Heuristieken FreeCell JAVA library

# Inhoud

Dit document is een korte beschrijving voor het gebruik van de FreeCell Java library. Deze library is in het voorjaar van 2010 geschreven in de programmeertaal Java voor het vak heuristieken. Als er vragen/opmerkingen zijn over het (gebruik van het) programma dan kan je direct contact opnemen met:

* Daan van den Berg ([daanvandenberg1976@gmail.com](mailto:daanvandenberg1976@gmail.com))
* Patrick van Rietschoten ([prn250@few.vu.nl](mailto:prn250@few.vu.nl))

De source code is niet direct beschikbaar, maar mocht je geïnteresseerd zijn in het lezen en/of bewerken van de code dan kun je ons ook contacten.

De volgende bestanden heb je nodig:

* FreeCell Library: <http://www.few.vu.nl/~prn250/heuristieken/FreeCell.jar>
* Javadoc: <http://www.few.vu.nl/~prn250/heuristieken/javadoc/>
* Voorbeeld: <http://www.few.vu.nl/~prn250/heuristieken/example.java>

We zullen hieronder alleen de library beschrijven. Na het doorlezen van dit document zouden de Javadoc en het voorbeeld voor zich moeten spreken.

# Library

De library bestaat uit twee hoofdonderdelen, een grafische user interface (FreeCellGUI) en een implementatie van het FreeCell spel. Van de implementatie van het FreeCell spel zijn twee versies beschikbaar, een simpele maar efficiënte implementatie (FreeCellGame) en daarbovenop is een uitgebreidere maar langzamere implementatie gemaakt (RegisteredGame, die FreeCellGame extend). Als laatste heb je de Move class en de meer uitgebreide RegisteredMove class (die de Move class extend):

Figuur 1 - Schematische beschrijving FreeCell library

## FreeCell-Game en Registered-Game

Het grootste verschil tussen een FreeCellGame object en een RegisteredGame object is dat in de laatste de uitgevoerde moves worden opgeslagen en ongedaan kunnen worden en dat er gebruik gemaakt wordt van de RegisteredMove class in plaatst van de Move class. Maar let op, als je een oplosser schrijft is het efficiënter om een FreeCellGame object in combinatie met backtracking te gebruiken aangezien de een ongedane-move ook opgeslagen wordt.

## FreeCell-GUI

De gebruikersinterface kan op meerdere manieren gebruikt worden. In alle gevallen moet de FreeCellGUI een RegisteredGame object meekrijgen zodat uit dit object informatie over de verdeling van kaarten, vorige moves etc. kunnen worden gehaald en zodat eventuele nieuwe moves kunnen worden opgeslagen.

De eerste, intuïtieve manier om de gebruikersinterface te gebruiken is dat de gebruiker een spel opstart en deze oplost, daarna kan het spel opgeslagen worden voor latere analyse. Als de speler een move maakt dan wordt eerst gecontroleerd of deze move toegestaan is volgens de FreeCell regels, daarna wordt deze move uitgevoerd en opgeslagen in het RegisteredGame object.

Daarnaast is het ook mogelijk om de gebruikersinterface te gebruiken om (zonder de gebruiker) een animatie van een of meerdere moves uit te voeren. Hierbij wordt niet gecheckt of een move toegestaan is. Desondanks wordt de move wel uitgevoerd en opgeslagen in het RegisteredGame object. Een call naar een animatie-methode is non-blocking, oftewel de move wordt in een queue opgeslagen en uitgevoerd als dat zo uitkomt, de methode returnt wel gelijk.

## Move en RegisteredMove

Deze twee classes worden gebruikt om moves te representeren, een move is geen object van het type Move, maar een int[]. Waarin de i-de plek in het array overeenkomt met een bepaalde constante uit de Move class. In het voorbeeld wordt hier een duidelijk voorbeeld van gegeven.

## Overige onderdelen

Naast de bovenstaande hoofdonderdelen bevat de library nog een aantal extra packages:

### utils

Deze package bevat een Util class die handige methodes bevat voor onder andere het printen van bepaalde spelonderdelen. Ook is een Notification class beschikbaar die tekst aan de gebruiker laat zien.

### freecell.management

Deze package bevat allerlei classes die gebruikt zijn om het spel te implementeren, over veel van deze classes hoef je, je niet druk te maken. Table, Free en Base zijn respectievelijk de columns, vrije cellen en de base cellen. Dan heb je ook nog de Deck en Card classes die geen uitleg nodig hebben. De belangrijkste classes zijn Move en RegisteredMove