

# Library Web Services – Sample Applications

## Karen Coombs

### Code Sample: Embedding Google Calendar Data in a Web Page

#### The HTML

```
<html>
<head>
<title>Google Calendar Test</title>
<script type="text/javascript" language="JavaScript" src="../script_libraries/jquery.js"></script>
<script type="text/javascript" language="JavaScript" src="../script_libraries/jquery-ui.js"></script>
<script type="text/javascript" language="JavaScript" src="../script_libraries/date.js"></script>
<script type="text/javascript" language="JavaScript" src="google_calendar.js"></script>
<link type="text/css" href="../css/base/ui.all.css" rel="stylesheet" />
<link type="text/css" href="calendar.css" rel="stylesheet" />
</head>
<body>
<div id="sidebar">
<h4>Calendar LITA Events at ALA Midwinter 2010</h4>
<div id="calendar"></div>
</div>
<div id="main">
<h4>Some Title Here</h4>
<p>Body text here</p>

</div>
</body>
</html>
```

#### Explanation

1. This is really just an HTML page. I've structured it to have two content regions: div#sidebar and div#main
2. It has two CSS files
  - calendar.css – just lays out the two columns
  - ../css/base/ui.all.css – the CSS for the JQuery-UI library
3. Need to add the appropriate script libraries: JQuery, JQuery-UI and a data script library
4. Add a div with the ID calendar in order to insert the Calendar information into a right sidebar

#### The Javascript

```
$(document).ready(function(){
    // add List of Calendar Items
    $(function addCalendar(){
        $.getJSON( "http://www.google.com/calendar/feeds/csr34g1n3nk0squg9eb0u7laoc
%40group.calendar.google.com/private-2a276d4a03b15896f0e1379ba2d7c081/full?alt=json-in-
```

```

script&start-min=2010-01-16T00:00:00&start-max=2010-01-
19T23:59:59&orderby=starttime&callback=?",
    function (data) {
        $('div#calendar').append('<ul class="calendar_entries"></ul>');
        $.each(data.feed.entry, function(i,item){
            $('calendar_entries').append('<li class="calendar_entry"><span
style="display:none">' + item.id.$t + '</span><a href="' + item.link[0].href + '">' + item.title.$t +
'</a></li>');
        });
    });
});

// CalendarInformation Information Popup
$('calendar_entry').live('click', (function(event){
event.preventDefault();
var entry_id = $(this).children("span").text();
$("body").append('<div id="calendar_detail"></div>');
$("#calendar_detail").dialog({
    bgiframe: true,
    height: 500,
    width: 600,
    modal: true,
    title: 'Event Details',
    close: function(event, ui) {
        $("#calendar_detail").dialog('destroy');
        $("#calendar_detail").remove();
    }
});
$("#calendar_detail").append('<p>Loading please wait...</p> ');
$.getJSON(entry_id + "?alt=json-in-script&callback=?",
    function(data){
        var start_info = data.entry.gd$when[0].startTime;
        start_info = start_info.substr(0,start_info.indexOf('.'));
        start_info = Date.parse(start_info);
        var start_month = start_info.getMonth();
        start_month++
        var start_day = start_info.getDate();
        var start_year = start_info.getFullYear();
        var start_hour = start_info.getHours();
        var start_minute = start_info.getMinutes();
        start_minute = start_minute + "";
        if (start_minute.length == 1) {
            start_minute = "0" + start_minute;
        }
        var a_p = "";
        if (start_hour < 12) {
            a_p = "AM";
        } else {

```

```

    a_p = "PM";
  }
  if (start_hour == 0) {
    start_hour = 12;
  }
  if (start_hour > 12) {
    start_hour = start_hour - 12;
  }
  var start_time = start_hour + ':' + start_minute + a_p;

  // Format Event End time
  var end_info = data.entry.gd$when[0].endTime;
  end_info = end_info.substr(0, end_info.indexOf('.'));
  end_info = Date.parse(end_info);
  var end_month = end_info.getMonth();
  end_month++;
  var end_day = end_info.getDate();
  var end_year = end_info.getFullYear();
  var end_hour = end_info.getHours();
  var end_minute = end_info.getMinutes();
  end_minute = end_minute + '';
  if (end_minute.length == 1) {
    end_minute = "0" + end_minute;
  }

  var a_p = "";
  if (end_hour < 12) {
    a_p = "AM";
  } else {
    a_p = "PM";
  }
  if (end_hour == 0) {
    end_hour = 12;
  }
  if (end_hour > 12) {
    end_hour = end_hour - 12;
  }
  var end_time = end_hour + ':' + end_minute + a_p;

  $('#div#calendar_detail').empty();
  $('#div#calendar_detail').append('<h4>' + data.entry.title.$t + '</h4>');
  $('#div#calendar_detail').append('<p>' + data.entry.content.$t + '</p>');
  $('#div#calendar_detail').append('<ul id="time"></ul>');
  $('#div#calendar_detail').append('<li>Starts: ' + start_month + '/' +
start_day + '/' + start_year + ' ' + start_time + '</li>');
  $('#div#calendar_detail').append('<li>Ends: ' + end_month + '/' + end_day
+ '/' + end_year + ' ' + end_time + '</li>');
  $('#div#calendar_detail').append('<li>Location: ' +
data.entry.gd$where[0].valueString + '</li>');

```

```
});
```

```
});
```

```
});
```

## **Explanation**

1. Get the JSON feed of the calendar from Google Calendar
  - a) This url can be obtained by looking at your Google Calendar Public/Private URLs
  - b) You can tell it a date range to get – which this script does
  - c) You can tell it to sort by something – this script sorts by start time. The default sort is last updated
2. Once you have the JSON, loop through the entries and create
  - a) HTML which creates a bulleted list with links
  - b) Make sure each list item also has a span in it which contains the entry id field while use this later
3. Create a function which allows the links when clicked on to bring up more detailed information in a popup.
  - a) Gather the individual entry url from the link clicked on's span tag
  - b) Use this with the alt=json-in-script parameter to retrieve information for that event.
  - c) Parse the information retrieved back from Google on the Event into a dialog window
    - i. This has a large section of data manipulation code into order to get the data in a human readable format.

## Code Sample: Incorporating Flickr Photos as a Gallery Script

### The HTML

```
<html>
<head>
  <meta http-equiv="content-type" content="text/html; charset=UTF-8">
  <script src="../script_libraries/jquery.js"></script>
  <script src="../script_libraries/jquery-cycle/jquery.cycle.min.js"></script>
  <script src="flickr_gallery.js"></script>
  <style>
    #images #photo { width:250px; padding: 15px; border: 1px solid #ccc; background-color:
#eee; }
    #images #photo img {border:none;}
  </style>
</head>
<body>
  <div id="images"></div>
  <div id="flickrNav">
    <a id="prev" href="#">Prev</a> <a id="next" href="#">Next</a>
  </div>
  <p id="link"><a href="http://www.flickr.com/search/?w=all&q=c4110+oclc&m=tags">More
code4lib 2010 preconference photos</a></p>
</body>
</html>
```

### Explanation

1. This is really just an HTML page.
2. It has inline styles which will help create the image gallery  
#images #photo { width:250px; padding: 15px; border: 1px solid #ccc; background-color: #eee; }  
#images #photo img {border:none;}
3. Need to add the appropriate script libraries: JQuery, JQuery-Cycle
4. Add a div with the ID images in order to insert the Flickr photo information
5. Make sure Previous and Next links exist in a div with the ID flickrNav to facilitate navigation through the photos

### The Javascript

```
$.getJSON("http://api.flickr.com/services/feeds/photos_public.gne?tags=c4110,oclc&lang=en-us&format=json&jsoncallback=?", function(data){
  $.each(data.items, function(i,item){
    var photo_info = '<div id="photo"><a href="' + item.link + "'></a><p>' + item.title + '</p></div>';
    $("#images").append(photo_info);
  });
});
```

```
$('#images').cycle({  
  fx: 'fade',  
  speed: 'normal',  
  timeout: 0,  
  next: '#next',  
  prev: '#prev'  
});  
});
```

## Explanation

1. Get the JSON feed of the images matching the desired tags from Flickr
  - a) Information on how to construct this URL is at <http://www.flickr.com/services/feeds/>
  - b) You can tell it a set of tags to match – which this script does
  - c) You can tell it to get photos from a particular user or users
2. Once you have the JSON, loop through the entries and create
  - a) HTML which consists of a set of div tags each which contains the image linked back to the larger image at Flickr and a caption for the photo
3. Use the JQuery Cycle plugin to allow the photos to be displayed in a gallery fashion
  - a) Make sure you set
    - i. the fx (effect) parameter
    - ii. speed – for cycling
    - iii. next – element which represents the next button
    - iv. prev – element which represents the previous button

## Code Sample: Incorporating Delicious Links

### The HTML

```
<html>
<head>
<title>Delicious Links Test</title>
<script type="text/javascript" language="JavaScript" src="../script_libraries/jquery.js"></script>
<script type="text/javascript" language="JavaScript" src="../script_libraries/jquery-ui.js"></script>
<script type="text/javascript" language="JavaScript" src="../script_libraries/date.js"></script>
<script type="text/javascript" language="JavaScript" src="delicious.js"></script>
<link type="text/css" href="../css/base/ui.all.css" rel="stylesheet" />
<link type="text/css" href="delicious.css" rel="stylesheet" />
</head>
<body>
<div id="sidebar">
<h4>Links for Web Services Workshop</h4>
<div id="delicious_links"></div>
</div>
<div id="main">
....
</div>
</body>
</html>
```

### Explanation

1. This is really just an HTML page. I've structured it to have two content regions: div#sidebar and div#main
2. It has a CSS file, delicious.css – which just lays out the two columns
3. Need to add the appropriate script library: JQuery
4. Add a div with the ID delicious\_links in order to insert the Delicious links into a right sidebar

### The Javascript

```
$(document).ready(function(){
    // add List of Delicious Items
    $(function addLinks(){
        $.getJSON( "http://feeds.delicious.com/v2/json/librarywebchic/drupal+modules?
callback=?",
        function (data) {
            $('div#delicious_links').append('<ul class="links"></ul>');
            $.each(data, function(i,item){
                $('li.links').append('<li class="delicious_link"><a href="' + item.u +
">' + item.d + '</a></li>');
            });
        });
    });
});
```

```
});
```

### **Explanation**

1. Go fetch the items from Delicious that you want to incorporate using the JSON feed.
2. Loop through the items returned
3. Build a bulleted list which contains the title of the link which is linked to the web page
4. Add a link to all More items in Delicious with the tag

## Code Sample: Incorporating YouTube Videos

### The HTML

```
<html>
<head>
<title>YouTube Test</title>
<script type="text/javascript" language="JavaScript" src="../../script_libraries/jquery.js"></script>
<script type="text/javascript" language="JavaScript" src="../../script_libraries/jquery-ui.js"></script>
<script type="text/javascript" language="JavaScript" src="youtube.js"></script>
<link type="text/css" href="../../css/base/ui.all.css" rel="stylesheet" />
<link type="text/css" href="youtube.css" rel="stylesheet" />
</head>
<body>
<div id="sidebar">
<h4>YouTube Videos</h4>
<div id="youtube_links"></div>
</div>
<div id="main">

</div>
</body>
</html>
```

### Explanation

1. This is really just an HTML page. I've structured it to have two content regions: div#sidebar and div#main
2. It has a CSS file, youtube.css – which just lays out the two columns
3. Need to add the appropriate script libraries: JQuery and JQuery-UI
4. Add a div with the ID youtube\_links in order to insert the list of YouTube videos links into a left sidebar

### The Javascript

```
$(document).ready(function(){
    // add List of YouTube Items
    $(function addLinks(){
        $.getJSON( "http://gdata.youtube.com/feeds/api/videos?q=mashathon+oclc&alt=json-
in-script&orderby=published&start-index=11&max-results=10&v=2&callback=?",
        function (data) {
            $('#youtube_links').append('<ul class="videos"></ul>');
            $.each(data.feed.entry, function(i,item){
                $('.videos').append('<li class="youtube_link"><a href="' +
item.link[0].href + "'> + item.title.$t + '</a><span id="video_id" style="display:none"> +
item.link[3].href + '</span></li>');
            });
        });
    });
});
```

```

// YouTube Video Load
$('.youtube_link').live('click', (function(event) {
event.preventDefault();
var video_id = $(this).find('span#video_id').text();
$('#div#main').empty();
$("#div#main").append('<div id="youtube_video"></div>');
    $('#youtube_video').append('<p>Loading please wait...</p> ');
    $.getJSON(video_id + '&alt=json-in-script&v=2&callback=?',
        function(data) {
            $('#div#youtube_video').empty();
            $('#div#youtube_video').append('<h4>' + data.entry.title.$t + '</h4>');
            $('#div#youtube_video').append('<object style="height: 344px; width:
425px">');

            var movieInfo = data.entry.content.src;
            $('#div#youtube_video object').append('<param name="movie" value="" +
movieInfo + "">');

            $('#div#youtube_video object').append('<param name="allowFullScreen"
value="true">');

            $('#div#youtube_video object').append('<param
name="allowScriptAccess" value="always">');
            $('#div#youtube_video object').append('<embed src="" + movieInfo + ""
type="application/x-shockwave-flash" allowfullscreen="true" allowScriptAccess="always"
width="425" height="344">');
            $('#div#youtube_video').append('<p>' +
data.entry.media$group.media$description.$t + '</p>');

        });

    });
});
});

```

## Explanation

1. Go fetch the list of videos from YouTube that you want to incorporate using the JSON feed.
2. Loop through the items returned
3. Build a bulleted list which contains the url for the feed for that video in a span, title of the video which is linked to the web page for the video
4. Create function for displaying video within main content of page when link is clicked
5. Use JSON to get the feed for the video which is clicked
6. From video feed retrieve and video title, video embeddable url and video description
7. Add these item to the main portion of the screen

## Code Sample: WorldCat Search API + LibraryThing Reviews + Google Book Preview

### The form

```
<form action="worldcat_search_mashup.php" method="get">
  <p>
    <label for="AddWorldCatSearch-SearchType">Search Type</label>
    <select name="AddWorldCatSearch-SearchType" id="AddWorldCatSearch-SearchType">
      <option value="srw.kw">Keyword</option>
      <option value="srw.ti">Title</option>
      <option value="srw.au">Author</option>
    </select>
  </p>
  <p>
    <label for="AddWorldCatSearch-SearchString">Search </label>
    <input type="text" id="AddWorldCatSearch-SearchString" name="AddWorldCatSearch-SearchString" value="" />
  </p>
  <p>
    <label for="AddWorldCatSearch-LibraryLimit">Limit by Specific Library (OCLC Symbol)</label>
    <input type="text" id="AddWorldCatSearch-LibraryLimit" name="AddWorldCatSearch-LibraryLimit" value="" />
  </p>
  <input type="submit" value="Search"/>
</form>
```

### Form explanation

Select box for the different indexes which can be searched

Text box for the search string text

Text box for limiting by a particular library's OCLC symbol

### The URL Request

```
http://worldcat.org/webservices/catalog/search/worldcat/sru?query=srw.kw%3D%22ambient+findability%22+and+srw.li%3D%22TXH%22&recordSchema=info%3Asrw%2Fschema%2F1%2Fmarxml&servicelevel=full&wskey=12345kac
```

### URL Request Explanation

1. HTTP Request to WorldCat's Search API Service
2. Specify request protocol - sru
3. Specify the search being submitted and its components with query
4. Specify response format with recordSchema
5. Specify the service level with servicelevel
6. Identify the developer with wskey

### Request with PHP

```

if ( strlen($_REQUEST['AddWorldCatSearch-SearchString']) > 0 ) {
    $searchURL = 'http://worldcat.org/webservices/catalog/search/sru?query=' .
$_REQUEST['AddWorldCatSearch-SearchType'] . '%3D%22' .
urlencode($_REQUEST['AddWorldCatSearch-SearchString']) . '%22';

    if ( isset($_REQUEST['AWC_libraryLimit']) ) {
        $searchURL = $searchURL . '+and+srw.li%3D%22' . $_REQUEST['AddWorldCatSearch-
LibraryLimit'] . '%22';
    }

    $searchURL = $searchURL . '&maximumRecords=10';

    $searchURL .= '&wskey=' . $WorldCatAPIKey;
    $xml = simplexml_load_file($searchURL);

```

### Explanation

1. Make sure that a search was passed
2. Build the request URL by using the variables submitted via the form
3. Send request to server and load result as XML

### Use PHP to get the ISBNs from the search results

```

// Go get the ISBNs in the search results
$xml->registerXPathNamespace("marc", "http://www.loc.gov/MARC21/slim");
$google_ids = "";
$lt_ids = "";

$isbn = $xml->xpath("//marc:record/marc:datafield[@tag='020']/marc:subfield[@code='a']");
foreach ((array)$isbn as $isbn) {
    if (strlen($isbn) > 1) {
        $lt_ids = $lt_ids . substr($isbn, 0, strpos($isbn, " "));
        if ($isbn != end($isbn)) {
            $lt_ids = $lt_ids . ',';
        }
    }
}

```

### Explanation

1. Gather all the ISBNs from the records
2. Create an arrays of these ISBNs : one for getting information from LibraryThing

### Parse and display with PHP

```

$xml->registerXPathNamespace("marc", "http://www.loc.gov/MARC21/slim");

foreach($xml->xpath("//marc:record") as $book ) {
    $book['xmlns:marc'] = 'http://www.loc.gov/MARC21/slim';

```

```

$field = simplexml_load_string($book->asXML());
$title = $field->xpath("marc:datafield[@tag='245']/marc:subfield[@code='a']");
$publisher = $field->xpath("marc:datafield[@tag='260']/marc:subfield[@code='b']");
$publication_date = $field-
>xpath("marc:datafield[@tag='260']/marc:subfield[@code='c']");
$isbn = $field->xpath("marc:datafield[@tag='020']/marc:subfield[@code='a']");
if (strpos($isbn[0], " ") > 0) {
    $isbn_1 = substr($isbn[0], 0, strpos($isbn[0], " "));
} else {
    $isbn_1 = $isbn[0];
}
$author = $field->xpath("marc:datafield[@tag='100']/marc:subfield[@code='a']");
$oclcnumber = $field->xpath("marc:controlfield[@tag='001']");

echo '<div class="record">';
if (strlen($isbn_1) > 0) {
    // check Open Library for a cover
    $image_url = 'http://covers.openlibrary.org/b/isbn/' . $isbn_1 . '-S.jpg';
    $image_size = getimagesize($image_url);
    if ($image_size[0] > 1 and $image_size[1] > 1) {
        echo '';
    } else {
        // check LibraryThing for a cover
        $image_url = 'http://covers.librarything.com/devkey/' .
$librarything_key . '/small/isbn/' . $isbn_1;
        $image_size = getimagesize($image_url);
        if ($image_size[0] > 1 and $image_size[1] > 1) {
            echo '';
        }
    }
}
echo '<p><a href="http://www.worldcat.org/oclc/' . $oclcnumber[0] . '"><span>' .
$title[0] . '</span></a>';
if ( strlen($isbn_1) > 1 ) {
    echo '<br/>';
    echo '<span id="LT_' . $isbn_1 . '"></span><br/>';
    echo '<noscript><a href="http://www.librarything.com/isbn/' . $isbn_1 . '">View
Book Information at LibraryThing</a></noscript>';
    echo '<br/>';
    echo '<script type="text/javascript">';
    echo 'GBS_insertPreviewButtonPopup("ISBN:' . $isbn_1 . '")';
    echo '</script>';
    echo '<noscript><a href="http://books.google.com/books?vid=ISBN' . $isbn_1 .
">Book Info at Google Books</a></noscript>';

}
echo '</p>';
echo '<br clear="all"/>';
echo '</div>';

```

```
}
```

## Explanation

1. Add the marc namespace in order to properly select nodes.
2. Use xpath to gather a set of all the marc records returned.
3. Loop through each marc record as book and pull out specific information using xpath: title, publisher, publication\_date, isbn, author, and oclc number.
4. If cover image is available from OpenLibrary or LibraryThing print cover
5. Print HTML to display each book's title with a link to WorldCat.org
6. Add a span id as a placeholder for LibraryThing rating information to be gathered via Javascript and a noscript tag to create a link to the item at LibraryThing if Javascript is disabled
7. Add a second span with a script tag and javascript for possible link to preview or fulltext via GoogleBooks. The javascript will add a link if preview or fulltext is available. A noscript tag will create a link to book info at Google Books if Javascript is disabled.

## Add Javascripts to gather ratings

```
<script type="text/javascript">
    function LTpop(booksInfo){
        for (i in booksInfo) {
            var book = booksInfo[i];
            if (book.link) {
                var desc = "";
                var rating = " ";
                if (book.reviews && (book.reviews != '0')) {
                    desc = book.reviews + ' reviews' ;
                }
                if (book.rating) {
                    rating = ' <img src="" + book.rating_img + ""/>' ;
                    $('#LT_' + book.id).append('<a href="" + book.link + "">' +
desc + '@ LibraryThing</a>' + rating );
                }
            }
        }
    }
</script>
<script type="text/javascript" src='http://www.librarything.com/api/json/workinfo.js?ids=<?php echo $lt_ids ?>&callback=LTpop'></script>
```

## Javascript Explanation

The set of scripts gets information from LibraryThing. The first portion uses the ISBNs to retrieve a JSON object with reviews/ratings for those ISBNs from LibraryThing. The second portion takes the information returned from LibraryThing as JSON and parses it. If ratings exist then it creates and appropriate image of said rating (number of stars) and inserts it into the blank span with the appropriate id (the one with the same ISBN) in the HTML.

## Code Sample: WorldCat Holdings + Google Maps

### The form

```
<form action="worldcat_holdings_map.php" method="get">
  <p>
    <label for="OCLCNumber">OCLC Number</label>
    <input type="text" id="OCLCNumber" name="OCLCNumber" value="<?php echo
$_REQUEST['OCLCNumber']?>" />

  </p>
  <p>
    <label for="ZipCode">Zip Code </label>
    <input type="text" id="ZipCode" name="ZipCode" value="" />
  </p>
  <input type="submit" value="submit"/>
</form>
```

### Form explanation

1. Enter OCLC Number
2. Enter Zip Code where you want to get item

### The URL Request

```
http://www.worldcat.org/webservices/catalog/content/libraries/12345/?
wskey=kac123&location=04849&libtype=
```

### URL Request Explanation

/libraries/##/ - This is where the OCLC Number you are requesting information about goes  
wskey – WorldCat Search API key  
location – zip code  
libtype – library type (where 1 = academic, 2 = public, 3 = government, and 4 = other)

### Request with PHP

```
if ( (($isbn or $issn or $oclcnum) and ($oclcSym or $location or $ip)) or ($location) ) {
```

```
    // build request
    $holdingsURL = 'http://www.worldcat.org/webservices/catalog/content/libraries/';
    if (strlen($isbn) > 0) {
        $holdingsURL = $holdingsURL . 'isbn/' . $isbn;
    } elseif (strlen($issn) > 0) {
        $holdingsURL = $holdingsURL . 'issn/' . $issn;
    } elseif (strlen($oclcnum) > 0) {
        $holdingsURL = $holdingsURL . $oclcnum;
    } else {
    }
}
```

```

$holdingsURL = $holdingsURL . '?';
if (strlen($oclcSym) > 0) {
    $holdingsURL = $holdingsURL . 'oclcSymbol=' . $oclcSym;
} elseif (strlen($location) > 0) {
    $holdingsURL = $holdingsURL . 'location=' . $location;
} elseif (strlen($lat) > 0 and strlen($lon) > 0) {
    $holdingsURL = $holdingsURL . 'lat=' . $lat . 'lon=' . $lon;
} else {
    $holdingsURL = $holdingsURL . 'ip=' . $ip;
}

if (strlen($libstype) > 0) {
    $holdingsURL = $holdingsURL . '&libtype=' . $libstype ;
}

$holdingsURL = $holdingsURL . '&wskey=' . $WorldCatAPIKey ;

// go get the XML
$xml = simplexml_load_file($holdingsURL);

```

## Explanation

1. Make sure the and identifier (OCLC number, ISBN, ISSN) and locality information or merely locality exists
2. See if type of library exists
3. Build the URL from the variables passed by the form
  - This will change for different types of identifiers
4. Send request to server and load result as XML

## Parse the XML with PHP

```

$holdingCount = count($xml->xpath('//holding'));
if ($holdingCount > 0) {
    echo "status: "holdings";";
    echo "holdings": [";
    $id_number = 0;
    foreach($xml->xpath('//holding') as $library ) {
        $field = simplexml_load_string($library->asXML());
        $id_number++;
        $identifier = $field->institutionIdentifier->value;
        $name = $field->physicalLocation;
        $address = $field->physicalLocation . ' ' . $field->physicalAddress->text;
        $url = $field->electronicAddress->text;
        $copies = $field->holdingSimple->copiesSummary->copiesCount;

        echo '{';
        echo "id_number": " . $id_number . " ";
        echo "identifier": " . $identifier . " ";
    }
}

```

```

        echo "name": " . $name . "';
        echo "address": " . $address . "';
        echo "url": " . $url . "';
        echo "copies": " . $copies . "'";
        echo '>';
        if ($id_number != $holdingCount) {
            echo '!';
        }
    }
    echo ']';
} else {
    echo "status": "no holdings";
}
?>

```

### Explanation

1. Use xpath to count all the holdings returned.
2. If at least 1 holdings returned, use xpath to gather a set of all the holdings.
3. Loop through each holding as library and pull out specific information using xpath: identifier, name, address, catalog url, and copies.
4. Use the information retrieved to build json that represents this information

### The HTML

```

<script type="text/javascript" language="JavaScript" src="../script_libraries/jquery.js"></script>
<script src="http://maps.google.com/maps?
file=api&v=2.73&key=yourgooglemapskey" type="text/javascript"></script>
<script type="text/javascript" language="JavaScript" src="worldcat_map.js"></script>
<style type="text/css">
<!--
#oclcNum, #zip { display: none;}
-->
</style>
</head>
<body>
<div id="oclcNum"><?php echo $oclcNum ?></div>
<div id="zip"><?php echo $zip ?></div>
<div id="map_canvas" style="width: 800px; height: 600px"></div>

```

### Explanation

1. Add JQuery Javascript library
2. Add GoogleMaps script library with appropriate API key
3. Hide the OCLC Number and Zip from display
4. Add div to contain map canvas

### The Javascript

```

$(function getMap(){
    var map = null;
    var geocoder = null;
    var oclcNum = $('#oclcNum').text();
    var zip = $('#zip').text();
    if (GBrowserIsCompatible()) {
    map = new GMap2(document.getElementById("map_canvas"));
    map.setCenter(new GLatLng(37.4419, -122.1419), 7);
        geocoder = new GClientGeocoder();
        $.getJSON( "http://www.librarywebchic.net/mashups/library_holdings.php?oclcnum="
+ oclcNum + "&location=" + zip + "&callback=?",
            function (data) {
                $.each(data.holdings,function(i,holding) {
                    geocoder.getLatLng(
                        holding.address,
                        function(point) {
                            if (!point) {
                                alert(holding.address + " not found");
                            } else {
                                if (holding.id_number == 1) {
                                    map.setCenter(point, 7);
                                }
                                var marker = new GMarker(point, {title:holding.name});
                                var label = '<p><strong>' + holding.name +
'</strong></p><p><a href="' + holding.url + "'>See library catalog</a></p>';
                                map.addOverlay(marker);
                                marker.bindInfoWindowHtml(label);
                            }
                        }
                    );
                });
            }
        );
    map.addControl(new GSmallMapControl());
    }
});

```

## Explanation

1. Find the OCLC Number and ZipCode in the webpage
2. Make JSON request to libraryholdings.php using OCLCNumber and Zip Code
3. Retrieve JSON object
4. Loop through holdings returned via JSON
5. For each holding add point to map with InfoWindow

## Code Sample: Journal Enhancement Script

### The HTML

```
<html>
<head>
<meta http-equiv="content-type" content="text/html; charset=ISO-8859-1">
<link type="text/css" href="../css/base/ui.all.css" rel="stylesheet" />
<script src="../script_libraries/jquery.js"></script>
<script src="../script_libraries/jquery-ui.js"></script>
<script src="journal_list.js"></script>
</head>
<body>

<div class="Journal"><p><a href="http://www.worldcat.org/oclc/2243594">Journal of academic
librarianship.</a>
<span id="issn" style="display: none;">0099-1333</span>
<br/>
<span id="toc"></span>
</p>
</div>

<div class="Journal"><p><a href="http://www.worldcat.org/oclc/7251238">Library issues.</a>
<span id="issn" style="display: none;">0734-3035</span>
<br/>
<span id="toc"></span>
</p>
</div>

</body>
</html>
```

### Explanation

This is a basic HTML page which contains two journal results.

1. Add JQuery and JQuery UI script libraries
2. Add JQuery UI CSS
3. Add the journal\_list.js script
4. Make sure your ISSNs are in spans with an id = issn

### The Javascripts

```
$(document).ready(function(){

    $(function getISSNs(){
```

```

        $('.Journal').each(
            function (i){
                var ISSN = $(this).find('span#issn').text();
                var self = this; // capture this
                $.getJSON( "http://xissn.worldcat.org/webservices/xid/issn/"+
ISSN + "?method=getMetadata&format=json&fl=*&callback=?",
                    function (data) {
                        if (data.stat = 'ok') {
                            if (data.group[0].list[0].peerreview == 'Y') {
                                $(self).find('span#issn').after('<span
id="peer_reviewed"> Peer Reviewed</span>');
                            }
                            if (data.group[0].list[0].rssurl.length > 0) {
                                var rss = data.group[0].list[0].rssurl;
                                $(self).find('span#toc').append('<a
href="">See Latest Table of Contents</a>');
                            }
                        }
                    });
            });
        );
    });

    $('span#toc').click(function(event){
        event.preventDefault();
        var ISSN = $(this).prevAll('span#issn').text();
        $(this).parents('div.Journal').append('<div id="journal_toc"></div>');
        var self = this; // capture this
        $.getJSON( "http://xissn.worldcat.org/webservices/xid/issn/"+ ISSN + "?
method=getMetadata&format=json&fl=*&callback=?",
            function (data) {
                rss = data.group[0].list[0].rssurl;
                $.get("feed_process.php?feedURL=" + rss,
                    function(data){
                        $
(self).parents('div.Journal').find('div#journal_toc').append(data);
                        $("#journal_toc").dialog({
                            bgiframe: true,
                            height: 500,
                            width: 600,
                            modal: true,
                            title: 'Table of Contents for Recent Issue'
                        });
                    });
            });
    });
});

```

```
});
```

## Explanation

1. Find all the divs with the Journal class
2. Within each of these get the value of span with the issn ID
3. Take ISSN and send it to xISSN service to retrieve information in JSON format
4. If the peerreview field in the JSON is set to 'Y' add a new span with the text “Peer Reviewed”
5. If the rssurl field exists create a link that says “See Latest Table of Contents”
6. Create a function which handles what happens when “See Latest Table of Contents” link is clicked
  - a) Figure out which Journal div the link which is being in is clicked
  - b) Get the ISSN from this particular div.Journal
  - c) Create a new div with the ID = journal\_toc
  - d) Pass the ISSN to xISSN to retrieve the rssurl field
  - e) Take the rssurl field and send it to the feed\_process.php
  - f) Insert data retrieved from feed\_process.php script into new div#journal\_toc
  - g) Use JQuery-UI to define div#journal\_toc as a dialog window which pops-up

## The PHP script

```
<?php
if ( strlen($_REQUEST['feedURL']) > 0) {
    $feed_url = $_REQUEST['feedURL'];
    require_once('simplepie.inc');

    $feed = new SimplePie();
    $feed->set_feed_url($feed_url);
    $feed->init();
    $feed->handle_content_type();

    echo '<ul>';
    foreach ($feed->get_items(0, 5) as $item) {
        echo '<li><a href="'. $item->get_permalink() . "'>' . $item->get_title() . '</a></li>';
    }
    echo '</ul>';
    echo '<p><a href="' . $feed->get_permalink() . "'>' . $feed->get_title() . '</a></p>';
} else {
    echo '<p>Please submit a valid feed URL</p>';
}
?>
```

## Explanation

1. Get the feedURL passed to the script
2. Load the SimplePie PHP Library
3. Use passed feedURL to to request the feed

4. Parse the first 5 items in the feed into a bulleted list

## Code Sample: New York Times Book Reviews Script

### The HTML

```
<html>
<head>
<meta http-equiv="content-type" content="text/html; charset=UTF-8"/>
<title>The girl with the dragon tattoo </title>
<script type="text/javascript" language="JavaScript" src="..script_libraries/demo_config.js"></script>
<script type="text/javascript" language="JavaScript" src="..script_libraries/jquery.js"></script>
<script type="text/javascript" language="JavaScript" src="..script_libraries/jquery-ui.js"></script>
<script src="http://books.google.com/books/previewlib.js"></script>
<script type="text/javascript" language="JavaScript" src="nyt.js"></script>
<link type="text/css" href="..css/base/ui.all.css" rel="stylesheet" />

<link rel="stylesheet" type="text/css" media="all" href="..mobile_cat/mobile.css" />

</head>
<body>
<div id="oclcSym"></div>
<div class="record">
...
  <tr>
    <th>ISBN</th>
    <td>9780307269751</td>
  </tr>
  <tr>
...

  <!--Add Ratings and Reviews -->

  <div id="ratings">
    <h4><a href="http://www.librarything.com/isbn/9780307269751">Ratings from
LibraryThing</a></h4>
    <p><span id="LT_9780307269751"></span></p>
  </div>

  <div id="fulltext">
    <script type="text/javascript" language="JavaScript">
GBS_insertPreviewButtonPopup("ISBN:9780307269751");
</script>
  </div>
...
</body>
```

</html>

## Explanation

1. Make sure to load the JQuery and JQuery-UI code libraries.php
2. Make sure the nyt.js javascript is loaded
3. Create a table which had a row with the ISBN information
4. Make sure div#ratings exists

## The Javascript

```
$(function NYTBestsellers() {
    var ISBN = $("th:contains('ISBN')").next().text();
    if (ISBN.length > 0) {
        $.getJSON( server_path + "/nyt_json.php?isbn=" + ISBN + "&callback=?",
            function (data) {
                // check for status being ok to proceed
                if (data.num_results > 0) {
                    $('div#ratings').after('<div id="nyt"><p><strong>New York Times
Bestseller</strong></p></div>');
                    // figure out if there is a review and link to it
                    if (data.reviews[0].book_review_link.length > 0) {
                        $('div#nyt').append('<p><a href="' +
data.reviews[0].book_review_link + "'>See Review from New York Times</a></p>');
                    } else {
                        return;
                    }
                    if (data.reviews[0].sunday_review_link.length > 0) {
                        $('div#nyt').append('<p><a href="' +
data.reviews[0].sunday_review_link + "'>See Sunday Review from New York Times</a></p>');
                    } else {
                        return;
                    }
                }
            }
        );
    } else {
        return;
    }
});
```

## Explanation

1. Find the ISBN field and get it
2. Check to make sure an ISBN was obtained
3. If so, take ISBN and send it to the nyt\_json.php script to retrieve information in JSON format
4. If num\_results is greater than 0, create a div to hold New York Times Review information and insert it after the div#ratings
5. Put into this div the text “New York Times Bestseller” and links to Review from New York Times and Sunday Review from New York Times if they exist

## The PHP

```
<?php
require_once('demo_config.inc');

if (strlen($_REQUEST['isbn']) > 0) {
    $isbn = $_REQUEST['isbn'];
}

if ((strpos($_SERVER['HTTP_REFERER'], $_server_address) > 0) or
is_null($_SERVER['HTTP_REFERER'])) {
    $domain = true;
} else {
    $domain = false;
}

if (strlen($_REQUEST['callback']) > 0) {
    echo $_REQUEST['callback'];
    echo '{';
} else {
    echo '{';
}
// Check to see that ISBN was passed

if ($isbn and $domain == true) {

    // build request
    $itemURL = 'http://api.nytimes.com/svc/books/v2/lists/best-sellers/history.xml?isbn=' . $isbn;

    $itemURL = $itemURL . '&api-key=' . $NYT_API ;

    // go get the XML
    $xml = simplexml_load_file($itemURL);
    $itemCount = count($xml->xpath('/book'));
    if ($itemCount > 0) {
        $weeks_on_list = $xml->xpath('/ranks_history/rank[position()=1]/weeks_on_list');
        echo "status": "ok",;
        echo "num_results": ' . $itemCount . ',;
        echo "weeks_on_list": ' . $weeks_on_list[0] . ',;
        echo "reviews": [;
        $id_number = 0;
        $reviewCount = count($xml->xpath('/reviews/review'));
        foreach($xml->xpath('/reviews/review') as $review ) {
            $field = simplexml_load_string($review->asXML());
            $id_number++;
            $book_review_link = $field->book_review_link;
            $sunday_review_link = $field->sunday_review_link;
```

```

        echo '{';
        echo "id_number": " . $id_number . "'";
        echo "book_review_link": " . $book_review_link . "'";
        echo "sunday_review_link": " . $sunday_review_link . "'";
        echo '};
        if ($id_number != $reviewCount) {
            echo '!';
        }
    echo ']';
}
} else {
    echo "status": "ok",';
    echo "num_results": 0';
}
} else {
    echo "status": "error missing parameter";
}

if (strlen($_REQUEST['callback']) > 0) {
    echo '});
} else {
    echo '};
}
?>

```

### Explanation

1. Get the ISBN from the query string
2. Build the request to the NY Times API based on the ISBN
3. Load the XML file returned
4. If at least one result is returned then build JSON including:
  - number of results
  - weeks on list
  - reviews
5. Loop through each review and return:
  - ☒ id\_number
  - ☒ book\_review\_link
  - ☒ sunday\_review\_link

## Code Sample: New York Times Movie Reviews Script

### The PHP

```
<div id="reviews">

<?php

//If Video, check New York Times for Movie Reviews
if ($format == 'video') {
    $nyt_movies_request = "http://api.nytimes.com/svc/movies/v2/reviews/search.xml?query=" .
str_replace(" ", "+", $title[0]) . "&api-key=" . $NYT_Movies_API;
    $nyt_movies_response = simplexml_load_file($nyt_movies_request);

    foreach ($nyt_movies_response->xpath('//review') as $movie) {
        if (strpos($publication_date[0], substr($movie->dvd_release_date, 0, 4)) > 0 and
strtolower($movie->display_title) == strtolower($title[0])) {
            $nyt_review_url = $movie->xpath('link[@type="article"]/url');
            echo '<h4>New York Times Movie Review</h4>';
            echo '<p>' . $movie->capsule_review . '</p>';
            echo '<p><a href="' . $nyt_review_url[0] . '">Full New York Times Review</a></p>';
        }
    }
}
?>

</div>
```

### Explanation

1. Check to see if the format of the item is video (format is obtained earlier in the script)
2. If format is video then build a request to the NY Times Movie Reviews API
3. Send the item title to the API as part of the request. Make sure to replace any spaces with “+”
4. Load the XML returned from the request
5. Loop through each review returned and check
  - a) that the publication year matches the year of the DVD release
  - b) that the item title matches the title of the movie being reviewed
6. If these two criteria are met, print the text “New York Times Movie Review” along with the capsule review and link to Full New York Times Review

## Code Sample: Last.fm Similar Artists

### The PHP

```
<?php
    // add code to get related Music from Last.fm
    if ($format == 'music recording') {
        echo '<div id="similar_artists">';
        echo '<h4>Other Artists You Might Like</h4>';
        echo '<ul style="list-style-type:none;">';
        $artist_fname = substr($author_name, strpos($author_name, ',')+2);
        $artist_lname = substr($author_name, 0, strpos($author_name, ','));
        $last_fm_request = 'http://ws.audioscrobbler.com/2.0/?
method=artist.getsimilar&artist=' . $artist_fname . ' . $artist_lname . '&limit=10&api_key=' .
$lastfm_developer_key ;
        $last_fm_xml = simplexml_load_file($last_fm_request) or die("xml response not
loading");
        foreach($last_fm_xml->xpath('//artist') as $artist ) {
            $field = simplexml_load_string($artist->asXML());
            $name = $field->name;
            $image = $field->xpath("image[@size='medium']");
            echo '<li style="padding-bottom:1em;"><img src="" . $image[0] ."" alt="" .
$name . ""/>';
            echo '<br/>';
            echo '<a href=" ../mobile_cat/catalog.php?AddWorldCatSearch-
SearchType=srw.au&AddWorldCatSearch-SearchString=' . $name . "">' . $name . '</a></li>';
        }
        echo '</ul>';
        echo '<p><a href="http://www.last.fm/music/' . $artist_fname . '+' .
$artist_lname . '/+similar">Other Similar Artists via Last.fm</a></p>';
        echo '</div>';
    }
?>
```

### Explanation

1. Check to see if the format of the item is a music recording (format is obtained earlier in the script)
2. If format is music recording then build a request to the Last.fm API
3. Break up the author name into last name and first name to reorder. Last.fm wants it sent FirstName LastName in query
4. Send the item author first and last name to the API as part of the request.
5. Load the XML returned from the request
6. Loop through each artist returned
7. Get the medium sized image from each artist returned
8. Print the image of the artists as well as their name which is a link back to the search page to perform an author search for this artist.