Table of contents

Introductory Remarks

This tutorial uses the commands parameters and values explained in the PluginLIST documentation on this website and will guide you through the process of creating a simple table to display tracker data in very small steps.

The structure for the steps is as following:

- first use a descriptive heading and optional description
- second show the code
- third explain "what we did" with the code
- fourth explain the "result" of what we did with the code.
- optionally, where it makes sense, the fifth part is a demo display

Why our devs recommend us to use {LIST()} and to stop using {trackerlist}:

With the plugin {LIST()} you can do the same as you can do with plugin {trackerlist}.
On the one hand, {trackerlist} is quite simple to use where {LIST()} needs some more in depth knowledge.
On the other hand, {LIST()} has a number of advantages over {trackerlist} and once the principles are understood it is not so difficult with {LIST()} to get tracker data displayed on a wiki page.

Advantages of {LIST()} over {trackerlist}:

- Because {LIST()} builds upon "Unified Search", it loads the data several times faster than {trackerlist}
- {LIST()} has more options, with the result that the capabilities of {trackerlist} are only a small subset of the capabilities of {LIST()}.
- The quality and quantity of opportunities with {LIST()} convinced our developers so much, that they decided to not continue supporting {trackerlist} anymore from Tiki 15 onwards, although being still usable at least for a while, due to backwards compatibility where older Tikis will be upgraded.

Disadvantage of {LIST()} over {trackerlist}:

- {LIST()} has a somewhat steep learning curve. You need a bit longer to get the first result (however you can do much more with it).

What you need to know first

{LIST()} has basically two main facets.
First you "filter" data, meaning you let the plugin {LIST()} search the data you want to display.
The search is accomplished by the so called "Unified Search" feature, which has to be active in your Tiki.
Second you display the filtered data with a huge number of options.

note

On an average shared hosting you can work with "Unified Search" and the option "MySQL Full-text Search as search engine". "Lucene" is possible as well, but not recommended and likely at some day not supported anymore, whilst "Elasticsearch" requires an "Elasticsearch" installation on a dedicated root server.
Elasticsearch does not need to be installed on the same server and it is possible to use an external Elasticsearch service, either a self-hosted service or a paid service.

In this tutorial we use the standard shared hosting setup with Unified Search + MySQL Full-text Search.

× ✓
This said, we conclude, that plugin {LIST()} uses a "filter command" to filter arbitrary data from the database and some kind of "output command" to display the filtered data. "Format commands" optionally extend the output commands.

Task: How to filter and display data from a certain tracker?

This website has a tracker which contains the features of Tiki (we call it *Feature Tracker*) and certain descriptive data about those features. Thus we want to display some data of this tracker for demonstration.

0. basic syntax

Please mind, that the sort order of the parameters inside the commands is indifferent (meaning the order is not important). Same as with wiki plugin parameters it has no effect which parameter you write first. But it makes sense to stick to some general order (like a best practice) to make larger LIST plugins better readable by default, which might only perhaps make sense for you alone, but truly makes sense in a team based work environment.

```
{LIST()}
{inlinecommand1 parameter1="value1" parameter2="value2"}
{TAGCOMMAND2(parameter3="value3")}
{inlinecommand3 parameter4="value4" parameter5="value5"}
{inlinecommand4 parameter6="value6" parameter7="value7"}
{TAGCOMMAND2}
{TAGCOMMAND5(parameter8="value8")}
{inlinecommand6 parameter9="value9" parameter10="value10"}
{inlinecommand7 parameter11="value11" parameter12="value12"}
{TAGCOMMAND5}
{LIST}
```

Hints:

The commands inside the List plugin in the "tag format" (written in capitals) are usual OUTPUT and FORMAT, whilst the inline commands (written in lower case) are usual filter, column, display etc.

Two things can be very confusing for beginners:

a) Sometimes output and format can (or have to) be used as inline commands as well

b) The names of commands, parameters or values can be the same, but have complete different effects / meanings. This is one reason, why the to date existing comprehensive LIST documentation tends to be only little useful for *list-beginners* or for non-tecchies.

1. LIST alone

```
{LIST()}
Body of the LIST plugin containing the various commands
{LIST}
```

What we did:
We just created the List plugin with only some descriptive text inside, which has no effect on the result.
lind: The body shall contain commands (in curly brackets). Wiki text is only for comments - in basic use cases. You are better off to avoid any non-commands, if you do no
know what you do.

By only creating the empty List plugin, we already triggered a search query (actually an unspecified search query, searching for simply everything without any output directive).
'e only *told* the List plugin: "you exist, so look what is in Tiki (in the database), once this wiki page (where we use the empty plugin) is opened in a browser." But we did no
tell the List plugin what to do with the data.

Result:
The result of this code would be a list of about 50 items out of the database, actually wiki pages, as the List plugin without specific commands uses some default configurations related to search and pagination. The result can be different on other Tikis.
The result of this code has no real value, as we have no control about the results at this point (not yet without additional commands).

We do not display the result here, to not waste too much space with a long list of unusable results.

2. filtering the data from one tracker

info

In the inline command "*filter*" you usually want to specify the type of object you are filtering (where filtering is limiting the search result to a subset of all database content). As explained above, an empty List plugin would just search for all content that is indexed in Unified Search. As LIST is build upon *Unified Search* you use *Unified Search Fields* to specify the filtered objects.
This is done with the parameter *field* in the inline command *filter*.

You find a list of the available fields and a table which features make use of which fields here: **Search and List from Unified Index**

```
{LIST()}
{filter field="tracker_id" content="7"}
{LIST}
```

What we did:
Our "Feature Tracker" has the Id=7. The filter command (like most other commands as well) needs a parameter *field* to define *what sort of data* we want to filter.

Here we filter for a tracker Id, so we have the parameter *field* with the value *tracker_Id*.
Please mind the underscore instead of empty space: *field*="*tracker_id*"!

Next, the filter command needs some specification *which* tracker Id we are looking for. There fore we use the parameter '*content* with the value 7 .

{filter field="tracker_id" content="7"} filters for all data that matches with the search for tracker_Id=7, where it does not look for the tracker itself, but for the content of this tracker (we will understand this very point later, just take it as given for now).

Result:
The List plugin would display the first X items of the tracker with Id=7 (on this site 50 items of the "*Feature Tracker*", given no other admin changes configurations in the meantime).

is result as well has no value for us, but at least we already have broken the issue down to the tracker data we are looking for instead of filtering simply all existing data the database.

We do not display the result here, to not waste too much space with a long list of unusable results.

3.1 using the table template instead of the default simple list

```
{LIST()}
{filter field="tracker_id" content="7"}
{OUTPUT(template="table")}
{OUTPUT}
{LIST}
```

What we did:
Using the output command with the parameter *template*="*table*", we *told* the plugin list to use a predefined template for displaying the output.
Plugin List in Tiki 15 knows the following templates (status March 2016):

- table (optionally to be combined with tablesorter)
- medialist (to display a fancy list)
- carousel (to display a Bootstrap Slideshow from a file gallery)
- count (to display the number of items of a search query)
- there are more advanced opportunities, but here we stick to the very basics
for advanced output see here: LIST - advanced output command

Result:
ow the plugin List does not anymore show the tracker items - neither as a list, nor as a table - BUT it displays a *pagination* with pagination links "below" an empty (invisib
table.

t the time of writing this page, the pagination offers 3 pages. As today this Tiki is configured to a standard pagination of 50 items and the tracker Id=7 contains the curre
features of Tiki 15, we can assume that we have about more than 100 and less than 151 features in Tiki.

«
1 (current)
2
3
»

But why the tracker items are not displayed? The above information seems not to be very useful without any of the features displayed.

Easy to answer:
At first we know, that the List plugin already filtered the items of tracker Id=7, so it "*spot on knows*" about all of the items.
Secondly the Output command with the parameter *template="table"* "*knows*" that a pagination might be useful for us: The template table just contains a pagination by default and displays when settings apply in the background, for ex. our global settings.
On the other hand the template table *needs* additional information about which columns to display to being able display at least something.
Thus: no columns defined equals no content displayed.

3.2 limiting the number of (to be) displayed results with the pagination command

```
{LIST()}
{pagination max=5}
{filter field="tracker_id" content="7"}
{OUTPUT(template="table")}
{OUTPUT}
{LIST}
```

What we did:
We obviously use a (the) pagination command to limit the number of displayed results, for the case we would manage to get them displayed.

Result:
No change in display, but we are prepared for the next step: actually showing some items from the tracker.

4. defining columns and display tracker items

Please keep in mind:
At this point we need to know the permanent names and the Id numbers of the fields of tracker Id=7

```
{LIST()}
{pagination max=5}
{filter field="tracker_id" content="7"}
{OUTPUT(template="table")}
{column field="tracker_field_f_62" label="Name" mode="raw"}
{column field="tracker_field_f_93" label="Type" mode="raw"}
{OUTPUT}
{LIST}
```

What we did:
/e defined two columns, which will be used (and is required) by the template table ... surely a table needs columns, like a car needs tires. A table without columns display
n content - see above No. 3..
he command "*column*" needs a parameter "*field*" to specify which of the data, which was filtered before will be displayed. Obviously the first column will display the track
field with the **permanent name** tracker_field_f_62 ... again mind the underscore instead of empty space: *field*="*tracker_field_f_62*".
The parameter "*label*" defines the label of the heading of the column.
he parameter "*mode*" is important for more advanced use. At this point we should be satisfied, that we want to use "*raw* text" as output mode and we should just settle f
the fact, that a link to the tracker item is defined in the tracker field definition at /tiki-admin_tracker_fields.php?trackerId=XYZ and in the FORMAT command (which we
tackle later on).

Result:
Now we have everything in place for a first basic and useful result.
We paginated to max 5 items to not wasting too much space on this page and below we see a table with the two defined columns, based on the search query on tracker
Id=7 and the output command on the fieldId 62 **by referencing its permanent name** tracker_field_f_62 and fieldId 93 **by referencing its permanent name**
tracker_field_f_93:

Name	Type
Chat	Section
Open standards & protocols	Other
Platform independence	Other
File Gallery	Section
Multilingual	Transversal

«
1 (current)
2
3

5. the parameter "*content*"

As seen above, we used the parameter *content* in the command *filter* to specify, which tracker (respectively tracker_Id) we are looking for.
The parameter *content* can be used as well for example in the command *column*, where it limits the displayed subset of the filtered search results?

Reminder:

- first we filter a subset of the data in the Tiki database with a unified search query from the List plugin (we say filter and List does the search for the resulting subset o the data we want to use.
- second we define which part of this subset of data will be displayed.
- third we define how the displayed date will be formatted=rendered (how it will be actually looking, if different from a default presetting).

The parameter *content* helps us with the second task: defining the part of the subset of the Tiki database data (filter / search results) we actually want to display.

Example:
hen a tracker field is a category field and we defined a category tree for this field, a column with the parameter "field" and the value ''tracker_field_my_category_field" w display the categories of the displayed items and an empty field when an item is not categorised. At this point the table will display all items of the tracker, whether categorised or not.
But when we add the parameter *content* and therein add a number of category Ids of the above mentioned category tree, then the table will only show items which are categorised with those categories, named in the value of the parameter *content*. Other categorised or non-categorised items will not be displayed anymore.
The parameter *content* works with other field types than category as well.

```
a code example to be added
```

What we did:

Result:

6. getting tablesorter into the game

⚠ Do not forget ...

... to activate tablesorter when using for the first time on a site, cause otherwise it might happen, that it won't work. Please make sure, that you activate the **preference *jQuery Sortable Tables*** (We might add a comment to the preference on day to mention that "jQuery Sortable Tables" about equals "tablesorter". Sorry for the confusion.).

You find the preference in /tiki-admin.php?page=features#contentadmin_features-interface.

tablesorter will only work in plugin List from Tiki 15 onwards

ℹ tip

For now you can find the tablesorter parameters here at the trackerlist wiki page - the options and parameters are identical with both, the LIST and the trackerlist plugin:
https://doc.tiki.org/PluginTrackerList#content_index1-5

```
{LIST()}
{pagination max=5}
{filter field="tracker_id" content="7"}
{OUTPUT(template="table")}
{column field="tracker_field_f_62" label="Name" mode="raw"}
{column field="tracker_field_f_93" label="Type" mode="raw"}
{tablesorter server="n" sortable="type:reset" tsortcolumns="type:text|type:text" tsfilters="type:text|type:text" tspaginate="max:10"
tscolselect="critical|5|6"}
{OUTPUT}
{LIST}
```

What we did:

Result:

Name	Type
Chat	Section
Open standards & protocols	Other

Name	Type
Platform independence	Other
File Gallery	Section
Multilingual	Transversal

Alias

- Tutorial: Display Tracker data with plugin List