massmedia Documentation

Release 1.3b3

me

October 19, 2016

Contents

1 Configuration Settings **3**
- 1.1 MMEDIA_IMAGE_EXTS . . . 3
- 1.2 MMEDIA_VIDEO_EXTS . . . 3
- 1.3 MMEDIA_AUDIO_EXTS . . . 3
- 1.4 MMEDIA_FLASH_EXTS . . . 3
- 1.5 MMEDIA_DOC_EXTS . . . 4
- 1.6 MMEDIA_INFO_QUALITY . . . 4
- 1.7 MMEDIA_THUMB_SIZE . . . 4
- 1.8 MMEDIA_EXTRA_MIME_TYPES . . . 4
- 1.9 MMEDIA_FS_TEMPLATES . . . 4
- 1.10 MMEDIA_LOCAL_IMPORT_TMP_DIR . . . 4
- 1.11 GRAB_API_KEY . . . 5
- 1.12 GRAB_API_URL . . . 5

2 Mass Media Reference **7**
- 2.1 Models . . . 7

3 TinyMCE and Filebrowsing **9**

4 You Tube Collections **11**
- 4.1 Settings . . . 11
- 4.2 Getting the playlist URL . . . 11
- 4.3 Creating the Collection . . . 13

5 YouTube Collections and Templates **15**
- 5.1 mm_youtube template tag library . . . 15

6 Rendering media in templates **23**

7 Indices and tables **25**

Python Module Index **27**

Contents:

CHAPTER 1

Configuration Settings

1.1 MMEDIA_IMAGE_EXTS

Recognized extensions for image files. It should be a `list` or `tuple` of `strings` without the leading period. **Default:**

```
MMEDIA_IMAGE_EXTS = ('bmp','gif','ico','cur','jpg','jpeg','pcx',
                     'png','psd','tga','tiff','wmf','xcf','bmp',
                     'wmf','apm','emf')
```

1.2 MMEDIA_VIDEO_EXTS

Recognized extensions for video files. It should be a `list` or `tuple` of `strings` without the leading period. **Default:**

```
MMEDIA_VIDEO_EXTS = ('asf','wmv','flv','mov','mpeg','mpg','mpe','vob',
                     'qt','mp4','m4v','rm','avi','ogm')
```

1.3 MMEDIA_AUDIO_EXTS

Recognized extensions for audio files. It should be a `list` or `tuple` of `strings` without the leading period. **Default:**

```
MMEDIA_AUDIO_EXTS = ('asf','aif','aiff','aifc','flac','au','snd','mid',
                     'midi','mpa','m4a','mp1','mp2','mp3','ra','xm',
                     'wav','ogg')
```

1.4 MMEDIA_FLASH_EXTS

Recognized extensions for SWF files. It should be a `list` or `tuple` of `strings` without the leading period. **Default:**

```
MMEDIA_FLASH_EXTS = ('swf',)
```

1.5 MMEDIA_DOC_EXTS

Recognized extensions for generic document files. It should be a `list` or `tuple` of `strings` without the leading period. **Default:**

```
MMEDIA_DOC_EXTS = ('pdf','xls','doc')
```

1.6 MMEDIA_INFO_QUALITY

The `hachoir_metadata` library allows you to specify how it parses metadata, from 0.0 (fastest) to 1.0 (best quality). **Default:**

```
MMEDIA_INFO_QUALITY = 1.0
```

1.7 MMEDIA_THUMB_SIZE

The size of automatically generated thumbnails for images. It is a two integer tuple for width and height. The default is 100px wide by 80px high. **Default:**

```
MMEDIA_THUMB_SIZE = (100, 80)
```

1.8 MMEDIA_EXTRA_MIME_TYPES

The Python standard library includes a MIME types map of file extensions to MIME type. However, this map may be incomplete. This setting extends the default mapping. It is a `dict` of dot-extensions mapped to MIME type. **Default:**

```
MMEDIA_EXTRA_MIME_TYPES = {'.flv':'video/x-flv',}
```

1.9 MMEDIA_FS_TEMPLATES

There are two ways to manage the templates used to render content: via the admin or through files on the computer's file system. If this setting is `False`, there will be a `MediaTemplate` model for managing the templates by MIME type. By default, this setting is `True` and templates are managed through the path `massmedia/`. Massmedia looks in a hierarchy for the correct template. For example, an image with a MIME type of `image/jpeg`

```
/massmedia/image/jpeg.html
/massmedia/image/generic.html
/massmedia/generic.html
```

Where it first looks for the most specific template, and then falls back to more generic templates. **Default:**

```
MMEDIA_FS_TEMPLATES = True
```

1.10 MMEDIA_LOCAL_IMPORT_TMP_DIR

TODO

1.11 GRAB_API_KEY

In order to import items from Grab Network, you need an API key.

1.12 GRAB_API_URL

In order to import items from Grab Network, you'll also need an API URL.

CHAPTER 2

Mass Media Reference

Contents:

2.1 Models

CHAPTER 3

TinyMCE and Filebrowsing

Add to project settings:

```
TINYMCE_FILEBROWSER = True
```

In TINYMCE_DEFAULT_CONFIG add:

```
extended_valid_elements "-p[class|style]," and "-div[class|style],"
```

Create URL definition:

```
url(r'^browse/', 'massmedia.views.browse', name="fb_browse"),
```

CHAPTER 4

You Tube Collections

In version 0.8 preliminary support for external services was added. The first service supported was creating a `Collection` of a youtube.com playlist.

Creating a collection from a YouTube playlist **will not import the videos** on the playlist. Because you can change the playlist independently of the Django admin interface, it would be difficult to keep it in sync with YouTube. It is easy to use the Google YouTube API to get all the videos dynamically. See *YouTube Collections and Templates*.

4.1 Settings

The YouTube API requires 3 settings, `EMAIL`, `USERNAME`, `PASSWORD`:

```
MASSMEDIA_SERVICES = {
    'YOUTUBE': {
        'EMAIL': '',
        'USERNAME': '',
        'PASSWORD': '',
    },
}
```

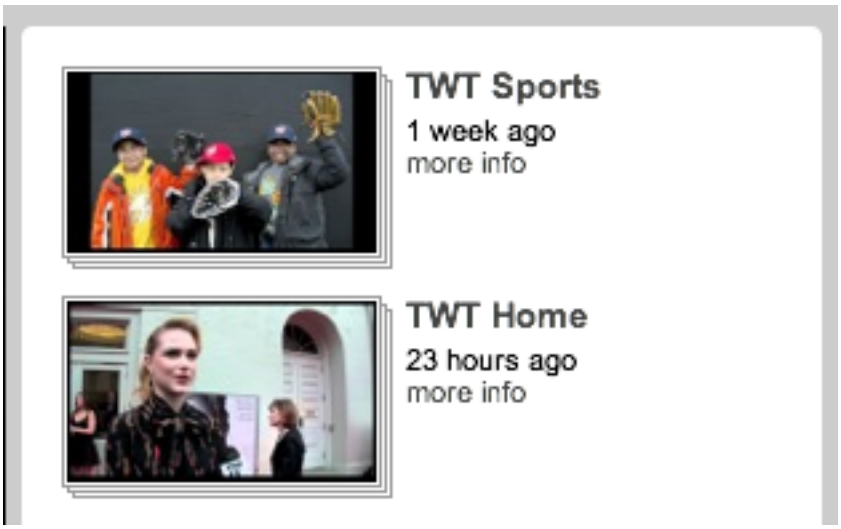
4.2 Getting the playlist URL

YouTube has several different URLs that reference the playlist, depending on how you get to the playlist page.

4.2.1 User's playlist page

If you go to any user's page and click on the "Playlists" button

you can select a playlist from the list on the right.



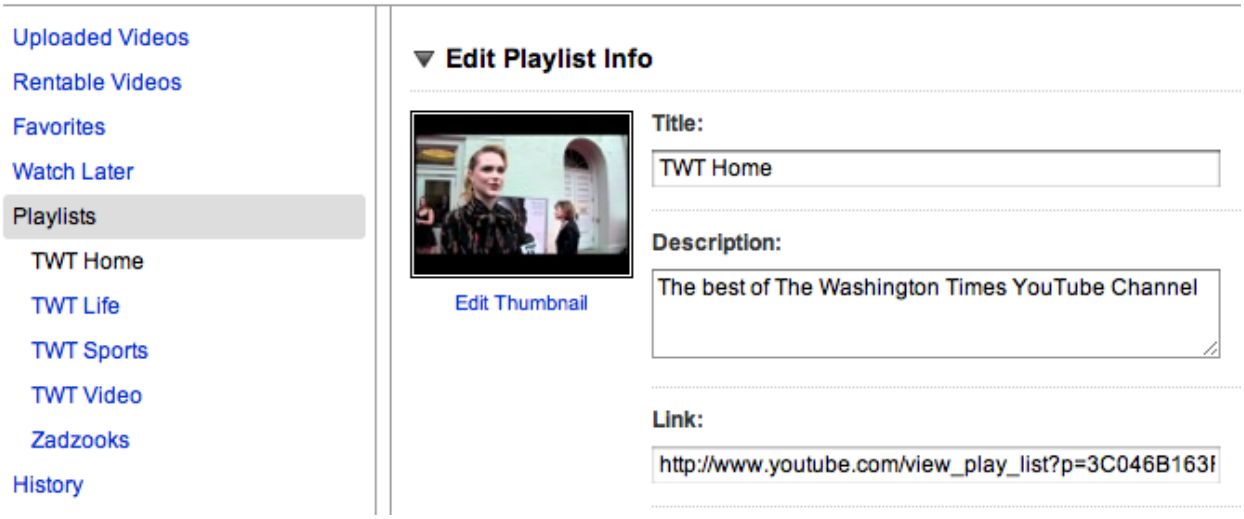
The address bar address will change to look similar to:

```
http://www.youtube.com/washingtontimes#p/c/4A35EB2544D73557
```

4.2.2 Playlist management page

If you go to **My Videos & Playlists**

and select **Playlists** from the list on the left.



You can use the **Link:** field URL; which is formatted like:

```
http://www.youtube.com/view_play_list?p=4A35EB2544D73557
```

If you look at the URL in the **Embed:** field, it contains a URL like:

```
http://www.youtube.com/p/4A35EB2544D73557?hl=en_US&amp;fs=1
```

This URL requires you to chop off the question mark and everything after it:

```
http://www.youtube.com/p/4A35EB2544D73557
```

4.3 Creating the Collection

1. Create a new `Collection` object in the admin.
2. Give the new collection a name
3. Enter the playlist URL retrieved earlier into the **External URL** field.
4. Select the **Site**.
5. Save.

For information on displaying the collection, see *YouTube Collections and Templates*.

CHAPTER 5

YouTube Collections and Templates

5.1 mm_youtube template tag library

The `mm_youtube` template tag library handles all the communication with YouTube, plus there are some extra helper tags and filters.

- *{% get_youtube_feed %}*
- *{% split %}*
- *format_seconds filter*

5.1.1 {% get_youtube_feed %}

Usage: `{% get_youtube_feed <object> as <result> %}`

This tag onverts the `massmedia.models.Collection` `<object>` into an object with the information about the YouTube playlist. The information is stored in the variable specified as `<result>`. The `<result>` object has two attributes: metadata and entries. The metadata contains information about the playlist, and entries is a list of the videos in the playlist.

result.metadata

author *Type:* A list of dictionaries containing `name` and `uri` fields.

Example:

```
{{ result.metadata.author.0.name }}
{{ result.metadata.author.0.url }}
```

could result in:

```
washingtontimes
http://gdata.youtube.com/feeds/api/users/washingtontimes
```

category *Type:* A list of strings.

Example:

```
{% for cat in result.metadata.category %}
    {{ cat }}
{% endfor %}
```

could result in:

```
http://gdata.youtube.com/schemas/2007#playlist
washington
news
politics
```

items_per_page *Type:* A string representation of a number for the default number of items retrieved per page.

Example:

```
{{ result.metadata.items_per_page }}
```

could result in:

```
25
```

link *Type:* A list of dictionaries containing `href`, `rel`, and `type` fields.

Example:

```
{{ result.metadata.link.0.href }}
{{ result.metadata.link.0.rel }}
{{ result.metadata.link.0.type }}
```

could result in:

```
http://www.youtube.com/view_play_list?p=3C046B163FA3957C
alternate
text/html
```

Comment: Typically there are four links with `rel` attributes of:

- `alternate`
- `http://schemas.google.com/g/2005#feed`
- `http://schemas.google.com/g/2005#batch`
- `self`

logo *Type:* A string of the URL to the logo set for this playlist.

Example:

```
{{ result.metadata.logo }}
```

could result in:

```
http://www.youtube.com/img/pic_youtubelogo_123x63.gif
```

playlistId *Type:* A string representing the ID of the playlist.

Example:

```
{{ result.metadata.playlistId }}
```

could result in:

```
3C046B163FA3957C
```

start_index *Type:* A string representation of a item number of a paginated result list.

Example:

```
{{ result.metadata.start_index }}
```

could result in:

```
1
```

subtitle *Type:* A string of the subtitle of the playlist

Example:

```
{{ result.metadata.subtitle }}
```

could result in:

```
The best of The Washington Times YouTube Channel
```

title *Type:* A string of the title of the playlist

Example:

```
{{ result.metadata.title }}
```

could result in:

```
TWT Home
```

total_results *Type:* A string representation of the number of items returned.

Example:

```
{{ result.metadata.total_results }}
```

could result in:

```
15
```

updated *Type:* A string representation of the date this playlist was last modified.

Example:

```
{{ result.metadata.updated }}
```

could result in:

```
2011-04-13T16:37:29.000Z
```

result.entries

author *Type:* A list of dictionaries containing `name` and `uri` fields.

Example:

```
{{ result.entries.0.author.0.name }}
{{ result.entries.0.author.0.url }}
```

could result in:

```
washingtontimes
http://gdata.youtube.com/feeds/api/users/washingtontimes
```

category *Type:* A list of strings.

Example:

```
{% for cat in result.entries.0.category %}
    {{ cat }}
{% endfor %}
```

could result in:

```
http://gdata.youtube.com/schemas/2007#playlist
{'label': 'Entertainment', 'term': 'Entertainment'}
evan rachel wood
the conspirator
robert redford
mildred pierce
liz glover
dawne langford
gucci
jill stuart
washington
d.c.
ford's theater
```

comments *Type:* A list of dictionaries containing `count_hint` and `href` fields.

Example:

```
{% for item in result.entries.0.comments %}
    {{ item.count_hint }}
    {{ item.href }}
{% endfor %}
```

could result in:

```
0
http://gdata.youtube.com/feeds/api/videos/DLyt64ZtZcw/comments
```

content *Type:* A string

Example:

```
{{ result.entries.0.content }}
```

could result in:

```
Liz Glover chats with Evan Rachel Wood at the D.C. Premiere of "The Conspirator."
```

description *Type:* A string

Example:

```
{{ result.entries.0.description }}
```

could result in:

```
Liz Glover chats with Evan Rachel Wood at the D.C. Premiere of "The Conspirator."
```

link *Type:* A list of dictionaries containing `href`, `rel`, and `type` fields.

Example:

```
{{ result.entries.0.link.0.href }}
{{ result.entries.0.link.0.rel }}
{{ result.entries.0.link.0.type }}
```

could result in:

```
http://www.youtube.com/watch?v=DLyt64ZtZcw&feature=youtube_gdata
alternate
text/html
```

Comment: Typically there are six links with `rel` attributes of:

- `alternate`
- `http://gdata.youtube.com/schemas/2007#video.responses`
- `http://gdata.youtube.com/schemas/2007#video.related`
- `http://gdata.youtube.com/schemas/2007#mobile`
- `related`
- `self`

media *Type:* A dictionary.

Comment: Because the `media` field is complex, each field is discussed separately.

media.category *Type:* A list of dictionaries with `label`, and `text` fields.

Example:

```
{% for cat in result.entries.0.media.category %}
    {{ cat.label }}
    {{ cat.text }}
{% endfor %}
```

could result in:

```
Entertainment
Entertainment
```

media.content *Type:* A list of dictionaries with fields: `duration`, `expression`, `isDefault`, `medium`, `type`, `url`, and `{http://gdata.youtube.com/schemas/2007}format`.

Example:

```
{% for item in result.entries.0.media.content %}
    {{ item.duration|format_seconds:"i:s" }}
    {{ item.expression }}
    {{ item.isDefault }}
    {{ item.medium }}
    {{ item.type }}
    {{ item.url }}
    {{ item.{http://gdata.youtube.com/schemas/2007}format }}
    ---------------------------------
{% endfor %}
```

could result in:

```
01:05
full
true
video
application/x-shockwave-flash
http://www.youtube.com/v/DLyt64ZtZcw?f=playlists&app=youtube_gdata
5
---------------------------------
```

```
01:05
full

video
video/3gpp
rtsp://v4.cache4.c.youtube.com/CiULENy73wIaHAnMZW2G6628DBMYDSANFEgGUglwbGF5bGlzdHMM/0/0/0/video.
1
---------------------------------
01:05
full

video
video/3gpp
rtsp://v4.cache5.c.youtube.com/CiULENy73wIaHAnMZW2G6628DBMYESARFEgGUglwbGF5bGlzdHMM/0/0/0/video.
6
---------------------------------
```

media.description *Type:* A string

Example:

```
{{ result.entries.0.media.description }}
```

could result in:

```
Liz Glover chats with Evan Rachel Wood at the D.C. Premiere of "The Conspirator."
```

media.description *Type:* A string representing the number of seconds of the movie's duration.

Example:

```
{{ result.entries.0.media.duration|format_seconds:"i:s" }}
```

could result in:

```
01:05
```

media.keywords *Type:* string of comma-delimited keywords

Example:

```
{% split result.entries.0.media.keywords ", " as keywords %}
{% for i in keywords %}
    {{ i }}
{% endfor %}
```

could result in:

```
evan rachel wood
the conspirator
robert redford
mildred pierce
liz glover
dawne langford
gucci
jill stuart
washington
d.c.
ford's theater
```

media.player *Type:* A URL string

Example:

```
{{ result.entries.0.media.player }}
```

could result in:

```
http://www.youtube.com/watch?v=DLyt64ZtZcw&feature=youtube_gdata_player
```

media.thumbnail *Type:* A list of dictionaries with `height`, `time`, `url`, and `width`.

Example:

```
{% for i in result.entries.0.media.thumbnail %}
    <img height="{{i.height}}" width="{{i.width}}" src="{{i.url}}" alt="{{time}}">
{% endfor %}
```

could result in:

```
<img height="240" width="320" src="http://i.ytimg.com/vi/DLyt64ZtZcw/0.jpg" alt="00:00:32.500">
<img height="90" width="120" src="http://i.ytimg.com/vi/DLyt64ZtZcw/1.jpg" alt="00:00:16.250">
<img height="90" width="120" src="http://i.ytimg.com/vi/DLyt64ZtZcw/2.jpg" alt="00:00:32.500">
<img height="90" width="120" src="http://i.ytimg.com/vi/DLyt64ZtZcw/3.jpg" alt="00:00:48.750">
```

media.title *Type:* A string

Example:

```
{{ result.entries.0.media.title }}
```

could result in:

```
Evan Rachel Wood at the D.C. Premiere of "The Conspirator"
```

position *Type:* A string representing the position of the item in the results.

Example:

```
{{ result.entries.0.position }}
```

could result in:

```
1
```

statistics *Type:* A dictionary with fields `favorite_count` and `view_count`.

Example:

```
{{ result.entries.0.statistics.favorite_count }}
{{ result.entries.0.statistics.view_count }}
```

could result in:

```
0
151
```

title *Type:* A string

Example:

```
{{ result.entries.0.title }}
```

could result in:

```
Evan Rachel Wood at the D.C. Premiere of "The Conspirator"
```

updated *Type:* A string representation of the date this playlist was last modified.

Example:

```
{{ result.metadata.updated }}
```

could result in:

```
2011-04-13T16:37:29.000Z
```

5.1.2 {% split %}

Usage: `{% split <string or variable> [<split_str>] as <result> %}`

This tag is useful for converting a string to a list of strings. Without the `split_str` parameter, it will split by space.

Example:

```
{% split "1 2 3 4" as result %} {# result == ['1', '2', '3', '4'] #}
{% split "1, 2, 3, 4" ", " as result %} {# result == ['1', '2', '3', '4'] #}

{# assuming commalist == "1,2,3,4" #}
{% split commalist "," as result %} {# result == ['1', '2', '3', '4'] #}
```

5.1.3 format_seconds filter

Usage: `{{ secondsvar|format_seconds:"i:s"}}`

This filter formats a number of seconds using the format provided. It is the same formatting options for Django's date filter.

CHAPTER 6

Rendering media in templates

Each type of media has a customizable template for displaying a thumbnail version and a detail version

They are stored in `massmedia/mediatypes/`

Templates for each type of media are based on MIME type

Each item looks for a template in this order:

For detail templates:

```
massmedia/mediatypes/<base_type>/<specific_type>_detail.html
massmedia/mediatypes/<base_type>/generic_detail.html
massmedia/mediatypes/generic_detail.html
```

For thumbnail templates:

```
massmedia/mediatypes/<base_type>/<specific_type>_thumb.html
massmedia/mediatypes/<base_type>/generic_thumb.html
massmedia/mediatypes/generic_thumb.html
```

As an example, an item with a MIME type of `application/x-shockwave-flash` would look for its templates in:

```
massmedia/mediatypes/application/x-shockwave-flash_detail.html
massmedia/mediatypes/application/generic_detail.html
massmedia/mediatypes/generic_detail.html
```

These templates are only snippets of html for rendering one item and should not contain extra HTML tags (like `<head>` or `<body>`)

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

m

massmedia.models, 7

M

massmedia.models (module), 7