WhiteStarUML Tutorial

Készítette:Simon Balázs, BME IIT, 2015.

Telepítés

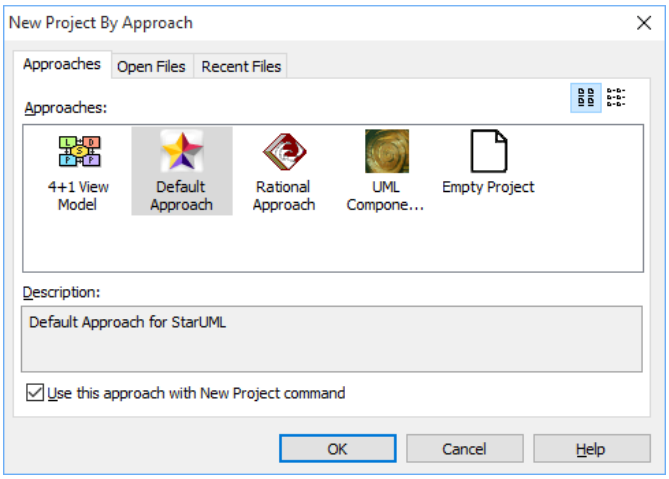
Töltsük le és telepítsük a WhiteStarUML-t:

http://sourceforge.net/projects/whitestaruml/

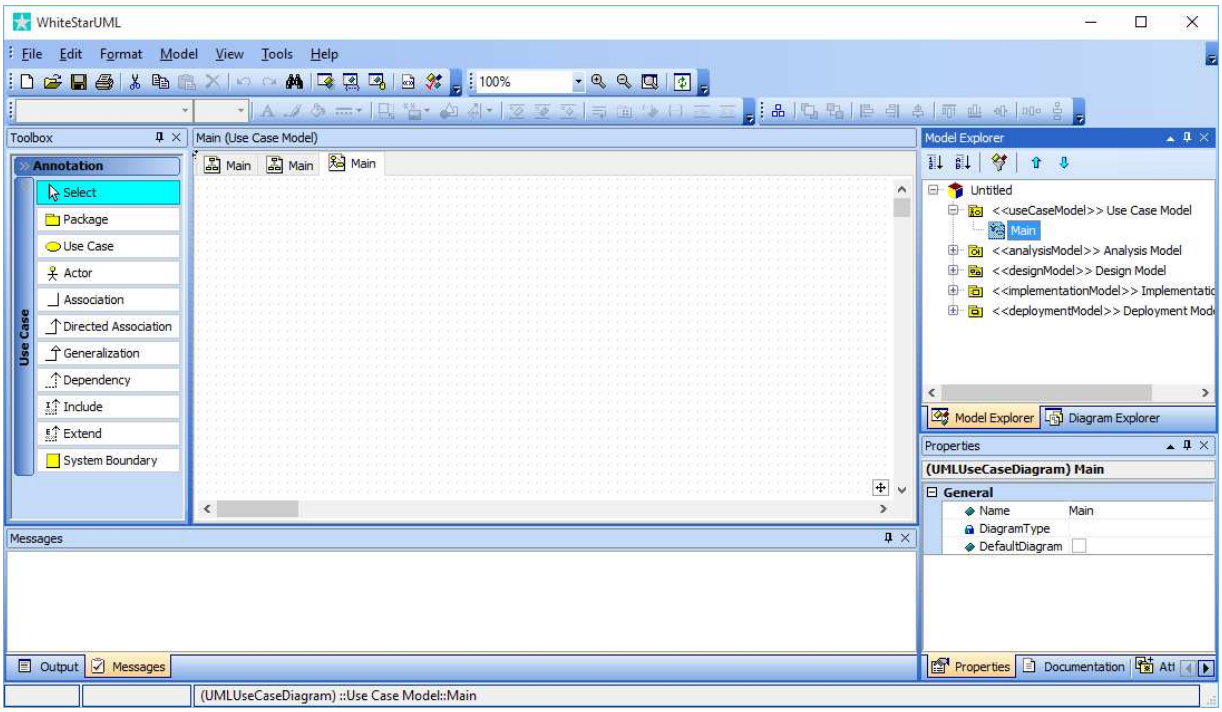
A HSZK laborgépein az eszköz már telepítve van.

Indítás

Indítsuk el a WhiteStarUML-t, és válasszuk ki a **Default Approach**-ot az új projekthez:



A főablak a következőképpen néz ki:



A középső területen lehet grafikusan szerkeszteni az UML diagramokat.

Bal oldalon található a paletta (**Toolbox**), ahonnan az UML modell megfelelő elemei elérhetők.

Jobb oldalon fent (**Model Explorer**) található a diagramok fastruktúrában ábrázolt képe, amely hasznos lehet abban az esetben, ha a grafikus szerkesztőben bizonyos elemek nem választhatók ki közvetlenül (pl. operációk paraméterei).

Jobb oldalon lent (**Properties**) szerkeszthetők a kiválasztott UML elemek tulajdonságai (pl. név, típus, stb.).

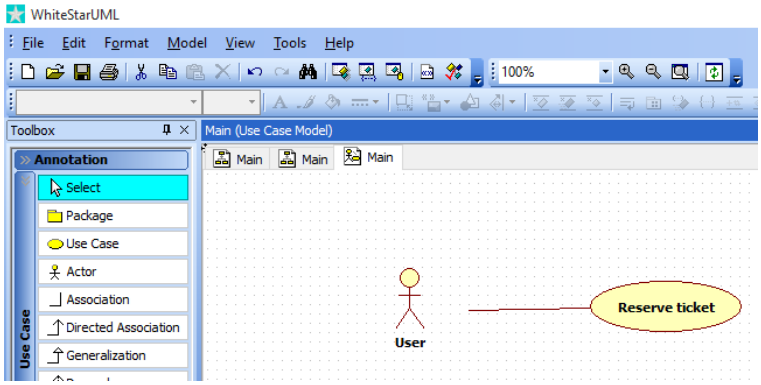
Use-case diagram készítése

A **Model Explorer**-ben dupla kattintással válasszuk ki a **Use Case Model** alatt a **Main** diagramot.

Kattintsunk a palettán a pálcikaemberen, majd pedig a diagramon, és nevezzük el **User**-nek. Helyezzünk egy use-case elemet is a diagramra, és nevezzük el **Reserve ticket**-nek.

Rajzoljunk egy asszociációt a pálcikaember és a use-case közé. Ehhez kattintsunk a palettán az **Association** elemen, majd a diagramon nyomjuk le a bal gombot a pálcikaemberen és engedjük fel a use-case elem fölött.

A diagram így néz ki:



Osztálydiagram készítése

Duplakattintással válasszuk ki a **Model Explorer**-ből a **Design Model** alatti **Main** diagramot.

Vegyünk fel egy Cinema nevű osztályt:

A világoskék dobozkával attribútumokat (UML property), a piros dobozkával operációkat (UML operation) vehetünk fel.

A felvett attribútumok és operációk tulajdonságainak szerkesztéséhez a **Model Explorer** és a **Properties** ablakok használhatók.

Sokszor egyszerűbb a pontos UML szintaxissal beírni az adott elem láthatóságát, nevét, típusát, mert azt a WhiteStarUML megfelelően képes feldolgozni.

Vegyünk fel például egy attribútumot a következő szintaxissal:

```
- address: String
```

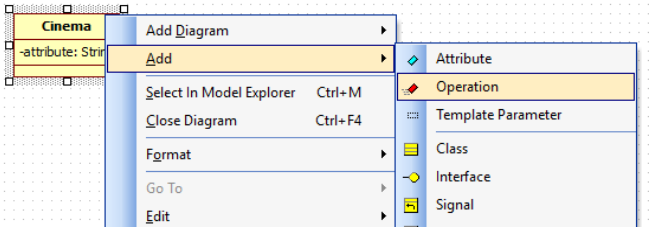
Ellenőrizzük a **Model Explorer** és a **Properties** ablakokban, hogy a tulajdonságai megfelelők-e.

Vegyünk fel egy operációt is:

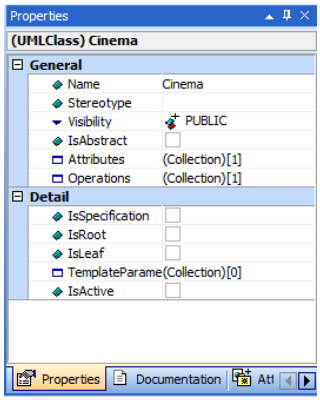
```
+ ReserveTicket(row: int, column: int): boolean
```

Ellenőrizzük a **Model Explorer** és a **Properties** ablakokban, hogy a tulajdonságai megfelelők-e.

A világoskék és piros dobozkákhoz tartozó menüpontok előhívhatók az osztályon való jobb gombbal kattintással is:

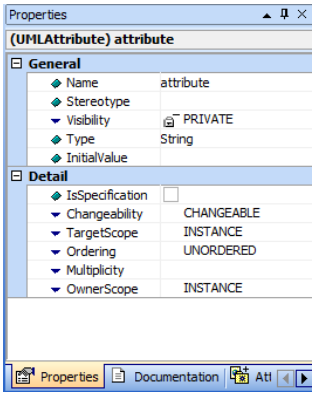


Egy osztály tulajdonságai a **Properties** ablakban:



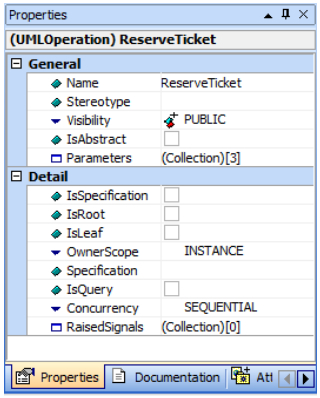
Itt lehet például beállítani azt, hogy mi az osztály neve, absztrakt legyen-e, milyen sztereotípiái vannak (pl. <<interface>>), stb.

Egy attribútum tulajdonságai:



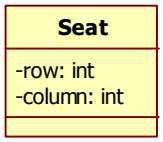
Itt lehet állítani az attribútum nevét, típusát, láthatóságát, és azt, hogy statikus legyen-e (OwnerScope=CLASSIFIER).

Egy operáció tulajdonságai:

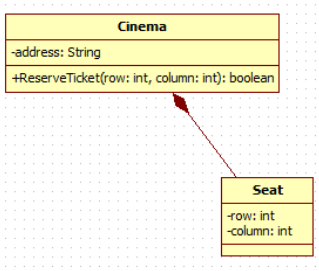


Itt állítható az operáció neve és láthatósága, illetve az, hogy az operáció absztrakt-e, statikus-e, stb.

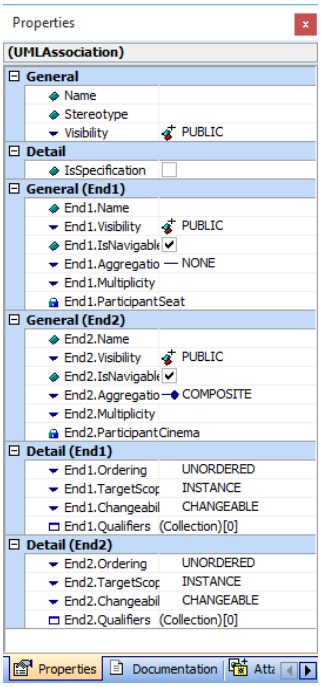
Vegyünk fel még egy osztályt **Seat** néven:



Vegyünk fel egy kompozíciós asszociációt (**Composition**) is a **Cinema** és a **Seat** közé. A **Seat** osztályon nyomjuk le a bal egérgombot, és a **Cinema** osztályon engedjük fel:



Válasszuk ki az asszociációt, és a tulajdonságok ablakban szerkeszthetők a tulajdonságai:



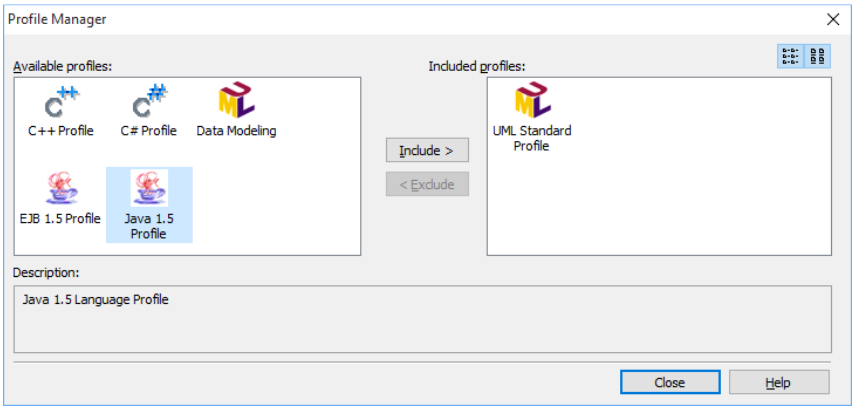
Itt szerkeszthetők az asszociáció két végének adatai (pl. számosság, navigálhatóság, qualifier-ek, stb.).

Hasznos információk

A diagramról többféleképpen lehet törölni elemet. A **Del** billentyű hatására csak a grafikus nézete tűnik el az elemnek, de a modellben (**Model Explorer**) még megmarad. A végleges törléshez onnan is törölni kell az adott elemet. A **Ctrl+Del** billentyűkombináció hatására a grafikus nézetből és a diagramból is törlődik az elem.

Kódgenerálás

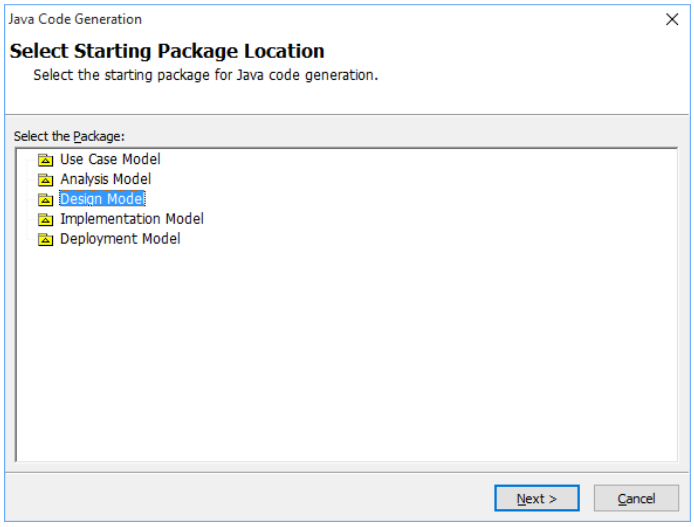
Kódgeneráláshoz először hozzá kell rendelni a Java profile-t a projekthez. Ezt a **Model > Profiles...** menüpont alatt tehetjük meg:



Válasszuk ki a **Java 1.5 Profile**-t és az **Inlcude** gombbal adjuk hozzá a projekthez, majd a **Close** gombbal zárjuk be az ablakot.

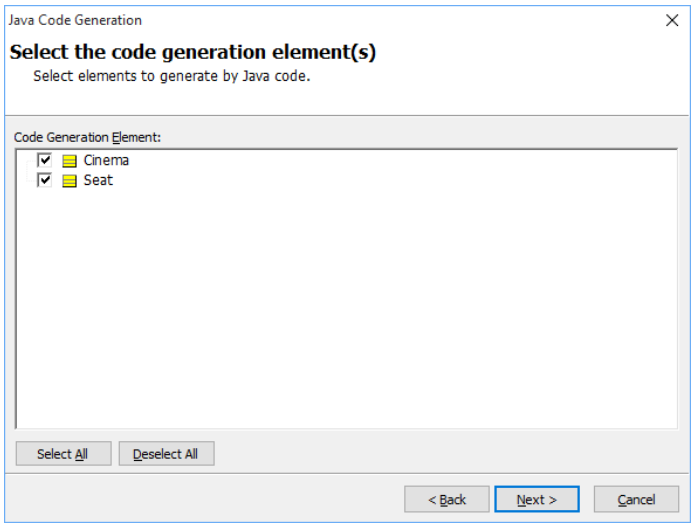
Mielőtt a Java kódot előállítanánk, célszerű a kimeneti könyvtárat előre létrehozni az operációs rendszerben, mert a generátor varázslója erre nem ad lehetőséget.

A Java kód előállítását a **Tools > Java 1.5 > Generate Code...** menüpont segítségével tehetjük meg:

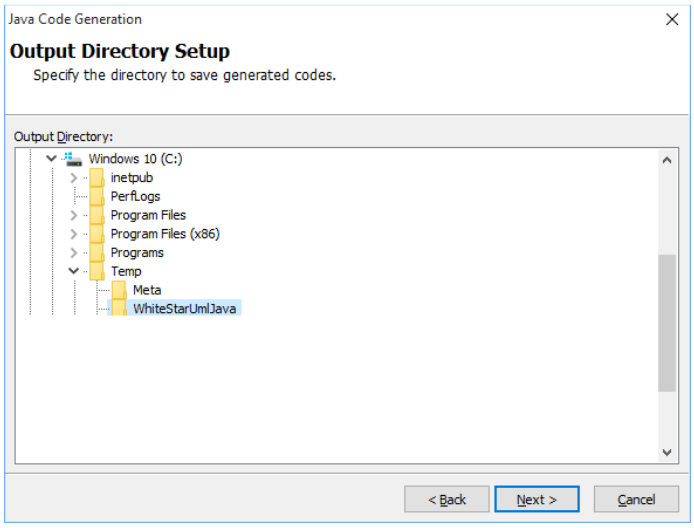


Ha az osztálydiagramot az **Analysis Model** alatt hoztuk létre, akkor azt válasszuk, ha pedig a **Design Model** alatt készítettük, akkor ezt válasszuk ki.

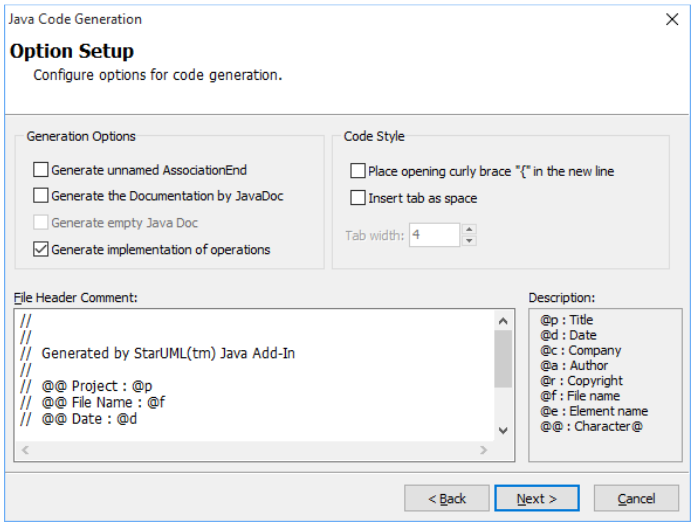
Ezután kattintsunk a **Next**-re:



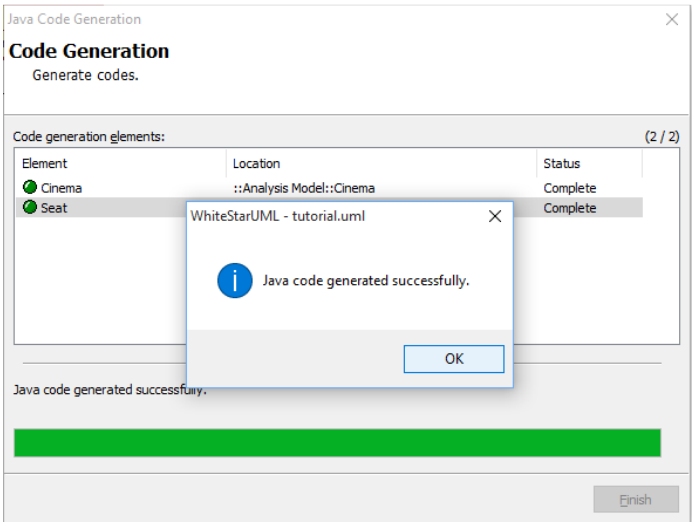
Itt a válasszuk ki az osztályokat, amelyekhez kódot szeretnénk generálni (célszerűen mindenhez, így a Select All-ra érdemes kattintani), majd kattintsunk a **Next** gombra, és válasszuk ki a korábban létrehozott kimeneti könyvtárat:



Kattintsunk a **Next**-re. Ha adtunk meg dokumentációt az egyes modellelemekhez, akkor megtarthatjuk a pipát a **Generate the Documentation by JavaDoc** mellett, egyébként vegyük ki. Pipáljuk ki a **Generate implementation of operations** elemet:



Kattintsunk a **Next**-re, és végül kapunk egy értesítést a generálás sikerességéről:



Sajnos a WhiteStarUML kódgenerátora nem túl okos. Nem kezeli például a több multiplicitású asszociációvégeket, és a konstruktorokat sem ismeri fel, így void típust generál nekik.

Megjegyzés: az OpenAmeos nevű eszköz sokkal jobb (sőt, programozható) kódgenerátorral rendelkezik, azonban a grafikus szerkesztőfelülete kevésbé kényelmes. Érdeklődők számára az eszköz itt érhető el: https://www.scopeforge.de/cb/project/8