

Keret (ablak) kezelés

Az ablakok kezeléséhez a *javax.swing* csomag osztályait fogjuk használni, mivel ezek platformfüggetlenséget és könnyű kezelhetőséget biztosítanak.

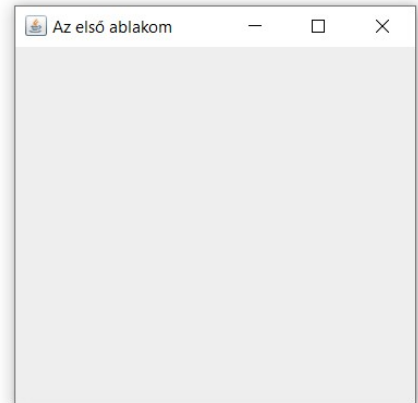
Példa: Hozzunk létre egy üres keretet, fejléccel.

Megbeszélés:

1. *JFrame* osztály,
2. ablakméret beállítása: *setSize()* metódus,
3. fejléc szövege: *setTitle()* metódus,
4. ablak bezárása: *EXIT_ON_CLOSE*,
5. az ablak megjelenítése: *setVisible()* metódus.

Megoldás:

```
package jframe_1;
import javax.swing.JFrame;
public class JFrame_1 {
    public static void main(String[] args) {
        JFrame Keret = new JFrame();
        Keret.setSize(500, 500);
        Keret.setTitle("Az első ablakom");
        Keret.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Keret.setVisible(true);
    }
}
```



Példa:

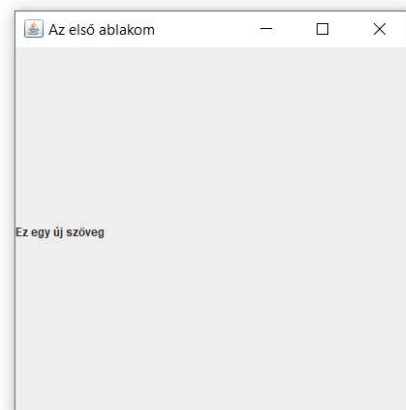
Készítsük el az előző feladatot úgy, hogy új osztályt hozunk létre az ablak kezelésére. Egészítsük ki a feladatot egy szöveg kiírással és az ablak automatikus méretezésével.

Megbeszélés:

1. *JLabel* osztály,
2. ablakméret automatikus beállítása: *pack()* metódus.

Megoldás:

```
package jframe_2;
import javax.swing.JFrame;
public class JFrame_2 {
    public static void main(String[] args) {
        Keret k = new Keret();
        k.Ablak();
    }
}
class Keret extends JFrame{
    public void Ablak(){
        setTitle("Az első ablakom");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        add(new JLabel("Ez egy új szöveg"));
        //
        setSize(500, 500);
        pack();
        setVisible(true);
    }
}
```



Hallgatói feladat: Hozzunk létre a képernyő négy sarkában egy-egy ablakot (internet).

Példa:

Hozzunk létre egy tetszőleges méretű ablakot egy megadott pozícióba. Írjunk ki legalább három sor szöveget különböző beállításokkal.

Megbeszélés:

1. ablak helyének megadása: *setLocation()* metódus,
2. az elrendezés beállítása *setLayout()* metódus,
3. szövegdoz helyének és méretének megadása *setBounds(x, y, w, h)*,
4. szövegdoz hátterszínének megadása: *setBackground()* metódus,
5. szöveg színének megadása: *setForeground()* metódus,
6. a *Color* osztály,
7. szöveg vízszintes igazítása a szövegdozban belül: *setHorizontalAlignment()* metódus,
8. szöveg függőleges igazítása a szövegdozban belül: *setVerticalAlignment()* metódus,

Megoldás:

```
package jframe_3;
import java.awt.Color;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.SwingConstants;

public class Jframe_3 {
    public static void main(String[] args) {
        Keret k = new Keret();
        k.Ablak();
    }
}

class Keret extends JFrame{
    JLabel szoveg;
    public void Ablak(){
        setTitle("A második ablakom");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(500, 500);
        setLocation(200, 200);
        setLayout(null);
        szoveg = new JLabel("Ez az első sor szövege");
        szoveg.setOpaque(true);
        szoveg.setBackground(Color.blue);
        szoveg.setForeground(Color.WHITE);
        szoveg.setHorizontalAlignment(SwingConstants.LEFT);
        szoveg.setVerticalAlignment(SwingConstants.TOP);
        szoveg.setBounds(10, 10, 200, 50);
        add(szoveg);
        szoveg = new JLabel("Ez a második sor szövege");
        szoveg.setOpaque(true);
        szoveg.setBackground(Color.YELLOW);
        szoveg.setForeground(Color.CYAN);
        szoveg.setHorizontalAlignment(SwingConstants.CENTER);
```

```

        szoveg.setVerticalAlignment(SwingConstants.CENTER);
        szoveg.setBounds(100, 100, 200, 50);
        add(szoveg);
        szoveg = new JLabel("Ez a harmadik sor szövege");
        szoveg.setOpaque(true);
        szoveg.setBackground(Color.green);
        szoveg.setForeground(Color.red);
        szoveg.setHorizontalAlignment(SwingConstants.RIGHT);
        szoveg.setVerticalAlignment(SwingConstants.BOTTOM);
        szoveg.setBounds(200, 200, 200, 50);
        add(szoveg);
        setVisible(true);
    }
}

```



Hallgatói feladat: Hozzunk létre egy ablakot, amelyben a magyar zászló színeiben elhelyez egy odaillő vers három sorát.

Példa:

Kérjünk be billentyűzetről neveket. Számoljuk meg a nevekben lévő karaktereket, majd mentjük el egy szöveges fájlba a nevek sorszámát, a neveket és a nevek hosszát. A fájl tartalmát egy ablakba írjuk ki.

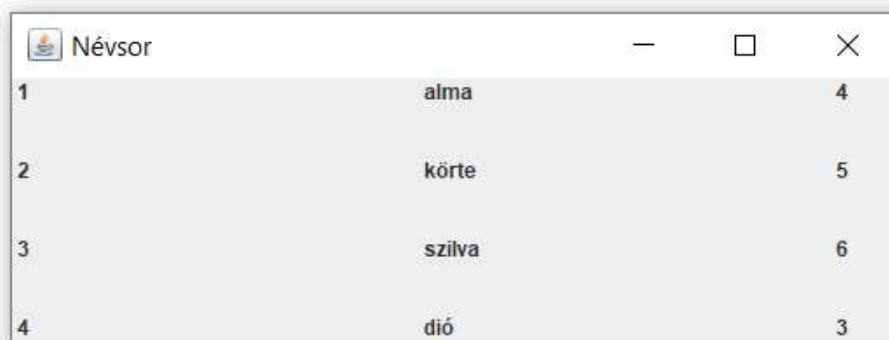
Megoldás:

```
package label_1;
import java.awt.GridLayout;
import static java.lang.System.out;
import java.util.Scanner;
import java.util.StringTokenizer;
import java.io.File;
import java.io.PrintStream;
import java.io.IOException;
import javax.swing.JFrame;
import javax.swing.JLabel;
public class Label_1 {
    public static void main(String[] args) {
        Beolvas be = new Beolvas();
        Kiír ki = new Kiír();
        ki.Ki();
    }
}
class Beolvas{
    public Beolvas(){
        Scanner billentyu = new Scanner (System.in, "ISO8859_2");
        String nev;
        int ssz = 1;
        try{
            try (PrintStream file = new PrintStream("Fájl.txt")) {
                out.print("Kérek egy nevet: ");
                nev = billentyu.nextLine();
                while(nev.length() != 0){
                    file.printf("%d-%s-%d\n", ssz, nev, nev.length());
                    ssz++;
                    out.print("Kérek egy nevet: ");
                    nev = billentyu.nextLine();
                }
            }
        }
        catch (IOException error) {
            System.err.println("Hiba: " + error.getMessage());
        }
    }
}
```

```

class Kiír extends JFrame {
    public void Ki() {
        String sor;
        int sorok=0;
        StringTokenizer token;
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        try (Scanner file = new Scanner(new File("Fájl.txt"))) {
            while(file.hasNext()) {
                sor = file.nextLine();
                sorok++;
                token = new StringTokenizer(sor, "-");
                add(new JLabel(" " + (String) token.nextElement()));
                add(new JLabel((String) token.nextElement()));
                add(new JLabel((String) token.nextElement()));
            }
            file.close();
        }
        catch (IOException error) {
            System.err.println("Hiba: " + error.getMessage());
        }
        finally{
            setTitle("Névsor");
            setLayout(new GridLayout(sorok, 3, 200, 30));
            pack();
            setVisible(true);
        }
    }
}

```



1	alma	4
2	körte	5
3	szilva	6
4	dió	3

Hallgatói feladat: Hozzunk létre egy listát a következő tagokat tartalmazó objektummal:

int azon;
string név;
float fizetés;

Billentyűzetről töltse fel néhány adattal (pl. azon = 0 végjelig). Feltöltés után a lista tartalmát megformázva írja ki egy ablakba.

Példa:

Igazítsuk középre az ablakot és tiltsuk le az átméretezés lehetőségét.

Megbeszélés:

1. az ablak méretének lekérdezése *getScreenSize()* metódus,
2. ablak méretének tárolása, *Dimension* osztály.
3. *setResizable(false)* metódus,
4. *FlowLayout()* metódus.

Megoldás:

```
package label_2;
import java.awt.*;
import javax.swing.*;

public class Label_2 {
    JFrame ablak;
    JLabel cimke1;
    JLabel cimke2;
    public static void main(String[] args) {
        Ablak ob = new Ablak();
    }
}

class Ablak extends JFrame {
    public Ablak(){
        JFrame ablak = new JFrame("Ablak középre");
        JLabel szöveg1 = new JLabel("Első sor");
        JLabel szöveg2 = new JLabel("Második sor");
        ablak.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        ablak.setLayout(null);
        ablak.setSize(800, 600);
        setCenter(ablak);
        ablak.setResizable(false);
        szöveg1.setBounds(10, 10, 200, 30);
        szöveg2.setBounds(30, 30, 200, 30);
        ablak.add(szöveg1);
        ablak.add(szöveg2);
        ablak.setVisible(true);
    }

    private static void setCenter(Window frame){
        Dimension közép = Toolkit.getDefaultToolkit().getScreenSize();
        int x = (int) ((közép.getWidth() - frame.getWidth()) / 2);
        int y = (int) ((közép.getHeight() - frame.getHeight()) / 2);
        frame.setLocation(x, y);
    }
}
```

Példa:

Írjuk ki egy szöveges fájl tartalmát egy görgethető ablakba.

Megbeszélés:

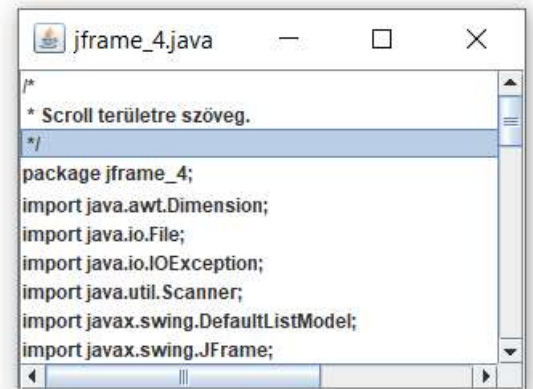
1. a *JList* osztály,
2. a *JScrollPane* osztály.

Megoldás:

```
package jframe_4;
import java.awt.Dimension;
import java.io.File;
import java.io.IOException;
import java.util.Scanner;
import javax.swing.DefaultListModel;
import javax.swing.JFrame;
import static javax.swing.JFrame.EXIT_ON_CLOSE;
import javax.swing.JList;
import javax.swing.JScrollPane;

public class Jframe_4 {
    public static void main(String[] args) {
        Kiír k = new Kiír();
        k.Ki();
    }
}

class Kiír extends JFrame {
    public void Ki() {
        String sor;
        DefaultListModel<String> lm = new DefaultListModel<>();
        JList<String> jl = new JList<>(lm);
        JScrollPane scroll = new JScrollPane(jl);
        scroll.setPreferredSize(new Dimension(300, 200));
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setLocation(400,400);
        try (Scanner file = new Scanner(new File("src\\jframe_4\\jframe_4.java"))) {
            while(file.hasNext()) {
                sor = file.nextLine();
                lm.addElement(sor);
            }
            file.close();
        }
        catch (IOException error) {
            System.err.println("Hiba: " + error.getMessage());
        }
        finally{
            add(scroll);
            setTitle("jframe_4.java ");
            pack();
            setVisible(true);
        }
    }
}
```



Példa:

Jelenítsünk meg egy képet egy teljesképernyős ablakban.

Megbeszélés:

1. a *paintComponent()* metódus a grafikus megjelenítéshez szükséges, a rendszer hívja meg,
2. várakozás a kép betöltésére: *MediaTracker* osztály, *waitForID()* metódus,
3. a *getImage()* metódus,
4. a *gr.drawImage(image, xbf, ybf, xja, yja, null)* metódus.

Megoldás:

```
package image_2;
import java.awt.Graphics;
import java.awt.Image;
import java.awt.MediaTracker;
import java.awt.Toolkit;
import javax.swing.JFrame;
import javax.swing.JPanel;
public class Image_2 {
    public static void main(String[] args) {
        Kép k = new Kép();
        k.kép();
    }
}
class Kép extends JFrame{
    public void kép(){
        setSize(Toolkit.getDefaultToolkit().getScreenSize());
        getContentPane().add(new Kép_be());
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setVisible(true);
    }
}
class Kép_be extends JPanel{
    private Image im;
    public Kép_be(){
        MediaTracker tr = new MediaTracker(this);
        im = Toolkit.getDefaultToolkit().getImage("Kép.jpg");
        tr.addImage(im, 0);
        try{
            tr.waitForID(0);
        }
        catch(InterruptedException err){}
        finally{
            tr.removeImage(im, 0);
        }
    }
    @Override
    protected void paintComponent(Graphics gr){
        super.paintComponent(gr);
        gr.drawImage(im, 100, 100, 400, 400, this);
    }
}
```


Feladatok:

1. Írjon programot, amely e program forrásállományát (.java fájl) kiírja a képernyőre egy keretben.
2. Írjon programot, amely e program forrásállományát (.java fájl) kiírja a képernyőre egy keretben, két színű, soronként váltakozó háttérszínnel.
3. Írjon programot, amely e program forrásállományát (.java fájl) kiírja a képernyőre két keretbe úgy, hogy minden páratlan sort az egyikbe és minden páros sort a másikba ír.
4. Írjon programot, amely e program forrásállományát (.java fájl) kiírja a képernyőre két keretbe úgy, hogy minden páratlan sort az egyikbe és minden páros sort a másikba ír, valamint a szöveg karakterszínét soronként váltogatja.
5. Írjon programot, amely e program forrásállományát (.java fájl) kiírja a képernyőre soronként egy-egy keretbe úgy, hogy a keret címe legyen a sor szövege.
6. Írjon programot, amely e program forrásállományát (.java fájl) kiírja a képernyőre egy keretbe úgy, hogy a szöveg két hasábos legyen.
7. Olvassuk be billentyűzetről egy téglatest paramétereit (3 oldal hossza). Egy-egy módszer segítségével számolja ki a térfogatát és a felszínét. A három paramétert a térfogat és a felszín értékét egy ablakban jelenítse meg.