

**SZÉCHENYI ISTVÁN EGYETEM**  
**GÉPÉSZMÉRNÖKI, INFORMATIKAI ÉS VILLAMOSMÉRNÖKI KAR**  
**MATEMATIKA ÉS SZÁMÍTÁSTUDOMÁNY TANSZÉK**



**FÉLÉVES FELADAT**

**Fotel**

**Balics Attila Ádám, Balogh Levente**  
Programtervező informatikus BSc

**Győr, 2025**

**ÉRTÉKELŐLAP**  
**A FÉLÉVES PROGRAMOZÁSI FELADATOKHOZ**

**Hallgató(k):**.....

**Értékelő neve:**.....

**Feladat:**.....**Összpontszám:**.....

**1. Dokumentáció:.....pont**

- **4..5:** A dokumentáció teljesíti mind a tartalmi, mind a formai követelményeket, azaz kellő részletességgel tartalmazza az előírt részeket, gondosan szerkesztett, áttekinthető, stílus és nyelvhelyességi szempontból is jó.
- **2..3:** Kisebb hiányosságok vannak a jó esethez képest.
- **0:** Lényeges hiányosságok vannak a jó esethez képest (pl. erősen keverednek vagy teljesen hiányoznak egyes részek, 'ömlesztett' szöveg, sok elírási, helyesírási hiba).

**2. A program funkcionalitása, működése:.....pont**

- **4..5:** A program tartalmazza az összes előírt funkciót, biztonságosan és helyesen működik.
- **2..3:** Kisebb hiányosságok vannak a jó esethez képest (pl. hiányzó funkciók; a program 'normál' tesztadatokkal biztonságosan és helyesen működik, de a 'szélső' esetek kezelése 'bizonytalan'; nem ellenőrzött input).
- **0:** A program a megadott forrásanyagokból az előírt környezetben nem állítható elő (pl. szintaktikus hiba), vagy 'normál' tesztadatokkal is helytelen eredményeket és/vagy futási hibákat produkál.

**3. A program megvalósítása (szerkezet, küllem, használat):.....pont**

- **4..5:** A program megfelelően strukturált, moduláris; tetszetős küllemű és könnyen kezelhető.
- **2..3:** Kisebb hiányosságok vannak a jó esethez képest (pl. publikus adattagok, a tanultnál alacsonyabb színvonalú menükezelés, 'összecsapott' sűrű, nehézkes kezelés).
- **0:** Lényeges hiányosságok vannak a jó esethez képest (pl. az objektum-orientált programozási elvek következetes be nem tartása, 'átláthatatlan' forráskód).

**4. Bemutató:.....pont**

- **4..5:** A bemutató mind tartalmát, mint küllemét tekintve megfelel az előírásoknak.
- **2..3:** Kisebb hiányosságok vannak a jó esethez képest (pl. túl részletes leírások felsorolások helyett; ábrák, futási képek hiánya).
- **0:** Lényeges hiányosságok vannak a jó esethez képest (pl. hiányzó részek; sok elírási, helyesírási hiba).

**Értékelés:**

- Ha valamely értékelési szempont szerint a munka 0 értékű, akkor nem elfogadható.
- Az egyes szempontok azonos súllyal számítanak, tehát a maximális összpontszám 20.
- Az érdemjegyek ponthatárai: 0-7:1, 8-10:2, 11-13:3, 14-16:4, 17-20:5.

<b>1. Bevezetés</b>	<b>3</b>
<b>2. Tervezési dokumentáció</b>	<b>4</b>
<b>2.1. Fejlesztői környezet</b>	<b>4</b>
2.1.1. Hardver:	4
2.1.2. Operációs rendszer:	4
2.1.3. Fejlesztői környezet:	4
2.1.4. Programozási nyelv:	4
2.1.5. Könyvtárak:	4
<b>2.2. A program főbb osztályai és kapcsolataik (UML osztálydiagram)</b>	<b>4</b>
2.2.1. Fotel kapcsolatai	4
2.2.2. ArmCharViewPanel kapcsolatai	4
2.2.3. FileManager kapcsolatai	4
2.2.4. MenuBuilder kapcsolatai	4
2.2.5. KeyBindManager kapcsolatai	4
<b>2.3. Osztályleírások</b>	<b>5</b>
2.3.1. Main	5
2.3.2. Fotel	5
2.3.3. ArmChairModel	5
2.3.4. ArmChairViewPanel	5
2.3.5. FileManager	5
<b>2.4. Főbb algoritmusok</b>	<b>5</b>
2.4.1. Véletlenszerű példány generálása	5
2.4.2. Rajzolás	5
2.4.3. Fájlkezelés	5
<b>2.5. Menürendszer</b>	<b>6</b>
2.5.1. Fájl	6
2.5.2. Nézet	6
2.5.3. Súgó	6
<b>3. Felhasználói útmutató</b>	<b>7</b>
<b>3.1. A program indítása</b>	<b>7</b>
<b>3.2. Fő ablak felépítése</b>	<b>7</b>
<b>3.3. Fő funkciók</b>	<b>7</b>
3.3.1. Méretek állítása	7
3.3.2. Színek állítása	7
3.3.3. Fájl műveletek (Fájl menü és gyorsbillentyűk)	7
3.3.4. Randomizálás (Nézet menü és gyorsbillentyűk)	7
3.3.5. Súgó	7
3.3.6. Kilépés	8
3.3.7. Fókusz mód	8
3.3.8. Fókusz nézet választás	8
<b>3.4. Egyéb kezelési lehetőségek</b>	<b>8</b>
<b>3.5. Hibaüzenetek, Figyelmeztetések</b>	<b>8</b>
<b>4. Melléklet</b>	<b>9</b>



# 1. Bevezetés

A féléves feladat keretében egy fotel, objektum orientált modellezését és kezelőprogramjának elkészítését kaptunk feladatul. A program célja, hogy bemutasson egy fotel példányt, valamint különböző vetületek (felülnézet, oldalnézet, előlnézet) segítségével szemléltetni tudja annak felépítését.

A program az alábbi főbb funkciókat valósítja meg:

- Generálás:  
Véletlenszerű paraméterekkel létrehoz egy új fotel példányt.
- Mentés:  
A példányt fájlba menti a felhasználó által megadott név alapján.
- Betöltés:  
Fájlból betölt egy meglévő példányt.
- Törlés:  
Egy adott azonosítóval rendelkező példányt eltávolít.
- Rajzolás:  
Grafikus bemenettel létrehozható, illetve módosítható egy példány.
- Bemutató:  
Megjeleníti a példány jellemzőit és vetületeit. A feladat során törekedtünk a méretarányok helyes arányainak betartására, valamint az objektumorientált elvek – mint például az osztályok megfelelő szétválasztása és a felelősségi körök világos elhatárolása – betartására. A fejlesztés során Java nyelvet és az objektumorientált tervezés eszközeit használtunk, különös figyelmet fordítva a jól strukturált, áttekinthető kódra és a felhasználóbarát működésre.

## 2. Tervezési dokumentáció

### 2.1. Fejlesztői környezet

#### 2.1.1. Hardver:

IBM PC kompatibilis számítógép, minimum 2 GB RAM, 1 GHz processzor, 100 MB szabad tárhely.

#### 2.1.2. Operációs rendszer:

Windows 10 vagy újabb, Linux, vagy macOS.

#### 2.1.3. Fejlesztői környezet:

IntelliJ IDEA.

#### 2.1.4. Programozási nyelv:

Java 17 vagy újabb.

#### 2.1.5. Könyvtárak:

Java SE (Swing, AWT).

### 2.2. A program főbb osztályai és kapcsolataik (UML osztálydiagram)

(lásd 2.2. ábra)

#### 2.2.1. Fotel kapcsolatai

ArmChairModel – model

ArmChairViewPanel – viewPanel

FileManager – fileManager

MenuBuilder – menuBuilder

KeyBindManager – keyBindManager

#### 2.2.2. ArmCharViewPanel kapcsolatai

ArmChairModel – model

#### 2.2.3. FileManager kapcsolatai

Fotel – parent

ArmChairModel – model

#### 2.2.4. MenuBuilder kapcsolatai

Fotel – parent

#### 2.2.5. KeyBindManager kapcsolatai

Fotel – parent

## 2.3. Osztályleírások

### 2.3.1. Main

Feladata: A program belépési pontja, elindítja a Swing GUI-t.

Fő metódus: `main(String[] args)`

### 2.3.2. Fotel

Feladata: A fő ablak, menürendszer, vezérlés.

Fő adattagok: `ArmChairModel` példány, `ArmChairViewPanel`, menü.

Fő metódusok: `updateView()`, `normalViewMenu()`, `focusViewMenu()`

(lásd 2.3.2.1., 2.3.2.2. és 2.3.2.3. ábra)

### 2.3.3. ArmChairModel

Feladata: Egy fotel példány adatainak tárolása (méretek, színek).

### 2.3.4. ArmChairViewPanel

Feladata: A fotel grafikus megjelenítése különböző nézetekben (elől-, oldal-, felülnézet).

Fő metódusok: `drawFrontView()`, `drawSideView()`, `drawTopView()`

(lásd 2.3.4.1., 2.3.4.2. és 2.3.4.3. ábra)

### 2.3.5. FileManager

Feladata: Fájlba mentés, betöltés, törlés.

Fő metódusok: `saveToFile()`, `loadFromFile()`, `saveLocal()`, `loadLocal()`, `deleteLocal()`

(lásd 2.3.5.1., 2.3.5.2., 2.3.5.3., 2.3.5.4. és 2.3.5.5. ábra)

## 2.4. Főbb algoritmusok

### 2.4.1. Véletlenszerű példány generálása

A fő méretek és színek (szélesség, magasság, mélység, stb.) előre megadott tartományból véletlenszerűen kerülnek kiválasztásra.

(lásd 2.4.1. ábra)

### 2.4.2. Rajzolás

A `paintComponent(Graphics g)` metódusban a modell aktuális állapotától függően, különböző színű és méretű téglalapok, vonalak jelennek meg.

A nézetek között menüből lehet váltani.

### 2.4.3. Fájlkezelés

A mentés és betöltés során a modell adatai szöveges formátumban kerülnek kiírásra/beolvasásra.

Az azonosító alapján történik a fájlok elnevezése.

(lásd 2.4.3.1. és 2.4.3.2. ábra)

## 2.5. Menürendszer

### 2.5.1. Fájl

Megnyitás

Mentés

Törlés

Kilépés

(lásd 2.5.1. ábra)

### 2.5.2. Nézet

Előlnézet

Oldalnézet

Felülnézet

Randomizálási opciók

(lásd 2.5.2. ábra)

### 2.5.3. Súgó

Súgó megnyitása

(lásd 2.5.3. ábra)



## 3. Felhasználói útmutató

### 3.1. A program indítása

A program futtatásához Java 17 vagy újabb szükséges. Futtassa a mellékelt Fotel.bat file-t és az alkalmazás elindul.

(lásd 3.1. ábra)

### 3.2. Fő ablak felépítése

Bal oldali panel: Méretállító csúszkák (szélesség, magasság, mélység) és színválasztó gombok (alapszín, párna, láb, díszpárna).

Középső rész: Négy nézet (előlnézet, oldalnézet, felülnézet, méretek).

Menüsor: Fájl, Nézet, Súgó menük.

(lásd 3.2. ábra)

### 3.3. Fő funkciók

#### 3.3.1. Méretek állítása

A bal oldali csúszkákkal vagy a [1], [2], [3] billentyűkkel választható ki a fókusz (szélesség, magasság, mélység).

A fókuszált csúszka értéke a nyilakkal növelhető/csökkenthető (jobb/bal/fel/le).

Shift + nyíl nagyobb lépés.

(lásd 3.3.1. ábra)

#### 3.3.2. Színek állítása

A színválasztó gombokkal ([4] – [7]) külön-külön módosítható az alapszín, párna, láb, díszpárna színe.

(lásd 3.3.2. ábra)

#### 3.3.3. Fájl műveletek (Fájl menü és gyorsbillentyűk)

Mentés helyileg: [Ctrl + S] – Fotel mentése a program könyvtárába.

Mentés másként: [Ctrl + Shift + S] – Mentés tetszőleges helyre.

Betöltés: [Ctrl + O] – Mentett fotel betöltése a program könyvtárából.

Betöltés máshonnan: [Ctrl + Shift + O] – Betöltés tetszőleges helyről.

Törlés: [Ctrl + Del] – Mentett fotel törlése.

(lásd 3.3.3. ábra)

#### 3.3.4. Randomizálás (Nézet menü és gyorsbillentyűk)

Minden randomizálása: [Ctrl + R]

Méretek randomizálása: [Ctrl + T]

Színek randomizálása: [Ctrl + Z]

(lásd 3.3.4. ábra)

#### 3.3.5. Súgó

[Ctrl + H] vagy Súgó menüpont – részletes használati útmutató.

(lásd 3.3.5. ábra)

### 3.3.6. Kilépés

[Esc] vagy Fájl → Kilépés  
(lásd 3.3.6. ábra)

### 3.3.7. Fókusz mód

Belépés [8]  
Kilépés [0]  
(lásd 3.3.7. ábra)

### 3.3.8. Fókusz nézet választás

Előlnézet [1]  
Oldalnézet [2]  
Felülnézet [3]  
(lásd 3.3.8. ábra)

## 3.4. Egyéb kezelési lehetőségek

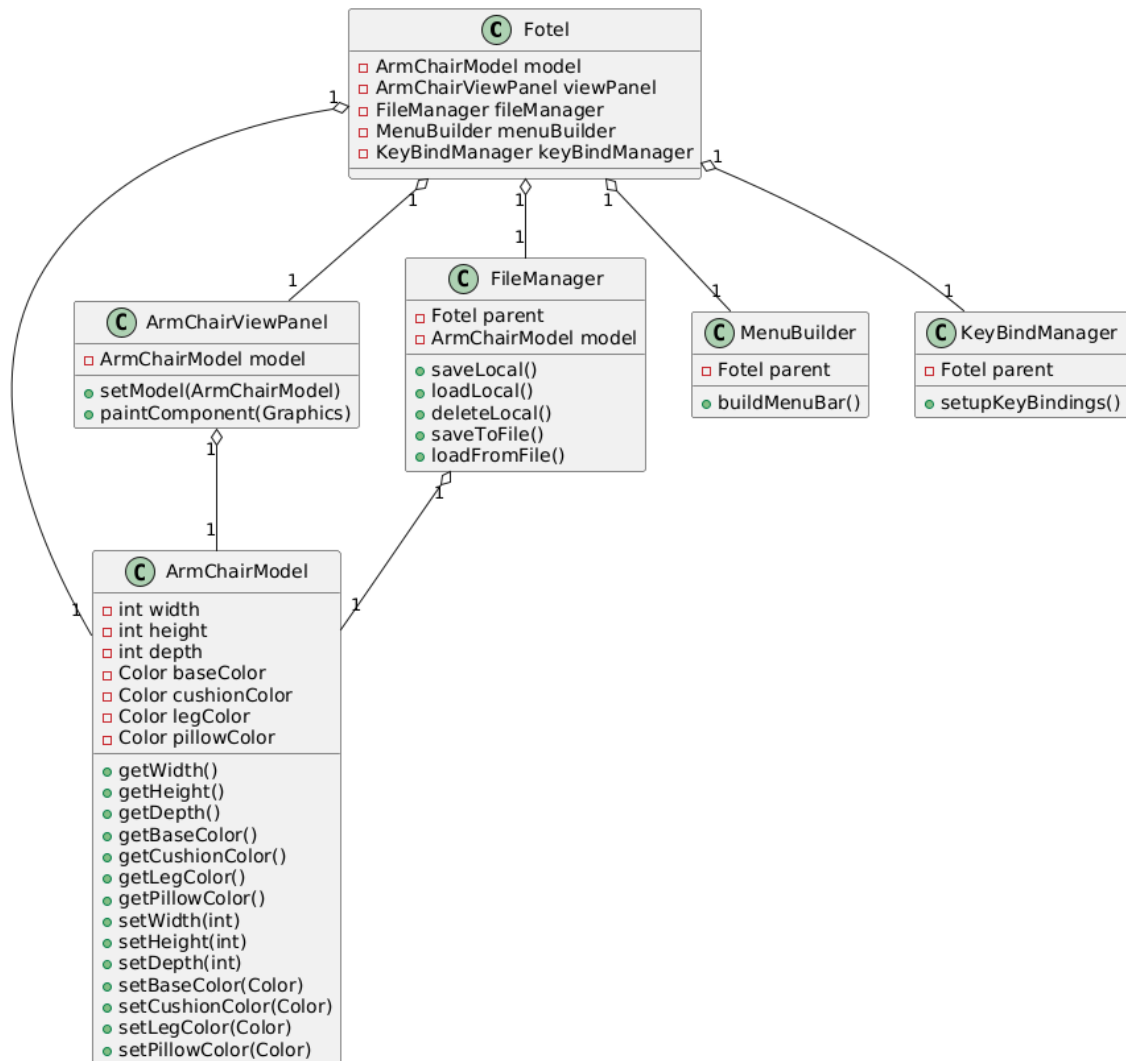
A fókuszált csúszka gyors kiválasztása: [1] – szélesség, [2] – magasság, [3] – mélység.  
Színválasztó gyors elérés: [4] – alapszín, [5] – párna, [6] – láb, [7] – díspárna.

## 3.5. Hibaüzenetek, Figyelmeztetések


Hibás vagy hiányzó fájlnev esetén a program figyelmeztet. Sikeres mentés, betöltés, törlés után visszajelzést kap. A program minden művelet után automatikusan frissíti a nézeteket.

## 4. Melléklet

### 2.2. UML diagram



### 2.3.2.1. updateViews()



```
1 public void updateView() {
2     getContentPane().removeAll();
3
4     switch (setFocusOnView) {
5         case 0:
6             normalViewMenu();
7             break;
8         case 1:
9             focusViewMenu();
10            break;
11    }
12    new KeyBindingManager(this, model).setupKeyBindings();
13
14    revalidate();
15    repaint();
16 }
```

### 2.3.2.2. normalViewMenu()



```
1 private void normalViewMenu() {
2     setTitle("Fotel");
3
4     JPanel controlPanel = createControlPanel();
5     add(controlPanel, BorderLayout.WEST);
6
7     JPanel gridPanel = new JPanel(new GridLayout(2, 2));
8     gridPanel.add(new ArmChairViewPanel("Előlnézet", model) {
9         @Override
10        protected void drawView(Graphics g) {
11            drawFrontView(g);
12        }
13    });
14
15     gridPanel.add(new ArmChairViewPanel("Oldalnézet", model) {
16         @Override
17        protected void drawView(Graphics g) {
18            drawSideView(g);
19        }
20    });
21
22     gridPanel.add(new ArmChairViewPanel("Felülnézet", model) {
23         @Override
24        protected void drawView(Graphics g) {
25            drawTopView(g);
26        }
27    });
28
29     gridPanel.add(new ArmChairViewPanel("Méretek", model) {
30         @Override
31        protected void drawView(Graphics g) {
32            drawSizes(g);
33        }
34    });
35
36     add(gridPanel, BorderLayout.CENTER);
37
38     setSize(1000, 700);
39     setLocationRelativeTo(null);
40
41     setupFocusHandling();
42     getContentPane().setFocusable(true);
43     getContentPane().requestFocusInWindow();
44
45     setVisible(true);
46
47     new KeyBindingManager(this, model).setupKeyBindings();
48 }
```

### 2.3.2.3. focusedViewMenu()



```
1 private void focusViewMenu() {
2     setTitle("Fókuszált Nézet - Fotel");
3
4     JPanel gridPanel = new JPanel(new GridLayout(1, 1));
5
6     ArmChairViewPanel viewPanel = new ArmChairViewPanel("Nézet", model) {
7         @Override
8         protected void drawView(Graphics g) {
9             switch (focusedViewIndex) {
10                 case 0:
11                     drawFrontView(g);
12                     break;
13                 case 1:
14                     drawSideView(g);
15                     break;
16                 case 2:
17                     drawTopView(g);
18                     break;
19             }
20         }
21     };
22 }
```

### 2.3.4.1. Rajzolás előlnézet

```
1  protected void drawFrontView(Graphics g) {
2      int width = model.getWidth();
3      int height = model.getHeight();
4      int armWidth = model.getArmWidth();
5      int legHeight = model.getLegHeight();
6
7      int seatHeight = height / 4;
8      int x = (getWidth() / 2) - (width / 2);
9      int y = getHeight() - 60;
10
11     g.setColor(model.getLegColor());
12     g.fillRect(x + 5, y, 10, legHeight);
13     g.fillRect(x + width - 15, y, 10, legHeight);
14     g.setColor(Color.BLACK);
15     g.drawRect(x + 5, y, 10, legHeight);
16     g.drawRect(x + width - 15, y, 10, legHeight);
17
18     g.setColor(model.getBaseColor());
19     g.fillRect(x, y - (int) (height * 0.8) + legHeight, armWidth, (int) (height * 0.8) - legHeight);
20     g.fillRect(x + width - armWidth, y - (int) (height * 0.8) + legHeight, armWidth,
21         (int) (height * 0.8) - legHeight);
22     g.fillRect(x + armWidth, y - height + legHeight, width - 2 * armWidth, height - seatHeight);
23     g.setColor(Color.BLACK);
24     g.drawRect(x, y - (int) (height * 0.8) + legHeight, armWidth, (int) (height * 0.8) - legHeight);
25     g.drawRect(x + width - armWidth, y - (int) (height * 0.8) + legHeight, armWidth,
26         (int) (height * 0.8) - legHeight);
27     g.drawRect(x + armWidth, y - height + legHeight, width - 2 * armWidth, height - seatHeight);
28
29     g.setColor(model.getCushionColor());
30     g.fillRect(x + armWidth, y - seatHeight, width - 2 * armWidth, seatHeight);
31     g.setColor(Color.BLACK);
32     g.drawRect(x + armWidth, y - seatHeight, width - 2 * armWidth, seatHeight);
33
34     g.setColor(model.getPillowColor());
35     g.fillRect(x + armWidth + 5, y - seatHeight - height / 4 - 1, width - (armWidth * 2) - 10, height / 4);
36     g.setColor(Color.BLACK);
37     g.drawRect(x + armWidth + 5, y - seatHeight - height / 4 - 1, width - (armWidth * 2) - 10, height / 4);
38 }
```

### 2.3.4.2. Rajzolás oldalnézet

```
1  protected void drawSideView(Graphics g) {
2      int height = model.getHeight();
3      int depth = model.getDepth();
4      int armWidth = model.getArmWidth();
5      int legHeight = model.getLegHeight();
6
7      int x = (getWidth() / 2) - (depth / 2);
8      int y = getHeight() - 60;
9
10     g.setColor(model.getLegColor());
11     g.fillRect(x + 5, y, 10, legHeight);
12     g.fillRect(x + depth - 15, y, 10, legHeight);
13     g.setColor(Color.BLACK);
14     g.drawRect(x + 5, y, 10, legHeight);
15     g.drawRect(x + depth - 15, y, 10, legHeight);
16
17     g.setColor(model.getBaseColor());
18     g.fillRect(x, y - height + legHeight, armWidth, (int) (height * 0.2));
19     g.fillRect(x, y - (int) (height * 0.8) + legHeight, depth, (int) (height * 0.8) - legHeight);
20     g.setColor(Color.BLACK);
21     g.drawRect(x, y - height + legHeight, armWidth, (int) (height * 0.2));
22     g.drawRect(x, y - (int) (height * 0.8) + legHeight, depth, (int) (height * 0.8) - legHeight);
23 }
```

### 2.3.4.3. Rajzolás felülnézet

```
1  protected void drawTopView(Graphics g) {
2      int width = model.getWidth();
3      int depth = model.getDepth();
4      int armWidth = model.getArmWidth();
5
6      int x = (getWidth() / 2) - (width / 2);
7      int y = (getHeight() / 2) - (depth / 2);
8
9      g.setColor(model.getBaseColor());
10     g.fillRect(x, y, armWidth, depth);
11     g.fillRect(x + width - armWidth, y, armWidth, depth);
12     g.fillRect(x + armWidth, y, width - 2 * armWidth, armWidth);
13     g.setColor(Color.BLACK);
14     g.drawRect(x, y, armWidth, depth);
15     g.drawRect(x + width - armWidth, y, armWidth, depth);
16     g.drawRect(x + armWidth, y, width - 2 * armWidth, armWidth);
17
18     g.setColor(model.getCushionColor());
19     g.fillRect(x + armWidth, y + armWidth, width - 2 * armWidth, depth - armWidth);
20     g.setColor(Color.BLACK);
21     g.drawRect(x + armWidth, y + armWidth, width - 2 * armWidth, depth - armWidth);
22
23     g.setColor(model.getPillowColor());
24     g.fillRect(x + armWidth + 5, y + armWidth + 1, width - (armWidth * 2) - 10, armWidth);
25     g.setColor(Color.BLACK);
26     g.drawRect(x + armWidth + 5, y + armWidth + 1, width - (armWidth * 2) - 10, armWidth);
27 }
```



### 2.3.5.1. saveToFile()

```
1 public void loadFromFile() {
2     JFileChooser fileChooser = new JFileChooser();
3     fileChooser.setDialogTitle("Betöltés fájlból");
4
5     int userSelection = fileChooser.showSaveDialog(parent);
6     if (userSelection == JFileChooser.APPROVE_OPTION) {
7         try {
8             File fileToLoad = fileChooser.getSelectedFile();
9             if (!fileToLoad.getName().toLowerCase().endsWith(".fotel")) {
10                 throw new Exception("A fájlnak .fotel kiterjesztésűnek kell lennie");
11             }
12             loadFromFile(fileToLoad);
13         } catch (Exception ex) {
14             JOptionPane.showMessageDialog(parent, "Hiba a betöltés közben:\n" + ex.getMessage(), "Hiba",
15                 JOptionPane.ERROR_MESSAGE);
16         }
17     }
18 }
```

### 2.3.5.2. loadFromFile()

```
1 public void saveToFile() {
2     JFileChooser fileChooser = new JFileChooser();
3     fileChooser.setDialogTitle("Mentés fájlba");
4
5     int userSelection = fileChooser.showSaveDialog(parent);
6     if (userSelection == JFileChooser.APPROVE_OPTION) {
7         try {
8             File fileToSave = fileChooser.getSelectedFile();
9             if (!fileToSave.getName().toLowerCase().endsWith(".fotel")) {
10                 fileToSave = new File(fileToSave.getAbsolutePath() + ".fotel");
11             }
12             PrintWriter writer = new PrintWriter(fileToSave);
13
14             writeToFile(writer);
15         } catch (Exception ex) {
16             JOptionPane.showMessageDialog(parent, "Hiba a mentés közben:\n" + ex.getMessage(), "Hiba",
17                 JOptionPane.ERROR_MESSAGE);
18         }
19     }
20 }
```

### 2.3.5.3. saveLocal()

```
1 public void saveLocal() {
2     SwingUtilities.invokeLater(() -> {
3         JPanel panel = new JPanel(new BorderLayout(10, 10));
4
5         JTextField filenameField = new JTextField(20);
6         JPanel inputPanel = new JPanel();
7         inputPanel.add(new JLabel("Fájl név:"));
8         inputPanel.add(filenameField);
9
10        JButton saveButton = new JButton("Mentés");
11        JButton cancelButton = new JButton("Vissza");
12
13        JPanel buttonPanel = new JPanel();
14        buttonPanel.add(saveButton);
15        buttonPanel.add(cancelButton);
16
17        panel.add(inputPanel, BorderLayout.CENTER);
18        panel.add(buttonPanel, BorderLayout.SOUTH);
19
20        JDialog dialog = new JDialog(parent, "Mentés", true);
21        dialog.setContentPane(panel);
22        dialog.pack();
23        dialog.setLocationRelativeTo(parent);
24
25        saveButton.addActionListener(_ -> {
26            String filename = filenameField.getText().trim();
27            if (filename.isEmpty()) {
28                File fileToSave = new File(filename);
29                if (!fileToSave.getName().toLowerCase().endsWith(".fotol")) {
30                    fileToSave = new File(fileToSave.getAbsolutePath() + ".fotol");
31                }
32                try (PrintWriter writer = new PrintWriter(fileToSave)) {
33
34                    writeToFile(writer);
35
36                    dialog.dispose();
37                } catch (IOException ex) {
38                    JOptionPane.showMessageDialog(parent, "Hiba a mentés közben:\n" + ex.getMessage(), "Hiba",
39                        JOptionPane.ERROR_MESSAGE);
40                }
41            } else {
42                JOptionPane.showMessageDialog(parent, "Hiba a mentés közben:\n A fájl név nem maradhat üressen.",
43                    "Hiba", JOptionPane.ERROR_MESSAGE);
44            }
45        });
46
47        cancelButton.addActionListener(_ -> dialog.dispose());
48
49        dialog.setVisible(true);
50    });
51 }
```

### 2.3.5.4. loadLocal()

```
1 public void loadLocal() {
2     JFrame frame = new JFrame("Fotel választó");
3     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
4     frame.setSize(400, 150);
5     frame.setLayout(new BorderLayout());
6
7     String rootPath = System.getProperty("user.dir");
8     File rootDir = new File(rootPath);
9
10    JComboBox<String> dropdown = new JComboBox<>();
11
12    File[] fotelFiles = rootDir.listFiles((_, name) -> name.endsWith(".fotel"));
13    if (fotelFiles != null) {
14        for (File file : fotelFiles) {
15            String baseName = file.getName().replaceFirst("\\.fotel$", "");
16            dropdown.addItem(baseName);
17        }
18    }
19
20    JButton loadButton = new JButton("Betöltés");
21    JButton returnButton = new JButton("Vissza");
22
23    JPanel topPanel = new JPanel();
24    topPanel.add(new JLabel("Válaszd ki a betöltendő modellt"));
25    topPanel.add(dropdown);
26
27    JPanel bottomPanel = new JPanel();
28    bottomPanel.add(loadButton);
29    bottomPanel.add(returnButton);
30
31    frame.add(topPanel, BorderLayout.NORTH);
32    frame.add(bottomPanel, BorderLayout.SOUTH);
33
34    loadButton.addActionListener(_ -> {
35        try {
36            File fileToLoad = new File(Objects.requireNonNull(dropdown.getSelectedItem()) + ".fotel");
37
38            loadFromFile(fileToLoad);
39            frame.dispose();
40        } catch (Exception ex) {
41            JOptionPane.showMessageDialog(parent, "Hiba a betöltés közben:\n" + ex.getMessage(), "Hiba",
42                JOptionPane.ERROR_MESSAGE);
43        }
44    });
45
46    returnButton.addActionListener(_ -> frame.dispose());
47
48    if (dropdown.getItemCount() == 0) {
49        JOptionPane.showMessageDialog(parent, "Nincs betölthető modell!", "Figyelmeztetés",
50            JOptionPane.WARNING_MESSAGE);
51        frame.dispose();
52        return;
53    }
54    frame.setVisible(true);
55 }
```

### 2.3.5.5. deleteLocal()

```
1 public void deleteLocal() {
2     JFrame frame = new JFrame("Fotel választó");
3     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
4     frame.setSize(400, 150);
5     frame.setLayout(new BorderLayout());
6
7     String rootPath = System.getProperty("user.dir");
8     File rootDir = new File(rootPath);
9
10    JComboBox<String> dropdown = new JComboBox<>();
11
12    File[] fotelFiles = rootDir.listFiles((_, name) -> name.endsWith(".fotel"));
13    if (fotelFiles != null) {
14        for (File file : fotelFiles) {
15            String baseName = file.getName().replaceFirst("\\.fotel$", "");
16            dropdown.addItem(baseName);
17        }
18    }
19
20    JButton deleteButton = new JButton("Törlés");
21    JButton returnButton = new JButton("Vissza");
22
23    JPanel topPanel = new JPanel();
24    topPanel.add(new JLabel("Válaszd ki a törlendő modellt"));
25    topPanel.add(dropdown);
26
27    JPanel bottomPanel = new JPanel();
28    bottomPanel.add(deleteButton);
29    bottomPanel.add(returnButton);
30
31    frame.add(topPanel, BorderLayout.NORTH);
32    frame.add(bottomPanel, BorderLayout.SOUTH);
33
34    deleteButton.addActionListener(_ -> {
35        try {
36            String selected = (String) dropdown.getSelectedItem();
37            if (selected == null)
38                return;
39            File fileToDelete = new File(System.getProperty("user.dir"), selected + ".fotel");
40            boolean deleted = fileToDelete.delete();
41            if (deleted) {
42                dropdown.removeItem(selected);
43                JOptionPane.showMessageDialog(parent, "Törlés sikeres!", "Törlés", JOptionPane.INFORMATION_MESSAGE);
44                if (dropdown.getItemCount() == 0) {
45                    frame.dispose();
46                }
47            } else {
48                JOptionPane.showMessageDialog(parent,
49                    "A fájl törlése nem sikerült!\nTeljes elérési út: " + fileToDelete.getAbsolutePath(),
50                    "Hiba", JOptionPane.ERROR_MESSAGE);
51            }
52        } catch (Exception ex) {
53            JOptionPane.showMessageDialog(parent, "Hiba a törlés közben:\n" + ex.getMessage(), "Hiba",
54                JOptionPane.ERROR_MESSAGE);
55        }
56    });
57
58    returnButton.addActionListener(_ -> frame.dispose());
59
60    frame.setVisible(true);
61 }
```

### 2.4.1. Randomizálás

```
1 public void Randomize(boolean color, boolean size) {
2     if (color) {
3         Random rnd = new Random();
4         baseColor = new Color(rnd.nextInt(256), rnd.nextInt(256), rnd.nextInt(256));
5         cushionColor = new Color(rnd.nextInt(256), rnd.nextInt(256), rnd.nextInt(256));
6         legColor = new Color(rnd.nextInt(256), rnd.nextInt(256), rnd.nextInt(256));
7         pillowColor = new Color(rnd.nextInt(256), rnd.nextInt(256), rnd.nextInt(256));
8     }
9     if (size) {
10        Random rnd = new Random();
11        width = rnd.nextInt(160, 300);
12        height = rnd.nextInt(160, 250);
13        depth = rnd.nextInt(160, 240);
14    }
15 }
```

### 2.4.3.1. Mentés fájlba

```
1 private void writeToFile(PrintWriter writer) {
2     writer.print(model.getWidth() + ";");
3     writer.print(model.getHeight() + ";");
4     writer.print(model.getDepth() + ";");
5     writer.print(model.getBaseColor().getRed() + ";" + model.getBaseColor().getGreen() + ";"
6         + model.getBaseColor().getBlue() + ";");
7     writer.print(model.getCushionColor().getRed() + ";" + model.getCushionColor().getGreen() + ";"
8         + model.getCushionColor().getBlue() + ";");
9     writer.print(model.getLegColor().getRed() + ";" + model.getLegColor().getGreen() + ";"
10        + model.getLegColor().getBlue() + ";");
11     writer.print(model.getPillowColor().getRed() + ";" + model.getPillowColor().getGreen() + ";"
12        + model.getPillowColor().getBlue() + "\n");
13
14     writer.close();
15     JOptionPane.showMessageDialog(parent, "Mentés sikeres!", "Mentés",
16         JOptionPane.INFORMATION_MESSAGE);
17 }
```


### 2.4.3.2. Betöltés fájlból

```
1 private void loadFromFile(File fileToLoad) throws FileNotFoundException {
2     try (Scanner sc = new Scanner(fileToLoad)) {
3         String[] data = sc.nextLine().split(";");
4         model.setWidth(Integer.parseInt(data[0]));
5         model.setHeight(Integer.parseInt(data[1]));
6         model.setDepth(Integer.parseInt(data[2]));
7         model.setBaseColor(
8             new Color(Integer.parseInt(data[3]), Integer.parseInt(data[4]), Integer.parseInt(data[5]));
9         model.setCushionColor(
10            new Color(Integer.parseInt(data[6]), Integer.parseInt(data[7]), Integer.parseInt(data[8]));
11        model.setLegColor(
12            new Color(Integer.parseInt(data[9]), Integer.parseInt(data[10]), Integer.parseInt(data[11]));
13        model.setPillowColor(
14            new Color(Integer.parseInt(data[12]), Integer.parseInt(data[13]), Integer.parseInt(data[14]));
15    }
16    JOptionPane.showMessageDialog(parent, "Betöltés sikeres!", "Betöltés", JOptionPane.INFORMATION_MESSAGE);
17 }
```

### 2.5.1. Fájl menü


```
1 private JMenu createFileMenu() {
2     JMenu fileMenu = new JMenu("Fájl");
3
4     fileMenu.add(createMenuItem("Mentés [Ctrl + S]", _ -> new FileManager(parent, model).saveLocal()));
5     fileMenu.add(createMenuItem("Mentés másként [Ctrl + Shift + S]", _ -> new FileManager(parent, model).saveToFile()));
6
7     fileMenu.add(createMenuItem("Betöltés [Ctrl + O]", _ -> {
8         new FileManager(parent, model).loadLocal();
9         parent.updateSliders();
10        parent.repaint();
11    }));
12
13    fileMenu.add(createMenuItem("Betöltés máshonnan [Ctrl + Shift + O]", _ -> {
14        new FileManager(parent, model).loadFromFile();
15        parent.updateSliders();
16        parent.repaint();
17    }));
18
19    fileMenu.add(createMenuItem("Törlés [Ctrl + DELETE]", _ -> new FileManager(parent, model).deleteLocal()));
20
21    fileMenu.add(createMenuItem("Kilépés [Esc]", _ -> System.exit(0)));
22
23    return fileMenu;
24 }
```

### 2.5.2. Nézet menü





```
1 private JMenu createViewMenu() {
2     JMenu viewMenu = new JMenu("Nézet");
3
4     viewMenu.add(createMenuItem("Minden Randomizálása [Ctrl + R]", _ -> {
5         model.Randomize(true, true);
6         parent.updateSliders();
7         parent.repaint();
8     }));
9
10    viewMenu.add(createMenuItem("Méretek randomizálása [Ctrl + T]", _ -> {
11        model.Randomize(false, true);
12        parent.updateSliders();
13        parent.repaint();
14    }));
15
16    viewMenu.add(createMenuItem("Színek randomizálása [Ctrl + Z]", _ -> {
17        model.Randomize(true, false);
18        parent.updateSliders();
19        parent.repaint();
20    }));
21
22    return viewMenu;
23 }
```

### 2.5.3. Súgó



```
1 private JButton createHelpButton() {
2     JButton helpButton = new JButton("Súgó");
3     helpButton.setFocusable(false);
4     helpButton.addActionListener(_ -> Fotel.help());
5     helpButton.setBorderPainted(false);
6     helpButton.setContentAreaFilled(false);
7     helpButton.setFocusPainted(false);
8     helpButton.setOpaque(false);
9     helpButton.setToolTipText("Súgó [Ctrl + H]");
10    return helpButton;
11 }
```

### 3.1. A program indítása

 Fotel.bat	15/06/2025 14:06	Windows Batch File	1 KB
 Fotel.jar	19/06/2025 18:12	JAR File	27 KB

### 3.2. Fő ablak

Vetület Modell – Fotel

Fájl Nézet Súly

Beállítások

Szélesség [1]

Magasság [2]

Mélység [3]

Alapszín [4]

Párna szín [5]

Láb szín [6]

Díszpárna szín [7]

Fókusz [8]

Előnézet

Oldalnézet

Felülnézet

Méret

Szélesség: 1.44 m

Magasság: 1.2 m

Mélység: 0.8 m

Valós Felület: 7.772 m<sup>2</sup>

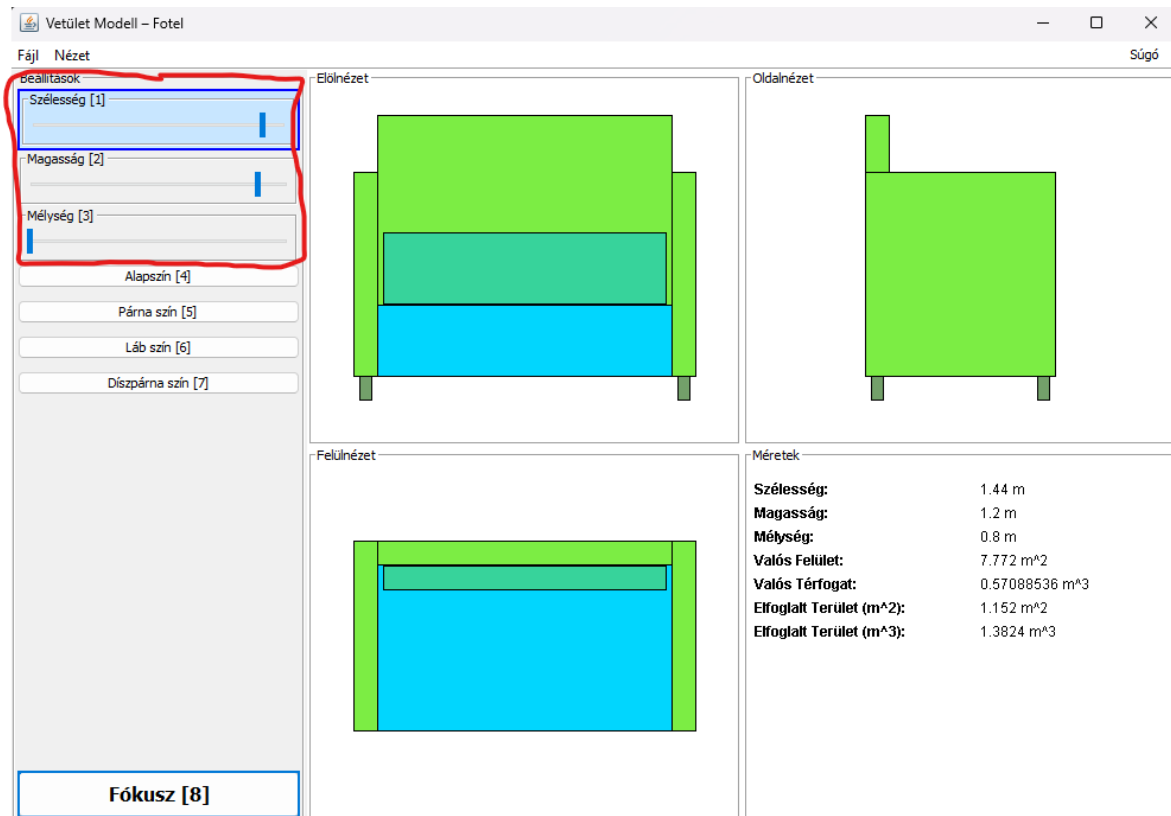
Valós Térfogat: 0.57088536 m<sup>3</sup>

Elfoglalt Terület (m<sup>2</sup>): 1.152 m<sup>2</sup>

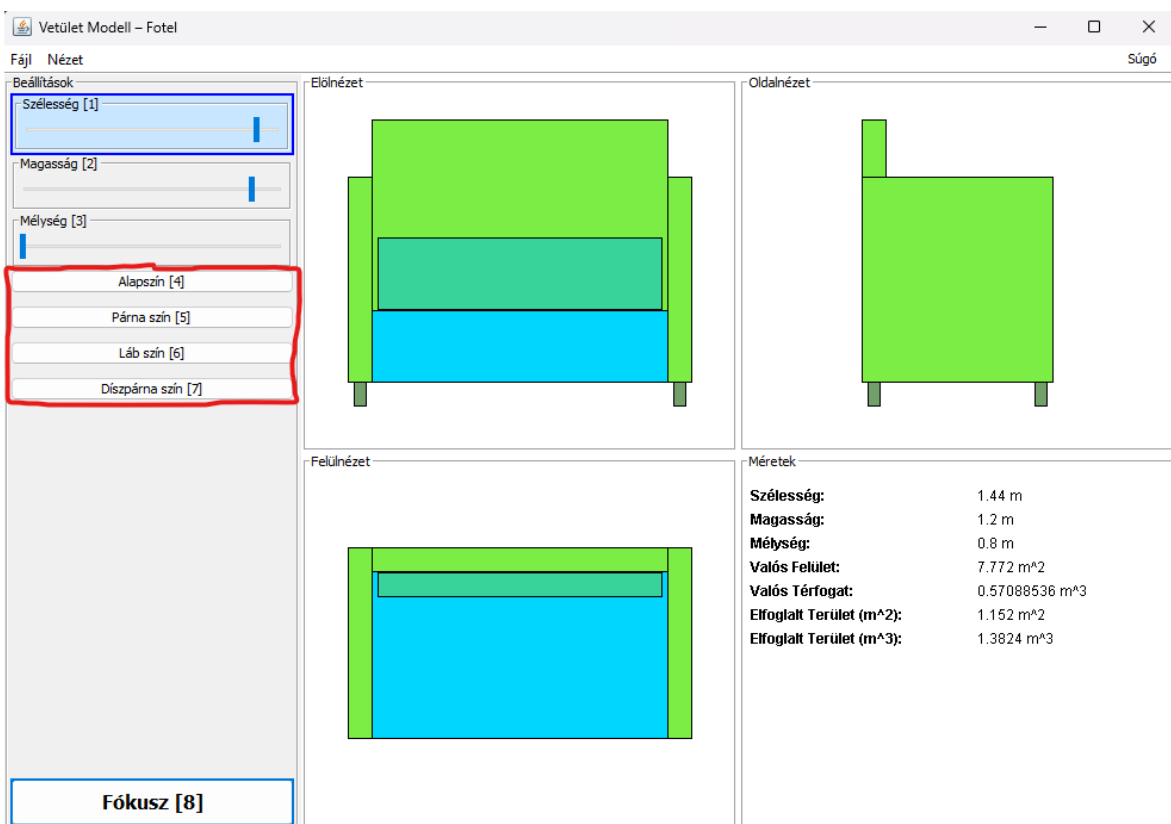
Elfoglalt Terület (m<sup>3</sup>): 1.3824 m<sup>3</sup>



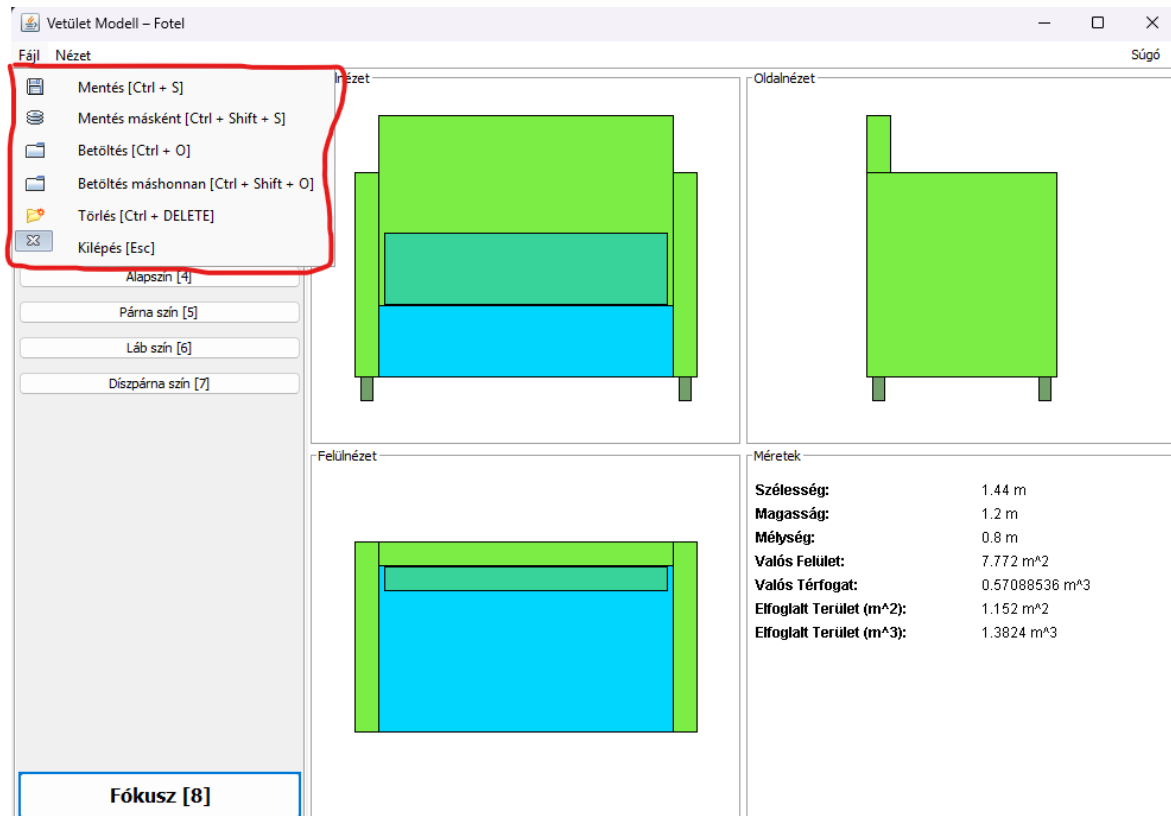
### 3.3.1. Méretek állítása



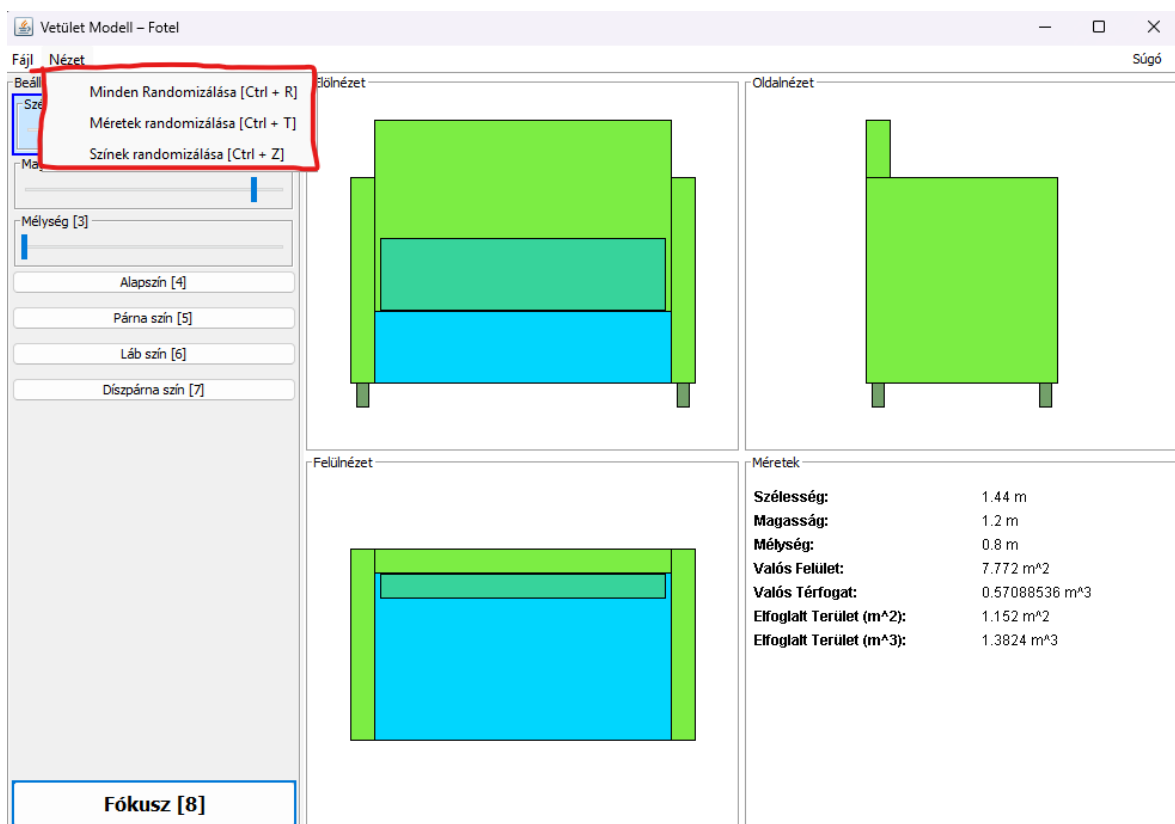
### 3.3.2. Színek állítása



### 3.3.3. Fájlműveletek



### 3.3.4. Randomizálás



### 3.3.5. Súgó

Súgó

Fotel Súgó

Névjegy

Készítette:  
Balics Attila Ádám - Z58T3N  
Balogh Levente HOAFBT

Használati útmutató

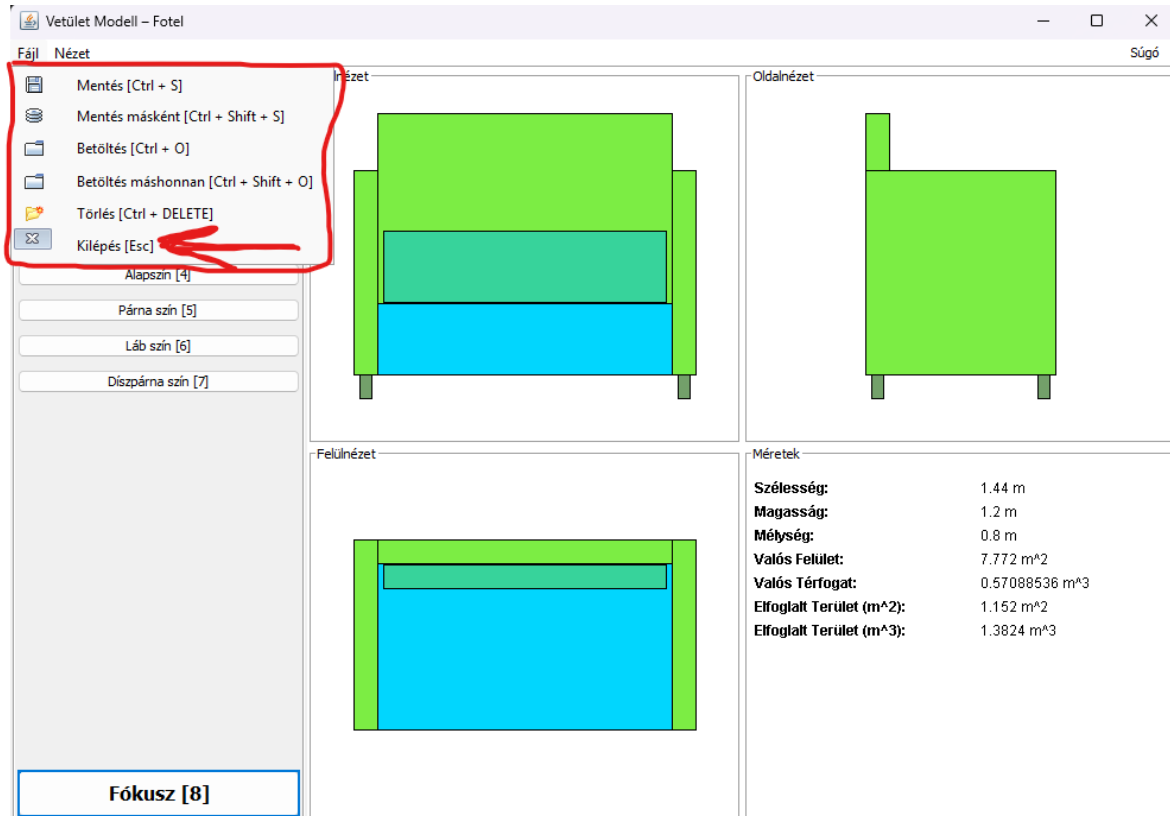
- [1] Szélesség állítása a Fel-Le, Jobb-Bal nyilakkal, **Shift**-tel nagyobb lépés.
- [2] Magasság állítása a Fel-Le, Jobb-Bal nyilakkal, **Shift**-tel nagyobb lépés.
- [3] Mélység állítása a Fel-Le, Jobb-Bal nyilakkal, **Shift**-tel nagyobb lépés.
- [4] Alapszín állítása.
- [5] Párna szín állítása.
- [6] Láb szín állítása.
- [7] Díszpárna szín állítása.
- [8] Nézetek fókuszba helyezése
  - [1] Előlnézet fókuszba helyezése
  - [2] Oldalnézet fókuszba helyezése
  - [3] Felülnézet fókuszba helyezése
  - [0] Vissza szerkesztés módba

Gyorsbillentyűk

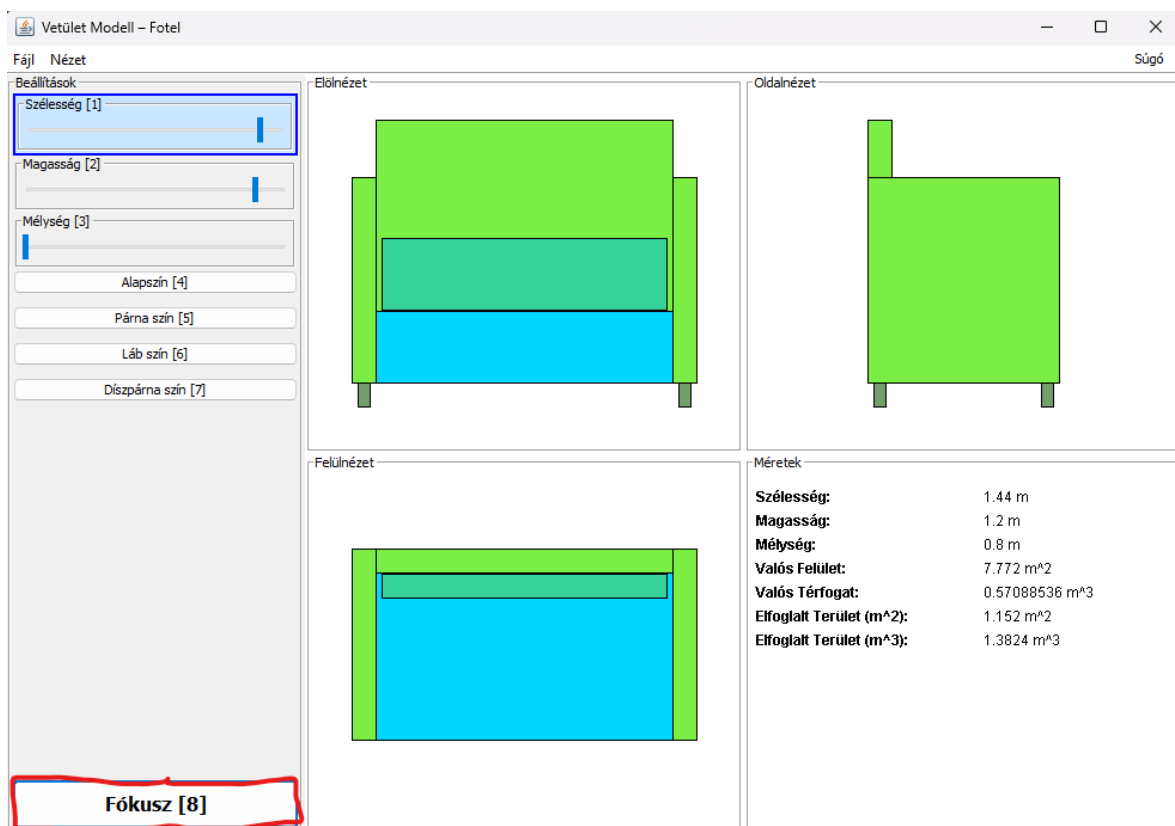
- Ctrl + S Fotel mentése helyileg
- Ctrl + Shift + S Fotel mentése tetszőleges helyre
- Ctrl + O Fotel betöltése
- Ctrl + Shift + O Fotel betöltése tetszőleges helyről
- Ctrl + Del Fotel törlése
- Ctrl + Z Fotel randomizálása szín szerint
- Ctrl + T Fotel randomizálása méret szerint
- Ctrl + R Fotel randomizálása minden szerint
- Ctrl + H Súgó
- Alt + F Fájl menü megnyitása
- Alt + V Nézet menü megnyitása

© 2025 Fotel

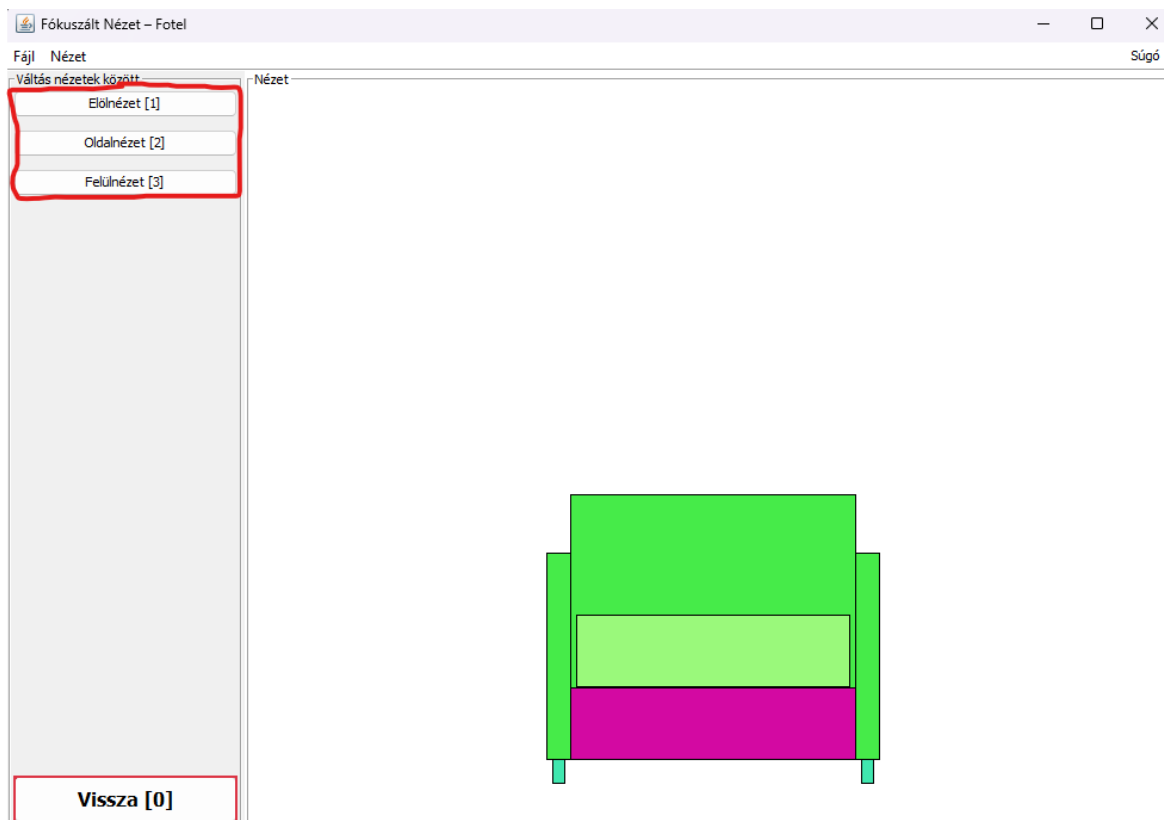
### 3.3.6. Kilépés



### 3.3.7. Fókusz mód



### 3.3.8. Fókusz nézet választás



# Irodalomjegyzék

- [1] Java SE Documentation. <https://docs.oracle.com/javase/8/docs/>
- [2] Az OOP féléves feladat készítési szabályzata – SZE 2025