Софтуерни шаблони за проектиране (Software Design Patterns)

Изготвил:

Диньо Динев Фак. № M24133

Тема:

Ханойски кули Project

19.06.2013

София

Contents

[1. Изисквания за приложението 3](#_Toc328282847)

[2. Избор на технологии 3](#_Toc328282848)

[3. Дизайн на системата 3](#_Toc328282849)

[4. Реализация на приложението 10](#_Toc328282850)

# 

# Изисквания за приложението

Заданието за разработка на приложението е дефинирано по следния начин:

Да се създаде прототип на играта Ханойски кули с персистентни данни (запазване на статуса на играта, текущ, краен за даден именуван играч), с един потребител (играч) и едно административно приложение.

Освен изисквания към приложението има и няколко допълнителни изисквания за начина, по който трябва да е направен дизайна на системата. Това са:

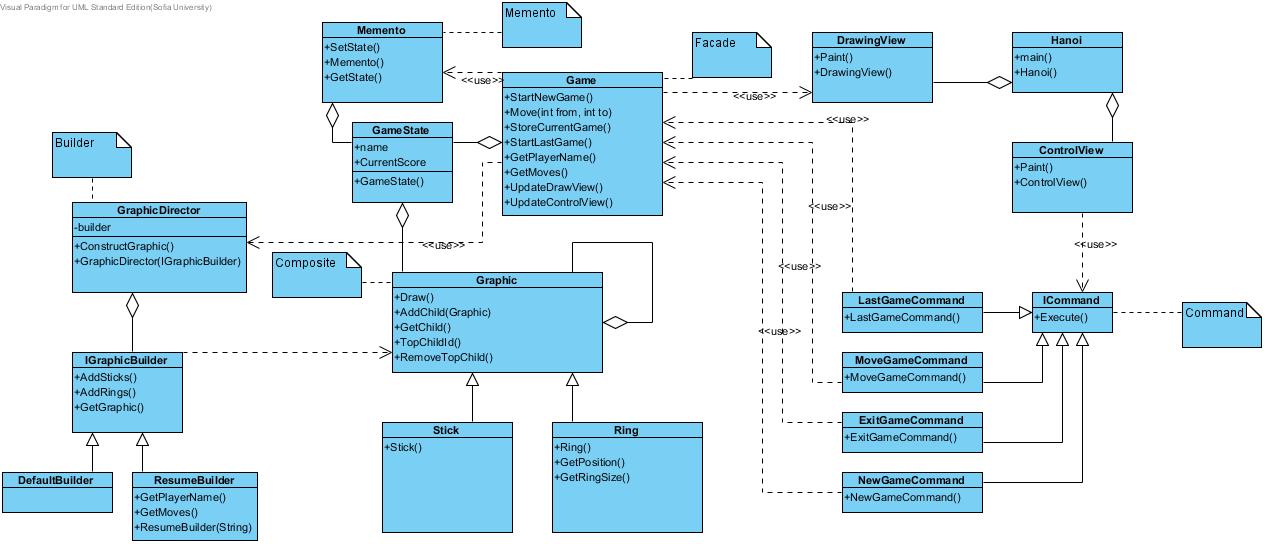
1. В приложението трябва да се използват поне 4 GoF шаблона за проектиране.
2. От всяка една група (създаващи, структурни и поведенчески шаблони) да има поне един използван шаблон.
3. Използваните шаблони да са свързани по подходящ начин като структура и поведение.

# Избор на технологии

След анализ на изискванията беше взето решение да се разработи desktop приложение. За език за програмиране е избран Java, тъй като почти всички примери в лекциите бяха разработени на Java. Това ще направи по-лесно използването на изучаваните шаблони за проектиране. За запазване на персистентни данни ще се използва xml файл, който ще се записва в папката на приложението. Потребителския интерфейс в Java е реализиран посредством Java Swing компоненти.

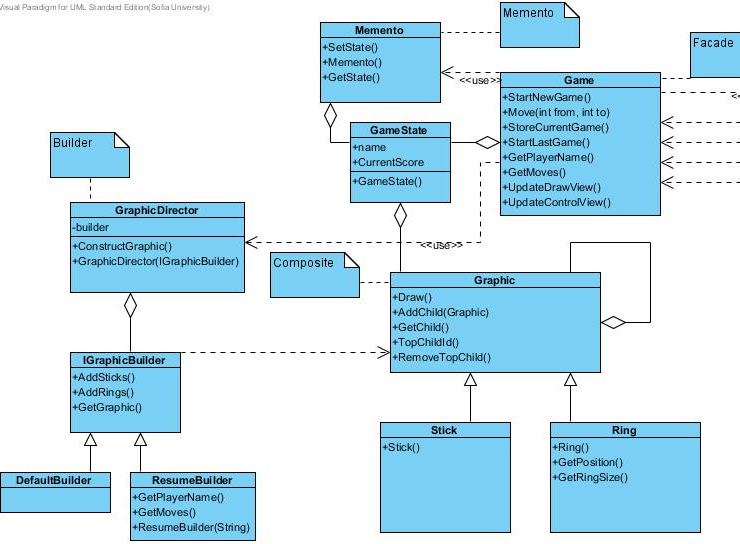
# Дизайн на системата

На фигура 1 е показана клас диаграмата, която е реализирана в приложението:

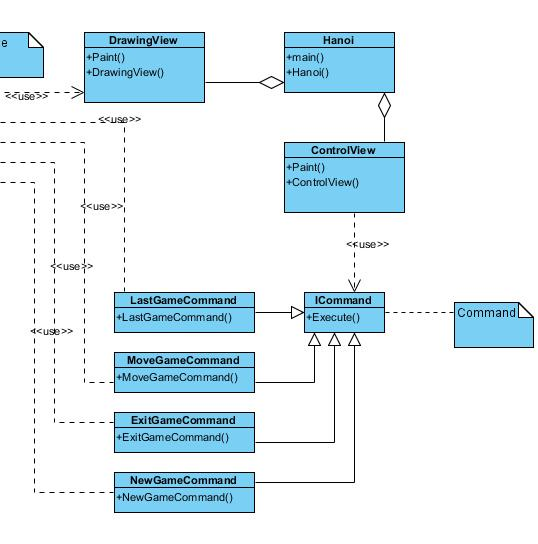


Фиг. 1 Цялостна клас диаграма на приложението.

Тъй като диаграмата е съставена от голям брой класове и трудно се побира на една страница, на фигура 2 и 3 тя е показана уголемена и отрязана съответно отляво и отдясно.

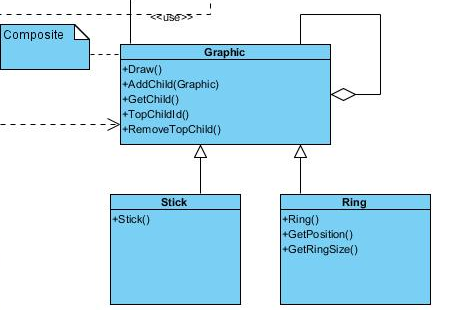


Фиг. 2 Уголемена лява част на клас диаграмата.



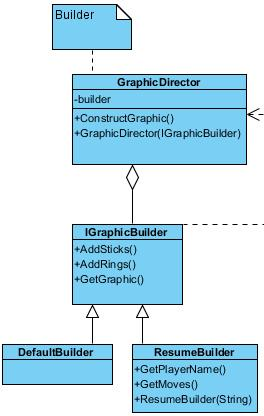
Фиг. 3 Уголемена дясна част на клас диаграмата.

Изображенията в приложението се представят като йерархична структура, която е реализирана посредством шаблона **композиция** (composite). Най-отгоре в йерархията се намира класът Graphic. Той реализира някои базови методи като addChild, removeChild и др. Също така той съдържа структура с референции към същия обект. По този начин ще се реализира обект Graphic, който ще има 3 подобекта за стълбове (stick), които от своя страна ще съдържат кръгове (ring). По този начин с едно извикване на Draw метода на базовия клас ще се изчертае цялата структура. Частта от клас диаграмата, я която е реализиран шаблона **композиция** (composite) е показана на фигура 4.



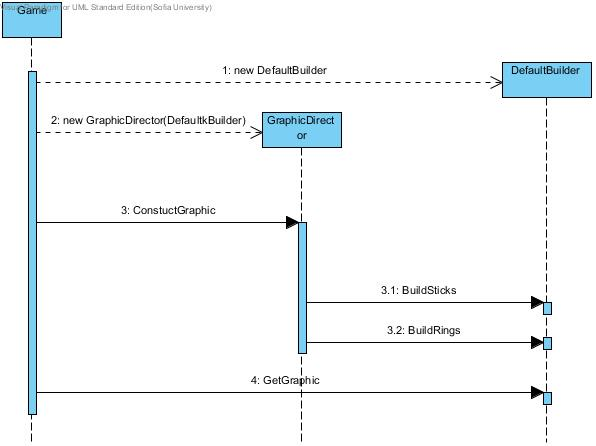
Фиг. 4 Реализация на шаблона композит.

За да се създаде тази сложна структура от данни е използват **шаблона строител**. Той има за цел да обособи създаването на структурата в отделен модул. Освен това шаблонът е реализиран посредством абстрактен клас за строител и два конкретни класа, които го наследяват. Това са **DefaultBuilder** и **ResumeBuilder**. Първият ще построи структурите в начално положение, докато втория ще изчете xml файл и ще създаде структура спрямо този файл. По този начин се зарежда последно, използваната игра. Частта от клас диаграмата, която реализира шаблонът строител е показана на Фиг. 5.



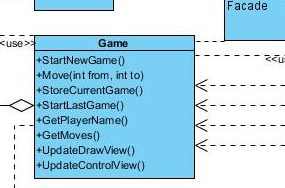
Фиг. 5 Реализация на шаблона строител.

Комуникацията между обектите при създаване е показана на фигура 6 посредством диаграма на последователността.



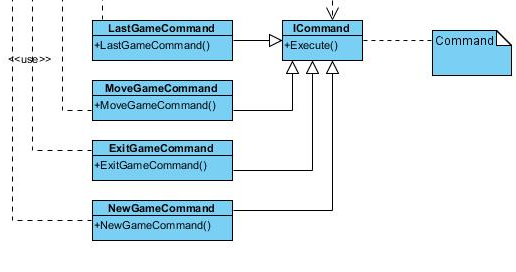
Фиг. 6 Диаграма на последователността при използване на строител.

Друг шаблон, който е използван в приложението е фасада. Класът Game служи като фасада, като за целта предоставя интерфейс от по-високо ниво, през които си комуникират компонентите отговарящи за потребителския интерфейс и останалите. Частта от клас диаграмата, която реализира фасада е показана на фигура 7.



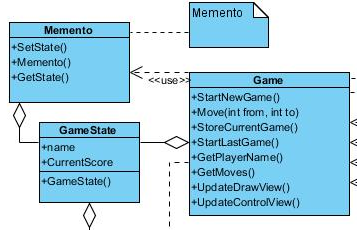
Фиг. 7 Реализация на фасада в приложението.

Поведенчески шаблони са използвани два. На първо място това е **шаблонът команда**. Той се използва за препредаване на командите по общ интерфейс към фасадата Game. Реализиран е абстрактен клас ICommand и четири команди наследници. Използването на шаблона се налага, защото при натискането на бутоните на приложението е необходимо да се изпращат заявки до обектите, без да се знае нищо за операцията. Реализацията на шаблона команда може да се види на Фиг. 8.



Фиг. 8 Реализация на команда в приложението.

Последния шаблон, който е реализиран в приложението е **Momento (спомен).** Той се използва, за да се удовлетвори изискването за персистентни данни. Чрез него се запазва текущото състояние на програмата в xml файл и след това може да се възстанови при повторно стартиране. Реализацията на спомен е показана на фигура 9.

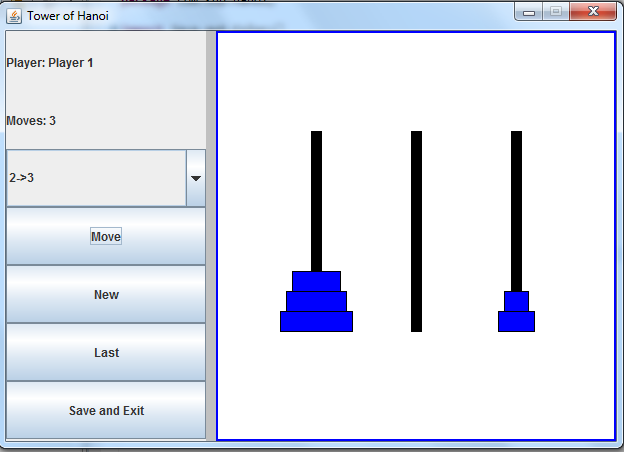


Фиг. 9 Реализация на шаблона спомен в приложението.

Това означава, че в цялото приложение са реализирани 5 шаблона за проектиране. Един за създаване, два структурни и два поведенчески. Това е с един повече от изискванията.

# Реализация на приложението

Изглед на имплементираното приложението може да се види на фигура 10.



Фиг. 10 Общ вид на създаденото приложение.

Данните, които се записват в xml файл изглеждат по следния начин:

<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<game>

<playerName>Player 1</playerName>

<moves>3</moves>

<stick>

<ring>5</ring>

<ring>4</ring>

<ring>3</ring>

</stick>

<stick/>

<stick>

<ring>2</ring>

<ring>1</ring>

</stick>

</game>