Louie Documentation

Release 1.1

Patrick K. O'Brien and contributors

January 08, 2016

Contents

Contents:

CHAPTER 1

About

Louie is based on PyDispatcher. Here is a very good detailed description of what both PyDispatcher and Louie offer you, based on the description available at the PyDispatcher website.

PyDispatcher provides a multiple-producer-multiple-consumer signal-registration and routing infrastructure, suitable for use in multiple contexts.

The dispatcher mechanism is particularly useful when constructing Model-View-Controller style applications where it is not desirable to have the Model objects aware of the event model.

To be more concrete about what PyDispatcher does for you, here are some specifics:

- A centralized service delivers messages to registered objects in the local process. You can register any number of functions or other callable objects which can receive signals from senders.
  - Registration can be for any sender, particular sending objects, or "anonymous" messages (messages where the sender is None).
  - Registration can be for all signals, or particular signals.
  - A single signal will be delivered to all appropriate registered receivers, so that multiple registrations do not interfere with each other.
  - The sender or receiver need not be be dispatcher-aware. Any Python object, except for *None*, can act as a sender, and any callable object can act as a receiver.
  - The system uses weak references to receivers wherever possible.
  - Object lifetimes are not affected by PyDispatcher registrations. When your object goes away, the registrations related to the object also go away.
  - References to common transient objects (instance methods in particular) are stored as compound weak references.
  - Weak references can be disabled on a registration-by-registration basis.
- It allows rich signal types. Signals must simply be hashable objects; they are otherwise opaque to the dispatch mechanism.
- Positional and keyword arguments may be attached to a signal when sending. For each receiver, PyDispatcher sends each receiver the arguments that they expect; other arguments are silently dropped. Thus, receivers can be general in nature, even ignoring all arguments, or they can be specific, accepting whichever arguments it needs.

CHAPTER 2

Changes

This document provides a detailed list of changes made to Louie, including differences between PyDispatcher and the initial release of Louie.

2.1 Changes from PyDispatcher to Louie 1.0

2.1.1 Packaging and Distribution

- Louie uses setuptools for managing its placement in the Python package hierarchy.

2.1.2 Naming and Importing

- The package name for Louie is *louie*.
- The preferred way of using Louie is to only import the *louie* package, e.g.:

```
import louie
louie.connect(...)
louie.send(...)
```

- Function and method names are lowercase_with_underscores, to conform to PEP-0008.

2.1.3 Plug-ins

- Louie provides globally-registered plug-ins that augment various aspects of Louie's operation.
- Available plug-ins include the following:
  - *QtWidgetPlugin* knows how to handle Qt widgets that still exist as Python objects, but whose C++ objects have been destroyed.
  - *TwistedDispatchPlugin* converts Louie's default synchronous signal dispatching behavior to an asynchronous behavior based on Twisted Deferred objects.

Contributors

The following people have contributed code to PyDispatcher and Louie:

- Patrick K. O'Brien
- Mike C. Fletcher
- Matthew R. Scott
- Brad Arndt
- Vasily Evseenko

Louie provides Python programmers with a straightforward way to dispatch signals between objects in a wide variety of contexts. It is based on PyDispatcher, which in turn was based on a highly-rated recipe in the Python Cookbook.

Louie is licensed under The BSD License.

CHAPTER 4

Louie Requirements

Python 2.3 or higher.

Tested against Python 2.6 through 3.5.

CHAPTER 5

Installing Louie

Louie uses pip for installation, and is distributed via the Python Package Index.

In many modern Python environments, and also starting with Python 3.4, pip installed by default.

Run this command:

```
pip install louie
```

CHAPTER 6

Upgrading Louie

Run this command to upgrade Louie to the latest release:

```
pip install -U Louie
```

CHAPTER 7

Development

You can track the latest changes in Louie using the Github repo.

7.1 Using git

Clone the Louie repo using git, e.g.:

```
git clone https://github.com/11craft/louie
```

Run this command inside your git repo directory to use Louie directly from source code in that directory:

```
cd louie
pip install -e .
```

If you want to revert to the version installed in `site-packages`, you can do so:

```
pip uninstall louie
```

CHAPTER 8

Indices and tables

- genindex
- modindex
- search