MANUAL

ob1 advanced search

DOC# 1000-2

PLUGIN VER# 1.03B

GETTING STARTED

DOWNLOAD THE PLUGIN

First thing you do is to go get the plugin if you haven't already.

Download from RDS
Download from Textpattern Resources

INSTALL THE PLUGIN

Go to your web site and log in to the Textpattern interface.
I recommend that you have your web site in developing mode when installing this plugin since otherwise you won't see any errors, if any. I also recommend you duplicating your default page and creating a search section associated to it. If you do not know how to do these tasks maybe this plugin isn't for you or you need to get help from someone who knows a little bit more about Textpattern. The forum is a great place to find some help.

CHECKLIST

- Create a search section, either duplicate the default page or create a new associated to the section
- Install the plugin and activate it.

Already done? Lovely.

SETTING IT UP

On the newly created page for the search section, remove the article tag currently in it and replace it with:

```
<txp:ob1_advanced_search/>
```

Note: Whatever you do, **don't load the page** just yet!

QUICKSTART

If you want the fast approach of to the plugin just add the attribute setup="1" to the already added tag, load the page and it will install the standard indexes, section, pages, forms and CSS.
The standard install makes title, body, excerpt and the custom field one and two search able in occurrences.
After this is done and nothing big has exploded on the screen telling you something has gone bonkers, revert the tag to it's first state without the setup="1" attribute and reload and voila, a search form has appeared.

If something went wrong or you had some error it's my hope that it explains what went wrong. Otherwise see Reporting errors/feats.

Now for us other users that wants to select what this plugin does before it does it, read on.

AVAILABLE PREFERENCES

Now it's time to check the plugins default preferences and change it to reflect how you want your users to be able to interact with the plugin.

`allowed_usage`

Through this you set what kind of searches the user can do. If a feature is disabled here the user can not add the feature to the GET string since it is also blocked in the actual searching.
Note that there is one more allowed usage but for some of the word scenarios. This was the easiest way to solve this so bare with me.
I also do not recommend removing the words type.

By default the install allows all types of searches to be made.

Allowed types:
- results_per_page
- words
- without_words
- occurrences
- first_seen
- date_between
- date_future

results_per_page
Defines if the user can choose how many results are displayed per page.

words
Defines if the user can use word searches (all words, exact phrase and at least one word). Usage is defined under allowed_words_usage.

without_words
Add this if you want the user to be able to search without specified words.

occurrences
If you want your users to be able to choose the fields they search in. What fields are defined under the occurrences.

first_seen
Makes the user able to search in admin defined time intervals (ex last month).

date_between
Allows the user to do a date-range specified search.

date_future
Makes it possible for the user to search for articles with posted dates in the up coming future. If date_between isn't specified this type has no use.

Note: If you specify later on that the first_seen <select> should search in the up coming weeks/months and so on this type will not block that. It only takes care of what the user him/herself types in. You're an admin, why should your decisions be questioned?

`allowed_words_usage`

Declare text searches available and in what order they appear in the <select>.

all_words
Matches all the words separatly against the occurrences.

exact_phrase
Allows the user to do the vanilla Textpattern search way. Searches with exact phrases typed in by the user.

at_least_one_word
Lets the user do a more generic search where they do not know exactly what they are looking for. See it as a good feature if you have users who can't decide what they are looking for.

allowed_words_usage_name

We have to have names to the above usage features right? What good would a select that stated all_words etc. do? And what would this plugin do if I didn't make it relatively easy to change language in it? You're right, not much.

Now comes the tricky part. You need to specify this in the same order as allowed_words_usage or the text in the <select> won't add up to what the user chooses.
Meaning:
If you have defined the allowed_words to be at_least_one_word,all_words this comma-separated list should state in English:

```
with at least one of the
words,with all the words
```

(without the line-break of course)

Note: As you might have guessed already the first value you add to the allowed_words_usage will be the default search form.

first_seen

Here is where you add in what time periods the first_seen <select> should use. As with allowed_words_usage, the first value becomes the default.

The time period is defined in how many minutes have passed so if you are going to do any changes, grab the calculator and start counting away.

If you want future dates (for instance up coming week) you add a negative value.

The wild card * (star) specifies any time.

Note: I do recommend the wild card * being the default, otherwise the default search will use the time period specified by you.

first_seen_name

This is where you actually tell the user what all those numbers you just calculated mean in human terms. Like allowed_words_usage_name this list needs to be in the same order as you specify first_seen.

occurrences

You probably by now think you got the hang of it, but then I pull you back in.
Most part is same-same with first_seen but in steps custom fields to the arena.
Here you define where the user will be able to do searches and what indexes will be created in the database.
Like first_seen it has the wild card * to define all fields and I do recommend you keeping this as the default (or first) type.
Other types allowed is any field name of the textpattern table which is a CHAR, VARCHAR or TEXT field.

On a standard Textpattern install this means:

- AuthorID
- LastModID
- Title
- Title_html
- Body
- Body_html
- Excerpt
- Excerpt_html
- Image
- Category1
- Category2
- AnnotateInvite
- Section
- override_form
- url_title
- custom1-10

Since you can define other names to the last ones, custom1-custom10, if you want them available, just add %custom to occurrences and the plugin will take care of using the names you have chosen for them in the admin preferences.

Occurrences example:

```
*,Title,Body,Keywords,%custom
```

You specify which, or all, of the custom fields to search in the preference of search_custom later described.

Note: %custom has to currently be the last value.

occurrences_name

Like previous _name preferences this creates the actual text supplied to the user with one difference; you can here add %customname into a value and it will change it to whatever the custom field is named by you.

Example if one of your custom fields names are Shoe size:

```
in the %customname of the
article
```

results in

```
in the shoe size of the
article
```

Sure. What article has it's own shoe size? It's early morning and I got writers-block, nuff said. The value will be returned lower-cased.

Note: Like %custom, %customname has to be the last value.

results

This is where it get's even more tricky, this one specifies (hold on to your pantihose) the text added to the results <select>. Phew. Default value: ' results'.

results_per_page

Define how many results per page the user can choose from. And yes, it's comma-separated.

results_cookie

Do you want the user to save the results chosen in a cookie? 1 or 0 does the trick here.

forms

This defines the name of the forms used by the plugin. It needs to have five values and in a specific order, namely:

- Big search form
- Search header form
- Small search form
- Search results form
- Error form

BIG SEARCH FORM

The **big search form** is what is presented when the user enters the search section as well as wherever you specify in show_search_form. It can contain some dynamic text that is wrapped in % signs. These are then altered when the form is out putted by the plugin.

%search%
This is added in the form action and depending on what URL mode the site has changes it to the search section or ./.

%search_hidden%
Here we add a hidden <input> if the URL mode is messy so Textpattern shows the correct section when searching. If not messy-mode, this is removed.

%select_display_results%
Changes to a <select> with the info from the results_per_page and results preferences.

%words%
Changes to an text <input>. Why use this instead of just adding a <input> tag directly? Well, this way, it adds the information the user altered if the form is displayed again (for instance on no results found). Linked with the %select_words_how%.

%select_words_how%
Creates a <select> on the different types of word searching allowed.

%without_words%
Like %words%, adds an text <input>.

%select_occurrences%
You guessed right, creates an <select> tag with the occurrences.

%select_first_seen%
More dynamically created <select> tags, this time first_seen.

%startdate%
Adds a text <input> for the date searches start date. This also gets a class added to it called dInput so we can alter the sizes (if wanted).

%enddate%
Almost the same as %startdate% just that it creates the <input> for the end date.

Note: All, except %search% and %search_hidden%, are of course not added if not allowed in the allowed_usage.

SEARCH HEADER FORM

Search header is the form that contains what will be displayed when a user has done a search.
Like the big search form it contains some things that will be altered in the output and these are the following:

%start%
Changed to the first number of the results shown. For instance, on the first page and results have been found, changed to 1. But on page two on a 10 results/page search it shows 11.

%end%
Outputs the end result of the page. For instance, if three results were found, outputs 3.

%total%
Output the total search results.

%article%
Outputs article/articles depending if one or more results. No worries, it uses the Textpattern installs chosen language so no need to change anything here.

%results% and %no_results%
Now what about these two? They are wrapped around the contents to show. Wrap %results% %/results% around what should displayed if results were found and %no_results% %/no_results% around the contents to display if the search came up blank.

SMALL SEARCH FORM

Contains the %search%, %search_hidden% and %words% as explained in the big search form.

SEARCH RESULTS FORM

Not highly interested, it's by default the standard search form already in Textpattern. Want another? Specify the name here instead.

ERROR FORM

This is what will be out putted if there were user errors in the search. This form will be added below the header form (that will be removed of the contents inside %results% and %no_results%).
It contains just one dynamic thing, namely:

%error_list%
The output will loop through all errors and output them here. It has the attribute wraptag if you want the output wrapped into something (
, etc.). Like anywhere else in Textpattern you just define the name of the tag, not the tag itself, like wraptag="li".

search_section

This is where you define the section name and title that will be used by the plugin.

search_custom

Feel like searching some custom fields? Define the numbers of the fields here.

Note: The fields have to be CHAR, VARCHAR or TEXT, otherwise there will be errors.

show_search_form

Here you set in what scenarios the big search form will be displayed. Allowed values are:

- empty
- no_results
- before_results
- after_results

The list is comma-separated and no need to think of what order they are in.
The values are quite self explanatory so I see no need to go into them here.

error_search_short

Defines the error message shown when the search string is less then two characters long or no dates have been specified to search for.

error_date_larger

Defines the error message shown when the end date specified by the user is larger then the start date.

error_date_future

Defines the error message shown when the user have specified dates in the future and it is not allowed in the allowed_usage.

setup

What you want the setup to check when called. Allowed values are:

- index
- section
- forms
- css

index
If the setup should check and add/alter the database indexes or not.

section
If you want the setup to check that the search section and page (sorry, not separated just yet) exists.

forms
Do you want the setup to check and add the default forms? Then this is a must value.

css
If allowed, checks the default css if it contains the id used in the big search form.

This list is comma-separated and you do not need to think of the order.

SETUP EXAMPLE

Now, let's play with the following scenario: You want the plugin in Swedish (I'm Swedish, so wonder why I chose that scenario?) and you only want the user to be able to do word searches, they can change how many results per page is displayed, they can search the title, body and custom field 1 and the big search form should be displayed on empty or a no results call.
So how do we do this?

In the test page you have created (hopefully) where you already have the original call we do the following.

Change the plugin call to set the values needed before the setup is called.

```
<txp:ob1_advanced_search
set="allowed_usage;
occurrences;
show_search_form;
search_custom;
allowed_words_usage_name;
occurrences_name;
results;
error_search_short"
value="results_per_
page,words,without_words;
*,Title,Body,%custom;
empty,no_results;
1;
med alla ord,med den exakta
frasen,med åtminstone ett ord;
var som helst i artikeln,i
titeln av artikeln,i texten
av artikeln,i %customname av
artikeln;
 resultat;
Sökströngen för kort (minst
två tecken behövs)"/>
```

I've added some line breaks for readability but you should of course use a one line call.

Note: You can of course separate every set to it's own plugin call if you want even better readability. It will of course decrease performance a tiny, tiny bit but...

I recommend having your Textpattern install in development status when doing the setup or error output won't happen. Not that I hope you will get errors, but of course there is a possibility.

Now load up the page in your browser.

Not all error messages are displayed on the page itself but in the code. Show the source of the page and check at the end of the code for trace information what has happened and what failed.
The more "heavier" errors are of course displayed in the browser directly.
Fix any errors that might have occurred and reload the page until it comes up blank where the plugin call is made.

Pre-check complete? Time to make the call of the setup process. Add another plugin call below your set/value call with the attribute setup="1".
What? Do you have to do the set/value call again? Isn't the preferences stored in the database? To this I of course have to answer no. I thought of having it saved in the preferences but got to the conclusion not to for the time being. Future version may but that's not for us to discuss here.

After reloading the page in the browser redo the check for errors in the source.

No errors? Lovely. Now the plugin has created what it needs to function. Remove the attribute setup="1" from the plugin call. When you now reload the page the default big search form should display.

Since we told it not to use some of the features it looks a bit strange with it's empty parts and therefore needs to be altered. Go into the admin side and alter the forms as you wish. I won't go into this since I do hope you know what to do.

The plugin should now be up and running. Sweet.

REPORTING ERRORS/FEATS

- Found an error you can't make go away?
- Want a feature added to the plugin?
- Want to add your praise to my state of genius?

Go to the Textpattern forum and find the ob1_advanced_search thread.

PLANNED FEATURES

I am planning to add id and class attributes to the different %selects% and inputs created through the output, making some of the selects to hidden, add start- and end word

to the %select%s and add some default text attribute for the inputs.
I wish I had the time and knowledge to start looking at caching searches but who knows?

CHANGE LOG

v1.0b [2007-09-17]

- First release

v1.01b [2007-09-17]

Bug fix

- Fixed when no %select_words_how% chosen Textpattern returned an error in debug mode.

v1.02b [2007-09-18]

Bug fix

- Fixed when it saves the results/page cookie and no results/page have been defined in the query

New features

- Added search_score to the $thisarticle variable. Therefor you can now type out the score each search match had with some <txp:php> in the search_results form. Yes, a plugin that does this for you is on the way.

v1.03b [2007-10-05]

Bug fix

- Didn't add table prefix to the textpattern table when creating indexes
- Didn't filter out the sections not to search in a standard search

New features

- uninstall attribute for removing the indexes created by the plugin (forms etc you have to remove by the TXP interface)
- sections attribute for limiting the search to specified sections