Own subject5 - Hyper Fabio Battle

Tekijät:

- 80391C Juho Salmi juho.salmi@aalto.fi
- 78713T Matti Pekkanen matti.pekkanen@tkk.fi
- 53761M Riku Luostarinen riku.luostarinen@tkk.fi

Viimeisin päivitys:

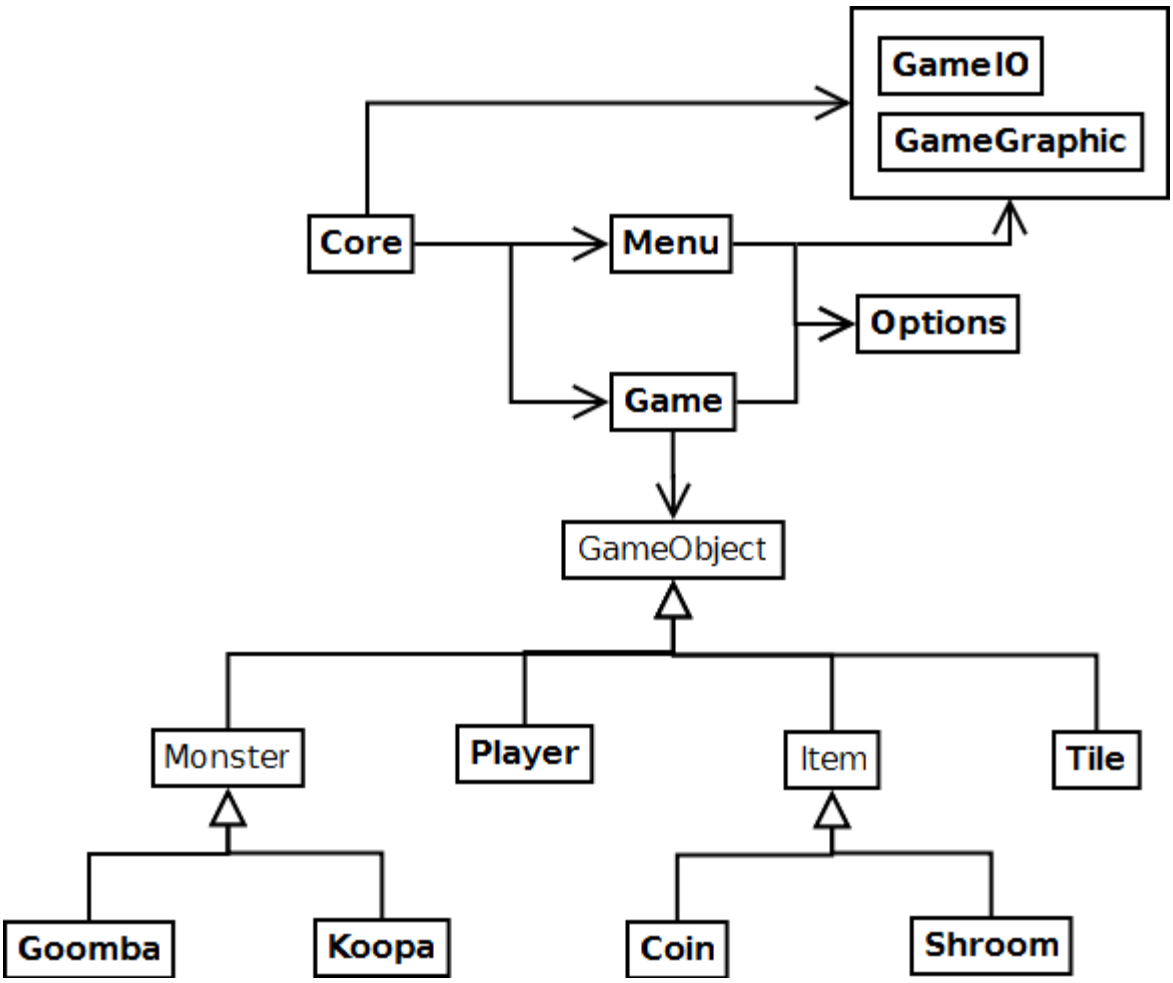
7.11.2010

Ohjelman kuvaus ja vaatimusten määrittely

Ohjelma on Super Mario Brothers 3 Battle mode klooni. Pelissä kaksi pelaajaa ohjaavat pelihahmoja, jotka keräävät kolikkoja. Kolioita ansaitaan tuhoamalla erilaisia hirviöitä. Peli jatkuu kunnes viisi kolikkoa on kerätty, tai kunnes toinen pelaajista kuolee. Pelin voittaja on se, joka kerää enemmän kolikkoja tai selviytyy hengissä.
Peliin toteutamme kaksi pelihahmoa, joita ohjataan asetusvalikosta säädettävillä näppäimillä. Toteutamme yhden kartan, jossa pelataan. Kartta koostuu liikkuvista paloista, joita vasten hyppäämällä ylläoleva hirviö taintuu. Peliin tulee vähintään kaksi hirviötyyppiä, Goomba ja Koopa Troopa. Grafiikat toteutetaan 2D:nä ja pelissä tulee olemaan erillinen taustamusiikki ja ääniefektejä.
Ohjelman luokkarakenne (kuva 1.) perustuu perintään ja on suunniteltu siten, että uusien hirviöiden, karttojen ja erilaisten bonus-esineiden lisääminen ajan salliessa on mahdollista. Esimerkiksi peliin voi lisätä pelaajan kasvattavia supersieniä, hetkellisen kuolemattomuuden antavia tähtiä tai vaikeampia vihollisia.

Ohjelman arkkitehtuuri

Ohjelman arkkitehtuuri on esitetty kuvassa 1.



Kuva 1. Ohjelman luokkarakenne. Lihavoimattomat luokat ovat abstrakteja. Kolmionuoli kuvaa perintää, ja avoin nuoli kuvaa luokan käyttävän toista luokkaa

Ohjelma käynnistyy Core-luokasta, joka alustaa Menu- ja Game-luokat sekä kaikki pelissä tarvittavat grafiikka- ja ääniobjektit GameGraphic- ja GameIO-luokkiin. Ohjelman aloitusvalikko on sijoitettu luokkaan Menu, josta voidaan aloittaa uusi peli, jatkaa olemassa olevaa peliä, lopettaa ohjelma tai säätää pelin asetuksia, kuten ääniä ja pelissä käytettäviä näppäimiä. Kaikki asetukset talletetaan Options-luokkaan.

Itse pelilogiikka on luokassa Game. Core-luokka antaa Gamelle alustuksessa referenssin pelissä käytettäviin Options-, GameGraphic- ja GameIO-objekteihin. Game-objektin konstruktorissa alustetaan kaikki kyseenomaisella pelikerralla käytettävät pelaajat, hirviöt, kolikot ja muut kenttään

sijoitetut esineet.

Ohjelma-arkkitetuuri hyödyntää tehokkaasti perintää: Kaikkilla kentässä esiintyvillä esineillä on abstrakti yläluokka GameObject. Kentässä esiintyvät hirviöt (luokka Monster), pelaajat (luokka Player), esineet (luokka Item) ja kentän palat (luokka Tile) perivät kaikki tämän luokan. Edellä mainituista luokista Monster sekä Item ovat edelleen abstrakteja, s.e. eri hirviötyypit (toteutuksessa ainakin Goomba ja Koopa) sekä eri esineet (mm. Coin) perivät nämä.

Suunnittelimme jossain vaiheessa kentässä esiintyvien GameObject-luokan perivien luokkien jakamista eläviin ja elottomiin sekä niitä vastaavien abstraktien yläluokkien toteuttamista GameObject-luokan alle. Emme kuitenkaan tässä vaiheessa nähdeet tätä tarpeelliseksi, mutta saatamme toteuttaa em. luokkahierarkiamuutoksen myöhemmin, mikäli se osoittautuu aiheelliseksi.

Alla vielä lyhyt kuvaus jokaisesta luokasta:

Core

Alustaa Menu- ja Game-luokat sekä kaikki pelissä tarvittavat grafiikka- ja ääniobjektit GameGraphic- ja GameIO-luokkiin. Sisältää ohjelman pääsilmukan ja käynnistää Menun sekä toteuttaa käyttäjän menussa valitsemat toiminnot.

Menu

Pelivalikko, jonka rakenne on pääpiirteissään seuraava:

New Game

Continue (näytettään jos valikkoon on siirrytty kesken pelin)

Options

-> alivalikko eri asetuksille, mm. näppäimet, äänet...

Exit

Päätasolla tehdyt valinnat palautetaan Core-luokalle, joka reagoi niihin valinnan mukaisella tavalla (aloittaa pelin, lopettaa ohjelman jne.)

Game

Itse pelilogiikka. Luokka omistaa erilaiset kenttään sijoitetut objektit, kuten pelaajat, hirviöt, kolikot yms. Vastaa myös näppäimistökomentojen lukemisesta.

GameIO

Luokka (struct), jossa säilytetään IO:hon liittyvät objektit, kuten GraphicContext sekä ikkuna. Luokasta luodaan jokaisella ohjelman suorituskerralla ainoastaan yksi instanssi (singleton). Alustuksen hoitaa Core-luokka.

GameGraphic

Luokka (struct), jossa säilytetään graafiikkaan ja ääneen liittyvät objektit. GameGraphic-instansseja luodaan ohjelman suorituksen aikana kaksi kappaletta: yksi Menu:lle ja yksi itse pelille (Game), joilla molemmilla on omat taustakuvat, musiikit, äänet jne. Sekä Menu että Game jakavat kuitenkin yhteisen GameIO-objektin, jonka kautta GameGraphic:n sisältämä grafiikka renderöidään ikkunaan.

Options

Luokka, jossa säilytetään peliasetukset. Options-luokalle alustetaan konstruktorissa oletusasetuset, joita käyttäjä voi sitten halutessaan muuttaa päävalikon Options-valinanna alta aukeavan alivalikon kautta.

GameObject

Abstrakti yläluokka kaikille pelikentässä esiintyville objekteille (pelaajat, hirviöt, esineet, kentän palat). Kaikille objekteille yhteiset ominaisuudet, kuten sijainti, nopeus, törmäyksiin liittyvä laskenta jne. on sijoitettu tähän luokkaan.

Monster

Kaikille hirviöille yhteinen abstrakti yläluokka. Perii GameObject-luokan.

Goomba

Goomba-hirviö, joka perii Monster-luokan. Sisältää kyseiselle hirviölle ominaiset piirteet, kuten liikeradan ja tiedon siitä, miten hirviö voidaan tappaa.

Koopa

Goomba-luokkaa vastaava luokka Koopa-hirviölle.

Player

Luokka pelaajalle (Fabio/Uolevi). Sisältää tiedon pelaajan nykyisestä tilasta, kuten monestako osumasta pelaaja kuolee ja tiedon siitä, montako kolikkoa pelaaja on kerännyt. Perii GameObject-

luokan.

Item
Abstrakti yläluokka kaikille kentässä oleville esineille, kuten kolikoille, sienille, tähdille jne. Perii GameObject-luokan.

Coin
Kolikko, joita keräämällä peli voidaan voittaa. Perii luokan Item.

Shroom
Sieni, jota syömällä Fabio/Uolevi kasvaa pojasta mieheksi (vrt. armeija). Mieheksi kasvanut Fabio/Uolevi muuttuu takaisin pikkupojaksi osuessaan hirviöön (ts. kestää kaksi osummaa hirviöön). Pieni pelaaja taas kuollee ensimmäisestä osumasta hirviöön. Shroom-luokka perii Item-luokan.

Tile
Kentän pala. Perii GameObject-luokan.

Työnjako

Katsomme, että projektin voi jakaa loogisesti kolmeen osaan:
1. GameObject, ja sen aliluokat
2. Core sekä ohjelman tukiluokat GameGraphic, GameIO ja Options sekä Menu
3. Pelilogiikan toteutus Game-luokkaan

Aiomme aloittaa ohjelman toteuttamisen määrittelemällä ohjelman luokkien väliset rajapinnat yhdessä. Samalla suunnittelemme koko ohjelman yhdistävän Core-luokan. Tämän jälkeen jokainen voi itsenäisesti alkaa toteuttamaan yhtä kolmesta edellämainitusta osasta. Projektin edetessä voimme arvioida onko työnjako ollut tasapuolinen, ja tarvittaessa uudelleenjakaa tehtäviä työmäärän tasaamiseksi. Suunniteltu työnjako on, että Matti Pekkanen toteuttaa osan 1, Juho Salmi osan 2 ja Riku Luostarinen osan 3.

Ryhmän tarkoitus on työskennellä mahdollisimman paljon yhdessä samassa paikassa, toteuttamassa omia osiaan. Näin projektin edistymisen seuraaminen helpottuu ja ryhmän on mahdollista ratkaista mahdollisia ongelmatilanteita yhdessä. Myös työmäärän seuranta on helpompaa. Kaikki koodi

dokumentoidaan Doxygen-syntaksin mukaisesti. Tämä helpottaa loppudokumentaation tekemistä, sekä toisten koodin ymmärrettävyyttä. Ryhmä käyttää kommunikoinnissa Facebookia, IRCiä sekä Instant Messaging -sovelluksia. Koodi säilytetään ja jaetaan SVN:n kautta kurssilla opetetulla tavalla.

Testaus

Core-osuuden tekijä pystyy testaamaan sovelluksen toimintaa, koska se ei tarvitse muiden osien toiminnallisuuksia toimiakseen. Game- ja GameObject-luokkien testaamiseksi GameObject-luokkaan ja sen aliluokkiin toteutetaan aluksi funktiorunkoja, jotta luokkien välistä rajapintaa voidaan testata. Myöhemmässä vaiheessa Game- ja GameObject-luokkia pyritään toteuttamaan rinnakkain, koska ne käyttävät toistensa toiminnallisuuksia. Ts. Kun jokin GameObject-luokasta tai sen aliluokista valmistuu, siihen liittyviä toiminnallisuuksia voidaan testata Game-luokan puolelta. Esim. kun Player-luokka on toteutettu, voidaan Game-luokkaan toteuttaa pelaajien liikkeisiin liittyvä logiikka. Projektin loppupuolella voimme testata ohjelmaa kokonaisuutena. Testauksessa käytetään normaalin toiminnallisuustestauksen lisäksi Valgrindia ja tarvittaessa erillistä debugger-ohjelmaa.

Ajankäyttöarvio

Viikko 44: Suunnitelman laatiminen

Viikko 45: Core-luokan suunnittelu, sisäisten rajapintojen määrittely

Viikko 46: Aloitetaan loppudokumentointi koodauksen rinnalla.

Toteutetaan alkuvalikko, staattiset oletusasetukset, pelin käynnistäminen. Uuden pelin ja siihen liittyvien objektien alustaminen, syötteen lukeminen, grafiikan piirtäminen. GameObject-luokan ja sen aliluokkien funktioiden runkojen määrittely ja toteutus.

Viikko 47: Luokkarunkojen täyttäminen ja funktioiden oikeat toteutukset valmiiksi kaikkiin osiin.

Viikko 48: Ohjelman osien yhdistäminen, ensimmäiset pelitestit, loppudokumentin tekeminen.

Viikko 49: Pelin ja dokumentin viimeistely, Demo.

Ulkoiset kirjastot

Ohjelman toteutuksessa vakiokirjastojen lisäksi ClanLib 2.2 -pelikehityskirjastoa.