Widgets, Gadgets, and Mobile Apps for Libraries: Tips, Code Samples, Explanations, and Downloads

Michael Sauers
Technology Innovation Librarian
Nebraska Library Commission
msauers@nlc.state.ne.us

Jason A. Clark
Head of Digital Access and Web Services
Montana State University Libraries
jaclark@montana.edu

Karen A. Coombs
Head of Web Services
University of Houston Libraries
kacoombs@uh.edu

Mobile/Gadget/Widget Design – Building Blocks

1. LEARN – learn and use the platform (iPhone, Blackberry, browser, WWW)
2. THINK MODULAR – think of atomic library services; can they work in this environment?
3. DESIGN and TEST – pick the pieces you need, format those pieces for display
4. DISPLAY – work within the confines of the platform, keep it simple

Mobile/Gadget/Widget Design – Multiple Endpoints

- Facebook, MySpace
- iGoogle, Blogs
- Content/Learning Management Systems (moodle, Drupal, etc.)
- Mobile Devices
- "An opportunity to broadcast our library signals" and get into workflow of our users

Links – Getting Started with Widgets, Gadgets and Mobile Apps

- WidgetBox: http://www.widgetbox.com/
- Google Gadgets: http://code.google.com/apis/gadgets/
- Designing for the Mobile Web: http://www.sitepoint.com/article/designing-for-mobile-web/

* View more related links at: http://delicious.com/tag/cil2009+w9

Portable Libraries – Sample Applications

- "TERRA: The Nature of Our World" Google Gadget
- Mobile Digital Collections with Flickr

* View samples and download code at http://www.lib.montana.edu/~jason/files.php

Step #1: TERRA Google Gadget – Create XML container

XML source:

```
<?xml version="1.0" encoding="UTF-8" ?>
<Module>
        <ModulePrefs title="TERRA: The Nature of Our World"
                title_url="http://www.lifeonterra.com/"
                description="Browse and search TERRA: The Nature of Our World for nature videos"
                screenshot="http://www.lifeonterra.com/meta/images/terra-gadget-screen.jpg"
                thumbnail="http://www.lifeonterra.com/meta/images/terra-gadget-thumb.jpg"
                author="Montana State University Libraries"
                author_location="Bozeman, MT, USA"
                author_email="jaclark@montana.edu"
                >
                <Require feature="tabs" />
                <Require feature="dynamic-height" />
                <Require feature="setprefs"/>
        </ModulePrefs>
        <UserPref name="show_date" display_name="Show Dates?" datatype="bool"/>
        <UserPref name="show_summ" display_name="Show Summaries?" datatype="bool"/>
        <UserPref name="num_entries" display_name="Number of Entries:" />
...
```

Step #1: TERRA Google Gadget – Explanation

• Google Gadget uses XML for instructions
• Name the modules assigned to the gadget <Module>
• List the requirements and user preferences needed to direct the xHTML and javascript to do something

Step #2: TERRA Google Gadget – Add behavior w/xHTML and javascript

XML source:

```
<Content type="html">
        <![CDATA[
        <script type="text/javascript">
        // Initialize tabs, designate the tab named "TERRA Feed" as
        // the tab selected by default.
        var tabs = new _IG_Tabs(__MODULE_ID__, "TERRA Feed");
        function init() {
        // Create the tab and define a corresponding <div> in the
        // HTML portion of the gadget. Add static content to the <div>.
        tabs.addTab("TERRA Feed", "feed");
        tabs.addTab("TERRA Search", "search");

        }
        // Call init function to initialize and display tabs.
        _IG_RegisterOnloadHandler(init);
        </script>

        <div id="feed" style="display:none">
        <p><img src="http://www.lib.montana.edu/meta/img/msu_favicon.ico" alt="terra video rss
feed" /> <strong>TERRA: The Nature of Our World</strong></p>
        <style> #content_div { font-size: 80%; margin: 5px; background-color: #FFFFBF;} </style>

        <div id=content_div></div>
```

```
<script type="text/javascript">

// Get userprefs
var prefs = new _IG_Prefs(__MODULE_ID__);
var showdate = prefs.getBool("show_date");
var summary = prefs.getBool("show_summ");
var entries = prefs.getInt("num_entries");

// If user wants to display more than 9 entries, display an error
// and set the value to 9, the max allowed.
if (entries > 9)
{
alert("You cannot display more than 9 entries.");
entries = 9;
}

// Use the _IG_FetchFeedAsJSON() function to retrieve core feed data from
// the specified URL. Then combine the data with HTML markup for display in
// the gadget.
_IG_FetchFeedAsJSON(
"http://feeds.feedburner.com/Terravideos",
function(feed) {
if (feed == null){
alert("There is no data.");
return;
}

// Start building HTML string that will be displayed in gadget.
var html = "";
// Access the fields in the feed
//html += "<div><b>" + feed.Title + "</b></div>";
//html += "<div>" + feed.Description + "</div><br>";

// Access the data for a given entry
if (feed.Entry) {
for (var i = 0; i < feed.Entry.length; i++) {
html += "<div>"
+ "<a target='_blank' href='" + feed.Entry[i].Link + "'>"
+ feed.Entry[i].Title
+ "</a> ";
if (showdate==true)
{
// The feed entry Date field contains the timestamp in seconds
// since Jan. 1, 1970. To convert it to the milliseconds needed
// to initialize the JavaScript Date object with the correct date,
// multiply by 1000.
var milliseconds = (feed.Entry[i].Date) * 1000;
var date = new Date(milliseconds);
html += date.toLocaleDateString();
html += " ";
html += date.toLocaleTimeString();
}
if (summary==true) {
html += "<br><i>" + feed.Entry[i].Summary + "</i>";
}
html += "</div>";
}
```

```
        }

        _gel("content_div").innerHTML = html;
        // The rest of the function parameters, which are optional: the number
        // of entries to return, and whether to return summaries.
        }, 5, summary);

        </script>
        <a href="http://feeds.feedburner.com/Terravideos" target="_blank">+ subscribe</a>
        </div>

        <div id="search" style="display:none">
        <p><img src="http://www.lib.montana.edu/meta/img/msu_favicon.ico" alt="terra video
search" /> <strong>TERRA: The Nature of Our World</strong></p>
        <form action="http://www.lifeonterra.com/results.php" method="post" target="_blank">
        <input type="hidden" name="max" value="25" />
        <input type="hidden" name="low" value="0" />
        <input class="txt_search" type="text" name="keyword" maxlength="35" size="25" />&nbsp;
<input type="submit" id="submit" class="submit" value="Search" />
        </form>
        <a href="http://www.lifeonterra.com/search.php" target="_blank">+ advanced search</a>
        </div>
        ]]>
</Content>
```

Step #2: TERRA Google Gadget – Explanation

• Using javascript methods that are part of Google Gadget API to control action/behavior of the gadget
    `<script type="text/javascript">, var tabs, _IG_RegisterOnloadHandler(init);`
• XHTML markup for tabbed display and `<div id=content_div>` to hold dynamic content
    `<div id="feed" style="display:none">, <div id="search" style="display:none">`
• Use javascript to assign and parse feed, print out dynamic content
    `var prefs, _IG_FetchFeedAsJSON("http://feeds.feedburner.com/Terravideos",`

Step #3: TERRA Google Gadget – Upload to server and submit to Google

• Test gadget at http://code.google.com/apis/gadgets/docs/legacy/gs.html#Scratchpad
• If you want to brand your gadget, will need to create image screenshot and thumbnail (see below)
• Instructions for submitting gadget - http://www.google.com/ig/submit

Step #1: Mobile Digital Collections – Create xHTML markup

xHTML source:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<head>
<title>Portable Libraries, Mobile/iPhone View of Content : Montana State University Libraries</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<meta name="viewport" content="width=500">
<!-- <meta name="viewport" content="width=320"> 480 is width of iphone sideways -->
<link rel="shortcut icon" href="/meta/img/msu_favicon.ico" />
<link rel="stylesheet" type="text/css" media="print" href="/meta/styles/print.css" />
<link rel="stylesheet" type="text/css" media="handheld" href="/meta/styles/mobile.css" />
...

<h1>Portable Libraries, <span>Mobile/iPhone View of Content</span><small>from MSU
Libraries</small></h1>
<div class="container">
        <h2 class="trigger"><a href="#">Dudes Enroute to Buffalo Stampede (Ca. 1920's)</a></h2>
        <div class="toggle_container">
                <div class="block">
                        <h3>Dudes Enroute to Buffalo Stampede (Ca. 1920's)</h3>
                        <a title="Dudes Enroute to Buffalo Stampede (Ca. 1920's)"
href="http://www.flickr.com/photos/msulibrary/3329345848/"><img
src="http://farm4.static.flickr.com/3646/3329345848_f7e0d9ef23_s.jpg" alt="Dudes Enroute to
Buffalo Stampede (Ca. 1920's)" /></a>
                        <p><strong>Date Uploaded:</strong> Wed, 4 Mar 2009 12:49:42 -0800</p>
                </div>
        </div>
</div>
```

Step #1: Mobile Digital Collections – Explanation

- The markup framework for our mobile app - “digital skeleton”
- Tags that inform display - <meta name="viewport" content="width=500" />
- Tags that javascript will use to create actions - <h2 class="trigger">, <div class="toggle_container">

Step #2: Mobile Digital Collections – Add behavior with javascript

javascript source:

```
<script type="text/javascript" src="http://code.jquery.com/jquery-latest.js"></script>
<script type="text/javascript">
$(document).ready(function(){

        $(".toggle_container").hide();

        $("h2.trigger").toggle(function(){
                $(this).addClass("active");
                }, function () {
                $(this).removeClass("active");
        });

        $("h2.trigger").click(function(){
                $(this).next(".toggle_container").slideToggle("slow,");
        });

});
</script>
```

Step #2: Mobile Digital Collections – Explanation

- Load the Prototype javascript library
- Tell it which xHTML tags to use - <h2 class="trigger">, <div class="toggle_container">
- Assign action using internal function with Prototype library
  hide(), addClass("active"), slideToggle("slow,")

* Learn more about Prototype internal functions - http://www.prototypejs.org/api

Step #3: Mobile Digital Collections – Add style with CSS, images

CSS source:

```
/* =miscellaneous and special global settings
---------------------------------------------- */
body {
	font: 10px normal Arial, Helvetica, sans-serif;
	margin: 0;
	padding: 0;
	line-height: 1.7em;
}
*, * focus {
	outline: none;
	margin: 0;
	padding: 0;
}
.container {
	width: 500px;
	margin: 0 auto;
}
h1 {
	font: 4em normal Georgia, 'Times New Roman', Times, serif;
	text-align:center;
	padding: 20px 0;
	color: #aaa;
}
h1 span {
	color: #666;
}
h1 small{
	font: 0.3em normal Verdana, Arial, Helvetica, sans-serif;
	text-transform:uppercase;
	letter-spacing: 1.5em;
	display: block;
	color: #666;
}
h2.trigger {
	padding: 0 0 0 50px;
	margin: 0 0 5px 0;
	background: url("../img/h2_trigger_a.gif") no-repeat;
	height: 46px;
	line-height: 46px;
	width: 450px;
	font-size: 2em;
	font-weight: normal;
	float: left;
}
h2.trigger a {
	color: #fff;
	text-decoration: none;
	display: block;
}
h2.trigger a:hover {
	color: #ccc;
}
h2.active {
	background-position: left bottom;
}
.toggle_container {
	margin: 0 0 5px;
	padding: 0;
	border-top: 1px solid #d6d6d6;
	background: #f0f0f0 url("../img/toggle_block_stretch.gif") repeat-y left top;
	overflow: hidden;
	font-size: 1.2em;
	width: 500px;
	clear: both;
}
.toggle_container .block {
	padding: 20px;
	background: url("../img/toggle_block_btm.gif") no-repeat left bottom;
}
.toggle_container .block p {
	padding: 5px 0;
	margin: 5px 0;
}
.toggle_container h3 {
	font: 2.5em normal Georgia, "Times New Roman", Times, serif;
	margin: 0 0 10px;
	padding: 0 0 5px 0;
	border-bottom: 1px dashed #ccc;

}
.toggle_container img {
	float: left;
	margin: 10px 15px 15px 0;
	padding: 5px;
	background: #ddd;
	border: 1px solid #ccc;
}
```

Step #3: Mobile Digital Collections – Explanation

• Create rules for display and format using our mobile.css file
• <div> tags are given absolute widths, <h2>, <img>, <a> tags styled for mobile (e.g., display:block)

Step #4: Mobile Digital Collections – Upload to server and test it out

• Test mobile view with iPhoney (http://www.marketcircle.com/iphoney/) or search Google for "iPhone emulator"

Final Thoughts

• Start with simple display formats – basic xHTML and CSS
• Keep experimenting and learning with a single widget/gadget/mobile service
• Train yourself to recognize when library web services can become atomic as widgets, gadgets, etc.