**Szoftver laboratorium 3 Nagy Hazi Feladat**

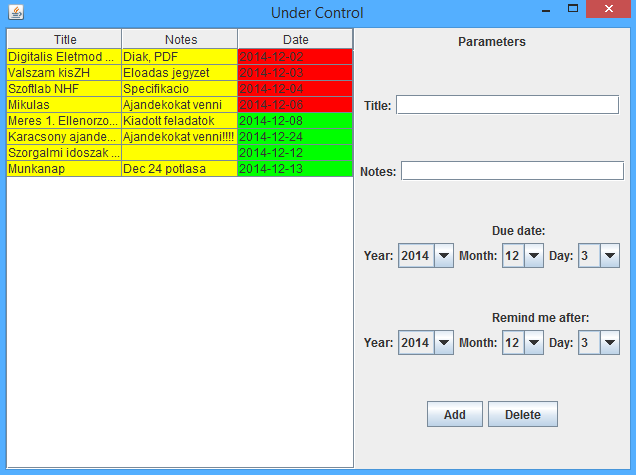
**Pápai Attila**

**X6YVJM**

**Hazi feladat ismertetese:**

A program egy nagyon szimpla “To-do List”.

A felhasznalonak lehetosege van Taskokat felvenni. Egy Tasknak 4 parametere van. Cim, jegyzetek, esedekes datum, emlekezteto napja.

A program egy grafikus alkalmazas. 2 reszre van felbontva a felulet. Baloldali resz a lista, ahol megjelennek az eddig felvett taskok. Jobboldalon lehet felvenni uj taskokat, illetve meglevoket torolni.

Megkotesek az alkalmazassal kapcsolatban:

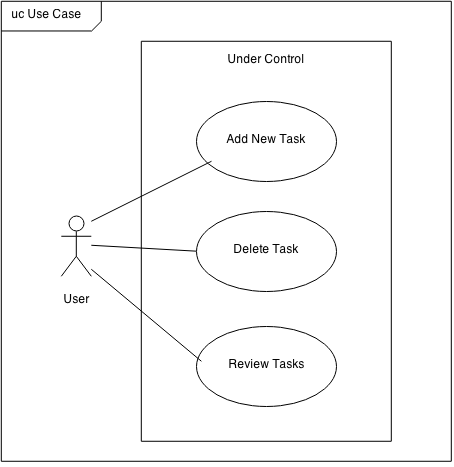
1. A listaban nem szerepelhet ugyanaz a task ketszer. Egy task akkor egyezik meg egy masikkal, ha a neve ES az esedekes datuma is megegyezik.
2. Felhasznalo csak olyan taskokat vehet fel, amik teljesitik az alabbi kriteriumokat:
   1. Az emlekezteto datum napja nem lehet elobb, mint az esedekes datum
   2. A tasknak van neve (cime)
   3. Az esedekes nap nincs a multban. Alapbol a program ugy allitja be a datumokat, hogy a multbeli napokat ne lehessen kivalasztani, azonban egy trukkel ezt ki lehet kerulni. Legalabbis addig, amig az „Add” gombra nem kattint, mert akkor a program visszadobja a taskot.
   4. A task nem szerepel a listaban.

Termeszetesen a program tamogatja a torles lehetoseget is. Ha a felhasznalo torolni szeretne egy taskot, annyit kell csinalnia, hogy kijelolje az adott task egyik parameteret a listaban, ekkor az egesz sor ki lesz jelolve, es rakattint a „Delete” gombra. Ha nincs kijelolve egyetlen egy task sem, az alkalmazas nem torol semmit, hibauzenetet dob.

Az alkalmazas indulaskor alapertelmezetten tolti be az adatokat. Ez a „tasks.dat” fajl. Ha nem letezik, ures a listaval indul a program.

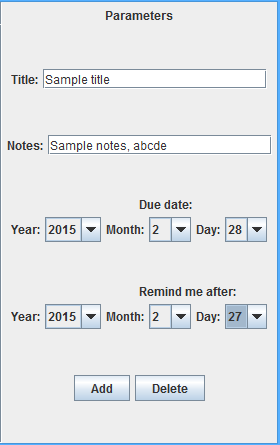
Kilepeskor automatikusan ment a program, szinten a „tasks.dat” fajl tarolja a tartalmat a tablazatnak.

**Funkciok ismertetese:**

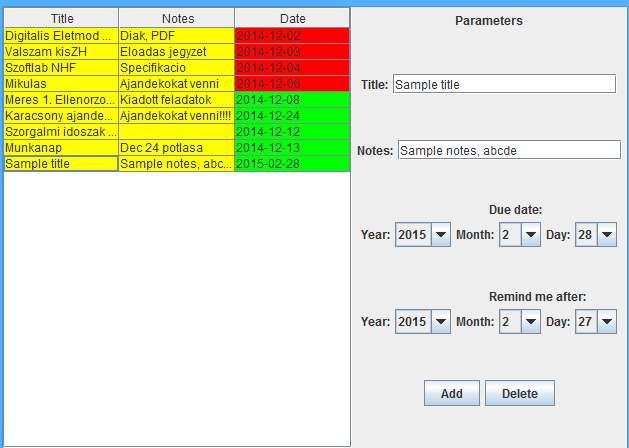


Itt lathatjuk, hogy ebben a programban 3 fele funkciot erhet el a felhasznalo:

1. Add New Task:

Miutan a felhasznalo kitoltotte a title mezot, lehetosege van felvenni uj taskot. Ezt az abran bekeretezett “Add” gomb segitsegevel lehet megtenni.

1. Delete Task



Kijelolt elem (sor)

A felhasznalo, miutan kijelolt egy elemet, a “Delete” gomb megnyomasa utan ki tudja torolni azt.

1. Review Tasks

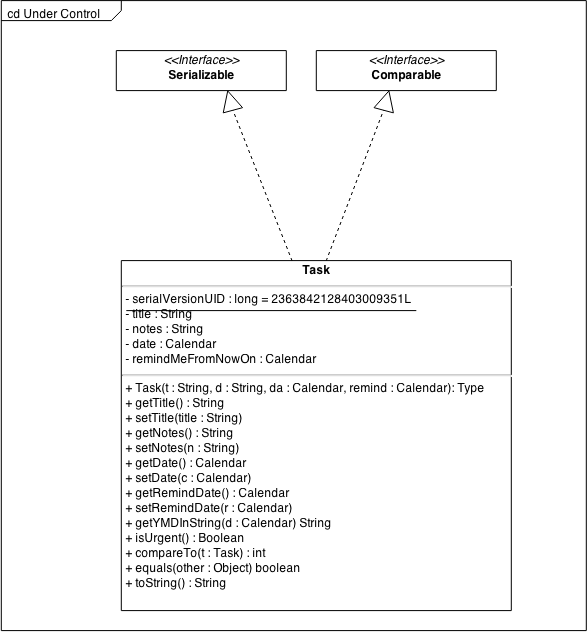


A felhasznalo miutan felvett legalabb 1 taskot, a listaban mar lathatja is a lenyeges informaciokat. A szinekrol es a sorrendrol kesobb less szo.

**Megoldas menete, UML**

Celszerunek lattam az alkalmazast a kisebb elemtol a nagyobbig felepiteni.

**Task class:**



Lathato, hogy nincs default konstruktor, ilyet nem engedunk meg. Alapveto getterek es setterek, hogy a classal kenyelmes legyen dolgozni.

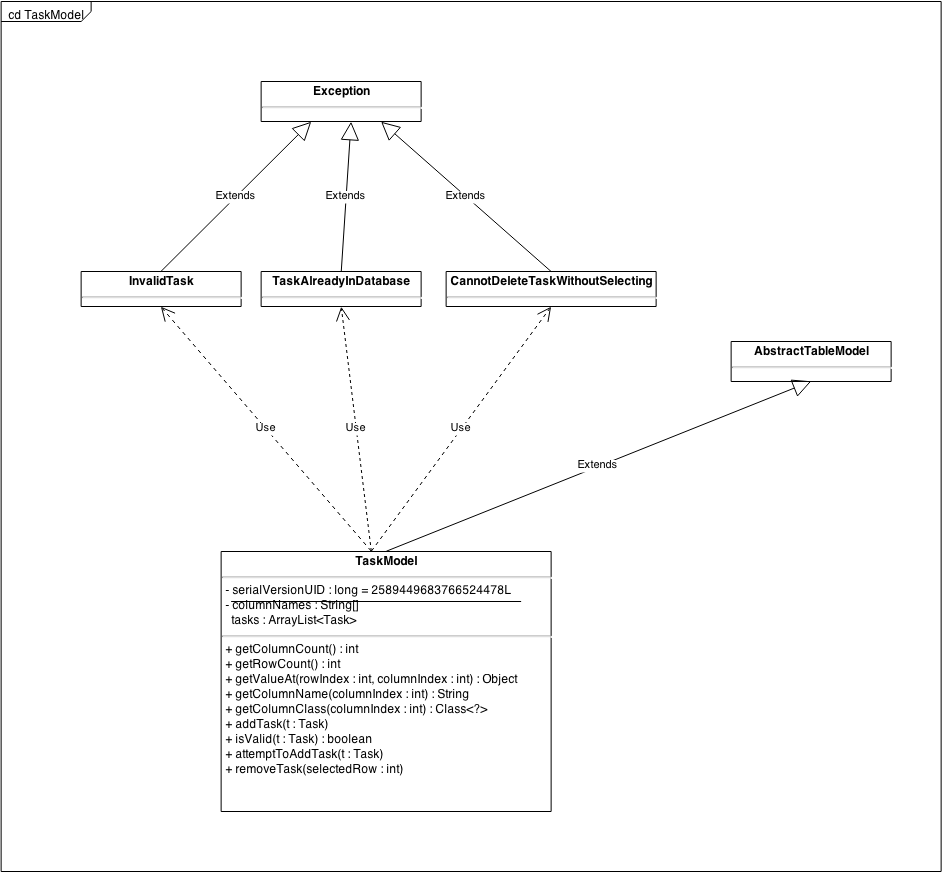
A *getYMDInString* osztaly arra szolgal, hogy a Calendar formatumban levo datumot, valami emberileg olvashatova tudjuk tenni. Jelen esetben ez a kovetkezo kep nez ki: pl.: “2014-01-11”.

Az *isUrgent*() metodus arra szolgal, hogy megtudjuk, hogy a task fontos, aktualis-e. Ebben a programban akkor aktualis egy task, ha a emlekezteto napjat mar tulleptuk.

*compareTo*(t : Task) metodus erdekes, mert nem azt adja eredmenynek, amit alapbol az ember gondolja. A metodus azert fontos, mert ha kesobb ossze akarunk hasonlitani ket taskot, akkor nem az less elobb, amelyik idoben is elobb van, hanem amelyiknek az emlekeztetoje van elobb.

*equals*(other : Object) metodus akkor ad vissza true erteket, ha a ket tasknak megegyezik a neve es a esedekes datuma.

**TaskModel class:**



Lathato, hogy ez az osztaly mar hasznal kiveteleket, illetve az ose az AbstractTableModel osztaly.

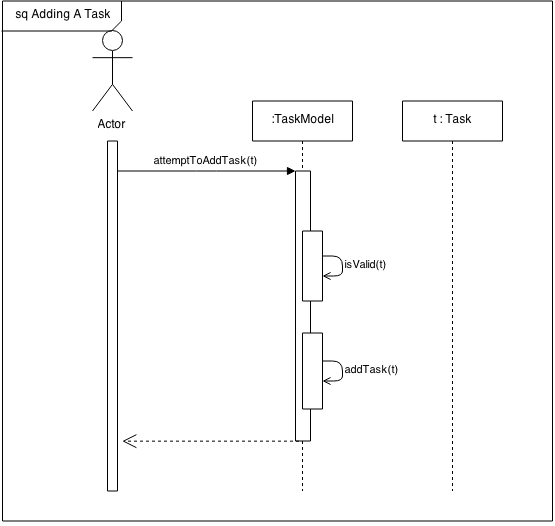
Az *addTask*(t : Task) metodus felelos a task listahoz adasaert. Itt meg tortenik egy ellenorzes, ekkor meg nem biztos, hogy bekerul a listaba (TaskAlreadyInDatabase).

Az *isValid*(t : Task) megnezi, hogy az emlekezteto elobb van-e, mint az esedekes nap.

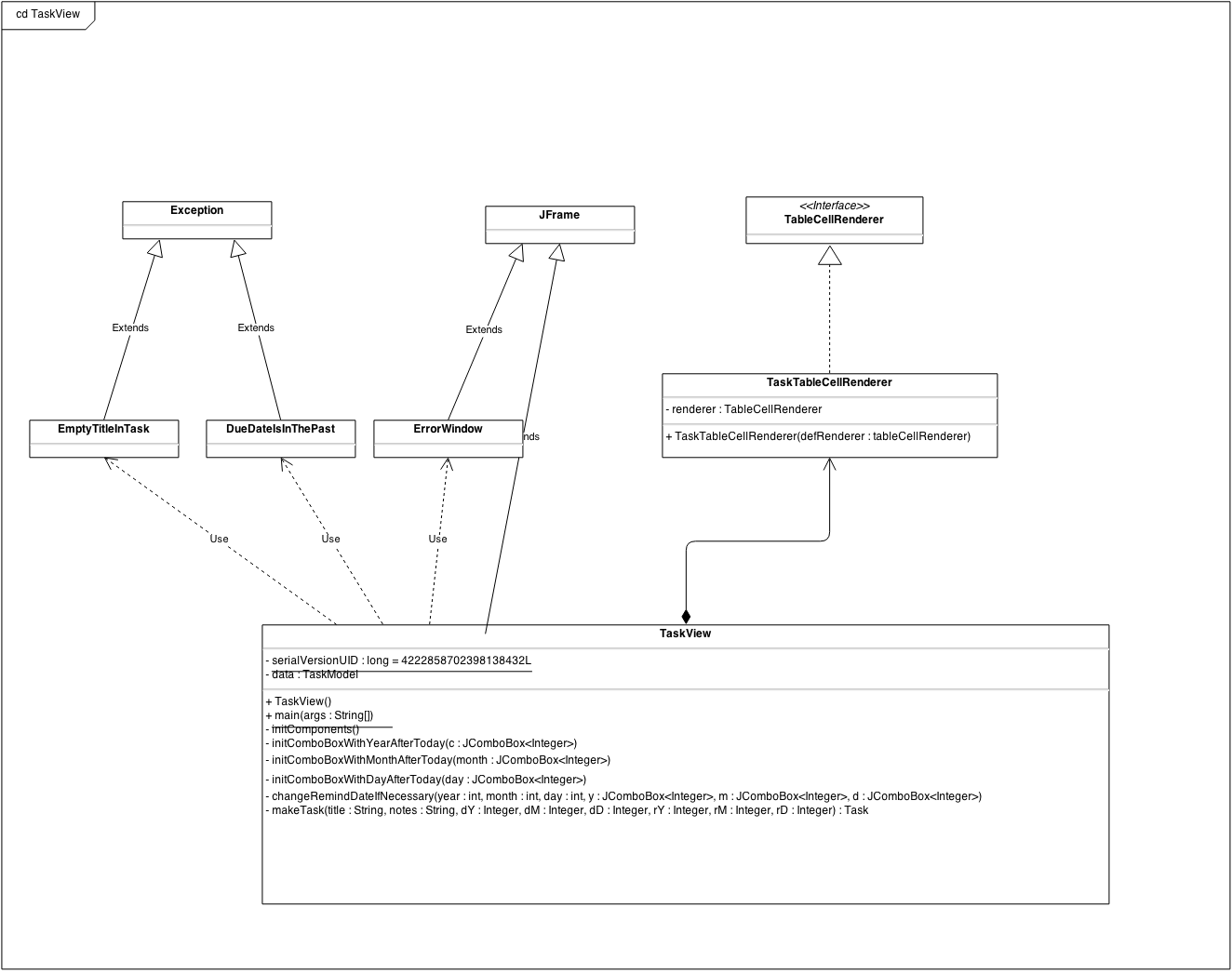
*attemptToAddTask*(t : Task) hivja meg az isValid es addTask fuggvenyt

*removeTask*(selectedRow : int) tavolitja el az elemet a listabol.

Legjobban a metodusok viszonyat az alabbi szekvencia diagram irja le:



**TaskView Class:**

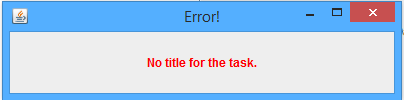


A program nagyon sokban hasonlit a SwingMVC labor kiadott feladatara. Itt is megprobaltam elkuloniteni a modelt, a viewt, es a controller.

Itt is hasznalom a TableCellRenderer osztalyt, ami a szinezesert felelos.

A hibakezelesre hoztam letre a kulonbozo Exception osztalyokat, illetve az ErrorWindow osztalyt. Mukodes kozben valahogy igy nez ki:

1. A felhasznalo valami olyat probal csinalni, ami a program mukodesere veszelyes lenne, vagy csak szembemegy a program logikajaval
2. Megprobalja a program elkapni a try-catch blokkokban
3. Tudatja a felhasznaloval a kovetkezo modon (ez csak egy pelda az ErrorWindowra):



**Taskok sorrendje:**



1. A task akkor piros, ha az *isUrgent*() fuggveny true erteket ad vissza.
2. A sorrendet az emlekezteto datum hatarozza, az van elobb, amelyiknek korabban van a remindere.

**Teszteles:**

JUnitot hasznalok a tesztelesre.

A 2 osztaly, aminek a legfontosabb metodusait tesztelem:

1. Task
2. TaskModel

Ezek a tesztek fokent arra mennek ki, hogy az alapveto funkciokat, exception dobasokat, illetve program logikaval kapcsolatos dontesek eredmenyet nezi.

Az eredmeny:

