Table of Contents

Taxon

What is Taxon?

Taxon is an Open Source project to automatically classify a text according to a given taxonomy.

The resulting metadata can be used as tags on a web page, as metadata in a record management system, as keys in a mail distribution system and many other places.

Metadata

Metadata is moving into more and more systems. Some metadata are easy to obtain from the system, like the time a document is created, the author if you are logged in and many more.

But one is much more complicated to create than the others – the subject. The subject is the very short description – usually one or two words - of what the text is about. This particulary piece of metadata requires an understanding of the content of the text.

Most record/knowledge management systems are capable of handling metadata, including the subject, but very few are capable of actually creating the subject metadata. And that's where Taxon comes in. Taxon is created with the specific purpose of creating the subject metadata.

Automated classification

Classification is the process of applying metadata or tags to a text. Classification has been used in libraries for centuries and is now slowly moving into knowledge management.

Taxonomies

A taxonomy is a structure for placing texts in classes and subclasses. It is a bit like the file manager where the directories/folders are the classes and the file are the terms. The terms in each class define the class.

The Open Source project

Taxon is an Open Source project. It is licensed under GPL 3.

Strategy for version numbers

Major version number 1.x

The general idea is to let the major version number reflect the version of the API, so that all versions within 1.x have the same API and should be able to communicate as well as using the taxonomies.

Subversion number x.0

The subversion number reflects minor updates, bugs fixes, speed improvement etc. of Taxon.

Current version

As of 3/5-2013 the version is 2.1.2.

Sponsorship

Being a sponsor for Taxon means that you can have a feature implemented faster than might have been the plan.

It also means that you can have your ideas and features implemented as long as they have a general value for many users.

Sponsors

The scripts

- getallclasses.php
- getclasses.php
- getallterms.php
- getterms.php

are all sponsored by the Danish municipality Syddjurs Kommune (www.syddjurs.dk).

Installing Taxon

Create a directory for Taxon to live in, e.g. taxon/. Enter the directory.

Installing Taxon core

Unpack the file taxon-X.y.tar.gz with

```
tar -zxvf  taxon-X.y.tar.gz
```

Now there is a directory called system/. The directory contains all the files for Taxon.

The files

documentation/

- COPYRIGHT.txt
- LICENSE.txt
- README.txt
- Taxon-2.1.pdf
- VERSION.txt

includes/

- niceJSON.php

taxon/

- classify.php
- getallclasses.php
- getclasses.php
- getallterms.php
- getterms.php

taxonomies/

- test.json
- test_lookup.json

test/

- test_getallclasses.php
- test_getclasses.php
- test_getallterms.php
- test_getterms.php
- test_taxon.php

tools/

make_lookup_taxonomy.php

update_taxonomy.php

That's it!

Taxon is now installed on your system.

To test Taxon go to the system/test/ directory and type:

```
php test_taxon.php
```

Enter 'test' as the name of the taxonomy and 'Denmark' as the sample text.

If everything is okay the result is:


```
{
  "01":
  {
    "title":"EU countries",
    "exclusive":0,
    "hidden":0,
    "thresholdWeight":5,
    "thresholdCount":1,
    "thresholdCountUnique":1,
    "terms":
    {
      "denmark":
      {
        "weight":5,
        "count":1,
        "firstpos":1,
        "hits":
        {
          "denmark":1
```

```
            },
            "requireTerm":"",
            "excludeOnTerm":"",
            "requireClass":"",
            "excludeOnClass":"",
            "required":0,
            "hidden":0
          }
      },
      "scoreCount":1,
      "scoreWeight":5,
      "scorePosition":1,
      "scoreFirstPosition":10,
      "scoreTotal":15,
      "classificationMethod":"Full classification",
      "scoreConfidenceCoefficient":100
}
```


Installing a taxonomy

Taxon uses taxonomies in the JSON format, see the section on 'JSON taxonomy format' below. To install a taxonomy, copy your taxonomy to the taxonomies/ directory in the system/ folder. Go to the tools/ directory and run:

```
php make_lookup_taxonomy.php ../taxonomies/<NAME-OF-YOUR-TAXONOMY>
```

You now have a new taxonomy file in the taxonomies/ directory, called <NAME>_lookup.json.

Now Taxon is ready for use. Enjoy!

The API

Calling Taxon (classify.php)

Include the file classify.php, located in system/taxon/.

Description

Classifies the $text according to the $taxonomy with respect to the $settings.

```
string classify($taxonomy, $text, $settings = array())
```

Parameters

$taxonomy is the path and the name of the lookup taxonomy to use. The path can be relative to the calling directory or absolute. (Note: This is different than Taxon 1.0.)

$text is the text to classify in plain UFT-8 text.

$settings is an array of settings, possibly empty.
Possible settings are:

numberResultsReturned, a number equal to or greater than 0. 0 means all.

ignoreTermConstraints, either 0 or 1.
- 0 means use the full classification.
- 1 ignores the requireTerm, excludeOnTerm, requireClass, ExcludeOnClass as well as the class thresholds.

onNoResultsIgnoreTermConstraints, either 0 or 1.
- 0 If a full classification returned 0 results do nothing.
- 1 If a full classification returned 0 results another classification is run with the ignoreTermConstraints.

onNoResultsUseAlternativeTaxonomy, either 0 or 1.
- 0 If a full classification returned 0 results do nothing.
- 1 If a full classification returned 0 results another classification is run with the onNoResultsUseAlternativeTaxonomy.

returnShortResult, either 0 or 1.
- 0 Return the full result.
- 1 Return short result, i.e. only ID and title.

Return value

A JSON string with the found classes if any. The string "No results" is returned on no results.

```
  {
"[0-9.]+":
{
  "title":"<STRING>",
  "exclusive":[0|1],
  "hidden":[0|1],
  "thresholdWeight":[0-9]+,
  "thresholdCount":[0-9]+,
  "thresholdCountUnique":[0-9]+,
  "terms":
  {
    "<STRING>":
    {
      "weight":[0-9]+,
      "count":[0-9]+,
      "firstpos":[0-9]+,
      "hits":
      {
        "<STRING>":[0-9]+
      },
      "requireTerm":"<STRING>",
      "excludeOnTerm":"<STRING>",
      "requireClass":"<STRING>",
      "excludeOnClass":"<STRING>",
      "required":[0|1],
      "hidden":[0|1]
    }
  },
  "scoreCount":[0-9]+,
  "scoreWeight":[0-9]+,
  "scorePosition":[0-9]+,
```

```
    "scoreFirstPosition":[0-9]+,
    "scoreTotal":[0-9]+,
    "classificationMethod":"<STRING>",
    "scoreConfidenceCoefficient":0-100
}
```

"scoreCount":[0-9]+

is the number of

```
"scoreWeight":[0-9]+,
"scorePosition":[0-9]+,
"scoreFirstPosition":[0-9]+,
"scoreTotal":[0-9]+,
"classificationMethod":"<STRING>",
"scoreConfidenceCoefficient":0-100
```

Alternative taxonomy

As of version 2.0 Taxon is capable of using an alternative taxonomy if the full classification returned no results.

The alternative taxonomy has the same JSON format as the full taxonomy.

There are several ways to build and use an alternative taxonomy, e.g. using the TaxonHub Client.

The TaxonHub Client is exposes a web page to the users and collects their suggestions for new terms in the alternative taxonomy. The terms are not qualified, i.e. they have default weights and settings. The suggestions are sent to the TaxonHub for later qualification. This kind of alternative taxonomy should be used with care.

You can supply your own ready-made taxonomy as the alternative taxonomy or you can provide your system to build the alternative taxonomy.

Get all the classes (getallclasses.php)

Include the file getallclasses.php, located in system/taxon/.

Description

Retrieves all the classes in the $taxonomy.

```
string getallclasses($taxonomy, $settings = array())
```

Parameters

$taxonomy is the path to the taxonomy to use. The path can be relative to the calling directory or absolute.

$settings is an array of settings, possibly empty.
Possible settings are:
    return-type: empty or 'json'. Default is empty which returns the result as text.

Return value

The list of all classes. Depending on return-type the result is in text or JSON.

Get all terms from all classes (getallterms.php)

Include the file getallterms.php, located in system/taxon/.

Description

Retrieves all the terms from all the classes in the $taxonomy.

```
string getallterms($taxonomy, $settings = array())
```

Parameters

$taxonomy is the path to the taxonomy to use. The path can be relative to the calling directory or absolute.

$settings is an array of settings, possibly empty.
Possible settings are:
    return-type: empty or 'json'. Default is empty which returns the result as text.

Return value

The list of all terms from all classes. Depending on return-type the result is in text or JSON.

Get all terms from a given class (getterms.php)

Include the file getterms.php, located in system/taxon/.

Description

Retrieves the terms from the class in the $taxonomy identified by $classid.

```
string getterms($taxonomy, $classid)
```

Parameters

$taxonomy is the path to the taxonomy to use. The path can be relative to the calling directory or absolute.

$classid identifies the class.

Return value

A JSON string with the found terms if any. Empty on no results.

Get all classes containing a given term title (getclasses.php)

Include the file getclasses.php, located in system/taxon/.

Description

Retrieves the classes containing the $term_title from the $taxonomy.

```
string getclasses($taxonomy, $term_title)
```

Parameters

$taxonomy is the path to the taxonomy to use. The path can be relative to the calling directory or absolute.

$term_title is the title of the term in plain UTF-8 text.

Return value

A JSON string with the found classes if any. Empty on no results.

JSON taxonomy format

```
{
   "system" :
   {
      "defaults" :
      {
         "numberResultsReturned":[0-9]+,
         "ignoreTermConstraints":[0|1],
         "onNoResultsIgnoreTermConstraints":[0|1],
         "onNoResultsUseAlternativeTaxonomy":[0|1],
         "returnShortResult":[0|1],
      },
      "versions":
      {
         "taxonomy_version":[0-9.x]+,
         "build":[0-9]+,
         "taxon_version":[0-9.x]+,
      },
      "information":
      {
         "taxonomy_name":"<STRING>",
      }
   },
   "classes":
      {
         "<STRING [0-9\.]+>":
         {
            "id":"<STRING [0-9\.]+>",
            "title":"<STRING>",
            "hidden":[0|1],
            "exclusive":[0|1],
            "thresholdWeight":[0-9]+,
            "thresholdCount":[0-9]+,
            "thresholdCountUnique":[0-9]+,
            "terms":
```

```
            {
               "<STRING>":
               {
                  "title": "<STRING>",
                  "weight":[0-9]+,
                  "required":[0|1],
                  "requireTerm": "<STRING>",
                  "excludeOnTerm": "<STRING>",
                  "requireClass": "<STRING [0-9\.]+>",
                  "excludeOnClass": "<STRING [0-9\.]+>" ,
                  "prefix": "<STRING>" ,
                  "suffix": "<STRING>"
               },
               ...
            }
         },
         ...
      }
}
```

Defaults

"numberResultsReturned":[0-9]+

The maximal number of results to return.

A number equal to or greater than 0. 0 means all.

Examples:

```
0
3
```

"ignoreTermConstraints":[0|1]

Ignores the requireTerm, excludeOnTerm, requireClass, ExcludeOnClass as well as the class thresholds.

0 means use the full classification. 1 means ignore term constraints.

Examples:

```
0
1
```

"onNoResultsIgnoreTermConstraints":[0|1]

If a full classification returned 0 results another classification is run ignoring term constraints.

0 Do nothing.
1 Run again ignoring the term constraints.

Examples:

0
1

"onNoResultsUseAlternativeTaxonomy":[0|1]

If a full classification returned 0 results another classification is run using the alternative taxonomy.
The alternative taxonomy is build through the taxonhub-client which gathers suggestions from the users in an alternative taxonomy. These suggestions are not qualified and should be used with care.

0 Do nothing.
1 Run again using the alternative taxonomy.

Examples:

0
1

"returnShortResult":[0|1]

Return the results with only the ID and the title. This can be used to display the result directly to the user.

0 Return normal result.
1 Return short result.

Examples:

0
1

Versions

"taxonomy_version":[0-9.x]+

The version of the taxonomy. This is controlled by the editor.

Examples:

2
2.3.x

"build":[0-9]+

The build number of the taxonomy. This is increased by 1 on every save.

Examples:

2
278

"taxon_version":[0-9.x]+

The version of Taxon to which this taxonomy correlates. For this version it should be 2.x.

Examples:

2.x

Information

"taxonomy_name":<STRING>

The name of the taxonomy. This is controlled by the editor.

Examples:

MyTax

Class parameters

"id":"<STRING [0-9\.]+>"

The ID of the class. Note that the ID is also the key for the class in the JSON tree.

Contains the digits 0-9 and the character '.'.

Examples:

01
01.02
01.02.03

"title":"<STRING>"

Title of the class – without the ID.

Contains characters.

Examples:

EU countries
Asia
Africa

"hidden":[0|1]

Whether to include the class in the result.
The class can be hidden, e.g. used as the target class a requireClass without the class being returned in the result.

Contains a number. The number can be 0 to not hide the class and therefor include the class in the result or 1 to hide the class and thereby excluding it from the result.

Default: 0. The default is used in various places, i.e. in the classification.

Examples:

0
1

"exclusive":[0|1]

Whether the class is exclusive in the result.

If the class is exclusive it is only returned if no other class with the same parent is returned.

Contains a number. The number can be 0 or 1.

Default: 0. The default is used in various places, i.e. in the classification.

Examples:

0
1

"thresholdWeight":[0-9]+

The threshold weight of the class.
The weight of the class is the sum of the weights of each term in class found in the text. The weight of a term is calculated by multiplying the number of hits by the weight of the term.
The sum of the weights of the terms must be greater or equal to the tresholdWeight for the class to be included in the result.

Contains a number. The number can be negative, 0 or positive.

Default: 5. The default is used in various places, i.e. in the classification.

Examples:

5
10

"thresholdCount":[0-9]+

The threshold count of the class.
The count of the class is the sum of the hits of each term in the class found in the text.
The sum of the hits of the terms must be greater or equal to the tresholdCount for the class to be included in the result.

Contains a number. The number can be 0 or positive.

Default: 1. The default is used in various places, i.e. in the classification.

Examples:

1
10

"thresholdCountUnique":[0-9]+

The threshold count of unique terms in the class.
The count of unique terms in the class is the sum of the hits of unique terms in the class found in the text.
The count of the hits of the terms must be greater or equal to the tresholdCountUnique for the class to be included in the result.

Contains a number. The can be 0 or positive.

Default: 1. The default is used in various places, i.e. in the classification.

Examples:

1
2

"terms": {}

An array of terms, see below.

Term parameters

"title": "<STRING>"

Title of the term. Note that the Title is also the key for the term in the JSON tree.

Contains characters.

Examples:

Denmark

Spain

"weight": [0-9]+,

The weight of the term.

The weight of the term is used to indicate the importance of the term.

Default: 5.

Examples:

2

5

"required": [0|1],

The term is required.
The term must appear at least once in the text for the class to fire.

Default: 0. The default is used in various places, i.e. in the classification.

Examples:

0
1

"requireTerm": "<STRING>",

The required term(s).
At least one of the required terms must appear at least once in the text for the term to fire.
The terms may or may not be a term in this or other classes.

Contains a list of strings separated by |. The can be empty. The strings in the list will be trim'ed.

Default: "". The default is used in various places, i.e. in the classification.

Examples:

term
term1|term2|term3

"excludeOnTerm": "<STRING>",

The excluding term(s).
If least one of the excluding terms appear at least once in the text the term is excluded.
The terms may or may not be a term in this or other classes.

Contains a list of strings separated by |. The can be empty. The strings in the list will be trim'ed.

Default: "". The default is used in various places, i.e. in the classification.

Examples:

term
term1|term2|term3

"requireClass": "<STRING>",

The required class(es).
The required class must also be fired for this term to be fired. This is used on more common terms that might be used in different contexts, e.g. Jaguar is both a car, an animal and a operating system.
The required class can be another class or the same class as the term belongs to. If the required class is the same as the class of the term, another term in the class is required for the term to be fired.

Contains a class ID.

Default: "". The default is used in various places, i.e. in the classification.

Examples:

01
01.02.03

"excludeOnClass": "<STRING>" ,

The excluding class(es).
If the excluding is fired this class is removed from the result.

Contains a class ID.

Default: "". The default is used in various places, i.e. in the classification.

Examples:

01
01.02.03

"prefix": "<STRING>" ,

The prefix(es) to a term.
In some languages there are some grammatical prefixes.

Contains a list of strings separated by |. The can be empty. The strings in the list will be trim'ed.

Default: "". The default is used in various places, i.e. in the classification.

Examples:

The|a
ge

"suffix": "<STRING>"

The suffix(es) to a term.
In some languages there are some grammatical suffixes.

Contains a list of strings separated by |. The can be empty. The strings in the list will be trim'ed.

Default: "". The default is used in various places, i.e. in the classification.

Examples:

```
ing|ed
er|en|erne
```

KNOWN BUGS

Multiple occurrences of the same term with different suffixes.

If a term occurs two different places in the taxonomy with two different suffixes, the generated lookup taxonomy has the combined suffixes.

TO-DO

A list of things to do in future versions. Just because ideas make to this list does not mean that they will make it into Taxon.

Handle multiple class IDs in requireClass and ExcludeOnClass

It should be possibly to require or exclude on more than one class. The system should be able to understand something like '01.02|04.08'.

Add RequireClass and ExcludeOnClass to classes

At the moment the RequireClass and ExcludeOnClass are used on terms. In the future it would be nice to have the same functionality on classes. This would be especially nice in combination with shadow classes.

WEIRD STUFF

Point excludeOnClass to the same class of the term

As it is now the term will always be excluded. This is because the term forces a hit in the class and then the term itself is excluded.

Another possible approach could be to only exclude the term if another term in the class was found.