|  |  |
| --- | --- |
|  | Project parkeersimulator  Door: Sam van Putten, Dave Nienhuis, Zerin Sulejmanovic, Niels Wildeboer en Alex Atacho  **Project Design & Build 1.3** |

**Inhoudsopgave**

[Probleemanalyse 1 2](#_Toc474013845)

[Probleemanalyse 2 3](#_Toc474013846)

[Model View Controller 4](#_Toc474013847)

[UML-diagram (voor aanpassingen) 5](#_Toc474013848)

[De belangrijkste uitbreidingen 6](#_Toc474013849)

[Reflectie CityParking-project 7](#_Toc474013850)

[Parkings (Project parkeergarage) 7](#_Toc474013851)

[Project reflectie 8](#_Toc474013852)

# Probleemanalyse 1

Deze week zijn wij bezig geweest met de parkeersimulator in zijn huidige vorm. De volgende zaken vielen ons op:

* Er is geen mogelijkheid voor abonnementhouders om bepaalde delen van de parkeergarage voor een bepaalde tijd te boeken. Er is altijd een wisseling van blauwe en rode auto’s gaande, er moet ook een mogelijkheid komen waar een auto of een groep auto’s meerdere ticks kan blijven staan.
* In de SimulatorView zijn geen knoppen aanwezig; hij kan niet gepauzeerd of gestopt worden.
* Er is een verdieping waar geen gebruik van wordt gemaakt. De simulator reikt maar tot de tweede rij van de tweede verdieping. Meerdere keren testen heeft geresulteerd in ongeveer hetzelfde resultaat.
* Er staat helemaal geen tekst in de SimulatorView. Het is voor de kijker onmogelijk om te zien hoeveel auto’s er per dag binnenkomen of hoeveel er op een bepaald moment in de garage staan.

Hier heeft het bedrijf uiteraard niet veel aan. De code moet zo aangepast worden dat de SimulatorView knoppen heeft die het programma kunnen manipuleren: bijvoorbeeld een pauzeknop, een knop om de simulatie met 1 tick per keer te laten verlopen, etc. Ook moet er een log bij waarin staat hoeveel auto’s er per dag binnenkomen, hoeveel auto’s er op het huidige tijdstip staan, etc.

# Probleemanalyse 2

* **Als ik het programma meerdere keer draai, krijg ik dan hetzelfde resultaat? Is dat goed of slecht?**

Als je hem meerdere keren draait krijg je steeds een ander resultaat en in mijn ogen is dat een stuk realistischer dan elke keer hetzelfde resultaat. In het echte leven parkeren mensen ook niet altijd op hetzelfde plekje elke dag.

* **Is er een patroon te herkennen in het resultaat?**

Niet echt, alleen dat dat de parkeergarage van voor af aan vult en niet ineens ergens bij het einde een auto heeft staan.

* **Is dit ook wat ik in de werkelijkheid verwacht?**

Nee, in werkelijkheid zijn er ook auto’s die niet goed naast elkaar passen en daarom aan de andere kant van de parkeergarage parkeren omdat ze daar meer ruimte hebben of omdat daar de uitgangen toegankelijker zijn.

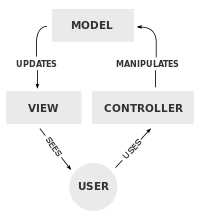
* **Is de simulator eenvoudig uit te breiden/te onderhouden? Wat zijn de beperkingen?**

Omdat alle classes goed verspreid zijn over losse Java bestanden is het makkelijker te onderhouden en te gebruiken. Als je dan aanpassingen wil doen hoef je niet bang te zijn dat je andere classes hiermee beïnvloed. Hier zijn volgens mij geen beperkingen aan.

* 1. **Wat kan ik op internet vinden over dit soort optimalisatie problemen?**

Er staan veel informatieve filmpjes over coupling op Youtube. Deze zullen van pas komen tijdens het coderen.

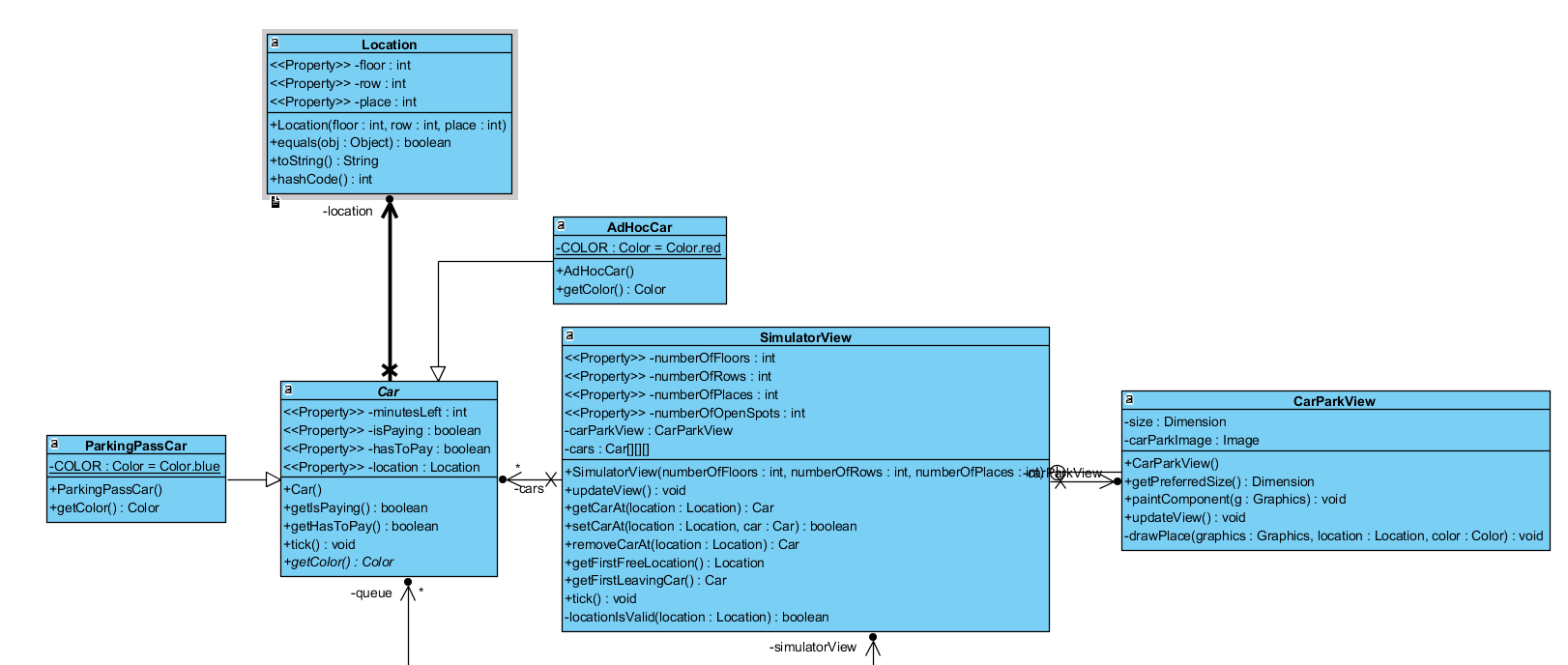
# Model View Controller

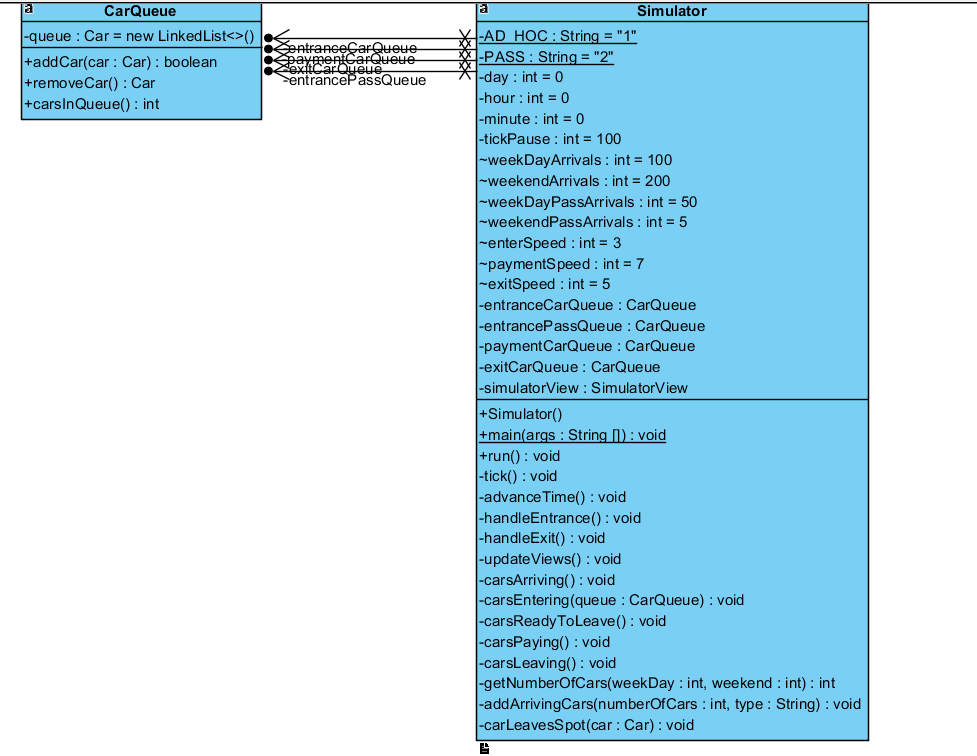


Dit is een representatie van hoe een Model View Controller werkt. In principe is het zo: je gebruikt JFrame of JPanel om een View aan te maken waarin je Buttons aanmaakt. Deze View krijgt input van de simulator. Wanneer de user op een van de Buttons klikt (controller) wordt het View geüpdate, wat de User kan zien. In het geval van de parkeersimulator willen wij een pauzeknop aanbrengen, alsook een knop om de ticks handmatig te laten verlopen (met 1 of 10 stappen).

In het project zijn we al bepaalde zaken tegengekomen die niet kloppen met het MVC-model. Zo zit de main-methode al in simulator, is er geen apart model/controller etc.

# UML-diagram (voor aanpassingen)





# De belangrijkste uitbreidingen

Zoals eerder vermeld moest de simulatie aan bepaalde eisen voldoen: een speciale verdieping hebben voor abonnementhouders, knoppen aan de SimulatorView toevoegen en er staat ook geen tekst in de SimulatorView waardoor het onmogelijk is om te zien hoeveel auto’s er per dag binnen komen en hoeveel er op een bepaalde moment in de garage geparkeerd zijn.

Tot nu toe hebben we knoppen aan de simulatie toegevoegd en de main methode. Ook is er een onderscheid van auto’s toegevoegd van abonnees en adhoc auto’s. Deze auto’s hebben hun eigen kleur gekregen. Dit is het lijst van werkende knoppen in de simulatie:

* 1. Start: start de simulatie
  2. Minuut: gaat minuut verder
  3. Uur: gaat een uur verder
  4. Pauze: pauzeert de simulatie
  5. Stop: stopt de simulatie

Er is tekst aan de SimulatorView toegevoegd, zodat je een betere beeld krijgt van de simulatie, qua kosten en auto’s. Dit is een lijst van de tekst dat te zien is:

* 1. Totaal aantal auto’s
  2. Aantal AdHocCars
  3. Aantal ParkingpassCard
  4. Aantal AbboCars
  5. Totaal opbrengst
  6. Aantal minuten

Reflectie verslagen

Reflectie verslag: Sam van Putten

# Reflectie CityParking-project

Deze periode moesten wij voor het vak Project Design & Build een java-applicatie van een parkeergarage verbeteren. Dit was een vrij lastig proces; doordat de vakken OOP1 en OOP2, waarvoor volgens onze docent een half jaar uitgetrokken moest worden, in 8 weken werden gegeven was het moeilijk voor mij om de stof goed op te nemen. Hierdoor kon ik ook niet goed meekomen met het huiswerk: na week 1 liep ik al vrij snel achter. De moeilijkheidsgraad steeg zeer snel. Dat zorgde wel voor moeilijkheden bij het project: bij bepaalde problemen wist ik niet wat ik moest doen omdat ik nog niet tot een bepaald punt van OOP2 was gekomen.

Ik ben wel van mening dat BlueJ en Eclipse fijne programma’s zijn om in te werken en heb ondanks de snelheid aardig wat opgestoken van dit vak. Ik hoop wel dat in de toekomst vergelijkbare vakken niet in veel kortere tijd worden gegeven dan eigenlijk nodig is.

**Bijdrage Sam:**

* Knoppen start, step, pause, en quit toegevoegd. Start zorgt ervoor dat de app gaat runnen middels een booleanvariable *run* (de standaardwaarde is false, wanneer je op start klikt wordt deze true en alleen wanneer de variabele true is kan de simulator lopen). Pause stelt de booleanwaarde weer in op false, dus wanneer er op deze knop wordt geklikt pauzeert de simulator. Step zorgt ervoor dat het model eerst pauzeert – daarna wordt pas een tick gegeven. Het leek ons dat het weinig zin had om een extra tick toe te voegen wanneer het programma nog liep, dat zou ervoor zorgen dat er twee minuten in kortere tijd werden afgespeeld. Tenslotte heb ik een quit-knop aangemaakt; wanneer op deze knop wordt geklikt wordt het venster met de simulator gesloten.
* Start() uit main methode verwijderd; het model start nu pas wanneer er op de startknop wordt gedrukt.
* Nieuwe klasse AbboCar (auto’s met abonnement) aangemaakt.
* Velden aangemaakt voor het aantal auto’s. Cars is het totaalaantal: de simulator is zo ingesteld dat AdHocCar + ParkingPassCar + AbboCar het totaal aantal auto’s weergeeft in de simulator. Verder is in carsEntering ingesteld dat er elke keer als er een bepaald type auto binnenkomt deze bij dat type wordt opgeteld (bijvoorbeeld als er 3 AdHocCars, 2 ParkingPassCars en 1 AbboCar de garage inkomen krijgen de labels voor de respectievelijke auto’s dat aantal en worden deze aantallen voor de abstracte klasse Car opgeteld en weergegeven, dus Cars = 6 in dit geval.). Ook zijn er JLabels gemaakt die deze aantallen binnen de simulator weergeven. Tenslotte zijn binnen de updateView-methode methoden toegevoegd die het aantal auto’s per tick ophaalt, waardoor het correcte aantal constant in de simulator wordt weergegeven.
* ActionListeners toegevoegd in klasse Controller en klasse MainWindow. Hierdoor krijgen de knoppen werking.
* CarParkView uit SimulatorView gehaald en in een aparte klasse gestopt, dit ter bevordering van de MVC-structuur.

Reflectie verslag: Dave Nienhuis

# Parkings (Project parkeergarage)

Ik vond, net zoals mijn teamgenoten dit project een hele uitdaging. Dit komt omdat de JAVA kennis gewoonweg onder de maat was om dit project goed uit te kunnen voeren. Verder heb ik wel veel geleerd tijdens dit project betreffende JAVA en samen werken met code in een groep. De taakverdeling van de groep was opzich wel goed, de één was iets beter in JAVA dan de ander dus die waren voornamelijk bezig met het coderen en de mensen die minder waren in het codeer gedeelte gingen meer bezig met de verslagen die bij het project horen.

Ook hoop ik dat volgende periode iets minder snel gaat qua de leerstof van bepaalde vakken ( in dit geval OOP 1/2) zodat je meer tijd hebt om de opdracht goed te maken en daardoor beter te snappen wat je nou precies aan het doen bent met de code.

Reflectie verslag: Zerin Sulejmanovic

# Project reflectie

Het project parkeergarage Cityparking Groningen vond ik persoonlijk een erg lastig project. Voor dit project moesten we een simulatie maken van een parkeergarage. De reden dat ik dit project pittig en lastig vond is omdat de vakken OOP1 en OOP2 in een hele korte tijd aan ons uitgelegd waren. Volgens de docent hadden wij hier een half jaar voor nodig, maar wij hebben hiervoor maar acht weken gekregen. Ten eerste is dit een korte periode om alle lesstof door te nemen en het te begrijpen. Ten tweede was de uitleg die wij gekregen hebben tijdens de lessen matig. Verder vond ik ook de methode van het boek BlueJ niet slim aangepakt, het boek begint niet met de basis uitleg, maar je moest meteen werken met een script zonder dat je weet wat het precies inhoudt. Dit is bij het vak PHP anders aangepakt. We waren als eerst begonnen met het leren van de basis en zo gingen we verder, stap voor stap. Met het vak OOP werden we volgens mij gevoel in het diepe gegooid, want zoals ik eerder genoemd had de lessen waren versneld en de methode werkte niet goed voor mij. Zelf heb ik hierdoor niet veel inbreng gehad met het coderen, puur omdat ik het niet begreep. Geprobeerd heb ik het wel, maar zonder positief resultaat.

Ik hoop door mijn ervaringen van het project uit periode een en periode twee, dat periode drie een betere aanpak zal hanteren dan periode twee.

**Bijdrage Zerin:**

* Grotendeels rapport gemaakt
* Tijdens de groepsbijeenkomsten proberen te helpen
* Emotionele steun geweest

Reflectie verslag Alejandro Atacho.

# Reflectie verslag Parkings

Voor dit project vond ik dat we heel onvoorbereid waren met Java om het te doen. Denkt wel dat dit project veel leuker/beter zou gaan als we meer van Java wisten. Het was heel veel worstelen om alleen één stap te nemen op het project. Ik vond dat 2de periode overhaast was om met een project zoals dit te laten doen sommige mensen waren beter dan anderen in programmeren dankzij ervaring van vorige opleiding dus die konden veel meer doen vergeleken met mensen die pas nieuw waren. Periode 1 was tenminste te doen ok al was je niet goed in PHP en HTML een beetje rond kijken op google en video’s bekijken had je wel veel geleerd vergeleken met project 2. Wat wel een voordeel van de project was dat we hebben leren samenwerken beter, maar anders dan dat gaf deze project alleen maar stress en depressie aan anderen. Ik hoop dat periode drie niet ons op straat gooit net als periode twee.

**Bijdrage Alejandro:**

* Tijdens de groepsbijeenkomsten proberen te helpen met informatie zoeken
* Helpen denken/zoeken van codes om errors te fixen
* Andere groepen checken om te zien hoe achter/ver we waren en soms voor tips vragen