

The purpose of this article

This article is the complete beginner's guide on how to use Express signed to sign your S60 application.

Even though it is tuned to serve Flash developers. This may be useful for Symbian developers as well. Please note, that symbiansigned.com provides [an Express signed guide](#) too, but the order of the steps is not the same and some of the instructions assume former Symbian knowledge. Some of the text here is copied from that document.

Contents

- 1 Overview
 - ◆ 1.1 Plan B: Web Runtime
- 2 Creating a Symbian stub application
- 3 Preparatory steps
 - ◆ 3.1 Step 1. Purchasing a Publisher ID
 - ◆ 3.2 Step 2. Creating a Symbian Signed account
 - ◆ 3.3 Step 3. Creating a Key/Certificate pair
 - ◇ 3.3.1 Option 1. Using the Tcp12p8 tool
 - ◇ 3.3.2 Option 2. Using OpenSSL
 - ◆ 3.4 Step 4. Obtaining a UID
 - ◆ 3.5 Step 5. Testing your Publisher ID
 - ◇ 3.5.1 Creating a Certificate Request file
 - ◇ 3.5.2 Requesting Developer Certificate
 - ◆ 3.6 Step 6. Purchasing a Content ID
- 4 The actual signing process
 - ◆ 4.1 Step 7. Signing your files with Publisher ID
 - ◇ 4.1.1 Option 1. Doing it manually
 - ◇ 4.1.2 Option 2. Using Carbide C++
 - ◇ 4.1.3 Option 3. Using Forum Nokia Flash packager
 - ◆ 4.2 Step 8. Signing your application in Express Signed

Overview

Signing Applications is a method by which you can certify software as being safe for end users to install. It can be used to indicate that the installation package has been tested, and is supplied from a trusted and authentic source. For commercial software it is advisable - and to be eligible for entry into the Ovi Store - all

Step_by_Step_instructions_to_Express_signing

Symbian applications must be signed.

Self signed applications prompt an installation warning of "untrusted software" to the end user.

Signed applications not only remove this warning, but open up the possibility for additional capabilities to be granted. For example - Flash Lite will normally ask the user for permission every time a network connection is required. With a certified signed application it is possible that this permission is granted once by the user and never asked again. Also being part of a Symbian application, you can add functionality not normally available for Flash apps, as well as increasing the stack memory size.

Plan B: Web Runtime

Please note, that SIS is not the only way of packaging a Flash app. You can also use a Web Runtime Widget on supported devices. There is a nice tutorial for that in this Wiki titled appropriately How to package Flash content in a Widget

Creating a Symbian stub application

This guide expects you to have a Symbian stub application created. [[Here's a nice tutorial for it](#)]

Preparatory steps

Step 1. Purchasing a Publisher ID

If you already have a Publisher ID issued by TC TrustCenter, go to [Step 2](#).

For the purpose of identifying your company as a trusted party, it is required that you have an official Publisher ID, issued by TC TrustCenter. The Publisher ID is valid for one year at a time.

To complete this step you may have to wait up to 2 weeks for your application to be approved

1. Have your credit card ready (cost is 200 US Dollars / year).
2. Go to <http://www.trustcenter.de/order/publisherid/dev>
3. Fill in all necessary data, follow the instructions closely.
4. Submit your application.
5. Wait for the request to be processed.

During processing, to be able to verify the existence of your company and to verify your identity, TC TrustCenter may ask for additional documents to be submitted or posted. When approved, you will receive further information in about a week. If you are renewing an application, it is a lot faster.

Note! ACS Publisher IDs supplied by VeriSign cannot be used for Express Signed submissions, although existing ACS Publisher IDs remain valid for Certified Signed.

Step 2. Creating a Symbian Signed account

If you already have a Symbian signed account, go to [Step 3](#).

First go to <https://www.symbiansigned.com/app/page/preregister> and give your email address. Then fill in all required data and submit the application. You will receive further instructions to the email address you gave.

Note! *Symbian Signed only accepts registration from privately registered domains or company domains; public email domains and ISP domains are not accepted.*

Step 3. Creating a Key/Certificate pair

If you have a previously created *.key and *.cer file pair you created from your Publisher ID (*.pfx), go to [Step 4](#).

Option 1. Using the Tcp12p8 tool

Note! *You need a Windows PC to execute this step*

1. Download the TC-ConvertP12 -tool from [here](#)
2. Create a new folder and unzip TC-ConvertP12.zip in that folder
3. Copy your Publisher ID file (*.pfx) into that folder
4. Open a command prompt window and change to that folder
5. Type the following command:

```
Tcp12p8.bat <yourfile.pfx> <password> <yourkeyfile.key> <yourcerfile.cer>
```

where <yourfile.pfx> is your Publisher ID file and <yourfile.key> and <yourfile.cer> are the key and certificate files to be generated.

Option 2. Using OpenSSL

In case you do not have access to a Windows PC you can still use OpenSSL

1. Install OpenSSL
 1. Instructions for Mac [here](#)
 2. Instructions for Linux [here](#)
2. Copy the Publisher ID file into the **bin** folder in your OpenSSL installation.
3. Open a console/terminal/command prompt at the **bin** folder
4. Type

```
openssl pkcs12 -in <yourfile.pfx> -nokeys -clcerts -out <yourcert.cer>
```

Step_by_Step_instructions_to_Express_signing

where `<yourfile.pfx>` is your Publisher ID file and `<yourfile.cer>` is the certificate file to be generated.

5. Type

```
openssl pkcs12 -in <yourfile.pfx> -nocerts -out <yourcert.key> -nodes
```

where `<yourfile.pfx>` is your Publisher ID file and `<yourfile.key>` is the key file to be generated.

6. You have now successfully created the key and certificate files. Make note of the location or copy the `*.cer` and `*.key` files to a preferred location

Step 4. Obtaining a UID

Every Symbian application needs a unique identifier. For testing purposes you can use a UID from the test range: `0xE0000000...0xEFFFFFFF`. For a commercial application you need to get one assigned especially for your application from Symbian Signed. If you have already obtained a protected range UID for your application proceed to [Step 7](#).

1. Login to [Symbian Signed](#)
2. Go to [UID request form](#)
3. After you click on **Submit** your UID is shown. Copy the UID and use in your project or save for using in an online service.

Step 5. Testing your Publisher ID

If you have not done so before, you need to test at least once, that your publisher ID is valid. This is done by creating an Open Signed offline request. As a result you will get a *Developer Certificate* or DevCert in the Symbian developer jargon. This Certificate is a key and value pair you can use to sign your own application and bind it to a device's [IMEI](#) code ([more info](#)). After creating the DevCert the option for purchasing *Express Signed Content IDs* will become available. If you have made an Open Signed *offline* request earlier, you can go to [Step 6](#).

Creating a Certificate Request file

Note! *You need a Windows PC to use the DevCertRequest tool*

1. Get the DevCertRequest tool from [here](#)
2. Install the tool by double clicking on the downloaded file
3. Click on **Start menu / Symbian OS Tools / Developer Certificate Request / DevCertRequest**
 1. select the name and location of the `*.csr` file.
 2. select the key and certificate pair generated in [Step 3](#).
 3. check that the information displayed is correct
 4. fill in the [IMEI](#) codes of the devices used for testing
 5. select the capabilities needed for your application. The list of required capabilities can be found from your `*.mmp` file (usually in your projects *group* folder)

Step_by_Step_instructions_to_Express_signing

6. if the final screen looks OK, Click **Finish** to generate the *.csr file.
4. You can now close the DevCertRequest tool and proceed to [requesting a Developer Certificate](#)

Requesting Developer Certificate

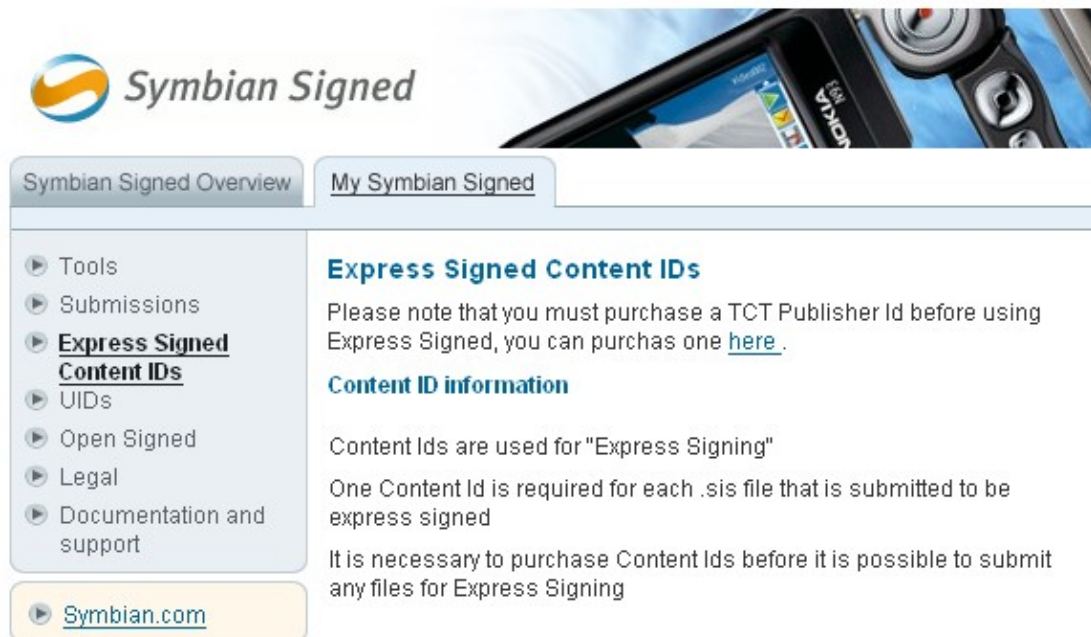
1. Go to [the DevCert request form](#)
2. Fill in the required information and press **Send**

The DevCert (*.cer) is now created, you can download it from [My DevCerts](#) To use the DevCert for signing, use the instructions in [Step 7.](#), but instead of the *.csr file generated in Step 3, use the DevCert *.cer file.

Step 6. Purchasing a Content ID

To be able to sign your application with Express signed, you need to purchase an Express signed Content ID, also referred to as TCT Content ID. If you already have at least one you have purchased earlier, proceed to [the signing process.](#)

1. Log in to [Symbian Signed](#)
2. Make sure, you have the option to purchase content IDs available (*see image below*). If not, go back to [Step 5](#)



Screenshot from Symbian Signed

3. Use a credit card or PayPal to pay for a Content Id

The actual signing process

Now that you have successfully completed all the preparatory steps, you are now ready to Express Sign your application.

Step 7. Signing your files with Publisher ID

Now you have the key and certificate files, identifying you or your company. These files are used to sign your application's SIS-package, so it can be Express signed at Symbian Signed.

Note! *Your application needs to have a UID from the protected range see [Step 4](#).*

Option 1. Doing it manually

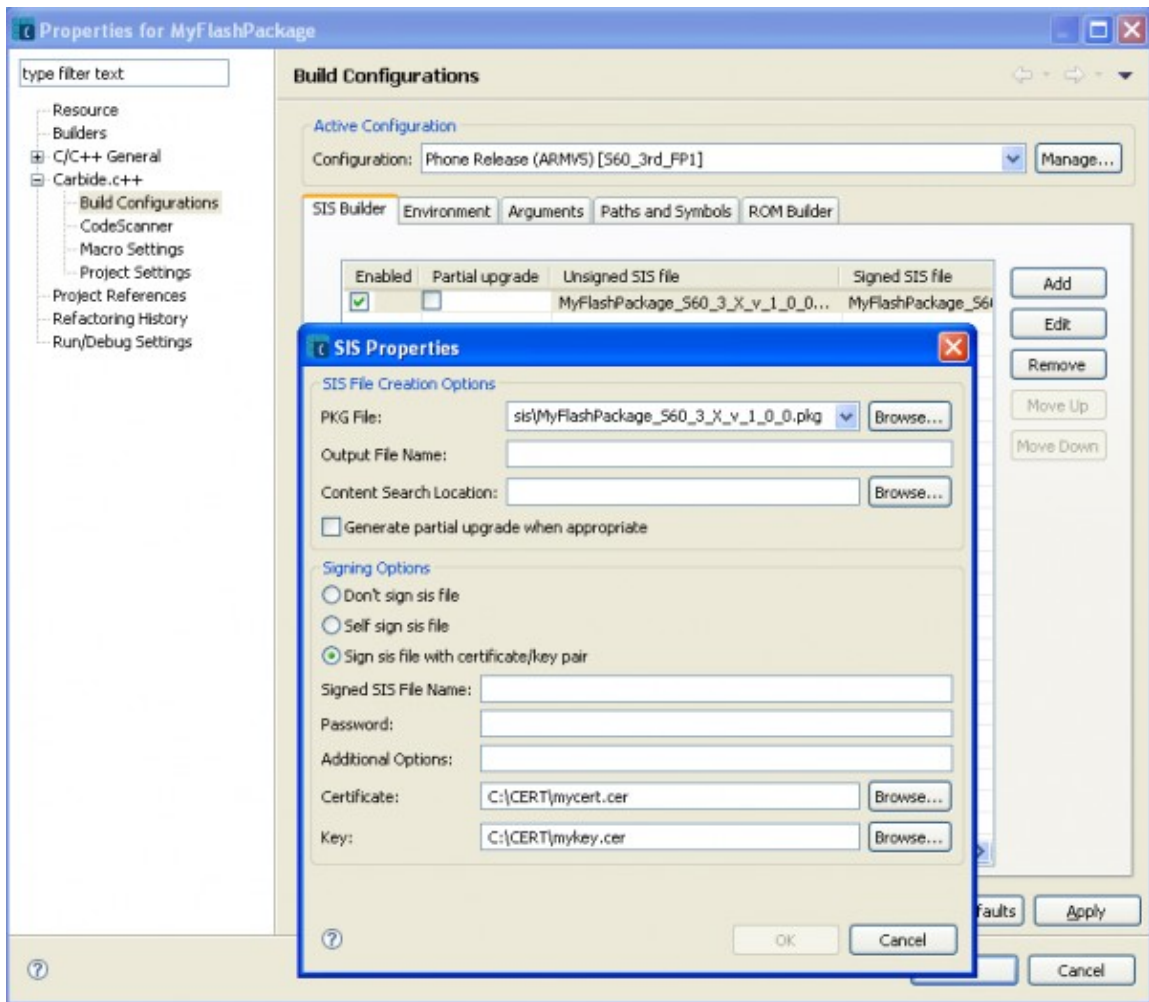
An example project and instructions can be found for example in the [Flash Lite Developers Library](#)

Option 2. Using Carbide C++

To use Carbide.c++ you need to have it installed on a Windows PC. Instructions for installing Carbide.c++ and the S60 SDK can be found [here](#).

If you have created your application with Carbide C++, you need to edit the build configuration so it uses your key/certificate pair:

1. In Carbide, right-click on the root folder in the project explorer window or select the root folder and press *Alt+Enter*.
2. In the Properties-dialog box that pops up, select **Carbide.c++** and then select **Build Configurations**.
3. Make sure **Phone release** is selected as active configuration.
4. In the **SIS Builder** tab either select an existing set or click **Add** to create a new one. If you leave the **...name** fields empty your project name is used by default.*(see image below)*



5. After pressing **OK** twice, build your project (*Ctrl+B*) and the sis files will appear in the *sis* folder of the project. The one you need is named **.sisx* or whatever you chose, if you filled in the *Signed SIS file name* field.

Option 3. Using Forum Nokia Flash packager

By far the If you are packaging a Flash Lite application using the online service

1. Open the tool [www.forum.nokia.com/flashpackager here]
2. Follow the instructions and fill in the required data
3. On the advanced tab fill in your UID and upload your **.key* and **.cer* files
4. The SIS file will be automatically downloaded to your computer

Step 8. Signing your application in Express Signed

Congratulations! You have gone through all the previous steps and are now ready to submit your application to be Express Signed. Most of the steps need not be repeated again until your Publisher ID expires.

1. Log in to [Symbian Signed](#)
2. Go to the [Express Signed page](#)

Option 2. Using Carbide C++

Step_by_Step_instructions_to_Express_signing

3. Follow the step by step instructions there to Express Sign your application