

Application_signing_for_Dummies

- "I juz bout tis phon and now symbian wont let me uzit!! gief mani baak, lpl"

Hum, are you a:
end user?
student?
freeware developer?

Good, this guide is for YOU.

Contents

- 1 Using a mobile device -vs- publishing applications for a device
- 2 How to sign applications for free?
- 3 The steps
- 4 What if this does not work?

Using a mobile device -vs- publishing applications for a device

The main idea behind the Symbian application model is that an end user need not care about signing applications - or about developer certificates. These are tools used by an application developer, a person or a company that develops mobile applications. In this model the two roles (developer / end-user) have the following responsibilities:

Developer company:

- create
- test
- sign
- push to market

End-user:

- purchase / download
- install
- use

How to sign applications for free?

If you are a freeware developer - or an end user with some mad coding skilz - you may want to be able to develop your own stuff for your device (or any other device for that matter, no IMEI restriction here) for free. You don't want to buy a Publisher ID, nor would you pay for any external testing or signing.

Well, this is all possible.

If your code uses only user grantable capabilities, you can self-sign it. What this means is that you call one tool (on the command prompt) to make your project; you call another to create a SIS file; you call a third to generate certificate / key pair; you call a fourth command to sign with those cert/key.

Voila! You can now publish your application, and it will install on any device! (well depending on binary compatibility and device/platform feature support, not to mention manufacturer, ofc).

Remember ESeries devices do not allow self signed application to be installed by default

[link](#)

The steps

To make a *S60 3rd Ed (Symbian OS v9)* application you have to:

1. write code (make sure you use a UID from the unprotected range, 0xe0000000 - 0xefffffff is a good starting range)
2. test it on the EMULATOR (almost as good as a real device, can be used to run demos too)

-- when your code is good enough

3. make a SIS
4. make your own certificate and key
5. sign it (the SIS file)
6. run it on the device

7. Help your fellow developer by writing a constructive post on how you have triumphed and managed to sign and share your application :)

What if this does not work?

Well, there are limitations to this model, namely, the more system critical capabilities that require certified testing of the application. Such testing will cost various amounts of money, and such testing / verification systems can be misused if not properly monitored. Hence a software publisher needs to be authenticated by a trusted party (enter the Publisher ID).

For more information on how to proceed *if* self-signing did not work for you, check out [Symbian Signed](#).