Y2 CS-A1121 - TASOHYPPELY TEKNINEN SUUNNITELMA

Nimi: Essi Tallgren
Opiskelijanumero: 710756
Koulutusohjelma: Bioinformaatioteknologia
Vuosikurssi: 2
Päivämäärä: 4.3.2020

Ohjelman rakennesuunnitelma

Luokat:
Menu: Alkuvalikon sisältävä luokka. Siinä on metodit kuten QPushButton jolla voidaan valita valikon eri vaihtoehdoista mieleinen, open_controls() jolla voidaan siirtyä controls info ikkunaan ja open_level1, open_level2,... metodit joilla voidaan avata haluttu taso.

Fox: Pelihahmon tiedot ja toiminnan sisältävä luokka. Siinä on metodit kuten jump() joka määrittä hyppäämisen ja movement() joka määrittää hahmon liikkumisen vasemmalle ja oikealle.

World: Luokka jossa on 2D maailman pohja, joka on tehty laatikkomaisista olioista. Luokan metodien avulla maailmaan voi lisätä viholliset ja ympäristön. Metodeja ovat esimerkiksi add_block() jolla lisätään oliot jotka tuodaan GUI luokasta.

GUI: RobotWorldissa olleen GUI luokan tapaan tämä luokka piirtää pelin Worldin ja mahdollistaa käyttäjän vuorovaikutuksen pelin maailman kanssa. Metodeja ovat esimerkiksi add_block_graphics_item() joka lisää olion Worldiin ja add_enemy_graphics_item() joka lisää vihollisen Worldiin.

CollisionDetection: Luokka joka käsittelee törnäyksen tunnistuksen. Siinä on esimerkiksi metodi check_distance() joka tarkastaa että pelihahmo ei ole liian lähellä esinettä ja stop_speed() joka asettaa nopeuden nollaksi siihen suuntaan missä on olio johon pelihahmo koskee.

Tämän lisäksi on pienempiä apuluokkia jotka sisältävät esimerkiksi koordinaatit (Coordinates) ja muut tarvittavat pienemmät osat jotka kannattaa jättää omiin luokkiinsa.

Käyttötapauskuvaus

Käyttäjän ajaessa ohjelman, hän saapuu alkuvalikkoon jossa on usea valinta joiden välillä hän voi siirtyä W ja S napeilla ylös ja alas. Painamalla SPACE näppäintä voidaan valita yksi vaihtoehdoista, joita on “Controls” joka kertoo mitä eri näppäimillä voi tehdä pelissä ja toiset vaihtoehdot ovat “Level 1”, “Level 2”, ja niin edelleen, riippuen kuinka monta tasoa pelissä

on. Viimeisenä valintavaihtoehtona on “Quit” jolla voidaan poistua pelistä jo heti alkuvalikossa. Kun käyttäjä valitsee tietyn tason ohjelma luo tason World ja GUI luokan avulla ja päädytään pelimaisemaan, jossa on pelihahmo, ympäristö ja viholliset. Liikkuessa hahmolla päästään eteenpäin kartalla ja osuessaan ansoihin tai vihollisiin ilmestuu “Defeat” ilmoitus jossa on valinnat “Try again”, “Menu” ja “Quit” jolla voidaan joko aloittaa uudestaa, lopettaa tai mennä takaisin alkuvalikkoon. Osuessaan aarteeseen tason lopussa ilmestyy “Victory” ikkuna, jossa on valinnat “Next level”, “Quit” ja “Menu” jolla pääsee joko seuraavaan tasoon, alkuvalikkoon tai pois pelistä. Valintoja tehdessä siirrytään joko taaksepäin edelliseen luokkaan (kuten Menuun) tai eteenpäin uuteen tasoon, jolloin GUI piirtää uuden tason.

Algoritmit

- Hyppy: Alussa ylöspäin kohdistuva nopeus on nolla. SPACE näppäintä painamalla ylöspäin tulee nopeutta ja alaspäin suuntautuu kiihtyvyys, kuten normaalielämän fysiikan laeissa, joka hidastaa ylöspäin kohdistuvan nopeuden ja aiheuttaa takaisin alas putoamisen. Kun tullaan takaisin maankamaralle, törmäyksen tunnistuksen avulla huomataan ettei voida mennä alemmaksi ja alaspäin suuntautuva nopeus muuttuu nollaksi.

  Nopeus y-akselin suunnassa = v0 - a * t
  Kiihtyvuus = - a
  Paikka = x0 + v0 * t - ½ * a * t^2

- Liikkuminen: Painamalla A ja D näppäintä päästään liikkumaan sivulle, alussa nopeus kiihtyy nollasta nopeuteen x jolla hahmo liikkuu pelissä. Osuessaan esineeseen nopeus siihen suuntaan muuttuu nollaksi tai lopettamalla näppäimen painamisen nopeus muuttuu nollaksi suurella kiihtyvyydellä (eli hahmo ei liiku suuria määriä eteenpäin kun näppäimen painaminen loppuu, mutta pysähtyminen näyttää kuitenkin luonnolliselta kun se ei tapahdu välittömästi). Painamalla W näppäintä tapahtuu hyppy jonka aikana voidaan liikkua sivusuunnassa painamalla A ja D nappia. Itse pelissä S nappi ei tee mitään, vain alkuvalikossa.

  Nopeus = v

- Törmäyksen tunnistus: Kun hahmo tulee tietylle etäisyydelle kappaletta, sen kiihtyvyys pois päin kappaleesta kasvaa nopeasti ja hahmon nopeus kohti kappaletta nollaantuu kun se on siinä kiinni. Tämä puskurialue estää töksähtävän pysähtymisen.

Tietorakenteet

Karttatiedoston lukemisessa voidaan käyttää listaa jakamalla rivin merkit välimerkin kohdalta listan jäseniksi. Tällä tavalla voimme helpommin lukea tiedostoa rivi kerrallaan. Tiedot kuten

maksiminopeus, kiihtyvyys ja hyppykorkeus voidaan tallentaa liikkumista käsittelevään luokkaan luokkamuuttujiksi, jolloin näitä voidaan käyttää myös luokan ulkopuolella. Tiedot joita ei tarvita kuin yhdessä luokassa tallennetaan normaaleihin muuttujiin.

Kirjastot

Käytettävät kirjastot: PyQt

Aikataulu

Aikataulu ja alustava järjestys missä aion ohjelman tehdä:
Ohjelman pohjustus ja luokkien alustus: 4h
Grafiikkojen luominen ja suunnittelu: 5h
Menu: 3h
Taso tiedostojen suunnittelu: 1h
Worldin liikkumattomien olioiden luominen (maa, piikit…): 2h
Worldin pohjustaminen, eli liikkumattomien olioiden laittaminen sceneen (GUI): 5h
Hahmon luominen ja siihen liittyvät fysiikat: 4h
Törmäyksen tunnistus (tätä vaihetta tehdään melkein samanaikaisesti kuin seuraavaa): 5h
Hahmon laittaminen worldiin ja siihen liittyvä hienosäätö: 4h
Vihollisten luominen ja laittaminen worldiin: 4h
Voittaminen ja häviäminen sekä siihen liittyvät ilmoitukset: 4h
Hienosäätöä, osien yhteen kasaamista, lisäominaisuuksien tekemistä: 10-20h
Testaaminen: 5-10h

Yksikkötestaussuunnitelma

Aion testata esimerkiksi kartan luomisen, törmäyksen tunnistuksen, menun sekä voiton ja häviämisen. Suurimmin osan näistä toiminnoista voidaan testata systemaattisella uudelleen pelaamisella ja kokeilulla ajamalla ohjelman uudestaan ja uudestaan ja tutkailemalla liikkuuko ja toimiiko kaikki pelin osat tahdotulla tavalla. Esimerkiksi törmäyksen tunnistus voidaan testata helposti ajamalla ohjelman ja ohjaamalla pelihahmon erilaisia objekteja kohti eri kulmista ja eri nopeuksilla. Kartan luominen voidaan testata muilla tavoilla helpommin kuin nuo muut ominaisuudet. Tässä voi esimerkiksi tehdä niin, että ohjelma tarkistaa onko kartta oikean kokoinen ja onko siinä oikeat oliot ja jos jotain puuttuu tai on liikaa, tulee asiasta virheilmoitus ja myös jos kartasta puuttuu jotain erittäin oleellista, voi tehdä myös niin, ettei kartta edes avaudu.

Kirjallisuusviitteet ja linkit

https://en.wikipedia.org/wiki/Platform_game
https://en.wikipedia.org/wiki/Collision_detection
http://zetcode.com/gui/pyqt5/tetris/
RobotWorld tehtävä
Y2 kurssin materiaalit ja esimerkkiohjelmat