CrownPeak

WEB EXPERIENCE MANAGEMENT

Search Implementation and Best Practices

Preparation

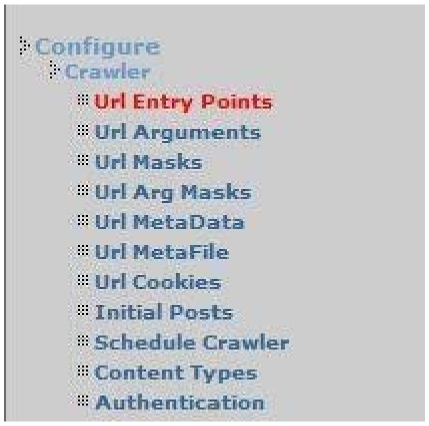
Before you begin you should start with a mockup for the search results page. Having the HTML and css already implemented for the search results will make the search implementation much easier.

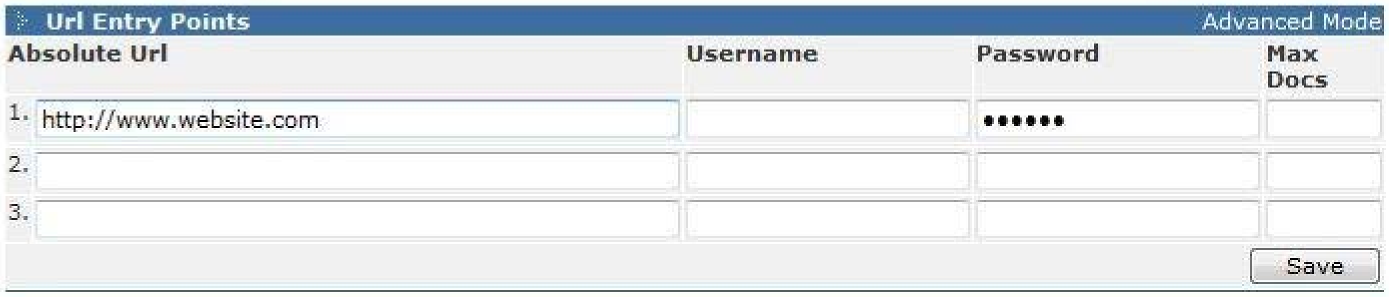
Ensure that you have the correct URL to spider and that the URL is publicly accessible. The site should be linked to some extent in order to start any tests.

Submit a ticket to the CrownPeak systems team to get an instance created. Although you have the ability to create search instances directly from the search administration screens, we need to have the systems team create the instances in order to properly load balance the servers. Our systems team will need to know the name of the instance and the URL for the site.

Basic Search Implementation

Once the instance has been created, start by double checking to see that the URL is entered correctly under Configure > Crawler > URL Entry Points. The URL entry point is the URL where search will start crawling your site. Note that most staging sites will require a username and password in order to spider the site. This needs to be added to the Configure->Crawler->URL Entry Points. If the client uses a non-http authentication technique (like Windows NTML Authentication) we CANNOT spider their site. Regarding the staging site, you will need to ask them to support the basic HTTP auth technique. Note that if they are using cookies for authentication that can be setup in the Configure->Crawler->URL Cookies interface. If the cookies change once per login you will need to setup the Configure->Crawler->Initial Posts with the login URL and arguments in order to "log" the crawler into the system prior to crawling.





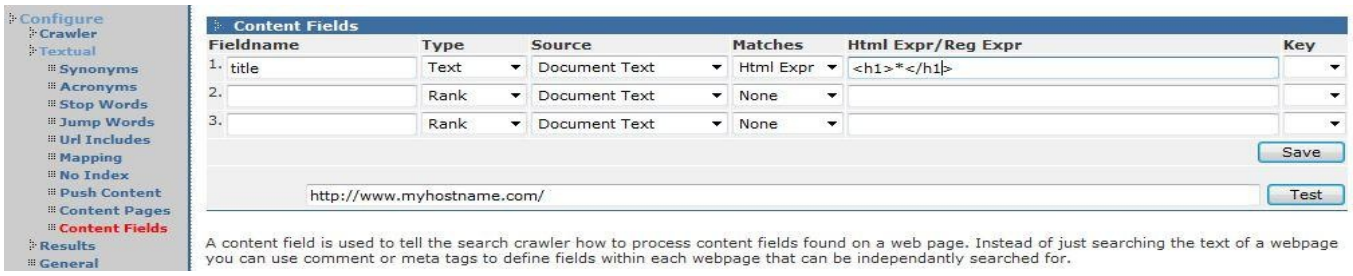
	Absolute Url	Username	Password	Max Docs
1.	http://www.website.com		••••••	
2.				
3.				

Did we spider enough urls? Check with the client for a page count to see if we are in the ball park. If we are short check for Javascript links, flash links, etc. and add additional links into the entry points to help the spider index the entire site. Note that if they have a site map this is a good url to add as an entry point. Additionally, we could create a custom page that includes links to pages not being linked from elsewhere on the site and add this custom page to the url entry points.

- Did we spider too many urls? Look for url arguments that cause looping of the same page but with a new argument that does not change the content. To reduce these duplicate pages use the Configure->Crawler->Url Arg Masks to eliminate certain arguments from being considered when unique urls are determined.
- Ensure that the right pages are being spidered. Note that by default the crawler will respect the robots.txt and meta tags in order to reduce indexing. If pages are missing check the robots.txt file of their site (located at www.domain.com/robots.txt) and/or the meta tags present in each page. Try switching them off in the Configure->General (checkboxes) and reindex the site.
- Ensure that pages that should not be indexed are not. Often printer versions of pages will be included in the index. These can either be excluded using the Configure->Crawler->Url Masks using something like "exclude –print.asp$" which will exclude all pages that end with –print.asp from being indexed.

Once spidering is complete do a quick search test using the search systems search box (located at the top of the browser). This should use the base template and return some results. If successful, you are now ready to improve the data extracted from the site and tweak the template's HTML for the search results.

First and foremost, there will be title extraction. Most likely, the default of extracting the title from the HTML <title>*</title> tags is not sufficient. The client should be reminded that this tag will be used for all other internet search engines and thus should be as clear and clean as possible. If it is not possible to alter the <title></title> tag you may have to extract the title from other parts of the page. This can be done using HTML expression syntax. For example, to extract a title from H1 tags use <h1>*</h1> for the select html expr in the Configure->Textual->Content Fields. If you want to default to another tag and then H1 use the | as an OR. For example, <h2>*</h2>|<h1>*</h1> would check for h2 first and if it is not found it will check for h1 tags. This allows the title to be drawn from the page in various places.



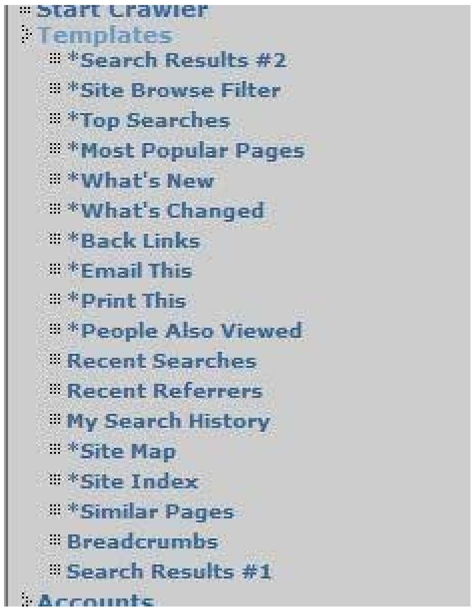
	Fieldname	Type	Source	Matches	Html Expr/Reg Expr	Key
1.	title	Text	Document Text	Html Expr	<h1>*</h1>	
2.		Rank	Document Text	None		
3.		Rank	Document Text	None		

http://www.myhostname.com/

A content field is used to tell the search crawler how to process content fields found on a web page. Instead of just searching the text of a webpage you can use comment or meta tags to define fields within each webpage that can be independantly searched for.

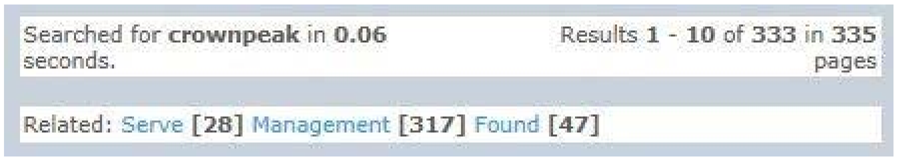
Often the client includes the site name in the title tags. This can be removed using the [] optional tags. For example, <title>[CrownPeak \|]*</title> would strip the text CrownPeak from the title tag IF the title tag starts with CrownPeak. Also note the use of the \ character to escape characters that may mean other functional items as in the case of a | being an OR.

Once the title is cleaned up, you can move on to the template. The default template used is normally called “Search Results #1.” You can find the template under Templates->Search Results #1.



The template should have some default code in it which you could modify as needed. Essentially, you would need to modify the HTML to render the HTML based on your mockup.

From the mockup, you need to determine the content that will be returned by search. This includes data about the search like: the term being searched for, current results page number, total number of items found, total number of pages, search term suggestions, related items, etc...



Searched for **crownpeak** in **0.06** seconds.

Results **1 - 10** of **333** in **335** pages

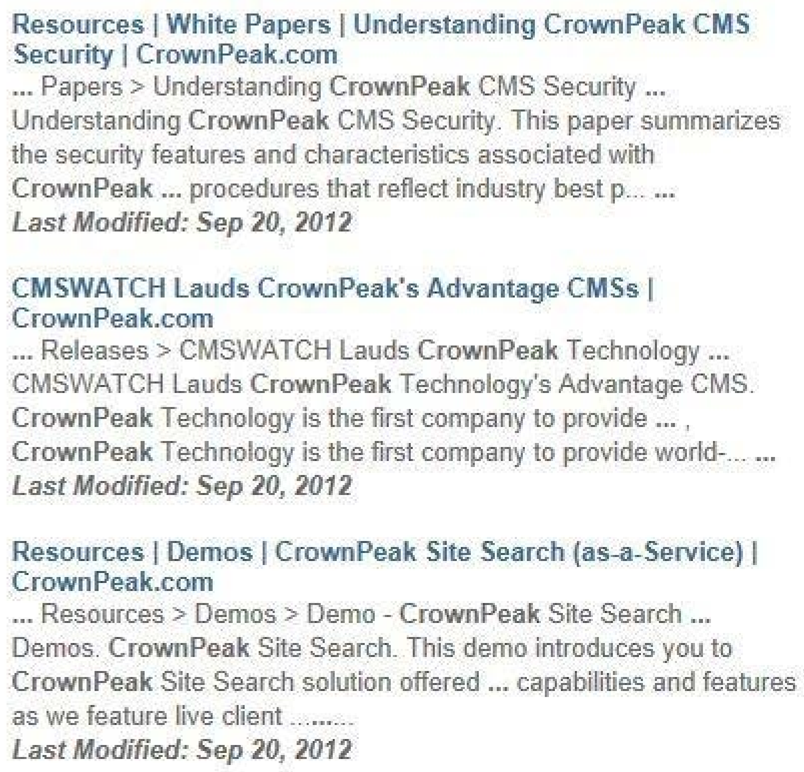
Related: Serve **[28]** Management **[317]** Found **[47]**

The default template code for this is listed below:

```
<table width=100% cellpadding=0 cellborder=0>

<tr>

<td>Searched the website for <b><#cgi.q#></b>.</td>

<td align=right>Results <b><#start_found#></b> - <b><#end_found#></b> of
<b><#total_found#></b> in <b><#total_documents#></b> pages.</td>

</tr>

</table>
```

```
<br><#if isNotNull="spelling"#>

Did you mean:&nbsp;

<#foreach variable="spelling" use="spell"#>

<a href="<# system.link_url #>?q=<#spell.keywords#>&<# querystring exclude="q,page"
#>"><#print text=spell.term_context#></a>

&nbsp;<b>[<#spell.count#>]</b>&nbsp;

<#/foreach#>

<br>

<#/if#><#if isNotNull="related"#>

Related:&nbsp;

<#foreach variable="related" use="relate"#>

<a href="<# system.link_url #>?q=<#relate.term#>&<# querystring exclude="q,page"
#>"><#relate.term#></a>

&nbsp;<b>[<#relate.count#>]</b>&nbsp;

<#/foreach#>

<br>

<#/if#>

<#if isNotNull="acronyms"#>

<#foreach variable="acronyms" use="acro"#>

<b><#acro.term#></b> may also mean <i><#acro.definition#></i><br>

<#/foreach#>

<br>

<#/if#>
```

You will also want to determine the block of html that contains the repeating list of items.



Resources | White Papers | Understanding CrownPeak CMS Security | CrownPeak.com
... Papers > Understanding CrownPeak CMS Security ... Understanding CrownPeak CMS Security. This paper summarizes the security features and characteristics associated with CrownPeak ... procedures that reflect industry best p... ...
Last Modified: Sep 20, 2012

CMSWATCH Lauds CrownPeak's Advantage CMSs | CrownPeak.com
... Releases > CMSWATCH Lauds CrownPeak Technology ... CMSWATCH Lauds CrownPeak Technology's Advantage CMS. CrownPeak Technology is the first company to provide ... , CrownPeak Technology is the first company to provide world-... ...
Last Modified: Sep 20, 2012

Resources | Demos | CrownPeak Site Search (as-a-Service) | CrownPeak.com
... Resources > Demos > Demo - CrownPeak Site Search ... Demos. CrownPeak Site Search. This demo introduces you to CrownPeak Site Search solution offered ... capabilities and features as we feature live client ....
Last Modified: Sep 20, 2012

The default code for this list is shown below. You would need to modify this loop to render the HTML you need for each result item.

```
<#foreach variable="results" use="result"#>

<p><a href="<# result.link_url #>"><#print variable="result.icon"#>&nbsp;<#if
isnotnull="result.title_context" #><#print variable="result.title_context"
crop=80#><#else#>Unknown<#/if#></a>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;<a href="<#
result.link_url #>" target="_blank"><font size=-2>[new window]</font></a>

<br>

<font size=-1><#print variable="result.body_context"#></font>

<br>

<font size=-2><font color=#008000><#result.display_path#> - <#result.size#> -
<#result.modified_date#></font></font>

<br>

<#/foreach#>
```

Adding Custom Content

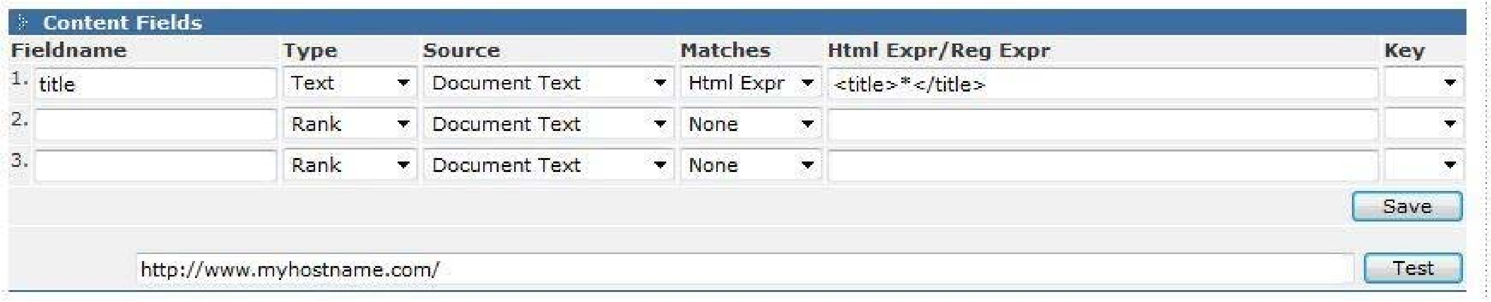
By default, the search will automatically use the contents of the <title> tag for search result titles unless an alternate is specified in step #4 above. The abstract will be a section of text pulled from any text on the page. The text used for the abstract will be the text surrounding and including the search term. From the code in step #4 above this is rendered using the `<#print variable="result.title_context" crop=80#>` and `<#print variable="result.body_context"#>` tags. If it is determined that custom content needs to be displayed in the search results, for example publication dates, alternate abstracts or sub titles, you will need to determine if the content can be found within a standard HTML tag on the page being indexed. If not, you will need to add meta tags for the custom content to all pages that apply.

The names of the meta tags can be arbitrary but as a best practice you should only use letters and the underscore character for the meta name.

```
<meta name="page_title" value="News Article Title" />

<meta name="publication_date" value="08/26/2012" />
```

After adding the custom meta tags, make sure to re-publish all pages and ensure the meta content is in place on the site by viewing the page's source within your browser.

Regardless of whether the custom content can be found in standard html tags or if you added custom meta tags, you will need to configure search to pull in the content and store them in content fields for use in your search templates. You would need to configure these under Configure > Textual > Content Fields.



Content Fields

	Fieldname	Type	Source	Matches	Html Expr/Reg Expr	Key
1.	title	Text	Document Text	Html Expr	<title>*</title>	
2.		Rank	Document Text	None		
3.		Rank	Document Text	None		

The Fieldname is arbitrary and should consist of letters or the underscore. Refrain from using the fieldnames such as title or body as these have special meanings to search.

In general, the Type of 'Text' is sufficient for basic search configuration needs. Search will index the content and store them with the type of text for use in displaying in the search results. You will only need to use other types for more advanced configurations that will not be discussed in this document.

If you added custom meta tags to the page, you should select Document Meta Tags from the Source drop-down with Matches set as None and Html Expr/Reg Expr should be blank. If your meta tag names don't match your field names you would need to specify Document Body for the source, Matches Html Expr and the Html Expr entered should match your meta tags. For example, for the meta tag <meta name="your_meta_tag_name" content="This is my content" /> you would enter <meta name="your_meta_tag_name" content="*" /> where * indicates the value you wish to store for the content field.

If the custom content is contained within a standard html tag, you should select Document Body from the Source dropdown with Matches set as Html Expr. For the Html Expr/Reg Expr field, enter the tag that the content is contained in. For example, if the subtitle is contained within an <h2 class="subtitle"></h2> field you would enter `<h2 class="subtitle">*</h2>`

Save the configuration and enter a url in the field below and click the Test button to check if the content fields are setup properly.

After you have tested your content field configurations re-crawl the site again using the Start Crawler interface.

Update your search template to render the custom content field. In the result loop you would just need to print out the content field using `<#print variable="result.your_content_field_name" #>`.

For example, the following would render the custom content field for each result item.

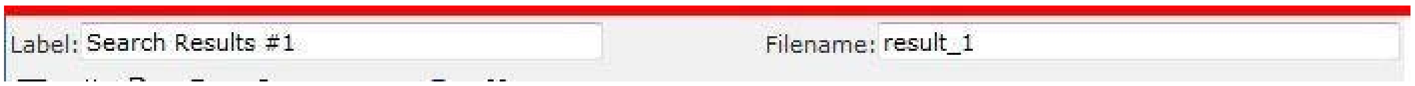
```
<#foreach variable="results" use="result"#>

<# print variable="result.your_content_field_name" #>

<p><a href="<# result.link_url #>"><#print variable="result.icon"#>&nbsp;<#if
isnotnull="result.title_context" #><#print variable="result.title_context"
crop=80#><#else#>Unknown<#/if#></a>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;<a href="<#
result.link_url #>" target="_blank"><font size=-2>[new window]</font></a>

<br>

<font size=-1><#print variable="result.body_context"#></font>

<br>

<font size=-2><font color=#008000><#result.display_path#> - <#result.size#> -
<#result.modified_date#></font></font>

<br>

<#/foreach#>
```

CMS Template Implementation

Within the CMS you would need to implement a search wrapper page. The search wrapper page is essentially your search results page that includes all html in your mockup excluding the items rendered by search. To include the HTML returned from search, all you would need to do is include the following JavaScript on the search wrapper page where the HTML should be rendered.

```
<script type="text/javascript">

 document.write('<'+'script type="text/javascript"
src="http://search.crownpeak.com/cpt_search/result_1?'+location.search+'&format=js&account=
xxxxxxxxxxxx" /></'+'script>');

</script>
```

The items to note on the src is the search template being used (i.e. result_1) and the account number. You could get the search template name at the top from the search template editing interface (i.e. Templates > Search Results #1). This should show up in the Filename field.



Label: Search Results #1

Filename: result_1

The account number can be found at the top of the search admin interface.

Note that the javascript passes the query string of the page along to search (i.e. location.search). By default search looks at the query string parameter q= for the search term. If you need to use an alternate query string parameter, for example query=, you would need to edit your search template and change all references to cgi.q to cgi.query.

For example, instead of

```
<#search

page=cgi.page

title=cgi.q

body=cgi.q

use="results"

promotions="promo"

related="related"

spelling="spelling"

acronyms="acronyms"

sortby=cgi.sort

#>
```

Use this:

```
<#search

page=cgi.page

title=cgi.query

body=cgi.query

use="results"

promotions="promo"

related="related"

spelling="spelling"

acronyms="acronyms"

sortby=cgi.sort

#>
```

You will also need to update all pages that have a search forms to link to your search results page and pass the search parameters in a query string. For example:

```
<form name=”search” action=”/search-results.html” method=”get”>

<input type=”text” name=”q” value=””>

<input type=”submit” name=”submit” value=”Go” />

</form>
```

If needed, you could add javascript validation to the form to prevent empty queries.