Integration with external sites

amusewiki.org
Optional

The are the same options you can find in the editing page. Use a value of 1 to activate them.

fix_links Make all links active

fix_typography Smart quotes

fix_nbsp remove_nbsp Unconditionally remove all the invisible non-breaking spaces in the text

show_nbsp Make all non-breaking spaces explicit and visible as ~~

fix_footnotes Rearrange the footnotes

curl -F __auth_user=$username -F __auth_pass=$password -F body='<test.muse' -F fix_footnotes=1 -F message='Edit my library via api' - https://staging.amusewiki.org/remote/edit/library/my-library

Beware that some filters, like the footnote filter, may trigger errors which are returned to the caller, e.g.:

```json
{
    "error": {
        "references": 1,
        "references_found": "[2]",
        "footnotes": 0,
        "footnotes_found": "",
        "differences": '@@ -0,0 +1 @@
+[2]
'
    }
}
```
curl -F __auth_user=$username \
-F __auth_pass=$password \
-F body='<test.muse' \
-F message='Edit my special via api' \
https://staging.amusewiki.org/remote/edit/special/my-special

Or:
curl -F __auth_user=$username \
-F __auth_pass=$password \
-F body='<test.muse' \
-F message='Edit my library via api' \
https://staging.amusewiki.org/remote/edit/library/my-library

Response:
{
  "job":"https://staging.amusewiki.org/tasks/status/31864",
  "url":"https://staging.amusewiki.org/special/my-special"
}

or an error:

{"error":"This text cannot be edited"}

As shown above, the endpoint in /remote/edit/ with the full URI of the text appended.

Parameters

Mandatory
__auth_user The username
__auth_pass The password
message A commit message to explain what you did
body The full body in Muse format (in the examples above, curl is reading from a file)

Embedding an existing text

Say the URL of the text you want to embed is https://amusewiki.org/library/bookbuilder-tutorial
you can add an iframe with https://amusewiki.org/library/bookbuilder-tutorial/embed to have the infobox with the downloads.

Example:

<iframe src="https://amusewiki.org/library/bookbuilder-tutorial/embed" width="700px" height="200px">
</iframe>

Non-interactive creation of texts

This procedure lacks any human interaction and it relies completely on the HTML conversion to be good enough. Worst case scenario, you are going to get a text which needs fixing.

You need an existing user on the site.

The API endpoint is /remote/create, e.g. https://staging.amusewiki.org/remote/create.

You need to do a POST request with some mandatory and some optional parameters

Request:

$ curl -F __auth_user=$username \
-F __auth_pass=$password \
-F title=Test \
-F textbody='Hello <em>hello</em>' \
https://staging.amusewiki.org/remote/create

Response is a JSON string, which holds the URL of the text which is going to be created and the job URL with the publishing.

{
  "url":"https://staging.amusewiki.org/library/test",
  "job":"https://staging.amusewiki.org/tasks/status/17097"
}

If you are missing some mandatory parameters you get an error:
curl -F __auth_user=$username \
   -F __auth_pass=$password \
   -F textbody='Hello <em>hello</em>' \
   https://staging.amusewiki.org/remote/create

Response:

{"error":"Missing mandatory title and textbody parameters"}

If you miss the authentication parameters, you get a 401 response.
To create a new special page, append `special` to the endpoint:

curl -F __auth_user=$username \
   -F __auth_pass=$pwd \
   -F title='My special' \
   -F textbody='Hello <em>special</em>' \
   https://staging.amusewiki.org/remote/create/special

Response:

{
   "job": "https://staging.amusewiki.org/tasks/status/31848",
   "url": "https://staging.amusewiki.org/special/my-special"
}

**Parameters**

**Mandatory**
- `__auth_user` The username
- `__auth_pass` The password
- `title` The text title
- `textbody` The HTML body

**Optional**
- `subtitle` Text subtitle
- `author` Text author
- `notes` HTML string with the notes
- `teaser` HTML string with the teaser
- `LISTtitle` Text for indexing
- `SORTauthors` List of authors, comma or semicolon separated list
- `SORTtopics` List of topics, comma or semicolon separated list
- `date` Date of the text
- `uid` Unique identifier (for multilanguage support)
- `cat` Fixed category list (separated by space)
- `slides` Boolean if slides are required (if the site sets that)
- `source` String with the source notes
- `lang` ISO code language (e.g. `en`)
- `pubdate` Publication date

**Non-interactive editing**

Editing pages via API works the same way. However, a word of caution: while using the web interface you get warned if there are pending revisions, via API you unconditionally spawn a revision and publish it. If by chance you have other users editing it, they will get conflicts, have your or their changes overwritten.

---

1 Actually, `/remote/create` is an alias for `/remote/create/library`