assignFAST: An Autosuggest-Based Tool for FAST Subject Assignment

Rick Bennett,
Edward T. O'Neill,
and Kerre Kammerer

ABSTRACT

Subject assignment is really a three-phase task. The first phase is intellectual—reviewing the material and determining its topic. The second phase is more mechanical—identifying the correct subject heading(s). The final phase is retyping or cutting and pasting the heading(s) into the cataloging interface along with any diacritics, and potentially correcting formatting and subfield coding. If authority control is available in the interface, some of these tasks may be automated or partially automated.

A cataloger with a reasonable knowledge of Faceted Application of Subject Terminology (FAST)[1,2] or even Library of Congress Subject Headings (LCSH)[3] can quickly get to the proper heading but usually needs to confirm the final details—was it plural? am I thinking of an alternate form? is it inverted? etc. This often requires consulting the full authority file interface. assignFAST is a web service that consolidates the entire second phase of the manual process of subject assignment for FAST subjects into a single step based on autosuggest technology.

BACKGROUND

Faceted Application of Subject Terminology (FAST) Subject Headings were derived from the Library of Congress Subject Headings (LCSH) with the goal of making the schema easier to understand, control, apply, and use while maintaining the rich vocabulary of the source. The intent was to develop a simplified subject heading schema that could be assigned and used by nonprofessional cataloger or indexers. Faceting makes the task of subject assignment easier. Without the complex rules for combining the separate subdivisions to form an LCSH heading, only the selection of the proper heading is necessary.

The now-familiar autosuggest[4,5] technology is used in web search and other text entry applications to help the user enter data by displaying and allowing the selection of the desired text before typing is complete. This helps with error correction, spelling, and identification of commonly used terminology. Prior discussions of autosuggest functionality in library systems have focused primarily on discovery rather than on cataloging.[6-11]

Rick Bennett (Rick_Bennett@oclc.org) is a Consulting Software Engineer in OCLC Research, **Edward T. O'Neill** (oneill@oclc.org) is a Senior Research Scientist at OCLC Research and project manager for FAST, and **Kerre Kammerer** (kammerer@oclc.org) is a Consulting Software Engineer in OCLC Research, Dublin, Ohio.

The literature often uses synonyms for autosuggest, such as autocomplete or type-ahead. Since assignFAST can lead to terms that are not being typed, autosuggest seems most appropriate and will be used here.

The assignFAST web service combines the simplified subject choice capabilities of FAST with the text selection features of autosuggest technology to create an in-interface subject assignment tool. Much of a full featured search interface for the FAST authorities, such as searchFAST,[12] can be integrated into the subject entry field of a cataloging interface. This eliminates the need to switch screens, cut and paste, and make control character changes that may differ between the authority search interface and the cataloging interface. As a web service, assignFAST can be added to existing cataloging interfaces.

In this paper, the actual operation of assignFAST is described, followed by how the assignFAST web service is connected to an interface, and finally by a description of the web service construction.

assignFAST Operation

An authority record contains the Established Heading, See headings, and control numbers that may be used for linking or other future reference. The relevant fields of the FAST record for **Motion Pictures** are shown here:

Control Number	fst01027285
Established Heading	**Motion pictures**
See	**Cinema**
See	**Feature films -- History and criticism**
See	**Films**
See	**Movies**
See	**Moving-pictures**

In FAST, the facet of each heading is known. **Motion pictures** is a topical heading. The See references are unauthorized forms of the established heading. If someone intended to enter **Cinema** as a subject heading, they would be directed to use the established heading **Motion pictures**.

For a typical workflow, the subject cataloger would need to leave the cataloging interface, search for "cinema" in an authority file interface, find that the established heading was **Motion Pictures**, and return to the cataloging interface to enter the established heading.

The figure below shows the same process when assignFAST is integrated into the cataloging interface. Without leaving the cataloging interface, typing only "cine" shows both the See term that was initially intended and the Established Heading in a selection list.

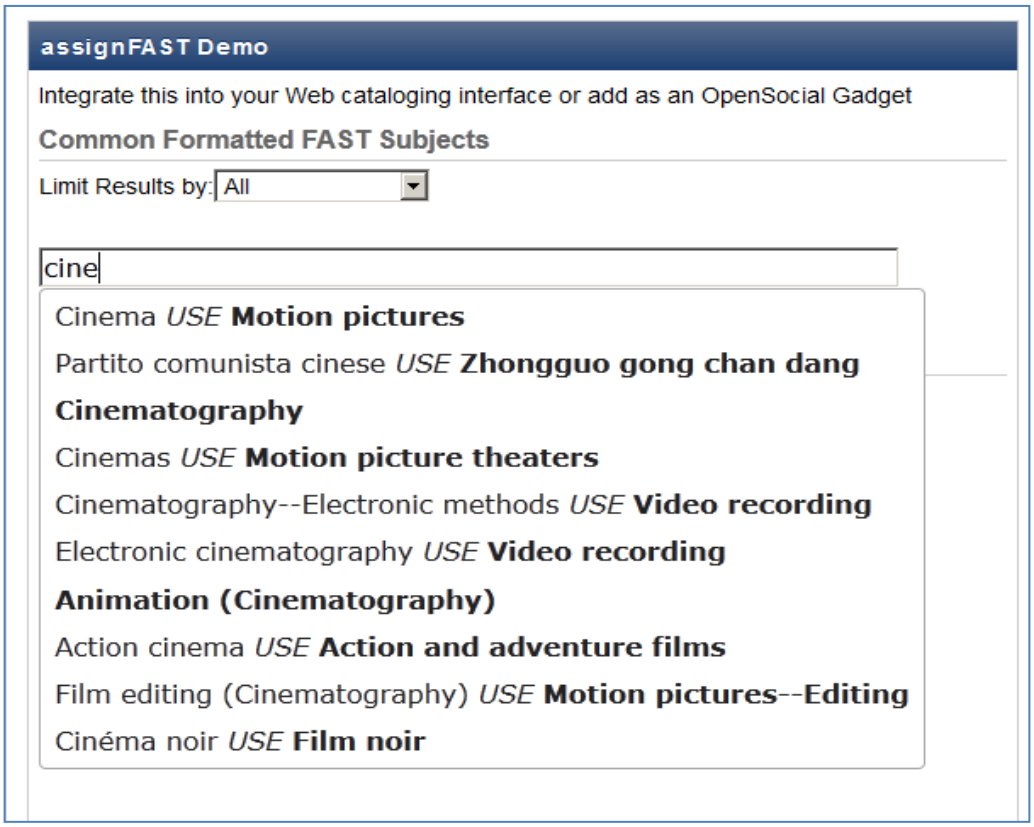

Figure 1. assignFAST typical selection choices. *Selecting "Cinema USE **Motion pictures**" enters the Established term, and the entry process is complete for that subject.*

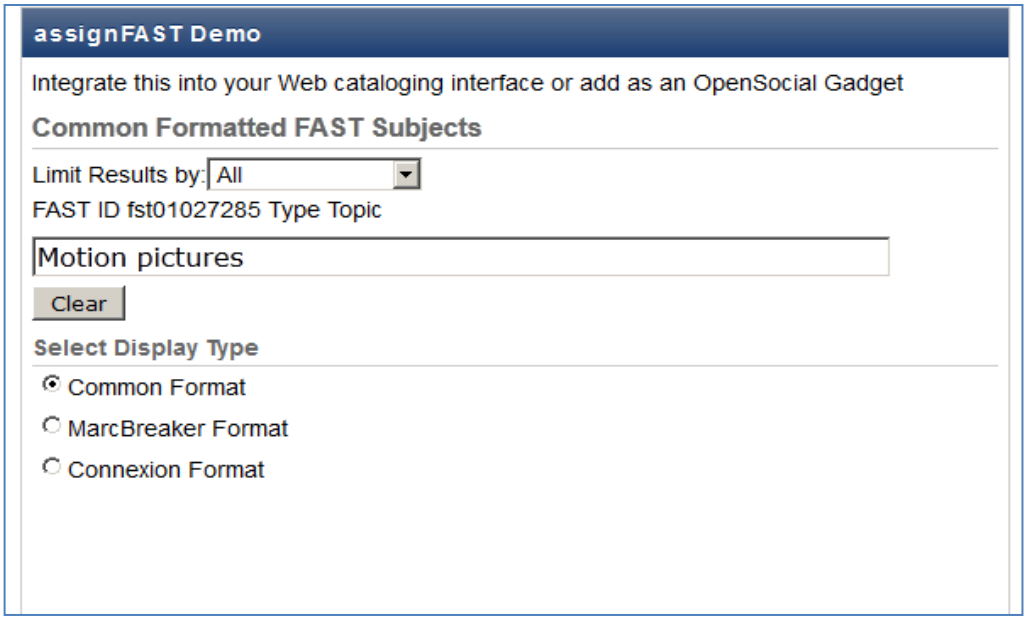

Figure 2. assignFAST selection result. *The text above the entry box provides the fast ID number and facet type.*

As a web service, assignFAST headings can be manipulated by the cataloging interface software after selection and before they are entered into the box. For example, one option available in the assignFAST Demo is MARCBreaker format.[13] MARCBreaker combines MARC field tagging and allows diacritics to be entered using only ASCII characters.

Using MARCBreaker output, assignFAST returns the following for "São Paulo, Brazil":

=651 7$aBrazil$zS{tilde}ao Paulo$0(OCoLC)fst01205761$2fast

In this case, the output includes MARC tagging of 651 (geographic), as well as subfield coding ($z) that identifies the city within Brazil, that it's a FAST heading, and the FAST control number. The information is available in the assignFAST result to fill one or multiple input boxes and to reformat as needed for the particular cataloging interface.

Addition to Web Browser Interfaces

As a web service, assignFAST could be added to any web-connected interface. A simple example is given here to add assignFAST functionality to a web browser interface using JavaScript and jQuery (http://jquery.com). These technologies are commonly used, and other implementation technologies would be similar. Example files for this demo can be found on the OCLC Developers Network under assignFAST.[14]

The example uses the jQuery.autocomplete function.[15] First, the script packages jquery.js, jquery-ui.js, and the style sheet jquery-ui.css are required. Version 1.5.2 of JQuery and version 1.8.7 for jquery-ui was used for this example, but other compatible versions should be fine. These are added to the html in the script and link tags.

```
<script src="http://code.jquery.com/jquery-1.5.2.min.js"></script>
<script src="http://ajax.googleapis.com/ajax/libs/jqueryui/1.8.7/jquery-ui.min.js"
type="text/javascript"></script>
<script src="http://experimental.worldcat.org/fast/assignfast/js/assignFASTComplete.js"
type="text/javascript"></script>
<script src=" http://ajax.googleapis.com/ajax/libs/jqueryui/1.8.7/jquery-ui.min.js"
type="text/javascript" ></script>
<link href="http://ajax.googleapis.com/ajax/libs/jqueryui/1.8.7/themes/base/jquery-ui.css "
rel="stylesheet" type="text/css">
```

The second modification to the cataloging interface is to surround the existing subject search input box with a set of div tags.

```
<span id="extraInformation "></span>
<div class="ui-widget ">
   <input id="existingBox " type="text " size="60 "></input>
</div>
```

The final modification is to add JavaScript to connect the assignFAST web service to the search input box. This function should be called from <body onload= ..>

```
function setUpPage() {
// connect the autoSubject to the input areas
 jQuery('#existingBox').autocomplete(   {
    source: autoSubjectExample,
    minLength: 1,
    select: function(event, ui) {
     jQuery('#extraInformation').html("FAST  ID <b>" + ui.item.idroot + "</b> Facet <b>"+
getTypeFromTag(ui.item.tag)+  "</b>");
    } //end select
  }
 ).data( "autocomplete"  )._renderItem = function( ul, item ) { formatSuggest(ul, item);};

} //end setUpPage()
```

The *source: autoSubjectExample* tells the autocomplete function to get the data from the *autoSubjectExample* function, which in turns calls the assignFAST web service. This is in the assignFASTComplete.js file.

In *select: function*, the *extraInformation* text is rewritten with additional information returned with the selected heading. In this case, the fast number and facet are displayed.

The generic *_renderItem* of the jQuery.autocomplete function is overwritten by the *formatSuggest* function (found in assignFASTComplete.js) to create a display that differentiates the See from the Authorized headings that are returned in the search. The version used for this example shows:

> See Heading *USE* **Authorized Heading**
>
> when a See heading is returned, or simply the
>
> **Authorized Heading**
>
> otherwise.

WEB SERVICE CONSTRUCTION

The autosuggest service for a FAST heading was constructed a little differently than the typical autosuggest. For a typical autosuggest for the term **Motion picture** from the example given above, you would index just that term. As the term was typed, **Motion picture** and other terms starting with the text entered so far would be shown until you resolved the desired heading. For example, typing in "mot" might give

> **Motion pictures**
>
> **Motion picture music**
>
> **Employee motivation**

Diesel motor

Mothers and daughters

For the typical autosuggest, the term indexed is the term displayed and is the term returned when selected.

For assignFAST, both the Established and See references are indexed. However, when typing resolves a See heading, both the See heading and its Established Heading are displayed. Only the Established Heading is selected, even if you are typing the See heading. For assignFAST, the "mot" result now becomes

Features (Motion pictures) *USE* **Feature films**

Motion pictures

Motorcars (Automobiles) *USE* **Automobiles**

Motion picture music

Background music for motion pictures *USE* **Motion picture music**

Motion pictures for the hearing impaired *USE* **Films for the hearing impaired**

Documentaries, Motion picture *USE* **Documentary films**

Mother of God *USE* **Mary, Blessed Virgin, Saint**

The headings in assignFAST are ranked by how often they are used in WorldCat, so headings that are more common appear at the top. To place the Established Heading above the See heading when they are similar, the Established Heading is also ranked higher than the See for the same usage. assignFAST can also be searched by facet, so if only topical or geographic headings are desired, only headings from these facets will be displayed.

The web service uses a SOLR[16] search engine running under Tomcat.[17] This provides full text search and many options for cleaning and manipulating the terms within the index. The particular option used for assignFAST is the EdgeNGramFilter.[18] This option is used for autosuggest and has each word indexed one letter at a time, building to its entire length. The index for "cinema" will then contain "c," "ci," "cin," "cine," "cinem," and "cinema." SOLR handles UTF-8 encoded Unicode for both input and output. The assignFAST indexes and queries are normalized using FAST normalization[19] to remove punctuation, diacritics, and capitalization. FAST normalization is very similar to NACO normalization, although in FAST normalization the subfield indicator is replaced by a space and no commas retained.

assignFAST is accessed using a REST request.[20] REST requests consist of URLs that can be invoked via either HTTP POST or GET methods, either programmatically or via a web browser.

http://fast.oclc.org/searchfast/fastsuggest?&query=[query]&queryIndex=[queryIndex]&queryReturn=[queryReturn]&suggest=autosuggest&rows=[numRows]&callback=[callbackFunction]

where

<table>
<tr><th>parameter</th><th>description</th></tr>
<tr><td>query</td><td>The query to search</td></tr>
<tr><td>queryIndex</td><td>The index corresponding to the FAST facet. These include
<table>
<tr><th>name</th><th>description</th></tr>
<tr><td>suggestall</td><td>All facets</td></tr>
<tr><td>suggest00</td><td>Personal names</td></tr>
<tr><td>suggest10</td><td>Corporate names</td></tr>
<tr><td>suggest11</td><td>Events</td></tr>
<tr><td>suggest30</td><td>Uniform titles</td></tr>
<tr><td>suggest50</td><td>Topicals</td></tr>
<tr><td>suggest51</td><td>Geographic names</td></tr>
<tr><td>suggest55</td><td>Form/Genre</td></tr>
</table></td></tr>
<tr><td>queryReturn</td><td>Information requested list, comma separated. These include:
<table>
<tr><th>names</th><th>description</th></tr>
<tr><td>idroot</td><td>FAST Number</td></tr>
<tr><td>Auth</td><td>Authorized Heading, formatted for display with—as subfield separator</td></tr>
<tr><td>Type</td><td>alt or auth—indicates whether the match on the queryIndex was to an Authorized or See heading</td></tr>
<tr><td>Tag</td><td>MARC Authority tag number for the heading—100= Personal name, 150 = Topical, etc.</td></tr>
<tr><td>Raw</td><td>Authorized Heading, with subfield indicators. Blank if this is identical to auth (i.e., no subfields)</td></tr>
<tr><td>breaker</td><td>Authorized Heading in marcbreaker format. Blank if this is identical to raw (i.e., no diacritics)</td></tr>
<tr><td>indicator</td><td>Indicator 1 from the Authorized Heading</td></tr>
</table></td></tr>
<tr><td>numRows</td><td>headings to return maximum restricted to 20</td></tr>
<tr><td>Callback</td><td>the callback function name for jsonp</td></tr>
</table>

Table 1. assignFAST web service results description.

Example Response:

http://fast.oclc.org/searchfast/fastsuggest?&query=hog&queryIndex=suggestall&queryReturn=suggestall%2Cidroot%2Cauth%2Ctag%2Ctype%2Craw%2Cbreaker%2Cindicator&suggest=autoSubject&rows=3&callback=testcall

yields the following response:


```
testcall({
  "responseHeader":{
    "status":0,
    "QTime":148,
    "params":{
      "json.wrf":"testcall",
      "fl":"suggestall,idroot,auth,tag,type,raw,breaker,indicator",
      "q":"suggestall:hog",
      "rows":"3"}},
  "response":{"numFound":1031,"start":0,"docs":[
      {
        "idroot":"fst01140419",
        "tag":150,
        "indicator":"  ",
        "type":"alt",
        "auth":"Swine",
        "raw":"",
        "breaker":"",
        "suggestall":["Hogs"]},
      {
        "idroot":"fst01140470",
        "tag":150,
        "indicator":"  ",
        "type":"alt",
        "auth":"Swine--Housing",
        "raw":"Swine$xHousing",
        "breaker":"",
        "suggestall":["Hog  houses"]},
      {
        "idroot":"fst00061534",
        "tag":100,
        "indicator":"1",
        "type":"auth",
        "auth":"Hogarth,   William, 1697-1764",
        "raw":"Hogarth,   William,$d1697-1764",
        "breaker":"",
        "suggestall":["Hogarth,   William, 1697-1764"]}]
  }})
```


Table 3. Typical assignFAST JSON data return.

The first response heading is the Use For headin **Hogs,** which has the Authorized heading **Swine**. The second is the Use For heading for **Hog houses**, which has the Authorized heading **Swine--Housing**. This Authorized heading is also given in its raw form, including the $x subfield separator, which is unnecessary for the first heading. The third response matches the Authorized heading for **Hogarth, William, 1697–1764**, which is also given in its raw form. The breaker (MARCBreaker) format is only added if it differs from the raw form, which is only when diacritics are present.

CONCLUSIONS

Subject assignment is a combination of intellectual and manual tasks. The assignFAST web service can be easily integrated into existing cataloging interfaces, greatly reducing the manual effort required for a good subject data entry, increasing the cataloger's productivity.

REFERENCES

1. Lois Mai Chan and Edward T. O'Neill, *FAST: Faceted Application of Subject Terminology, Principles and Applications* (Santa Barbara, CA: Libraries Unlimited, 2010), http://lu.com/showbook.cfm?isbn=9781591587224.
2. OCLC Research Activities associated with FAST are summarized at http://www.oclc.org/research/activities/fast.
3. Lois M. Chan, *Library of Congress Subject Headings: Principles and Application: Principles and Application* (Westport, CT: Libraries Unlimited, 2005).
4. "Autocomplete," Wikipedia, last modified on October 1, 2013, http://en.wikipedia.org/wiki/Autocomplete.
5. Tony Russell-Rose, "Designing Search: As-You-Type Suggestions," *UX Magazine*, article no. 828, May 16, 2012, http://uxmag.com/articles/designing-search-as-you-type-suggestions.
6. David Ward, Jim Hahn, and Kirsten Feist, "Autocomplete as a Research Tool: A Study on Providing Search Suggestions," *Information Technology & Libraries* 31, no. 4 (December 2012), 6–19.
7. Jon Jermey, "Automated Indexing: Feeding the AutoComplete Monster," *Indexer* 28, no. 2 (June 2010), 74–75.
8. Holger Bast, Christian W. Mortensen, and Ingmar Weber, "Output-Sensitive Autocompletion Search," *Information Retrieval* 11 (August 2008), 269–286.
9. Elías Tzoc, "Re-Using Today's Metadata for Tomorrow's Research: Five Practical Examples for Enhancing Access to Digital Collections," *Journal of Electronic Resources Librarianship* 23, no. 1 (January–March 2011)

10. Holger Bast and Ingmar Weber, "Type Less, Find More: Fast Autocompletion Search with a Succinct Index," *SIGIR '06 Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (New York: ACM, 2006), 364–71.
11. Demian Katz, Ralph LeVan, and Ya'aqov Ziso, "Using Authority Data in VuFind," *Code4Lib Journal* 11 (June 2011).
12. Edward T. O'Neill, Rick Bennett, and Kerre Kammerer, "Using Authorities to Improve Subject Searches," in Maja Žumer, Sandra K. Roe, and Edward T. O'Neill, eds., "Beyond Libraries—Subject Metadata in the Digital Environment and Semantic Web," special issue, *Cataloging & Classification Quarterly* 52, no. 1/2 (in press).
13. "MARCMaker and MARCBreaker User's Manual," Library of Congress, Network Development and MARC Standards Office, revised November 2007, http://www.loc.gov/marc/makrbrkr.html .
14. "OCLC Developers Network—assignFAST," submitted September 28, 2012, http://oclc.org/developer/services/assignfast [page not found]
15. "jQuery autocomplete," accessed October 1, 2013, http://jqueryui.com/autocomplete.
16. "Apache Lucene—Apache Solr," accessed October 1, 2013, http://lucene.apache.org/solr.
17. "Apache Tomcat," accessed October 30, 2013, http://tomcat.apache.org.
18. "SOLR Wiki—Analyzers Tokenizers TokenFilters," last edited October 29, 2013, http://wiki.apache.org/solr/AnalyzersTokenizersTokenFilters.
19. Thomas B. Hickey, Jenny Toves, and Edward T. O'Neill, "Naco Normalization: A Detailed Examination of the Authority File Comparison Rules," *Library Resources & Technical Services* 50, no. 3 (2006), 166–72.
20. "Representational State Transfer," Wikipedia, last modified on October 21, 2013, http://en.wikipedia.org/wiki/Representational_State_Transfer.