PADERBORN UNIVERSITY
The University for the Information Society

Introducing Extension SemanticScribunto

EMWCon Fall 2017

IMT: Zentrum für Informations- und Medientechnologien

Tobias Oetterer
Center for Information and Media Technologies

Topics

- Scribunto
- Data retrieval
- Data storage
- Tooltips
- Query building
- Documentation references

Tobias Oetterer
Center for Information and Media Technologies

A short Introduction to Scribunto and Lua

Scribunto is an extension that allows for the embedding of lua scripts in MediaWiki.

> “[Lua] is a lightweight multi-paradigm programming language designed primarily for embedded systems and clients”.
>
> https://en.wikipedia.org/wiki/Lua_(programming_language)

The introduction of Scribunto addresses performance issues in complex templates, tries to give a tool to write better code, and provides a way to divide display code and logic code. For people coming from a coding background it also offers more familiarity that “coding” in templates.

For further reading, please see [0].

[0]: https://en.wikibooks.org/wiki/Scribunto:_An_Introduction

Tobias Oetterer
Center for Information and Media Technologies

How does Scribunto interface with extensions?

When using Scribunto you have to ways of handling MediaWiki functionalities:

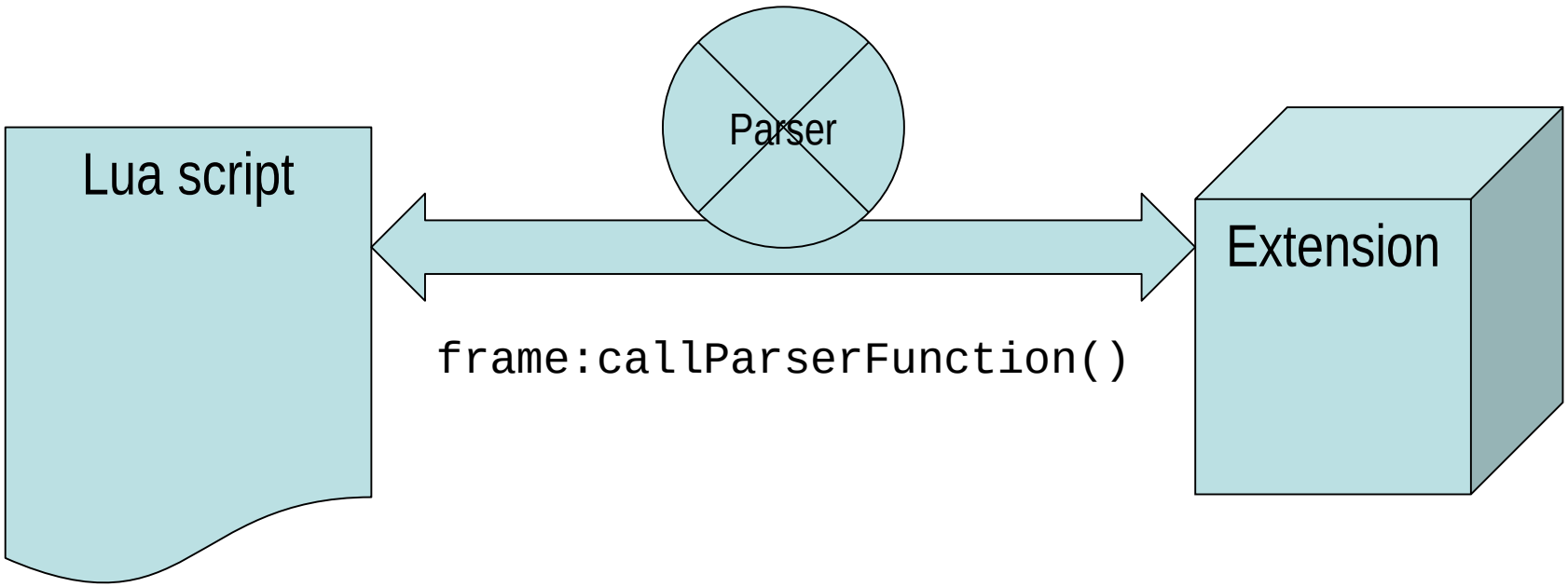
- Native Lua libraries
- Parser function calls

The first option is preferable since it is normally better designed, more performant and easier to integrate. Unfortunately, the libraries have to be provided by the extension developer. In our case we are in luck: SMW has the Extension:SemanticScribunto.

If there is no native support, Scribunto can call any parser function through a frame. However, return values (if present) are always strings.

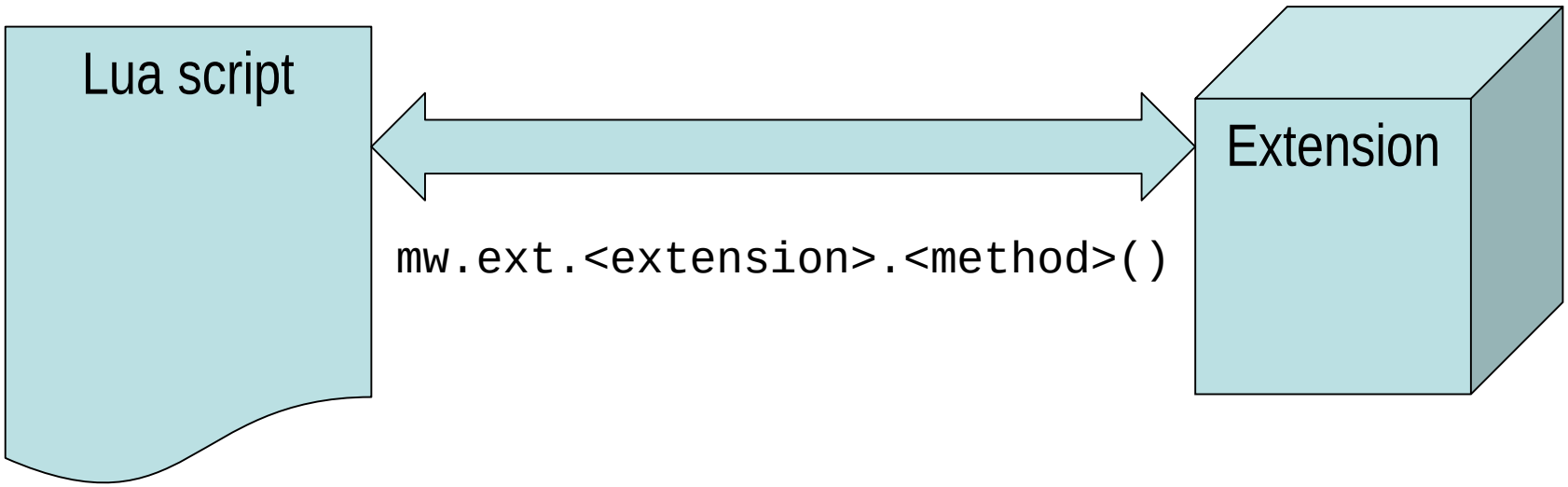
Tobias Oetterer
Center for Information and Media Technologies

How does Scribunto interface with extensions?



Tobias Oetterer
Center for Information and Media Technologies

The preferred way



Tobias Oetterer
Center for Information and Media Technologies

Data retrieval

Tobias Oetterer
Center for Information and Media Technologies

Data retrieval

Three functions are available for data retrieval:

1. mw.smw.getPropertyType
2. mw.smw.ask
3. mw.smw.getQueryResult

Tobias Oetterer
Center for Information and Media Technologies

mw.smw.ask

```
result = mw.smw.ask('[[Category:Comic books]][[Has comic book
writer::Neil Gaiman]]|?#-=page|?Has main character=protagonist|?Has
comic book artist#-|?Creation date|limit=2|mainlabel=-')
```

```
(table)[2]:
        1: (table)[0]:
                Creation date: (string) '14 décembre 2016 08:39:18'
                protagonist: (table)[2]:
                        1: (string) 'Dream (comics)'
                        2: (string) 'Death (DC Comics)'
                page: (string) 'Sandman - Dream Country'
                Has comic book artist: (table)[5]:
                        2: (string) 'Malcolm Jones III'
                        3: (string) 'Steve Oliff'
                        1: (string) 'Kelley Jones'
                        4: (string) 'Charles Vess'
                        5: (string) 'Colleen Doran'
        2: (table)[0]:
                Creation date: (string) '14 décembre 2016 08:30:19'
                protagonist: (table)[4]:
                        1: (string) 'Dream (comics)'
                        2: (string) 'Cain and Abel (comics)'
                        4: (string) 'Lucifer (DC Comics)'
                        3: (string) 'John Constantine'
                page: (string) 'Sandman - Preludes and Nocturnes'
                Has comic book artist: (table)[3]:
                        1: (string) 'Sam Kieth'
                        2: (string) 'Mike Dringenberg'
                        3: (string) 'Malcolm Jones III'
```

Tobias Oetterer
Center for Information and Media Technologies

mw.smw.getQueryResult

```
(table)[0]:
        serializer: (string) 'SMW\Serializers\QueryResultSerializer'
        meta: (table)[0]:
                hash: (string) 'c558d584ba2d286f0a077bc34db2a801'
                count: (number) 2
                time: (string) '0.010596'
                source: (string)
                offset: (number) 0
        printrequests: (table)[4]:
                1: (table)[0]:
                        key: (string)
                        label: (string)
                        format: (boolean) false
                        mode: (number) 2
                        redi: (string)
                        typeid: (string) '_wpg'
                2: (table)[0]:
                        key: (string) 'Has_comic_book_artist'
                        label: (string) 'Has comic book artist'
                        format: (string) '-'
                        mode: (number) 1
                        redi: (string)
                        typeid: (string) '_wpg'
                4: (table)[0]:
                        key: (string) 'Has_main_character'
                        label: (string) 'protagonist'
                        format: (string)
                        mode: (number) 1
                        redi: (string)
                        typeid: (string) '_wpg'
                3: (table)[0]:
                        key: (string) '_CDAT'
                        label: (string) 'Creation date'
                        format: (string)
                        mode: (number) 1
                        redi: (string)
                        typeid: (string) '_dat'
        version: (number) 2
        results: (table)[2]:
```

Tobias Oetterer
Center for Information and Media Technologies

mw.smw.getQueryResult

```
1: (table)[0]:
        exists: (string) '1'
        displaytitle: (string)
        printouts: (table)[0]:
                protagonist: (table)[2]:
                        1: (table)[0]:
                                exists: (string) '1'
                                displaytitle: (string)
                                namespace: (number) 0
                                fulltext: (string) 'Dream (comics)'
                                fullurl: (string) 'https://sandbox.semantic-mediawiki.org/wiki/Dream_(comics)'
                        2: (table)[0]:
                                exists: (string)
                                displaytitle: (string)
                                fullurl: (string) 'https://sandbox.semantic-mediawiki.org/wiki/Death_(DC_Comics)'
                Creation date: (table)[1]:
                        1: (table)[0]:
                                raw: (string) '1/2016/12/14/8/39/18/0'
                                timestamp: (string) '1481704758'
                Has comic book artist: (table)[5]:
                        2: (table)[0]:
                                exists: (string)
                                displaytitle: (string)
                                namespace: (number) 0
                                fulltext: (string) 'Malcolm Jones III'
                                fullurl: (string) 'https://sandbox.semantic-mediawiki.org/wiki/Malcolm_Jones_III'
                        3: (table)[0]:
                                exists: (string)
                                displaytitle: (string)
                                namespace: (number) 0
                                fulltext: (string) 'Colleen Doran'
                                fullurl: (string) 'https://sandbox.semantic-mediawiki.org/wiki/Colleen_Doran'
        namespace: (number) 0
        fulltext: (string) 'Sandman - Dream Country'
        fullurl: (string) 'https://sandbox.semantic-mediawiki.org/wiki/Sandman_-_Dream_Country'
```

Tobias Oetterer
Center for Information and Media Technologies

Data storage

Tobias Oetterer
Center for Information and Media Technologies

Data storing

Two functions are available for data storing:

1. mw.smw.set
2. mw.smw.subobject

```
if result == true then
    return 'Your data was stored successfully in a subobject\n<pre>' .. mw.dumpObject(args) .. '</pre>'
else
    return 'An error occurred during the subobject storage process. Message reads ' .. result.error
end
```

Tobias Oetterer
Center for Information and Media Technologies

Tooltips

Tobias Oetterer
Center for Information and Media Technologies

mw.smw.info

To complete the set of available function, there is also an implementation for the #info parser function. It takes one or two arguments:

1. The text, you want to display in the tooltip
2. [optional] the type of tooltip image to use [0].

```
output = mw.smw.info( 'nice tooltip to display a warning', 'warning' )
```

[0]: https://www.semantic-mediawiki.org/wiki/Help:Adding_tooltips

Tobias Oetterer
Center for Information and Media Technologies

Query building

Tobias Oetterer
Center for Information and Media Technologies

Topics

```
-- example query definition using a string
-- ask and getQueryResult
local query = '[[Category:Comic books]][[Has comic book writer::Neil
Gaiman]]|?#-=page|?Has main character=protagonist|?Has comic book artist#-|?Creation
date|limit=2|mainlabel=-'
local result = mw.smw.ask(query)
-- set
local query = 'has author=User:Oetterer|has keyword=documentation|has keyword=example|has
keyword=ssc|has keyword=SemanticScribunto'
local result = mw.smw.set(query)
-- subobject
local query = 'has firstname=Daisy|has lastname=Duck'
local result = mw.smw.subobject(query, 'LadyDuck')

-- example definition using a table
-- ask and getQueryResult
local query = { '[[Category:Comic books]][[Has comic book writer::Neil Gaiman]]',
'?#-=page', '?Has main character=protagonist', '?Has comic book artist#-', '?Creation
date', 'limit=2', 'mainlabel=-' }
local result = mw.smw.getQueryResult(query)
-- set
local query = { 'has author=User:Oetterer', 'has keyword=documentation', 'has
keyword=example', 'has keyword=ssc', 'has keyword=SemanticScribunto' }
local result = mw.smw.set(query)
-- since now, we have a number indexed table where lua maintaines order, we can do
something like this, as well:
local query = { 'has author=User:Oetterer', 'has keyword=documentation;example;
ssc;SemanticScribunto', '+sep=;' }
local result = mw.smw.set(query)
-- subobject
local query = { 'has firstname=Daisy', 'has lastname=Duck' }
local result = mw.smw.subobject(query, 'LadyDuck')

-- example definition using a table with key/value pairs (where applicable
-- ask and getQueryResult
local query = { '[[Category:Comic books]][[Has comic book writer::Neil Gaiman]]',
['?#-']='page', ['?Has main character']='protagonist', '?Has comic book artist#-',
'?Creation date', limit=2, mainlabel='-' }
local result = mw.smw.ask(query)
-- set
-- obviously set cannot use key/value pairs, with multiple values for one property. Also,
you cannot use +sep as key, because we don't have the fixed order we need! :(
-- this works. but why?
local query = { ['has author']='User:Oetterer', ['has keyword']='documentation', 'has
keyword=example', 'has keyword=ssc', 'has keyword=SemanticScribunto' }
local result = mw.smw.set(query)
```

Tobias Oetterer
Center for Information and Media Technologies

Documentation references

You can find documentation about SemanticScribunto on

https://github.com/SemanticMediaWiki/SemanticScribunto/tree/master/docs

There are also some examples and a little bit more “hands-on” documentation available on

https://sandbox.semantic-mediawiki.org/wiki/Semantic_Scribunto_example

And

https://sandbox.semantic-mediawiki.org/wiki/Category:Semantic_Scribunto_example

Tobias Oetterer
Center for Information and Media Technologies

Try it yourself

What do you need to try it yourself?