PROJEKTISUNNITELMA

TASOHYPPELY-peli

1.Henklötiedot

Nimi: Vadim Kuznetsov

Opiskelijanumero: 907271

Koulutusohjelma: Elektroniikka ja sähkötekniikka, tekniikan kandidaatti ja diplomi-insinööri

Vuosikurssi ja päiväys: 1. vuosikurssi 26.02.2021

2. Yleiskuvaus ja vaikeustaso

Tasohyppely on peli, jossa pelaaja ohjaa hahmoa kentän läpi hyppimällä erilaisten tasojen päälle. Tavallisesti pelissä on vaikeutettavia objektia kuten ansoja ja vihollisia. Teen pelin 2D:na.

Päätin suorittaa tehtävää ainakin keskivaikealla tasolla, kun se on ensimmäinen projektini. Samoin en halunnut sen olemaan liian helppo. Jos pystyn teen projektin vaikealla tasolla.

Keskivaikean tason kriteerit:

-graafinen käyttöliittymä.

- toimiva törmäyksen tunnistus. Ohjelman täytyy tunnistaa, kun kaksi tai useampi objekti törmäävät. Esimerkiksi, jos pelaajan hahmo juoksee estettä päin, niin ohjelman tulisi huomata tämä ja estää pelihahmon eteneminen esteen läpi.

-pelissä täytyy olla selkeät ehdot voittamiselle ja häviämiselle.

-pelissä täytyy olla vähintään yksi valmis kenttä.

-grafiikat voivat olla esimerkiksi palloja tai neliöitä.

-kiinnitä huomioita laajennettavuuteen (esimerkiksi uusien kenttien luominen).

-yksikkötestit edes jollekin osalle ohjelmaa

3. Käyttötapauskuvaus ja käyttöliittymän luonnos

Ohjelma avaa ikkunaa, minne se piirtää tason, sen objektit ja pelihahmon. Ohjelma reagoi käyttäjän painamiin näppäimiin, jotka ovat “W” ylös, “S” alas, ”A” vasemmalle ja “D” oikealle. Kun käyttäjä painaa näitä nappeja ohjelma piirtää hahmon liikettä kyseiseen suuntaan. Yleensä muut hahmot liikkuvat käyttäjältä riippumatta ja ohjelma piirtää sen. Voiton ehto on pomon voittaminen.

4. Ohjelman rakennesuunnitelma

Täällä hetkellä voin keksiä vain pari luokkaa.

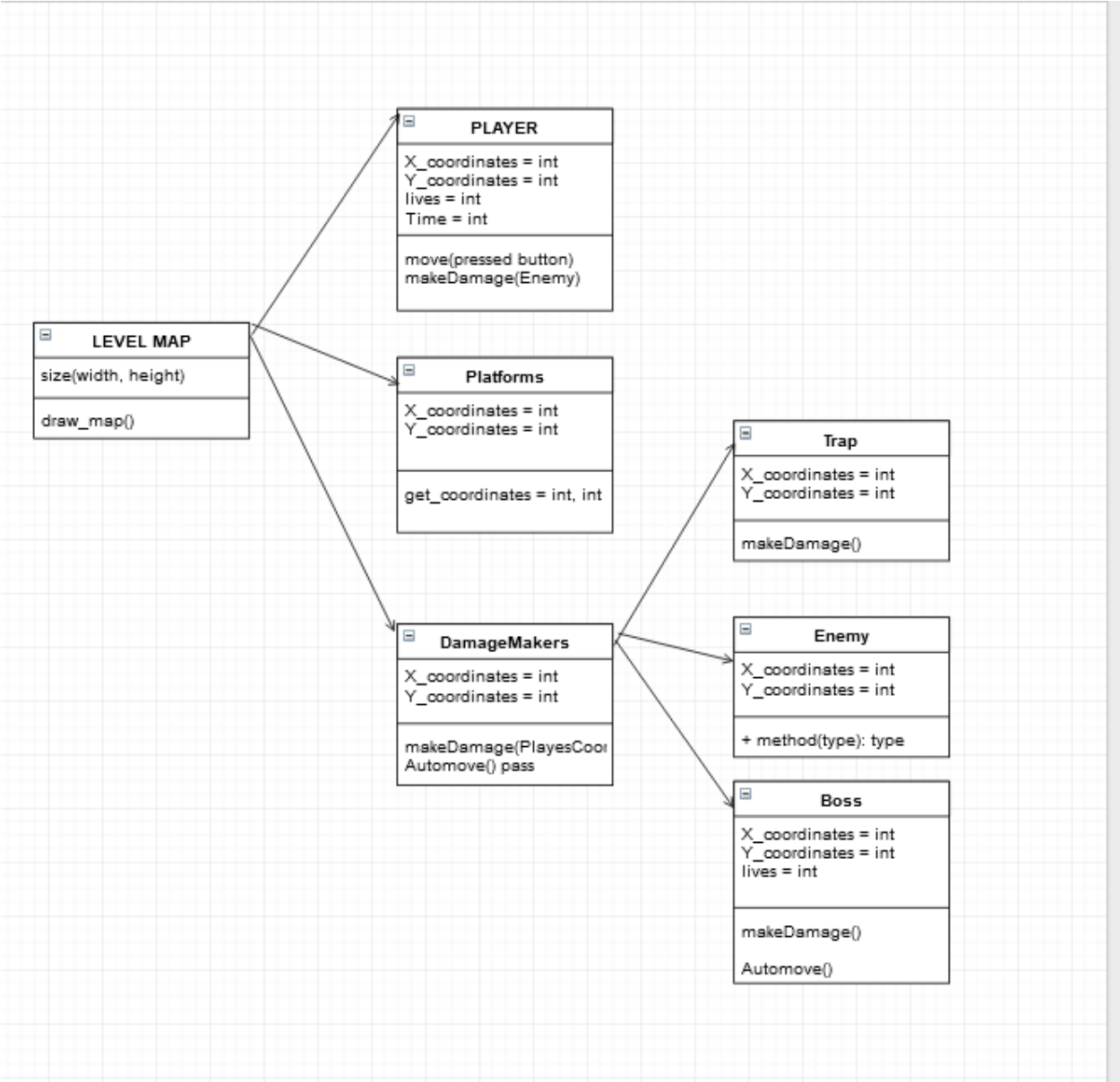
Class Player() = Kuvaa pelihahmoa ja sisältää liikkumiseen tarkoitetut metodit, kuten estojen tarkistus kaikissa suunnissa. Samoin se sisältää tiedot pelaajan parametreistä kuten elämien määrä ja pisteiden määrä. On parametri, joka kertoo sijainnin xy-koordinaatistossa.

Class Platform() = Kuvaa tasoa, josta koostu pelitaso. Siitä koostuu pintaa, jolla pelaaja liikkuu. Myös seinät ja katot koostuvat niistä. Luokka ei sisältää tärkeitä metodia ja myös sisältää tiedot omista xy-koordinaateista.

Class DamageMakers() = Kuvaa yläluokka, joka koostuu useimmista alaluokista, jotka voivat murhata pelihahmoa. Ne voivat olla liikkuvia vihollisia tai paikalla pysyviä ansoja. Suunnitelmani mukaan ne antavat iskun pelihahmolle, kun niiden xy-koordinaatit ovat samat. Tärkein metodi, joka tarkistaa objektin ja pelihahmon koordinaattien yhtäsuuruutta ja suoritta iskun, jos ne ovat samat.

Class Boss() = Kuvaa tason pomoa, joka on DamageMakers:n alaluokka. Se on tavallista suurempi vihollinen, jolla omituisen iskut ja enemmän elämiä, minkä takia sen ei voi tappaa yhdellä iskulla. Luokka sisältää vain sille kuluvat metodit, jotka kuvaavat sen erityistä liikettä ja

hyökkäystä.



5. Tietorakenteet

Minun tietojani mukaan pelissä ei tarvitaan paljon tietorakenteita. Täällä hetkellä luulen, että joudun käyttämään listoja merkeistä, jonka piirtofunktio muuta tarvittavaan muotoon. Esim. pelitaso voidaan kuvata listana, joka sisältää katkoviivoja ja nollia. Tällöin (-) on pinta ja 0 on tyhjä paikka, jonka yli on hypättävä. Se olisi helppo tapa kuvata tasoa, ennen kun ohjelma piirtää sen. Samoin se on helppo tapa lisätä uudet tasot.

6. Tiedostot ja tiedostoformaatit

Projektini ei käsittele tiedostoja.

7. Algoritmit

Ideani mukaan suurin osa metodeista perustuu xy-koordinaattien vertailun muiden objektien kanssa. Jos ne ovat yhtäsuuret jotain tapahtuu. Muiden hahmojen liike tulee tapahtumaan silmukoiden avulla. Eli täällä hetkellä luulen, että kaikki algoritmit tulevat koostumaan vain if-else-ehdoista ja silmukoista.

8. Testaussuunnitelma

Luulen, että pelejä ei voi testa muuten, kuin pelamalla niitä useita kertoja läpi. Nykyisetkin suuret projektit sisältävät virheitä julkaisun jälkeen. Mielestäni tärkeimmät asiat, mitkä ei saa sallia ovat pelin kaatuminen ja pelihahmojen putoaminen pois pelitasosta. Viimeiset kohdan voidaan korjata rakentamalla tasolla seiniä.

9. Kirjastot ja muut työkalut

Tutkimuksiani mukaan kaikki tavalliset ihmiset tekevät tämänkaltaisia projektia PyGame kirjaston avulla, koska se sisältää kaikki tarvittavat metodit ja on suosituin kirjasto pelejä varten. Mutta sitä ei saa käyttää tehtävänanton mukaan.

Ehkä voisin käyttää toistaa kirjastoa Pyglet, joka on myös tarkoitettu pelien kirjoittamisen.

Muuten joudun käyttämään vain PyQt, joka vain piirtää, tällöin teen ihan kaikki metodit itse.

10. Aikataulu

Ensin minulle on suoritettava viimeiset tehtäväkierrokset. Ehkä ehdin tehdä niitä täällä viikonloppuna.

Olen aina tehnyt ohjelmointia viikonloppuisin ja jatkan samoin. Yritän tehdä, kunnes onnistun.

Mielestäni on loogista aloittaa siitä, että ensin teen funktion, joka piirtää pelitasoa, seuraava askel on hahmon piirtäminen ja saaminen sen liikkumaan. Sitten vihollisten piirtäminen ja saaminen niitä liikkumaan itsestään. Sitten vuorovaikutusmetodien kirjoittaminen. Sitten viimeistely ja sitten testaus.

11. **Kirjallisuusviitteet ja linkit**

Netti on täynnä ohjeita siitä, miten tehdään tasohyppelypeliä, mutta niissä kaikissa käytettään Pygame, mutta ehkä silti voin oppia niistä jotain ja kirjoittaa metodia itse.

https://habr.com/ru/post/193888/

12. Liitteet