# Plugin JQ

This [wiki plugin](), introduced in [Tiki3](), allows jQuery JavaScript code to be inserted on a page.

See also [PluginHTML]() and [PluginJS]().

## 1.1. REQUIREMENTS

The jQuery feature must be enabled at **Control Panels > Features > Experimental > jQuery (checkbox)**.

## 1.2. PARAMETERS

Add jQuery JavaScript code
*Introduced in Tiki 3.*
[Go to the source code]()
*Preferences required:* wikiplugin_jq

| Parameters | Accepted Values |
| --- | --- |
| (body of plugin) | |
| lang | |
| nojquery | |

| Description |
| --- |
| JavaScript code |
| Language to apply JQuery to |
| Optional markup for when JavaScript is off |

| Default | Since |
|---------|-------|
| | 13.0 |
| | 3.0 |

`notonready`

# Do not execute on document ready (execute inline)

3.0

× 
⚠ Syntax changed since Tiki 7

Keep in mind that default syntax for tracker fields changed since Tiki7, so that when referring them from within plugin jQuery, you have to update the syntax.

Tracker field IDs and names have changed to improve consistency: all tracker fields now use `ins_XX`, where `XX` is the field Id.

For more information, read: http://doc.tiki.org/Tiki7#Upgrade_notes

And basic syntax changed after Tiki 6:

A Simple jQuery line of code looks like this:

**Standard jQuery**

$("p").show("slow");

When using the Tiki JQ Plugin in old versions 3 to 5, you needed to start you code with `jq`:

**Tiki jQuery Call**

$jq("p").show("slow");

# 1.3. EXAMPLES

## 1.3.1. SIMPLE ONE

A Simple jQuery line of code looks like this:

**Standard jQuery**

```
$("p").show("slow");
```

## 1.3.2. USE JQUERY DATE PICKER TO INPUT DATE IN A FORM

Example with 2 date fields, one for the "date in" and one for the "date out".
*name = the name of your input field, you can use any name permitted by html5*

```
{JQ()}$( 'input[name="date_in"]' ).datepicker(); $( 'input[name="date_out"]' ).datepicker();{JQ}
```

Note: you can change the date format (or may be other parameter) directly in the plugin code.

For example here we changed the date format for yyyy-mm-dd

```
{JQ()}$( 'input[name="cf_1214"]' ).datepicker({dateFormat: "yy-mm-dd"}); $( 'input[name="cf_1216"]'
).datepicker({dateFormat: "yy-mm-dd"});{JQ}
```

To check proper parameters refer to : http://api.jqueryui.com/datepicker/

## 1.3.3. USING JQ PLUGIN TO ENHANCE TRACKER PLUGIN

You can show or hide fields (the whole div containing it) based on another input condition.
Here a sample that display a field (the whole div containing it) if a checkbox with the value "other" is checked.
*If you use it don't forget to change the checkbox name and the div class*

**jQuery code to display a field based on a checkbox**

```
{JQ()} $('input[value="other"]').change(function(){ if ($(this).is(":checked")) { $('.tracker_field118').show(); } else {
$('.tracker_field118').hide(); } }).change(); {JQ}
```

## 1.3.4. SPECIAL NOTES:

When following above guide, you can only call the included script from the getScript handler. That means, you cannot just include the script under look and feel area, and then use that script from the JQ plugin - the order of loading the script and using it, will conflict. In short: Call your custom script from the getScript handler.~/tc~

## 1.3.5. USING JQ PLUGIN TO ENHANCE PRETTY TRACKER FORMS

Q plugin won't substitute the pretty tracker form, fields or design. In one sentence; With JQ plugin you can **add functionality** to a already working pretty tracker.

A good practice is to create your tracker, your wiki page, do the design you need and test that everything is working fine. Then without touching what's been already done you can add the JQ plugin at the bottom of the wiki page.

fields IDs are a little different than the ones used in the tracker or the pretty tracker. For instance a tracker field id **1** which is a tex field should specify as `ins_1` . If it is a textarea it is `ins_1` also, If a date you have to specify (if need all of them) `ins_1Day` , `ins_1Month` , `ins_1year` . Note also that the syntax use the html form element. Input is an input field, select if for selectable options, etc… To check what is exactly required to use JQ with your tracker you can look at the source of the page (use Firebug or any other tool) and use the exact syntax.

## 1.3.6. PREFILL TRACKER FIELDS

### 1.3.6.1. DIRECTLY AT THE WIKI PAGE

**Basic Example A**

For instance, if we want to pre-load some values in a tracker plugin form, where you have text field id's 20, 21, 22, and a textarea field id 23, and you want some text to be preloaded in the fields 22 and 23 of your form, we can use something like the syntax below (in a tiki 3, 4, 5 or 6 site as such):

**jQuery code since Tiki7+ to add in the wiki page where the tracker plugin call is located**

```
{$f_22} {JQ()} $("#ins_22").val("foo or bar"); // to prefill a text field, option1 $("#ins_23").val("first line of text\nsecond line\nthir
line"); // to prefill a text area $("#ins_3572").text("Text"); // to prefill a text field, option 2
$('[name="ins[3496]"]')[1].checked=true; // to prefill a checkbox $("select[name=ins_91]").val("foo") // to prefill a drop down fiel
$("select[name='ins_92[]']").val("6") // to prefill a category field shown as a drop down {JQ}
```

And if you want those values to be pre-filled in your form only when there is not previously saved content in that tracker field for that item (remember you can edit tracker items also from wiki pages), the syntax is:

**jQuery code since Tiki7+ to add in the wiki page where the tracker plugin call is located**

{JQ()} if (!$("#ins_22").val()) { $("#ins_22").val("foo or bar"); } if (!$("#ins_23").val()) { $("#ins_23").val("first line of text\nsecond line\nthird line"); } if (!$("#ins_3572").val()) { $("#ins_3572").text("Text"); } if (!$('[name="ins[3496]"]')[1].checked) { $('[name="ins[3496]"]')[1].checked=true; } if (!$("select[name=ins_91]")[0].selectedIndex) { $("select[name=ins_91]").val("foo") } {JQ}

## Basic Example B

You can use some value defined in some other HTML `input` in the same wiki page:

{HTML()}<input type="hidden" name="testingr" id="testingr" value="temp3" />{HTML}

In this case below, field 91 is a dropdown field:

{JQ()} if (!$("select[name=ins_91]")[0].selectedIndex) $("select[name=ins_91]").val($("#testingr").val()) {JQ}

## Basic Example C
Using the id & name "routput" generated by a PluginR call:

{DIV(class="hidden")}{R()}a<-1:10; cat(mean(a));{R}{DIV} {JQ()} if (!$("#other_ins_92").val()) $("#other_ins_92").val($("#routput").text()){JQ}

## Basic Example D

You can use some value defined in some other html input in the same wiki page:

```
{HTML()}<input type="hidden" name="mydate" id="mydate" value="{display name="tracker_field_mydate" format="date"}" />{HTML}
```

In this case below, field 46 is a text field, and we want it hidden for the end user, that's why we end up appending the last jQuery part `.parents(".form-group").hide()`:

```
{JQ()} if (!$("#ins_46")) $("#ins_46").val($("#mydate").val()).parents(".form-group").hide() ; {JQ}
```

## 1.3.6.2. BY MEANS OF A WIKI PAGE USED AS A SMARTY TEMPLATE ('PRETTY TRACKERS')

This case is needed when you are adding some prefill by means of a smarty template which resides on a wiki page.

This is very useful when you use PluginR in advanced mode with Pretty Trackers, for instance, to mix R code and HTML forms to collect parameter data managed by Tiki trackers. The Smarty template allows inserting data from tracker item fields in R scripts.

Example:

**Prefilling tracker item fields since Tiki7+ from smarty templates in wiki pages**

```
{wikiplugin _name=jq}if (!$("#ins_22").val()) $("#ins_22").val("foo or bar"); {/wikiplugin} {wikiplugin _name=jq}if (!$("#ins_23").val()) $("#ins_23").val("first line of text\nsecond line\nthird line"); {/wikiplugin} {wiki} * field 22 is: {$f_22} * field
```

23 is: {$f_23} {/wiki}

## 1.3.7. PRE-FILL WITH CSS STYLES

The pre-filling performed by jQuery can be done with some style, to indicate the user that this content comes from a suggestion when the field in the tracker form is empty (yet) for that item.

Thus, in the first example above ("**Prefill tracker fields > Directly at the wiki page**"), if you want those values to be prefilled in your form only when there is not previously saved content in that tracker field for that item **and** with some CSS style like `background-color` light yellow, font `color` grey and in *italics*, the syntax is:

**jQuery code since Tiki7+ to add in the wiki page where the tracker plugin call is located**

```
{JQ()} if (!$("#ins_22").val()) { $("#ins_22").val("foo or bar"); $("#ins_22") .css('background-color', 'lightyellow') .css('font-style',
'italic') .css('color', 'grey') .focus(function() { $(this) .css('background-color', '') .css('font-style', '') .css('color', ''); }); } if
(!$("#ins_23").val()) { $("#ins_23").val("first line of text\nsecond line\nthird line"); $("#ins_23") .css('background-color',
'lightyellow') .css('font-style', 'italic') .css('color', 'grey') .focus(function() { $(this) .css('background-color', '') .css('font-style', '')
.css('color', ''); }); } {JQ}
```

## 1.3.8. CONDITIONAL DISPLAY OF FIELDS

### 1.3.8.1. EXAMPLE 1

Imagine that you have a form with 15 fields. Field number 10 is a checkbox checked by default, and you want that when field 10 is checked, field 11 (a text area, for instance) is shown. But when field 10 is not checked, fields 12, 13, 14 & 15 are shown instead.

You can get that behavior with the following jQuery syntax:

```
{JQ()} $("input[name=ins_10]").click(function(){ if (this.checked) { $("input[name=ins_10]").parents("tr:first + tr ~
tr").fadeOut('fast'); $("input[name=ins_10]").parents("tr:first + tr ~ tr:first").fadeIn('fast'); } else {
$("input[name=ins_10]").parents("tr:first + tr ~ tr").fadeIn('fast'); $("input[name=ins_10]").parents("tr:first + tr ~
tr:first").fadeOut('fast'); } }); $("input[name=ins_10]").parents("tr:first + tr ~ tr").hide(); $("input[name=ins_10]").parents("tr:firs
+ tr ~ tr:first").show(); {JQ}
```

**Explanation "for humans"**:

.parents("tr:first + tr ~ tr") means: match the first parent which is a <tr>, go to the next <tr> ("+ tr") and then pick all following
<tr> siblings ("~ tr")

"tr:first + tr ~ tr:first" is much the same but only select the first sibling

1.3.8.2. EXAMPLE 2

Imagine that you have a form with 15 fields, where you always want to show the first 9 fields, and the last two (14 and 15), and fiel
10 controls which fields are shown from the numbers 11, 12, 13. Field number 10 is a drop down box, with possible values: empty
value (default), TRUE and FALSE. And imagine that you want that when field 10 is set as FALSE, field 11 (a text area, for instance) i
shown but not fields 12 (text field), and 13 (drop down with other text field). But when field 10 is set as TRUE, field 11 is hidden, an
fields 12 and 13 are shown. And when field 10 is not set (empty value, the default option), fields 11, 12 & 13 are all hidden.

You can get the following behavior with the following jQuery syntax:

**Only the text fields for typing are hidden, not the whole row of the html form**

```
{JQ()} $("select[name=ins_10]").change(function(){ if ($(this).val() === 'FALSE') { $("input[name=ins_11]").hide();
$("input[name=ins_12]").show(); $("select[name=ins_13]").show(); $("input[name=other_ins_13]").show(); } else if ($(this).val(
=== 'TRUE') { $("input[name=ins_11]").show(); $("input[name=ins_12]").hide(); $("select[name=ins_13]").hide();
```

```
$("input[name=other_ins_13]").hide(); } else { $("input[name=ins_11]").hide(); $("input[name=ins_12]").hide();
$("select[name=ins_13]").hide(); $("input[name=other_ins_13]").hide(); } }); $("input[name=ins_11]").hide();
$("input[name=ins_12]").hide(); $("select[name=ins_13]").hide(); $("input[name=other_ins_13]").hide(); {JQ}
```

## 1.3.8.3. EXAMPLE 3

This is the same example as Example 2 but in this case, you want to hide the whole row from the table in the form (and not just hiding the field for typing).

You can get the following behavior with the following jQuery syntax:

**The whole row of the form is hidden for the tracker fields indicated**

```
{JQ()} $("select[name=ins_10]").change(function(){ if ($(this).val() === 'FALSE') {
$("input[name=ins_11]").parents("tr:first").hide(); $("input[name=ins_12]").parents("tr:first").show();
$("input[name=other_ins_13]").parents("tr:first").show(); } else if ($(this).val() === 'TRUE') {
$("input[name=ins_11]").parents("tr:first").show(); $("input[name=ins_12]").parents("tr:first").hide();
$("input[name=other_ins_13]").parents("tr:first").hide(); } else { $("input[name=ins_11]").parents("tr:first").hide();
$("input[name=ins_12]").parents("tr:first").hide(); $("input[name=other_ins_13]").parents("tr:first").hide(); } });
$("input[name=ins_11]").parents("tr:first").hide(); $("input[name=ins_12]").parents("tr:first").hide();
$("input[name=other_ins_13]").parents("tr:first").hide(); {JQ}
```

## 1.3.8.4. EXAMPLE 4

This is an example that shows how to iterate over a collection of input text fields in a tracker. First the text field is selected and an event that responds to keyboard input is set for each text field. When text is entered, the text field below the active one, will be shown. The process repeats until there are no more hidden text fields left to show. Note: this code will only work with 1 tracker on

the page. If you include more trackers on the same page, the script needs to be modified a bit. (see Example 4.1 below)

**The whole row of the form is hidden for the tracker fields indicated**

```
{JQ()} $(".wikiplugin_tracker input").each(function() { $(this).keyup(function() { $(this).parents("tr").next().show("slow", "swing");
}).parents("tr").hide(); }); $(".wikiplugin_tracker input:first").parents("tr").show(); {JQ}
```

## 1.3.8.5. EXAMPLE 4.1

This example shows how to achieve the same functionality as above, but with two different trackers on the same page.

**The whole row of the form is hidden for the tracker fields indicated**

```
{JQ()} $("#editItemForm1 input").each(function() { $(this).keyup(function() { $(this).parents("tr").next().show("slow", "swing");
}).parents("tr").hide(); }); $("#editItemForm1 input:first[type=text]").parents("tr").show(); $("#editItemForm2
input").each(function() { $(this).keyup(function() { $(this).parents("tr").next().show("slow", "swing"); }).parents("tr").hide(); });
$("#editItemForm2 input:first[type=text]").parents("tr").show(); {JQ}
```

It is worth noting that `#editItemForm1` is the form containing the first tracker and `#editItemForm2` is the form containing the second tracker.

## 1.3.8.6. EXAMPLE 5

Hide a field shown in the tracker form which does not contain any text field or checkbox, etc. That's the case of the auto-increment type field, which is shown in the PluginTracker form at item creation time, but it could be very well hidden for the sake of simplicity.

If the auto-increment tracker field has id 149, then the code is:

## The whole row of the form is hidden for the auto-increment tracker field indicated

{JQ()} $("label[for='ins_149']").parents("tr:first").hide(); {JQ}

### 1.3.8.7. EXAMPLE 6

This example has 1 radio button fields (field 38) as 0=no, or 1=yes. And when the user presses in the option 1=yes, there are 3 fields shown below (fields 39, 46, 47).

In addition, the field with the radio button has some CSS background color provided, which matches the background color of the fields shown when the option "yes" is selected. And when selecting "no", there is an attempt to clear any values which have been provided already to the fields that are being hidden.

**Code**

```
{JQ()} $("input[name=ins_38]").change(function(){ if ($(this).val() === '0') {
$("input[name=ins_39]").parents("tr:first").hide("slow"); $("input[name=ins_39]").val("");
$("input[name=ins_46]").parents("tr:first").hide("slow"); $("input[name=ins_46]").val("");
$("input[name=ins_47]").parents("tr:first").hide("slow"); $("input[name=ins_47]").val(""); } else if ($(this).val() === '1') {
$("input[name=ins_39]").parents("tr:first").show("slow"); $("input[name=ins_46]").parents("tr:first").show("slow");
$("input[name=ins_47]").parents("tr:first").show("slow"); $("input[name=ins_39]").parents("tr:first") .css('backgrund-color',
'lightyellow') .css('font-style', 'italic') .css('color', 'grey') .focus(function() { $(this) .css('background-color', '') .css('font-style', '')
.css('color', ''); }); $("input[name=ins_46]").parents("tr:first") .css('background-color', 'lightyellow') .css('font-style', 'italic')
.css('color', 'grey') .focus(function() { $(this) .css('background-color', '') .css('font-style', '') .css('color', ''); });
$("input[name=ins_47]").parents("tr:first") .css('background-color', 'lightyellow') .css('font-style', 'italic') .css('color', 'grey')
.focus(function() { $(this) .css('background-color', '') .css('font-style', '') .css('color', ''); }); } else {
$("input[name=ins_39]").parents("tr:first").hide("slow"); $("input[name=ins_39]").val("");
$("input[name=ins_46]").parents("tr:first").hide("slow"); $("input[name=ins_46]").val("");
```

```
$("input[name=ins_47]").parents("tr:first").hide("slow"); $("input[name=ins_47]").val(""); } });
$("input[name=ins_39]").parents("tr:first").hide(); $("input[name=ins_46]").parents("tr:first").hide();
$("input[name=ins_47]").parents("tr:first").hide(); $("input[name=ins_38]").parents("tr:first") .css('background-color',
'lightyellow') .focus(function() { $(this) .css('background-color', ''); }); {JQ}
```

### 1.3.8.8. OTHER EXAMPLES: DYNAMIC (PRETTY) TRACKERS

See http://doc.tiki.org/Pretty+Tracker#Dynamic_Pretty_Trackers to learn other examples of advanced usage of this JQ plugin with trackers.

### 1.3.9. EMPTY TEXTAREA FIELD ON LOAD

If textarea tracker field is number 64, this would be the code:

**Code**

```
{JQ()} $("textarea[name=ins_64]").val(""); {JQ}
```

### 1.3.10. SHOW TEXT INPUTS WITH CONTENT BUT NON-EDITABLE

In some cases you may want to show the contents of some tracker field to the user without that chance to edit it. You can achieve this with this simple jQuery code below, for a tracker field number 1, in a wiki page with PluginTracker and some itemId passed in the URL:

**Code**

{JQ()} $("#ins_1").prop('readonly', true); {JQ}

# 1.3.11. DISABLE SOME CHECKBOXES WHEN NO CONTENT FOUND IN THE CORRESPONDING CELL

The code below disabled checkboxes from the items in a table generated with PluginListExecute where the 7th column (6th for jQuery, since it starts counting from 0) has no content in the cell. In addition, it disables the select all checkbox, since for some reason allowed to select the disabled ones by the time of this writing:

## Code

{JQ()} $("#wplistexecute-1 tbody tr").each(function() { $("input[type=checkbox]", this).prop("disabled", $.trim($("td:nth(6)", this).text()) === ""); $("input[name=selectall]").prop("disabled", true); // select all seems to alway also select disabled ones }); {JQ}

See the type of output produced:



Click to expand

## 1.3.12. CHANGES IN DROPDOWN SELECTORS WITH CHOSEN JQ PLUGIN

You need to add some "**.trigger('chosen:updated');**"

From

**Code**

```
$("select[name=ins_162]").val("ongoing"); $("select[name='ins_160[]']").val("26");
```

To

**Code**

```
$("select[name=ins_162]").val("ongoing").trigger('chosen:updated');
$("select[name='ins_160[]']").val("26").trigger('chosen:updated');
```

Note that in the previous example, field 162 is a standard dropdown field, and field 160 is a category field with a dropdown display mode (notice the single quotes in that category-dropdown case surrounding the **ins_160[]**.

## 1.3.13. DISABLE DROPDOWN SELECTORS WITH CHOSEN JQ PLUGIN

You can disable the field 162 (being a dropdown field), when chosen jq plugin is in use, with a syntax like:

**Code**

## 1.3.14. REDUCE THE SELECTION IN A SELECT FIELD

We wish to restrict a long select field (field 658) to only 3 values (45,48,50) and hide the rest.
The first trick is, this script cannot function before Tiki has filled the select with values, which happens dynaically after the page has loaded.
For this to be possible, activate option *Look & Feel â†' Pagination â†' Object Selectors Events*
(since 21.5 and after)
Then :

**Code**

```
// Put the values we want to keep in an array var $to_be_kept = []; $to_be_kept.push(45); $to_be_kept.push(48);
$to_be_kept.push(50); // Detect that selects are drawn by Tiki $(document).on("ready.object_selector", function (event,
fieldName) { // Check that this is the select we are interested in // console.log('select "' + fieldName + '" is available')); if (
fieldName === "ins_658" ) { $('#trackerinput_658 select option').hide(); // Hide all select options $('#trackerinput_658 select
option[value=""]').show(); // If you wish, keep the empty selection available // Show the options we want to keep
$($to_be_kept).each(function(index,value){ $('#trackerinput_658 select option[value="trackeritem:' + value + '"]').show(); //
Syntax for a multi-select field } $('#trackerinput_658 select option').trigger("chosen:updated"); // If the chosen library is activated
} }
```

Field 658 is a multi-select field. For non multi-select, the line would be

```
$('#trackerinput_658 select option[value="' + value + '"]').show(); // Syntax for a single-selection field
```

## 1.3.15. HIDE FIELDS IN TIKI 15+ (BOOTSTRAP)

Use the CSS selector for the field 162 (being a dropdown field), like:

**Code**

```
{JQ()} $(".tracker_field162").parent().hide(); {JQ}
```

See examples at the profile "Conditional_Display_in_Forms_14"

## 1.3.16. HIDE FIELDS IN TIKI18+ (BOOTSTRAP)

**Code**

```
{JQ()} $("select[name=ins_1]").change(function(){ if ($("select[name=ins_1]").val() == 'example - example - My Value 1 with \'
apostrophe') { // \' $(".tracker_field2").show("slow", "swing"); $(".tracker_field3").hide("slow", "swing");
$(".tracker_field4").hide("slow", "swing"); } else if ($("select[name=ins_1]").val() == 'example - My Value 2') {
$(".tracker_field2").hide("slow", "swing"); $(".tracker_field3").show("slow", "swing"); $(".tracker_field4").show("slow", "swing"); }
else { $(".tracker_field2").hide("slow", "swing"); $(".tracker_field3").hide("slow", "swing"); $(".tracker_field4").hide("slow",
"swing"); } }); $(".tracker_field2").hide(); $(".tracker_field3").hide(); $(".tracker_field4").hide(); {JQ}
```

# 1.3.17. ADVANCED: JQUERY IN STATIC TEXT TRACKER FIELD TO SYNCHRONIZE USER SELECTOR TRACKER FIELDS

From the use case partly covered in the profile "Custom_Work_Pricing"

## 1.3.17.1. SIMPLEST CASE

**Code**

```
{JQ()} var user_fields = ['230', '302', '303']; jQuery.map(user_fields, function(f){ $('#user_selector_'+f).change(function(sel){ var users = []; jQuery.map(user_fields, function(f){ users = users.concat($('#user_selector_'+f).val()); }); $('#user_selector_229').val(users); }); }); $('#user_selector_229').closest('.form-group').hide(); $('#trackerinput_143').closest('.form-group').hide(); {JQ}
```

Explanation:

- Fields with Id 230 (Account Managers), 302 (Clients), 303 (Contractors) correspond to user selector tracker fields, one for a different user group.
- Field with Id 229 is the generic user selector field, listing all users from all groups, that will be used internally to control permissions on item owners, so that each user can see his/her own items only. (the permission to view items is not granted to the whole tracker, but only items belonging to each user will be seen by them.
- Field 143 is the field with this static text field containing this JQuery code.

## 1.3.17.2. UPDATING SOME STATUS IN ONE OTHER TRACKER

And in case you also want to update some field from another tracker, you can do so with an ajax call fired from inside the jquery plugin. See this enhanced version:

**Code**

```
{JQ()} var user_fields = ['230', '302', '303']; $.map(user_fields, function(f){ $('#user_selector_'+f).change(function(sel){ var user
= []; jQuery.map(user_fields, function(f){ users = users.concat($('#user_selector_'+f).val()); });
$('#user_selector_229').val(users); $('input[name="ins_241[]"]').map(function(i,el){ $.ajax({ type: 'POST', url: 'tiki-
ajax_services.php', dataType: 'json', data: { controller: 'tracker', action: 'update_item', trackerId: 13, itemId: $(el).val(), ins_314:
} }); }); }); }); $('#user_selector_229').closest('.form-group').hide(); $('#trackerinput_143').closest('.form-group').hide(); {JQ}
```

Explanation:

- field 241 is the item_link field from tracker 10 that shows records from tracker 13
- field 314 is the field from tracker 13 that controls the custom processing status of the item (users have been syncronized already - value 2 - or not - value 1). We want to modify the value in this field from whatever it was (usally 2, synchronized) and reset to value 1 (unprocessed).
  This way, some PluginListExecute can filter these "unprocessed" values and run some action on them, plus reset the status flag to 2 (processed), etc. So that permissions based on the user selector field can be set in synchrony among trackers.

## 1.3.17.3. UPDATING SOME STATUS VALUE IN TWO OTHER TRACKERS

And if you want to update several other trackers, not just one, you could use this other syntax. Note that it updates also tracker 11

**Code**

```
{JQ()} var user_fields = ['230', '302', '303']; $.map(user_fields, function(f){ $('#user_selector_'+f).change(function(sel){ var user
 = []; jQuery.map(user_fields, function(f){ users = users.concat($('#user_selector_'+f).val()); });
$('#user_selector_229').val(users); $('input[name="ins_241[]"]').map(function(i,el){ $.ajax({ type: 'POST', url: 'tiki-
ajax_services.php', dataType: 'json', data: { controller: 'tracker', action: 'update_item', trackerId: 13, itemId: $(el).val(), ins_314:
} }); }); $('input[name="ins_272[]"]').map(function(i,el){ $.ajax({ type: 'POST', url: 'tiki-ajax_services.php', dataType: 'json', data
{ controller: 'tracker', action: 'update_item', trackerId: 11, itemId: $(el).val(), ins_320: 1 } }); }); }); });
$('#user_selector_229').closest('.form-group').hide(); $('#trackerinput_143').closest('.form-group').hide(); {JQ}
```

This way, as in the previous case, some PluginListExecute can filter these "unprocessed" values and run some action on them, plus reset the status flag to 2 (processed), etc. So that permissions based on the user selector field can be set in synchrony among trackers.

## 1.3.18. MAKE A CLICKABLE BUTTON WITH A CUSTOM ICON

For when using PluginButton isnt possible for whatever reason.

**Code**

```
{DIV(class="my-button btn btn-primary")}{icon name="add"}[my-hyperlink|Click me!]{DIV} {JQ()} $('div.my-
button').click(function(){ window.location.href=$(this).find('a').attr('href'); }); {JQ}
```

In the example above, we want the div to act as the button but the hyperlink is only on the 'click me' text. By adding a click event on the div, you can fetch the hyperlink which now make the whole div clickable.

## 1.3.19. ALLOW ONLY LOWER CASE LETTERS IN A TRACKER FIELD

n this example we want to allow the user to type only lowercase, so that we replace the typed letters to lowercase at typing time, i the search form produced by PluginCustomSearch:

**Code**

```
{JQ()} $("#customsearch_0_24").on('keyup', function(e){ if( e.keyCode >= 65 && e.keyCode <= 90 ) {
$(this).val($(this).val().toLowerCase()); } }); {JQ}
```

Replace `customsearch_0_24` with the right value for your field selector.

This is useful with search/folter boxes (from PluginCustomSearch, PluginTrackerFilter or similar use cases with filter fields) in which he stored content or text strings are in lowercase, and the search is case sensitive, so that searching the uppercase versions of the string will show no matches.

## 1.3.20. EMPTY TRACKER DATE (DATEPICKER) FIELD WHEN USED AS FILTER

In some edge cases, you may want to empty a date (DatePicker) tracker field in form to filter records from a tracker. This can happen, for instance, in cases when you wnat to have PluginPivotTable show a filter field in top of the results. Example:

```
{PIVOTTABLE(data="tracker:2" )} {filter field="tracker_field_mydate" editable="content"} {PIVOTTABLE}
```

In that case, if you get the date field pre-filled with the current date and time (as set in the preferences of that tracker field), you

can empty that field with syntax like:

{JQ()} $("#list_filter1 .jscal > .form-control").val("") {JQ}

## 1.3.21. FORCE SEARCH MODULE FILTER FOR SPECIFIC FEATURES OBJECTS

The Search Module options allows a filter to be displayed to search object from a specific feature. For exemple you can search for a string but only for Articles, Forums, Wiki Pages, etc. However it is up to the user to set it.

Using JQuery you can force the filter for a specific feature and with some CSS styles you can hide the label and filter selector. The user won't see anything but a search field.

**Hiding fields**

{HTML()} <style> #page_yourpageID .module.box-search div.input-group label.col-form-label {display: none} #page_yourpageI .module.box-search div.input-group div.col-auto {display: none} </style> {HTML}

**Jquery forced selection to serch among Articles only**

{JQ()} $(document).ready(function(){ $("select[name='filter~type']").val("article") }); {JQ}

## 1.4. JQUERY ON CUSTOM SEARCH RESULTS

If you want to run some jquery code on the results of the CustomSearch, you need to do it in a different way than usual, since Custom Search loads results by Ajax and doesn't allow to run javascript code by default. See param "**callbackscript**" at the CustomSearch documentation page to see how to avoid this issue and get your jQuery run.

## 1.5. TRACKER FIELD VALIDATION

See also: Tracker Field Validation > **Adding custom per-page validation**

## MORE JQUERY TIPS & TRICKS

For more tips and tricks using jQuery, visit:
http://api.jquery.com/

For more information on jQuery selectors:
http://api.jquery.com/category/selectors/