
livewatch Documentation

Release 0.3.0

Moccu GmbH & Co. KG

January 25, 2016

1	What is django-livewatch?	3
2	Features	5
3	Usage	7
3.1	Installation	7
3.2	Configuration	7
3.2.1	LIVELWATCH_EXTENSIONS	7
3.3	Usage	8
3.4	Extending Livewatch	9
3.5	Contribution	9
3.5.1	Django Code	9
3.6	Changelog	10
3.6.1	0.3.0 2016-01-25	10
3.6.2	0.2.5 2015-04-09	10
3.6.3	0.2.4 2015-04-02	10
3.6.4	0.2.3 2015-03-31	10
3.6.5	0.2.2 2015-03-31	10
3.6.6	0.2.1 2015-03-26	10
3.6.7	0.2.0 2015-03-26	10
3.6.8	0.1.1 2015-03-20	10
3.6.9	0.1.0 2015-03-20	11
4	Api documentation	13
4.1	livewatch	13
4.1.1	livewatch package	13
5	Indices and tables	15
	Python Module Index	17

livewatch.de integration for Django projects.

What is django-livewatch?

django-livewatch is a module to integrate your services from livewatch.de. Livewatch has some usefull urls to check if your services are up or down.

Features

- cache integration
- `celery` integration
- `django-rq` integration

3.1 Installation

- Install `django-livewatch` (or [download from PyPI](#)):

```
pip install django-livewatch
```

- If you use `livewatch` with `celery` add it to `INSTALLED_APPS` in `settings.py`:

```
INSTALLED_APPS = (  
    # other apps  
    'livewatch',  
)
```

- Include `livewatch.urls` in your `urls.py`:

```
urlpatterns += patterns('',  
    (r'^livewatch/', include('livewatch.urls')),  
)
```

3.2 Configuration

3.2.1 LIVEWATCH_EXTENSIONS

If you want to use it with `cache` or `django-celery` or `django-rq` you have to update the `LIVEWATCH_EXTENSIONS` setting.

cache

Make sure that you have a `cache` installed and configured.

```
# Example with cache support  
LIVEWATCH_EXTENSIONS = (  
    'livewatch.extensions.cache:CacheExtension',  
)
```

django-celery

Make sure that you have `celery` installed. You can use the `celery` extra target for that.

```
$ pip install django-livewatch[celery]
```

```
# Example with celery support
LIVEWATCH_EXTENSIONS = (
    'livewatch.extensions.rq: CeleryExtension',
)
```

Celery has to be configured in your `project.celery` module that defines the celery app. For more details see the official [celery documentation](#).

Please make sure to load tasks from `INSTALLED_APPS`:

```
app.autodiscover_tasks(lambda: settings.INSTALLED_APPS)
```

If you don't load all tasks from the `INSTALLED_APPS` setting, please use `CELERY_IMPORTS` to register the livewatch tasks.

```
# In settings.py
# Activate livewatch.tasks
CELERY_IMPORTS = (
    'livewatch.tasks',
)
```

django-rq

Make sure that you have `rq` installed. You can use the `rq` extra target for that.

```
$ pip install django-livewatch[rq]
```

```
# Example with rq support
LIVEWATCH_EXTENSIONS = (
    'livewatch.extensions.rq: RqExtension',
)
```

Hint: If you use `celery` or `rq`, you have to ensure that a cache is running!

For details on writing your own extensions, please see the [Extending Livewatch](#) section.

3.3 Usage

Before you can use `django-livewatch`, you have to install and configure it. Please see [Installation](#) for more details.

To integrate `django-livewatch` with [livewatch.de](#) you can use the following URLs:

- `/livewatch/`
- ... if you're using the cache extension:
- `/livewatch/cache/`
- ... if you're using the celery extension:
- `/livewatch/celery/`

... if you're using the rq extension:

- /livewatch/rq/

3.4 Extending Livewatch

```
from livewatch.extensions.base import BaseExtension

class FooExtension(BaseExtension):
    name = 'foo'

    def check_service(self, request):
        # check that service is running
```

If you use a task queue service like celery or rq you can inherit your custom class from the TaskExtension class

```
from livewatch.extensions.base import TaskExtension

class BarExtension(TaskExtension):
    name = 'bar'

    def run_task(self):
        # check that execution of a task works
```

3.5 Contribution

If you like to contribute to this project please read the following guides.

3.5.1 Django Code

To install all requirements for development and testing, you can use the provided requirements file.

```
$ make devinstall
```

Testing the code

django-livewatch uses `py.test` for testing. Please ensure that all tests pass before you submit a pull request. `py.test` also runs PEP8 and PyFlakes checks on every run.

This is how you execute the tests and checks from the repository root directory.

```
$ make tests
```

If you want to generate a coverage report, you can use the following command.

```
$ make coverage
```

Or if you want to generate a HTML version of the coverage report, use the following command.

```
$ make coverage-html
```

The generated HTML files are located in the `htmlcov` folder.

Documentation

django-livewatch uses Sphinx for documentation. You find all the source files in the `docs/source` folder.

To update/generate the HTML output of the documentation, use the following command:

```
$ make docs
```

Please make sure that you don't commit the build files inside `docs/build`.

3.6 Changelog

3.6.1 0.3.0 2016-01-25

- Added logging if services/checks fail
- Added support for Django 1.8 and 1.9 (actually just enabling tests)

3.6.2 0.2.5 2015-04-09

- Another packaging fix for python 2.6

3.6.3 0.2.4 2015-04-02

- Main URL now checks for all registered extensions

3.6.4 0.2.3 2015-03-31

- Fix packaging for python 2.6

3.6.5 0.2.2 2015-03-31

- Restore python 2.6 compatibility

3.6.6 0.2.1 2015-03-26

- Add compatibility with Django 1.5

3.6.7 0.2.0 2015-03-26

- Lots of improvements to our test-suite
- Documentation updates
- Add new extension for testing `django.core.cache`

3.6.8 0.1.1 2015-03-20

- Various documentation updates

3.6.9 0.1.0 2015-03-20

- Initial public release

Api documentation

4.1 livewatch

4.1.1 livewatch package

Subpackages

livewatch.extensions package

Submodules

livewatch.extensions.base module

```
class livewatch.extensions.base.BaseExtension
    Bases: object

    check_service (request)

    name = None

class livewatch.extensions.base.TaskExtension
    Bases: livewatch.extensions.base.BaseExtension

    check_service (request)

    name = 'taskextension'

    run_task ()
```

livewatch.extensions.celery module

```
class livewatch.extensions.celery.CeleryExtension
    Bases: livewatch.extensions.base.TaskExtension

    name = 'celery'

    run_task ()
```

livewatch.extensions.rq module

```
class livewatch.extensions.rq.RqExtension
    Bases: livewatch.extensions.base.TaskExtension

    name = 'rq'
```

```
run_task()
livewatch.extensions.rq.livewatch_update_rq_task(key)
```

Module contents

Submodules

livewatch.tasks module

livewatch.urls module

```
urlpatterns = [
    url(r'^$', LiveWatchView.as_view(), name='livewatch'),
    url(r'^(?P<service>\w+)/$', LiveWatchView.as_view(), name='livewatch-service'),
]
```

livewatch.utils module

```
livewatch.utils.get_extensions(reload_extensions=False)
```

livewatch.views module

```
class livewatch.views.LiveWatchView(**kwargs)
    Bases: django.views.generic.base.View
    get(request, *args, **kwargs)
```

Module contents

Indices and tables

- `genindex`
- `modindex`
- `search`

I

`livewatch`, [14](#)
`livewatch.extensions`, [14](#)
`livewatch.extensions.base`, [13](#)
`livewatch.extensions.celery`, [13](#)
`livewatch.extensions.rq`, [13](#)
`livewatch.tasks`, [14](#)
`livewatch.utils`, [14](#)
`livewatch.views`, [14](#)

B

BaseExtension (class in livewatch.extensions.base), 13

C

CeleryExtension (class in livewatch.extensions.celery), 13

check_service() (livewatch.extensions.base.BaseExtension method), 13

check_service() (livewatch.extensions.base.TaskExtension method), 13

G

get() (livewatch.views.LiveWatchView method), 14

get_extensions() (in module livewatch.utils), 14

L

livewatch (module), 14

livewatch.extensions (module), 14

livewatch.extensions.base (module), 13

livewatch.extensions.celery (module), 13

livewatch.extensions.rq (module), 13

livewatch.tasks (module), 14

livewatch.utils (module), 14

livewatch.views (module), 14

livewatch_update_rq_task() (in module livewatch.extensions.rq), 14

LiveWatchView (class in livewatch.views), 14

N

name (livewatch.extensions.base.BaseExtension attribute), 13

name (livewatch.extensions.base.TaskExtension attribute), 13

name (livewatch.extensions.celery.CeleryExtension attribute), 13

name (livewatch.extensions.rq.RqExtension attribute), 13

R

RqExtension (class in livewatch.extensions.rq), 13

run_task() (livewatch.extensions.base.TaskExtension method), 13

run_task() (livewatch.extensions.celery.CeleryExtension method), 13

run_task() (livewatch.extensions.rq.RqExtension method), 13

T

TaskExtension (class in livewatch.extensions.base), 13