Xetor

Tietotekniikan sovellusprojekti

Jani Lirkki
Olavi Paananen
Raimo Pitkänen
Jussi Räisänen

Sovellussuunnitelma
5. huhtikuuta 2004
Versio 1.0

Jyväskylän yliopisto
Tietotekniikan laitos

Xetor-projektin tietoja

Tekijät: Jani Lirkki (jaallirk@cc.jyu.fi)
Olavi Paananen (laolpaan@cc.jyu.fi)
Raimo Pitkänen (rapitkan@cc.jyu.fi)
Jussi Räisänen (jtraisan@cc.jyu.fi)

Yhteystiedot: Työtila AgC224.1, puhelinnumero (014) 260 4967, sähköpostilistan osoite on xetor@korppi.jyu.fi

Työn nimi: Sovellussuunnitelma, Xetor-projekti

Työ: Sovellussuunnitelma tietotekniikan sovellusprojektiin.

Tiivistelmä: Tämä dokumentti on Jyväskylän yliopistossa keväällä 2004 toteutettavan Xetor-projektin sovellussuunnitelma. Dokumentissa kuvataan, kuinka sovellus toteutetaan. Lisäksi käydään läpi ohjelmointiin liittyviä käytäntöjä ja sovelluksen sisäistä rakennetta sekä yhteyksiä muihin ohjelmiin ja tiedostoihin.

Avainsanat: XML, editori, Java, puunäkymä, tekstinäkymä, esikatselunäkymä, elementti, attribuutti, käyttöliittymä

Dokumentin versiohistoria

Versio	Päiväys	Tehnyt	Muutokset
0.1	15.03.2004	JR,RP,JL	
0.2	22.03.2004	JR,RP,JL,OP	Tarkennettu kuvauksia valikoista ja käyttöliittymästä
0.3	25.03.2004	JL,OP,RP,JR	Dokumentin ulkoasu tarkastettu huolellisesti ja kuvat päivitetty ajan tasalle
0.4	31.03.2004	JL,OP,RP,JR	Dokumentti muotoiltu katselmoitavaan kuntoon
1.0	02.04.2004	JL,OP,JR,RP	Katselmoinnissa havaitut virheet korjattu

Tekijöiden lyhenteet

JL: Jani Lirkki
OP: Olavi Paananen
RP: Raimo Pitkänen
JR: Jussi Räisänen

Sisältö

1 Johdanto

Xetor on Jyväskylän yliopiston tietotekniikan laitoksen kevään 2004 sovellusprojekti. Projekti toteuttaa informaatioteknologian tiedekunnalle XML-editorin, jossa XML-dokumentin muokkaus on mahdollista puunäkymän ja tekstinäkymän kautta. Kevään 2004 Xetor-projektiryhmään kuuluvat tietotekniikan opiskelijat Jani Lirkki, Olavi Paananen, Raimo Pitkänen ja Jussi Räisänen. Tilaajana toimii Jyväskylän yliopiston informaatioteknologian tiedekunta.

Xetor XML-editori toteutetaan tämän suunnitelman pohjalta. Tavoitteena on antaa niin selkeä kuva sovelluksen toiminnasta ja rakenteesta sekä yhteyksistä muihin sovelluksiin ja tiedostoihin, että Java-ohjelmointiin perehtynyt lukija pystyisi ohjelmoimaan sovelluksen, vaikka tietenkään suunnitelmassa ei pystytä kuvaamaan aivan kaikkia yksityiskohtia. Tässä dokumentissa ei käsitellä erikseen sovellukselle asetettuja tavoitteita tai käyttötapauksia, sillä ne on listattu vaatimusmäärittelyssä.

Luvussa 2 esitellään aiheeseen oleellisesti liittyviä termejä. Sovelluksen rakenne ja luokkakaaviot esitetään luvussa 3. Käyttöliittymän ulkoasua, valikkoja ja toimintoja käydään puolestaan läpi luvussa 4. Lähdekoodin ulkoasuun liittyvistä asioista kuten kommentointityylistä ja nimeämiskäytännöstä kerrotaan luvussa 5. Sovelluksen testaukseen liittyvät asiat kuvataan luvussa 6. Kappaleessa 7 kerrotaan sovellukseen liittyvistä tiedostoista ja kappaleessa 8 on lyhyt yhteenveto tästä sovellussuunnitelmasta. Lisäksi lopussa on liitteenä esimerkit mahdollisista kieli- ja asetustiedostoista sekä kuva ohjelman kaikista valikoista.

2 Termejä

Tässä luvussa on kuvattu projektin aihealueeseen liittyviä termejä. Termit on jaoteltu aihepiirin yleisiin termeihin ja sovellukseen liittyviin käsitteisiin.

2.1 Aihepiirin yleisiä termejä

Attribuutti on elementin ominaisuus, joka tarkentaa elementin tilaa.

CSS eli *Cascading Style Sheets* on XML-dokumenttien ulkoasua kuvaava kieli. [1]

DOM eli *Document Object Model*, on alusta- ja kieliriippumaton oliorajapinta, joka antaa skriptien ja ohjelmien päästä käsiksi dokumentin sisältöön, rakenteeseen ja tyyliin. [2]

DTD on XML-kielissä käytettävä dokumenttityypin määrittelytiedosto. [8]

Elementti on osa, josta XML-dokumentit koostuvat. Elementit voidaan määritellä DTD:ssä.

Gecko on alustariippumaton selainmoottori, joka on toteutettu osana Mozilla-projektia. [5]

HTML eli *Hypertext Markup Language* on standardi merkintäkieli, jolla kuvaillaan www-sivujen sisällön rakenne. [3]

Hyvin muodostettu on käsite, jolla kuvataan XML-dokumentin rakennetta. Dokumenttia kutsutaan hyvin muodostetuksi, jos sen rakenne on XML-spesifikaation minimisääntöjen mukainen. [8]

ISO-8859-1 on merkistö, joka sisältää ASCII-merkistöön kuuluvien normaalien aakkosten, numeroiden ja yleisimpien välimerkkien lisäksi useimmat länsi- ja pohjoiseurooppalaisten kielten tarvitsemat aksentoidut merkit.

ISO-8859-15 on ISO-8859-standardiin kuuluva merkistö, joka on tarkoitettu käyttöön Euroopan alueelle. Se on muokattu ISO-8859-1:n pohjalta ja siihen on esimerkiksi lisätty euro-merkki.

Java on Sunin kehittämä laitteistoriippumaton olio-ohjelmointikieli. [4]

Java-pavut (engl. *JavaBeans*) ovat Java-ohjelmointikielellä luotuja komponentteja. [4]

JAXP eli *Java API for XML Processing* on XML-jäsentimien käytön mahdollistava yhtenäinen rajapinta. [4]

JDK eli *Java Development Kit* on Java-ohjelmien standardi kehitysympäristö Sun Microsystemsiltä. [4]

JDom on Java-pohjainen DOM XML-tiedostojen käsittelyyn. JDom on tehty samaan tarkoitukseen kuin DOM, mutta on helppokäyttöisempi. [11]

JRE eli *Java Runtime Environment* on apuohjelmisto, joka tarvitaan Java-ohjelmien ajamiseen. JRE pitää sisällään mm. ympäristöön sopivan Java-virtuaalikoneen. [4]

Käyttöjärjestelmä on ohjelmisto, joka ohjaa tietokonetta ja siihen kytkettyjä oheislaitteita.

Käyttötapaus (engl. *use case*) on käyttäjän tai sovelluksen toimintoa tietyn tehtävän suorittamiseksi kuvaava dokumentti.

Linux on suosittu käyttöjärjestelmä.

Mozilla on tehokas ja ilmainen standardien mukainen WWW-selain, jonka lähdekoodi on vapaasti käytettävissä. [5]

PNG *Portable Network Graphics* on W3-yhteenliittymän (W3C) vuonna 1996 määrittelemä suositus tiedostomuodoksi häviöttömälle, siirrettävälle ja tehokkaasti pakatulle rasterikuvalle.

Prosessointiohje on XML-dokumenttiin liitetty komento tai ohje, jonka XML-jäsennin välittää dokumenttia käsittelevälle sovellukselle. [9]

Selain on toiminto tai ohjelmisto, jolla selataan tietokantaa. Internet-verkossa asiakasohjelmisto, joilla selataan www-palvelimen sivuja. [6]

UTF-8 on vaihtelevan pituinen merkistön koodaustapa. Merkkikoodista riippuen yksi merkki vie tallennettuna yhdestä neljään tavua.

Validi on käsite, jolla kuvataan XML-dokumentin DTD:n mukaisuutta. Dokumenttia kutsutaan validiksi, jos se on muodostettu DTD:n mukaiseksi.

Windows on laajaan käyttöön levinnyt käyttöjärjestelmä Microsoftilta.

XHTML eli *Extensible Hypertext Markup Language* on XML-muotoinen WWW-dokumenttien kuvaukseen käytettävä kieli. [7]

XML eli *Extensible Markup Language* on metakieli, jolla määritellään rakenteellisia merkkauskieliä. [8]

XPath on yksinkertainen kyselykieli, jolla voidaan hakea erilaisia tietoja XML-dokumentin sisältä.

XSL eli *Extensible Stylesheet Language* on XML-pohjainen kieli, jonka avulla voidaan XML-dokumentti muuntaa toiseen XML-dokumentin muotoon tai toiseen formaattiin. [10]

2.2 Sovellukseen liittyvät käsitteet

CSS-ominaisuus	on CSS-tiedoston määre, jonka avulla voidaan määrittää XML-dokumentin ulkoasua. [1]
CSS-valitsin	sitoo CSS-ominaisuudet XML-dokumentin elementteihin. [1]
Dialogi	on ikkuna, jonka avulla käyttöliittymän ja käyttäjän välinen kommunikointi tapahtuu.
Esikatselunäkymä	näyttää XML-dokumentin ulkoasun tyylimäärityksineen.
Leikepöytä	graafisten käyttöliittymien ominaisuus, jossa tietoja (tekstiä ja grafiikkaa) voidaan viedä edelleen käsittelyä varten erilliseen käyttöliittymästä varattuun muistiin (leikepöytään) ja tarvittaessa poimia sieltä. [6]
Näkymäikkuna	on ikkuna, joka sisältää puu-, teksti- ja esikatselunäkymän dokumenttiin.
Ominaisuuseditori	(*Element inspector*)näyttää XML-elementin attribuutit ja niiden arvot. Ominaisuuseditorin kautta voi myös muokata elementin CSS-tyylimäärityksiä.
Puunäkymä	(*Tree view*) on hierarkkinen näkymä, jossa voidaan havainnollisesti muokata XML-dokumentin rakennetta.
Tekstidokumentti	eli tekstitiedosto on tiedosto, jossa on tekstiä (kirjaimia, numeroita ja symboleita), mutta ei muotoilukoodeja. Se voi olla ASCII-tiedosto, jonka useimmat tietokoneet pystyvät lukemaan. [6]
Tekstinäkymä	(*Text view*) näyttää dokumentin tekstisisällön.
Validointi	on operaatio, jossa tarkistetaan, että XML-dokumentti on hyvinmuodostettu ja validi.
Virhealue	on rajattu alue tekstinäkymästä, jossa virheilmoitukset esitetään validoinnin epäonnistuessa.
XML-dokumentti	on dokumentti, jossa tieto esitetään XML-muodossa. [8]

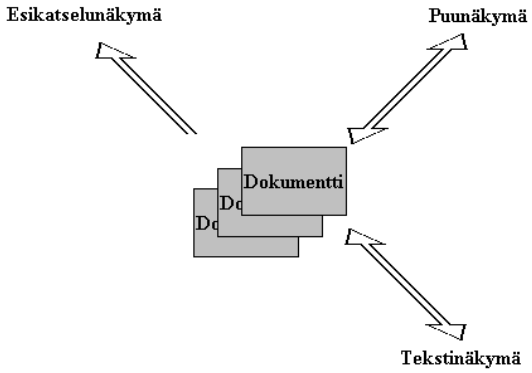
3 Rakenne

Tässä luvussa esitellään sovelluksen yleisrakennetta ja luokkakaaviota. Lisäksi kuvataan luokkien vastuualueet.

3.1 Yleisrakenne

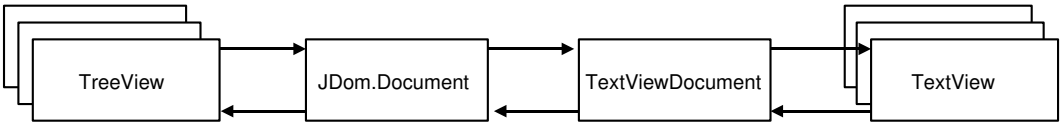
Tässä luvussa esitetään rakennekaavioilla sovelluksen yleisrakenne ja sen kehitys. Lisäksi perustellaan miksi lopulliseen ratkaisuun päädyttiin.

Alkuperäinen idea oli, että puu- ja tekstinäkymä muokkaisivat suoraan samaa `JDom.Document`-oliota. Tämän toteutuksen ongelmaksi osoittautui kuitenkin se, että `JDom.Document`-oliolla voi käsitellä vain rakenteisia dokumentteja, mutta sovelluksella pitää pystyä käsittelemään dokumentteja riippumatta siitä, ovatko ne rakenteisia.



Kuva 3.1: Alkuperäinen idea näkymien suhteesta dokumenttiin.

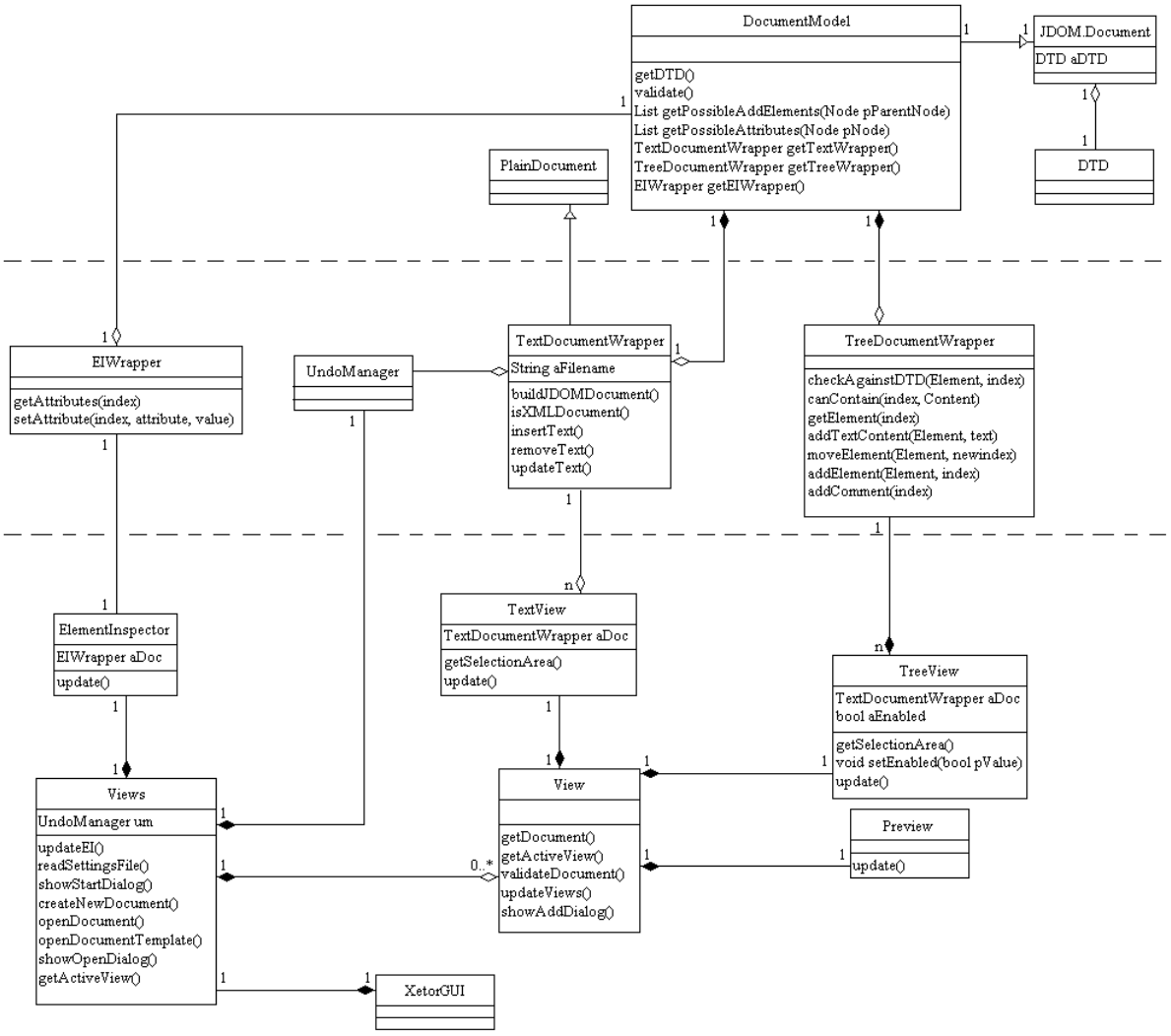
Edellä esitettyjen ongelmien vuoksi päädyttiin kuvan 3.2 esittämään rakenteeseen. Tässä ratkaisussa `TextView`-komponentti kommunikoi oman `TextViewDocument`-komponentin kanssa ja `TextViewDocument`-komponentti hoitaa kommunikoinnin `JDom.Document`-olion kanssa. `TreeView`-komponentti pystyy muokkaamaan `JDom.Document`-oliota suoraan rakenteisesti.



Kuva 3.2: Dokumenttia muokkaavat näkymät.

3.2 Luokkakaavio

Kuvassa 3.3 on esitetty sovelluksen UML-luokkakaavio. Luokkakaavio on jaettu katkoviivoilla kolmeen osaan, jotka esittävät kolmitasoarkkitehtuurin kerroksia. Luokkakaavion luokat on siis jaettu alhaalla oleviin käyttöliittymäluokkiin, keskellä oleviin ohjausluokkiin ja ylhäällä oleviin liiketoimintaluokkiin. Tarkemmin luokkien toiminnasta on selitetty luvussa 3.3.



Kuva 3.3: UML-luokkakaavio.

3.3 Luokkien kuvaukset

Tässä luvussa esitetään luokkakaaviossa esitettyjen luokkien vastuualueet.

3.3.1 Käyttöliittymäkomponentit

XetorGUI

`XetorGUI` on sovelluksen pääluokka ja se sisältää sovelluksen `main`-metodin. `Main`-metodin tarkoituksena on luoda sovelluksen pääikkuna.

Views

`Views`-luokan tärkeimpänä tehtävänä on `view`-olioiden ja `ElementInspector`-olion hallinta sekä viestinvälityksen hoitaminen käyttöliitymän eri osien välillä. Lisäksi `Views`-luokka sisältää koko sovellukselle yhteisen `UndoManager`-olion.

View

`View`-luokka on se osa käyttöliittymää, jossa näytetään muokattavan dokumentin sisältö. `View`-luokka koostuu puu-, teksti- ja esikatselunäkymistä. `View`-luokka tietää, mitä dokumenttia se muokkaa. Mikäli luokan muodostajalle ei tuoda `DocumentModel`-oliota parametrina, muodostaja luo uuden `DocumentModel`-olion.

ElementInspector

`ElementInspector` on käyttöliittymän komponentti, jossa voi muokata valitun elementin attribuutteja ja CSS-tyylimäärityksiä. Tarkemmin tämän luokan toiminta on selitetty kappaleessa 4.9.

TextView

`TextView` on käyttöliittymän komponentti, jossa voi muokata dokumenttia tekstimuotoisena. Tarkemmin tämän luokan toiminta on selitetty kappaleessa 4.6.

TreeView

`TreeView` on käyttöliittymän komponentti, jossa dokumentti näytetään puun muodossa. Puun kautta voi myös muokata dokumenttia. Tarkemmin tämän luokan toiminta on selitetty kappaleessa 4.5.

Preview

`Preview` on käyttöliittymän komponentti jossa dokumentti näytetään tyylimäärityksien

mukaisesti muotoiltuna. Tarkemmin tämän luokan toiminta on selitetty kappaleessa 4.8.

3.3.2 Ohjausluokat

UndoManager

`Undomanager`-luokka sisältää listan sovelluksen viimeisimmistä komennoista, jotka käyttäjä voi kumota. `Undomanager` osaa tallentaa käyttäjän määrittelemän määrän komentoja. `UndoManager` osaa palauttaa sovelluksen kaikkien tietämiensä komentojen suorittamista edeltäneisiin tiloihin. `UndoManager`-luokka sisältyy Java-kielen standardikirjastoon.

EIWrapper

`EIWrapper` on rajapintaluokka, jonka kautta käyttöliittymän `ElementInspector`-luokka voi muokata ja näyttää dokumenttia. Se määrittelee vain tarpeelliset metodit, joilla dokumentista voi muokata puunäkymässä valitun elementin attribuuttien arvoja tai hakea käyttöliittymän ominaisuuseditoriin elementin kaikkien attribuuttien nimet ja arvot.

TextDocumentWrapper

`TextDocumentWrapper` on rajapintaluokka, jonka kautta käyttöliittymän `TextView`-luokka voi muokata ja näyttää dokumenttia. `TextDocumentWrapper` on peritty luokasta `PlainDocument`.

TreeDocumentWrapper

`TreeDocumentWrapper` on rajapintaluokka, jonka kautta käyttöliittymän `TreeView`-luokka voi muokata ja näyttää dokumenttia.

3.3.3 Liiketoimintaluokat

DocumentModel

`DocumentModel`-luokka on `Document`-rajapinnan toteuttava luokka, joka on peritty luokasta `PlainDocument`. Jokaisesta samaa dokumenttia esittävästä tekstinäkymästä on viite samaan `DocumentModel`-olioon. `DocumentModel` osaa ilmoittaa siinä tapahtuneista muutoksista sitä esittäville näkymille. Mikäli `DocumentModel` sisältää XML-muotoisen dokumentin, osaa se validoida sen. Se myös tietää validoinnista tulleet virheet.

DTD

DTD on tiedosto, joka pitää sisällään dokumenttityypin määrittelyn.

JDOM.Document

JDOM-kirjaston `Document`-luokka pitää sisällään DOM-muotoisen esityksen XML-muotoisesta dokumentista. `Document`-luokkaan voidaan myös liittää viittaus DTD:hen.

PlainDocument

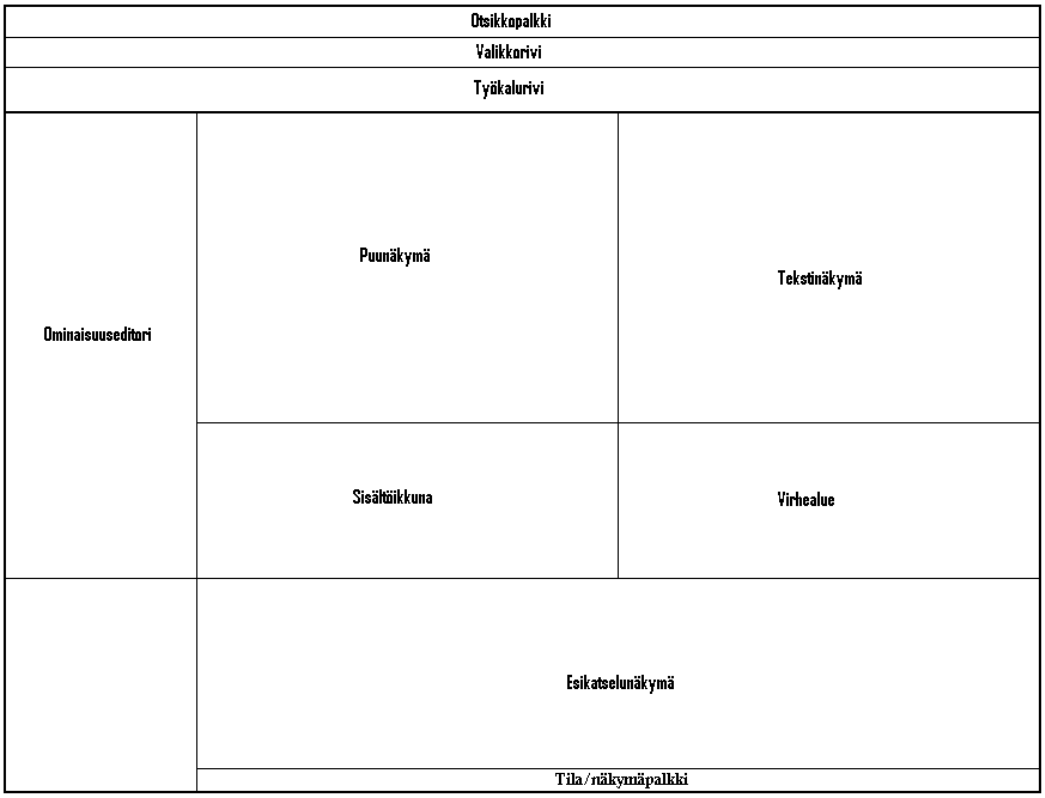
`PlainDocument` pitää sisällään tekstimuotoisen dokumentin. Tämä luokka löytyy JDK:n version 1.4.2 kirjastosta `javax.swing.text`.

4 Käyttöliittymä

Tässä luvussa kuvataan käyttöliittymän rakennetta ja toimintaa. Käyttöliittymän toiminnot ja ulkoasu selostetaan yksityiskohtaisesti. Lisäksi kerrotaan kuvien avustuksella käyttöliittymän ulkoasusta erilaisissa käyttötilanteissa.

4.1 Rakenne

Käyttöliittymän rakenne on jaettu kehyksien avulla pienempiin kokonaisuuksiin. Jokaisella kehyksellä on oma toiminnallinen tarkoituksensa. Kuvasta 4.4 näkyy käyttöliittymän rakenne.



Kuva 4.4: Käyttöliittymän rakenne.

Käyttöliittymä koostuu siis seuraavista osista: otsikkopalkki, valikkorivi, työkalurivi, ominaisuuseditori, puunäkymä, sisältöikkuna, tekstinäkymä, virhealue, esikatselunäkymä ja tila/näkymäpalkki. Seuraavissa alaluvuissa on käsitelty yksityiskohtaisesti jokaisen osan sisältämät toiminnallisuudet.

4.2 Otsikkopalkki

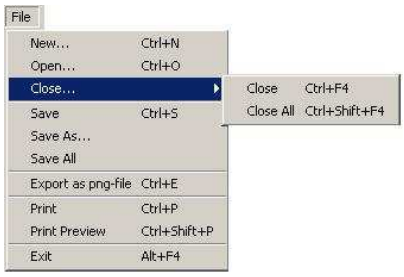
Käyttöliittymän otsikkopalkki käsittää normaalit ikkunan sulkemis-, suurennus- ja pienennystoiminnot. Lisäksi otsikkopalkissa lukee sovelluksen nimi.

4.3 Valikkorivi

Käyttäjä voi suorittaa toimintoja käyttöliittymän yläosassa olevien toimintovalikoiden kautta. Valikkorakenteesta on kerrottu seuraavissa aliluvuissa.

4.3.1 Tiedosto-valikko (File)

Tiedosto-valikko sisältää yleisiä dokumentteihin kohdistuvia toimintoja.



Kuva 4.5: Tiedosto-valikko.

Uusi (New)-toiminnon avulla käyttäjä voi luoda uuden dokumentin. Käyttäjälle avautuu dialogi, josta hän voi valita valmiin mallipohjan (kuvattu luvussa 7.1) tai tyhjän dokumentin. Valinnan jälkeen uusi dokumentti luodaan painamalla dialogilla olevaa `OK`-painiketta tai näppäimistön `Enter`-nappulaa. Käyttäjä voi myös peruuttaa dokumentin luomisen painamalla dialogin `Peruuta`-painiketta tai näppäimistön `Esc`-näppäintä.

Tallenna (Save)-toiminnon avulla käyttäjä voi tallentaa dokumentin. Dokumenttia ensi kertaa tallennettaessa käyttäjälle avautuu dialogi, johon hänen täytyy syöttää dokumentin nimi ja tallennushakemisto. Tämän jälkeen tallennus tapahtuu painamalla dialogin `OK`-painiketta tai näppäimistön `Enter`-näppäintä ja käyttäjä voi peruuttaa dokumentin tallennuksen painamalla dialogin `Peruuta`-painiketta tai näppäimistön `Esc`-näppäintä. Onnistuneen tallennuksen jälkeen dokumenttia esittävien näkymäikkunoiden tilapalkissa kerrotaan dokumentin olevan tallennettu.

Tallenna nimellä (Save as)-toiminnon avulla käyttäjä voi myös tallentaa dokumentin. Käytettäessä `Tallenna nimellä` -toimintoa käyttäjän on aina syötettävä nimi, jolla dokument-

ti tallennetaan ja tallennushakemisto. Dokumentti tallennetaan painamalla nimen kirjoittamisen jälkeen dialogilla olevaa `OK`-painiketta tai näppäimistön `Enter`-näppäintä. Käyttäjä voi myös peruuttaa dokumentin tallentamisen painamalla dialogin `Peruuta`-painiketta tai näppäimistön `Esc`-näppäintä.

Tallenna kaikki (Save all)-toiminnon avulla käyttäjä voi tallentaa kaikki avoinna olevat dokumentit. `Tallenna kaikki` vastaa siis `tallenna`-toiminnon kutsumista erikseen jokaiselle avoinna olevalle dokumentille.

Avaa (Open)-toiminnon avulla käyttäjä voi avata olemassa olevan dokumentin erillisestä tiedostonvalintadialogista. Käyttäjän on myös mahdollista avata useita dokumentteja samalla dialogilla valitsemalla avattavat dokumentit `Ctrl`-näppäin alas painettuna.

Tallenna png-kuvaksi (Export as png-file)-toiminnolla käyttäjä voi luoda puunäkymästä png-muotoisen kuvatiedoston.

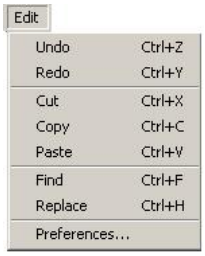
Tulosta (Print)-toiminnolla käyttäjä voi tulostaa dokumentin. Tekstidokumentin tulostamisen lisäksi käyttäjällä on mahdollisuus tulostaa puunäkymä tai puunäkymän osa.

Esikatselu (Print preview)-toiminnolla käyttäjä voi katsella mahdollisen tulostuksen ulkoasua.

Lopeta (Exit)-toiminnolla käyttäjä voi sulkea sovelluksen. Jos avoinna olevia dokumentteja on tallentamatta, käyttäjältä kysytään, haluaako hän tallentaa muutokset. Lisäksi tallennetaan asetustiedostoon kappaleen 7.3.1 mukaiset asetukset.

4.3.2 Muokkaa-valikko (Edit)

Muokkaa-valikko sisältää dokumentin muokkaamiseen liittyviä toimintoja.



Kuva 4.6: Muokkaa-valikko.

Kumoa (Undo)-toiminnon avulla käyttäjä voi kumota viimeisimmän muutoksen dokumentissa.

Toista (Redo)-toiminnon avulla käyttäjä voi kumota viimeisimmän `Kumoa`-komennon, ellei hän ole muokannut dokumenttia sen jälkeen.

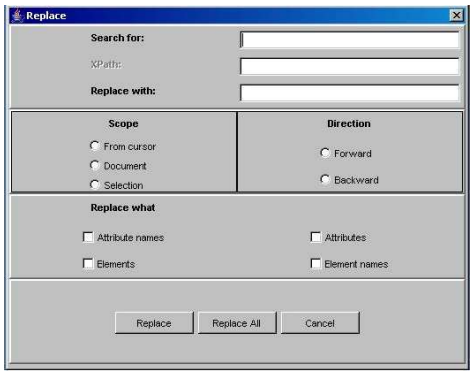
Leikkaa (Cut)-toiminnolla käyttäjä voi siirtää valitsemansa kohteen leikepöydälle. Siirron yhteydessä kohde häviää dokumentista.

Kopioi (Copy)-toiminnon avulla käyttäjä voi kopioida valitsemansa kohteen leikepöydälle. Kohde ei häviä dokumentista.

Liitä (Paste)-toiminnon avulla käyttäjä voi liittää dokumenttiin kohteen leikepöydältä. `Liitä`-toiminto säilyttää kohteen leikepöydällä.

Etsi (Find)-toiminnon avulla käyttäjä voi etsiä dokumentista merkkijonoja. Käyttäjä syöttää etsittävän merkkijonon dialogiin, valitsee valintaruuduista etsitäänkö attribuutteja, elementtejä vai niiden sisältöä ja painaa dialogissa olevaa `Etsi`-painiketta tai näppäimistön `Enter`-näppäintä. Jos etsitty kohde löytyy dokumentista, puunäkymässä etsitty elementti muuttuu valituksi ja tekstinäkymässä näkymä siirtyy dokumentin kohtaan, josta etsittävä merkkijono löytyy ja löydetty merkkijono korostetaan. Käyttäjä voi suorittaa useita hakuja peräkkäin ja sulkea dialogin, kun haut on suoritettu. Dialogin avulla käyttäjä voi suorittaa dokumenttiin myös XPath-haun.

Etsi ja korvaa (Find and replace)-toiminnon avulla käyttäjä voi etsiä ja korvata dokumenttista merkkijonon esiintymiä. Käyttäjälle avautuu erillinen dialogi, johon hän syöttää etsittävän sekä korvaavan merkkijonon ja mihin suuntaan korvaus suoritetaan. Jos käyttäjä on valinnut dokumentista jonkin alueen, kysytään myös suoritetaanko korvaus koko dokumenttiin vai valitulle alueelle. Mikäli kyseessä on XML-dokumentti, käyttäjä valitsee myös mitä seuraavista haluaa korvata: elementtien nimiä, attribuutteja, attribuuttien sisältöä vai elementtien sisältöä. Kuvassa 4.7 on esitelty Korvaa-dialogin rakenne. Etsimisen voi suorittaa dokumenttiin myös XPath-haulla.



Kuva 4.7: Korvaa dialogi.

Asetukset (Preferences)-toiminnon avulla käyttäjä voi muuttaa ohjelman käyttöasetuksia erillisestä dialogista. Muutettavat asetukset on dialogissa jaoteltu seuraaville välilehdille:

- **Yleiset (General)**-välilehdellä voi muuttaa seuraavia sovelluksen yleisiä asetuksia:
  - **Kumoushistorian pituus (Undo history length)**-kohdassa käyttäjä voi määritellä kuinka monta komentoa tallentuu kumoushistoriaan käyttäjän kumottavaksi. Pienemmällä historian koolla sovelluksen käytettävyys hitaammalla tietokoneella parantuu.
  - **Tallenna asetukset lopetettaessa (Save state on exit)**-kohdassa käyttäjä voi valita, tallennetaanko sovelluksen ulkoasuun ja käyttöön vaikuttavat asetukset lopetettaessa. Tallennetut asetukset voidaan palauttaa, kun ohjelma seuraavan kerran avataan.
  - **Etsi/Korvaa oletusasetukset (Find/Replace defaults)**-kohdassa käyttäjä voi määritellä, mihin etsi-korvaa-toiminto oletuksena vaikuttaa: elementtien nimiin, attribuutteihin, attribuuttien sisältöön vai elementtien sisältöön.
  - **Käytettävä merkistö (Default character map)**-kohdassa käyttäjä voi määritellä oletusmerkistön, jota ehdotetaan käytettäväksi uutta dokumenttia luotaessa. Tuetut merkistöt ovat UTF-8, ISO-8859-1 ja ISO-8859-15.
- **Puunäkymä (Tree view)**-välilehdellä voi muuttaa seuraavia puunäkymään vaikuttavia asetuksia:
  - **Näytä attribuutit puunäkymässä (Show attributes in tree view)**-kohdassa käyttäjä voi valita näytetäänkö puunäkymässä myös attribuutit.
  - **Pikanäppäimet (Keyboard shortcuts)**-kohdassa käyttäjä voi määritellä pikanäppäinkomentoja elementtien lisäykselle.
- **Tekstinäkymä (Text view)**-välilehdellä voi muuttaa seuraavia tekstinäkymään vaikuttavia asetuksia:
  - **Validointiväli (Validation interval)**-kohdassa käyttäjä voi määrittää aikavälin, jolloin automaattinen validointi tapahtuu. Käyttäjä syöttää kenttään arvon, joka kertoo validointiaikavälin sekunteina.
  - **Automaattinen validointi (Automatic validation)**-kohdassa käyttäjä voi valita, onko automaattinen validointi päällä. Asetukselle on oma valintaruutu.
  - **Näytä rivinumerot (Show line numbers)**-kohdassa käyttäjä voi valita, näytetäänkö tekstinäkymässä rivinumerot. Asetukselle on oma valintaruutu.
  - **Tekstin rivitys (Word wrap)**-kohdassa käyttäjä voi valita, rivitetäänkö tekstinäkymän teksti automaattisesti. Asetukselle on oma valintaruutu.

- **Värit (Colours)**-välilehdellä voi muuttaa seuraavia väreihin vaikuttavia asetuksia standardidialogilla:

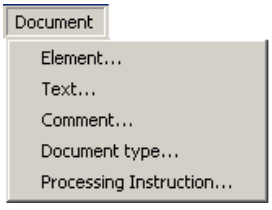
  - **Virhealueen värit (Error area colours)**-kohdassa käyttäjä voi määritellä virheikkunan virheilmoitusten ja taustan värit. Käyttäjä voi määrittää, mikä virheikkunan taustan väri on onnistuneen validoinnin jälkeen ja mikä epäonnistuneen. Värien valinta tapahtuu valitsemalla listasta kohde ja sille väri.
  - **Tekstinäkymän värit (Text view colours)**-kohdassa käyttäjä voi määritellä tekstin eri osille värit, joilla osat esitetään tekstinäkymässä. Valinta tapahtuu valitsemalla listasta tekstin osa ja sille väri.
  - **Puunäkymän värit (Tree view colours)**-kohdassa käyttäjä voi määrittää, millä väreillä puunäkymän eri osat esitetään. Värien valinta tapahtuu valitsemalla listasta kohde ja sille väri.

- **Kirjasimet (Fonts)**-välilehdellä voi muuttaa seuraavia kirjasimiin vaikuttavia asetuksia standardidialogilla:

  - **Tekstinäkymän kirjasin (Text view font)**-kohdassa käyttäjä voi määritellä tekstinäkymässä käytettävän kirjasintyylin ja sen koon.
  - **Puunäkymän kirjasin (Tree view font)**-kohdassa käyttäjä voi määritellä puunäkymässä käytettävän kirjasintyylin ja sen koon.

4.3.3 Lisää-valikko (Add)

Lisää-valikon kautta käyttäjä voi lisätä dokumenttiin elementtejä, tekstiä, kommentteja ja prosessointiohjeita. Lisäksi voidaan valita dokumentissa käytettävä DTD, jos sitä ei ole aiemmin valittu.



Kuva 4.8: Lisää-valikko.

Elementti (Element)-toiminnon avulla käyttäjä voi lisätä elementtejä dokumenttiin. Elementin lisäys voi tapahtua puu- tai tekstinäkymästä. Tekstinäkymää käytettäessä siirretään kursori oikeaan paikkaa ja lisätään elementti, joko Lisää-valikon kautta tai näppäimistöltä

painamalla `Alt+Insert`. Puunäkymässä lisättävän elementin paikka valitaan puusta hiirellä ja lisätään elementti Lisää-valikon kautta. Jos DTD on käytössä, sovellus näyttää käyttäjälle listan mahdollisista lisättävistä elementeistä. Käyttäjä valitsee lisättävän elementin hiirellä tai näppäimistöllä listasta ja painaa dialogin `OK`-painiketta. Jos lisättävällä elementillä on DTD:n mukaan attribuutteja, voi käyttäjä syöttää niille arvot erillisellä dialogilla elementin valitsemisen jälkeen.

Teksti (Text)-toiminnon avulla käyttäjä voi lisätä tekstiä elementtien sisälle. Tämä toiminto on käytössä vain dokumenttia puunäkymässä muokattaessa. Ensin käyttäjä valitsee puusta elementin, jonne haluaa tekstiä lisätä. Tekstin lisäys tapahtuu Lisää-valikon kautta. Käyttäjälle avautuu dialogi, jonne hän voi syöttää lisättävän tekstin. Hyväksyminen tapahtuu dialogin `OK`-painikkeen avulla.

Kommentti (Comment)-toiminnon avulla käyttäjä voi lisätä kommentteja dokumenttiin puunäkymästä. Puunäkymästä valitaan haluttu paikka kommentille ja lisäys tapahtuu Lisää-valikosta valitsemalla `Kommentti`. Käyttäjälle avautuu dialogi, jonne hän voi syöttää lisättävän kommentin. Hyväksyminen tapahtuu dialogin `OK`-painikkeen avulla.

Dokumenttityyppi (Document type)-toiminnon avulla käyttäjä voi lisätä dokumenttiin DTD-tiedoston. DTD:n lisäys tapahtuu Lisää-valikosta valitsemalla `Dokumenttityyppi`. Käyttäjälle avautuu dialogi, josta hän valitsee haluamansa DTD-tiedoston hakemistosta tai verkosta syöttämällä kohteen osoitteen.

Prosessointiohje (Processing Instruction)-toiminnon avulla käyttäjä voi lisätä dokumenttiin prosessointiohjeita. Lisäys tapahtuu Lisää-valikosta valitsemalla kohdan `Prosessointiohje`. Käyttäjälle avautuu dialogi, johon syöttää lisättävän prosessointiohjeen ja painaa dialogin `OK`-painiketta.

4.3.4 Työkalut-valikko (Tools)

Työkalut-valikko sisältää dokumentin muuntamiseen ja käsittelyyn liittyviä toimintoja.

Kuva 4.9: Työkalut-valikko.

Validoi (Validate)-toiminnon avulla käyttäjä voi suorittaa dokumentin validoinnin.

Automaattinen validointi (Automatic Validation) -toiminnon ollessa päällä, dokumentin validointi suoritetaan tietyin väliajoin.

XSL muunnos (XSL Transform)-toiminnon avulla käyttäjä voi muuntaa XML-dokumentin toiseen muotoon.

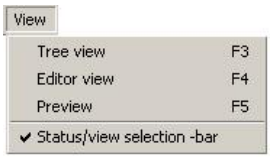
Muodosta DTD (Generate DTD)-toiminnon avulla käyttäjä voi muodostaa dokumentille DTD:n. Toiminto kerää DTD:n määrittämiseksi tarvittavat tiedot muokattavasta XML-dokumentista.

4.3.5 Kieli-valikko (Language)

Kieli-valikossa esitetään käyttöliittymän mahdolliset kielet, joista käyttäjä voi valita mieleisensä. Tuettuja kieliä ovat ainakin suomi ja englanti.

4.3.6 Näkymä-valikko (View)

Näkymä-valikko sisältää ikkunoiden näyttämiseen liittyviä toimintoja.



Kuva 4.10: Näkymä-valikko.

Näkymä-valikon toiminnot kohdistuvat sillä hetkellä aktiivisena olevaan näkymäikkunaan.

Puunäkymä (Tree view)-toiminnon avulla käyttäjä voi piilottaa tai tuodan näkyviin puunäkymän.

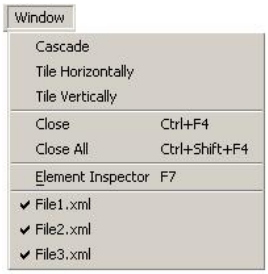
Tekstinäkymä (Text view)-toiminnon avulla käyttäjä voi piilottaa tai tuoda näkyviin tekstinäkymän.

Esikatselunäkymä (Preview)-toiminnon avulla käyttäjä voi piilottaa tai tuoda näkyviin esikatselunäkymän.

Tila/näkymäpalkki (Status/View selection -bar)-toiminnon avulla käyttäjä voi piilottaa tai tuoda näkyviin ikkunan alareunassa olevan tila/näkymäpalkin.

4.3.7 Ikkuna-valikko (Window)

Ikkuna-valikko sisältää ikkunoiden asetteluun liittyviä toimintoja.



Kuva 4.11: Ikkuna-valikko.

Limitä (Cascade)-toiminnon avulla käyttäjä voi limittää avatut näkymäikkunat.

Järjestä leveyssuunnassa (Tile Horizontally)-toiminnon avulla käyttäjä voi jakaa ruudun leveyssuunnassa auki olevien näkymäikkunoiden kesken.

Järjestä pystysuunnassa (Tile Vertically)-toiminnon avulla käyttäjä voi jakaa ruudun pystysuunnassa auki olevien näkymäikkunoiden kesken.

Sulje (Close)-toiminnon avulla käyttäjä voi sulkea aktiivisen näkymäikkunan.

Sulje kaikki (Close All)-toiminnon avulla käyttäjä voi sulkea kaikki avoinna olevat näkymäikkunat.

Ominaisuuseditori (Element Inspector)-toiminnon avulla käyttäjä voi valita, näytetäänkö ominaisuuseditori.

Avoinna olevat näkymät-toimintojen avulla käyttäjä voi vaikuttaa siihen, mitkä näkymäikkunat näytetään ruudulla ja mitkä ovat piilossa.

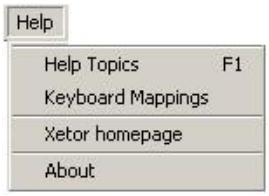
4.3.8 Ohje-valikko (Help)

Ohje-valikosta käyttäjä saa apua ohjelman käyttöön ja tietoa ohjelmasta.

Avustus (Help Topics)-toiminnon avulla käyttäjä voi saada apua ohjelman käyttöön.

Näppäinkomennot (Keyboard Shortcuts)-toiminnon avulla käyttäjä näytetään lista sovelluksessa käytettävissä olevista pikanäppäimistä.

Xetorin kotisivut (Xetor homepage)-toiminnon avulla käyttäjä pääsee Xetor XML-editorin kotisivuille. Toiminto avaa Xetor XML-editorin kotisivun käyttäjän oletusselaimessa.



Kuva 4.12: Ohje-valikko.

Xetor XML editorista (About)-toiminnon avulla käyttäjä saa tietoja sovelluksesta.

4.4 Työkalurivi

Työkalurivillä on pikavalinnat tärkeimmille toiminnoille. Pikavalintoja on uuden dokumentin luomiselle sekä dokumentin avaamiselle että tallentamiselle.

4.5 Puunäkymä

Puunäkymässä kuvataan dokumentin rakenne puumaisesti. Se vastaa ulkoasultaan ja osittain näppäinassosiaatioiltaan Windows-käyttöjärjestelmän resurssienhallinnan hakemistonäkymää. Puun solmut voivat olla joko XML-elementtejä tai tekstiä. Tekstistä näytetään puunäkymässä vain alku ja sen sisällä olevat elementit lapsisolmuina. Puunäkymässä käyttäjä voi muokata dokumentin rakennetta. Validoinnin epäonnistuessa puunäkymä lukittuu ja puunäkymää voi muokata vasta, kun virheet on korjattu tekstinäkymässä. Puunäkymän yhteydessä on sisällönmuokkausikkuna, jossa näytetään puussa valitun elementin sisältö.

4.5.1 Puun osan valitseminen

Puun osan voi valita sekä hiiren vasemmalla että oikealla painikkeella. Vasemmalla painikkeella valitaan vain puun osa ja oikealla painikkeella avataan lisäksi konteksivalikko. Puun osia valittaessa voidaan käyttää myös Windowsin resurssienhallinnasta tuttuja `Shift-` ja `Ctrl`-näppäimiä, joita käytettäessä valinta onnistuu vain hiiren vasemmalla painikkeella. Pitämällä valinnan aikana `Shift`-näppäintä pohjassa, käyttäjä voi valita puusta yhtenäisen osan. `Ctrl`-näppäintä pohjassa pitämällä voidaan valita useita puun osia samaan aikaan. Puunäkymän sisältöikkuna on tyhjä useiden osien ollessa samaan aikaan valittuina.

4.5.2 Elementin ja/tai tekstin lisääminen

Käyttäjä valitsee ensin puusta paikan, johon haluaa lisätä uuden elementin tai tekstiä. Tämän jälkeen hän valitsee kontekstivalikosta kohdan `Lisää`. Tällöin käyttäjälle avautuu dialogi, josta hänen on mahdollista valita, lisääkö hän pelkän elementin, vain tekstiä vai molemmat. Samasta dialogista käyttäjä voi määrittää myös, tehdäänkö lisäys ennen valittua puun solmua, sen jälkeen vai itse solmuun. Käyttäjän valitessa pelkän tekstin lisäämisen, lisätään teksti automaattisesti itse solmuun. Mikäli lisäys joihinkin kolmesta kohdasta ei ole spesifikaation mukainen, poistetaan käyttäjältä mahdollisuus valita kyseinen kohta. DTD:n ollessa valittu dialogissa on lista mahdollisista lisättävistä elementeistä.

Käyttäjä voi määritellä haluamilleen elementeille pikanäppäinyhdistelmät, joita painamalla suoritetaan elementin lisäys. Pikanäppäimiä elementin lisäyksille voi määritellä valikkorivin Muokkaa-valikosta löytyvän `asetukset`-toiminnon kautta.

4.5.3 Leikepöydälle kopiointi

Käyttäjä valitsee puusta kopioitavan osan, jonka jälkeen kopiointi tapahtuu jollakin seuraavista tavoista:

- näppäinyhdistelmällä `Ctrl+C`,
- valitsemalla kontekstivalikosta toiminnon `Kopioi` tai
- valitsemalla Muokkaa-valikosta toiminnon `Kopioi`.

4.5.4 Leikepöydältä liittäminen

Käyttäjä valitsee hiiren oikealla painikkeella puusta kohdan, johon haluaa lisätä leikepöydän sisällön. Ruudulle avautuu kontekstivalikko, josta käyttäjä valitsee `Liitä`-toiminnon. Leikepöydällä oleva teksti lisätään valittuun kohtaan, jonka jälkeen dokumentti validoidaan välittömästi.

4.5.5 Elementin poistaminen

Elementin voi poistaa hiirellä, jolloin käyttäjä valitsee poistettavan elementin hiiren oikealla painikkeella. Avautuvasta kontekstivalikosta hän valitsee kohdan `Poista`. Valitun elementin voi poistaa myös painamalla `Delete`-näppäintä.

4.5.6 Elementin siirtäminen

Käyttäjä tarttuu hiirellä siirrettävään elementtiin, siirtää osoittimen elementin uuteen paikkaan ja vapauttaa hiiren napin. Mikäli siirron jälkeen dokumentti ei ole enää validi, siirtoa ei suoriteta ja käyttäjälle annetaan ilmoitus epäonnistuneesta toiminnosta.

4.5.7 Tekstin lisääminen elementin sisälle

Käyttäjä valitsee hiiren vasemmalla painikkeella puusta elementin, jonka sisälle haluaa tekstiä lisätä. Jos elementillä on jo tekstisisältö, se tulee näkyviin sisällönmuokkausikkunaan. Käyttäjä kirjoittaa haluamansa tekstin sisällönmuokkausikkunaan. Lisätty teksti päivittyy välittömästi dokumenttiin.

4.5.8 Elementin lisääminen tekstin sisälle

Käyttäjä valitsee tekstiä sisältävän elementin puusta. Sen jälkeen käyttäjä valitsee lisättävän elementin paikan sisällönmuokkausikkunasta siirtämällä kursorin haluttuun paikkaan tai maalaamalla tekstin, jonka ympärille haluaa lisätä elementin. Hiiren oikeaa nappia painamalla avautuu näkyviin kontekstivalikko, jossa ylimpänä vaihtoehtona on `Lisää elementti`. Valitsemalla toiminnon avautuu näkyviin sama dialogi kuin kohdassa 4.5.2. Kontekstivalikko sisältää myös tavalliset komennot kuten `leikkaa, liitä` ja `kopioi`.

4.6 Tekstinäkymä

Tekstinäkymä esittää XML-dokumentin tekstimuotoisena. Jokainen rivi alkaa rivinumero-osalla, jonka taustaväri on harmaa. Lähdekoodi esitetään koodikirjasimella ja mikäli dokumentti on XML-, HTML-, DTD-, PHP- tai CSS-dokumentti, esitetään lähdekoodi väreillä eroteltuna.

4.6.1 Elementin lisääminen

Käyttäjä voi lisätä elementin kolmella eri tavalla.

Lisääminen hiirellä Käyttäjä voi lisätä elementin valitsemalla hiiren oikealla painikkeella avautuvasta valikosta toiminnon `Lisää`. Esille tulevasta dialogista hän voi valita lisättävän

elementin ja määrittää sen attribuutit. Mikäli käyttäjä on tätä ennen valinnut tekstiä, lisätään elementti valitun tekstin ympärille. Muutoin elementti lisätään valittuun kohtaan dokumenttia.

Lisääminen pikanäppäintoiminnon avulla Käyttäjä voi lisätä elementin pikanäppäinyhdistelmän `Alt+Insert` avulla. Tällöin näytetään sama dialogi kuin edellisessä kohdassa.

Lisääminen avustetusti Mikäli käyttäjä kirjoittaa tekstinäkymässä merkin '<' ja odottaa hetken, esille tulee lista elementeistä, jotka on mahdollista lisätä kyseiseen kohtaan.

4.7 Virheikkuna

Tekstinäkymään liittyy olennaisena osana virheikkuna. XML-dokumenttia tekstinäkymässä muokattaessa käyttäjä voi tehdä virheitä ja validoinnin epäonnistuessa virheilmoitukset esitetään virheikkunassa. Virheilmoituksesta ilmenee kuvaus virheestä ja rivinumero, jolla mahdollinen virhe on. Virheilmoituksesta on myös linkki tekstinäkymän virheelliselle riville. Virhealueen taustaväri on vihreä dokumentin ollessa hyvin muodostettu tai validi ja muulloin taustaväri on punainen. Virheikkunassa esitetään virheilmoituksia myös käyttäjän avatessa epävalidin dokumentin.

4.8 Esikatselunäkymä

Esikatselunäkymä esittää dokumentin kuten Internet-selain. Esikatselunäkymässä käyttäjä näkee tyylimäärityksien vaikutuksen dokumentin ulkoasuun, mikä mahdollisesti auttaa käyttäjää XML-dokumenttien tekemisessä.

4.9 Ominaisuuseditori

Ominaisuuseditorilla voi muokata elementin attribuuttimäärityksiä ja CSS-tyylimäärityksiä. Ominaisuuseditorissa on välilehdillä eroteltuina elementin attribuuttimääritykset ja CSS-tyylimääritykset. Pakolliset attribuutit on korostettu. CSS-määritykset on ryhmitelty CSS-valitsimien mukaan.

4.9.1 CSS-ominaisuuden lisääminen

Käyttäjä syöttää uuden ominaisuuden nimen listan lopussa olevaan tyhjään tekstilaatikkoon ja painaa sen vieressä olevaa `Lisää ominaisuus` -nappia. Tämän jälkeen ominaisuus lisätään muiden ominaisuuksien perään ja kursori siirtyy juuri lisätyn elementin arvoa määrittävään tekstilaatikkoon, johon käyttäjä voi määritellä ominaisuudelle arvon. Tekstikentän sisällöksi alustetaan lisättävän ominaisuuden oletusarvo. Mikäli oletusarvoa ei ole, kentän sisältö jätetään tyhjäksi.

4.9.2 CSS-ominaisuuden poistaminen

Käyttäjä valitsee poistettavan ominaisuuden ja painaa ominaisuuseditorissa olevaa `Poista`-painiketta. Vaihtoehtoinen tapa on painaa hiiren oikealla painikkeella ominaisuuden nimeä, jolloin esiin tulee kontekstivalikko. Nyt poistaminen onnistuu kontekstivalikosta löytyvällä `Poista`-toiminnolla.

4.9.3 Attribuutin tai CSS-ominaisuuden muokkaaminen

Käyttäjä valitsee ominaisuuden arvo-kentän aktiiviseksi ja syöttää uuden arvon.

4.9.4 CSS-valitsimen lisääminen

Käyttäjä voi lisätä uuden CSS-valitsimen syöttämällä sen nimen ominaisuuseditorin valitsimen lisäyskenttään ja painamalla `Lisää`-painiketta.

4.10 Tila/Näkymäpalkki

Tilapalkissa näytetään lyhyitä tekstimuotoisia ilmoituksia suoritetuista toiminnoista.

Näkymäpalkista käyttäjä voi valita esillä olevat näkymät. Palkissa on valintaruudut puu-, teksti- ja esikatselunäkymille. Käyttäjä voi painaa haluamansa näkymän valintaruudusta esille tai piiloon. Taulukossa 4.10 on listattu tekstejä, jotka voivat ilmestyä tilapalkkiin.

Teksti suomeksi	**Teksti englanniksi**	**Kuvaus**
Dokumentti tallennettu	Document saved	Dokumentti on tallennettu eikä siihen ole tehty sen jälkeen muutoksia.
Validoidaan	Validating	Validointiprosessi on käynnissä.
Validointi onnistui	Validation ok	Validointi onnistui.
Validointi epäonnistui	Validation failed	Validointi epäonnistui.

Taulukko 4.1: Tilapalkin tilat.

4.11 Xetor XML-editorin pikanäppäimet

Pikanäppäin	Toiminto
CTRL+N	Avaa uuden dokumentin
CTRL+O	Avaa tallennetun dokumentin
CTRL+F4	Sulkee valitun dokumentin
CTRL+S	Tallentaa valitun dokumentin
CTRL+A	Tallentaa dokumentin nimellä
CTRL+SHIFT+S	Tallentaa kaikki avoinna olevat dokumentit
CTRL+SHIFT+P	Näyttää tulostuksen esikatselunäkymän
CTRL+P	Tulostaa valitun dokumentin
CTRL+E	Tallentaa puunäkymän png-muodossa
ALT+F4	Sulkee ohjelman
CTRL+Z	Kumoaa edellisen dokumentin muokkaustoiminnon
CTRL+SHIFT+Z	Toistaa viimeksi kumotun toiminnon
CTRL+X	Siirtää valitun tekstin leikepöydälle
CTRL+V	Liittää tekstin leikepöydältä valittuun kohtaan
CTRL+A	Aktivoi valitun dokumentin sisällön
CTRL+F	Näyttää etsi-dialogin
CTRL+H	Näyttää etsi-korvaa-dialogin
F3	Tekstinäkymän näyttäminen
F4	Puunäkymän näyttäminen
F5	Esikatselunäkymän näyttäminen
F7	Ominaisuuseditorin näyttäminen
CTRL+SHIFT+V	Validoi dokumentin
CTRL+F10	Avaa asetuksista välilehden pikanäppäimien määrittämiseksi
CTRL+ENTER	Avaa linkitetyn dokumentin
CTRL+TAB	Aktivoi seuraavan ikkunan
CTRL+SHIFT+TAB	Aktivoi edellisen ikkunan
CTRL+SHIFT+F4	Sulkee kaikki avoinna olevat dokumentit
CTRL+SHIFT+F	Etsii valittua merkkijonoa uudestaan
F1	Avaa ohjeen
ALT+INSERT	Avaa elementtien lisäysdialogin

Lisäksi käyttäjä saa määritellä itse pikanäppäimiä elementtien lisäykselle. Näille toiminnoille on varattu kaikki sellaiset muotoa ALT + ’merkki’ olevat pikanäppäinyhdistelmät, jotka eivät ole varattuja. Merkki voi olla mikä tahansa näppäimistön merkki.

5 Lähdekoodin ulkoasu

Tässä luvussa esitellään sovellusten dokumentointi- ja ohjelmointikäytäntöä.

5.1 Kommentointityyli

Lähdekoodin dokumentoinnissa noudatetaan yhtenäistä käytäntöä. Ohjelmakoodi kommentoidaan sovitun käytännön mukaisesti englanniksi. Jokaisen lähdekooditiedoston alkuun kirjoitetaan projektin nimi, tekijän nimi, päivämäärä, tiedoston nimi, vuosiluku ja merkintä käytetystä lisenssistä sekä riittävä ja selkeä kuvaus lähdekoodin tarkoituksesta. Jokaisen luokan sekä metodin toiminta ja tarkka muutoshistoria kommentoidaan erikseen ennen sen esittelyä. Tärkeimmät tiedot lähdekoodin lisäys- ja muutoshistoriasta - muutoksen tekijä, päivämäärä ja tehdyt muutokset - kommentoidaan lisäksi projektin CVS-versiohistoriaan.

Ohjelmakoodin sisennyksissä käytetään kahta välilyöntiä. Koodirivit ovat maksimissaan 80 merkkiä pitkiä. Kolmannelta osapuolelta saatuihin lähdekoodeihin kommentoidaan vain ne kohdat, joita projektiryhmä on muokannut.

Yhteen lähdekooditiedostoon kirjoitetaan ainoastaan yksi luokka ja siihen kiinteästi liittyvät luokat, kuten tapahtumankäsittelijäluokat käyttöliittymän lähdekoodissa. Lähdekoodista muodostetaan dokumentointi käyttäen JavaDoc-sovellusta.

5.2 Nimeämiskäytäntö

Ohjelmakoodin luokat, paketit, attribuutit, metodit, oliot, parametrit ja muuttujat nimetään englanniksi. Pakettien (Package) nimet, luokkien metodit ja paikallisten muuttujien nimet kirjoitetaan pienellä alkukirjaimella. Attribuuttien ja parametrien nimet kirjoitetaan isolla alkukirjaimella.

Mikäli muuttuja on luokan attribuutti, lisätään sen alkuun pieni a-kirjain ja mikäli muuttuja on metodikutsun parametri, lisätään sen alkuun pieni p-kirjain. Muiden lähdekoodin nimiin ei tällaista unkarilaisen notaation mukaista alkukirjainta lisätä.

Luokkien nimet kirjoitetaan isolla alkukirjaimella ja tiedoston nimen etuliite on aina luokan nimi ja tiedostopääte on java. Muodostajametodien nimet kirjoitetaan luokkien nimien tapaan.

Mikäli jokin nimi koostuu useasta sanasta, sanat kirjoitetaan yhteen ja muiden paitsi ensimmäisen sanan alkukirjaimet isolla. Ensimmäiseen kirjaimeen sovelletaan edellä esitettyjä

käytäntöjä.

Nämä menettelytavat tulevat esille luvun 5.3 koodiesimerkistä. Taulukko 5.2 esittää tiivistetysti nimeämiskäytännön koodin eri osissa.

Kohde	**Alkukirjain**	**Isolla/pienellä**	**Esimerkki**
Paketti	-	pienellä	`xetorGUI`
Luokka	-	isolla	`Validator`
Metodi (muodostaja)	-	isolla	`Validator()`
Metodi (ei muodostaja)	-	pienellä	`validate()`
Attribuutti	`a`	isolla	`aBuffer`
Parametri	`p`	isolla	`pWrittenText`
Paikallinen muuttuja	-	pienellä	`tempString`

Taulukko 5.2: Koodauskäytäntö.

5.3 Lähdekoodiesimerkki

Alla oleva lähdekoodiesimerkki kuvaa aiemmin mainittua nimeämiskäytäntöä ja kommentointityyliä:

```
package xetorExamples;

/**
 * <p>Title: Xetor</p>
 * <p>Description: XML-editor</p>
 * <p>Copyright: Copyright (c) 2004.
 *    Licensed under the Academic Free License version 2.0.</p>
 * <p>Company: University of Jyväskylä</p>
 * @author Jani Lirkki
 * @version 0.002
 *
 * XetorCodeExample.java
 * Created 23.3.2004
 *
 * This code represents the coding style of Xetor-project.
 * The coding protocol is as follows:
 *
 *   Target            &  first letter & big/small & example
 * -------------------------------------------------
 *   Packet                   & - &        small   & xetorGUI
 *   Class                    & - &        big     & Validator
 *   Method (constructor)     & - &        big     & Validator()
```

```
 *   Method (not constructor) & - &        small   & validate()
 *   Attribute                & a &        big     & aBuffer
 *   Parameter                & p &        big     & pWrittenText
 *   Local variable           & - &        small   & tempString
 */

public class XetorCodeExample {
  private String aStr;

  /** Creates a new instance of XetorCodeExample. */
  public XetorCodeExample() {
    aStr = new String();
  }


  /** Creates a new instance of XetorCodeExample.
   * @param pStr String to set initially.
   */
    public XetorCodeExample(String pStr) {
      aStr = new String(pStr);
    }

  /**
   * Sets aStr value.
   * @param vstr String to set.
   * @exception NullPointerException is thrown
   * if vstr is null.
   *
   * Changes:
   *   23.3.2004 JL: Set to thow an NullPointerException if pStr is null
   */
    public void setString(String pStr) throws NullPointerException {
      if (pStr == null)
        throw new NullPointerException();
      aStr = pStr;
    }

    /**
     * Gets str value.
     * @return String str.
     */
    public String getString() {
      return aStr;
    }


    /**
     * Clears str value.
```

```
     */
    public void clear() {
      setString("");
    }

}
```

6 Testaus

Tässä kappaleessa kuvataan sovelluksen testaamiseen liittyviä asioita, kuten testausperiaatteet ja testattavat asiat. Sovelluksen toteutuksen aikana tullaan suorittamaan JUnit-yksikkötestausta tärkeimmille komponenteille. Lisäksi Xetor-projekti toteuttaa testaussuunnitelman, jossa esitetään sovelluksen hyväksymistestaus.

6.1 Testausperiaatteet

Projektiryhmä testaa kaiken tuottamansa ohjelmakoodin. Ohjelmoinnin aikana suoritetaan JUnit-yksikkötestausta, jossa ohjelmakoodia testataan luokkakohtaisesti. Yksikkötestauksessa luokan ohjelmoija tekee testausohjelman, jonka avulla testauksesta saadaan automaattista. JUnit-yksikkötestauksesta kerrotaan tarkemmin luvussa 6.2. Testauksen aikana mahdollisesti löydetyt virheet korjataan ja korjaukset testataan. Lopulta luokka hyväksytään ja yksikkötestaus on päättynyt tämän luokan osalta.

Kun ryhmä on tuottanut iteraatioon liittyvät tärkeimmät luokat, ryhmä siirtyy integraatiotestausvaiheeseen. Integraatiotestauksessa tarkistetaan, toimivatko ohjelmiston pienemmät osaset yhdessä määritellyn mukaisesti.

Kun iteraatio alkaa olla valmis, siirtyy ryhmä järjestelmätestausvaiheeseen. Järjestelmätestien aikana tarkastetaan, toimiiko järjestelmä määritysten mukaisesti. Järjestelmätestaus on määritelty testaussuunnitelmassa. Järjestelmätestauksen aikana tilaaja suorittaa alpha-testausta, tutkien täyttääkö toteutettu järjestelmä tilaajan asettamat vaatimukset. Alpha-testausta jatketaan, kunnes tilaaja ja ryhmä ovat hyväksyneet järjestelmän ominaisuudet.

Mikäli ryhmä tai tilaaja löytää virheitä järjestelmän toiminnasta, korjataan järjestelmää. Ryhmä korjaa tarvittavia luokkia ja testaus aloitetaan muokattujen luokkien osalta yksikkötestausvaiheesta lähtien.

6.2 JUnit-yksikkötestaus

Menetelmän ideana on ajaa testattavan luokan metodeita keksityissä testitapauksissa. Tapaukset ovat yleensä staattisia ja ne testaavat ensin toiminnan tavallisella syötteellä, jonka jälkeen edetään erikoistapauksiin sekä rajatapauksiin. Esimerkiksi metodille, joka ottaa syötteekseen kokonaislukuja välillä 3-6, voidaan ensin antaa syötteeksi 5, sitten 3 ja 6 ja viimeisenä vaikka 2 ja 7. Viimeinen tarkistaa, että metodi reagoi halutulla tavalla myös virheelliseen syötteeseen.

Metodien mahdollisia palautusarvoja verrataan toivottuun tulokseen. Tämän takia yleensä käytetään kiinteitä syötteitä. Toivottuja vasteita ei tarvitse laskea, vaan ne voidaan kirjoittaa testiin vakioina. Jokaiseen vertailuun on hyvä kertoa mahdollisimman valaisevasti, mitä testataan.

7 Tiedostot

Tässä luvussa kuvataan sovelluksessa käytettäviä tiedostoja. Tiedostot on tallennetaan XML-muodossa ja niiden rakennetta on esitelty tarkemmin alaluvuissa 7.2 ja 7.3. Lisäksi käytössä on valmiita dokumenttipohjia, joita esitellään luvussa 7.1.

7.1 Dokumenttipohjat

Käyttäjä voi aloittaa uuden dokumentin valmiista dokumenttipohjasta. Dokumenttipohja sisältää tietyn dokumenttityypin perusrakenteen. Valmiit dokumenttipohjat sijaitsevat niille varatussa hakemistossa. Sovellukseen lisätään ainakin XHTML 1.0 Strict, XHTML basic ja XHTML 1.1 -dokumenttipohjat.

7.2 Sanasto

Sanasto-tiedostossa on määritelty kaikki ohjelman käyttöliittymässä esiintyvät sanat suomeksi ja englanniksi XML-muodossa.

Dokumentin juurielementti on `<xetorlanguages>`. Sillä on lapsielementti `<language>`. Language elementillä on attribuuttina `name`, joka kertoo käytettävän kielen nimen. `Language`-elementillä on puolestaan lapsina `<word>`-elementtejä, joilla on attribuuttina `base`. `Base` toimii avainsanana ohjelmakoodissa. Seuraavana on esitelty käytettävän XML-dokumentin DTD (lang.dtd) ja esimerkki XML-dokumentin rakenteesta on mukana tämän dokumentin liitteenä.

```
<!ELEMENT language (word)*>
<!ATTLIST language name CDATA #IMPLIED>

<!ELEMENT word (#PCDATA)>
<!ATTLIST word base CDATA #IMPLIED>

<!ELEMENT xetorlanguages (language)*>
```

7.3 Asetukset

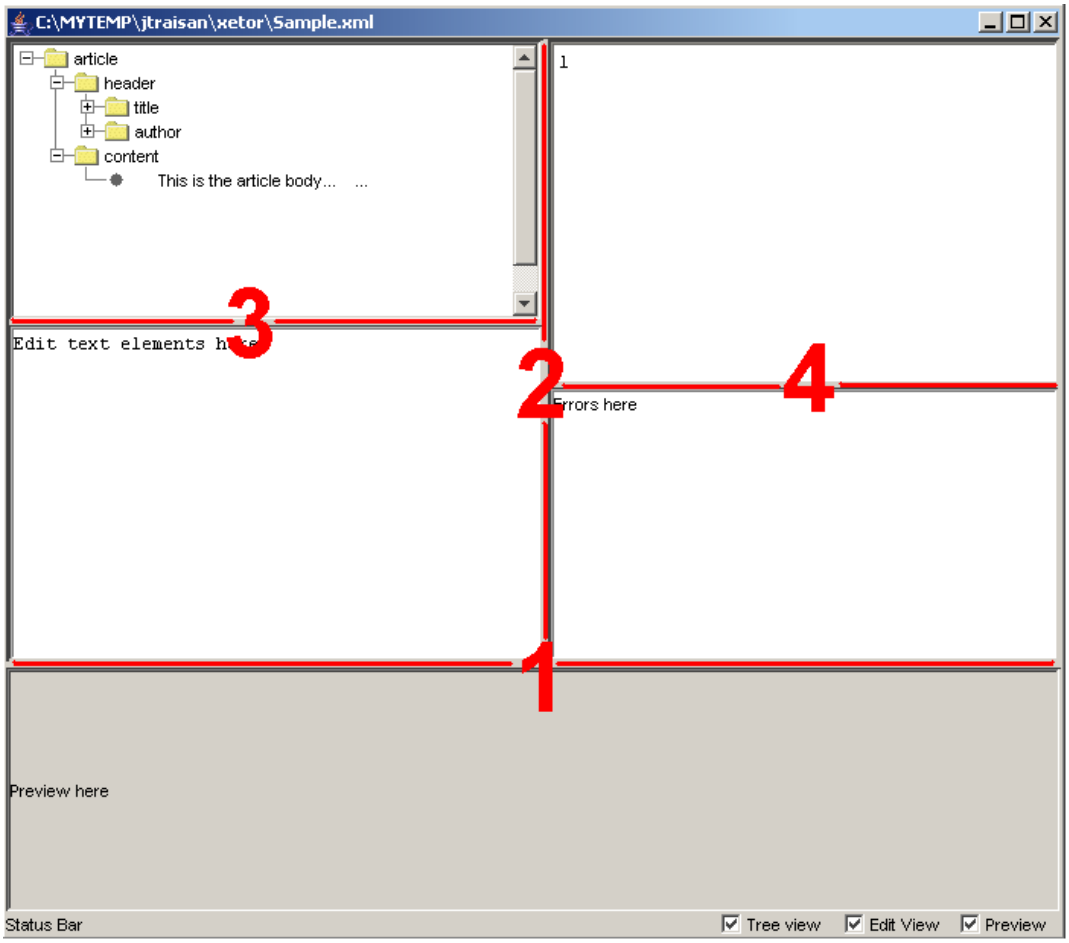
Editorin asetukset tallennetaan XML-muotoiseen dokumenttiin, kun ohjelma suljetaan. Näitä tietoja käytetään sovelluksen seuraavassa käynnistyksessä sovelluksen tilan saattamiseksi lopettamista edeltäneeseen tilaan. Kappaleessa 7.3.1 kerrotaan, mitä asetuksia tallennetaan

ja kappaleessa 7.3.2 on esitelty asetustiedoston DTD (settings.dtd). Esimerkki asetustiedostosta on liitteenä.

7.3.1 Tallennettavat asetukset

Sovelluksen asetustiedostoon kirjoitetaan seuraavat tiedot:

- Ikkunan korkeus, leveys ja sijainti.
- Tieto aktiivisena olevasta ikkunasta.
- Avoinna olevat dokumentit ja niistä avoinna olevat näkymät.
- Dokumentti-ikkunoiden sijainnit ja koot.
- Dokumentti-ikkunoissa olevien jakopaneelien jakajien sijainnit (Esitetty kuvassa 8.14):
    1. Jakaja, joka erottaa esikatselunäkymän teksti- ja puunäkymästä sekä virhe- ja sisältöikkunasta.
    2. Jakaja, joka erottaa tekstinäkymän ja virheikkunan puunäkymästä ja sisältöikkunasta.
    3. Puunäkymän ja sisällönmuokkausikkunan välillä oleva jakaja.
    4. Tekstinäkymän ja virhealueen välillä oleva jakaja.
- Ominaisuuseditorin sijainti, korkeus, leveys.



Kuva 7.13: Näkymäikkunan jakopaneelien jakajat.

7.3.2 Asetustiedoston DTD

```
<!ELEMENT activeframe (#PCDATA)>
<!ELEMENT common(windowbottomright|activeframe|elementinspector
                |windowtopleft)*>
<!ELEMENT divpos (#PCDATA)>
<!ELEMENT doc (views|filename)*>
<!ELEMENT eibottomright (y|x)*>
<!ELEMENT eitopleft (y|x)*>
<!ELEMENT elementinspector (eibottomright|eitopleft|isopen)*>
<!ELEMENT filename (#PCDATA)>
<!ELEMENT framebottomright (y|x)*>
<!ELEMENT frametopleft (y|x)*>
<!ELEMENT horizontaldiv (#PCDATA)>
<!ELEMENT isopen (#PCDATA)>
<!ELEMENT opendocs (doc)*>
<!ELEMENT preview (isopen)*>
```

```
<!ELEMENT textview (isopen|divpos)*>
<!ELEMENT treeview (isopen|divpos)*>
<!ELEMENT verticaldiv (#PCDATA)>
<!ELEMENT view (frametopleft|treeview|framebottomright|viewid|textview|
               verticaldiv|preview|horizontaldiv)*>
<!ELEMENT viewid (#PCDATA)>
<!ELEMENT views (view)*>
<!ELEMENT windowbottomright (y|x)*>
<!ELEMENT windowtopleft (y|x)*>
<!ELEMENT x (#PCDATA)>
<!ELEMENT xetorsettings (common|opendocs)*>
<!ELEMENT y (#PCDATA)>
```

8 Yhteenveto

Xetor-projektiryhmä toteuttaa XML-editorin informaatioteknologian tiedekunnan käyttöön. Tässä dokumentissa on kuvattu sovelluksen toimintoja, rakennetta ja käyttöliittymää sekä kerrottu testauksesta. Myös koodauskäytäntö on selvitetty ja sitä on havainnollistettu esimerkillä. Editorin käyttöliittymä on kuvattu hyvin yksityiskohtaisesti toteutusvaiheen ongelmien minimoimiseksi. Xetor XML-editori toteutetaan tämän suunnitelman pohjalta.

Lähteet

[1] World Wide Web Consortium, ”Cascading Style Sheets, level 2 CSS2 Specification”,http://www.w3.org/TR/CSS2/, 8-3-2004.

[2] World Wide Web Consortium, ”Document Object Model (DOM) Level 2 Core Specification”, http://www.w3.org/TR/2000/REC-DOM-Level-2-Core-20001113/, 8-3-2004.

[3] World Wide Web Consortium,”HyperText Markup Language (HTML) Home Page”,http://www.w3.org/MarkUp/, 9-3-2004.

[4] Sun Microsystems, ”Java Technology”, http://java.sun.com/, 9-3-2004.

[5] Dave Shea, ”mozilla - home of the mozilla, firefox, and camino web browsers”, http://www.mozilla.org, 9-3-2004.

[6] Jaakohuhta Hannu, ”Suuri tietotekniikan tietosanakirja”, Suomen Atk-kustannus Oy, Helsinki, 1999.

[7] World Wide Web Consortium, ”XHTML 1.0 The Extensible HyperText Markup Language (Second Edition)”, http://www.w3.org/TR/xhtml1/, 9-3-2004.

[8] World Wide Web Consortium, ”Extensible Markup Language (XML) 1.0”, http://www.w3.org/TR/1998/REC-xml-19980210, 8-3-2004.

[9] Ruini Henri, ”Englanti - suomi -sanasto, v. 0.7 (XML)”, http://www.cs.helsinki.fi/u/ruini/structure/xml/sanasto.html, 9-3-2004.

[10] World Wide Web Consortium, ”Extensible Stylesheet Language (XSL) Version 1.0”, http://www.w3.org/TR/xsl/, 8-3-2004.

[11] JDOM Project,”JDOM v1.0beta10-dev API Specification”, http://www.jdom.org/docs/apidocs/index.html, 30-3-2004.

Liite 1

Esimerkki kielitiedostosta.

```
<xetorlanguages>
  <language name="Suomi">
    <word base="xetorxmleditor">Xetor XML-editori</word>
    <word base="file">Tiedosto</word>
    <word base="new...">Uusi...</word>
    <word base="open...">Avaa...</word>
    <word base="close...">Sulje...</word>
    <word base="close">Sulje</word>
    <word base="closeall">Sulje kaikki</word>
    <word base="save">Tallenna</word>
    <word base="saveas...">Tallenna nimellä...</word>
    <word base="saveall">Tallenna kaikki</word>
    <word base="export">Tallenna png-kuvana...</word>
    <word base="print">Tulosta</word>
    <word base="printpreview">Esikatselu</word>
    <word base="exit">Poistu</word>
  </language>

  <language name="English">
    <word base="xetorxmleditor">Xetor XML-editor</word>
    <word base="file">File</word>
    <word base="new...">New...</word>
    <word base="open...">Open...</word>
    <word base="close...">Close...</word>
    <word base="close">Close</word>
    <word base="closeall">Close all</word>
    <word base="save">Save</word>
    <word base="saveas...">Save as...</word>
    <word base="saveall">Save all</word>
    <word base="export">Export as png-file...</word>
    <word base="print">Print</word>
    <word base="printpreview">Print preview</word>
    <word base="exit">Exit</word>
  </language>
</xetorlanguages>
```

Liite 2

Esimerkki asetustiedostosta.

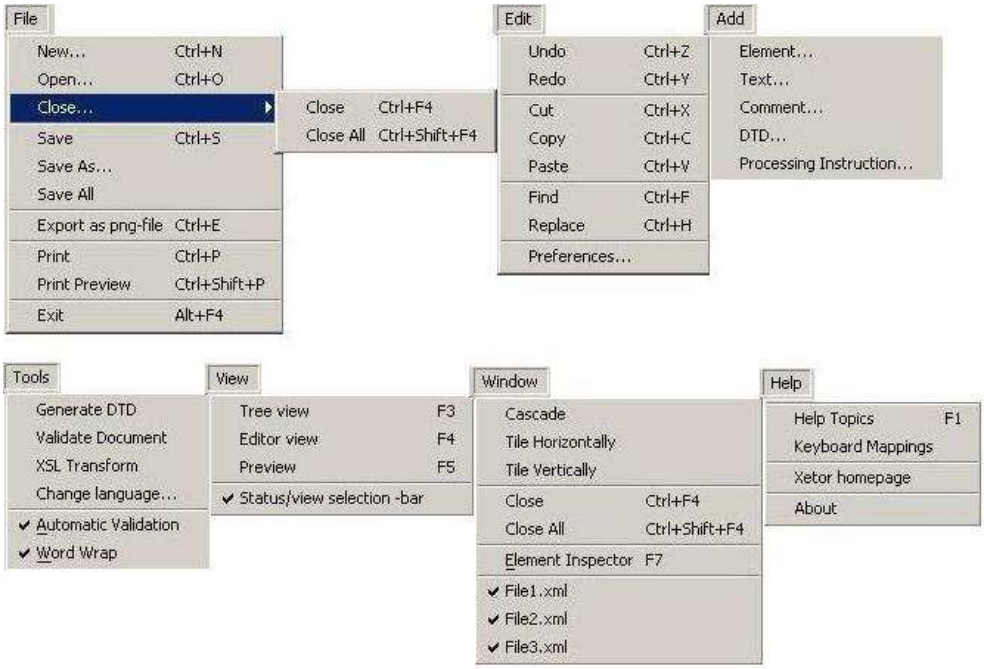
```
<xetorsettings>

  <common>
    <windowtopleft>
      <x>10</x>
      <y>20</y>
    </windowtopleft>
    <windowbottomright>
      <x>200</x>
      <y>300</y>
    </windowbottomright>
    <activeframe>3</activeframe>
    <elementinspector>
      <isopen>true</isopen>
      <eitopleft>
        <x>10</x>
        <y>20</y>
      </eitopleft>
      <eibottomright>
        <x>200</x>
        <y>300</y>
      </eibottomright>
    </elementinspector>
  </common>

  <opendocs>
    <doc>
      <filename>C:\kissa\koira.xml</filename>
      <views>
        <view>
          <viewid>1</viewid>

          <frametopleft>
            <x>50</x>
            <y>70</y>
          </frametopleft>

          <framebottomright>
            <x>180</x>
            <y>250</y>
          </framebottomright>

          <horizontaldiv>30</horizontaldiv>
```

```
      <verticaldiv>30</verticaldiv>

      <treeview>
        <isopen>true</isopen>
        <divpos>20</divpos>
      </treeview>

      <textview>
        <isopen>true</isopen>
        <divpos>30</divpos>
      </textview>

      <preview>
        <isopen>true</isopen>
      </preview>
    </view>
  </views>
</doc>
<doc>
  <filename>c:\koira\kissa.xml</filename>
  <views>
    <view>
      <viewid>2</viewid>

      <frametopleft>
        <x>50</x>
        <y>70</y>
      </frametopleft>

      <framebottomright>
        <x>180</x>
        <y>250</y>
      </framebottomright>

      <horizontaldiv>30</horizontaldiv>
      <verticaldiv>30</verticaldiv>

      <treeview>
        <isopen>true</isopen>
        <divpos>20</divpos>
      </treeview>

      <textview>
        <isopen>true</isopen>
        <divpos>30</divpos>
      </textview>

      <preview>
```

```
            <isopen>true</isopen>
          </preview>
        </view>
        <view>
          <viewid>3</viewid>

          <frametopleft>
            <x>50</x>
            <y>70</y>
          </frametopleft>

          <framebottomright>
            <x>180</x>
            <y>250</y>
          </framebottomright>

          <horizontaldiv>30</horizontaldiv>
          <verticaldiv>30</verticaldiv>

          <treeview>
            <isopen>true</isopen>
            <divpos>20</divpos>
          </treeview>

          <textview>
            <isopen>true</isopen>
            <divpos>30</divpos>
          </textview>

          <preview>
            <isopen>true</isopen>
          </preview>
        </view>
      </views>
    </doc>
  </opendocs>
</xetorsettings>
```

Liite 3

Kuva valikkorakenteesta.



Kuva 8.14: Kaikki ohjelman valikot.