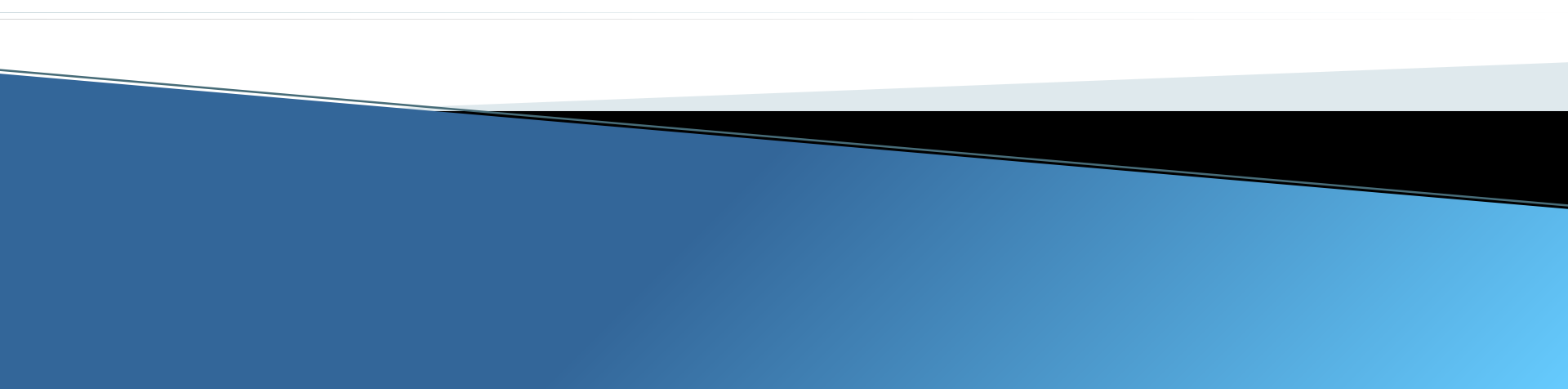
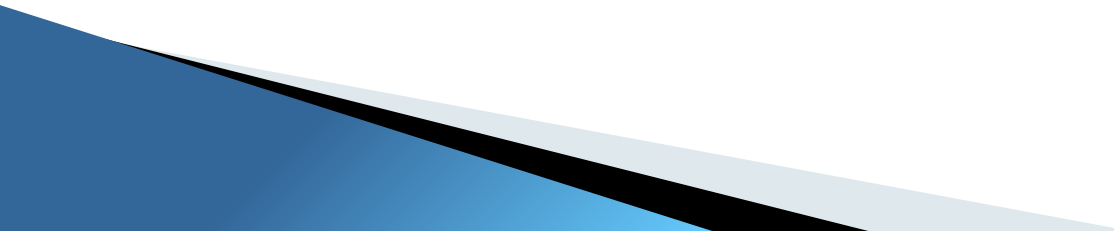


# Interfaze Grafikoak

SOFTWARE INGENIARITZA



# EDUKIAK

- ▶ Motibazioa eta helburuak
  - ▶ GUI osagaiak
  - ▶ Objektu grafikoen hierarkiak (AWT, Swing)
  - ▶ Layout kudeatzaileak
- 

# Motibazioa

- ▶ Programazio lengoaia modernoek Erabiltzaile Interfaze Grafikoak (GUI - Grafical User Interface) garatzeko tresnak eskeini
- ▶ Javak hurrengoa egiten du posible:
  - GUI-en diseinu eta programazio azkar eta sinplea.
  - AWT (Abstract Window Toolkit) klase paketea.
  - **Swing** klase paketea: AWT-ren eboluzioa, klase eta malgutasun gehiago eskeintzen ditu.

# Helburuak

- ▶ GUIak eraikitzeko Java klaseen hierarkiaren diseinua ulertu
- ▶ GUI-ak eraikitzeko Eclipse programazio ingurunea erabiltzen ikasi

# Java programa motak

## ► Aplikazioak

Zuzenean Java ingurune batetan exekutatu. Motak:

- Kotsola modua
  - ✓ Teklatu bitarteko interakzioa
  - ✓ Testuan oinarritutako interfazea
- Interfaze grafikodun aplikazioak (GUI)
  - ✓ Datu sarrera eta irteerarako leiho grafikoak
  - ✓ Ikonoak
  - ✓ Sarrera gailuak (arratoia, teklatua)
  - ✓ Interakzio zuzena

## ► Applet-ak

Nabigatzaile batetan (edo applet bistaratzailean, Appletviewer) exekutatzen diren aplikazio txikiak


# GUIrentzako osagai bibliotekak

- ▶ Abstract Windowing Toolkit (AWT)
  - Plataformaren menpeko GUI tresna.
  - JDK 1.1.5 bertsiora arte, estandarra.
- ▶ **Swing**
  - Plataformaren GUI tresna independentea
  - Funtzionalitate eta API berriak
    - Irisgarritasun APIa beharrian berezien erabiltzaileentzat

# Oinarrizko osagaiak

- ▶ GUI osagaiak (*widgets*): interfazeko objektu bisualak
  - ✓ Programa grafikoa, osagai anidatuen multzoa: leihoak, kontendoreak, menuak, botoiak, e.a..
- ▶ Kokapen kudeatzaileak (*layout managers*)
  - ✓ Interfazaren osagai grafikoaren antolaketa kudeatu
- ▶ Grafiko eta testu sorrera – Graphics klasea
  - ✓ Marrak, irudiak, koloreztatzea,...
- ▶ Interaktibitatea: ebentoen kudeaketa
  - ✓ Teklatua
  - ✓ Sagua

# Edukiontzia eta osagaiak

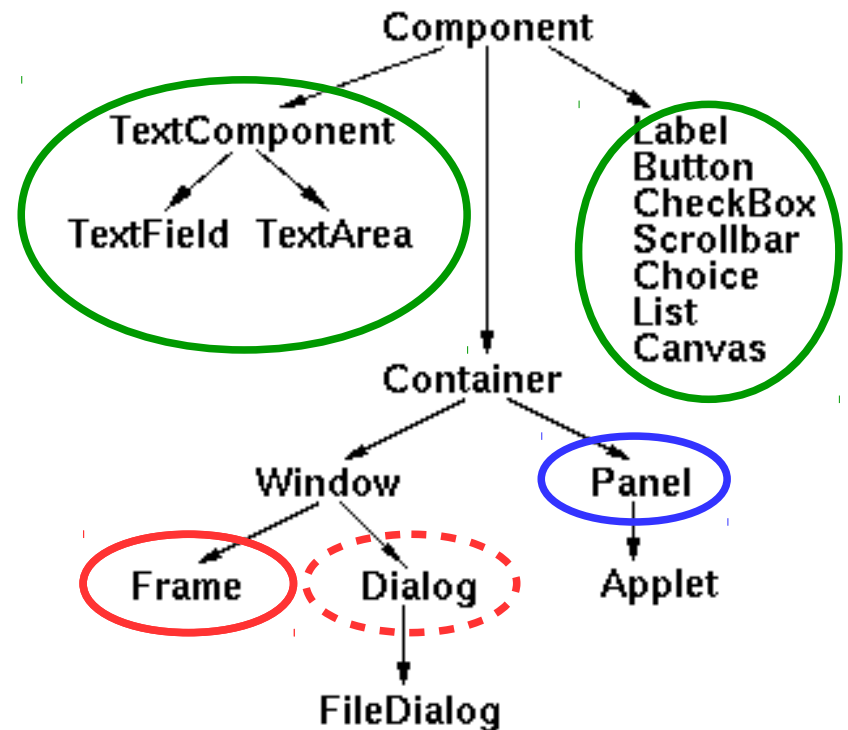
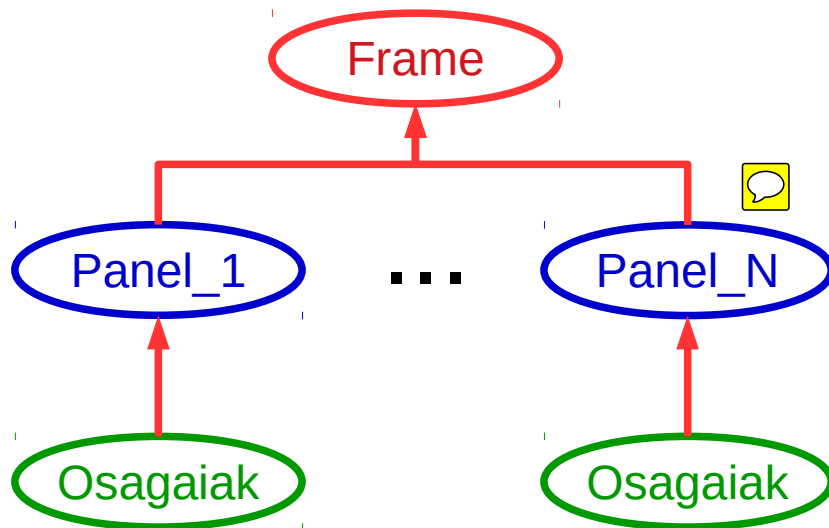
- ▶ Java GUIak bi motako elementutan oinarritu:
  - ✓ Edukiontzia (*containers*): beste osagaiak taldekatu eta erakutsi.
  - ✓ Osagaiak (*components*): botoiak, etiketak, scrollbarak, edukiontzia, e.a..
- ▶ GUI guztiek, gutxienez, *container* bat:
  - ✓ *JFrame*: leiho nagusia 
  - ✓ *JDialog*: elkarrizketa leihoa
  - ✓ *JApplet*: applet leihoa




# Edukiontziak eta osagaiak

- ▶ Java GUIa eraikitzeko, leiho baten eremuan osagaiak gehitu.
- ▶ Leiho bat edukiontzi bat da; hots, osagai multzo bat bere baitan duen elementua.

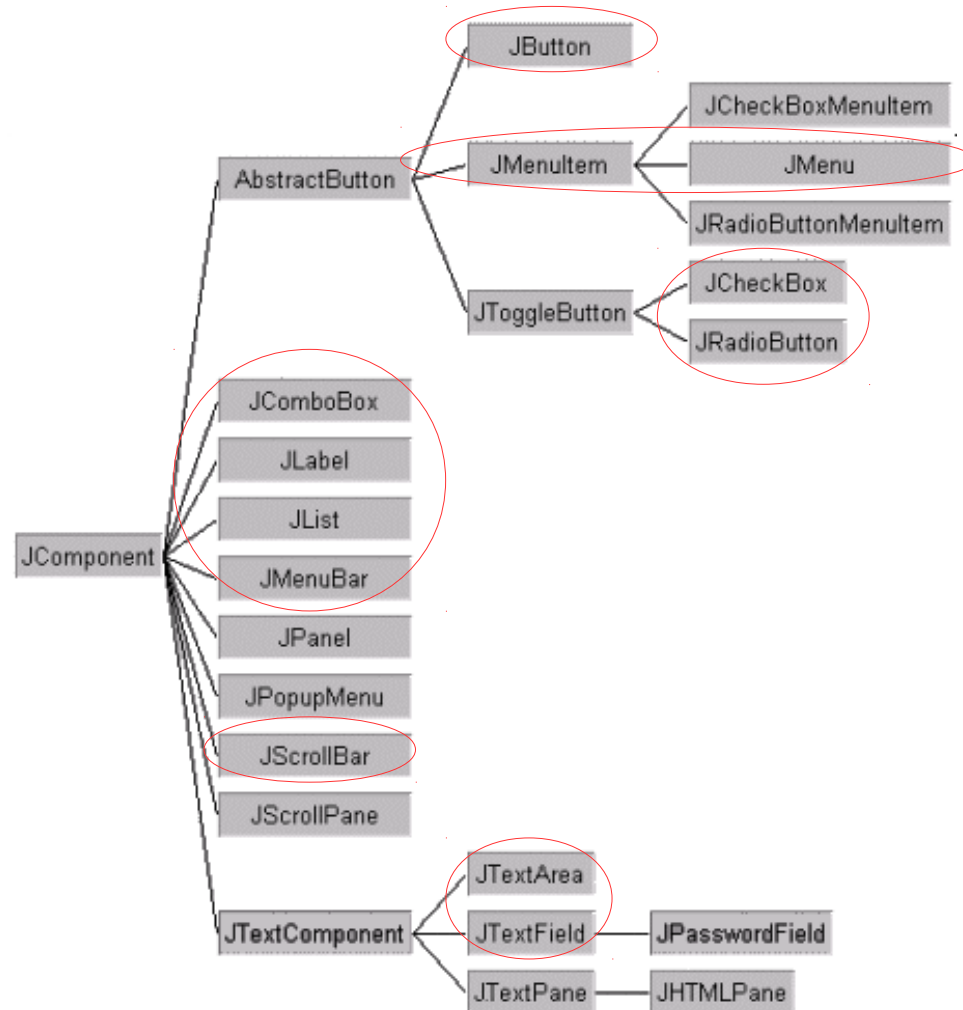
# AWT osagaien hierarkia



# Swing osagaiak

- ▶ Edukiontzia: beste osagai (edo edukiontzia) barnean
  - ✓ Osagaiak edukiontzia gehitu ahal dizkiogu. Batzuetan, entitate bakarra bezala tratatu.
  - ✓ Diseinu kudeatzaile (*layout*)  baten bitartez, osagaiek pantailan duten kokapena kudeatu.
  - ✓ Adibidez: **JPanel**, **JFrame**, **JApplet**
- ▶ Erabiltzaile-interfazearen osagaiak: botoiak, zerrendak, menuak, testu eremuak, e.a.
- ▶ Leihoak eraikitzeko osagaiak: leihoak, markoak, menu barrak, elkarriketa leihoak, e.a.

# Swing hierarkia



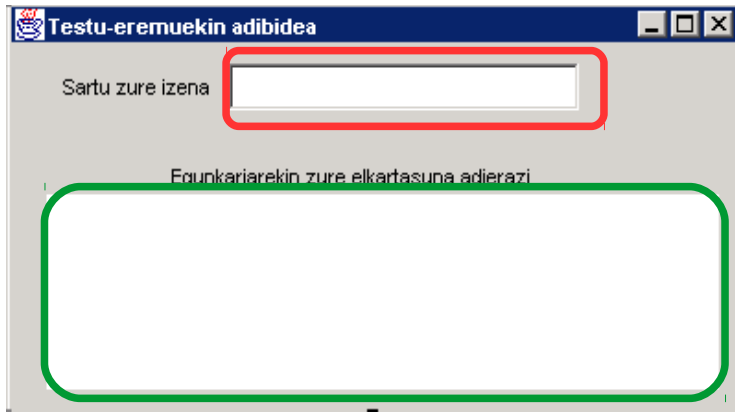
# Osagaiak: JTextField

- ▶ **TextField (JTextField) klasea:**
  - Sarrea moduan, testu-lerro bakarra sartzeko gunea.
  - Irtera moduan, testu-lerro bakarra ateratzeko gunea.

# Osagaiak: JTextField

- ▶ **TextArea (JTextArea) klasea:**
  - Sarrea moduan, hainbat testu-lerro sartzeko gunea.
  - Irtera moduan, hainbat testu-lerro ateratzeko gunea.

# Adibidea: JTextField+JTextArea



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

```
public class TestuEremuak extends JFrame {  
    JLabel jLabel1 = new JLabel();  
    JTextField jTextField1 = new JTextField();  
    JLabel jLabel2 = new JLabel();  
    JTextArea jTextArea1 = new JTextArea();  
  
    TestuEremuak(){  
        try { jbInit();}  
        catch (Exception e) {e.printStackTrace();}  
    }  
  
    public jbInit(){  
        this.setTitle("Testu-eremuekin adibidea");  
        jLabel1.setText("Sartu zure izena");  
        jTextField1.setColumns(25);  
        jLabel2.setText("Egunkariarekin zure elkartasuna adierazi");  
        jTextArea1.setColumns(50);  
        jTextArea1.setRows(10);  
        this.getContentPane().add(jTextArea1, null);  
        this.getContentPane().add(jLabel2, null);  
        this.getContentPane().add(jTextField1, null);  
        this.getContentPane().add(jLabel1, null);  
    }  
    public static void main(String[] args){  
        Frame frame = new TestuEremuak();  
        frame.setVisible(true);  
    }  
}
```

# Osagaiak: Botoiak

- ▶ **Button (JButton) klasea:**
  - Botoiak sortu eta, ondoren, ekintzak burutu



# Adibidea: Botoiak

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

public class Botoiak extends JFrame
{
    JButton jButton1 = new JButton();
    JButton jButton2 = new JButton();
    JButton jButton3 = new JButton();

    public Botoiak() {
        this.setTitle("Botoien adibidea");
        jButton1.setText("Ireki");
        jButton2.setText("Gorde");
        jButton3.setText("Ezeztatu");
        this.getContentPane().add(jButton3, BorderLayout.EAST);
        this.getContentPane().add(jButton2, BorderLayout.CENTER);
        this.getContentPane().add(jButton1, BorderLayout.WEST);
    }

    public static void main(String[] args){
        Frame frame = new Botoiak();
        frame.setVisible(true);
    }
}
```

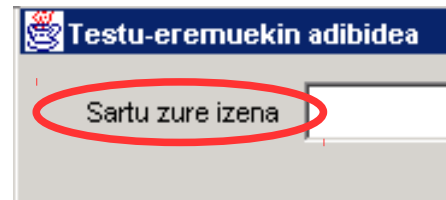


# Osagaiak: Etiketak



## ► Label (JLabel) klasea:

- Etiketak (*label*), erabiltzaileari informazioa emateko. Beste osagaien lagungarri.
- Etiketek kontenedoretan hiru lerratze:
  - Label.LEFT
  - Label.CENTER
  - Label.RIGHT



# Osagaiak: CheckBox-ak

## ► **Checkbox (JCheckBox/JRadioButton) klasea:**

- Erabiltzaileak aukera desberdinen artean bat hautatu dezake.
- Aukera aktibatu/desaktibatu egin daiteke.
- **Oharra:**
  - Karratuak direnean, aukera bat baino gehiago egin.
  - Borobilak direnean, aukera bakarra egin.

# Adibidea: CheckBox-ak



Aukera bakarrean klik egin ahal izateko,  
*ButtonGroup* berean sartu

```
private void initialize() {
    setTitle("CheckBox eta RadioButton");
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(100, 100, 450, 300);
    contentPane = new JPanel();
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
    contentPane.setLayout(new BorderLayout(0, 0));
    setContentPane(contentPane);
    contentPane.add(getPanel(), BorderLayout.NORTH);
    contentPane.add(getPanel_1(), BorderLayout.CENTER);
    buttonGroup = new ButtonGroup();
}

private JPanel getPanel() {
    if (panel == null) {
        panel = new JPanel();
        panel.add(getLblProbintzia());
        panel.add(getRdbtnAraba());
        panel.add(getRdbtnGipuzkoa());
        panel.add(getRdbtnBizkaia());
    } return panel;
}

private JPanel getPanel_1() {
    if (panel_1 == null) {
        panel_1 = new JPanel();
        panel_1.add(getLblAfizizioaAukeraEzazu());
        panel_1.add(getChckbxMusika());
        ...
    } return panel_1;
}

private JRadioButton getRdbtnBizkaia() {
    if (rdbtnBizkaia == null) {
        rdbtnBizkaia = new JRadioButton("Bizkaia");
        buttonGroup.add(rdbtnBizkaia);
    } return rdbtnBizkaia;
}
```

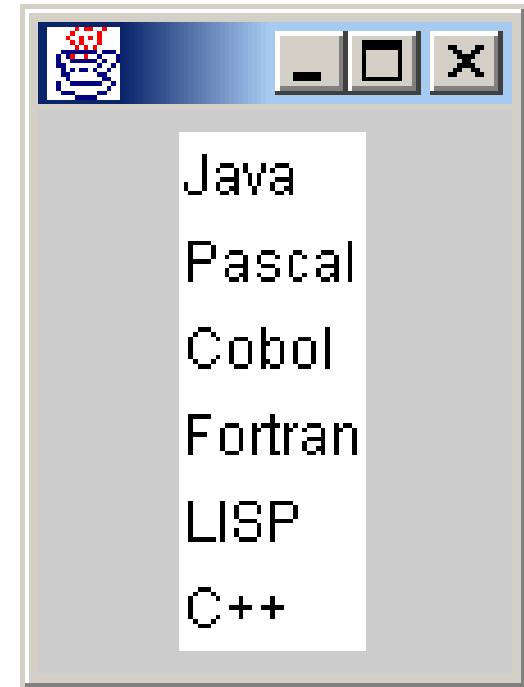
# Osagaiak: Zerrendak

## ► **List (JList) klasea:**

- Aukera desberdinak eskaini erabiltzaileari, zerrenda formatuan.
- Zerrenda *scroll* pantaila txiki batean azaldu, espazioa aurrezteko.

# Adibidea: JList zerrenda

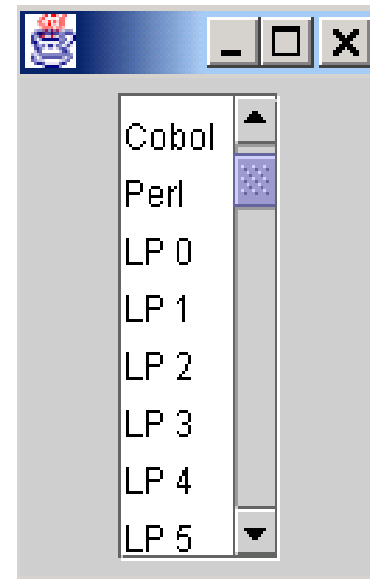
```
import javax.swing.*;
import java.awt.*;
import java.util.*;
public class Listak extends JFrame
{
    JList jList1; //new gero egingo da
    DefaultListModel elementuak = new DefaultListModel();
    JPanel jPanel1 = new JPanel();
    public Listak(){
        this.setTitle("Listen adibidea");
        elementuak.addElement("Java");
        elementuak.addElement("Pascal");
        elementuak.addElement("Cobol");
        elementuak.addElement("Perl");
        jList1 = new JList(elementuak);
        jPanel1.add(jList1, null);
        this.getContentPane().add(jPanel1, null);
        elementuak.addElement("LISP");
    }
    public static void main(String[] args){
        Frame frame = new Listak();
        frame.setVisible(true);
        frame.elementuak.addElement("C++");
        frame.setVisible(true);
    }
}
```



# Adibidea: JScrollPane zerrenda

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

public class ListakScrollekin extends JFrame {
    JPanel jPanel1 = new JPanel();
    JList jList1;
    Vector elementuak = new Vector();
    public ListakScrollekin(){
        this.getContentPane().add(jPanel1, null);
        elementuak.addElement("Java");
        elementuak.addElement("Pascal");
        elementuak.addElement("Cobol");
        elementuak.addElement("Perl");
        jList1 = new JList(elementuak);
        JScrollPane j = new JScrollPane(jList1);
        //Lista sartzen dugu scroll-a duen panel batean
        jPanel1.add(j,null);
        for (int i=0;i<50;i++) elementuak.addElement("LP "+i);
        pack();
    }
    public static void main(String[] args){
        Frame frame = new ListakScrollekin();
        frame.setVisible(true);
    }
}
```



# Adibidea: JComboBox zerrenda

```
import javax.swing.*;
import java.awt.*;
import java.util.*;
public class ComboBoxak extends JFrame {
    DefaultComboBoxModel elementuak = new DefaultComboBoxModel ();
    JPanel jPanel1 = new JPanel();
    JComboBox jComboBox1;//new gero egingo da

    public ComboBoxak(){
        elementuak.addElement("Java");
        elementuak.addElement("Pascal");
        elementuak.addElement("Cobol");
        elementuak.addElement("Perl");
        jComboBox1 = new JComboBox(elementuak);
        jPanel1.add(jComboBox1, null);
        this.getContentPane().add(jPanel1, null);
        elementuak.addElement("LISP");
        pack();
    }
    public static void main(String[] args){
        Frame frame = new ComboBoxak();
        frame.setVisible(true);
    }
}
```

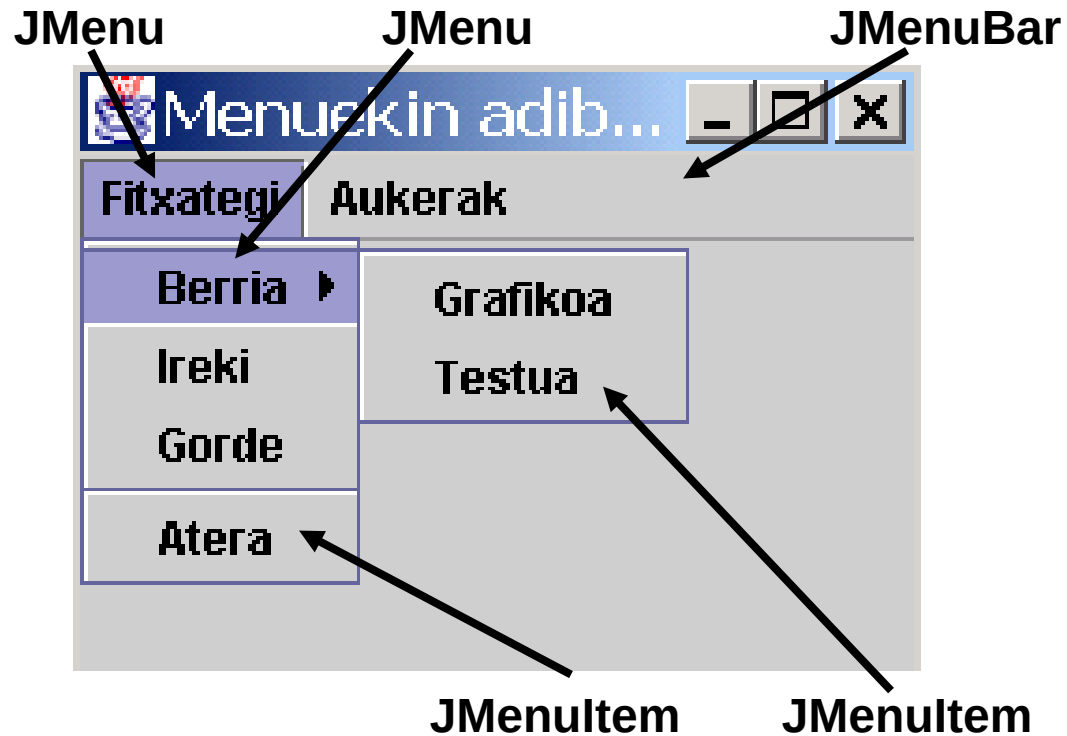




# Osagaiak: Menuak

- ▶ **Menu (JMenu/JMenuItem/JMenuBar):**
  - Aplikazioak garatzerako erabilgarria.
  - Orain arte ikusitako osagaiek baino egitura konplexuagoa.

# Osagaiak: Menuak



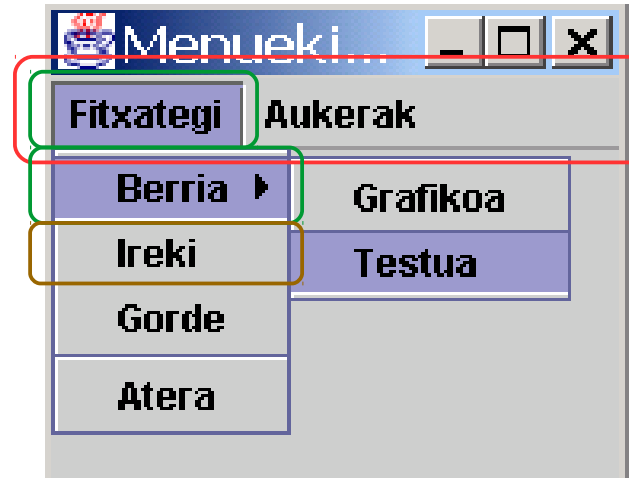
# Adibidea: Menuak

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

public class Menuak extends JFrame {
    JMenuBar menuBarra = new JMenuBar();
    JMenu fitxategi = new JMenu(); JMenu aukerak = new JMenu();
    JMenu berria = new JMenu(); JMenuItem ireki = new JMenuItem();
    JMenuItem gorde = new JMenuItem(); JMenuItem atera = new JMenuItem();
    ButtonGroup bg = new ButtonGroup();
    JMenuItem testua = new JMenuItem(); JMenuItem grafikoa = new JMenuItem();

    public Menuak() {
        this.setJMenuBar(menuBarra); this.setTitle("Menuekin adibidea");
        fitxategi.setText("Fitxategi"); aukerak.setText("Aukerak");
        berria.setText("Berria"); grafikoa.setText("Grafikoa");
        testua.setText("Testua"); ireki.setText("Ireki");
        gorde.setText("Gorde"); atera.setText("Atera");
        testua.setText("Testua");
        berria.add(grafikoa); berria.add(testua); berria.add(testua);
        fitxategi.add(berria); fitxategi.add(ireki); fitxategi.add(gorde);
        fitxategi.addSeparator(); fitxategi.add(atera);
        menuBarra.add(fitxategi);
        menuBarra.add(aukerak);
    }

    public static void main(String[] args){
        Frame frame = new Menuak();
        frame.setVisible(true);
    }
}
```



# Ariketa

Sor ezazu hurrengo itxurako panela:

Aukera bakarrean  
soilik klik egin  
beharko da

Erabiltzailea sartu

Botoi bat sakatu

A1	A2
B1	B2
C1	C2

☐ 1. aukera ☐ 2. aukera

Gustuko duzu? ☐ Bai ☐ Ez

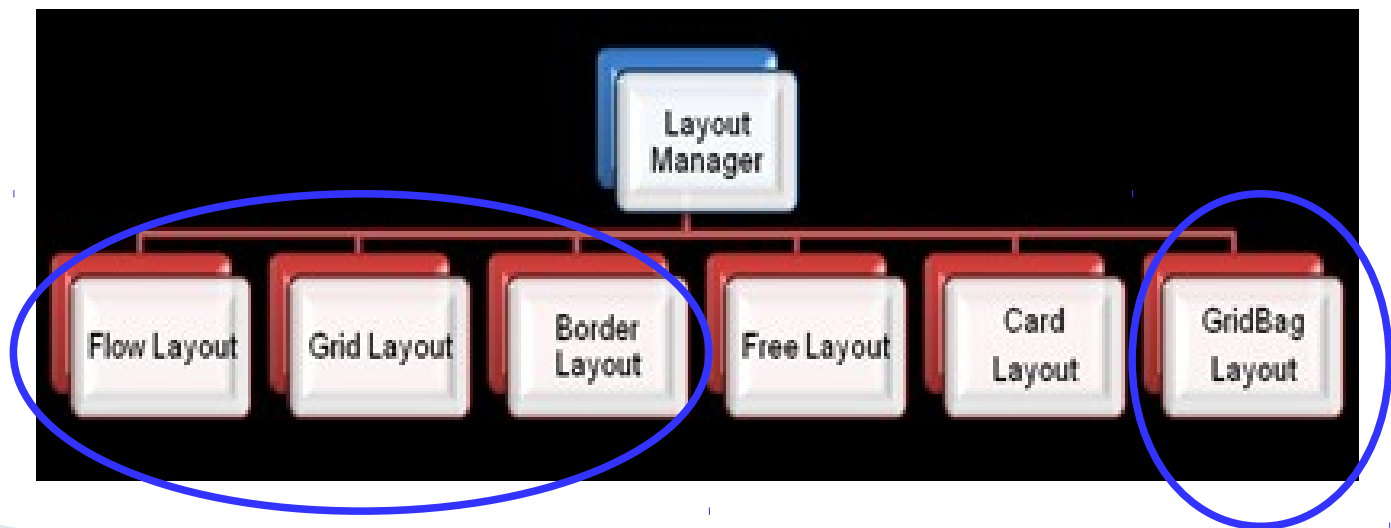
# Layout: Motibazioa

Zer hartu behar da kontutan GUI diseinuan?

- ▶ Objektuen **berdimentsionatzea**, leihoaren tamaina aldatzerakoan.
- ▶ **Kokapen egituratuagoa**

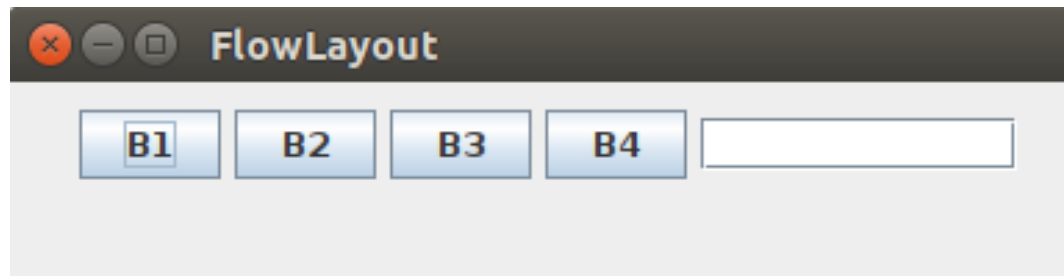
# Layout: deskribapena

- ▶ Layout kudeatzaileek objektu grafikoak kontenedore baten antolatu.
- ▶ Osagaien itxura zehatu, baita tamaina eta posizioa ere.



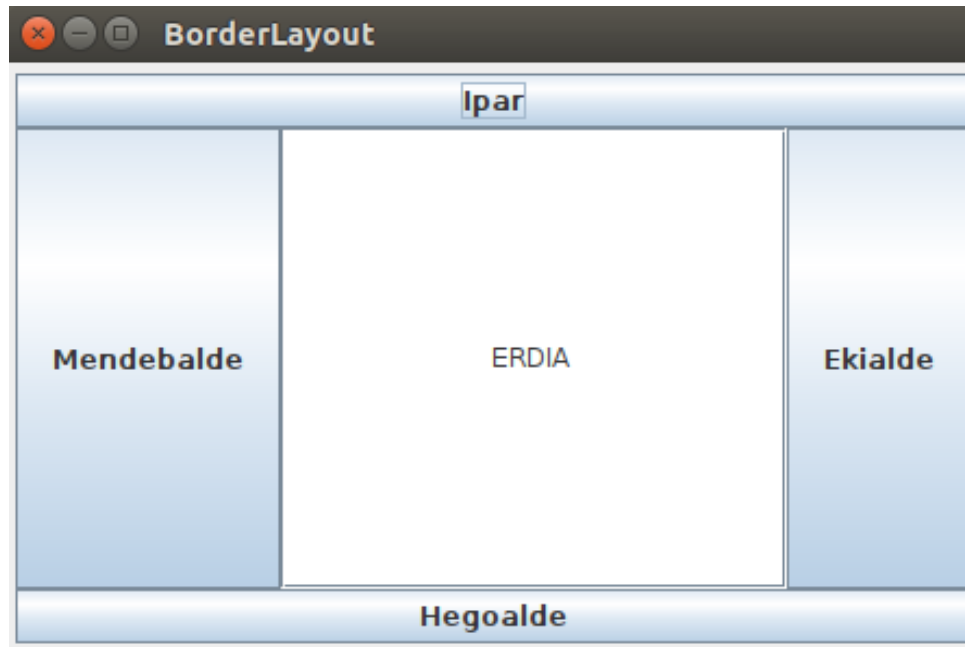
# Layout: FlowLayout

Osagaiak lerroetan edo zutabetan antolatu, eraikitzailean adierazitakoaren arabera. Lerroa edo zutabea betetzerakoan, berri bat hasi.



# BorderLayout

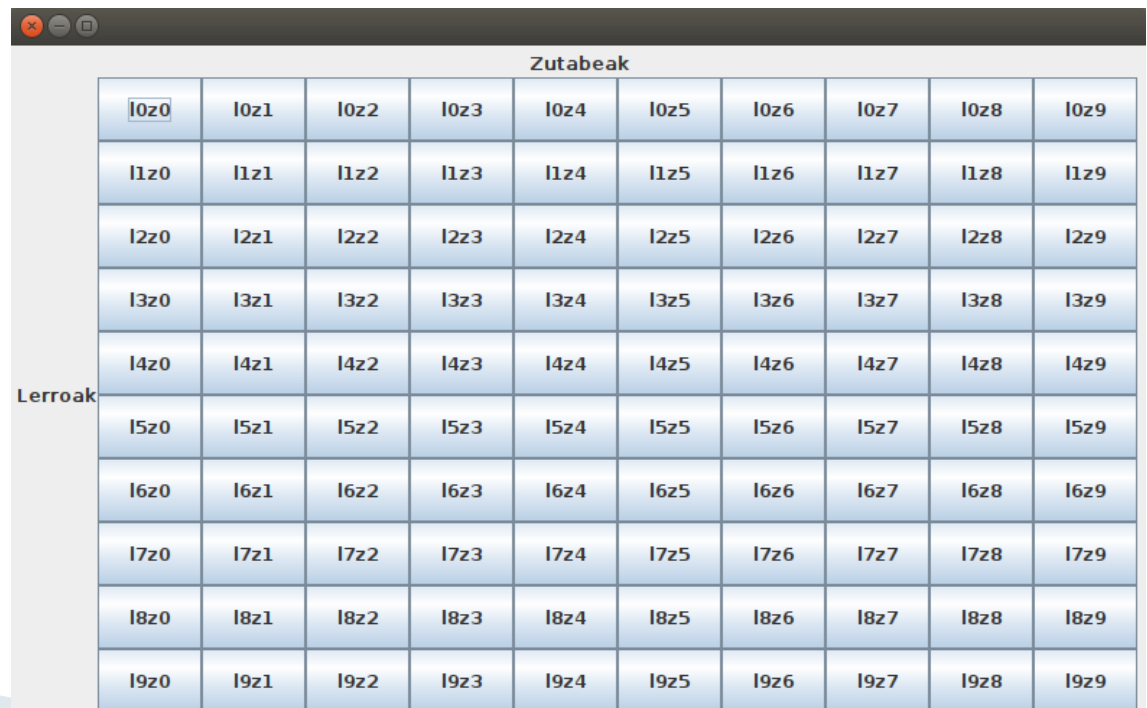
Osagaiak bost eskualdetan antolatu: iparralde, ekialde, hegoalde, mendebalde eta erdia.





# GridLayout

- ▶ Osagaiak bi dimentsiotako taulan antolatu, eta gelaxka oro tamaina berekoa.
- ▶ Gelaxka bakoitzean osagai bat kokatu.
- ▶ Osagaiak ezkerretik eskuinera eta lerroz lerro bete.



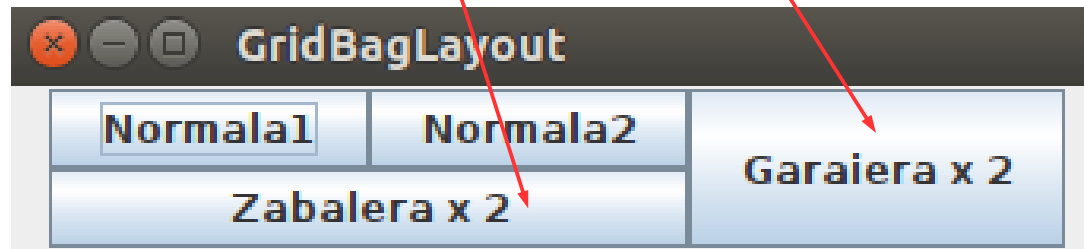
	l0z0	l0z1	l0z2	l0z3	l0z4	l0z5	l0z6	l0z7	l0z8	l0z9
	l1z0	l1z1	l1z2	l1z3	l1z4	l1z5	l1z6	l1z7	l1z8	l1z9
	l2z0	l2z1	l2z2	l2z3	l2z4	l2z5	l2z6	l2z7	l2z8	l2z9
	l3z0	l3z1	l3z2	l3z3	l3z4	l3z5	l3z6	l3z7	l3z8	l3z9
	l4z0	l4z1	l4z2	l4z3	l4z4	l4z5	l4z6	l4z7	l4z8	l4z9
Lerroak	l5z0	l5z1	l5z2	l5z3	l5z4	l5z5	l5z6	l5z7	l5z8	l5z9
	l6z0	l6z1	l6z2	l6z3	l6z4	l6z5	l6z6	l6z7	l6z8	l6z9
	l7z0	l7z1	l7z2	l7z3	l7z4	l7z5	l7z6	l7z7	l7z8	l7z9
	l8z0	l8z1	l8z2	l8z3	l8z4	l8z5	l8z6	l8z7	l8z8	l8z9
	l9z0	l9z1	l9z2	l9z3	l9z4	l9z5	l9z6	l9z7	l9z8	l9z9

# GridBagLayout

- ▶ Osagaiak bi dimentsiotako taulan antolatu, baina gelaxkek ez dute tamaina berekoak izan behar.
- ▶ Gelaxka barruko elementuek **GridBagConstraints**:
  - **gridx, gridy**: elementuaren koordenatuak
  - **gridwidth, gridheight**: elementuak beteko dituen gelaxka kopurua.
  - **fill**: birdimentsionatzea mota (*BOTH, HORIZONTAL, VERTICAL*)
  - **weightx, weighty**: birdimentsionatze maila
  - **ipadx, ipady**: elementuaren tamainaren gehigarria
  - **insets**: elementuaren kanpoko betegarri tamaina

# Adibidea: GridBagLayout

```
GridBagConstraints gbc_btnNormala = new GridBagConstraints();
gbc_btnNormala.gridx = 0;
gbc_btnNormala.gridy = 0;
contentPane.add(getBtnNormala(), gbc_btnNormala);
GridBagConstraints gbc_btnNormala_1 = new GridBagConstraints();
gbc_btnNormala_1.gridx = 1;
gbc_btnNormala_1.gridy = 0;
contentPane.add(getBtnNormala_1(), gbc_btnNormala_1);
GridBagConstraints gbc_btnGaraieraX = new GridBagConstraints();
gbc_btnGaraieraX.fill = GridBagConstraints.BOTH;
gbc_btnGaraieraX.gridheight = 2;
gbc_btnGaraieraX.gridx = 2;
gbc_btnGaraieraX.gridy = 0;
contentPane.add(getBtnGaraieraX(), gbc_btnGaraieraX);
GridBagConstraints gbc_btnZabaleraX = new GridBagConstraints();
gbc_btnZabaleraX.fill = GridBagConstraints.HORIZONTAL;
gbc_btnZabaleraX.gridwidth = 2;
gbc_btnZabaleraX.gridx = 0;
gbc_btnZabaleraX.gridy = 1;
contentPane.add(getBtnZabaleraX(), gbc_btnZabaleraX);
```



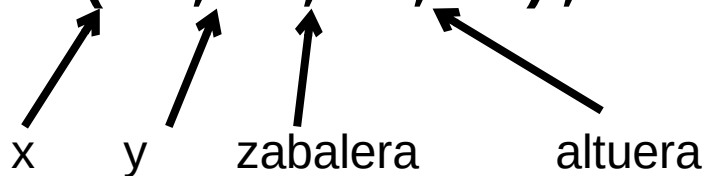
# Layout

- ▶ Osagaia koordenatu zehatzetan jartzeko, *layout* kendu

```
osagaia.setLayout(null);
```

- ▶ Esaterako:

```
this.setLayout(null);  
textField1.setBounds(15, 20, 50, 60);
```



x y zabalera altuera

The diagram shows four arrows pointing from labels below to the parameters of the `setBounds` method in the code above. The label 'x' points to the first parameter '15'. The label 'y' points to the second parameter '20'. The label 'zabalera' (width) points to the third parameter '50'. The label 'altuera' (height) points to the fourth parameter '60'.

Erabiltzailea sartu

☐ A  
☒ B  
☐ C

Botoi bat sakatu

A2	B1
A1	C1
C1	B2

☐ 1. aukera ☐ 2. aukera

Gustuko duzu? ☐ Bai ☐ Ez

Erabiltzailea sartu

☐ A  
☒ B  
☐ C

Botoi bat sakatu

A2	B1
A1	C1
C1	B2

☐ 1. aukera ☐ 2. aukera

Gustuko duzu? ☐ Bai ☐ Ez

**Layout-aren abantaila:**  
leihoaren forma aldatzen  
badugu, osagaiak  
automatikoki egoera  
berrira egokitu!!

```

this.setSize(new Dimension(400, 300));
jPanel1.setBounds(new Rectangle(0, 0, 392, 273));
jPanel1.setLayout(null);
jCheckBox1.setLabel("Opc. 1");
jCheckBox1.setBounds(new Rectangle(130, 165, 130, 55));
jCheckBox2.setLabel("Opc. 2");
jCheckBox2.setBounds(new Rectangle(260, 165, 130, 55));
jPanel3.setBounds(new Rectangle(0, 165, 130, 55));
jPanel3.setLayout(null);
jPanel2.setBounds(new Rectangle(0, 238, 392, 35));
jPanel2.setLayout(null);

```

**Layout barik, osagai guztien koordenatuak definitu beharra!!**

Erabiltzailea sartu

Botoi bat sakatu

A1	A2
B1	B2
C1	C2

☐ 1. aukera ☐ 2. aukera

Gustuko duzu? ☐ Bai ☐ Ez

► Baina...

- Framea berdimentsionatzean osagaian zeuden lekuan geratzen dira.

Erabiltzailea sartu

Botoi bat sakatu

☐ A

☐ B

☐ C

A1	A2
B1	B2
C1	C2

☐ 1. aukera ☐ 2. aukera

Gustuko duzu? ☐ Bai ☐ Ez

Erabiltzailea sartu

Botoi bat sakatu

☐ A

☐ B

☐ C

A2
A1
C1

Erabiltzailea sartu

Botoi bat sakatu

☐ A

☐ B

☐ C

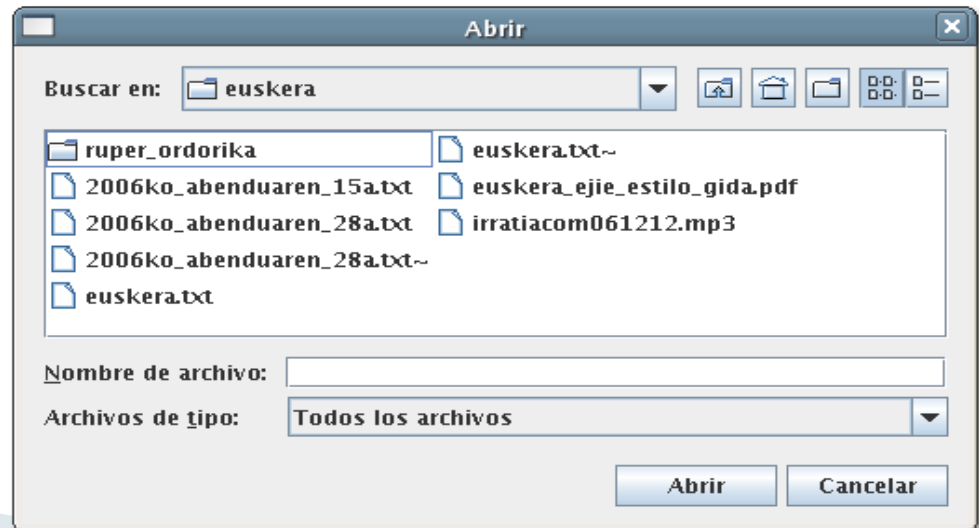
A2
A1
C1

☐ 1. aukera

Gustuko duzu? ☐ Bai

# Beste edukiontzi batzuk

- ▶ **Dialog/JDialog** klaseak
  - Erabiltzailearen datuak irakurtzeko leihoa.
  - MODAL ezaugarria jartzen badiogu, aktiboa dagoen bitartean ezin izango da beste leiho batetara aldatu.
- ▶ **JFileChooser** klasea





# Layout Ariketa 1

Sor ezazu hurrengo panela:

Aukera bakarrean  
soilik klik egin  
beharko da

Erabiltzailea sartu

Botoi bat sakatu

A1	A2
B1	B2
C1	C2

☐ 1. aukera ☐ 2. aukera

Gustuko duzu? ☐ Bai ☐ Ez

# Layout Ariketa 1

The image shows a Java Swing window titled "Erabiltzailea sartu" (User Login). The window is divided into four main regions by a red border, illustrating the use of BorderLayout:

- Top Region (Green border):** A label "Botoi bat sakatu" (Click a button) is positioned at the top, aligned to the center. It is associated with the `BorderLayout.NORTH` constraint.
- Center Region (Blue border):** A panel containing a grid of 6 cells (3 rows by 2 columns). The cells are labeled A1, A2, B1, B2, C1, and C2. To the left of the grid are three radio buttons labeled A, B, and C. Below the grid are two checkboxes labeled "1. aukera" and "2. aukera". This panel is associated with the `BorderLayout.CENTER` constraint.
- Bottom Region (Yellow border):** A panel containing the text "Gustuko duzu?" (Do you like it?) followed by two checkboxes labeled "Bai" (Yes) and "Ez" (No). This panel is associated with the `BorderLayout.SOUTH` constraint.

Annotations and Layout Details:

- Panel BorderLayout:** The entire window content area is managed by a `BorderLayout` object.
- Panel GridLayout(4,3) BorderLayout.CENTER:** The central grid of 6 cells is managed by a `GridLayout(4,3)` within the center region.
- Panel FlowLayout-ekin BorderLayout.SOUTH:** The bottom panel is managed by a `FlowLayout` within the south region.