LINKÖPINGS UNIVERSITET

TDP005 Projekt: Objektorienterat system

Kodgranskning

Författare

Martin Kuiper, marku849@student.liu.se
Jim Teräväinen, jimte145@student.liu.se

INSTITUTIONEN FÖR DATAVETENSKAP

Höstterminen 2019
Version 1.0

9 december 2020

1 Revisionshistorik

Ver.	Revisionsbeskrivning	Datum
1.0	Första version efter gruppvis granskning av koden.	201209

2 Möte

Grupperna träffades över nätet i ett röstsamtal där vi diskuterade uppgiften. Vi skickade sedan vår egen kod till varandra som zip-filer tillsammans med designspecifikationen. Vi bestämde att vi skulle granska varandras kod i 2 timmar och sedan återsamlas för diskussion. Mötet ägde rum den 9:e december klockan 13:00.

3 Granskat projekt

Vi granskade grupp B1s projekt vilket är ett top-down skjutarspel där man ska försvara sig från fiender som går emot spelaren.

3.1 Designspecifikation gentemot verklighet

När det kommer till vad som står i designspecifikationen gentemot vad som syns i koden och spelet så verkar det generellt som att det stämmer överens väldigt bra. Den största skillnaden är att det tillkommit en klass vi namn 'living entity' i koden som helt saknas i arvsdiagrammet.

Designspecifikationen är inte väldigt djupgående och täcker inte så mycket av spelet men det som finns att läsa går att återfinna i så gott som exakt samma form i spelet. Vi kan bara anta att dokumentet uppdaterats flitigt under arbetets gång så att det konstant innehåller den senaste versionen av funktionen.

3.2 Strukturering

I koden används många olika 'managers' som hanterar varsin slags entitet och hur de ska bete sig på spelplanen. Alla dessa verkar ärva en del av sina egenskaper från en 'basemanager' men inte alla. Vi anser att man ska se över namngivningen här för att inte skapa förvirring då alla 'managers' rimligtvis bör ärva från grundklassen.

Funktioner används ofta utan att referera till den förälderklassen där den ligger. Det blev svårt att läsa koden och förstå hur den fungerade i vissa fall när funktioner som inte har tillräckligt tydliga namn kallas av barn. Det var ibland svårt att hitta filen som funktionens definition låg i.

Just nu så hanterar varje sorts karaktär sin visuella del direkt i objektets konstruktor vilket lätt blir långt. Det kanske hade gått att bryta ut den hanteringen i en egen funktion högre upp i heirarkin för att minska koduppprepning.

3.3 Klasstruktur

Strukturen känns ibland förvirrande då det är svårt att läsa ut vilka klasser som påverkar varandra. Vi misstänker att det har att göra med manager-strukturen som har många olika barnklasser som interagerar med varandra. För att motverka förvirring så hade fler kommentarer och tydliga funktionsanrop kunnat hjälpa en utomstående person att koppla ihop allt. Viktigast av allt skulle vara ett uppdaterat arvsdiagram med samtliga klasser inritade.

3.4 Saker som funkar

Generellt så är spelet på god väg att bli både roligt och bekvämt att spela. Kontrollerna känns naturliga och rörelsen både känns och ser bra ut. Vi känner att spelet redan nu har mycket av den polering som utmärker ett seriöst spel. Oavsett effektivitet så verkar all kod stabilt skriven och det både känns och syns i spelet.

4 Vårt projekt

I denna sektion bemöts kritiken som vi fick av grupp B1 (även benämnt som granskningsgruppen) på vårt projekt.

4.1 Kommentarer

B1 påpekade att kommentarerna i koden inte var nödändiga i många fall. I vissa fall ansåg de till och med att de var störande och förklarade saker inte behöver förklaras. Vi avser att gå igenom vår kod och tänka en extra gång på hur vi gjort våra kommentarer och lägga till, ta bort och redigera där det är passande.

4.2 Const och Ref(&)

B1 tyckte även att vår användning av 'const' och referens '&' var sporadisk och att vi missat möjliga referenser och konstanta variabler på ett par ställen. Vi är väl medvetna om att vi ofta glömmer att tänka över hurvida konstanta referenser kan användas i någon kombination. Planen är att åtgärda det i samma genomgång som kommentarerna.

4.3 Destruktorer

Vi fick en kommentar om faktumet att alla våra klasser implicit använder 'default' destruktorerna. Granskningsgruppen var oroliga att detta skulle kunna leda till minnesläckor då vi inte hanterar vad som ska hända med pekare när objekten raderas. Vi är inte lika oroliga då vi endast använder smartpekare som i teorin ska hantera sig själva. Test med Valgrind har inte heller visat på några minnesläckor.

4.4 Kodstruktur

Granskningsgruppen hade i sin genomgång hittat ett par ställen där indenteringen på koden var felaktig. Detta finner vi högst otroligt då vår IDE hanterar allt som har med indentering att göra men även detta kommer att undersökas vid genomgången.

Däremot hade de ett mycket bra tips på hur man kan strukturera sina funktioner i H-filerna. Igenom att tänka på funktionerna utifrån vilka som logiskt hör ihop så kan man styckesindela koden och få den mycket mer läsbar. Detta ska vi definitivt göra vid genomgång av koden och i framtiden.

4.5 Returvärde

Denna punkt var egentligen resultatet av ett missförstånd då granskningsgruppen trodde att vår 'update' funktion returnerade ett boolianskt värde utan anledning. Vi förtydligade varför detta är fallet och då tog de tillbaka kommentaren. Men det pekar på ett annat problem där det inte är tydligt nog varför 'update' returnerar en boolian, något som vi ska undersöka hur vi löser. Just nu lutar det mot att göra tydliga kommentarer och att srkiva bra dokumentation.