Flemming Lundahl

Group 1

Jacob Kurtzhals [au]

Tobias

Mikkel

Flemming Lundahl [au]

Software Design – Mandatory Assignment 3

Indholdsfortegnelse

[1 Chosen design pattern 2](#_Toc500154432)

[1.1 Purpose 2](#_Toc500154433)

[1.2 Type 2](#_Toc500154434)

[1.3 Structure 2](#_Toc500154435)

[1.4 Dynamics 2](#_Toc500154436)

[1.5 Consequenses 2](#_Toc500154437)

[2 Related patterns 2](#_Toc500154438)

[3 Implementation 2](#_Toc500154439)

[4 Usefulness 2](#_Toc500154440)

[4.1 Pros 2](#_Toc500154441)

[4.2 Cons 2](#_Toc500154442)

# 1 Memento Pattern

Vi har valgt at gå i dybden med patterent Memento. Igennem denne rapport skulle enhver som læser den, gerne få styr på hvad dette pattern skal bruges til, hvordan det er bygget op og hvordan man bruger det, ved hjælp af vores implementations eksempel skrevet i C#, Wishlist.

## 1.1 Formål

Formålet med memento pattern er at give muligheden for et objekt, at gå tilbage til et tidligere state. I den kommende implementation, implementerer vi en wishlist, som består af flere wishes. Ideen er så her, at vi kan ændre på vores wishlists og ved hjælp af Memento patterent, gå tilbage til et tidligere state af vores wishlist, hvis vi ikke er tilfredse med vores ændringer.

Hvordan det hele er struktureret for at gøre dette muligt, kommer vi tilbage til senere.

## 1.2 Type

Memento er et adfærds pattern, da der ikke bliver oprettet nye objekter, men ændrer på et objekt, i dette tilfælde et objekts state. Med state menes der at vi selvfølgelig kan gå tilbage til et tidligere, men også gå frem igen, hvis vi først et gået mindst et state tilbage.

### 1.3 Struktur

Memento pattern består af de tre klasser, memento, originator og caretaker. Disse har hver deres opgave, i målet om at gemme forskellige states af et objekt.

### 1.3.1 Memento

Et Memento objekt er hvad et originator objekt gemmes i, når dets state tidligere states skal gemmes.

### 1.3.2 Originator

Står for at oprette et Memento objekt, som indeholder orignatorens state. Den bruger også Memento objekter til at gendanne sig til et tidligere stadie.

### 1.3.3 Caretaker

Caretaker skal blot holde på Memento, den må ikke operere på dem.

### 1.4 Dynamics

## 1.5 Consequenses

Der skal muligvis bruges mange ressourcer, på at gemme de forskellige states.

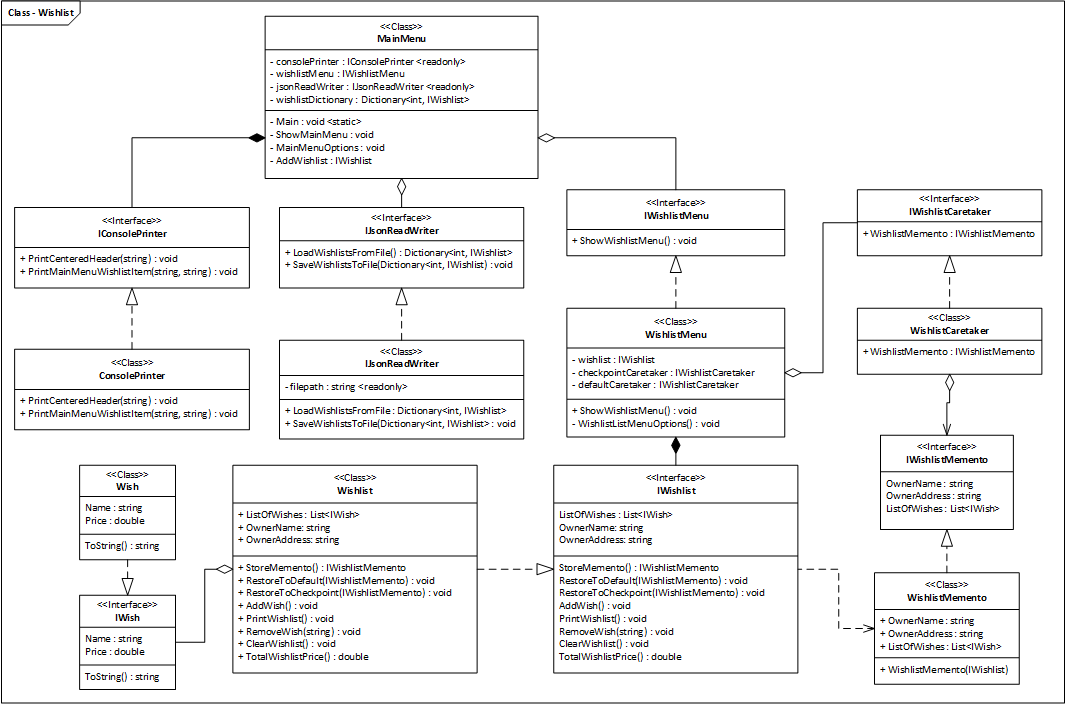
# 2 Related patterns

Command og Iterator

# 3 Implementation

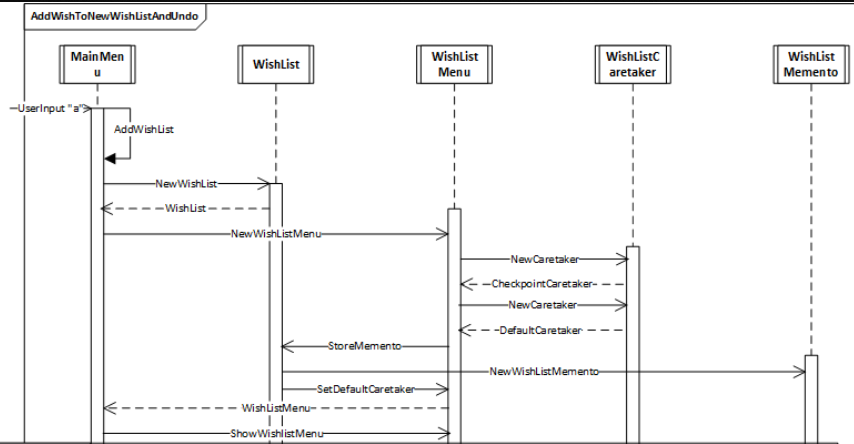
Vores implementation af Memento Pattern, er en ønskeliste. Her kan brugeren indsætte nye *wish’s* ind på sin *wishlist*. Hvis der indsættes et *wish*, som brugeren har fortrudt, kan han via en *undo* kommando, gå tilbage til det stadie, hvor dette ønske ikke var på listen.

## 3.1 Klassediagram

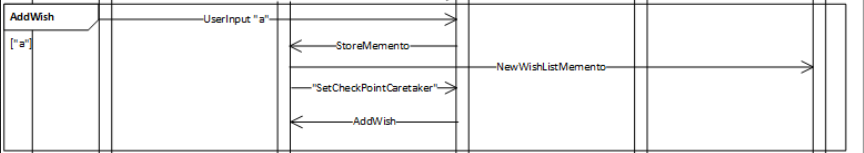


Figur . Dette klassediagram viser vores implementation af Memento pattern.

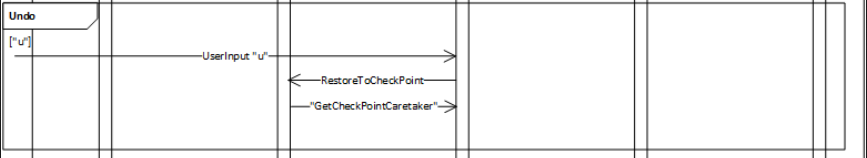
## 3.2 Sekvensdiagrammmer



### 3.2.1 Add command



### 3.2.2 Undo command



## 3.3 Originator klasse

I vores implementering er originator klassen vores wishlist. Den står for alt det praktiske, som vores wishlist skal kunne, samt at gemme vores Memento states og restore dem. Nedestående kodeudsnit, viser hvordan vi med metoden RestoreToCheckpoint, ændrer vores state, til hvad der ligger i checkpointWishlistMemento, hvilket er det senest gemte state.

public void RestoreToCheckpoint(IWishlistMemento checkpointWishlistMemento)

{

if (checkpointWishlistMemento != null)

{

OwnerName = checkpointWishlistMemento.OwnerName;

OwnerAddress = checkpointWishlistMemento.OwnerAddress;

ListOfWishes = checkpointWishlistMemento.ListOfWishes;

}

}

## 3.4 Memento klasse

Memento klassen ligner meget Originator klassen, dog uden alle funktionerne. Dette skyldes at Memento objektet gemmer det state som Originator klassen er i og derfor bliver nød til at have de samme attributter som Originator klassen.

public WishlistMemento(IWishlist wishlist)

{

OwnerAddress = wishlist.OwnerAddress;

OwnerName = wishlist.OwnerName;

if(wishlist.ListOfWishes != null)

ListOfWishes = new List<IWish>(wishlist.ListOfWishes);

}

## 3.5 Caretaker klasse

Caretaker klassen skal blot holde på de gemte Memento klasser, har derfor blot en liste af disse i sig.

public class WishlistCaretaker : IWishlistCaretaker

{

public IWishlistMemento WishlistMemento { get; set; }

}

# 4 Usefulness

Transaktions bruger Memento pattern.

## 4.1 Pros

## 4.2 Cons