Using Event Tracking to Enhance Library Web Interfaces

Scott Hanrath
Assistant Dean, Information Technology and Discovery Services
University of Kanas Libraries

Abstract

Event tracking allows for more fine-grained tracking of clicks, searches, and other user-initiated actions than is available through typical web analytics like page view and visit statistics. This session describes how staff at the University of Kansas Libraries have used event tracking in Google Analytics to monitor the Libraries online presence broadly, including the use of specific features of the Libraries website and third-party online applications, including the website homepage, our online catalog, a databases A-Z list, and a discovery system. Considerations discussed include the technical skills necessary to implement event tracking, challenges in tracking events on hosted systems, and how we've used event tracking to inform our efforts to assess and improve our services.

Review of Literature

Libraries have long recognized the value provided by analyzing the use of their websites. As online services have grown in importance to libraries, this value has increased. Turner notes that as providing access points for e-content becomes a primary function for library websites, "it becomes more and more important for libraries to work hard work to maintain a website that provides effective access points and information about these electronic information resources" (262). A web analytics program can provide library staff charged with managing websites with powerful and insightful means for ensuing the effectiveness of their websites.

Black defines web analytics as "the objective tracking, collection, measurement, reporting, and analysis of quantitative Internet data to optimize Web sites" (1). Loftus identifies two primary functions of a successful web analytics program: "to illustrate the value of the Web site, and to provide practical intelligence to the site's managers and content owners to use in improvements to the site" (2). This session will focus on the latter goal, demonstrating how event tracking can be used as a flexible and convenient method for gathering such practical intelligence.

Traditional web analytics using server logs can provide a basic picture of how a library website is used, including measures such as visit timing and lengths, technology used, and popular content (Black). However Turner notes that measures such as the virtual website visits reported as part of the annual Association of College and Research Libraries survey while useful, are crude and argues that libraries should borrow from the e-commerce world to create specific Key Performance Indicators (KPIs) for library websites that can be closely tied to defined organizational goals (263). Event tracking is well-suited to tracking such indicators.

The literature on web analytics and library websites consistently stresses that web analytics alone are an insufficient means of understanding how a website is used or evaluating its effectiveness and should be supplemented by methods such as user interviews, user surveys, and usability testing (Loftus, 6). As Black summarizes: "The statistics are the trail left by the user, but they do not explain the motivations behind that behavior" (10). The experience of the University of Kansas Libraries has been that interplay between methods is two-way: web analytics can both illuminate issues to be further probed by methods such as usability testing and be used to evaluate the effectiveness of changes made as a result of user testing or interviews.

Event Tracking in Google Analytics

Google Analytics (http://www.google.com/analytics/) is a free web-based tool for tracking web analytics. After registering for a Google Account, website managers create a profile for the website they wish to track. The tool provides a tracking code and a snippet of JavaScript code that must be added to each page of the website that should be tracked. Many popular Content Management Systems have features to simplify adding the tracking code. Google's Help documentation provides further details. The following discussion of event tracking assumes that a Google Analytics profile has been created and that the initializing tracking code has been added to all pages where events will be tracked.

Unlike web analytics methods that use server logs as data, Google Analytics creates data based on user interactions within the browser. Server logs will often undercount page views (defined as a user viewing a page in the browser) due to browser caching. When a user makes a repeat visit to a page, her browser will commonly use a locally saved (or "cached") copy of the page rather than requesting the page again from the server; because the server never receives a second request for the page, that interaction is not included in the server log. It is thus invisible to log-based analytics. Because Google Analytics uses JavaScript code in the browser to track page views when the browser loads the page, whether from the cache or from a server requests, it can yield more accurate counts.

Such in-browser tracking is not full-proof: errors generated by invalid HTML or other JavaScript code running on a page can prevent page views from being tracked. Browser add-ons designed to protect user privacy, such as the Ghostery add-on for Firefox (https://addons.mozilla.org/en-us/firefox/addon/ghostery/) can also prevent page views and events from being tracked. It is important for consumers of web analytics data to keep in mind that any method of web analytics will produce an approximation of user behavior, not a 100% accurate reporting.

The in-browser, JavaScript-based method used by Google Analytics provides opportunities for tracking more than simple page views. In this method, a page view is simply an event that occurs in the browser. The framework readily allows virtually any other "event" that a user initiates in the browser to be tracked as well. This ability is especially powerful in the post-Web 2.0 environment, where dynamic JavaScript widgets such as tabs and accordion menus and the dynamic loading of content via Asynchronous JavaScript and XML (AJAX) requests have been common. The widespread adoption of these practices means that users

now do much more on a given page than simply follow links to a different page, amd thus reduces the explanatory power of tracking page views as a method for understanding user interactions with a website. In what follows I will provide examples of such in-page events, provide the basic technical steps required to track them, and demonstrate how the University of Kansas Libraries have used the resulting to data to inform decisions about enhancing our web interfaces, including interfaces beyond our library website such as our catalog and discovery systems.

Example 1. Libraries Website Homepage

Basic page view and visit analytics tell us that our library website homepage is the most used of our libraries-provided online resources. To learn more about how users interact with the homepage, we began tracking a number of different events on the page.

Links

Anyone who has been in discussions about the content of a library website homepage knows that the links included on that page and the prominence of those links is a contentious topic. Using event tracking to provide data about which links are actually clicked can help inform decision-making. The simplest case for tracking a link click is to add an "onclick" handler to that link that tells Google Analytics to track the event.

The most common use of Google Analytics event tracking allows for three pieces of data to be tracked with the event: a category, an action, and a label. An example of the syntax of using an onclick handler to track a link click is:

```
<a href="#" onClick="_gaq.push(['_trackEvent', ‘Cateogry’,
‘Action’, 'Label']);">Link 1</a>
```

However, manually adding onlick handlers to each link is tedious. Further, we were interested not just in tracking events on the homepage in general, but also tracking which regions of the homepage were being used the most. Our homepage has sections for navigation, services, news, etc. and knowing more about which links from what sections are being used helps us better understand what our users are coming to our homepage for.

The JavaScript library jQuery is an invaluable tool for simplifying event-tracking. While this does increase the level of technical skills needed to implement event-tracking to include basic experience with JavaScript and Cascading Stylesheets (CSS), it expands the potential uses considerably. Using jQuery's selector syntax to retrieve information about a given page element – such as the text or URL of a link, or all of the links in an element with a given id – allows for adding richer data to the events being tracked. For example, to track clicks on all of the links in within the "Help" block of our homepage (see fig. 1), we first note that the HTML div element that contains the help links has the id attribute "homeHelp".

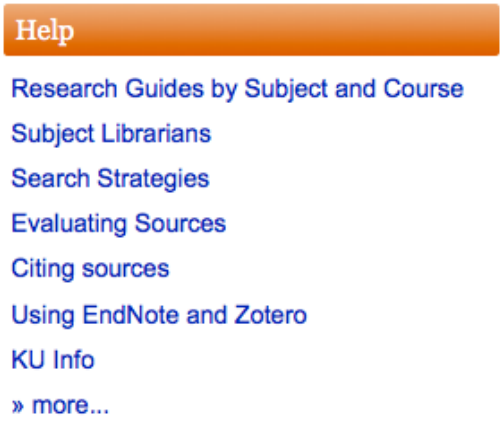

Figure 1. Help block

That allows us to create a jQuery selector to refer to all of the links in that section: ("#homeHelp a") and then to add an onclick handler to each of those links. We can further use the jQuery "this" idiom and the "text" function to retrieve the text of each link. That yields the following JavaScript to be run when the page loads:

```
$("#homeHelp a").click(function(){
_gaq.push(["_trackEvent","Home Block Link","Help",$(this).text()]);
});
```

This codes tracks clicks on every link in the "Help" section of the page, using the Google Analytics event category "Home Block Link", the action "Help", and the text of the link as the label. In the Google Analytics Reporting interface, we are thus able to generate a report like the example in fig. 2.

	Event Label	Total Events ↓	Total Events
		12,528 % of Total: 1.22% (1,027,258)	12,528 % of Total: 1.22% (1,027,258)
1.	Research Guides by Subject and Course	8,019	64.01%
2.	» more...	1,378	11.00%
3.	Subject Librarians	802	6.40%
4.	Citing sources	737	5.88%
5.	Search Strategies	538	4.29%
6.	Using EndNote and Zotero	527	4.21%
7.	Evaluating Sources	244	1.95%
8.	KU Info	153	1.22%

Fig. 2 Report

In this case the event tracking data informed our decision about the relative placement of links in this list. The high usage of the "Research Guides by Subject and Course" link compared to the other links in the list was evidence that it might be best placed as the first link in the list.

We are further able to easily compare the aggregate link clicks across all of the home page regions we have defined and for which we have enabled similar tracking, as in fig 3.

	Event Action	Total Events ↓	Total Events
		139,572 % of Total: 13.59% (1,027,258)	139,572 % of Total: 13.59% (1,027,258)
1.	Left Nav	98,583	70.63%
2.	Help	12,528	8.98%
3.	About	12,335	8.84%
4.	Services	7,766	5.56%
5.	Hours Today	6,105	4.37%
6.	Right Nav	2,255	1.62%

Fig. 3 Report

Similarly, where we provide A to Z title links for our databases, we track the 26 individual links as a group using an event category and action rather than individually; it's more useful to know how often the A-Z list is used than it is to know that "M" is a frequently used link (though that data is available too).

Additional JavaScript can be added to handle cases where the link uses an image, rather than text. Again, jQuery selectors are used to retrieve the necessary information for the event. But rather than retrieving the value for the event action using the jQuery "text" function, we first use the jQuery "children" function to determine if the anchor element contains any images. Then we use the "alt" attribute of image as the action value, as in this example code:

```
$("#section a").click(
  function(){
    // retrieve the text of the link
    var a = $(this).text();
    // does the link contain any images?
    if($(this).children("img").length > 0){
      // retrieve the alt attribute of the img (assume one)
      a = $(this).children("img").attr("alt");
    }
    _gaq.push(["_trackEvent","Home Block Link","Section", a])
  }
);
```

As we prepare for a homepage refresh and a move to a new CMS, the data provided by using event tracking to capture link clicks will be used to inform design decisions and to evaluate those decisions on the new site.

Tabs and other Widgets

Like many library websites, the University of Kansas Libraries homepage offers a set of tabs, each providing a different search option. Even tracking is used to track each time a tab is used and each time a search is launched. The tabs themselves simply alter which elements on the page are displayed; clicking a tab doesn't result in a new page view. Page views and visit alone don't provide any information about interactions with our tabs. Similarly, our home page displays daily hours information for our two largest locations and offers daily hours information about all of our locations when the user clicks a "More…" link, resulting in a drop-down element positioned above the page (see fig 4).

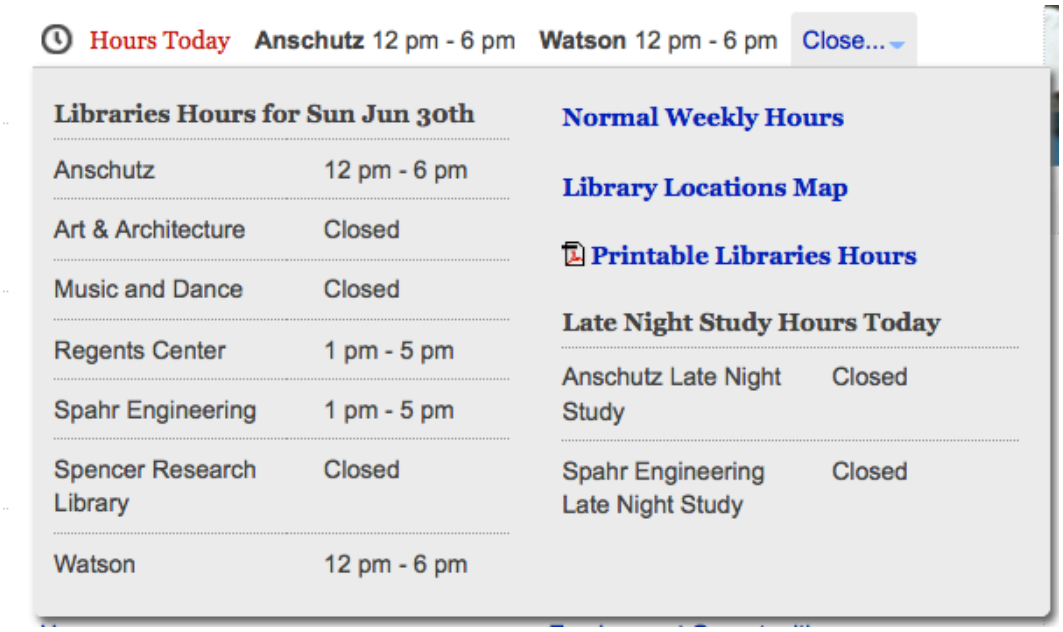

Fig. 4 Hours drop-down

Using events to track uses of the tabs and the hours drop-down is essentially the same as tracking link clicks. All that's required is to an add an onclick handler to the anchor elements that trigger a tab change or hours drop-down and add the necessary category, action, and/or label. In cases where a widget library such as jQuery UI or the Twitter Bootstrap framework is used to provide the widgets such as tabs, it's often possible to use functionality provided by the framework to trigger the event tracking and retrieve the necessary information for the event (e.g., Bootstrap's Tab widget includes "show" and "shown" events which can be used to define functions to track events).

After adding the daily hours drop down to our homepage, we were able to use the event tracking data to determine how frequently the "More" option was used to display hours for

additional locations. Combined with looking at trends in the number of page views for the Libraries main weekly and term hours pages since the addition of the home page hours, the event analytics give us a better sense of what hours information our users find most useful in the context of the library home page, allowing us to further adjust our home page going forward.

The search forms within the tabs all direct users outside of our website, to our discovery layer, the online catalog, etc. We use Google Analytics on those sites too, so we do have page view and visit statistics available. But these searches are available from multiple entry points on different sites. It's simpler to isolate which of those page views originated from the homepage by using event tracking on the homepage itself.

In addition to tracking the number of searches initiated from each tab, we are able to use event tracking to capture the queries themselves. To retrieve the queries from the forms, we ensure that the home page HTML markup places all of our search forms within a container element with the id "searchTabs," that each form has a meaningful id attribute to distinguish the forms from one another, and that the text input boxes used for the search queries all have a class attribute of "searchbox". We then use "Library Search Tabs Search" as the event category, the ids of the search forms as the event label, and the value of the text entered into the search box (as retrieved by the jQuery "val" function) as the label, as in the following example code:

```
$("#searchTabs form").submit(function(){
  _gaq.push(["_trackEvent",
             "Library Search Tabs Search",
             // the id attribute of the form being submitted
             $(this).attr("id"),
             // the value of the field with the class searchbox (the query)
             $(this).find(".searchbox").val()])
});
```

Tracking the queries entered into the search forms in our various tabs provides useful data about user search behavior. The resulting data can show us the distribution of searches across our tabs (see fig. 5), the most used search queries across the tabs (see fig. 6), or a number of other reports along dimensions such as time. The data can be used to inform decisions about the number, order, and placement of search tabs; the kinds of help documentation that should be available; and how frequently or infrequently searched for resources might be promoted.

	Event Action	Total Events	Total Events
		184,154 % of Total: 17.93% (1,027,258)	184,154 % of Total: 17.93% (1,027,2
1.	searchForm	73,462	39.89%
2.	catalogsearch	67,108	36.44%
3.	articlesearch	29,342	15.93%
4.	ejournalsearch	14,068	7.64%

Fig. 5 Report

	Event Label	Total Events	Total Events
		182,822 % of Total: 17.80% (1,027,258)	182,822 % of Total: 17.80% (1,027,258)
1.	science	147	0.08%
2.	jstor	146	0.08%
3.	pubmed	143	0.08%
4.	journal of music therapy	96	0.05%
5.	new york times	96	0.05%
6.	nature	93	0.05%
7.	scifinder	92	0.05%
8.	web of science	80	0.04%
9.	proquest	79	0.04%

Fig. 6 Report

While the provided interface allows website managers to look at the event data in a number of different ways, Google Analytics allows data to be exported to formats such as Comma-Separated Values (CSV) or Excel to be further analyzed using other software packages. This can be particularly helpful when analyzing data such as search queries, which may need to be cleaned before being effectively analyzed. An Application Programming Interface (API) is also available, allowing programmatic access to event data for use in other applications or for further analytics (see Morton-Owens and Hanson for an example of using the Google Analytics API to populate a dashboard).

Example 2. Catalog and Discovery

As with our library website-proper, the University of Kansas Libraries employs event tracking on other web-based interfaces it provides to users. Adding web analytics to

specialized third-party applications can be more difficult than on web pages under full local control. It often requires considerable access to application configuration and customization, in addition to the JavaScript skills required in the examples above. Many such applications provide their own set of analytics or statistics, which can be quite useful. Where possible, though, adding event tracking (and other basic analytics) to specialized library web interfaces can provide additional data designed to help answer questions specific to the organization.

For example, we have deployed event tracking in response to a basic question from our collection development team: "do users access electronic resources via their catalog records?" Gathering data to answer this question required several steps. First, our Voyager online catalog needed to have a Google Analytics profile with tracking enabled; we were able to achieve this by adding the necessary JavaScript to the online catalog via a configuration change, where we likewise included the jQuery JavaScript library. Second, the markup used by the online catalog was analyzed to identify jQuery selectors that could be used to retrieve the link elements for electronic resources. This was considerably more difficult than adding events to our website homepage. We have less control over the markup and could not, for example, add class and id attributes to page elements to make it easier to find the links we were concerned with. Because the markup on our catalog uses HTML tables to display field labels and values, we eventually created a method that depended on finding the correct table cells for the electronic resource link field label, then retrieving the link element from the following table cell. This is a brittle method: if the field label changes, the JavaScript needs to be updated or the events will stop being tracked. However, after installing this event tracking our collections development group can now receive regular reports on the use of electronic resources from catalog records and monitor those trends over time.

In 2012 we began providing the Primo discovery environment to our users. As we launched the new service we wanted to track the use of new features available in the environment, such as facets, both to determine the extent and nature of their use and to inform our on-going decisions about how to configure the features. Primo offers extensive reporting tools, but we found the event tracking with Google Analytics allowed us to create data tailored to our specific questions.

For example, the facets in Primo allow for a set number of facet values to appear as links the user can click to narrow the results. At the end of that list is a "More options" link that allows the user to select from a greater number of facet values in a pop-up list (see fig 7).

Fig. 7. Facets

We were interested in how often one of the "initial" facet values was selected versus the "More options" link. As with the online catalog, adding this event tracking required analyzing the underlying markup in Primo and creating jQuery selectors to retrieve the facet heading, facet value links, and more options link for each facet. This was achieved through a combination of parent/child element transversal and using the jQuery "hasClass" method to test elements for the presence of class attributes that Primo uses for different elements in the facet markup, as in this simplified example:

```
// for all the list items in every list with class EXLFacetsList
$('.EXLFacetsList li').each(function(){
    var facetLi = $(this);
    // retrieve the facet label by walking the markup to find the
    // element that occurs just before the facet list
    var facet = facetLi.parent().prev().text();
    var type;
    // determine whether this is an initially presented facet value
    // or a "More options" link
    if (facetLi.hasClass('EXLFacet')){
      type = 'initial';
    }
    if (facetLi.hasClass('EXLFacetsDisplayMore')){
      type = 'showmore';
    }
    // track an event every time the link in the facet list
    // item is clicked
    facetLi.find('a').each(function(){
      $(this).click(function(){
        _gaq.push(['_trackEvent', 'Facet Click', facet, type]);
      });
```

```
    });
});
```

The data from tracking facet clicks allows us to see which facets are used most often (see fig 8) and the relative use of initially presented facet values versus the more option link both for the facet sets as a whole or for a particularly selected facet (see fig 9).

	Event Action	Total Events	Total Events
		5,261 % of Total: 12.24% (42,990)	5,261 % of Total: 12.24% (42,990)
1.	Format	1,469	27.92%
2.	Topic	1,160	22.05%
3.	Show only	1,051	19.98%
4.	Date	667	12.68%
5.	Author/Creator	331	6.29%

Fig. 8 Report

	Event Label	Total Events	Total Events
		5,161 % of Total: 12.01% (42,990)	5,161 % of Total: 12.01% (42,990)
1.	initial	3,597	69.70%
2.	showmore	1,564	30.30%

Fig 9. Report

This data has been used to inform decisions about the ordering of facets on the page to place the more frequently used facets at the top of the list. It has also revealed patterns in the use of initial facet values versus the more options link that vary depending on the facet, a finding that might be useful to consider when crafting instruction for our discovery tool.

Adding event tracking to page content generated dynamically through AJAX requests poses an additional challenge. Because the AJAX content does not exist at the time the page is loaded, but is instead often loaded to the document based on a user action, there is nothing to add an onclick or other handler to at that time. Instead, methods such as delegated events in jQuery must be used to ensure that events are tracked on AJAX-loaded content. In cases where a third-party application is already using the jQuery library for AJAX requests, but the version is an older one that does not support event delegation (as was the case for the first version of Primo we began using) the jQuery "ajaxSuccess" method offers an option for event tracking on content added by an AJAX call based on the URL of the request as returned by the AJAX options object. In the absence of either option, the underlying application code may need to be modified to enable event tracking. In that instance the

values of the data provided by the event tracking must be weighed against the effort required to maintain the modified application.

Conclusion

Using Google Analytics to track events across the various online platforms offered by a library can provide data to organization efforts to assess website effectiveness. When employed across platforms, event tracking can offer a more nuanced description of user behavior than that found in traditional web analytics such as page views and visits. Adding event tracking requires some staff expertise in web technologies like JavaScript. While adding event tracking to third-party applications can be more difficult than adding event tracking to locally managed websites, custom event tracking on such applications can augment data already provided by those applications to better meet local needs.

Works Cited

Black, Elizabeth. "Web Analytics: A Picture of the Academic Library Web Site User." *Journal of Web Librarianship* 3.1 (2009): 3-14. Web. 29 June 2013.

"Bootstrap." *getbootstrap.com*. Bootstrap, n.d. Web. 29 June 2013. <http://getbootstrap.com/>

"Event Tracking - Web Tracking (ga.js)." *developers.google.com*. Google Developers, 1 March 2013. Web. 29 June 2013. <https://developers.google.com/analytics/devguides/collection/gajs/eventTrackerGuide>

"Get Answers to Your Google Analytics Questions." *google.com/analytics.* Google.com, n.d. Web. 29 June 2013. <http://www.google.com/analytics/support/index.html>

"jQuery." *jQuery.com.* jQuery Foundation, n.d. Web. 29 June 2013. <http://jquery.com/>

Loftus, Wayne. "Demonstrating Success: Web Analytics and Continuous Improvement." *Journal of Web Librarianship* 6.1 (2012): 45-55. Web. 29 June 2013.

Morton-Owens, Emily, and Karen Hanson. "Trends at a Glance: A Management Dashboard of Library Statistics." *Information Technology and Libraries* 31.3 (2012): 36-51.

Turner, Steven. "Web Statistics 2.0: Using Google Analytics to Measure Library Website Effectiveness." *Technical Services Quarterly* 27.3 (2010): 261-278. Web. 29 June 2013.

"Widgets | jQuery User Interface API Documentation." *jQueryui.com*. jQuery Foundation, n.d. Web. 29 June 2013. < http://api.jqueryui.com/category/widgets/>