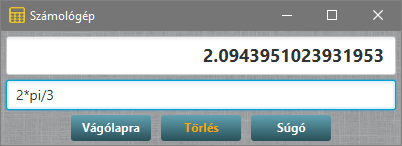
# 14. Számológép

## Feladat

Ebben a leckében egy számológép programot készítünk:

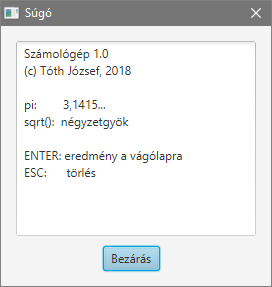


A beírt kifejezés eredménye azonnal megjelenik a kijelzőn, kivéve, ha nem lehet kiszámítani. A műveleti sorrendet figyelembe veszi számításkor.

Ha ENTER-t nyomunk, vagy a Vágólapra gombra kattintunk, akkor az eredmény a vágólapra kerül.

A Törlés gombra kattintva, vagy az ESC megnyomása esetén a kifejezés törlődik.

A Súgó gomb megnyomásakor ez a párbeszédpanel jelenik meg:



Amíg a Súgó ablak látható, nem lehet visszaváltani a fő ablakra.

## Előkészületek

Kezdj egy új FXML Java projektet *szamologep* néven!

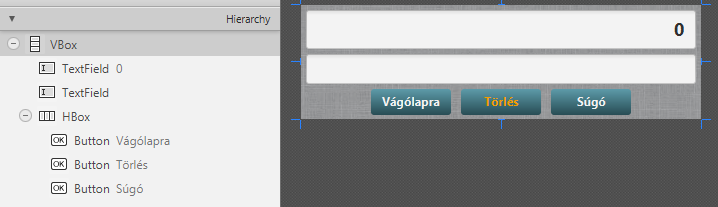
Nyisd meg az FXML fájlt a Scene Builder-rel!

Töröld le az automatikusan létrehozott elemeket! (Mivel ezeknek van fx:id-jük, a törlés előtt figyelmeztet a program.)

Töröld a Controller fájlból is a felesleges kódot!

## Jelenet elkészítése

Az ablakban a VBox tárolót fogjuk használni. Ebben két szöveges mezőt helyezünk el, alattuk pedig egy HBox tárolót, amelybe három gomb kerül:



A VBox-ra állíts be 5 pixel térközt (spacing) és minden oldalára 5 pixel padding-et! Preferált szélessége legyen 400, magassága a számított érték!

A felső szövegmező szövege legyen 0, amely 18 pontos, félkövér betűkkel jelenjen meg, jobbra igazítva (Alignment: CENTER\_RIGHT)! Ne legyen szerkeszthető, és fx:id-je legyen txtEredmeny!

Az alsó szövegmező betűi 14 pontosak legyenek, maradjon a balra igazítás és a szerkeszthetőség! fx:id-je legyen txtKifejezes!

Az alsó HBox igazítása legyen CENTER, térköze 10 pixel!

A HBox-ba kerülő gombok preferált szélessége legyen 80 pixel!

A Vágólapra gomb legyen alapértelmezett (Default Button)! Ez azt jelenti, hogy az Enter lenyomásával is ki lehet választani. Kiválasztásakor a vagolapAction nevű eseménykezelő metódus fusson le!

A Törlés gomb legyen Cancel Button! Ez azt jelenti, hogy az Esc lenyomásával is ki lehet választani. Kiválasztásakor a torlesAction nevű eseménykezelő metódus fusson le! fx:id-je legyen btnTorol!

A Súgó gomb legyen kiválasztásakor a sugoAction nevű eseménykezelő metódus fusson le!

Add meg a controller osztály nevét a bal alsó sarokban (Document, Controller)!

Ezután nyisd meg a View, Show Sample Controller Skeleton paranccsal a controllerbe másolandó kódot, és másold a vágólapra a szükséges deklarációkat!

Mentsd a jelenetet, és válts vissza a NetBeans-re!

Nézd át a generált FXML dokumentumot!

Másold a kódot a vágólapről a controller osztályba!

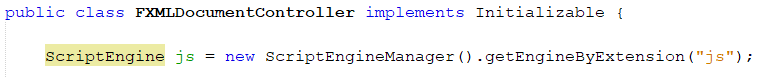
## Számítások

A kiszámítandó kifejezést a txtKifejezes szöveges mezőbe írjuk be.

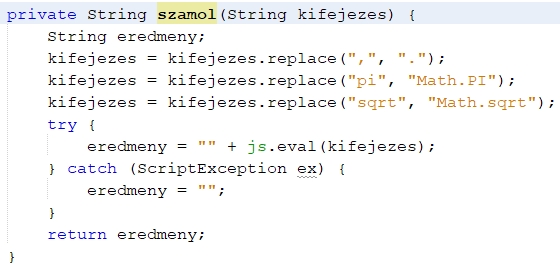
Sajnos a Java-ban nincs olyan metódus, amely kiszámítaná egy paraméterként kapott kifejezés értékét. Viszont a script nyelvekben (JavaScript, Python, …) van erre elehetőség.

A Java-ban pedig arra van lehetőség, hogy a script nyelveken írt programokat lefuttassuk. Ebben a példában JavaScript-et fogunk futtatni.

Ehhez először létre kell hozni egy script motort:



Ezután készítsd el a szamol metódust, amely kiszámítja a paraméterként kapott kifejezés értékét, majd visszaadja az eredményt:



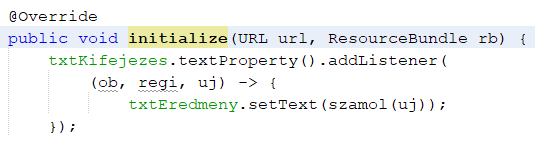
A kifejezésben csak tizedespont szerepelhet, tizedesvessző nem. Ezért a beírt vesszőket a számítás előtt pontokra cseréljük.

A pi és az sqrt elé odatesszük a Math. szöveget, hogy ezt ne kelljen begépelni.

A számítást a try részben lévő sor végzi. A js scriptmotor eval metódusa lefuttatja a kapott utasításokat. Ha ez egy kifejezés, akkor kiszámítja az értékét.

Ha nem sikerült a számítás, akkor egy ScriptException kivétel keletkezik. Ekkor az eredményt üresre állítjuk.

A kifejezés változásakor mindig le kell futatni a szamol metódust. Ezt úgy érjük el, hogy a szöveges mezőhöz egy ChangeListener metódust rendelünk a controller initialize metódusában:

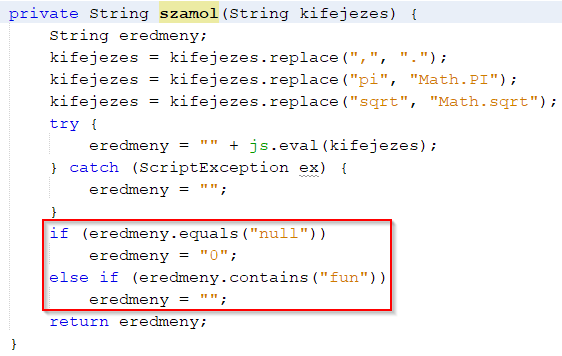


Futtasd a programot, és próbáld ki néhány számítással! Figyeld meg, hogy a műveleti sorrendet is helyesen kezeli!

Néhány apróbb módosítással még használhatóbbá tehetjük a számológépet:

* Ha minden karaktert törlünk a kifejezésből, akor a null felirat jelenik meg. Ezt el kell tüntetni.
* A függvény beírásakor még így is különböző szövegek jelennek meg, ezeket is el kell tüntetni.

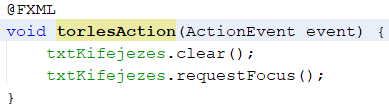
Egészítsd ki a szamol metódust a következőkkel:



Próbáld ki!

## Törlés

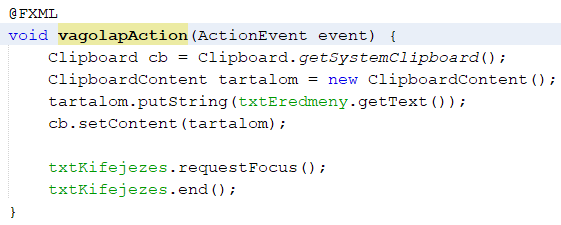
A Törlés gomb vagy az Esc megnyomásakor a teljes kifejezést törölni kell. Készítsd el az eseménykezelőt:



A clear metódus törli a szöveges mező tartalmát, a requestFocus pedig odaviszi a kurzort. Próbáld ki!

## Vágólap

Az eredményt a következő módon tudod a vágólapra másolni:

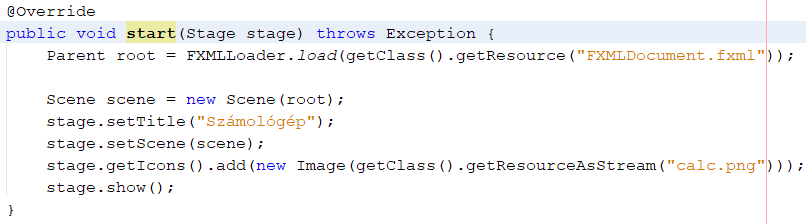


A szöveges mező end metódusa a szöveg végére viszi a kurzort.

## Ikon módosítása

Állítsd be az ablak címét Számológépre!

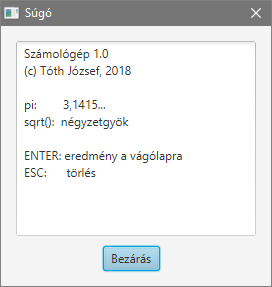
Másold a kapott calc.png képet az src/szamologep mappába, és egészítsd ki a Szamologep osztályban a start metódust így:



Futtasd a programot, és figyeld meg az ikont a bal felső sarokban!

## Súgó ablak

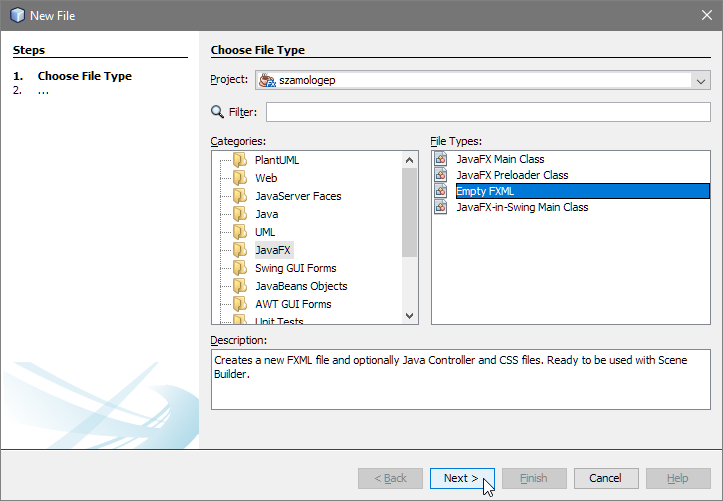
Ha a felhasználó megnyomja meg a Súgó gombot, akkor egy új ablakban jelenjen meg a súgó!



Ez az ablak legyen **modális**, vagyis addig ne lehessen visszamenni a Számológép ablakra, amíg ezt be nem zártuk! Átméretezésre, nagyításra, kicsinyítésre nincs szükség.

Először külön elkészítjük az ablak tartalmát, majd a gomb eseménykezelőjéből megnyitjuk.

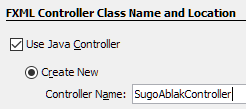
Kattints a szamologep csomagra a jobb gombbal, majd válaszd a New/Other parancsot! A megjeleno ablakban válaszd a JavaFX/Empty FXML lehetőséget:



A következő ablakban az FXML név legyen SugoAblak!

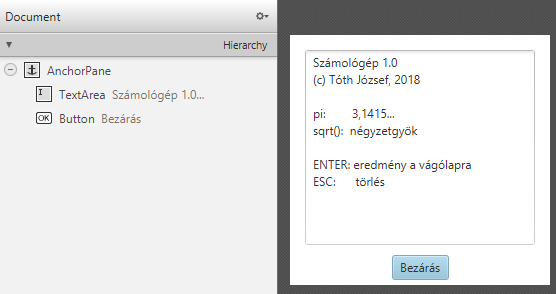


Ezután add meg a Controller osztály nevét:



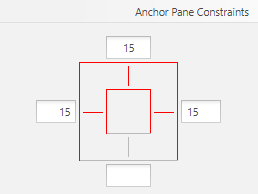
CSS fájl most nem szükséges.

Ezután készítsd el a Súgó ablak tartalmát az alábbiak szerint:



Az AnchorPane mérete legyen 260x250 képpont!

A TextArea vezérlőbe írd be a több soros szöveget! Ne legyen szerkeszthető! A Layout lapon állítsd be, hogy az ablak felső és két oldalsó szélétől 15 pixel távolságra legyen! Magassága 195 px legyen!



Próbáld ki, mi történik, ha módosítod az ablak méretét!

A gomb vízszintesen középre kerüljön (húzás), megnyomásakor a bezarAction metódus fusson le! fx:id-je legyen btnBezar!

Add meg a controller osztály nevét a bal alsó sarokban (Document, Controller)!

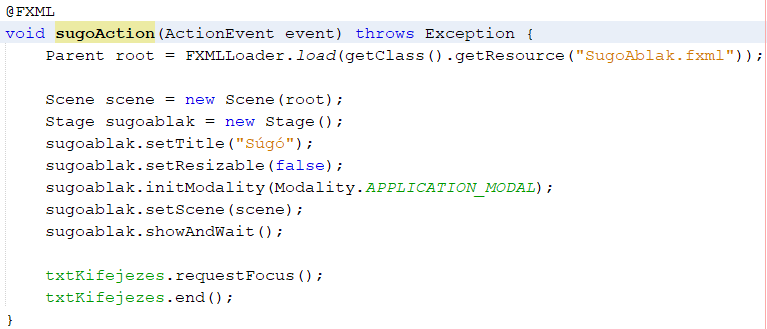
Ezután nyisd meg a View, Show Sample Controller Skeleton paranccsal a controllerbe másolandó kódot, és másold a vágólapra a szükséges deklarációkat!

Mentsd a jelenetet, és válts vissza a NetBeans-re!

Nézd át a generált FXML dokumentumot!

Másold be a kódot a vágólapról a SugoAblakController osztályba!

Válts át az FXMLDocumentController osztályra, és készítsd el a sugoAction metódust így:



Ez majdnem úgy működik, mint a fő ablak indítása a Szamologep osztályban (egy részét onnan lehet másolni), de vannak különbségek:

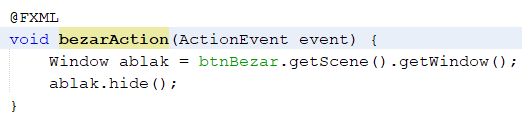
* nekünk kell létrehozni az új Stage-et,
* a showAndWait metódussal nyitjuk meg az ablakot,
* beállítjuk a modalitást.

Próbáld ki a Súgó ablakot! Figyeld meg, hogy amíg nem zárod be, addig nem tudsz visszaváltani a számológép ablakára!

Most tedd megjegyzésbe a az initModality metódus hívásának sorát, és próbáld ki így is a programot! Mi a különbség?

Válts át a SugoAblakController osztályra!

A Bezár gomb megnyomásakor csak a Súgó ablakot kell bezárni (elrejteni), a programot nem. Ezt így lehet megvalósítani:

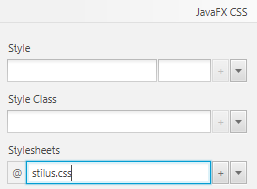


Próbáld ki!

## Formázás stílusokkal

Hozz létre egy új CSS fájlt a szamologep csomagban (New, Other, Other, Cascading Style Sheet)!

Rendeld hozzá az FXMLDocument-ben a Scene Builder segítségével a VBox tárolóhoz! (Így a benne lévő elemekre is fog vonatkozni.)

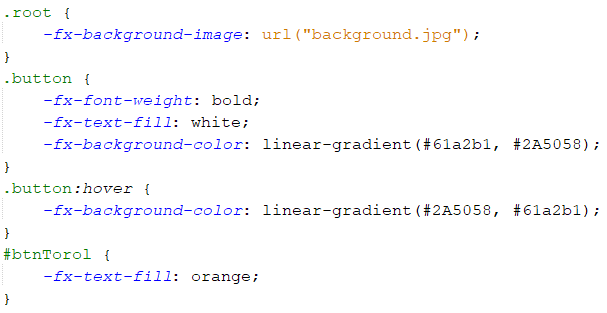


Mentsd a módosítást, majd válts a NetBeans-ben a stilus.css fájlra!

A CSS fájlban a selectorokat a következőképpen használhatjuk:

* Ha azt szeretnénk, hogy egy adott típusú elem, például egy gomb minden előfordulására vonatkozzon a beállítás, akkor osztályt használunk (például .button az összes gombra vonatkozik).
* Ha egy adott fcx:id-jű elemre akarunk megadni egy beállítást, akkor azonosítót használunk (például #btnTorol).

Készítsd el a következő beállításokat a CSS fájlban, és próbáld ki őket!



Próbáld ki!

## Feladatok

1. Nézd végig az eseménykezelőket! Melyiket mikor indítja el a program?
2. Készíts JAR fájlt a programból, és futtasd le!