Flask-Discussion

Release 0.1.0

Rafael Medina García

Mar 01, 2020

CONTENTS

1 User Guide 3

1.1 Quickstart 3

1.2 Macros 4

1.3 Configuration 7

Index 9

Flask-Discussion is an extension for Flask that adds support for several discussion/comment systems to your application. Simply use the provided *macros* in your templates:

```
{% import "flask_discussion/helper.html" as discussion %}

<html>
    <body>
        {{ discussion.render_comments(title="Page title", identifier="my-page", url=
↪"http://mypage.com") }}
    </body>
</html>
```

CHAPTER

ONE

USER GUIDE

1.1 Quickstart

First install the extension:

```
pip install Flask-Discussion
```

And then initialize it in your application:

```
from flask import Flask
from flask_discussion import Discussion

discussion = Discussion()

def init_app():
    app = Flask(__name__)

    # Set config values
    # .....

    discussion.init_app(app)
```

This will register the extension templates (which contain the macros for each comment system) with your application, making them available in your own templates.

If you want to be able to swap comment systems through the configuration of your application, you may use the macro defined in `flask_discussion/helper.html`:

```
{% import "flask_discussion/helper.html" as discussion %}

<html>
    <body>
        {{ discussion.render_comments(title="Page title", identifier="my-page", url=
↪"http://mypage.com") }}
    </body>
</html>
```

Note that the macro **receives any keyword argument and will relay the appropriate information to the corresponding system-specific macro** (i.e. each comment system may use only specific information, therefore you should provide all possible values for all comment systems).

However, if you only want to use a specific comment system, you may import its macros directly. For instance, for Disqus:

```
{% import "flask_discussion/disqus.html" as disqus %}

<html>
    <body>
        {{ disqus.render_comments(identifier="my-page", url="http://mypage.com", title="Page title") }}
    </body>
</html>
```

Check the next sections to see configuration variables available for each supported comment system.

1.2 Macros

Flask-Discussion exposes a set of Jinja macros for use in Flask application templates. Note that macros for each system will require specific configuration values.

1.2.1 Helper macro

The **helper macro** renders the appropriate comment system depending on your configuration. Internally, it calls the system-specific macros in a transparent way. This is useful in case you want to support several comment systems by just modifying the configuration of your application instead of its code.

Available in the `flask_discussion/helper.html` template.

`helper.`**`render_comments`**`()`

Render the comment system specified in the `DISCUSSION_SYSTEM` configuration key. If no valid value is provided, an error will be shown.

Note that this macro **accepts any keyword argument** and forwards the appropriate values to the system-specific macro called underneath. Therefore, if you intend to support several systems, you should specify all the values that would be required.

Example usage:

```
{% import "flask_discussion/helper.html" as discussion %}

<html>
    <body>
        {# Specify values used by several comment systems #}
        {{ discussion.render_comments(title="Page title", identifier="my-page", url="http://mypage.com") }}
    </body>
</html>
```

1.2.2 Disqus macros

These macros integrate Disqus comments into an application.

Available in the `flask_discussion/disqus.html` template.

`disqus.`**`render_comments`**`(`*identifier*, *url*, *title*`)`

Render Disqus comments using the provided parameters and embed the JS client.

Parameters

- **`identifier`** – Unique page identifier (e.g. blog post ID)
- **`url`** – URL of the page
- **`title`** – Title of the page

Example usage:

```
{% import "flask_discussion/disqus.html" as disqus %}

<html>
    <body>
        {{ disqus.render_comments("my-page", "http://mypage.com", "Page title") }}
    </body>
</html>
```

`disqus.`**`render_comment_count`**`()`

Render Disqus comment count.

This should be called before the closing `</body>` tag. Make sure to include `#disqus_thread` in the `href` attribute of the link for which to show the comment count.

Example usage:

```
{% import "flask_discussion/disqus.html" as disqus %}

<html>
    <body>
        <a href="http://mypage.com#disqus_thread">Comments</a>
        {{ disqus.render_comment_count() }}
    </body>
</html>
```

1.2.3 Isso macros

These macros integrate Isso comments into an application.

Available in the `flask_discussion/isso.html` template.

`isso.`**`embed_client`**`()`

Embed Isso script.

This can either be used to globally include the script (e.g. in the head of the page) or called as part of the `isso.render_comments()` macro.

Example usage:

```
{% import "flask_discussion/isso.html" as isso %}

<html>
    <body>
        <a href="http://mypage.com">Visit my page</a>

        {{ isso.embed_client() }}
    </body>
</html>
```

`isso.`**`render_comments`**(*identifier=""*, *title=""*, *include_script=true*)

Render Isso comments.

Parameters

- **`identifier`** – Optional identifier of the page
- **`title`** – Optional title of the page
- **`include_script`** – Set to `false` to not call the `isso.embed_client()` macro (e.g. for cases in which the client is embeded globally).

Example usage:

```
{% import "flask_discussion/isso.html" as isso %}

<html>
    <body>
        {{ isso.render_comments("my-page", "Page title") }}
    </body>
</html>
```

`isso.`**`render_comment_count`**()

Render Isso comment count.

This **cannot be use at the same time as the full Isso client**. Make sure to include `#isso_thread` in the `href` attribute of the link for which to show the comment count.

Example usage:

```
{% import "flask_discussion/isso.html" as isso %}

<html>
    <body>
        <a href="http://mypage.com#isso_thread">Comments</a>
        {{ isso.render_comment_count() }}
    </body>
</html>
```

1.3 Configuration

Flask-Discussion supports several comment systems, as well as configuring individual aspects of each of them through the configuration of your application. Each of these configuration keys follows the naming convention `DISCUSSION_X_Y` where `X` is the name of the comment system and `Y` the name of the configuration parameter (usually being the same name as specified in the documentation of said system).

In order to use the helper macro `render_comments()` you need to set the value of the `DISCUSSION_SYSTEM` key to one of the supported systems:

- `"disqus"`
- `"isso"`

Below is a list of specific configuration keys for all supported comment systems:

1.3.1 Disqus Configuration

The following configuration keys are available for Disqus comments:

`DISCUSSION_DISQUS_SHORTNAME`	Disqus shortname configured in the Admin panel

1.3.2 Isso Configuration

The following configuration keys are available for Disqus comments. Note that all the keys correspond to the data attributes used to configure the Isso client:

`DISCUSSION_ISSO_URL`	URL where the Isso server resides. **Required**.
`DISCUSSION_ISSO_CSS`	Set to `False` to prevent Isso from automatically appending the stylesheet. **Defaults to** `True`.
`DISCUSSION_ISSO_LANG`	Override useragent's preferred language with one of the languages available in Isso (two letter code). **Defaults to** `en`.
`DISCUSSION_ISSO_REPLY_TO_SELF`	Set to `True` when server spam guard is configured with `reply-to-self = true`.
`DISCUSSION_ISSO_REQUIRE_AUTHOR`	Set to `True` when server spam guard is configured with `require-author = true`.
`DISCUSSION_ISSO_REQUIRE_EMAIL`	Set to `True` when server spam guard is configured with `require-email = true`.
`DISCUSSION_ISSO_REPLY_NOTIFICATIONS`	Set to `True` when reply notifications is configured in the server with `reply-notifications = true`.
`DISCUSSION_ISSO_MAX_COMMENTS_TOP`	Number of top level comments to show by default. If some comments are not shown, an `"X Hidden"` link is shown. Set to `"inf"` to show all or `"0"` to hide all.
`DISCUSSION_ISSO_MAX_COMMENTS_NESTED`	Number of nested comments to show by default. If some comments are not shown, an `"X Hidden"` link is shown. Set to `"inf"` to show all or `"0"` to hide all.
`DISCUSSION_ISSO_REVEAL_ON_CLICK`	Number of comments to reveal upon clicking the `"X Hidden"` link.
`DISCUSSION_ISSO_AVATAR`	Set to `True` to enable avatar generation or `False` to disable it.
`DISCUSSION_ISSO_AVATAR_BG`	Set avatar background color. Any valid CSS color will do.
`DISCUSSION_ISSO_AVATAR_FG`	Set avatar foreground color. Up to 8 colors are possible. The default color scheme is based on this color palette. Multiple colors must be separated by space. If you use less than eight colors and not a multiple of 2, the color distribution is not event.
`DISCUSSION_ISSO_GRAVATAR`	Set to `True` to use gravatar images instead of generating svg images. You have to set `DISCUSSION_ISSO_AVATAR` to `False` if you want to use this. Otherwise both the gravatar and avatar svg image will show up. This requires setting the `gravatar = true` configuration in the server.
`DISCUSSION_ISSO_VOTE`	Set to `True` to enable voting feature on the client side.
`DISCUSSION_ISSO_VOTE_LEVELS`	List of vote levels used to customize comment appearance based on score. Provide comma-separated values (e.g. `"0,5,10,25,100"`) or a JSON array (e.g. `"[-5,5,15]"`). For example, the value `"-5,5"` will cause each `isso-comment` to be given one of these 3 classes: • `isso-vote-level-0` for scores lower than -5 • `isso-vote-level-1` for scores between -5 and 4 • `isso-vote-level-2` for scores of 5 and greater These classes can then be used to customize the appearance of comments (e.g. put a star on popular comments).
`DISCUSSION_ISSO_FEED`	Set to `True` to enable the addition of a link to the feed for the comment thread. The link will only be valid if the appropriate setting, in `[rss]` section, is also enabled server-side.

INDEX

D

disqus.render_comment_count() (*built-in function*), 5

disqus.render_comments() (*built-in function*), 5

H

helper.render_comments() (*built-in function*), 4

I

isso.embed_client() (*built-in function*), 5

isso.render_comment_count() (*built-in function*), 6

isso.render_comments() (*built-in function*), 6