Documentation System Release 9.7

The Sage Development Team

Jul 21, 2024

CONTENTS

1	Sage docbuild main	1
2	Documentation builders	5
3	Build options	11
4	Sphinx build script	13
5	Sphinx build configuration	15
6	Utilities	17
Py	Python Module Index	
Inc	Index	

SAGE DOCBUILD MAIN

This module defines the Sage documentation build command:

sage --docbuild [OPTIONS] DOCUMENT (FORMAT | COMMAND)

If FORMAT is given, it builds DOCUMENT in FORMAT. If COMMAND is given, it returns information about DOCUMENT.

Run sage --docbuild to get detailed explanations about arguments and options.

Positional arguments:

DOCUMENT	name of the document to build. It can be either one
	of the documents listed by -D or 'file=/path/to/FILE' to build_
→documentation	
	for this specific file.
FORMAT or COMMAND	document output format (or command)

Standard options:

-h,help	show a help message and exit
-H,help-all	show an extended help message and exit
-D,documents	list all available DOCUMENTs
-F,formats	list all output FORMATs
-C DOC,commands DOC	list all COMMANDs for DOCUMENT DOC; use 'all' to list all
-i,inherited	include inherited members in reference manual; may be slow,
→may fail for PDF output	
-u,underscore	include variables prefixed with '_' in reference
	manual; may be slow, may fail for PDF output
-j,mathjax,jsmath	ignored for backwards compatibility
no-plot	do not include graphics auto-generated using the ' plot'
→markup	
include-tests-blocks	include TESTS blocks in the reference manual
no-pdf-links	do not include PDF links in DOCUMENT 'website';
	FORMATs: html, json, pickle, web
warn-links	issue a warning whenever a link is not properly
	resolved; equivalent to 'sphinx-opts -n' (sphinx option:_
→nitpicky)	
check-nested	check picklability of nested classes in DOCUMENT 'reference'
no-prune-empty-dirs	do not prune empty directories in the documentation sources
-N,no-colors	do not color output; does not affect children
-q,quiet	work quietly; same asverbose=0
-v LEVEL,verbose LEVEL	report progress at LEVEL=0 (quiet), 1 (normal), 2
	(continues on next nage)

```
(info), or 3 (debug); does not affect children
-o DIR, --output DIR
→this directory(info), or 3 (debug); does not affect children
if DOCUMENT is a single file ('file=...'), write output to_
```

Advanced options:

-S OPTS,sphinx-opts OPTS	pass comma-separated OPTS to sphinx-build; must
	precede OPTS with '=', as in '-S=-q,-aE' or '-S="-q,-aE"'
-U,update-mtimes	before building reference manual, update
	modification times for auto-generated reST files
-k,keep-going	do not abort on errors but continue as much as
	possible after an error
all-documents ARG	if ARG is 'reference', list all subdocuments of
	en/reference. If ARG is 'all', list all main documents

class sage_docbuild.__main__.IntersphinxCache

Bases: object

Replace sphinx.ext.intersphinx.fetch_inventory by an in-memory cached version.

fetch_inventory(app, uri, inv)

Return the result of sphinx.ext.intersphinx.fetch_inventory() from a cache if possible. Otherwise, call sphinx.ext.intersphinx.fetch_inventory() and cache the result.

sage_docbuild.__main__.format_columns(lst, align='<', cols=None, indent=4, pad=3, width=80)
Utility function that formats a list as a simple table and returns a Unicode string representation.</pre>

The number of columns is computed from the other options, unless it's passed as a keyword argument. For help on Python's string formatter, see

https://docs.python.org/library/string.html#format-string-syntax

```
sage_docbuild.__main__.get_formats()
```

Return a list of output formats the Sage documentation builder will accept on the command-line.

```
sage_docbuild.__main__.help_commands(name='all')
```

Append and return a tabular list of commands, if any, the Sage documentation builder can run on the indicated document. The default is to list all commands for all documents.

sage_docbuild.__main__.help_description(s=", compact=False)
Append and return a brief description of the Sage documentation builder.

If 'compact' is False, the function adds a final newline character.

- sage_docbuild.__main__.help_documents()
 Append and return a tabular list of documents, including a shortcut 'all' for all documents, available to the Sage
 documentation builder.
- sage_docbuild.__main__.help_examples(s=")
 Append and return some usage examples for the Sage documentation builder.

```
sage_docbuild.__main__.help_formats()
```

Append and return a tabular list of output formats available to the Sage documentation builder.

Bases: argparse.Action

Print an extended help message for the Sage documentation builder and exits.

Bases: argparse.Action

Print a help message for the Sage documentation builder.

The message includes command-line usage and a list of options. The message is printed only on the first call. If error is True during this call, the message is printed only if the user hasn't requested a list (e.g., documents, formats, commands).

```
sage_docbuild.__main__.help_usage(s=", compact=False)
```

Append and return a brief usage message for the Sage documentation builder.

If 'compact' is False, the function adds a final newline character.

```
class sage_docbuild.___main__.help_wrapper(option_strings, dest, nargs=None, const=None,
```

default=None, type=None, choices=None, required=False, help=None, metavar=None)

Bases: argparse.Action

A helper wrapper for command-line options to the Sage documentation builder that print lists, such as document names, formats, and document-specific commands.

sage_docbuild.__main__.main()

```
sage_docbuild.__main__.setup_logger(verbose=1, color=True)
Set up a Python Logger instance for the Sage documentation builder.
```

The optional argument sets logger's level and message format.

EXAMPLES:

```
sage: from sage_docbuild.__main__ import setup_logger, logger
sage: setup_logger()
sage: type(logger)
<class 'logging.Logger'>
```

sage_docbuild.__main__.setup_parser()

Set up and return a command-line ArgumentParser instance for the Sage documentation builder.

DOCUMENTATION BUILDERS

This module is the starting point for building documentation, and is responsible to figure out what to build and with which options. The actual documentation build for each individual document is then done in a subprocess call to Sphinx, see *builder_helper()*. Note that

- The builders are configured with build_options.py;
- The Sphinx subprocesses are configured in conf.py.

DocBuilder is the base class of all Builders. It has builder helpers html(), latex(), pdf(), inventory(), etc, which are invoked depending on the output type. Each type corresponds with the Sphinx builder format, except that pdf is Sphinx latex builder plus compiling latex to pdf. Note that Sphinx inventory builder is not native to Sphinx but provided by Sage. See sage_docbuild/ext/inventory_builder.py. The Sphinx inventory builder is a dummy builder with no actual output but produces doctree files in local/share/doctree and inventory.inv inventory files in local/share/inventory.

The reference manual is built in two passes, first by *ReferenceBuilder* with inventory output type and secondly with ``html`` output type. The *ReferenceBuilder* itself uses *ReferenceTopBuilder* and *ReferenceSubBuilder* to build subcomponents of the reference manual. The *ReferenceSubBuilder* examines the modules included in the subcomponent by comparing the modification times of the module files with the times saved in local/share/doctree/reference.pickle from the previous build. Then new rst files are generated for new and updated modules. See get_new_and_updated_modules().

After trac ticket #31948, when Sage is built, *ReferenceBuilder* is not used and its responsibility is now taken by the Makefile in \$SAGE_ROOT/src/doc.

class sage_docbuild.builders.AllBuilder

Bases: object

A class used to build all of the documentation.

get_all_documents()

Return a list of all of the documents.

A document is a directory within one of the language subdirectories of SAGE_DOC_SRC specified by the global LANGUAGES variable.

EXAMPLES:

```
sage: from sage_docbuild.builders import AllBuilder
sage: documents = AllBuilder().get_all_documents()
sage: 'en/tutorial' in documents # optional - sage_spkg
True
sage: documents[0] == 'en/reference'
True
```

```
class sage_docbuild.builders.DocBuilder(name, lang='en')
```

Bases: object

INPUT:

- name the name of a subdirectory in SAGE_DOC_SRC, such as 'tutorial' or 'bordeaux_2008'
- lang (default "en") the language of the document.

changes(*args, **kwds)

clean(*args)

html(*args, **kwds)

htmlhelp(*args, **kwds)

inventory(*args, **kwds)

json(*args, **kwds)

latex(*args, **kwds)

linkcheck(*args, **kwds)

pdf()

Build the PDF files for this document.

This is done by first (re)-building the LaTeX output, going into that LaTeX directory, and running 'make all-pdf' there.

EXAMPLES:

```
sage: from sage_docbuild.builders import DocBuilder
sage: b = DocBuilder('tutorial')
sage: b.pdf() #not tested
```

pickle(*args, **kwds)

```
web(*args, **kwds)
```

```
class sage_docbuild.builders.ReferenceBuilder(name, lang='en')
```

Bases: sage_docbuild.builders.AllBuilder

This class builds the reference manual. It uses DocBuilder to build the top-level page and ReferenceSubBuilder for each sub-component.

get_all_documents(refdir)

Return a list of all reference manual components to build.

We add a component name if it's a subdirectory of the manual's directory and contains a file named 'index.rst'.

We return the largest component (most subdirectory entries) first since they will take the longest to build.

EXAMPLES:

...,
'reference/valuations']

class sage_docbuild.builders.ReferenceSubBuilder(*args, **kwds)

Bases: sage_docbuild.builders.DocBuilder

This class builds sub-components of the reference manual. It is responsible for making sure that the auto generated reST files for the Sage library are up to date.

When building any output, we must first go through and check to see if we need to update any of the autogenerated reST files. There are two cases where this would happen:

- 1. A new module gets added to one of the toctrees.
- 2. The actual module gets updated and possibly contains a new title.

auto_rest_filename(module_name)

Return the name of the file associated to a given module

EXAMPLES:

cache_filename()

Return the filename where the pickle of the reference cache is stored.

clean_auto()

Remove all autogenerated reST files.

get_all_included_modules()

Return an iterator for all modules which are included in the reference manual.

get_all_rst_files(exclude_sage=True)

Return an iterator for all rst files which are not autogenerated.

get_cache()

Retrieve the reference cache which contains the options previously used by the reference builder.

If it doesn't exist, then we just return an empty dictionary. If it is corrupted, return an empty dictionary.

get_modified_modules()

Return an iterator for all the modules that have been modified since the documentation was last built.

get_module_docstring_title(module_name)

Return the title of the module from its docstring.

get_modules(filename)

Given a filename for a reST file, return an iterator for all of the autogenerated reST files that it includes.

get_new_and_updated_modules()

Return an iterator for all new and updated modules that appear in the toctrees, and remove obsolete old modules.

get_sphinx_environment()

Return the Sphinx environment for this project.

get_unincluded_modules()

Return an iterator for all the modules in the Sage library which are not included in the reference manual.

print_included_modules()

Print all of the modules that are included in the Sage reference manual.

print_modified_modules()

Print a list of all the modules that have been modified since the documentation was last built.

print_new_and_updated_modules()

Print all the modules that appear in the toctrees that are newly included or updated.

print_unincluded_modules()

Print all of the modules which are not included in the Sage reference manual.

save_cache()

Pickle the current reference cache for later retrieval.

update_mtimes()

Update the modification times for reST files in the Sphinx environment for this project.

write_auto_rest_file(module_name)

Write the autogenerated reST file for module_name.

class sage_docbuild.builders.ReferenceTopBuilder(*args, **kwds)

Bases: sage_docbuild.builders.DocBuilder

This class builds the top-level page of the reference manual.

pdf()

Build top-level document.

class sage_docbuild.builders.SingleFileBuilder(path)

Bases: sage_docbuild.builders.DocBuilder

This is the class used to build the documentation for a single user-specified file. If the file is called 'foo.py', then the documentation is built in DIR/foo/ if the user passes the command line option "-o DIR", or in DOT_SAGE/ docbuild/foo/ otherwise.

class sage_docbuild.builders.WebsiteBuilder(name, lang='en')

Bases: sage_docbuild.builders.DocBuilder

clean()

When we clean the output for the website index, we need to remove all of the HTML that were placed in the parent directory.

In addition, remove the index file installed into the root doc directory.

html()

After we have finished building the website index page, we copy everything one directory up.

In addition, an index file is installed into the root doc directory.

sage_docbuild.builders.build_many(target, args, processes=None)

Thin wrapper around $sage_docbuild.utils.build_many$ which uses the docbuild settings NUM_THREADS and ABORT_ON_ERROR.

sage_docbuild.builders.build_other_doc(args)

sage_docbuild.builders.build_ref_doc(args)

sage_docbuild.builders.builder_helper(type)

Return a function which builds the documentation for output type type.

sage_docbuild.builders.get_builder(name)

Return an appropriate Builder object for the document name.

DocBuilder and its subclasses do all the real work in building the documentation.

THREE

BUILD OPTIONS

This module defines options for building Sage documentation.

SPHINX BUILD SCRIPT

This is Sage's version of the sphinx-build script. We redirect stdout and stderr to our own logger, and remove some unwanted chatter.

```
class sage_docbuild.sphinxbuild.SageSphinxLogger(stream, prefix)
```

Bases: object

This implements the file object interface to serve as sys.stdout/sys.stderr replacement.

close()

flush()

isatty()

raise_errors()

Raise an exceptions if any errors have been found while parsing the Sphinx output.

EXAMPLES:

```
sage: from sys import stdout
sage: from sage_docbuild.sphinxbuild import SageSphinxLogger
sage: logger = SageSphinxLogger(stdout, "doctesting")
sage: logger._log_line("This is a SEVERE error\n")
[doctestin] This is a SEVERE error
sage: logger.raise_errors()
Traceback (most recent call last):
....
OSError: This is a SEVERE error
```

write(str)

writelines(sequence)

sage_docbuild.sphinxbuild.runsphinx()

sage_docbuild.sphinxbuild.term_width_line(text)

SPHINX BUILD CONFIGURATION

This file contains configuration needed to customize Sphinx input and output behavior.

sage_docbuild.conf.add_page_context(app, pagename, templatename, context, doctree)

- sage_docbuild.conf.call_intersphinx(app, env, node, contnode)
 Call intersphinx and make links between Sage manuals relative.
- sage_docbuild.conf.check_nested_class_picklability(app, what, name, obj, skip, options)
 Print a warning if pickling is broken for nested classes.
- sage_docbuild.conf.debug_inf(app, message)
- sage_docbuild.conf.find_sage_dangling_links(app, env, node, contnode)
 Try to find dangling link in local module imports or all.py.
- sage_docbuild.conf.nitpick_patch_config(app)
 Patch the default config for nitpicky

Calling path_config ensure that nitpicky is not considered as a Sphinx environment variable but rather as a Sage environment variable. As a consequence, changing it doesn't force the recompilation of the entire documentation.

```
sage_docbuild.conf.setup(app)
```

```
sage_docbuild.conf.skip_member(app, what, name, obj, skip, options)
```

To suppress Sphinx warnings / errors, we

- Don't include [aliases of] builtins.
- Don't include the docstring for any nested class which has been inserted into its module by sage.misc. NestedClassMetaclass only for pickling. The class will be properly documented inside its surrounding class.
- Optionally, check whether pickling is broken for nested classes.
- Optionally, include objects whose name begins with an underscore ('_'), i.e., "private" or "hidden" attributes, methods, etc.

Otherwise, we abide by Sphinx's decision. Note: The object obj is excluded (included) if this handler returns True (False).

SIX

UTILITIES

exception sage_docbuild.utils.RemoteException(tb) Bases: Exception

Raised if an exception occurred in one of the child processes.

class sage_docbuild.utils.RemoteExceptionWrapper(exc)
 Bases: object

Used by child processes to capture exceptions thrown during execution and report them to the main process, including the correct traceback.

exception sage_docbuild.utils.WorkerDiedException(message, original_exception=None) Bases: RuntimeError

Raised if a worker process dies unexpected.

sage_docbuild.utils.build_many(target, args, processes=None)

Map a list of arguments in args to a single-argument target function target in parallel using multiprocessing.cpu_count() (or processes if given) simultaneous processes.

This is a simplified version of multiprocessing.Pool.map from the Python standard library which avoids a couple of its pitfalls. In particular, it can abort (with a RuntimeError) without hanging if one of the worker processes unexpectedly dies. It also has semantics equivalent to maxtasksperchild=1; that is, one process is started per argument. As such, this is inefficient for processing large numbers of fast tasks, but appropriate for running longer tasks (such as doc builds) which may also require significant cleanup.

It also avoids starting new processes from a pthread, which results in at least two known issues:

- On versions of Cygwin prior to 3.0.0 there were bugs in mmap handling on threads (see trac ticket #27214#comment:25).
- When PARI is built with multi-threading support, forking a Sage process from a thread leaves the main Pari interface instance broken (see trac ticket #26608#comment:38).

In the future this may be replaced by a generalized version of the more robust parallel processing implementation from sage.doctest.forker.

EXAMPLES:

```
sage: from sage_docbuild.utils import build_many
sage: def target(N):
....: import time
....: time.sleep(float(0.1))
....: print('Processed task %s' % N)
sage: _ = build_many(target, range(8), processes=8)
Processed task ...
```

```
Processed task ...
```

Unlike the first version of build_many which was only intended to get around the Cygwin bug, this version can also return a result, and thus can be used as a replacement for multiprocessing.Pool.map (i.e. it still blocks until the result is ready):

```
sage: def square(N):
....: return N * N
sage: build_many(square, range(100))
[0, 1, 4, 9, ..., 9604, 9801]
```

If the target function raises an exception in any of the workers, build_many raises that exception and all other results are discarded. Any in-progress tasks may still be allowed to complete gracefully before the exception is raised:

```
sage: def target(N):
....: import time, os, signal
....: if N == 4:
....: # Task 4 is a poison pill
....: else:
....: else:
....: time.sleep(float(0.5))
....: print('Processed task %s' % N)
```

Note: In practice this test might still show output from the other worker processes before the poison-pill is executed. It may also display the traceback from the failing process on stderr. However, due to how the doctest runner works, the doctest will only expect the final exception:

```
sage: build_many(target, range(8), processes=8)
Traceback (most recent call last):
...
raise ZeroDivisionError("rational division by zero")
ZeroDivisionError: rational division by zero
...
raise worker_exc.original_exception
ZeroDivisionError: rational division by zero
```

Similarly, if one of the worker processes dies unexpectedly otherwise exits non-zero (e.g. killed by a signal) any in-progress tasks will be completed gracefully, but then a RuntimeError is raised and pending tasks are not started:

```
sage: def target(N):
....: import time, os, signal
....: if N == 4:
....: # Task 4 is a poison pill
os.kill(os.getpid(), signal.SIGKILL)
else:
```

```
....: time.sleep(float(0.5))
....: print('Processed task %s' % N)
sage: build_many(target, range(8), processes=8)
Traceback (most recent call last):
....
WorkerDiedException: worker for 4 died with non-zero exit code -9
```

PYTHON MODULE INDEX

S

sage_docbuild.__main__, 1
sage_docbuild.build_options, 11
sage_docbuild.builders, 5
sage_docbuild.conf, 15
sage_docbuild.sphinxbuild, 13
sage_docbuild.utils, 17

INDEX

Α

add_page_context() (in module sage_docbuild.conf), 15 AllBuilder (class in sage_docbuild.builders), 5 auto_rest_filename() (sage_docbuild.builders.ReferenceSubBuilder method), 7

В

build_many() (in module sage_docbuild.builders), 8 build_many() (in module sage_docbuild.utils), 17 build_other_doc() (in module sage_docbuild.builders), 8 build_ref_doc() (in module sage_docbuild.builders), 8 builder_helper() (in module sage_docbuild.builders), 8

С

<pre>cache_filename() (sage_docbuild.builders.ReferenceSub method), 7</pre>	ק ו ge
call_intersphinx() (in module sage_docbuild.conf),	90
15	ge
changes() (<i>sage_docbuild.builders.DocBuilder</i>	
method), 6	ge
<pre>check_nested_class_picklability() (in module</pre>	ge
sage_docbuild.conf), 15	
clean() (sage_docbuild.builders.DocBuilder method), 6	
clean() (sage_docbuild.builders.WebsiteBuilder	ge
method), 8	
clean_auto() (sage_docbuild.builders.ReferenceSubBuild	ler
method), 7	ge
<pre>close() (sage_docbuild.sphinxbuild.SageSphinxLogger</pre>	
method), 13	ge
	-
D	
<pre>debug_inf() (in module sage_docbuild.conf), 15</pre>	ge
DocBuilder (class in sage_docbuild.builders), 5	-
bocburruer (cluss in suge_uocoulla.oullaers), 5	

F

fetch_inventory() (sage_docbuild.__main__.Intersph *method*), 2

<pre>find_sage_dangling_links()</pre>	(in	module
sage_docbuild.conf), 15		
<pre>flush() (sage_docbuild.sphinxbuild.SageSphinxLogger</pre>		
<i>method</i>), 13		

format_columns() module (in sage_docbuild.__main__), 2

G

	get_all	_documents()
	-	(sage_docbuild.builders.AllBuilder method), 5
	get_all	_documents()
	-	(sage_docbuild.builders.ReferenceBuilder
		method), 6
	get_all	_included_modules()
	-	(sage_docbuild.builders.ReferenceSubBuilder
		method), 7
	get_all	_rst_files()
	-	(sage_docbuild.builders.ReferenceSubBuilder
		method), 7
uł	b g atabui	<pre>lder() (in module sage_docbuild.builders), 8</pre>
	get_cac	he() (sage_docbuild.builders.ReferenceSubBuilder
		method), 7
	get_doc	<pre>uments() (in module sage_docbuild.builders),</pre>
		8
	get_for	<pre>mats() (in module sage_docbuildmain), 2</pre>
	get_mod	ified_modules()
		(sage_docbuild.builders.ReferenceSubBuilder
		method), 7
	get_mod	ule_docstring_title()
		(sage_docbuild.builders.ReferenceSubBuilder
ild	der	method), 7
	get_mod	ules() (sage_docbuild.builders.ReferenceSubBuilder
		method), 7
	get_new	_and_updated_modules()
		(sage_docbuild.builders.ReferenceSubBuilder
		<i>method</i>), 7
	get_sph	inx_environment()
		(sage_docbuild.builders.ReferenceSubBuilder
		<i>method</i>), 7
	get_uni	ncluded_modules()
ir	ıxCache	(sage_docbuild.builders.ReferenceSubBuilder
		method), 7

Н

help_commands()	(in	module
sage_docbuild	_main), 2	
help_description()	(in	module
sage_docbuild	<i>main</i>), 2	
help_documents()	(in	module
sage_docbuild	_main_), 2	
help_examples()	(in	module
sage_docbuild	<i>main</i>), 2	
help_formats() (in mode	ule sage_docbuil	dmain),
2	Ū.	
help_message_long	(class	in
sage_docbuild	<i>main</i>), 2	
help_message_short	(class	in
sage_docbuild	<i>main</i>), 2	
<pre>help_usage() (in module</pre>		_ <i>main</i>), 3
help_wrapper (class in sa	ige_docbuildn	nain), 3
html() (sage_docbuild.bui	-	
html() (sage_doc		
method), 8		
htmlhelp() (sage_	docbuild.builder	s.DocBuilder
method), 6		
1		

L

IntersphinxCache (class in sage_docbuild.__main__), 2 inventory() (sage_docbuild.builders.DocBuilder method), 6 isatty() (sage_docbuild.sphinxbuild.SageSphinxLogger method), 13

J

json() (sage docbuild.builders.DocBuilder method), 6

L

latex() (sage_docbuild.builders.DocBuilder method), 6 linkcheck() (sage docbuild.builders.DocBuilder *method*), 6

Μ

main() (in module sage docbuild. main), 3 module sage_docbuild.__main__,1 sage_docbuild.build_options, 11 sage_docbuild.builders, 5 sage_docbuild.conf, 15 sage_docbuild.sphinxbuild, 13 sage_docbuild.utils, 17

Ν

nitpick_patch_config() (in module sage_docbuild.conf), 15

Ρ

<pre>pdf() (sage_docbuild.builders.DocBuilder method), 6</pre>
pdf() (sage_docbuild.builders.ReferenceTopBuilder
<i>method</i>), 8
<pre>pickle() (sage_docbuild.builders.DocBuilder method),</pre>
6
<pre>print_included_modules()</pre>
(sage_docbuild.builders.ReferenceSubBuilder
method), 7
<pre>print_modified_modules()</pre>
(sage_docbuild.builders.ReferenceSubBuilder
method), 8
<pre>print_new_and_updated_modules()</pre>
(sage_docbuild.builders.ReferenceSubBuilder
method), 8
<pre>print_unincluded_modules()</pre>
(sage_docbuild.builders.ReferenceSubBuilder
method), 8
К

<pre>raise_errors() (sage_docbuild.</pre>	sphinxbuild.Sag	geSphinxLogger	
<i>method</i>), 13			
ReferenceBuilder (class in sage_docbuild.builders), 6			
ReferenceSubBuilder (class in			
sage_docbuild.builders),	, 7		
ReferenceTopBuilder	(class	in	
sage_docbuild.builders), 8			
RemoteException, 17			
RemoteExceptionWrapper	(class	in	
sage_docbuild.utils), 17			
<pre>runsphinx() (in module sage_down)</pre>	cbuild.sphinxbu	<i>ild</i>), 13	

S

sage_docbuild.__main__ module, 1 sage_docbuild.build_options module, 11 sage_docbuild.builders module, 5 sage_docbuild.conf module, 15 sage_docbuild.sphinxbuild module, 13 sage_docbuild.utils module, 17 SageSphinxLogger (class in sage_docbuild.sphinxbuild), 13 save_cache() (sage_docbuild.builders.ReferenceSubBuilder method), 8 set_intersphinx_mappings() (in module sage_docbuild.conf), 15 setup() (in module sage_docbuild.conf), 15 setup_logger() (in module sage_docbuild.__main__), 3

skip_member() (in module sage_docbuild.conf), 15

Т

term_width_line() (in module sage_docbuild.sphinxbuild), 13

U

W