

Programozás alapjai 3.

Házi feladat

VÁRADI RICHÁRD TAMÁS

XA5OZH

A FELADAT ISMERTETÉSE

A feladat, amit választottam ebben a félévben az egy szövegszerkesztő alkalmazás, amellyel kódokat lehet szerkeszteni.

A programnak lesz egy olyan grafikus interfésze, amelyet négy féle módon lehet megváltoztatni:

- Metal
- CDE/Motif
- Nimbus
- GTK+

Ezen felül még lehet változtatni a kód színezésének a módját is:

- Eclipse
- Idea
- Default-alt
- Monokai
- Default
- Visual Studio
- Dark

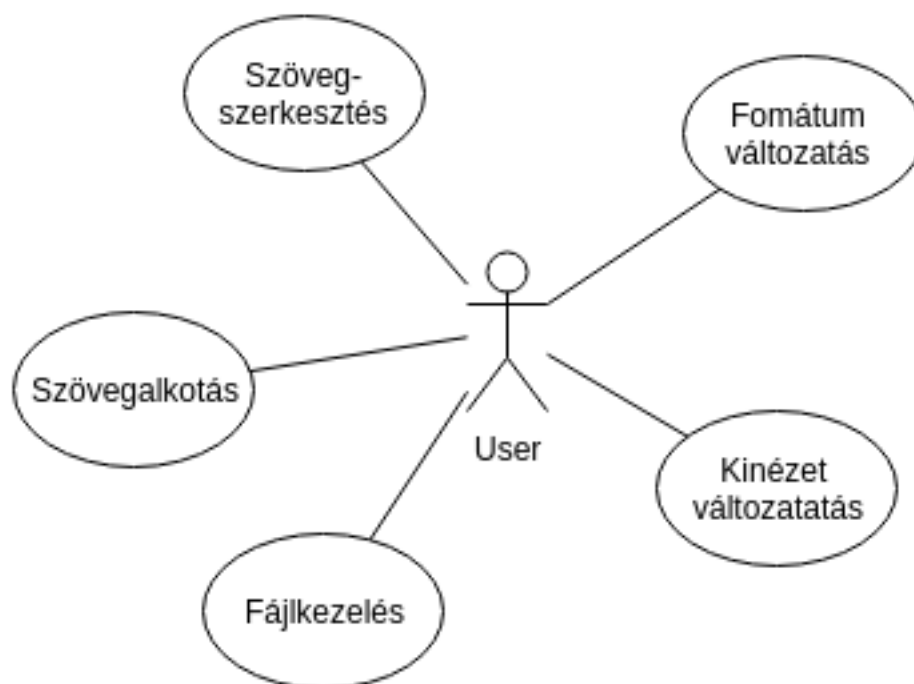
Lehet emellett új fájlt létrehozni, menteni, elmenteni máshogy. Aztán vannak még ilyen szerkesztéshez használatos funkciói is. Ezek rendre a megszokott műveletek: visszavonás, kivágás, másolás, beillesztés, törlés, keresés, keresés következő, csere, kijelöl mindent és még az akutális dátumot is betudjuk szűrni. Aztán tudunk még formátumot változtatni, vagyis lehet tördelni a kódot és lehet állítani a betűk méretét és stílusát.

Maga a szerkesztő felület támogatja a sorok számozását és még az összecsukást is, ami annyit tesz, hogy ha nagyon sok hasonló kezdetű sor szerepel egymás után, akkor összelehet őket csukni, hogy ne zavarjanak és ilyenkor eltűnnek, de egy kis ikon megjelenik az első sora mellett, amellyel újra ki lehet bontani. Ugyan ez igaz a függvényekre vagy osztályokra is. Ezek mellett még van autómátikus kiegészítés is. A szerkesztő az alábbi nyelveknek a szintaxisát tudja kezelni:

- | | | |
|----------------|--------------|--------------|
| • actionscript | • html | • php |
| • asm | • java | • properties |
| • bbcode | • javascript | • python |
| • c | • json | • ruby |
| • clojure | • jsp | • sas |
| • cpp | • latex | • scala |
| • cs | • lisp | • sql |
| • css | • lua | • tcl |
| • delphi | • makefile | • unix |
| • dtd | • mxml | • vb |
| • fortran | • nsis | • bat |
| • groovy | • perl | • xml |

USE-CASE

DIAGRAM



Leírások

Cím	Fájlkezelés
Leírás	Ezzel tud a felhasználó fájlokat kezelni.
Forgatókönyv	1. A menübáron megnyomva a file pontot lenyílik egy fül, amellyel új fájlt tud létrehozni.
Alternatív forgatókönyv	1.A.1 A felhasználó a New pontot kiválasztva megtud nyitni egy új fájlt.
Alternatív forgatókönyv	1.A.2 Ha éppen dolgozott valamin a felhasználó és azt nem mentette el, akkor felugrik egy ablak, hogy biztos-e benne.
Alternatív forgatókönyv	1.B.1 Kitudja választani még az Open... pontot is és akkor feljön egy ablak, amin keresztül kitudja választani a megnyitandó fájlt.
Alternatív forgatókönyv	1.B.2 Ha éppen dolgozott valamin a felhasználó és azt nem mentette el, akkor felugrik egy ablak, hogy biztos-e benne.
Alternatív forgatókönyv	1.B.3 Ha meggondolja magát, akkor be is tudja zárni az ablakot.
Alternatív forgatókönyv	1.C.1 Eltudja menteni, az aktuális munka fájlt a Save ponttal.
Alternatív forgatókönyv	1.C.2 Ha még ez előtt nem volt még egyszer sem mentve a fájl, akkor megkérdezi, hogy hova és milyen néven szeretnénk menteni.

Alternatív forgatókönyv	1.D.1 Ha a Save as... pontot választja, akkor feljön egy olyan ablak, amin keresztül máshova vagy más kiterjesztéssel tud fájlokat menteni.
Alternatív forgatókönyv	1.E.1 Ha az Exit pontot kiválasztja, akkor bezáródik az alkalmazás.

Cím	Szövegalkotás
Leírás	A felhasználónak lesz egy tere, amiben tud szöveg alkotni
Forgatókönyv	1. A felhasználó belekattint a szövegdobozba és elkezd írni
Alternatív forgatókönyv	1.A.1 Ha írásvédett a fájl, akkor nem tud beleírni a felhasználó.

Cím	Szövegszerkesztés
Leírás	Lehet a szövegben apró módosításokat véghez vinni.
Forgatókönyv	Lehet visszavonni, szöveg be- és kivinni.
Alternatív forgatókönyv	1.A.1 Vissza tud vonni változtatásokat
Alternatív forgatókönyv	1.B.1 Ki tudja vágni a kijelölt szöveget.
Alternatív forgatókönyv	1.C.1 Tud másolni.
Alternatív forgatókönyv	1.D.1 Tud beilleszteni.
Alternatív forgatókönyv	1.E.1 Tud keresni a szövegben.
Alternatív forgatókönyv	1.E.2 Tudja lépteni a keresést.
Alternatív forgatókönyv	1.F.1 Tud cserélni a szövegben.
Alternatív forgatókönyv	1.G.1 Ki lehet jelölni az egész szöveget.
Alternatív forgatókönyv	1.H.1 Betudja szűrni az aktuális dátumot.

Cím	Formátum változtatás
Leírás	A szöveg formátumát tudja változtatni.
Forgatókönyv	Megváltoztatja a szöveget.
Alternatív forgatókönyv	1.A.1 Tudja tördelni a szöveget.
Alternatív forgatókönyv	1.B.1 Tudja változtatni a betűket.
Alternatív forgatókönyv	1.B.2 Lehet a betűkészletet változtatni.
Alternatív forgatókönyv	1.B.3 Lehet a betűstílust változtatni.
Alternatív forgatókönyv	1.B.4 Lehet a méretet változtatni.

Cím	Kinézet változtatás
Leírás	A kinézetét a programnak meglehet változtatni.
Forgatókönyv	Tud választani, hogy a szövegdoboz kinézetén vagy a program kinézetén szeretne változtatni.
Alternatív forgatókönyv	textbf1.A.1 Tudja változtatni a program kinézetét a feladat leírásában található témák között.
Alternatív forgatókönyv	1.A.2 Tudja változtatni a szövegdoboz kinézetének szintaxisát a feladat leírásában található témák között.

VÁRADI RICHÁRD TAMÁS, XA5OZH



Osztályok és interfészek

Interfészek	
MenuConstants	Ez lényegében csak egy tároló osztály, amely a menü elemeinek a nevét tartalmazza.

Osztályok	
FindDialog	Ezzel lehet létrehozni a keresési ablakot és ebben hajtódik végre maga a keresés is.
FontChooser	Ezzel lehet létrehozni a betűtípus választási ablakot és ebben hajtódik végre maga a logika is.
LookAndFeelMenu	Ezzel lehet megváltoztatni a program kinézetének témáját.
LookAndFeelMenuListener	Ez figyel, hogy milyen témát választottunk ki a program megjelenésének.
Norns	Ez hozza létre magát a GUI-t és ebben vannak egyéb menü belüli logikai megvalósítások.
FileOperation	Ez hozza létre azokat az ablakokat, amin keresztül lehet fájlt menteni vagy azokat megnyitni. A logika is itt tárolódik.
ChangeSyntaxStyleListener	Ez nézi, hogy milyen szintaxis színezést választottunk ki.
ChangeSyntaxStyle	Ezzel lehet megváltoztatni a szövegmezőbe írt kód színezését.
NornsTest	Ebben vannak a teszt esetek megírva.
TestRunner	Ez futattja a teszteket.

Interfész MenuConstants

```
public interface MenuConstants
```

Ezek azok a konstansok, amelyeket felhasználunk, mikor megalkotjuk a menüsávot a **Norns** osztályban. Vagyis ezek a változók lesznek a fő menüpontok és az összes többi pontnak a neve is.

Változók	
final String	fileText
final String	editText
final String	formatText
final String	viewText
final String	helpText
final String	fileNew
final String	fileOpen
final String	fileSave
final String	fileSaveAs
final String	fileExit
final String	editUndo
final String	editCut
final String	editCopy
final String	editPaste
final String	editDelete

final String	editFind
final String	editFindNext
final String	editReplace
final String	editSelectAll
final String	editTimeDate
final String	formatWordWrap
final String	formatFont
final String	viewStatusBar
final String	helpHelpTopic
final String	helpAboutNorns
final String	aboutText

VÁLTOZÓK ADATAI

Ezeket nagyon beszédes nevei miatt nem fejtem ki jobban, kivéve ezt.

aboutText

```
public final String aboutText
```

Ez egy kis html üzenetet fész fel, amiben pár szó található a programról és az e-mail címem.

Osztály FindDialog

```
public class FindDialog extends JPanel implements ActionListener
```

Ebben az osztályban valósul meg a keresési és a kicseréli ablak is, mivel ez a kettő ugyanazon az ablakon működik csak nem mindig látható minden gomb rajta, ezért nem szedtem őket szét kettő különböző osztályba.

Változók	
static final long	serialVersionUID
RSyntaxTextArea	textArea
int	lastIndex
JLabel	replaceLabel
TextField	findWhat
JTextField	replaceWith
JCheckBox	matchCase
JRadioButton	up
JRadioButton	down
JButton	findNextButton
JButton	replaceButton
JButton	replaceAllButton
JButton	cancelButton
JPanel	direction
JPanel	buttonPanel
JPanel	findButtonPanel
JPanel	replaceButtonPanel
JDialog	dialog

A függvények és leírásuk:

```
public FindDialog(RSyntaxTextArea textArea)
```

Ez egy konstruktor, amivel létrehozuk azt a dialógust, amely a keresésekhez szolgál. Csak egyetlen ablakot csinál meg és azon annak megfelelően jelenít meg elemeket, hogy éppen mely menüpont van nyitva.

```
public void enableDisableButtons()
```

Eldönti azt, hogy a gombok közül mit lehet nyomkodni. Ha nincs még semmilyen szöveg írva, akkor mindent lezár.

```
public void actionPerformed(ActionEvent ev)
```

Figyeli azokat a gombokat, amik megjelennek és a megfelelő függvényeket hívogatja meg a hatására.

```
public int findNext()
```

Megkeresi a következő olyan szót, ami megfelel annak, amit talált és annak az indexét visszaadja.

```
public void findNextWithSelection()
```

Megkeresi a következőt, aztán kijelöli.

```
public void replaceNext()
```

Megkeresi a következő olyan szót, ami illik a keresetre, majd azt kicseréli.

```
public int replaceAllNext()
```

Ugyan azt csinálja, mint az előző függvény csak ez végig megy a teljes szövegen, majd az össze keresésnek megfelelő szót kicseréli arra, amire szeretnénk.

```
public void showDialog(Component parent, boolean isFind)
```

Ezzel lehet létrehozni a dialógus ablakot.

Osztály FontChooser

```
public class FontChooser extends JPanel
```

Itt valósul meg az ablak, amelyben kilehet választani, hogy milyen betűstílust szeretnénk alkalmazni, milyen mérettel és milyen formázással.

Változók	
static final long	serialVersionUID
Font	thisFont
JList	jFace
JList	jStyle

JList	jSize
JDialog	dialog
JButton	okButton
RSyntaxTextArea	textArea
boolean	ok

A függvények és leírásuk:

```
public FontChooser(Font withFont)
```

Ebben a konstruktorban kéri le, hogy az operációs rendszerünk milyen betűtípusokat képes támogatni és beállítja a stílust, formázást és méretet is. Aztán ezekből megalkotja a dialógus ablakot annak minden elemével együtt.

```
public Font createFont()
```

Ez állítja be, hogy választásaink alapján milyen betűket szeretnénk használni és annak megfelelő típussal tér vissza.

```
public boolean showDialog(Component parent, String title)
```

Ez jeleníti meg magát az ablakot.

Osztály LookAndFeelMenu

```
public class LookAndFeelMenu
```

Ez az osztály szolgálja azt a célt, hogy a felhasználó megtudja magának változtatni a GUI kinézetét.

A függvények és leírásuk:

```
public static void createLookAndFeelMenuItem(JMenu jmenu, Component cmp)
```

Lekérdezi, hogy milyen témák vannak, majd csinál egy menüpontot, amiben ezeket a stílusokat réteges formában elhelyezi, majd ezekre kattintva megváltozik a GUI kinéézete.

Osztály LookAndFeelMenuListener

```
class LookAndFeelMenuListener implements ActionListener
```

Ez figyeli a *LookAndFeelMenuListener* osztályt és ez változtatja meg a kinézetet.

Változók	
String	classname
Component	jf

A függvények és leírásuk:

```
public LookAndFeelMenuListener(String cln, Component jf)
```

Ez csak egy mezei konstruktor, ami beállítja a változóit.

```
public void actionPerformed(ActionEvent ev)
```

Figyeli, hogy milyen változások történtek és, aszerint cselekszik.

Osztály ChangeSyntaxStyle

```
public class ChangeSyntaxStyle
```

Ezzel lehet megváltoztani a szövegmező színezését.

Változók	
static String[][]	types

Ebben van tárolva azoknak a témáknak az elérési útvonala, amiket használ.

A függvények és leírásuk:

```
public static void createChangeSyntaxStyle(JMenu jmenu, RSyntaxTextArea rsta)
```

Betölti az összes lehetséges szintaxis színezést és csinál belőlük egy rétegzet menüőpontot. Aminél a megfelelő témát kiválasztva megváltozik.

Osztály ChangeSyntaxStyleListener

```
class ChangeSyntaxStyleListener implements ActionListener
```

Figyeli a *ChangeSyntaxStyle* osztályt és ez változtat.

Változók	
String	type
RSyntaxTextArea	rs

A függvények és leírásuk:

```
public ChangeSyntaxStyleListener(String th, RSyntaxTextArea rsta)
```

Ez csak egy mezei konstruktor, ami beállítja a változóit.

```
public void actionPerformed(ActionEvent ev)
```

Figyeli, hogy milyen változások történtek és, aszerint cselekszik.

Osztály FileOperation

```
public class FileOperation
```

Ez az osztály valósítja meg a fájlok megnyitását, mentését és magát a dialógus ablakot.

Változók	
Norns	nor
boolean	saved
boolean	newFileFlag
String	fileName
String	applicationTitle
File	fileRef
JFileChooser	chooser

A függvények és leírásuk:

```
public boolean isSave()
```

Visszatér azzal, az információval, hogy elvan-e mentve a dokumentum.

```
public void setSave(boolean saved)
```

Ha majd rányomunk a mentés gombra, akkor ez állítja be, hogy mentettünk.

```
public String getFileName()
```

Lekérdezi a megnyitott file-nak a nevét.

```
public void setFileName(String fileName)
```

Betudjuk állítani, hogy a file-nak mi legyen a neve.

```
public FileOperation(Norns nor)
```

Konstruktor, ami program megnyitásakor létrehoz egy üres dokumentumot.

```
public boolean saveFile(File temp)
```

Fájlok elmentésére szolgál.

```
public boolean saveThisFile()
```

Ezzel lehet menteni, mikor egy létező fájlba írtunk bele.

```
public boolean saveAsFile()
```

Ezzel, akkor lehet menteni, mikor egy másik formátumban vagy más helyre akarunk menteni. Szóval "Mentés másként..."

```
public boolean openFile(File temp)
```

Megnyit egy paraméterül kapott file-t és, annak a szövegét hozzáfűzi a szövegmezőre.

```
public void openFile()
```

Itt állítja be a FileChooser dialógusnak a megjelenítését, hogy mikor és milyen esetekben mit kell felhozni.

```
public void updateStatus(File temp, boolean saved)
```

Visszajelzést add arról, hogy sikerült-e menteni vagy megnyitni egy fájlt. Olyan formában, hogy beállítja a megfelelő változókat.

```
public boolean confirmSave()
```

Létrehoz egy dialógust, hogy biztos elszeretnénk-e menteni az adott dokumentum változtatásait.

```
public void newFile()
```

Üres dokumentum létrehozásakor beállítja a változókat olyan értékre, ami megfelel egy új fájlnak.

Osztály Norns

```
public class Norns implements ActionListener, MenuConstants
```

Ez lényegében egy controller osztály, amelyben összefutnak a különböző osztályok metódusai.

Változók	
JFrame	frame
RSyntaxTextArea	textArea
JLabel	statusBar
JLabel	textIndicator
JPanel	panel
String	fileName
boolean	saved
String	applicationName
String	searchString
String	replaceString
int	lastSearchIndex
FileOperation	fileHandler
FontChooser	fontDialog
FindDialog	findReplaceDialog
JMenuItem	cutItem
JMenuItem	copyItem
JMenuItem	deleteItem
JMenuItem	findItem
JMenuItem	findNextItem
JMenuItem	replaceItem
JMenuItem	selectAllItem

A függvények és leírásuk:

```
public Norns()
```

Ebben a konstruktorban jön létre maga a fő ablak és annak a beállításai.

```
public CompletionProvider createCompletionProvider()
```

Csinál egy olyan *DefaultCompletionProvider* objektumot, amely egy ilyen bufferként működik és a megadott fájlból kiolvassa a megfelelő kulcsszavakat tárolja. Ebből tudunk majd kiegészítéskor válogatni.

```
public void setSyntax(String ext)
```

Ez egy olyan függvény, ami paraméterül kapja a fájl típusát és asszerint állítja be a szöveg színezését.

```
public void actionPerformed(ActionEvent ev)
```

Ez a fő vezérő, ami figyel, hogy milyen esetekre, hogyan kellene válaszolnia.

```
public JMenuItem createMenuItem(String s, int key, JMenu toMenu,  
    ActionListener al)
```

Menüelemek létrehozására szolgál a megadott menüben, aminek nincs gyorsbillentyűje.

```
public JMenuItem createMenuItem(String s, int key, JMenu toMenu, int aclKey,  
    ActionListener al)
```

Itt is menüelemeket lehet létrehozni, de itt már állítunk be nekik gyorsbillentyűt is.

```
public JCheckBoxMenuItem createCheckBoxMenuItem(String s, int key, JMenu  
    toMenu, ActionListener al)
```

Ilyen kipipálós menüelemek létrehozásához tesz hozzá. Ezek, akkor jók, mikor több fajta kinézetből választahatunk és ott jelölni kell, hogy mi az aktuális.

```
public JMenu createMenu(String s, int key, JMenuBar toMenuBar)
```

Ezek hozzák létre, a fő menüpontokat a menüsávon.

```
public void createMenuBar(JFrame frame)
```

Ezzel megalkotjuka teljes menüsávot, amiben felhasználjuka *MenuConstants*- beli változókat is, mint menü nevek.

```
public static void main(String[] args)
```

Ebben hívjuk meg a *Norns* konstruktorát, amivel elindul a programunk.

Osztály NornsTest

```
public class NornsTest
```

Ebben az osztályban írtam meg a 10 darab tesztet.

Változók	
Norns	nor

A függvények és leírásuk:

```
public void setUp() throws Exception
```

A tesztek előtt beállítjuk mindent.

```
public void testCreateMenuBar()
```

Leteszteljük, hogy ténylegesen létrehozza-e a menüsávot.

```
public void testCreateCompletionProvider()
```

Megnézzük azt, hogy feltöltötte-e a kulcsszavakat.

```
public void testCreateMenu()
```

Tud-e menüt létrehozni.

```
public void testSetSyntax()
```

Beállítja-e a szintaxis színezését.

```
public void testIsSave()
```

Megnézi, hogy sikerült-e elmenteni.

```
public void testSetSave()
```

Elmentjük és megnézzük, hogy tényleg elmentődött-e.

```
public void testGetFileName()
```

Megnézzük, hogy tudjuk-e kérni a fájlok nevét.

```
public void testSetFileName()
```

Megnézzük, hogy tudjuk-e állítani a fájl nevét.

```
public void testConfirmSave()
```

Megnézzük, hogy a mentés megerősítő panel jól működik-e.

```
public void tesCreateFont()
```

Létretudunk-e hozni egy betűstílust.

Osztály TestRunner

```
public class TestRunner
```

Itt futnak le a tesztek, ha valamilyen hiba van, akkor kiírja egyébként csak annyit közöl velünk, hogy *true*, a minden teszt jól lefutott.

A függvények és leírásuk:

```
public static void main(String[] args)
```

Végig iterálja a teszteket, majd ha hibát talál jelzi.

Használati útmutató

Fordítás:

Én ennek a programnak a megírásához nem használtam semmilyen fejlesztőkörnyezetet csak egy szövegszerkesztőt és a terminált. Szóval, ha leszeretnéd fordítani a kódot, akkor az alábbi parancsot kell a terminálba beírnod, mikor benne vagy a forrás mappában.

```
javac -cp "?:./rsyntaxtextarea.jar:autocomplete.jar:junit.jar" -d . *.java
```

Futattás:

Ha szeretnéd ekindítani az alkalmazást, akkor az fenti feltételek mellett ezt a parancsot kell beírni a terminálba:

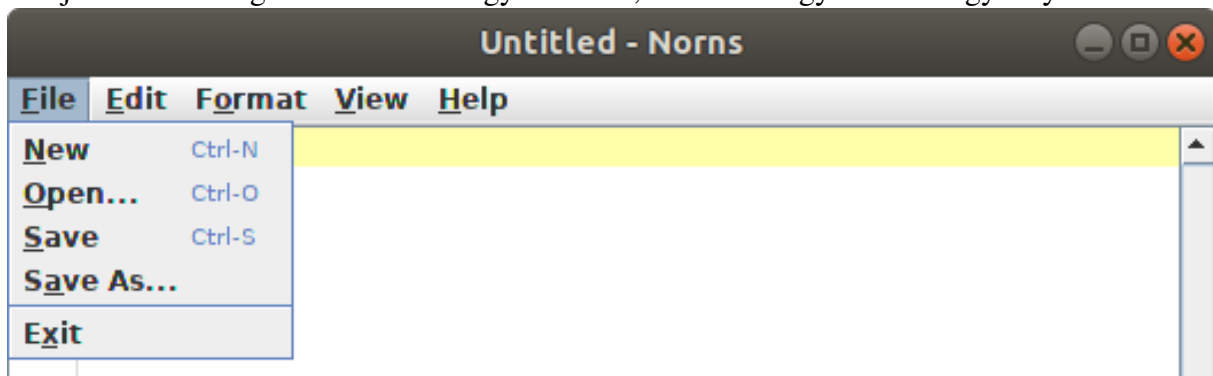
```
java -cp '?:./rsyntaxtextarea.jar:autocomplete.jar:junit.jar' nor.Norns
```

Ha pedig a tesztek eredményére vagy kíváncsi:

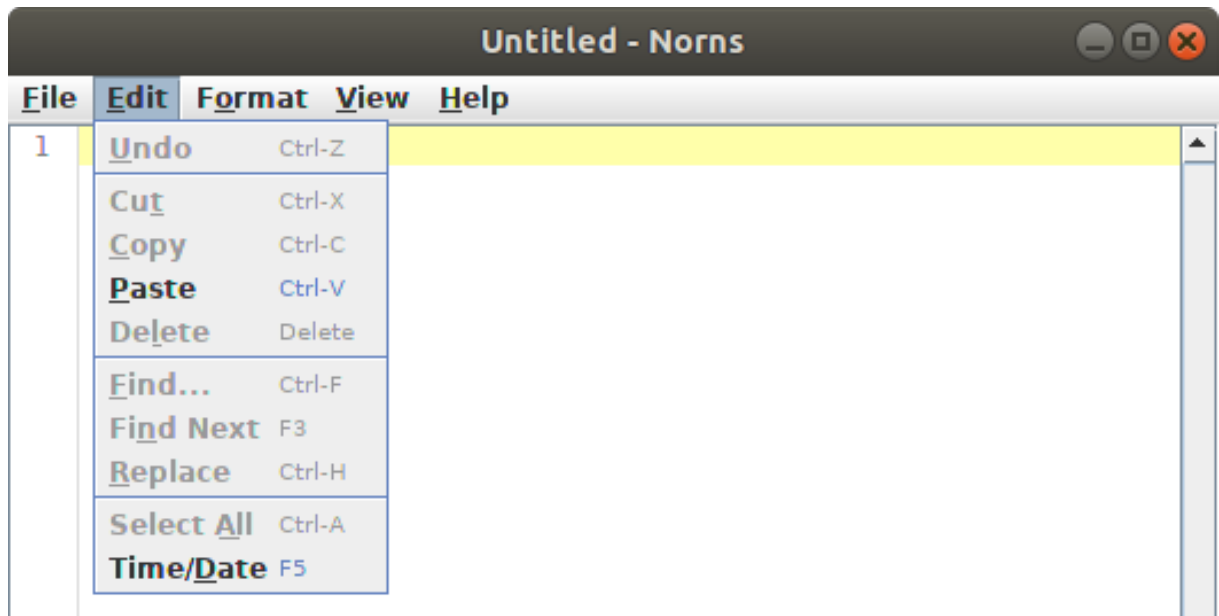
```
java -cp '?:./rsyntaxtextarea.jar:autocomplete.jar:junit.jar' nor.TestRunner
```

Használat:

Mikor egy felhasználó megnyitja ezt az alkalmazást, akkor alapjáraton egy üres szövegmezővel találja szemben magát. De ezen van egy menüsáv, aminek az egyes elemei így helyezkednek el:

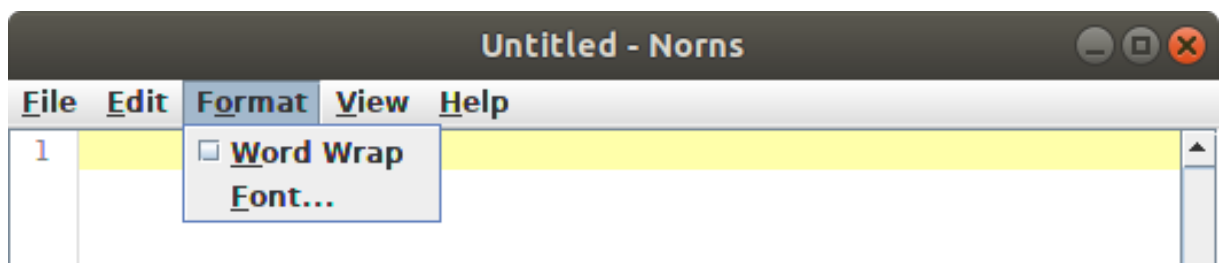


- **New:** Létre lehet hozni egy új dokumentumot.
- **Open...:** Meglehet nyitni egy új doksit.
- **Save:** Mikor csak simán menteni szeretnénk egy meglévő fájlra.
- **Save As...:** Mikor máshogyan akarunk elmenteni egy fájlt.
- **Exit:** Ha kisseretnénk lépni a programból.

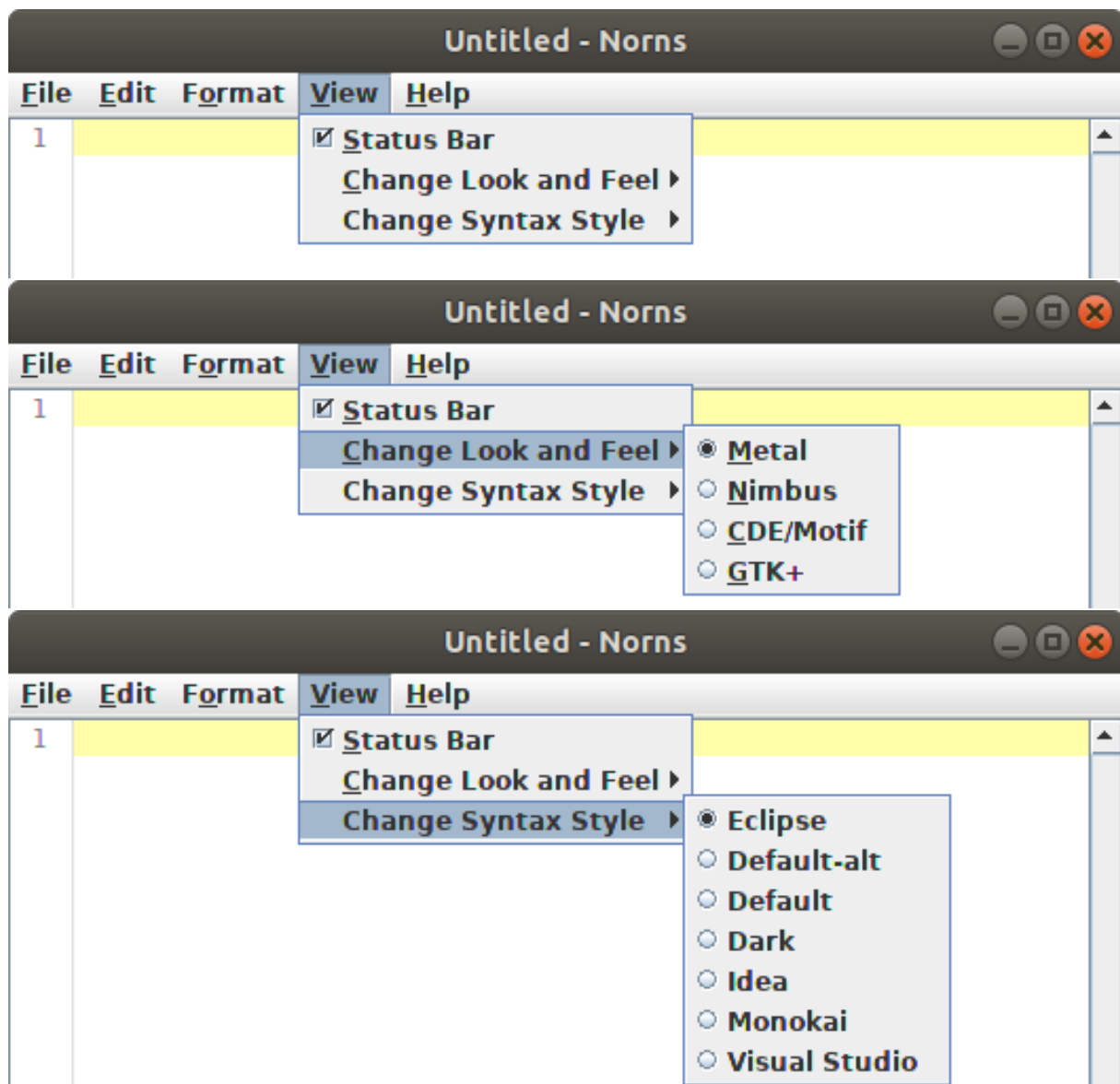


- **Undo:** Viszavonást lehet vele megvalósítani.
- **Cut:** Kitudunk vágni egy darabot a szövegből.
- **Copy:** Tudunk másolni szöveget.
- **Paste:** Betudunk illeszteni egy szöveget.
- **Delete:** Kitudjuk törölni a kijelölt részt.
- **Find...:** Megnyit egy keresési panelt, ahol tudunk keresni szöveget.
- **Find Next:** A következő egyező részt kijelöli.
- **Replace:** Kicseréli a következő olyan szöveget, ami egyezik.
- **Select All:** Kiválasztja az egész szöveget.
- **Time/date:** Beszúrja az aktuális dátumot ilyen formában.

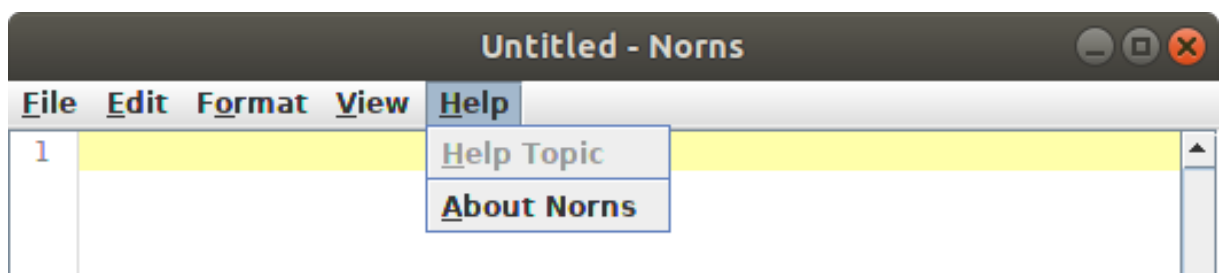
Sun Nov 25 09:24:51 CET 2018



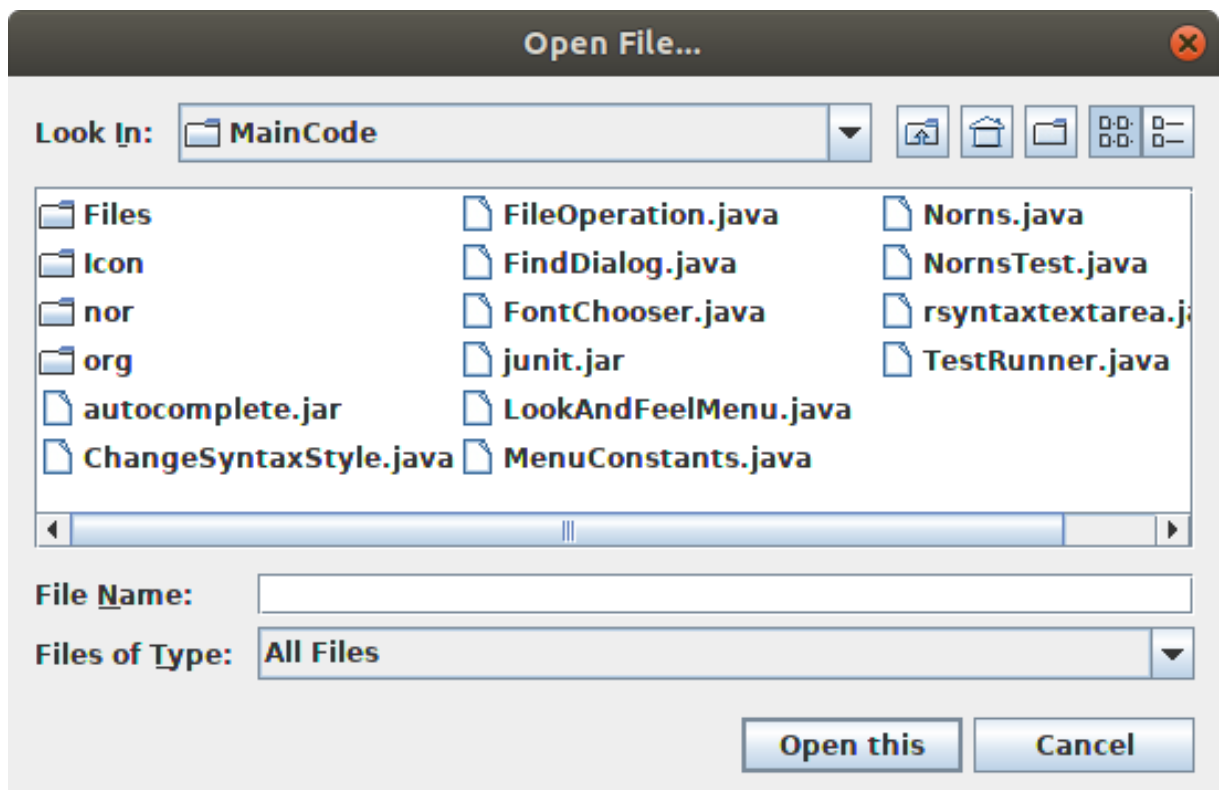
- **Word Wrap:** Megtöri a szöveget.
- **Font:** Előhozza a szöveg stílusának beállítását.



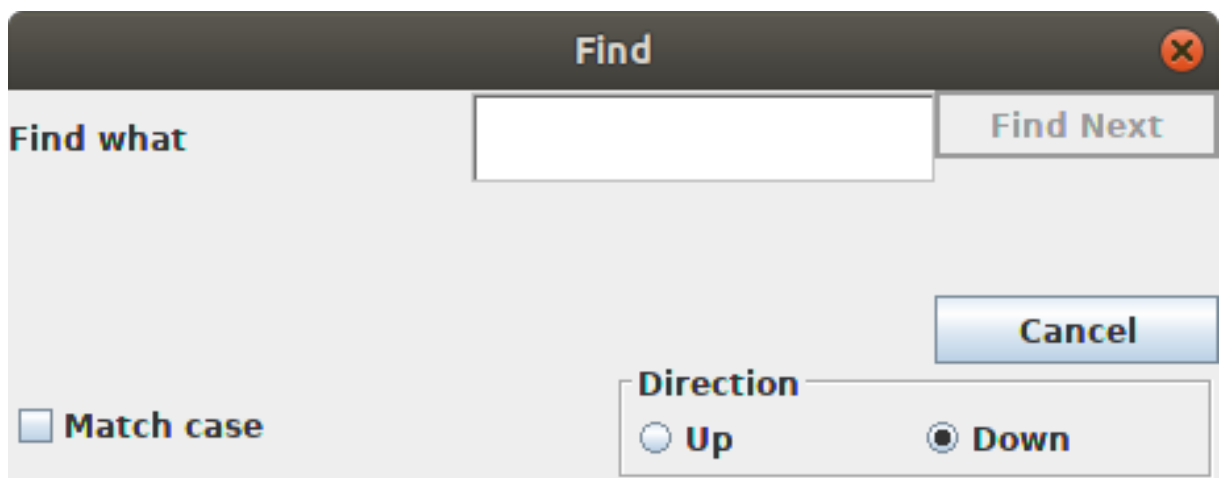
- **Status Bar:** Eltünteti vagy megjeleníti a Status Bar-t.
- **Change Look and Feel:** Kilehet választani a GUI kinézetét.
- **Change Syntax Style:** Meglehet változtatni a szintaxis színezést.



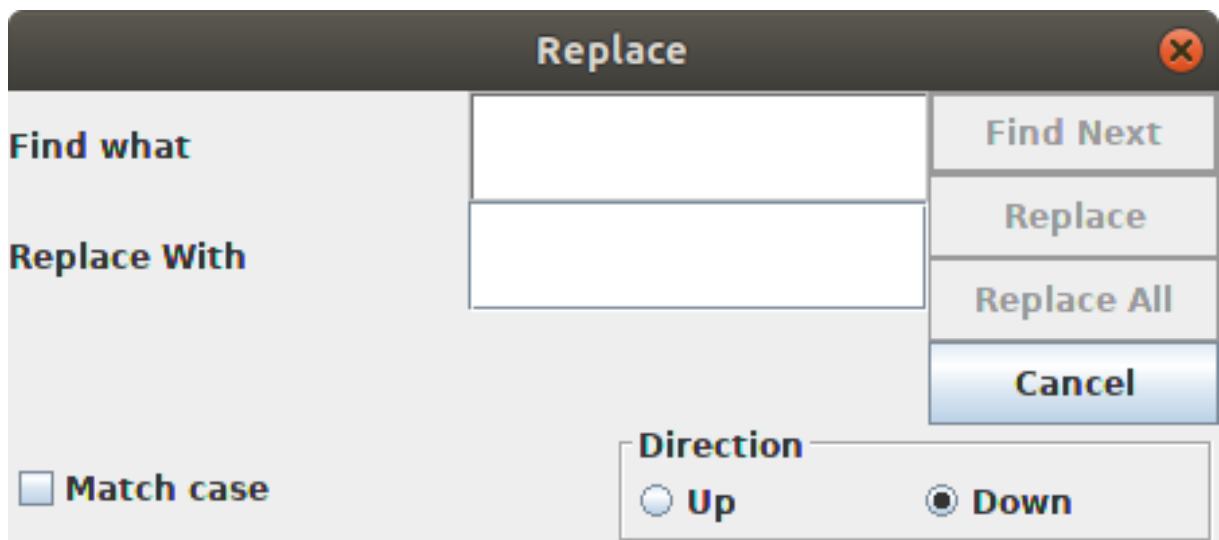
- **Help Topic:** Ha lenne weblap hozzá itt lehetne elérni.
- **About Norns:** Egy kis szöveget ír ki.



Ha megszeretnénk nyitni egy fájlt vagy elmenteni másképp, akkor egy ilyen ablakot fogunk kapni.

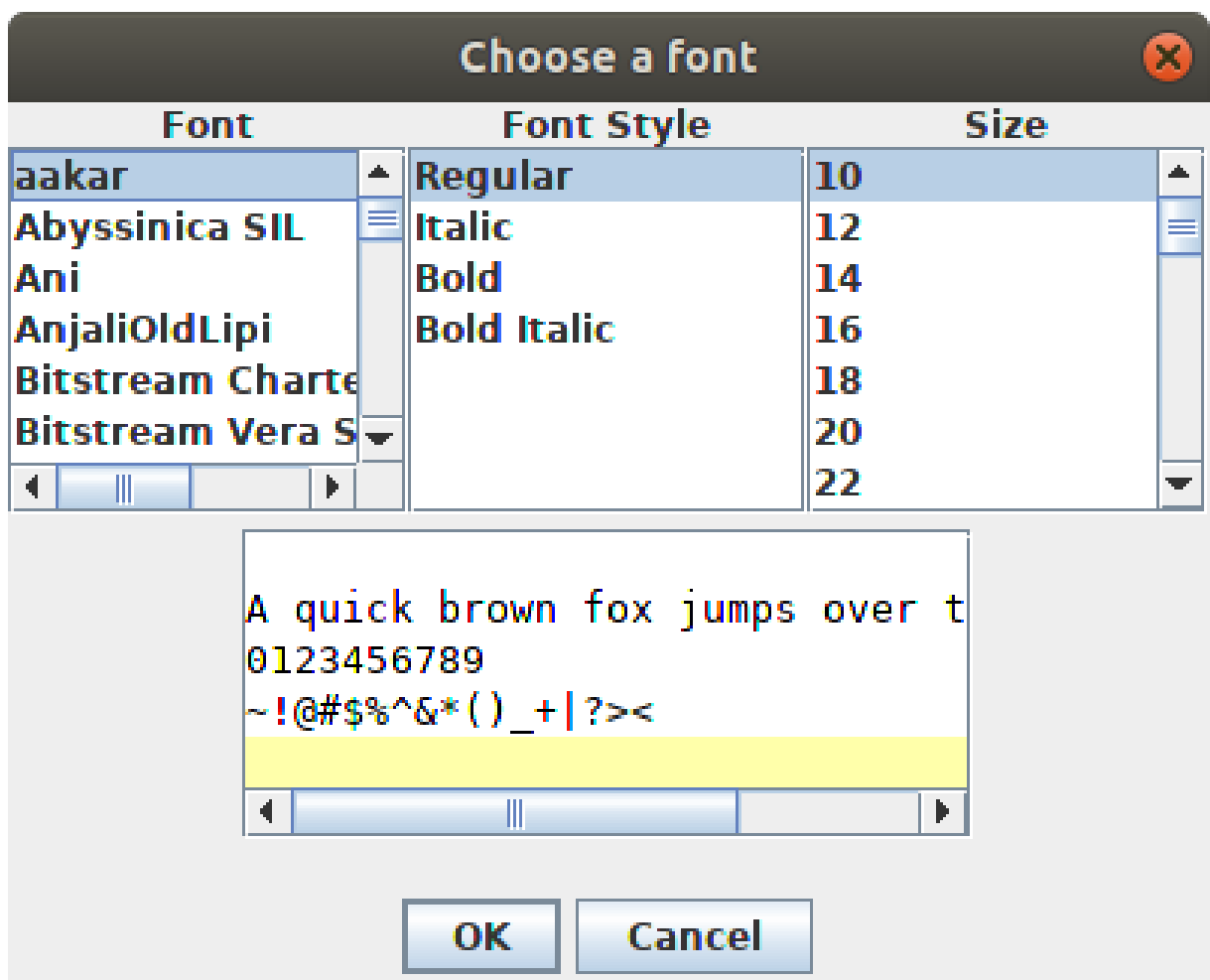


Mikor keresni szeretnénk egy szót, de nem akarjuk kicserélni csak megtalálni, akkor egy ilyen ablakot kapunk.



The 'Replace' dialog box has a title bar with a close button. It contains two input fields: 'Find what' and 'Replace With'. To the right of these fields are three buttons: 'Find Next', 'Replace', and 'Replace All'. Below these buttons is a 'Cancel' button. At the bottom left is a checkbox labeled 'Match case'. At the bottom right is a 'Direction' section with two radio buttons: 'Up' and 'Down', where 'Down' is selected.

Ha szeretnénk keresni, de azt ki is szeretnénk cserélni.

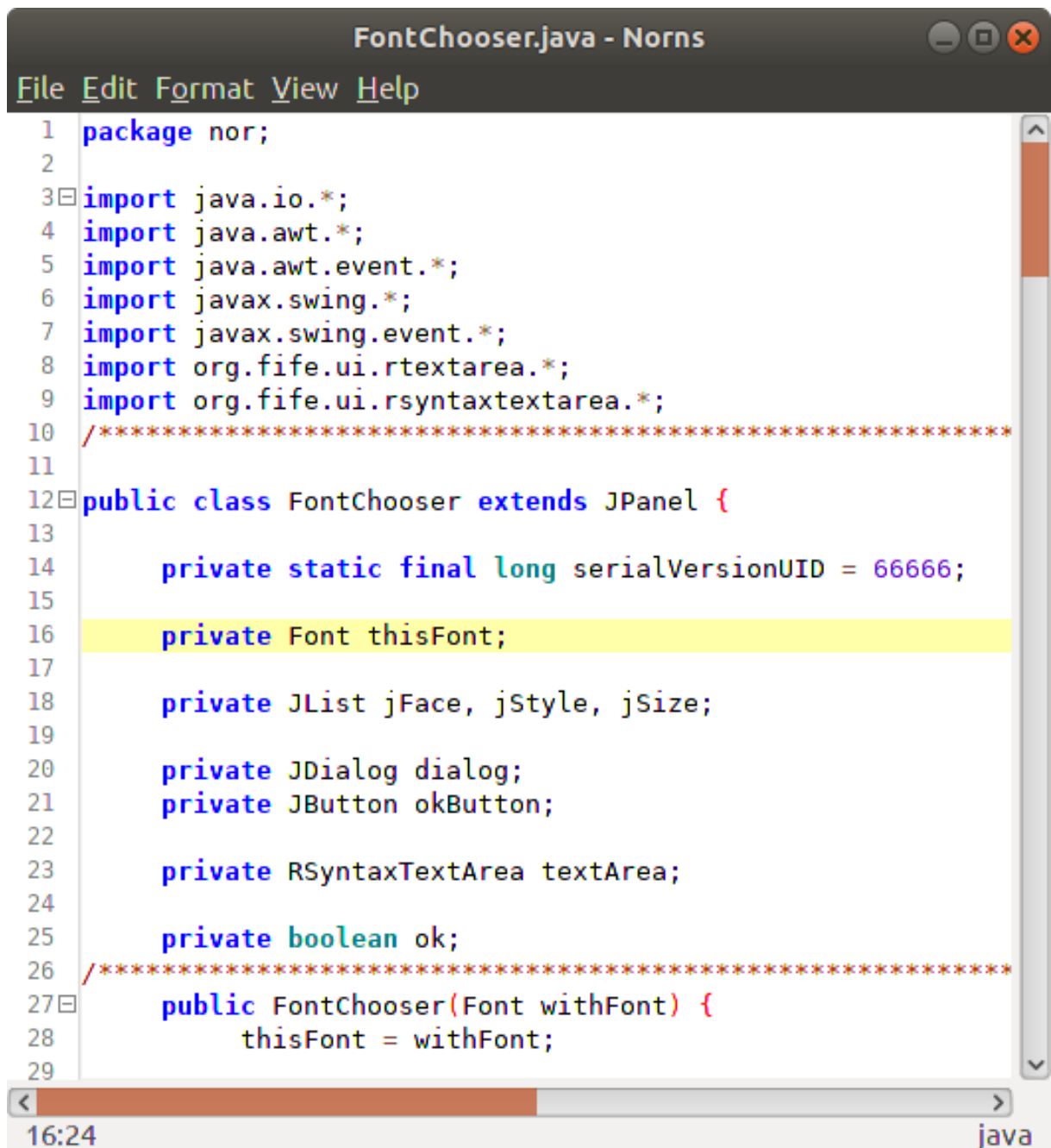


The 'Choose a font' dialog box has a title bar with a close button. It features three columns: 'Font', 'Font Style', and 'Size'. The 'Font' column lists various fonts like 'aakar', 'Abyssinica SIL', 'Ani', 'AnjaliOldLipi', 'Bitstream Charter', and 'Bitstream Vera S'. The 'Font Style' column lists 'Regular', 'Italic', 'Bold', and 'Bold Italic'. The 'Size' column lists sizes from 10 to 22. Below the columns is a preview window showing the text 'A quick brown fox jumps over t' followed by a list of characters: '0123456789', '~!@#\$%^&*()_+|?><'. At the bottom are 'OK' and 'Cancel' buttons.

Itt kitudjuk választani a szöveg:

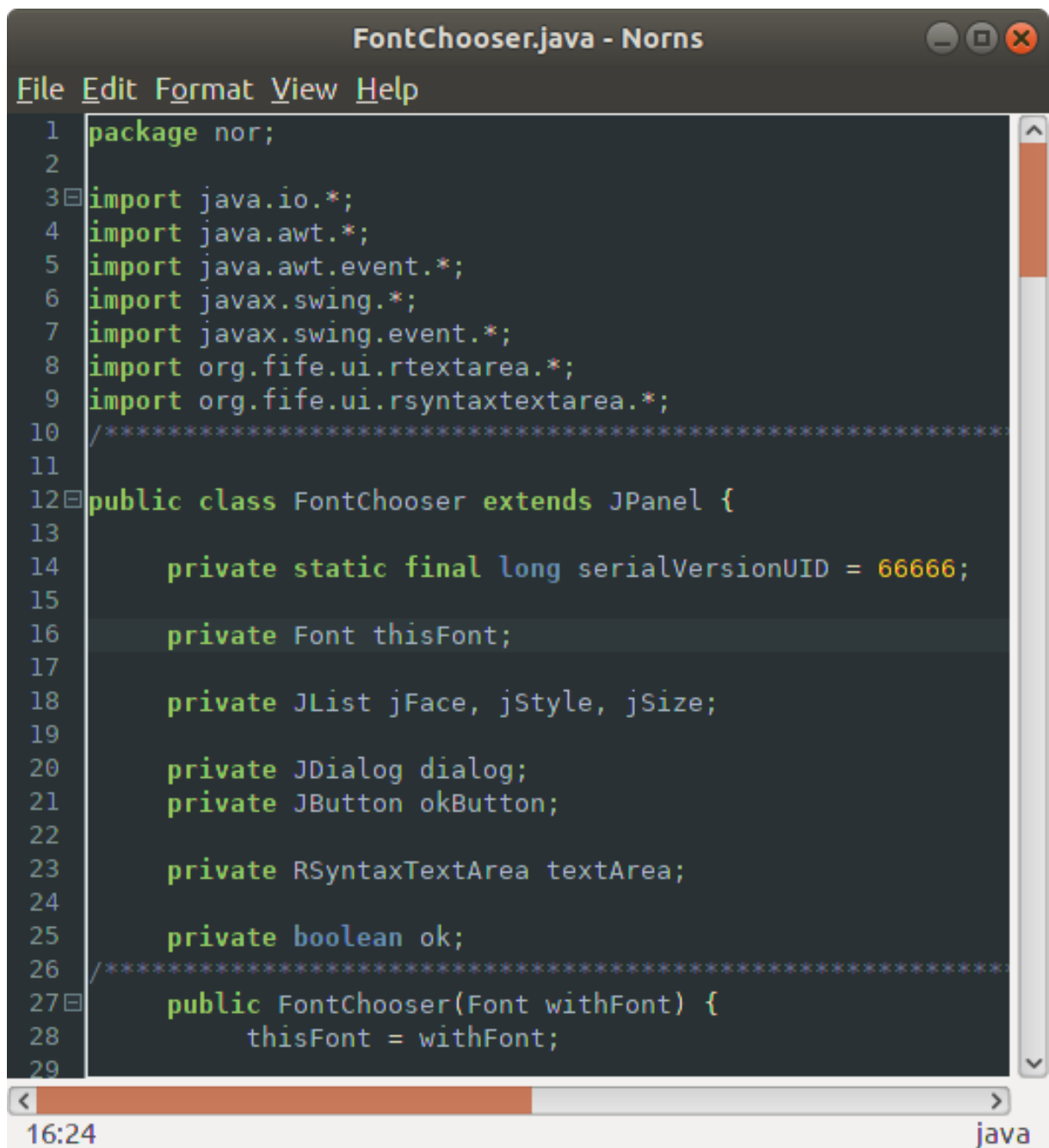
- Stílusát.
- Formázását.
- Méretét.

És még emellé van egy kis szövegünk is, amin látszik, hogy mit változtattunk.



```
FontChooser.java - Norns
File Edit Format View Help
1 package nor;
2
3 import java.io.*;
4 import java.awt.*;
5 import java.awt.event.*;
6 import javax.swing.*;
7 import javax.swing.event.*;
8 import org.fife.ui.rtextarea.*;
9 import org.fife.ui.rsyntaxtextarea.*;
10 /**
11
12 public class FontChooser extends JPanel {
13
14     private static final long serialVersionUID = 66666;
15
16     private Font thisFont;
17
18     private JList jFace, jStyle, jSize;
19
20     private JDialog dialog;
21     private JButton okButton;
22
23     private RSyntaxTextArea textArea;
24
25     private boolean ok;
26 /**
27 public FontChooser(Font withFont) {
28     thisFont = withFont;
29
16:24 java
```

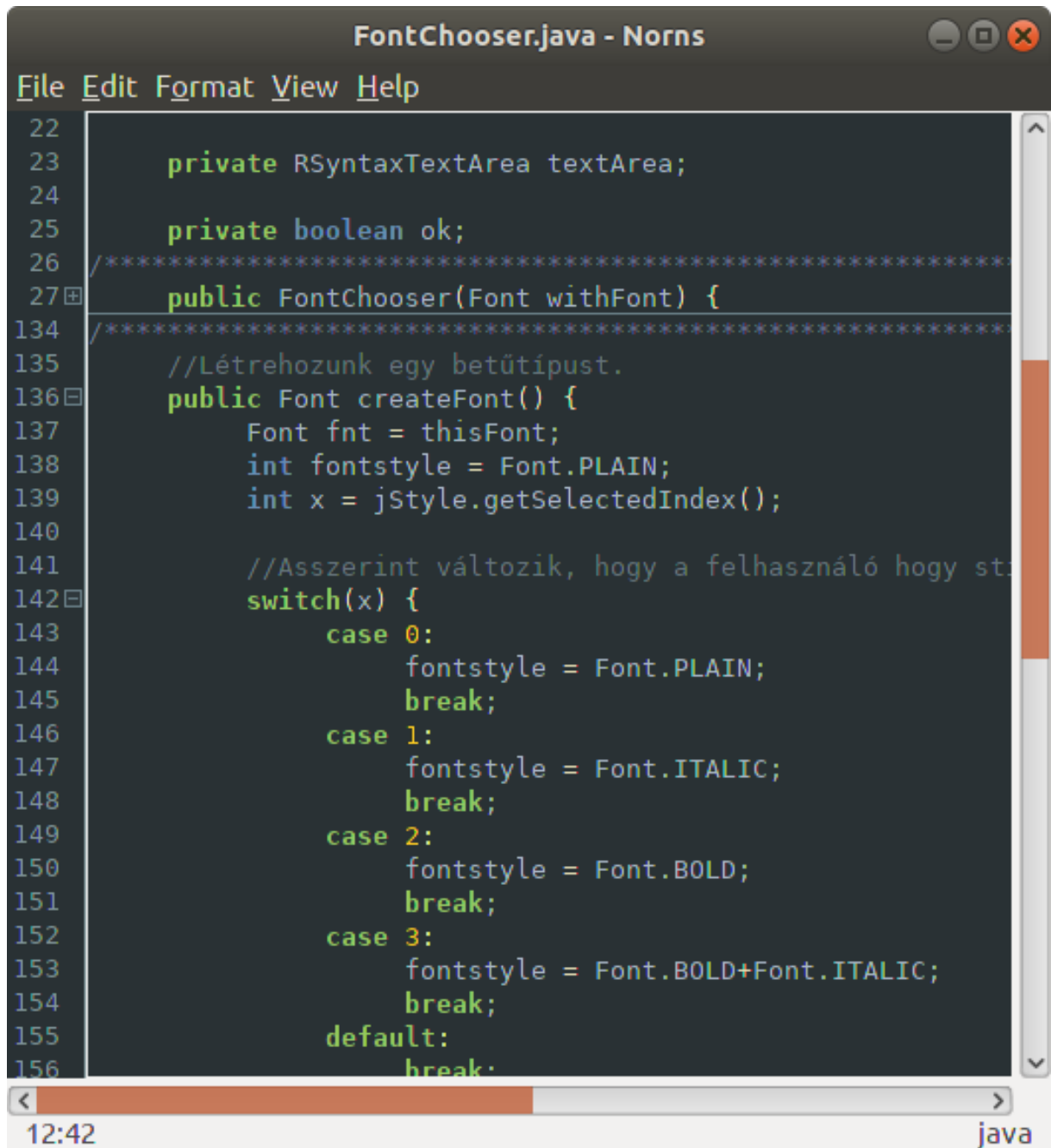
Ha megváltoztatjuk a GUI kinézetét pl.: GTK+ kinézetűre.



```
FontChooser.java - Norns
File Edit Format View Help
1 package nor;
2
3 import java.io.*;
4 import java.awt.*;
5 import java.awt.event.*;
6 import javax.swing.*;
7 import javax.swing.event.*;
8 import org.fife.ui.rtextarea.*;
9 import org.fife.ui.rsyntaxtextarea.*;
10 /*****
11
12 public class FontChooser extends JPanel {
13
14     private static final long serialVersionUID = 66666;
15
16     private Font thisFont;
17
18     private JList jFace, jStyle, jSize;
19
20     private JDialog dialog;
21     private JButton okButton;
22
23     private RSyntaxTextArea textArea;
24
25     private boolean ok;
26 /*****
27 public FontChooser(Font withFont) {
28     thisFont = withFont;
29
```

16:24 java

Ha megváltoztatjuk a szintaxis színezését mondjuk a Dark-ra.



```
FontChooser.java - Norns
File Edit Format View Help
22
23     private RSyntaxTextArea textArea;
24
25     private boolean ok;
26     /**
27     public FontChooser(Font withFont) {
134
135     //Létrehozunk egy betűtípust.
136     public Font createFont() {
137         Font fnt = thisFont;
138         int fontstyle = Font.PLAIN;
139         int x = jStyle.getSelectedIndex();
140
141         //Asszerint változik, hogy a felhasználó hogy st
142         switch(x) {
143             case 0:
144                 fontstyle = Font.PLAIN;
145                 break;
146             case 1:
147                 fontstyle = Font.ITALIC;
148                 break;
149             case 2:
150                 fontstyle = Font.BOLD;
151                 break;
152             case 3:
153                 fontstyle = Font.BOLD+Font.ITALIC;
154                 break;
155             default:
156                 break;
157
12:42 java
```

Ha használjuk az összecsukó funkciót, akkor monjuk ilyen függvényekt lehet összecsukni és, akkor az nem látszik.

Megvalósítás:

A felhasznált eszközök:

- **Operációs rendszer:** Linux
- **Szövegszerkesztő:** Atom
- **Dokumentációhoz használt eszköz:** Latex

A feladat megoldása körülbelül 20-25 órát vett igénybe.