

# **Semantisches Wissensmanagement im Unternehmen: Konzepte, Technologien, Anwendungen**

Prof. Dr. Stefan Linus Zander

Kapitel 3: Einführung in Semantic MediaWiki

# Inhalte des 3. & 4. Kapitels

1. Einführung Wiki-Systeme
2. Semantic MediaWiki
3. Grundbegriffe und Definitionen
4. Sprach- und Datenstrukturelemente
5. Modellierung von Gegenstandsbereichen als Wissensgraph (~> [Modelling Information](#))
6. Kodierung von Faktenwissen (~> [Encoding Information](#))
7. Semantische Suche (~> [Retrieving Information](#))
8. Advanced Data Modelling Topics<sup>1</sup>
9. Extensions for Knowledge Graph Creation<sup>1</sup>

---

<sup>1</sup> Inhalte des 4. Teils der Vorlesung

# Inhalte dieses Foliensatzes

1. Einführung Wiki-Systeme
2. Eigenschaften von Wiki-Systemen
3. Semantik MediaWiki
4. Grundlegende Definitionen
5. Sprach- und Datenstrukturelemente
  - i. Pages
  - ii. Namespaces
  - iii. Categories
  - iv. Properties
  - v. Concepts
  - vi. Templates

# 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<sup>1</sup>

<sup>1</sup> 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<sup>1</sup>
  - Kennzeichnung ob ein Verweis existiert oder nicht<sup>2</sup>
- Wiki ist **frei**
  - Quelltext: Gnu General Public License
  - Jede/r kann im Rahmen der Möglichkeiten tun und lassen was er/sie/es 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<sup>2</sup>
  - 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

<sup>1</sup> "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.

<sup>2</sup> Das sind große Vorteile gegenüber dem bestehenden Web.

<sup>3</sup> Angelehnt an Rick Hegewald, "Ontologien und (semantische) Wikis", Problemseminar Ontologie-Management Institut für Informatik - Universität Leipzig

# Semantic MediaWiki

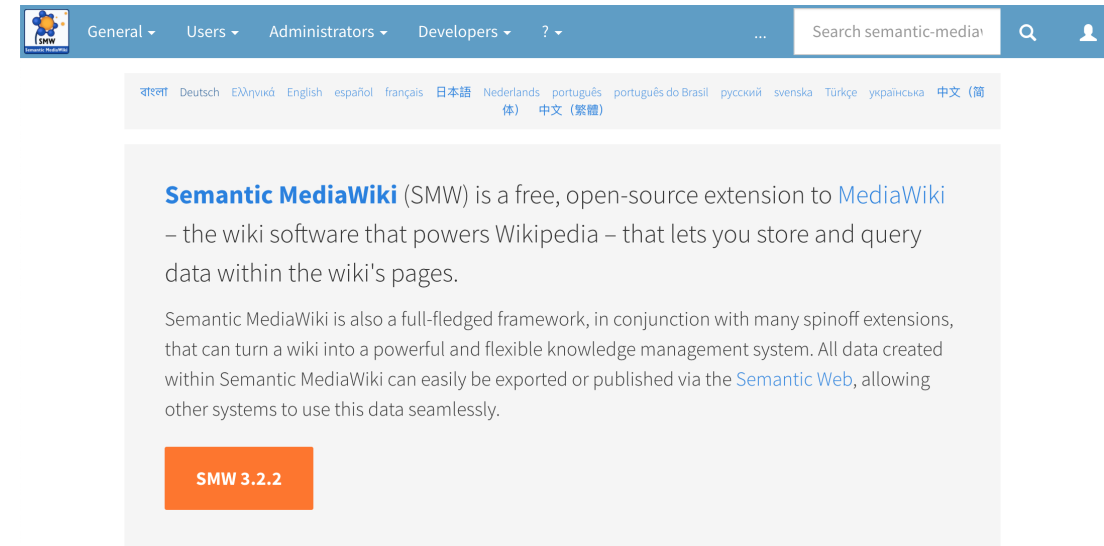
- Extension für MediaWiki, erstmals veröffentlicht 2005
- Erweitert MediaWiki mit **Ontologie-basierten Sprachkonzepten** für den Aufbau und die Verarbeitung von Daten in Form **semantischer Wissensgraphen**
  - properties, datatypes, concepts, subobjects
- Erfunden und entwickelt am KIT durch **Markus Krötzsch**<sup>1</sup>, **Denny Vrandečić**<sup>2</sup> und **Max Völkel**<sup>3</sup>, ehemalige Wissenschaftliche Mitarbeiter am Institut für Angewandte Informatik und Formale Beschreibungsverfahren (AIFB)<sup>4</sup>
- Überführt Wiki-Systeme in **semantische Wissensbasen**

<sup>1</sup> [https://iccl.inf.tu-dresden.de/web/Markus\\_Krötzsch](https://iccl.inf.tu-dresden.de/web/Markus_Krötzsch)

<sup>2</sup> [https://de.wikipedia.org/wiki/Denny\\_Vrandečić](https://de.wikipedia.org/wiki/Denny_Vrandečić)

<sup>3</sup> <https://www.maxvoelkel.de/>

<sup>4</sup> <https://aifb.kit.edu/web/Hauptseite>



# Why Semantic MediaWiki ?

Wikis are a great tool for **collecting** and **sharing knowledge** in communities and organizations. This knowledge is mostly **contained** within **texts** and multimedia files, and is thus easily accessible for human readers. But though wikis are very good for storing and retrieving individual facts, they are less useful for getting queried or aggregated information.

As a simple example, let's say you use a wiki that stores information about projects related to your organization. You have the following simple question:

*"What are the active projects that were started in 2012?"*

This should be an easy question to answer, but in fact it's not - you would have to read through all of the pages about projects every time you wanted to answer the question. Text searches won't necessarily help. Categories could help to some extent, although they're not an ideal tool and maintaining them can become complex (see below). And there doesn't exist an artificial intelligence tool that could help with this task either.

Semantic MediaWiki enables wikis to make their knowledge computer-processable, so that you can find and display the answer to this question - and to many more.

---

Source: [https://www.semantic-mediawiki.org/wiki/Help:Introduction\\_to\\_Semantic\\_MediaWiki](https://www.semantic-mediawiki.org/wiki/Help:Introduction_to_Semantic_MediaWiki)

# Semantic MediaWiki ist ein offenes, kollaboratives Wissensmanagementsystem

## Vorteile

- Jede angemeldete Nutzer\_in kann Content erstellen (*anyone can edit*)
- Einfach zu erlernen und zu nutzen (*easy to learn*)
- Unmittelbare Contenterstellung und -nutzung (*instant publish*)
- Kollaborative Ontologie- und Inhaltserstellung (*collaboration*)
- Änderungsverfolgung (*versioning and tracking*)
- Unterstützt den Aufbau von Communities (*community building*)
- Beinhaltet eine semantische Wissensbasis (*knowledge base*)
- Agile Entwicklung (*agile development*)

## Nachteile

- Anyone can edit
- Open to spam 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!

# Grundlegende Definitionen (1/3)

**Definition** Ein **Datenmodell** ist ein Modell der zu beschreibenden und verarbeitenden Daten eines Anwendungsbereichs und ihrer Beziehungen zueinander.

Quelle: <https://wirtschaftslexikon.gabler.de/definition/datenmodell-28093/version-251730>

- Im Kontext der **formalen Abbildung eines Gegenstandsbereichs** (engl. Universe of Discourse) repräsentiert das Datenmodell eine konkrete Ausprägung des Gegenstandsbereichs. ~> Das Datenmodell ist also eine konkrete Instanz des Gegenstandsbereichs
- Im Kontext der **semantischen Wissensmodellierung** repräsentiert das Datenmodell das **assertionale Wissen** eines Wissensgraphen bzw. einer Wissensbasis.

## Wichtig

Das **Datenmodell** bezeichnet eine konkrete Ausprägung eines Ausschnitts der Wirklichkeit, welcher mit einem zugrunde liegenden **Datenstrukturmodell** beschrieben ist. Im Kontext semantischer Wissensmodellierung ist dies idR ein **gerichteter Graph**.

# Grundlegende Definitionen (2/3)

**Definition** **Datenmodellierung** bezeichnet ein Verfahren zur formalen Abbildung der in einem definierten Kontext relevanten **Entitäten** (Dinge) sowie ihrer Eigenschaften und Beziehungen. Hauptziel ist die eindeutige Definition und Spezifikation der in einem Informationssystem zu verwaltenden Elemente, ihrer für die Informationszwecke erforderlichen Attribute und der Zusammenhänge.

Quelle: Angelehnt an <https://de.wikipedia.org/wiki/Datenmodellierung>

Im Kontext semantischer Wissensgraphen spricht man anstelle von Datenmodellierung eher von **Wissensmodellierung**.

**Definition** Der Begriff **Entität** stammt aus der Philosophie und bezeichnet etwas, das existiert – ein Seiendes, einen konkreten oder abstrakten Gegenstand. In diesem Sinn wird der Begriff der Entität in der Regel als Sammelbegriff verwendet, um so unterschiedliche Gegenstände wie Dinge, Eigenschaften, Relationen, Sachverhalte oder Ereignisse auf einmal anzusprechen.

Quelle: Übernommen und angepasst von <https://de.wikipedia.org/wiki/Entität>

In semantischen Wissensgraphen werden **Entitäten** als **dereferenzierbare Ressourcen** mit einer eindeutigen **URI** repräsentiert.

Entität ist hierbei der allgemeine Oberbegriff und bezeichnet sowohl **ABox**- als auch **TBox-Elemente**.

Eine Entität ist damit mit einem **Term** eines **kontrollierten Vokabulars** bzw. einer **Ontologie** gleichzusetzen.

# Grundlegende Definitionen (3/3)

**Definition Wissensrepräsentation** (engl. Knowledge Representation) dient im Rahmen der Wissensmodellierung dazu, mit Hilfe von formalen Sprachen und Repräsentationsframeworks Wissen in Wissensbasierten Systemen formal abzubilden. Eine Sammlung auf diese Weise repräsentierten Wissens wird als **Wissensbasis** (engl. Knowledge Base) bezeichnet.

Quelle: Eigene Definition angelehnt an <https://de.wikipedia.org/wiki/Wissensrepräsentation>

- Semantic MediaWiki liegt das vom **W3C** standardisierte **Resource Description Framework (RDF)** zugrunde.
- Mittels **RDF** lässt sich sowohl assertionales als auch terminologisches Wissen in einem semantischen Wissensgraph abbilden.
- Daten aus Semantic MediaWiki werden in einem solchen **RDF-Graphen** exportiert und mittels einer **OWL-Ontologie** beschrieben.

**Definition Wissensmodellierung** (engl. Knowledge Engineering) dient der Abbildung von Wissen in wissensbasierten Systemen. Sie ist Teilgebiet des Wissensmanagements und der Künstlichen Intelligenz.

Quelle: Auszug übernommen aus <https://de.wikipedia.org/wiki/Wissensmodellierung>

- Die Wissensmodellierung wird idR mittels **formaler, logik-basierter Sprachen** durchgeführt
- Ziel ist das kodierte Wissen "berechenbar" zu machen, d.h., Widersprüche oder Schlussfolgerungen automatisch zu **inferrieren**.
  - d.h. zusätzliches "Wissen" automatisch aus bestehendem Wissen abzuleiten

Source: <http://groups.csail.mit.edu/medg/ftp/psz/k-rep.html>

# Datenstrukturelemente

**Definition** Ein **Datenstrukturelement** 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

**Semantic MediaWiki** definiert 8 **Datenstrukturelemente**:

1. **Seite** (engl. Page)
2. **Namensraum** (engl. Namespace)
3. **Kategorie** (engl. Category)
4. **Attribut** (engl. Property)
5. **Datentyp** (engl. Datatype)<sup>1</sup>
6. **Konzept** (engl. Concept)
7. **Vorlage** (engl. Template)<sup>2</sup>
8. **Subobjekt** (engl. Subobject)

<sup>1</sup> Werden zusammen mit Punkt 4 behandelt

<sup>2</sup> Templates sind kein Datenstrukturelement im eigentlichen Sinne aber ein mächtiges und sinnvolles Tool zum Datenmanagement in MediaWiki

# **Datenmodellierungselemente im Detail**

- Seiten und Namensräume**

# Page

**Definition** A **page** in the context of Semantic MediaWiki corresponds to a single, uniquely identifiable resource, accessible via a URI. A page serves as container for encoding (semantic) information in a Wiki system. It is used to encode both **assertional (ABox)** and **terminological (TBox) knowledge**.

Source: Individual Definition

- Each page belongs to a specific and defined **namespace**
  - pages with no explicitly given namespace belong to the "main" namespace per default
  - the namespace determines the semantics of the page's content, i.e., how its contents is processed and interpreted
- **Pages** are used to
  - describe elements from a universe of discourse (the ABox) as well as
  - elements (vocabulary terms) used to create a data model for representing elements from the universe of discourse (the TBox)
- Semantic MediaWiki defines a **syntax** similar to markdown for encoding page content

# 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-existent 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<sup>1</sup>.

Quelle: eigene Definition

- Every wiki page belongs to **one specific namespace**
  - Namespaces become part of the page's title, e.g., `Help:Namespaces/de`<sup>3</sup>.
  - When **no namespace** is given during page creation, the page will be created in the **main namespace**<sup>2</sup>.
- 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

<sup>1</sup> MediaWiki provides 18 default namespaces. See [https://www.mediawiki.org/wiki/Help:Namespaces/en#Standard\\_namespaces](https://www.mediawiki.org/wiki/Help:Namespaces/en#Standard_namespaces) for a list of standard namespaces provided by MediaWiki

<sup>2</sup> The main namespace is not displayed in a page's URL but used internally by the mediawiki engine.

<sup>3</sup> Please note how subpages are used to implement [multilanguage support](#) on the official MediaWiki help pages.

# Namespaces – Zusammenfassung

- Namensräume dienen der **Gruppierung** von logisch zusammengehörigen Seiten
- **Funktionen** können auf bestimmte Namensräume eingeschränkt werden
- Mediawiki hat standardmäßig **18 Namensräume**
  - Neue Namensräume können hinzugefügt werden
- Textseiten stehen standardmäßig im Namenraum **Main**
  - wird nicht separat in der URL angezeigt
- Der Namensraum definiert, wie der Inhalt einer Seite verarbeitet wird, d.h., die Semantik einer Seite

# **Datenmodellierungselemente im Detail**

## **– Kategorien**

# 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\\_](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-Inheritance, Cycles, ...

## c) Linking to a Category

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

---

<sup>1</sup> 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<sup>1</sup>

```
[[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"



# **Datenmodellierungselemente im Detail**

## **– Beziehungen**

# 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

- 1.) a wikipage to one or many **wikipages**
- 2.) a wikipage to a **datatype value**
- 3.) 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 definition 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**<sup>1</sup> 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](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.

<sup>1</sup> Semantic MediaWiki provides 61 special properties; here is a list of them: [https://www.semantic-mediawiki.org/wiki/Help:Special\\_properties](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 annotations about its **type** and **allowed values**.
  - `[[Has type::{some type}]]` sets the **type** a property's values adhere to
  - `[[Allowed value::{value}]]` allows to **restrict** possible values; it can be entered multiple times on the declaration page

## 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](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)

<sup>1</sup> More information can be found here: [https://www.semantic-mediawiki.org/wiki/Help:Property\\_naming](https://www.semantic-mediawiki.org/wiki/Help:Property_naming)

<sup>2</sup> "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

- A Property needs to be **declared** in the `Property` namespace
- Consider **property naming** recommendations
- Use appropriate **datatype** depending on the envisioned object value<sup>1</sup>

## B) Creating Annotations

- Use it in **in-text annotations**
- Alternatively: **silent annotations** using the `#set` parser function

---

<sup>1</sup> List of supported datatypes: [https://www.semantic-mediawiki.org/wiki/Help/List\\_of\\_datatypes](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
}}
```

Equivalent Property Notation:

```
[[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](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](https://www.semantic-mediawiki.org/wiki/Help:Setting_values)



# **Datenmodellierungselemente im Detail**

## **– Konzepte**

# 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

## Application Scenarios

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

# Function

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

## 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

Source: Example taken from <https://www.semantic-mediawiki.org/wiki/Help:Concepts>

## 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  
}}
```

# **Datenmodellierungselemente im Detail**

## **– Vorlagen**

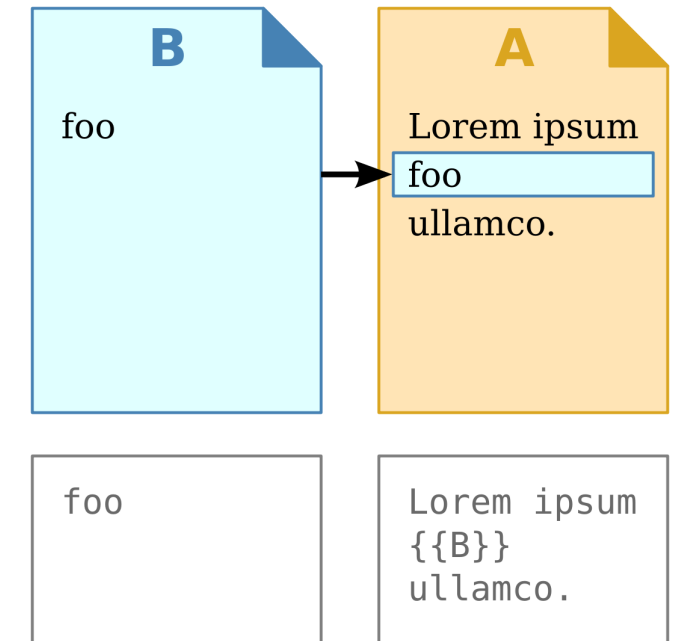
# Templates

**Definition** A **template** is a wiki page defined in the `Template:` namespace that contains **transcludable content** which can be used in other wiki pages. Combined with properties and page forms, it is a very powerful feature to minimize **semantic drift** in a wiki system.

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
- ... can be used in any place in a page



# 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 included in distinct locations during template invocation.

Quelle: Eigene Definition angelehnt an

Transcluded content can be controlled by **three** distinct **commands**<sup>1</sup>

- `<includeonly>` – the text will only be used when the page is transcluded onto another page; it will not appear on the page itself.
- `<onlyinclude>` – the only text should be transcluded onto another page, but will appear on the page itself
- `<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.

<sup>1</sup> see <https://www.mediawiki.org/wiki/Transclusion> for more information about transclusion



# 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**<sup>1</sup> 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**

---

<sup>1</sup> 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**<sup>1</sup> 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}}}`

Syntax:

=====

`{{{Parameter_name}}}`      OR      `{{{Parameter_number}}}`

Example:

=====

`{{{project}}}`      OR      `{{{1}}}`

<sup>1</sup> 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)

# Creating Templates

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

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

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

## Usage

In a wiki page – **without parameters** specified

```
{{Template_name}}
```

**With parameters** on the page where the template is used

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

# 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>
```

# **Next Topic: 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 encode **machine-processable semantics** that can be used in knowledge graphs

...those machine-processable semantics are defined by the **ontology language** and the KRF used to define an ontology

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

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

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

...RDF can also be used to encode terminological knowledge to form a **vocabulary** (=ontology) that can be used in **knowledge graphs**

...An RDF-graph is a set of **resolvable IRIs** structured in form of triples that can be **serialized** using different formats (e.g. **n3**, **n-triples**)

# How do we create Knowledge Graphs using Semantic MediaWiki ?

In the next lecture, we will learn how we can use SMW's knowledge representation framework to encode information as **knowledge graphs**