

# Spiff Signal Release User Documentation

Signal/event mechanism for Python

Samuel Abels

May 3, 2009

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
1.1	Why Spiff Signal? . . . . .	2
1.2	Legal Information . . . . .	2
1.3	Contact Information & Feedback . . . . .	2
<b>2</b>	<b>Overview</b>	<b>3</b>

## 1 Introduction

### 1.1 Why Spiff Signal?

*Spiff Signal* provides a simple signal/event mechanism for Python.

### 1.2 Legal Information

*Spiff Signal* and this handbook are distributed under the terms and conditions of the GNU GPL (General Public License) Version 2. You should have received a copy of the GPL along with *Spiff Signal*. If you did not, you may read it here:

<http://www.gnu.org/licenses/gpl-2.0.txt>

If this license does not meet your requirements you may contact us under the points of contact listed in the following section. Please let us know why you need a different license - perhaps we may work out a solution that works for either of us.

### 1.3 Contact Information & Feedback

If you spot any errors, or have ideas for improving *Spiff Signal* or this documentation, your suggestions are gladly accepted. We offer the following contact options:

**Google Groups:** <http://groups.google.com/group/spiff-devel/>  
**Bug tracker:** <http://code.google.com/p/spiff-signal/issues/list>  
**Phone:** +49 176 830 40288  
**Jabber:** [knipknap@jabber.org](mailto:knipknap@jabber.org)

## 2 Overview

*Spiff Signal* provides one single class only: *Trackable* implements an interface for sending and receiving signals. To send a signal an object inherits from *Trackable* and calls the *signal\_emit*-Method, as can be seen in the following code:

```
from SpiffSignal import Trackable

class WatchMe(Trackable):
    def __init__(self):
        Trackable.__init__(self)

    def do_something(self):
        self.signal_emit('did-something', 'hello world')
```

To retrieve the signal a subscriber may register with the sender using the *signal\_connect* method:

```
def my_callback(arg):
    print arg

foo = WatchMe()
foo.signal_connect('did-something', my_callback)
foo.do_something()
```

The example calls the *my\_callback* function whenever the *did\_something* signal is sent by the *WatchMe* class.

*Spiff Signal* provides additional methods for unregistering existing connections and for retrieving additional information. For a complete list of the methods supported by *Trackable* please refer to our API documentation.