

---

# **gs.group.messages.text Documentation**

*Release 1.0.0*

**GroupServer.org**

April 28, 2016



<b>1</b>	<b>Support for displaying the plain-text version of a post</b>	<b>1</b>
1.1	<code>gs.group.messages.text</code> API	1
1.2	Changelog	4
1.3	Indices and tables	4
1.4	Resources	4



---

## Support for displaying the plain-text version of a post

---

**Author** Michael JasonSmith

**Contact** Michael JasonSmith <mpj17@onlinegroups.net>

**Date** 2015-10-30

**Organization** GroupServer.org

**Copyright** This document is licensed under a [Creative Commons Attribution-Share Alike 4.0 International License](#) by [OnlineGroups.net](#).

Contents:

### 1.1 `gs.group.messages.text` API

There are three parts to the API provided by the `gs.group.messages.text` product. The *split message* code separates the bottom quoting and signatures from the rest of the message. The *HTML body* code will format the parts of the message, using the *matcher* code.

#### 1.1.1 Split message

An email message is normally in two parts: the actual body of the message, and then some trailing bottom quoting and signatures. The *SplitMessage* named tuple represents this duality, while the *split\_message()* function does the actual splitting. Both parts of the message can be fed into the *HTMLBody* class to generate the markup.

`gs.group.messages.text.SplitMessage (:class:'collections.namedtuple')`

The 2-tuple containing the strings representing

0.The main body of the message (`intro`) and

1.The rest of the message, including the bottom-quoting and the footer (`remainder`).

`gs.group.messages.text.split_message (messageText, max_consecutive_comment=12, max_consecutive_whitespace=3)`

Split the message into main body and the footer.

##### Parameters

- **messageText** (*str*) – The text to process.
- **max\_consecutive\_comment** (*int*) – The maximum number of lines of quoting to allow before snipping.

- **max\_consecutive\_whitespace** (*int*) – The maximum number of lines that just contain whitespace to allow before snipping.

**Returns** 2-tuple, containing the strings representing the main-body of the message, and the footer.

**Return type** *SplitMessage*

Email messages often contain a footer at the bottom, which identifies the user, and who they work for. However, GroupServer has lovely profiles which do this, so normally we want to snip the footer, to reduce clutter.

In addition, many users only write a short piece of text at the top of the email, while the remainder of the message consists of all the previous posts. This method also removes the *bottom quoting*.

Originally a ZMI-side script in `Presentation/Tofu/MailingListManager/lscripts`.

### 1.1.2 HTML body

The *HTMLBody* class will format a plain-text message as HTML. The changes that are made include the following.

- The characters that would cause issues with the XML are escaped. This includes " and ' characters.
- Each line is placed within a `<span>` element, with the CSS class set to `line`.

```
<span class="line">Like this</span>
```

- Lines that start with `>` but not `>From` are considered quotes, and given the additional CSS class `muted`.

```
<span class="line muted">&gt; Like this</span>
```

- The words of the line markup by the *matcher* classes.

**class** `gs.group.messages.text.HTMLBody` (*originalText*)

The HTML form of a plain-text email body.

**Parameters** **originalText** (*str*) – The original (plain) text

**\_\_iter\_\_** ()

The marked-up lines in the main body

**\_\_unicode\_\_** ()

The main part of the HTML body, as a Unicode string

**\_\_str\_\_** ()

The main part of the HTML body, as an ASCII string. Non-ASCII characters are replaced with XML entities.

**markup** (*line*)

Markup the line, and the words in the line

**Parameters** **line** (*str*) – The line to mark up.

**Returns** An HTML form of the line: the characters escaped, the words marked up, and surrounded in a `<span>` element.

**Return type** *str*

**markup\_words** (*line*)

Mark up the words on the line

**Parameters** **line** (*str*) – The line to mark up

**Returns** The line with the words marked up

**Return type** *str*

### 1.1.3 Matcher

The *matcher* classes

- Test that a word matches, and
- Produce a substitute for the word.

They all inherit from the *Matcher* class.

**class** `gs.group.messages.text.Matcher` (*matchRE*, *subStr*, *weight=10*)  
Match a word, by a regular expression, and make a substitution

**Parameters**

- **matchRE** (*str*) – The regular expression used to check if there was a match (see `re.match()`)
- **subStr** (*str*) – The string specifying the substitution (see `re.sub()`)

**re = None**

The regular expression used to make the match. The flags `re.I`, `re.M`, and `re.U` are set.

**match** (*s*)

Does the string match the regular expression?

**Parameters** **s** (*str*) – The string to evaluate

**Returns** `True` if the string matches the regular expression, `False` otherwise.

**Return type** `bool`

**sub** (*s*)

Substitute the string in for the substitution string

**Parameters** **s** (*str*) – The string to process

**Returns** The new string substituted in `self.subStr`

**Return type** `unicode`

### Instances

Four instances of the *Matcher* class are provided to make the following changes to the email.

- Words in *\*asterisk\** characters are made bold

`gs.group.messages.text.boldMatcher = <gs.group.messages.text.matcher.Matcher object>`

Turn words within *\*asterisk\** characters into bold-elements. This is as close as GroupServer gets to implementing a wiki.

- Email addresses are made clickable

`gs.group.messages.text.emailMatcher = <gs.group.messages.text.matcher.Matcher object>`

Turn email addresses (`person@example.com`) into clickable `mailto:` links. Surrounding text (such as parenthesis) is added to the *link text*, while the address is extracted and used as for the *link target*.

- Site names starting with *www* are made clickable.

`gs.group.messages.text.wwwMatcher = <gs.group.messages.text.matcher.Matcher object>`

Turn site names that start with *www* (`www.example.com`) into clickable `http://` links.

- URLs (`http` and `https`) are made clickable.

```
gs.group.messages.text.uriMatcher = <gs.group.messages.text.matcher.URIMatcher object>
```

Turn URIs (both `http` and `https`) into clickable links. If the link is particularly long (over 64 characters) then small text will be used (`<a class="small">`). Leading and trailing characters (like parenthesis) will be used in the *link text* while just the URL will be used for the *link target*.

## 1.2 Changelog

### 1.2.1 1.1.0 (2016-03-31)

- Adding the quotation symbols (“> “) to wrapped lines.
- Tweaking the bottom-quoting detection code so messages from Mozilla Thunderbird (like [this message](#) are handled correctly.

### 1.2.2 1.0.0 (2015-10-30)

Initial version. Prior to the creation of this product the code for handling the plain-text version of the posts was split between `gs.group.messages.post.text` and `gs.group.list.email.html`.

This product provides the utilities and functions that supports the displaying of these posts, particularly the conversion of the plain-text post to HTML, documented in [the API](#).

The **actual rendering** of the messages is carried out in either:

1. `gs.group.messages.post.text.base` for the plain-text version of the post shown on the web, or
2. `gs.group.list.email.html` for the HTML version of the plain-text message (the *pseudo HTML*) that is used in email messages.

## 1.3 Indices and tables

- [genindex](#)
- [modindex](#)
- [search](#)

## 1.4 Resources

- Documentation <https://groupserver.readthedocs.io/projects/gsgrouppagemessagetext>
- Code repository: <https://github.com/groupserver/gs.group.messages.text/>
- Questions and comments to <http://groupserver.org/groups/development>
- Report bugs at <https://redmine.iopen.net/projects/groupserver>



## Symbols

`__iter__()` (gs.group.messages.text.HTMLBody method),  
2  
`__str__()` (gs.group.messages.text.HTMLBody method),  
2  
`__unicode__()` (gs.group.messages.text.HTMLBody  
method), 2

## B

`boldMatcher` (in module gs.group.messages.text), 3

## E

`emailMatcher` (in module gs.group.messages.text), 3

## H

`HTMLBody` (class in gs.group.messages.text), 2

## M

`markup()` (gs.group.messages.text.HTMLBody method),  
2  
`markup_words()` (gs.group.messages.text.HTMLBody  
method), 2  
`match()` (gs.group.messages.text.Matcher method), 3  
`Matcher` (class in gs.group.messages.text), 3

## R

`re` (gs.group.messages.text.Matcher attribute), 3

## S

`split_message()` (in module gs.group.messages.text), 1  
`SplitMessage` (in module gs.group.messages.text), 1  
`sub()` (gs.group.messages.text.Matcher method), 3

## U

`uriMatcher` (in module gs.group.messages.text), 3

## W

`wwwMatcher` (in module gs.group.messages.text), 3