Android CTS

|  |  |  |  |
| --- | --- | --- | --- |
| Approvals | |  |  |
| Name | Function | Date | Signature |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Revision History

| **Rev.** | **Date** | **Author** | **Description** |
| --- | --- | --- | --- |
| 1 |  |  | Initial draft |

Table of Contents

[1 Introduction 6](#_Toc413663529)

[1.1 Purpose 6](#_Toc413663530)

[1.2 Scope 6](#_Toc413663531)

[1.3 Intended Audience 6](#_Toc413663532)

[1.4 Terms, Acronyms and Abbreviations 6](#_Toc413663533)

[1.5 References 6](#_Toc413663534)

[1.6 Document Conventions 7](#_Toc413663535)

[2 Design Overview 7](#_Toc413663536)

[2.1 Why we need CTS? 7](#_Toc413663537)

[2.2 Running the CTS 7](#_Toc413663538)

[2.2.1 Setting up your host machine 7](#_Toc413663539)

[2.2.2 Setting up your devices 8](#_Toc413663540)

[2.2.3 Running CTS test 8](#_Toc413663541)

[2.3 Interpreting the CTS test results 10](#_Toc413663542)

[3 Known Risks and Issues 10](#_Toc413663543)

[3.1 Risks 10](#_Toc413663544)

[3.2 Open Issues 10](#_Toc413663545)

Table of Figures

Table of Tables

# Introduction

The document is used to describe android CTS(Compatibility Test Suite) test.

## Purpose

Specify the purpose of the document in this section.

The CTS is a free, commercial-grade test suite, available for [download](https://source.android.com/compatibility/downloads.html). The CTS represents the "mechanism" of compatibility. The CTS runs on a desktop machine and executes test cases directly on attached devices or an emulator. The CTS is a set of unit tests designed to be integrated into the daily workflow (such as via a continuous build system) of the engineers building a device. Its intent is to reveal incompatibilities early on, and ensure that the software remains compatible throughout the development process.

This design document is used to describe the overview of CTS and also the details steps of how to run CTS test.

## Scope

Briefly describe the scope of this document. What project(s) it is associated with and anything else that is affected or influenced by this document.

This Software Design Document contains all CTS related issues.

## Intended Audience

Specify who the audience targeted by this document is.

## Terms, Acronyms and Abbreviations

List in this section any terms, acronyms, and abbreviations required for understanding this work product.

Below are the terms, acronyms, and abbreviations used within this document. Additional project-specific terms can be found in the project glossary [GLOS].

|  |  |
| --- | --- |
| Term, Acronym, Abbreviation | Definition |
| CTS | Compatibility Test Suite |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

## References

List in this section any external references and/or documents which are necessary to understand this work product.

|  |  |  |
| --- | --- | --- |
| Abbreviation | Reference Element | Description |
| Google CTS Home Page | <https://source.android.com/compatibility/cts-intro.html> |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## Document Conventions

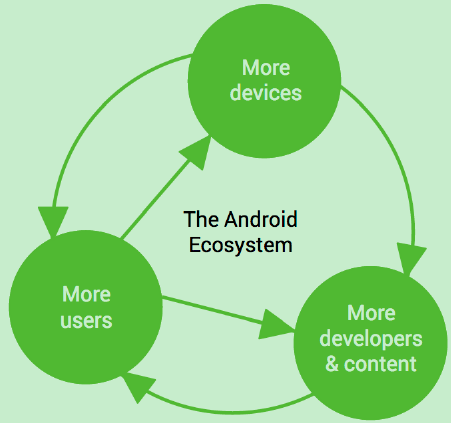
Document any particular conventions used throughout this document, including font styles used to distinguish different information elements and graphical notations.

Describe key design principles that should be followed, and standards that apply. Examples may include the use of certain design methods such as object-oriented analysis and design, specific design patterns such as observer, factory, etc, preference for persistent data rather than in-memory data, etc. Ideally, reference external documents in case of design principles and standards applicable across multiple software components and/or projects done by an organization/team.

# Design Overview

## Why we need CTS?

The Android compatibility program makes it easy for mobile device manufacturers to develop compatible Android devices.



* Users want a customizable device.

A mobile phone is a highly personal, always-on, always-present gateway to the Internet. We haven't met a user yet who didn't want to customize it by extending its functionality. That's why Android was designed as a robust platform for running aftermarket applications.

* Developers outnumber us all.

No device manufacturer can hope to write all the software that a person could conceivably need. We need third-party developers to write the apps users want; so the Android Open Source Project aims to make it as easy and open as possible for developers to build apps.

* Everyone needs a common ecosystem.

Every line of code developers write to work around a particular phone's bug is a line of code that didn't add a new feature. The more compatible phones there are, the more apps there will be. By building a fully compatible Android device, you benefit from the huge pool of apps written for Android, while increasing the incentive for developers to build more of those apps.

Generally, building a compatible device is a three-step process:

1. *Obtain the*[*Android software source code*](http://source.android.com/source/index.html). This is the source code for the Android platform that you port to your hardware.
2. *Comply with the*[*Android Compatibility Definition Document (CDD)*](http://source.android.com/compatibility/android-cdd.pdf). The CDD enumerates the software and hardware requirements of a compatible Android device.
3. *Pass the*[*Compatibility Test Suite (CTS)*](http://source.android.com/compatibility/cts-intro.html). Use the CTS as an ongoing aid to compatibility during the development process.

After complying with the CDD and passing the CTS, your device is now Android compatible. Android apps in the ecosystem will have a consistent experience on your device.

## Running the CTS

### Setting up your host machine

1. Make sure you have a recent version of Android Debug Bridge(adb), to install adb, download Android SDK tool and set up an existing IDE:

[*http://developer.android.com/sdk/index.html#ExistingIDE*](http://developer.android.com/sdk/index.html#ExistingIDE)

[*http://developer.android.com/sdk/installing/index.html*](http://developer.android.com/sdk/installing/index.html)

Ensure ‘adb’ is in your system path:

export PATH=$PATH:$HOME/androidsdklinux\_x86/platformtools

1. Download the CTS package(CTS&CTS media) matching the Android version and all the ABIs(Application Binary Interface) your device support. Unzip to your local folder:



### Setting up your devices

CTS should be executed on user build devices only. Follow these instructions:

1. Make sure your device is running a user build
2. Plug in an empty SD card.
3. Factory data reset the device
4. Set your device’s language to English(United States) from settings->Language&Input->Language
5. Turn on the location setting if there is a GPS or WIFI/Cellular network available
6. Connect to a Wifi network that supports IPv6 and has an internet connect(For SZ mobility CTS test, please choose DD-WRT-CTS, the passwd is 12345678)
7. Make sure no lock pattern is set on the device(Settings -> Security -> Screen Lock -> None)
8. Check the “USB Debugging” development option(Settings -> Developer options -> USB debugging)
9. Connect the host machine that will be used to test the devie, and “Allow USB debugging” for the computer’s RSA key fingerprint
10. Check Settings -> Developer options -> Stay Awake
11. Check Settings -> Developer options -> Allow mock locations
12. For android-cts-4.4\_r3, setup your device to run the device administration tests:

adb install –r android-cts/repository/testcases/CtsDeviceAdmin.apk

On the device, Settings -> Security -> Select device administrators, enable only the two:

android.deviceadmin.cts.CtsDeviceAdminReceiver\*

Make sure android.deviceadmin.cts.CtsDeviceAdminDeactivatedReceiver and any other preloaded device administrators stay disabled in the same menu.

1. If device supports video codecs, the CTS media files must be copied to the device
   1. Go to the unzipped media folder
   2. Change the file permissions: chmod u+x android-cts-media-1.1/copy\_media.sh
   3. Run the “bash copy\_media.sh all”(this will copy all files default)

### Running CTS test

#### Fully test

1. Make sure java 1.6, in sz server environment, you can use source /usr/local/android/jdk-1.6 to configure.
2. Launch the CTS console with **root permission** by running the cts-tradefed script from the folder where the CTS package has been unzipped, e.g.

# ./android-cts/tools/cts-tradefed

1. You can start default test plan(Containing all of the test packages) by appending:

run cts –plan CTS

Enter list plans to see a list of test packages in the repository

Enther list packages to see a list of test packages in the repository

1. Alternately, you may run the CTS plan of your choosing from the command line using:

Run cts –plan <plan\_name>

1. View test process and results reported on the console
2. While a device is running tests, it must not be used for any other tasks and must be kept in a stationary position (to avoid triggering sensor activity) with the cameras pointing an object that could be focused.
3. Do not press any keys on the device while CTS is running. Pressing keys or touching the screen of a test device will interfere with the running tests and may lead to test failures.

#### Test unfinished plan

1. Launch the CTS console by running the cts-tradefed script, e.g.

$ ./android-cts/tools/cts-tradefed

1. Input command: l r

Session Pass Fail Not Executed Start time Planname

0 7 0 0 2012.01.16\_16.09.19 NA

1 13 0 0 2012.01.16\_16.13.01 NA

2 19 9 17120 2012.01.18\_13.43.56 CTS

1. Based on the information, you can continue your wanted session:

run cts –continue-session <session\_id>

#### Test failed cases

1. Input command in CTS console: l r

Session Pass Fail Not Executed Start time Planname

0 7 0 0 2012.01.16\_16.09.19 NA

1 13 0 0 2012.01.16\_16.13.01 NA

2 19 9 17120 2012.01.18\_13.43.56 CTS

1. Add derived plan: add derivedplan – plan <plan\_name> -s <session\_id> -r fail(if you want test notExecuted cases, please use “-r notExecuted”, for timeout cases, please use “-r timeout”)
2. run cts –plan <your\_plan\_name>

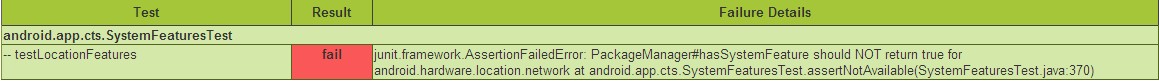
#### Test single package/case

1. Input command in CTS console to list all packages: l p
2. run cts –c <package\_name>, for example, to test android.bluetooth, input command:

run cts –c android.bluetooth

1. run specific case:

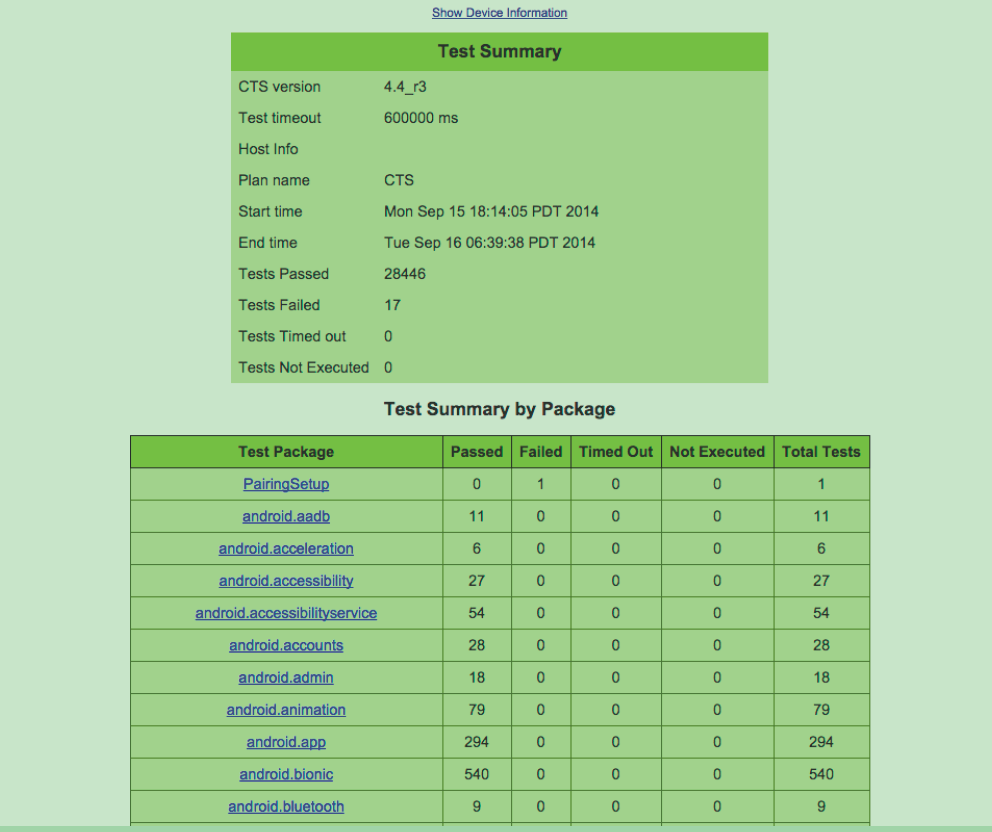
run cts -c package\_name --m method



## Interpreting the CTS test results

The test results are placed in the file:

$CTS\_ROOT/android-cts/repository/results/<start time>.zip, inside the zip, the testResult.xml file contains the actual results.



This is followed by details of the the actual tests that were executed. The report lists the test package, test suite, test case, and the executed tests. It shows the result of the test execution—pass, fail, timed out, or not executed. In the event of a test failure details are provided to help diagnose the cause.

Provide an overview of the design specific to the items identified in the scope. In particular, provide information about the relationships between the design elements. Specify the structure of the design elements identified in the scope. Use, for example, a UML class diagram showing the elements and their relationships. Describe how relevant non-functional requirements have been addressed.

List any considered design alternatives and provide rationales for selecting as well as not selecting a certain design.

# Known Risks and Issues

N.A

## Risks

N.A

Describe any risks associated with the software design and proposed mitigation strategies. No information already captured in other places should be repeated here. If you are maintaining a central risk register for your project or using a tool such as JIRA or MKS for capturing and tracking your risks then this section is not required and can be removed. Alternatively, you can provide a link to those documents or a link to a filter for relevant issues in your issue tracking tool. Make sure to capture “Design” as source for design-related risks.

## Open Issues

List any open issues (i.e., TBDs, pending decisions, information that is needed, conflicts awaiting resolution, and the like) which need further investigation and/or attention. No information already captured in other places should be repeated here. If you are using a central action register or issue tracking tool such as JIRA or MKS for your project then this section is not required and can be removed. Alternatively, you can provide a link to those documents or a link to a filter for relevant issues in your issue tracking tool.

N.A