Testkit-Lite User Guide

Content

1. Introduction

This document gives an overview of Testkit-Lite and a guide on how to run testing for Core and WebAPI components using this great tool. You could find usages examples of commands and describe them here.

2. Testkit-Lite Overview

Testkit-lite is a test runner with command-line interface. It accepts XML test descriptor files as input and drive automatic testing execution. Many options are provided to meet your various testing requirement. More usage examples are shown in **Running Testkit-Lite** section

3. Prerequisites

- For Testkit-Lite to work, three packages should have been installed. Please run below command to install those packages
    - python(version>=2.6)
    - python-lxml
    - man

```
$ sudo zypper -n install python python-lxml man
```

- Testkit-Lite recognizes and drives your test cases through a XML test descriptor file. Please read the quick-start of [**how to write TestKit-Lite test case]** to know how to create a valid test descriptor file.
- The test script, grogram or HTML web test page described in test descriptor files should be accessible with system default PATH environment variable. If not, please give full path there to make it accessible by TestKit-Lite.

4. Installing Testkit-Lite

- **To Install from source code:**

1) Run below command to check out the latest source code:

```
$ git clone git@github.com:testkit/testkit-lite.git
```

2) Navigate to the work directory "testkit-lite" after finishing code check-out, run below command to finish installation:

```
$ python ./setup.py build && pythoh ./setup.py install
```

5. Testkit-Lite Options

- **Mandatory options**

Options	Description
-f <test_descriptor_file>.xml	Specify one or more test descriptor files
-e <Web Runtime Environment>	*Only required for WebAPI testing* Specify Web Runtime Environment to run WebAPI testing, e.g. *WRTLauncher <test_widget_name >*

- **Optional options**

Options	Description
-A	Run both of manual and auto test cases. If this option is not used, Testkit-Lite will run only auto test cases.
-M	Only run manual test cases.
-O <test_result_file>	Specify the name of result file. Testkit-Lite will locate this result file under /opt/testkit/lite/latest/ by default
--fullscreen	Run WebAPI testing in full screen mode

6. Running Testkit-Lite

- **To show help info**

```
$ testkit-lite  --help       #show help info
```

- **To get statistics info from test descriptor file without testing**

```
$ testkit-lite  -f  /PATH/TO/<test_descript_file>.xml  -D
```

- **To run Non-WebAPI test cases**

```
$ testkit-lite  -f /PATH/TO/<test_descriptor_file>.xml
```

- **To run WebAPI test cases**

```
$ testkit-lite  -e  "WRTLauncher  <widget _name>" -f
/PATH/TO/<test_descriptor_file>.xml
```

- **To run test cases with multiple test descriptor files**

  Testkit-Lite can accept as many test descriptor files at one time as your want. One way is just sequentially listing all test descriptor files after '**–f**' option

```
$ testkit-lite  -f  /PATH/TO/<test_descript_file>.xml, …,
/PATH/TO/<test_descript_file_more>.xml
```

Alternatively, Testkit-Lite also supports **test configure file** for you to conveniently aggregate all test descriptor files to be run in one test cycle.

- List all test descriptor files line-by-line in a plain-text mode file

```
/PATH/TO/<test_descriptor_1>.xml
/PATH/TO/<test_descriptor_2>.xml
[More]
```

- Run below command to start testing in the order of test descriptor files sorted in the test configure file

```
$ testkit-lite  --testxmlconfig /PATH/TO/<test_config_file>
```

Like test descriptor file, Testkit-Lite accepts as many test configure files as you want. You could decompose sets of test descriptor files in different test configure files, and sequentially listed after '**--testxmlconfig**' option

```
$ testkit-lite  --testxmlconfig
/PATH/TO/<test_config_file_1>,…,/PATH/TO/<test_config_file_more>
```

- **To run a subset of test cases in test descriptor file with filters**

  Using filter option, you could select out and run a subset of test cases from the test descriptor file according to the filters. You could use below filters to select test cases:

Filter	Description
--type	Filter test cases by test case type: • functional_positive • functional_negative • security • performance • reliability • portability • maintainability • compliance • user_experience
--priority	Filter test cases by test case priority: • P0 • P1 • P2
--category	Filter test cases by test category: • Netbook • IVI • TV
--status	Filter test case by test case status: • ready • approved • designed

You could give multiple values for each filter, e.g. to select test cases of both P0 and P1 priority:

```
testkit-lite  --priority P0 P1 -f <test_descriptor_file>.xml
```

A group of filters could be used at one time, and Testkit-Lite will perform the **AND** logic when selecting test cases, e.g. to select test cases by both of **Priority** and **Category** filter:

```
testkit-lite  --priority P0 P1 --category  Netbook IVI -f <test_descriptor_file>.xml
```

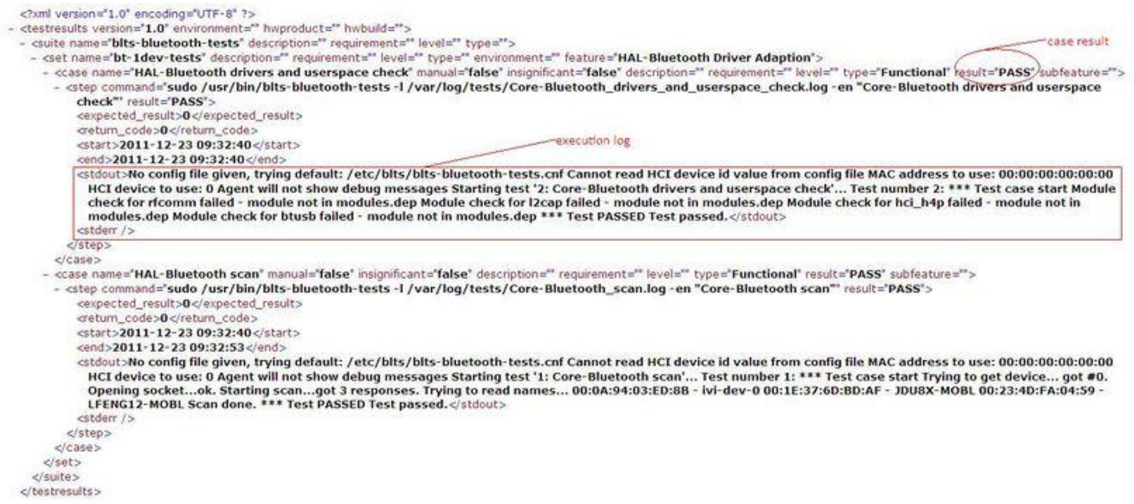
You could freely combine all those usages in one command to meet your testing requirement, e.g. to select and run test cases from two test descriptor files by filters of **priority**, **category** and **status**:

```
testkit-lite  --priority P0 P1 --category  Netbook IVI --status ready -f
<test_descriptor_file_1>.xml <test_descriptor_file_2>.xml
```

7. Checking Test Reports

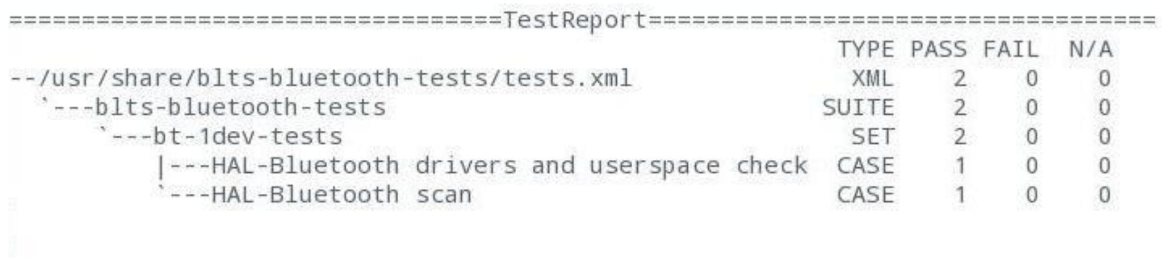
After Testkit-Lite successfully completes all test cases execution, you could get XML, Text test reports under /opt/testkit/lite/latest. Here are examples of each type of test report

XML Test Report



```
<?xml version="1.0" encoding="UTF-8" ?>
- <testresults version="1.0" environment="" hwproduct="" hwbuild="">
  - <suite name="blts-bluetooth-tests" description="" requirement="" level="" type="">
    - <set name="bt-1dev-tests" description="" requirement="" level="" type="" environment="" feature="HAL-Bluetooth Driver Adaption">
      - <case name="HAL-Bluetooth drivers and userspace check" manual="false" insignificant="false" description="" requirement="" level="" type="Functional" result="PASS" subfeature="">
        - <step command="sudo /usr/bin/blts-bluetooth-tests -l /var/log/tests/Core-Bluetooth_drivers_and_userspace_check.log -en "Core-Bluetooth drivers and userspace
            check"" result="PASS">
            <expected_result>0</expected_result>
            <return_code>0</return_code>
            <start>2011-12-23 09:32:40</start>
            <end>2011-12-23 09:32:40</end>
            <stdout>No config file given, trying default: /etc/blts/blts-bluetooth-tests.cnf Cannot read HCI device id value from config file MAC address to use: 00:00:00:00:00:00
              HCI device to use: 0 Agent will not show debug messages Starting test '2: Core-Bluetooth drivers and userspace check'... Test number 2: *** Test case start Module
              check for rfcomm failed - module not in modules.dep Module check for l2cap failed - module not in modules.dep Module check for hci_h4p failed - module not in
              modules.dep Module check for btusb failed - module not in modules.dep *** Test PASSED Test passed.</stdout>
            <stderr />
          </step>
        </case>
      - <case name="HAL-Bluetooth scan" manual="false" insignificant="false" description="" requirement="" level="" type="Functional" result="PASS" subfeature="">
        - <step command="sudo /usr/bin/blts-bluetooth-tests -l /var/log/tests/Core-Bluetooth_scan.log -en "Core-Bluetooth scan"" result="PASS">
            <expected_result>0</expected_result>
            <return_code>0</return_code>
            <start>2011-12-23 09:32:40</start>
            <end>2011-12-23 09:32:53</end>
            <stdout>No config file given, trying default: /etc/blts/blts-bluetooth-tests.cnf Cannot read HCI device id value from config file MAC address to use: 00:00:00:00:00:00
              HCI device to use: 0 Agent will not show debug messages Starting test '1: Core-Bluetooth scan'... Test number 1: *** Test case start Trying to get device... got #0.
              Opening socket...ok. Starting scan...got 3 responses. Trying to read names... 00:0A:94:03:ED:8B - ivi-dev-0 00:1E:37:6D:BD:AF - JDU8X-MOBL 00:23:4D:FA:04:59 -
              LFENG12-MOBL Scan done. *** Test PASSED Test passed.</stdout>
            <stderr />
          </step>
        </case>
      </set>
    </suite>
  </testresults>
```

Text Test Report



```
==================================TestReport==================================
                                                           TYPE  PASS  FAIL  N/A
--/usr/share/blts-bluetooth-tests/tests.xml                 XML     2     0    0
  `---blts-bluetooth-tests                                SUITE     2     0    0
      `---bt-1dev-tests                                     SET     2     0    0
          |---HAL-Bluetooth drivers and userspace check    CASE     1     0    0
          `---HAL-Bluetooth scan                           CASE     1     0    0
```