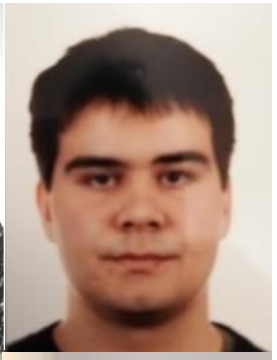
02312, 62531, 62532

INDLEDENDE PROGRAMMERING, UDVIKLINGSMETODER TIL IT-SYSTEMER OG VERSIONSSTYRING OG TESTMETODER

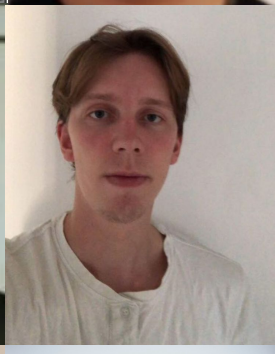
CDIO 3

Krzysztof Kisiel
s192884



Oliver Olsen
s205443

Johannes Jensen
s205413



Gustav Fauser
s205446

Jonathan Hansen
s205415

Alaa Hadood
s205477

Timeplan

Tid i timer	Krav og analyse	Design	Dokumentering af Implementering	Test	Programmering
Navn					
Gustav Fauser					
Oliver Olsen					
Krzysztof Kisiel					
Johannes Jensen					
Jonathan Høj					
Alaa Hadood					

Summary

Indholdsfortegnelse

Introduktion

IOOuterActive (os) har fået til at opgave at designe og implementere et Monopoly Junior spil. Der skal implementeres de væsentligste elementer fra brætspillet, for at spillet kan spilles. Spillet skal kunne spilles mellem 2-4 spillere.

Krav

For at spillet lever op til kundens forventninger opstilles der en række krav som spillet skal leve op til. Kravene er opstillet ud fra kundens vision(CDIO del 3, 2020, stbn)

1. Få et beløb når du passerer start feltet.
2. Tag chancekort læg det nederst i bunken.
3. Implementering af forskellige chancekort (beslutter vi sammen).
4. Fængsel betal for at komme ud eller brug "Get out of jail" kortet.
5. Parkering / Besøg i fængsel.
6. Der skal være mulighed for at købe et felt hvis man lander på det.
7. Hvis en spiller ejer et felt, skal andre spillere betale husleje, når det lander på dette.
8. Hvis spilleren ejer flere af samme farve felt koster det mere.
9. Man skal kunne definere sin brik(farve/type/pattern/fill).
10. Man skal kunne se hvem der har købt ethvert felt.
11. Når en spiller lander på et felt, skal de fortsætte fra det felt på næste tur.
12. Spillet skal spilles af 2 til 4 spillere.

Analyse

Der skal udarbejdes følgende artifacts (udover kravliste):

- Use case diagram
- Eksempler på use case beskrivelser - vælg mindst én, der beskrives fully dressed
- Domænemodel
- Et eksempel på systemsekvensdiagram
- Et eksempel på sekvensdiagram
- (Et) Designklassediagram

Der er vigtigt, at der er tydelig sammenhæng mellem beskrivelser og diagrammer. Det skal således være muligt at trace fra kravliste til analysedokumentation, designdokumentation og implementering. Beskriv ligeledes jeres brug af GRASP-mønstre.

Use cases

- Spil spil
- Kast terning
- Vælg farve/brik

Domænemodel

GRASP-mønstre

Klasser

For at lave vores Monopoly-spil, så har vi vurderet, at vi vil få brug for følgende klasser:

Dice(Skal ikke opdateres)

teks

Player (Oliver)

tekst

PlayerList (Oliver)

Field (Johannes)

FieldList (Jonathan)

Board (Jonathan)

Game (Oliver)

Logic (Johannes)

Chance-Card (Gustav)

Account (Alaa)

GameGUI (Krzysztof)

Menu (Krzysztof)

Translator (Alaa)

Design

Systemsekvensdiagram

Sekvensdiagram

Designklassediagram

Implementering

- Lav passende konstruktører.
- Lav passende get og set metoder.
- Lav passende toString metoder.
- Lav en klasse GameBoard der kan indeholde alle felterne i et array.
- Tilføj en toString metode der udskriver alle felterne i arrayet.
- Lav det spil kunden har bedt om med de klasser I nu har.
- Benyt GUI'en. Gui' skal importeres fra Maven: Maven repository

Dokumentation

- Forklar hvad arv er.
- Forklar hvad abstract betyder.
- Fortæl hvad det hedder hvis alle fieldklasserne har en landOnField metode der gør noget forskelligt.
- Dokumentation for test med screenshots.
- Dokumentation for overholdt GRASP.

Konfiguration

Konfigurationsstyring

Udviklingsplatformen er alt det software i bruger under udviklingen af jeres projekt. Produktionsplatformen er alt det software der skal bruges til at køre jeres færdige program. I dette projekt er de ens. I skal dokumentere platformens dele med versionsnummer så den kan genskabes til senere brug. Jeres platform består af operativsystem, java, og IntelliJ samt biblioteket matadorgui.jar der hentes fra Maven.
I bedes definere hvordan I vil sikre jer, at I på et hvert tidspunkt vil kunne
finde den sidste nyeste version af samtlige artefakter OG hvordan I vil
sikre jer at dokumentationen altid er opdateret for jeres system
– Hvor er filerne?
– Hvordan finder vi den nyeste version?
– Hvordan sikrer vi at vi ved om versionen er opdateret?
– Hvordan finder vi versioner som passer sammen?
Endvidere skal i beskrive hvordan man importerer jeres projekt i IntelliJ fra git, og hvordan man kører jeres program.
Det er også et krav at I bruger Maven til at hente junit.

Test

Test

Lav mindst tre testcases med tilhørende fremgangsmåde/testprocedure og testrapporter.
Lav mindst én Junit test til centrale metoder. Inkludér code coverage dokumentation.
Lav mindst én brugertest. Husk at brugeren skal være en der ikke kan kode.

Projektplanlægning

Konklusion

Kilder

stbn, CDIO del 3, 02312 / 02313 / 02314 / 62532 Efterår 2020 revision 8.
https://docs.google.com/document/d/1UsCw9jeOvZlthzUrxIg0sF8ez1xwk-64oh6KivQvzHI/edit#
(Hentet: 02-11-2020)

Versionsstyring

Lav et lokal Git-repository som en del af IntelliJ-projektet.
Alternativt kan afleveres et link til et repository på nettet eks.

https://github.com/cbudtz/TestRepo42.git.

Rapporten skal indeholde en vejledning i hvordan man importerer Git-repository i IntelliJ.