Kryds & Bolle - Obligatorisk opgave #2

Afleveret af

1. Anders Rene Thomsen
2. Daniel Rahbek Schultz
3. Kasper Lyster Nørbygaard
4. Nicklas Kramer Andersen

Dato: 13/10-2023

Indholdsfortegnelse

1. Introduktion **3**
1.1. Baggrund **3**
1.2. Problemdefinition **3**
1.3. Produkt vision **3**
2. Pre-Game/Baggrund **4**
2.3. Indledende produkt backlog **4**
2.4. Arkitektur **5**
3.1. Sprint planlægning **6**
3.3. GUI **7**
3.5.1. Kodeeksempler **8**
3.7. Sprint anmeldelse **9**
3.8. Sprint retrospektiv **9**
9. Bilag **10**
9.1. Bilag 1 Gruppe Kontrakten **10**

1. Introduktion

Dette projekt omhandler spillet TicTacToe, som man kender det i et online format. Selve spillets opbygning foregår via et spillebræt på 3x3 størrelse. Når den første spiller har placeret sin første brik på pladen, er spillet i gang. Spillet handler om at man skal placere tre brikker på stribe, og dermed vinder spillet. Dette kan gøres både vertikalt, horisontalt og diagonalt. Spillet bliver som regel spillet som et “2 player game”, men kan også spilles mod en AI. Spillet kan virke simpelt, men det er muligt at skabe en fælde, der gør, at du kan vinde på 2 måder og dermed vinde spillet.

1.1. Baggrund

Denne opgave er baseret på det klassiske spil, kaldet TicTacToe. Projektets opgave er at genskabe spillet og dets givne funktioner i IntelliJ IDEA. Projektet er produceret af gruppe nummer 2 “Bolle og kryds”, bestående af studerende fra Erhvervsakademi Sydvest i Esbjerg. Spillet kommer til at indeholde følgende funktioner: Spillet skal være i stand til at finde en vinder, lave en game over-metode og nulstille spillet tilbage, således at et nyt spil kan starte.

1.2. Problemdefinition

Opgaven er at genskabe spillet TicTacToe, med et sæt af givne obligatoriske opgaver. Det er eventuelt muligt at lave videreudvikling på spillet. De givne opgaver, vil blive forklaret længere nede i afsnit 2.3. “Indledende produkt backlog".

1.3. Produkt vision

Den indledende plan er, at få et godt overblik over den allerede eksisterende kode. Hvordan koden fungerer og hænger sammen. Derefter videreudvikles der ud fra den eksisterende kode, indtil et færdigt produkt er opnået. Dermed opfyldes kravene til alle de obligatoriske opgaver forklaret i afsnit 2.3. Hvis der er tid til overs, vil der blive lavet yderligere udvikling af spillet, hvilket vil skabe mere værdi for spillet, samt dets funktionalitet. Dette vil også gøre at flere får lyst til at spille, hvilket altid er godt.

2. Pre-Game/Baggrund

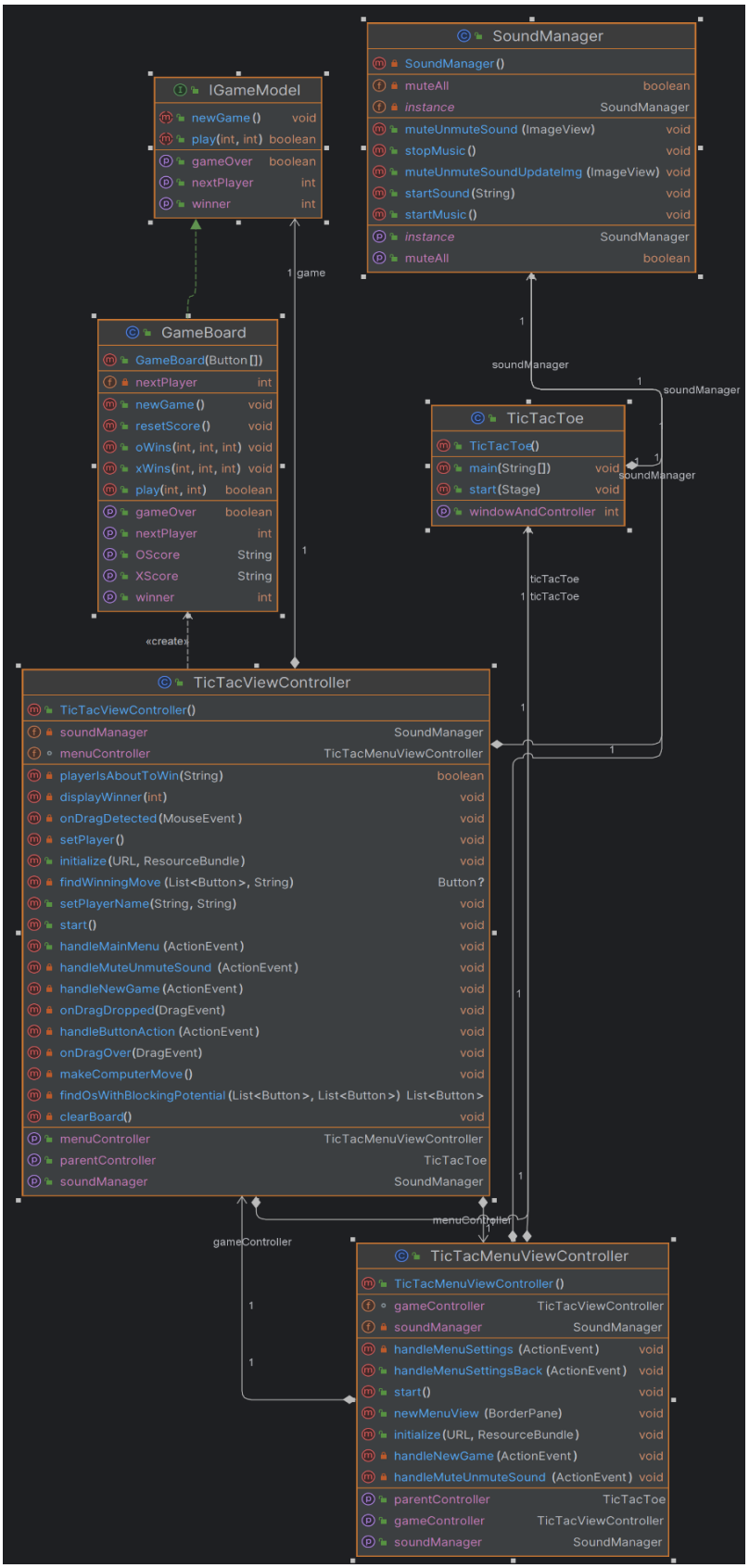
2.3. Indledende produkt backlog

I dette projekt, er der blevet udleveret et antal krav, som skal opfyldes. Der er ikke særlig mange krav. Grunden til dette er, at opgaven handler mere om at forstå hvordan man opbygger et program optimalt. Kravene for opgaven lyder som følgende:

Lav en ordentlig implementering af GameBoard klassen i den udleveret kode, så hele applikationen fungerer perfekt. Derudover skal der kunne findes en vinder.

2.4. Arkitektur

Der vil i dette afsnit kort blive beskrevet, hvordan de forskellige klasser interagerer med hinanden, ved hjælp af et Unified Modeling Language (UML) som ses til højre.



IGameModel:

Er et interface af Gameboard som skal implementeres og har aggregation til TicTacViewController.

SoundManager:
Har aggregation til TicTacViewController, TicTacMenuViewController og TicTacToe.

GameBoard:
Er en implementering af IGameModel. Det betyder den skal indeholde de metoder der er defineret i IGameModel

TicTacToe:
Har aggregation til TicTacMenuViewController og TicTacViewController.

TicTacViewController:
Er en implementering af GameBoard og har aggregation til TicTacMenuViewController. og TicTacToe

TicTacMenuViewController: Har aggregation til TicTacViewController. og TicTacToe

3. Sprint 1

3.1. Sprint planlægning

D. 09-10-2023
På dagen:

Gruppen mødtes kl. 09:15 og lagde planer for dagen. Planerne bestod i at uddelegere individuelle opgaver til gruppens medlemmer. Nogle medlemmer arbejdede på kodningsaspektet af opgaven, andre på spillets visuelle design og andre med at skrive rapporten.

D. 10-10-2023
På dagen:

Gruppen mødtes kl. 09:15 og lagde planer for dagen. På dagen blev der implementeret en “Drag and drop” funktion i spillet, således at når de første seks brikker er placeret på brættet, kan der nu rykkes rundt med dem. Der blev også fiflet med en bot til 1 player, som dog ikke er særlig intelligent tænkende.

D. 11-10-2023
På dagen:

Gruppen mødtes kl. 09:15 og lagde planer for dagen. På denne dag blev der arbejdet med at lave en “soundmanager” klasse, som skal styre alt lyd i spillet. Derudover blev botten også gjort lidt klogere, men stadig mulig at besejre.

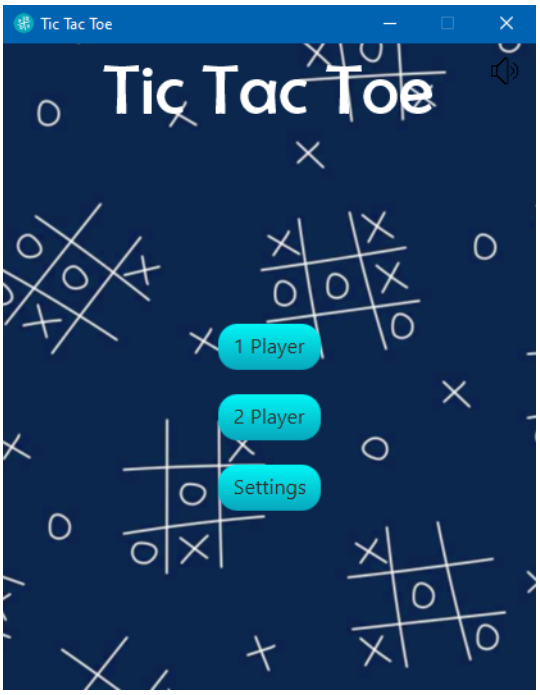
D. 12-10-2023
På dagen:

Gruppen mødtes kl. 09:15 og lagde planer for dagen. Dagen startede ud med at få merged hele projektet sammen, da soundmanager var lavet i en anden branch. Derudover blev der debugget diverse fejl i koden og ændret på noget af GUI.

D. 13-10-2023
På dagen:

Gruppen mødtes kl. 09:15. Koden blev gennemgået, såfremt alle forstod hvad alt koden betyder, samt hvad der gør hvad. Rapporten blev herefter gennemlæst, foretaget korrekturlæsning og afleveret ligeså med koden i en ZIP.

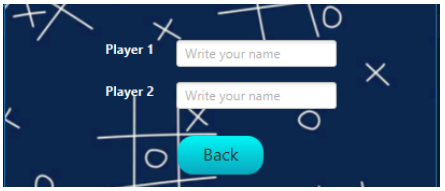
3.3. GUI



Billede nr. 1

Billede nr. 2

Som det kan ses på billede nr. 1, er der en game menu skærm, som er blevet tilføjet. Det er nu blevet en mulighed, at indtaste navn på både player 1 og player 2. Hvilket ikke tidligere var en mulighed. Dette kan ses på billede nr. 3. Der er derudover også implementeret en mute funktion, hvilket slår lyden fra i hele programmet. På billede nr. 2 er der blevet lavet en "Menu" knap. Hvilket med et tryk på den, kan føre dig tilbage til menuen. En "New Game" knap er også tilføjet, hvilket giver spilleren mulighed for at starte et nyt spil, såfremt dette ønskes af spilleren. Det visuelle design/udtryk af spillet, er også blevet ændret. Der er blandt andet blevet tilføjet en ny baggrund til spillet, samt ændret farven på spillets overskrift, fra sort til hvid. For at det visuelle kan ses.



Billede nr. 3

En score counter er også blevet tilføjet til spillet. Dette gør at man hele tiden, kan holde styr på hvor mange runder af spillet, enten player 1 eller player 2 (computer i 1 player) har vundet. Det er nu også muligt, at se hvilken spillers tur det er, via et display placeret mellem scoren. Scoren blev automatisk indstillet til 0, når man går til hovedmenuen. Så kan spillet starte på en frisk, og alle får en chance til.

3.5. Implementering

3.5.1. Kodeeksempler

Koden for at vinde

I et klassisk spil kryds og bolle, er der altid én vinder. Det skulle vi også implementere i dette spil. Denne implementering blev løst ved, først at skabe en array over de 8 forskellige måder man kan vinde et spil på, for at gøre dette så nemt som muligt har vi tilgivet hvert spillefelt en værdi fra 0 til 8. *Eksempel 1* viser denne implementering.

```
private int[][] winConditions =
        {
                {0, 1, 2}, {3, 4, 5}, {6, 7, 8}, // Rows
                {0, 3, 6}, {1, 4, 7}, {2, 5, 8}, // Columns
                {0, 4, 8}, {2, 4, 6} // Diagonals
        };
```

Eksempel 1

Dernæst skulle vi bruge en metode som tjekker, om der er nogen som har vundet spillet. Her blev der skabt isGameOver() metoden som set i *eksempel 2*. Denne metode består af to loops, som hver især tjekker alle spillefelterne og sammenligner dem med de win conditions, som blev forklaret tidligere. I den øverste loop i *eksempel 2* tjekker den de spilfelter markeret med “X” og hvis den får et match med de 8 forskellige måder at vinde på, så returnerer den “True” - hvilket i en anden del af koden får spillet til at stoppe.

```
public boolean isGameOver()
{
    // Check X win conditions
    for (int[] winCondition : winConditions) {
        if (buttons[winCondition[0]].getText().equals("X") &&
            buttons[winCondition[1]].getText().equals("X") &&
            buttons[winCondition[2]].getText().equals("X")) {


            return true;
        }
    }

    // Check O win conditions
    for (int[] winCondition : winConditions) {
        if (buttons[winCondition[0]].getText().equals("O") &&
            buttons[winCondition[1]].getText().equals("O") &&
            buttons[winCondition[2]].getText().equals("O")) {


            return true;
        }
    }

    return false;
}
```

Eksempel 2

I vores spil bruges der lyde, for at skabe en god oplevelse for brugeren. Til at starte med, havde man fundet en løsning, hvor at hver gang man skulle slå lyden fra i spillet, skulle man individuelt slå lyden fra på hvert enkelte lydklip. Det vil i længden blive træls, når man har mange lyde. Det blev løst ved at skabe en liste, hvor man kan bruge indholdet i, ved at kalde det ved deres navn. Det gjorde at det var muligt at slå alt lyden fra. Ved at løbe hele map listen igennem, ved hjælp af en for løkke, og dermed slukke alle eksisterende lyde i spillet. Musikken blev gjort for sig selv, da den skal køre konstant, og normalt vil lydeffekter og musik deles op i lydindstillinger i større programmer, da nogle ønsker det ene frem for det andet.

```
public void muteUnmuteSound(ImageView img) {
    if (!muteAll == true) {
        for (AudioClip sound : soundMap.values()) {
            if (sound != null)
                sound.setVolume(0.0);
        }
        img.setImage(new Image( url: "tictactoe/gui/images/unmute.png"));
        backgroundMusic.setMute(true); // Mute
        muteAll = true;
    } else {
        for (AudioClip sound : soundMap.values()) {
            if (sound != null)
                sound.setVolume(1.0);
        }
        img.setImage(new Image( url: "tictactoe/gui/images/mute.png"));
        backgroundMusic.setMute(false); // unMute
        muteAll = false;
    }
}
```

3.7. Sprint anmeldelse

Overordnet set, er alle de obligatoriske opgaver indenfor spillet blevet gennemført, samt at alt fungerer som det skal. Spillet er nu i stand til at opfange, hvorvidt et felt inden for spillebrættet på 3x3 format, er optaget eller ej. Dette betyder at, spillet kan genkende, hvorvidt et træk er muligt eller ej. Derved kan det afgøres, om det er den næste spillers tur. Spillet har dermed også fået en “2 player” funktion, således at man rent faktisk kan spille mod hinanden. Derudover har spillet fået en “game menu”, hvor der kan vælges at starte et nyt spil, ved tryk på knappen. Spillet er nu i stand til at genkende, når en spiller har fået tre på stribe. Dermed i stand til at erklære en vinder i spillet.
Hvis en vinder ikke findes indenfor de første seks træk, er det nu blevet en mulighed via “drag and drop”, at fortsætte med at rykke rundt på brikkerne, indtil en vinder er fundet. Dette giver større mulighed for at skabe et strategisk spil.

3.8. Sprint retrospektiv

Projektet gik overordnet set rigtig godt og efter planen. Projektet blev fuldført i den forstand, at alle de “mandatory” opgaver som der blev stillet, også blev gennemført. Der var i dette projekt større fokus på forbedring af “coupling and cohesion" sammenlignet med tidligere projekter, således at alt koden ikke blev samlet under samme klasse, men spredt bedre ud i forskellige klasser.

9. Bilag

9.1. Bilag 1 Gruppe Kontrakten

Gruppekontrakt

Projektnavn: Bolle og kryds

Projektets forfaldsdato:13. oktober kl 14.00

Holdkammerater:

Navn	Kontaktmetode 1#	Kontaktmetode 2#
Nicklas	25708507	nicand02@easv365.dk
Daniel	Discord	dansch01@easv365.dk
Kasper	24670506	kasnoe01@easv365.dk
Anders	30321240	andtho01@easv365.dk

Holdaftaler og grundregler:

Problem	Aftale
Personlige interaktioner	Platformen for møder er Discord. Arbejdsmængden er 5 timer om dagen. Fra 09:00 -12:00 og fra 13:00-15:00. (Fredag 14:00) Gruppen mødes personligt, hvis det er aftalt.

Roller og ansvar	**Lederen: Anders** Leder diskussioner og kontrollerer konsensus i gruppen. **Arrangøren:** Planlæg mødes, hvis det er nødvendigt, og sørg for, at alle får information. **Forskeren: Os alle** Forsker i kodning **Fejlfinderen: Os alle** Omskriv koder og fejlfinding. **Forfatteren: Vi alle sammen** Skriv og fejlfind sammen. **Redaktøren:** Korrekturlæser dokumentet inden indsendelse.

Fordeling af arbejdsbyrden	Anders : Leder, forsker, troubleshooter, skribent. Daniel : Forsker, fejlfinder, skribent. Nicklas : Arrangør, forsker, fejlfinder, skribent. Kasper : Redaktør, forsker, fejlfinder, skribent.
Håndtering af konflikter	Hvis der opstår konflikt, stopper vi det, vi laver, og taler om de problemer, vi støder på. Hvis vi ikke kan løse problemet som voksne, må vi ty til en 1v1 i et spil kryds og bolle - vinderen har ret.
Konsekvenser	Hvis størstedelen af gruppen mener at medlemmet ikke er arbejdsom, samarbejdsvillig og uvillig til at færdiggøre projektet, får de en advarsel af gruppen. Hvis det fortsætter efter advarslen, kan medlemmet smides ud af gruppen efter drøftelse med en underviser.

Beskrivelse af projektet:

Den anden obligatoriske opgave, der handler om et meget simpelt brætspil: Tic-Tac-Toe. Spillet er et to-spiller spil, og brættet er et 3 x 3 bræt. På skift lægger de to spillere en genstand i et felt på brættet. En spiller vinder, hvis hun eller han får tre i en række, i en kolonne eller i en diagonal.

Hvilke produkter vil gruppen skabe for at fuldføre dette projekt?

Gruppen vil lave koder til at levere produktet og producere en skriftlig tekstfil på dansk med billeder for at beskrive hensigten med gruppen af produktet.

Hvilken forskning er nødvendig for at udføre dette projekt?

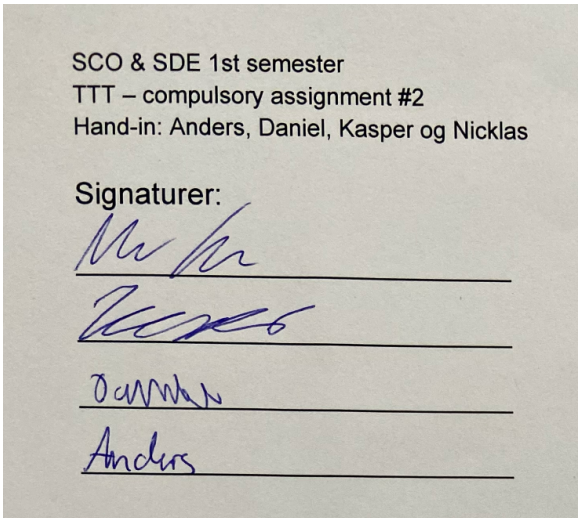
Adfærdsegenskaber, der passer til vores vision.

Motivering:

For at levere et godt produkt ville det have været godt. At få viden og erfaring om kodning, struktur af kodning, struktur på at skrive et projekt og at beskrive produktet.

Foreslået tidslinje:

	Hvornår skal det være færdigt	Hvem er primært ansvarlig
Del 1 (Mandag) 09-10-2023	• Gruppekontrakt • Obligatoriske opgaver.	Gruppe.
Del 2 (Tirsdag) 10-10-2023	• Valgfrie opgaver.	Gruppe.
Del 3 (Onsdag) 11-10-2023	• Valgfrie opgaver.	Gruppe.
Del 4 (Torsdag) 12-10-2023	• Fejlretning. • Rapport.	Gruppe.
Slutprodukt (Fredag) 13-10-2023	• Aflever opgave. (Deadline 14.00)	Gruppe.


SCO & SDE 1st semester
TTT – compulsory assignment #2
Hand-in: Anders, Daniel, Kasper og Nicklas

Signaturer:

Anders