|  |  |
| --- | --- |
| Reflectie-verslag  Leertaak 2: Project Vossen & Konijnen | oVER  Het reflectieverslag van het project Vossen & Konijnen. Dit project wordt uitgevoerd in periode twee van het eerste jaar.  Door  Frank Noorlander, Rick van der Poel en Tsjeard de Winter |

Reflectie-verslag

Leertaak 2: Project Vossen & Konijnen



**Auteurs:** Frank Noorlander, Rick van der Poel en Tsjeard de Winter

**Opdrachtgever**: Dhr. Heijnen

**Organisatie**: Hanzehogeschool Groningen

**Datum** 24-01-2015

**Locatie**: Groningen

# Samenvatting

Inhoudsopgave

[Samenvatting 2](#_Toc409877482)

[Verklarende woordenlijst 4](#_Toc409877483)

[Voorwoord 5](#_Toc409877484)

[probleemstelling 6](#_Toc409877485)

[Analyse huidige situatie 6](#_Toc409877486)

[Hoofdstuk 1 7](#_Toc409877487)

[Hoofdstuk 2 8](#_Toc409877488)

[Hoofdstuk 3 9](#_Toc409877489)

[Conclusie 10](#_Toc409877490)

[Advies / aanbevelingen / actieplan 11](#_Toc409877491)

[Noten 12](#_Toc409877492)

[Literatuurlijst 13](#_Toc409877493)

[Bijlage 1 14](#_Toc409877494)

[Bijlage 2 15](#_Toc409877495)

[Register 16](#_Toc409877496)

# Verklarende woordenlijst

# Voorwoord

Voor u ligt het reflectieverslag van het project Vossen & Konijnen. Het project wordt uitgevoerd in de tweede periode van het eerste schooljaar, in opdracht van de Hanzehogeschool. De projectleden zijn: Frank Noorlander, Rick van der Poel en Tsjeard de Winter

Dit verslag zal een reflectie zijn op de werkzaamheden die we als projectgroep hebben verricht om dit project tot een succes te brengen. Meer informatie over het project vindt u in de volgende hoofdstukken.Inleiding

# Probleemstelling

Onze opdrachtgever; de Hanze Hogeschool heeft veel verzoeken gekregen om een applicatie te maken waarmee het de levenscyclus van dierenpopulaties kan simuleren. Omdat er tegenwoordig veel meer aandacht is voor de natuur en het behouden ervan zijn er verschillende partijen geïnteresseerd in zo’n applicatie.

De opdrachtgever heeft zelf al een poging gedaan om een dergelijke applicatie te maken. Deze applicatie genaamd ‘*Vossen & Konijnen’* voldoet echter niet aan de eisen om een goede simulatie te genereren.

Als projectgroep hebben we de taak gekregen om de huidige applicatie van de opdrachtgever aan te vullen en te verbeteren om tot een realistisch resultaat te komen. De meegegeven applicatie is momenteel erg simpel, dat betekend dat we ons momenteel alleen zullen gaan richten op een bosgebied met daarin vossen en konijnen. Later tijdens dit project zal de applicatie echter uitgebreid worden met meer functionaliteiten.

Als de applicatie een succes wordt, kan het gebruikt kunnen worden om veel verschillende dierenpopulaties te simuleren en daarmee een enorme schat aan informatie te verkrijgen. Deze informatie kan op zijn beurt helpen de natuur en haar inwoners te behouden.

# Analyse huidige situatie

In de huidige versie van het programma dat hebben gekregen van de opdrachtgever is het mogelijk om via een visuele simulator het leven van vossen en konijnen te simuleren.

De vossen en konijnen hebben de volgende eigenschappen:

* Beide dieren hebben een maximale leeftijd. Als deze leeftijd overschreden wordt zullen de dieren verdwijnen uit het programma.
* Beide dieren hebben de mogelijkheid om te paren.
* De vossen kunnen de konijnen opeten als deze naast elkaar komen te staan in de simulator.
* De vossen kunnen dood gaan als ze te lang geen konijnen gegeten hebben.

Zoals hier boven al blijkt, is de applicatie nog erg kaal. De enige overlevingscondities waar de applicatie momenteel rekening mee houdt zijn de leeftijd en de honger. Er wordt bijvoorbeeld geen rekening gehouden met weersomstandigheden, andere dieren die hoger in de voedselketen staan zoals beren of jagers, vervuiling enzovoort.

Het doel van deze applicatie is om de levenscyclus van dierenpopulaties zo goed en realistisch mogelijk te simuleren. Om dat doel te kunnen realiseren moet de applicatie uitgebreid worden met nieuwe elementen.

Verbeteringen aan het programma

MVC

De applicatie zoals wij deze aangeleverd kregen bevatte naast het gebrek aan functionaliteit ook gebrek aan flexibiliteit. Wanneer er een functie toegevoegd of juist gewijzigd moest worden moest er teveel veranderd worden waardoor dit veel tijd in beslag nam. Wij hebben ervoor gekozen om het programma flexibeler te maken door het MVC-model toe te passen. Door dit model veranderd er niks aan het uiterlijk van de applicatie maar is wel duidelijk merkbaar tijdens het wijzigen of toevoegen van functies.

**Wat is MVC?**

MVC staat voor *Model, View, Controller* en is een ontwerppatroon dat het ontwerp van toepassingen opdeelt in drie onderdelen met elk hun eigen verantwoordelijkheden. Door het opdelen wordt de leesbaarheid en herbruikbaarheid van de code vergroot.

Het belangrijkste onderdeel is het *model*. Deze zorgt voor de informatie door de applicatie logica toe te passen op de data. Het model weet welke *views* bij hem horen en geeft steeds een melding aan al deze *views* dat de informatie veranderd is.

De *view* zorgt dat de informatie op het scherm getoond wordt. Wanneer de *view* gemaakt wordt, geeft deze door aan een *model* dat hij van zijn informatie afhankelijk is. De *view* doet zelf geen verwerking van de gegevens en geeft ook niks aan het *model* door.

De *controller* zorgt voor de invoer van gebruikers. Alle knoppen, tekstvelden, menu’s, etc. zijn dus van de controller. De controller wordt aan één of meerdere *models* verbonden en geeft elke gebruikers invoer direct door aan het *model* zonder iets met de informatie te doen.

Wanneer deze drie onderdelen goed geïmplementeerd zijn kan je heel makkelijk een nieuwe view toevoegen of weghalen zonder dat de applicatie daardoor niet werkt. Maar dit geldt ook voor controller en view hoewel deze vaak wat in elkaar verwikkeld zijn, kunst is om dit zo beperkt mogelijk te houden.

**MVC in Vossen en Konijnen**

In onze applicatie moest er op minstens 5 plekken code gewijzigd worden voordat we een nieuwe view toe konden toevoegen. Een nieuwe view zou bijvoorbeeld een cirkel- of staafdiagram kunnen zijn. Daarom is er in

Hoe kan V&K de MVC structuur krijgen?

Klein stukje over threads > wat zijn threads? Voordeel van gebruik in V&K?

*N.B. Het belangrijkste doel van MVC is om de verschillende onderdelen van de applicatie los te koppelen van elkaar. Een goede check om te kijken of je MVC implementatie correct is, is om te kijken of je een nieuwe view of controller kunt toevoegen zónder dat je het model moet aanpassen.*

The reason because it is not working right now is that the Java thread that handle the Gui refreshing also handles the events of the Listeners. So when you call the setText() method it tells the Gui thread (called EDT for Event Dispatch Thread) to update the Component, but it can't be done right now because the EDT is currently in the actionPerformed() method executing your code.

So I think you should put the code that do whatever work and change the text of your JLabel in a new thread. So the EDT starts it in the actionPerformed() and is then free to update the GUI when the text of your JLabel changes.

Something like this :

public void actionPerformed(ActionEvent e) {

Thread t = new Thread(new Runnable() {

myLabel.setText("Processing");

//Do the job

myLabel.setText("Processed");

});

t.start();

}

Ideally the setText() method and others that alter the component have to be called from the EDT itself to avoid bugs... This is not the case in the example I gave you. If you want to do it use this method :

SwingUtilities.invokeLater(new Runnable() {

public void run() {

myLabel.setText("my text");

}

};

Verplichte uitbreidingen

Bonus uitbreidingen

Conclusie

Advies / aanbevelingen / actieplan

Noten

Literatuurlijst

Bijlage 1

Bijlage 2

Register