**Mr. Glass**

Olio-ohjelmointi 1

Olli Nissinen H8593

Miika Avela H4211

Dokumentointi

4/2015

Ohjelmistotekniikan koulutusohjelma

Tekniikan ja liikenteen ala

Sisällysluettelo

[1 Yleiskuvaus 2](#_Toc436244400)

[1.1 Projektin kuvaus ja toimeksianto 2](#_Toc436244401)

[1.2 Pelin kuvaus 2](#_Toc436244402)

[1.3 Käytetyt teknologiat 2](#_Toc436244403)

[1.4 Tekijät 3](#_Toc436244404)

[2 Työnjako ja vastuualueet 3](#_Toc436244405)

[3 Aikataulu 4](#_Toc436244406)

[3.1 Suunniteltu ajankäyttö 4](#_Toc436244407)

[3.2 Toteutunut ajankäyttö 4](#_Toc436244408)

[4 Alkuperäisen- ja toteutuneen vertailua 9](#_Toc436244409)

[4.1 Alkuperäinen- ja toteutunut UML- kaavio 9](#_Toc436244410)

[4.2 Alkuperäinen kenttä ja toteutuneet kentät 10](#_Toc436244411)

[4.3 Suunniteltu ja toteutunut valikko 13](#_Toc436244412)

[5 Testaus, ongelmakohdat ja niiden ratkaisut 14](#_Toc436244413)

[6 Itsearviointi projektista 15](#_Toc436244414)

# Yleiskuvaus

## Projektin kuvaus ja toimeksianto

Toimeksiannon saimme koulun olio-ohjelmointi opettajalta, Ari Rantalalta. Työn tavoitteena oli tutustua C++ ohjelmointiin ja oppia mahdollisimman paljon 2D pelin tekemisestä SFML kirjastolla. Aiheenamme on perinteinen 2D tasohyppelypeli, missä hahmo yrittää päästä paikasta A paikkaan B osumatta esteisiin. Peli on suunnattu kohderyhmältään tasohyppelyn ystäville.

## Pelin kuvaus

Mr. Glass2 on C++ tehty perinteinen tasohyppelypeli, missä pelin ideana on saada päähenkilö, Mr. Glass ¤%/¤%/¤%/TARINAN IDEA TÄHÄN!!!!!=¤!(!”&!%. Reissu ei kuitenkaan suju aivan ongelmitta, sillä matkan varrella sankarimme kohtaa erilaisia vaaroja joista hänen pitää selvitä yhtenä kappaleena. Vaikutteita pelin luonne on saanut vanhoista tasohyppelypeleistä kuten super mario world ja ’n’ The Way of the Ninja.

## Käytetyt teknologiat

Vielä mietintävaiheessa pelin ulkoasua pohdittiin piirtelemällä laatikoita NinjaMock:lla, mutta tämä todettiin nopeasti hankalakäyttöiseksi ja melko rajoittuneeksi ohjelmaksi, joten suunnittelu tapahtui pääosin Paintin ja lehtiön avustuksella. Varsinainen graafinen toteutus suorittui pieniltä osin koulussa Adobe Fireworks CS6:lla, sekä pääosin kotona Adobe Photoshop CS6:lla. Pixel-art –tekniikoiden opiskelussa tietoa haettiin Googlen välityksellä. Luokkakaavion suunnittelussa käytettiin heti alusta alkaen Violet UML editoria. Kaikki ohjelmointi ja testaus tapahtui VisualStudio 2015:sta. Projektinhallinta ja -seuranta toteutettiin Githubin projektikansiolla, mitä päivitettiin jokaisten muutosten jälkeen, sekä yhteydenpito samanaikaisesti hoidettiin pääosin Mumble VoiP-ohjelman välityksellä. Myös projektipalavereja pidettiin kasvokkain muutaman kerran viikossa.

## Tekijät

Projektin tekijöinä toimivat kaksi Jyväskylän ammattikorkeakoulun, ohjelmistotekniikan linjan, toisen vuoden opiskelijaa. Miika Avela ja Olli Nissinen. Olio-ohjelmointi 2 kurssille tultaessa kummankin C++:n lähtötaso perustui ohjelmoinnin perusteet kurssiin.

# Työnjako ja vastuualueet

Vastuualueet jaoimme alustavasti suunnitelman jälkeen siten, että molemmille tulisi suunnilleen yhtä paljon tehtävää. Ongelman ilmestyessä miettisimme ratkaisua kuitenkin yhdessä. Alueet jakautuivat seuraavasti:

**Olli:**

Collision, tilemap.

**Miika:**

Valikot.

**Yhteistä:**

Pixel Art, dokumentointi, UML-kaaviot, peli-idean kehittäminen, kenttäsuunnittelu.

# Aikataulu

## Suunniteltu ajankäyttö

Lähtökohtaisena ajankäytön suunnitelmana oli käyttää tunteja seuraavasti:

Ideointi ja Suunnitelman teko 5h

Graaffinen suunnittelu 15h

Pixel art 20h

Ohjelmointi 85h

Testaus 10h

Dokumentaatio 5h

**yht 140h**

## Toteutunut ajankäyttö

Todellinen ajankäyttö ja työn edistyminen. Pelin testaus on sisällytetty ohjelmointiin:

**Viikko 48**

Aiheeksi valittiin 2D tasohyppelypeli. Suunniteltiin pelille tarina ja alustava luokkakaavio.

Olli: Ideankehittäminen 2h, Collision 4h, Tilemap 3h, Pelaaja luokka 1h, Dokumentaatio 1h.

Miika: Ideankehittäminen 2h, Pelaaja luokan liikkuminen kellon mukaan 1h.

**Viikko 49**

**Viikko 7**

**Viikko 8**

**Viikko 9**

Ajastimen vaihto tickseihin pitkän mietinnän jälkeen, luokkakaavion viimeistelyä, lisää objektien piirtämistä, hyppääminen, painovoima, sekä alustavan suunnitelman palautus.

Olli: Ohjelmointia 10h, Suunnitelman viimeistely ja palautus 1,5h, pixel art 3h

Miika: 2h pixel art, Suunnitelman viimeistely ja palautus 1,5h, ohjelmointia 5h

**Viikot 10 - 11**

Päävalikon alavalikot ja näiden toiminnallisuus.

Miika: 6h ohjelmointia, 4h pixel art

**Viikko 12**

Ensimmäisen kentän luominen, collisionin bugin debuggaus ja korjaaminen.

Olli: 9h ohjelmointia

**Viikko 13**

Objektien luomista, sirkkeli, piikit jne. ja niiden collision. Menun valitsin ja sen animaatio.

Olli: 6h ohjelmointia

Miika: 4h ohjelmointia

**Viikko 14**

Uusia objekteja, näiden toiminnallisuus ja pidemmät testausajat.

Olli: 7h ohjelmointia

Miika: 6h ohjelmointia

**Viikko 15**

Peli- ja päävalikkotaustan tekoa ja muiden kuvien piirtämistä, collisionin tarkentamista.

Olli: 6h ohjelmointia, 2h pixel art

Miika 3h ohjelmointia, 2h pixel art, 2 graafinen suunnittelu

**Viikko 16**

Maapalikoille muotojen haku, mediatracker, animointia, shattered valikko, mitä tapahtuu hajotessa, pistelaskuri, uuden UML- kaavion teko, kanuunan ja kanuunan kuulan tekeminen ja piirtäminen.

Olli: ohjelmointia 7h, 1h pixel art

Miika: ohjelmointia 9h, pixel art 2h

**Viikko 17**

Pistelaskurin kirjoitus tiedostoon ja luku uuden kentän alkaessa, hajoamisanimaatio + interaktivointi, kolme tasoa lisää, cannonball ja cannon korjausta, liekkiobjectin teko ja collision, virheiden etsintää ja korjausta. Esimerkiksi hypystä ja blockista löytyneiden bugien korjaus, highscore-valikko.

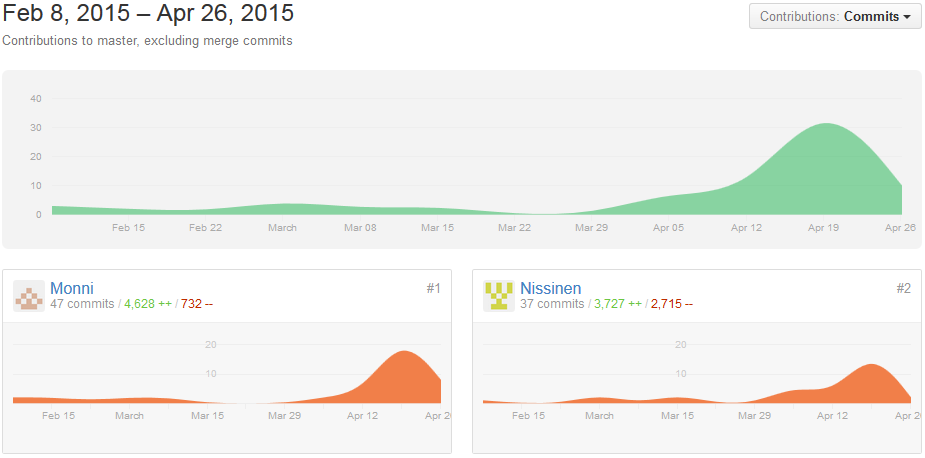
Miika: ohjelmointia 15h, dokumentaatiota 2h, pixel art 2h

Olli: Dokumentaatiota 8h, ohjelmointia 3h

**Yhteensä:**

Olli: 80,5h

Miika: 78,5h

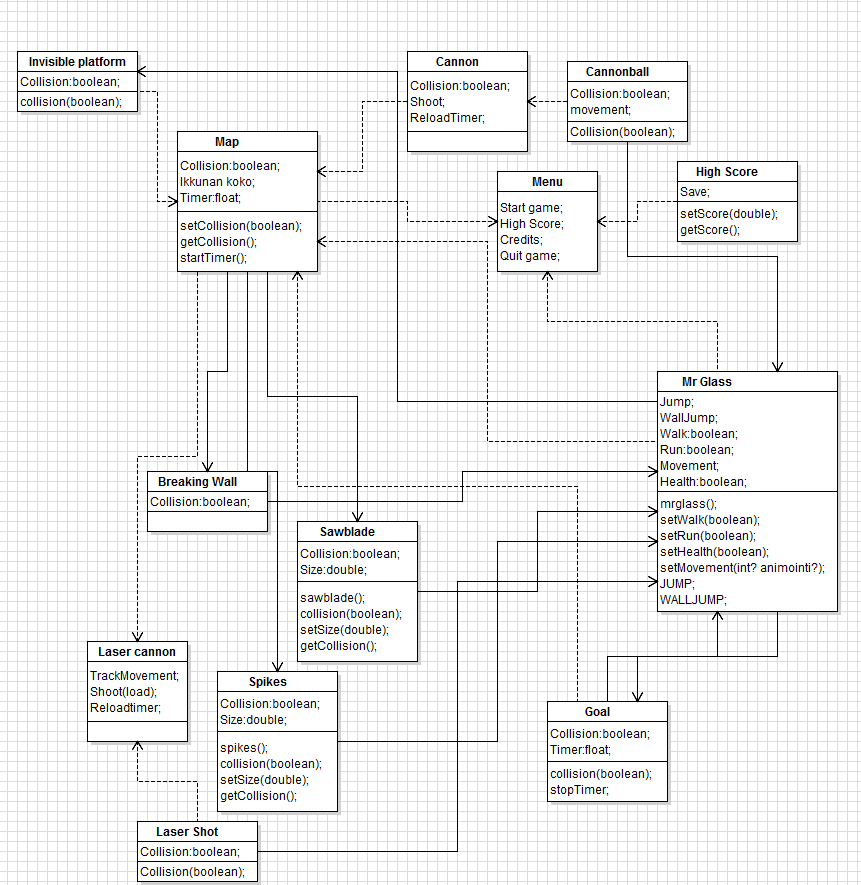


Taulukko 1: Github projektin edistyminen

Vaikka taulukosta 1 voisi olettaa projektin varsinaisen työn alkaneen vasta viime hetkellä, näin ei kuitenkaan ole. Todellisuudessa teimme projektia suhteellisen tasaisesti. Alussa päivitykset olivat huomattavasti suurikokoisempia ja lopussa pienempiä, sekä bugien korjausta.

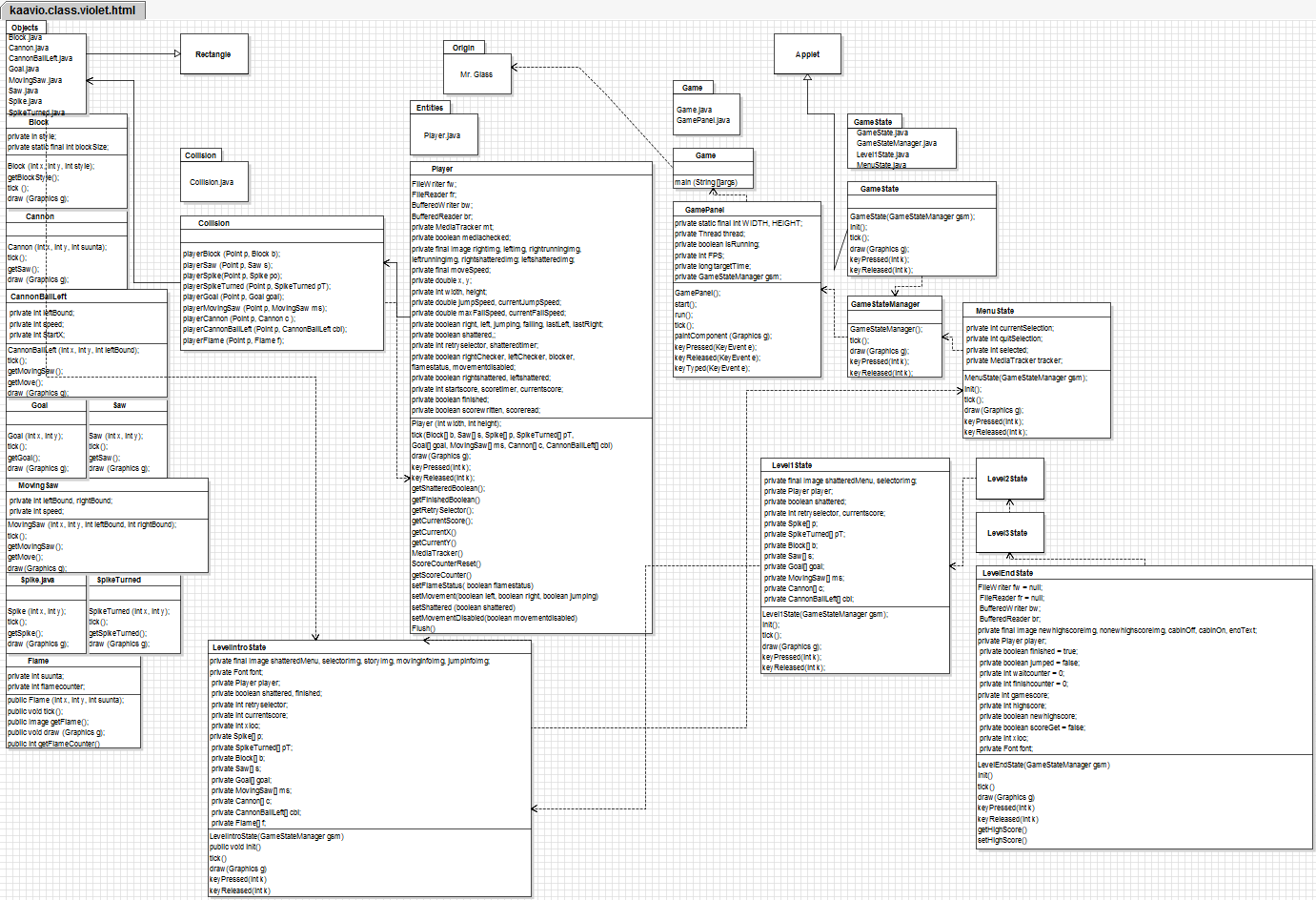
# Alkuperäisen- ja toteutuneen vertailua

## Alkuperäinen- ja toteutunut UML- kaavio



Kuva 1: Alkuperäinen UML-kaavio

Kuvan 1 alkuperäinen UML-kaavio on jälkeenpäin katsottuna hieman hutaisten tehty, kun vertaa lopulliseen. Kyseisessä kaaviossa ei ole huomioitu peliä alustavia tekijöitä ollenkaan, eikä siinä ole selkeää ”punaista lankaa”.

  
Kuva 2: Lopullinen UML-kaavio

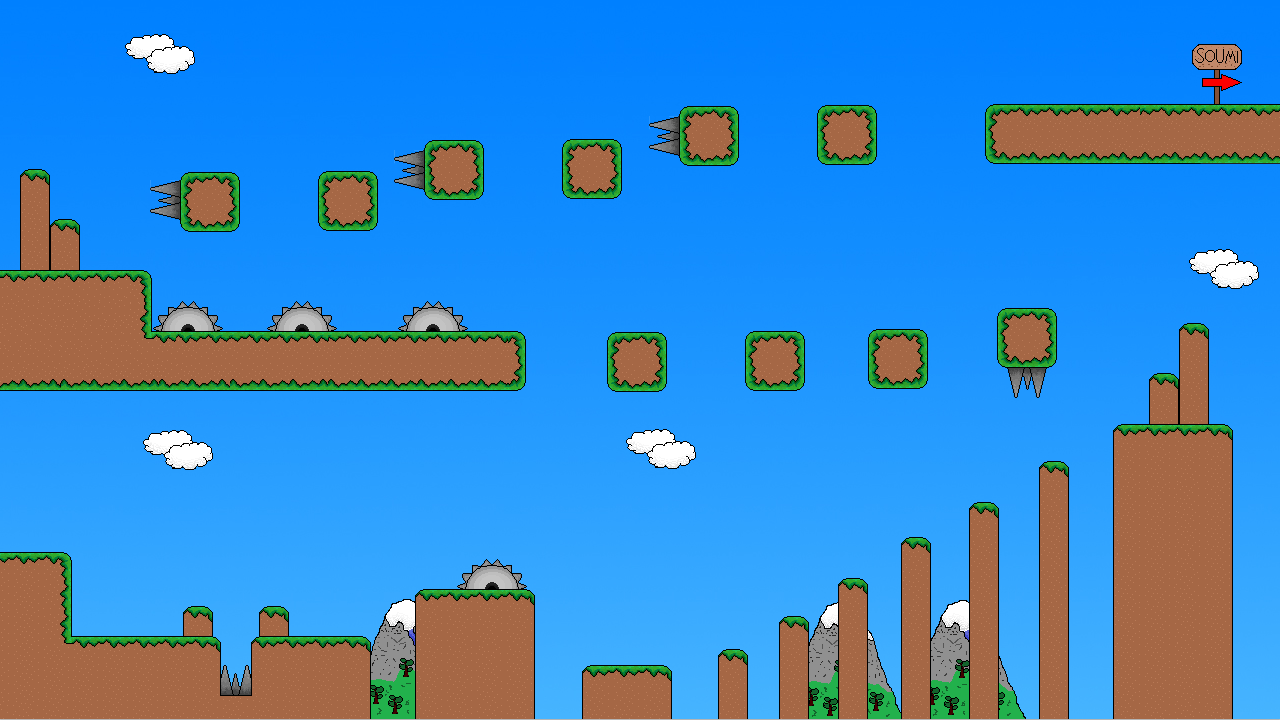
Kuvan 2 lopullisessa UML-kaaviossa on huomioitu peliä alustavat tekijät ja se on huomattavasti selkeämpi kokonaisuus, kuin alkuperäinen. Lopullisessa kaaviossa on havaittavissa edellä mainittu ”punainen lanka”.

## Alkuperäinen kenttä ja toteutuneet kentät

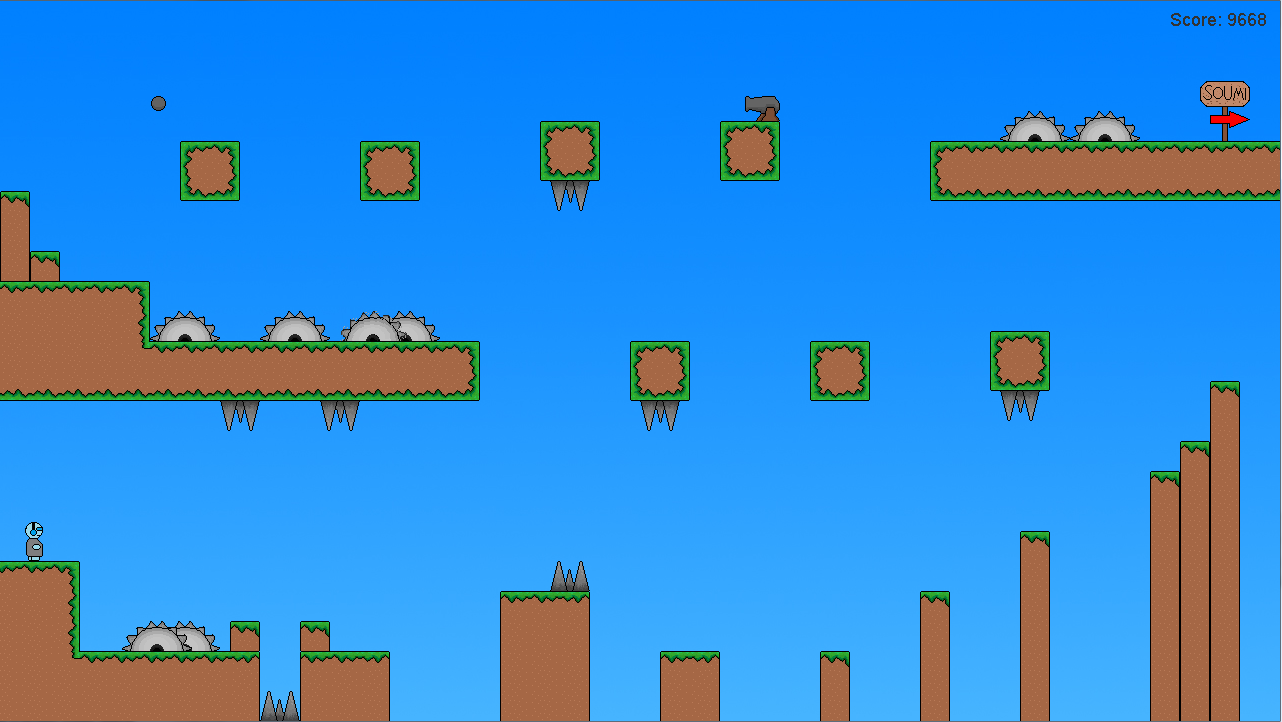
Alkuperäinen idea oli tehdä yksi kenttä, mutta teimme neljä pelattavuuden takia. Tutorial kentän, kaksi vaikeampaa kenttää ja loppuun ”animaatiokentän”. Yksi kentistä eroaa alkuperäisestä suunnitelmasta hyvin vähän. Tosin huomasimme kentän olevan turhan staattinen ja lisäsimme siihen joitakin objekteja pelattavuuden ja ulkonäön vuoksi, kuten alla olevista kuvista voidaan huomata.



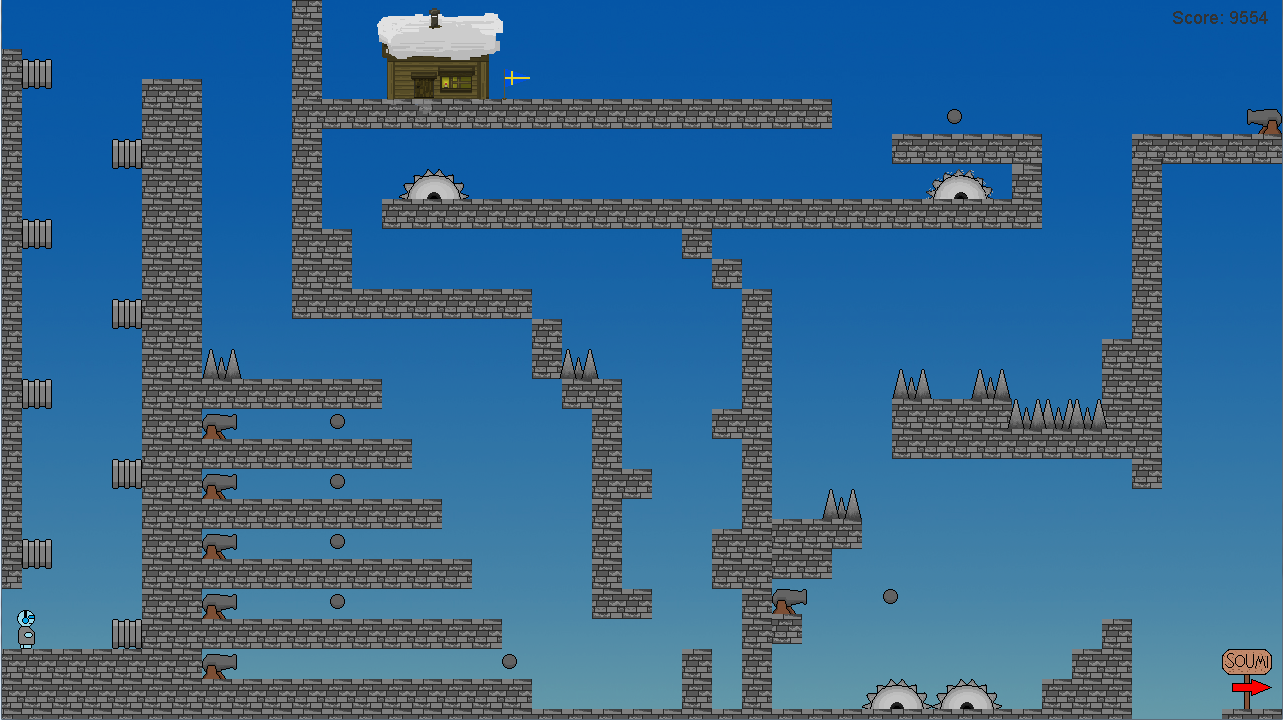
Kuva 3: Tutorial kenttä



Kuva 4: Alkuperäinen kenttäsuunnitelma



Kuva 5: Lopullinen toinen kenttä



Kuva 6: Kolmas kenttä



Kuva 7: Animoitu kenttä

## Suunniteltu ja toteutunut valikko

Alkuperäisessä suunnitelmassa päävalikon ulkoasu näytti seuraavanlaiselta:

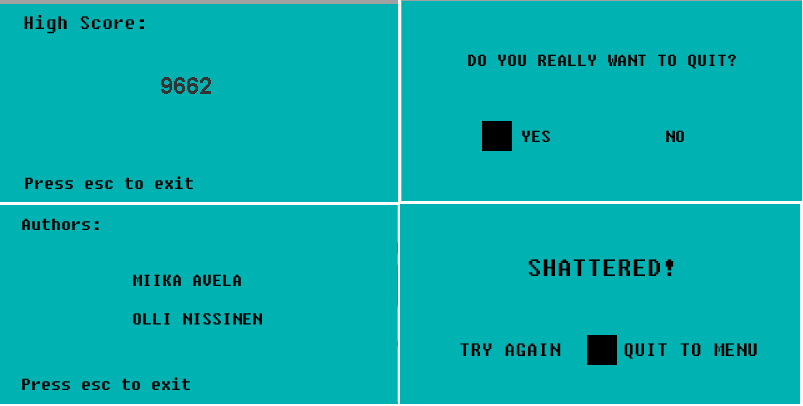


Kuva 8: Alkuperäinen päävalikko

Myöhemmin mietittyämme totesimme valikon olevan hieman tylsännäköinen ja päätimme parannella sitä visuaalisesti ja käytännöllisistä syistä luoda alavalikot:



Kuva 9: Lopullinen päävalikko



Kuva 10: Alavalikot ja ilmoitukset

# Testaus, ongelmakohdat ja niiden ratkaisut

Vaikka testasimme peliä aina kun lisäsimme siihen jotain uutta, ei ongelmakohdilta vältytty. Varsinkin projektin loppuvaiheessa testasimme peliä jatkuvasti, jotta löytäisimme mahdolliset virheet pelattavuudesta. Testautimme peliä myös ystävillä, jotta saisimme pelattavuudesta mahdollisimman hyvän. Pieniä virheitä löytyi mutta korjasimme jokaisen minkä löysimme. Suurimmaksi ongelmakohdaksi muodostui ajastimen käyttö, mikä sai pelin hidastuessa välillä hahmon jumittumaan seinän sisälle kokonaan. Tämä ratkaistiin vaihtamalla ajastin tickseihin, jolla asia korjaantui. Ticksit pitivät pelin pyörimisnopeuden jatkuvasti samana tietokoneen nopeuden kanssa hetkellisistä hidastumisista välittämättä. Tästä huolimatta collissionissa oli vielä virhe, jota metsästettiin debuggerin kanssa. Virhe ei ollut sama vaan osoittautui, että palikan sivuseinät olivat yhden pikselin liian korkealla, mikä sai hahmon välillä pysähtymään tasaisella. Pienempään ongelmakohtaan törmäsimme kun haimme ensimmäistä tasoa menu valikosta. Sillä halusimme luoda pelin samaan ikkunaan käytettävyyden vuoksi, emmekä uutta ikkunaa.

# Itsearviointi projektista

Mielestämme projekti on ollut hyvin opettavainen ja mielenkiintoinen kokonaisuus. Tunnilla tehdyt asiat ovat olleet oikein hyviä ja hyödyllisiä esimerkkejä, on silti suuremman kokonaisuuden hallitseminen vaikeaa ilman aukotonta suunnitelmaa. Projektin edetessä olemme huomanneet UML-kaavion tärkeyden ja kuinka se helpottaa koodin kirjoittamista sekä projektin toiminnallisuuden ymmärtämistä.

Lopputuloksesta olemme erittäin tyytyväisiä ja ylpeitä, sillä peli näyttää hyvältä ja toimii erinomaisesti. Tulevaisuudessa suunnittelimme tekevämme useamman kentän, jotta tästä paketista tulisi laajempi kokonaisuus. Mainittakoon lopuksi, että kaikki pelissä käytetyt grafiikat ovat itse tehtyjä ja suunniteltuja.