PySlot Documentation

Release 1.0.0

Julien Kauffmann

February 17, 2016

Contents

1 Installation **3**

2 Basic usage **5**

3 Table of contents **7**
3.1 API 7

4 Indices and tables **9**

PySlot a dead-simple signal/slot library for Python.

CHAPTER 1

Installation

You may install it by using *pip*:

```
pip install pyslot
```

CHAPTER 2

Basic usage

PySlot provides two signal classes:

- `Signal` for basic signals in mono-threaded code.
- `ThreadSafeSignal` for cross-thread signal instance usage.

Both have the same interface and can be used like so:

```
from pyslot import Signal

def greet(name, msg):
   print("{name} says: {msg}".format(name=name, msg=msg))

signal = Signal()
signal.connect(greet)

signal.emit("alice", msg="Hi Bob !")
signal.emit("bob", msg="Hi Alice !")
```

The `connect` function takes a weak-reference to any callable that will in turn be called whenever the `emit` method gets called.

A signal can be connected to several callables, which will all be called *in their registration order*.

It is also possible to disconnect a callable from the signal by calling the `disconnect` method with that callable as the single argument. Since only a weak-reference to the callable is kept, destroying the callable will implicitely disconnect it from the signal as well.

Note: No matter what variant of the *Signal* class you use, it is **always** safe for a callable to call `disconnect`, be it for itself or for another callable.

CHAPTER 3

Table of contents

3.1 API

PySlot comes with two different signal classes:

class `pyslot.`**`Signal`**
A basic, synchronous, signal implementation.

> **Warning:** This class is **NOT** thread-safe.
> In particular, attempting to connect, disconnect or emit the signal simultaneously from different threads has unspecified behaviour.
> If you need that kind of thread-safety and can pay the cost for it, look at the *`ThreadSafeSignal`* class instead.

`connect`(*callback*)
Connects a new callback to this signal.

Parameters **`callback`** – The callback to connect.

callback will be called whenever *emit* gets called on the *Signal* instance.

A weak reference is kept, meaning that if the callback gets destroyed, it is unregistered from the signal automatically.

This design choice helps avoiding circular references in user-code.

Note: Connecting the same callback twice or more will cause the callback to be called several times per *emit* call.

You will have to call *disconnect* as many times as the *connect* call was called to unregister a callback completely.

`disconnect`(*callback*)
Disconnects a callback from this signal.

Parameters **`callback`** – The callback to disconnect.

> **Warning:** If the callback is not connected at the time of call, a `ValueError` exception is thrown.

Note: You may call *disconnect* from a connected callback.

emit(**args*, ***kwargs*)
Emit the signal.

Parameters

- **args** – The arguments.
- **kwargs** – The keyword arguments.

All the connected callbacks will be called synchronously in order of their registration.

class pyslot.**ThreadSafeSignal**
A thread-safe, synchronous, signal implementation.

Note: This class is thread-safe. You may connect, disconnect or emit the signal from different threads.

connect(*callback*)
Connects a new callback to this signal.

Parameters **callback** – The callback to connect.

callback will be called whenever *emit* gets called on the *Signal* instance.

A weak reference is kept, meaning that if the callback gets destroyed, it is unregistered from the signal automatically.

This design choice helps avoiding circular references in user-code.

Note: Connecting the same callback twice or more will cause the callback to be called several times per *emit* call.

You will have to call *disconnect* as many times as the *connect* call was called to unregister a callback completely.

disconnect(*callback*)
Disconnects a callback from this signal.

Parameters **callback** – The callback to disconnect.

Warning: If the callback is not connected at the time of call, a `ValueError` exception is thrown.

Note: You may call *disconnect* from a connected callback.

emit(**args*, ***kwargs*)
Emit the signal.

Parameters

- **args** – The arguments.
- **kwargs** – The keyword arguments.

All the connected callbacks will be called synchronously in order of their registration.

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

C

connect() (pyslot.Signal method), 7
connect() (pyslot.ThreadSafeSignal method), 8

D

disconnect() (pyslot.Signal method), 7
disconnect() (pyslot.ThreadSafeSignal method), 8

E

emit() (pyslot.Signal method), 8
emit() (pyslot.ThreadSafeSignal method), 8

S

Signal (class in pyslot), 7

T

ThreadSafeSignal (class in pyslot), 8