Web-baseret uno

Klasse	*3.Q*
Fag	*Programmering C*
Årgang	*2012 - 2015*
Skole	*HTX Sukkertoppen*
Medlemmer	*Bo Hans Thomsen, 16/03/1996* *Frederik Fredslund Lassen, 27/06/1996*

Indledning

I dette projekt vil vi bygge det velkendte og desværre varemærkebeskyttet Uno spil, med moderne webteknologier. Kortene er svg vektorgrafik, brugergrænsefladen bygget i CSS og HTML med Bootstrap, med funktionalitet på brugersiden skrevet i JavaScript med stor fokus på genbrug og objektorienteret programmering, så meget det nu er muligt i JavaScript. Denne prototype er lavet til kun at fungere på en computer, men logiken er bygget så den nemt kan genbruges på serversiden i NodeJS.

Projektet skal inkludere en bred blanding af logik. Der skal være logik til at lave et dæk, dele kort ud, vise, spille og ikke mindst tjekke om kortet er gyldigt.

```
if hånd.lagte.længde == 0
    if kort.farve == øverste.farve || kort.værdi == øverste.værdi || kort.farve == 'wild'
        return Kort er gyldigt
else
    if kort.værdi == øverste.værdi ||
    ( øverst_lagte.værdi == +4 && kort.værdi == +2 ) ||
    ( øverst_lagte.værdi == +2 && kort.værdi == +4 )
        return Kort er gyldigt

return Kort er ugyldigt
```

- Pseudokode der viser logiken der vurdere om et kort er gyldigt

Projektet er interessant fordi det giver mulighed for at vise en bred vifte af forskellige programmeringsteknikker, implementere logik med færest linjer kode og så objektorienteret så muligt, med mest mulig genbrug af kode. Spillet kan både laves så alt kører på brugersiden som vores prototype er eller en blanding af server- og user-side med brug af de mest moderne web-teknologier som NodeJS og WebSockets.

På følgende flowdiagram er bruger interaktionen omkring kortene vist, med placering af kort, tjek af kortets gyldighed og enten ignorering af kortet eller placering af kortet på toppen af dækket.

Læsevejledning

Rapporten begynder med den introduktion til projektet du lige har læst oven over, hvor den vigtigste logik er vist med pseudokode og en brugerinteaktion er vist med et flowdiagram.

Herefter følger et afsnit om programmeringsteori, hvor de vigtigste elementer i programmering forklares. Dette afsnit inkludere ikke kun teori der er relevant for projektet, men generel teori.

Efter læseren er sat ind i teorien bag, følger en forklaring af den endelig implementering af de problemer der er forklaret indtil nu, med logiken i spillet Uno og de forskellige datatyper.

Efter det følger et afsnit med brugervejledning i brug af produktet og læsning af koden og efterfølgende test af selve produktet.

Til sidst afsluttes med en konklusion på projektforløbet.

Programmeringsteori

Der findes flere forskellige former for programmeringssprog, der findes typestærke programmeringssprog hvor en informationsbeholder, kaldet en variabel, kun kan indeholde data af en type data. I nogle sprog skal variablerne defineres og typen angives før at variablen kan bruges, mens i andre typestærke sprog bestemmer fortolkeren hvilken type at variablen skal indeholde, et eksempel på et typestærkt sprog med krævet variabel initialisering kunne være C++.
Modsat findes der sprog som JavaScript og PHP hvor variabler kan indeholde forskellige typer data, uden at variablen geninitialiseres.

Variable

En variabel er en organiseret opbevaring af information. En variabel kan have mange forskellige typer. Af ofte brugte typer findes der en instans af en klasse, en array der er en samling af variabler, en char der er et tegn, en streng der er en samling af tegn, en integer der er et heltal, en float der er et kommatal, en boolean der er en logisk datatype der enten er sand eller falsk og en tuple der er en liste der ikke kan ændres efter den er oprettes.

En array fungere på forskellige måder i forskellige sprog, i typestærke sprog indeholder en array/list kun en type data, fordi at hver element så fylder det samme i hukommelsen- der skal arrayen initialiseres med en type og længde så de forskellige elementer kan lægges på række i hukommelsen.

Betinget udførsel

Betinget udførsel er en meget vigtig styringsfunktion i programmering, der kan begrænse udførelsen af en blok kode. Der kan opsættes uendelige antal betingelser, og hvis betingelsen ikke er sand for den første gruppe, kan der fortsættes med en ny blok og liste af betingelser. En standard blok af kode kan sættes til at udføres, hvis ingen af de forskellige mulige grupper af betingelser er blevet sand.

```
if age < 15
    print Stadig ung, ingen ret til sex
else if age < 16
    print Ret til sex, men ingen alkohol
else if age < 18
    print Ret til at købe alkohol med lav procent
else
    print Voksen ret til alt
```

- Betinget udførsel forklaret, men de vigtigste betingelser

Betingelsen sættes oftest med if og skal evaluere til sand for at blokken eksekveres, ellers fortsættes til næste betingelse inde for gruppen, hvis den findes eller til else hvis ingen evaluere til sand eller der ikke er flere, selvfølgelig kun hvis der findes en else.

En anden måde at opsætte udsagn der kan betinge udførelsen af en blok kode er ved at bruge switch, hvor en værdi evalueres, derefter opsættes nogle forskellige betingelser med en medfølgende kodeblok og en mulig kodeblok der kan køres hvis ingen af betingelsen evalueres til at være sand.

```
switch blomsterfarve
    case rød
        print Kærlighed, åhh kærlighed
    break

    case pink
        print Lille smule underligt O.o
    break

    default
        print Bestem dig
```

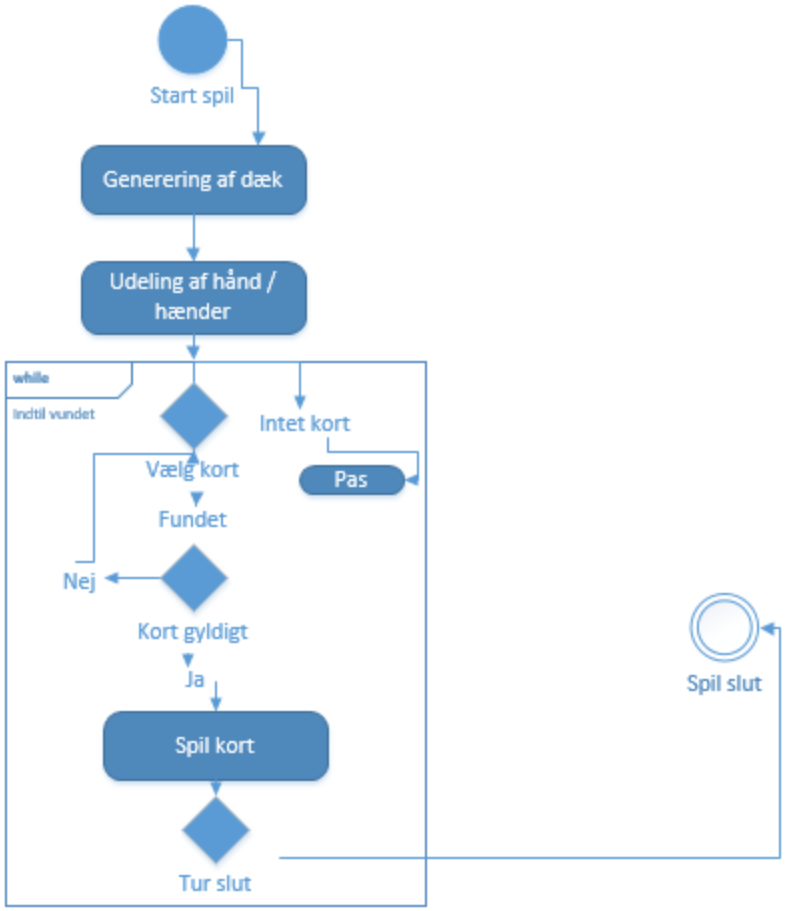
Løkker

Løkker er en serie af operationer der bliver gentaget indtil en eller flere opsatte bestemmelser er mødt. Af forskellige løkker kan nævnes while der gentages indtil den eller de betingelser der er givet returnere boolean sand, en for løkke der kører et af brugeren forudbestemt antal gange og hver gang enten forøger eller formindskes en index variabel og et forEach der kører en gang per element i den valgte liste.

Med løkker kan man udføre operationer per element, eksempelvis vise en liste af elementer på skærmen med for eller forEach, en operation - som det at opdatere skærmen udføres hver gang CPU'en kører et tick med while eller lave en vilkårlig værdi og tjekke om den allerede er brugt.

Analyse af problemstilling

I Uno er der et sæt af kort og der er et sæt af regler der skal implementeres. Fire forskellige kort farver med tal fra 0 - 9 + skiftretning + springover + tag to kort op. Ved siden af det er der tag fire kort op + vælg farve og skift farve.



Et kort kan lægges hvis det enten er af farven wild, har samme nummer eller samme farve som det der ligger. Flere kort af samme værdi kan lægges oven på så længe lager haves.

Derefter spilles der i ture. Hvis spilleren intet kan lægges trækkes der et kort af gangen fra bunken, maks 3, hvis ingen af dem passer - siger brugeren pas og turen går videre.
Når spilleren lægger det næst sidste kort, skal der klikkes på Uno, ellers tildeles 3 kort.

Den der første kommer af med alle kort, vinder.

Implementering

Koden er implementeret med stor fokus på Objektorienteret programmering og genbrug af kode. Hver type af spilobjekt er implementeret som sin egen klasse og så styrer de forskellige objekter hinanden. Hvert kort er et objekt med sine egenskaber implementeret. Et kort har en farve (rød. grøn, blå, gul og blandet), en funktion / nummer, et unikt guid der bruges til at finde det specifikke kort der er spillet i brugergrænsefladen,- på hånden, stakken før kortet spilles og i bunken, en højde, en bredde, et offset og en funktion der kan visualisere kortet - det er dog en funktion der udnytter en klasse specifikt bygget til at visualisere.

Spillebrættet har sit eget objekt, der indeholder brugerens hånd, dækket og de spillede kort. Brættet står for alt fra brugerinput, visning af dækket og validering af om et kort er korrekt.

Dækket er implementeret som sit eget objekt, der indeholder de kort der ligger, hvilke farver der eksisterer, hvilke numre og andre typer af kort - og hvor mange kort der skal uddeles. Det indeholder funktionalitet til at læse størrelsen af dækket, tilføje et eller flere kort, bygge et dæk, blande dækket, trække et kort eller trække flere kort.

Hånden er implementeret seperat den indeholder en liste over kort på hånden, hvilke kort brugeren er ved at spille og funktionalitet til at læse størrelsen, tilføje kort og finde et specifikt kort.

Brugervejledning + Test

Fordi at programmet er lavet som en hjemmeside, er logiken skrevet i JavaScript, der først kompileres når det skal bruges og derfor ligger det ikke i kompilerede filer. Koden kan derfor ses i js mappen under game.js - hvor alt logiken ligger, resten af de filer der ligger der er tredjeparts biblioteker vi inkludere. Bootstrap til layout, jQuery til nem og hurtig manipulation af DOM'en, der er den træ-struktur at en hjemmeside er opbygget i, et plugin til at håndtere tastegenveje, jQuery UI der er et bibliotek med forskellige visuelle elementer som foreksemepl drag-n-drop og til sidst et bibliotek til at lave fyrværkeri animationer. I index.html er brugerflade elementerne bygget op i markup sproget HTML og i css/cards.css ligger de æstetiske ændringer vi har lavet i CSS.

Selve programmet og brugergrænsefladen kan ses og testes ved at åbne index.html, der ligger i den afleverede folder.

Vi har under udviklingen af projektet testet funktionaliteten af først vores forskellige klasser igennem, hvilket ligger inde i cards mappen, hvor første eksperiment er lavet, sammen med et converterings script skrevet i Python, der ændre svg filen til den opsætning vi ønsker.

Da selve produktet var færdigt har vi testet det med nogle forskellige brugere og selv forsøgt os med scenarier vi selv kunne opsætte.

I spillet findes fire forskellige farver (rød, grøn, gul og blå) indenfor hver farve findes der to kort for hvert nummer mellem 0 og 9 plus to springover kort, to træk to op kort og to skift retningskort, udover farverne findes der fire trækfirekort op og fire vælg farvekort.

Det første der sker når spillet åbnes er at der opbygges et dæk af samtlige af de valgte kort, hver spiller får derefter 7 kort på deres hånd, i vores tilfælde er der kun en spiller, der lægges også et kort øverst i bunken der er det kort der diktere næste kort.

Spilleren skal derefter vælge et kort, enten skal det kort der lægges have samme farve som det der ligger i bunken eller samme værdi, derefter må der lægges så mange af samme værdi oven på som spilleren har på hånden - derefter trykkes der på Play. Hvis spilleren ingen kort har der passer, kan der trykkes maks 3 gange på Draw, hvorefter der gives et nyt kort. Hvis ingen af dem passer springes brugeren over.

Konklusion

Produktet er blevet bygget i kun webteknologier og server-side kode kan nemt implementeres ved at udnytte en stor del af den eksisterende kode, ved at bruge NodeJS på serversiden.

En stor del af koden genbruges igennem forløbet, der er opbygget objektorienteret. Hver type har sit eget objekt - kort, spillebræt, hånd og deck - på der laves flere instanser af samme type og andre dele har en standard kontaktflade til den funktionalitet og data som objektet tilfører.