Kurs programowania

Wykład 5

Wojciech Macyna

31 marzec 2016

Klasa `java.awt.Panel`

Klasa `Panel` jest równocześnie komponentem (czyli może być wstawiana tam gdzie i inne komponenty) i kontenerem (można do niej wstawiać inne komponenty).
Panel służy również często do rysowania – wtedy warto aby być podwójnie buforowany (wtedy działa dużo szybciej).

Główne konstruktory

- `public Panel()`
- `public Panel(LayoutManager m)`
- `public JPanel(boolean isDoubleBuffered)`

Główne metody

Pojawią się w trakcie następnego wykładu.

Klasa `java.awt.Dialog`

Główne konstruktory

- `public Dialog(Frame owner, String title, boolean modal)`

`modal` wskazuje czy okienko dialogowe ma blokować okno główne.

Główne metody

- `public void setVisible(boolean v)`

Przykład – dialogDemo.java

```
1  import java.awt.*;
2  import java.awt.event.*;
3  class dialogDemoWindowAdapter extends WindowAdapter {
4    public void windowClosing(WindowEvent e) { System.exit(0); } }
5  public class dialogDemo extends Frame implements ActionListener {
6    private Dialog myDialog;
7    private Button myFrameButton, myDialogButton;
8    public dialogDemo() {
9      myFrameButton = new Button("Dialog");
10     myFrameButton.addActionListener(this);
11     myDialogButton = new Button("OK");
12     myDialogButton.addActionListener(this);
13     myDialog = new Dialog(this, "Dialog Window", true);
14     myDialog.setSize(320,240);
15     myDialog.add(myDialogButton);
16     add(myFrameButton);
17     addWindowListener(new dialogDemoWindowAdapter()); }
18   public void actionPerformed(ActionEvent e) {
19     if( e.getActionCommand().equals("OK") ) myDialog.setVisible(false);
20     else myDialog.setVisible(true); }
21   public static void main(String[] args) {
22     Frame f = new dialogDemo();
23     f.setBounds(100,100,640,480);
24     f.setVisible(true); }
25 }
```

Klasy `java.awt.MenuBar`, `java.awt.Menu` i `java.awt.MenuItem`

`java.awt.MenuBar`

- `public MenuBar()` - konstruktor
- `public Menu add(Menu m)` - dodanie menu
- Dodanie do `Frame` przez metodę `setMenuBar(MenuBar mb)`

`java.awt.Menu extends MenuItem`

- `public Menu(String label)` - konstruktor
- `public Menu add(MenuItem m)` - dodanie wpisu/menu
- `public void addSeparator()` - dodanie linii separatora

`java.awt.MenuItem`

- `public MenuItem(String label)` - konstruktor
- `public void addActionListener(ActionListener l)` - dodanie słuchacza
- `public String getActionCommand()` - podanie etykiety

Przykład – menuDemo.java (1/2)

```
1  import java.awt.*;
2  import java.awt.event.*;
3  class menuDemoWindowAdapter extends WindowAdapter {
4    public void windowClosing(WindowEvent e) { System.exit(0); }
5  }
6  public class menuDemo extends Frame implements ActionListener {
7    private Label myLabel;
8    private MenuBar myMenu;
9    private Menu menu1, menu2, submenu;
10   private MenuItem i1, i2, i3, i4, i5, exit;
11
12   public void actionPerformed(ActionEvent e) {
13     if( e.getActionCommand().equals("Exit") ) System.exit(0);
14     else myLabel.setText("Wybrano␣"+e.getActionCommand());
15   }
16   public static void main(String[] args) {
17     Frame f = new menuDemo();
18     f.setBounds(100,100,640,480);
19     f.setVisible(true);
20   }
```

Przykład – menuDemo.java (2/2)

```
21    public menuDemo() {
22      myLabel = new Label("Start",Label.CENTER);
23      myMenu = new MenuBar();
24      menu1 = new Menu("Menu 1");
25      menu2 = new Menu("Menu 2");
26      myMenu.add( menu1 );
27      myMenu.add( menu2 );
28      submenu = new Menu("Podmenu");
29      menu1.add( submenu );
30      menu1.addSeparator();
31      i1 = new MenuItem("Akcja 1"); i1.addActionListener(this);
32      i2 = new MenuItem("Akcja 2"); i2.addActionListener(this);
33      i3 = new MenuItem("Akcja 3"); i3.addActionListener(this);
34      i4 = new MenuItem("Akcja 4"); i4.addActionListener(this);
35      i5 = new MenuItem("Akcja 5"); i5.addActionListener(this);
36      exit = new MenuItem("Exit");  exit.addActionListener(this);
37      submenu.add( i1 );
38      submenu.add( i2 );
39      submenu.add( i3 );
40      menu1.add( exit );
41      menu2.add( i4 );
42      menu2.add( i5 );
43      setLayout(new GridLayout(1,1));
44      setMenuBar( myMenu );
45      add(myLabel);
46      addWindowListener(new menuDemoWindowAdapter());
47    }
48  }
```

Klasa `java.awt.Checkbox`

Główne konstruktory

- `public Checkbox()`
- `public Checkbox(String label)`
- `public Checkbox(String label, boolean selected)`

Główne metody

- `public void setState(boolean state)`
- `public boolean getState()`

Klasa `java.awt.TextArea`

Główne konstruktory

- `public TextArea()`
- `public TextArea(int rows, int columns)`
- `public TextArea(String label)`
- `public TextArea(String label, int rows, int columns)`

Główne metody

- `public void append(String str)`
- `public String getText()`
- `public void setText(String str)`
- `public int getColumns()` | `public int getRows()`

Inne komponenty

Opis pozostałych komponentów graficznych można znaleźć w dokumentacji Javy.

Procesy zewnętrzne

Uruchamianie procesu zewnętrznego

Metoda `public Process exec(String command) throws IOException` umożliwia wykonywanie procesu zewnętrznego.

Obiekt `Process` - główne metody

- `void destroy()`
- `InputStream getErrorStream()`
- `InputStream getInputStream()`
- `OutputStream getOutputStream()`

Szczegółowe informacje o strumieniach na następnych wykładach.

Przykład

ExecDemo.java

```
1  import java.io.*;
2
3  public class ExecDemo {
4    public static void main(String[] args) {
5      try {
6        Process child =
7  Runtime.getRuntime().exec("cmd␣/c␣dir␣*.java");
8        InputStream in = child.getInputStream();
9        int c;
10       while ((c = in.read()) != -1)
11         System.out.print((char)c);
12       in.close();
13     }
14     catch(IOException e) { }
15   }
16 }
```

Przykład z GUI– Program.java (1/2)

```
1  import java.awt.*;
2  import java.awt.event.*;
3  import java.io.*;
4  class MyWindowAdapter extends WindowAdapter {
5    public void windowClosing(WindowEvent e) { System.exit(0); }
6  }
7  class MyButtonAdapter implements ActionListener {
8    Program p;
9    MyButtonAdapter(Program p) { this.p = p; }
10   public void actionPerformed(ActionEvent e) { p.action(); }
11 }
12 class MyButton extends Button {
13   MyButton(Program p) {
14     super("Wykonaj"); addActionListener(new MyButtonAdapter(p)); }
15 }
16 class MyFrame extends Frame {
17   MyFrame(Program p) {
18     super("Program");
19     setBounds(100,100,640,480);
20     addWindowListener(new MyWindowAdapter());
21     setFont(new Font(Font.SANS_SERIF,Font.PLAIN,20));
22     setLayout(new BorderLayout());
23     MyButton akcja = new MyButton(p); add(p.dane,BorderLayout.NORTH);
24     p.wynik = new TextArea(25,80); add(akcja,BorderLayout.SOUTH);
25     p.dane = new TextField(80); add(p.wynik,BorderLayout.CENTER);
26     setResizable(false);
27   }
28 }
```

Przykład z GUI– Program.java (1/2)

```
29  public class Program {
30    MyFrame frame;
31    TextArea wynik;
32    TextField dane;
33    void action() {
34      try {
35        Process child = Runtime.getRuntime().exec( dane.getText() );
36        BufferedReader in = new BufferedReader(
37                              new InputStreamReader(child.getInputStream()));
38        String c;
39        wynik.setText("");
40        while ((c = in.readLine()) != null) wynik.append(c+"\n");
41        in.close();
42      }
43      catch(IOException e) {
44        wynik.setText(e.getMessage());
45      }
46      catch(IllegalArgumentException e) {
47        wynik.setText(e.getMessage());
48      }
49      dane.setText("");
50    }
51    public static void main(String[] args) {
52      Program p = new Program();
53      p.frame = new MyFrame(p);
54      p.frame.setVisible(true);
55    }
56  }
```