User manual Content Platform
Version 0.1

Edgar Klerks (Sanoma Digital BV)

August 21, 2014

Version 0.1 of the Content Platform manual
api key: 0796b8160a7544ccbfc645195591ecaf7ecbeabb

Contents

1 Introduction

This document covers various usage patterns of the Dynamic Content Platform (DCP). It will try to be extensive; (un)fortunately the Content Platform offers a powerful interface for searching, therefore it is not possible to give all possible usage patterns in this manual. This manual tries to give an overview and shows the user how to combine various patterns, hoping it gives the user a taste of what the possibilities are. The manual will also encourages the user to experiment with several features.

When something can degrade your developer experience, it will display the following box:

You should read me

If you miss this, then working with the Content Platform will be difficult

When something can make your experience better, it will be displayed in the blue box:

You could read me

But for your general understanding it won't matter to much

When we have a working example (which you can paste into your code or browser), this box will be used:

I am an example

```
And you could try me out
```

If you don't like the boxes, please type "Idontlikeboxes" in Google, it will learn you how to add a like box to your webpage.

2 Overview

I want to skip to...

This section is a small overview, for the people who are in a hurry. The following recommendations are done:

- For the impatient there is the option to skip to section Quick start (for the impatient), which will provide enough material to start immediately.
- For the experienced, on could jump to Schema reference for a refence of the searchable fields.

The DCP is a RESTful API, which can communicate in various message formats. The primary goal of the Content Platform is searching and to ease the task for the user, advanced analysis is done onto the items of the Content Platform, which will be stored as searchable metadata. The Content Platform is built on SOLR , a very advanced facetting search engine. The Content Platform exposes part of the SOLR api to the user. This enables the user to create advanced queries for his[1] applications.

A user receives an API key, which provide access to the Content Platform. The key can constrained the accesible content, but it cannot constrain the search options, the Content Platform provides to a user.

3 Operations

In this section the provided search interface is described and some example queries are presented. We will start with some simple queries and end with more complex and advanced ones. The user should experiment with various options and especially try to combine the presented search possibilities.

Suggested reading order

First the user should look at the Resources, which are the prime entry points for the search features of the Content Platform. Then the user can either start with Searching or Faceting.

3.1 Definitions

API Key
The personalized key, which provides access to the Content Platform

Boosting
An operation, which modifies the order of the search results by changing the weight of the results according to a pattern

Collection
A constraint on an API key, which narrows the search on certain fields

Faceting
An operation, which groups the data following a specified pattern

Resource
An uri, which provides access to an interface of the Content Platform

SOLR
The internal search engine of the Content Platform

[1] Of course it could be her, the Content Platform sees developers as asexual, but its 'applications' looks rather bad. There is no target gender groups analysis done at the moment, although it would be interesting.

Schema
A description of all fields which should be searchable for the user

Searching
An operation, which filters and orders the data following a specified pattern

3.2 Resources

This section describes the endpoints and resources of the Content Platform. The Content Platform offers several resources to accomodate search patterns. For certain often used operations there are shortcuts, but a developer will use the advanced interface. The base url should always be:

`http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/`

This base url build up from the uri of the server, the specification of the API resource, the version of the API (currently 1.0), your key and finally the resource you want to use. The version is included to allow the user to use an old version of the API.

Before you go further

Before the user can access the Content Platform, the api key should be received and the IP addresses should be known by the Content Platform. Also note that the user should use the address: `http://ce.snmmd.nl`http://ce.snmmd.nl. Click the hyperlink to see if you have access!

The resource in table 3.2 can be used for searching.

resource	description
item/<item_id>/	retrieve an item
item/<item_id>/elements/	retrieve content elements of an item
item/<item_id>/pages/	retrieve content element per page
item/<item_id>/related/	retrieve related items to this item
item/<item_id>/related-images/	retrieve related images to this item
search/	The main search resource, which defaults to search/include_details
search/exclude_details/	Search subresource, which exclude nested images
search/include_details/	Search subresource, which include nested images
magazine/<publication name>/	retrieve all issues of a magazine from <publication name>
magazine/<publication name>/<issue name>/	retrieve information about a certain issue

Table 1: Resources used for searching and retrieving

All these resources can and should be queried by a simple GET request with url encoded parameters. There are several parameters relevant to the user. Table Resources shows the features of the search resource.

parameter	description
q	A search query which highlight results, search to title, description and content
..	and adds some preset boost values
solrq	A search query, which is passed directly to SOLR. Default searches title.
fq	A filtering query, which don't boost the search terms or highlight terms
b	Boosting of a query, changes the weight of the results following a pattern
rows	The number of rows returned
start	The offset of the row to start
sort	Sort queries on a field
format	Change the document format (xml, json, html)

Table 2: Search parameters

The index resource has only one parameter (see 3.2)

parameter	description
format	Change the document format (xml, json, html)

Table 3: Item parameters

Now let explore the syntax for filter queries and queries.

4 Searching

4.1 Query syntax and examples

The syntax follows the syntax of `http://wiki.apache.org/solr/SolrQuerySyntax`SOLR Syntax which is quite complicated. This manual provides the user a practical and easily machine generated syntax, but it will not describe the SOLR syntax fully instead it will present a case and then dive into the various syntactic instruments a user can use to reach his goal.

Url encoding is omitted for readability, but the developer should be aware he has to encode the request query prior to testing an example. Most modern browsers do this automatically.

Use your key

The url we are using in the examples is:
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/

We first look into a simple search strategy, where SOLR decides how the Content Platform is searched. Later we will examine a more complex approach.

4.1.1 Simple searches

The simplest form of searching is a full text search. This we can do without any special syntax. The searched keywords should be separated by spaces in the **q** (see table 3.2) parameter. Let's say we want to search for quacking ducks.

Searching for quacking ducks in Dutch

```
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/?q=kwaakeend
```

We can modify the meaning of the keywords a bit, a user could use **+/-** to change the role of the keyword. In the following example we don't search quacking ducks (in dutch again):

Searching for not quacking ducks in Dutch

```
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/?q=-kwaak+eend
```

Now the Content Platform excludes all articles with the word kwaak(quack) in it, and includes all articles with the word eend (duck). The results are rather curious [2] at the time of writing of this manual:

```
"Worden deze <em>eendjes</em> platgewalst?"
```

As you can see, the relevant word is highlighted

SOLR query versus Filter query

When using a solr query ("q") the search terms gets automatically highlighted in the returned document, when using a filter query the highlighting feature will not be present, but the search will be faster. Thus a filter query only filters.

The simple search queries are especially suited for human generated searches.
If we want to find the phrase "quacking duck" then we should enclose it in quotes:

Searching for quacking ducks revisited

```
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/?q=
"kwakendeeend"
```

4.1.2 Searching specific fields

To search specific fields, we have various instruments to our disposal. Let us first examine the solr query (solq).

We would like to get all articles from a certain magazine, we could use the field operator(**:**):

```
<field>:<search expression>
```

Back to the example. So we want all articles of "Panorama Blad" and don't want any preset boosting, we can ask this to the Content Platform like this:

Searching for panorama blad

```
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/?solrq=
publication:"PanoramaBlad"
```

And of course we could **not** search for "Panorama Blad":

Searching not for panoram blad

```
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/?solrq=
-publication:"PanoramaBlad"
```

See for more operators 4.1.3.

[2] Which is of course correct, flattened ducks are physically unable to quack, although philosophically it would be correct to state, that ducks are defined by their quacking, thus undefined by their flattening. Unfortunately, such subtleties are not provided in the search capacities yet.

4.1.3 Combining searches

Watch your fields

If you are searching fields, which aren't contained in the Content Platform, you will get a **400 Bad request**, for an overview of fields see Schema reference.

Of course we don't want to search one field, we want to search many fields and combine them to meaningful queries. This can be achieved by the operators presented in 4.1.3.

Let's present some use cases. We want to search for all publication of a specific magazine in a time range of begin 2012 and end 2013 about dangerous feeding habits.

First we built the query.

We want to specify one publication.

Search all viva articles, publication selection

```
publication:"Viva Blad"
```

Then we want to specify our time range.

Search all viva articles, time range

```
created:[2011-12-31T23:59:59.999Z TO 2013-01-01T23:59:59.999Z]
```

And then add our keywords:

Search all viva articles, keywords

```
gevaarlijke eetgewoonte
```

The complete query will become thus;

Search all viva articles about dangerous feeding habits between begin 2012 and end 2013

```
publication:"Viva Blad" AND created:[2011-12-31T23:59:59.999Z TO 2013-01-01T23:59:59.999Z]
AND gevaarlijke eetgewoonte
```

Click to test the `http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/?solrq=publication:"VivaBlad"ANDcreated:[2011-12-31T23:59:59.999ZTO2013-01-01T23:59:59.999Z]ANDgevaarlijkeeetgewoontenquery`.

Strangely enough no articles about this fascinating subject are to be found in the Content Platform.

Operator examples

Some examples of operators which are useful for searching the Content Platform.

AND	narrowing the search	description:(wonen AND leven)
OR	broadening the search	title:eend OR title:kwaak
-	excluding search expr	-publication:"Panoram Blad"
+	include search expr	description:(+wonen OR -buiten)
()	grouping search expr	description:(wonen OR -buiten)
:	field selecting	description:wonen
[a TO b]	range query	date_created_cl:[* TO NOW]
""	quote text	title:"feeding hazards"

4.2 Filtering search results

In the previous sections, we only talked about solrq, which gives us free highlighting. But sometimes we have fields, which we don't want to highlight. If this is the case, we are not searching on these fields, but we are filtering on these fields. The syntax of a filter query is the same as the syntax for a solr query (see Searching).

We can combine the two queries. Lets filter on all publications which contain the word crimineel, but where we don't want to filter on and a user defined query, which we want to highlight.

Searching in the criminal database with the keyword muziek

```
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/?solrq=
description:muziek&fq=description:crimineel
```

Which clearly shows there are some connections between music and criminals, although this is interesting, it is more interesting to us, that the results are highlighted as expected:

```
naar top 40-<em>muziek</em>, klassieke <em>muziek</em>...
```

The writer of this manual enjoys breakcore, which is btw not researched in the original article, but is very alike top 40.

5 Faceting

Faceting is essentially grouping of search queries, which keeps track of the amount of found articles. The Content Platform is quite big and faceting can be a useful strategy to help the user searching. It breaks down a query in facets.

For the developer it can be a handy tool to get an overview of all the possible field values (just like a group by statement in SQL would do). We follow the syntax of `http://wiki.apache.org/solr/SolrFacetingOverview`Solr.

To explore the possibilities of faceting, we create a couple of examples as starting point for the user.

Facetting fields

The Content Platform search resource (see 3.2) understands the following parameters for facetin queries:

name	description
facet.field	the field on which we want to facet
facet.query	a query on which to facet

5.1 Simple field facet

As user, we want to have all the editions of “Libelle Blad”, to do this we can facet on issue:

Faceting libelle blad

```
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/?solrq=
publication:"LibelleBlad";facet.field=issue
```

[3]

The results of this query, will have all issues sorted:

Faceting result

```
facet_fields": { "issue": { "2010-08": 519, "2010-09": 452, "2010-10": 621, "2010-20":
848, "2010-21": 536, "2010-22": 473, "2010-23": 564, "2010-24": 619, "2010-25": 436,
"2010-26": 420, "2010-27": 496, "2010-28": 595, ...
```

Thus showing all available issues. By sorting on date the issues can be ordered chronologically.

5.2 Query facet

Sometimes we want to specify more exactly on what to facet. Such a query is called a facet query. Unfortunately the Content Platform doesn’t have this in the current version.

6 Sorting

Sorting can be done with the parameters **sort**. Two examples are presented, one for sorting ascending and one for sorting descending.

Sorting on date ascending

```
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/?sort=
createdasc
```

[3]The semicolon can be used as separator too, see the `http://www.w3.org/TR/html401/appendix/notes.html#h-B.2.2`w3c notes on the subject.

Sorting on date descending

`http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/?sort=createddesc`

7 Retrieving an Item

To retrieve an item in full details, the user can retrieve it as follow:

Retrieving an Item

`http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/item/51859a2e3b21db04847ba3c3`

The parameters of the item resource are listed in Resources.

8 Boosting

While boosting is outside the scope of this manual (at this moment), the eager user can find information about boosting on the `http://wiki.apache.org/solr/SolrRelevancyFAQ#index-time_boosts` SOLR site. See the table at Resources for the boosting parameters[4].

[4] The right answer is the query parameter "b".

9 Quick start (for the impatient)

Before you go further

Before the user can access the Content Platform, the api key should be received and the IP addresses should be known by the Content Platform. Also note that the user should use the address: `http://ce.snmmd.nl`http://ce.snmmd.nl. Click the hyperlink to see if you have access!

This section is a fast forward through all the examples. You can click on the urls and open it in a webbrowser. You should not forget to change your key in the url.

Simple searching

Difference between q and solrq

The **q** parameter searches title, content and description and can be changed by the DCP. The parameter **solrq** is send directly to SOLR and is passed unchanged. Solr only searches in the title.

Full text search on title, content, description:

Searching for quacking ducks in Dutch

`http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/?q=kwaakeend`

Search for non quacking ducks with **q**.

Searching for not quacking ducks in Dutch

```
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/?q=-kwaak+eend
```

Searching phrases:

Searching for quacking ducks revisited

```
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/?q=
"kwakendeeend"
```

Searching specific fields

Search some specific field:

Searching for panorama blad

```
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/?solrq=
publication:"PanoramaBlad"
```

Negative search on field:

Searching not for panoram blad

```
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/?solrq=
-publication:"PanoramaBlad"
```

More complicated searches, search with a range on created in a certain publication and full text search:

Search all viva articles about dangerous feeding habits between begin 2012 and end 2013

```
publication:"Viva Blad" AND created:[2011-12-31T23:59:59.999Z TO 2013-01-01T23:59:59.999Z]
AND gevaarlijke eetgewoonte
```

run `http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/?solrq=publication:"VivaBlad"ANDcreated:[2011-12-31T23:59:59.999ZTO2013-01-01T23:59:59.999Z]ANDgevaarlijkee`

Note that we use AND. See Combining searches for more operators.

Filtering query

Filter query and highlighting query mixing. Filter queries follow same syntax as normal query.

Searching in the criminal database with the keyword muziek

```
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/?solrq=
description:muziek&fq=description:crimineel
```

Faceting query

Group a field and count the number of items associated:

Faceting libelle blad

```
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/?solrq=
publication:"LibelleBlad";facet.field=issue
```

Index

10 Schema reference

The following fields are searchable through the API.

field_name	type	required	description
authors	string	optional	authors of article
available_in_source_feed	boolean	optional	if the item is in the source feed
category	string	optional	human set classification
classification	string	optional	classifications
classification_set_by_hand	boolean	optional	is class. set manually?
cluster	string	optional	cluster of publication
colors	string	optional	colors in image
comments	text	optional	comments on article
content	text	optional	body of article
content_suggestions	text_suggestions	optional	generated suggestion vectors
content_type	string	optional	Article, MagazineArticle, Image, MagazineImage, Magazin
created	tdate	optional	Creation date of article
date_created_cl	date	optional	added to content library
date_modified_cl	date	optional	last changed content library
description	text	optional	description of item
dpi	int	optional	dots per inch (images only)
extra_texts	text	optional	extra texts in article
faces	int	optional	number of faces in the picture (works rarely)
fuzzy_hash	string	optional	special hash for determine if content shares features wit
height	int	optional	height of a picture
id	string	required	primary key for a document
image_classification	string	optional	classification derived from images which are alike this im
image_main_classification	string	optional	primary classification
image_types	string	optional	filetype of image
images	string	optional	images of item
is_archive	boolean	optional	item is in archive (woodwing)
is_rest	boolean	optional	item is rest material
issue	string	optional	issue of the magazine
key_phrases	string	optional	most important phrases in document
keywords	string	optional	most important words in document
last_modified	date	optional	last modification date of original document
license_embargo_end_date	date	optional	date when embargo (period of exclusiveness) ends
license_embargo_start_date	date	optional	date when embargo (period of exclusiveness) starts
license_level	int	optional	License number (internal use)
license_name	string	optional	Human license name
locations	string	optional	Found locations in article
long_text_score	double	optional	score of quality of text (0.4>) is generally reasonable
long_text_word_count	int	optional	word count
magazine	string	optional	related magazine
mediatool_id	string	optional	id of mediatool (internal)
organizations	string	optional	organizations in text
orientation	string	optional	orientation of image
origin	string	optional	origin of item (feed, woodwing..)
original_id	string	optional	original id
original_url	string	optional	original url

Table 4: Schema reference table a-o

field_name	type	required	description
perceptual_hash	string	optional	internal usage
perceptual_words	text	optional	internal usage
persons	string	optional	persons found in text
pixels	int	optional	number of pixels (image only)
popularity	externalPopularityScore	optional	measured popularity of item
possible_sharing	string	optional	Unknown?
product_classification	string	optional	classification for home decoration product
product_classification_set_by_hand	boolean	optional	is the property set by hand
publication	string	optional	associated publication from item
resource	string	optional	resource from item
reusability	string	optional	is item reusable
section	string	optional	section of item
sentiment	string	optional	is the item's tone negative or positive (or n
share_states	string	optional	Unknown?
space_classification	string	optional	classification for home decoration space s
space_classification_set_by_hand	boolean	optional	is the property set by hand
spelling	spell	optional	Internal field
style_classification	string	optional	classification for home decoration styles
style_classification_set_by_hand	boolean	optional	is the property set by hand
subject	text	optional	What is the subject of the item .
subsection	string	optional	subsection of the item
tags	textgen	optional	tags found in item
title	text	optional	title of item
url	string	optional	url for webcontent of item
width	int	optional	width of image

Table 5: Schema reference table p-w

Element objects

field_name	type	required	description
content	markdown	optional	content of the element
coordinates	list	optional	coordinates of the element on the page
editable	boolean	required	if this element could be edited by an editor in chief
element$_{type}$	string	required	HeadingElement, IntroElement, TextElement, StreamerElement, CreditsEl
guid	string	optional	unique identifier of this element
is_deleted	boolean	required	if this element is deleted by an editor in chief
modified_content	string	optional	if edited by an editor in chief the modified content
original_order	int	required	natural order of the element
modified_order	int	optional	order if it's edited by an editor in chief
style_code	int	required	style code of the element

Table 6: Schema reference elements

11 Bonus section: Frequently Asked Questions

When I click on a "woodwing9.publisher.intra url" it doesn't work.
WoodWing is the workflow system we use to make magazines. As it is only accessible by Sanoma empoyees, you can't access it from outside the sanoma network.

The item object is so big that I don't know where to start. Which attributes do I really need?
For a simple result response we suggest to use title, description, created en medium_url. To also display a the content use the content attribute.

I only know how to process XML, can you make me a special XML endpoint?
By just adding the queryparamter format=xml at the end of an api endpoint it will automatically return a valid xml file

Can you please provide more examples for my usecase?
Of course, here are some common queries:

I want all articles with "aardbei"

```
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/exclude_
details/?q=aardbei&fq=(content_type:ArticleORcontent_type:MagazineArticle)
```

I want all images with "aardbei"

```
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/exclude_
details/?q=aardbei&fq=(content_type:ImageORcontent_type:MagazineImage)
```

I want all items from an online source with "aardbei"

```
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/exclude_
details/?q=aardbei&fq=(content_type:ArticleORcontent_type:ImageORcontent_type:Video)
```

I want all items from a paper source with "aardbei"

```
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/exclude_
details/?q=aardbei&fq=(content_type:MagazineImageORcontent_type:MagazineArticle)
```

I want all items from publication Viva Blad with "aardbei"

```
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/exclude_
details/?q=aardbei&fq=publication:"VivaBlad"
```

I want all items from publication Viva Blad with aardbei and only images

```
http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/exclude_
details/?q=aardbei&fq=publication:"VivaBlad"&fq=(content_type:ImageORcontent_type:
MagazineImage)
```

I can only process XML

```
Add format=xml at the end e.g.

http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/search/exclude_
details/?q=aardbei&fq=publication:"VivaBlad"&fq=(content_type:ImageORcontent_type:
MagazineImage)&format=xml
```

I want to all issues from a Viva Blad

`http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/magazine/VivaBlad/`

I want to retrieve all pages in a magazine

`http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/magazine/VivaBlad/32-2014/`

I want to retrieve all elements from an article

`http://ce.snmmd.nl/api/1.0/0796b8160a7544ccbfc645195591ecaf7ecbeabb/item/53c63196fba98412ab835e16/elements`