Ακροατές

- Λύση: Να ορίσουμε τους ακροατές που χρειάζεται το παράθυρο μας ως εσωτερικές κλάσεις

- Υπενθύμιση: μια εσωτερική κλάση ορίζεται μέσα σε μία άλλη κλάση και την βλέπει μόνο η κλάση που την ορίζει

- Πλεονεκτήματα:
  - Οι κλάσεις είναι πλέον τοπικές στον κώδικα που τις καλεί, μπορούμε να επαναχρησιμοποιούμε τα ίδια ονόματα
  - Οι κλάσεις έχουν πρόσβαση σε ιδιωτικά πεδία

```
JButton redButton = new JButton("Red");
redButton.setBackground(Color.RED);
redButton.addActionListener(new RedListener());
buttonPanel.add(redButton);

JButton whiteButton = new JButton("White");
whiteButton.setBackground(Color.WHITE);
whiteButton.addActionListener(new WhiteListener());
buttonPanel.add(whiteButton);

JButton blueButton = new JButton("Blue");
blueButton.setBackground(Color.BLUE);
blueButton.addActionListener(new BlueListener());
buttonPanel.add(blueButton);
```

Ορισμός των εσωτερικών κλάσεων-ακροατών

```
private class RedListener{
    public void actionPerformed(ActionEvent e)
    {
        redPanel.setBackground(Color.RED);
    }
}

private class WhiteListener{
    public void actionPerformed(ActionEvent e)
    {
        whitePanel.setBackground(Color.WHITE);
    }
}

private class BlueListener{
    public void actionPerformed(ActionEvent e)
    {
        bluePanel.setBackground(Color.BLUE);
    }
}
```

Οι εσωτερικές κλάσεις έχουν πρόσβαση στα ιδιωτικά αντικείμενα πάνελ

Ανώνυμες κλάσεις

- Τα αντικείμενα-ακροατές είναι ανώνυμα αντικείμενα
  - `redButton.addActionListener(new RedListener());`
- Μπορούμε να κάνουμε τον κώδικα ακόμη πιο συνοπτικό ορίζοντας μια ανώνυμη κλάση
  - Ο ορισμός της κλάσης γίνεται εκεί που τον χρειαζόμαστε μόνο και υλοποιεί ένα Interface
  - Δεν συνίσταται αλλά μπορεί να το συναντήσετε σε κώδικα που δημιουργείται από IDEs

```
redButton.addActionListener(new ActionListener()
 {
    public void actionPerformed(ActionEvent e){
          redPanel.setBackground(Color.RED);
    }
  }
);
```

Ο ορισμός της κλάσης
Χρησιμοποιούμε το όνομα του interface

Menu

- Μπορούμε να δημιουργήσουμε ένα drop-down menu χρησιμοποιώντας την κλάση JMenu.

```
JMenu colorMenu = new JMenu("Add Colors");

JMenuItem redChoice = new JMenuItem("Red");
redChoice.addActionListener(this);
colorMenu.add(redChoice);

JMenuItem whiteChoice = new JMenuItem("White");
whiteChoice.addActionListener(this);
colorMenu.add(whiteChoice);

JMenuItem blueChoice = new JMenuItem("Blue");
blueChoice.addActionListener(this);
colorMenu.add(blueChoice);

JMenuBar bar = new JMenuBar( );
bar.add(colorMenu);
setJMenuBar(bar);
```

Δημιουργεί ένα drop-down menu

Δημιουργεί τις επιλογές του μενού και τις προσθέτει στο μενού

Δημιουργεί ένα menu bar στην κορυφή του παραθύρου και προσθέτει το menu σε αυτό

Text Box

- Μπορούμε να δημιουργήσουμε ένα πεδίο κειμένου με την κλάση JTextField.
  - Το JTextField δημιουργεί ένα text box μίας γραμμής
  - Διαβάζουμε και γράφουμε κείμενο στο text box με τις μεθόδους getText() και setText(String).
- Για ένα πεδίο κειμένου μεγαλύτερο από μία γραμμή μπορούμε να χρησιμοποιήσουμε την κλάση JTextArea

Παράδειγμα

```
        JTextField name = new JTextField(NUMBER_OF_CHAR);
        namePanel.add(name, BorderLayout.SOUTH);

        JButton actionButton = new JButton("Click me");
        actionButton.addActionListener(this);
        buttonPanel.add(actionButton);

        JButton clearButton = new JButton("Clear");
        clearButton.addActionListener(this);
        buttonPanel.add(clearButton);
```

```
public void actionPerformed(ActionEvent e)
{
        String actionCommand = e.getActionCommand( );

        if (actionCommand.equals("Click me"))
            name.setText("Hello " + name.getText( ));
        else if (actionCommand.equals("Clear"))
            name.setText("");
        else
            name.setText("Unexpected error.");
}
```

Pop-up Windows

- Αν θέλουμε να δημιουργήσουμε παράθυρα διαλόγου μπορούμε να χρησιμοποιήσουμε την κλάση JOptionPane
  - Πετάει (pops up) ένα παράθυρο το οποίο μπορεί να μας ζητάει είσοδο, ή να ζητάει επιβεβαίωση.
  - Η δημιουργία και η διαχείριση των παραθύρων γίνεται με στατικές μεθόδους.

```
import javax.swing.JOptionPane;

public class PopUpDemo
{
  public static void main(String args[])
    boolean done = false;
    while (!done){
      String classes =
        JOptionPane.showInputDialog("Enter number of classes");
      String students =
        JOptionPane.showInputDialog("Enter number of students");
      int totalStudents =
        Integer.parseInt(classes)*Integer.parseInt(students);

      JOptionPane.showMessageDialog(null,
        "Total number of students = "+totalStudents);

      int answer =
        JOptionPane.showConfirmDialog(null,
        "Continue?",
        "Confirm",
        JOptionPane.YES_NO_OPTION);
      done = (answer == JOptionPane.NO_OPTION);
    }
    System.exit(0);
  }
}
```

Εμφανίζει ένα παράθυρο διαλόγου που ζητάει από τον χρήστη να δώσει είσοδο. Η είσοδος αποθηκεύεται στο String που επιστρέφεται

Το αντικείμενο (component) που είναι πατέρας του pop-up, null η default τιμή

Εμφανίζει ένα παράθυρο που τυπώνει ένα μήνυμα

Η ερώτηση στο χρήστη

Τίτλος παραθύρου

Εμφανίζει ένα παράθυρο επιβεβαίωσης

Τύπος επιβεβαίωσης

Σταθερά για την επιλογή (YES_OPTION για ΝΑΙ)

Άλλοι τύποι επιβεβαίωσης:

- OK_CANCEL_OPTION
- YES_NO_CANCEL_OPTION

Icons

- Μπορούμε να βάλουμε μέσα στο GUI μας και εικονίδια
- Παράδειγμα

Δημιουργεί ένα εικονίδιο από μία εικόνα

```
ImageIcon dukeIcon = new ImageIcon("duke_waving.gif");
JLabel dukeLabel = new JLabel("Mood check");
dukeLabel.setIcon(dukeIcon);
```

Προσθέτει το εικονίδιο σε ένα label

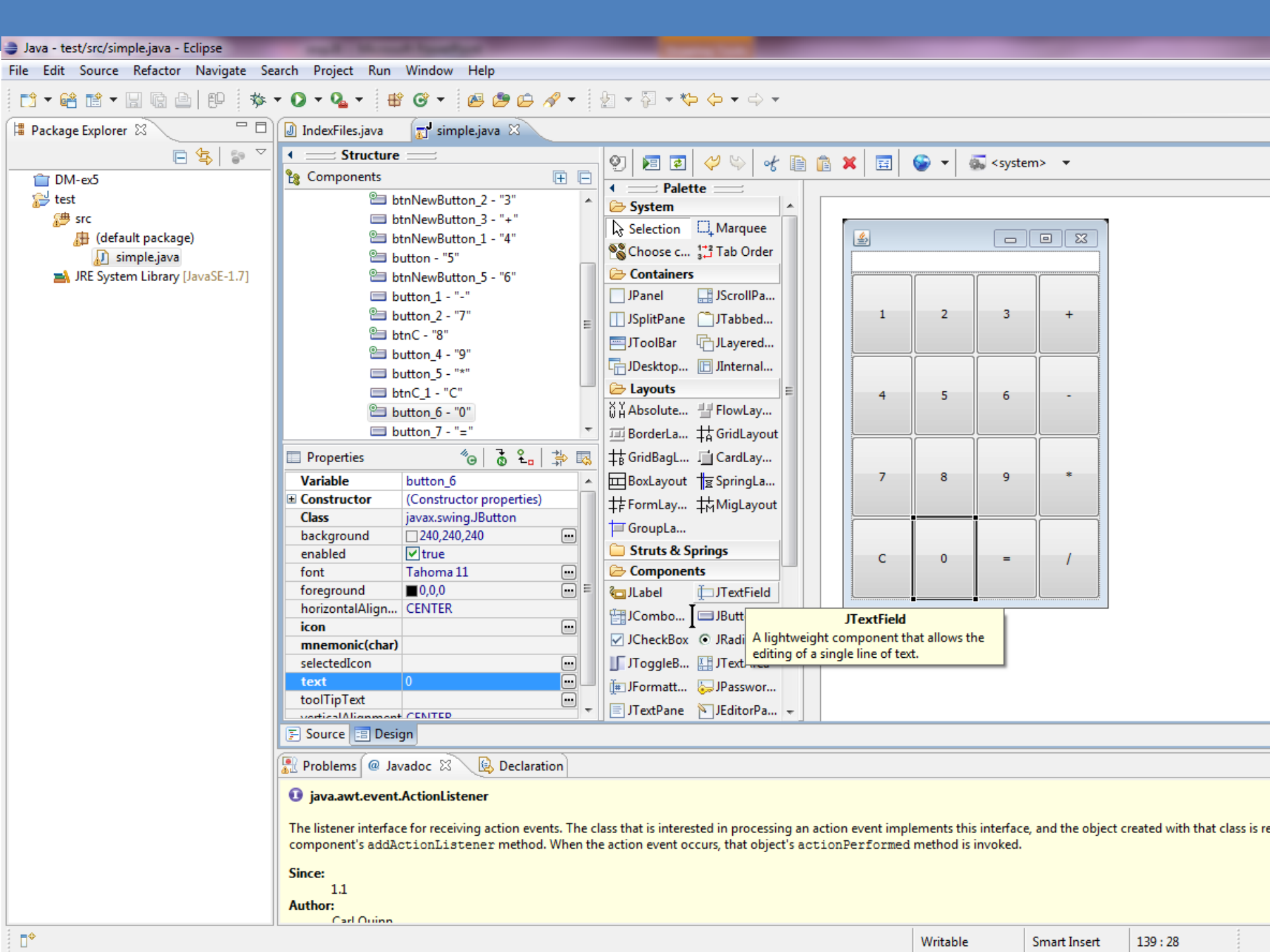
```
ImageIcon happyIcon = new ImageIcon("smiley.gif");
JButton happyButton = new JButton("Happy");
happyButton.setIcon(happyIcon);
```

Προσθέτει το εικονίδιο σε ένα button

Eclipse

- Η eclipse (αλλά και άλλα IDEs) μας δίνει πολλά έτοιμα εργαλεία για την δημιουργία GUIs
- Εγκαταστήσετε το plug-in Windows Builder Pro

- Παράδειγμα: Δημιουργήστε μια αριθμομηχανή.


Java - test/src/simple.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
DM-ex5
test
src
(default package)
simple.java
JRE System Library [JavaSE-1.7]
IndexFiles.java
simple.java
Structure
Components
btnNewButton_2 - "3"
btnNewButton_3 - "+"
btnNewButton_1 - "4"
button - "5"
btnNewButton_5 - "6"
button_1 - "-"
button_2 - "7"
btnC - "8"
button_4 - "9"
button_5 - "*"
btnC_1 - "C"
button_6 - "0"
button_7 - "="
Properties
Variable button_6
Constructor (Constructor properties)
Class javax.swing.JButton
background 240,240,240
enabled true
font Tahoma 11
foreground 0,0,0
horizontalAlign... CENTER
icon
mnemonic(char)
selectedIcon
text 0
toolTipText
Palette
System
Selection
Marquee
Choose c...
Tab Order
Containers
JPanel
JScrollPa...
JSplitPane
JTabbed...
JToolBar
JLayered...
JDesktop...
JInternal...
Layouts
Absolute...
FlowLay...
BorderLa...
GridLayout
GridBagL...
CardLay...
BoxLayout
SpringLa...
FormLay...
MigLayout
GroupLa...
Struts & Springs
Components
JLabel
JTextField
JCombo...
JCheckBox
JToggleB...
JFormatt...
JPasswor...
JTextPane
JEditorPa...
JTextField
A lightweight component that allows the editing of a single line of text.
1 2 3 +
4 5 6 -
7 8 9 *
C 0 = /
Source
Design
Problems
Javadoc
Declaration
java.awt.event.ActionListener
The listener interface for receiving action events. The class that is interested in processing an action event implements this interface, and the object created with that class is re
component's addActionListener method. When the action event occurs, that object's actionPerformed method is invoked.
Since:
1.1
Author:
Writable
Smart Insert
139 : 28

Δημιουργία κώδικα

- Τα IDEs μας επιτρέπουν να διαχωρίζουμε το design από τον κώδικα
  - Το πλεονέκτημα είναι ότι έχουμε ένα WYSIWYG interface με το οποίο μπορούμε να σχεδιάσουμε το GUI
  - Το μειονέκτημα είναι ότι δημιουργείται πολύς κώδικας αυτόματα ο οποίος δεν είναι πάντα όπως τον θέλουμε

Δημιουργία κώδικα

- Η δημιουργία ενός κουμπιού δημιουργεί αυτό τον κώδικα

```
JButton button_6 = new JButton("0");
panel.add(button_6);
```

- Αν πατήσουμε πάνω στο κουμπί (double-click) δημιουργείται ο ακροατής του κουμπιού αυτόματα ως μια ανώνυμη κλάση

```
JButton button_6 = new JButton("0");
button_6.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
    }
});
panel.add(button_6);
```

Δημιουργία κώδικα

- Η δημιουργία ενός κουμπιού δημιουργεί αυτό τον κώδικα

```
JButton button_6 = new JButton("0");
panel.add(button_6);
```

- Αν πατήσουμε πάνω στο κουμπί (double-click) δημιουργείται ο ακροατής του κουμπιού αυτόματα ως μια ανώνυμη κλάση
  - Εμείς συμπληρώνουμε τον κώδικα

```
JButton button_6 = new JButton("0");
button_6.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        textField.setText(textField.getText()+"0");
    }
});
panel.add(button_6);
```

ΕΠΙΣΚΟΠΗΣΗ

Θέματα που καλύψαμε

- Γενικές έννοιες αντικειμενοστραφούς προγραμματισμού
- Βασικά στοιχεία Java
- Κλάσεις και αντικείμενα
  - Πεδία, μέθοδοι, δημιουργοί, αναφορές
- Σύνθεση και συνάθροιση αντικειμένων
  - Πώς να φτιάχνουμε μεγαλύτερες κλάσεις με μικρότερα αντικείμενα - σχεδίαση
- Κληρονομικότητα, Πολυμορφισμός
- Συλλογές δεδομένων
- Εξαιρέσεις, I/O
- Γραφικά περιβάλλοντα

Αντικειμενοστραφής Προγραμματισμός

- Αν και το μάθημα έγινε σε Java, οι βασικές αρχές είναι οι ίδιες και για άλλες αντικειμενοστραφείς γλώσσες, και μπορείτε να μάθετε πολύ γρήγορα μια οποιαδήποτε άλλη γλώσσα προγραμματισμού
  - Μπορείτε να μάθετε C# σε μια βδομάδα
  - Η C++ είναι λίγο πιο μπερδεμένη γιατί πρέπει να κάνετε μόνοι σας τη διαχείριση μνήμης

Εξετάσεις

- Οι εξετάσεις θα είναι με ανοιχτά βιβλία και σημειώσεις
- Οι ερωτήσεις θα είναι στο πνεύμα των εργαστηρίων και των ασκήσεων
  - Κατά κύριο λόγο θα είναι προγραμματιστικές, αλλά μπορεί να σας ρωτήσουμε να ονομάσετε ένα μηχανισμό, ή να εξηγήσετε γιατί συμβαίνει κάτι
- Καλή επιτυχία!