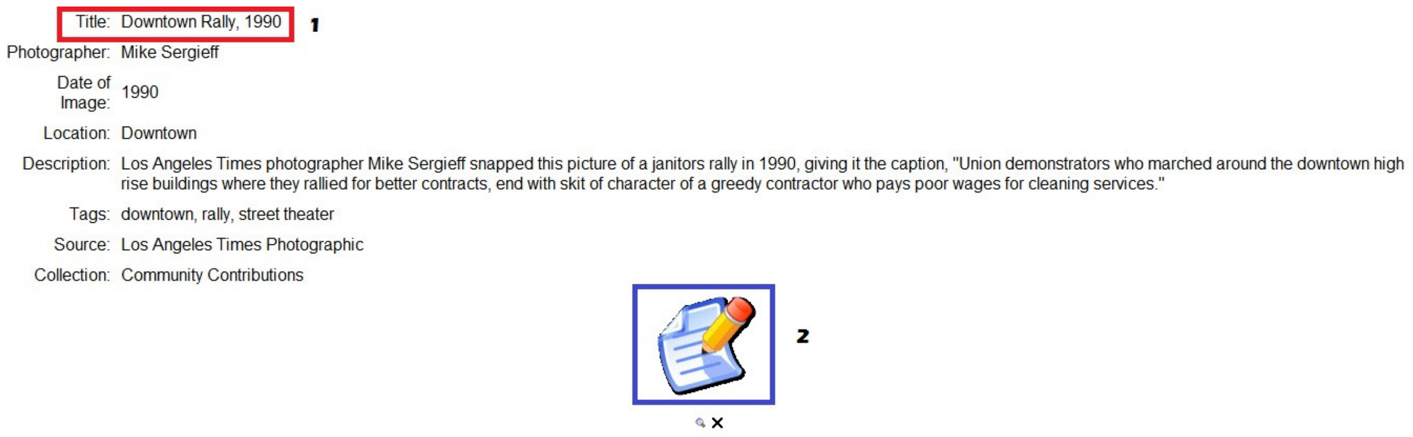
Linking database information through JavaScript template in Moodle Database Activity Module

Say you have two databases in your Moodle site, and you want to send data from one database to the other one to fill in a form. An easy way to do this is by sending the required data through the url and utilizing JavaScript.

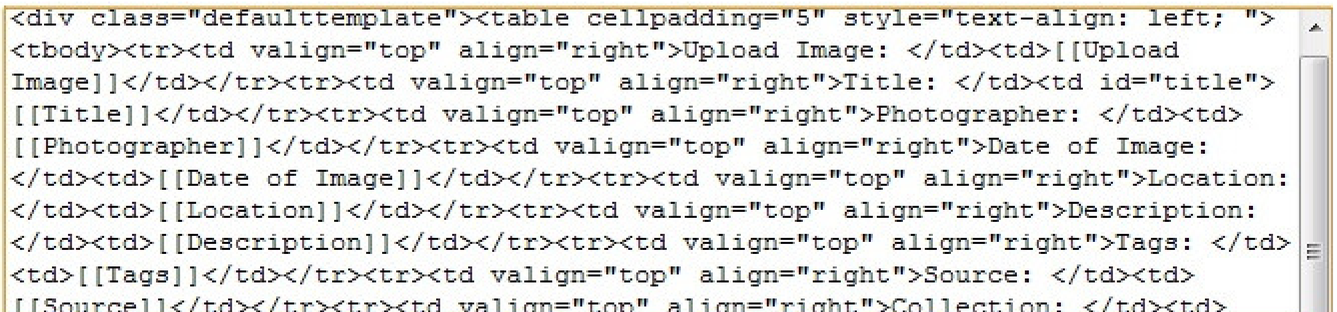
Sample 1



Title:	Downtown Rally, 1990
Photographer:	Mike Sergieff
Date of Image:	1990
Location:	Downtown
Description:	Los Angeles Times photographer Mike Sergieff snapped this picture of a janitors rally in 1990, giving it the caption, "Union demonstrators who marched around the downtown high rise buildings where they rallied for better contracts, end with skit of character of a greedy contractor who pays poor wages for cleaning services."
Tags:	downtown, rally, street theater
Source:	Los Angeles Times Photographic
Collection:	Community Contributions

In Sample 1, lets say we want to send data like the title to another database. Through HTML and JavaScript, we turn the paper icon into a button, as you can see in Sample 2. This is an <a tag, meaning an anchor tag. Automatically, through HTML, this will lead to the link specified, meaning the `href=https://classes.sscnet.ucla.edu/mod/data/edit.php?d=91&url=##moreurl##&title=[[Title]]`

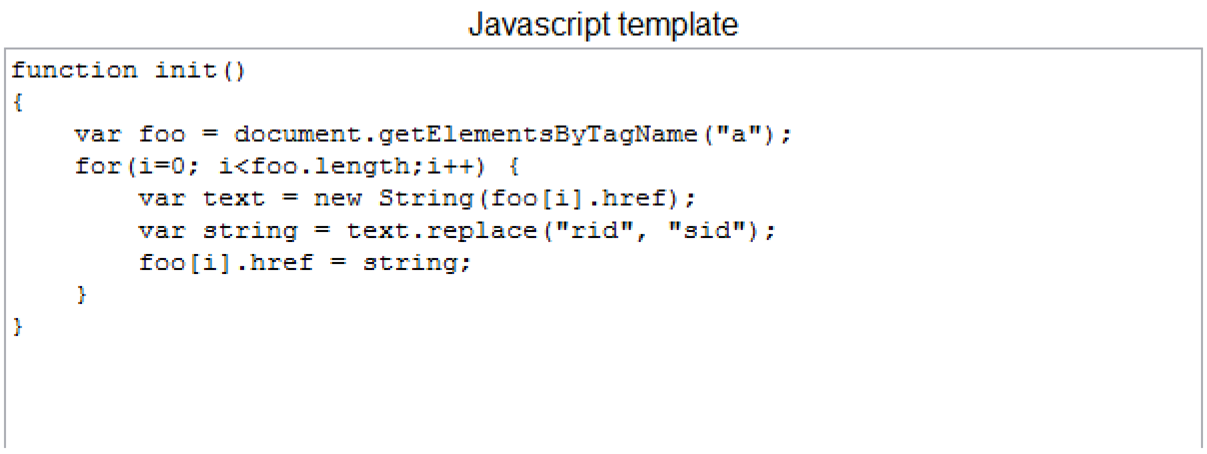
Sample 2



```
<div class="defaulttemplate"><table cellpadding="5" style="text-align: left; ">
<tbody><tr><td valign="top" align="right">Upload Image: </td><td>[[Upload
Image]]</td></tr><tr><td valign="top" align="right">Title: </td><td id="title">
[[Title]]</td></tr><tr><td valign="top" align="right">Photographer: </td><td>
[[Photographer]]</td></tr><tr><td valign="top" align="right">Date of Image:
</td><td>[[Date of Image]]</td></tr><tr><td valign="top" align="right">Location:
</td><td>[[Location]]</td></tr><tr><td valign="top" align="right">Description:
</td><td>[[Description]]</td></tr><tr><td valign="top" align="right">Tags: </td>
<td>[[Tags]]</td></tr><tr><td valign="top" align="right">Source: </td><td>
[[Source]]</td></tr><tr><td valign="top" align="right">Collection: </td><td>
```

So right now, we have a paper icon that links to a specified website; however, this website is not correct because the ##moreurl## contains a string that creates an error. This is where javascript comes in, which will dynamically change the website to the correct one when clicked, explaining why the <a tag has the **onclick = "init()"**. When clicked, the function **init()** will run.

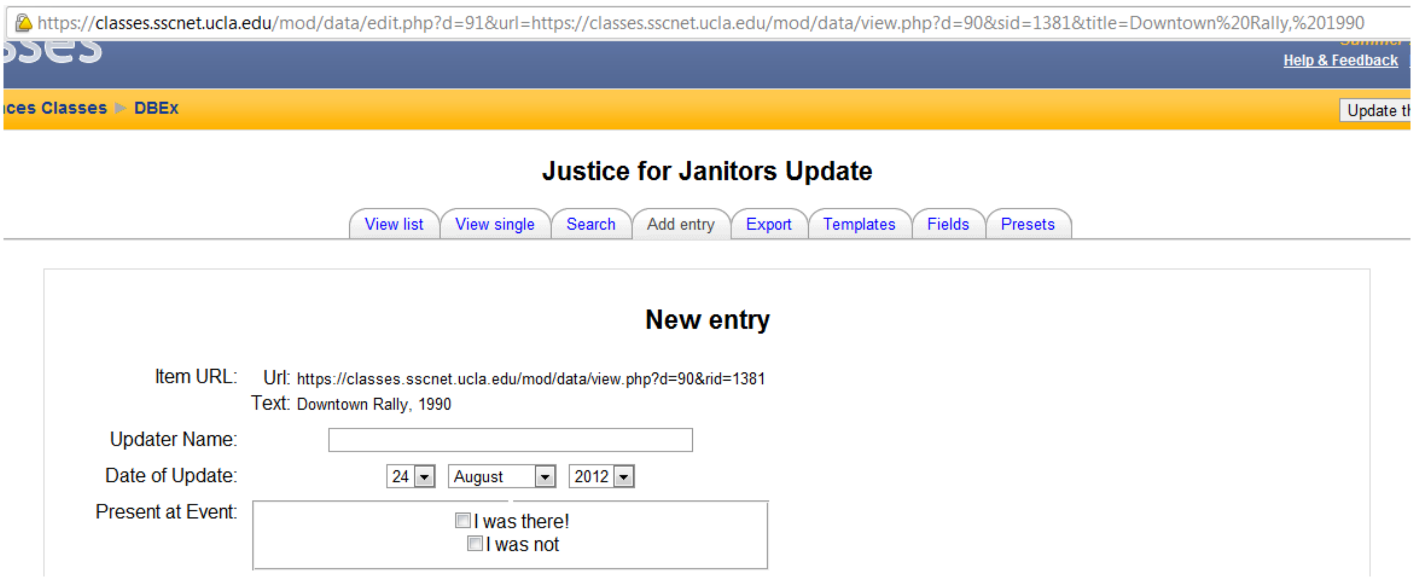
Sample 3


Javascript template

```
function init()
{
    var foo = document.getElementsByTagName("a");
    for(i=0; i<foo.length;i++) {
        var text = new String(foo[i].href);
        var string = text.replace("rid", "sid");
        foo[i].href = string;
    }
}
```

In Sample 3, we see the **init()** function. The first line declares that the variable foo is equal to all the elements with the tag name "a". This fills foo with an array of all the anchor tags in the document, including the one described above in Sample 2.
Running through the array with a for loop, the variable text is defined as the anchor tag's href, the website it links to, and then it replaces the string "rid" with "sid". By using JavaScript, when the button is clicked, it will search through the anchor tags and replace the required string so the website linked will be correct.

Sample 4


https://classes.sscnet.ucla.edu/mod/data/edit.php?d=91&url=https://classes.sscnet.ucla.edu/mod/data/view.php?d=90&sid=1381&title=Downtown%20Rally,%201990

Help & Feedback
ces Classes ▶ DBEx
Update th

Justice for Janitors Update

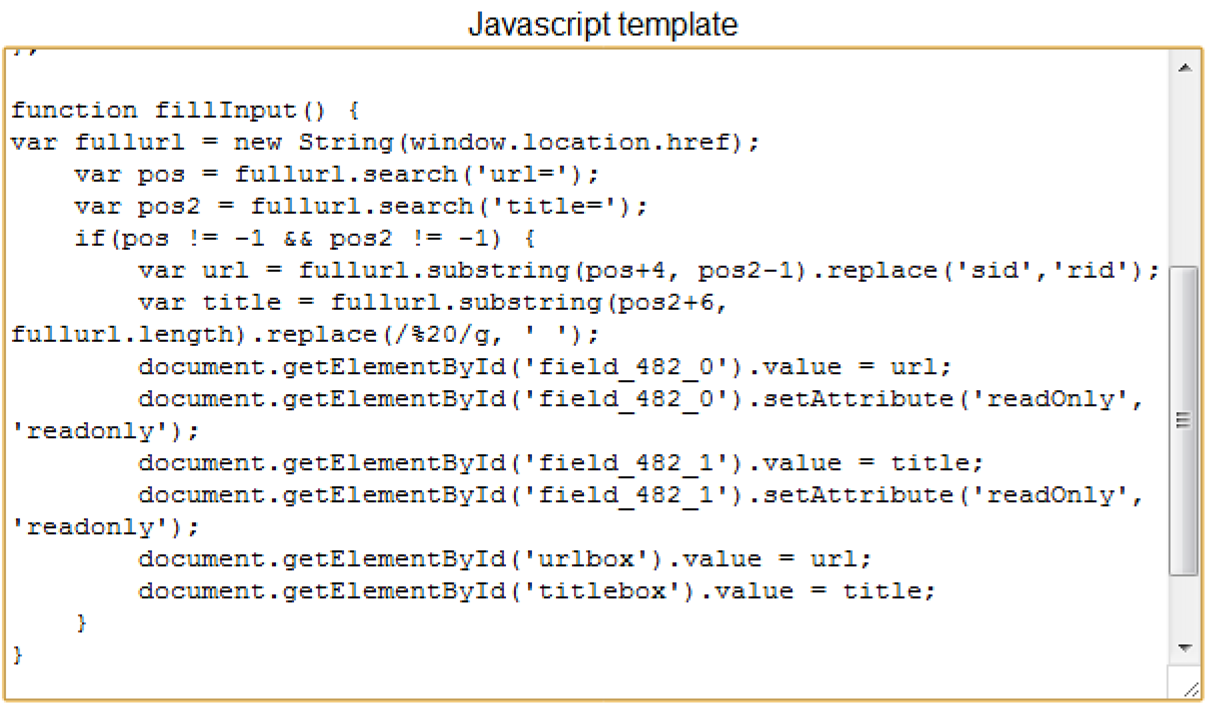
View list | View single | Search | Add entry | Export | Templates | Fields | Presets

New entry

Item URL:	Url: https://classes.sscnet.ucla.edu/mod/data/view.php?d=90&rid=1381 Text: Downtown Rally, 1990
Updater Name:	
Date of Update:	24 August 2012
Present at Event:	☐ I was there! ☐ I was not

In Sample 4, we can see the url and the displayed database. The complete url contains the needed data of the partial url, which is the ##moreurl## but with rid = sid. The title has also been sent through the url, but to place it in the database requires more JavaScript.

Sample 5

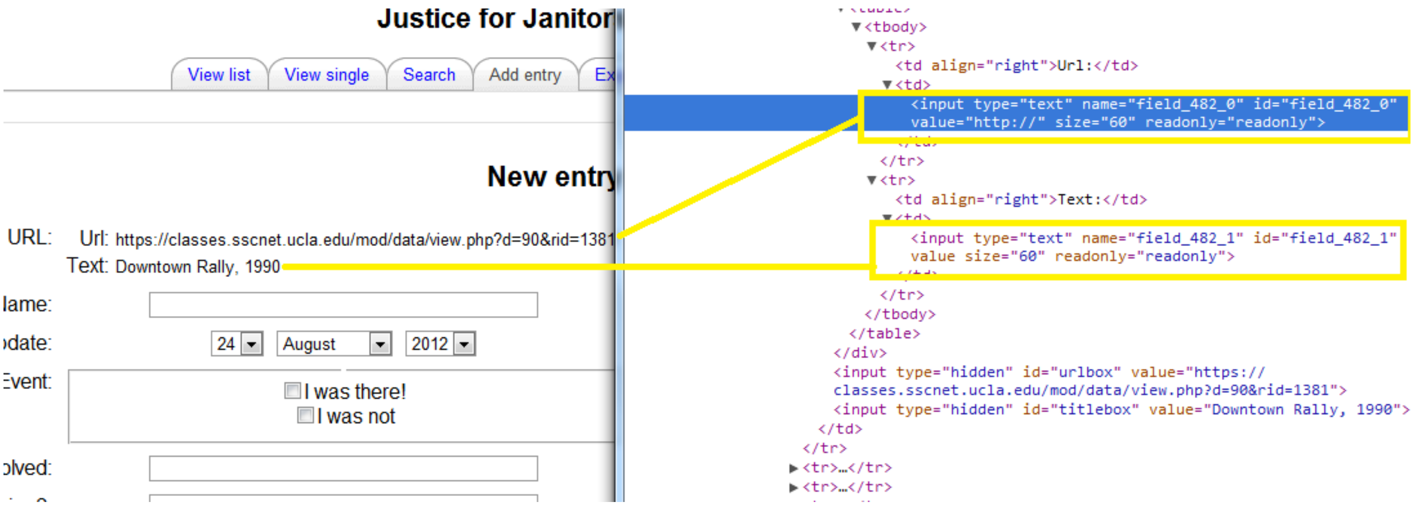


Javascript template

```
function fillInput() {
var fullurl = new String(window.location.href);
    var pos = fullurl.search('url=');
    var pos2 = fullurl.search('title=');
    if(pos != -1 && pos2 != -1) {
        var url = fullurl.substring(pos+4, pos2-1).replace('sid','rid');
        var title = fullurl.substring(pos2+6,
fullurl.length).replace(/%20/g, ' ');
        document.getElementById('field_482_0').value = url;
        document.getElementById('field_482_0').setAttribute('readOnly',
'readonly');
        document.getElementById('field_482_1').value = title;
        document.getElementById('field_482_1').setAttribute('readOnly',
'readonly');
        document.getElementById('urlbox').value = url;
        document.getElementById('titlebox').value = title;
    }
}
```

In Sample 5, the function **fillInput()** will take the url and parse the required information into the template in Sample 4. In this function, we want the ##moreurl## and the title — data from the previous database that was transferred over. We search for the beginning of each field in the complete url, placing values in the variables pos/pos2. If both variables are defined, meaning they were found in the url, we can get the required data through substrings.

Sample 6



URL: Url: https://classes.sscnet.ucla.edu/mod/data/view.php?d=90&rid=1381
Text: Downtown Rally, 1990

24 August 2012

I was there!
I was not

```
▼<tbody>
  ▼<tr>
      <td align="right">Url:</td>
    ▼<td>
        <input type="text" name="field_482_0" id="field_482_0"
        value="http://" size="60" readonly="readonly">
    </tr>
  ▼<tr>
      <td align="right">Text:</td>
        <input type="text" name="field_482_1" id="field_482_1"
        value size="60" readonly="readonly">
    </tr>
  </tbody>
</table>
</div>
<input type="hidden" id="urlbox" value="https://
classes.sscnet.ucla.edu/mod/data/view.php?d=90&rid=1381">
<input type="hidden" id="titlebox" value="Downtown Rally, 1990">
</td>
</tr>
▶<tr>…</tr>
▶<tr>…</tr>
```

Now that we have the required data, we transfer them to the template by searching through the document. Going back to Sample 4, when we right click anywhere on the webpage, and then go to inspect element/view source, we can see the raw HTML of the page. Going over the template values in Sample 6, we see that **field_482_0** is the input box for the **URL:** and **field_482_1** is the **Title:** input box. Through JavaScript, we can dynamically fill in those values with the data from the url.

Thus our end result is seen in Sample 4 where we have a url with the data passed from one database to another to fill in a template!

Revision #2
Created Fri, Aug 24, 2012 8:59 PM by WONG, SARAH L
Updated Fri, Aug 24, 2012 9:05 PM by WONG, SARAH L