# STF Features

## Test script configuration handling

### XML configuration

XML testing file handles many technical configurations: server address, used tools and related configuration.

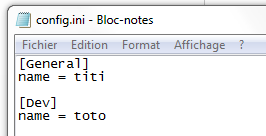
Business configuration can be done through the “unknown” parameters. These are parameters which are not known from the framework. They are added to a list of business parameters.

### Confg.ini configuration

XML configurations are done statically and must be duplicated through all the test suites (or using “testConfiguration” parameter). It’s not possible to have a centralized configuration which depends on test environment.

Example: the server URL depends on testing phase. They are not the same in production and in integration phase.

That’s why the “config.ini” file is made for. Each tested application can embed a config.ini file whose format is:



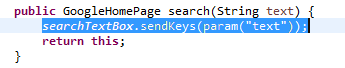
“General“ section is for common configuration (e.g: a database user name which does not depends on environment) and other sections are specific to named test environments. Here, we define a “Dev” environment. Then, when launching test, user MUST define the environment on which test will be run with the option “-Denv=Dev”

Keys defined in environment sections override the ones in “General” section.

This file must be located in <application root>/data/<application name>/config folder.

### Using configurations in test scripts

Each webpage can use the configurations defined above using (getting variable “text” from configuration):



## Cucumber tests

## TestNG tests

**TBD (extract example from STF)**

# Test with providers

## Test with Appium locally

### Application test on android

<test name=*"tnr\_appium\_mobile\_app"* parallel=*"false"*>

<!-- cucumber part -->

<parameter name=*"cucumberTests"* value=*"Infolidays"* />

<parameter name=*"cucumberTags"* value=*""* />

<parameter name=*"variables"* value=*""* />

<parameter name=*"browser"* value=*"\*android"* />

<parameter name=*"testType"* value=*"appium\_app\_android"* />

<parameter name=*"appiumServerURL"* value=*"http://localhost:4723/wd/hub"*/>

<parameter name=*"automationName"* value=*"Appium"*/>

<parameter name=*"platform"* value=*"Android"*/>

<parameter name=*"mobilePlatformVersion"* value=*"6.0"*/>

<parameter name=*"deviceName"* value=*"192.168.56.101:5555"*/>

<parameter name=*"app"* value=*"C:\Users\behe\Documents\Tests\Tests mobiles\infolidays-1.0.0-release.apk"*/>

<parameter name=*"appPackage"* value=*"com.infotel.mobile.infolidays"*/>

<parameter name=*"appActivity"* value=*"com.infotel.mobile.mesconges.view.activity.StartActivity"*/>

<!--browserName should be empty string when automating app-->

<parameter name=*"browserName"* value=*""*/>

<parameter name=*"newCommandTimeout"* value=*"120"*/>

<packages>

<package name=*"com.seleniumtests.core.runner.\*"*/>

</packages>

</test>

deviceName reflects the local device used to automate the test

## Test with SauceLabs

## Test with Testdroid

# Configurations

Below is the list of all parameters accepted in testing xml file.

TBD

# Interfacing with tests managers

## Squash TM/TA

STF can work with Squash TA by using an intermediate .java file. This file handles:

* Execution of test framework using a command line
* Generation of .ta files
* Update of test list

### Execution of test framework

This needs some environment variable configuration:

* Add a STF\_HOME environment variable pointing to the folder where STF is deployed
* Add a JAVA\_HOME\_STF environment variable pointing to a Java 8 installation. This is mandatory as for now (Squash TA 1.9), Squash only supports Java 7 whereas STF is built using Java 8.

Execution platform must be installed with Java 8

A typical command line would be:

%JAVA\_HOME\_STF%/bin/java -cp %STF\_HOME%/seleniumtestsframework.jar;%STF\_HOME%/plugins/${application}-tests.jar -Dbrowser=${TC\_CUF\_browser} -DcucumberTests=\"${TC\_CUF\_cucumberTest}\" -DcucumberTags=${TC\_CUF\_cucumberTags} -Dvariables=${TC\_CUF\_testVariables} -Denv=${TC\_CUF\_testEnvironment} org.testng.TestNG ${testngFile} -testnames ${testngName}

Each ${} is a variable passed by the .ta script when replacing variables by their actual value.

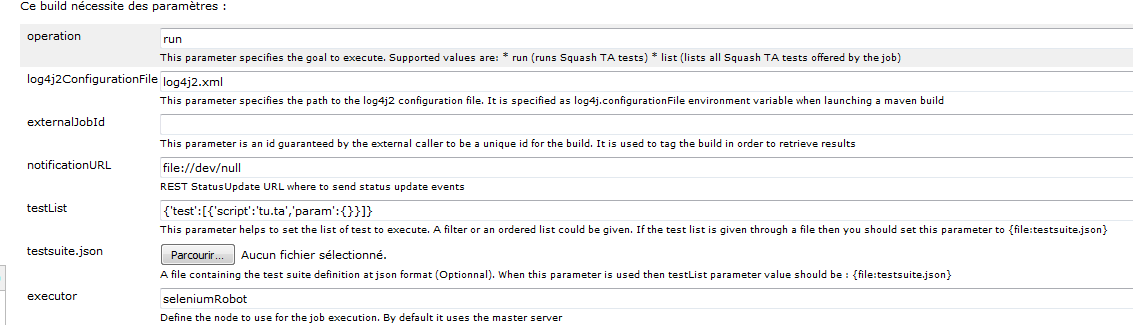
### Squash TM configuration

Launching of STF expects some variables passed to the script. These MUST be declared on Squash TM side (as Test Cas custom fields):

* browser
* testEnvironment



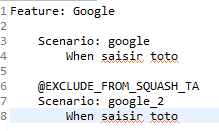
### TA files generation

Before each run, test script updates the list of .ta files so that they reflect the list of available Cucumber Scenario and/or TestNG tests. The generation can also be done launching the Jenkins job with the following parameters:  


Generation is done using a root test file names src/Squash ta/tests/<application>\_generic.ta where some fields will be replaced to reflect the cucumberTest, TestNG xml file name and TestNG test name.

#### Cucumber mode

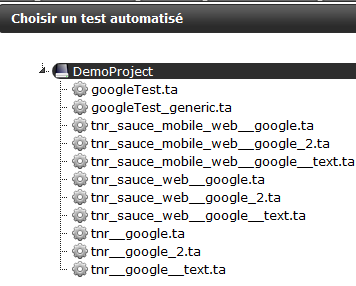
A .ta test script will be generated for each cucumber scenario combined with each TestNG test. When launching this type of .ta script, cucumber variables contained in the testNG.xml file will be overridden.

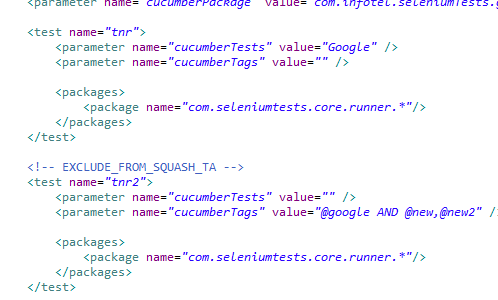
If some of the scenarios should not be available through Squash TA: precede them with the tag “@EXCLUDE\_FROM\_SQUASH\_TA”  


#### TestNG mode

A .ta test script will be generated for each TestNG test.

Below is an example of what has been generated for cucumber scenarios.



If some of the tests should not be available in Squash TA, precede them with the comment: <!-- EXCLUDE\_FROM\_SQUASH\_TA -->  


### Squash TA job configuration

By default, Squash TM receives the Squash TA test report. But, it’s useless when executing tests with STF as it does not contain any details. So we have to configure the STF report to be sent to Squash TM

Add the following line below « Publish HTML reports »



In maven command line (goals & options), replace  
-Dta.tmcallback.reportname= Squash\_TA\_HTML\_Report  
by  
-Dta.tmcallback.reportname=STF\_Report