# **DRLVMInternalTests**

Back to DRLVM Test Tracking

#### Summary

Status

**Building DRLVM tests** 

Running DRLVM tests

1. Running Smoke tests 2. Running C-Unit tests 3. Running Kernel tests 4. Running JVMTI tests

Back to Summary

#### **Status**

DRLVM tests consist in Smoke, C-unit, Kernel and JVMTI tests. All DRLVM tests are DRLVM acceptance tests so they must be run each time the contributor posts his/her patch for DRLVM. It is strictly recommended to test your patch with DRLVM tests in both debug and release build of DRLVM on Windows/Linux/x86/x86\_64 platforms.

Smoke tests are set of java-language programs with standard 'main' entrance which test different DRL VM subsystems like exceptions, GC, Threading and etc.. There are about 80 separate tests. The source of smoke tests are in .../working\_vm/vm/tests/smoke directory.

C-unit tests are set of C-language tests which validate Thread Manager component. C-unit tests are dynamically linked with hythr to test its functionality directly without creating full Java context. To test Java functionality of Thread Manager C-unit tests can be also linked with harmonyvm library and create Java environment through invocation API. There are 80+ tests in this suite.

Kernel tests are set of java-language programs with standard 'main' entrance which test DRLVM Kernel Classes such as java.lang.Class, java.lang.Thread or features such as Java5.0 Generics. There are 660+ tests in this suite.

JVMTI tests are intended to test DRLVM JVMTI support. Each of these tests consists of java test class and JVMTI agent written on C++. Execution of each test class is performed by DRLVM started with test agent.

DRLVM tests located in directories three started from trunk/working\_vm/tests.

Back to Summary

#### **Building DRLVM tests**

To build DRLVM tests you need just to run them in any configuration. For example, to be Kernel tests you may execute the following:

```
cd trunk/working_vm
ant -Dhy.cfg=release -Dtest.mode=jet kernel.test
```

Back to Summary

#### **Running DRLVM tests**

There are common rules of running all of the DRLVM tests.

You can set binary configuration thought a command line option or an environment variable:

```
-Dhy.cfg=[release|debug]
export BUILD_CFG=[release|debug]
```

By default 'debug' mode is set.

You may specify a run mode which is helpful while working on particular issue, for example:

```
-Dtest.mode=srv
```

Also you can specify a given test to be run, for example:

```
-Dtest.case=java.lang.ThreadTest
```

The known point is that if you what to run the test from direct command without ant adaptors, then get execution command from ant debug output, for example with the following command:

ant -d kernel.test

Back to Summary

#### 1. Running Smoke tests

To run all Smoke tests in all supported modes just execute from working\_vm directory:

ant smoke.test

You may see the results of run at directory like:

build/windows\_x86\_msvc\_release/tests/smoke.tests/reports

Back to Summary

## 2. Running C-unit tests

To run all *C-unit* tests in default mode need to execute from working\_vm directory:

ant cunit.test

You can see test execution results at directory like:

working\_vm/build/windows\_x86\_msvc\_release/tests/cunit.tests/reports

Back to Summary

### 3. Running Kernel tests

To run all *Kernel* tests in all supported modes just execute from working\_vm directory:

ant kernel.test

You may see the results of run at directory like:

build/windows\_x86\_msvc\_release/tests/kernel.tests/reports

Back to Summary

#### 4. Running JVMTI tests

To run all  $\emph{JVMTI}$  tests in all supported modes just execute from working\_vm directory:

ant jvmti.test

You may see the results of run at directory like:

build/windows\_x86\_msvc\_release/tests/jvmti.tests/reports

Back to Summary

Back to DRLVM Test Tracking