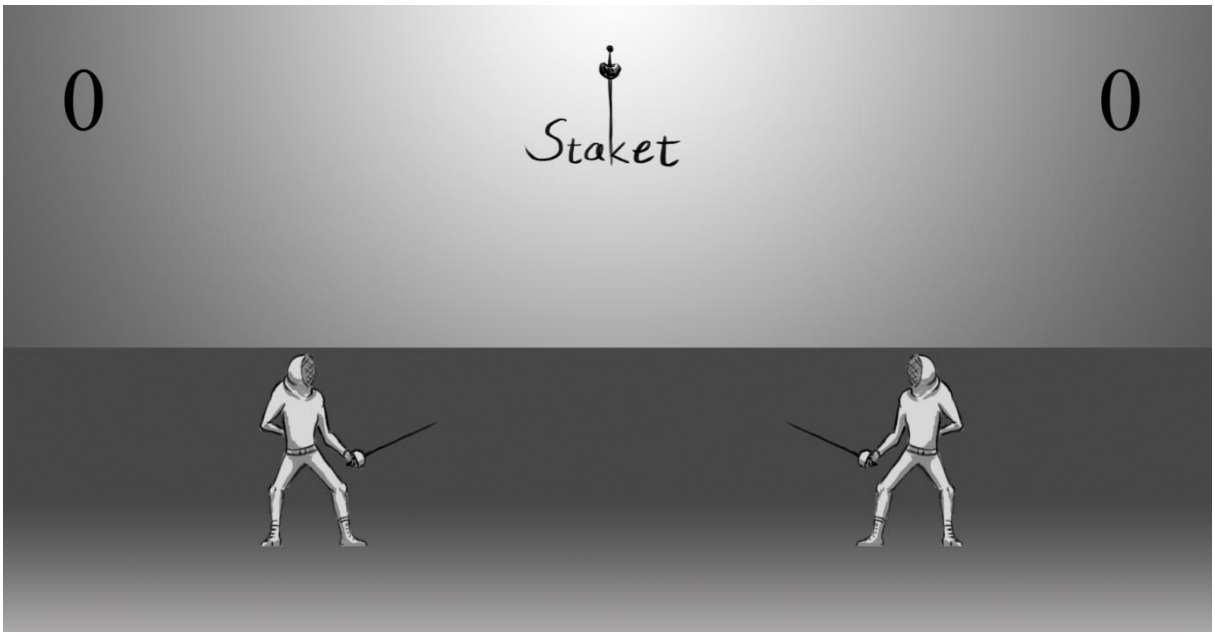
KTH

Staket

Ett riktigt intensivt fäktningsspel

Richard Uggelberg, Henry Eklind

5/15/2014



Staket är ett riktigt intensivt fäktningspel där de båda spelarna, framför samma skärm, skall fäktas till döden. Vinnaren vinner inte bara äran utan även ett poäng i spelets poängräkning. Matchen pågår så länge som spelarna orkar. Använd funktionerna för att gå framåt och bakåt, blocka och attackera till din fördel och ta hem segern!

Contents

Terminologi 2

Programbeskrivning 2

Användarbeskrivning 3

Användarscenarier 4

Scenario 1: 4

Input: 4

Output: 4

Scenario 2: 4

Input: 4

Output: 4

Testplan 6

Programdesign 7

Tekniska frågor 10

Arbetsplan 11

Till Övning 12 (2/5) 11

Till Övning 13 (9/5) 11

Till Övning 14 (16/5) 11

Sammanfattning 12

Staket

Terminologi

Staket = separerar tomter; namnet på den adligaste av sporter även kallat fäktning.

Värja = svärd som används inom sporten staket.

Användare = någon som kontrollerar en värjhållare.

Värjhållare = någon som håller en värja (se "Värja")

Programbeskrivning

Planen

Det handlar om staket, eller fäktning som det heter på svenska. Ett fightingspel där två spelare möter varandra.

Programmet kompileras och körs ifrån command-line. Direkt kastas användarna in i en match till "döden" (det har ännu inte bevisas ifall virtuella karaktärer faktiskt dör). Användarna har valet att kunna blockera sin motståndares värja eller att vifta i deras riktning. Karaktärerna kan även förflytta sig vänster och/eller höger om det så ska krävas. I krig och kärlek finns inga regler. Ifall någon av kämparna skulle dö kommer spelplanen automagiskt att återställas och en poäng ges till den vinnande värjhållaren. Användarna spelar tills de inte längre orkar bevittna blodutgjutelse.

Värjhållare A har tidigare under kvällen stött på värjhållare Bs syster. Dat plot twist. I och med detta utmanar B A på en (pub)runda staket™. För att visa att familjen är det viktigaste i hens värld.

Resultat

Spelet Staket går ut på att två spelar kontrollerar varsinn fäktare som, för att vinna spelat, ska döda den andra spelarens fäktare. Detta sker genom att spelaren höjer sin karaktärs värja när den är tillräckligt nära motståndaren således att värjan hamnar i motspelarens ansikte. Detta. som tidigare nämnt resulterar i en vinst. Dock kan spelaren som blir attackerad blockera attacken och sedan attackera själv. Spelarna kan även gå fram eller tillbaka för att undvika attacker eller för att ge sin egen karatär räckvidd att själv attackera. Spelet utspelar sig i två dimensioner längs ett plan som ska föreställa en motsvarighet till den längan som fäktare slåss längs i fäkningstävlingar.

Jämförelse

Det finns egentligen inga relevanta skillnader mellan vår vision och det vi nu har, i alla fall vad gäller idén. De funktioner som annorlunda från vår idé är mycket små och var alleles för specifika för att skriva ner. Alternetivt var det som vi ändrat något vi inte riktigt visste hur man gjorde och har därmed ändrat funktionen därefter för att få det att fungera.

Användarbeskrivning

Planen

Användare: goda vänner med meningsskiljaktigheter som vill slåss till döden, fast på låtsas.

Antaganden: har en kaffemaskin i datorn, alternativt fungerar Java; kan starta och kompilera program ifrån command-line; har sett och eller använt ett tangentbord förut; är etniskt svensk, australiensisk inföding eller något däremellan; vet vad sporten staket innebär; kan läsa och skriva; kommer gissa att WASD och piltangenterna är bra knappar att testa.

De är: människor (rimligtvis); vana datoranvändare; gamershs.

Resultat

Användare innefattar människor som har åtminstone någorlunda datorvana och som (förhoppningsvis) tycker om att spela spel mot varann. Användare ska ha en dator med java installerat och ska även kunna (eller kunna fråga om hjälp för att) starta programmet från command-line. Användare förväntas ha en uppfattning om vad sporten "fäktning" (Staket, i detta fall) går ut på.

Jämförelse

Ingen skillnad gentemot projektplanen förutom seriositetsnivån.

Användarscenarier

Planen

Scenario 1:

Vad de ser: Se figur 1.

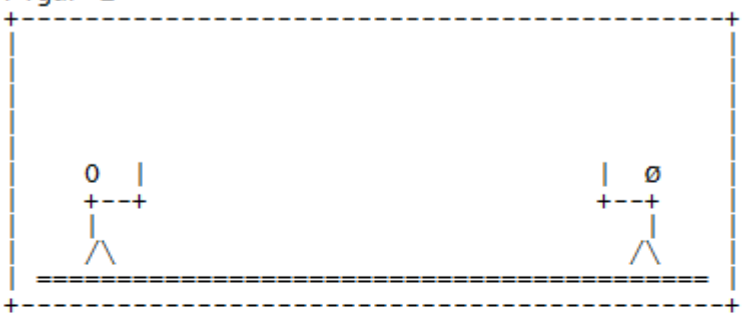
Input:

Användaren kompilerar och startar programmet via command-line. Väl inne i spelet har en match till döden redan börjat och användaren trycker frenetiskt på piltangenterna eller WASD för att vidarebefordra sin ilska till sin motståndares avatar.

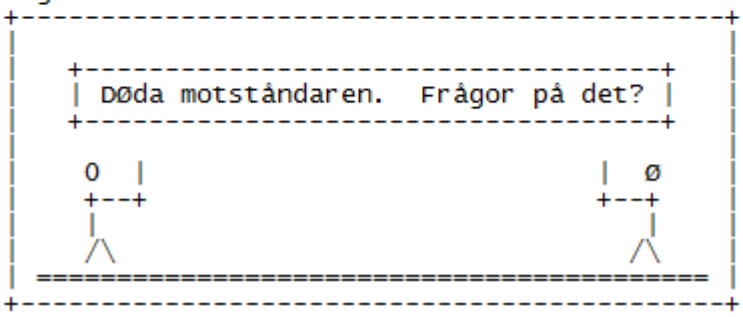
Output:

Karaktären på skärmen rör sig hatiskt mot sin virtuella motståndare samt svingar sin värja med obeskrivbar finess.

Figur 1



Figur 2



Scenario 2:

Vad de ser: Se figur 2.

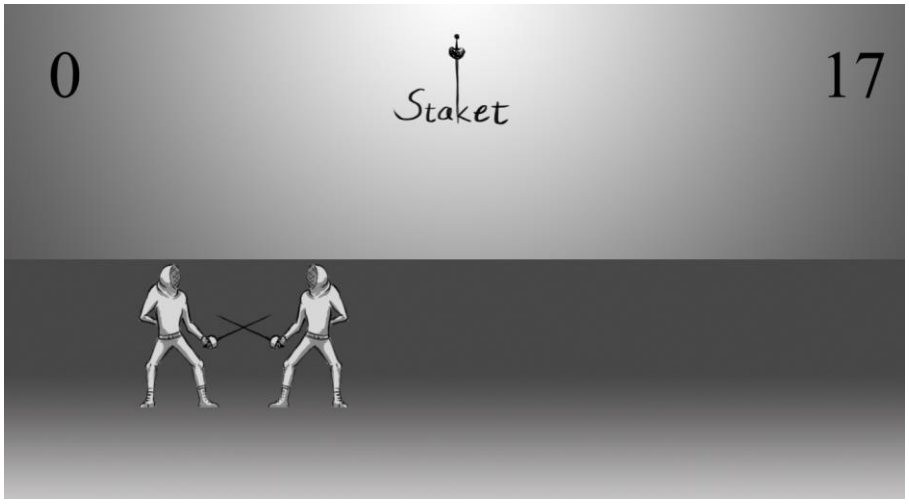
Input:

Användaren kompilerar och startar programmet via command-line. Eftersom denna användare inte är insatt i gentlemanna sporter som staketning behövs instruktioner. I verkligheten kommer de båda utmanarna överens om att gå igenom reglerna för sporten. Med finess klickar användaren på 'q', som står för qultur, för att utbilda sig inom sporten.

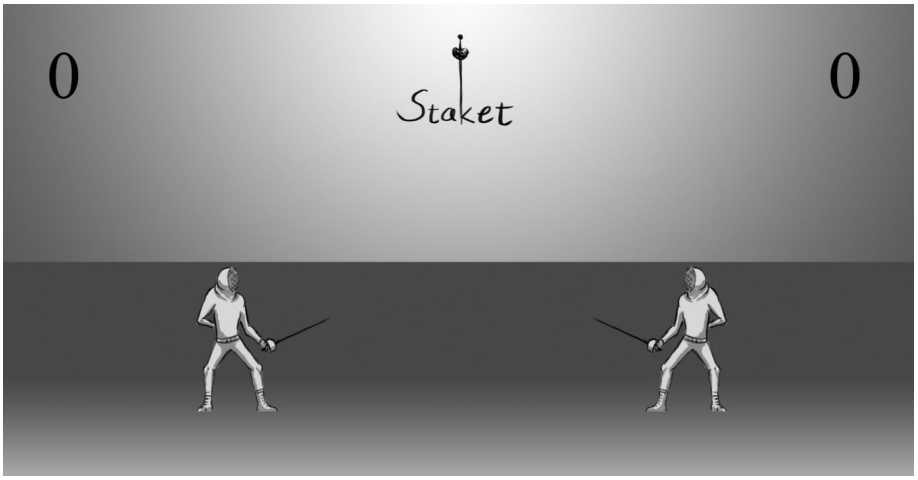
Output:

En nedvärderande låt börjar höras genom högtalarna för att påvisa användarens dräggighet. Samt visas en informativ ruta som förklarar tangenter och regler som sedan kan avisas med ett godtyckligt knapptryck för att sedan återgå till ärofylld kamp.

Resultat



Båda fäktarna står redo för att börja slåss. Härifrån kan spelarna göra lite vad de vill inom spelets ramar. Exempel kan vara saker såsom att gå framåt för att attackera motståndaren eller kanske dansa runt på (gå framåt och sedan bakåt väldigt fort) för att förvirra fienden.



Här har det båda fäktarna utkämpat sin kamp ett tag och den högra spelaren verkar ha det klara övertaget.

Jämförelse

Vi har hittills ingen egentlig instruktionsskärm då vi ansåg att det skulle vara överflödigt. Har du datorvana nog att starta spelet (från command-line) kan vi anta att spelaren kommer gissa att man ska använda WASD och piltangenterna för att spela spelet. Utöver denna lilla förändring är inget annorlunda.

Testplan

Planen

Vi planerar att ge två personer en anledning att slåss och sedan låter dem göra det. Fast på låtsas, såklart. Försökskaninerna kommer slåss minst 2147483648 gånger och varje gång testa att avvärja, attackera och förflytta sig runt på planen; de kommer även under denna tid glömma bort vad de håller på med och därför ta fram instruktionerna.

Resultat

Vi har testat spelet genom att låta oss själva samt nära och kära spela mot oss eller mot varann. Dessa har gått till på sådant sätt att man valt ett arbiträrt tal att spela till, exempelvis ”först till 10” där den spelare som nått 10 poäng först är den som utses till vinnare.

Jämförelse

Relevanta skillnader är sådana såsom att inga tester utförs på intruktionsskärmen (då den inte finns) och att vi sannolikt inte kommer utföra MAX_INT antal tester.

Programdesign

Planen

Staket

```
Staket är sammanslagningen av alla bakomliggande klasser för att starta spelet.

    loadPics
        Hämtar bilder.
```

Character

```
Character hanterar information om en värjhållare, bilder, position och poäng etc.

         // Hämtar karaktärens x-position. int X()
         // Hämtar karaktärens statiska y-position (karaktärer kan inte hoppa). int Y()
         // Hämtar aktuell animation ifrån bakomliggande sprite. Sprite getSprite() }
```

Sprite

```
Sprite är en samling av bilder för att kunna skapa animationer.

         // Hämtar in en animation ifrån en bildfil. void setSprite(String path)
         // Hämtar en speciell bild ifrån en sprite. Image getSprite(int num) }
```

KeyInputHandler

```
KeyInputHandler hanterar knapptryckningar.
         // Hookar ifall en knapp blir nedtryckt. public void keyPressed(KeyEvent e) }
```

Screen

```
Screen gör ett fullskärmsfönster.
         void setFullScreen(DisplayMode dm, JFrame window)
```

Resultat

Staket

```
Staket är själva spelklassen där mycket gemensam data finns samlad.
Spelkaraktärerna, deras startkoordinater till exempel.
    redraw
        Forcera en omritning av skärmen.
    paintComponent
        Override funktion som ritar objekt till skärmen.
    positionChallengers
        Förflyttar karaktärerna till deras start positioner.
    loadPics
        Ladda in karaktärernas animationer och bakgrunden.
    run
        Huvudmetoden som sedan startar skärmen.
```

Character

```
Character hanterar allt om enstaka karaktärer och deras placering.
    isBlocking
        Håller den här karaktären för nuvarande på att avvärja attacker?
    attackAnim, blockAnim
        Spela upp attack / blockerings animationen.
    moveRight, moveLeft
        Förflytta karaktären.
    stopAnimations
        Stoppa alla animationer.
```

KeyInputHandler

```
KeyInputHandler hanterar alla knapptryckningar och logiken som inträffar vid
sagd knapptryckning. Attack, blockering och förflyttningar lyssnas efter
här.
    charActions
        Tar reda på vilken spelare som tryckte på knappen för att
        sedan utföra kommandot. En convenience function som generaliserar
        koden för de båda spelarna.
    keyPressed
        Override function son krävs för att hooka på knapptryckningar.
    moveBack, moveForward
        Convenience functions för att flytta spelarna bakåt respektive
        framåt (vilket är motsatt för varandra eftersom de står ansikte mot
        ansikte).
    stopTimers
        Stanna båda fäktarnas animationer.
```

Screen

```
Screen möjligör fullskärmsrendrering.
    setFullScreen
        Självförklarande.
Spelkaraktärerna, deras startkoordinater till exempel.
```

Jämförelse

Skillnaderna mellan planen och resultatet är framför allt att resultatet är mer specifikt i vilka metoder som används då vi faktiskt skrivit koden vid det här laget. I övrigt avskaffade vi klassen Sprite() då det inte behövdes en egen klass för det.

Tekniska frågor

Planen

Hur ska vi hantera multipla knapptryckningar samtidigt? En ArrayList av knapptryckningar som töms var 50ms?

Hur ska poängräkningssystem se ut? Pongaktigt poängsystem?

Hur ska vi visa hjälptext för nya spelare? Konstant textremsa någonstans på skärmen?

Hur ska hitboxsystemet fungera? Beräkna avståndet mellan värja och karaktär. Är det <= 0 så är det exempelvis en träff. Hitboxsystemet kommer kräva trådar för att registrera attacker mot en karaktär som går in i en värja.

Hur ska attacker fungera? Se hitboxsystemet.

Hur ska avvärjning av attacker fungera? Ifall två värjor under någon gång under animationens gång har avstånd <= 0 till varandra sker en avvärjning och de båda karaktärerna förblir oskadda. Vid en avvärjning körs karaktärernas animationer baklänges motsvarande hur länge de kört. Dvs. ju senare en avvärjning sker desto mer fördelaktigt är det

Hur ska förflyttning fungera? Karaktärernas hitbox och animationer förflyttas i x-led.

Resultat

Är det något som fattas? Musik/Ljudeffekter?

Är det något som inte fungerar som det ska? Eventuella buggar osv.

Jämförelse

Vi lyckades lösa alla de problem vi hade på olika sätt. Exempel på problem vi löste eller kringgick skulle kunna vara att vi aldrig använde en egen Spriteklass då det inte behövdes utan alla sprites (bilder) importerades direkt till Staket.java (main klassen) utan problem. Det fanns redan javatyper för bilder som vi lade in i en hashmap för att göra dem lättåtkomliga.

Vi löste problemet med tråder vi trodde vi skulle ha. Detta gjordes med javatypen Timer. Det är Timer som står för alla delays i spelet.

Arbetsplan

Planen

Bly Spritekonstnär: Richard Uggelberg

Bly Utvecklare (fram till quarnevalen): Henry Eklind

Bly Utvecklare (under quarnevalen): Richard Uggelberg

Bly Utvecklare (efter quarnevalen): Henry Eklind

Till Övning 12 (2/5)

"Inlämning av projektplan samt muntlig lägesrapport."

Skriva klart detta dokument. Fixa en fungerande utvecklingsmiljö. Lyckas förflytta en bild på skärmen.

Till Övning 13 (9/5)

"Muntlig lägesrapport."

Richard: Fixa sprites Eventuellt logotyp (splash screen!) Fixa git Fixa musik

Henry: Skriva Character-klassen Skriva Sprite-klassen Implementera förflyttning Implementera poängsystemet Implementera attacksystemet Trådar i Java för hitboxsystemet

Till Övning 14 (16/5)

"Inlämning av slutrapport, färdig kod samt muntlig slutrapport."

Richard: Fixa animationer Skriva slutrapport Göra det som är kvar Fixa ljudeffekter Implementera 8dimensionell blodspillsmotor

Henry: Implementera avvärjning Skriva slutrapport

Resultat

Den arbetsplan vi satte ut har inte följts till punkt och pricka vad gäller datum och tid men alla uppgifter äro utförda förutom ljudeffter och den 8dimensionella blodspillsmotorn.

Jämförelse

Det uppgifter vi satte ut har uppfyllts, alltså finns inget mer att tillägga.

Sammanfattning

Sammanfattningsvis anser vi att detta har varit ett lycked projekt då hann klart med det vi ville inom den givna tidsramen utan att sätta en alldeles för låg ambitionsnivå. Spelet ser till stor del ut som vi vill att det gör och det har de funktioner som vi anser vara essentiella.

Skulle vi fortsatt arbeta på projektet skulle det vara mycket små förändringar i sådana fall. Om man vill ändra spelets utseende och funktionalitet skulle man behöva göra stora förändringar eller kanske skriva om det från början helt och hållet. Förändringar vi kan göra är att exempelvis göra bättre animationer eller "finare" sprites men ingen av dessa skulle tillföra speciellt mycket till spelupplevelsen.