

---

**fastwire**

**Mar 08, 2021**



---

## Contents:

---

<b>1</b>	<b>fastwire package</b>	<b>3</b>
1.1	Module contents . . . . .	3
1.2	fastwire.signal module . . . . .	3
1.3	fastwire.wire module . . . . .	5
1.4	fastwire.container module . . . . .	7
1.5	fastwire.box module . . . . .	7
1.6	fastwire.condition module . . . . .	8
1.7	fastwire.decorate module . . . . .	9
1.8	fastwire.wired module . . . . .	10
<b>2</b>	<b>Indices and tables</b>	<b>11</b>
	<b>Python Module Index</b>	<b>13</b>
	<b>Index</b>	<b>15</b>



Fastwire helps manage communication between different classes. There are many similar packages out there, but fastwire is (trying to be) more flexible and easy to use.



### 1.1 Module contents

Created on Sun Jul 21 22:06:32 2019

@author: Reuben

### 1.2 fastwire.signal module

Created on Sat Jul 27 08:56:13 2019

@author: Reuben

Signals can be connected to multiple callable receivers. The hold weak references to those callables, so that when they are garbage collected their references are removed automatically.

**class** fastwire.signal.**Signal** (*name=None, doc=None, receiver\_limit=None, condition=None, attrs=None*)

Bases: object

A class that can emit and receive data from multiple callables.

#### Parameters

- **name** (*str*) – A name of the wire [optional]
- **doc** (*str*) – A documentation string for the wire [optional]
- **receiver\_limit** (*int*) – Limit the number of receivers [optional]
- **condition** (*Condition*) – An optional signal Condition.
- **attrs** (*dict*) – Optional dictionary of signal attributes.

**add\_condition** (*condition*)

Add a condition that the signal must pass to be received

**Parameters** `condition` (`Condition`) – A Condition instance

**attrs**

The signal documentation

**connect** (`receiver`, `**receiver_kwargs`)

Store weakref of receiver function or method to call

**Parameters**

- **receiver** (`callable`) – A callable receiver
- **kwargs** – Optional key word arguments

**Returns** A receiver id that can be used to disconnect

**Return type** float

**disconnect** (`receiver_id`)

Disconnect a receiver

**Parameters** **receiver\_id** (`int`) – The id of the receiver.

**doc**

The signal documentation

**fetch** (`**kwargs`)

Get a return value from a single supplier

**Parameters** **\*\*kwargs** – Key word arguments.

**fetch\_all** (`**kwargs`)

Get return value from all connected callables

**Parameters** **\*\*kwargs** – Key word arguments.

**Returns** The list of return values

**Return type** list

**mute** ()

Prevent receivers from receiving signals

**n**

Number of receivers

**name**

The signal name

**receivers** ()

**receivers\_present**

A boolean check if receivers are present

**remove\_condition** (`name`)

Remove a condition that the signal must pass to be received

**Parameters** **name** (`str`) – The name of the condition to remove.

**reset** ()

Reset the signal

**unmute** ()

Allow receivers to receive signals (after a mute)



---

```
class fastwire.signal.SignalBox
```

```
Bases: fastwire.box.Box
```

```
A collection of SignalContainers
```

```
signal (name=None, doc=None, attrs=None, **kwargs)
```

```
Create or get a new signal instance in the active container
```

#### Parameters

- **name** (*str*) – A name of the wire/signal [optional]
- **doc** (*str*) – A documentation string for the wire/signal [optional]
- **attrs** (*dict*) – Optional dict of signal attributes.

```
class fastwire.signal.SignalContainer (cid=None)
```

```
Bases: fastwire.container.Container
```

```
A dictionary-like collection of Signal instances
```

```
signal (name=None, doc=None, attrs=None, **kwargs)
```

```
Create or get a new signal instance
```

#### Parameters

- **name** (*str*) – A name of the wire/signal [optional]
- **doc** (*str*) – A documentation string for the wire/signal [optional]
- **attrs** (*dict*) – Optional dict of signal attributes.

```
fastwire.signal.get_signal_box (name)
```

```
fastwire.signal.signal_box (name)
```

```
fastwire.signal.signal_container (name)
```

## 1.3 fastwire.wire module

Created on Sat Jul 27 08:57:12 2019

@author: Reuben

The wire module is about simple ‘Wire’ classes. Wire instances can be connected to one, and only one, receiver (a callable). Unlike Signals, Wire instances hold normal a normal reference to the callable.

```
class fastwire.wire.Wire (name=None, doc=None, attrs=None, **kwargs)
```

```
Bases: object
```

```
A simple instance that can be connected to one receiver
```

#### Parameters

- **name** (*str*) – A name of the wire [optional]
- **doc** (*str*) – A documentation string for the wire [optional]
- **\*\*attributes** – Optional key word arguments, which are stored as attributes of the signal.

```
attrs
```

```
The wire documentation
```

**connect** (*receiver*)

Connect the wire to a callable receiver

**Parameters** **receiver** (*callable*) – A receiver called by the wire.

**disconnect** ()

Disconnect the wire from its receiver

**doc**

The wire documentation

**mute** ()

Prevent the wire from calling the receiver

**name**

The wire name

**reset** ()

Fully reset the wire, disconnecting it if required

**set\_default** (*default*)

**unmute** ()

Allow the wire to call the receiver

**class** fastwire.wire.WireBox

Bases: *fastwire.box.Box*

A collection of SignalContainers

**wire** (*name=None, doc=None, attrs=None, \*\*kwargs*)

Create or get a new signal instance in the active container

**Parameters**

- **name** (*str*) – A name of the wire/signal [optional]
- **doc** (*str*) – A documentation string for the wire/signal [optional]
- **attrs** (*dict*) – Optional dict of signal attributes.

**class** fastwire.wire.WireContainer (*cid=None*)

Bases: *fastwire.container.Container*

A dictionary-like collection of Signal instances

**wire** (*name=None, doc=None, attrs=None, \*\*kwargs*)

Create or get a new wire instance

**Parameters**

- **name** (*str*) – A name of the wire/signal [optional]
- **doc** (*str*) – A documentation string for the wire/signal [optional]
- **attrs** (*dict*) – Optional dict of signal attributes.

fastwire.wire.get\_wire\_box (*name*)

fastwire.wire.wire\_box (*name*)

fastwire.wire.wire\_container (*name*)

## 1.4 fastwire.container module

Created on Wed Jul 31 12:59:51 2019

@author: Reuben

**class** fastwire.container.Container(*signal\_cls, cid=None*)

Bases: dict

A dictionary-like collection of Signal instances

**get** (*name=None, doc=None, attrs=None, must\_exist=False, \*\*kwargs*)

Get or create a new Signal or Wire instance

### Parameters

- **name** (*str*) – A name of the wire [optional]
- **doc** (*str*) – A documentation string for the wire [optional]
- **attrs** (*dict*) – Optional dict of signal attributes.
- **must\_exist** (*bool*) – True if the signal must already exist

**mute\_all** ()

Mute all signals in the container

**reset\_all** ()

Reset all wires in the contain

**unmute\_all** ()

Unmute all signals in the container

## 1.5 fastwire.box module

Created on Wed Jul 31 12:56:25 2019

@author: Reuben

**class** fastwire.box.Box(*container\_cls*)

Bases: object

A collection of containers

It's used to allow client classes to create sets of signals for specific instances. Other classes instantiated can call an instance of this class to get signals within the appropriate set. This can help to avoid signals getting mixed up.

**active**

Return the active container id

**add** (*cid=None, activate=True, remove\_with=None*)

Add a new container referenced with cid

### Parameters

- **cid** (*int, str*) – A reference for the container
- **activate** (*bool*) – Set the container as the active one
- **remove\_with** (*object*) – An object to associate the container with. When the object is garbage collected, its container and all signals within it will also be removed. This can be useful to avoid objects accumulating in memory.

**Returns** The container

**Return type** *Container*

---

**Note:** Container id is available via container.id.

---

**clear()**

Clear all containers in the box

**containers**

Return a dictionary of containers in the box

**deactivate** (*cid=None*)

Set the active container to 'default'

**Parameters** *cid* (*int*) – An optional container id. If this id is given, the active container will only be changed if this container is currently active.

**get** (*name=None, doc=None, attrs=None, \*\*kwargs*)

Create or get a new wire/signal instance in the active container

**Parameters**

- **name** (*str*) – A name of the wire/signal [optional]
- **doc** (*str*) – A documentation string for the wire/signal [optional]
- **attrs** (*dict*) – Optional dict of signal attributes.

**get\_active()**

Return the currently active container

**get\_container** (*cid=None*)

Return the container given by a container id

**remove** (*cid*)

Remove a container

**Parameters** *cid* (*int, str*) – The container reference

**remove\_with** (*obj, cid=None*)

Set this box to be removed when an object is garbage collected

**Parameters**

- **obj** (*object*) – The object to remove the container with
- **cid** (*int*) – The optional container id to remove. Defaults to the active container.

**reset\_all()**

Reset all wires in all containers

**set\_active** (*cid*)

Set the active container

**Parameters** *cid* (*int, str*) – The container reference

## 1.6 fastwire.condition module

Created on Sat Jul 27 08:58:09 2019

@author: Reuben

Conditions apply logic to whether or not particular receivers receive particular signals. They are designed to be open, so that you can add your own condition classes with your own logic.

```
class fastwire.condition.Condition
```

Bases: object

Template class for a signal condition

**Class attributes:** name (str): A class attribute that gives their name.

**check** (*\*\*kwargs*)

The main check call - must return a boolean

All Condition classes must implement a method called 'check'. The check method is passed a set of key word arguments provided by the sender and the receiver.

**name** = 'default'

## 1.7 fastwire.decorate module

Created on Sat Jul 27 08:58:26 2019

@author: Reuben

The decorate module contains a number of decorators which assist with automatically connecting methods and functions.

```
fastwire.decorate.ensure_signal_obj(signal, box, container, receiver_limit=None)
```

```
fastwire.decorate.fn_receive(s, box=None, container=None, **receiver_kwargs)
```

A decorator to connect methods to Signal instances automatically

### Parameters

- **s** ([Signal](#)) – A Signal instance, or list of Signal instances.
- **box** ([Box](#)) – [Optional] The box for the active container.
- **container** ([Container](#)) – [Optional] The container for the signal.
- **\*\*receiver\_kwargs** – Any number of key word arguments. These are passed to any Condition instances added to the Signal instance.

```
fastwire.decorate.fn_supply(s, box=None, container=None, **receiver_kwargs)
```

A decorator to set methods to be the sole source for a Signal

### Parameters

- **s** ([Signal](#)) – A Signal instance, or list of Signal instances.
- **\*\*receiver\_kwargs** – Any number of key word arguments. These are passed to any Condition instances added to the Signal instance.

```
fastwire.decorate.receive(s, box=None, container=None, **receiver_kwargs)
```

A decorator to connect methods to Signal instances automatically

### Parameters

- **s** ([Signal](#)) – A Signal instance, or list of Signal instances.
- **box** ([Box](#)) – [Optional] The box for the active container.
- **container** ([Container](#)) – [Optional] The container for the signal.

- **\*\*receiver\_kwargs** – Any number of key word arguments. These are passed to any Condition instances added to the Signal instance.

`fastwire.decorate.supply(s, box=None, container=None, **receiver_kwargs)`

A decorator to set methods to be the sole source for a Signal

#### Parameters

- **s** ([Signal](#)) – A Signal instance, or list of Signal instances.
- **box** ([Box](#)) – [Optional] The box for the active container.
- **container** ([Container](#)) – [Optional] The container for the signal.
- **\*\*receiver\_kwargs** – Any number of key word arguments. These are passed to any Condition instances added to the Signal instance.

## 1.8 fastwire.wired module

Created on Sat Jul 27 08:58:45 2019

@author: Reuben

This module provides mix-in classes to facilitate the decoration of methods.

**class** `fastwire.wired.Wired`

Bases: `object`

The mix-in class that enables method decoration to work

## CHAPTER 2

---

### Indices and tables

---

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





### f

- `fastwire`, 3
- `fastwire.box`, 7
- `fastwire.condition`, 8
- `fastwire.container`, 7
- `fastwire.decorate`, 9
- `fastwire.signal`, 3
- `fastwire.wire`, 5
- `fastwire.wired`, 10



**A**

active (*fastwire.box.Box* attribute), 7  
add() (*fastwire.box.Box* method), 7  
add\_condition() (*fastwire.signal.Signal* method), 3  
attrs (*fastwire.signal.Signal* attribute), 4  
attrs (*fastwire.wire.Wire* attribute), 5

**B**

Box (*class in fastwire.box*), 7

**C**

check() (*fastwire.condition.Condition* method), 9  
clear() (*fastwire.box.Box* method), 8  
Condition (*class in fastwire.condition*), 9  
connect() (*fastwire.signal.Signal* method), 4  
connect() (*fastwire.wire.Wire* method), 5  
Container (*class in fastwire.container*), 7  
containers (*fastwire.box.Box* attribute), 8

**D**

deactivate() (*fastwire.box.Box* method), 8  
disconnect() (*fastwire.signal.Signal* method), 4  
disconnect() (*fastwire.wire.Wire* method), 6  
doc (*fastwire.signal.Signal* attribute), 4  
doc (*fastwire.wire.Wire* attribute), 6

**E**

ensure\_signal\_obj() (in module *fastwire.decorate*), 9

**F**

fastwire (*module*), 3  
fastwire.box (*module*), 7  
fastwire.condition (*module*), 8  
fastwire.container (*module*), 7  
fastwire.decorate (*module*), 9  
fastwire.signal (*module*), 3  
fastwire.wire (*module*), 5  
fastwire.wired (*module*), 10

fetch() (*fastwire.signal.Signal* method), 4  
fetch\_all() (*fastwire.signal.Signal* method), 4  
fn\_receive() (in module *fastwire.decorate*), 9  
fn\_supply() (in module *fastwire.decorate*), 9

**G**

get() (*fastwire.box.Box* method), 8  
get() (*fastwire.container.Container* method), 7  
get\_active() (*fastwire.box.Box* method), 8  
get\_container() (*fastwire.box.Box* method), 8  
get\_signal\_box() (in module *fastwire.signal*), 5  
get\_wire\_box() (in module *fastwire.wire*), 6

**M**

mute() (*fastwire.signal.Signal* method), 4  
mute() (*fastwire.wire.Wire* method), 6  
mute\_all() (*fastwire.container.Container* method), 7

**N**

n (*fastwire.signal.Signal* attribute), 4  
name (*fastwire.condition.Condition* attribute), 9  
name (*fastwire.signal.Signal* attribute), 4  
name (*fastwire.wire.Wire* attribute), 6

**R**

receive() (in module *fastwire.decorate*), 9  
receivers() (*fastwire.signal.Signal* method), 4  
receivers\_present (*fastwire.signal.Signal* attribute), 4  
remove() (*fastwire.box.Box* method), 8  
remove\_condition() (*fastwire.signal.Signal* method), 4  
remove\_with() (*fastwire.box.Box* method), 8  
reset() (*fastwire.signal.Signal* method), 4  
reset() (*fastwire.wire.Wire* method), 6  
reset\_all() (*fastwire.box.Box* method), 8  
reset\_all() (*fastwire.container.Container* method), 7

## S

`set_active()` (*fastwire.box.Box method*), 8  
`set_default()` (*fastwire.wire.Wire method*), 6  
`Signal` (*class in fastwire.signal*), 3  
`signal()` (*fastwire.signal.SignalBox method*), 5  
`signal()` (*fastwire.signal.SignalContainer method*), 5  
`signal_box()` (*in module fastwire.signal*), 5  
`signal_container()` (*in module fastwire.signal*), 5  
`SignalBox` (*class in fastwire.signal*), 4  
`SignalContainer` (*class in fastwire.signal*), 5  
`supply()` (*in module fastwire.decorate*), 10

## U

`unmute()` (*fastwire.signal.Signal method*), 4  
`unmute()` (*fastwire.wire.Wire method*), 6  
`unmute_all()` (*fastwire.container.Container method*),  
7

## W

`Wire` (*class in fastwire.wire*), 5  
`wire()` (*fastwire.wire.WireBox method*), 6  
`wire()` (*fastwire.wire.WireContainer method*), 6  
`wire_box()` (*in module fastwire.wire*), 6  
`wire_container()` (*in module fastwire.wire*), 6  
`WireBox` (*class in fastwire.wire*), 6  
`WireContainer` (*class in fastwire.wire*), 6  
`Wired` (*class in fastwire.wired*), 10