Pynguin: Automated Unit Test Generation for Python

Lukasczyk, Kroiß, F. & Fraser. An empirical study of automated unit test generation for Python. *Empir Software Eng* **28** (2023)

Doi: <https://doi.org/10.1007/s10664-022-10248-w>

Pynguin – very easy to use tool

Install the tool: Open project for which the tool will be used, open terminal and install package: pip install pyguin

Access Environment Variables and create a new variable: PYNGUIN\_DANGER\_AWARE set to 1. Without doing this, when running the command to generate tests, a console error will be raised.

A screenshot of a computer

AI-generated content may be incorrect.

+ also some additional dependencies (pydot and graphviz):

A screenshot of a computer

AI-generated content may be incorrect.

Create a “tests” directory, where all the tests will be generated after running the pynguin command

--project-path .:path of the project (“.” Because it is inside the current directory)

--output-path ./tests: path to the folder where the generated tests will be placed in the end

--module-name: the name of the file that you wish to creates tests for

Advantage

-sppeed due to the fact that it generates tests automatically, without the need of the developer to write its unit tests

Usage

Run command: “pynguin –project-path . –output-path ./tests –module-name main”

A screenshot of a computer

AI-generated content may be incorrect.

We can see that 2 files with tests were generated inside tests folder, one of them is for successful testcases, the other for failed testcases. It also created one folder of eventual reports in case the testcases are more complex.