Android app单体测试以及code coverage生成手顺

Last edited by **caoquanli** 1 month ago

Android系统App单体测试手顺

Table of Contents

系统App单体测试以及CodeCoverage报告生成手顺。

1. 构建测试APK

这个例子将会以原生的Settings APK 单体测试为例子进行描述.Settings Apk的目录: **packages/apps/Settings** ,其单体测试的代码路径: **packages/apps/Settings/tests/unit**。

1.1. 编写单体测试APK代码

测试APK的语法请参考 Build instrumented unit tests，以及 AndroidJUnitRunner 的说明文档。

1.2. AndroidManifest.xml文件的编写

AndroidManifest.xml文件中最关键的点就是 **android:targetPackage**,这个值应该设置为 **被测试APK** 的包名。

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.android.settings.tests.unit"> <!--测试APK的包名-->

    <application>
        <uses-library android:name="android.test.runner" />
    </application>

    <instrumentation android:name="android.support.test.runner.AndroidJUnitRunner"
        android:targetPackage="com.android.settings" <!--被测试APK的包名-->
        android:label="Settings Test Cases">
    </instrumentation>

</manifest>
```

1.3. Android.mk 的构建

Android.mk文件中主要要包含必要的test相关的jar。 注意点如下：

- LOCAL_JAVA_LIBRARIES 中必须要包含 android.test.runner
- LOCAL_STATIC_JAVA_LIBRARIES至少要包含：
  - android-support-test
  - legacy-android-test
  - junit
  - android-support-test
  - 其他的静态库要看测试代码中是否有依赖，有依赖的也要添加进去

下面是SettingsUnitTests的Android.mk文件的写法，可以作为参考：

```
LOCAL_PATH:= $(call my-dir)
include $(CLEAR_VARS)

# We only want this apk build for tests.
```

```
LOCAL_MODULE_TAGS := tests
LOCAL_CERTIFICATE := platform

LOCAL_JAVA_LIBRARIES := android.test.runner

LOCAL_STATIC_JAVA_LIBRARIES := \
    android-support-test \
    espresso-core \
    legacy-android-test \
    mockito-target-minus-junit4 \
    truth-prebuilt \
    ub-uiautomator \
    junit \
    legacy-android-test \
    android-support-test \


# Include all test java files.
LOCAL_SRC_FILES := $(call all-java-files-under, src)

LOCAL_PACKAGE_NAME := SettingsUnitTests
LOCAL_COMPATIBILITY_SUITE := device-tests

LOCAL_INSTRUMENTATION_FOR := Settings <!--被测试的APK-->.

include $(BUILD_PACKAGE)
```

1.4. 编写AndroidTest.xml文件

AndroidTest.xml文件是为了适配Android TradeFaderaiton框架，通过这个文件可以让TF框架自动化运行我们的单体测试。

AndroidTest.xml的注意点如下：

- target_preparer标签中的value应该是测试APK的名字，和Android.mk文件中的LOCAL_PACKAGE_NAME变量名字一致
- CodeCoverageTest 注意点
    - package属性的value，应该写成测试APK的包名
    - runner 要写成 AndroidJUnitRunner
    - metadata-file要等于被测试APK的coverage.em文件的路径
    - source-dir写成被测试APK的代码路径

下面是SettingsUnitTests的AndroidTest.xml的写法，可以作为参考：

```
<configuration description="Runs Settings Test Cases.">
    <target_preparer class="com.android.tradefed.targetprep.InstallBuildEnvApkSetup">
        <option name="apk-name" value="SettingsUnitTests.apk" />  <!--测试apk-->
    </target_preparer>
    <option name="test-suite-tag" value="apct" />
    <option name="test-tag" value="SettingsUnitTests" />
    <test class="com.android.tradefed.testtype.CodeCoverageTest" >
        <option name="package" value="com.android.settings.tests.unit" />  <!--测试APK的包名
        <option name="runner" value="android.support.test.runner.AndroidJUnitRunner" />
        <option name="metadata-file" value="out/target/common/obj/APPS/Settings_intermediat
        <option name="source-dir" value="packages/apps/Settings/src" />  <!--被测试APK代码路径
    </test>
</configuration>
```

1.5. 生成coverage.em文件

为被测试的APK生成测试报告，需要被测试APK的instrumention信息，这个信息保存在coverage.em文件中。首先需要为被测试的APK生成这个文件。 在源代码树根目录下运行如下命令：

```
$ EMMA_INSTRUMENT_STATIC=true mmma packages/apps/Settings -j8
```

命令运行成功之后，在out目录下，相应被测试APK的中间物目录中会生成coverage.em文件。注意 **一定要确保成功的生成了coverage.em文件！！！** 。 对于Settings来说，它的coverage.em文件位于: out/target/common/obj/APPS/Settings_intermediates/coverage.em 。

1.6. 全编系统，烧录系统镜像

对于系统APK来说，因为其APK被打包到system.img或者是vendor.img，这两个分区在user版本中是read-only，无法修改。所以需要全编系统，将instrumention之后的apk烧录到机器上才能进行单体测试 并为之生成coverage报告。

1. 全编系统的命令如下：

```
$ EMMA_INSTRUMENT_STATIC=true make -j8
```

2. 烧录系统。经过上面的全编命令，Settings的APK被instrumention，并打包到了系统镜像中。我们需要把这个镜像烧录到开发板中。

2. 编译生成测试APK

在源码目录下运行 make SettingsUnitTests,在out/target/product/mt2712/data/app/目录下生成测试SettingsUnitTests.apk

3. 使用TF框架运行单体测试并生成单体测试报告

1. 请确保先编译了tradefed，编译请参考 Android Trade Federation框架单体测试
2. 运行 tradefed.sh run packages/apps/Settings/tests/unit/AndroidTest.xml
3. 如果运行成功，终端上会有如下log信息

```
 01-08 10:02:47 I/TestInvocation: Invocation was started with cmd: packages/apps/Setting
01-08 10:02:47 I/TestInvocation: Starting invocation for 'SettingsUnitTests' with '[ Bu
01-08 10:02:47 W/FileUtil: Chmod is not supported by this OS.
01-08 10:02:47 I/InstallBuildEnvApkSetup: Installing SettingsUnitTests.apk on GFEDCBA09
01-08 10:02:51 W/FileUtil: Chmod is not supported by this OS.
01-08 10:02:51 I/TextResultReporter: Saved device_logcat_setup log to /tmp/0/SettingsUn
01-08 10:02:51 I/RemoteAndroidTest: Running am instrument -w -r   -e timeout_msec 30000
01-08 10:02:55 I/InvocationToJUnitResultForwarder: run com.android.settings.tests.unit
........
01-08 10:06:27 I/CodeCoverageTest: coverage file from device: /tmp/coverage_32694487620
01-08 10:06:27 W/FileUtil: Chmod is not supported by this OS.
01-08 10:06:27 I/TextResultReporter: Saved com.android.settings.tests.unit_runtime_cove
01-08 10:06:27 I/CodeCoverageTest: create report dir: out/target/testcases/coverage/com

01-08 10:06:30 I/CodeCoverageTest: Jacoco code coverage report generation success
01-08 10:06:30 W/FileUtil: Chmod is not supported by this OS.
01-08 10:06:30 I/TextResultReporter: Saved device_logcat_test log to /tmp/0/SettingsUni
01-08 10:06:30 W/FileUtil: Chmod is not supported by this OS.
01-08 10:06:30 I/TextResultReporter: Saved device_logcat_teardown log to /tmp/0/Setting
01-08 10:06:30 W/FileUtil: Chmod is not supported by this OS.
01-08 10:06:30 I/TextResultReporter: Saved host_log log to /tmp/0/SettingsUnitTests/inv
```

4. Settings覆盖率报告在out/target/testcases 目录下.