# **Frappe**

# **Unit Testing**

**(Source: <https://frappeframework.com/docs/v14/user/en/guides/automated-testing/unit-testing> )**

**(Video: <https://youtu.be/qYQpBHdQXWE?t=264> )**

## **Introduction**

Frappe provides some basic tooling to quickly write automated tests. There are some basic rules:

1. Test can be anywhere in your repository but must begin with test\_ and should be a .py file.
2. Tests must run on a site that starts with test\_. This is to prevent accidental loss of data.
3. Test stubs are automatically generated for new DocTypes.
4. Frappe test runner will automatically build test records for dependant DocTypes identified by the Link type field (Foreign Key)
5. Tests can be executed using bench run-tests
6. For non-DocType tests, you can write simple unittests and prefix your file names with test\_.

## **Writing Tests for DocTypes**

### **Writing DocType Tests:**

1. Test cases are in a file named test\_[doctype].py
2. You must create all dependencies in the test file
3. Create a Python module structure to create fixtures / dependencies

## **Running Tests**

This function will build all the test dependencies and run your tests. You should run tests from "frappe\_bench" folder. Without options all tests will be run.

bench run-tests

If you need more information about test execution - you can use verbose log level for bench.

bench --verbose run-tests

### **Options:**

*--app <appname>--doctype <doctype>--test <specifictest>--module <module> (Run a particular module that has tests)--profile (Runs a Python profiler on the test)--junit-xml-output<pathtoxml> (The command provides test results in the standard XUnit XML format)*

#### **Example for app:**

All applications are located in folder: "~/frappe-bench/apps". We can run tests for each application.

*- frappe-bench/apps/erpnext/- frappe-bench/apps/erpnext\_demo/- frappe-bench/apps/frappe/*

bench run-tests --app erpnext

bench run-tests --app erpnext\_demo

bench run-tests --app frappe

#### **Example for doctype:**

frappe@erpnext:~/frappe-bench$ bench run-tests --doctype "Activity Cost".

----------------------------------------------------------------------

Ran 1 test in 0.008s

OK

#### **Example for test:**

Run a specific case in User:

frappe@erpnext:~/frappe-bench$ bench run-tests --doctype User --test test*\_get\_*value.

----------------------------------------------------------------------

Ran 1 test in 0.005s

OK

#### **Example for module:**

If we want to run tests in the module:

/home/frappe/frappe-bench/apps/erpnext/erpnext/support/doctype/issue/test\_issue*.py*

We should use module name like this (related to application folder)

erpnext*.support.doctype.issue.test\_issue*

**Example:**

frappe@erpnext:~/frappe-bench$ bench run-tests --*module* "erpnext.stock.doctype.stock\_entry.test\_stock\_entry"

...........................

----------------------------------------------------------------------

Ran 27 tests *in* 30.549s

#### **Example for profile:**

frappe@erpnext:~/frappe-bench$ bench run-tests --doctype "Activity Cost" --profile

.

----------------------------------------------------------------------

Ran 1 test *in* 0.010s

OK

9133 *function* *calls* (8912 primitive calls) *in* 0.011 *seconds*

*Ordered* *by*: *cumulative* *time*

*ncalls* *tottime* *percall* *cumtime* *percall* *filename*:*lineno*(*function*)

2 0.000 0.000 0.008 0.004 /*home*/*frappe*/*frappe*-*bench*/*apps*/*frappe*/*frappe*/*model*/*document*.*py*:187(insert)

1 0.000 0.000 0.003 0.003 /*home*/*frappe*/*frappe*-*bench*/*apps*/*frappe*/*frappe*/*model*/*document*.*py*:386(\_validate)

13 0.000 0.000 0.002 0.000 /*home*/*frappe*/*frappe*-*bench*/*apps*/*frappe*/*frappe*/*database*.*py*:77(sql)

255 0.000 0.000 0.002 0.000 /*home*/*frappe*/*frappe*-*bench*/*apps*/*frappe*/*frappe*/*model*/*base\_document*.*py*:91(get)

12 0.000 0.000 0.002 0.000

#### **Running Tests without creating fixtures or before\_tests hook**

* When you are building a feature it is useful to write tests without building test dependencies (i.e build fixtures for linked objects), with --skip-test-records
* You can also skip the test initialisation script with --skip-before-tests

Example

bench --site school*.erpnext.local* run-tests --doctype "Student Group" --skip-test-records --skip-before-tests

## **FrappeTestCase**

FrappeTestCase is Frappe Framework specific TestCase class extended from unittest.TestCase. Inherting this class in your tests ensures:

1. frappe.local.flags and other most used local proxies are reset after test case runs.
2. database - a new [database transaction](https://frappeframework.com/docs/v14/user/en/api/database" \l "database-transaction-model) is started before testcase begins and rolled back after tests are finished.

Usage

*# app/doctype/dt/test\_dt.py*

*from* frappe.tests.utils *import* FrappeTestCase

*class* *TestDt*(FrappeTestCase): @classmethod

*def* *setUpClass*(cls):

*super*().setUpClass() *# important to call super() methods when extending TestCase.*

...

## **Writing XUnit XML Tests**

### **How to run:**

bench run-tests --junit-xml-output=/reports/junit\_test.xml

### **Example of test report:**

<testsuite tests="3">

<testcase classname="foo1" name="ASuccessfulTest">

<testcase classname="foo2" name="AnotherSuccessfulTest">

<testcase classname="foo3" name="AFailingTest">

<failure type="NotEnoughFoo"> details about failure </failure>

</testcase></testcase></testcase></testsuite>

It’s designed for the CI Jenkins, but will work for anything else that understands an XUnit-formatted XML representation of test results.

### **Jenkins configuration support:**

1. You should install xUnit plugin - https://wiki.jenkins-ci.org/display/JENKINS/xUnit+Plugin
2. After installation open Jenkins job configuration, click the box named “Publish JUnit test result report” under the "Post-build Actions" and enter path to XML report: (Example: reports/\*.xml)