

# SWING

## ZAGADNIENIA:

- wprowadzenie,
- kontenery i komponenty,
- LayoutManager,
- komponenty tekstowe.

## MATERIAŁY:

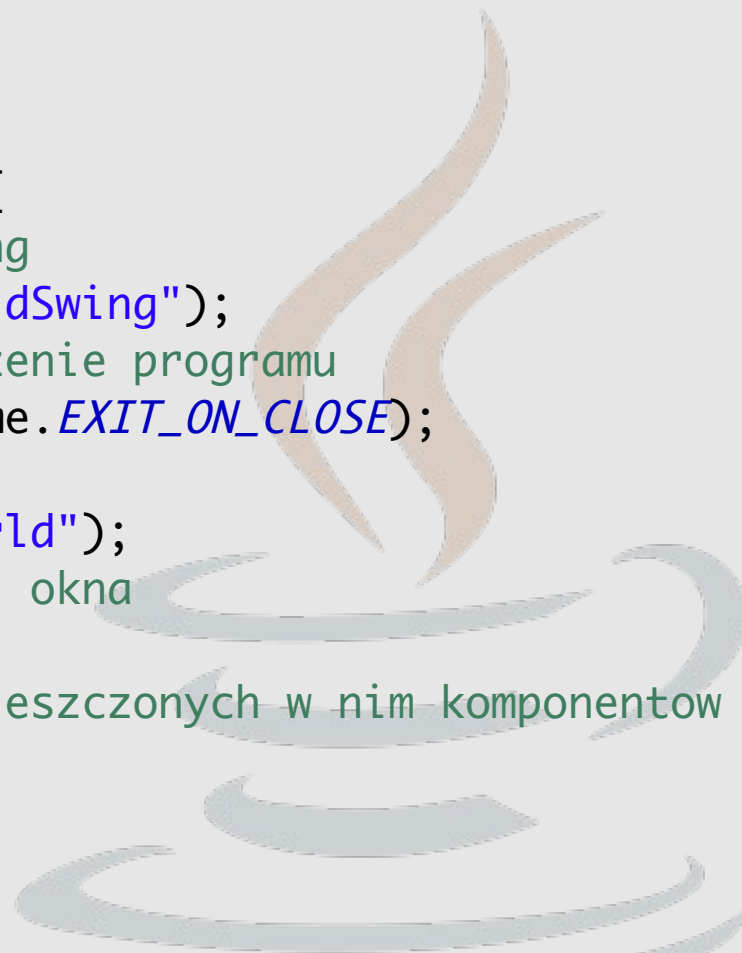
<http://docs.oracle.com/javase/tutorial/uiswing/>



# SWING

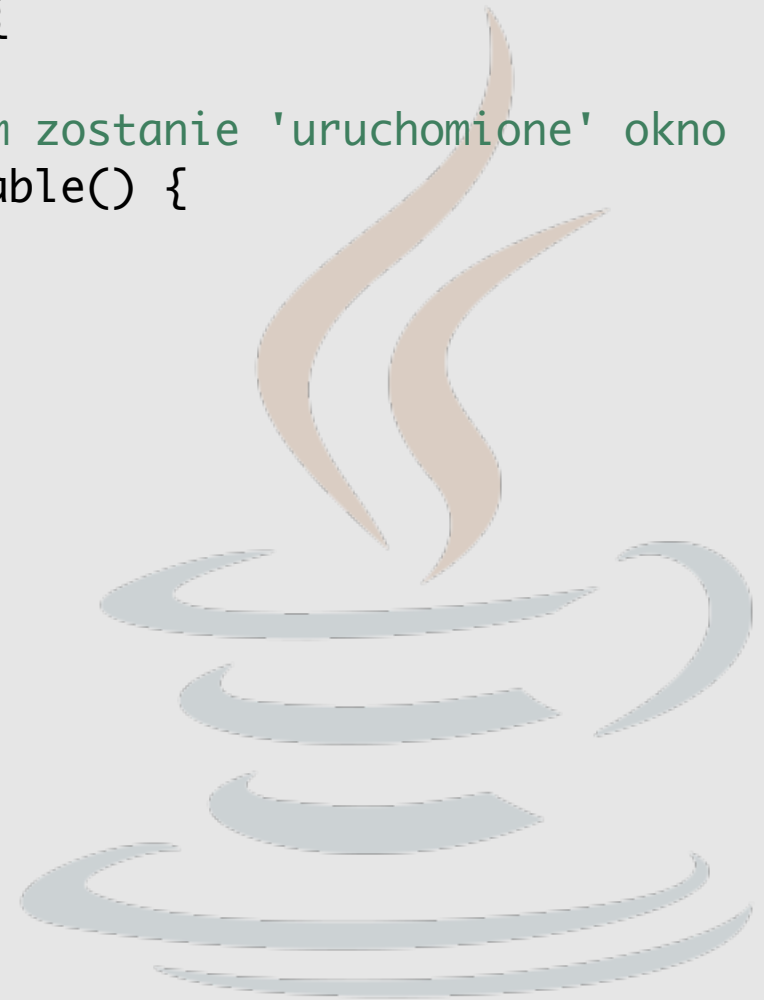
```
import javax.swing.*;

public class HelloWorldSwing {
    private static void createAndShowGUI() {
        // nowe okno o tytule HelloWorldSwing
        JFrame frame = new JFrame("HelloWorldSwing");
        // zamknięcie okna spowoduje zakończenie programu
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        // nowy napis
        JLabel label = new JLabel("Hello World");
        // napis jest dodawany do zawartosci okna
        frame.getContentPane().add(label);
        // dopasowanie rozmiarow okna do umieszczonych w nim komponentow
        frame.pack();
        // wyświetlenie okna
        frame.setVisible(true);
    }
}
```



# SWING

```
public static void main(String[] args) {  
  
    // stworzenie nowego watku, w którym zostanie 'uruchomione' okno  
    SwingUtilities.invokeLater(new Runnable() {  
        public void run() {  
            createAndShowGUI();  
        }  
    });  
}
```



# KONTENERY

Każda aplikacja wykorzystująca Swing używa co najmniej jeden kontener najwyższego poziomu.

- JFrame,
- JDialog,
- JApplet.

W takim kontenerze umieszcza się kolejne kontenery lub komponenty:

```
frame.getContentPane().add(yellowLabel, BorderLayout.CENTER);
```

```
// A tutaj tworzymy panel (kontener) dodajemy do niego komponenty  
// a następnie umieszczamy go w kontenerze najwyższego poziomu (frame)
```

```
JPanel contentPane = new JPanel(new BorderLayout());  
contentPane.setBorder(someBorder);  
contentPane.add(someComponent, BorderLayout.CENTER);  
contentPane.add(anotherComponent, BorderLayout.PAGE_END);  
frame.getContentPane().add(contentPane)
```

# KONTENERY

Za rozmieszczenie komponentów w kontenerze odpowiedzialny jest obiekt typu **LayoutManager**. Możemy go określić za pomocą metody **setLayout()**. Domyślnie kontenery używają instancji **FlowLayout()**. Jeśli chcemy "samodzielnie" określać pozycje komponentów należy usunąć LayoutManagera: **container.setLayout(null)**.

Do kontenerów najwyższego poziomu można także dodać menu: **frame.setJMenuBar()**.

# KOMPONENTY

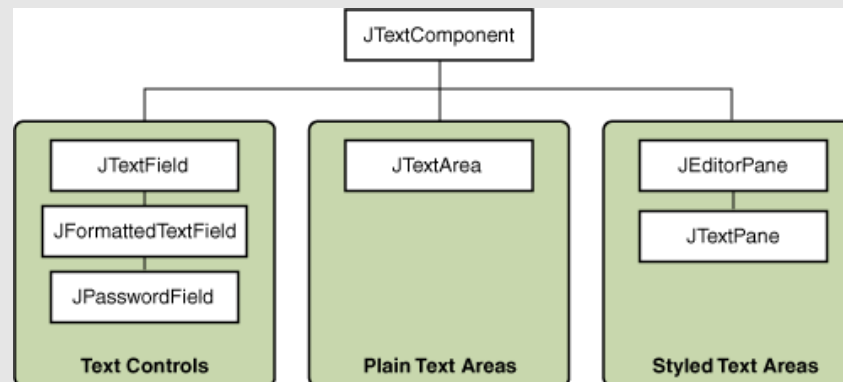
Wszystkie komponenty rozszerzają klasę **JComponent**. Klasa **JComponent** implementuje następujące funkcjonalności:

- podpowiedzi (**setToolTipText()**),
- ramki (**setBorder()**),
- styl (**UIManager.setLookAndFeel()**),
- dodatkowe właściwości (**setClientProperties()**),
- rozmiary (layout)
- przystępność (accessibility)
- przeciągnij i upuść,
- podwójne buforowanie,
- wiązanie klawiszy.



# KOMPONENTY TEKSTOWE

Komponenty tekstowe (**JTextComponent**) dzielą się na trzy kategorie:



# KOMPONENTY

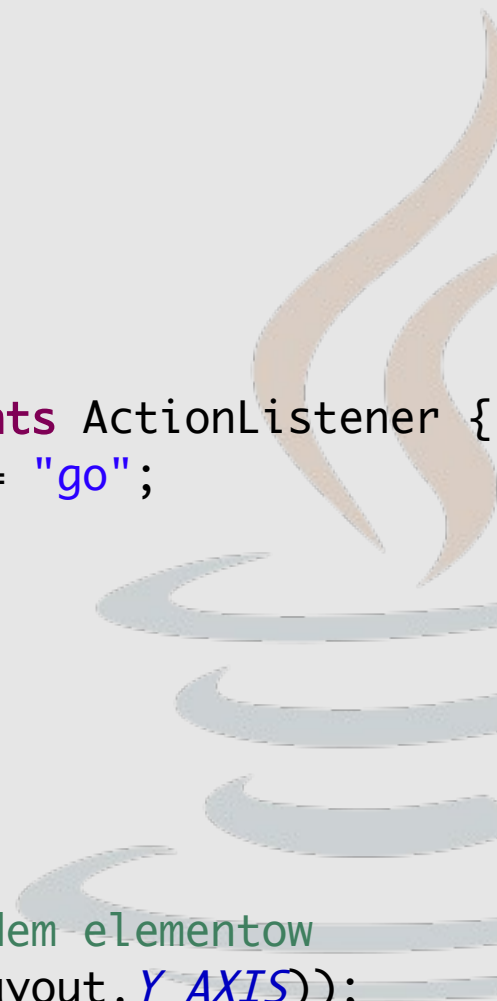
```
import java.awt.*;
import java.awt.event.*;
import java.io.*;
import java.net.URL;

import javax.swing.*;

public class Browser extends JFrame implements ActionListener {
    private static final String COMMAND_GO = "go";

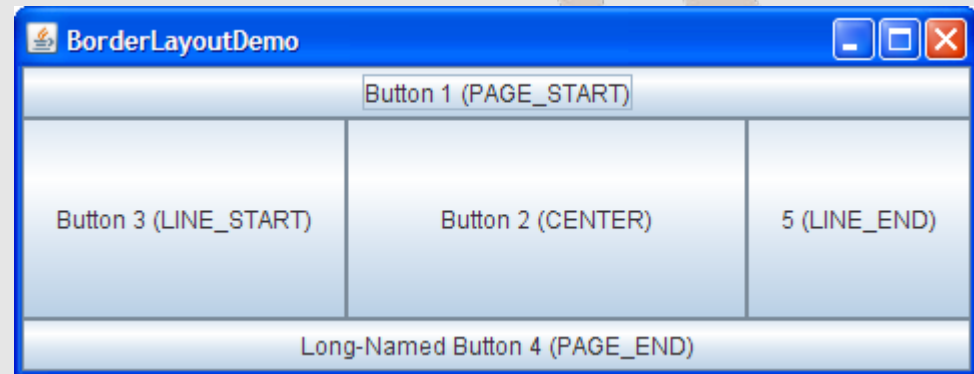
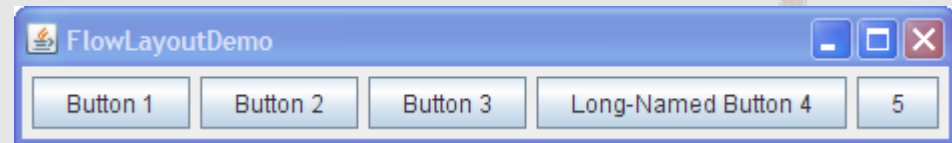
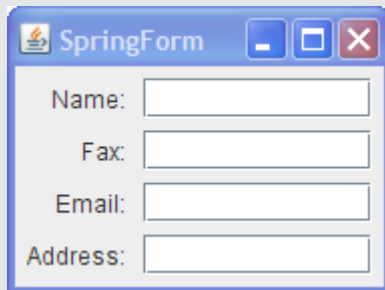
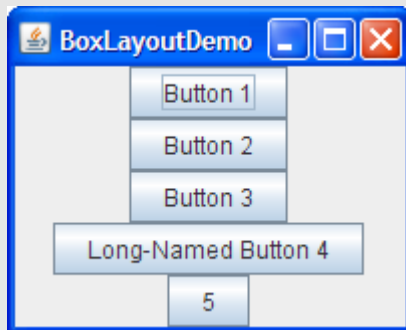
    private JEditorPane webpage;
    private JTextField url;
    private JTextArea htmlPage;

    private JPanel createMainPanel() {
        JPanel mp = new JPanel();
        // panel z kolumnowym (Y_AXIS) układem elementów
        mp.setLayout(new BorderLayout(mp, BorderLayout.Y_AXIS));
    }
}
```

A large, faint watermark of the Java logo is visible on the right side of the slide. It consists of a stylized coffee cup with steam rising from it, rendered in a light gray color.



# LAYOUT MANAGER

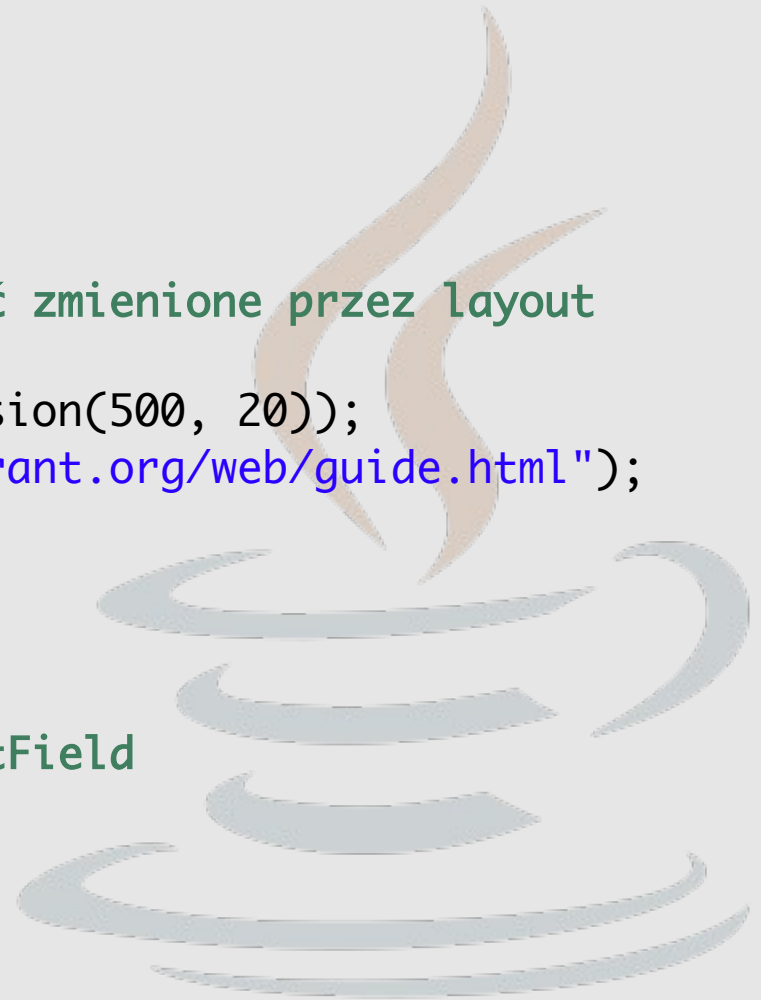


# KOMPONENTY

```
// panel z domyślnym FlowLayout
JPanel p = new JPanel();
this.url = new JTextField();

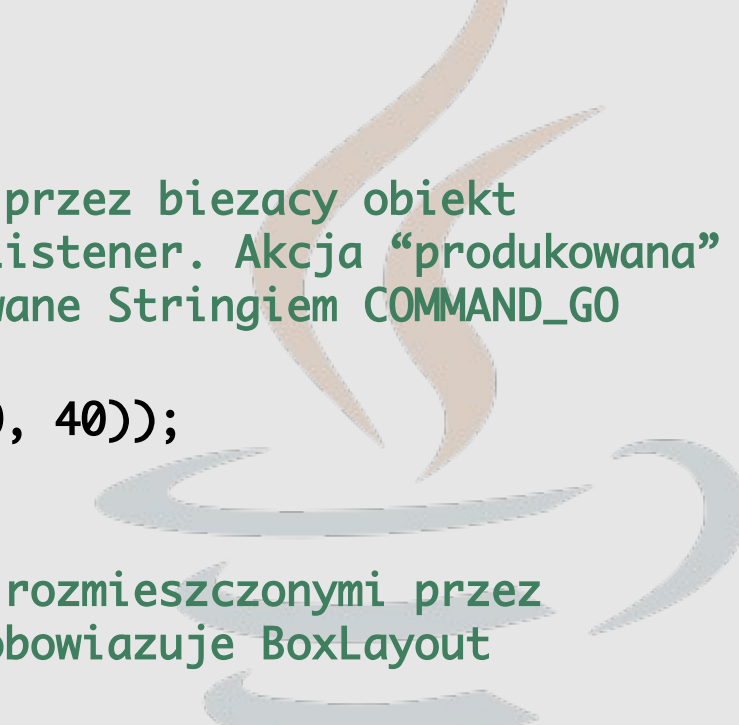
// sugerowane rozmiary – mogą zostać zmienione przez layout
// managera
this.url.setPreferredSize(new Dimension(500, 20));
this.url.setText("http://www.simongrant.org/web/guide.html");
JLabel l = new JLabel("adres");
// label opisujący url
l.setLabelFor(this.url);

// dodajemy do panelu JLabel i JTextField
p.add(l);
p.add(this.url);
```



# KOMPONENTY

```
// tworzymy przycisk
JButton b = new JButton("Go");
b.setActionCommand(COMMAND_GO);
// ktorego "akcje" beda obslugiwane przez biezacy obiekt
// (implementujacy interfejs ActionListener. Akcja "produkowana"
// przez przycisk bedzie identyfikowane Stringiem COMMAND_GO
b.addActionListener(this);
b.setPreferredSize(new Dimension(100, 40));
// dodanie przycisku do panelu
p.add(b);
// dodanie panelu p (z komponentami rozmieszczonymi przez
// FlowLayout) do panelu, w ktorym obowiazuje BorderLayout
mp.add(p);
```



# KOMPONENTY

```
this.webpage = new JEditorPane();
this.htmlPage = new JTextArea();
try {
    // wczytujemy zawartosc strony "startowej"
    this.setPage(
        new URL("http://www.simongrant.org/web/guide.html"));
} catch (IOException e) { }
// Tworzymy panel z zakladkami
JTabbedPane tp = new JTabbedPane();
tp.setPreferredSize(new Dimension(600, 400));
// pole tekstowe webpage umieszczamy wewnatrz panela
// scrollowanego. Dzieki temu zawartosc okienka bedzie mogla
// zajmowac wiecej miejsca niz widok
JScrollPane sp = new JScrollPane(this.webpage);
// zakladka "page" bedzie zawierac webpage (wewnatrz JScrollPane)
tp.add("page", sp);
```



# KOMPONENTY

```
// zakładka "html" będzie zawierać htmlPage (wewnątrz JScrollPane)
sp = new JScrollPane(this.htmlPage);
tp.add("html", sp);
```

```
// przygotowany JTabbedPane zostaje dodany do panelu mp
mp.add(tp);
return mp;
```

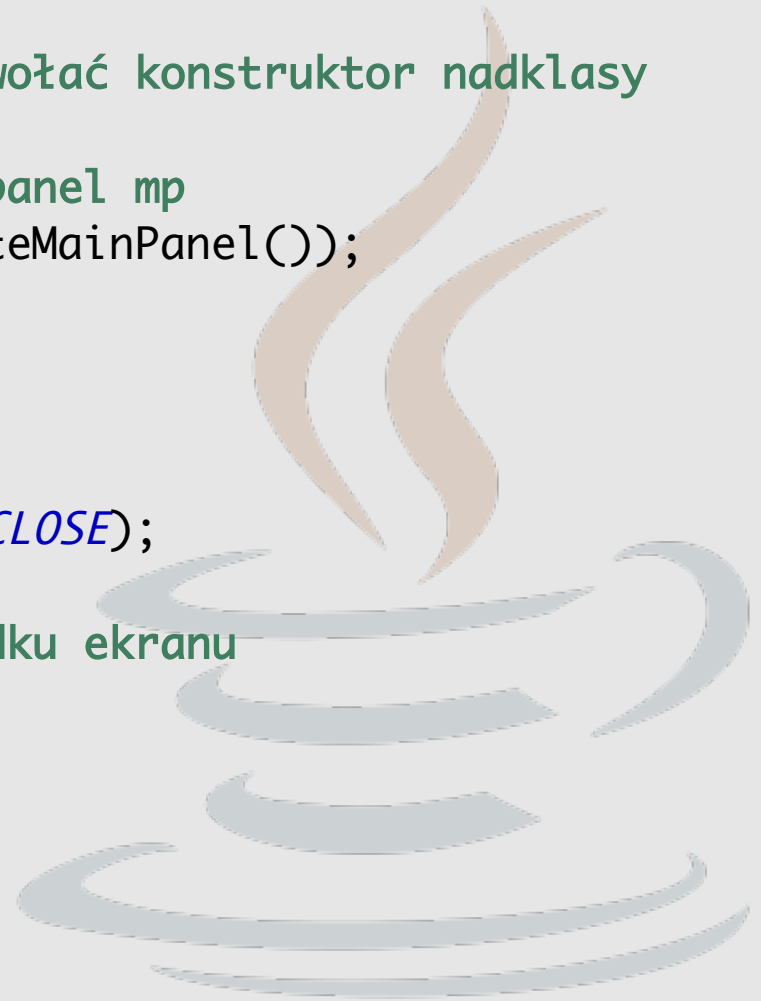
```
}
```

```
private void setPage(URL page) throws IOException {
    String s;
    this.webpage.setPage(page);
    BufferedReader br = new BufferedReader(new InputStreamReader(
        page.openStream()));

    while ((s = br.readLine()) != null)
        this.htmlPage.append(s + "\n");
}
```

# KOMPONENTY

```
public Browser() {  
    // zawsze na początku powinniśmy wywołać konstruktor nadklasy  
    super();  
    // zawartoscia okna Browser bedzie panel mp  
    this.getContentPane().add(this.createMainPanel());  
}  
  
public static void createAndShow() {  
    Browser b = new Browser();  
    b.setDefaultCloseOperation(EXIT_ON_CLOSE);  
    b.pack();  
    // okno zostanie umieszczone na srodku ekranu  
    b.setLocationRelativeTo(null);  
    b.setVisible(true);  
}
```



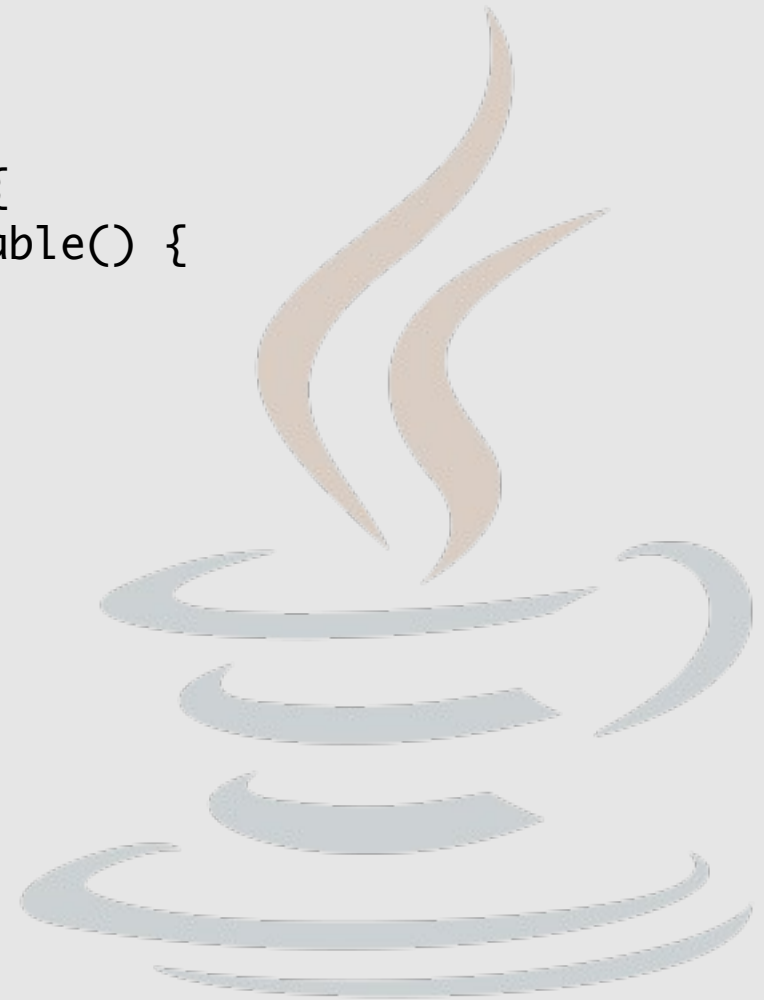
# KOMPONENTY

```
// Interfejs ActionListener implementuje jedna metode, ktora jest
// uruchamiana gdy nastapi zdarzenie na komponencie nasluchiwany
// przez ten obiekt. Informacje o zrodle akcji sa przekazywane przez
// argument(ActionEvent)
@Override
public void actionPerformed(ActionEvent e) {
    if (COMMAND_GO.equals(e.getActionCommand())) {
        try {
            // przeladowujemy strone
            this.setPage(new URL(this.url.getText()));
        } catch (IOException e2) {
            this.webpage.setText(
                "Problem z adresem " + this.url.getText());
            this.htmlPage.setText(
                "Problem z adresem " + this.url.getText());
        }
    }
}
```



# KOMPONENTY

```
// uruchomienie programu
public static void main(String[] args) {
    SwingUtilities.invokeLater(new Runnable() {
        public void run() {
            createAndShow();
        }
    });
}
```





DZIĘKUJĘ ZA UWAGĘ