VAASAN AMMATTIKORKEAKOULU
VASA YRKESHÖGSKOLA
UNIVERSITY OF APPLIED SCIENCES

Kari Filpus

FLASH-POHJAINEN LÄÄKELASKU-HARJOITUSOHJELMA

Tekniikka ja liikenne
2012

VAASAN AMMATTIKORKEAKOULU
Tietotekniikan koulutusohjelma

TIIVISTELMÄ

Tekijä	Kari Filpus
Opinnäytetyön nimi	Flash-pohjainen lääkelaskuharjoitusohjelma
Vuosi	2012
Kieli	suomi
Sivumäärä	33
Ohjaaja	Pirjo Prosi

Noin neljä vuotta sitten aloitin yhteistyön vaasalaisen matematiikanlehtorin Sinikka Vuorenmaan kanssa. Hän on kehittänyt metodin, jolla sellaisetkin oppilaat, joilla on ollut vaikeuksia oppia mm. prosenttilaskuja, ovat hänen mukaansa metodin avulla oppineet ratkaisemaan niitä sujuvasti.

Sinikka Vuorenmaa on käyttänyt tätä metodia myös opettaessaan sairaanhoitajia Vaasan keskussairaalassa erilaisten lääkelaskujen suorittamisessa. Tästä syntyi ajatus Flash-pohjaisesta opetus-/harjoitusmateriaalista, jota sairaanhoitajat voisivat käyttää valmistautuessaan neljän vuoden välein pidettäviin kokeisiin.

Ohjelma pyrittiin tekemään mahdollisimman ergonomiseksi, jotta sen käyttäminen koettaisiin mielekkääksi. Tästä huolimatta jouduimme tekemään ohjelmaan aika-ajoin muutoksia kentältä tulleen palautteen vuoksi. Myös virheiden korjaaminen vei aikaa. Ohjelma onkin nähty tarpeelliseksi, koska suuri osa Suomen sairaanhoitopiireistä on ottanut sen käyttöön.

Avainsanat oppiminen, ratkaiseminen, menetelmä

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Tietotekniikan koulutusohjelma

ABSTRACT

Author	Kari Filpus
Title	Flash based application for practising medical sums
Year	2012
Language	Finnish
Pages	33
Name of Supervisor	Pirjo Prosi

About four years ago I started to co-operate with lector Sinikka Vuorenmaa. She lives in Vaasa and teaches mathematics in Vaasa vocational institute. She has invented a method that according to her, also those students who have had problems at how to learn mathematics, have also managed to learn, for example percent sums, quite easily.

Sinikka Vuorenmaa has used this method for teaching nurses in Vaasa Central Hospital to solve various types of medical calculations. This resulted in the idea of making a Flash based learning and practice application. With this application the nurses can practise their nursing skills so that they can participate in the medical examination which is provided every fourth year.

We tried to make this application as easy to use and ergonomic as possible. Still we had to make changes based on the feedback by the users and that took some time. This application has, indeed, proved to be considered useful because a great amount of healt care districts in Finland have taken this application into use so that their nurses can practise medical sum calculation skills.

Keywords Learning, solving, method

TERMISTÖ

Flash	Alkujaan Macromedian omistama ja nykyään Adoben kehittämä multimedia-alusta interaktiivisten animaatioiden, videoiden ja web-sivujen esittämiseen.
Fla	Flashin projektitiedostomuoto.
Swf	Flashin esitystiedostomuoto.
Aikajana	Timeline, jonka frameihin voidaan sijoittaa grafiikkaa ym. animaatioiden aikaansaamiseksi.
_root	Flash-kehittimen pääaikajanan osoite.
Stage	Flash-kehittimen rootissa sijaitseva näyttämö.
Layer	Flash-kehittimen animaatiokerros.
Frame	Yksittäinen kehys aikajanalla, joka voi sisältää grafiikkaa, ääntä tai ActionScriptiä.
Lukupää	Aikajanan frameja lukeva liikkuva osoitin.
Fps	Frames per seconds. Nopeus, jolla aikajanan framet luetaan.
Preloader	Flash-esityksen alussa tarvittaessa näytettävä animaatio, joka kertoo käyttäjälle esityksen latautumistilanteen.
MovieClip	Flashin luokka, josta voidaan luoda tarvittavat instanssit animaatioita ym. varten.
Button	Flashin luokka, josta voidaan luoda tarvittavat napit esitykseen.
Library	Flashin kirjasto, jossa on kaikki esityksessä käytetty graafinen materiaali. Voi sisältää myös äänimateriaalia. Visuaalisesti mukana ohjelmakehittimessä.

Sound	Flashin sound-luokka, jota ohjataan ActionScriptin avulla.
ActionScript	Flashin hallintaan käytettävä ohjelmointikieli.

SISÄLLYS

1 JOHDANTO

Noin neljä vuotta sitten keskustelin Vaasan ammattiopistossa matematiikkaa opettavan lehtori Sinikka Vuorenmaan kanssa hänen kehittämästään opetusmetodista. Hän kertoi pitävänsä sairaanhoitajille opetustunteja Vaasan keskussairaalassa yhdessä Vaasan ammattikorkeakoulun lehtorin Mikko Peltolan kanssa. Opetuksen tarkoituksena oli valmistaa sairaanhoitajia tuleviin lääkelaskennan kokeisiin. Sairaanhoitajat joutuvat aina neljän vuoden välein suorittamaan lääkelaskennan kokeen, jossa heidän täytyy osoittaa virheetöntä osaamistaan erilaisissa lääkelaskentatilanteissa. Tällaisia ovat mm. erityyppiset annos-, muunto- ja prosenttilaskut.

Keskustelussa kävi ilmi, että he käyttivät opetuksessaan tukimateriaalina PowerPoint-esitystä. Ehdotin heille, että toteuttaisin internetin kautta käytettävän opetusmateriaalin. Tässä päädyin Flash-ohjelmalla tehtävään esitykseen. Tämä perustui siihen, että materiaalissa käytetään paljon animaatioita sekä ääntä. Flash mahdollistaa lisäksi tarvittavan interaktiivisuuden, joten opiskelijoilla on mahdollisuus vaikuttaa esityksen käyttämiseen. Tämän vuoksi Flash oli kaiken kaikkiaaan luonnollinen valinta opetusalustaksi.

Aluksi tarkoitus oli tehdä sovellus vain Vaasan sairaanhoitopiirin käyttöön. Myöhemmin sovelluksen käyttäjien määrä on kuitenkin kasvanut huomattavasti, koska useat sairaanhoitopiirit Suomessa halusivat myös ottaa sen omaan käyttöönsä.

Tulevaisuudessa Flash-projektien määrä tullee vähenemään ja siirtymään HTML5-tekniikalla toteutettaviksi. Varsinkin Apple on ajanut maailmalla kovasti Flash-tekniikan korvaamista HTML5-tekniikalla. Esim. iPad-tabletti sekä iPhone eivät tue Flashiä. Lisäksi Flash-ohjelman omistaja Adobe on ilmoittanut lopettavansa Flash-tuen kehittämisen mobiilialustoille. Myös Mac-tietokoneiden uudesta Lion-käyttöjärjestelmästä on poistettu tuki, jolla Flash-esityksiä on voitu ajaa näytönohjaimen laskentatehoa hyväksi käyttäen. Flash-tuki on kuitenkin edelleen kaikissa tietokoneiden selaimissa.

2 FLASH-TEKNIIKKA

2.1 Historia

Flash on alun perin Macromedian 1996 kehittämä tekniikka vektorimuotoisten animaatioiden esittämiseen internetissä. Aluksi tarjolla oli vain yksinkertainen frame-pohjainen aikajana animaatioiden esittämiseen. Myöhemmin vuonna 2000 Flash 5:een lisättiin ActionScript 1.0. Tämä mahdollisti animaatioelementtien (MovieClip) hallitsemisen koodaamalla. Tässä vaiheessa voitiin jo saada aikaiseksi pelityyppisiä sovelluksia. (Lauri Paakinaho: Flash-peliohjelmointi. Opinnäytetyö 2011 pdf.)

Vuonna 2003 Macromedia toi markkinoille Flash-version 7. Tässä versiossa oli mukana ActionScript 2.0. Ohjelmalla on mahdollista kirjoittaa omia as-tiedostoja. Näihin as-tiedostoihin voidaan kirjoittaa luokkia ja esitellä muuttujat ja luokan metodit. Näistä luokista voi sitten luoda esitykseen omat instanssit, joten olio-ohjelmointi on ollut mahdollista jo versiosta 2.0 lähtien.

ActionScript 3.0 tuli puolestaan markkinoille vuonna 2007 ja on puhtaasti olio-ohjelmointiin perustuva. Version toi markkinoille Adobe ostettuaan Macromedian vuonna 2005. Tämän jälkeen ohjelman ulkoasua onkin muokattu vastaamaan Adoben CS-tuoteperheen muita jäseniä. Nykyinen versio on CS6.

2.2 Ohjelmakehitin

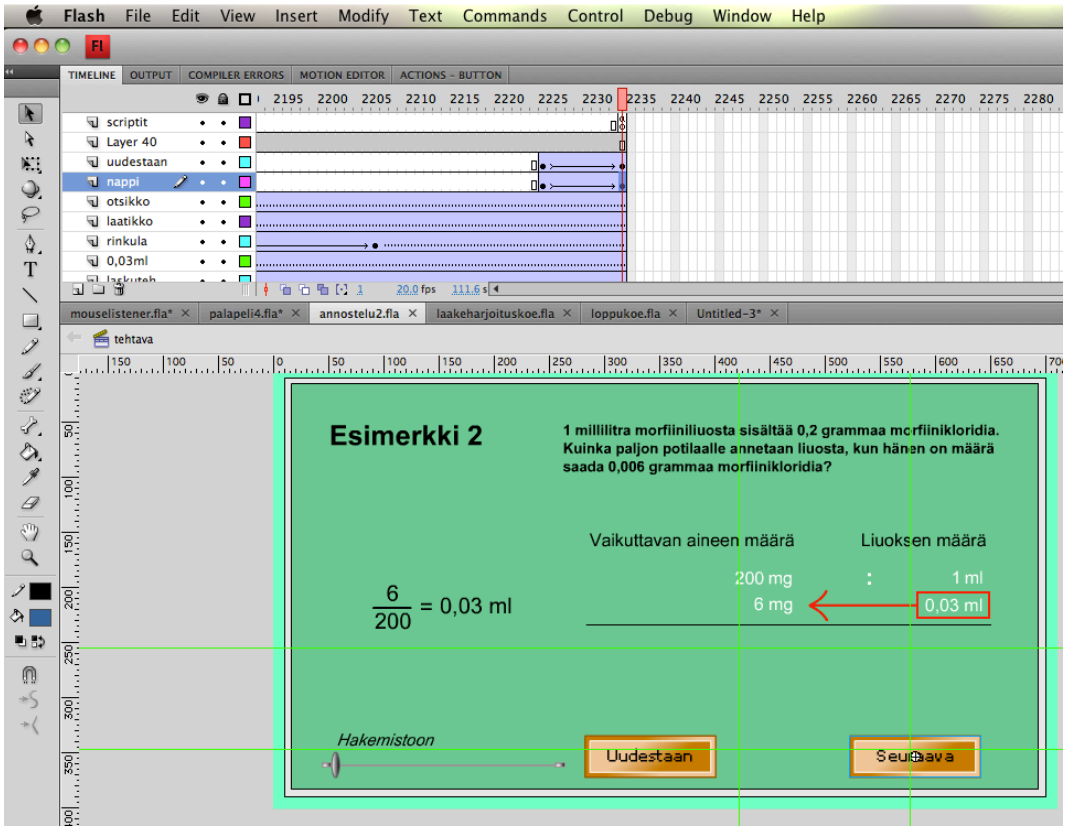
2.2.1 Flash-grafiikka ja ääni

Flash-grafiikka perustuu vektorigrafiikkaan, joten sitä voidaan skaalata ilman laadun heikkenemistä. Ohjelma kykenee näyttämään myös bittikarttakuvia, kuten jpeg-kuvia. Tällöin kuvaa ei kuitenkaan voida skaalata sen laadun heikentymättä. Kuvan suurentamisessa laatu heikkenee eniten, koska esitettävät pikselit joudutaan suurentamaan, joten kuvan tarkkuus heikkenee. Toisaalta kuvan pienentämi-

sessä Flash joutuu puolestaan poistamaan pikseleitä ja laskemaan niiden tilalle uusia pikseleitä ympärillä olevien pikseleiden RGB-arvojen mukaan. Lisäksi ohjelma joutuu tekemään antialiasoinnin pikseleiden väliin, ettei kuva näyttäisi revityltä pienennyksen jälkeen. Kuvan pienentäminen onnistuu täydellisesti ainoastaan silloin, kun kuvan vaaka- ja pystysuunnasta poistetaan joka toinen pikseli. Tällöin kuitenkin kuvan pinta-ala putoaa neljäsosaan alkuperäisestä. Tästä voikin ymmärtää sen, miksi antialiasointia joudutaan käyttämään bittikarttakuvien pienentämisessä.

Flash-ohjelmassa on asetukset sille, miten sinne tuotuja bittikarttakuvia voidaan käyttää. Ohjelmassa on oletuksena se, että kaikki kuvat näytetään laadultaan 75 %:n tasoisina jpeg-kuvina. Tätä asetusta voidaan säätää halutuksi. Toinen mahdollisuus on näyttää kuvat sellaisenaan. Jos halutaan käyttää oletusarvoista määritystä, tulee silloin huomioida se, että myös ohjelmaan tuodut jpeg-kuvat kompressoituvat uudelleen, joten kuvan laatu heikkenee silloin entisestään. Tällöin kannattaakin käyttää alun perin hyvälaatuista kuvaa, kuten esim. 24 bittistä PNG-kuvaa. Jälkimmäisessä vaihtoehdossa tulee puolestaan kuvankäsittelyssä huolehtia siitä, että kuvat ovat optimoituja käyttötarkoitukseensa.

Animaatio rakennetaan rootin aikajanalle, MovieClippien aikajanoille tai molempiin. Aikajanalle on mahdollista luoda kerroksia eli layereitä. Näihin kerroksiin voidaan sitten sijoittaa grafiikkaa, ääntä tai ActionScriptiä. Layerit mahdollistavat lisäksi eri grafiikkaelementtien animoimisen erikseen. Samalla voidaan näin helposti huolehtia siitä, mitkä grafiikkaelementit ovat toistensa päällä, kuva 1.



Kuva 1. Animaation layer-rakenne

Äänen käyttämistä Flashissä voidaan toteuttaa kahdella eri tavalla. Ensimmäinen tapa on implementoida ääni esitykseen itseensä, jolloin esityksen koko luonnollisesti kasvaa äänen koon osalta. Toinen mahdollisuus on lukea se jostain ulkopuolisesta tiedosta tai internetistä. Tällöin menetelmän etuna on se, että esitystiedoston koko ei kasva äänen osalta, joten esitys voi alkaa toimia nopeammin.

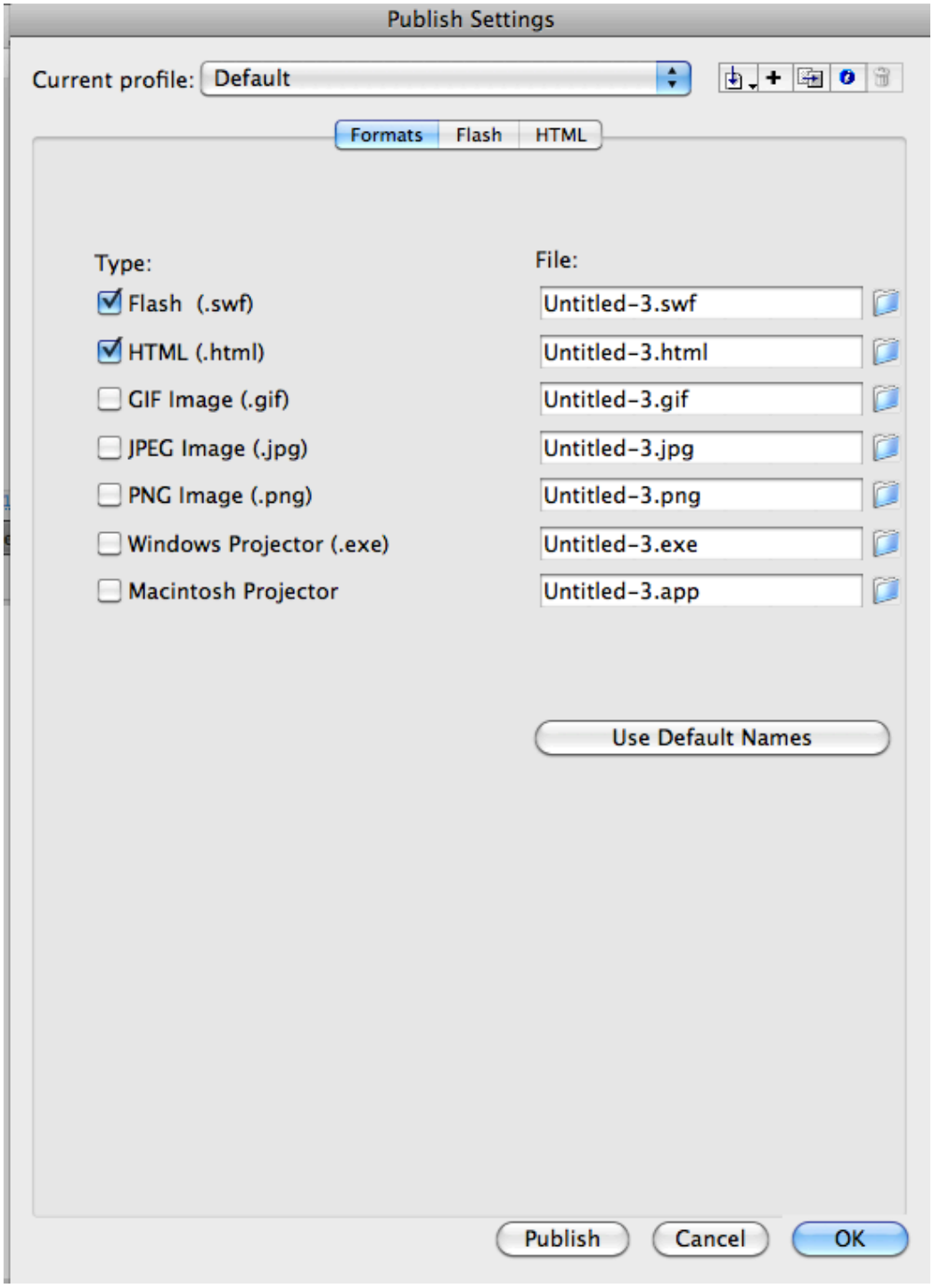
Mikäli ääni on esityksen sisällä, sitä voidaan käyttää siellä kahdella eri tavalla. Ääni voidaan laittaa aikajanalle ja soittaa sieltä lukupään avulla. Äänen ominaisuudeksi on asetettava joko event- tai stream-muoto. Event-muotoisessa tilassa ääni soitetaan mikäli esityksen lukupää vain sattuu siihen koskemaan. Vaikka lukupää pysähtyisi, soi ääni kuitenkin loppuun saakka. Tällainen äänen asetustila on hyödyllinen esim. lyhyiden ääniefektien toistamisessa.

Stream-muotoisessa äänessä lukupää toistaa äänileikettä liikkuessaan sen päällä. Tällöin saavutetaan mahdollisuus tahdistaa ääni ja grafiikka toisiinsa. Tämä on erityisen tärkeää toistettaessa sellaisia multimediaesityksiä, joissa grafiikka täytyy tahdistaa puhujan spiikin kanssa.

Sekä event-, että streammuotoiselle äänelle löytyy ohjelmasta omat laatuasetuksensa. Ääni toistetaan oletuksena MP3-muotoisena ja laatu voidaan asettaa 8 kbps – 160 kbps. 16 kbps asti ääni on mono. Yli 16 kbps ääni voidaan asettaa joko monoksi tai stereoksi. Äänen toistomuotoa voidaan tarvittaessa muuttaa, joten ääntä voidaan ajaa jopa kompressoimatta.

Valmiin esityksen tiedostopääte on SWF. Tämä tiedosto pitää sisällään vektori- ja bittikarttagrafiikan sekä animaatiotiedot. SWF pitää sisällään myös esityksen audiotiedot sekä interaktiivisuuden. Selaimeen puolestaan tarvitaan oma Flash-player, jotta SWF-tiedoston sisältö voidaan toistaa. Flash-kehittimen oma projekti- tiedostomuoto on FLA.

SWF-tiedosto luodaan Windows-versiossa painamalla CTRL+ENTER-näppäinyhdistelmää. Macintosh OSX-käyttöjärjestelmässä käytetään CMD+ENTER-yhdistelmää. Esitystiedosto voidaan luoda myös Publish Settings-asetuksen kautta. Tällöin projektista on mahdollista luoda myös muitakin tiedostoja, kuten valmis HTML-kehys SWF:n esittämiseen. Ohjelmasta on mahdollisuus luoda myös omalla playerillä varustetut projector-tiedostot sekä Windows- että Macintosh-käyttöjärjestelmään. Tällaisten tiedostojen esittämiseen ei tarvita silloin selainta, vaan esitys voidaan toistaa millä tahansa PC:llä tai Mac-koneella, kuva 2.



Kuva 2. Ohjelman julkaisuasetusten määrittäminen

2.2.2 MovieClip

Monimutkaiset graafiset hierarkiat ja pelityyppiset tapahtumat ovat mahdollisia vain MovieClip-luokan avulla. Flashissä MovieClip on määritelty luokaksi ja siitä on sen vuoksi mahdollista luoda instansseja. Nämä MovieClip-instanssit ovat visuaalisesti graafisia animaatio-olioita, joita on mahdollista ohjata ActionScript-ohjelmointikielellä.

MovieClip pitää sisällään oman aikajanan eli timelinen ja tuo aikajana toimii samoin kuin root-aikajanakin. Näitä MovieClippejä on mahdollista sijoittaa toistensa sisään, jolloin ne muodostavat keskenään monimutkaisia hierarkioita. Samoin kuin oikeassa elämässä, esim. autoa ei ajatella lukemattomien osiensa summana, vaan objektina, samoin MovieClipitkin muodostavat keskenään kokonaisuuksia, jotka voidaan nähdä yhtenä objektina.

MovieClipit kykenevät välittämään tietoa toisilleen ja tämä puolestaan mahdollistaa monimutkaisten syy-seuraussuhteiden rakentamisen ActionScript-kielen avulla. Jokainen MovieClip voi katsoa tapahtumia omalta näkökulmaltaan. Esim. biljardipelissä pallo voi pitää sisällään koodin joka tutkii, osuiko joku toisista palloista siihen itseensä. Kun tällainen tapahtuu, koodi huolehtii pallo-objektin siirtämisestä x- ja y-akselistossa niillä parametreillä, jotka tuossa tilanteessa pallolle välitetään. Lopputuloksena on aidon tuntuinen pallojen törmäystapahtuma.

Varsinainen Flash-esitys toimii stagen päällä. Stage näkyy käyttäjälle visuaalisesti valkoisena alueena root-scenessä. Tämä on ikään kuin näyttämö, jossa näyttelijät (MovieClipit) esiintyvät. Stage itsessään on Flashin luokka, joten jokainen stagella oleva MovieClip on tämän luokan aliluokka. Kun MovieClippeihin halutaan syöttää käskyjä, täytyy tietää, missä tasossa ko. MovieClip sijaitsee hierarkiassa. Ajatellaan esimerkiksi tilannetta, jossa root-tasolta halutaan välittää tietoa stagella olevan MovieClipin sisällä olevalle MovieClipille. Tällöin käskyyn täytyy sisällyttää tiedostopolku, jotta käskyt menevät oikeaan osoitteeseen. Esim. neliön sisällä olevan toisen neliön kääntäminen 10 astetta myötäpäivään tapahtuisi seuraavasti:

```
var kulma:Number = 10;

mc1.mc2._rotation += kulma;
```

MovieClip-luokka pitää sisällään ominaisuuksia kuten koko, x-, y-sijaintia stagella, alpha-arvo ja color-arvo. Näihin ominaisuuksiin voidaan vaikuttaa kirjoittamalla MoviClipin perään pisteen sekä ”_”-merkin. Alla muutamia esimerkkejä.

```
mc._x = 100; // tarkoittaa mc-nimisen MovieClipin sijainnin määräämistä vaakasuunnassa 100 pikseliä stagen vasemmasta reunasta oikealle päin.

mc._y = 100; // mc:n sijainnin määrääminen stagen yläreunasta 100 pikseliä alaspäin.

mc._width = 200; // mc:n leveyden määrääminen 200 pikseliksi.

mc._height = 200; // mc:n korkeuden määrääminen 200 pikseliksi.

mc._alpha = 50; // mc:n opasiteetin määrääminen 50%:ksi.
```

MovieClipin (mc) värin määritys (color) tapahtuu seuraavasti:

```
var c:Color = new Color(mc);

c.setRGB(0xFFFFFF);
```

Em. scriptillä asetetaan mc-niminen MovieClip väriltään valkoiseksi.

MovieClipeille voidaan antaa myös käsky toistaa sen oma sisäinen animaatio. Tällöin MovieClipin oman aikajanan lukupää lähtee liikkeelle ja toistaa siihen rakennetun animaation. Esim. mc1-MovieClipin sisällä olevalle mc2-MovieClipille voidaan root-tasolla antaa käsky seuraavasti:

```
mc1.mc2.play();
```

MovieClippejä voidaan myös monistaa ja hävittää käyttäen ActionScriptiä. Esimerkiksi monistamalla ”mc”-niminen MovieClippi ”mc1”-nimiseksi tapahtuu käyttämällä duplicateMovieClip-metodia esim. seuraavan esimerkin mukaisesti.

```
var i:Number = 1;

duplicateMovieClip(mc,”mc”+i,_root.getNextHighestDepth());
```

Viimeinen parametri ”_root.getNextHighestDepth()” määrittelee sen, että syntyvä kopio syntyy root-tasolle ja sijaitsee MovieClip-pinon päällimmäisenä. Ilman tätä määrittelyä kopio ei näkyisi stagella. Kopion poistaminen tapahtuu käyttämällä removeMovieClip-metodia seuraavasti:

```
removeMovieClip(_root[”mc”+i]);
```

2.2.3 ActionScript 2.0

Työssäni käytin ActionScript 2.0-versiota, koska olin tottunut käyttämään sitä aikaisemmissa projekteissa. Vaikka versio onkin jo melkein kymmenen vuotta vanha, ei se silti ole mahdollisuuksiltaan heikko. Tällä versiolla voi toteuttaa pitkälti samoja tehtäviä kuin versiolla 3.0. Tehtävässäni pääpaino olikin enemmän sovelluksen graafisella ulkoasulla ja laskentametodin visuaalisella toimivuudella kuin tehokkaalla peliohjelmoinnilla, mihin versio 3.0 on puolestaan soveltuu parhaiten. Sovellukseni haastavimmat koodiosuudet liittyivät tehtäväosioiden harjoitussivuihin. Näissä sivuissa koodi toimii proseduraalisesti, eli ne eivät sisällä olio-ohjelmointia.

ActionScript 2.0:n tarkempi dokumentaatio löytyy Adoben sivuilta osoitteesta: http://help.adobe.com/en_US/AS2LCR/Flash_10.0/help.html?content=Part2_AS2_LangRef_1.html

2.2.4 Tietoturva

Adobe on julkaissut säännöllisesti turvallisuuspäivityksiä koskien Flash-playeriä. Päivitykset ovat saattaneet koskea muistin hallintaa, mutta myös vakavampia ongelmia. Eräs tällainen korjaus koski ns. nolla-päivän ongelmaa. Tällöin hyökkääjä saattoi päästä käsiksi käyttäjän tietokoneeseen joko siten, että käyttäjä avasi saastuneen sähköpostilinkin tai kävi saastuneella web-sivustolla. Tästä löytyy artikkeli osoitteesta:

http://gcn.com/articles/2012/02/17/ecg-adobe-zero-day-flash-flaw.aspx

Lisää Flash-korjauksista löytyy osoitteesta:

http://www.locklizard.com/adobe-flash-security.htm

Toinen Flashin ominaisuus on sen tapa tallentua selaimen välimuistiin. Tästä seuraa se, että sovelluksia voi useinkin käyttää, vaikka oikeudet olisikin käyttäjältä poistettu. Tätä tapahtuu varsinkin siinä tapauksessa, kun kirjautuminen tapahtuu SWF-sivun kautta. Ongelmaa voi yrittää rajoittaa sillä, että jokainen SWF-tiedosto avataan PHP-sivun kautta. Tällöin PHP-sivulle voi määrittää, ettei seuraavaksi avattava SWF-tiedosto tallennu selaimen välimuistiin. Kuitenkin Flash-esitykset koostuvat usein SWF-sivuista, jotka avautuvat edellisen SWF:n kutsumana. Tämän vuoksi on hankalaa estää täydellisesti Flash-esityksen latautumista välimuistiin.

Alla esimerkki PHP-scriptistä.

```
<?php
header('Content-type: application/x-shockwave-flash');
header('Expires: Thu, 01 Jan 1970 00:00:00 GMT, -1');
header('Cache-Control: no-cache, no-store, must-revalidate');
header('Pragma: no-cache');
header('location: http://www.jokuosoite/index.swf');
echofile_get_contents('http://www.jokuosoite/index.swf');
?>
```

On hyvin tärkeää asentaa kaikki uusimmat Flash-player-päivitykset, koska niissä on usein mukana myös monia tietoturvaa parantavia korjauksia.

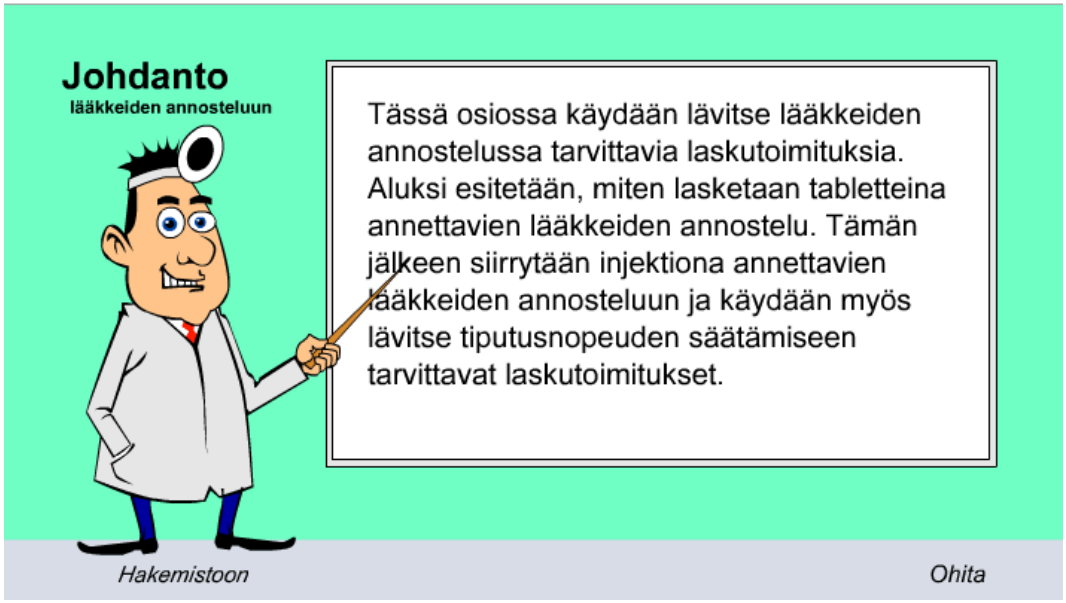
3 FLASH-PROJEKTIN VAIHEET

3.1 Suunnitteluvaihe

3.1.1 Idean sisäistäminen

Koska projektissa oli kyse laskentametodin visualisoinnista, oli hyvin tärkeää käydä yhdessä keskusteluja siitä, millainen esitystapa vastaisi parhaiten opetettavaa metodia. Keskustelujen jälkeen oli selvää, että esityksen tulee muodostua havainnollisista animaatioista, joita viedään eteenpäin selostusäänellä. Lisäksi esityksen tuli olla sekä suomen- että ruotsinkielinen.

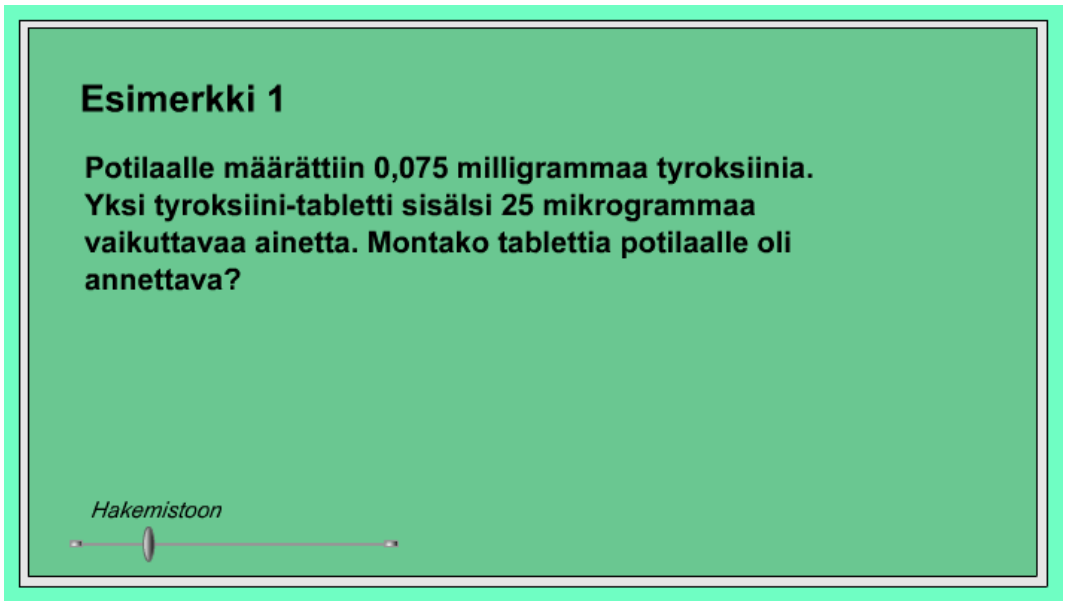
Metodissa käytetään taulukkomenetelmää, jossa käsiteltävät luvut asetetaan taulukkoon sovitulla tavalla, jolloin vastaus saadaan mekaanisesti jakamalla ja kertomalla ne keskenään, **kuva 9.**



Kuva 3. Tehtäväosion johdantosivu

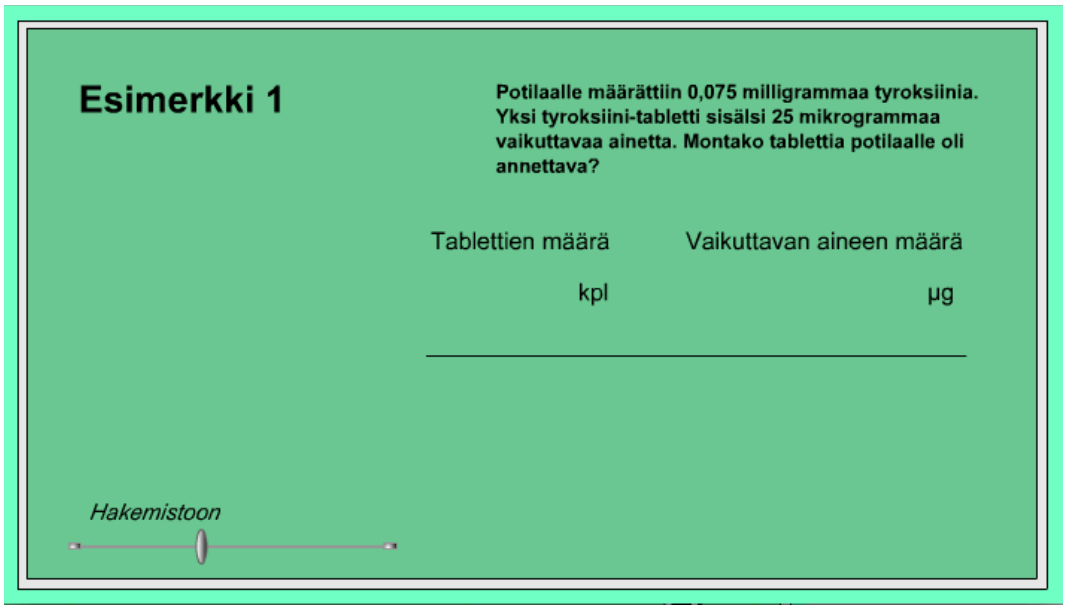
Aluksi animaatiohahmo esittelee tehtäväosion sisällön, jonka jälkeen esitys siirtyy varsinaiseen tehtäväosioon. Tarvittaessa johdannosta voidaan siirtyä takaisin hakemistoon painamalla ”Hakemistoon”-nappia. Johdanto voidaan myös ohittaa haluttaessa painamalla ”Ohita”-nappia.

Seuraavaksi esitellään ratkaistava tehtävä, kuva 4.



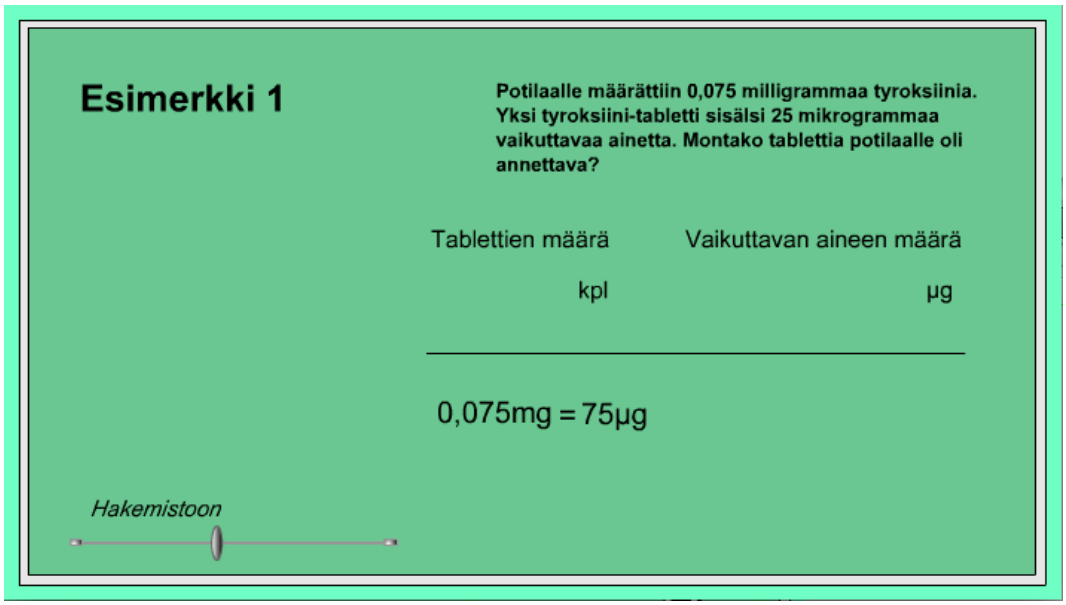
Kuva 4. Varsinainen tehtävä

Tämän jälkeen luodaan tehtävään sopiva taulukko, johon voidaan sijoittaa tiedossa olevat arvot, kuva 5.



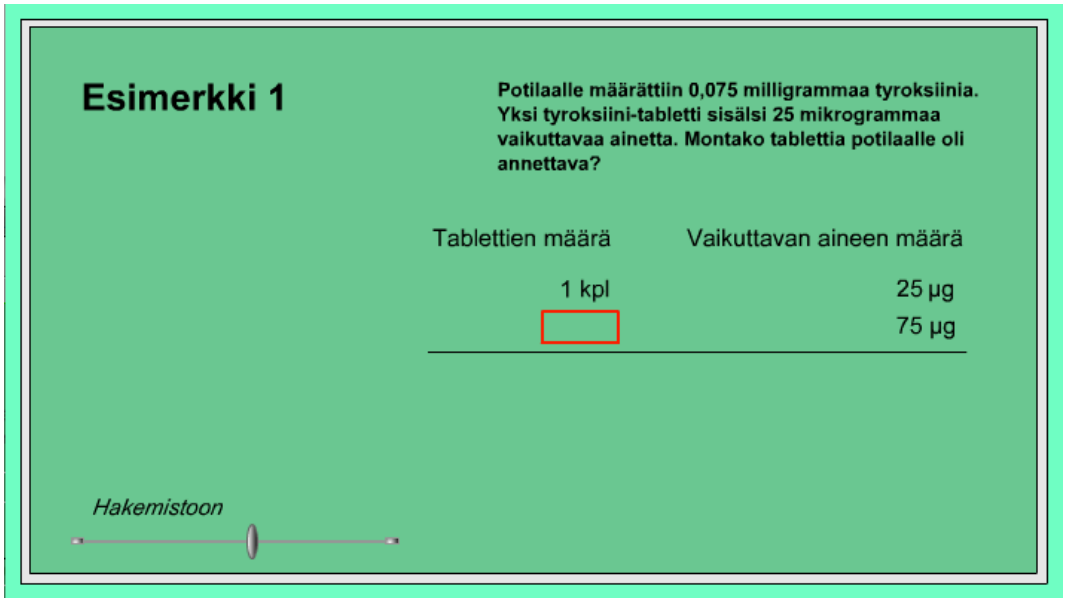
Kuva 5. Taulukon visuaalinen ulkoasu

Seuraavaksi suoritetaan tarvittavat yksikkömuunnokset, kuva 6.



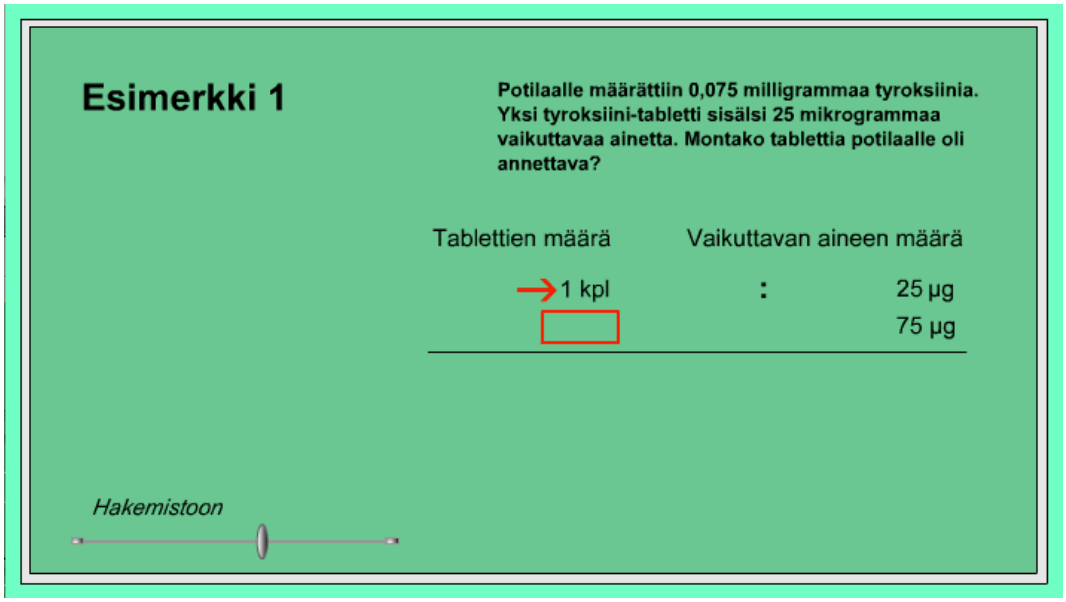
Kuva 6. Milligrammojen muuntaminen mikrogrammoiksi

Tämän jälkeen sijoitetaan numerot oikeille paikoilleen ja laitetaan punainen laatikko kysytyn luvun paikalle. Tässä tapauksessa tablettien määrän kohdalle, kuva 7.



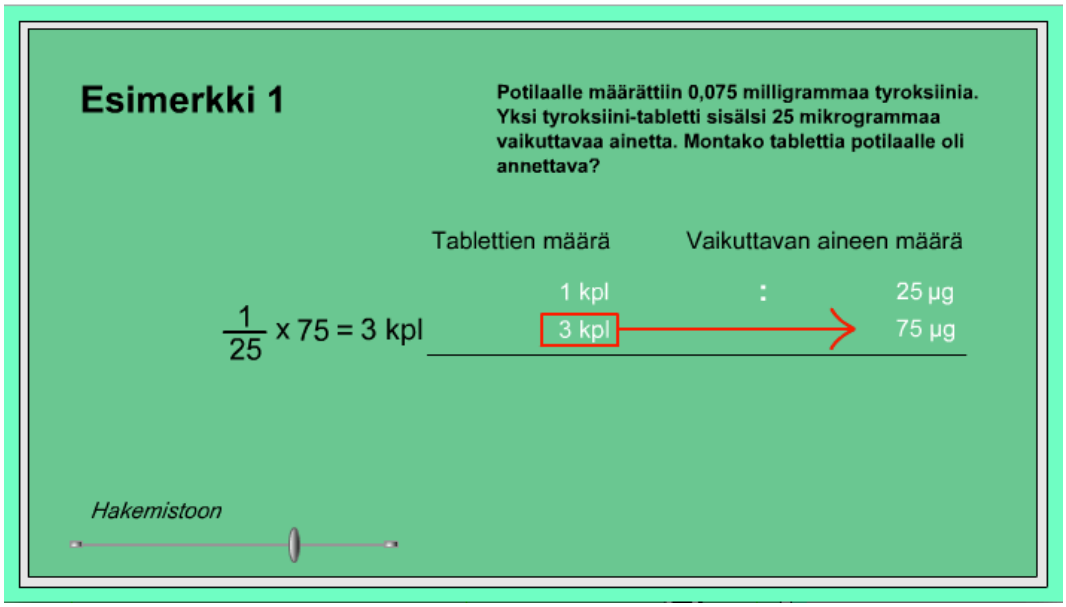
Kuva 7. Numeroiden sijoitus

Seuraavaksi laitetaan jakomerkki sen rivin kohdalle, joka on täynnä. Tällä kertaa yläriville, kuva 8.



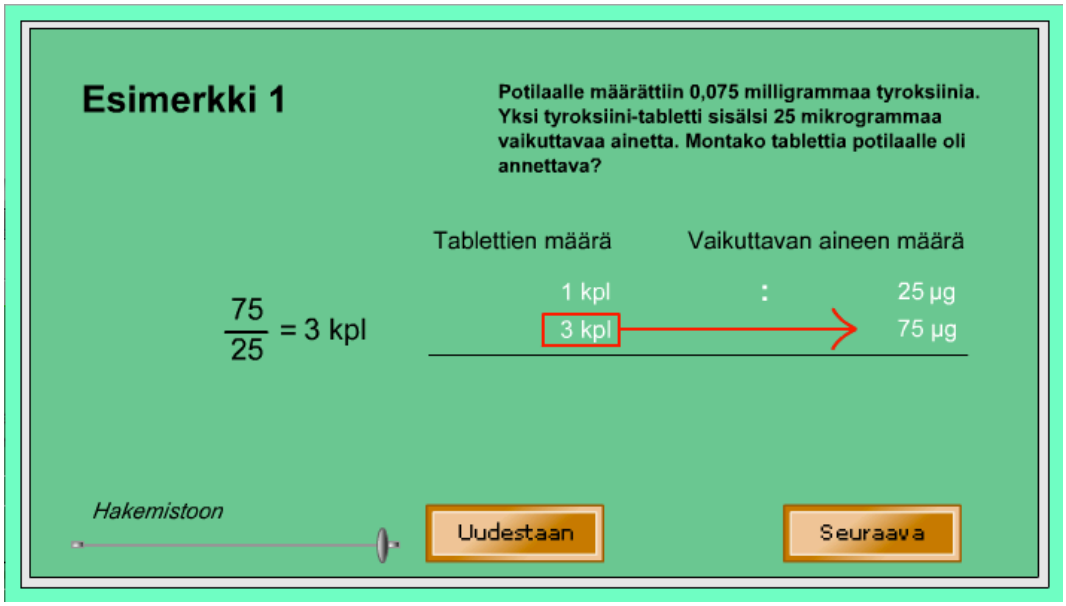
Kuva 8. Jakomerkin sijoittaminen

Lopuksi punaisesta laatikosta piirretään viiva viereiseen lukuun päin, jolloin saadaan tietää laskusuunta. Tämän jälkeen selitetään laskutapa ja vastaus sijoitetaan punaisen laatikon sisään, kuva 9.



Kuva 9. Laskusuunnan ratkaiseminen

Lopuksi kerrotaan vielä tarvittaessa kuinka ratkaisu voidaan muuttaa yksinkertaisempaan muotoon. Tässä vaiheessa käyttäjälle avautuu mahdollisuus katsoa sama esitys uudelleen tai siirtyä seuraavaan esitykseen. Esitystä voidaan lisäksi haluttaessa ”kelata” eteen- tai taaksepäin ala-vasemmalla olevalla säätimellä missä tahansa esityksen vaiheessa, kuva 10.



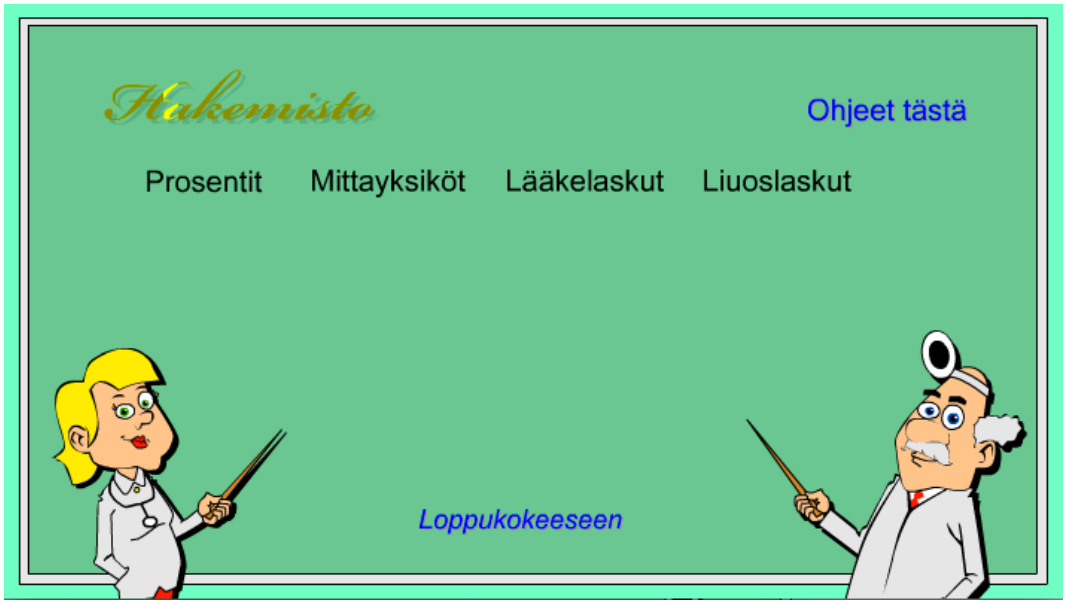
Kuva 10. Esityksen lopputilanne

3.1.2 Ulkoasun suunnittelu

Ensimmäinen ajatus oli se, että sovelluksen käyttäminen olisi mahdollisimman helppoa. Tämän vuoksi päätettiin tehdä sovellus siten, että se täyttää kokonaan tietokoneen näytön. Tehtävä oli suhteellisen yksinkertainen toteuttaa, koska Flash-grafiikka on skaalautuvaa, vektorimuotoista grafiikkaa. Toinen vaatimus oli se, että kaikki tarpeellinen tuli näkyä näytössä tehtäväsivun alusta asti. Minkäänlaisia alasvetovalikkoja ei kesken esitystä tarvitse käyttää, joten jopa hiiren käyttö jää sovelluksessa mahdollisimman vähäiseksi. Oppilas voi keskittyä pelkästään metodin opiskeluun erilaisten tehtäväesimerkkien avulla.

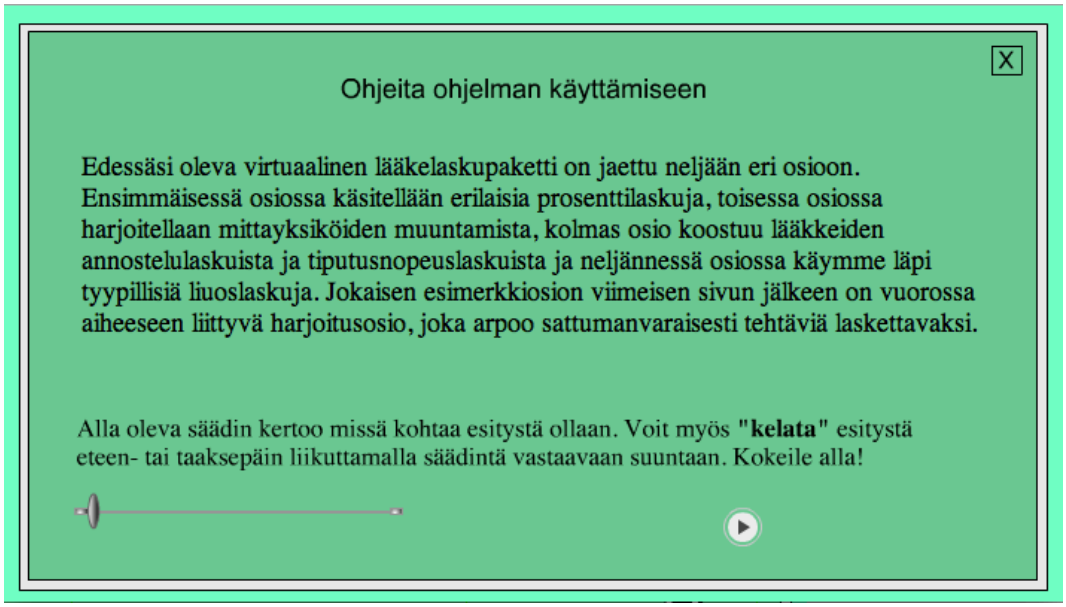
Värimaailmaksi valittiin vihreä, koska se on rauhoittava väri ja taustan ilmeeksi ”liitutaulu”. Ehdotettiin myös, että käytettäisiin jotain animaatiohahmoa, joka selittäisi tehtäväosioon liittyviä periaatteita. Tämä hahmo tulisi esiin jokaisen esitysosion ensimmäisessä sivussa. Ehdotusta pidettiin hyvänä, joten päätimme käyttää lääkäri- ja sairaanhoitajahahmoa. Nämä hahmot tilattiin Savon Sanomissa työskentelevältä veljeltäni graafikko Timo Filpukselta. Häntä pyydettiin tekemään hahmot nuoresta ja vanhasta lääkäristä sekä sairaanhoitajasta. Hän toteutti nuo hahmot Adobe Illustrator-ohjelmalla. Adobe Illustrator-ohjelma tuottaa Flash-kehittimen käyttämää vektorigrafiikkaa. Saatuani hahmot, ne muutettiin Flash-kehittimessä sellaisiksi, että niitä pystyi animoimaan. Lopuksi tehtiin hahmoihin tarvittavat koodit sopivien liikkeiden aikaansaamiseksi.

Sovelluksen rakenne on jaettu selkeisiin osa-alueisiin, joihin pääsee käsiksi etusivun hakemiston kautta. Hakemistosivulla on lisäksi ”Ohjeet tästä”-nappi, jota painamalla saa tietoa sovelluksen käytöstä. Tähän hakemistoon on pääsy kaikilta tehtäväsivuilta, kuva 11.



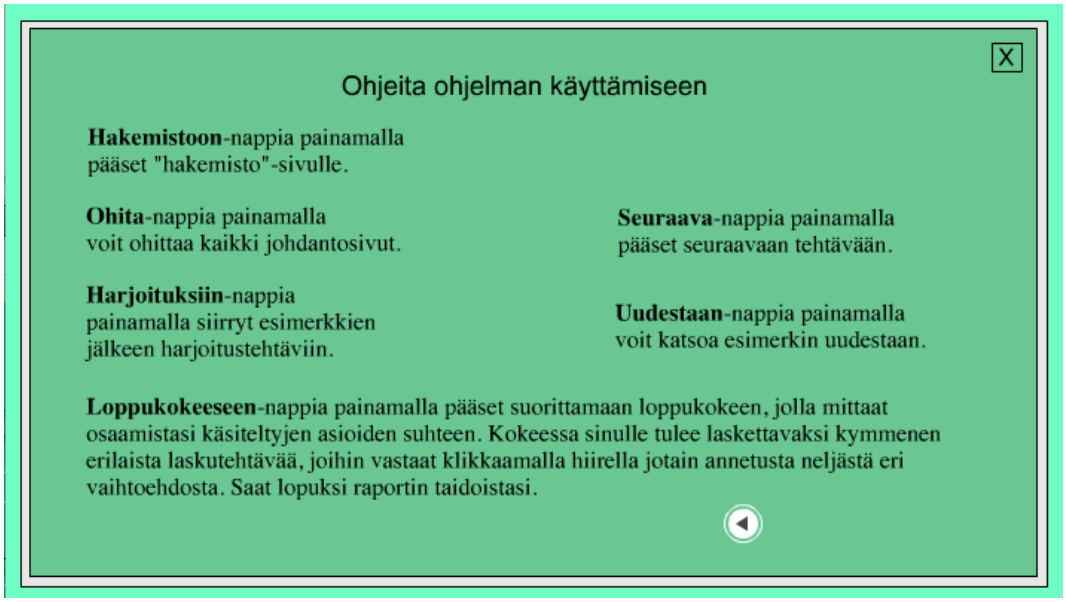
Kuva 11. Hakemistosivun ulkoasu

Pienestä kolmiosta alemmassa kuvassa pääsee klikkaamalla seuraavaan ohjesivuun, kuva 12.



Kuva 12. Ohjesivun ulkoasu

Oikeassa yläkulmassa olevaa rastia klikkaamalla pääsee takaisin hakemistoon ja pientä kolmiota klikkaamalla puolestaan edelliselle ohjesivulle, kuva 13.



Kuva 13. Viimeisen ohjesivun ulkoasu

Jokaisen tehtäväosion lopussa on harjoitussivu, jossa oppilas voi harjoitella laskemaan annettuja laskuesimerkkejä. Sitä mukaan, kun oppilas vastaa kysymyksiin, hänelle kerrotaan oikeiden vastausten prosenttiosuus. Halutessaan oppilas voi katsoa vastauksen ja laskutavan painamalla antamansa vastauksen jälkeen ”Katso vastaus”-nappia. Tuolloin näyttöön tulee näkyviin koko laskuprosessi. Nämä osiot toimivat luuppeina eli alkavat uudestaan, kun kaikki kysymykset on läpikäyty. Samalla prosenttilaskuri nollautuu ja kaikki alkaa alusta, kuva 14.

Kuva 14. Tehtäväosion harjoitussivu

Alla tilanne ”Katso vastaus”-napin painalluksen jälkeen. Vastauksen voi poistaa painamalla nappia uudelleen. Tällöin napin alla lukee ”Poista vastaus”, kuva 15.

Kuva 15. Vastaustilanteen ulkoasu

Lopuksi oppilas suorittaa ns. loppukokeen, jossa on kymmenen kysymystä painotettuna osioiden koon suhteessa. Näihin kysymyksiin vastattuaan oppilas saa raportin onnistumisestaan. Tähän kokeeseen voi osallistua suoraan myös hakemiston kautta. Kysymykset arvotaan taulukosta, joten ne ovat joka kerta erilaiset, samoin myös vastausvaihtoehdot. Tässäkin osiossa oppilaan on mahdollista nähdä laskutapa painamalla ”Katso vastaus”-nappi, kuva 16.

Kuva 16. Loppukokeen ulkoasu

Alla kuva loppuraportista, kuva 17.

Kuva 17. Loppuraportin ulkoasu

Oppilaalla on milloin tahansa mahdollisuus siirtyä seuraavaan osioon ohittamalla tehtävät ”Ohita”-napilla. Harjoitussivusta on myös pääsy takaisin hakemisto-sivulle, jonka kautta oppilas voi valita myös jonkin toisen tehtäväosion. Peruslähtökohta on kuitenkin se, että oppilas käy läpi järjestyksessä kaikki osiot, sekä niihin liittyvät harjoitustehtävät. Sovellus on rakennettu siten, että siirtyminen seuraavaan osioon käy aina ”Ohita”-napin kautta. Esimerkkisivujen vaihto tapahtuu puolestaan painamalla ”Seuraava”-nappia aina esityksen lopussa.

Esimerkkisivujen vasemmassa alalaidassa on lisäksi säädin, jolla esitystä voidaan ”kelata” tarpeen tullen. Tämä antaa oppilaalle mahdollisuuden katsoa jotain kohtaa esityksestä uudelleen. Samalla säätimen sijainti näytöllä kertoo karkeasti esityksen jäljellä olevan keston, kuva 18.

Kuva 18. Säädin, jolla esitystä voi halutessa kelata eteen ja taaksepäin

3.2 Toteutusvaihe

3.2.1 Käyttöalusta

Sovellus toimii kaikissa selaimissa niissä olevan Flash-laajennuksen avulla. Tämän ansiosta esitys on aina saman näköinen riippumatta siitä, millä käyttöjärjestelmällä esitystä katsotaan. Flashillä on kuitenkin ikävä ominaisuus tallentua selaimen välimuistiin, joten sitä yritettiin ratkaista myöhemmin PHP-scriptillä. Tämä ei kuitenkaan onnistunut kuin ensimmäisen sivun osalta, joten päätettiin ratkaista ongelma myöhemmin siinä yhteydessä, kun toteutetaan tietokantapohjainen asiakashallintajärjestelmä.

Tällä hetkellä sovelluksen ensimmäinen sivu on PHP-sivu, mikä on jäänteenä yrityksestä estää selainta tallentamasta Flash-sivuja välimuistiinsa. Väärinkäytön mahdollisuutta yritettiin kuitenkin vähentää rakentamalla ohjelman rakenteen sellaiseksi, että se ei suostu menemään esitykseen, ellei kirjautumista ole suoritettu. Vaikka joku henkilö yrittäisi ohittaa kirjautumisen tietämällä jonkin sivun nimen, ei hän voisi käyttää tuota sivua, vaan sovellus ohjaisi hänet tuossa tilanteessa takaisin kirjautumissivulle. Kirjautumissivu on myös tällä hetkellä Flash-sivu.

3.2.2 Koodausvaihe

Koodaamisessa käytettiin Flash-kehittimessä olevaa Action Script 2.0 versiota. Syynä tähän oli se, että olin tottunut käyttämään tuota versiota aikaisemmissa sovelluksissani. Uusin ActionScript-versio on nykyään 3.0. Suurimmat erot liittyvät versioiden välillä siihen, että 3.0 muistuttaa enemmän Java-ohjelmoinnista tuttua luokka-ajattelua. Myös syntaksi muistuttaa Java-kielessä käytettyä syntaksia. Action Script 2.0 toimii myös karkeasti olio-perusteisesti, mutta muistuttaa enemmän syntaksiltaan JavaScriptiä.

Sovelluksen root-aikajanalla sijaitsee esityksen lataamiseen liittyviä käskyjä. Lisäksi siinä tutkitaan, onko käyttäjä kirjautunut sisään. Esityssivujen napeissa puolestaan on käskyt, joko näyttää sivu uudestaan tai kutsua esiin uusi esityssivu.

Tämä on erilaista verrattuna ActionScript 3.0:aan. ActionScript 3.0:ssa ei voida laittaa nappeihin itseensä koodia, vaan kaikki koodi on löydyttävä yhdestä paikasta, tässä tapauksessa, root-aikajanalta. Näin voidaan toimia toki myös ActionScript 2.0:ssa. Jos nimeää napit tietyllä tavalla, on silloin mahdollista sijoittaa koodit samoin root- tasolle kuin ActionScript 3.0:ssa.

Root-aikajanalta löytyy myös ohjauskäskyt animaatiohahmojen liikkeiden ohjaamiseen suhteessa ääneen. Flashissä ei ole mahdollisuutta saada esiin tietoa äänen varsinaisesta voimakkuusvaihtelusta, jolloin tuon tiedon käyttäminen animaatiohahmojen liikeohjaukseen olisi ollut helppoa. Flash tietää ainoastaan sen, onko ääni päällä vai ei sekä sen, kuinka suuri on äänen kokonaisvoimakkuus. Tämän vuoksi jouduttiin manuaalisesti kirjoittamaan aikajanalle niihin kohtiin käskyt, joissa animaatiohahmo on puhuvinaan ja ne joissa lupaa puhumiseen ei enää ole, jotta hahmojen liikkeet olisivat synkassa esityksen äänen kanssa.

Suurimmat koodit liittyvät harjoitus- ja loppukoeosioihin. Näissä osioissa kaikki kysymys- ja vastausmateriaalit ovat taulukoissa, joista ne sitten arvotaan näytettäväksi sovitulla tavalla. Vastauskohdat ovat myös aktiivisia ”nappeja” ja pitävät sisällään koodit, joissa on mukana tieto siitä onko vastaus oikea ja sen, ettei samaa kysymystä arvota enää uudelleen. Lisäksi niissä on koodia siitä, näytetäänkö ”Katso vastaus”-nappia ja sen kautta on mahdollisuus katsoa oikea vastaus ja laskutapa.

Haastavin koodattava osio oli loppukoe, koska kysymykset tuli painottaa eri tehtäväosioiden koon suhteen. Lisäksi tiedon vieminen siitä, mikä oli oikea vastaus oli haastavaa. Tämä tieto täytyi kuitenkin saada esiin, koska sitä käytetään kun oppilas haluaa tietää laskutavan ”Katso vastaus”-nappia painamalla.

3.3 Muutokset

3.3.1 Käyttäjien palaute

Alun alkaen ohjelmaa oli tarkoitus käyttää opetuksen tukena, joten joitain toimintoja ei siihen alussa kuulunut. Näitä ovat esim. ”Katso vastaus”-mahdollisuus ja mahdollisuus hypätä kesken kaiken hakemistoon. Myös loppukoe oli alkujaan suunniteltu oikeaksi kokeeksi, joka suoritetaan vasta harjoittelun lopuksi.

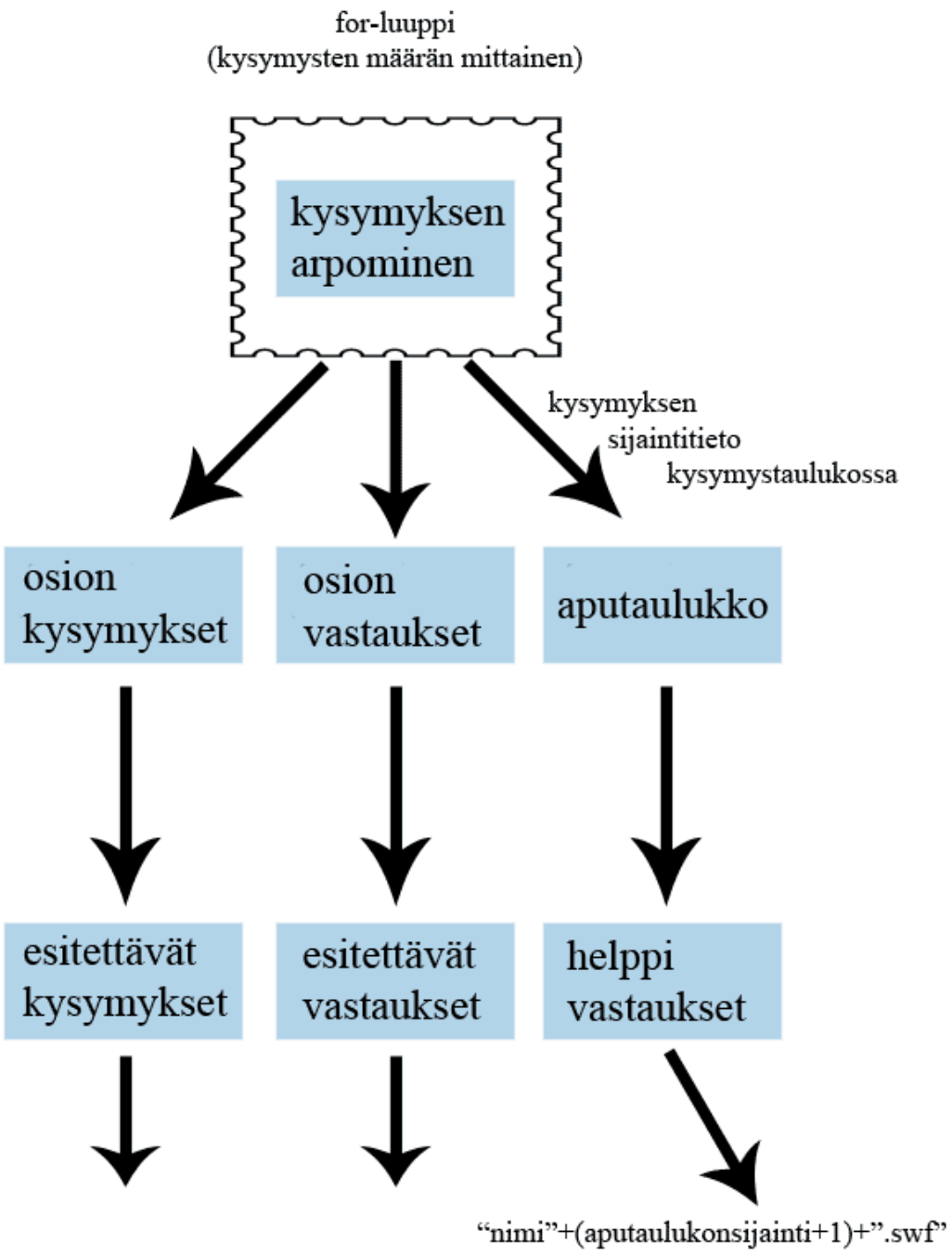
Käyttäjät kokivat kuitenkin loppukokeenkin vain harjoitteluksi, joten myös sinne oli pakko tehdä ”Katso vastaus”-nappi ja mahdollisuus palata kesken kaiken hakemistoon. Käyttäjät kokivat siis ohjelman toisella tavalla kuin miten olimme sen itse ajatelleet. Tämä voidaan huomata aikaisemmista projekteista pitävän samalla tavalla paikkansa. Ohjelman suunnittelija ei useinkaan osaa nähdä kaikkia niitä toimintoja, joita käyttäjät puolestaan sinne kaipaavat. Siksi osa suunnittelusta kuluu siihen, että sovellusta joudutaan hienosäätämään, jotta se vastaisi paremmin käyttäjien toiveita.

Koska esitys pitää sisällään paljon manuaalisesti tehtyä materiaalia, käyttäjät ovat löytäneet sieltä myös virheitä. Näitä ovat olleet suoranaiset virheet vastauksissa sekä epäloogisuudet itse ohjelman toiminnoissa. Esimerkiksi ”Seuraava”-napista ei siirryttykään seuraavaan sivuun, vaan jonnekin muualle. Näiden korjaaminen on myös vienyt aikaa.

3.3.2 Toimintojen lisääminen

Käyttäjiltä saadun palautteen perusteella ohjelmaan tehtiin muutokset koskien mahdollisuutta palata hakemistoon sekä laskutavan näyttämiseen tarvittaessa. Nämä aiheuttivat välillä ongelmia, varsinkin loppukokeen kanssa. Loppukoe oli alun perin suunniteltu kokeeksi, joka täytyi suorittaa alusta loppuun, joten laskutavan näyttäminen oli tilanteessa haastavin asia.

Ohjelmaan luotiin aputaulukot, jotta saatiin tietää mikä laskutapa esitetään loppukokeessa. Näihin taulukoihin tallennettiin tiedot kysymystilanteesta. Koska vastausvaihtoehtoja on neljä, ohjelmaan tehtiin neljä taulukkoa, jotka olivat kukin oman osionsa mittaisia. Näihin tallennettiin suoraan luvut nollasta eteenpäin riippuen osion kysymysten vaihtoehtomäärästä. Näistä taulukoista **siirrettiin** lopuksi luvut arvonnan perusteella seuraavaan aputaulukkoon, joten tähän taulukkoon muodostuneiden lukujen perusteella ohjelma tietää, mikä vastaustapa tulee esittää kysyjälle. Kaikki kysymykset ja niihin liittyvät vastaukset on siis arvottu jo siinä vaiheessa, kun loppukoeosiota aletaan esittää, kuva 19.



Kuva 19. Osion kysymysten ja vastausten hallinnointi

4 FLASH-TEKNIIKAN TULEVAISUUS

Flash-tekniikan korvaajaksi on kehitetty HTML5. HTML5 on yhteistyöprojekti World Wide Web Consortium (W3C) ja Web Hypertext Application Technology Working Group (WHATWG) välillä. Vuonna 2006 nämä päättivät yhdessä luoda uuden version HTML:stä (W3C).

HTML5:een päätettiin luoda uusia ominaisuuksia, jotka perustuvat vanhoihin tekniikoihin, kuten HTML, CSS ja JavaScript. Tarkoituksena oli vähentää ulkopuolisten lisäosien, kuten Flash, tarvetta. Samoin haluttiin vähentää scriptauksen tarvetta ja siirtyä käyttämään HTML:stä tuttua muotoilukieltä. HTML5:n tuli myös olla laiteriippumaton eli samat sivut näkyvät sekä selaimilla että mobiililaitteilla.

Uusia ominaisuuksia ovat esim. piirto-ominaisuudet, videotoisto-ominaisuus, sisältöelementit kuten <article>, <footer>, <header>, <nav>, <section>. Uutena tulivat myös kalenterikontrolli, date-, time-ominaisuudet, email, url ja search.

HTML5-tekniikka on kuitenkin vielä tiensä alussa. Markkinoilla olevien editorien ominaisuudet ovat vielä rajalliset. Esim. äänen käyttö on haastavaa. Jotta ääntä voitaisiin käyttää esityksessä, sen ohjaaminen täytyy hoitaa erikseen Javascriptillä. Erilaisilla editoreilla, kuten Applen OSX-käyttöjärjestelmässä toimiva Purple, voidaan kuitenkin suhteellisen helposti rakentaa visuaalisesti koko sivuston grafiikka sekä sen animaatiot. Tämän lisäksi myös napit voidaan rakentaa editoreilla toimiviksi, joten HTML5-sivujen määrä kasvaa varmasti tulevina aikoina. Monet isot toimijat ovat jo siirtyneet HTML5-tekniikkaan. Tästä esimerkkinä Yle Areena. Peliohjelmointiin Purplesta ei kuitenkaan ole. Nappeihin liittyvät scriptit täytyy valita olemassa olevista vaihtoehdoista, joten vapaus varsinaiseen ohjelmointiin on rajoittunutta.

Web-sivujen tekniikkana se varmasti yleistyy hyvinkin nopeasti, koska sivut ovat yhteensopivia sekä web-selaimissa että mobiilialustoilla. Ongelmaksi muodostuu tällä hetkellä äänen käyttö sekä erilaiset syy-seuraus-hierarkiat, joita tarvitaan peliohjelmoinnissa. Tämän vuoksi HTML5 ei nähdäkseni kykene kovinkaan nopeas-

ti korvaamaan kaikkea sitä, mitä on aikaisemmin tuotettu Flash-tekniikalla. On kuitenkin erittäin mielenkiintoista seurata, mihin suuntaan HTML5 tulee kehittymään. Uskon sen aikanaan syrjäyttävän Flashin, kunhan em. ongelmat saadaan ratkaistua. Editorien tulee myös olla käytettävyydeltään nykyisen Flash-kehittimen tyyppinen. Animaatioiden rakentamiseen kun on aina tarvittu visuaalisia välineitä.

Aika näyttää missä tahdissa Flash-tekniikka vähenee ja tilalle tulee HTML5. Kuitenkin monien kaupallisten sovellusten muuttaminen HTML5-pohjaisiksi vie aikaa. Myös useat peliportaalien kautta pelattavat pelit on toteutettu Flash-tekniikalla. Näiden kaikkien sovellusten muuttaminen HTML5:ksi merkitsisi ylimääräisiä kustannuksia, joten henkilökohtaisesti en usko Flash-tekniikan katoavan käytöstä kovinkaan nopeasti. Web-sivujen toteutuksessa siirtyminen HTML5-tekniikkaan tapahtuu varmasti nopeaan tahtiin, koska ko. tekniikka mahdollistaa mm. sivujen skaalaamisen erikokoisille näytöille. Koska mobiilisovellusten määrä on lisääntynyt viime vuosina kovasti, sivujen yhteensopivuus sekä tietokoneiden että mobiilinäyttöjen kanssa onkin merkittävä asia.

HTML5 ei ole vielä virallinen standardi, eivätkä kaikki selaimet tue kaikkia sen ominaisuuksia. Merkittavimmät selaimet (Safari, Chrome, Opera, Internet Explorer) lisäävät kuitenkin jatkuvasti uusia HTML5-ominaisuuksia uusimpiin versioihinsa (w3schools.com.).

5 JOHTOPÄÄTÖKSET

Projektia tehdessäni sain huomata sen, että käytin siihen aikaa enemmän kuin alun perin arvioin. Tämä johtui virheiden korjaukseen käytetystä ajasta. Lisäksi käyttäjiltä tulleen palautteen perusteella jouduin tekemään esitykseen tarvittavat toiminnalliset muutokset, jotka eivät alun perin kuuluneet esityksen sisältöön. Kaikki tämä lisäsi projektin toteutukseen kulunutta kokonaisaikaa. En kuitenkaan hämmästynyt ylimääräisen ajan käyttämisestä, koska aikaisemmissa projekteissa vuosien varrella olen myös joutunut uhraamaan ylimääräistä aikaa eri syistä. Varsinkin virheiden korjaus on ollut yksi suurimmista syistä projektien valmistumisen pidentymiseen.

Esitys on ollut kaiken kaikkiaan toimiva ja saanut kiitosta käyttäjien taholta. Varsinkin esityksessä käytetyt laskuja havainnollistavat animaatiot on koettu havainnollisiksi. Lisäksi animaatiohahmoista on pidetty. Tämän johdosta voin todeta esityksen olleen onnistunut ja valittu tyyli sopiva. Flash on myöskin tekniikkana ollut tarkoitukseeni sopiva.

Käyttäjämäärien kasvu on kuitenkin aiheuttanut sen, että tällä hetkellä käytössämme oleva manuaalisesti hoidettu asiakashallinta ei ole tarpeeksi tehokas tapa hoitaa asiakkaiden ohjelmalisenssitietoja. Tämän vuoksi tarkoituksenani on rakentaa tulevaisuudessa tietokantapohjainen asiakashallintajärjestelmä, johon asiakkaiden yhteystiedot, käyttäjätunnukset ja salasanat tallennetaan. Samalla järjestelmä huolehtisi automaattisesti asiakkaiden käyttäjäoikeuksista, kuten mahdollisuudesta kirjautua serverille sekä lisenssien jäljellä olevasta käyttöajasta.

Suunnitelmissa onkin uusien esitysten valmistaminen projektin pohjalta. Tarpeita tämän tyyppisille tuotteille tuntuu olevan niin peruskouluissa kuin ammattiopistoissakin.

6 LÄHTEET

Flash-kehittimen historia:

Lauri Paakinaho: Flash-peliohjelmointi. Opinnäytetyö 2011 pdf. Viitattu 13.12.2012.

Flash-ActionScript 2.0 dokumentaatio. Lähde: Adobe.com. Viitattu 13.12.2012. Saatavissa: http://help.adobe.com/en_US/AS2LCR/Flash_10.0/help.html?content=Part2_AS2_LangRef_1.html

Flash-turvallisuuspäivityksistä. Lähde: GCN.com Viitattu 13.12.2012. Saatavissa: http://gcn.com/articles/2012/02/17/ecg-adobe-zero-day-flash-flaw.aspx

Flash-korjauksista. Lähde: locklizard.com. Viitattu 13.12.2012. Saatavissa: http://www.locklizard.com/adobe-flash-security.htm

HTML5-dokumentaatio. Lähde: W3C. Viitattu 13.12.2012. Saatavissa: http://www.w3.org/html/wg/drafts/html/master/Overview.html

HTML5-dokumentaatio. Lähde: w3schools.com. Viitattu 13.12.2012. Saatavissa: http://www.w3schools.com/tags/default.asp