Cordova Test Suite User Guide

Version 1.0

Contents

1   Introduction

This document provides method to run Crosswalk based Cordova Test Suite. Currently the target platform is Android only. You can use the following method to run it with testkit-lite. Testkit tool-chain includes 3 components:

-    testkit-lite: a command-line interface application deployed on Host

-    testkit-stub: a test stub application deployed on Device

-    tinyweb: a web service application deployed on Device

2   Cordova Web Testing Architecture

●   Cordova Web Testing on Android

•   Architecture
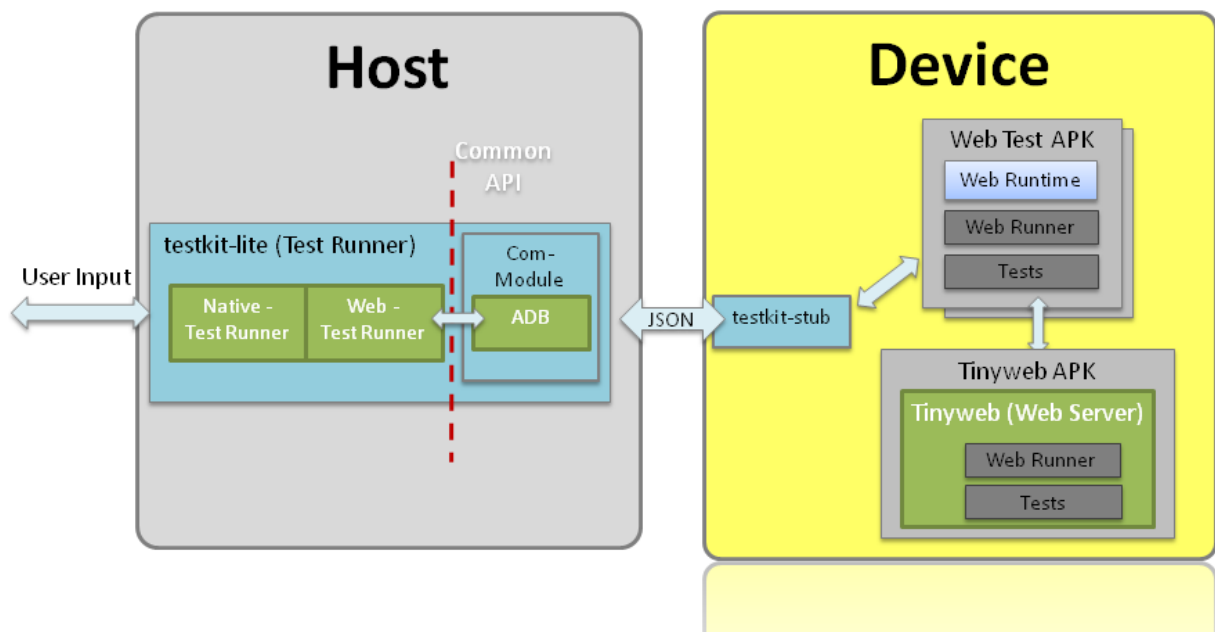
There are two types of Webapi tests:

-    Web service dependent

Client side is a stub test package which link to remote web runner, no local TCs and web runner, thus avoid cross origin issue.

Server side includes tinyweb, webrunner and TCs.

- Web service independent

Self contained test package which include all things - web runner, TCs.

3 Install testkit-lite on Host

- Deploy testkit-lite
    - Install dependency python-requests (version>1.0)

        $ sudo apt-get install python-pip

        $ sudo pip install requests

    - Install testkit-lite from source code in GitHub

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

        $ cd testkit-lite && sudo python setup.py install

4 Crosswalk based Cordova System Requirements

- Java JDK 1.5 or greater

    http://www.oracle.com/technetwork/java/javase/downloads/index.html
- Apache ANT 1.8.0 or greater http://ant.apache.org/bindownload.cgi
- Android SDK http://developer.android.com
- Python 2.7 or greater https://www.python.org/download/
- Node.js 0.10.24 or greater http://nodejs.org/download/

5 Crosswalk based Cordova Developer Tools

The Cordova developer tooling is split between general tooling and project level tooling.

- General Commands

    ./bin/create [path package activity]

    create the ./example app or a cordova android project

    ./bin/check_reqs

    checks that your environment is set up for cordova-android development

    ./bin/update [path]

    updates an existing cordova-android project to the version of the framework

- Project Commands

    ./cordova/clean

    cleans the project

    ./cordova/build

    calls `clean` then compiles the project

    ./cordova/log

    stream device or emulate logs to stdout

    ./cordova/run

    calls `build` then deploys to a connected Android device. If no Android device is detected, will launch an emulator and deploy to it.

    ./cordova/version

    returns the cordova-android version of the current project

6   Web Runtime and Web API Test on Crosswalk based Cordova

- Deploy Android ADT bundle (Android SDK, IDE included) and Android NDK

    -Deploy Android ADT bundle by referring to below link

    http://developer.android.com/sdk/installing/bundle.html

-Deploy Android NDK by referring to below link

http://developer.android.com/tools/sdk/ndk/index.html

● Deploy adb Tool to Host

-Append Android SDK's tools and platform-tools directories to PATH environment

$ export PATH=${PATH}:/path/to/adt-bundle-<version>/sdk/tools:
/path/to/adt-bundle-<version>/sdk/platform-tools

● Download Crosswalk binaries from:

https://download.01.org/crosswalk/releases/android-x86/canary/crosswalk-<version>-x86.zip
https://download.01.org/crosswalk/releases/android-arm/canary/crosswalk-<version>-arm.zip

● In Web Runtime testing, there are two methods to build test builds base on two kinds of test sources:

Build Crosswalk based Cordova app with cordova.tar.gz (there should be cordova.tar.gz in crosswalk-<version>-x86/arm.zip, but it's not ready in download.01.org)

1. Unzip cordova.tar.gz extracted from crosswalk-<version>-x86/arm.zip

   tar zxvf /path/to/cordova.tar.gz

2. /path/to/crosswalk-cordova-android/bin/create testapp com.example.testapp testapp --shared

3. $ cd testapp

4. Copy web source code (e.g. index.html with some contents) to assets/www

5. ./cordova/build

6. ./cordova/run

If you can't get cordova.tar.gz in decompressed crosswalk-<version>-x86/arm.zip, please refer to the steps as below:

1. Extract XWalkCoreLibrary in Crosswalk builds

   Download crosswalk builds from

   https://download.01.org/crosswalk/releases/android-x86/canary/crosswalk-<version>-x86.zip
   https://download.01.org/crosswalk/releases/android-arm/canary/crosswalk-<version>-arm.zip
   $ unzip crosswalk-<version>-arm.zip <path>/
   $ tar zxvf /path/to/crosswalk-<version>-arm/xwalk_core_library.tar.gz

2. Checkout crosswalk-cordova-android

   $ git clone https://github.com/crosswalk-project/crosswalk-cordova-android.git
   The default branch is master.
   $ cd crosswalk-cordova-android
   $ git branch

   Sometimes you may need to switch to target branches e.g. crosswalk-4, if not, please use default master branch.
   $ git checkout -b crosswalk-4 origin/crosswalk-4
   Please check the detailed available branch list via "git branch -a", the "crosswalk-4" etc., branches keep updating by developer.

3. Import XWalkCoreLibrary by linking it to framework folder of crosswalk-cordova-android

   $ ln -s /path/to/xwalk_core_library /path/to/crosswalk-cordova-android/framework/
   Make sure there is only one xwalk_core_library folder rather than xwalk_core_library/xwalk_core_library under /framework/.

   The steps 1~4 are equal to unzipped cordova.tar.gz.

4. /path/to/crosswalk-cordova-android/bin/create testapp com.example.testapp testapp --shared

5. $ cd testapp

6. Copy web source code (e.g. index.html with some contents) to assets/www

7. . /cordova/build

8. . /cordova/run

- Set Permissions
  Some HTML5 APIs which access devices require developers to set appropriate permissions in AndroidManifest.xml to work correctly. For example, if your app calls getUserMedia, it needs to add

      <uses-permission android:name="android.permission.RECORD_AUDIO" />
      <uses-permission android:name="android.permission.CAMERA" />
  into AndroidManifest.xml in /path/to/testapp folder.

The Cordova Mobile Spec test doesn't need testkit-lite etc., tools to run the test, but for Web Runtime and Web API tests, please run the following steps:

- Deploy testkit-stub and launch it
    - Make binary for testkit-stub from source code in GitHub

    ```
    $ git clone git@github.com:testkit/testkit-stub.git
    $ cd testkit-stub/android/jni/ && /path/to/android-ndk-<version>/ndk-build
    ```
    - Import project testkit-stub to Android developer Tool by location testkit-stub/android
    - Export the android project to APK and install APK to android device

    ```
    $ adb install /path/to/TestkitStub.apk
    ```
    - Launch testkit-stub by clicking the testkit-stub App icon in launcher

- Deploy tinyweb and launch it
    - Make binaries for tinyweb from source code in GitHub

```
$ git clone git@github.com:testkit/tinyweb.git
```

```
$ cd tinyweb/android/native/jni/ && /path/to/ android-ndk-<version>/ndk-build
```

- Copy tinyweb/android/native/libs/ to folder tinyweb/android/assets/system/libs/
- Import project tinyweb to Android developer Tool by location tinyweb /android
- Export the android project to APK and install APK to android device

```
$ adb install /path/to/TinywebTestService.apk
```

- Launch tinyweb by clicking the tinyweb app icon in launcher

- Pack test suite package
  Please see *Web_Test_Suite_Packaging_Guide*, Chapter 3.1 "*Pack Web Test Suite Packages for Android*", detailed steps for Cordova test suites package are added.
  Note: For Android device, the default APK package mode of Crosswalk based Cordova is embedded mode.

- Install test suite on Android device

  ```
  $ unzip -o <test_suite_name>-<version>.apk.zip -d /path/to/
  $ /path/to/opt/<test_suite_name>/inst.sh
  ```

- Launch WRT test with lite

  ```
  $ testkit-lite -f /path/to/opt/<test_suite_name>/tests.xml --comm androidmobile
  ```

- Uninstall test suite

  ```
  $ /path/to/opt/<test_suite_name>/inst.sh -u
  ```

7 Cordova Mobile Spec Test on Crosswalk based Cordova

- Build and run Cordova Mobile Spec test build (named as cordova_mobile_spec-debug.apk) on Android

1. Extract XWalkCoreLibrary in Crosswalk builds

   Download crosswalk builds from

   https://download.01.org/crosswalk/releases/android-x86/canary/crosswalk-<version>-x86.zip

   https://download.01.org/crosswalk/releases/android-arm/canary/crosswalk-<version>-arm.zip

   $ unzip crosswalk-<version>-arm.zip <path>/

   $ tar zxvf /path/to/crosswalk-<version>-arm/xwalk_core_library.tar.gz

2. Checkout crosswalk-cordova-android

   $ git clone https://github.com/crosswalk-project/crosswalk-cordova-android.git

3. The default branch is master.

   $ cd crosswalk-cordova-android
   $ git branch

   Sometimes you may need to switch to target branches e.g. crosswalk-4, if not, please use default master branch.
   $ git checkout -b crosswalk-4 origin/crosswalk-4
   Please check the detailed available branch list via "git branch -a", the "crosswalk-4" etc., branches keep updating by developer.

4. Import XWalkCoreLibrary by linking it to framework folder of crosswalk-cordova-android

   $ ln -s /path/to/xwalk_core_library /path/to/crosswalk-cordova-android/framework/
   Make sure there is only one xwalk_core_library folder rather than xwalk_core_library/ xwalk_core_library under /framework/.

5. Fetch Cordova Mobile Spec test cases:

   $ git clone git@github.com:apache/cordova-mobile-spec.git
   $ cd cordova-mobile-spec
   $ git checkout -b 3.3.0 3.3.0

6. Create mobile spec app:

```
$ /path/to/crosswalk-cordova-android/bin/create mobilespec org.apache.mobilespec
mobilespec --shared
$ cd mobilespec
$ cp -r /path/to/cordova-mobile-spec/* assets/www (Please don't accept to overwrite
the cordova.js)
$ cp -r /path/to/cordova-mobile-spec/config.xml res/xml/config.xml
```

7. Set up Cordova Mobile Spec plugins environment. Recommend to use plugman to install plugins. You need to have [node.js](#) installed. Then install plugman by:

```
$ npm install -g plugman
```

8. Add Cordova Mobile Spec plugins for Crosswalk based Cordova, please refer to [full supported plugin list](#):

```
$ cd mobilespec
$ plugman install --platform android --project ./ --plugin https://git-wip-
us.apache.org/repos/asf/cordova-plugin-device.git#r0.2.5
$ plugman install --platform android --project ./ --plugin https://git-wip-
us.apache.org/repos/asf/cordova-plugin-network-information.git#r0.2.5
$ plugman install --platform android --project ./ --plugin https://git-wip-
us.apache.org/repos/asf/cordova-plugin-battery-status.git#r0.2.5
$ plugman install --platform android --project ./ --plugin https://git-wip-
us.apache.org/repos/asf/cordova-plugin-device-motion.git#r0.2.4
$ plugman install --platform android --project ./ --plugin https://git-wip-
us.apache.org/repos/asf/cordova-plugin-device-orientation.git#r0.3.3
$ plugman install --platform android --project ./ --plugin https://git-wip-
us.apache.org/repos/asf/cordova-plugin-geolocation.git#r0.3.4
$ plugman install --platform android --project ./ --plugin https://git-wip-
us.apache.org/repos/asf/cordova-plugin-media.git#r0.2.6
$ plugman install --platform android --project ./ --plugin https://git-wip-
us.apache.org/repos/asf/cordova-plugin-file.git#r0.2.5
$ plugman install --platform android --project ./ --plugin https://git-wip-
us.apache.org/repos/asf/cordova-plugin-file-transfer.git#r0.4.0
$ plugman install --platform android --project ./ --plugin https://git-wip-
us.apache.org/repos/asf/cordova-plugin-dialogs.git#r0.2.4
$ plugman install --platform android --project ./ --plugin https://git-wip-
```

us.apache.org/repos/asf/cordova-plugin-splashscreen.git#r0.2.5

$ plugman install --platform android --project ./ --plugin https://git-wip-us.apache.org/repos/asf/cordova-plugin-console.git#r0.2.5

$ plugman install --platform android --project ./ --plugin https://git-wip-us.apache.org/repos/asf/cordova-plugin-camera.git#r0.2.5

$ plugman install --platform android --project ./ --plugin https://git-wip-us.apache.org/repos/asf/cordova-plugin-media-capture.git#r0.2.8

$ plugman install --platform android --project ./ --plugin https://git-wip-us.apache.org/repos/asf/cordova-plugin-vibration.git#r0.3.5

$ plugman install --platform android --project ./ --plugin https://git-wip-us.apache.org/repos/asf/cordova-plugin-globalization.git#r0.2.4

$ plugman install --platform android --project ./ --plugin https://git-wip-us.apache.org/repos/asf/cordova-plugin-contacts.git#r0.2.5

$ plugman install --platform android --project ./ --plugin https://git-wip-us.apache.org/repos/asf/cordova-plugin-inappbrowser.git#r0.2.4

$ plugman install --platform android --project ./ --plugin assets/www/cordova-plugin-whitelist

9. According to Splash Screen API Spec, you may need to add following statement into the onCreate method of the class that extends DroidGap:

   super.setIntegerProperty("splashscreen", R.drawable.splash);
   in /path/to/mobilespec/src/org/apache/mobilespec/mobilespec.java

   The .java file path maps to package activity etc., package parameters in step 6 "mobilespec org.apache.mobilespec mobilespec"

```
public void onCreate(Bundle savedInstanceState)
{
  super.onCreate(savedInstanceState);
  super.init();
  super.setIntegerProperty("splashscreen", R.drawable.splash);
  super.loadUrl(Config.getStartUrl());
}
```

10. Connect the Android test device to host (adb enabled), build and run:

$ cd /path/to/mobilespec

$ ./cordova/build

Add "--debug" switch if "remote debugging" feature is needed to run the test

$ ./cordova/build --debug

$ ./cordova/run

- The alternate way is copy test apk from /path/to/mobilespec/bin/mobile_spec-debug.apk to device, install it.
- Run Cordova API (Cordova Mobile Spec) test cases in app on test device.