# click-log Documentation

***Release 0.3.2***

**Markus Unterwaditzer & contributors**

**Jun 06, 2018**

# Contents

CHAPTER 1

# click-log

Integrates logging with click.

- Documentation
- Source code

CHAPTER 2

# License

Licensed under the MIT, see `LICENSE`.

# CHAPTER 3

## Getting started

Assuming you have this Click application:

```
@click.command()
def cli():
    click.echo("Dividing by zero.")

    try:
        1 / 0
    except:
        click.echo("ERROR: Failed to divide by zero.")
```

Ignore the application's core functionality for a moment. The much more pressing question here is: How do we add an option to not print anything on success? We could try this:

```
@click.command()
@click.option('--quiet', default=False, is_flag=True)
def cli(quiet):
    if not quiet:
        click.echo("Dividing by zero.")

    try:
        1 / 0
    except:
        click.echo("ERROR: Failed to divide by zero.")
```

Wrapping if-statements around each `echo`-call is cumbersome though. And with that, we discover logging:

```
import logging
logger = logging.getLogger(__name__)
# More setup for logging handlers here

@click.command()
@click.option('--quiet', default=False, is_flag=True)
def cli(quiet):
```

(continues on next page)

(continued from previous page)

```
    if quiet:
        logger.setLevel(logging.ERROR)
    else:
        logger.setLevel(logging.INFO)

    ...
```

Logging is a better solution, but partly because Python's logging module aims to be so generic, it doesn't come with sensible defaults for CLI applications. At some point you might also want to expose more logging levels through more options, at which point the boilerplate code grows even more.

This is where click-log comes in:

```
import logging
logger = logging.getLogger(__name__)
click_log.basic_config(logger)

@click.command()
@click_log.simple_verbosity_option(logger)
def cli():
    logger.info("Dividing by zero.")

    try:
        1 / 0
    except:
        logger.error("Failed to divide by zero.")
```

The output will look like this:

```
Dividing by zero.
error: Failed to divide by zero.
```

The `error:`-prefix will be red, unless the output is piped to another command.

The *`simple_verbosity_option()`* decorator adds a `--verbosity` option that takes a (case-insensitive) value of `DEBUG`, `INFO`, `WARNING`, `ERROR`, or `CRITICAL`, and calls `setLevel` on the given logger accordingly.

---

**Note:** Make sure to define the *simple_verbosity_option* as early as possible. Otherwise logging setup will not be early enough for some of your other eager options.

---

# CHAPTER 4

# API

click_log.**basic_config**(*logger=None*)
Set up the default handler (*ClickHandler*) and formatter (*ColorFormatter*) on the given logger.

click_log.**simple_verbosity_option**(*logger=None*, **names*, ***kwargs*)
A decorator that adds a *–verbosity, -v* option to the decorated command.

Name can be configured through `*names`. Keyword arguments are passed to the underlying `click.option` decorator.

## 4.1 Classes

**class** click_log.**ClickHandler**(*level=0*)

**class** click_log.**ColorFormatter**(*fmt=None*, *datefmt=None*)

CHAPTER 5

# Indices and tables

- genindex
- modindex
- search

# Python Module Index

## C

# Index