

# Archived:Express Signed for Flash Lite apps



**Archived:** This article is [archived](#) because it is not considered relevant for third-party developers creating commercial solutions today. If you think this article is still relevant, let us know by adding the template `{{ReviewForRemovalFromArchive|user=~~~~|write your reason here}}`.

We do not recommend Flash Lite development on current Nokia devices, and all Flash Lite articles on this wiki have been archived. Flash Lite has been removed from all Nokia Asha and recent Series 40 devices and has limited support on Symbian. Specific information for Nokia Belle is available in [Flash Lite on Nokia Browser for Symbian](#). Specific information for OLD Series 40 and Symbian devices is available in the [Flash Lite Developers Library](#).



**Note:** Starting 1st July 2013, there would be NO need to purchase the Publisher Id to verify Symbian Signed account which further is used in Express Signing of apps, thereby eliminating Step 1, 3, 4,6,8 completely.For more info see: [User\\_guide:\\_Symbian\\_Signed](#)

This article is the complete beginner's guide for Flash Lite developers on how to use [Express signed](#) to sign your Symbian app. This article may be useful for developers of other runtimes on Symbian. Note, that parts of this document were copied from a document on the defunct [developer.symbian.com](#) wiki "ExpressSignedGuide-HowtoSignApplicationsUsingExpressSigned".

## Overview



24 Jan  
2010

There are a number of alternatives for signing/installing Flash Lite apps on Symbian. Apps may be:

- Symbian Signed, using Express or Certified Signed
- [Self Signed](#) - but will prompt an installation warning of "untrusted software" on installation
- Packaged as a [Symbian Web Runtime Widget \(.wgz\)](#) on [supported devices](#)

Apps signed using a digital certificate from Symbian Signed will install without warnings - indicating that they have been tested, and originate from an authenticated source. Symbian Signed apps can be granted additional capabilities over Self Signed apps - among other things, you can add functionality not normally available for Flash apps, increase the [stack size](#) available to your app, and grant network access permanently on installation rather than having to request access every time the network is used. Using Symbian Signed is recommended for commercial software, and is required if the app is to be sold through Nokia Store.

This article explains how to [Express sign](#) an app. This is a low-cost signing method where apps are self-tested to conform to the testing criteria (they may be audited).

## Prerequisites

This guide expects you to have created a Symbian stub application, as described in [Create Flash Applications with Carbide.c++](#).

## 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 <https://www.trustcenter.de/RetailStore/cid/CustomerData!input.action>
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/signedui/user/signup> 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>
```

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. Verifying the Symbian Signed Account

Once the publisher id is obtained successfully, the symbian signed account needs to be verified first before any application signing process can be done. Following steps should be followed for verifying:

1. Login to [Symbian Signed](#)
2. On right-hand side of the page (just below the user name), **Verify Account** label appears(if the account is unverified), click it to go the verification page
3. On verification page, upload a sample sis(x) file, signed from the same publisher id which is obtained in [Step 3](#). (a sample sis file for signing is also available on the same page)
4. Press the submit/verify button.

If everything is fine, the page gets loaded with the message that your symbian signed account has now been verified other wise if there exist issues(like the publisher id is expired etc.) then you are notified about the same. Next time you login to your Symbian-Signed account, you can notice the label just below the user name, **verified account** & you can now proceed with the signing & other processes.

### Step 5. 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. Click on the tab **Manage UIDs**
3. Enter the Number of UIDs required under the **Request UIDs** heading, along with other details
4. Press the **Request UIDs** button below the page & if the request is successful, the allocated UIDs will appear on top of the page

### Step 6. 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 7](#).

#### 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 you \*.mmp file (usually in your projects *group* folder)
  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](#)



**Note:** The tool/process stated in Step 6 is no longer used. Testing/verifying the publisher id is done as soon as symbian signed account is verified using the newly granted publisher id, as described in [Step 4](#).

### Requesting Developer Certificate

1. Login on to your [Symbian Signed](#) account
  2. Click on the tab **Development Certificate**
  3. Down the page, under the heading **Add IMEIs**, add the IMEIs to be included in the certificate(a text file containing IMEIs can also be uploaded)
  4. After adding IMEIs, press the **Download Certificate** button on right hand side of the page which successfully downloads the certificate on the pc
- To use the DevCert for signing, use the instructions in [Step 8.](#), but instead of the \*.cer file generated in Step 3, use the DevCert \*.cer file.

### Step 7. 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. Click on the tab "**My content ids**". This page displays the details(history) of purchased/consumed content ids, & along with that an option to purchase new content ids as well(a button on right hand side named "**Purchase Content Ids**")
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 8. 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 5](#).*

#### Option 1. Doing it manually

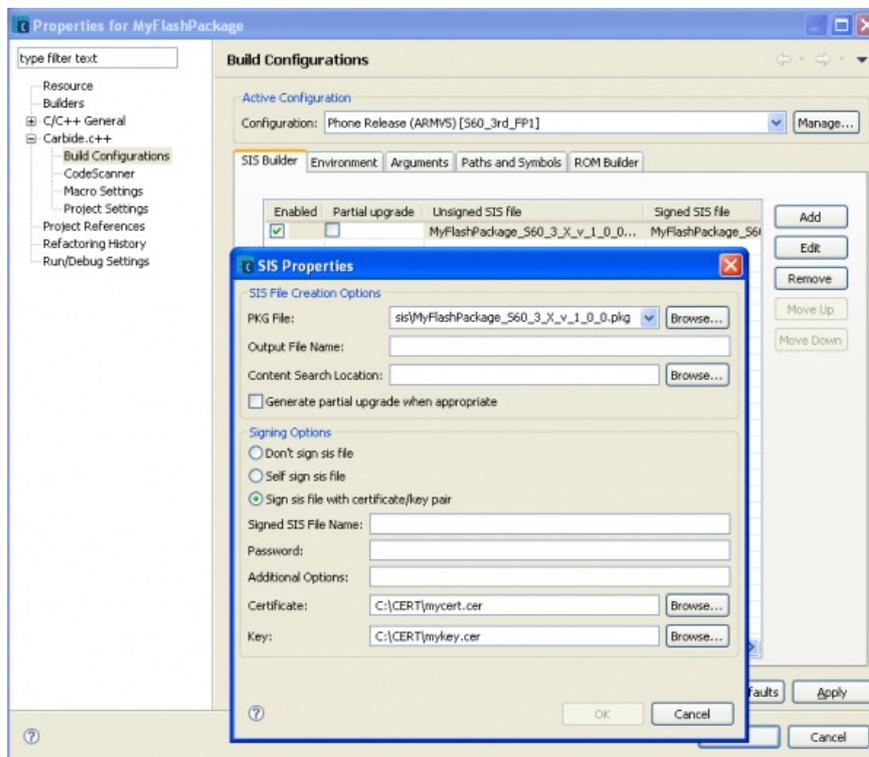
An example project and instructions can be found for example in the [Flash Lite Developers Library](#)

#### 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.



**Note:** The file signed with a publisher ID is **not** installable.

To be able to install it or sell it in the OVI store, you need to get it signed by Symbian signed. See the instructions below on how to Express sign the SIS file

## Step 9. 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. Click on the tab **Submissions** & on right hand side of the page click on the button **Submit App for Signing**
3. fill in all the details as asked in the form & click **Submit App for analysis** for uploading the file for Express Signing.

