Aalto-yliopisto
Sähkötekniikan korkeakoulu
Elektroniikka ja Sähkötekniikka

Dokumentaatio

Defence

Samuli Mononen
426477
3. vuosikurssi
5.5.2017

Sisältö

1 Yleiskuvaus

Projektin aiheena on pikkupeliformaatti (tower defence), jossa erilaiset viholliset yrittävät päästä ennalta määritettyä reittiä pitkin maaliin. Pelaajan tavoitteena on estää vihollisten liike asettamalla reitin läheisyyteen torneja, joiden tehtävänä on ampua ja siten tuhota viholliset, ennen kuin ne saavuttavat maalin.

Pelissä kilpaillaan keräämällä pisteitä ja etenemällä mahdollisimman korkealle tasolle. Pelaaja saa pisteitä tuhoamistaan vihollisista ja etenee seuraavalle tasolle, kun tason jokainen vihollinen on joko tuhottu tai päässyt karkuun, eli maaliin. Peli päättyy, kun kymmenen vihollista on päässyt karkuun pelaajan torneista huolimatta.

Pelin vaikeusaste muuttuu luonnollisesti pelin edetessä. Vihollisten määrä ja voimakkuus kasvavat, mutta vastaavasti pelaaja voi pelin edetessä rakentaa tehokkaampia torneja. Pelaajan tulee kuitenkin taktikoida puolustuksensa tarkoin, sillä torneja voi rakentaa vain rajallisen määrän. Pelaaja tarvitsee tornien rakentamiseen rahaa, jota hän saa tuhoamalla vihollisia. Vaikka pelin loppupuolella pelaaja on saanut paljon rahaa, ja kyennyt siten rakentamaan paljon torneja, häneltä voi myös loppua tila tai tornien kantama kesken, sillä torneja ei voi rakentaa päällekkäin.

Pelaaja voi valita kolmesta eri vaihtoehdosta, "Easy", "Medium" ja "Hard", joiden vaikeusaste tulee suoraan siitä, että kenttä on lyhyempi, eli pelaajalla on vähemmän aikaa torjua viholliset.

2 Käyttöohje

Ohjelma käynnistetään komentoriviltä syöttämällä komento "python main.py". Tätä varten tulee olla asennettuna python 3.6 ja grafiikkakirjasto QT:n versio 5.8. Käynnistyksen jälkeen käyttäjälle avautuu koko ruudun kokoinen ikkuna, jossa on kaksi näppäintä ja ohjeteksti. Vasemmanpuoleinen kolmio kuvaa "Play-nappulaa, jota klikkaamalla käyttäjä siirtyy valitsemaan vaikeusastetta. Oikeanpuoleinen kuvake kuvaa "High Scores-nappulaa, jolla käyttäjä voi siirtyä tarkastelemaan ennätystuloksia.

Mikäli käyttäjä siirtyy ennätystuloksiin, hän voi siirtyä takaisin alkuvalikkoon oikeasta alakulmasta löytyvää kuvaketta klikkaamalla.

Ennen kuin pelaaja pääsee siirtymään itse pelitilaan, hänen tulee valita kenttä kolmesta eri vaihtoehdosta (Easy, Medium, Hard). Kentät eroavat toisistaan vain siten, että ne ovat eri pituisia.

Pelitilassa käyttäjä ohjaa peliä lisäämällä pelialueelle "torneja", joiden tehtävänä on torjua pelialueella liikkuvat viholliset. Käyttäjä lisää tornin klikkaamalla tornin kuvaketta ruudun alareunasta ja tämän jälkeen klikkaa pelialueelle ilmestyineitä rakennuskuvakkeita, jolloin torni alkaa rakentua pelialueelle. Klikkaamalla tornia käyttäjä voi vaihtaa perustetta, jolla torni valitsee kohteensa. Näitä ovat: ensimmäisin, lähin, vahvin ja heikoin.

Peli jatkuu, kunnes 10 vihollista on päässyt maaliin, jolloin peli pyytää käyttäjältä nimimerkkiä ennätystuloksia varten. Nimimerkin tulee olla alle 15 merkkiä pitkä, eikä siinä tule olla tyhjän tilan merkkejä, kuten välilyöntejä tai sarkaimia, sillä nämä poistetaan automaattisesti.

3 Ohjelman rakenne

3.1 Class Main

Kutsuu Setup ja LoadHighScores luokkia, joiden avulla ladataan pelin ja ennätystulosten tiedot. Luokka myös aloittaa QApplicationin ja lopulta kutsuu GUI-luokkaa, joka käynnistää itse ojelman.

3.2 Class Setup

Luokka lataa pelin tiedot annetusta tiedostosta. Tätä kirjoittaessa tiedosto on "Game_data.txt". Tiedot erotellaan välimerkeillä, ja luokka ei välitä välimerkkien määrästä, tai laadusta; välilyöntejä ja sarkaimia hyväksytään rajoittamaton määrä, kunhan, että tietojen välissä on vähintään yksi tällainen merkki. Luokka osaa myös hypätä tyhjien rivien ja kommenttien yli.

Luokka sisältää myös poikkeuksen class CorruptedGameData(Exception), jota luokka hyödyntää, mikäli annetussa tiedostossa on virheitä ja luokka ei kykene lukemaan sitä. Toiseksi luokka käyttää loader_function.py tiedostossa määritettyä EOFReached-poikkeusta, joka nostetaan, kun tiedosto on luettu loppuun.

3.2.1 load_waves(file, game)

Lataa tiedot vihollisaalloista ja asettaa ne parametriin "game".

3.2.2 load_missile(file, game)

Lataa tiedot ammuksista ja asettaa ne parametriin "game".

3.2.3 load_tower(file, game)

Lataa tiedot torneista ja asettaa ne parametriin "game".

3.2.4 load_enemy(file, game)

Lataa tiedot vihollisista ja asettaa ne parametriin "game".

3.2.5 load_route(file, game)

Lataa reittipisteet ja asettaa ne parametriin "game".

3.2.6 load_board(file)

Lataa luokan Board tarvitsemat tiedot, eli leveyden, korkeuden ja ruudun leveyden ja luo luokan Board niiden mukaisesti. Lopuksi metodi palauttaa luodun Board-luokan. (Board)

3.2.7 load_game(file, game)

Lataa peliluokan Game tarvitsemat tiedot, kuten elämät, rahamäärän ja ruudunpäivitysnopeuden ja määrittää ne parametriin "game". Palauttaa muokatun "game"parametrin, eli Game luokan. (Game)

3.2.8 load_all(input)

Kutsuu kaikkia yllä mainittuja luokan Setup metodeja ja määrittää niiden avulla pelin alkuarvot tiedoston "Game_data.txt", eli parametrin input mukaisesti. Alustaa luokan Game ja sisällyttää siihen kaikki ladatut tiedot. Lopuksi palauttaa luodon Game luokan. (Game)

3.3 Class LoadHighScores

Luokkaa käytetään ennätystulosten lataamiseen annetusta tiedostosta. Tätä tekstiä kirjoittaessa tiedosto on "High_scores.txt". Luokan parametriksi annetun tekstin tulee täyttää tietyt muotoilukriteerit, jotka on kerrottu tämän dokumentin luvussa 6.1.

Luokan Setup mukaisesti, tämä luokka hyödyntää omaa CorruptedHighScoresData-luokkaa, jonka avulla ilmaistaan, milloin tiedosto on luettu loppuun.

3.3.1 load_scores(file, error$_text$)

Palauttaa 10 ennätystulosta tiedoston siitä kohdasta, jolta ennätystulokset halutaan lukea. Metodia saa siis parametrikseen tiedoston "file", joka on luettu siihen kohtaan, josta nämä 10 riviä halutaan lukea. Metodi palauttaa listan luetuista riveistä. (List)

3.3.2 load(input)

Palauttaa kaikki ladatut ennätystulokset sanakirjana, jonka avaimina on "Easy", "Medium"ja "Hard". Tiedosto, josta tiedot luetaan saadaan parametrina "input". (Dict)

3.4 Class Game

3.4.1 setup(game_information)

Asettaa pelin tiedot Game_data.txt -tiedoston mukaisesti. Asetettavat tiedot ovat: rahamäärä, elämät, aika tasojen välillä, aika vihollisten luomisen välillä ja ruudunpäivitysnopeus.

3.4.2 initialize

Nollaa pelin rahamäärän ja elämät niihin arvoihin, jotka on määritetty Game_data.text -tiedostossa.

3.4.3 get_money

Palauttaa pelin rahamäärän. (int)

3.4.4 get_points

Palauttaa pelin pisteet. (int)

3.4.5 get_lives

Palauttaa pelin elämäpisteet. (int)

3.4.6 get_time_between_waves

Palauttaa ajan tasojen välissä. (int)

3.4.7 get_enemy_spawn_interval

Palauttaa ajan vihollisten luomisen välillä yhdellä tasolla. (int)

3.4.8 get_fps

Palauttaa ruudunpäivitysnopeuden. (int)

3.4.9 get_board

Palauttaa pelikentän. (Board)

3.4.10 get_enemy_types

Palauttaa määritetyt vihollistyypit. (Dict)

3.4.11 get_tower_types

Palauttaa määritetyt tornityypit. (Dict)

3.4.12 get_missile_types

Palauttaa määritetyt ammustyypit. (Dict)

3.4.13 get_route_points

Palauttaa määritetyt reittipisteet. (List)

3.4.14 check_dead_enemies

Tarkistaa, onko pelikentällä kuolleita vihollisia, eli vihollisia, joiden elämäpisteet ovat kuluneet loppuun. Lisää pelin pisteisiin vihollisen arvon mukaisen määrän pisteitä ja poistaa vihollisen pelikentältä.

3.4.15 add_enemy_type(type_name, enemy)

Lisää vihollistyypin "enemy"pelin enemy_types -sanakirjaan ja nimeää sen "enemy_type":ksi.

3.4.16 add_tower_type(type_name, tower)

Lisää tornityypin "tower"pelin tower_types -sanakirjaan ja nimeää sen "type_name":ksi.

3.4.17 add_missile_type(type_name, missile)

Lisää ammustyypin "missile"pelin missile_types -sanakirjaan ja nimeää sen "missile_type":ksi.

3.4.18 add_route(route_name, route_points)

Lisää reitin "route_points"pelin routes-listaan ja nimeää sen "route_name":ksi.

3.4.19 lose_life

Vähentää pelistä yhden elämän.

3.4.20 increase_money(money)

Kasvattaa pelin rahamäärää parametrin "money"verran.

3.4.21 buy(cost)

Vähentää pelin rahamäärää parametrin "cost"verran.

3.4.22 increase_points(points)

Kasvattaa pelin pistemäärää parametrin "points"verran.

3.4.23 set_board(board)

Asettaa peliin pelikentän "board".

3.4.24 set_enemy_spawn_interval(interval)

Asettaa pelin vihollisten luomisväliksi ajan "interval".

3.4.25 next_wave

Palauttaa listan vihollisista, jotka tulee lisätä pelikenttään seuraavalla tasolla ja kasvattaa pelin tasoa yhdellä kutsumalla funktiota "advance_to_next_wave". (List)

3.5 Class GUI(QtWidgets.QMainWindow)

Huolehtii pelin ikkunoiden ja grafiikan piirtämisestä. Luokka myös kutsuu siihen tallennettuja aliluokkia, kuten Game, Enemy, Tower ja Missile ja pyörittää samalla näin koko pelin logiikkaa. GUI periytyy QT:n QMainWindow -luokasta.

3.5.1 get_game

Palauttaa GUI:n game-objektin. (Game)

3.5.2 add_all_graphics

Kutsuu erillistä add_board_graphics funktiota, joka piirtää kentän grafiikat aiemmin määritetyn reitin perusteella. Funktio myös asettaa TextGraphicsItemit pelinäyttöön kutsumalla add_text_graphics_items -metodia. Kolmanneksi metodi lisää pelinäyttöön tornien valintakuvakkeet kutsumalla add_buildable_tower_graphics_items -funktiota.

3.5.3 add_enemies

Asettaa seuraavat lisättävät viholliset kutsumalla next_wave -funktiota. Metodi myös puolittaa vihollisten lisäysajan jokaisen kymmenennen "vihollisaallon"kohdalla.

3.5.4 add_enemy_graphics_item

Lisää näyttöön vihollisen grafiikkaobjektin. Mikäli viholliselle on määritetty varjo, metodi lisää myös sen. Objekti ja mahdollinen varjo tallennetaan GUI:n enemy_graphics_items -listaan. Lopuksi metodi käynnistää ajastimen, jonka loppuessa seuraava vihollinen voidaan lisätä.

3.5.5 add_tower_base_graphics_item(tower)

Lisää näyttöön tornin "tower"pohjan grafiikkaobjektin ja lisää sen GUI:n tower_graphics_items -listaan.

3.5.6 add_explosion_graphics_item(location, explosion_type, time)

Lisää näyttöön räjähdyksen "explosion_type-tyyppisen grafiikkaobjektin sijaintiin "location". Objekti tallennetaan myös explosion_graphics_items -listaan.

3.5.7 add_text_graphics_items

Lisää näyttöön QLabel-objektin, joka tulostaa näyttöön tietoja, kun hiiri viedään pelinäytön objektien päälle. Esimerkiksi kun hiiri viedään rakennettavan tornin valikko-objektin päälle, näyttöön tulostuu: "Costs: X1, Range: X2, Damage: X3, Builds in: X4 s, Reloads in X5, s", jossa XN on jokaisen tornin ominainen numeroarvo.

Metodi pitää huolen myös TextGraphicsItem -objektien piirtämisestä, joilla esitetään esimerkiksi tietoa pelaajan pisteistä ja elämäpisteistä ja tallentaa ne play_text_items -listaan.

3.5.8 add_buildable_tower_graphics_items

Lisää pelinäyttöön tornien grafiikkaobjekteja, joita klikkaamalla pelaaja voi rakentaa pelilaudalle torneja. Nämä objektit lisätään buildable_graphics_items -listaan. Tornit on "hard koodattu"ohjelmaan, joten vaikka niiden tiedot ladataankin Game_data.txt -tiedostosta, ei ohjelma osaa lisätä uusia tornien valikko-objekteja vain lisäämällä niitä tekstitiedostoon.

3.5.9 add_building_slots(tower_type)

Metodia kutsutaan, kun pelaaja on klikannut tornin valikko-objektia. Tämän jälkeen metodi piirtää rakennuskuvakkeet niihin ruutuihin, joihin pelaajan on mahdollista rakentaa torneja. Kuvakkeet tallenetaan building_slots -listaan. Kuvakkeille annetaan arvoksi "tower_type", joka ilmaisee, minkä tyyppinen torni mahdollisesti klikattavaan ruutuun rakennetaan.

3.5.10 update_all

Päivittää kaikki grafiikkaobjektit kutsumalla kutakin objektin päivityksestä huolehtivaa funktiota.

3.5.11 update_enemies

Päivittää viholliset sekä niiden grafiikkaobjektit. Metodi huolehtii sekä vihollisten liikuttamisesta, että kuolleiden vihollisten tarkastamisesta ja poistamisesta. Mikäli vihollisia ei ole jäljellä, metodi kutsuu add_enemies -metodia lisäämään vihollisia. Metodi tarkastaa myös, että kaikki kyseisen aallon viholliset tulee lisätyksi peliin. Vihollisten lisäämisen ja liikuttamisen jälkeen vastaavat toimenpiteet suoritetaan niiden grafiikkaobjekteille.

3.5.12 update_towers(towers)

Päivittää tornit ja niiden grafiikkaobjektit. Metodi huolehtii tornien ampumisesta ja tarkistaa, onko torni rakennettu valmiiksi. Mikäli kyllä, tornin päälle rakennetaan tykki ja torni aloittaa ampumisen. Tornien osalta tarkistetaan, että niiden tykki on suunnattu ammuttavaa vihollista kohti, ja metodi huolehtiikin tornin grafiikkaobjektin suunnan päivityksen.

3.5.13 update_missiles(missiles)

Päivittää ammukset ja niiden grafiikkaobjektit. Metodi huolehtii ammusten liikuttamisesta ja tarkistaa, ovatko ne saavuttaneet maalin, tai onko niiden kohde jo päässyt itse maaliin. Molemmissa tapauksissa ammus ja sen grafiikkaobjekti poistetaan.

Mikäli ammus on suuntautuva, myös sen kohteen sijainti päivitetään ja sen suunta käännetään vihollista kohti.

3.5.14 update_explosions

Päivittä räjähdysten grafiikkaobjektit. Poistaa objektit, joiden "elinaika"on kulunut loppuun.

3.5.15 update_text_graphics_items

Päivittää pelinäytön tekstiobjektit.

3.5.16 is_game_over

Tarkistaa onko peli loppunut. Mikäli kyllä, metodi pysäyttää ruudun päivityksen, poistaa näytöstä kaikki grafiikkaobjektit ja nollaa kaikki objekteja sisältävät listat.

Tämän jälkeen metodi piirtää näyttöön ohjetekstin käyttäjänimen pyytämistä ennätystuloksia varten ja luo ikkunan, johon käyttäjä voi syöttää käyttäjänimensä.

3.5.17 check_points

Tarkistaa, riittikö pelaajan pistemäärä pääsemään ennätystuloksiin kutsumalla add_high_scores funktiota. Metodi myös huolehtii siitä, että tuloksiin tallennetaan täsmälleen 15 merkkiä pitkä käyttäjänimi, ja että nimimerkki itsessään ei

sisällä välimerkkejä. Mikäli nimi on liian lyhyt, sen perään lisätään välilyöntejä niin kauan, että 15 merkin raja täyttyy. Myös, mikäli nimi jätetään tyhjäksi, metodi muuttaa sitä lisäämällä sen paikalla numeron 1, että ennätystulokset pysyvät kunnossa.

Lopuksi funktio näyttää ennätystulokset kutsumalla show_high_scores -metodia.

3.5.18 main_menu

Piirtää näytölle alkuvalikon, jossa on ohjetekstin lisäksi kaksi kuvaketta. Kolmion muotoisella pelaaja siirtyy pelaamaan ja toisella ennätystuloksiin. Tekstiobjekti (TextGraphicsItem) ja grafiikkaobjektit (QPixmapItem) tallennetaan menu_graphics_items -listaan.

3.5.19 choose_difficulty

Piirtää ruudulle ohjetekstit (TextGraphicsItem) ja kolme kuvaketta (QPixmapItem), joita klikkaamalla pelaaja valitsee pelin vaikeusasteen eli samalla eri kentän.

Metodi myös poistaa aiemmat päävalikon kuvakkeet ruudusta ja vaihtaa menu_graphics_items -listaan tämän metodin tekstit ja kuvakkeet.

3.5.20 play

Aloittaa pelaamisen poistamalla ensin vaikeusastevalikon teksti- ja grafiikkaobjektit näytöltä ja menu_graphics_items -listasta. Tämän jälkeen metodi aloittaa QTimer-aikalaskurin, jolla pyöritetään pelin logiikan päivitystä.

3.5.21 show_high_scores

Poistaa näytöltä aiempien valikoiden teksti- ja grafiikkaobjektit ja piirtää näytölle ennätystulokset. Ennätystulokset tallennetaan menu_graphics_items -listaan.

3.6 Class Board

Luokka huolehtii pelin vihollisten, tornien ja ammusten tallentamisesta. Luokkaan tallennetaan myös pelilautan ruutujen tiedot, eli onko ruudussa torni, reitti, vai onko se tyhjä. Luokka tallennetaan Game-luokkaan.

3.6.1 initialize

Alustaa luokan tiedot alkuperäisiksi, kun peliä pelataan samalla käynnistyskerralla useampia kertoja.

3.6.2 get_width

Palauttaa kentän leveyden. (Int)

3.6.3 get_height

Palauttaa kentän korkeuden. (Int)

3.6.4 get_squares

Palauttaa kentän ruudut taulukkona. (List in List)

3.6.5 get_square_size

Palauttaa kentän ruudun koon. (Int)

3.6.6 get_enemies

Palauttaa viholliset listana. (List)

3.6.7 get_towers

Palauttaa tornit listana. (List)

3.6.8 get_missiles

Palauttaa ammukset listana. (List)

3.6.9 get_route_points

Palauttaa reittipisteet listana. (List)

3.6.10 get_waves

Palauttaa vihollisaallot listana. (List)

3.6.11 get_current_wave

Palauttaa tämänhetkisen aallon numeron. (Int)

3.6.12 get_enemy_start_location

Palauttaa vihollisten alkupisteen listana, jonka ensimmäisessä alkiossa on x-koordinaatti, ja toisessa y-koordinaatti. (List)

3.6.13 amount_of_enemies_on_next_wave

Palauttaa seuraavan aallon vihollisten määrän. (Int)

3.6.14 get_difficulty

Palauttaa valitun vaikeustason. (String)

3.6.15 set_route_points(route_points, difficulty)

Asettaa pelilaudalle reittipisteet listan "route_points"mukaisesti ja asettaa vaikeustasoksi "difficulty":n.

3.6.16 add_wave(enemy, amount)

Lisää kenttään aallon, jossa on "amount"määrä vihollisia "enemy".

3.6.17 add_tower(tower_information, position)

Lisää kenttään tornin tietojen "tower_information"mukaisesti sijaintiin "position". Metodi myös tarkistaa, onko valittu sijainti jo varattu tai että onko ruutu kentän ulkopuolella. Mikäli kyllä, metodi tulostaa virhetekstin.

3.6.18 add_missile(missile)

Lisää kenttään ammuksen "missile".

3.6.19 add_route

Lisää reitin aiemmin määritettyjen reittipisteiden mukaisesti. Metodi siis merkitsee reittipisteiden lisäksi reitiksi myös niiden vaaka- tai pystysuorasti välissä olevat ruudut. Lopuksi metodi määrittää ensimmäisen reittipisteen vihollisten aloituspisteeksi.

3.6.20 kill_enemy(enemy)

Poistaa vihollisen "enemy"kentästä.

3.6.21 add_enemy(enemy, position)

Lisää vihollisen "enemy"sijaintiin "position".

3.6.22 is_adding_enemies

Palauttaa totuusarvon, joka ilmaisee vihollisten lisäämistä. Metodi palauttaa arvon Tosi, mikäli vihollisia ollaan juuri lisäämässä.

3.6.23 set_adding_enemies

Muuttaa vihollisten lisäämisen totuusarvon todeksi.

3.6.24 set_enemies_added

Muuttaa vihollisten lisäämisen totuusarvon epätodeksi.

3.6.25 advance_to_next_wave

Kasvattaa luokan muuttujaa "current_wave"yhdellä. Muuttuja ilmaisee, millä tasolla pelaaja on kyseisellä hetkellä.

3.7 Class Enemy

Mallintaa pelikentällä liikkuvaa vihollista. Vihollisen tavoitteena on päästä maaliin, ennen kuin sen elämäpisteet menevät nollaan. Mikäli näin kuitenkin käy, vihollinen poistetaan pelikentältä. Maaliin päästessään pelaaja menettää yhden elämän.

3.7.1 get_type

Palauttaa vihollisen tyypin. (String)

3.7.2 get_hitpoints

Palauttaa vihollisen elämäpisteet. (Int)

3.7.3 get_armour

Palauttaa vihollisen suojuksen. (Int)

3.7.4 get_speed

Palauttaa vihollisen nopeuden. (Int)

3.7.5 get_worth

Palauttaa vihollisen arvon. (Int)

3.7.6 get_image

Palauttaa vihollisen kuvatiedoston nimen. (String)

3.7.7 get_shadow

Palauttaa vihollisen mahdollisen varjon nimen. (String)

3.7.8 get_position

Palauttaa vihollisen paikan pelikentän taulukossa. (List)

3.7.9 get_location

Palauttaa vihollisen sijainnin listana, jonka alkioina on x- ja y-koordinaatit. (List)

3.7.10 get_degrees

Palauttaa suunnan, johon vihollinen on kääntynyt. Suunta indikoi myös suuntaa, jossa vihollisen seuraava saavutettava reittipiste on. (Int)

3.7.11 set_location(location)

Asettaa vihollisen sijainnin listan "location"mukaisesti.

3.7.12 set_location(position)

Asettaa vihollisen sijainnin listan "position"mukaisesti siten, että "positionin"arvot kerrotaan kentän ruutujen koolla, sillä "position"ilmaisee sijainnin kentän ruudun paikkana, eli positiota ruutujen taulukossa.

3.7.13 move(speed = 0)

Liikuttaa vihollista parametrin "speed"mukaisesti. Mikäli metodille ei ole annettu parametria, vihollinen ei liiku, sillä silloin parametri "speed"asetetaan nollaksi. Metodi tarkistaa, onko vihollinen saavuttanut maalin liikkumisen jälkeen ja mikäli kyllä, se palauttaa totuusarvon Tosi, muutoin Epätosi. Samalla metodi vähentää pelaajalta yhden elämän. (Boolean)

Metodi pitää myös huolen, että vaikka vihollinen saavuttaakin vuorolla seuraavan reittipisteen, se liikkuu jäljellä olevan nopeuden verran seuraavaa reittipistettä kohti. Tässä tapauksessa myös vihollisen suunta käännetään kohti kyseistä reittipistettä. (Boolean)

3.7.14 set_next_destination

Asettaa viholliselle seuraavan reittipisteen. Mikäli nykyinen reittipiste on vihollisen määränpää, metodi palauttaa totuusarvon Tosi, muutoin Epätosi. (boolean)

3.7.15 is_dead

Palauttaa totuusarvon sen mukaan, onko vihollinen kuollut vai ei, eli onko sen elämäpisteet kuluneet nollaan. Mikäli vihollinen on kuollut, metodi palauttaa totuusarvon Tosi, muutoin Epätosi. (Boolean)

3.7.16 reduce_hitpoints(damage)

Vähentää vihollisen elämäpisteitä parametrin "damage"verran. Kuitenkin siten, että parametristä "damage"vähennetään vihollisen suojaus "armour".

3.8 Class Square

Mallintaa pelikentän ruutuja. Jokainen pelikentän ruutu sisältää yhden Square luokan, joka kuvaa, mitä ruutu sisältää. Mikäli ruutu on tyhjä, sen tyyppi on Square.EMPTY, mikäli reitillä, Square.ROUTE ja mikäli siinä on torni, Square.TOWER.

3.8.1 set$_{e}$*mpty*

Määrittää ruudun tyhjäksi.

3.8.2 set$_{r}$*oute*

Määrittää ruudun reittipisteeksi, mikäli siinä ei ole vielä tornia tai aiempaa reittipistettä. Palauttaa totuusarvon Tosi, mikäli asettaminen onnistuu ja totuusarvon Epätosi, mikäli asettaminen ei onnistu.

3.8.3 set$_{t}$*ower*

Määrittää ruudun sisältävän tornin, mikäli siinä ei ole vielä tornia tai aiempaa reittipistettä. Palauttaa totuusarvon Tosi, mikäli asettaminen onnistuu ja totuusarvon Epätosi, mikäli asettaminen ei onnistu.

3.8.4 contains

Palauttaa numeron 0, 1 tai 2 sen mukaan, mitä ruutu sisältää. Mikäli se on tyhjä, metodi palauttaa 0:n, mikäli siinä on reitti, palauttaa 1:n ja mikäli siinä on torni, palauttaa 2:n. (Int)

3.9 Class Tower

Mallintaa tornia, joita pelaaja asettaa pelikentälle, ja jotka ampuvat vihollisia.

3.9.1 get_type

Palauttaa tornin tyypin. (String)

3.9.2 get_damage

Palauttaa tornin tekemän vahingon. (Int)

3.9.3 get_shoot_range

Palauttaa tornin ampumisetäisyyden. (Int)

3.9.4 get_reload_time

Palauttaa tornin latausajan. (Int)

3.9.5 get_build_time

Palauttaa tornin rakennusajan. (Int)

3.9.6 get_cost

Palauttaa tornin hinnan. (Int)

3.9.7 get_image

Palauttaa tornin tykin kuvakkeen nimen. (String)

3.9.8 get_base_image

Paluttaa tornin pohjan kuvakkeen nimen. (String)

3.9.9 get_target_type

Palauttaa kohteen tyypin, joka enumeraation mukaisesti palautuu numerona. (Int)

3.9.10 get_location

Palauttaa tornin sijainnin listana, jonka alkioina on x- ja y-koordinaatit. (List)

3.9.11 is_building

Palauttaa totuusarvon sen mukaan, rakennetaankto tornia vielä, vaiko ei. (Boolean)

3.9.12 get_degrees

Palauttaa tornin kulman suhteessa määritettyyn viholliseen. (Int)

3.9.13 select_target

Etsii viholliset, jotka ovat tornin ampumissäteen sisällä ja valitsee sen jälkeen vihollisen muuttujan "target_type"mukaisesti. Palauttaa tämän vihollisen. (Enemy)

3.9.14 shoot_first_enemy(enemies)

Asettaa tornin ampumaan heikointa tornin ampumissäteen sisällä olevaa vihollista kohti. Palauttaa tämän vihollisen. (Enemy)

3.9.15 shoot_closest_enemy(enemies)

Asettaa tornin ampumaan lähintä tornin ampumissäteen sisällä olevaa vihollista kohti. Palauttaa tämän vihollisen. (Enemy

3.9.16 shoot_strongest_enemy(enemies)

Asettaa tornin ampumaan vahvinta tornin ampumissäteen sisällä olevaa vihollista kohti. Palauttaa tämän vihollisen. (Enemy)

3.9.17 shoot_weakest_enemy(enemies)

Asettaa tornin ampumaan heikointa tornin ampumissäteen sisällä olevaa vihollista kohti. Palauttaa tämän vihollisen. (Enemy)

3.9.18 shoot

Tarkistaa voiko torni ampua, eli luoda uuden Missile-objektin. Mikäli kyllä, "reloading"muuttujan totuusarvo muutetaan todeksi ja QTImer-laskuri latausajalle käynnistetään. Lopuksi tornin tykin suunta käännetään vihollista kohti.

3.9.19 stop_reloading

Asettaa muuttujan "reloadin"totuusarvon epätodeksi. Ilmaisee, että torni ei enää lataa ja on valmis ampumaan.

3.9.20 set_location(location)

Asettaa tornin sijainnin listan "location"mukaisesti.

3.9.21 build

Asettaa tornin rakennetaan "building"muuttujan totuusarvon epätodeksi. Ilmaisee, että tornin rakentaminen on valmis.

3.9.22 change_target_type

Muuttaa tornin kohteen valinnan perustetta. Perustetta merkitään seuraavasti:

- Tower.FIRST = Torni ampuu ensimmäiseksi luotua vihollista
- Tower.CLOSEST = Torni ampuu lähimpänä olevaa vihollista
- Tower.STRONGEST = Torni ampuu vahvinta vihollista
- Tower.WEAKEST = Torni ampuu heikointa vihollista.

Valinnan muuttaminen tapahtuu kiertäen siten, että metodi tarkistaa tornin nykyisen tyypin ja siirtyy seuraavaan näin: Tower.FIRST → Tower.CLOSEST → Tower.STRONGEST → Tower.WEAKEST → Tower.FIRST → jne.

3.10 Class Missile

Mallintaa pelikentällä liikkuvaa ammusta. Mikäli pelikentällä on torneja, jotka ovat valmiita ampumaan, ja niiden ampumisetäisyydellä on vihollisia, tornit luovat ammuksia. Ammukset poistetaan, kun niiden kohde, eli vihollinen joko poistetaan, tai ammus saavuttaa kohteensa. Tällöin ammus on osunut viholliseen, ja vihollisen elämäpisteet vähentyvät.

3.10.1 get_type

Palauttaa ammuksen tyypin. (String)

3.10.2 get_image

Palauttaa ammuksen kuvatiedoston nimen. (String)

3.10.3 get_damage

Palauttaa ammuksen tekemän vahingon. (Int)

3.10.4 get_location

Palauttaa ammuksen sijainnin listana, jonka alkioina on x- ja y-koordinaatit. (List)

3.10.5 get_degrees

Palauttaa ammuksen suunnan. (Int)

3.10.6 initialize(tower, target)

Alustaa ammuksen sen luoneen tornin "tower"mukaisesti ja asettaa ammuksen kohteeksi parametrin "target"mukaisen vihollisen. Metodi myös tarkistaa, onko tornin synnyttämä ammus staattinen vai ei, eli että muuttaako ammus määränpäätänsä ampumisen jälkeen. Mikäli kyllä, ammus ohjautuu sille määrättyä vihollista kohti, ja mikäli ei, ammus suuntautuu sitä pistettä kohti, jossa määrätty vihollinen oli ammuksen luomishetkellä.

3.10.7 move(enemies, GUI)

Liikuttaa ammusta. Mikäli ammus saavuttaa määränpäänsä, kohteen, eli vihollisen elämäpisteitä vähennetään. Lisäksi, mikäli ammus on staattinen, tai sen tyyppi on "DoubleRocketLauncher", myös ammuksen räjähdysalueen sisällä olevien vihollisten elämäpisteet vähenevät.

Metodi myös piirtää pelikentälle räjähdyksen grafiikkaobjektin ja lisää sen GUI:n explosion_graphics_items -listaan, mikäli ammukselle on määritetty räjähdyskuvake.

Metodi palauttaa totuusarvon Tosi, mikäli ammus on osunut määränpäähänsä, tai mikäli ammuksen kohde on poistettu. Muussa tapauksessa metodi palauttaa totuusarvon Epätosi. Tällöin, ja jos ammus ei ole staattinen, sen suuntaus määritetään uudelleen kohti vihollista.

3.11 Class BoardGraphicsItem(QtWidgets.QGraphicsPixmapItem)

Grafiikkaobjekti perii QT:n QGraphicsPixmapItemin ja sijoittaa QPixmapin parametrin "placement"mukaiseen ruutun. Luokka siis sijoittaa pelikentälle oikean ruudun kuvakkeen parametrin "type"mukaan. Luokka osaa myös valita oikean vaikeusasteen mukaisen kuvan parametrin "difficulty"avulla. Kuva sijoitetaan näytölle käyttämällä parametria "square_size".

3.12 Class EnemyGraphicsItem(QtWidgets.QGraphicsPixmapItem)

Vihollisen grafiikkaobjekti, joka periytyy QGraphicsPixmapItemistä.

3.12.1 get_enemy

Palauttaa objektin vihollisen. (Enemy)

3.12.2 get_shadow

Palauttaa objektin varjon. (QGraphicsPixmapItem)

3.12.3 has_shadow

Palauttaa totuusarvon Tosi, mikäli objektilla on varjo, muutoin Epätosi.

3.12.4 update_graphics

Päivittää objektin grafiikat kutsumalla updatePosition, updateRotation ja update_shadow metodeja.

3.12.5 updatePosition

Päivittää objektin sijainnin.

3.12.6 updateRotation

Päivittää objektin suunnan.

3.12.7 update_shadow

Päivittää objektin varjon.

3.13 Class TowerGraphicsItem(QtWidgets.QGraphicsPixmapItem)

Tornin grafiikkaobjekti. Periytyy QGraphicsPixmapItemistä ja vaihtaa kohteen valintaan käytettävää tornin ominaisuutta klikkaamalla.

3.13.1 get_tower

Palauttaa objektin tornin. (Tower)

3.13.2 update_graphics

Kutsuu updatePosition metodia ja mikäli torni on rakentunut, metodi kutsuu myös updateRotation metodia.

3.13.3 build

Tarkistaa, onko torni rakentunut, mikäli kyllä, asettaa oman is_built parametrin todeksi ja palauttaa totuusarvon Tosi. Muussa tapauksessa Epätosi. (Boolean)

3.13.4 updatePosition

Päivittää tornin sijainnin.

3.13.5 updateRotation

Päivittää tornin suunnan.

3.13.6 mousePressEvent(event)

Mikäli objektia klikataan, metodi muuttaa tornin kohteen valintaa seuraavaan vaihtoehtoon.

3.14 Class MissileGraphicsItem(QtWidgets.QGraphicsPixmapItem)

Ammuksen grafiikkaobjekti, joka periytyy QGraphicsPixmapItemistä.

3.14.1 get_missile(self)

Palauttaa objektin ammuksen. (Missile) return self.missile

3.14.2 update_graphics(self)

Päivittää objektin sijainnin ja suunnan kutsumalla updatePosition ja update-Rotation -metodeja.

3.14.3 updatePosition(self)

Päivittää objektin sijainnin.

3.14.4 updateRotation(self)

Päivittää objektin suunnan.

3.15 Class ExplosionGraphicsItem(QtWidgets.QGraphicsPixmapItem)

Räjähdyksen grafiikkaobjekti, joka periytyy QGraphicsPixmapItemistä. Piirretään, mikäli ammus osuu kohteeseensa ja sille on määritetty räjähdysgrafiikkaobjekti.

3.15.1 time_passed(self)

Palauttaa jäljellä olevan ajan, jonka räjähdys on kentällä ennen poistoa. (Int)

3.16 Class BuildTowerGraphicsItem(QtWidgets.QGraphicsPixmapItem)

Rakennettavan tornin valikko-objektin grafiikkaobjekti, joka periytyy QGraphicsPixmapItemistä. Piirretään pelinäytön alareunaan, jota klikkaamalla pelaaja voi aloittaa tornin rakentamisen pelikentälle, mikäli hänellä on tarpeeksi rahaa.

3.17 get_tower

Palauttaa objektin tornin. (Tower)

3.18 set_building(boolean)

Asettaa tornin building muuttujan totuusarvon muuttujan "boolean"mukaisesti. Tosi merkitsee, että tornia ollaan rakentamassa, eikä uusia torneja voida tällöin luoda. Epätosi taas päinvastaista.

3.19 mousePressEvent(event)

Objektia klikattaessa objekti luo uuden tornin ja asennuspaikan kuvakkeet kentälle kutsumalla GUI:n add_building_slots -metodia.

3.20 Class BuildingSlotGraphicsItem(QtWidgets.QGraphicsPixmapItem)

Tornin rakennuspaikan grafiikkaobjekti. Rakennusapaikan kuvakkeet luodaan kentälle, kun pelaajalla on tarpeeksi rahaa ja hän klikkaa BuildTowerGraphicsObjektia.

3.20.1 get_tower_type

Palauttaa objektin tornin tyypin. (String)

3.20.2 mousePressEvent(self, event)

Objektia klikattaessa objekti luo sen omaan sijaintiin tyypin "tower_type"mukaisen tornin.

3.21 Class PlayerName(QWidget)

Luokkaa luo uuden ikkunan perien QWidgetin. Ikkunaan lisätään QLineEdit, jolla voidaan lukea pelaajan haluama nimimerkki ennätystuloksia varten.

3.21.1 get_text

Palauttaa kenttään kirjoitetun tekstin. (String)

3.22 Class TextItem(QGraphicsTextItem)

Objekti perii QGraphicsTextItemin ja sen mukaisesti piirtää näyttöön parametrin "text"mukaisen tekstin, jonka kooksi asetetaan parametrin "size"mukainen koko. Luokkaan on valmiiksi määritetty fontin väriksi valkoinen ja fontiksi "Comic Sans", jotta pelin fonttimaailma saataisiin pidettyä mahdollisimman yhdennäköisenä. Lopuksi objekti sijoitetaan parametrina annettuihin "x":n ja "y":n mukaisiin koordinaatteihin.

3.22.1 set_text(text)

Asettaa objektin tekstiksi parametrin "text"mukaisen tekstin.

4 Algoritmit

Algoritmit on kerätty pääosin omiin funktioihinsa ja jaettu omiin tiedostoihinsa koodin selkeyttämiseksi. Tiedostoja on neljä; add_board_graphics.py, add_high_score.py, coordinates.py ja loader_function.py. Tiedostojen sisältämät algoritmit on esitetty alla.

4.1 add_board_graphics.py

Tiedosto sisältää kaksi funktiota; add_board_graphics ja contains funktiot. Näistä contains-funktio on vain add_board_graphics -funktion apufunktio. Add_board_graphics -funktio selvittää, mitkä ruuduista ovat reittipisteitä ja piirtää niihin reitin, sekä niiden ympärille reitin muodot.

Funktio käy läpi kentän taulukon jokaisen alkion ja tarvittaessa samalla, mitkä ympäröivistä ruuduista ovat reittipisteitä. Helpoin tapaus on tilanne, jossa piste itsessään on osa reittiä, tällöin kuvake on helppo valita. Samoin on myös tilanne, jossa pisteen ympärillä ei ole ollenkaan reittiä.

Vaativin esimerkki on se, jossa piste itsessään ei ole reitillä, mutta sen oikeassa yläkulmassa oleva ruutu on. Tällöin tulee ensin tarkistaa ylä, ala, vasen

ja oikea ruutu ja jos mikään niistä ei täytä ehtoa, joutuu jokaisen kulman tutkimaan erikseen.

Contains-funktiolla kutsutaan yksittäisen ruudun omaa contains metodia, jonka avulla selvitetään, onko ruutu tyhjä, vaiko ei.

4.2 add_high_score

Funktio selvittää, onko pelaajan saavuttama pistemäärä tietyllä tasolla suurempi kuin samalla tasolla 10 ensimmäisen pistemäärän suuruus. Mikäli näin on, funktio kirjoittaa tiedoston uusiksi niin, että se lisää oikealle riville uuden ennätystuloksen.

Tämä on toteutettu siten, että funktio tallentaa kohdan, josta se löytää pienemmän tuloksen, lukee tämän jälkeen muistiin kaikki seuraavat rivit, siirtyy aiempaan kohtaan ja kirjoittaa uuden ennätystuloksen, jonka perään kirjoitetaan loput luetut tiedot.

4.3 coordinates

Tiedosto sisältää kolme eri funktiota; distance, new_location ja direction.

4.3.1 Distance

Funktiolla voidaan määrittää kahden annetun pisteen etäisyys toisistaan. Tämä tehdään Pythagoraan lauseen mukaisesti siten, että lasketaan x- ja y-koordinaattien erotus, otetaan erotuksista toiset potenssit, summataan ne yhteen ja otetaan tästä neliöjuuri.

4.3.2 New_location

Funktiolla voidaan määrittää uusi sijainti, kun liikutaan parametrin "speed"pituinen matka annettuja koordinaatteja kohti annetuista alkukoordinaateista lähtien.

Tätä varten muodostetaan yksikkövektori, jota varten tarvittavat kaavat löytyvät lähdeluettelosta. Yksikkövektorin laskemisen jälkeen riittää, että sitä kerrotaan parametrilla "speed", josta saadaan suoraan uusi sijainti.

Yksikkövektorin laskenta tapahtuu periaatteessa siten, että lasketaan x- ja y-koordinaattien erotukset ja muodostetaan niistä vektori. Tämän jälkeen lasketaan vektorin pituus laskemalla Pythagoraan lauseella hypotenuusa Distancefuntion tapaan. Tämän jälkeen vektori jaetaan sen pituudella, josta saadaan tulokseksi haluttu yksikkövektori.

4.3.3 Direction

Funktiolla voidaan määrittä suunta, johon vihollisen, tornin tai ammuksen tulee kääntyä, että se osoittaa kohteensa suutaan. Funktio saa paremetreinään nykyisen sijainnin, kohteen sijainnin ja nykyisen suunnan.

Laskentaa varten tulee jälleen laskea erotukset x- ja y-suunnassa. Sitten suunta saadaan tämän kaavan avulla laskemalla: suunta = math.degrees(math.atan(movement_y_axis

/ movement_x_axis)), joka tarkoittaa siis sitä, että siirtymä y-suunnassa jaetaan siirtymällä x-suunnassa, minkä jälkeen jakolaskusta otetaan arctan ja lopuksi se muutetaan vielä radiaaneista asteisiin.

Funktiossa on myös huomioitu nollalla jakaminen sekä arctanin ominaisuus osata laskea kulma vain välille -90 - 90 astetta. Tämä tehdään tarkistamalla suunnan muutosten merkkejä.

5 Tietorakenteet

Pääasiassa tietoa on varastoitu listoihin. Listoihin on tallennettu esimerkiksi viholliset, tornit ja ammukset. Samoin näiden grafiikkaobjektit, sekä kaikki muut grafiikkaobjektit on tallennettu omiin listoihinsa. Listat valittiin niiden muutettavuuden takia. Pythonilla alkion poistaminen, etsiminen ja lisääminen onnistuu vaivatta ja mikä tärkeintä, listan läpikäyminen for loopilla on yksinkertaista. Tätä tarvittiin esimerkiksi eri objektien liikkumisen tarkistamiseen.

Tietoa on tallennettu myös sanakirjoihin ja tupleihin. Tupleihin on tallennettu reittipisteet, koska on haluttu varmistaa, että niitä ei voi vahingossakaan muuttaa ohjelman sisällä. Sanakirjoja puolestaan on käytetty esimerkiksi eri vihollis-, torni- ja ammustyyppien tallentamiseen. Tämän on toteuttettu siten, että sanakirjan avaimena on vihollisen, tornin tai ammuksen tyyppi ja avaimen parina lista, joka sisältää kaiken tarvittavan informaation. Sanakirja valittiin tähän tarkoitukseen siksi, että sen avulla pystyttiin välttämään tilanne, jolloin ohjelman tulisi tietää tiedon tarkka sijainti. Näin ohjelmaan jätettii optio lisätä esimerkiksi uusia vihollistyyppejä huomattavasti helpommin.

6 Tiedostot

Ohjelma käyttää tietojen tallentamiseen teksti- (.txt) ja kuvatiedostoja (.png).

6.1 Tekstitiedostot

Käytettyjä tekstitiedostoja on kaksi. "Game_data-tiedostoon on tallennettu pelin informaatio ja "High_scores-tiedostoon ennätystulokset. Suurin osa ohjelman muuttujista päätettiin tallentaa tekstitiedostoihin, sillä niiden muokkaaminen nopeutui näin merkittävästi. Esimerkiksi vihollisten ja tornien voimakkuuden tasapainottaminen on huomattavasti nopeampaa, kun se voidaan tehdä erillistä tekstitiedostoa muokkaamalla, eikä lähdekoodiin tarvitse koskea. Toisaalta ennätystulosten tallentamista varten täytyy ylipäätään olla olemassa erillinen tiedosto, sillä muuten tulokset nollaantuisivat jokaisella käynnistyskerralla, tai ohjelman tulisi kyetä muokkaamaan omaa lähdekoodiaan.

Tekstitiedostot valittiin ohjelman tietojen tallennusmuodoksi niiden yksinkertaisuuden vuoksi. Vaihtoehtona oli .csv tai .xml tiedostot, mutta niiden lukeminen ja muokkaaminen havaittiin vaativammaksi kuin tavallisten .txt tiedostojen hyödyntäminen. Lisäksi tekstitiedostot aukeavat kaikilla laitteilla ja

käyttöjärjestelmillä varmasti ja ilman ylimääräisiä muotoiluja (vrt. Excel, LibreOffice tai OpenOffice).

Tietojen tallentamisessa pyrittiin tekemään tiedostosta ihmisystävällinen. Tästä syystä muokattavaksi tarkoitettuja tietoja ei tallennettu "yhteen pötköön", vaan rivi kerrallaan, jolloin ihmisen on mahdollista lukea ja ymmärtää tiedoston tietoja vaivatta. Eri tietolohkot on eroteltu # -merkillä, jonka jälkeen lohkon tiedot on erotettu rivillä toisistaan välimerkeillä (whitespace).

Lisäksi haluttiin, että tiedostoja voidaan lukea, vaikka niiden muotoilu muuttuisikin hiukan. Tästä syystä tietojen lukeminen on toteutettu siten, että ohjelma osaa lukea tiedot välimerkkien määrästä riippumatta ja että tiedosto osaa sivuuttaa ylimääräiset välilyönnit. Oleellista on, että tiedot yhdellä rivillä on erotettu toisistaan vähintään yhdellä väli-, tai sarkainmerkillä. Nämä toteutettiin "Game_data.txt-tiedoston lukemista varten. Valitettavasti "High_scores.txt-tiedoston osalta näitä ei voitu toteuttaa, sillä muutoin ennätystulosten kirjoittamista ei olisi saatu toimimaan. Tämä johtuu siitä, että tekstitiedostoon kirjoittaessa tietoja kirjoitetaan merkki, eikä rivi kerrallaan ja tiedosto ei pysynyt halutussa muodossa, mikäli käyttäjänimen pituudeksi sallittiin eri pituisia nimiä. Tästä ei kuitenkaan haluttu luopua, joten tiedoston muotoiluksi valikoitui ihmiselle helposti luettava, mutta vaikeasti korjattava muoto.

Muotoilu on toteutettu siten, että #-merkin jälkeen ilmaistaan kentän taso, jonka perään on listattu ilman turhia välejä ennätystulokset parhaimmasta alkaen. Yhdellä rivillä on ensin nimi, sitten välilyöntejä niin kauan, että nimen ja välimerkkien yhteenlaskettu summa on 15 merkkiä. Tämän jälkeen on asetettu tason numero ja kaksi sarkainmerkkiä, jonka jälkeen tulee vapaapituinen pistemäärä.

6.2 Kuvatiedostot

Kuvatiedostoja käytetään vihollisten, tornien, ammusten, räjähdysten ja kentän piirtämiseen. Kuvat on tallennettu "images-kansioon, jossa on lisäksi kolme alikansiota kentän kuvatiedostoille.

Valikoiden ja kentän kuvatiedostot on koodattu suoraan lähdekoodiin, sillä niiden muuttamiselle ei havaittu ohjelman kehityksen aikana tarvetta. Mikäli ohjelman käyttäjä haluaa muuttaa kentän kuvakkeita, tulee hänen vain vaihtaa kentän piirtämiseen käytetyt 14 kuvaketta uusiin, vastaavasti nimettyihin kuvatiedostoihin. Vastaavasti voidaan menetellä valikon kuvakkeiden kanssa.

Muut kuvat on spesifioitu "Game_data-tiedostoon, kukin kuva esimerkiksi kyseisen vihollisen tai tornin tietolohkoon.

Kuvatiedostoksi valittiin .png, sillä se säilyttää kuvien läpinäkyvyyden. Valintaan vaikutti myös se, että käytetyt kuvatiedostot oli tallennettu .png muodon lisäksi vektorigrafiikkana (.svg), jonka käyttäminen muodostui turhan vaativaksi. Muutaman kuvan tapauksessa alkuperäiset kuvat eivät olleet alkuperäisinä tarpeeksi suuria, mutta muokkaamisen jälkeen nekin tallennettiin .png muotoon yhtenäisen tiedostomuotojen käytön edistämiseksi.

7 Testaus

Ohjelma täyttää kaikki teknisen suunnitelman liitteessä 2 esitetyt testit. Ainoa suunnitelmassa esitetty kohta, jota testeissä ei ole, on ääniefektin toistaminen, mutta tämä ei ole vika, sillä ääniä ei toteutettu peliin ollenkaan.

Testausta ei kuitenkaan suoritettu suunnitelmassa esitetyllä tavalla, vaan siitä poiketen huomattiin, että yksikkötestaus ei ole graafisen käyttöliittymän tapauksessa kaikista mielekkäin vaihtoehto. Tämä on perusteltua, koska käyttäjä hallitsee ohjelmaa kokonaan graafisen käyttöliittymän kautta. Periaatteessa ainoat virhetilanteet, joita käyttäjä voi aiheuttaa, ovat klikkaamalla aiheutetut virheet, sekä nimen syöttäminen ennätystuloksia varten. Kaikkea mahdollista kuitenkin klikattiin, eikä vikaa saatu syntymään. Myös ennätystuloksiin syötettiin mahdollisimman paljon virhedataa, eri pituisia ja eri merkkisiä merkkijonoja ja pitkän korjaamisen jälkeen ennätystulosten kirjoittamisen pitäisi toimia nyt ongelmitta.

Testaamatta jäi kuitenkin tilanne, jossa pelaaja pelaa pelin loppuun, katsoo ennätystulokset ja aloittaa uuden pelin. Samoin todella epäselvästi kirjoitetun "Game_data.txt":n testaaminen jäi puolitiehen, sillä ohjelmoitaessa testattiin, että ylimääräiset välit ja välimerkit eivät sotke ohjelmaa ja todettiin, että käyttäjän tulee osata kirjoittaa tiedot näiden vaatimusten mukaisesti, jolloin ongelmia ei pääse syntymään.

8 Ohjelman tunnetut puutteet ja viat

Pelin päättyessä joko vihollisaaltojen loppumiseen tai todennäköisemmin pelaajan elämäpisteiden loppumiseen, ohjelma pyytää pelaajalta nimimerkkiä ennätystuloksia varten. Tällöin ohjelma avaa uuden ikkunan, johon pelaajan tulee syöttää nimimerkkinsä. Yrityksistä huolimatta ikkunaa ei saada suljettua, eikä kenttää nimen syöttämistä varten saada luotua pääikkunaan, joten ikkuna jää auki ennätystulosten esittämisen jälkeenkin.

Toinen puute ohjelmassa on ennätystulosten luotettava kirjoittaminen. Ohjelma ei osaa pitää tuloksissa vain 10 ennätystulosta, vaan ennätystulosten tallennustiedosto kasvaa loputtomiin, mikäli peliä pelataa useita kertoja niin, että tulokset pääsevät listalle. Tämä voisi teoriassa aiheuttaa ongelmia, mikäli peli julkaistaisiin esimerkiksi verkkoon.

Lisäksi ei ole varmaa, toimiiko pelin aloittaminen uudelleen yhden päättyneen pelin jälkeen oikein. Pelin aloittaminen uudelleen ei kuulunut alkuperäiseen suunnitelmaan, mutta koska se saatiin implementoitua muuhun työmäärään nähden vaivatta, ominaisuus päätettiin jättää lopulliseen ohjelmaan.

On myös syytä mainita, että vaikka peliä varten luotiin tekstitiedosto, josta pelin tietoja voidaan muokata, kaikki siinä olevat tiedot eivät ole suoraan muokattavissa. Esimerkiksi ruudun leveys ja korkeus ovat arvoja, joita ei voi muuttaa ilman muuttamatta reittejä. Myöskään esimerkiksi uusia torneja ei voi lisätä ilman, että pelin koodiin kirjoitetaan paikat, joihin tornien luomiskuvakkeet voidaan laittaa.

9 Ohjelman heikkoudet ja vahvuudet

Projektin onnistunein osuus on ehdottomasti tavoitteeni huomioiden näyttävä grafiikka. Peliin saatiin sulavasti liikkuvia objekteja, eikä vain neliöitä tai palloja, jotka liikkuvat epämääräisesti. Lisäksi kentän automaattinen piirto koordinaattien mukaan toimii omien testieni mukaan ongelmitta. Pelikentästä voidaan näin ollen tehdä täysin erilainen vain muokkaamalla muutamaa numeroa.

Toinen merkittävä onnistuminen liittyy tiedostojen lukemiseen. Alunperin tiedot oli tarkoitus kirjoittaa suoraan koodiin, mutta lopulta suurin osa tiedoista laitettiin tekstitiedostoon, josta ohjelmaa on helppo muokata. Pelin vihollisaaltoja voidaan muokata täysin ja olemassa oleviem vihollisten, tornien ja ammusten tietoja voidaan muuttaa täysin, myös niiden kuvakkeita. Tämä oli merkittävä parannus suunnitelmaan nähden.

Ohjelman suurin heikkous on kattavan testauksen puuttuminen. Vaikka ohjelmaa on testattu pelaamalla sitä pitkiäkin aikoja, ei varmuutta kaikista erikoistilanteista ole. Tästä syystä esimerkiksi ennätystulosten kirjoittamisessa voi muodostua jokin ongelma, mikäli käyttäjä onnistuukin syöttämään jonkin merkkijonon, jota ohjelma ei osaakaan hallita. Myös mikäli pelaaja käynnistää pelin uudelleen ensimmäisen pelikerran jälkeen, ilman kattavaa testaamista on mahdollista, että esimerkiksi jokin grafiikkaobjekti jäisi poistamatta ja kummittelisi uudella pelikerralla.

Toinen heikkous liittyy pelin tasapainottamiseen. Pelin torneja ei ole tasapainotettu tarpeeksi, joten on hyvinkin varmaa, että jokin torneista on ylivoimainen toisiin torneihin nähden. Lisäksi myöskään vihollisaaltoja, eikä vihollisten arvoa ole kunnolla määritetty, joten on mahdollista, että jokin alkupäänkin aalloista on mahdoton torjua ja että joistakin vihollisista saa joko liikaa tai liian vähän rahaa, että pelaaja voisi rakentaa järkevän määrän torneja.

Kolmantena ohjelma ei tarjoa tarpeeksi palautetta käyttäjälle. Ohjelman kuvakkeet eivät välttämättä ole tarpeeksi selkeitä, että ne näyttäisivät nappuloilta, eikä pelaaja esimerkiksi saa mitään virheilmoitusta, mikäli hän yrittää rakentaa tornia, jota varten hänellä ei ole tarpeeksi rahaa. Pelaajalle ei myöskään esitetä graafisesti, mille alueelle torni ampuu, joten pelaajan on haastavaa asettaa torneja tehokkaasti.

10 Poikkeamat suunnitelmasta

Suunnitelmasta poiketen työhön ei toteutettu ollenkaan ääniä. Tämä olisi vaatinut kokonaan uuden elementin lisäämistä peliin, eikä sitä nähty välttämättömänä ominaisuutena, joten ohjelman kehityksessä keskityttiin muun toiminnallisuuden kehittämiseen.

Myös grafiikka on täysin erilaista kuin suunnitelmassa mainitut lähdeviittaukset. Ohjelmaa tehdessä havaittiin muiden grafiikkapakettien skaalautuvan paremmin tätä ohjelmaa varten, joten grafiikat vaihdettiin kokonaan uusiin.

Lisäksi testaus toteutettiin kokonaan eri tavalla. Vaikka kaikki suunnitelman mukaiset kohdat testattiin, ei yksikkötestausta toteutettu ollenkaan.

11 Toteutunut työjärjestys ja aikataulu

Projektiin käytetty työmäärä vastasi hyvin lähelle toteutunutta työmäärää. Suunnitelmassa työmääräksi arvioitiin yli 100 tuntia, joka tuli käytettyä. Näin ollen arvio oli oikeassa suuruusluokassa ja oikeastaan hyvin lähellä toteutunutta arviota.

Työjärjestys aikataulultaan sen sijaan ei toteutnut ollenkaan. Itsenäisenä työnä ohjelmointia toteutettiin luonnollisesti silloin, kun sille oli aikaa ja tästä syystä työtä tehtiin eri viikoilla, kuin suunnitelmaan oli merkitty.

Työjärjestys muutoin piti suurimmilta osin paikkaansa. Merkittävin ero oli siinä, että toisin kuin suunnitelmassa oli määritetty, työtä tehdessä ohjelmointi aloitettiin suoraan tallennustiedoston lukemisen kirjoittamisella. Näin ollen ensimmäisellä viikolla pelikenttään ei voinut vielä lisätä torneja, mutta tiedostojen lukeminen siltä osin, kun niitä oli datatiedostoon syötetty, toimi ongelmitta. Tästä eteenpäin ohjelmointi eteni suunnitellusti - ensin toteutettiin logiikka ja tämän jälkeen grafiika.

12 Arvio lopputuloksesta

Lopputulos täyttää suunnitelman vaatimukset ja graafinen käyttöliittymä näyttää paremmalta, kuin mitä projektia aloittaessa osasi vielä ennustaa. Myös toiminnallisuutta on riittävästi ja kaikki halutut toiminnot saatiinkin implementoitua peliin.

Ohjelmassa on kuitenkin vielä paljon puutteita ja kehitettävää, kuten osioissa 8 ja 9 on mainittu. Nämä kohdat tulisi korjata, ennen kuin ohjelman voisi julkaista yleiseen käyttöön, tai että se olisi edes mielekäs pelata.

Ongelmallista on myös tekstitiedoston käyttäminen ennätystulosten tallentamiseen. Vaikka sen lukeminen toimii erittäin hyvin, olisi kirjoittamisen takia pitänyt käyttää jotain taulukkomuotoon tallennettavaa tiedostoa, sillä silloin kirjoittamisen ongelmilta olisi vältytty kokonaan, ja toimintavarmuus olisi saatu toimimaan yhtä varmasti, kuin "Game_data.txt-tiedoston lukeminen.

13 Viitteet

- Qt 5.8 Documentation `http://doc.qt.io/qt-5/index.html`
- QGraphicsSimpleTextItem `https://doc.qt.io/qt-5/qgraphicssimpletextitem.html`
- QGraphicsPixmapItem `http://doc.qt.io/qt-5/qgraphicspixmapitem.html`
- QGraphicsTextGraphicsItem `http://http://doc.qt.io/qt-5/qgraphicstextitem.html`
- QLineEditItem `http://doc.qt.io/qt-5/qlineedit.html`

- QLabel http://doc.qt.io/qt-5/qlabel-members.html
- QWidget http://doc.qt.io/qt-5/search-results.html?q=qwidget
- QMainWindow http://doc.qt.io/qt-5/qmainwindow.html
- QScene http://doc.qt.io/qt-5/search-results.html?q=qscene
- Finding the Unit Vector of a Vector http://www.dummies.com/education/math/calculus/finding-the-unit-vector-of-a-vector/
- The Distance Formula http://www.mathwarehouse.com/algebra/distance_formula/index.php
- The Python Standard Library https://docs.python.org/3.3/library/index.html
- Moving from A(x,y) to B(x1,y1) http://gamedev.stackexchange.com/questions/23447/moving-from-ax-y-to-bx1-y1-with-constant-speed

14 Liitteet

14.1 Testilista

Alla listattu tärkeimpiä testattuja kohtia:

- Kaikkien vihollisten tulee syntyä automaattisesti
- Kaikkia torneja tulee voida luoda
- Jokaisen tornin pitää pystyä luomaan oma ammuksensa
- Tornin vihollistyyppiä pitää pystyä muuttamaan
- Torneja pitää pystyä asettamaan kentän jokaiseen ruutuun
- Torneja ei tule voida asettaa reitille
- Torneja ei tule voida asettaa toistensa päälle
- Ammuksen osuessa viholliseen
  - Ammuksen tulee kadota
  - Osumisefektin ääni tulee toistaa
  - Mahdollinen efektikuva tulee piirtää
  - Vihollisen elämäpisteiden tulee vähentyä
    - Mikäli vihollisen elämäpisteet loppuvat
      - Vihollinen tulee poistaa
      - Pisteiden tulee lisääntyä

        - Rahan tulee lisääntyä

- Pelaajan elämien loppuessa
    - Pelin tulee päättyä
    - Ennätystulos tulee tallentaa "Highscores.txt" tiedostoon
- Vihollisen karatessa
    - Pelaajan elämien tulee vähentyä
    - Vihollista seurannut ammus tulee poistaa
- Ennätystulosten tulee poistaa nimimerkistä ylimääräiset välit
- Ennätystulosten tulee lisätä nimimerkin perään tarpeeksi välilyöntejä
- Ennätystulosten tulee ylipäätään tallentua, mikäli pisteet ovat suuremmat, kuin edelliset pisteet samalla tasolla