

Semantic MediaWiki quick reference (1 of 2)

Covers MediaWiki extensions Semantic MediaWiki 1.4, Semantic Result Formats 1.4, Semantic Internal Objects 0.1, Semantic Compound Queries 0.2, Semantic Forms 1.8 and others

Properties

Special pages: Types, CreateProperty, Properties, UnusedProperties, Browse, SearchByProperty

Defining a triple:

```
[[property-name::value]]
{{#set:property-name=value}} (silent definition)
```

Defining a property:

```
[[Has type::type-name]]
[[Allows value::enum-value]] (for enumerations)
```

Property types: Page (default), String, Text, Code, Number, URL, Email, Date, Boolean, Geographical coordinate, Temperature

Defining an internal object:

```
{{#set_internal:object-to-page-property|prop1=val1|prop2=val2|...}}
```

Example:

```
{{#set_internal:Is president of|Has name=George Washington|Has start year=1789|Has end year=1797}}
```

Defining a recurring event:

```
{{#set_recurring_event:property=date property|start=start date|end=end date|period=number|unit={day|week|month|year}|include=additional dates|exclude=excluded dates}}
```

Notable settings for LocalSettings.php

```
$smwgLinksInValues - set to true to allow wiki-links within property values
$smwgNamespacesWithSemanticLinks - all namespaces that can hold semantic values
$smwgShowFactbox - set to SMW_FACTBOX_NONEMPTY to display factbox at bottom of pages
$smwgInlineErrors - set to false to hide SMW error messages
```

Semantic templates

Special pages: CreateTemplate, Templates

Simple semantic template definition:

```
Field-label: [[property-name::{{{field-name}}}|[[Category:category-name]]]]
```

Alternate declaration of property in template:

```
{{#declare:property-name=field-name}}
```

Setting a field to represent multiple property values:

```
{{#arraymap:value|delimiter|var|formula|new_delimiter}}
{{#arraymaptemplate:value|template-name|delimiter|new_delimiter}}
{{#declare:property-name=field_name#list}} (silent definition)
```

Inline queries

Special pages: Ask

Syntax: `{{#ask:parameter1|parameter2|...}}`

Standard parameters:

<code>query conditions</code>	<code>order={asc desc random}</code>	<code>intro=</code>
<code>?property-name=label</code>	<code>headers={show hide plain}</code>	<code>outro=</code>
<code>limit=</code>	<code>mainlabel=</code>	<code>searchlabel=</code>
<code>offset=</code>	<code>link={none subject all}</code>	<code>format=</code>
<code>sort=</code>	<code>default=</code>	

Query conditions: `[[Category:category-name]]... [[Concept:concept-name]]... [[property-name::value]]...`

Subquery: `[[prop1.prop2.prop3::value]]`

Inverse query: `[-property-name::page-name]]`

Comparators: `::` (equals), `::<` (less than), `::>` (greater than), `::~` (like), `::!` (not)

Format types:

standard: list, ol, ul, table, broadtable, category, rss, csv, json, embedded, template, count, debug; *from Semantic Result Formats:* timeline, eventline, calendar, graph, process, googlebar, googlepie, exhibit, outline, sum, average, min, max, bibtex, vcard, icalendar; *from Semantic Maps:* map, googlemaps, yahoo maps, openlayers; *from Semantic Gallery:* gallery

Notes on formats:

- list, ol, ul, category, template and calendar formats allow the 'template' parameter
- sum, average, min and max require 'limit' to be greater than number of queried pages
- for calendar, use #calendarstartdate and #calendarenddate to limit query to current month

Sample query: To create a table of all cities in Europe and their populations, sorted by population:

```
{{#ask:mainlabel=City|[[Category:Cities]]|[[Has country.Has continent::Europe]]?Has population=Pop.|sort=Has population}}
```

Running a compound query:

```
{{#compound_query:query1|query2|...|joint parameters}}
Each query's parameters should be separated by semicolons.
```

Sample compound query: to show a map with one icon for clothing stores and another for fast food restaurants:

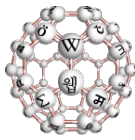
```
{{#compound_query:|[[Category:Clothing stores]]?Has coordinates;icon=Shirt.png|[[Category:Restaurants]]|[[Has cuisine::Fast food]]?Has coordinates;icon=Hamburger.png|format=googlemaps|height=400|width=600}}
```

Notable settings for LocalSettings.php

```
$smwgQComparators - set to '<|>|!~' to allow '~' operator
$smwgQDefaultLimit - default number of results displayed on a page; default is 50
$smwgQMaxInlineLimit - maximum results displayed on a page; default is 500
$smwgQMaxLimit - maximum results retrieved for a query; default is 10000
```

Concepts

Defining a concept: In the 'Concept' namespace:
`{{#concept:query-conditions|description}}`



Semantic MediaWiki quick reference (2 of 2)

Covers MediaWiki extensions Semantic MediaWiki 1.4, Semantic Forms 1.8, Semantic Drilldown 0.7, External Data 0.9 and others

Forms

Special pages: CreateForm, Forms, CreateCategory, CreateClass, RunQuery

Simple form definition:

```

{{{for template|template-name}}
field-label: {{{field|field-name}}}
{{end template}}
Free text:
{{{standard input|free text}}}
{{{standard input|save}}}

```

Form tags:

info field
for template choosers
end template standard input

'info' parameters:

add title= partial form
edit title= page name=
includeonly free text

'for template' parameters:

label= chooser=
multiple chooser caption=
strict

'field' parameters:

input type= hidden
size= mandatory
rows= restricted
cols= uploadable
class= values=
maxlength= default=
list preload=
delimiter= property=
default filename=
values from category=
values from concept=
autocomplete
no autocomplete
autocomplete on property=
autocomplete on category=
autocomplete on concept=
autocomplete on namespace=
autocomplete from url=
remote autocompletion

'standard input' types:

free text preview
minor edit changes
watch cancel
summary run query
save

'standard input' parameters:

label= class=

Input types

Field type	Default input type	Other allowed input types
Page, String, Number, URL, Email, etc.	text	textarea, category
Text, Code	textarea	text
Date	date	
Enumeration	dropdown	radiobutton
Boolean	checkbox	
List of Page, String, etc.	text	textarea, categories
List of Enumerations	checkboxes	listbox

More input types: *from Semantic Maps:* googlemaps, yahoo maps, openlayers;
from Semantic Forms Inputs: combobox, datepicker, regexp

Form input:

```

{{{#forminput:form-name|size|value|button_text|
query_string}}}

```

Form input query string options:

template_name[field_name]=
namespace= super_page=

Setting the 'edit with form' tab:

(in a namespace page (*project-namespace: namespace-name*) or category page)
[[Has default form::form-name]]
(in a regular or template page)
[[Page has default form::form-name]]

Pointing red links to forms:

(in a property page)
[[Has default form::form-name]]
[[Has alternate form::form-name]]
(in a namespace page)
[[Has default form::form-name]]

Making red links create pages automatically:

(in a property page)
[[Creates pages with form::form-name]]

Forms that set an automatic page name

In form definition:

```

{{{info|page name=...<template-name[field-
name]> ... <unique number>}}}

```

Linking to the form:

```

{{{#formlink:form-name|link_text|link_type|
query_string}}}

```

Notable settings for LocalSettings.php

\$sfgRenameEditTabs - set true to change 'edit with form' to 'edit', 'edit' to 'edit source'
\$sfgRenameMainEditTab - set to true to change 'edit' to 'edit source'
\$wgAmericanDates - set to true to show month first in date input
\$sfg24HourTime - set to true to display time input in 24-hour notation

Drilldown

Special pages: CreateFilter, Filters, BrowseData

Defining a filter:

```

[[Covers property::property-name]]
[[Has value::value]]
[[Use time period::Year|Month]]
[[Gets values from category::category-name]]
[[Has input type::{combobox|date range}]]
[[Requires filter::Filter:filter-name]]
[[Has label::label]]

```

In a category page:

```

[[Has filter::Filter:filter-name]]
[[Has display parameters::parameters]]
'parameters' should have same structure as #ask
query parameters, but separated by semicolons
__HIDEFROMDRILLDOWN__
__SHOWINDRILLDOWN__

```

Notable settings for LocalSettings.php

\$sdgFiltersSmallestFontSize, \$sdgFiltersLargestFontSize - font sizes for "tag cloud" display of filter values

External data

Getting data from a URL:

```

{{{#get_external_data:URL|format|external-variable-
name==filter-value|...|local-variable-name=external-
variable-name|...}}

```

Getting data from a database, LDAP server:

```

{{{#get_db_data:server=server_id|from=from-clause|
where=where-clause|data=data-mappings}}
{{{#get_ldap_data:domain=domain-id|filter=ldap-filter|
data=data-mappings}}

```

For #get_db_data and #get_ldap_data, data mappings should be separated by commas.

Displaying external data:

```

{{{#external_value:variable_name}}
{{{#for_external_table:expression}}
The expression value in #for_external_table should
contain variables in the form {{{variable_name}}}.

```

Storing external data - examples:

```

Storing an external value with SMW:
The capital of {{{PAGENAME}}} is [[Has capital:
{{{#external_value:capital}}]].

```

Storing (and displaying) a table of external values:

```

{{{#for_external_table:{{{#set_internal:Is leader of|Has
name= {{{name}}}|Has start year= {{{start year}}}|Has
end year= {{{end year}}}} {{{name}}}} {{{start year}}},
{{{end year}}}}. }}

```

Notable settings for LocalSettings.php

\$edgStringReplacements - array for hiding API keys
\$edgAllowExternalDataFrom - array for an API
"whitelist" - necessary if API keys are being hidden
\$edgCacheTable - database table for caching data
DB and LDAP access have other required settings.