

Index

■ A

add() method, 35
assertEqual() method, 35, 55
assertRaises() method, 60–62
assertTrue(), 55
Automated unit testing, 20

■ B

Benevolent Dictator for Life, 1

■ C

Coding and file naming
 conventions, 101–102

■ D, E, F

Docstrings, 20
 advantage, 20
 in Python, 21–24
Doctest, 19, 24–25, 29, 101
 advantages and disadvantages, 28–29
 failing tests, 26–27
 separate test file, 27–28

■ G, H

Geany, 14
Guido van Rossum at the Centrum
 Wiskunde & Informatica, 1

■ I, J, K

id() methods, 56
inspect.stack()[0][3] method, 34
Integrated development environment
 (IDE), 12

■ L, M

Lazy loading, 85

■ N

National Research Institute for
 Mathematics and
 Computer Science, 1
Nose, Python, 65, 101
 advantages over unittest, 83–84
 disadvantages, 84
 fixtures for classes, modules, and
 methods, 70–72
 alternate names, 75
 assert_equals() method, 75–76
 for functions, 72–74
 for packages, 74
 getting help, 68
 getting started with, 66
 installing
 on Linux OS, 65
 on MacOS and Windows, 66
 pytest support, 93
 report generation
 color output in console, 82–83
 HTML reports, creating, 81
 running unittest tests, 83
 XML reports, creating, 80
 test case, 67
 test discovery, 69–70
 testing code, organizing, 68–69
 testing tools, 77
 ok_ and eq_, 77–78
 @raises() decorator, 78–79
 @timed() decorator, 79–80
 test modules, 67–68
 verifying installation, 66
Nose2, 65, 84–85, 87, 101

■ **O**

OS schedulers, 101

■ **P, Q, R**

PyCharm, 15

pytest, 85, 87

- command-line options, 98
 - generating plain result, 99
 - help, 98
 - JUnit-style logs, 99
 - profiling test execution
 - duration, 99
 - stopping after the first (or n)
 - failures, 98
 - test report to online pastebin
 - service, 99

fixtures, 93–96

 pytest.raises(), 97–98

 scope, 96–97

overview, 87

with py.test command, 89

simple test, 88

support for unittest and nose, 93

TDD with, 102–107

test class and package, 90–91

test discovery, 91

xUnit-style of fixtures, 91–92

Python

community support, 5

docstring in, 21–24

easy to learn, 2

easy to maintain, 3

easy to read, 3

extensible, 4

extensive libraries, 4

Geany, 14

high-level language, 3

history, 1

IDLE, 13

installation

 debian, ubuntu, and derivatives, 7

 Fedora and CentOS, 8

 linux, 7

 MacOS X, 8

 windows, 8–9, 11

interactive mode, 11

interpreted, 3

memory management, 5

normal mode, 12

object-oriented programming
paradigms, 4

open source, 3

portable, 3

powerful, 5

PyCharm, 15

PyDev Plugin for Eclipse, 14

Python 2 *vs.* Python 3, 5–7

Python 3, 5

rapid prototyping, 4

robust, 4

simple, 2

PyUnit, 31

■ **S**

setUpModule() method, 38

shortDescription() method, 56

Software testing, 19

automated unit testing, 20

docstrings, 20

 advantage, 20

 in Python, 21–24

doctest, 24–25

 advantages and disadvantages,

 28–29

 failing tests, 26–27

 separate test file, 27–28

test automation, 19

unit testing, 19

■ **T**

tearDownModule() methods, 38

Test automation, 19

Test discovery, coding and file naming
conventions, 101–102

Test-driven development (TDD), 102–107

Test module, 36

■ **U**

Unittest/unit testing, 19, 29, 101

advantages of nose over, 83

assertions, 55

assertRaises(), 60–62

benefits of automated, 20

coding conventions, 54–55

command-line options and help,
42–46

creation, test package, 46–47

- development and test code
 - separate directories, 49–50, 52–53
 - single directory, 48–49
- exceptions, test case, 59
- failing a test, 57–58
- organizing code, 48
- Python libraries, 62
- pytest support, 93
- PyUnit, 31
- test discovery, 53–54
- test execution, 40–42
- test file/module, 36–37
- test fixtures, 37–38

- test methods, 34
- `unittest.main()`, 39–40
- usage, 32–33
- useful methods, 56
- verbosity control, 35–36
- xUnit, 31–32

■ V, W

- Verbosity control, 35–36

■ X, Y, Z

- xUnit, 31–32