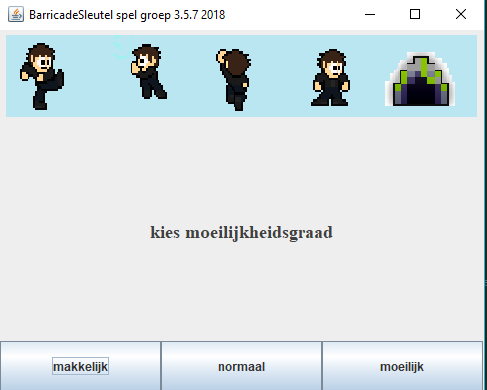
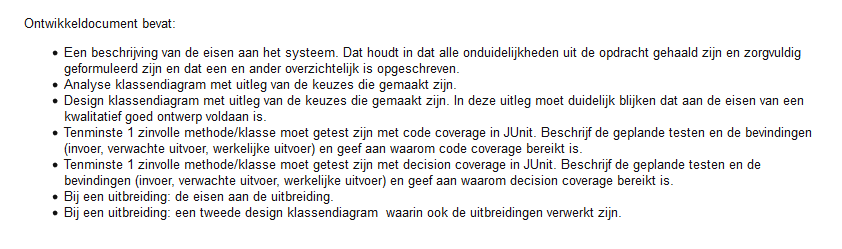
**Ontwikkeldocument Groep 3.5.7**

Door: Mucahit Coskun, Jurian van Hoorn en Taric Sherif

Datum: 03/04/2018



Inhoudsopgave:



Requirements:

Beschrijving van de eisen aan het systeem:

Analyse klassendiagram:

Design klassendiagram:

Toelichting bij ontwerp:

JUnit code coverage:

JUnit decision coverage:

Requirements:

De requirements voor dit project hebben wij uit de Projectbeschrijving gehaald die op Blackboard staat. Het betreffende document vindt U [Hier](https://drive.google.com/open?id=1TwJtHmyBijFDir-xdbuBaEORXwvyymd0). Wij hebben dit document bestudeerd en aan de hand van de informatie die wij uit het document konden halen de beschrijving van de eisen opgesteld.

Beschrijving van de eisen aan het systeem:

Wikipedia citaat:”De **MoSCoW-methode** is een wijze van prioriteiten stellen in onder meer de [software engineering](https://nl.wikipedia.org/wiki/Software_engineering). De eisen aan het resultaat van een [project](https://nl.wikipedia.org/wiki/Project) worden ermee ingedeeld. Het is een afkorting, waarvan de letters staan voor:[[1]](https://nl.wikipedia.org/wiki/MoSCoW-methode#cite_note-1)

* **M** - **must haves**: deze eisen ([*requirements*](https://nl.wikipedia.org/wiki/Requirement)) *moeten* in het eindresultaat terugkomen, zonder deze eisen is het product niet bruikbaar;
* **S** - **should haves**: deze eisen zijn zeer gewenst, maar zonder is het product wel bruikbaar;
* **C** - **could haves**: deze eisen zullen alleen aan bod komen als er tijd genoeg is;
* **W** - **won't haves**: deze eisen zullen in dit project niet aan bod komen maar kunnen in de toekomst, bij een vervolgproject, interessant zijn.“

Hieronder staat de beschrijving van de eisen aan het systeem:

must have

* Een speelveld in een vierkant vlak
* Speler die kan lopen naar het eindveld
* Een speelveld heeft de muren
* Een speelveld heeft de barricades met het nummer
* Een speelveld heeft de sleutel met het nummer
* Speler kan barricade openen met passende sleutel
* Speler kan eerst sleutel pakken
* Speler kan sleutel meerdere keren gebruiken
* Speler kan één sleutel in zijn zak hebben
* Als speler een sleutel pakt, verdwijnt het uit het veld, dan zit sleutel in de zak van de speler
* Je krijgt een melding als een sleutel niet past
* De barricade verdwijnt als de sleutel wel past
* Speler wordt verplaatst d.m.v. pijltjestoetsen
* Halverwege spel moet het mogelijk zijn hetzelfde speelveld opnieuw te starten
* Pop up scherm wanneer speler succesvol is begeleid naar het eindpunt

Vervolg op volgende pagina..

* Spel heeft knoppen om spel te starten, een spel te kiezen en een +spel te stoppen
* Speler loopt niet door een muur heen
* Speler loopt niet uit het speelveld
* Spel heeft een moeilijkheidsgraad

could have

* Een uitbreidbare functie applicatie
* De afbeeldingen voor een speler
* Speelveld heeft een willekeurige spelobjecten.

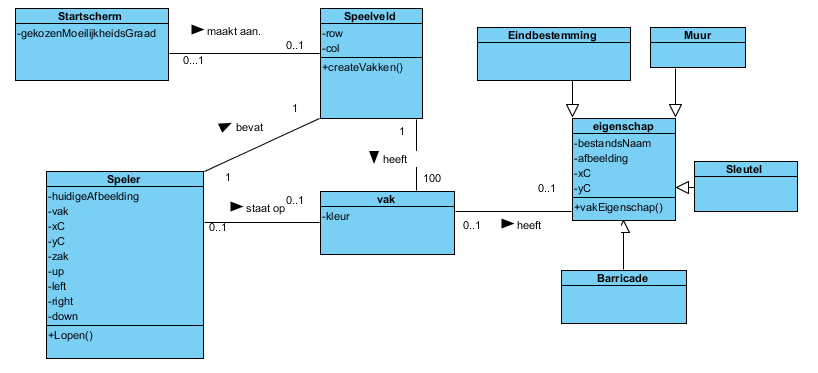
should have

* N.v.t

wont have

* N.v.t

Analyse klassendiagram:



Dit is ons Analyse Klassendiagram

Design klassendiagram:

Toelichting bij ontwerp:

JUnit code coverage:

JUnit decision coverage: