Guide to Symbian Signed Testing

This article is an in-depth look at Symbian Signed testing. It's not expected that everyone using Symbian Signed needs to read this document, but it provides a useful reference if you have questions about how your application may be affected by a particular test.

We also actively encourage contributions to this guide from developers, testers and others interested in the Symbian Signed process. Please feel free to update this document based on your experiences of the Test Criteria. **Please note: The test criteria document has been changed lately, and not all changes have been reflected on this page yet**.

General Symbian Signed Testing Guidelines

Who do the test criteria apply to?

All applications submitted to Symbian Signed - through either Express Signed and Certified Signed - must comply with the Symbian Signed Test Criteria.

What to do before you submit your application

If this is your first time using Symbian Signed, then you should check our article which gives you the Ten Top Tips To Passing Symbian Signed Testing so that you avoid failing the tests.

Before you submit your application to Express Signed or Certified Signed, you should check it against the test criteria. If you see any problems, then you should contact the Symbian Signed team at Symbian to discuss it before making your submission. Some applications will require waivers, and some others will require a bit of help from Symbian before you can pass the testing, and it's much easier for us to help you with that before you submit than after your application has failed testing.

Submission Checks

The submission checks are carried out before testing begins on your application. If you fail on any of these checks, then your application will not be tested by the test house and the application will be return with "failed" status and will not be signed.

Check 1

Every submission must pass this test.

The portal will check that your submission is signed with a Publisher ID.

During testing the test house will verify that the Publisher ID used is owned by the submitting developer.

There are no waivers acceptable to this check; if you do not use a Publisher ID which you own to sign your submission you will fail testing.

If an Express Signed audit fails on this test case then action will automatically be taken against your account.

Check 2

You should always give your application a UID from the protected range and which is allocated to the account you're using for submission.

If it's not possible for you to do this, then you should contact the Symbian Signed team at Symbian (symbiansigned@symbian.org) to discuss what can be done.

Generally, no waivers will be granted against this test.

Check 3

This check simply ensures that all information has been provided by the submitter. When running this check, the test house will ensure that you have completed the declarative statement for Restricted Capabilities, and that you have provided a readme.txt file containing all relevant information with your submission.

Test 1 - Installation

Testing Guidelines

Most applications will pass this test easily. The majority of applications install using the normal system installer and will present an icon in the application manager once installed. Some applications place their icons into folders under the application manager, so if the icon isn't in the normal place, do check for new folders. An application still passes this test if its icon is in a sub-folder, although this behaviour makes the phone cluttered, and is not recommended.

If the application doesn't present an icon in the usual way once installed, then the developer should provide extra information with the submission to explain how the test house check that it's installed. If the developer hasn't provided this information, then the tester will contact the developer to ask for the information. If the developer doesn't provide the information, then this test will be failed.

Once the developer provides the required information to the test house, testing can continue.

If it's not possible to find an icon for the application and the developer hasn't provided any information on how to verify that installation has been successful, then this test is failed.

When comparing the version numbers, some common sense should be used. 8.0.1(514) is considered close enough to 8.0.1 or 8,0,1 or 8.0.1 to pass the test. This step should only be failed when the version numbers are obviously different in meaning.

Advice for Developers

As mentioned above, most applications install in the usual way. In fact, as a developer, you would have to make a conscious effort to change this for your application.

If you have changed the way your application installs, though, you should make this clear with your submission so that the test house don't need to contact you for further information.

If your application icon isn't in the normal place, then in your readme.txt file, you should say where the icon can be found. If your application doesn't present an icon at all, then you need to give details of how to confirm it's been installed successfully.

Waivers

The only acceptable waiver for this test would be if an application is designed to be pre-installed onto a device, and this should be clearly explained when the waiver request is submitted.

No other waivers will be accepted against this test.

Test 2 - Application Start/Stop Behaviour

This test ensures that the user has control over whether the application is running on the device or not. There are two ways to close an application - from within the application itself using "Exit" or similar option, or from the system task manager. This test checks both of those methods of closing the application.

Testing Guidelines

You will need to note from Test 1 where to find the icon to start the application you're testing. Once that is done, running this test should be rather straight-forward.

Some applications do not appear in the task manager and the test house must exercise some judgement to decide whether this is necessary behaviour for the application or not. If the application does not appear in the task manager, then the developer must explain this during the submission process - any application which does not appear in the task manager without explanation will fail this test.

The test house will read the justification given by the developer. If there is a genuine reason why the application does not appear in the task manager, then it's acceptable that the application doesn't appear there. Cosmetic reasons are not justification enough.

However, the application should still provide a way for the user to exit even if it does not appear in the task manager. This could be provided as part of the application UI, or elsewhere on the device, but if it's not obvious then the developer should explain this during submission.

Any application which cannot be closed by the user requires a waiver, as described below.

Advice for Developers

Generally, there should be no reason why you want to prevent your application being stopped by the Task Manager or have it not appear there.

If there is a genuine reason why your application doesn't appear in the task manager, then you should explain this reason with

your submission.

If your application appears in the task manager, then the user must be able to kill it from there. If your application appears in the task manager, but can't be killed, you'll need to raise a waiver before making your submission.

Waivers

Applications which don't present an "exit" option or those which can't be stopped at all by the user will require a waiver. The waiver request should explain the reasons behind this and must provide technical justification.

Test 3 - Application Credentials

This step verifies that the developer provided accurate information with their submission.

Testing Guidelines

This test has a necessary element of subjectivity. There are two elements to this test.

Firstly, the simpler of the two tests checks that the application name on the device matches that provided by the developer at submission time. The names should match closely. For instance, if the application was submitted with the name "Ph Mgr" and is called "Phone Manager" on the device, that's acceptable. If it was called "Weather Report" during submission and "Phone Manager" on the phone, that wouldn't be.

The second part of this test establishes that the developer has described the application functionality accurately during submission. We can't expect the developer to write a long explanation of their application during submission, but we expect one or two sentences which sum up the application.

When checking that this statement is correct, the test house will take a pragmatic approach. For instance, if the developer says during submission "this application displays the latest weather reports for your area" then you should check that the application does indeed display weather reports - the application may have other functionality but as long as the developer has described the main functionality accurately, then this test should be passed.

The point of this test is largely to catch cases where the description given by the developer is misleading.

A blank or meaningless description will result in failure of this test.

Advice for Developers

This test should cause no problems for developers. When you make your submission, you simply need to give the name and description of your application accurately.

The description should be short and describe the main functionality of your application. You don't need to provide a description of all of the functionality, just a sentence of two which describes briefly what your application does. The description you provide will only be available to Symbian and the test house so there should be no reason not to provide the information.

Waivers

No waivers will be accepted for this test.

Test 4 - No disruption to voice calls

Phone calls are a key feature of every mobile phone, and this test verifies that the application does not prevent the user from making or receiving voice calls.

Testing Guidelines

The voice call testing must be performed with the application "installed and running". It's worth considering what we mean by "running" this case.

For applications which use the `MultimediaDD` capability, it's important to check that any audio streaming functionality that the application has doesn't interfere with the audio of the voice call. So, when running this test, you should ensure that if the application has audio streaming functionality, that it's actually streaming when you perform the testing.

For other applications, you should ensure that the functionality most likely to interfere with voice calls is currently being exercised within the application. For instance, if the application plays video, then you should use that functionality whilst conducting this test.

VoIP applications must have particular attention applied when being tested against this test case, but that is described in the section on Symbian Signed testing for VoIP applications.

What do we mean by "incoming call must be indicated"?

Testing Emergency Calls

We can't expect developers to make calls to the emergency services in order to pass this test, and so it is acceptable to submit your application without having run this test provided you have tested that your application does not prevent non-emergency voice calls being made.

However, this functionality will be tested by the test house as part of an Express Signed audit or Certified Signed test round and so your application must not prevent emergency calls being made.

Advice for Developers

The important thing with this test is to know in advance whether your application will require a waiver. If your application is designed to interfere with voice calls - for instance an application which places a parental lock on which numbers can be called from the phone - then you will need a waiver to pass this test.

It's also important to check that your application doesn't interfere with any aspect of voice calls. For instance, you should check that you don't do anything which prevents audio or visual notification of the incoming call.

Waivers

Any application which interferes with voice calls will need a waiver for this test. Waivers are now accepted for applications submitted through Express Signed, but you'll need to apply for the waiver in advance. In order to get the waiver approved, you'll need to explain in what way the application interferes with voice calls, and justify this in the context of your application functionality.

For instance, if your application is a parental lock for the device, then you need to state that in the waiver request.

Waivers will be granted where it's appropriate for the given application to do so. But, for instance, a waiver wouldn't be accepted which allowed a game to prevent the user making phone calls.

Test 5 - No disruption to text messages

Text messaging has become key functionality in phones. As such, this test ensures that the application does not unnecessarily restrict the user's ability to send and receive text messages.

Testing Guidelines

In many ways this test is similar to Test 4 which checks interference (or lack of it) with voice calls.

The application should be running in the sense that functionality should be running within the application which may potentially interfere with text messages.

Advice for Developers

If your application doesn't interact with messaging functionality at all, then this test should be passed easily by your application.

If your application does change the way users receive or send their text messages, then you must mention this along with your submission. You will fail this test if you do not mention in advance any behaviour which affects text messages or which affect the sending or receipt of text messages without informing the user.

Waivers

Again, this test has much in common with test 4. If your application interferes with text messages by design, then you must declare this by requesting a waiver for the behaviour

Test 6 - Auto-start behaviour

Some applications are designed to start when the phone starts. This test seeks to check that applications which do that are sufficiently checked to ensure that their auto-start behaviour doesn't threaten the user or the phone.

Testing Guidelines

Part of this involves knowing whether the application starts at boot or not. If the application doesn't start at boot time, then this test

doesn't apply. So the first thing to do is check whether the application is designed to start at boot time.

You also need to check that the user can turn the auto-start option off, and that when turned off, the application doesn't start at boot time.

You can refer back to how you checked that the application was running in Test 2 to see whether it's running or not after start.

Don't forget that it may take a short time after boot of the phone before the application starts.

Any application which doesn't present the option for the user to turn off autostart will fail this test.

Advice for Developers

If you don't use auto-start then your application doesn't need to pass this test. The vast majority of applications shouldn't auto-start with the phone boot anyway. Only if your application is providing a service to the user that they expect will always be on should you implement auto-start functionality.

If you're writing a game or utility, then you shouldn't implement auto-start and you do not have to worry at all about this test case.

If you do implement auto-start functionality, then you must provide the user with a method to turn it off. This can be provided in your application settings dialog.

Waivers

Any application which auto-starts at phone boot and which doesn't provide the user with a way to turn that functionality off will require a waiver for this test. Waivers will only be granted in exceptional circumstances, as there are generally no reasons why the user shouldn't be given the option to disable the application.

Test 7 - No disruption to key device applications

User data is important, and so any application on the device must not change the user's data without the user being aware of it. This test seeks to ensure that the only changes made to user data on the device are those which the user is aware of.

Testing Guidelines

The first step in this test is to populate the device with some data. You could do this manually, but it may be easier to keep a data set and bluetooth it onto the device (or get it onto the device in some other way) before beginning the testing.

We don't specify the exact data set with which the device must be populated, as we don't want to restrict the testing to only a certain number of contacts for instance.

Once you've run through the test cases, don't delete the data, as it's used again in a subsequent test.

There is an element of subjectivity in this test case, as some applications do change user data as part of their key function. In which case, the test house need to ensure that only the data specified by the application is changed. They key is that any changes to user data must be something the user has control over and must be aware of.

Advice for Developers

The key to passing this test is to make sure that you prompt the user before making any changes to their data. If the key function of your application is to alter the user data, then just ensure that you prompt the user before making any changes and give them the chance to cancel the change.

If your application isn't designed to change user data, then you shouldn't do it. For instance, a game shouldn't be making any changes to contacts, and a Twitter client shouldn't be deleting calendar entries.

Waivers

Only in very exception circumstances will waivers be accepted. You should contact Symbian before requesting a waiver for this test, as very few waivers will be accepted.

Test 8 - Un-install

It's vital that the user can remove from their device any applications which they not longer wish to have installed. This test ensures that it's possible for the user to remove the application completely.

Testing Guidelines

The first three test steps of this test are fairly self-explanatory and should not present any problems for testing.

The final test has some subjectivity included. Generally, applications must not leave more than 10k of data behind when they are uninstalled. The only exception to this allowed within the test would be for an application which downloads content at the user request. For instance, an application which downloads mp3 files from a music store would not necessarily be expected to delete all of the music which has been downloaded.

The key is that the user must be aware of what is being left behind. The test states user generated and downloaded content, and so anything above 10k which is left by the uninstall must be something which the user has either created or downloaded willingly and must be declared in the readme.txt. If the data isn't known to the user and isn't declared in the readme.txt file, then this test should be failed.

Advice for Developers

As with some of the other tests, the key to passing this is to keep the user informed. When the application is uninstalled, you could consider presenting the user with a prompt to explain what will, and won't be removed - giving them an option to remove all content if they want to.

You must also ensure that you provide information on what is left behind after uninstall. In your readme.txt file, you should provide a very brief explanation of all the data left behind when your application is uninstalled.

Sometimes, you may wish to protect the uninstall of your application so that the user cannot remove it. If you do this, you will need to not only provide justification of why you must restrict uninstall, but must explain to the test house how they can uninstall it. You could do this by use of a PIN code, etc.

Waivers

Generally, no waivers will be accepted.

Test 9 - Device Adaptation

Different devices can be produced from the same underlying Symbian platform, and it's important that any application which is designed for that platform is checked on a range of devices. This test is designed to ensure that the basic checks are done on the application on a range of devices.

Testing Guidelines

The first thing you need to do as the test house is to consult the device table to see which devices you should be testing on. The device table is available at Symbian Signed Device Table.

During submission the developer will have selected a primary device, and you don't need to run all of Test 9 on that primary device, as most of the testing has been covered by the other criteria. If the primary device supports multiple screen modes (for instance, portrait and landscape) then you should run just Step 4 of Test 9 on the primary device.

One you've done that, you should look up the device table which other lead devices you're required to test on.

The key point when testing is to remember that the aim is not to judge the quality of the application in terms of graphical display. As long as the key functions (exit, help, etc.) can be accessed and read, then the application should not fail this test on a minor graphical issue.

It is acceptable for an application to remain in one orientation when the UI rotates and still pass this test.

Advice for Developers

The first step to passing this test is to get your platform UIDs correct. This may seem an onerous task, but it's something you only need to do once for your application and so it's worth taking the time to make sure they are correct. You should include in the package file the UIDs of those platforms your application is designed to run on. If your application is designed only for one particular handset, then just specify the product UID of that handset.

The table of lead devices has been simplified and can be found at https://www.google.com/a/symbian.org/ServiceLogin?service=ah&passive=true&continue=https://appengine.google.com/_ah/conflogin%3Fcontinue%3Dhttp://tiny.symbian.org/a/symbian.org%253Fcontinue%253Dhttp%25253A//tiny.symbian.org/devicetable&ltmpl=&sig=42ea7cd4e12cc1e1acdda25ba834f5e9

Many manufacturers provide remote testing solutions to allow you to test your application on their device even if you don't have physical access to the handset.

If your application must remain in one orientation even when the device is rotated, then you are free to implement it this way and still pass this test.

Waivers

Testing VoIP Applications

What is a VoIP Application?

For the purposes of this testing, we define a VoIP application to be any application which allows the user to speak to another user from within the application using either 3G, GPRS or WiFi connectivity and which uses either the MultimediaDD or NetworkCotrol capability (or both)

Test 10

This test will ensure that emergency calls can still be made when the VoIP application is use. There are multiple scenarios which must be tested, because VoIP applications have potential to interfere with emergency calls in a number of ways. Unlike the emergency call testing in 4, which can be skipped if facilities are not available to the test house to test this, this testing is mandatory for VoIP applications and the test house must be equipped to run this test successfully.