Semantisches Wissensmanagement im Unternehmen: Konzepte, Technologien, Anwendungen

Prof. Dr. Stefan Linus Zander

Kapitel 3: Einführung in Semantic MediaWiki

Inhalte

Semantic MediaWiki (TODO: Change)

...is a content management software for

- Richly formatted hypertext documents
- Structured data

... based on the "Wiki Way":

- User-governed
- Collaborative
- Easy

...is based on MediaWiki (MW):

- Wiki engine for managing hypertext
- Very popular
  - Wikipedia, Wikia, >100,000 sites
- User interface in hundreds of languages
- Free, Open Source, open development
  - Some full-time developers (Wikimedia staff)
- Written in PHP and JavaScript, main DB MySQL

...is implemented as a MediaWiki extension:

- Modular add-on
  - Fully compatible with current and recent MW
- Free, Open Source, open development
  - Driven by volunteers: no staff
  - Integrated with MW development
  - Current maintainers: MK, Jeroen De Dauw
- Written in PHP and JavaScript, main DB MySQL
  - RDF DBs supported as secondary storage

...is well-established and stable

- Created in 2005
- Used on hundreds of sites (impossible to count)
- Extensive documentation in many languages
  - http://semantic-mediawiki.org/
  - Current maintainer: Karsten Hoffmeyer
- Twice-yearly user conference SMWCon
- Organisational support by OSDA (charity)
- One of the biggest MW extensions
  - Code, people, commit activity, user community

Was ist ein Wiki ?

Definition Ein Wiki ist ein webbasiertes Hypertext-System mit einer eigenen Markup-Sprache, das es Benutzenden ermöglicht, Webseiten direkt (d.h. ohne Programmierung) und online in einem Web-Browser zu erzeugen, zu lesen, zu verknüpfen und zu ändern.

Quelle: Angelehnt an Karin Haenelt, "Semantik im Wiki am Beispiel des MediaWiki und Semantic MediaWik", Fraunhofer, 2011.

Komponenten eines Wiki-Systems

Es gibt unterschiedliche Implementierungen von Wiki-Systemen; Die am häufigsten verwendeten Komponenten sind

- Datenbank
- Versionsverwaltung
- Suchfunktion

MediaWiki

MediaWiki ist die bedeutenste und weitverbreiteste Implementierung eines Wiki-Systems

- Open-source Wiki-Software
- Technische Basisplattform der Wikipedia
- Genutzt von zehntausenden Webseiten und Organisationen[1]

[1] Quelle: https://www.mediawiki.org/wiki/MediaWiki/de

Eigenschaften von Wiki-Systemen

- Wiki ist ein **Hypertext-System**
  - Seiten beinhalten Informationen zu einem Themengebiet
  - Darstellung von **thematischen Zusammenhängen** durch Verweise zwischen Seiten[1]
  - Kennzeichnung ob ein Verweis existiert oder nicht[2]
- Wiki ist **frei**
  - Quelltext: Gnu General Public License
  - Jede/r kann im Rahmen der Möglichkeiten tun und lassen was sie will
- Wiki ist **kollaborativ**
  - Zusammenarbeit mit anderen
  - schnelle Informationsbereitstellung und Korrektur
- Ein Wiki ist **niemals vollständig** oder perfekt
  - Es unterliegt einem **ständigen Prozess** von Erschaffung und Kollaboration
  - Mit einer inkludierten **Versionsverwaltung** der Inhalte[2]
  - Organisches Wachstum mit Semantic Drift
- Wiki ist von Natur aus **demokratisch**
  - Jeder hat die gleichen Rechte
  - damit erlaubt es eine Zusammenarbeit im Netz ohne Accounts und Passwörter
- Wiki ist **erweiterbar**
  - modularer Aufbau
  - Programmierschnittstellen

[1] "Wiki promotes meaningful topic associations between different pages by making page link creation almost intuitively easy". Bo Leuf, Ward Cunningham, "The Wiki Way: Quick Collaboration on the Web", Addison-Wesley, 2001.
[2] Das sind große Vorteile gegenüber dem bestehenden Web.
[3] Angelehnt an Rick Hegewald, "Ontologien und (semantische) Wikis", Problemseminar Ontologie-Management Institut für Informatik - Universität Leipzig

Semantic MediaWiki

- Extension für MediaWiki
- Erweitert MediaWiki mit Ontologie-basierten Sprachkonzepten
  - properties, concepts, subobjects
- Erfunden und entwickelt am KIT durch Markus Krötzsch, Denny Vrandečić[3] und Max Völkl, Institut für Angewandte Informatik und Formale Beschreibungsverfahren (AIFB)

[1] some url
[2] https://iccl.inf.tu-dresden.de/web/Markus_Krötzsch
[3] https://de.wikipedia.org/wiki/Denny_Vrandečić
[4]

Semantic MediaWiki ist ein offenes, kollaboratives Wissensmanagementsystem

Vorteile

- Jeder angemeldete Nutzer kann Content erstellen (anyone can edit)
- Einfach zu erlernen und zu nutzen
- Unmittelbare Contenterstellung und -nutzung (instant publish)
- Kollaborative Ontologie- und Inhaltserstellung (collaboration)
- Änderungsverfolgung
- Unterstützt den Aufbau von Communities (community building)
- Beinhaltet eine semantische Wissensbasis (knowledge base)
- Agile Entwicklung (agile development)

Nachteile

- Anyone can edit
- Open to span and vandalism
- Erfordert eine permanente Verbindung zum Server
- Information can become disorganized
- Editing is not as simple as Word
- Kein Berechtigungsmanagement "out-of-the-box"
- Semantic Drift
- Nur bedingt geeignet für Verschlusssachen
- Keine Kopplung von Inhalt und Sicherheitszonen

Quelle: "MediaWiki – Advantages of MediaWiki as a Content Mangement System" – Tutorial Part 2; https://youtu.be/nokM-3ZFwGs

Sprachelemente

Wikis stellen eine Reihe von Sprachelementen und Erweiterungen zur Erstellung und Verwaltung von Inhalten bereit:

- Seiten
- Namensräume
- Kategorien
- Vorlagen
- Magic Words
  - Parser Funktionen
  - Variablen
  - Behavior switches
- etc.

- Page Forms
- Semantic Result Formats
- Parser Functions
- Arrays
- etc.

Groß- und Kleinschreibung beachten!
MediaWiki und damit auch Semantic MediaWiki unterscheidet zwischen Groß- und Kleinschreibung bei Seitennamen etc. in der URL!

Datenstrukturelemente

Definition Ein Datenstrukturelement (ist Bestandteil einer ...) erlaubt die Spezifikation von Datenmodellen auf Basis des zugrunde liegenden Wissensrepräsentationsformalismus. Neben direkt in einem Wissensrepräsentationsformalismus verankerten Sprachelementen können auch weitere, unterstützende Elemente definiert sein.

Quelle: Eigene Definition; --> TODO: Unterscheidung zwischen Sprach- und Datenmodellelement

Semantic MediaWiki definiert 8 Datenstrukturelemente:

1. Seite (engl. Page)
2. Kategorie (engl. Category)
3. Attribute (engl. Properties)
4. Datentyp (engl. Datatype)
5. Namensraum (engl. Namespace) – kein DSE
6. Vorlage (engl. Template) - kein DSE
7. Subobjekt (engl. Subobject)
8. Konzept (engl. Concept)

MediaWiki: Syntax

Content Encoding

- Wiki pages are formatted hypertext documents
- Content is encoded in wikitext using a markdown-like syntax
- Wikitext input is converted into HTML output:

```
[[Link to wiki-page]]
[[Some page|link with custom text]]
[http://example.org Link to an outside URL]
==Header 1==
===Header 2=== (etc.)
''italics''
'''bold'''
* bulleted list
# numbered list
: indentation
```

Advanced Formatting Features

- Rich set of features for controlling content display
- Many HTML-like features supported (`<sub>`, `style`, ...)
- Advanced formatting elements:
  - Images
  - Tables

See online documentation for details
~> https://www.mediawiki.org/wiki/Help:Formatting

MediaWiki: The Structure of Page Names

Page names consist of 3 different parts

```
Namespace:Title/Subpagetitle
```

```
Example:   "User:Denny/Tests"
```

1. Namespace

- Prefixes, separated from title by colon `:`
- Not all prefixes that end in `:` are namespaces!
  - Available prefixes provided by MW, more can be added in configuration
  - Default: `Main` (empty), `User`, `Category`, `Template`, `Help`, `MediaWiki`, `File`, `Special`, `Project` (sitename)
- Purpose: distinguish basic "content types"
- Can have aliases (e.g. `File:` and `Image:`)

2. Pagetitle

- Defined during page creation
- MediaWiki determines whether page already exists
  - For existing pages, the page's content will be displayed
  - For non-existend pages, the edit view will be displayed

Page names are case-sensitive
MediaWiki distinguishes between upper- and lower-case letters in page names!

3. Subpagetitle

- Postfixes, separated from title by slash `/`
- Not all postfixes after `/` are subpages!
  - Enabled for certain namespaces
  - By default only for User and all Talk pages
- Often not appropriate for organising pages (rigid, hierarchical content structure)
- Small difference to pages with `/` in title
  - For example when moving pages
- Used in Wikipedia for multilingual page content

Namespaces

Definition Namespaces are prefixed in a page's URL and determine both the purpose and the model-theoretic semantics of a page. Based on the given namespace, the MediaWiki engine determines how to process the contents of a page[1].

Quelle: eigene Definition

- Every wiki page belongs to **one specific namespace**
  - Namespaces become part of the page's title, e.g., `Help:Namespaces/de`[3].
  - When no namespace is given during page creation, the page will be created in the **main namespace**[2].
- Namespaces determine the **purpose** of a page, i.e., how the contents of a page are interpreted by the wiki engine.
  - Example: Datatype information in property pages determine whether the specification of property values will lead to the creation of a new wiki page (in the default namespace) or whether property values are treated as data type values.
  - The wiki engine can also assess whether a certain **value holds for a property** or not (e.g., in the case of datatype `Date` or `telephone` etc.)
  - Contents of pages defined in the `Template:` namespace will be **transcluded** in other pages.
- The `Special` namespace is reserved for pages with fixed functionality (e.g. `Special:RunQuery`); no editing is possible

[1] MediaWiki provides 18 default namespaces. See https://www.mediawiki.org/wiki/Help:Namespaces/en#Standard_namespaces for a list of standard namespaces provided by MediaWiki
[2] The main namespace is not displayed in a page's URL but used internally by the mediawiki engine.
[3] Please note how subpages are used to implement mulit-language support on the official MediaWiki help pages.

Namespaces – Part 2

- Namensräume dienen der Gruppierung von logisch zusammengehörigen Seiten
- Funktionen können auf bestimmte Namensräume eingeschränkt werden
- Mediawiki hat standardmäßig **18 Namensräume**
- Textseiten stehen standardmäßig im Namenraum `Main`

Categories

Categories

Definition Categories are MediaWiki pages created in the `Category:` namespace. A category page represents a single category and allow for organizing other MediaWiki pages in predefined groups, represented by the category names.

Eigene Definition angelehnt an https://www.mediawiki.org/wiki/Help:Categories_

- MediaWiki ermöglicht das Klassifizieren von Seiten durch Kategorien
- Die Zuordnung einer Seite zu einer Kategorie erfolgt durch Einfügen von `[[Category:Categoryname]]` im Quelltext der Seite
- Eine Seite kann mehreren Kategorien zugeordnet werden
- Kategorien werden am Ende der Seite angezeigt
- Eine Hierarchiebildung der Kategorien ist durch Zuordnung von Kategorien zu Kategorien möglich

Using Categories

a) Adding a Page to a Category

- Links to Category pages mean: "page is in category"
  - Example: Add `[[Category:City]]` on page of `Cologne`

b) Defining Category Hierarchies

- Category Links on Category pages: "page is subcategory of"
  - Example: Add `[[Category:Settlement]]` on `Category:City`
- Category hierarchy can be any graph
  - Multi-Inheritage, Cycles, ...

c) Linking to a Category

- To create a link to a category, use a leading colon `:` before the category name[1]:
  - Example: `[[:Category:Help]]` --> Link displays as "Category:Help"
  - Example: `[[:Category:Help|Help category]]` --> The pipe `|` symbol allows to set an individual link text

[1] Without a leading colon, the current page would be added to the category

Category Hierarchies in more Detail

Mediawiki allows to build **category hierarchies**, i.e., it employs super- and sub-category concepts to resemble the 'broader' and 'narrower' relationships between categories known from taxonomical classification systems.

In order to make a category a sub-category of another category, add `[[Category:{Super_category}]]` to the sub-category page.

Example

```
On the category page 'city':
============================

[[Category:Settlement]] --> makes 'Settlement' the super category of category 'city'
```

Please note that category hierarchy can be any graph (multi-inheritance, cycles, etc.)

Working with Categories

1. Creating a new Category

- Create a new mediawiki page with a distinct name in the `Category` namespace
- Example

```
Category:Employee
```

--> Creates a new page named 'Employee' in the 'Category' namespace

- Enter additional information about the category on its page
- Link it to a category hierarchy

2. Adding Wikipages to a Category

To make a mediawiki page a member of a category, add a link to the category on the page[1]

```
[[Category:Employee]]
```

Adds the current page to the category "Employee"

> Category links can be placed at any location at a wiki page. However, due to maintenance and consistency reasons, it is good practice to define distinct locations in wiki pages for specifying category memberships.

3. Linking to a Category

To create a link to a category, use a leading colon before the category name (without this colon, the current page would be added to the category):

```
[[:Category:Help]]
```

--> Link displays as "Category:Help"

To change the link text, write the text inside the link tag after a pipe:

```
[[:Category:Help|Help category]]
```

--> Link displays as "Help category"

The Semantic Sauce

Introduction to Properties

Properties and datatypes are the basic way of encoding **semantic data** in Semantic MediaWiki.

Definition Properties define explicit relationships between wikipages, between wikipages and typed values, or between wikipages and subobjects. The explicit relationship is represented by the property definition page – a unique page created in the `property` namespace.

Source: Individual definition

Properties can link

- a wikipage to one or many wikipages
- a wikipage to a datatype value
- a wikipage to one or many subobjects

Some Introductory Remarks

- Properties do not just express a navigational plain link but a link with a certain meaning inherited from the property definiton page on the `property` namespace.
- The **semantics** of a property are determined by the annotations added to the property definition page.
    - e.g. defining value restrictions – so-called allowed values
    - e.g. defining a certain datatype – which in turn determines the interpretation of property values
        - e.g. treating 1,000 and 1000 as being equal in case of the datatype `Number`
    - e.g. defining taxonomical relationships between sub- and super-properties and thus inference in query answering

Special Properties

A property need to be **declared** before it can be used in annotations.

For the declaration, Semantic MediaWiki provides a set of so-called special properties[1] that allows for encoding property semantics.

Definition Special properties are a **predefined set of properties** with **built-in meaning** to control the behaviour of certain areas in a MediaWiki system. They–among other functions–are used to define the **semantics** of **individual properties** and are evaluated differently by the MediaWiki engine.

Source: individual definition based on https://www.semantic-mediawiki.org/wiki/Help:Special_properties

In consequence, we need to make a **distinction** between individually defined properties and special properties.

- Special properties control the behaviour of the Semantic MediaWiki system
- Properties represent individually defined vocabulary terms for expressing domain relationships.

Convention

- When we use the term "**property**", we always refer to individually defined properties used to model a universe of discourse.
- We use the term "**special property**", when we refer to the set of predefined properties provided by the Semantic MediaWiki engine.

[1] Semantic MediaWiki provides 61 special properties; here is a list of them: https://www.semantic-mediawiki.org/wiki/Help:Special_properties

Property Declaration

- Before a property can be used in annotations, it need to be **declared**.
- Properties are declared on their property definition pages.
- Property definition pages belong to the `property` namespace and their names represent the respective property in an annotation.
- Declarations are expressed as annotations, i.e., we use SMW's property syntax for defining a property's semantics (=declaration)
- A declaration usually involves an annotation about its type and allowed values.

Example

If we want to define a property, e.g. `has Population`, we first need to create a page with that name in the `Property` namespace and add a `Has type` special property on its declaration page with value `Number`.

~> `[[Has type::Number]]`

More information and examples can be found at https://www.semantic-mediawiki.org/wiki/Help:Property_declaration

Property Naming

Property naming is an important topic to avoid ambiguity and confusion and to minimize semantic drift.
It is good practice to create property names as a short **verb phrase**.

Example

`Germany's capital is Berlin. ↔ Berlin is the capital of Germany.`

Using a property called `capital` does not convey the intended meaning.

Better: `Germany's capital is [[Has capital::Berlin]]` <--> `Berlin is the capital of [[Is capital of::Germany]]`

Naming Recommendations

- Avoid reserved names – e.g. those used for magic words, special pages, behaviour switches, namespaces etc.
- Avoid using the name of a datatype – e.g. `[[Code::Qsdr-5t7Z-b99N]]` can not be changed to text etc.
- Avoid using certain kinds of punctuation – e.g. `::` (double colon), `-` (hyphen), `.` (dot), `|` (pipe), `#` (fence)

[1] More information can be found here: https://www.semantic-mediawiki.org/wiki/Help:Property_naming
[2] "Style Guidelines for Naming and Labeling Ontologies in the Multilingual Web" (https://dcpapers.dublincore.org/pubs/article/view/3626) contains some general guidelines how to name properties and URI's

Using Properties in SMW

The process of using properties in Semantic MediaWiki is twofold:

A) Creating Properties

- Property needs to be declared
- Consider property naming recommendations
- Use appropritate datatype depending on the envisioned object value[1]

B) Creating Annotations

- Using in-text annotations
- alternative: silent annotations using the `#set` parser function

[1] see https://www.semantic-mediawiki.org/wiki/Help:List_of_datatypes

Using Properties in Annotations

Properties are used in annotations the following way (a so-called in-text annotation)

Notation [[Property name::property value]]

This statement defines a value for the property of the given `Property name`.
The page on which the annotation is used displays the property value and **not the property assignment**.

Notations

- in-text annotation: `[[Is capital of::Germany]]` ~> displays Germany as HTML link on the page
- alternative text: `[[Is capital of::Germany|alternate text]]` ~> alternate text appears in place of the link
- hidden property: `[[Is capital of::Germany| ]]` ~> does not display any value at all

Silent Annotations using #set

The `#set` parser function allows to annotate data, i.e. assign values to a property, silently and avoid the `[[Property::value]]` syntax.

Example

```
{{#set:
 Has population=2,229,621
 |Located in country=France
}}
```

```
is similar to...

[[Has population::2,229,621| ]]
[[Located in country::France| ]]
```

The `set` parser function does not display anything, but saves the very same properties as data.

It is also possible to set multiple values to the same property:

```
{{#set:
 Has postcode=75001
 |Has postcode=75002
 ...
}}
```

```
{{#set:
 |Has text=fc00:123:8000::/%6;2001:db8::1428:57ab;2001:db8:0:8d3:0:8a2e:70:7344
 |+sep=;
}}
```

For full compatibility use the separator `|+sep=...` function to separate multiple values

Sources: (1) https://www.semantic-mediawiki.org/wiki/Help:Setting_values/Working_with_the_separator_parameter and (2) https://www.semantic-mediawiki.org/wiki/Help:Setting_values

Templates

Templates

Definition A template is a wikipage defined in the `Template:` namespace that defines content that can be transcluded in other wiki pages.

Quelle: eigene Definition

Templates

... can be created as any wiki page, but must be defined in the `Template:` namespace

... can contain almost any kind of wiki content

... can have parameters, the values of which will be inserted in the template's content during transclusion

... often used to embed semantic properties or subobjects in Semantic MediaWiki

- Templates allow for the inclusion of pre-defined content in wiki pages.
- This form of inclusion is called **Transclusion**

Templates: Transclusion

Definition Transclusion describes the process of embedding content defined in a template into another page. Transcluded content can be customized by parameters the values of which are are included in distinct locations during template invokation.

Quelle: Eigene Definition angelehnt an

Transcluded content can be controlled by **three** distinct commands

- `<includeonly>`
- `<onlyinclude>`
- `<noinclude>` – usually used for instructional content, i.e., how a template is to be used. This content will not be transcluded.

Always explicitly specify transcluding Content
It is recommended to explicitly markup the content in a template that is to be transcluded by using the provided commands and separate it from supplemental or instructional content that describes the usage of the template.

Templates: Tutorial Videos

Excellent tutorial videos about templates

YouTube hosts some excellent videos about the basic principles and formatting of templates as well as about the transclusion process and its controlling commands

- Introduction to templates
  https://youtu.be/IJ4BM5MFXmc
- Basic formatting of templates and transclusion commands
  https://youtu.be/SsLahlGX0Ls
- Template Variables
  https://youtu.be/X0QD5HT2qgc

Templates in Semantic MediaWiki

- Templates are often used for **harmonizing**[^1] semantic data and reducing semantic drift
  - ...by using pre-defined **semantic properties** in a template page
  - ...and by setting their allowed values through template **parameters** (often in conjunction with Page Forms)
  - ~> So, every page that transcludes a template contains the same semantic data and structure

[1] Harmonizing means to make something consistent and compatible

Customizing Template Content

Templates can have **parameters** that allow for passing individual data to template content that is to be transcluded.

Parameters within templates can either be specified...

- anonymously via the sequence of occurrence, i.e., `{{{1}}}`, `{{{2}}}`, etc.
- via specific parameter names, i.e., `{{{Parameter_name|default_value}}}`

In case a parameter is not set (i.e., it contains no value), a default value can be specified `{{{Parameter_name|default_value}}}`, `{{{1|default_value}}}`

Please note that parameters in templates need to be specified with three curly brackets `{{{Parameter_name||Parameter_number}}}` (N.B. '`||`' means 'OR' and is not part of the parameter syntax)

```
Syntax:
=======
{{{Parameter_name}}}       OR       {{{Parameter_number}}}

Example:
========
{{{project}}}              OR       {{{1}}}
```

Creating Templates

Every template needs to a have a **unique page name** in the `Template:` namespace; they can be created as any other wikipage.

Template URL: `{Semantic_MediaWiki_URL}\Template:Template_name`
Template Name: `Template_name`

```
<!-- additional content -->
...
[[property1::{{{parameter1}}}]]
[[property2::{{{parameter2}}}]]
...
<!-- additional content -->
```

Usage on a different wiki page – without parameters specified:

```
{{Template_name}}
```

Usage – with parameters specified on the page where the template it to be transcluded

```
{{Template_name
|parameter1=value1
|parameter2=value2
|...
}}
```

Templates can be used in any place in a page.

Templates: Syntax

```
<noinclude>
Dies ist die Vorlage zum Anlegen neuer nationaler und europäischer Förderprogramme.
Zum Anlegen eines neuen Förderprogramms einfach den folgenden Ausschnitt in den Quelltext
der neuen Seite kopieren und die Parameter entsprechend belegen:
<pre>
  {{Förderprogramm
   |Name=
   |Akronym=
   |Webseite=
   |Deadline=
   |Beschreibung=
   |Sonstige_Informationen=
  }}
</pre>
Das Template sollte im oben dargestellten Format genutzt werden.
Bis auf das Attribut Deadline können alle Felder mit Freitext befüllt werden;
Das Attribut Deadline erwartet Datumsangaben (nicht 'Ende September' sondern '30.09.2016').
</noinclude>
<includeonly> <!-- **** Hier beginnt das eigentliche Template **** -->
  [[Category:Förderprogramm]]
  [[Name::{{{Name|}}}]]
  [[Akronym::{{{Akronym|}}}]]
  [[Webseite::{{{Webseite|}}}]]
  [[Deadline::{{{Deadline|}}}]]
  [[Beschreibung::{{{Beschreibung|}}}]]
  [[Sonstiges::{{{Sonstige_Informationen|}}}]]
</includeonly
```

Concepts

Concepts

Definition Concepts are pages in the `Concept:` namespace and allow to dynamically compute page memberships based on the evaluation of query conditions defined on the concept page.

Quelle: Eigene Definition angelehnt an https://www.semantic-mediawiki.org/wiki/Help:Concepts

Motivation

- Sometimes, it is useful to determine category membership based on the occurrence of some specific property values.
- Reviewing whether membership conditions are still satisfied and manually altering categories is cumbersome and error-prone

Example

- Automatically annotate all currently running projects with a dedicated category e.g. `Running Projects` based on the evaluation of start and end date

Function

Meta: Could also be a summary

Concepts

... are pages defined in the `Concept:` **namespace**

... serve as categories with individually evaluated memberships

... are declared using the `#concept` parser function

... dynamically link pages to categories based on formulated **query conditions**

... conditions are specified in the form of an `#ask` query

... results of the `#ask` query automatically become members of the concept

... can be **browsed** to view the contents of some concept – similar to category pages

... can be used in **semantic queries** just like categories

... are very useful in **Page Forms** for defining auto-completion values

Additional Remarks:

Concept pages can have **additional content** (e.g. wikitext) – but this text does not have any effect on the definition of the concept.

The `#concept` parser function can only be used on pages in the `Concept:` namespace

Working with Concepts

A) Creating a Concept

```
{{#concept:
 [[Category:Event]]
 [[Has planned start::> Jan 1 2012]]
 [[Has planned finish::< Dec 31 2012]]
 |Semantic MediaWiki Cons in the year 2012
  that have been announced on this wiki.
}}
```

- A page with the concept name has to be created in the `Concept:` namespace
- Parser function `#concept` is used to define concepts.
  - It's first parameter is a concept definition that defines the conditions for selecting pages
  - It's second parameter is a short text that describes the concept.
- A list of matching pages is printed on the concept page

Source: Example taken from <...>

B) Using a Concept

```
{{#ask:
 [[Concept:Semantic MediaWiki Cons 2012]]
 |?Has location=Location
 |format=table
 |headers=plain
 |mainlabel=Event
}}
```

Result

Event	Location
SMWCon Fall 2012	Cologne, Germany
SMWCon Spring 2012	Carlsbad, CA, USA

TODO: Will concepts be displayed on wiki pages?

Example 2: Concept for all currently running Projects

Instead of annotating all currently running projects with a specific category (that certainly will be invalid for some projects after a certain amount of time), we can define a dynamic category in form of a concept.

```
{{#concept:
 [[Category:Project]]
 [[Has planned start::< {{CURRENTYEAR}}-{{CURRENTMONTH}}-{{CURRENTDAY}}]]
 [[Has planned finish::> {{CURRENTYEAR}}-{{CURRENTMONTH}}-{{CURRENTDAY}}]]
 |All currently running projects
}}
```

Concepts (Part 2)

Concepts

Definition Concepts are pages in the `Concept:` namespace and allow to dynamically compute page memberships based on the evaluation of query conditions defined on the concept page.

Quelle: Eigene Definition angelehnt an https://www.semantic-mediawiki.org/wiki/Help:Concepts

Problem

- Sometimes, it is useful to determine memberships to categories based on the occurence of some property values.
- Reviewing whether membership conditions are still satisfied and manually altering categories is cumbersome and error-prone

Example

- Automatically annotate all currently running projects with a dedicated category based on the evaluation of start and end date

Application Scenarios

- Concepts are useful, when the evaluation of query conditions is complex and/or needed in many `#ask` queries.
- Concepts help in simplifing semantic queries

Concepts: Syntax and Creation

Example

The following concept called `Concept:Semantic MediaWiki Cons 2012` describes Semantic MediaWiki conferences held in 2012[1]:

```
{{#concept:
 [[Category:Event]]
 [[Has planned start::> Jan 1 2012]]
 [[Has planned finish::< Dec 31 2012]]
 |Semantic MediaWiki Cons in the year 2012 that have been announced on this wiki.
}}
```

Creating Concepts

1. Create a new page in the `Concept:` namespace
2. Provide a suitable name for the concept (cf. naming categories)
3. Use the `#concept` parser function to define the query conditions
4. Provide a readable description for the concept as a second optional parameter
5. Add additional information when necessary

[1] Example is taken from https://www.semantic-mediawiki.org/wiki/Help:Concepts

Concepts: Usage

Concept pages

... can be browsed to view the contents of some concept, similar to category pages

... can be included in semantic queries like categories

Syntax

```
{{#ask:
 [[Concept:Semantic MediaWiki Cons 2012]]
 |?Has location=Location
 |format=table
 |headers=plain
 |mainlabel=Event
}}
```

Result

```
Event                      | Location
-----------------------------------------------------
SMWCon Fall 2012 Cologne | Germany
SMWCon Spring 2012         | Carlsbad, CA, USA
```

Semantic Data Modelling

Recap

From the lecture about semantic knowledge graphs we know that...

...in the Web, we distinguish between information and non-information resources

...we use IRIs to identify things

...IRIs should be resolvable/dereferenceable through a process called content negotiation

...Ontologies are used to for encoding semantics that can be used in knowledge graphs

...Ontologies contain terminological (TBox) and assertional (ABox) knowledge

...RDF is the representation framework for encoding factual knowledge in the Web

...RDF resembles a triple pattern (subject – predicate – object)

...RDF can also be used to encode terminological knowledge to form a vocabulary that can be used in knowledge graphs

Lernen lernen

Aspekte (die 4 K's) zukünftigen Lernens

Leitfrage: Was sind die Elemente zukünftigen Lernens ?

<span style="font-font-size: 4em; color: red; font-weight: 500">K</span>ollaboration

<span style="font-size: 4em; color: red; font-weight: 500">K</span>ritisches Denken

<span style="font-size: 4em; color: red; font-weight: 500">K</span>reativität

<span style="font-size: 4em; color: red; font-weight: 500">K</span>ommunikation