# 9

# Događaji

- Događaji

# Događaji

U primjeru iz prethodnog poglavlja smo kreirali jedno korisničko sučelje, u čije okvire za tekst možemo unositi tekst, brojeve ili bilo što drugo. Nekako smo navikli da se klikom na gumb dogodi neka očekivana akcija. Ovdje bi nekako bilo za očekivati da se klikom na gumb *Zbroji* zbroje brojevi koji pišu u prvom i drugom okviru za tekst te da se rješenje upiše u posljednji okvir za tekst. Međutim klikom na gumb *Zbroji* ne događa se ništa. Zašto? Pa vrlo jednostavno nismo rekli što se treba dogoditi kada kliknemo na gumb.
Općenito, klikom na gumb kreira se jedan događaj. Taj događaj šalje poruku drugom objektu, koji ćemo nazvati *listener*. *Listener* raspolaže sa svim podacima o događaju (npr. zna koji je gumb pritisnut ako ih je više,...). Kada primi poruku o događaju *listener* izvršava neku radnju, tj. pokreće odgovarajuću metodu koju smo mu definirali.
Java ima nekoliko različitih klasa za praćenje različitih tipova događaja. Mi ćemo se u ovom trenutku fokusirati na događaje koji su povezani sa sučeljem. Takve događaje zvat ćemo *akcije*. Rad s događajima vezanim uz sučelje omogućava nam klasa **`ActionListener`**, koja sadrži samo zaglavlje metode **`actionPerformed`**. Tijelo metode **`actionPerformed`** ćemo definirati u vlastitoj klasi. Unutar tijela metode **`actionPerformed`** ćemo staviti kôd koji će se izvršiti kao reakcija na neki događaj.
Općenito ćemo klase koje sadrže samo zaglavlja metoda zvati **interface**. Dakle, **`ActionListener`** je zapravo **interface**, čija bi definicija mogla imati sljedeći oblik:

```
public interface ActionListener
{
	public void actionPeroformed (ActionEvent e)
}
```

S obzirom da metoda **`actionPerformed`** nema tijela, dakle nije definirana u *interfaceu*, nije moguće kreirati instancu interfacea **`ActionListener`**, no kako ga trebamo u svojoj klasi, ipak ga trebamo nekako u nju uključiti.

Općenito ćemo korištenje interfacea u klasi najaviti na sljedeći način:

```
public class ime_paketa implements ime_interfacea;
```

Za našu klasu Zbrajanje bi u tom slučaju zaglavlje imalo sljedeći oblik:

```
public class Zbrajanje extends JFrame implements ActionListener
```

Budući da se interface nalazi u posebnom paketu **`java.awt.event`**, morat ćemo i taj paket uključiti u klasu u kojoj ćemo obrađivati događaje.

Nadalje trebamo "registrirati" elemente na kojima želimo pratiti događaje. To ćemo napraviti izvršavanjem metode:

```
addActionListener (Object o)
```

nad određenim objektom. Kao što možemo primijetiti kao parametar metode **`addActionListener`** dolazi neki objekt. Radi se o tome da se može kreirati posebna klasa koja će obrađivati događaje i u tom slučaju se kao parametar navodi instanca te klase. Mi ćemo događaje uvijek obrađivati u klasi u kojoj će se događaj i dogoditi, stoga ćemo kao parametar uvijek pisati **`this`**. Dakle, naša metoda **`addActionListener`** uvijek će imati oblik:

```
ime_objekta.addActionListener (this)
```

Mi ćemo za sada samo pratiti događaje na gumbima, iako je moguće pratiti događaje i na ostalim elementima sučelja, no o tome ćemo govoriti nešto kasnije.

Na kraju trebamo definirati metodu **actionPerformed**, unutar koje ćemo definirati akcije za sve događaje koje pratimo.
Zaglavlje metode **actionPerformed** je:

```
actionPerformed (ActionEvent e)
```

Parametar e metode **actionPerformed** ima sve informacije o događajima, pa između ostaloga zna i nad kojim elementom je događaj izvršen. Metoda koja će nam vratiti ime objekta nad kojim je izvršen događaj je **getSource** () i pozvat ćemo ju nad objektom tipa **ActionEvent**.

Ponovimo još jednom. Za obrađivanje događaja trebamo napraviti sljedeće korake:

1. uključiti paket **java.awt.event**
2. implementirati *interface* **ActionListener** u svojoj klasi. To ćemo napraviti tako da u zaglavlju klase dodamo još **implements ActionListener**.
   Npr. **public class Zbrajanje extends JFrame implements ActionListener**
3. registrirati listener na elementu grafičkog korisničkog sučelja metodom **addActionListener (Objekt o).**
   Npr. **b.addActionListener (this).** this u ovom slučaju znači da je *listener* implementiran u klasi u kojoj se upravo nalazimo, i svi naši primjeri će izgledati tako.
4. definirati metodu **actionPerformed**, koja će sadržavati kôd koji će se izvršiti kao reakcija na neki događaj.

**Primjer 9 – 1:**
Izmijenimo *Primjer 8 – 2*, tako da se klikom na gumb *Zbroji* u treći okvir za tekst upiše zbroj brojeva koji su upisani u prva dva okvira za tekst.

**Rješenje**:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class Zbrajanje extends JFrame implements ActionListener
{
    private final int sirina = 220;
    private final int visina = 200;
    private Container c;
    private JTextField t1, t2, t3;
    private JButton b;

    public Zbrajanje()
    {
        setTitle ("Zbrajanje");
        setSize (sirina, visina);
        setDefaultCloseOperation (EXIT_ON_CLOSE);

        c = getContentPane ();
        c.setLayout (null);

        t1 = new JTextField ();
        t1.setSize (200, 25);
```

```
        t1.setLocation (5, 5);
        c.add (t1);

        t2 = new JTextField ();
        t2.setSize (200, 25);
        t2.setLocation (5, 45);
        c.add (t2);

        b = new JButton ();
        b.setText ("Zbroji");
        b.setSize (200, 25);
        b.setLocation (5, 85);
        //najavljujemo praćenje događaja nad gumbom b
        b.addActionListener (this);
        c.add (b);

        t3 = new JTextField ();
        t3.setSize (200, 25);
        t3.setLocation (5, 125);
        c.add (t3);

        setVisible (true);
    }

    //metoda koja uzima sadržaj prvog i drugog okvira za tekst
    //te ih zbraja i zbroj postavlja u treći okvir za tekst
    public void zbroji ()
    {
        int a = Integer.parseInt (t1.getText ());
        int b = Integer.parseInt (t2.getText ());
        int c = a + b;
        t3.setText ("" + c);
    }

    //definicija metode actionPerformed na bilo koji događaj
    //koji se prati će se izvršiti metoda zbroji ()
    public void actionPerformed (ActionEvent e)
    {
        zbroji ();
    }

    public static void main (String[] s)
    {
        Zbrajanje z = new Zbrajanje ();
    }
}
```

Primijetimo da smo u prošlom primjeru pratili samo jedan događaj i to onaj nad gumbom b, stoga smo rekli da se na bilo koji događaj izvrši metoda *zbroji ()*. Međutim često ćemo unutar programa morati pratiti više događaja i u tom slučaju uglavnom nećemo uvijek izvršavati istu metodu, stoga ćemo morati provjeriti nad kojim elementom je događaj izvršen i ovisno o elementu ćemo izvršiti odgovarajuću metodu.

Naziv objekta nad kojim je izvršena akcija dobit ćemo metodom **`getSource ()`** koju pozivamo nad objektom tipa **`ActionEvent`**.
Dakle, naša metoda **`actionPerformed`** iz prethodnog primjera mogla bi imati i sljedeći oblik:

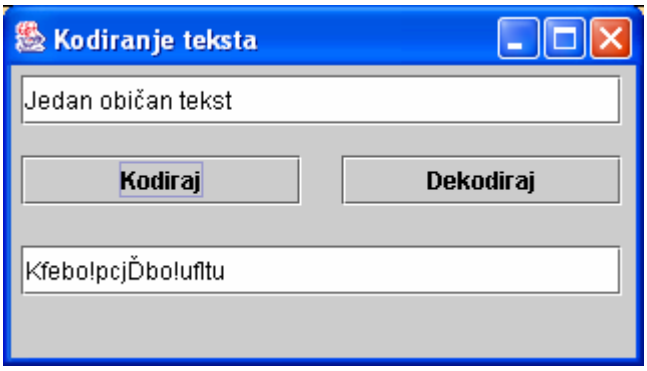
```
public void actionPerformed (ActionEvent e)
    {
```

```
        if (e.getSource () == b)
            zbroji ();
    }
```

# Zadaci za vježbu

1. Što sve moramo napraviti kako bismo mogli pratiti događaje na nekom elementu prozora?
2. Što je interface?
3. Napiši program koji će na ekranu crtati prozor kao na slici. Klikom na gumb ***Mijenjaj*** se iznos duljine u centimetrima treba pretvoriti u iznos duljine u inchima (1 inch = 2.54 cm).


4. Napiši program koji će crtati prozor kao na slici. Klikom na gumb Kodiraj se tekst iz gornjeg okvira za tekst treba kodirati i tako kodiran prepisati u donji okvir za tekst. Tekst ćemo kodirati tako da svaki znak zamijenimo znakom čiji je ASCII kôd za jedan veći od ASCII kôda odgovarajućeg znaka. Klikom na gumb Dekodiraj se tekst iz donjeg okvira za tekst treba dekodirati i tako dekodiran prepisati u gornji okvir za tekst.




5. Napiši program koji će crtati prozor kao na slici. Klikom na gumb OK gumb se treba pomaknuti u slučajnom smjeru za 10 pixela.
   Napomena: pri pomicanju gumba treba voditi računa da niti jedan dio gumba ne izađe izvan okvira prozora.

6. Napiši program koji će crtati ekran i na njemu jedan gumb. Kada se program učita na gumbu treba pisati tekst ***Broj: 0***. Svakim klikom na gumb tekst na gumbu se treba izmijeniti i to tako da se izmijeni broj pokraj teksta Broj (nakon prvog klika na gumbu treba pisati ***Broj: 1***, nakon drugog ***Broj: 2***,...)


7. Napiši program koji će na ekranu crtati prozor i dva gumba (Gumb 1 i Gumb 2). Na početku će biti aktivan samo prvi gumb (Gumb 1), klikom na prvi gumb on postaje neaktivan a aktivan postaje drugi gumb. Klikom na drugi gumb on postaje neaktivan a aktivan postaje ponovo prvi gumb,...