Harjoitustyö (TKO_2023)

Jyri Lehtonen (72039)

(alkuperäinen 29.1.2008)
päivitetty: 8.2.2008

1 Tehtävän kuvaus ja analysointi

1.1 Tehtävänanto

Tee Java-appletti, joka kysyy käyttäjältä asioita Java-kielestä. Appletti esittää aina kysymyksen ja käyttäjä antaa vastauksen klikkaamalla painiketta (Kyllä/Ei). Kysymyksiä voi olla esim. 15. Kun kysymykset on käyty läpi, appletti näyttää tulokset graafisesti esim. pylväinä (oikeat/väärät) vastaukset. Kysymykset tulee ladata tekstitiedostosta palvelimelta, josta appletti on ladattu.

1.2 Tehtävänannon analysointi

Harjoitustyön tehtävänä oli toteuttaa yksinkertainen tietovisasovellus. Tehtävänanto jätti ratkaisuvalintoja myös tekijän pohdittavaksi. Appletti ehdon tähden tietovisan on toimittava HTML ympäristössä.

Pyrin toteuttamaan yksinkertaisen tietovisan, jossa HTML sivulle saapumisen jälkeen voi avata ohjeet peliin tai käynnistää sen. Käynnistyksen jälkeen pelaajalla on vaihtoehtoina tietovisassa valita kumpaa vastaa (kyllä tai ei) ja vastata kysymykseen. Vastauksen jälkeen pelaaja voi siirtyä eteenpäin tietovisassa. Pistetilanteensa hän saa näkyviin valitsemalla pisteet painikkeen. Pelissä on yksinkertaiset säännöt. Oikein vastaamalla pelaaja saa pisteen, väärin vastaamalla hän menettää pisteen. Pelin kysymysten päättyessä pelaaja saa näkyviin tilastot hänen suorituksestaan. Tilastoissa näkyy kuinka monta kysymystä oli oikein ja väärin, ja näiden avulla laskettavat pisteet pylväsdiagrammina.

Tietovisa etenee siten, että pelaaja ensin valitsee tiedon vastaako kysymykseen kyllä tai ei. Tämän jälkeen hän valitsee vastaa painikkeen, jonka jälkeen vastaa painike katoaa hetkeksi ja seuraava painike ilmestyy tämän viereen. Seuraava painike tuo uuden kysymyksen ja palauttaa vastaa painikkeen. Pelaajalle tulostetaan tieto pelin kulusta jokaisessa valinnassa palauteruutuun, joka sijaitsee valintapainikkeiden alla. Tätä ruutua seuraamalla pelaaja tietää mitä pitää tehdä tai mitä hänen edellinen tekonsa aiheutti. Kun kierros on päättynyt, pelaaja voi aloittaa tilastojen katsomisen jälkeen uuden kierroksen.

2 Ratkaisuperiaate

Lähdin ratkaisemaan tehtävää pohtimalla tehtävänannon kannalta oleellisia luokkia ja niiden välisiä relaatioita. Alustavasti sovelluksen toteuttamista varten näin tarpeelliseksi seuraavat luokat; GUI-, Kysymys- ja Vastausluokka. Alustavasti oli myös suunniteltu vastauksien sisältyvän kysymyksien kanssa samaan tiedostoon, mutta ensimmäinen valmis toteutus sisälsi vain kysymykset tiedostosta. Kuitenkin koska appletit toimivat hieman eritavalla kuten "normaalit sovellukset", eikä tehtävänanto vaatinut vastauksien kohdalla mitään konkreettista, päädyin "yhdistämään" luokat GUI-luokkaan siten, että tietovisa olisi yhden luokan mittainen sovellus kolmen sijasta. Ensimmäisessä versiossa vastaukset olivat "kovakoodattuja" ohjelmakoodin sisään alkuperäisen suunnitelman vastaisesti. Tällöin jos halusi muuttaa kysymyksiä, joutui muuttamaan ohjelmakoodiakin. Viimeisessä versiossa (1.1) tämä muutettiin siten, että tietovisan kysymykset ja vastaukset sijaitsevat alkuperäisen suunnitelman mukaisesti samassa tekstitiedostossa, erotettuna välimerkillä. Enää ei ole tarpeellista muuttaa ohjelmakoodia, jos haluaa muuttaa kysymyksiä ja vastauksia.

GUI-luokka pitää sisällään tietovisan graafisen käyttöjärjestelmän, pelitekniikan, säännöt, vastauksien- ja kysymyksienkäsittelyn. Tämä luokka siis mallintaa koko tietovisan. Tehtävänannon mukaiset vastausominaisuudet kyllä ja ei, ovat mallinnettuina ympyräValinta1 `(CheckBox)` ja ympyräValinta2 `(CheckBox)` avulla. Kysymykset ladataan erilliseltä tiedostolta `(Kysymykset.txt)` sovellukseen. Tiedoston on sijaittava samassa kansiossa sovelluksen kanssa jotta kysymykset näkyisivät. Kysymyksien lukemisesta pelitekniikkaan vastaa `lueTiedosto()`-metodi. Tämä metodi tallentaa kysymykset tietovisan sisälle tietopankkiin `(ArrayList<String> kysymysLista)`, josta pelitekniikka esittää kysymykset yksi kerrallaan. Tiedosto, josta kysymykset haetaan, on tietovisaan määrätty kohtaan `luettavaTiedosto (String)`. Vastaukset kysymyksiin sijaitsevat samassa tiedostossa kysymyksien kanssa. Sovellus lukee vastaukset samalla muistiin, kun lukee kysymyksetkin. Metodi `lueTiedosto()` tallettaa vastaukset toiseen tietopankkiin `(ArrayList<String> vastausLista)`, josta pelitekniikka tarkistaa pelaajan vastatessa oikean ratkaisun kysymykseen. Tehtävänannossa ei mainittu, että kysymyksiä pitäisi sekoittaa pelikertojen välissä, joten koin tämän ratkaisun toimivaksi tässä tilanteessa. Tilastolliset tekijät tietovisaan suoritetaan piirtämällä suorakulmioita `(Rectangle2D)` joiden pituus vaihtelee pelaajan vastausten mukaisesti. Tilastollisuuden havainnoimisen helpottamiseksi tilastolliseen näkymään esitetään vielä rakenteita `(Line2D)` ja tekstiä `drawString()`.

Tietovisaan kehittyi heti ensimmäisestä kokonaisvaltaisemmasta testauksesta palauteruutu `(TextField)` graafiseen käyttöjärjestelmään. Tämän tilan avulla pystyin helposti kertomaan onnistuiko tekemäni sovellusmuutos. Tällä oli suurta vaikutusta testaamisen helpottumiseen. Tämä ruutu myös palvelee pelaajaa kertomalla hänelle toimintojensa vaikutuksen.

Tietovisan etenemiseksi oli välttämätöntä lisätä kuuntelijoita painikkeisiin `(ActionListener)`. Sovellus asettaa jokaiseen painikkeeseen `(Button)` kuuntelijan, ja pelaajan valitessa jonkun näistä, tapahtuma tarkastetaan `actionPerformed()`-metodista. Metodi sisältää kutsun kaikkien painikkeiden toimintoihin.

Koin työn hyväksi harjoitukseksi, joten suoritin kaksi ylimääräistä asiaa tietovisasovellukselle. Ensin suoritin ystäväpiirissä käytettävyystestauksen, jonka avulla muutin ohjelmaa siten, ettei siinä ole enää epäselviä tilanteita käyttäjän kannalta. Toiseksi suoritin ohjelmalle uudelleenpelattavuuden, eli tietovisan voi käynnistää uudelleen pelattavaksi "Pelaa uudelleen" napista `(Button)`.

3 Ohjelman ja sen osien kuvaaminen

3.1 Luokka GUI

Luokka sisältää kaiken tarpeellisen tietovisan toiminnalle.

3.1.1 Ilmentymämuuttujat

```
/*
           Harjoitustyö (TKO_2023) Kevät 2008

Tämä sovellus esittää interaktiivisen tietovisan. Sovelluksen on
tarkoitus pyöriä html sivuston seassa, täten sen toteutus on Appletti,
jonka luokka perii.

ActionListener toteutetaan myös, jotta graafisen käyttöliittymän
nappien painallukset havaittaisiin. Nappien panallus johtaa kutsun
metodille actionPerformed, joka selvittää jatkon mukaisen toiminnan.

Sovellus lukee kysymykset erilliseltä tiedostolta (luettavaTiedosto)
ja esittää ne pelin edetessä aina samassa järjestyksessä. Kysymyksiä
ei sekoiteta. Vastaukset kuuluvat (versio 1.1) jälkeen
kysymystiedostoon, erotettuna kysymyksestä merkillä (;) ->
(True/False) ohjaajan vaatimusten mukaisesti. Vastaukset eivät siis
enää ole "kovakoodattuja", eli kysymyksen ja vastauksen voi helposti
vaihtaa tietovisaan muuttamalla luettavaTiedosto (.txt) haluamalla
tavalla.

Versio 1.2: Koodin ulkoasun muotoilua ja ilmentymämuuttujien
            suojausmääreen lisäys. actionPerformed() metodin
            siistiminen siten, että itse toiminnot ovat omissa
            metodeissaan ja kyseinen metodi sisältää vain kutsun
            niihin. actionPerformed() metodin kutsuttavien metodien
            uudelleen nimeäminen, ja uusien metodien siirto omaan
            "rypäleeseen".

Tekijä: Jyri Lehtonen (72039), 28.1.2008
Versio: 1.1 (31.1.2008)
Versio: 1.2 (8.2.2008)
*/

public class GUI extends Applet implements ActionListener {

   //Graafisen käyttöliittymän komponentit.
     private Button okNappi;
     private Button nextNappi;
     private Button pisteetNappi;
     private Button aloitaNappi;
```

```
    private Button ohjeetNappi;
    private Button bootNappi;
    private Button statistic;
    private TextArea kysymysTila;
    private TextArea finaali;
    private TextField palauteTila;
    private CheckboxGroup valintaRyhmä;
     private Checkbox ympyräValinta1;
     private Checkbox ympyräValinta2;

  //Tietovisan kysymykset haetaan tiedostosta...
    private String luettavaTiedosto = "Kysymykset.txt";
    private StringBuffer strBuff;

  //Tietovisan pelitekniikkaan liittyvät komponentit.
    private int vipu=0;
    private int pisteet= 0;
    private int oikein=0;
    private int väärin=0;
    private int peliloppu=0;

  //Tietovisan käyttämät "tietopankit".
    private ArrayList<String> kysymysLista = new ArrayList();
    private ArrayList<String> vastausLista = new ArrayList();

  //Kuuntelijan vastausListan vaihtoehdot
    private String oikea="True";
    private String väärä="False";
```

3.1.2 Appletin alustaja

```
/*********************************************************************
* Tämä metodi ajaa appletissa itsensä, ja alustaa
* ohjelman käyntiin, luoden graafisen käyttöliittymän.
*          @.pre = selain tukee Appletteja ja riittävä Java versio.
*          @.post = html sivulla on alustettu GUI tietovisalle.
*********************************************************************/

     public void init() {

      //Graafisen käyttöliittymän alustukset.
        setLayout(null);
        okNappi = new Button("Vastaa");
        nextNappi = new Button("Seuraava");
        pisteetNappi = new Button("Pisteet");
        aloitaNappi = new Button("Aloita");
        ohjeetNappi = new Button("Ohjeet");
        bootNappi = new Button("Pelaa uudelleen");
        statistic = new Button("Tilastot");
        kysymysTila = new TextArea(100,100);
        finaali= new TextArea(100,100);
        palauteTila = new TextField("Initialisoitu...",100);
          valintaRyhmä = new CheckboxGroup();
          ympyräValinta1 = new Checkbox("Kyllä", valintaRyhmä, false);
          ympyräValinta2 = new Checkbox("Ei",    valintaRyhmä, false);
```

```
        //Graafisen käyttöliittmän sijoitukset.
          okNappi.setBounds(260,325,100,40);
          nextNappi.setBounds(150,325,100,40);
          pisteetNappi.setBounds(372,325,100,40);
          aloitaNappi.setBounds(10,175,125,75);
          ohjeetNappi.setBounds(10,262,125,75);
          bootNappi.setBounds(260,355,100,40);
          statistic.setBounds(60,325,100,40);
          kysymysTila.setBounds(75,50,475,100);
          finaali.setBounds(170,265,325,80);
          ympyräValinta1.setBounds(180,120,100,100);
          ympyräValinta2.setBounds(360,120,100,100);
          palauteTila.setBounds(170,250,260,60);

           //Käynnistyksen yhteydessä olevat valinnat.
            add(aloitaNappi);
            add(ohjeetNappi);
            add(kysymysTila);
            add(palauteTila);

           //Poistetaan pelaajan mahdollisuus muuttaa tekstejä.
            kysymysTila.setEditable(false);
            palauteTila.setEditable(false);
            finaali.setEditable(false);

           //Lisätään kuuntelijat Buttoneihin.
            okNappi.addActionListener(this);
            nextNappi.addActionListener(this);
            pisteetNappi.addActionListener(this);
            aloitaNappi.addActionListener(this);
            ohjeetNappi.addActionListener(this);
            bootNappi.addActionListener(this);
            statistic.addActionListener(this);


           setBackground(Color.lightGray);
           vastausLista.add(null);
           }
```

3.1.3 Appletin käynnistäjä

```
/**********************************************************************
* Appletin start metodi, joka käynnistuu alustusmetodin jälkeen.
*          @.pre = init() on suoritettu.
*          @.post = käynnistää kysymyksien lukemisen tiedostosta.
**********************************************************************/

public void start() {
palauteTila.setText("Initialisoitu... Käynnissä...");
kysymysTila.setText("Tervetuloa Javan applet tietovisaan!" + "\n");
       String prHtml = this.getParameter("luettavaTiedosto");
           if (prHtml != null) luettavaTiedosto = new String(prHtml);

}
```

3.1.4 Kuuntelijan toiminnot

```
*********************************************************************
* Kuuntelijan toiminnot (kaikki GUI:n buttonit)
*          @.pre = init() on suoritettu.
*          @.post = okNappi: varmistaa että kysymys on oikein,
*                     ja asettaa pelitekniikan ohjeiden mukaiseksi.
*                 aloitaNappi: Käynnistää pelin.
*                 nextNappi: siirtyy suoraan seuraavaan kysymykseen.
*                 pisteetNappi: palauterudussa näkyy pelaajan pisteet.
*                 ohjeetNappi: kysymysryydyssa näkyy pelin ohjeet.
*                 bootNappi: alustaa pelin uudelle kierrokselle.
*                 statistic: näyttää kierroksen tilastolliset tekijät.
*********************************************************************/

      public void actionPerformed(ActionEvent evt) {
            if (evt.getSource() == okNappi) {
               okNappiToiminnot();
            }

            if(evt.getSource() == aloitaNappi) {
               aloitaNappiToiminnot();
            }

            if(evt.getSource() == nextNappi) {
               nextNappiToiminnot();
            }

            if(evt.getSource() == pisteetNappi) {
               pisteetNappiToiminnot();
            }

            if(evt.getSource() == ohjeetNappi) {
               ohjeetNappiToiminnot();
            }

            if(evt.getSource() == bootNappi) {
               buuttaaNappiToiminnot();
            }

            if(evt.getSource() == statistic) {
               tilastoNappiToiminnot();
            }

      }
```

3.1.5 Kysymysten lukeminen tiedostosta

```
/*********************************************************************
 * Kysymysten lukeminen tiedostosta,
 * lataa ennaltamäärätyn tiedoston sisällön riveittän
 * kysymysLista ArrayListin indekseihin.
 *          @.pre = (luettavaTiedosto != null) & (kysymysTila != null)
 *          @.post = kysymysLista (ArrayList<String>) sisältää
 *                   tiedoston kysymykset.
 *********************************************************************/
```

```
public void lueTiedosto() {
           String rivi;
           URL url = null;

    try {
           url = new URL(getCodeBase(), luettavaTiedosto); }

    catch(MalformedURLException e){ }


    try {
           InputStream in = url.openStream();
           BufferedReader bf = new BufferedReader(
                                          new InputStreamReader(in));
           strBuff = new StringBuffer();

           for(int i=1; i < 16; i++){
              String merkki=";";

              rivi = bf.readLine();
              strBuff.append(rivi + "\n");

              int erotin = rivi.indexOf(merkki);
              String kysymys = rivi.substring(0,erotin);
              strBuff.append(kysymys + "\n");
              kysymysLista.add(kysymys);

              String vastaus = rivi.substring(erotin+1,rivi.length());
              strBuff.append(vastaus);
              vastausLista.add(vastaus);
        }


kysymysTila.append("Ladattu tiedosto: " +
                                luettavaTiedosto + "\n");
    }

    catch(IOException e){
      e.printStackTrace();
    }
  }
```

3.1.6 Uuden kysymyksen esittäminen

```
/*****************************************************************
 * Esittää uuden kysymyksen.
 *         @.pre = (kysymysLista =! null) & (kysymysTila != null)
 *         @.post = Seuraava kysymys on näkyvissä kysymysTila:ssa.
 *
 *****************************************************************/

 public void uusiKysymys() {
           kysymysTila.append((kysymysLista.get(vipu)).toString());
           vipu=vipu+1;
 }
```

3.1.7 Nappien toiminnot

```
/*****************************************************************
  * actionPerformed() metodin kutsuttavat metodit.
  * Mitä tapahtuu, kun valitaan jokin napeista,
  * joihin on liitetty kuuntelija (listener) joka
  * kutsuu actionPerformed metodia.
  *                   @.pre = vaadittavaa nappia on painettu.
  *                   @.post = nappi on toteuttanut toimintansa.
  *****************************************************************/

void okNappiToiminnot() {
           if ((ympyräValinta1.getState() == true) &&
            (ympyräValinta2.getState() == false) &&
            (vastausLista.get(vipu).equals(oikea))) {
                     palauteTila.setText("Oikein vastattu!");
                     pisteet=pisteet+1;
                     oikein=oikein+1;
                     remove(okNappi);
                     add(nextNappi);
         }
                     if(vipu==15) {
                     remove(okNappi);
                     remove(nextNappi);
                     add(statistic);
                     puhdistaKysymys();
                     palauteTila.setText("Peli on päättynyt!");
                               }

          if ((ympyräValinta1.getState() == false) &&
              (ympyräValinta2.getState() == true) &&
              (vastausLista.get(vipu).equals(oikea))) {
                     palauteTila.setText("Väärin vastattu!");
                     pisteet=pisteet-1;
                     väärin=väärin+1;
                     remove(okNappi);
                     add(nextNappi);
          }

           if ((ympyräValinta1.getState() == true) &&
               (ympyräValinta2.getState() == false) &&
               (vastausLista.get(vipu).equals(väärä))) {
                     palauteTila.setText("Väärin vastattu!");
                     pisteet=pisteet-1;
                     väärin=väärin+1;
                     remove(okNappi);
                     add(nextNappi);
           }

            if ((ympyräValinta1.getState() == false) &&
                (ympyräValinta2.getState() == true) &&
                (vastausLista.get(vipu).equals(väärä))) {
                     palauteTila.setText("Oikein vastattu!");
                     pisteet=pisteet+1;
                     oikein=oikein+1;
                     remove(okNappi);
                     add(nextNappi);
            }
```

```
                if(vipu==15) {
                remove(okNappi);
                remove(nextNappi);
                add(statistic);
                puhdistaKysymys();
                palauteTila.setText("Peli on päättynyt!");
                }

        if ((ympyräValinta1.getState() == false) &&
            (ympyräValinta2.getState() == false)) {
        palauteTila.setText("Valitse kyllä tai ei vastaukseksi.");
        }

}

        void nextNappiToiminnot() {
                puhdistaKentät();
                uusiKysymys();
                add(okNappi);
                remove(nextNappi);
        }

        void tilastoNappiToiminnot() {
                remove(statistic);
                remove(kysymysTila);
                remove(palauteTila);
                remove(pisteetNappi);
                remove(ympyräValinta1);
                remove(ympyräValinta2);
                add(finaali);
                add(bootNappi);
                peliloppu=1;
        }

        void pisteetNappiToiminnot() {
        palauteTila.setText("Pisteesi: " + pisteet +
        (" (oikein:")+oikein+(" väärin:")+väärin+")");
        }

        void ohjeetNappiToiminnot() {
        kysymysTila.setText("Ohjeet:" +("\n")+
        "Valitse Aloita pelin käynnistymiseksi." +("\n")+
        "Ensimmäinen kysymys tulee automaattisesti eteen,"+("\n")+
        "tämän jälkeen valitse vastaukseksi kyllä tai ei,"+("\n")+
        "ja paina Vastaa nappia. Sovellus kertoo miten"+("\n")+
        "pärjäsit."+("\n")+
        "Seuraava napilla saa uuden kysymyksen. "+("\n")+
        "Pisteet napista näkee painalluksen aikaisen
        pistetilanteen."+("\n")+
        "Pisteytys:"+("\n")+
        "Oikeasta vastauksesta saa 1:n pisteen."+("\n")+
        "Väärästä vastauksesta menettää 1:n pisteen."+("\n"));
        }
```

```
        void aloitaNappiToiminnot() {
                  puhdistaKysymys();
                  add(okNappi);
                  add(pisteetNappi);
                  palauteTila.setText("Peli on aloitettu!");
                  lueTiedosto();
                  uusiKysymys();
                  remove(aloitaNappi);
                  remove(ohjeetNappi);
                  add(ympyräValinta1);
                  add(ympyräValinta2);
        }

        //Asettaa pelin alkutilaan.
        void buuttaaNappiToiminnot() {
                  add(aloitaNappi);
                  add(ohjeetNappi);
                  add(kysymysTila);
                  add(palauteTila);
                  add(ympyräValinta1);
                  add(ympyräValinta2);
                  remove(pisteetNappi);
                  remove(statistic);
                  remove(finaali);
                  tyhjennäPisteet();
                  peliloppu=0;
                  remove(ympyräValinta1);
                  remove(ympyräValinta2);
                  kysymysTila.setText("Tervetuloa Javan
                            applet tietovisaan!" + "\n");
                  palauteTila.setText("Käynnistit tietovisan
                             uudelleen!");
    }
```

3.1.8 Pelitekniikkaan liittyvät metodit

```
/*****************************************************************
 * Muut lyhyet metodit jotka liittyvät tietovisaan.
 *****************************************************************/

  //Tyhjentää kysymys- ja palautetilaan.
     void puhdistaKentät() {
    palauteTila.setText("");
    kysymysTila.setText("");
   }

   //Tyhjentää vain kysymystilaan.
     void puhdistaKysymys() {
           kysymysTila.setText("");
    }
```

```
    //Kierroksen päätyttyä, näyttää tilastoja katsellessa
    //lyhyen kommentin suorituksesta, ja tulokset.
    void näytäFinaali() {
           if(pisteet==15){
            finaali.setText(("Täydellinen suoritus!")+("\n")+
                     ("Pisteesi: ") +pisteet+("\n")+
                     ("Oikein vastatut: ")+oikein+("\n")+
                     ("Väärin vastatut: ")+väärin+("\n"));
            }

            if((pisteet > 8) && (pisteet != 15)){
            finaali.setText(("Ihan hyvin osattu!")+("\n")+
                     ("Pisteesi: ") +pisteet+("\n")+
                     ("Oikein vastatut: ")+oikein+("\n")+
                     ("Väärin vastatut: ")+väärin+("\n"));

            }

            if(pisteet <= 8){
            finaali.setText(("Tiedoissasi on aukkoja...")+("\n")+
                     ("Pisteesi: ") +pisteet+("\n")+
                     ("Oikein vastatut: ")+oikein+("\n")+
                     ("Väärin vastatut: ")+väärin+("\n"));

            }

    }


    //Tyhjentää pelitekniset numeraaliset muuttujat.
    void tyhjennäPisteet() {
           vipu=0;
           oikein=0;
           väärin=0;
           pisteet=0;
           remove(bootNappi);
    }
```

3.1.9 Tilaston piirtämieen liittyvät metodit

```
/*****************************************************************
 * Tilaston piirtämiseen liittyvät metodit
 * @.pre = pystyy piirtämään grafiikkaa.
 * @.post = Tilastot pelaajan suorituksesta näkyy graafisesti.
 *****************************************************************/

  //Pirtämisiin liittyvä pakollinen metodi.
  public void paint(Graphics g) {
           annaTilastot(g);
  }

  //Jos havaitsee kierroksen olevan ohi, piirtää menestyksen
  //tietovisassa graafisesti taustalle.
  public void annaTilastot(Graphics g) {

    Graphics2D g2 = (Graphics2D) g;
    g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
        RenderingHints.VALUE_ANTIALIAS_ON);
```

```
                    if (peliloppu==1) {

    g2.setPaint(Color.red);
    g2.fill(new Rectangle2D.Double(100, 100, 70, oikein*10));
    g2.setPaint(Color.black);
    g2.drawString("Oikein vastattu", 100, 80);

    g2.setPaint(Color.red);
    g2.fill(new Rectangle2D.Double(280, 100, 70, väärin*10));
    g2.setPaint(Color.black);
    g2.drawString("Väärin vastattu", 280, 80);

    g2.setPaint(Color.red);
    g2.fill(new Rectangle2D.Double(460, 100, 70, pisteet*10));
    g2.setPaint(Color.black);
    g2.drawString("Pisteet", 460, 80);


    g2.setPaint(Color.gray);
    g2.draw(new Line2D.Double(100, 100, 525, 100));
    g2.drawString("0", 80, 100);

    g2.setPaint(Color.gray);
    g2.draw(new Line2D.Double(100, 150, 525, 150));
    g2.drawString("5", 80, 150);

    g2.setPaint(Color.gray);
    g2.draw(new Line2D.Double(100, 200, 525, 200));
    g2.drawString("10", 80, 200);

    g2.setPaint(Color.gray);
    g2.draw(new Line2D.Double(100, 250, 525, 250));
    g2.drawString("15", 80, 250);

    näytäFinaali();
           }

    else{setLayout(null);}

 }


} //end of GUI.class
```

4 Testausjärjestelyt

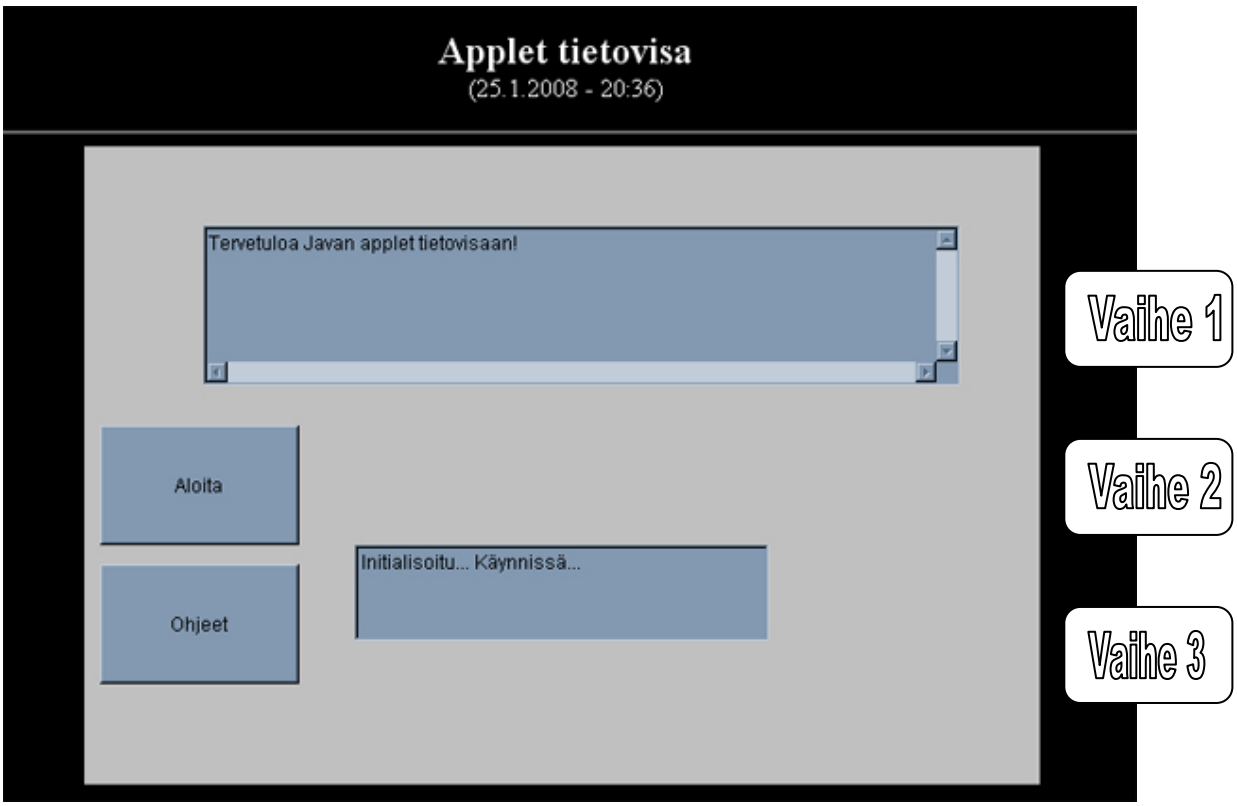
4.1 GUI-luokan testaus.

Testaussuunnitelma

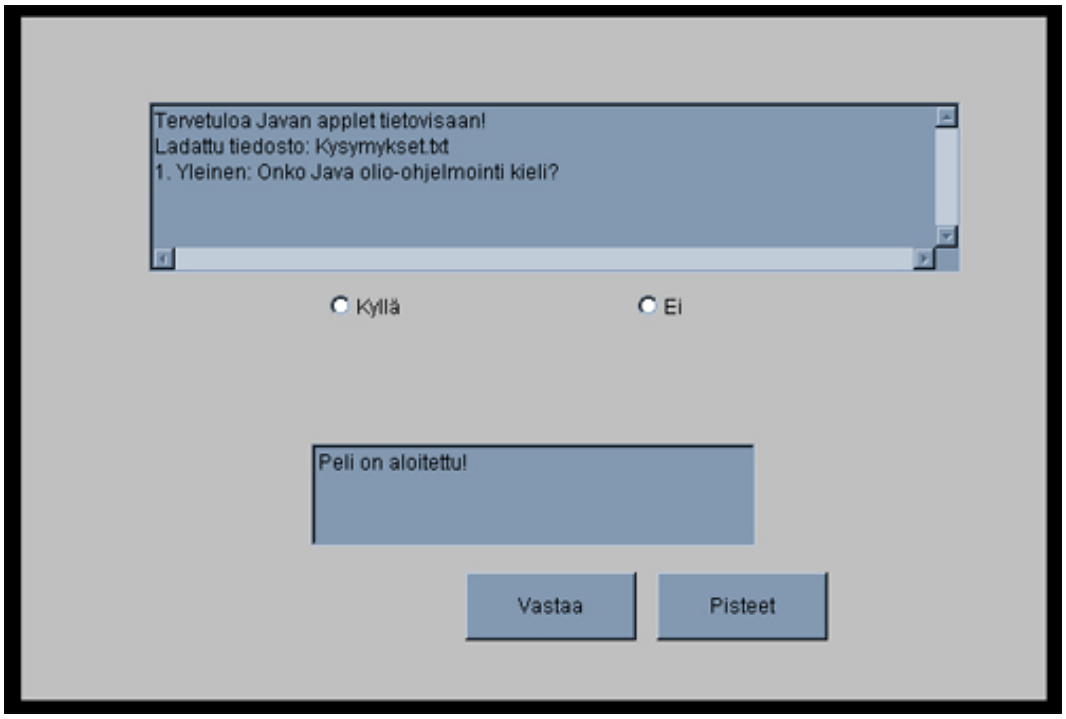
1. Vaihe: alustuksen testaus
2. Vaihe: käynnistymisen testaus
3. Vaihe: graafisen käyttöliittymän testaus
4. Vaihe: kuuntelijoiden testaus
5. Vaihe: kysymyksien testaus
6. Vaihe: pelitekniikan testaus
7. Vaihe: tilastojen testaus
8. Vaihe: Versio 1.2 kokonaisuuden testaus.

Havainnot

Testattaessa ruudulle tulostuu seuraavaa (vaiheet on merkitty kuvan viereen):



Ensimmäisessä vaiheessa, jotta tietovisa alustuisi (`init()`), tulisi palauteruudussa lukevan "Initialisoitu… ". Seuraavaksi applettien toiminnassa käsitellään `start()` metodi, joka ilmaisee toimintansa "Käynnissä…" tekstillä palauteruudussa. Kolmais vaihe on graafisen käyttöliittymän testaus. GUI:n testaus tulee jatkumaan koko sovelluksen ajan. Havaitaan testauksen etenevän suunnitelman mukaisesti kolmen ensimmäisen vaiheen jälkeen.



Vaihe 4

Vaihe 5

Vaihe 6

Neljännessä vaiheessa tarkistetaan kuuntelijoiden toiminta. Tämä tulee jatkumaan graafisen käyttöliittymän tavoin koko sovelluksen testauksen ajan. Graafinen käyttöliittymä on oikein sijoittunut ja käynnistyksen yhteydessä olevat valinnat toimivat kuten pitikin. Myös itse tietovisan käynnistämisen jälkeen valinnat toimivat ja tekevät tehtävänsä oikein.

Viidennessä vaiheessa keskitytään tietovisan kulkuun. Tarkastetaan lukeeko sovellus kysymykset ja vastaukset, ja osaako se yhdistää ne toisiinsa. Sovellus löysi kysymykset ja vastaukset oikein. Kuuntelijat toimivat oikein ja pelaaja sai palautetta toiminnoistaan.

Toiseksi viimeinen vaihe keskittyy tarkastamaan, että sovellus pyörii edellä olevien havaintojen mukaisesti koko suorituksen ajan. Ongelmia ei havaittu, peli päättyi ja testaus etenee suunnitelman mukaisesti.

Viimeisessä vaiheessa tarkastetaan graafisesti piirrettyjen tilastojen toiminta. Tilastot piirtyvät pelaajan menestyksen mukaan. Testaus on suoritettu ja ongelmia ei havaittu.

Vaihe 7

Uuden version testaus sujui ongelmitta. Testaus oli kokonaisvaltainen. Muutettu lähdekoodi kääntyi ongelmitta. Suoritin testauksen vaiheet 1-7 uudestaan tässä vaiheessa, eikä ongelmia ilmennyt. Päivitys onnistui.

5 Ohjelman käyttöohje

Ohjelma käynnistetään siirtymällä oikealle URL osoitteelle jossa sovellus sijaitsee. HTML sivusto käynnistää tietovisan suoraan ilman mitään valintoja, vaatimuksena on tarpeeksi uusi Java versio mikä tukee Appletteja ja Appletit on oltava päällä selaimessa. Peliteknisiä käyttöohjeita saa tietovisan käynnistyttyä valitsemalla valinnan "Ohjeet".

Osoite jossa ohjelma sijaitsee harjoitustyö kurssin aikana, ja samalla verkkotoiminnan testaus, on seuraava:

http://www.geocities.com/ascandancy/LueMinut.html