



# Semantic MediaWiki quick reference (1 of 3)

Covers MediaWiki 1.19 and MediaWiki extensions ParserFunctions 1.4, Semantic MediaWiki 1.8, Semantic Result Formats 1.8, Semantic Maps 2.0, Semantic Internal Objects 0.6, Semantic Compound Queries 0.3 and Semantic Forms 2.5

## MediaWiki syntax

"italics"  
'''bold'''  
''''bold and italics''''  
==standard header==  
===next-level header=== (...and so on)  
[[Internal link]]  
[[Internal link|alternate text]]  
[http://example.com text of external link]  
[[Category:Example]] (category tag)  
[:Category:Example]] (link to category)  
---- (horizontal line)  
\* bulleted item  
# numbered item  
:indentation  
::double indentation (...and so on)  
;term : definition  
[[File:Image-name.jpg|thumb|frame|Caption text]]  
{{Template name}} (call to template)  
#REDIRECT [[Page name]]

## ParserFunctions

Main functions: #expr, #if, #ifeq, #iferror, #ifexpr, #ifexist, #rel2abs, #switch, #time, #timel, #titleparts; from StringFunctions: #len, #pos, #rpos, #sub, #pad, #replace, #explode, #urlencode, #urlencode

## Inline queries

**Special pages:** Ask

**Syntax:** {{#ask:parameter1|parameter2|...}}, {{#show:page-name|parameter1|...}}

**Standard parameters:**

query conditions limit=  
?property-name=label offset=  
|+property-param=value default=  
sort= intro=  
order={asc|desc|random} outro=  
headers={show|hide|plain} searchlabel=  
mainlabel= format=  
link={none|subject|all}

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

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

**Inverse property:** [-property-name::page-name]]

**"Or" conditions:** [[Category:category-name]] OR [[property-name::value]],  
[[property-name:value1||value2]], [[page-name1||page-name2||page-name3]]

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

## 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 enums)

**Property types:** Page (default), String, Text, Code, Number, Date, Boolean, URL, Email, Telephone number, Temperature; from Semantic Maps: Geographic coordinate

**Defining an internal object:**  
{{#set\_internal:object-to-page-property|prop1=val1|prop2#list=val2|...}}

Example (on a page called "United States"):

```
{{#set_internal:Is president of|Has name=James Madison|Has start year=1801|Has end year=1809|Has vice president#list=George Clinton, Elbridge Gerry}}
```

**Property parameter types:** limit, order, align, index

**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}}
```

### Notable settings for LocalSettings.php

\$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}}

**Defining a subobject:**  
{{#subobject:-|prop1=val1|...|main-prop={{PAGENAME}}}}

**Defining a recurring event:**  
{{#set\_recurring\_event:property=date|property|start=start date|end=end date|period=number|unit={day|week|month|year}|week number=number|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

## 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}}
```

## Semantic Compound Queries

**Running a compound query:**  
{{#compound\_query:query1|query2|...|joint parameters}}  
Each query's parameters should be separated by semicolons, instead of pipes.

**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}}
```



# Semantic MediaWiki quick reference (2 of 3)

Covers MediaWiki extensions Semantic MediaWiki 1.8, Semantic Result Formats 1.8, Semantic Maps 2.0 and Semantic Drilldown 1.1

## SMW result formats

### Standard display formats:

**list** (default format if no printouts) - delimited list of values

**ol** - numbered vertical list

**ul** - bulleted vertical list

**table** - table of values, with property names as column headers

**broadtable** (default format if one or more printouts) - like table, but takes full width of page

**category** - display in the manner of MediaWiki category pages

**template** - like list, but "template" parameter required

*list, ol, ul, category and template formats allow the "sep", "template", "named args" and "userparam" parameters.*

*ol, ul and category allow the "columns" parameter.*

### Export formats:

**csv** - CSV (comma-separated values) format

**dsv** - DSV (delimiter-separated values) format

**json** - JSON (JavaScript-friendly) format

**rss** - RSS (syndication) format

**rdf** - RDF (Semantic Web) format

### Other formats:

**embedded** - full pages displayed

**count** - shows only number of results

**debug** - shows debugging information

## Semantic 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::{combo box|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

## Semantic Result Formats

### Date formats:

**timeline** - scrolling JavaScript-based timeline

**eventline** - like timeline, but allows multiple points per event

**calendar** - monthly calendar

*Use #calendarstartdate and #calendarenddate to limit results to current month*

**eventcalendar** - like calendar, but uses FullCalendar JS library

**timeseries** - aggregates pages by date, using the flot JS library

### Graphing formats:

**graph** - graph of pages as nodes, using the VizGraph application

**process** - workflow graphs, using VizGraph

### Charting formats:

**googlebar** - bar chart, using the Google API

**googlepie** - pie chart, using the Google API

**jqplotbar** - bar chart, using the jqPlot JavaScript library

**jqplotpie** - pie chart, using jqPlot

**jqplotchart** - various chart types, using jqPlot

**jqplotseries** - like jqplotchart, for multiple sets of data

*Relevant param: charttype (can be bar, line, donut, bubble or scatter)*

**d3chart** - various chart types, using the D3 JS library

*Relevant param: charttype (can be bubble or treemap)*

**sparkline** - small, inline charts, using the jquery.sparkline JS library

*Relevant param: charttype (can be bar, line, pie or discrete)*

**dygraphs** - charts of non-semantic raw data, with semantic annotations

*Relevant param: datasource*

*jqPlot, D3 and dygraphs formats allow the "height", "width", "charttitle" and "charttext" params, among others.*

*jqPlot, D3 and sparklines allow "distribution".*

### Page display formats:

**gallery** - display of images in a gallery or slideshow

**listwidget** - results separated into pages based on alphabetical and other sorting

*Relevant params: widget (can be alphabet, menu or pagination), pageitems*

**pagewidget** - scrolling through result pages

### Math formats:

**sum, average, min, max, median, product** - operations on numbers

*For all these formats, "limit" parameter must be greater than number of queried pages, for accurate results.*

### Other display formats:

**outline** - hierarchical display of properties

*Relevant param: outlineproperties*

**tree** - shows pages connected via a property

**oltree** - like tree, but a bulleted list

**tagcloud** - tag cloud display, based on frequency of values

**valuerank** - like tagcloud, but shows # of instances, without changing font size

**filtered** - JavaScript-based property drilldown

**array** - like list, but allows more delimiters

**hash** - like array, but meant to be used by HashTables extension

**incoming** - shows all incoming properties to each page

### Export formats:

**bibtex** - BibTeX bibliographic format

**icalendar** - iCalendar event information format

**vcard** - vCard personal information format

**feed** - both the RSS and Atom syndication types

## Maps

### Displaying a map:

```
{{#display_map:points=|center=|service=|geoservice=|width=|height=|zoom=|icon=|lines=|polygons=}}
```

Format for "points" parameter:

```
point 1~title 1~caption 1~marker 1;point 2~title 2~caption 2~marker 2
```

<display\_map> - takes same parameters, except for "points", which is passed in as tag value. Format for points value:

```
point 1|title 1|caption 1|marker 1
point 2|title 2|caption 2|marker 2
```

### Geocoding (displays coordinates):

```
{{#geocode:location=string or coordinates|service=|format=|directional=}}
```

## Semantic Maps

### Standard formats:

**googlemaps** - maps, using Google Maps service

**openlayers** - maps, using OpenLayers service

**map** - maps, using the default service specified

*Notable params: center, geoservice, width, height, zoom, icon, lines, polygon*

### Export formats:

**kml** - KML (Key Markup Language)



# Semantic MediaWiki quick reference (3 of 3)

Covers MediaWiki extensions Semantic Forms 2.5 and External Data 1.3

## Semantic Forms

**Special pages:** CreateForm, Forms, 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 standard input  
end template

**'info' parameters:**

create title= partial form  
edit title= page name=  
query title= query form on top

**'for template' parameters:**

label= strict  
multiple add button text=  
embed in field=

**'field' parameters:**

input type= hidden  
class= mandatory  
list restricted  
delimiter= default=  
property= preload=  
holds template

*...for text inputs:*

size=, uploadable, image preview,  
default filename=

*...for textarea inputs:*

autogrow, rows=, cols=, editor=

*...for 'category' and 'categories':*

top category=, height=, width=

*...for enums and autocomplete:*

values=, values from {property|  
category|concept|namespace}=

*...for enums:* show on

select=value1=>div-  
ID1;value2=>div-ID2

*...for autocomplete:* remote

autocomplete, values dependent  
on=, values from url=

**'standard input' types:**

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

**'standard input' parameters:**

label= class=

### Input types

Field type	Allowed input types (default is bolded)
Page	<b>text with autocomplete</b> , combobox, text
String, URL, Number, etc.	<b>text</b> , text with autocomplete, combobox
Text, Code	<b>textarea</b> , text
Date	<b>date</b> , datetime, year
Enumeration	<b>dropdown</b> , radiobutton
Boolean	<b>checkbox</b>
List of Page	<b>text with autocomplete</b> , text, textarea (with autocomplete)
List of String, etc.	<b>text</b> , text with autocomplete, textarea (with autocomplete)
List of enums	<b>checkboxes</b> , listbox

More input types: *from Semantic Maps:* map,  
googlemaps, openlayers;  
*from Semantic Forms Inputs:* datepicker,  
timepicker, datetimetype, menuselect, regexp;  
*from Semantic Image Input:* instantimage;  
*using CategoryTree:* category, categories

**Form input:**

```

{{{#forminput:form=|size=|default value=|button
text=query string=|popup|autocomplete on
category=|autocomplete on namespace=
remote autocomplete|query string params}}}
```

**Form input query string options:**

(values are separated by '&' if they're part of  
"query string=", or '|' if they're regular params)  
*template\_name*[*field\_name*]=  
namespace= super\_page=

**Setting the 'edit with form' tab:**

(in a namespace page (Project:*namespace-  
name*) or category page)  
[[Has default form::*form-name*]]  
(in a regular or template page)  
[[Page has default 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=|link text=|link type=|query
string=|target=|popup|query string params}}}
```

**Linking to Special:RunQuery:**

```

{{{#queryformlink:form=|link text=|link type=|query
string=|popup|query string params}}}
```

**Pointing red links to forms:**

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

**Making red links create pages automatically:**

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

**A link to edit a page automatically:**

```

{{{#autoedit:form=|target=|link text=|link type=|query
string=|reload}}}
```

### 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 'edt source'  
\$wgAmericanDates - set to true to show month  
first in date input  
\$sfg24HourTime - set to true to display time input  
in 24-hour notation

## External Data

**Getting data from a URL:**

```

{{{#get_web_data:url=URL|format=format|data=data-
mappings|filters=filters}}}
```

*Allowed formats: CSV, JSON, XML, GFF*

**Getting data from a database:**

```

{{{#get_db_data:server=server_id|from=from-clause|
where=where-clause|data=data-mappings}}}
```

**Getting data from an LDAP server:**

```

{{{#get_ldap_data:domain=domain-id|filter=ldap-filter|
data=data-mappings}}}
```

*"data", "filters" values should be comma-separated.*

**Displaying external data:**

```

{{{#external_value:variable_name}}}  
{{{#for_external_table:expression}}}  
The expression value in #for_external_table should  
contain variables of 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 a table of external values (requires Semantic  
Internal Objects extension):*

```

{{{#store_external_table:Is leader of|Has name=
{{{name}}}|Has start year= {{{start year}}}|Has end  
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.*