workerprocess Documentation

Release 0.1.2

Philip Cristiano

August 11, 2012

CONTENTS

WorkerProcess is a library that makes your day simpler by providing an easy base class for background workers.

```
from workerprocess import BaseWorker


class ExampleWorker(BaseWorker):

    def tick(self):
        time.sleep(1)


if __name__ == '__main__':
    ExampleWorker.main()
```

The *tick* method will be called in an infinite loop. WorkerProcess will handle *SIGTERM* and *SIGHUP*. You can override the default *SIGHUP* behavior (of nothing) by adding the method *sighup*. *startup* and *shutdown* methods are available as well that will run before and after the loop.

WorkerProcess can also limit the number of loops per second with the *max_ticks_per_second* variable. This supports floating point numbers as well so *.1* will run once every 10 seconds. This is mainly used for keeping the worker from going into a busy loop.

A full example:

```
from workerprocess import BaseWorker


class ExampleWorker(BaseWorker):

    max_ticks_per_second = 1

    def startup(self):
        print 'Starting...'

    def tick(self):
        print 'Tick!'

    def shutdown(self):
        print 'Shutting down.'

    def sighup(self):
        print 'Hanging up.'


if __name__ == '__main__':
    ExampleWorker.main()
```

CHAPTER

ONE

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*