Code concepts

Jake’s Restaurant

Contents

[Auteurs 3](#_Toc105536804)

[Inleiding 4](#_Toc105536805)

[Code concepten 4](#_Toc105536806)

[Functions 4](#_Toc105536807)

[Higher Order Functions 4](#_Toc105536808)

[Currying 5](#_Toc105536809)

[Closure 5](#_Toc105536810)

[Tuples 5](#_Toc105536811)

[Arrays 6](#_Toc105536812)

[Methods 6](#_Toc105536813)

[Classes 7](#_Toc105536814)

[Polymorphism 7](#_Toc105536815)

[Abstraction 7](#_Toc105536816)

[Inheritence 8](#_Toc105536817)

[Encapsulation 8](#_Toc105536818)

# Auteurs

|  |  |
| --- | --- |
| Naam | Leerling nummer |
| Jeremey Sas | 1028294 |
| Marchano Gopal | 1038165 |
| Michael Heerkens | 1028285 |
| Tobias Roessingh | 1042623 |

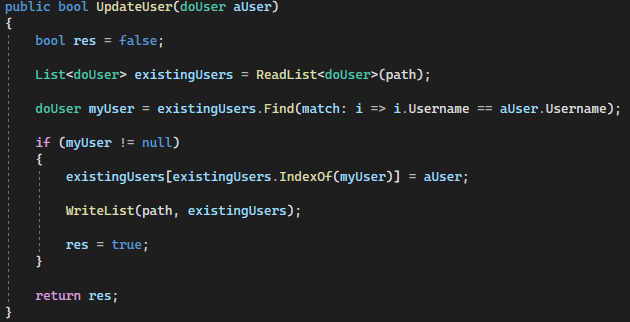
# Inleiding

Dit document is opgesteld als bewijs van interpretatie van verschillende code concepten die geleerd zijn tijdens de Development en Analysis lessen. Mochten bepaalde standaarden uit de lijst niet gebruikt zijn zullen we aangeven waarom de keuze gemaakt is dit niet te implementeren.

# Code concepten

## Functions

Het gebruik van functies is door ons in elke controller gebruikt, zie een van onze implementaties van een functie hieronder. Deze functie bewerkt een bestaande gebruiker uit het systeem, dit wordt onder andere gebruikt door het aanpassen van het profiel van een gebruiker, bijvoorbeeld het aanpassen van de voornaam van een aangemelde gebruiker.

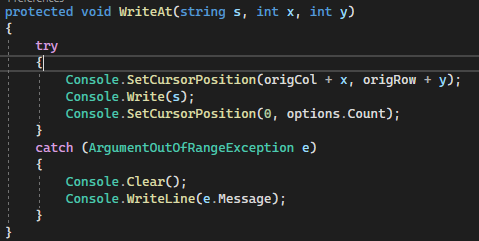


## Higher Order Functions

In onze applicatie hebben we geen gebruik gemaakt van Higher Order Functions, dit omdat in ons systeem design dit soort generieke manieren van functies niet nodig is, dit omdat alle functionaliteiten in onze Controllers van ons MVC-model staan. We hadden Higher Order Functions kunnen gebruiken voor bijvoorbeeld het uitlezen van een lijst van gebruikers en deze te filteren op basis van het door ons meegegeven filter.

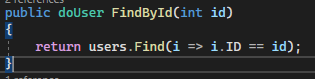
## Currying

We hebben niet veel gebruik gemaakt van code currying, maar voor ons menu hebben we dit geimplementeerd, zie de afbeelding hieronder, deze functie krijgt een string en twee integers mee om op een positie in de command line applicatie de cursor te zetten, vervolgens de string te schrijven die is meegegeven en vervolgens de cursor aan het begin van de opties te zetten.



## Closure

Het gebruik van closures in ons systeem is door ons geïmplementeerd voor het zoeken van objecten uit een lijst, zie hieronder de afbeelding van een stuk code, waarin we op basis van een gebruikers identificatie de gebruiker zoeken.

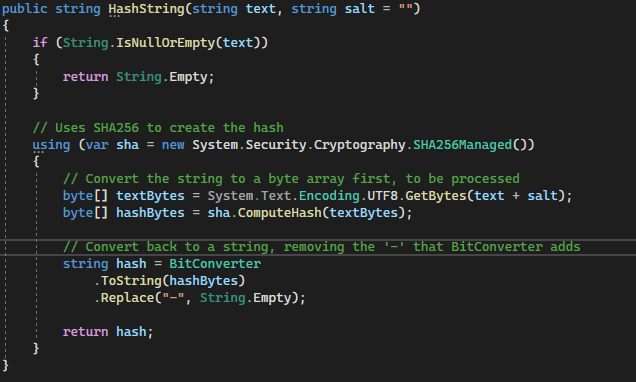


## Tuples

We hebben geen gebruikgemaakt van Tuples, dit omdat wij ervoor hebben gekozen om systeem breed gebruik te maken van de C# List. We hadden een List kunnen vervangen voor een Tuple om bijvoorbeeld de gebruikers lijst op te kunnen slaan in een Tuple van een gebruiker op basis van zijn gebruikers identificatie, door middel van een Tuple<int, DoUser>.

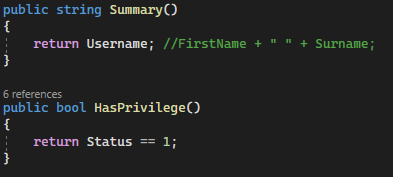
## Arrays

We hebben voor het beveiligen van de gebruikers gegevens een byte array gebruikt, zie afbeelding hieronder. In deze functie geven we een stuk tekst mee, bijvoorbeeld het wachtwoord van een gebruiker, samen met een zo genoemde salt code. Deze functie codeert daarna vervolgens het stuk tekst naar een SHA256 hash.



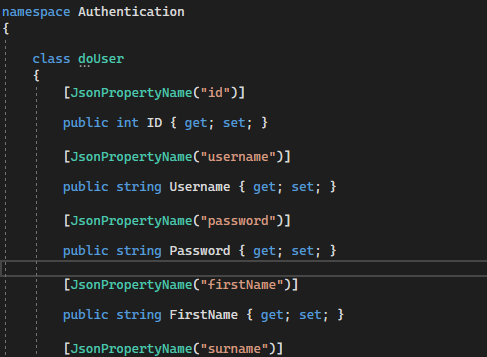
## Methods

In onze applicatie maken we gebruik van methods voor het uitlezen van bepaalde informatie van een object in een formaat dat voor elk object hetzelfde is. Zie hieronder het stuk code waar we zowel de method Summary als de method HasPrivilege hebben staan. De Summary method wordt gebruikt voor het uitlezen van alle objecten in precies hetzelfde formaat en de HasPrivilege methode wordt gebruikt voor het bepalen of de aangemelde gebruiker wel of geen administrator rechten heeft.



## Classes

In ons systeem hebben we uitgebreid gebruik gemaakt van classes, omdat we voor ons MVC-model voor al onze objecten een class hebben gemaakt. Neem in de afbeelding hieronder een voorbeeld van de gebruiker class ‘DoUser’



## Polymorphism

We hebben in ons systeem geen gebruik gemaakt van Polymorphism, dit waren we eerst van plan voor de overerving tussen klant en administrator, dit idee hadden we doorgenomen met de product eigenaar, maar deze had ons afgeraden dit te doen, omdat dit te ingewikkeld zou worden voor een simpel verschil.

Ook kunnen we dit in de applicatie integreren bij het toevoegen van een drank kaart, waar dan drank en voedsel aparte objecten zijn, maar polymorfische eigenschappen heeft, waardoor deze op dezelfde manier uitgelezen en besteld zou kunnen worden.

## Abstraction

We hebben in ons systeem geen gebruik gemaakt van abstraction, dit omdat we nergens een nuttige implicatie zagen met het huidige systeem ontwerp.

Om abstractie in het systeem te verwerken hadden we de distinctie in klant en administrator kunnen gebruiken, door hier een algemene abstracte gebruiker class voor te maken en hiervan twee overervingen te maken die in de basis beide aan moeten kunnen melden, profiel gegevens moeten kunnen weergeven en basisinformatie moet kunnen bevatten. Maar apart wel verschillende functionele als informatieve gegevens en methodes kan hebben.

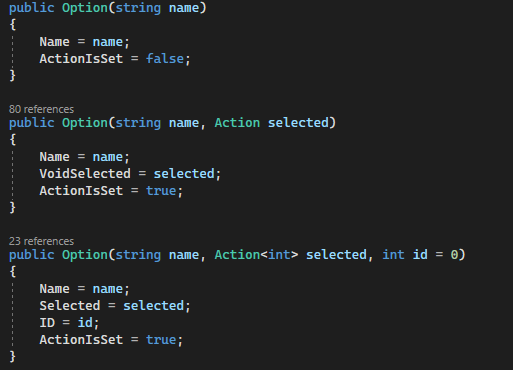
## Inheritence

In ons systeem hebben we inheritence gebruikt voor een view die alleen gebruikt kan worden door de administrator. Op deze manier hadden we de basis elementen van de normale view, die ook voor een klant beschikbaar zijn en konden we vervolgens voor de administrator methodes overschrijven. Zie afbeelding hieronder.



## Encapsulation

Voor onze menu items hebben we gebruik gemaakt van Encapsulation, dit door een class Option te maken, die intern een naam krijgt, maar vervolgens ook aangemaakt kan worden met een void callback functie of met een integer parameter callback met een bepaalde integer als identificatie en gebruik voor deze integer parameter callback. Zie abeelding hieronder voor de verschillende manieren van het construeren van de class Option.



Vervolgens heeft deze class op basis van zijn gegevens een ander gedrag in de applicatie, zo wordt de class Option met alleen een naam weergeven voor details, waar een gebruiker verder geen actie bij uit kan voeren. Daarentegen wordt de Option met een naam en een lege action gebruikt voor bijvoorbeeld de menu items om door het systeem te navigeren en vervolg menu’s te openen. Ten slotte wordt de class Option met een naam, een integer parameter action en een integer ID gebruikt voor bijvoorbeeld het selecteren van een specifiek item uit een lijst van meerdere items, neem hierbij een voorbeeld aan het selecteren van een specifieke gebruiker uit de gehele lijst.