How to add test cases on JTA

Abstract

This document is used to demonstrate how to add a regular test case to JTA. The newly added test case, as an example for this document, is used to test “touch” command. That is to say, “touch <file>” will be executed on the target machine. If “<file>” is created successfully, the test passes; otherwise it fails.

1. To make explanation easier, we make some assumptions here:
   a) The machine with JTA installed on it will be called "JTA machine" below. The IP address of JTA machine is 192.168.30.71.
   b) The machine, on which the test is supposed to be tested, will be called "target machine" below. The IP address of target machine is 192.168.30.64.

2. Login to JTA machine as "root" user.

3. Use the following command to check whether "Jenkins" service is working.

```
# /etc/init.d/jenkins status
```

   If message, like "Jenkins Continuous Integration Server is not running", is showed, please use the following command to start "Jenkins" service.

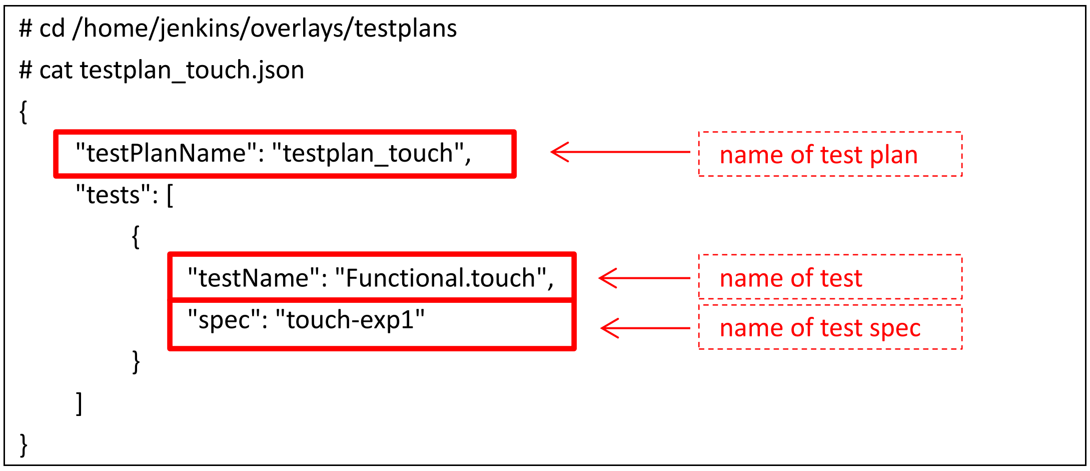
```
# /etc/init.d/jenkins start
```

4. The following table lists the files that should be added or fixed in order to add a test case for "touch" command.

file	usage
(optional) /home/jenkins/overlays/testplans	used for selecting "spec" for test cases, so that some variables in test_specs will be set to satisfy the requirement of the test.
(optional) /home/jenkins/overlays/test_specs	used for defining some variables for test. These variables are organized as "spec". In different "spec", variables will be defined differently
/home/jenkins/tests/Functional.touch/touch-script.sh	test start point that will be used to setup the test environment, execute the test and grab test result from target machine
/home/jenkins/tests/Functional.touch/touch-device.sh	test program that will be executed on the target machine to test "touch" command
/home/jenkins/overlays/boards/porter.board	configuration of target machine, touch-script.sh needs this to setup test environment
/home/jenkins/scripts/tools.sh	defining variables used to cross-build programs for target machine

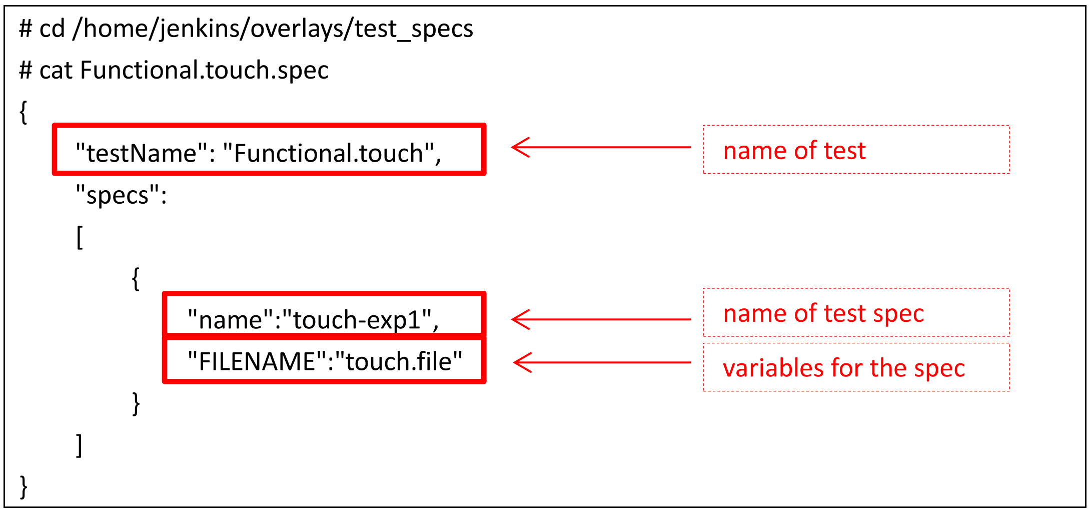
   "/home/jenkins/overlays/testplans" and "/home/jenkins/overlays/test_specs" are optional, only used when some special variables are needed for certain tests.
   More detailed information will be demonstrated in the next several steps.

5. Add “test plan” (optional)
   Add “testplan_touch.json” under “/home/jenkins/overlays/testplans”, and write it as the following example.



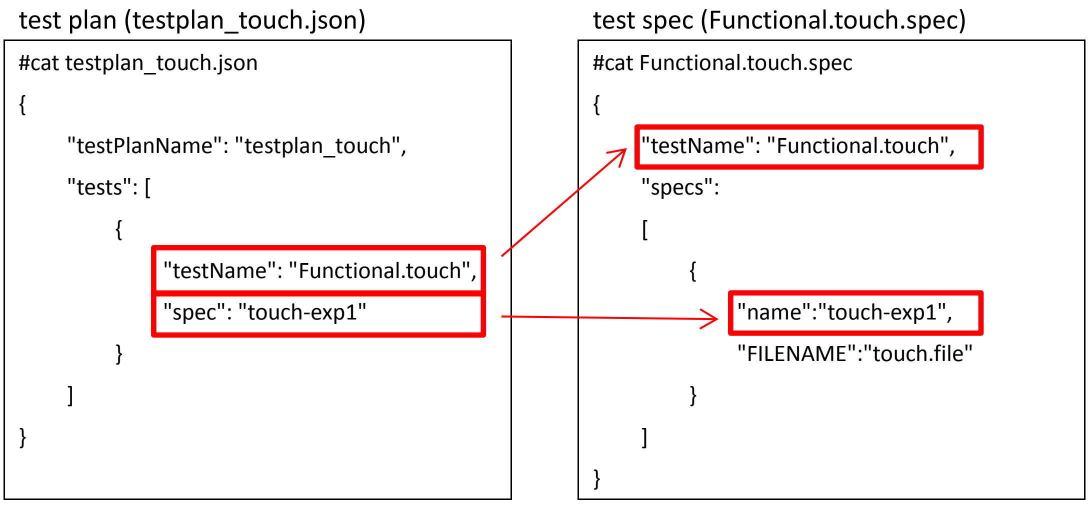
```
# cd /home/jenkins/overlays/testplans
# cat testplan_touch.json
{
    "testPlanName": "testplan_touch",
    "tests": [
        {
            "testName": "Functional.touch",
            "spec": "touch-exp1"
        }
    ]
}
```

6. Add “test spec” (optional)
   Add “Functional.touch.spec” under “/home/jenkins/overlays/test_specs”, and write it as the following example.



```
# cd /home/jenkins/overlays/test_specs
# cat Functional.touch.spec
{
    "testName": "Functional.touch",
    "specs":
    [
        {
            "name":"touch-exp1",
            "FILENAME":"touch.file"
        }
    ]
}
```

7. Relationship between “test plan” and “test spec”



test plan (testplan_touch.json)

```
#cat testplan_touch.json
{
    "testPlanName": "testplan_touch",
    "tests": [
        {
            "testName": "Functional.touch",
            "spec": "touch-exp1"
        }
    ]
}
```

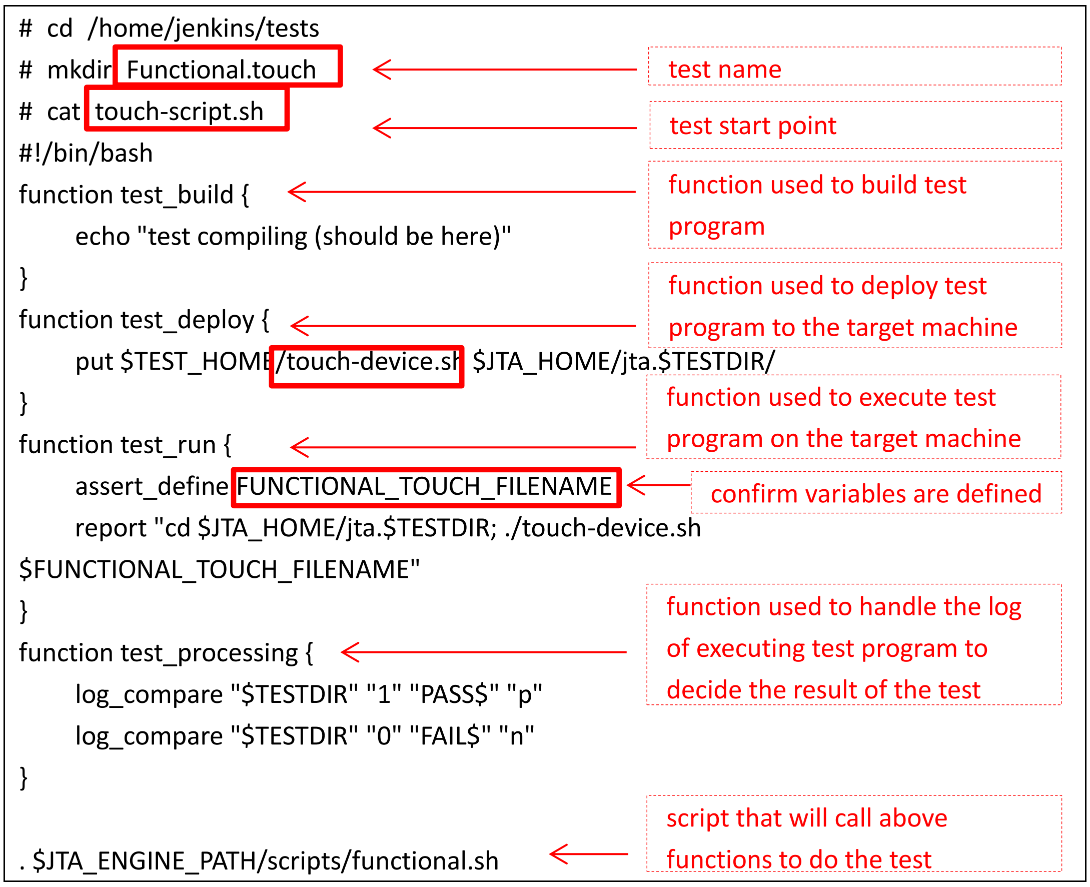
test spec (Functional.touch.spec)

```
#cat Functional.touch.spec
{
    "testName": "Functional.touch",
    "specs":
    [
        {
            "name":"touch-exp1",
            "FILENAME":"touch.file"
        }
    ]
}
```

8. Add test script

Create folder “Functional.touch” under “/home/jenkins/tests”, and under the folder add two files, “touch-script.sh” and “touch-device.sh”.

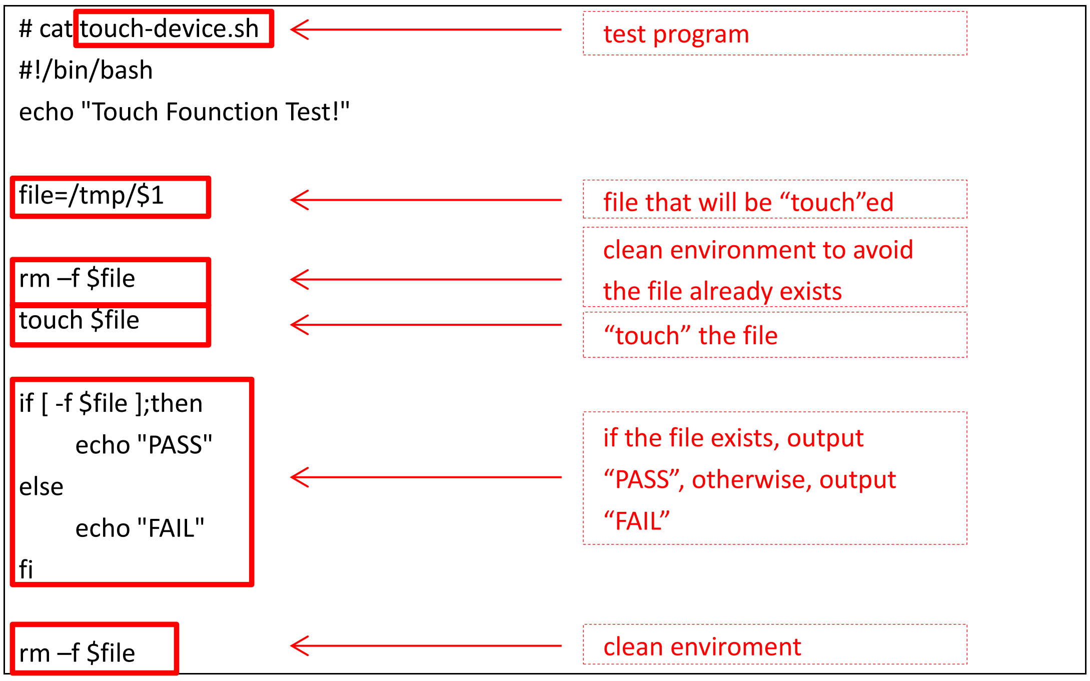
Follow the example below to write “touch-script.sh”.



```
#  cd  /home/jenkins/tests
#  mkdir  Functional.touch
#  cat  touch-script.sh
#!/bin/bash
function test_build {
        echo "test compiling (should be here)"
}
function test_deploy {
        put $TEST_HOME/touch-device.sh $JTA_HOME/jta.$TESTDIR/
}
function test_run {
        assert_define FUNCTIONAL_TOUCH_FILENAME
        report "cd $JTA_HOME/jta.$TESTDIR; ./touch-device.sh
$FUNCTIONAL_TOUCH_FILENAME"
}
function test_processing {
        log_compare "$TESTDIR" "1" "PASS$" "p"
        log_compare "$TESTDIR" "0" "FAIL$" "n"
}

. $JTA_ENGINE_PATH/scripts/functional.sh
```

test name
test start point
function used to build test program
function used to deploy test program to the target machine
function used to execute test program on the target machine
confirm variables are defined
function used to handle the log of executing test program to decide the result of the test
script that will call above functions to do the test

Follow the example below to write “touch-device.sh”. Be careful, “touch-device.sh” should gain the executable permission in order to be run on target machine.



```
# cat touch-device.sh
#!/bin/bash
echo "Touch Founction Test!"

file=/tmp/$1

rm –f $file
touch $file

if [ -f $file ];then
        echo "PASS"
else
        echo "FAIL"
fi

rm –f $file
```

test program
file that will be “touch”ed
clean environment to avoid the file already exists
“touch” the file
if the file exists, output “PASS”, otherwise, output “FAIL”
clean enviroment

9. Fix configuration of target machine
   Follow the example below to fix porter's related configuration, "porter.board" under "/home/jenkins/overlays/boards".

```
# cd /home/jenkins/overlays/boards
# cat qemu-arm.board
inherit "base-board"
include "base-params"

IPADDR="192.168.30.64"
LOGIN="root"
JTA_HOME="/home/a"
#PASSWORD="root"
PLATFORM="porter"
TRANSPORT="ssh"
ARCHITECTURE="arm"

#SATA_DEV="/dev/sda1"
#SATA_MP="/mnt/sata"

#USB_DEV="/dev/sdb1"
#USB_MP="/mnt/usb"

#MMC_DEV="/dev/mmcblk0p2"
#MMC_MP="/mnt/mmc"
```

IPADDR: IP address or hostname of target machine

LOGIN: user name for ssh login

#PASSWORD: password for ssh login, not needed for AGL yet

If you want to execute the test on other target machine, fix the related "*.board" file. You can also refer to "jta-guide.pdf" for more detailed information.

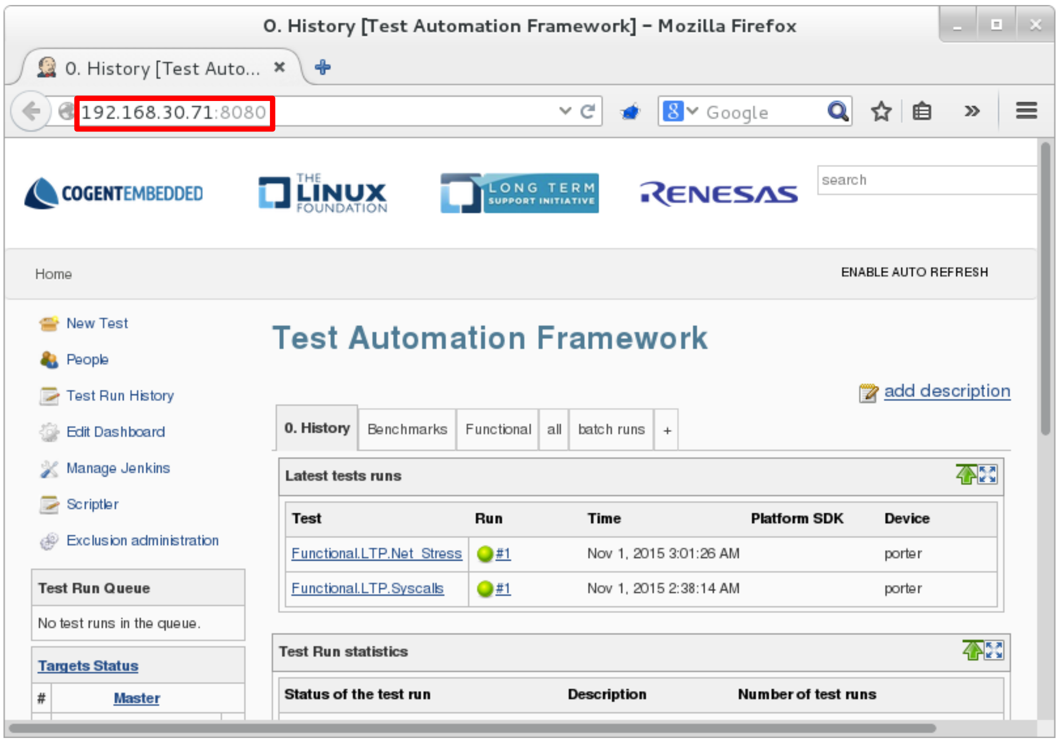
10. Fix variable definition used for corss-building
    Fix "tools.sh" under "/home/jenkins/scripts". Variables, like SDKROOT, PREFIX, HOST, and "source" are used to setup cross-build environment.

```
#  cd  /home/jenkins/scripts
#  cat  tools.sh
……
elif [  "${PLATFORM}"  =  "porter"  ];
then
            ORIG_PATH=$PATH
            PREFIX=arm-poky-linux-gnueabi
            source  /opt/poky-agl/1.0.0/environment-setup-cortexa15hf-vfp-neon-poky-linux-gnueabi
            SDKROOT=/opt/poky-agl/1.0.0/sysroots/cortexa15hf-vfp-neon-poky-linux-gnueabi/
            HOST=arm-poky-linux-gnueabi
```

["${PLATFORM}" = "porter"]: selected by "PLATFORM" variable in "*.board". Check step 9

```
            unset  PYTHONHOME
            env  -u  PYTHONHOME
……
```

11. Logon to JTA web interface. The URL should be “192.168.30.71:8080” here:



0. History [Test Automation Framework] – Mozilla Firefox

192.168.30.71:8080

Test Automation Framework

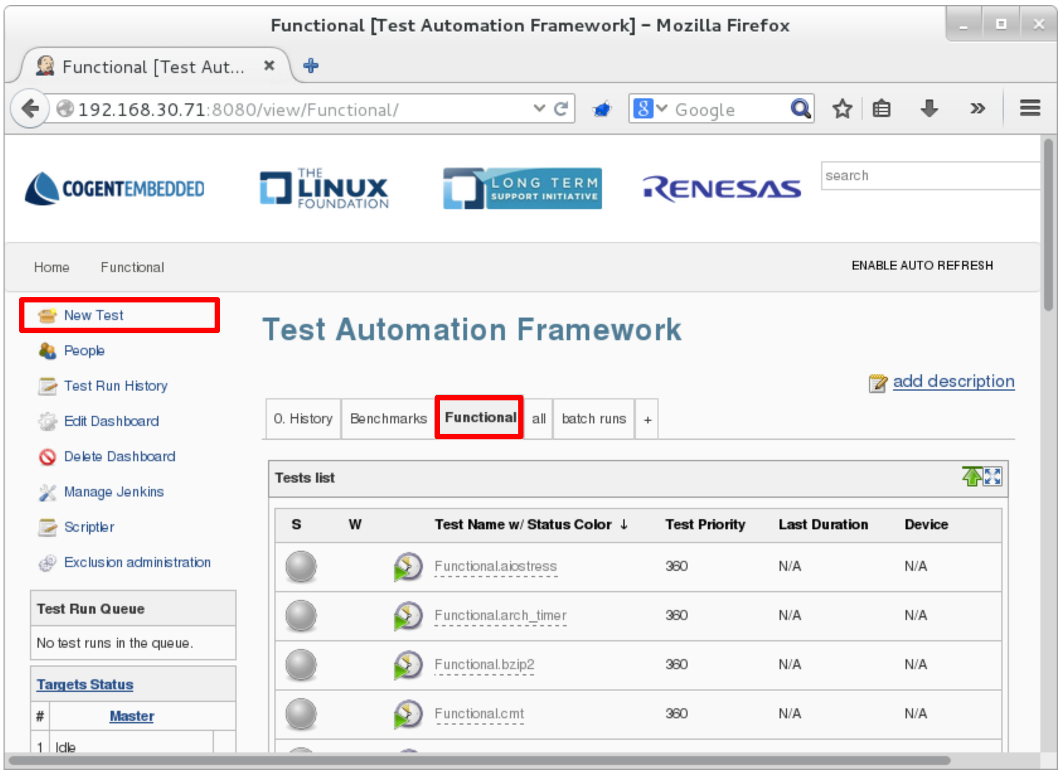
Latest tests runs

Test	Run	Time	Platform SDK	Device
Functional.LTP.Net_Stress	● #1	Nov 1, 2015 3:01:26 AM		porter
Functional.LTP.Syscalls	● #1	Nov 1, 2015 2:38:14 AM		porter

Test Run statistics

Status of the test run	Description	Number of test runs

12. Click “Functional” tag, then click “New Test” to create a new test case



Functional [Test Automation Framework] – Mozilla Firefox

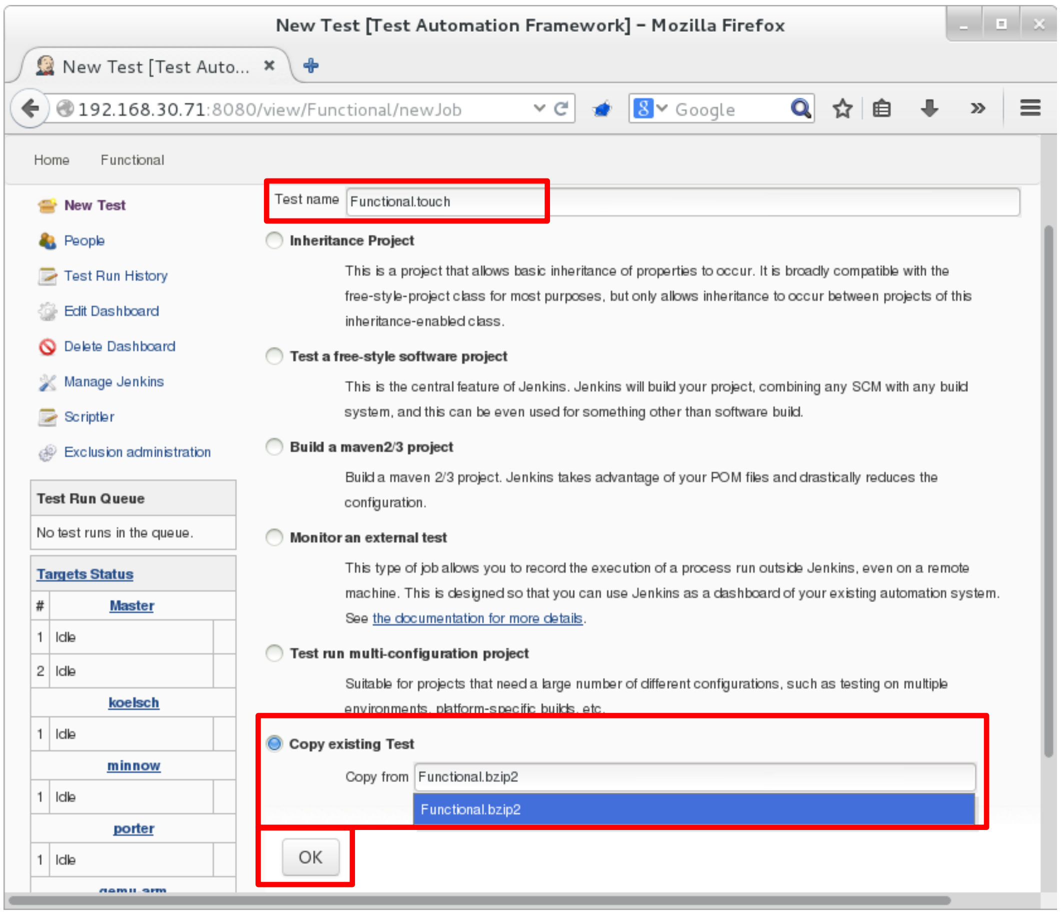
192.168.30.71:8080/view/Functional/

Test Automation Framework

Tests list

S	W	Test Name w/ Status Color ↓	Test Priority	Last Duration	Device
●		Functional.aiostress	360	N/A	N/A
●		Functional.arch_timer	360	N/A	N/A
●		Functional.bzip2	360	N/A	N/A
●		Functional.cmt	360	N/A	N/A

13. Input “Functional.touch” for “Test name”. Then check “Copy existing Test”, input “Functional.bzip2”. After all, click “OK”.


New Test [Test Automation Framework] - Mozilla Firefox

192.168.30.71:8080/view/Functional/newJob

Home Functional

Test Run Queue

No test runs in the queue.

Targets Status

#	Master
1	Idle
2	Idle
	koelsch
1	Idle
	minnow
1	Idle
	porter
1	Idle

Test name: Functional.touch

( ) **Inheritance Project**

This is a project that allows basic inheritance of properties to occur. It is broadly compatible with the free-style-project class for most purposes, but only allows inheritance to occur between projects of this inheritance-enabled class.

( ) **Test a free-style software project**

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

( ) **Build a maven2/3 project**

Build a maven 2/3 project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

( ) **Monitor an external test**

This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed so that you can use Jenkins as a dashboard of your existing automation system. See the documentation for more details.

( ) **Test run multi-configuration project**

Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

(✓) **Copy existing Test**

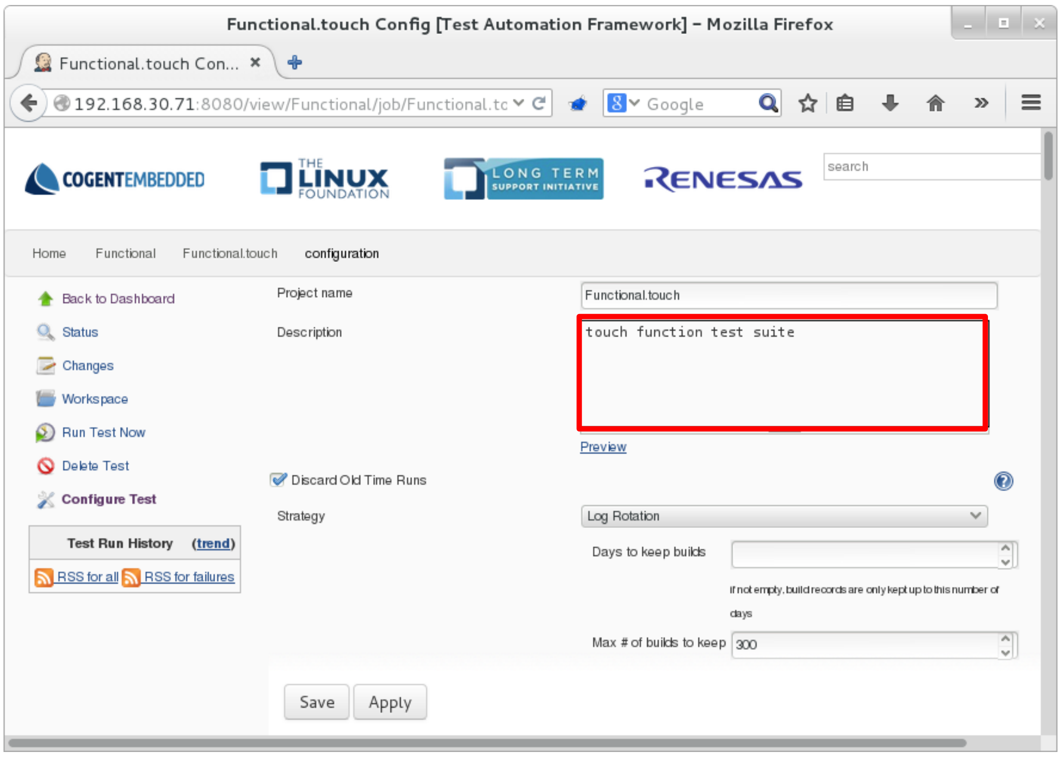
Copy from: Functional.bzip2

Functional.bzip2

OK

14. Fix configurations related to the test

1) test description:


Functional.touch Config [Test Automation Framework] - Mozilla Firefox

192.168.30.71:8080/view/Functional/job/Functional.tc

Home Functional Functional.touch configuration

Project name: Functional.touch

Description: touch function test suite

Discard Old Time Runs: ✓

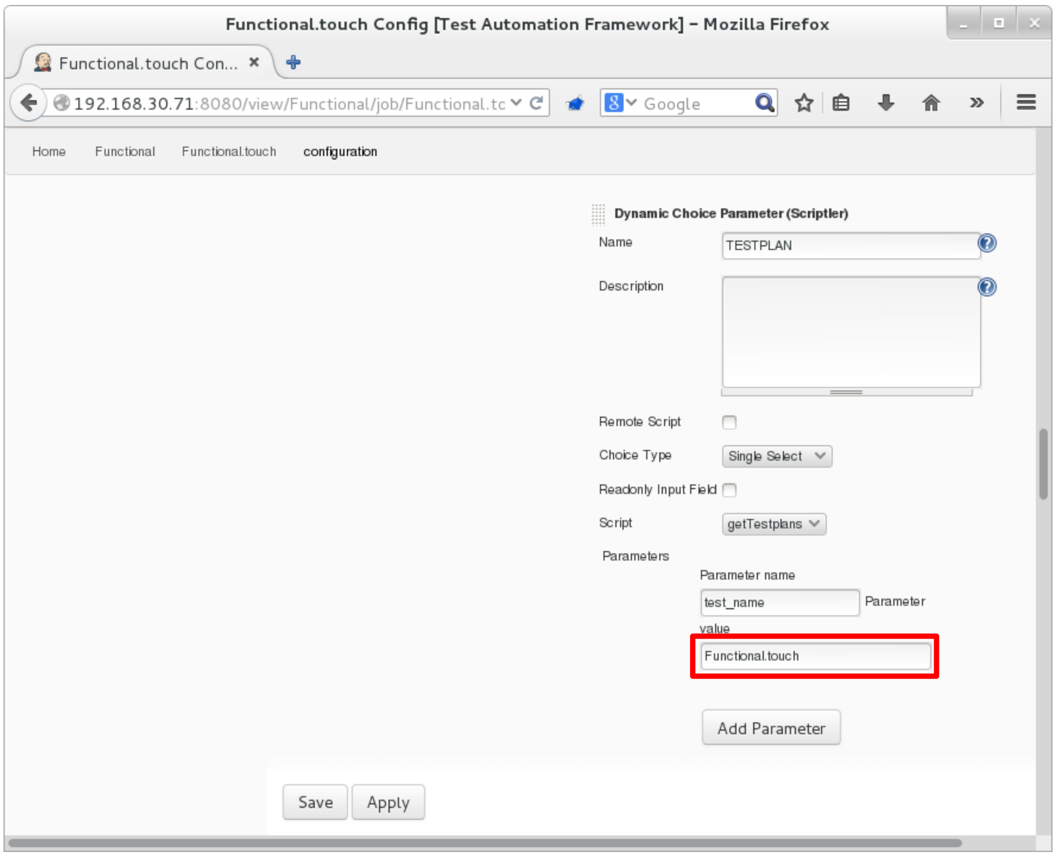
Strategy: Log Rotation

Days to keep builds:

if not empty, build records are only kept up to this number of days

Max # of builds to keep: 300

2) test name:


Functional.touch Config [Test Automation Framework] - Mozilla Firefox

192.168.30.71:8080/view/Functional/job/Functional.tc

Home Functional Functional.touch configuration

Dynamic Choice Parameter (Scriptler)

Name: TESTPLAN

Description:

Remote Script: ☐

Choice Type: Single Select

Readonly Input Field: ☐

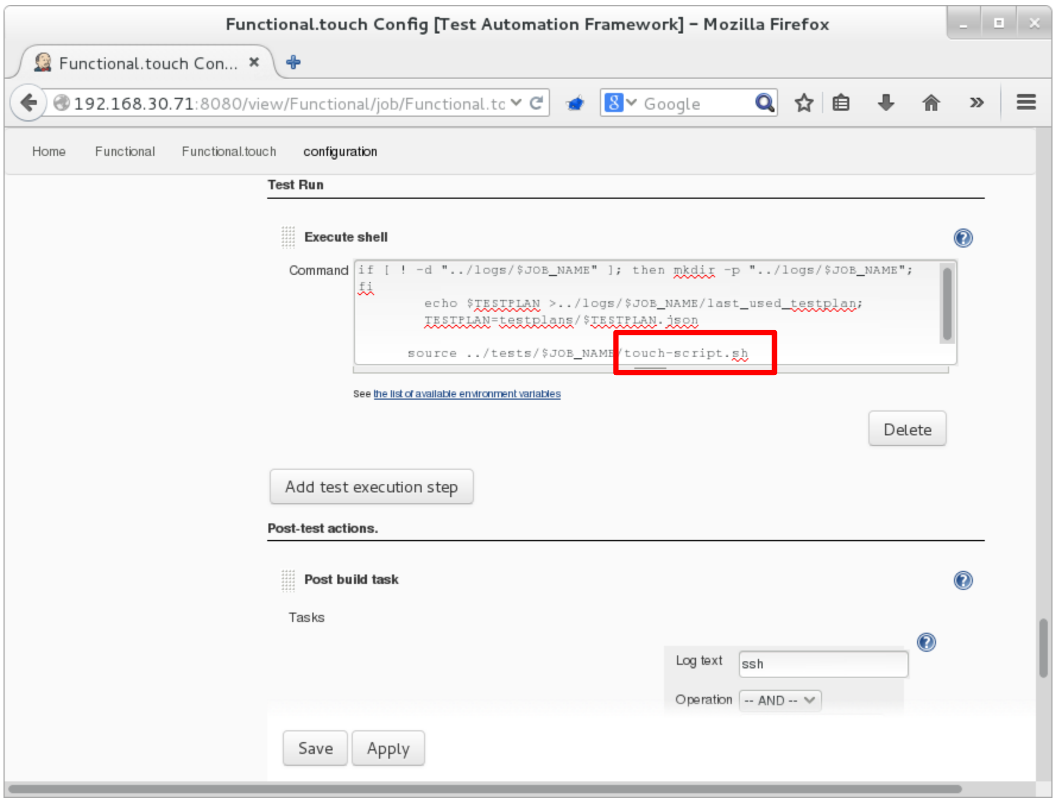
Script: getTestplans

Parameters

Parameter name: test_name

Parameter value: Functional.touch

3) test start point, it should be “touch-script.sh” here:



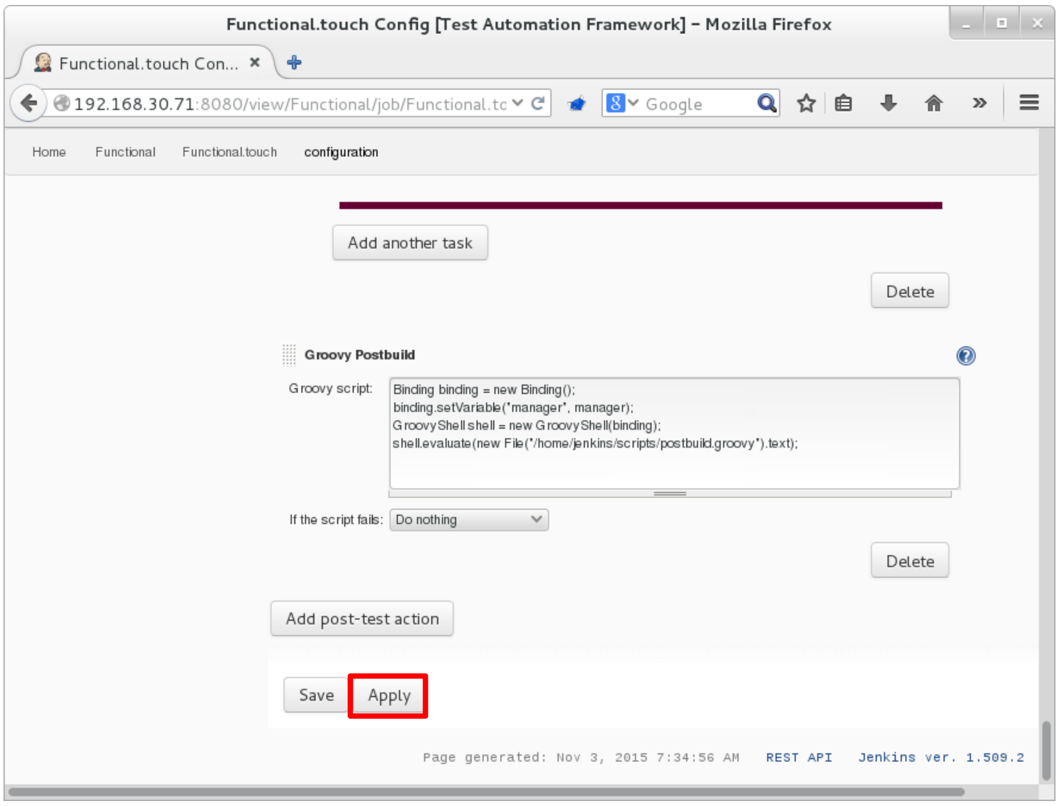
```
if [ ! -d "../logs/$JOB_NAME" ]; then mkdir -p "../logs/$JOB_NAME";
fi
        echo $TESTPLAN >../logs/$JOB_NAME/last_used_testplan;
        TESTPLAN=testplans/$TESTPLAN.json

      source ../tests/$JOB_NAME/touch-script.sh
```

See the list of available environment variables

Log text: ssh

Operation: -- AND --

4) click “Apply”, then the new test case is created:

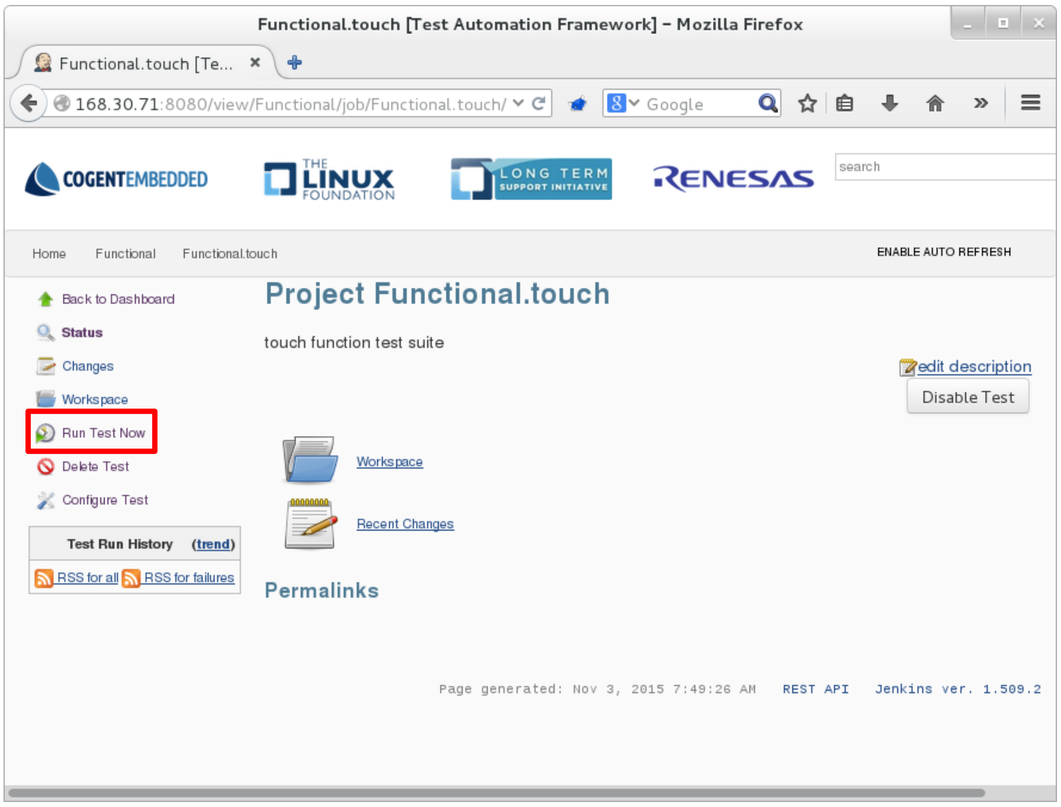


```
Binding binding = new Binding();
binding.setVariable("manager", manager);
GroovyShell shell = new GroovyShell(binding);
shell.evaluate(new File("/home/jenkins/scripts/postbuild.groovy").text);
```

If the script fails: Do nothing

Page generated: Nov 3, 2015 7:34:56 AM REST API Jenkins ver. 1.509.2

15. Clike “Run Test Now” on the left side.
    Choose “porter” for “Device”, check “Rebuild” and choose “testpaln_touch” for TESTPLAN.
    Then click “Run test” to start the test. The test progress will be showed in “Test Run History”.

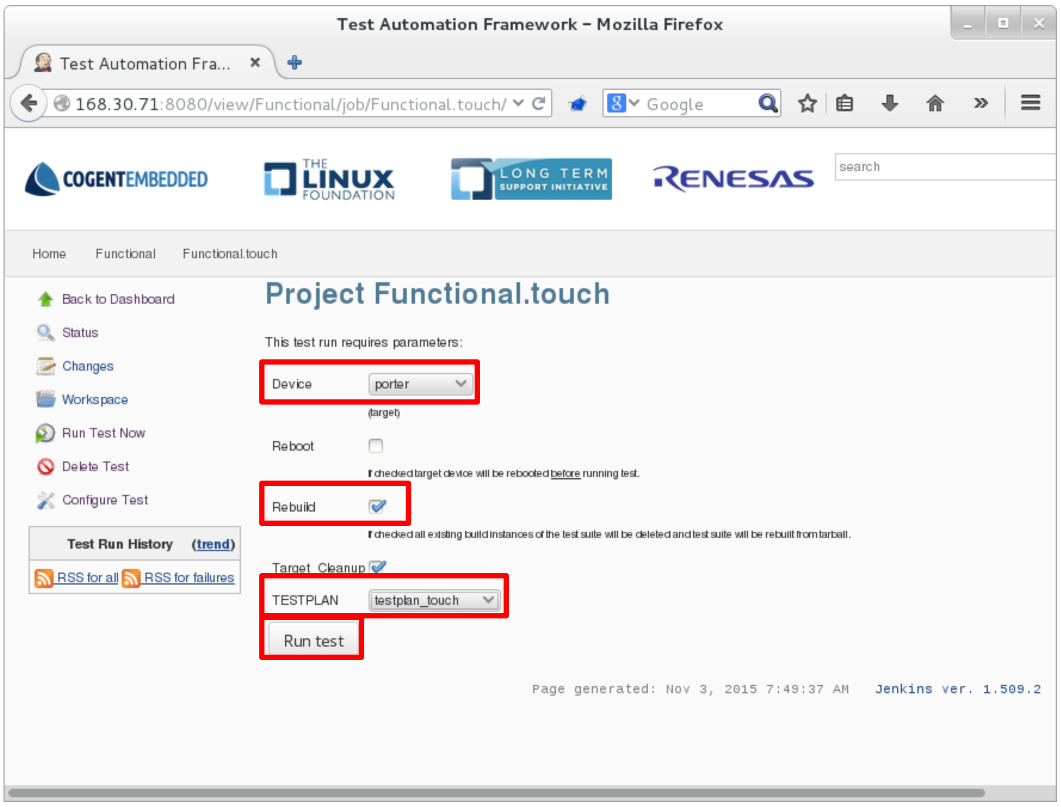


Project Functional.touch

touch function test suite

Permalinks

Page generated: Nov 3, 2015 7:49:26 AM REST API Jenkins ver. 1.509.2



Project Functional.touch

This test run requires parameters:

Device: porter
(target)

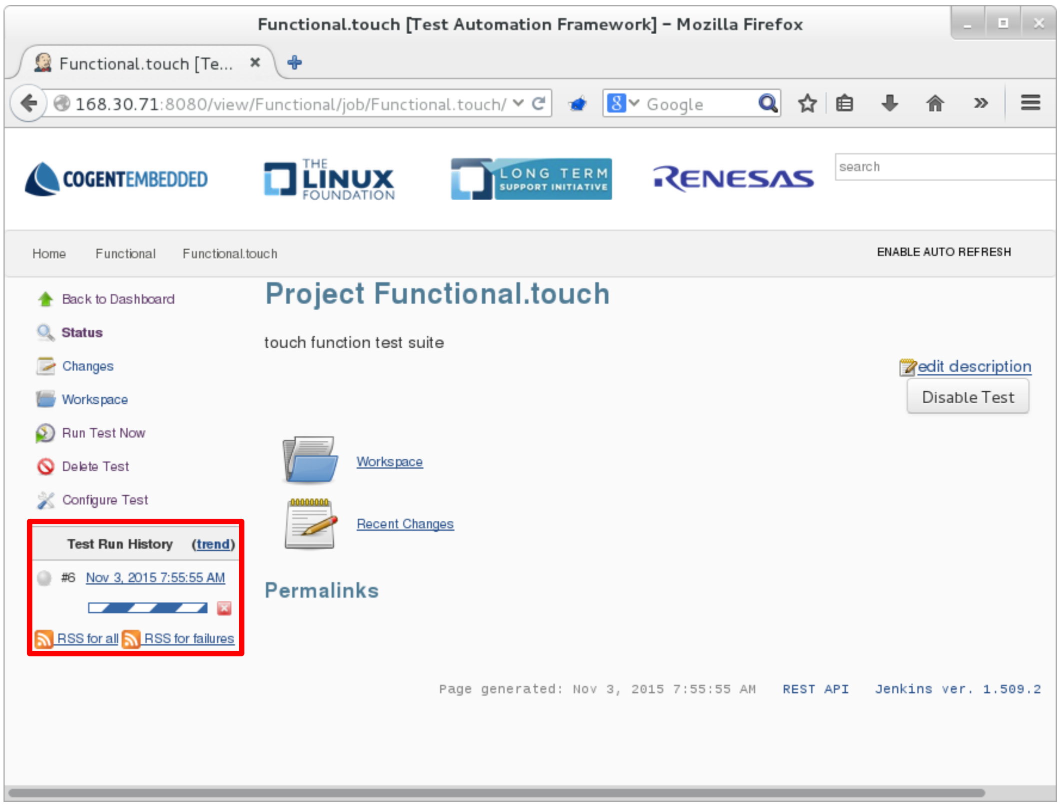
Reboot: ☐
If checked target device will be rebooted before running test.

Rebuild: ✓
If checked all existing build instances of the test suite will be deleted and test suite will be rebuilt from tarball.

Target_Cleanup: ✓

TESTPLAN: testplan_touch

Page generated: Nov 3, 2015 7:49:37 AM Jenkins ver. 1.509.2


Functional.touch [Test Automation Framework] - Mozilla Firefox

168.30.71:8080/view/Functional/job/Functional.touch/

Home Functional Functional.touch ENABLE AUTO REFRESH

Back to Dashboard
Status
Changes
Workspace
Run Test Now
Delete Test
Configure Test

Test Run History (trend)
#6 Nov 3, 2015 7:55:55 AM
RSS for all RSS for failures

Project Functional.touch

touch function test suite

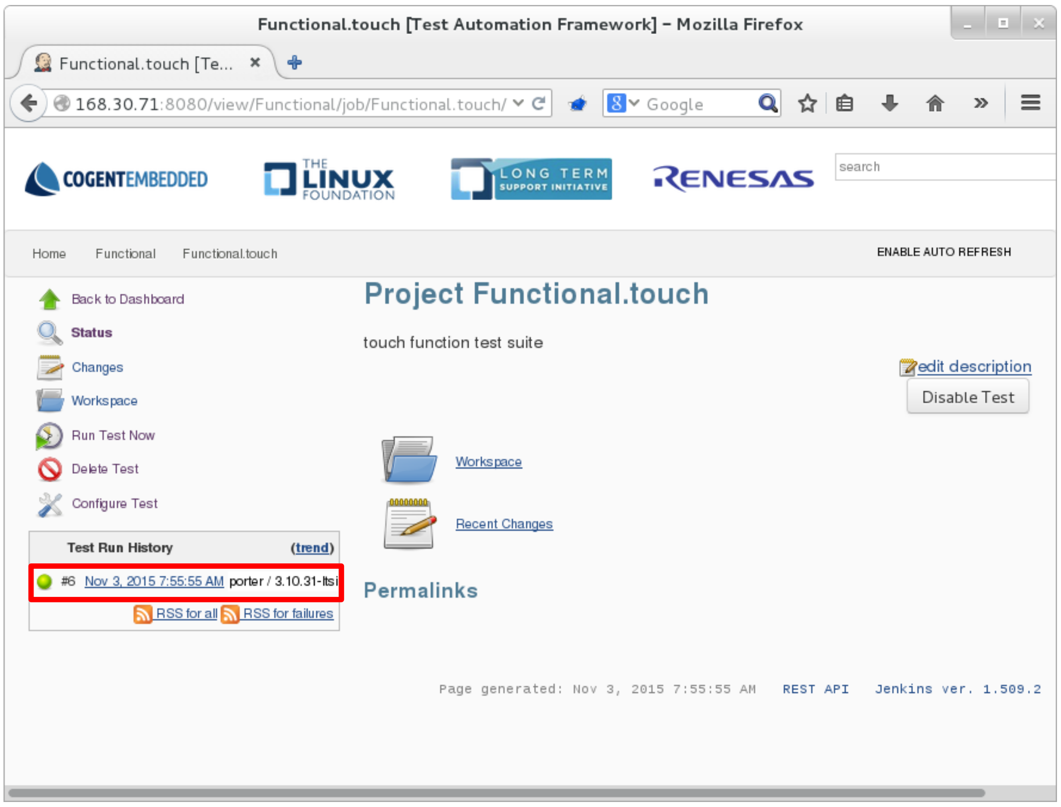
edit description
Disable Test

Workspace
Recent Changes

Permalinks

Page generated: Nov 3, 2015 7:55:55 AM REST API Jenkins ver. 1.509.2

16. If the test succeeded, a line with a green icon in front of it will be showed; otherwise, a red icon will be showed.
Click this line to get more information about this test.


Functional.touch [Test Automation Framework] - Mozilla Firefox

168.30.71:8080/view/Functional/job/Functional.touch/

Home Functional Functional.touch ENABLE AUTO REFRESH

Back to Dashboard
Status
Changes
Workspace
Run Test Now
Delete Test
Configure Test

Test Run History (trend)
#6 Nov 3, 2015 7:55:55 AM porter / 3.10.31-ltsi
RSS for all RSS for failures

Project Functional.touch

touch function test suite

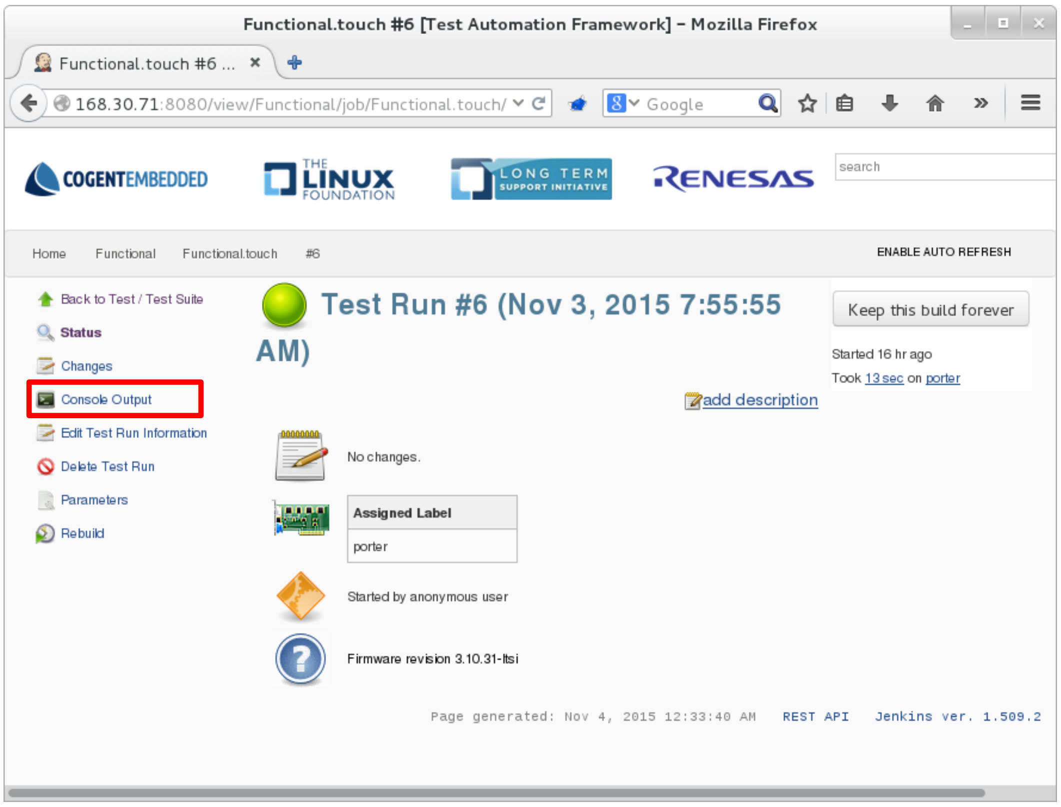
edit description
Disable Test

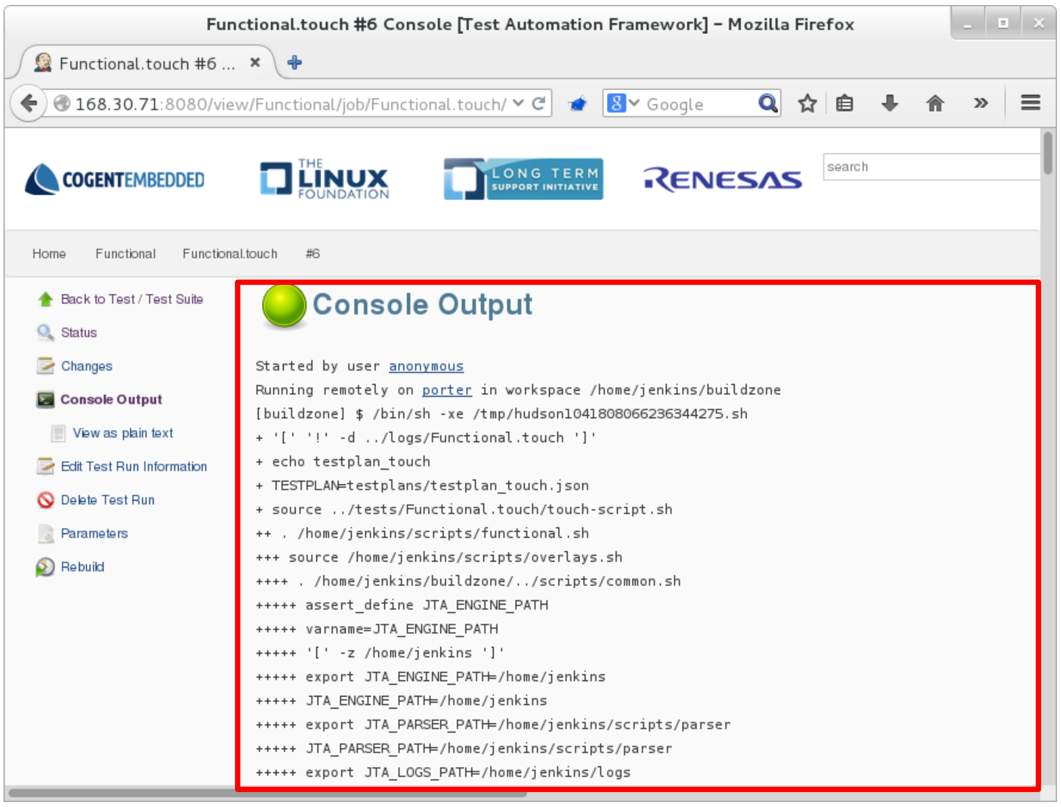
Workspace
Recent Changes

Permalinks

Page generated: Nov 3, 2015 7:55:55 AM REST API Jenkins ver. 1.509.2

17. Click “Console Output” on the left side, log of the test will be showed.


Functional.touch #6 [Test Automation Framework] - Mozilla Firefox
168.30.71:8080/view/Functional/job/Functional.touch/
Home Functional Functional.touch #6
ENABLE AUTO REFRESH
Back to Test / Test Suite
Status
Changes
Console Output
Edit Test Run Information
Delete Test Run
Parameters
Rebuild
Test Run #6 (Nov 3, 2015 7:55:55 AM)
Keep this build forever
Started 16 hr ago
Took 13 sec on porter
add description
No changes.
Assigned Label
porter
Started by anonymous user
Firmware revision 3.10.31-ltsi
Page generated: Nov 4, 2015 12:33:40 AM REST API Jenkins ver. 1.509.2


Functional.touch #6 Console [Test Automation Framework] - Mozilla Firefox
168.30.71:8080/view/Functional/job/Functional.touch/
Home Functional Functional.touch #6
Back to Test / Test Suite
Status
Changes
Console Output
View as plain text
Edit Test Run Information
Delete Test Run
Parameters
Rebuild
Console Output

```
Started by user anonymous
Running remotely on porter in workspace /home/jenkins/buildzone
[buildzone] $ /bin/sh -xe /tmp/hudson1041808066236344275.sh
+ '[' '!' -d ../logs/Functional.touch ']'
+ echo testplan_touch
+ TESTPLAN=testplans/testplan_touch.json
+ source ../tests/Functional.touch/touch-script.sh
++ . /home/jenkins/scripts/functional.sh
+++ source /home/jenkins/scripts/overlays.sh
++++ . /home/jenkins/buildzone/../scripts/common.sh
+++++ assert_define JTA_ENGINE_PATH
+++++ varname=JTA_ENGINE_PATH
+++++ '[' -z /home/jenkins ']'
+++++ export JTA_ENGINE_PATH=/home/jenkins
+++++ JTA_ENGINE_PATH=/home/jenkins
+++++ export JTA_PARSER_PATH=/home/jenkins/scripts/parser
+++++ JTA_PARSER_PATH=/home/jenkins/scripts/parser
+++++ export JTA_LOGS_PATH=/home/jenkins/logs
```