
Robot Framework Sphinx Contrib Library Documentation

Release 0.4.3

asko.soukka@iki.fi

September 25, 2016

1	Examples	3
1.1	Document with embedded tests	3
1.2	Document with a screenshot	3
1.3	Document with an annotated screenshot	5
2	Getting started	7

sphinxcontrib-robotframework is a [Sphinx](#)-extension, which executes embedded [Robot Framework](#) tests during `sphinx-build`.

sphinxcontrib-robotframework can be used in [doctest](#) way to validate examples shown in documentation or with [Selenium](#) and its Robot Framework integration, [Selenium2Library](#), to generate scripted screenshots during the documentation compilation time, for CI-generated up-to-date screenshots.

Examples

1.1 Document with embedded tests

With the Robot Framework space separated format, a minimal test suite must contain the `*** Test Cases ***` header and at least one test case, like:

```
*** Test Cases ***  
  
Foo is always foo  
    Should be equal    foo    foo
```

But the `*** Test Cases ***`-header may be followed by as many tests as required, like:

```
*** Test Cases ***  
  
Foo is still foo  
    Should be equal    foo    foo  
  
Foo is never bar  
    Should not be equal    foo    bar
```

1.2 Document with a screenshot

The fun with `sphinxcontrib-robotframework` starts in using it together with `Selenium2Library`.

These packages together would allow you to navigate any website, take screenshots when required and finally embed those screenshot into this very Sphinx-documentation. All this with just `sphinx-build`:

```
*** Settings ***  
  
Library    Selenium2Library  
  
Suite Teardown    Close all browsers  
  
*** Variables ***  
  
${BROWSER}    Firefox  
  
*** Test Cases ***  
  
Capture a screenshot of RobotFramework.org  
    Open browser    http://robotframework.org/    browser=${BROWSER}  
    Capture page screenshot    robotframework.png
```

ROBOT FRAMEWORK

Generic test automation framework for
acceptance testing and ATDD

[Introduction](#) [Examples](#) [Test libraries](#) [Tools](#) [Documentation](#)

[Support & Contact](#)

Robot Framework is a generic test automation framework for acceptance testing and acceptance test-driven development (ATDD). It has easy-to-use tabular test data syntax and it utilizes the keyword-driven testing approach. Its testing capabilities can be extended by test libraries implemented either with Python or Java, and users can create new higher-level keywords from existing ones using the same syntax that is used for creating test cases.

Robot Framework project is hosted on [GitHub](#) where you can find further documentation, source code, and issue tracker. Downloads are hosted at [PyPI](#). The framework has a rich ecosystem around it consisting of various generic [test libraries](#) and [tools](#) that are developed as separate projects.

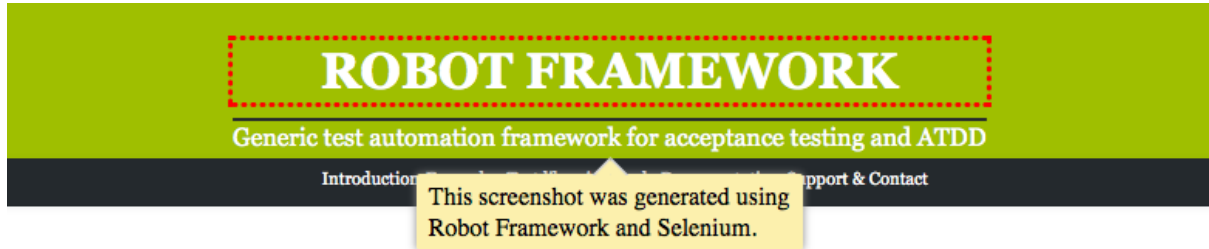
Robot Framework is operating system and application independent. The core framework is implemented using [Python](#) and runs also on [Jython](#) (JVM)

1.3 Document with an annotated screenshot

While [Selenium](#) has built-in support for capturing whole page screenshots, usually screenshots must be cropped and some times also annotated to make them useful in a documentation.

A Robot Framework library called [Selenium2Screenshots](#) provides a collection of re-usable keywords for cropping and annotating screenshots.

A cropped and annotated screenshot could look like this:



Note: The image cropping feature for `robotframework-selenium2screenshots` requires [PIL](#) or [Pillow](#).

Getting started

1. Install **sphinxcontrib-robotframework** into your virtualenv or require it as a dependency of your Sphinx-project.
2. Enable the extension and execution of embedded Robot Framework tests by adding the following lines into your Sphinx-project's `conf.py`:

```
extensions = ['sphinxcontrib_robotframework']

# Enable Robot Framework tests during Sphinx compilation
sphinxcontrib_robotframework_enabled = True

# Hide Robot Framework syntax from the Sphinx output by default
# (preferred, when you use the extension for scripted screenshots)
sphinxcontrib_robotframework_quiet = True
```

3. Write your Robot Framework tests in space separated form as contents of Docutils' `code`-directives with `robotframework-language`:

```
.. code:: robotframework

    *** Settings ***
    ...

    *** Variables ***
    ...

    *** Test cases ***
    ...
```

Each document may contain several `code`-directives, but their contents are concatenated into a single Robot Framework test suite before execution.

The output of each `code`-directive can be omitted by setting a special `:class: hidden`-option. (This is not a standard Sphinx-behavior, but a hard coded feature in **sphinxcontrib-robotframework**.)

4. Compile your documentation and see your tests being run.

Note: If you choose to use Robot Framework variables in your test cases, you can override values for those variables in your Sphinx-configuration file (`conf.py`) with:

```
sphinxcontrib_robotframework_variables = {
    "VARIABLE": "value"
}
```

When Sphinx *nitpicky* mode is enabled, failing Robot Framework run will raise Sphinx Error and leave Robot Framework log files in place.