RiveScript Coverage Plugin Documentation

Release 0.1.0

Joe Cool

Mar 29, 2020

Contents

A plug-in to measure code coverage in RiveScript with python

CHAPTER 1

Features

The RiveScript Coverage Plugin is a plugin for Coverage.py which extends that package to measure code coverage for RiveScript files. It uses code analysis tools and debug hooks provided by the RiveScript interpreter to determine which lines are executable, and which have been executed. It supports CPython version 3.6 and above and plugs into Coverage.py version 5.0 and above. It requires RiveScript 1.14.9 or above.

Documentation is on Read The Docs. Code repository and issue tracker are on GitHub.

CHAPTER 2

Getting Started

1. Use pip to install:

```
$ pip install rivescript_coverage_plugin
```

2. Create or edit your `.coveragerc` file and add this:

```
[run]
plugins = rivescript_coverage_plugin
```

3. Run the `coverage` command or `pytest` with the `--cov` option and your RiveScript files will automatically be included in the coverage analysis and subsequent report generation.

Note that just like with Python coverage, RiveScript files that are not executed at all will not be part of your coverage report. To add them, use the `source = .` or other more specific source specifier in the `[run]` section of your `.coveragerc` file or the `--source` command line option. See the Coverage Documentation section "Specifying source files" for more information on this.

2.1 More Information:

2.1.1 Installation

At the command line either via easy_install or pip:

```
$ easy_install rivescript_coverage_plugin
$ pip install rivescript_coverage_plugin
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv rivescript_coverage_plugin
$ pip install rivescript_coverage_plugin
```

2.1.2 Usage

To use RiveScript Coverage Plugin in a project, create or edit your `.coveragerc` file and add this:

```
[run]
plugins = rivescript_coverage_plugin
```

Now run the `coverage` command or `pytest` with the `--cov` option and your RiveScript files will automatically be included in the coverage analysis and subsequent report generation.

Note that just like with Python coverage, RiveScript files that are not executed at all will not be part of your coverage report. To add them, use the `source = .` or other more specific source specifier in the `[run]` section of your `.coveragerc` file or the `--source` command line option. See the *Coverage Documentation* section "Specifying source files" for more information on this.

2.1.3 Options

The options provided by the RiveScript Coverage Plugin are designed to help troubleshoot the plugin itself and should not be needed by normal users. For completeness, they are documented here. The options go in your `.coveragerc` file and are specified as follows:

```
[rivescript_coverage_plugin]
show_startup = True
show_parsing = True
show_tracing = True
clean_rs_objects = False
capture_streams = False
```

The default values are the opposite of what I show above. The options are defined as follows:

show_startup Show information about the plugin's startup sequence is shown, including version information and command line arguments. This is False by default.

show_parsing Show information about which lines of the RiveScript files are executable and which are not. This is False by default.

show_tracing Generate a complete trace of the execution of the RiveScript interpreter, including which lines are being marked as executed as we interpret the Debug output of the RiveScript interpreter. This is False by default.

clean_rs_objects Preserve the `_rs_objects_` temporary directory, where the RiveScript Coverage Plug creates files representing each `< object NAME python` in the RiveScript in order to trace it's execution. This directory will contain a `rs_obj_NAME.py` file for each object named `NAME` in your RiveScript. This is True by default, which means this directory will be removed after analysis.

capture_streams Capture coverage when RiveScript is dynamically created using the `rs.stream` method. In order to preserve these streams of RiveScript, a directory `_rs_streams_` is created and each stream is identified in a file named `s-SOURCEFILE1_LINENO1-SOURCEFILE2_LINENO2.rive`. Here, `SOURCEFILE1` identifies the python source file containing the direct caller of `rs.stream`, and `SOURCEFILE2` identifies a parent caller that is not located in the same source file. The `LINENOs` identify the line numbers where the call(s) occur. If multiple calls occur at the same place in the source code with different RiveScript strings being passed, then a 3-digit occurence count will be appended to the filename. This directory of streams is left in place after coverage analysis exits, so that when you run `coverage html`, it will be able to annotate the files with coverage information. If this is specified as `False`, then no coverage is captured for streams. (Since v1.1.0)

2.1.4 Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

Types of Contributions

Report Bugs

Report bugs at https://github.com/snoopyjc/rivescript_coverage_plugin/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" is open to whoever wants to implement it.

Implement Features

Look through the GitHub issues for features. Anything tagged with "feature" is open to whoever wants to implement it.

Write Documentation

RiveScript Coverage Plugin could always use more documentation, whether as part of the official RiveScript Coverage Plugin docs, in docstrings, or even on the web in blog posts, articles, and such.

Submit Feedback

The best way to send feedback is to file an issue at https://github.com/snoopyjc/rivescript_coverage_plugin/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

Get Started!

Ready to contribute? Here's how to set up *rivescript_coverage_plugin* for local development.

1. Fork the *rivescript_coverage_plugin* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/rivescript_coverage_plugin.git
```

3. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

Now you can make your changes locally.

4. When you're done making changes, check that your changes pass unit tests, including testing other Python versions with tox:

```
$ tox
```

To get tox, just pip install it.

5. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

6. Submit a pull request through the GitHub website.

Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 3.6, 3.7, and for PyPy. Check https://travis-ci.org/snoopyjc/rivescript_coverage_plugin under pull requests for active pull requests or run the `tox` command and make sure that the tests pass for all supported Python versions.

Tips

To run a subset of tests:

```
$ test/testrcp.sh
```

2.1.5 Credits

Development Lead

- Joe Cool <snoopyjc@gmail.com>

Contributors

None yet. Why not be the first?

2.1.6 History

0.1.0 (2020-01-16)

- First release on PyPI.

0.2.0 (2020-01-17)

- Fix Issues #1-#4

0.2.1 (2020-01-19)

- Fix Issue #5

0.2.2 (2020-01-19)

- Fix Issue #6

0.2.3 (2020-02-23)

- Fix Issue #7, #8

1.0.0 (2020-02-28)

- Add syntax highlighting, fixup some documentation issues, and fix issues #9-#14.

1.1.0 (2020-03-29)

- Fix issues #15-#18 and #21.

2.2 Feedback

If you have any suggestions or questions about **RiveScript Coverage Plugin** feel free to email me at snoopyjc@gmail.com.

If you encounter any errors or problems with **RiveScript Coverage Plugin**, please let me know! Open an Issue at the GitHub http://github.com/snoopyjc/rivescript_coverage_plugin main repository.