Lönnrotin kirjeiden digitaalinen julkaisu, julkaisualustan dokumentaatio

Maria Niku 02/2017

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
Kuvaviewer facsimile-kuvien näyttämiseen

-LanguageSwitcher, https://gitlab.com/TIME_LAS/Omeka_Plugin_SwitchLang/
Plugin käyttöliittymän kielen vaihtamiseen (suomi/ruotsi/englanti)

Käytettävä teema: Lönnrot (vaaleanvihreä header ilman taustakuvaa, etusivun tunnuskuva sinetti).

Selaus-/hakutulosten TEI-tiedostojen lataaminen zip-pakettina (Omekan browse-toiminnot ja Solr-haku): PHP:n ZipArchive(), vaatii PHP:n zip-lisäosan.

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"
/plugins/SwitchLanguage/languages/fi_FI.po, sv_SE.po

Pohjakieli on englanti, joka käännetään suomeksi ja ruotsiksi. Käännöstä vaativat merkkijonot vieweissä 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/[theme name]/css/style.css.

3a. Yläpaneeli

/themes/[theme name]/common/header.php, r. 108-141

r. 110-117: Julkaisun nimi, kielivalikko, SKS:n logo (sis. linkki SKS:n sivuille)
r. 122-130: linkit – selaa kirjeitä; selaa vastaanottajia (kokoelmaluettelo), tietosivu/-valikko (esipuhe, muita tietoja); ohjeita -> r. 142-146 dropdown, transkriptioiden merkinnät (SimplePages-sivu); laajennettu haku -> r. 137-140 hakukaavake-dropdown; r. 133: tekstihakukenttä (Solr)

3b. Etusivu

/themes/[theme name]/index.php

r. 3-26 (vasen palsta): julkaisun tunnuskuva (r. 4) esittelyteksti (asetetaan hallintapaneelissa); piilotettu featured item, feature collection ja featured exhibit (asetetaan hallintapaneelissa)

3c. Item-listaukset

Omeka käyttää samoja browse-toimintoja itemien listaukseen selaussivuilla, kokoelmien alla ja Omeka-haun hakutuloksissa. Tehdyt muutokset vaikuttavat siis näihin kaikkiin. Kuvailtujen muutosten lisäksi poistettu turhia otsikoita, muutettu otsikoiden kokoja jne.

/application/controllers/ItemsController.php, r. 187-189, _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, vastaanottaja (vars. otsikkokenttä (default otsikko, kirjoittaja, lisäysaika)

r. 33-36: painike selaustulosten (yleislistaus, kokoelmalistaukset, Omeka-haun tulokset) itemien TEI-tiedostojen lataamiseen zip-pakettina.

→ controller-koodi: application/libraries/Omeka/Controller/AbstractActionController.php

Defaultina viewiin lähetetään valmiiksi sivutetut hakutulokset (r. 123-138). Listaus näyttää 10 tulosta/sivu, jolloin zip-pakettiin tulisi vain senhetkisen sivun itemien TEI-tiedostot ja suuremmissa hakutuloksissa käyttäjä joutuisi lataamaan kunkin sivun tiedostot erikseen.

Tämän vuoksi noudetaan sivuttamattomat hakutulokset ($allRecords, r. 143). Koodi luo instanssin PHP:n ZipArchive-luokasta (r. 147), avaa väliaikaisen zip-tiedoston palvelimen tmp -kansioon (sys_get_temp_dir(), r. 149), kopioi selaustulosten TEI-tiedostot tmp-kansioon (r. 152-155) ja lisää kopiot zip-pakettiin (addFile(), r. 158-160). header() (r. 165-167) ja readfile() (r. 170) pakottavat zip-tiedoston latauksen. Lopuksi koodi poistaa väliaikaistiedostot (r. 171-172).

r. 49: lisätty ilmoitus, mikäli haku-/selaustulos tyhjä.

r. 61-: lisätty listausnäkymään kirjoitusaika, muodossa pp.kk.vvvv (tuodaan muodossa vvvv-kk-pp)

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. 27-:
switch-lauseke kokoelmien otsakkeiden käännöksille; tarvitaan, koska otsakkeiden arvot tulevat dynaamisesti

r. 42: Kokoelman otsikon linkki muutettu osoittamaan kokoelmaan kuuluvien kirjeiden listaukseen, link_to_items_browse(…) (default linkki erilliselle kokoelman esittelysivulle, link_to_collection())

r. 44-48: lisätty kokoelman kirjeiden määrä otsakkeen perään sulkeisiin.

r. 51-54: thumbnailista poistettu linkki erilliselle kokoelman esittelysivulle (link_to_collection())

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

3e. Itemien näyttösivu

/themes/[theme name]/items/show.php

Sivulla ladattavat skriptit (r. 3 – 8): /themes/[theme name]/javascripts (ks. 4. Skriptit)

Kuvien näyttämiseen käytetään UniversalVieweriä (r. 19-61). Viewerin oikealla puolella näytetään kuvakohtainen transkriptioteksti skrollattavassa divissä (r. 28-60) ja tämän päällä nuolipainikkeet kuvissa/sivuilla (r. 38-41), checkboxit merkintöjen ja popup-kommenttien /piilottamiseen (r. 31-32) sekä latauslinkki TEI-tiedostolle (r. 33-35, HTML5:n download-attribuutti).

Transkriptioteksti ladatataan div-kehykseen käyttäen PHP:n XSLTProcessor()-luokkaa (r. 43-59). DOMDocument()-luokan instanssi lataa itemiin liittyvän xml-tiedoston (r. 49-50), toinen instanssi lataa xsl-tyylitiedoston (r. 51-52) ja transformToXML()-metodi kääntä xml:n xhtml:ksi (r. 55).

4. Skriptit (javascript)

/themes/[theme name]/javascripts

Skriptit ladataan itemien näyttösivulla (/themes/[theme name]/items/show.php, ks. kappale 3e) lukuunottamatta tiedostoa header_menus.js, joka ladataan headerissa (/themes/[theme name]/common/header.php). Käytössä jqueryn version jquery-1.12.4.min.js

4a. page-formatting-xhtml.js

Jakaa xhtml:ksi käännetyn kirjeen transkriptiotekstin sivuihin itemien näyttösivulla (/themes/[theme name]/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. Toiminnot tarvitaan v. 2013 tehtyihin TEI-tiedostoihin. v. 2016-2017 tehdyissä TEI-tiedostoissa päädyttiin sulkemaan ja avaamaan kappaleet sivunvaihtotageja ennen ja niiden jälkeen.

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. 11-24

Sivunvaihtotagi tallennetaan child-muuttujaan (r. 14).

p-elementin alkuperäinen html-koodi jaetaan kahtia sivunvaihtotagin kohdalla ja saadut kaksi osaa tallennetaan content-arrayhin (r. 17).

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. 21-23).

3. p-kappaleen sisällä useampia sivunvaihtotageja, r. 27-67

Ensiksi luodaan tyhjä div-elementti välivaiheiden säilyttämistä varten (r. 30).

Halutaan löytää sivunvaihtotagien välinen sisältö.

Jokaisen p-elementin sisällä olevan sivunvaihtotagin kohdalla (r. 33) haetaan thisIndex-muuttujaan p-elementin html:n kohta, jossa sivunvaihtotagin sulkutagi päättyy (r. 36).

Jos käsittelyssä oleva sivunvaihtotagi ei ole p-elementin viimeinen (r. 39), seuraavan sivunvaihtotagin (r. 41) kohta p-elementin sisällä tallennetaan nextIndex-muuttujaan (43). thisIndex- ja nextIndex-muuttujien välinen sisältö haetaan substring()-metodilla (r. 45) ja sivunvaihtotagi sekä noudettu sisältö käärittynä p-tageihin liitetään aikaisemmin luotuun tyhjään div-elementtiin (r. 47-49).

Jos käsittelyssä oleva sivunvaihtotagi on p-elementin viimeinen (r. 51), noudetaan substring()-metodilla thisIndex-muuttujan ja p-elementin lopputagin välinen sisältö (r. 53) ja sivunvaihtotagi sekä noudettu sisältö p-tageihin käärittynä (r. 55-57).

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. 62). p-elementin alkuperäinen html-sisältö korvataan content-arrayn ensimmäisellä elementillä (r. 64) ja placeholder-divin lapsielementit liitetään järjestyksessä tämän perään (r. 66).

4. Sivujako, r. 72-80

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. 83). Myös sivunvaihtotagit piilotetaan (r. 86).

4b. 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:

Kommentoitavat termit ja niiden selitykset on listattu avain/arvo-pareina comments.js-tiedostossa. Koodi hakee avain-/arvoparien kohdalla (r. 6) jokaisen itemin transkriptiosta (#exhibit3b) kohdan, jossa avain esiintyy ensimmäisen kerran, poimii merkkijonon alkukohdasta seuraavan välilyöntiin ottaen huomioon tietyt välimerkit (r. 11-26) ja käärii merkkijonon <a>-tagiin, jonka sisällä kommentin sisältö -tageissa.

r. 40 – 71: kuvissa/sivuilla taaksepäin siirtyminen

Jos ollaan jo ensimmäisellä sivulla, nuolipainikkeen toiminto on estetty (r. 40, 45). Muussa tapauksessa koodi noutaa tämänhetkisen ja edellisen sivun (r. 53-54), piilottaa tämänhetkisen sivun (r. 56), näyttää seuraavan ja piilottaa muut (r. 57-59). Koodi hakee kuvaviewerin takaisinpäin-painikkeen (r. 62) ja laukaisee viewerin painikkeen, jos edellisen sivu ei ole samalle kuvalle kuin tämänhetkinen (kuvassa kaksi sivua yhdellä aukeamalla, r. 66-68). Lopuksi koodi pienentää sivulaskurin lukua yhdellä (r. 70).

r. 74-107: kuvissa/sivuilla eteenpäin siirtyminen

Samoin kuin edellä mutta päinvastaisessa järjestyksessä. Jos ollaan viimeisellä sivulla, nuolipainikkeen toiminto on estetty (r. 77). Muussa tapauksessa koodi hakee nykyisen ja seuraavan sivun (r. 85-86) ja kuvaviewerin seuraava-painikkeen (r. 89), näyttää seuraavan sivun ja piilottaa muut (r. 92-95). Jos seuraava sivu ei ole samalle kuvalle kuin nykyinen, koodi laukaisee viewerin seuraava-painikkeen (r. 100-102). Lopuksi koodi kasvattaa sivulaskurin lukua yhdellä (r. 106).

4c. toggles-xhtml.js

Tämän skriptin koodi määrittelee transkriptioteksteihin liittyvien checkboxien toiminnot, joilla näytetään/piilotetaan tekstin merkinnät (alleviivaukset, poistot, lisäykset) ja popup-kommentit.

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), alleviivattujen 'underline' (r. 31-38), epäselvien 'unclear' (r. 40-47) ja tyhjien kohtien 'gap' (r. 49-56). 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 = "#444444").

Popup-kommenttien näyttäminen/piilottaminen (r. 60-90) 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. 65-71: classExists() on apumetodi, joka hakee popup-kommentin 'comm'-luokan avulla ja tarkistaa, onko elementillä luokat 'tooltip bt' eli onko popup-kommentti näkyvillä.

r. 74-81: jos elementillä on luokat 'tooltip bt', koodi poistaa ne (r. 76), piilottaa a-linkkien tyylimäärittelyt (r. 77), estää tyylimuutokset kun hiiren osoitin viedään linkin päälle (hover(), r. 78) ja piilottaa popup-kommentin tekstin sisältävän span-elementin a-elementin sisällä (r. 79).

r. 81-88: jos elementillä ei ole luokkia 'tooltip bt', koodi lisää luokat (r. 83) ja palauttaa elementin alkuperäiset tyylimäärittelyt (r. 84-).

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 53-150 olevaan koodiin.

r. 53-91: indexItem() hakee jokaisesta Omekaan tallennetusta itemistä metadatakentät (r. 56-57) sekä muodostaa ($doc->setMultiValue() indeksoitaviksi valittujen kenttien arvoista indeksoitavat (r. 79-81, $field->indexKey()) ja faceteiksi valittujen kenttien arvoista facet-kentät (r. 85-89, $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. 61), tarpeettomat teiHeader- ja opener-elementit leikataan pois (r. 63-65) ja jäljelläoleva merkkijono sijoitetaan TEI-tiedoston url-osoitteen sisältävän metadata-kentän alkuperäisen arvon paikalle (r. 73).

Lisäksi TEI-tiedostosta halutaan noutaa facetiksi kirjeen kirjoituspaikan sisältävä kenttä. Tämä onnistuu PHP:n simplexml_load_file()-metodilla, joka tuo TEI-tiedoston objektiksi (r. 67). Noudetun kentän arvo liitetään TEI-tiedoston url-osoitteen sisältävään metadata-kenttään (r. 75). Kenttä on valittu pluginin käyttöliittymässä facetoitavaksi. Kentän kohdalla asetetaan facetiksi pelkästään kirjoituspaikkatieto, ei tekstiosuutta (r. 86).

r. 103-178: 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. 127), muodostetaan uusi SolrSearchField()-objekti (r. 130), merkitään se indeksoitavaksi (r. 132) ja liitetään SolrSearch-dokumenttiin (r. 135).

TEI-tiedostoista poimitaan vielä indeksoitavaksi kirjeiden vastaanottajan suku- ja etunimikentät. Objektiksi tuodusta TEI-tiedostosta haetaan tarvittavat kentät (r. 144-145) ja niille luodaan SolrSearchField()-objekti saman kaavan mukaan kuin kirjoituspaikan kohdalla (r. 138-147)

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. 158-170). Samanlainen muotoilu on tehtävä erikseen jokaisen erikoismerkin kohdalla.

Esimerkki (r. 158-161): 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. 82-145: 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. 109-111) 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. 149-260: 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. 184-186).

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. 201) 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 210-238 switch-lauseketta, 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

4. Hakutulosten TEI-tiedostojen lataus

index.php, r. 154-157: painike zip-paketin luomiseen ja lataamiseen
ResultsController, r. 77-110

Koodi toimii samoin kuin Omekan browse-toiminnoissa (ks. yllä, 3c) sillä erotuksella, että hakutulokset ovat SolrSearch-dokumentteja, joiden modelid-attribuutilla pitää ensin hakea oikea item ja tämän jälkeen itemin TEI-tiedoston osoitteen sisältävä kenttä (r. 88-89).

6. Omekan laajennettu haku

/application/views/scripts/items/search-form.php
/themes/[theme name]/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. 30-39, 54-63). Vakioasetuksissa valikossa on kaikki Omekan metadatakentät. Tässä versiossa on poistettu tarpeettomat kentät (r. 34-36) 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. 41-47, 64-73). Oletusarvoksi on asetettu "sisältää" (/themes/[theme name]/javascripts/header_menus.js, r. 8-10). 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 aikamääre-kentän arvot ovat muotoa VVVV-KK-PP eivätkä siis ole täsmälleen VVVV.

search-form.php, r. 7-8: Omekassa haussa korvataan hakukaavakkeen action-attribuutin arvo käsin laajennetun haun osoitteella. Jos käyttäjä tekee ensin Solr-haun, action-attribuutin arvoksi jää Solr-haun osoite eikä haku toimi.

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