Software Manual

Android Terminal Test

Author: Marco de Abreu

Version 1.0

Last update: August 2016

1 Table of Contents

2 Installation

This framework needs multiple software environments in order to operate. The following chapters will guide trough crucial installation steps.

2.1 Requirements

This application can only be run on Microsoft Windows due to the dynamic script compilation. You may be able to start it on UNIX environments, but executing a device-action will fail.

2.2 Java Environment

The Java Environment allows to start this application and provides dynamic script compilation features. Please download the latest Setup files and follow the instruction assistant.

JDK 7: http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html

JRE 8: http://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html

Please note down the installation paths you choose during the installation process.

2.3 Android SDK

The Android Software Development kit provides features to establish a connection between an android device and this application as well as executing dynamic scripts.

Go to https://developer.android.com/studio/index.html and download the windows installer for command line tools:

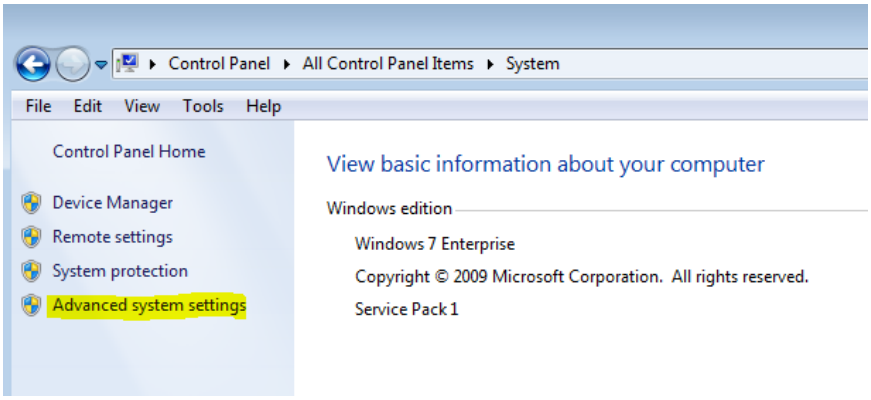
Get just the command line tools

If you do not need Android Studio, you can download the basic Android command line tools below.

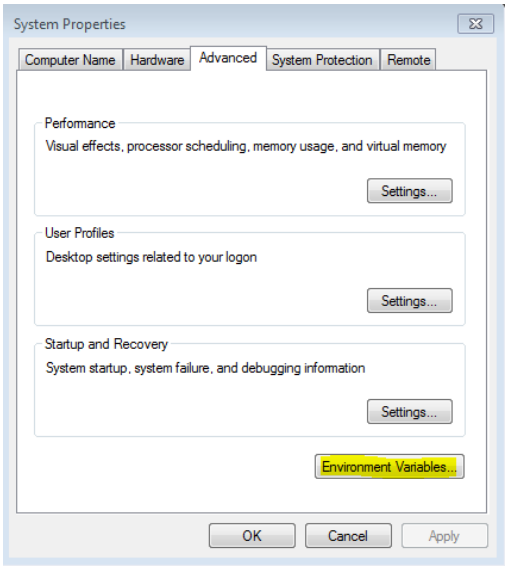
Platform	SDK tools package	Size	SHA-1 checksum
Windows	installer_r24.4.1-windows.exe	144 MB (151659917 bytes)	f9b59d72413649d31e633207e31f456443e7ea0b
	android-sdk_r24.4.1-windows.zip No installer	190 MB (199701062 bytes)	66b6a6433053c152b22bf8cab19c0f3fef4eba49
Mac OS X	android-sdk_r24.4.1-macosx.zip	98 MB (102781947 bytes)	85a9cccb0b1f9e6f1f616335c5f07107553840cd
Linux	android-sdk_r24.4.1-linux.tgz	311 MB (326412652 bytes)	725bb360f0f7d04eaccff5a2d57abdd49061326d

Also see the SDK tools release notes.

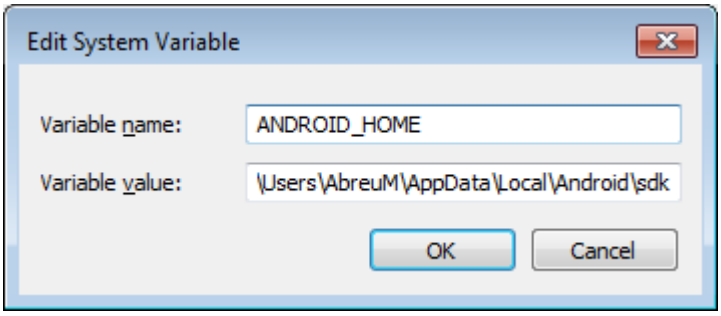
Finish the installation by providing the installation path in the ANDROID_HOME environment variable. Open the *Windows Control Panel*, go to System and access the Advances system settings on the left:



Click on Environment Variables… in the new dialog:



Add a new System variable by pressing New…. In the popped up dialog, enter “ANDROID_HOME” as name and the path to the Android SDK as value.



The path usually should be “C:\Users\USERNAME\AppData\Local\Android\sdk”. Note this path down for later usage.

2.4 Drivers

Android devices use manufacturer depended drivers to establish a connection. You may either install the drivers yourself or use a universal installer like the one you find at http://adbdriver.com/downloads/. Plug in the device you would like to use and open this tool to install the drivers.

You may need a serial cable to make use of the Koppelfeld-Script. Please refer to the manual of the USB-to-serial-converter you are using.

2.5 Application

Copy this application to a path you desired and open “AndroidTerminalTest.jar”. This will start the application. Starting the application the first time allows to reference the installed environments by providing the paths you noted down in the previous steps.

The first dialog with the title “Select javac.exe from JDK7” requires you to select the Java-Compiler of JDK7. You usually find it at the following path: “C:\Program Files\Java\jdk1.7.0_VERSION\bin\javac.exe\” Please make sure you are referencing the JDK7 path.

The second dialog with the title “Select dx.bat Android-SDK build-tools” requires you to select the Android-Converter of the Android SDK. You usually find it at the following path:
“C:\Users\USERNAME\AppData\Local\Android\sdk\build-tools\LATEST_VERSION\dx.bat”

In case you would like to reset the configuration, delete the file located at “/config/config.xml” relative to the application home directory.

3 Scripts

The actions of this application during the runtime are completely controlled by user-supplied scripts. Android terminal test does not introduce a new scripting language but uses regular java code to allow defining actions. The code has to meet the following requirements:

- Package specified
- Methods to be used by profiles have the correct parameters signature and are static
- Checked exceptions may be thrown

Scripts are able to interact with other files which allows complex class structures. One class may call methods of another class. The directory structure to save scripts is as following: “/scripts/TYPE/” whereas the type can be either “host” or “device” according to where the script may be executed. In case an external library should be used, place the .jar-File in the “script/TYPE/lib”-directory.

A script must always validate its behavior and only return if the expected outcome has been reached. Thrown exceptions will cause the execution to fail and be presented to the user, so try to keep them as informative as possible.

3.1 Script-Types

Dynamic scripts receive a runtime during the execution which varies depending on the target machine. Each type requires different things to be taken care of while developing scripts-actions.

3.1.1 Host

Host-scripts are executed on the computer running the application. A method to be used by a profile must have the following signature:

```
public static void METHODNAME(ScriptRuntimeContainer runtime) [throws ...] {
}
```

The `ScriptRuntimeContainer` supplies access to the following methods:

```
public Map<String, Object> getParameters();
public <T> T getParameter(String key);
public DataStorageHostProxy getDataStorage();
```

`getParameters()` and `getParameter(String key)` give access to the parameters supplied by the profile. `getDataStorage()` provides access to the data storage, which will be covered in chapter 3.2 „Storage“.

3.1.2 Device

Device-scripts are executed on the testing device running the client application. A method to be used by a profile must have the following signature:

```
public static void METHODNAME(DeviceRuntimeContainer runtime) [throws ...] {
}
```

The `DeviceRuntimeContainer` supplies access to the following methods:

```
public Map<String, Object> getParameters();
public <T> T getParameter(String key);
public DataStorageDeviceProxy getDataStorage();
public Context getAppContext();
```

The only addition in this case is `getAppContext()`. This method grants access to the context of the client application in order to use context-related android methods.

3.2 Storage

Some data may be required to be persisted longer than for the duration of the current method call. Therefor the developer of a script is able to use the `DataStorage`. This class provides scoped access to data, which allows to save data separately for every device and depending on the current execution state.

3.2.1 Scopes

The lifetime of data saved in the data storage depends on the storage scopes, which consist of the following:

Application: Data is shared with every script and may only be purged when the application is closed.

Device: Data is saved separately for every device and gets purged upon disconnection.

Profile: Data is saved during the execution of the current profile and will be purged when it finishes.

DeviceProfile: Combination of Device and Profile. Data will be purged after finishing the profile, but is kept separately for every device.

3.2.2 Usage

Data saved in the data storage has to be Serializable in order to allow the transfer between the host and client application. Access to the data storage is granted by the two following methods:

```
public <T extends Serializable> void saveData(String key, StorageScope scope, T
     data);
public <T extends Serializable> T getData(String key);
```

`saveData()` allows to save data in the storage. Keys should always be unique to prevent name clashing with different scripts or inconsistency due to scopes. Data already present under the same key will be overridden. The scope decides about the storage conditions.

getData() retrieves data assigned to the specified key while taking the specified scope into account.

4 Profiles

Profiles allow to define the testing criteria and actions to decide whether a device meets the requirements. Therefor it's possible to define actions which will be executed in sequence. There are various type of xml-elements you can use in a profile, which will be covered in this chapter.

4.1 General

The top-most-element of the profile consists of the following attributes:

```
<AttProfile id="TestId" description="Testdescription" name="Testname">
</AttProfile>
```

You may set the id to some test-case identifier, it should be unique. The name will be displayed in the user interface.

4.2 Actions

Actions are the core of this application and are the actual elements which allow to execute something on the host or device. Actions are represented by scripts, which have been introduced in chapter 3.

4.2.1 ActionHost

An ActionHost-Element has the following attributes:

```
<ActionHost path="UserInteraction/Message" method="showMessage" timeout="5000"
name="Show message">
```

The attribute “path” points to the java-file in the scripts directory. In this example the file would be located at “/scripts/host/UserInteraction/Message.java”. Inside this java-file there will be a method called “showMessage”, which matches the required method signature. A timeout value higher than 0 defines after how many milliseconds without returning the action will count as failed – this may be the case if the action waited for the results of its behavior but the event was not fired because of a bug on the device. This will be an indicator for the tester, that there may be an issue which needs to be verified manually. The attribute “name” will be shown to the user in the overview.

4.2.2 ActionDevice

An ActionDevice-Element has the following attributes:

```
<ActionDevice targetDevice="device1" path="Phone/Calls" method="startCall"
timeout="60000" name="Call Number">
```

The attribute “targetDevice” defines the alias of the device on which the action will be executed. A user will be able to assign an alias to a paired device. All other attributes are unchanged to ActionHost.

4.3 Parameters

Actions may be supplied with parameters during the runtime. The may always contain of a non-empty and unique key as well as some type of value.

4.3.1 TextParameter

A TextParameter-Element represents a constant value of the following signature:

```
<TextParameter key="message" value="Call with dynamic number" />
```

4.3.2 ParameterScriptDevice

A ParameterScriptDevice allows to evaluate a script on the device during the runtime in the same way actions are executed. The evaluated value is then used as a parameter for the parent action.

```
<ParameterScriptDevice targetDevice="device1" path="Phone/Informations"
method="getPhoneNumber" timeout="60000" key="phoneNumber"/>
```

4.3.3 ParameterScriptHost

A ParameterScriptHost allows to evaluate a script on the host during the runtime in the same way actions are executed. The evaluated value is then used as a parameter for the parent action.

```
<ParameterScriptHost path="UserInteraction/Message" method="getInput" timeout="0"
key="message"/>
```

4.4 Wait

The execution of a script may be paused for some timeframe with the following element:

```
<Sleep durationMs="10000" name="Wait 10s"/>
```

4.5 Loops

The execution of a script may also contain some repeating elements. This is achieved with a While-Element of the following signature:

```
<While name="Something>
    CONDITION
    CHILD-ACTION(s)
</While>
```

The first element in a while-element has to be a condition. All remaining elements may be regular actions following the exact same layout as before. An infinite loop could look like the following:

```
<While name="Something>
    <Equals>
        <TextParameter value="100"/>
        <TextParameter value="100"/>
    </Equals>

    <Sleep durationMs="10000" name="Wait 10s"/>
</While>
```

4.5.1 Conditions

Conditions may be made of logical and evaluateable elements. Logical elements take the result of an evaluateable element and aggregate it accordingly. Evalulatable elements actually make a comparison and decide whether the condition is met or not.

4.5.1.1 And

An And-Element may contain other conditions and only returns true, if all children were true as well. The element is of the following structure:

```
<And>
    <Equals>…</Equals>
    <Or>
        <Equals>…</Equals>
        <Equals>…</Equals>
    </Or>
</And>
```

4.5.1.2 Or

An Or-Element may contain other conditions and returns true as soon as at least one child is true. The element is of the following structure:

```
<Or>
    <Equals>…</Equals>
    <And>
        <Equals>…</Equals>
        <Equals>…</Equals>
    </And>
</Or>
```

4.5.1.3 Inverse

An Inverse-Element may contain a single other conditional element and inverse its value. A return value of false will be switched to true and vice versa. The usage is as following:

```
<Inverse><Equals>…</Equals></Inverse>
```

4.5.1.4 Equality

An Equality-Element actually compares two values and evaluates them. It must contain two parameters which follow the same structure as action-parameters. You may omit the key of the parameter, it will not be evaluated.

```
<Equals>
    <TextParameter value="test1/>
    <TextParameter value="test1/>
</Equals>
```
You may also check if two values are not equal

```
<Not-Equal>
    <TextParameter value="+491234567"/>
    <ParameterScriptDevice targetDevice="device1" path="Phone/Informations"
method="getPhoneNumber" timeout="60000"/>
</Not-Equal>
```

5 Troubleshooting

Device not appearing in devices-tab: In case the device does not appear, make sure the USB drivers are installed properly. You can also use the adb shell and use the command “devices list” to ensure the problem is not on the application level.

Client-App closing or not responding: Attach adb logcat to the device and look for error messages in the device log.

Host-App closing or not responding: Open app.log in the home directory of the application and check for errors.

Compilation-errors: Make sure you selected the correct compilers during the first start and that your source files don’t contain any errors like missing imports or libraries.

6 References

Titlepic: By Android Developers - Android, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=27871625