Plugin Web Service

Introduced in Tiki3 and improved in Tiki22

Use this wiki plugin to pull data from any JSON or SOAP enabled web service, e.g. those from Yahoo, or opendata portals based on CKAN.

PARAMETERS

Display remote information exposed in JSON or YAML or SOAP XML
Introduced in Tiki 3.
Go to the source code
Preferences required: wikiplugin_webservice

Parameters	Accepted Values

(body of plugin)

Description

Template to apply to the data provided. Template format uses smarty templating engine using double brackets as delimiter. Output must provide wiki syntax. Body can be sent to a parameter instead by using the `bodyname` parameter.

Default	Since

`url`

`params`

`service`

`template`

`bodyname` word

Complete service URL

Parameters formatted like a query: `param1=value1&amp;param2=value2`

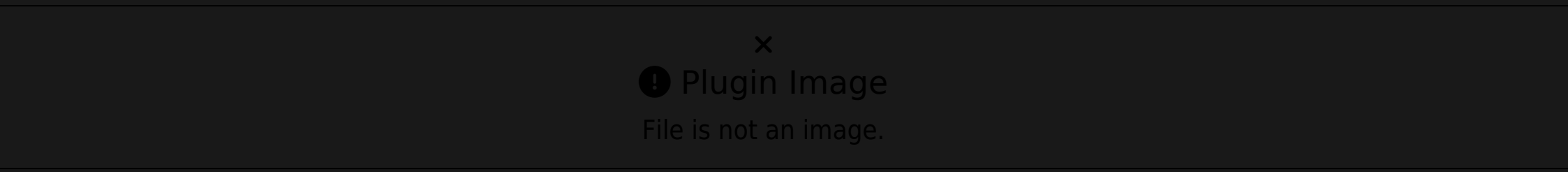
Registered service name.

For use with registered services, name of the template to be used to display the service output. This parameter will be ignored if a body is provided.

Name of the argument to send the body as for services with complex input. Named service required for this to be useful.


3.0
7.0
3.0
3.0
3.0

EXAMPLES



The result is you can embed a webservice-specific plugin tag, with one named parameter

And even change the output format by defining the format in a block after the tag:

1.1. WHY

This powerful plug-in allows you to bring external information in to Tiki easily. There are many contexts where existing legacy corporate systems or remote web sites contain useful information. This plug-in will allow you to display the information from these systems inside Tiki. It will allow administrators to set up information sources as easy and safe to use pre-configured wiki page plug ins.

Some good examples of this are:

- Including bug tracking information from an external source (such as Bugzilla) in documentation pages.
- Showing the results of images searches in **xxx** where the results are automatically refreshed.
- Displaying data from opendata portals as text, or in maps.

    - You can play with an example at: GeoCMS_Maps_22
- **More examples ...**

For a list of some of the web based information sources available have a look at, **put URL here**.

1.2. HOW

The Webservice plug-in consumes content from remote sites and customizes the way it is displayed.

- The remote site must export the data in either JSON or YAML, or XML using SOAP.
- The data must be publicly available as no authentication methods are currently supported.

There are multiple ways to display the information are supported, with various security and portability concerns.

1. Self-contained, all inline in a page
2. Defined (by an administrator) as a webservice
3. Defined (by an administrator) using a plugin alias.

The web service plugin is disabled by default. Also it is marked as unsafe, which means any use or modification requires validation by an administrator or a moderator with the required privileges.

1.3. OUTPUT TYPES

The output can generate:

- Smarty to generate wiki syntax

- Smarty to generate HTML
- Javascript to generate HTML

The most basic, and safest, way to display is to use Smarty templates to generate wiki syntax but using Smarty to generate HTML affords more flexibility.

1.4. HELLO, WORLD: YAHOO'S MADONNA EXAMPLE

The following is the result set from the Madonna example on http://developer.yahoo.com/javascript/json.html

Yahoo's Madonna Example

{"ResultSet":{ "totalResultsAvailable":"229307", "totalResultsReturned":"2", "firstResultPosition":"1", "Result":[{ "Title":"madonna 116", "Summary":"Picture 116 of 184", "Url":"http:\/\/www.celebritypicturesarchive.com\/pictures\/m\/madonna\/madonna-116.jpg", "ClickUrl":"http:\/\/www.celebritypicturesarchive.com\/pictures\/m\/madonna\/madonna-116.jpg", "RefererUrl":"http:\/\/www.celebritypicturesarchive.com\/pgs\/m\/Madonna\/Madonna%20picture_116.htm", "FileSize":"36990", "FileFormat":"jpeg", "Height":"530", "Width":"425", "Thumbnail":{ "Url":"http:\/\/scd.mm-b1.yimg.com\/image\/481989943", "Height":"125", "Width":"100" } }, { "Title":"madonna 118", "Summary":"Picture 118 of 184", ... } }] } }

If all is well with your installation, the following should display as below it:

{WEBSERVICE(url="http://search.yahooapis.com/ImageSearchService/V1/imageSearch?appid=YahooDemo&query=Madonna&output=json")} || Total Results Available|{{$response.ResultSet.totalResultsAvailable}} Total Results Returned|{{$response.ResultSet.totalResultsReturned}} || {{foreach from=$response.ResultSet.Result item=result}} * Title {{$result.Title}} {{/foreach}} {WEBSERVICE}

Result:

1.5. HELLO WORLD: ANOTHER EXAMPLE

Yahoo's Weather Forecast

{"query":{"count":1,"created":"2017-02-14T20:10:17Z","lang":"en-US","results":{"channel":{"units":{"distance":"mi","pressure":"in","speed":"mph","temperature":"F"},"title":"Yahoo! Weather - Nome, AK, US","link":"http://us.rd.yahoo.com/dailynews/rss/weather/Country__Country/*https://weather.yahoo.com/country/state/city-2460286/","description":"Yahoo! Weather for Nome, AK, US","language":"en-us","lastBuildDate":"Tue, 14 Feb 2017 11:10 AM AKST","ttl":"60","location":{"city":"Nome","country":"United States","region":" AK"},"wind":{"chill":"-15","direction":"23","speed":"25"},"atmosphere":{"humidity":"56","pressure":"982.0","rising":"0","visibility":"16.1"},"astronomy":{"sunrise":"10:2 am","sunset":"6:31 pm"},"image":{"title":"Yahoo! Weather","width":"142","height":"18","link":"http://weather.yahoo.com","url":"http://l.yimg.com/a/i/brand/purplelogo//uh/us/news-wea.gif"},"item":{"title":"Conditions for Nome, AK, US at 10:00 AM AKST","lat":"64.499474","long":"-165.405792","link":"http://us.rd.yahoo.com/dailynews/rss/weather/Country__Country/*https://weather.yahoo.com/country/state/city-2460286/","pubDate":"Tue, 14 Feb 2017 10:00 AM AKST","condition":{"code":"23","date":"Tue, 14 Feb 2017 10:00 AM AKST","temp":"3","text":"Breezy"},"forecast":[{"code":"30","date":"14 Feb 2017","day":"Tue","high":"6","low":"1","text":"Partly Cloudy"},{"code":"28","date":"15 Feb 2017","day":"Wed","high":"1","low":"-5","text":"Mostly Cloudy"},{"code":"30","date":"16 Feb 2017","day":"Thu","high":"-6","low":"-13","text":"Partly Cloudy"},{"code":"34","date":"17 Feb 2017","day":"Fri","high":"-14","low":"-16","text":"Mostly Sunny"},{"code":"28","date":"18 Feb 2017","day":"Sat","high":"-10","low":"-18","text":"Mostly Cloudy"},{"code":"26","date":"19 Feb 2017","day":"Sun","high":"-10","low":"-17","text":"Cloudy"},{"code":"26","date":"20 Feb 2017","day":"Mon","high":"-6","low":"-13","text":"Cloudy"},{"code":"28","date":"21 Feb 2017","day":"Tue","high":"-9","low":"-16","text":"Mostly Cloudy"},{"code":"28","date":"22 Feb 2017","day":"Wed","high":"-8","low":"-12","text":"Mostly Cloudy"},{"code":"24","date":"23 Feb

2017","day":"Thu","high":"36","low":"-13","text":"Windy"}],"description":"<![CDATA[\n
\nCurrent Conditions:\n
Breezy\n
\n
\nForecast:\n
 Tue - Partly Cloudy. High: 6Low: 1\n
 Wed - Mostly Cloudy. High: 1Low: -5\n
 Thu - Partly Cloudy. High: -6Low: -13\n
 Fri - Mostly Sunny. High: -14Low: -16\n
 Sat - Mostly Cloudy. High: -10Low: -18\n
\n
\nFull Forecast at Yahoo! Weather\n
\n
\n(provided by The Weather Channel)\n
\n]]>","guid":{"isPermaLink":"false"}}}}}}

×

Decoding JSON

Use an online JSON editor like this one to understand the structure of the JSON data.

If all is well with your installation, the following should display as below it:

{WEBSERVICE(url="https://query.yahooapis.com/v1/public/yql?q=select%20*%20from%20weather.forecast%20where%20woeid%20in%20(select%20woeid%20from%20geo.places(1)%20where%20text%3D%22nome%2C%20ak%22)&format=json")} {{$response.query.results.channel.item.title}} {FANCYTABLE(head="Date|Day|High|Low")} {{foreach from=$response.query.results.channel.item.forecast item=result}} {{$result.date}}|{{$result.day}}|{{$result.high}}|{{$result.low}} {{/foreach}} {FANCYTABLE} {WEBSERVICE}

THE DATA DISPLAYED LIVE

no longer working

{WEBSERVICE(url="https://query.yahooapis.com/v1/public/yql?q=select%20*%20from%20weather.forecast%20where%20woeid%2

0in%20(select%20woeid%20from%20geo.places(1)%20where%20text%3D%22nome%2C%20ak%22)&format=json")}
{{$response.query.results.channel.item.title}} {FANCYTABLE(head="Date|Day|High|Low")} {{foreach
from=$response.query.results.channel.item.forecast item=result}}
{{$result.date}}|{{$result.day}}|{{$result.high}}|{{$result.low}} {{/foreach}} {FANCYTABLE} {WEBSERVICE}

1.6. FULLER EXAMPLE

Note: this isn't such a good example because it doesn't work out of the box: you'll have to find your own bugzilla instance.

{WEBSERVICE(url=http://bugzilla.example.com/bugexport.pl?id=00002345)} !! {{$response.title}}
||ID|[http://bugzilla.example.com/bug/{{$response.id|escape|'url'}}|{{$response.id}}] Status|{{$response.status}}
Assignee|{{$response.assignee.full_name}} Open Date|{{$response.open_date}} Last Modification|{{$response.last_modif}}
{{foreach from=$response.comments item=comment}} *
[http://bugzilla.example.com/bug/{{$response.id|escape|'url'}}#{{$comment.id}}|{{$comment.title}}] by
''{{$comment.author.full_name}}'' on {{$comment.date}} {{/foreach}} {WEBSERVICE}

1.7. REGISTERING A WEBSERVICE

To use higher capabilities, such as HTML generation, a web service must be registered. A registered web service contains:

- A name
- A URL with parameters
- A type (REST / SOAP)
- A list of available templates

A template contains:

- A name
- The output format (use value `tikiwiki` or `html`)
- The engine used to execute the template (`smarty` or `javascript`)
- The content of the template (this is not a content or smarty template, this template is defined where you register the webservice.)

When the output of Smarty is Tiki syntax, Smarty elements use double curly braces rather than single ones to avoid conflicts with the syntax.

To register web services, visit tiki-admin.php?page=webservices on your tiki installation.(this is also where you define the body of the template used, you cannot reference an allready defined content template)

A registered web service can be then called e.g. from a wiki page using the Webservice plugin:

```
{WEBSERVICE(service=bugzilla,template=complete,id=00002345)/}
```

For registering a web service also see the example on Webservice Registration page.

1.8. BUNDLING THE WEBSERVICE IN A PLUGIN ALIAS

A **Plugin Alias** can be created to hide complexity by making the web service aspects transparent. The end user sees it as a norma plugin but internally, it's only a reserved name. When encountered, the parameters provided to it are forwarded to an other plugin Default values can be provided in the process.

For example, a BUGZILLA plugin could be created to access the bug information. This plugin would:

- Redirect to the WEBSERVICE plugin
- Always use *bugzilla* as the service
- Accept a required *id* parameter which would be passed directly to WEBSERVICE
- Accept an optional *template* parameter, which would provide the required documentation to indicate the supported templates (ex: status|link|short|complete). If not provided, it would use complete as the default value.
- Ignore any user input in the body
- Require no validation because all parameters are safe.

{BUGZILLA(id=00002345)/} {BUGZILLA(id=00002345,template=short)/}

Plugin Alias can be used for other purposes than web services, like shortcut plugins to various tracker lists and item configurations

To register plugin aliases, visit tiki-admin.php?page=plugins on your tiki installation.

1.9. IMPLEMENTATION NOTES

- Other scripting languages or templating engines could be added. However, because Tiki is

already bundled with Smarty, it was a natural choice.

1.10. SECURITY

When including content from remote sites, the primary issue is that to get any kind of special formatting, the content from the remote host must be provided as HTML. One one side, HTML is difficult to manipulate and can hardly be included seamlessly inside the page. Content will always appear to be different. On the other hand, the consumer becomes vulnerable to XSS attacks because unwanted JavaScript may be included in the remote HTML.

The webservice strategy mitigates this risk by applying an age-old concept: separate data from presentation. The web services only request the data from the remote host and nothing of the presentation aspects. The consumer side is responsible to display the information using their preferred display method. In the case of Tiki, Smarty is default option. Before being sent to the template engine, the received data is filtered to avoid XSS coming from the data. The presentation layer is assumed to be audited and safe.

1.11. PRESENTATION SUGGESTION

Even if security has to be insured, the remote host still is likely to know best how the content should be presented. Providers are encouraged to provide suggested templates to render the data. These templates can be audited by the site administrator to be used directly or serve as a sample to build custom templates.

The templates are to be provided as part of the body under the *_template* property.

The webservice registry presents the suggestions proposed by the remote hosts. These suggestions can be registered as valid templates.

1.12. RELATED

- Profile GeoCMS_Maps_22

ALIASES

Plugin Webservice | Plugin Web Services | PluginWebServices