Lönnrotin kirjeiden digitaalinen julkaisu, julkaisualustan dokumentaatio

Maria Niku 01/2017

Koodi kokonaisuudessaan: https://github.com/marianiku/omeka

(Omekan alkuperäinen koodi: https://github.com/omeka/Omeka)

Tämä dokumentaatio kuvaa julkaisualustan toteutuksessa Omekan koodiin tehdyt muutokset ja lisäykset. Kukin kooditiedosto, johon on tehty muutoksia, kuvataan erikseen. Omekan kooditiedostot, joihin ei ole tehty muutoksia, jätetään kuvaamatta.

Sisällysluettelo

1. Yleistä julkaisualustasta

Omeka, versio 2.x

Omekan lisäosat:

-CsvImport, https://github.com/Daniel-KM/CsvImportPlus
Datan ja tiedostojen tuonti Omekaan

-SolrSearch, https://github.com/scholarslab/SolrSearch
Apache Solr, vapaatekstihaku

-SimplePages, https://github.com/omeka/plugin-SimplePages
Yksinkertaiset html-sivut, esim. esipuhe, ohjeet

-UniversalViewer, https://github.com/Daniel-KM/UniversalViewer4Omeka
Vaihtoehtoinen Jqueryllä toteutetun viewerin kanssa, päätettävä kumpaa käytetään julkaisussa

-LanguageSwitcher, https://gitlab.com/TIME_LAS/Omeka_Plugin_SwitchLang/

Plugin käyttöliittymän kielen vaihtamiseen (suomi/ruotsi)

Käytettävä teema: default, /themes/default

2. Kieliasetukset

Default-kieli on suomi, vaihtoehtoinen kieli ruotsi. Kielen vaihtamiseen käytetään LanguageSwitcher-liitäntää

/application/languages/omeka.pot
/application/languages/fi_FI.po, sv_SE.po
/application/config/config.ini, r. 31: locale.name = "fi_FI"

Käännöstä vaativat merkkijonot on lueteltu /application/languages-kansion omeka.pot-tiedostossa. Käännökset listataan .po-tiedostoissa ja nämä on käännetty .mo-tiedostoiksi. Jokainen käännöstä vaativaa merkkijono ilmaistaan koodissa _('word to be translated'). Kääntäjä hakee .mo-tiedostosta merkkijonoa vastaavan käännöksen.

3. HTML-viewit

Tässä käsitellään etusivu, yläpaneeli, kokoelmien ja itemien listasivut sekä itemien näyttösivu. Hakutulosten HTML-viewit käsitellään hakua koskevien otsikkojen alla. HTML-elementtien CSS: /themes/default/css/style.css.

3a. Yläpaneeli

/themes/default/common/header.php, r. 108-141

r. 110-115: Julkaisun nimi ja SKS:n logo (sis. linkki SKS:n sivuille), kielivalikko
r. 118-124: linkit - kaikki kirjeet (mahd. ei mukaan); kirjeiden vastaanottajat (kokoelmaluettelo), esipuhe (SimplePages-sivu); ohjeita -> r. 136-140 dropdown -> transkriptioiden merkinnät (SimplePages-sivu); laajennettu haku -> r. 131-135 hakukaavake-dropdown
r. 127: tekstihakukenttä (Solr)

3b. Etusivu

/themes/default/index.php

r. 3-26 (vasen palsta): esittelyteksti; piilotettu featured item, featured collection ja featured exhibit (Omekan asetukset)

3c. Item-listaukset

Tehdyt muutokset vaikuttavat itemien listaukseen selaussivuilla, kokoelmien alla ja Omeka-haun hakutuloksissa. Kuvailtujen muutosten lisäksi poistettu turhia otsikoita, muutettu otsikoiden kokoja

/application/controllers/ItemsController.php, r. 186-189, _getBrowseDefaultSort():
Muutettu item-listausten järjestystä: kirjeiden kirjoitusaika & nouseva, return array('Dublin Core,Date', 'a') (defaultina lisäysaika & laskeva, return array('added', 'd'))

/application/views/scripts/items/browse.php, r. 8-10: piilotettu navigaatiopaneeli, jossa linkit hakusivulle ja kokoelmiin.

r. 18-23: muutettu/lisätty listausten sorttausvaihtoehtoja: kirjoitusaika, otsikko, kirjoittaja (default otsikko, kirjoittaja, lisäysaika)

r. 43: lisätty ilmoitus, mikäli haku-/selaustulos tyhjä.

r. 55-: lisätty listausnäkymään kirjoitusaika, muodossa pp.kk.vvvv (tuodaan muodossa vvvv-

r. 41–: muutettu listauksen sisältävä div skrollattavaksi (/themes/default/css/style.css)

3d. Kokoelmalistaukset

/application/controllers/CollectionsController.php, r. 29-37, browseAction()
Muutettu kokoelmien sorttausta: kokoelman nimi & nouseva, $this->_setParam('sort_field', 'Dublin Core, Title'), $this->_setParam('sort_dir', 'a') (ei default-asetusta)

/application/views/scripts/collections/browse.php, r. 21
Kokoelman otsikon linkki muutettu osoittamaan kokoelmaan kuuluvien kirjeiden listaukseen, link_to_items_browse(…) (default linkki erilliselle kokoelman esittelysivulle, link_to_collection())

r. 22-24: thumbnailista poistettu linkki erilliselle kokoelman esittelysivulle (link_to_collection())

r. 26-31: lisätty kokoelman kirjeiden määrä otsakkeen perään sulkeisiin.

Kokoelmaluettelon asettelu muutettu pystysuorasta galleriatyyppiseksi listaukseksi (r. 19-).

3e. Itemien näyttösivu

/themes/default/items/show.php

Sivulla ladattavat skriptit (r. 2 – 10): /themes/default/javascripts (ks. 4. Skriptit)

Kuvien näyttämiseen käytetään UniversalViewerіä (r. 29-36) tai Jqueryllä rakennettua viewerіä (r. 67-95). Viewerin oikealla puolella näytetään kuvakohtainen transkriptioteksti skrollattavassa divissä (r. 37-62 ja 96-121) ja tämän päällä nuolipainikkeet kuvissa/sivuilla (r. 42-45 ja 101-104),

checkboxit merkintöjen ja popup-kommenttien /piilottamiseen (r. 38-41 ja 97-100) sekä latauslinkki TEI-tiedostolle (r. 47-50, 118-121, HTML5:n download-attribuutti). Jokaisen itemin sivulla näytetään lisäksi linkki TEI-tiedostoon (r. 14-25).

Jqueryllä rakennetussa viewerissä on painikkeet kuvan alkuperäisen koon palauttamiseen, koko sivun näyttämiselle, koko sivun näytöstä poistumiselle ja metadatalistaukselle (r. 69-77). Jquery-viewerin toiminnot: ks. 4. Skriptit.
Transkriptioteksti ladatataan div-kehykseen käyttäen PHP:n XSLTProcessor()-luokkaa (r. 46-61 ja 105-120). DOMDocument()-luokan instanssi lataa itemiin liittyvän xml-tiedoston (r. 51-52/110-111), toinen instanssi lataa xsl-tyylitiedoston (r. 53-54/112-113) ja transformToXML()-metodi kääntä xml:n xhtml:ksi (r. 57/116).

4. Skriptit (javascript)

/themes/default/javascripts

Skriptit ladataan itemien näyttösivulla (/themes/default/items/show.php, ks. kappale 3e) lukuunottamatta tiedostoa header_menus.js, joka ladataan headerissa (/themes/default/common/header.php).

4a. jquery-1.12.4.min.js

jQuery-kirjasto, uusin Omekan tukema versio.

4b. page-formatting-xhtml.js

Jakaa xhtml:ksi käännetyn kirjeen transkriptiotekstin sivuihin itemien näyttösivulla (/themes/default/items/show.php, ks. 3e). Koodi käy läpi xhtml:ksi käännetyn transkription ( ) p-elementit, siirtää niiden sisällä olevat sivunvaihtotagit (=) samalle tasolle p-elementtien kanssa ja käärii sivunvaihtotagien väliset html-elementit div-tageihin. Jos sivunvaihtotagit ovat valmiiksi samalla tasolla p-kappaleen kanssa (esim. aina transkription ensimmäinen sivunvaihtotagi), koodi ei tee mitään.

1. Lähtötilanne

```
<div>
   <span class="pb_1">…</span>
   <p>
      …
      <span class="pb_2">…</span>
      …
    </p>
    <p>
      …
      <span class="pb_3">…></span>
      …
   </p>
```

```
   <p>
      …
   </p>
</div>
```

2. p-kappaleen sisällä yksi sivunvaihtotagi, r. 7-14

Sivunvaihtotagi tallennetaan child-muuttujaan (r. 9).

p-elementin alkuperäinen html-koodi jaetaan kahtia sivunvaihtotagin kohdalla ja saadut kaksi osaa tallennetaan content-arrayhin (r. 10).

replaceWith()-metodi korvaa p-elementin alkuperäisen html:n seuraavasti: content-arrayn ensimmäinen elementti käärittynä p-tageihin + sivunvaihtotagi + content-arrayn toinen elementti käärittynä p-tageihin (r. 11-13).

3. p-kappaleen sisällä useampia sivunvaihtotageja, r. 16-40

Ensiksi luodaan tyhjä div-elementti välivaiheiden säilyttämistä varten (r. 18).

Halutaan löytää sivunvaihtotagien välinen sisältö.

Jokaisen p-elementin sisällä olevan sivunvaihtotagin kohdalla (r. 20-35) haetaan thisIndex-muuttujaan p-elementin html:n kohta, jossa sivunvaihtotagin sulkutagi päättyy (r. 22).

Jos käsittelyssä oleva sivunvaihtotagi ei ole p-elementin viimeinen (r. 24-30), seuraavan sivunvaihtotagin (r. 25) kohta p-elementin sisällä tallennetaan nextIndex-muuttujaan (26). thisIndex- ja nextIndex-muuttujien välinen sisältö haetaan substring()-metodilla (r. 27) ja sivunvaihtotagi sekä noudettu sisältö käärittynä p-tageihin liitetään aikaisemmin luotuun tyhjään div-elementtiin (r. 28-29).

Jos käsittelyssä oleva sivunvaihtotagi on p-elementin viimeinen (r. 30-34), noudetaan substring()-metodilla thisIndex-muuttujan ja p-elementin lopputagin välinen sisältö ja sivunvaihtotagi sekä noudettu sisältö p-tageihin käärittynä (r. 32-33).

Placeholder-div:in sisältää nyt p-elementin alkuperäisen html-sisällön osat elementin sisällä olevasta ensimmäisestä sivunvaihtotagista elementin lopputagiin saakka:

```
<div>
   <span class="pb_1">…</span>
   <p>…</p>
   <span class="pb_2">…</span>
   <p>…</p>
   <p>…</p>
   <span class="pb_3">…</span>
   <p>…</p>
   <p>…</p>
</div>
```

Seuraavaksi p-elementin alkuperäinen html-sisältö jaetaan kahtia ensimmäisen elementin sisällä olevan ensimmäisen sivunvaihtotagin kohdalla ja osat tallennetaan content-arrayksi (r. 37). p-elementin alkuperäinen html-sisältö korvataan content-arrayn ensimmäisellä elementillä (r. 38) ja placeholder-divin lapsielementit liitetään järjestyksessä tämän perään (r. 39).

4. Sivujako, r. 44-50

Jokaisen sivunvaihtotagin välinen sisältö haetaan ja kääritään div-elementteihin <div class="page>".

5. Lopputilanne

```
<span class="pb_1">…</span>
<div class="page">
   <p>
      …
   </p>
</div>

<span class="pb_2">…</span>
<div class="page">
   <p>
     …
   </p>
   <p>
      …
   </p>
</div>
<span class="pb_3">…</span>
<div class="page">
   <p>
      …
   </p>
   <p>
     …
   </p>
</div>
```

Kun käyttäjä avaa itemin näyttösivun, lähtötilanne on, että näkyvillä on kirjeen ensimmäinen kuva ja ensimmäinen sivu, lopuksi kaikki muut paitsi ensimmäinen -elementti piilotetaan (r. 52-53). Myös sivunvaihtotagit piilotetaan (r. 54-55).

4c. jquery-image-viewer-xhtml.js

jQueryllä rakennetun kuva-viewerin toiminnot: kuvan liikuttaminen ja zoomaus, alkuperäisen koon palauttaminen, koko sivun näyttö ja siitä poistuminen, metadatavalikon avaaminen ja sulkeminen; kuvassa ja transkriptiossa eteen- ja taaksepäin siirtyminen.

1. Popup-kommentit, r. 7-34

Loopataan comments.js läpi avain (= kommentoitava käsite) / arvo (= selitys)-pareilla (r. 7). Haetaan kunkin avaimen ensimmäisen esiintymän indeksi transkriptiossa (r. 9) ja poimitaan merkkijono avaimen indeksistä seuraavaan välilyöntiin (r. 11-13). Jos avainta seuraa välimerkki, se liitetään merkkijonoon (r. 16 – 26). Lopuksi merkkijonon paikalle sijoitetaan popup-kommentin a-elementti ja sen sisälle avainta vastaava arvo span-elementin sisälle (r. 29-32)

2. Kuvissa/sivuilla siirtyminen, r. 38-110

Käyttäjän tullessa kirjeen sivulle näkyvillä on kirjeen ensimmäinen kuva (.pic), joten aluksi piilotetaan muut kuvat (r. 38). Asetetaan muuttujat sivujen (.page) ja kuvien järjestysluvun laskemiseksi, alkuarvona 0. (r. 41-42).

Seuraava-painiketta painaessa (r. 48-75) estetään aluksi painikkeen painaminen, jos ollaan viimeisen sivun kohdalla (r. 50-52). Haetaan nykyinen ja seuraava sivu ja kuva (r. 59-62). Jos ei olla jo viimeisellä sivulla (r. 64), näytetään seuraava sivu ja piilotetaan muut (r. 66). Jos seuraava sivu ei liity samaan kuvaan kuin nykyinen (= kuvassa ei ole kaksi sivua yhdellä aukeamalla), näytetään seuraava sivu, piilotetaan muut (r. 70-71) ja kasvatetaan kuvan järjestysnumeroa yhdellä (r. 66). Lopuksi kasvatetaan sivun järjestysnumeroa yhdellä (r. 73).

Edellinen-painiketta painaessa (r. 83-113) estetään painikkeen toiminta, jos ollaan valmiiksi ensimmäisessä kuvassa (r. 85-88). Haetaan nykyinen ja edellinen sivu ja kuva (r. 94-97). Jos ei olla ensimmäisellä sivulla (r. 99), näytetään edellinen kuva ja piilotetaan muut (r. 101). Jos edellinen ja nykyinen sivu eivät liity samaan kuvaan (= kuvassa ei ole kaksi sivua yhdellä aukeamalla), näytetään edellinen kuva, piilotetaan muut (r. 106-107) ja pienennetään kuvan järjestyslukua yhdellä (r. 109). Lopulta pienennetään sivun järjestyslukua yhdellä (r. 113).

3. Kuvien zoomaus

Zoomaustoiminto on sidottu kuvakehykseen (#picframe, r. 129) ja toimii hiiren pyörällä (jquery.mousewheel.min.js). Kullakin askeleella ohjelma laskee vaaka- ja pystysuuntaiset koordinaatit kuvakehyksessä ja kuvan päällä, määrittää uuden koon ja uudet koordinaatit muuttuneessa koossa sekä piirtää kuvan uudelleen uudessa koossa ja uusissa koordinaateissa siten, että zoomaus tapahtuu hiiren osoittimen kohdasta ulos- tai sisäänpäin. Kuvaa voi liikuttaa missä tahansa koossa.

Ennen ensimmäistä zoomausta/kuvan liikuttamista kuva on kehyksen keskikohdassa. Tarvittavat muuttujat:

r. 119: scale; kuvan skaala, lähtöarvo 1
r. 120-121: xLast, yLast; osoittimen viimeisimmät vaaka- ja pystysuuntaiset koordinaatit ruudulla = kuvakehyksessä, lähtöarvo 0
r. 122-123: xImage, yImage; osoittimen vaaka- ja pystysuuntaiset koordinaatit kuvan päällä, lähtöarvo 0
r. 124: xScreen, yScreen; tämänhetkiset koordinaatit ruudulla = kuvakehyksessä

Koska kuvakehys (#picframe) ei ole kiinni HTML body-elementin vasemmassa/ylälaidassa, osoittimen koordinaatit kuvakehyksessä eivät ole sama asia kuin osoittimen koordinaatit body-elementillä (e.pageX, e.pageY).

Tämänhetkiset koordinaatit kehyksessä saadaan vähentämällä e.pageX- ja e.pageY-koordinaateista kuvakehyksen reunan etäisyys vasemmasta ($(this).offset().left) ja ylälaidasta ($(this).offset().top (r. 135-136).
Jos koko sivun näyttö on päällä (r. 130-134), kuvakehyksen vasen reuna on sama kuin body-elementin vasen reuna. Tällöin xScreen-koordinaatti lasketaan vähentämällä kuvakehyksen puolikkaan ja kuvan puolikkaan summa e.pageX-koordinaatista (r. 132).

Seuraavaksi on löydettävä tämänhetkiset koordinaatit kuvan päällä kuvan tämänhetkisessä koossa. Nämä arvot saadaan lisäämällä koordinaattien edellisiin arvoihin (xImage, yImage) kehyksen tämänhetkisten ja edellisten koordinaattien erotus jaettuna kuvan skaalalla (r. 140-141).

Yksi askel suurentaa tai pienentää kuvaa 20% (r. 144-148). Kuvaa ei voi pienentää lähtöarvoa pienemmäksi eikä suurentaa yli 20-kertaiseksi (r. 149).

Uudet koordinaatit kehyksessä kuvan uudessa koossa saadaan jakamalla tämänhetkisten kehyksen koordinaattien ja kuvan koordinaattien erotus kuvan skaalalla (r. 152-153).

Tämänhetkiset kehyksen koordinaatit tallennetaan xLast- ja yLast-muuttujiin. Lopuksi CSS:n *transform* piirtää kuvan uudelleen (r. 160-163). *transform: scale* kasvattaa tai pienentää kuvan uuteen skaalaan; *transform: translate* siirtää kuvan uusiin koordinaatteihin. *transform: transform-origin* määrittää pisteen, josta kuva siirretään uusiin koordinaatteihin.

4. Kuvien siirtäminen

Kuvien siirtämistoiminto saadaan *jquery-ui.js*-kirjaston draggable()-metodista (r. 168-178).

5. Kuvaviewerin painikkeet

r. 181-188: Palauttamispainike (#origSize) palauttaa kuvan alkukokoon ja siirtää sen takaisin kuvakehyksen keskelle.

r. 192-207: Koko sivun näkymä-painike (#fullScreen) muuttaa kuvakehyksen koko näytön kokoiseksi lisäämällä kuvakehys-elementtiin luokan 'fullscreen' (r. 193), jonka ominaisuudet on määritelty CSS-tiedostossa. Koko sivun näkymässä sen painike on piilotettu ja koko sivun näkymästä poistumisen painike on näkyvillä (r. 194-195). Metadata-paneelin painike (r. 196) ja metadatapaneeli (r. 198-199) on piilotettu. Koko sivun näkymään siirtyminen muuttaa kuvakehyksen mittoja, minkä vuoksi ennen toimintoa lasketut skaala- ja koordinaattiarvot eivät enää päde. Tämän vuoksi koko sivun näkymään siirtyminen palauttaa kuvan lähtöarvot (r. 201-206).

r. 211-223: koko sivun näkymästä poistumisen painike palauttaa kuvakehyksen koon poistamalla elementistä fullscreen-luokan (r. 212), piilottaa painikkeen, näyttää koko sivun näkymä- ja metadatapaneelin painikkeet (r. 213-215) sekä nollaa kuvan skaala- ja koordinaattiarvot (r. 217-222).

r. 226-232: metadatapaanelin (#infopanel) näyttäminen ja piilottaminen tapahtyy jQueryn slideDown()- ja slideUp()-metodeilla.

4d. uv-image-viewer-xhtml.js

Tämän skriptin koodi määrittelee UniversalViewerin rinnalla näytettävän tekstikehyksen (#exhibit3b) toiminnot. Pieniä poikkeuksia lukuunottamatta ne ovat täysin samat kuin jQueryllä rakennetun viewerin kohdalla: popup-kommenttien luominen sekä siirtyminen eteen- ja taaksepäin kuvissa/tekstissä. UniversalViewerin koodia ei ole muutettu lukuunottamatta sitä, että kuvan omat siirtymispainikkeet on piilotettu.

r. 6-34: popup-kommenttien luominen tapahtuu samoin kuin jQuery-viewerin kohdalla. Ohjelma noutaa kommentoitavat kohdat comments.js-tiedostosta avain/arvo-pareina ja sijoittaa avaimen ensimmäisen esiintymiskerran kohdalle popup-elementit (a + span).

r. 40 – 81: kuvissa/sivuilla takaisinpäin siirtyminen tapahtuu samoin kuin jQuery-viewerin kohdalla, mutta koodi näyttää/piilottaa ainoastaan tekstin sivut halutulla tavalla ja laukaisee UV:n takaisin-painikkeen oikeissa kohdissa.

r. 74-107: kuvissa/sivuilla eteenpäin siirtyminen samoin kuin edellä, paitsi sivujen näyttäminen/piilottaminen päinvastoin ja UV:n eteenpäin-painikkeen laukaiseminen oikeissa kohdissa.

4e. toggles-xhtml.js

Tämän skriptin koodi määrittelee transkriptioteksteihin liittyvien checkboxien toiminnot, joilla näytetään/piilotetaan tekstin merkinnät (alleviivaukset, poistot, lisäykset) ja popup-kommentit. Toiminnot ovat identtiset jQuerylla rakennetun viewerin ja UniversalViewerin kohdalla.

Poistettujen kohtien (selaimessa yliviivaus ja punainen teksti) elementeillä on luokka 'del' (r. 9. 18). Lisättyjen kohtien (selaimessa superscript ja sininen teksti) luokka on 'sup' (r. 20-29) ja alleviivattujen kohtien 'underline' (r. 31-39). Merkintöjen näyttäminen ja piilottaminen tapahtuu elementtien css-määrittelyjä manipuloimalla. Merkintä on näkyvillä, kun elementillä on tiettyyn merkintään liittyvät css-määrittelyt (esim. poistoissa style.textDecoration = "line-through" ja style.color = "#f22b0e"). Merkintä on piilotettu, kun nämä css-määrittelyt on muutettu vastaamaan ympäröivän tekstin määrittelyjä (esim. poistoissa style.textDecoration = "none" ja style.color = "#292929").

Popup-kommenttien näyttäminen/piilottaminen (r. 42-72) tapahtuu elementtien luokkia manipuloimalla. Popup-kommenteilla (a-elementtejä) on apuluokka 'comm', jonka avulla elementit voidaan löytää riippumatta siitä, onko niillä popup-luokat ('tooltip bt') vai ei.

r. 47-53: classExists() on apumetodi, joka hakee popup-kommentin 'comm'-luokan avulla ja tarkistaa, onko elementillä luokat 'tooltip bt' eli onko popup-kommentti näkyvillä.

r. 56-63: jos elementillä on luokat 'tooltip bt', koodi poistaa ne (r. 58), piilottaa a-linkkien tyylimäärittelyt (r. 59), estää tyylimuutokset kun hiiren osoitin viedään linkin päälle (hover(), r. 60) ja piilottaa popup-kommentin tekstin sisältävän span-elementin a-elementin sisällä (r. 61).

r. 63-70: jos elementillä ei ole luokkia 'tooltip bt', koodi lisää luokat (r. 65) ja palauttaa elementin alkuperäiset tyylimäärittelyt.

5. Solr-haku

/plugins/SolrSearch/helpers/SolrSearch_Helpers_Index.php
/plugins/SolrSearch/controllers/ResultsController.php
/plugins/SolrSearch/views/shared/results/index.php

Ensimmäisen tiedoston koodiin tehdyt lisäykset mahdollistavat TEI-tiedostojen tekstisisällön sekä muutaman yksittäisen kentän (puuttuvat Omekan Dublin Core-metadatasta) tuominen indeksoitavaksi, jotta tekstihaku voidaan ulottaa niihin. Toisen tiedoston lisäykset liittyvät tiettyjen erikoismerkkien tunnistamiseen hauissa. Kolmannen tiedoston muutokset vaikuttavat hakutuloksissa näkyviin tietoihin.

1. Indeksoiminen (SolrSearch_Helpers_Index.php)

Muutoksia on tehty riveillä 51-178 olevaan koodiin.

r. 51-89: indexItem() hakee jokaisesta Omekaan tallennetusta metadatakentät (r. 54-55) sekä muodostaa ($doc->setMultiValue() indeksoitaviksi valittujen kenttien arvoista indeksoitavat (r. 77-79, $field->indexKey()) ja faceteiksi valittujen kenttien arvoista facet-kentät (r. 83-87, $field->facetKey()).

Tässä kohtaa myös TEI-tiedoston tekstiosuus on saatava indeksoitumaan. Tämä onnistuu siten, että tiedosto noudetaan merkkijonoksi PHP:n file_get_contents()-metodilla (r. 59), tarpeettomat teiHeader- ja opener-elementit leikataan pois (r. 61-63) ja jäljelläoleva merkkijono sijoitetaan TEI-tiedoston url-osoitteen sisältävän metadata-kentän alkuperäisen arvon paikalle (r. 71).

Lisäksi TEI-tiedostosta halutaan noutaa facetiksi kirjeen kirjoituspaikan sisältävä kenttä. Tämä onnistuu PHP:n simplexml_load_file()-metodilla, joka tuo TEI-tiedoston objektiksi (r. 65). Noudetun kentän arvo liitetään TEI-tiedoston url-osoitteen sisältävään metadata-kenttään (r. 73). Kenttä on valittu pluginin käyttöliittymässä facetoitavaksi. Kentän kohdalla asetetaan facetiksi pelkästään kirjoituspaikkatieto, ei tekstiosuutta (r. 84).

r. 101-178: itemToDocument() muodostaa jokaisesta Omekan itemistä SolrSeach-dokumentin.

Kirjeiden kirjoituspaikka halutaan vielä liittää SolrSearch-dokumenttiin erilliseksi indeksoitavaksi kentäksi, jotta tekstihaun voi kohdistaa siihen. TEI-tiedosto noudetaan uudestaan objektiksi (r. 125), muodostetaan uusi SolrSearchField()-objekti (r. 128), merkitään se indeksoitavaksi (r. 131) ja liitetään SolrSearch-dokumenttiin (r. 134).

TEI-tiedostoista poimitaan vielä indeksoitavaksi kirjeiden vastaanottajan suku- ja etunimikentät. Objektiksi tuodusta TEI-tiedostosta haetaan tarvittavat kentät (r. 143-144) ja niille luodaan SolrSearchField()-objekti saman kaavan mukaan kuin kirjoituspaikan kohdalla (r. 137-146)

2. Erikoismerkit tekstihaussa (ResultsController.php)

Lönnrot käyttää kirjeissään vaihdellen tiettyjä erikoismerkkejä tavallisten kirjeiden sijasta: esim. ç vs c, æ vs. ae ja én-loppuiset erisnimet aksentilla tai ilman. Haun on löydettävä molemmat versiot: jos käyttäjä hakee esim. sanaa 'lexicon', haun on löydettävä sekä 'lexicon' että 'lexiçon'.

Ongelma ratkeaa muodostamalla yhdistelmähakuja (OR) tapauksissa, jossa haettava merkkijono saattaa esiintyä myös erikoismerkin kanssa (r. 123-135). Samanlainen muotoilu on tehtävä erikseen jokaisen erikoismerkin kohdalla.

Esimerkki (r. 123-126): æ vs. ae

1. Käyttäjä hakee 'Europaeus' -> $query = 'Europaeus'
2. Europaeus esiintyy Lönnrotilla myös erikoismerkillä -> $query1 = alkuperäinen $query, jossa ae on korvattu merkkiä æ vastaavalla unicode-koodilla \u00E6
3. Muodostetaan yhdistelmähaku, jossa sanan molemmat versiot -> $query = 'Europaeus OR Europaæus'

Kun käyttäjä rajoittaa hakutuloksia klikkaamalla hakutulossivulla jotakin facetia, valittu facet liitetään kyselyyn, esim.

$query = 'Elmgren', $facet = 'language' -> $query = 'Elmgren AND language'

Jos muuttujan $query arvo on jotakin erikoismerkkiä varten muodostettu OR-haku, facetin kanssa kyselystä tulee seuraavanlainen:

$query = 'Castren OR Castrén', $facet = 'language' -> $query = 'Castren OR Castrén AND language'.

Tällöin haku ei osaa käsitellä OR- ja AND-määreitä oikein. Ongelma ratkeaa lisäämällä $query-muuttujan ympärille sulkeet, tuloksena '(Castren OR Castrén) AND language' (r. 153).

3. Hakutulosnäkymä (index.php)

Kaikki näkymät otsakkeet on muutettu suomenkielisiksi ja tiettyjä otsikkoja muutettu alkamaan isolla alkukirjaimella. Hakutulosten asettelua on muutettu: jokaiseen löydettyyn kirjeeseen liittyvät tiedostot muutettu näkymään vaakatasossa pystysuuntaisen asettelun sijaan.

r. 70-112: hakutulosten rajaamiseen käytettävät facetit näkyvät hakutulossivulla vasemmalla. Faceteiksi on valittu neljä kenttää: kokoelma, laji, kieli ja kirjoituspaikka. Lajilla on kolme mahdollista arvoa ja ne on määritelty TEI-tiedostojen tekstiosuuden sisältävän DIV-elementin attribuuttina: kirje, kirjekonsepti ja merkinta_konseptikirjassa. Kirjoituspaikka on varsinaisesti TEI-tiedoston url-osoitteen sisältävä kenttä, josta on muutettu otsikko (r. 80) ja määritetty indeksointivaiheessa facetiksi kirjoituspaikka-arvot (ks. yllä). Facet-listaus näyttää kunkin facetin otsikon sekä sen alla hakutuloksissa esiintyvät arvot.

Esimerkiksi kieli-facetin listaus voisi olla hakutuloksessa seuraava:

KIELI
suomi (12)
ruotsi (16)
ruotsi saksa (3)

r. 116-189: SolrSearch-pluginin default-asetuksissa jokaisessa hakutuloksessa näkyy otsikko klikattavana linkkinä; tietyn mittainen tekstipätkä, jossa löydetty merkkijono maalattuna; sekä itemiin liittyvät kuvat ja muut tiedostot thumbnail-kuvina.

Kirjeiden otsikot ovat saman kaavan mukaisia (Elias Lönnrot > Frans Johan Rabbe, Elias Lönnrot > Matthias Alexander Castrén). Jotta hakutulokset erottuisivat paremmin toisistaan, hakutulosten otsakkeisiin lisätään kirjeen numero ja kirjoituspäivä muodossa p.k.vvvv (r. 139-140).

Default-asetuksissa hakutuloslistaus ei erottele, mistä kentästä haettava merkkijono on löytynyt, esim:

* *Kajaani*
* terveisiä *Kajaani*sta, hyvä Weli

Tämän korjaamiseksi kenttien otsakkeet on saatava näkyville. Tapaa, jolla koodi tuo hakutulosobjektit (r. 154) muutetaan siten, että tuodaan pelkkien attribuuttien arvojen (alkup. foreach($results->highlighting->{$doc->id} as $field) sekä attribuuttien nimet että arvot ($results->highlighting->{$doc->id as $prop=>$field}. Indeksointivaiheessa (ks. yllä) indeksoitavat kentät on identifioitu avainkoodeilla, jotka on korvattava selkokielisillä otsikoilla, joista on oltava suomen- ja ruotsinkieliset versiot. Tähän tarkoitukseen käytetään rivien 168-196 switch-rakennetta, jossa kukin koodi korvataan kielitiedostojen avulla (ks. kappale 2) käännetyllä otsikolla. Kenttien koodit saattavat muuttua Solrin asennuksesta toiseen. Jos hakutuloksissa näkyy jossakin kentässä suomenkielisen sijasta koodi, koodi on tarkistettava Solrin hallintapaneelin kautta ja korjattava oikeaan kohtaan koodissa. Kentän otsikko sijoitetaan hakutuloksen eteen, esim.

* **Kirjoituspaikka:** *Kajaani*
* **Teksti:** terveisiä *Kajaani*sta, hyvä Weli

6. Omekan laajennettu haku

/application/views/scripts/items/search-form.php
/themes/default/common/header.php

Omekan default-teemassa laajennetulla haulla on oma sivunsa ja siihen pääsee erillisestä tabista itemien selaussivun kautta. Tässä rakennetta on muutettu niin, että hakukaavake on sijoitettu yläpaneeliin dropdown-valikkoon ja kaavakkeessa näkyviä kenttiä on karsittu ja muokattu. Kaavakke rakennetaan tiedostossa search-form.php ja ladataan div-elementin sisällä header.php:n riveillä 135-139. Hakutulokset näkyvät omalla sivullaan ja listausta määrittelevät samat asetukset kuin muiden item-listauksien kohdalla (ks. kappale 3)

Alkuperäisestä kaavakkeesta on poistettu haku ID:eillä, kokoelman nimellä, itemin lajilla, käyttäjän nimellä, julkisuusasetuksilla ja featured/non-featured-asetuksilla (https://github.com/omeka/Omeka/blob/master/application/views/scripts/items/search-form.php, r. 108-)

Kaavakkeeseen on jätetty kaksi dropdown-valikkoa sekä tekstikenttä hakusanoille. Edellisistä ensimmäinen sisältää hakukentät (search-form.php, r. 26-38, 53-62). Vakioasetuksissa valikossa on kaikki Omekan metadatakentät. Tässä versiossa on poistettu tarpeettomat kentät (r. 31-33) ja jätetty

jäljelle vastaanottaja (vars. kirjeen otsikko, josta löytyy vastaanottajatieto; nimi muutettu), kirjoittaja, kirjoitusvuosi, kieli (vars. aikamääre = kirjoitusaika; nimi muutettu) ja identifiointitunnus. Toinen dropdown-valikko sisältää hakumääreet: sisältää, ei sisällä, on täsmälleen, on tyhjä, ei ole tyhjä (r. 40-48, 63-72). Oletusarvoksi on asetettu "sisältää" (/themes/default/javascripts/header_menus.js, r. 28). Tämä johtuu siitä, että Omekan metadatakenttien arvot ovat merkkijonoja. Jos käyttäjä hakee esimerkiksi vuosilukua muodossa VVVV mutta valitsee hakumääreeksi "on täsmälleen", haku ei palauta mitään, koska aikamäärekentän arvot ovat muotoa VVVV-KK-PP eivätkä siis ole täsmälleen VVVV.

7. Tyylitiedostot (XSLT, CSS)

/files/original/TEI-to-HTML.xsl
/themes/default/css/style.css

TEI-tiedostot käännetään (ks. s. 3, 3e) yksinkertaisella XSL-tyylitiedostolla ja elementtien tyyliasetukset määritellään CSS-tyylitiedostossa.

1. TEI-to-HTML.xsl

r. 16-22: tei:lg ja tei:l – käytetään esittämään runosäkeitä; lg merkitsee yhtä runosäkeistöä ja käännetään p-elementiksi; runosäkeet säkeistön sisällä merkitään l-tagilla j käännetään span-elementeiksi, joiden lopussa on rivinvaihto.

r. 24-26: tei:opener – kirjeen kirjoituspaikka ja –aika; käännetään p-elementiksi, piilotettu CSS-tiedostossa

r. 32-34: tei:pb – sivunvaihtotagit; kussakin tagissa järjestysnumeron n-attribuutissa ja sivua vastaavan kuvan nimi facs-attribuutissa; pb-tagit käännetään span-elementeiksi, joille annetaan luokkanimeksi pb + kuvan nimi yhdistettynä tagin järjestysnumeroon

r. 36-46: tei:add, tei:hi, tei:del – lisäykset, alleviivaukset, poistot tekstissä; käännetään span-elementeiksi omine luokkanimineen

r. 48-54: tei:unclear – tekstin epäselvät kohdat, joihin transkriboijalla on esittää arvaus tulkinnasta; käännetään span-elementiksi luokkanimellä unclear; span-elementin sisään epäselvän kohdan ympärille lisätään kaarisulkeet

r. 56-61: tei:app – vaihtoehtoiset luennat (esim. vaihtoehtoiset sanajärjestykset), käyttö <app><lem>vaihtoehto1</lem><rdg>vaihtoehto2</rdg></app>; käännetään samanlaisiksi popup-kommenteiksi kuin tietosanakirjatyyppisten kommenttien kohdalla (ks. 4c, 1. ja 4d)

r. 63-68: tei:ref – tekstin sisäiset kommentit, käyttö <ref>erikoismerkki<note>Lönnrot käyttää erikoismerkkiä</note></ref>; käännetään popup-kommenteiksi kuten yllä

r.70-86: tei:table – taulukkorakenne, käyttö:

```
<table>
  <row>
    <cell>…</cell>
```

```
    </row>
</table>
```

Käännetään HTML:n vastaavaksi taulukkorakenteeksi

r. 88-90: tei:gap – tyhjät kohdat tekstissä; käännetään span-elementiksi, joka näkyy harmaana palkkina

r. 92-99: tei:rs – referenssi-tagi; käytetään tapauksissa, joissa on kirje ja kirjeen konsepti. Tällöin konseptissa on tekstitranskription sijasta viittaus kirjeeseen, jossa rs-tagitetun kirjeen identifiointitunnuksen kohdalle sijoitetaan a-linkki, joka hakee kirjeen identifiointitunnuksen perusteella.

2. style.css

Tyylitiedosto määrittelee kaikki Omekan tyyliasetukset. Vakiona tulevassa style.css-tiedostossa on tehty pieniä muutoksia (fontti, sivun leveys, värit ym.) valmiisiin asetuksiin sekä lisätty seuraavat:

r. 2375-2460: jQueryllä rakennettuun kuvavieweriin liittyvien elementtien tyyliasetukset

r. 2484-2512: UniversalViewerin rinnalla näkyvän tekstikehyksen tyyliasetukset

r. 2497-2509: tekstin poistoja, lisäyksiä ja alleviivauksia merkkaavat span-elementit

r. 2516-2596: popup-kommenttien tyyli- ja toimintoasetukset

r. 2604-2632: etusivun tyyliasetuksia

r. 2636-2676: yläpaneelin linkkien tyyliasetukset