TDP005 Projekt: Objektorienterad Programmering

Kodgranskning

Författare
Alexander Stolpe, alest170
Fredrik Jonsén,frejo105

Höstterminen 2014
Version 1.0

Innehållsförteckning

1. Revisionshistorik

Ver.	Revisionsbeskrivning	Datum
1.0	Första utskick av kodgranskningen.	141210

2. Medverkande grupper

- Fredrik Jonsén, Alexander Stolpe
- Goran Esmail, Harald Grant
- Emelie Olmås, Kenneth Börjesson

3. Granskningsmöte

Mötet hölls den 5 december 2014.

Det gick till på det sätt att varje grupp gick igenom sin kod för de andra grupperna, förklarade tanken bakom denna samt upplägget. Medan den redovisande grupper gjorde detta så kommenterade de andra grupperna med positiv samt negativ feedback och förde anteckningar. Därefter bjöd grupperna in varandra till respektive projekts gitlab-repo.

Mötet gick snabbt och smidigt med mindre diskussion runt koden då det antingen inte var så mycket kod skriven inom projektet eller att den var väldigt självbeskrivande.

4. Kodgranskning

4.1. Grupp 1. Harald och Goran

Projektet i allmänhet är väldigt bra med välgenomtänkt och snyggt skriven kod. Kommentarer används mycket även om den kan variera i stil.

Dock är mappstrukturen väldigt rörig, det ligger källkodsfiler direkt i roten och det är inte riktigt klart vad de olika mapparna innehåller utan att gå in i dom och se efter manuellt.

4.1.1. Klasser

Övergripande väldigt bra struktur och väl genomtänkta klasser. Man sätter endast de funktioner som är direkt relevanta för användaren tillgängliga och gömmer undan de interna medlemmarna över lag. Det används polymorfi på ett effektivt sätt och man har inte använt sig av för långa arvträd.

Nämnvärda klasser:

Game:

- Bra klassnamn som tydligt beskriver dess funktionalitet.
- Minimalt interface där endast konstruktor och en run-metod är tillgänglig för användaren. Vilket är väldigt bra.
- Klassen är helt inkapslad då det ända som användaren är tänkt att använda är medlemsfunktionen run.
- Klassen innehåller en del menu pekare som inte ser ut att fylla något syfte, och används aldrig. Dock är inte menu klassen gjord ännu och det är då svårt att se detta i dagsläget.

Game_Object:

- Klassen är uppdelad genom aggregation i ett Render_Object och ett Physics_Object, vilket ger en väldigt bra abstraktion.

Vec2:

- Klassen representerar en vektor vilket tydligt märkt på namnet.
- Den har inga interna operationer och allt är därför publikt, inklusive medlemsvariabler då de inte har några otillåtna värden, vilket acceptabelt.
- Klassen är dåligt kommenterad och utan kunskap om vektoroperationer så kan det vara svårt att förstå vad de olika operatorerna gör.

Line:

- Klassen använder makron för att definiera konstanter som används i klassen, vilket inte är optimalt.
- Övrigt väl genomtänkt datastruktur med nödvändiga operatorer dylikt definierat.

AABB:

- Hela klassen är publik, vilket inte är optimalt då det borde finnas restriktioner. Exempelvis så bör inte size kunna anta ett negativt värde, då de representerar boxens sidor.

Physics_Object:

- Klassen kan ta in negativa värden, fast den likt AABB skall representera den omslutande boxen. Någon sorts kontroll av detta vore en bra idé.

4.1.2. Variabler

Variabler har bra och beskrivande namn vilket gör att det oftast är väldigt lätt att sätta sig in och förstå vad variablerna innehåller och vad deras syfte i programmet är.

Dock kan nämnas att i flera fall så kontrollerar man inte om den data som kommer in till variablerna i konstruktorerna är giltig för klassens syfte, i huvudsak de klasser som hanterar en rektangel och som då använder en vektor för att lagra data.

4.1.3. Satser

Satserna är väl skrivna och ofta självdokumenterade, men på vissa håll i koden så skulle koden kunna dra nytta av mer beskrivande kommentarer för att tydliggöra vad de egentligen gör för att göra det mer lättläsligt samt spara tid för andra som kan tänkas gå igenom koden. Detta gäller mest de mer matematiska delarna av programmet.

Utöver det så håller de som tidigare nämnt en väldigt bra standard med tydlig och bra struktur och kodstil.

4.2. Grupp 2. Emelie och Kenneth

Under kodgranskningen så befann sig gruppen i tidigt av projektet och det fanns därför ingen större mängd att gå igenom. Det mesta av koden var för att lära sig grunderna i SDL och var därför varianter av guider och därför nämns inte koden i någon större utsträckning här.

Däremot så förstod man tydligt kodens syfte och det fanns med kommentarer som beskrev vad som hände i varje steg. Koden låg i sin helhet i main och det fanns då inga klasser att granska.

5. Feedback

De kommentar som vår grupp fick var i större del positiv. Grupperna ansåg att vi hade en tydlig struktur med bra variabler och abstraktion.

Dock så saknade vi helt kommentarer i vår kod, varken vanligt eller doxygen, vilket man tyckte skulle vara bra att skriva medan man kodar för att underlätta vid senare stadier. Vi hade även en del stilfel som vi inte rättat till under arbetets gång utan tänkt att detta är något man rättar till på slutet.

Det vi tog med oss från detta var bland annat att vi har skapat en mall för att i eclipse automatiskt formatera den skrivna koden med ett kommando.