Wykład 7
Programowanie wizualne- pakiet Swing

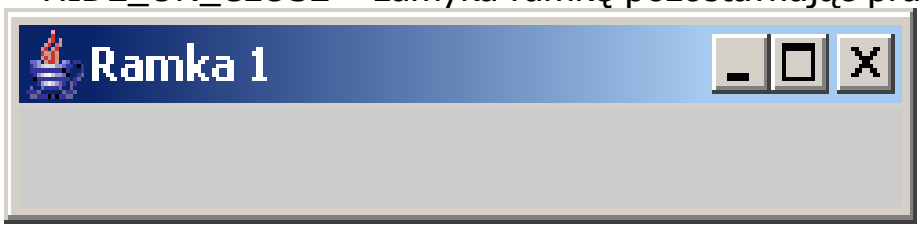
1. Główny obiekt interfejsu użytkownika - obiekt klasy JFrame

1.1. Przykład prostej aplikacji

- Zdefiniowanie klasy dziedziczącej po klasie JFrame z pakietu Swing (lub JWindow)
  public class Prosta_Aplikacja2 **extends** JFrame
  { //.....
  }
- Wywołania konstruktora klasy bazowej dla zapewnienia jej poprawnej inicjalizacji: **super**() (dwa konstruktory: **JFrame()** - okno nie posiada wtedy tytułu oraz **JFrame(String) -** okno posiada wtedy tytuł podany w parametrze konstruktora)
- Ustawienia rozmiaru okna ramki w pikselach za pomocą metody **setSize()**
- Obsługa zamykania okna
- Wyświetlenie ramki na ekranie metodą show() lub **setVisible(true)**

Ramka posiada przyciski *Minimalizuj, Maksymalizuj* oraz *Zamknij* zlokalizowane na pasku tytułu. W przypadku Javy domyślne zamknięcie ramki nie powoduje zamknięcia aplikacji. Aby to zmienić, należy wywołać metodę **setDefaultCloseOperation()** zawierającą jeden z czterech parametrów wywołania:

✓ EXIT_ON_CLOSE – zamyka aplikacje po zamknięciu ramki
✓ DISPOSE_ON_CLOSE – zamyka ramkę, usuwa obiekt reprezentujący ramkę, ale pozostawia pracującą aplikację,
✓ DO_NOTHING_ON_CLOSE – pozostawia ramkę otwartą i kontynuuje pracę aplikacji
✓ HIDE_ON_CLOSE – zamyka ramkę pozostawiając pracującą aplikację.




```
public class Prosta_Aplikacja1 extends JFrame
  { public Prosta_Aplikacja1()
     {  super("Ramka 1");// wywołanie dziedziczonego konstruktora i podanie tytułu okna
        setSize(250,50);      //rozmiar okna w pikselach
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);//obsługa zamykania aplikacji
        setVisible(true);  } //wyświetlenie okna

      public static void main(String[] arg)
       { Prosta_Aplikacja1 pr= new Prosta_Aplikacja1(); }
  }
```

1.2. Zamykanie okna aplikacji za pomocą bezpośredniej obsługi zdarzenia zamykania okna (zamiast wykorzystania metody **setDefaultCloseOperation**())

Czynności:

- utworzenie obiekt, który będzie monitorował stan okna
- Implementowanie za pomocą klasy **JFrame** interfejs **WindowsListener,** który będzie obsługiwał wszystkie możliwe zmiany stanu okna za pomocą implementowanych metod
  - ✓ windowClosed() – okno jest zamknięte
  - ✓ windowClosing() – okno jest zamykane
  - ✓ windowOpen() – okno jest widoczne
  - ✓ windowIconified() – okno zminimalizowane
  - ✓ windowDeiconified() – okno zmaksymalizowane
  - ✓ windowActivated() – okno powiązane z tym obiektem staje się aktywne i może przyjmować dane z klawiatury
  - ✓ windowDeactivated() - okno powiązane z tym obiektem staje się nieaktywne i nie może przyjmować danych z klawiatury

- lub wykorzystanie klasę **WindowAdapter** z pakietu awt, która ma zaimplementowane puste metody jak dla interfejsu **WindowListener**. Należy wykonać klasę dziedziczącą po **WindowAdapter** przedefiniowując wybraną metodę wg następującego przykładu

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

class Zamykanie_Aplikacji extends WindowAdapter
 {   public void windowClosing (WindowEvent e)
         { System.exit(0);}
 }

public class Prosta_Aplikacja2 extends JFrame
 {    public Prosta_Aplikacja2()
       { super ("Ramka 2");
         setSize(250,50);
         Zamykanie_Aplikacji wyjscie = new Zamykanie_Aplikacji();
         addWindowListener(wyjscie);
         //wstawienie obiektu monitorującego zamykanie okna wyjscie do aplikacji
     }
  public static void main (String[] arg)
     { Prosta_Aplikacja2 p2= new Prosta_Aplikacja2();
       p2.show(); }                          //wyświetlenie aplikacji
     }
```

2. Obsługa zdarzeń

Jest to reakcja na działania użytkownika programu. Obsługa zdarzeń realizowana jest za pomocą implementowania interfejsów, zwanych zarządcami zdarzeń.

Interfejsy- zarządcy obsługi zdarzeń

- **ActionListener –** obsługuje zdarzenia generowane przez użytkownika na rzecz danego składnika interfejsu (Np. kliknięcie przycisku)
- **AdjustmentListener –** obsługuje zdarzenie jako zmianę stanu składnika (np. przesuwanie suwaka w polu tekstowym)
- **FocusListener –** obsługuje zdarzenie od przejścia składnika w stan nieaktywny
- **ItemListener -** obsługuje zdarzenie od np. zaznaczenia pola wyboru
- **KeyListener -** obsługuje zdarzenie np. od wpisywania tekstu z klawiatury
- **MouseListener -** obsługuje zdarzenie od naciśnięcia klawiszy myszy
- **MouseMotionListener -** obsługuje zdarzenie od przesuwania wskaźnika myszy nad danym składnikiem
- **WindowListener -** obsługuje zdarzenie od okna np. minimalizacja, maksymalizacja, przesunięcie, zamknięcie

Wiązanie składnika z obsługą zdarzeń

Aby powiązać składnik interfesju z odpowiednim zarządcą zdarzeń, należy wykorzystać następujące metody:

- ***addActionListener()*** dla JButton, JCheckBox, JComboBox, JTextField, JRadioButton
- ***addAdjustmentListene()*** dla JScroolBar
- ***addFocusListener()*** dla wszystkich składników pakietu Swing
- ***addItemListener()*** dla JButton, JCheckBox, JComboBox, JTextField, JRadioButton
- ***addKeyListener()*** dla wszystkich składników pakietu Swing
- ***addMouseListener()*** dla wszystkich składników pakietu Swing
- ***addMouseMotionListener()*** wszystkich składników pakietu Swing
- ***addWindowListener()*** wszystkich obiektów typu JFrame oraz JWindow

Metody te muszą być zastosowane przed wstawieniem składnika do obiektu typu JPanel.

Metody obsługujące zdarzenia

Kiedy dana klasa implementuje interfejs, musi ona obsługiwać zdarzenia za pomocą metody, która jest wtedy wywoływana automatycznie, natomiast w programie trzeba ją odpowiednio przedefiniować, realizując sposób obsługi zdarzeń.

- Interfes ActionListener

  public void actionPerformed (ActionEvent evt)

  { //kod obsługi zdarzeń }
- Interfes AjustmentListener

  public void adjustmentValueChange (AdjustmentEvent evt)

  {//kod obsługi zdarzeń }
- Interfes FocusListener

  public void focusGained (FocusEvent evt) { //kod obsługi zdarzeń }

  public void focusLost (FocusEvent evt) { //kod obsługi zdarzeń }
- Interfes ItemListener

  public void itemStateChange (ItemEvent evt) { //kod obsługi zdarzeń }
- Interfes KeyListener

  public void keyPressed (KeyEvent evt) { //kod obsługi zdarzeń }

  public void keyReleased (KeyEvent evt) { //kod obsługi zdarzeń }

  public void keyTyped (KeyEvent evt) { //kod obsługi zdarzeń }
- Interfes MouseListener

  public void mouseClicked (MouseEvent evt) { //kod obsługi zdarzeń }

  public void mouseEntered (MouseEvent evt) { //kod obsługi zdarzeń }

  public void mouseExited (MouseEvent evt) { //kod obsługi zdarzeń }

  public void mousePressed (MouseEvent evt) { //kod obsługi zdarzeń }

  public void mouseReleased (MouseEvent evt) { //kod obsługi zdarzeń }
- Interfes MouseMotionListener

  public void mouseDragged (MouseEvent evt) { //kod obsługi zdarzeń }

  public void mouseMoved (MouseEvent evt) { //kod obsługi zdarzeń }
- Interfes WindowListener

  public void windowActivared (WindowEvent evt) { //kod obsługi zdarzeń }

  public void windowClosed (WindowEvent evt) { //kod obsługi zdarzeń }

  public void windowClosing (WindowEvent evt) { //kod obsługi zdarzeń }

  public void windowDeactvated (WindowEvent evt) { //kod obsługi zdarzeń }

  public void windowDeiconfied (WindowEvent evt) { //kod obsługi zdarzeń }

  public void windowIconfied (WindowEvent evt) { //kod obsługi zdarzeń }

  public void windowOpened (WindowEvent evt) { //kod obsługi zdarzeń }

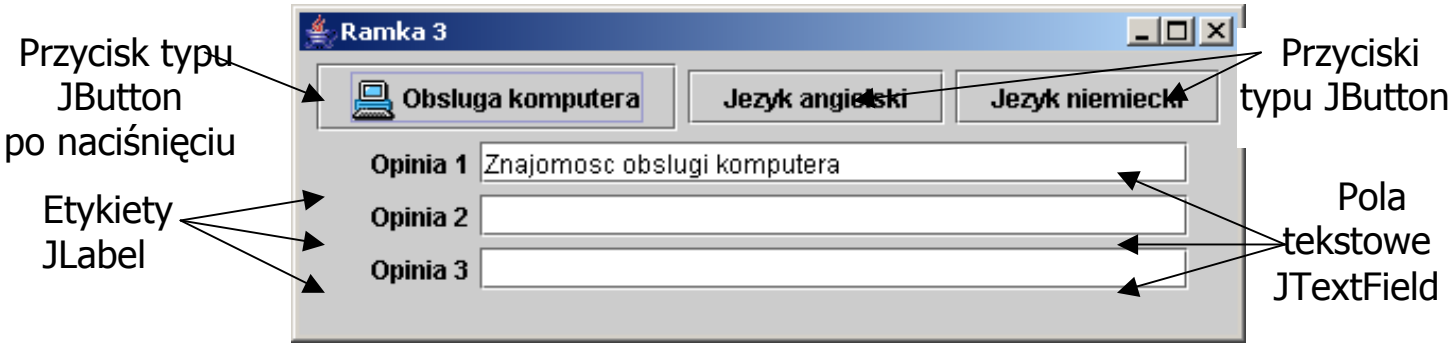
3. Tworzenie składników interfejsu użytkownika

- Dowolny składnik musi być dodany do kontenera (panel-obiekt klasy JPanel) za pomocą jego metody panel.add(skladnik), i po wywołaniu metody setContentPane(panel) wyświetlony na ekranie po wyświetleniu kontenera.
- Obsługa zdarzenia typu **Action** (JButton, JCheckBox, JComboBox, JTextField, JRadioButtion): naciśnięcie klawisza i wyświetlenie napisu w polu tekstowym.

Przyciski typu JButton, etykiety Jlabel oraz pola tekstowe JTextField, zdarzenia typu ActionEvent

Konstruktory:

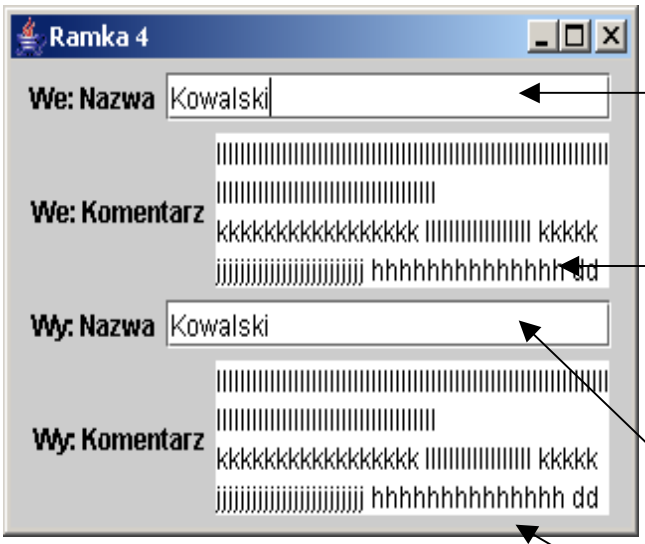
- ✓ JButton(String) - przycisk z etykietą tekstową
- ✓ JButton(Icon) - przycisk w postaci ikony graficznej
- ✓ JButton(String, Icon) – tworzy przycisk z etykietą tekstową i ikoną




```
import javax.swing.*;
import java.util.*;
import java.io.*;
import java.lang.*;
import java.awt.event.*;
 class Zamykanie_Aplikacji extends WindowAdapter
 {    public void windowClosing (WindowEvent e)
        { System.exit(0); }
 }
class Dane
{ String opinia1,opinia2,opinia3;
   Dane()
   { opinia1=opinia2=opinia3=null;}
}
public class Prosta_Aplikacja3 extends JFrame implements ActionListener
{ ImageIcon o1=new ImageIcon ("comp.blue.gif");
  JButton weopinia1 = new JButton ("Obsluga komputera", o1);      //obiekty-przyciski
  JButton weopinia2 = new JButton("Jezyk angielski");             //typu JRadioButton
  JButton weopinia3 = new JButton("Jezyk niemiecki");  // do wprowadzania danych
  JTextField wyopinia1=new JTextField(30);          //obiekty do wyświetlania danych
  JTextField wyopinia2=new JTextField(30);          // na ekranie typu JTextField
  JTextField wyopinia3=new JTextField(30);
  Dane dana = new Dane();                  //obiekt do zapamiętania danych wprowadzonych
```

```
 public Prosta_Aplikacja3()
  {   super("Ramka 3");
      setSize(450,160);
     JPanel panel=new JPanel();
      weopinia1.addActionListener(this); //this-obiekt monitorujący zdarzenie  typu Action
      weopinia2.addActionListener(this); //obiekty do wprowadzania danych
      weopinia3.addActionListener(this); // typu JRadioButton odbierają zdarzenia

      panel.add(weopinia1);              //wstawianie obiektów typu JRadioButton
      panel.add(weopinia2);              //do kontenera
      panel.add(weopinia3);
                                  //tworzenie etykiety dla pola tekstowego
      JLabel eopinia1= new JLabel("    Opinia 1",SwingConstants.RIGHT);
      panel.add(eopinia1);          //wstawianie etykiety i
      panel.add(wyopinia1);         //pola tekstowego do wyświetlania do kontenera
      JLabel eopinia2= new JLabel("    Opinia 2",SwingConstants.RIGHT);
      panel.add(eopinia2);
      panel.add(wyopinia2);
      JLabel eopinia3= new JLabel("    Opinia 3",SwingConstants.RIGHT);
      panel.add(eopinia3);
      panel.add(wyopinia3);
      Zamykanie_Aplikacji wyjscie = new Zamykanie_Aplikacji();
      addWindowListener(wyjscie);
      setContentPane(panel);
  }

  public void actionPerformed (ActionEvent evt) //metoda obsługująca zdarzenie Action
   { Object zrodlo = evt.getSource();
     if (zrodlo==weopinia1)              //jeśli naciśnięto przycisk "Obsługa komputera"
       dana.opinia1= new String("Znajomosc obslugi komputera");
    else if (zrodlo==weopinia2)                 //jeśli naciśnięto przycisk "Język angielski"
       dana.opinia2= new String("Znajomosc jezyka angielskiego");
    else if (zrodlo==weopinia3)                 //jeśli naciśnięto przycisk "Język niemiecki"
       dana.opinia3= new String("Znajomosc jezyka niemieckiego");
     wyopinia1.setText(dana.opinia1); //przekazanie tekstu związanego z naciśniętymi klawiszami
     wyopinia2.setText(dana.opinia2); // do składników tekstowych JTextField,
     wyopinia3.setText(dana.opinia3); // które zastosowano do wyświetlenia tekstu
     repaint();  }
  public static void main(String[] arg) throws Exception
   { Prosta_Aplikacja3 pr= new Prosta_Aplikacja3();
     pr.show();  }              //wyświetlenie okna
}
```

Obszary tekstowe JtextArea, zdarzenia typu FocusEvent



Obszar tekstowy JTextField do wprowadzania danych reagujący na stan aktywny innego składnika np. kliknięcie myszą innego składnika powoduje wyświetlenie tekstu w składniku wyjściowym Wy:Nazwa

Obszar tekstowy JTextArea do wprowadzania danych reagujący na stan aktywny innego składnika np. kliknięcie myszą innego składnika powoduje wyświetlenie tekstu w składniku wyjściowym Wy:Komentarz

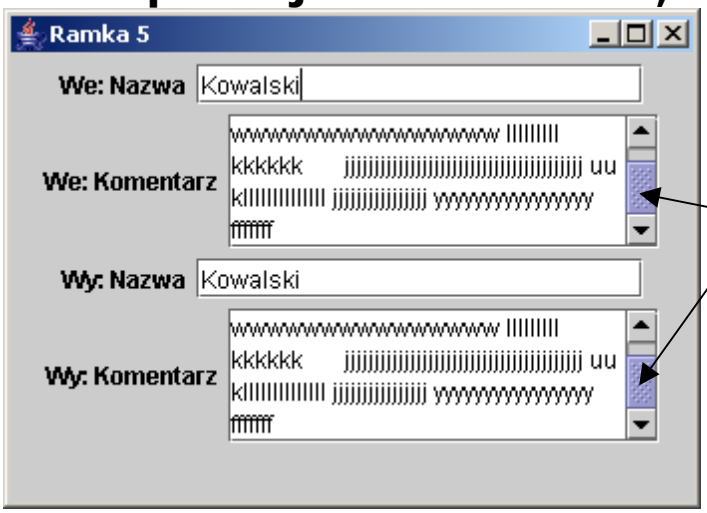
Obszar tekstowy JTextField do wyświetlania danych wprowadzonych w polu We:Komentarz

Obszar tekstowy JTextArea do wyświetlania danych wprowadzonych w polu We:Komentarz

```
import javax.swing.*;
import java.util.*;
import java.io.*;
import java.lang.*;
import java.awt.event.*;

class Zamykanie_Aplikacji extends WindowAdapter
 {  public void windowClosing (WindowEvent e)
      { System.exit(0); }
 }

class Dane
{ String nazwa, komentarz;
   Dane()
   { nazwa=komentarz=null; }
}

public class Prosta_Aplikacja4 extends JFrame implements FocusListener
{
    Dane dana = new Dane();                          //dane aplikacji
    JTextField nazwa = new JTextField(20);           //składniki do wprowadzania danych
    JTextArea komentarz = new JTextArea (4,18);      //JText oraz JTextArea
    JTextField wynazwa=new JTextField(20);           //składniki do wprowadzania danych
    JTextArea wykomentarz=new JTextArea(4,18);       //JText oraz JTextArea
```

```
 public Prosta_Aplikacja4()
  {   super("Ramka 12");
       setSize(320,220);
      JPanel panel=new JPanel();
      nazwa.addFocusListener(this);
       komentarz.addFocusListener(this);
       komentarz.setLineWrap(true); //zawijanie linii w oknie typu JTextArea
                                    //przeniesienie do następnej linii ostatniego wyrazu w linii, który się nie mieści
       komentarz.setWrapStyleWord(true);
       wykomentarz.setLineWrap(true);
       wykomentarz.setWrapStyleWord(true);
       JLabel etykieta_nazwy1          = new JLabel ("We: Nazwa");
       JLabel etykieta_komentarza1     = new JLabel  ("We: Komentarz");
       panel.add(etykieta_nazwy1);
       panel.add(nazwa);
       panel.add(etykieta_komentarza1);
       panel.add(komentarz);
       JLabel etykieta_nazwy2          = new JLabel ("Wy: Nazwa");
       JLabel etykieta_komentarza2     = new JLabel  ("Wy: Komentarz");
       panel.add(etykieta_nazwy2);
       panel.add(wynazwa);
       panel.add(etykieta_komentarza2);
       panel.add(wykomentarz);
       Zamykanie_Aplikacji wyjscie = new Zamykanie_Aplikacji();
       addWindowListener(wyjscie);
       setContentPane(panel); }

public void focusLost(FocusEvent evt)
{ Object zrodlo = evt.getSource();  // reakcja składników wejściowych na przejście ze stanu
  if (zrodlo==nazwa)                // aktywnego w nieaktywny np. kliknięcie innego składnika
     dana.nazwa= new String(nazwa.getText());          //zapamiętanie danych aplikacji
  else if(zrodlo==komentarz)
     dana.komentarz=new String(komentarz.getText());   //wyświetlenie danych aplikacji
   wynazwa.setText(dana.nazwa);
   wykomentarz.setText(dana.komentarz);
   repaint();  }                                       //odświeżenie widoku okna

public void focusGained (FocusEvent evt)
   { //metoda implementująca; do obsługi zdarzeń zastosowano jedynie focusLost }

 public static void main(String[] arg) throws Exception
   { Prosta_Aplikacja4 pr= new Prosta_Aplikacja4();
     pr.show();  }              //wyświetlenie okna
}
```

Panele przewijane JScroolPane, zdarzenia typu FocusEvent



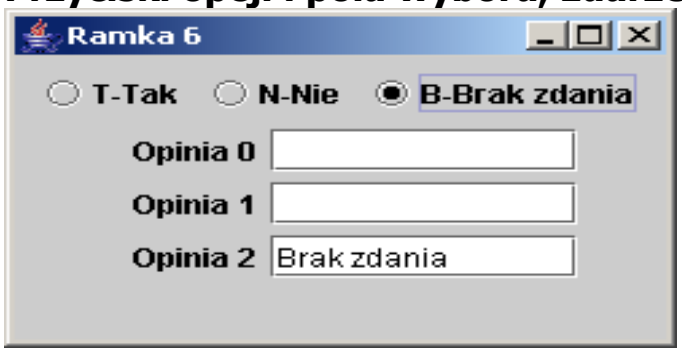
Suwaki skonfigurowane jako VERTICAL_SCROLLBAR_ALWAYS oraz HORIZONTAL_SCROLLBAR_AS_NEEDED

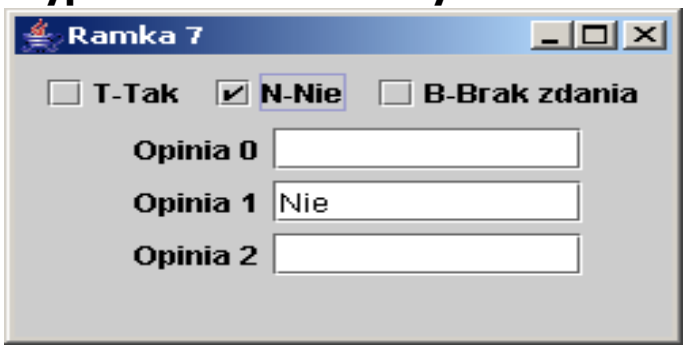
```
import javax.swing.*;
import java.util.*;
import java.io.*;
import java.lang.*;
import java.awt.event.*;

class Zamykanie_Aplikacji extends WindowAdapter
 { public void windowClosing (WindowEvent e)
    { System.exit(0); }
 }

class Dane
{ String nazwa, komentarz;
   Dane()   {nazwa=komentarz=null; }
}

public class Prosta_Aplikacja5 extends JFrame implements FocusListener
{ JTextField wynazwa=new JTextField(20);
  JTextArea wykomentarz=new JTextArea(4,18);
  Dane dana = new Dane();
  JTextField nazwa = new JTextField(20);
  JTextArea komentarz = new JTextArea (4,18);

  public Prosta_Aplikacja5()
   {     super("Ramka 5");
         setSize(350,250);
        JPanel panel=new JPanel();
         nazwa.addFocusListener(this);
         komentarz.addFocusListener(this);
         komentarz.setLineWrap(true);
         komentarz.setWrapStyleWord(true);
         wykomentarz.setLineWrap(true);
         wykomentarz.setWrapStyleWord(true);
        JLabel etykieta_nazwy1 = new JLabel ("We: Nazwa");
        JLabel etykieta_komentarza1 = new JLabel  ("We: Komentarz");
         panel.add(etykieta_nazwy1);              panel.add(nazwa);
         panel.add(etykieta_komentarza1);         panel.add(komentarz);
```

```
        JScrollPane obszar_przewijany1 = new JScrollPane
              (komentarz,
              ScrollPaneConstants.VERTICAL_SCROLLBAR_ALWAYS,
              ScrollPaneConstants.HORIZONTAL_SCROLLBAR_AS_NEEDED);
        panel.add(obszar_przewijany1);
        JLabel etykieta_nazwy2 = new JLabel ("Wy: Nazwa");
        JLabel etykieta_komentarza2 = new JLabel  ("Wy: Komentarz");
        panel.add(etykieta_nazwy2);              panel.add(wynazwa);
        panel.add(etykieta_komentarza2);         panel.add(wykomentarz);
        JScrollPane obszar_przewijany2 = new JScrollPane
              (wykomentarz,
               ScrollPaneConstants.VERTICAL_SCROLLBAR_ALWAYS,
              ScrollPaneConstants.HORIZONTAL_SCROLLBAR_AS_NEEDED);
        panel.add(obszar_przewijany2);
        Zamykanie_Aplikacji wyjscie = new Zamykanie_Aplikacji();
        addWindowListener(wyjscie);
        setContentPane(panel);
    }

    public void focusLost(FocusEvent evt)
    {  Object zrodlo = evt.getSource();
      if (zrodlo==nazwa)
            dana.nazwa= new String(nazwa.getText());
      if (zrodlo==komentarz)
          dana.komentarz= new String(komentarz.getText());
      wynazwa.setText(dana.nazwa);
      wykomentarz.setText(dana.komentarz);
      repaint();
    }

    public void focusGained (FocusEvent evt)
     {  }

     public static void main(String[] arg) throws Exception
     { Prosta_Aplikacja5 pr= new Prosta_Aplikacja5();
       pr.show(); }
}
```

Suwaki konfiguruje się za pomocą konstruktora:
JScrollPane(Component) – tworzy panel przewijany zawierający dany składnik
JScrollPane(Component, **int**, **int**)-tworzy panel przewijany zawierający dany składnik oraz konfiguruje suwak poziomy i pionowy. Drugi i trzeci parametr to:
VERTICAL_SCROLLBAR_ALWAYS, VERTICAL_SCROLLBAR_AS_NEEDED, VERTICAL_SCROLLBAR_NEVER, HORIZONTAL_SCROLLBAR_ALWAYS, HORIZONTAL_SCROLLBAR_AS_NEEDED, HORIZONTAL_SCROLLBAR_ NEVER

Przyciski opcji i pola wyboru, zdarzenia typu ActionEvent i KeyEvent

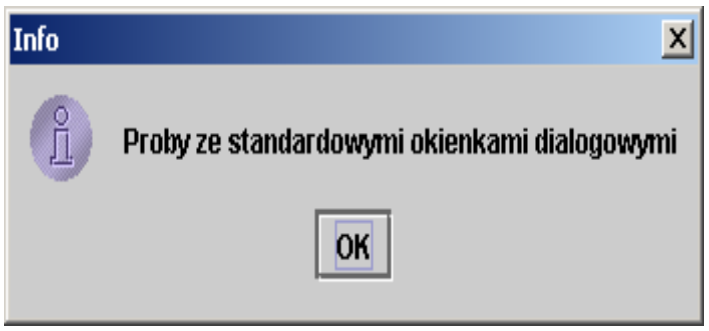


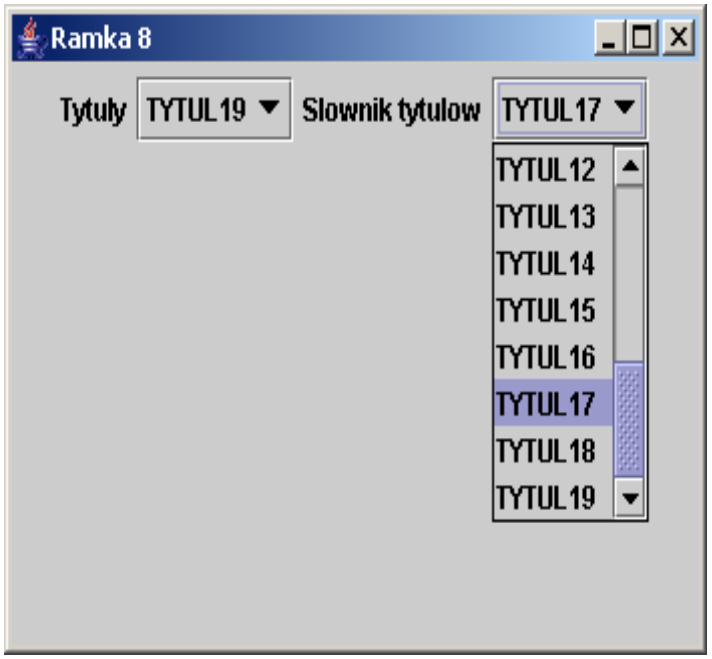


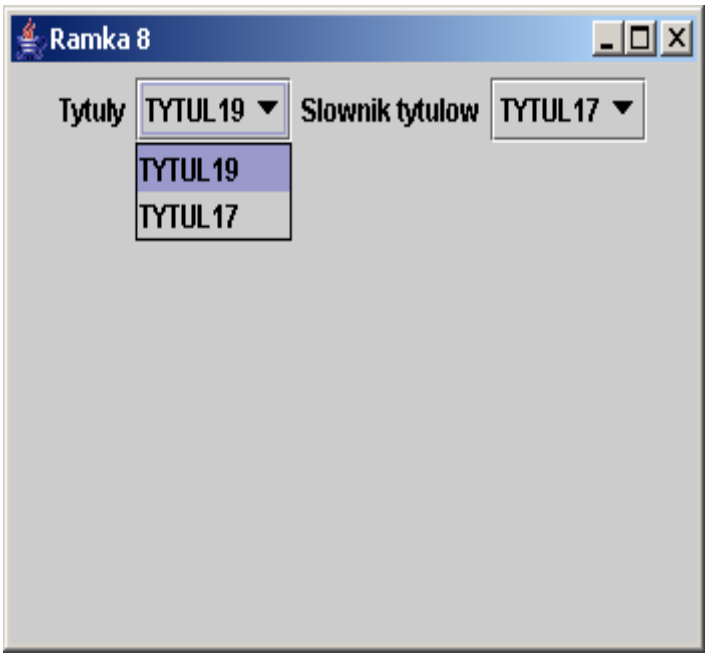

```
import javax.swing.*;
import java.util.*;
import java.io.*;
import java.lang.*;
import java.awt.event.*;

class Dane
{ String opinia1, opinia2, opinia3;
   Dane()   {opinia1=opinia2=opinia3=null;}
}
public class Prosta_Aplikacja6 extends JFrame implements KeyListener, ActionListener
{ JRadioButton[] wybor = new JRadioButton[3]; //lub JCheckBox- obiekty do  wprowadzanie danych
   Dane dana = new Dane();                    //dane aplikacji
   JTextField wyopinie[] = new JTextField[3];       //składniki ITextField do wyświetlania danych

  public Prosta_Aplikacja6()
   {   super("Ramka 6");
       setSize(250,160);
       setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
       JPanel panel=new JPanel();
       wybor[0]=new JRadioButton("T-Tak",true);          //lub JCheckBox
       wybor[1]=new JRadioButton("N-Nie");               //lub JCheckBox
       wybor[2]=new JRadioButton("B-Brak zdania");       //lub JCheckBox
       ButtonGroup wybory = new ButtonGroup();
       for (int i=0; i<wybor.length; i++)
         { wybor[i].addKeyListener(this);       //obiekt odbierający zdarzenia od klawiatury
           wybor[i].addActionListener(this);  //oraz od działań użytkownika (np. klikanie myszy)
           wybory.add(wybor[i]);              //wstawianie przycisków do kontenera
           panel.add(wybor[i]); }
       JLabel eopinie[]= new JLabel [3];
       for (int i=0; i<eopinie.length;i++)
         {  eopinie[i]= new JLabel("    Opinia "+i,SwingConstants.RIGHT);
            wyopinie[i]= new JTextField(10); // tworzenie pól tekstowych do wyświetlania
            panel.add(eopinie[i]);                //wstawianie etykiet do kontenera
            panel.add(wyopinie[i]); }             //wstawianie pól tekstowych do kontenera
        setContentPane(panel);}
```

```
 public void keyPressed (KeyEvent evt)
  { char znak=evt.getKeyChar();    //reakcja na przycisku na naciśnięcie klawisza
    if (znak =='t')
      { wybor[0].setSelected(true); //uaktywnienie przycisku po naciśnięciu odpowiadającego klawisza
        dana.opinia1= "Tak";          //zapamiętanie aktualnych danych - alternatywnych
        dana.opinia2=dana.opinia3= ""; }
     else if (znak=='n')
      { wybor[1].setSelected(true);
        dana.opinia2="Nie";
        dana.opinia1= dana.opinia3= ""; }
     else if (znak=='b')
      { wybor[2].setSelected(true);
        dana.opinia3= "Brak zdania";
        dana.opinia1= dana.opinia2= "";}
     wyopinie[0].setText(dana.opinia1);        //ustawienie pól tekstowych do wyświetlania
     wyopinie[1].setText(dana.opinia2);        //aktualnych danych
     wyopinie[2].setText(dana.opinia3);
     repaint();                                //odświeżenie widoku okna
   }

  public void keyReleased (KeyEvent evt)
      {   }
  public void keyTyped (KeyEvent evt)
      {   }

  public void actionPerformed (ActionEvent evt)
 {  Object zrodlo=evt.getSource();       //reakcja na przycisku na działanie użytkownika
    if ( zrodlo ==wybor[0])                    // (kliknięcie na wybrany przycisk)
      { dana.opinia1= "Tak";
        dana.opinia2= dana.opinia3= ""; }
     else if ( zrodlo ==wybor[1])
      { dana.opinia2= "Nie";
        dana.opinia1= dana.opinia3= "";}
     else if ( zrodlo ==wybor[2])
      { dana.opinia3= "Brak zdania";
        dana.opinia1= dana.opinia2= "";}
     wyopinie[0].setText(dana.opinia1);        //ustawienie pól tekstowych do wyświetlania
     wyopinie[1].setText(dana.opinia2);        //aktualnych danych
     wyopinie[2].setText(dana.opinia3);
     repaint();    }                          //odświeżenie widoku okna

  public static void main(String[] arg)
   { Prosta_Aplikacja6 pr= new Prosta_Aplikacja6();
     pr.show();  } //wyświetlenie okna
}
```

Listy rozwijane, zdarzenia typu ItemEvent, okno typu MessageDialog








```
import javax.swing.*;
import java.util.*;
import java.io.*;
import java.lang.*;
import java.awt.event.*;

class Zamykanie_Aplikacji extends WindowAdapter
 {  public void windowClosing (WindowEvent e)
     { System.exit(0);  }
 }

public class Prosta_Aplikacja8 extends JFrame     implements ItemListener
{   JComboBox tytuly = new JComboBox(); //lista rozwijana do wyświetlania wprowadzonych
                                                                    //danych
                                                dobranych ze słownika
    Vector dana=new Vector();                   //przechowywanie danych w aplikacji
    Vector dane_slownika= new Vector();  //wewnętrzna struktura danych dla drugiej listy
    JComboBox slownik = new JComboBox(dane_slownika); // rozwijanej, która pełni rolę
                                                                    //słownika danych
```

```
public Prosta_Aplikacja8()
  {    super("Ramka 8");
       setSize(350,250);
       JPanel panel=new JPanel();
       JLabel etykieta_tytulow = new JLabel("Tytuly");
       panel.add(etykieta_tytulow);
       panel.add(tytuly);
       for (int i=0; i<20;i++)
           dane_slownika.add("TYTUL"+i);
       JLabel etykieta_slownika = new JLabel("Slownik tytulow");
       panel.add(etykieta_slownika);
       slownik.addItemListener(this);
       panel.add(slownik);
       Zamykanie_Aplikacji wyjscie = new Zamykanie_Aplikacji();
       addWindowListener(wyjscie);
       setContentPane(panel);
  }

  public void itemStateChanged(ItemEvent evt)
  {  int zrodlo = evt.getStateChange();        //reakcja na wybranie pozycji w liście rozwijanej slownik
     if (zrodlo==ItemEvent.SELECTED)
      { dana.add(evt.getItem()); //zapisanie danych aplikacji do Vectora
        tytuly.addItem(evt.getItem()); //wstawienie danych do listy rozwijanej wyświetlające
       }             //dane aplikacji pobrane z listy nieedytowalnej, pełniącej rolę słownika danych
     repaint();     //odświeżenie widoku okna
  }

  void poczatek()              //standardowe okienko dialogowe typu MessageDialog
   {JOptionPane.showMessageDialog(
       null,                                          //składnik nadrzędny okna dialogowego
       "Proby ze standardowymi okienkami dialogowymi", //wyświetlany komunikat
       "Info",                                        //tytuł okna
       JOptionPane.INFORMATION_MESSAGE);       //typ ikony wyświetlanej w okienku (i)
    }

  public static void main(String[] arg)
   { Prosta_Aplikacja8 pr= new Prosta_Aplikacja8();
     pr.poczatek();
     pr.show();          //wyświetlenie okna
    }
  }
```