Create Flash Applications with Carbide.c++

Warning: We do not recommend Flash Lite development on current Nokia devices, and all Flash Lite articles on this wiki have been archived.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.

This article explains how to add a plugin for Carbide.c++ to quickly compile your Flash project into a Symbian installable application for S60 3rd edition and 5th edition Nokia Symbian devices. If you are interested in packaging into a WRT widget, see the article How to package Flash content in a Widget. If you want to package Flash for Series 40 phones, go and see this document about the NFL format.

03 Jan 2010

Prerequisites

You need to have access to a computer with Windows operating system. If you already have a working Symbian SDK and Carbide.c++ IDE installed, you can skip to *Step 1*.

Download and install Symbian SDK

The Symbian SDKs are free. You are only required to register as Nokia Developer member, which is also free. You can find the following SDKs here

- Symbian Belle (release notes)
- Symbian^3 1.0 (release notes)
- Nokia N97 (Nokia release notes)
- 5th Edition (release notes)
- 3rd Edition (release notes)

Note: *Read the release notes* corresponding to your choice of the SDK. From the release notes you will learn:
- If you need to have installed some 3rd party software like ActivePerl and Java Runtime Environment (JRE).
- Known issues (e.g. installation problems and known bugs or limitations)

If you have Windows Vista, also see Using Nokia SDKs and Tools on Windows Vista.

Once you are sure you have all the necessary pieces of the SDK puzzle, install the SDK on your PC.

Install Carbide.c++

Carbide.c++ is a free IDE from Nokia, you can download and install it from the link below

- Download Carbide.c++

Creating and testing a Flash Stub app

Follow the steps below to create a Flash Stub application. Please note, that to be able to publish sis file in the Nokia Store you need to have a publisher account for the Nokia Store and to sign the sis file. More information about signing in the chapter *Ready for Signing*

Step 1. Add the Wizard template for Flash Stub

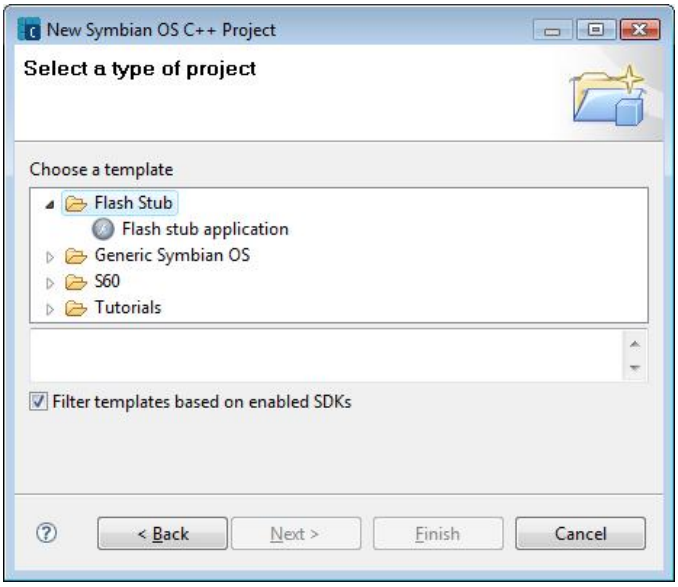
(only needed once)

- Make sure Carbide.c++ is not running and closed.
- Unzip and drop this .jar file to <installation path>/plugins *(for example C:\Apps\Nokia\Carbide.c++ v2.0\plugins)*

Step 2. Open Carbide and create a new stub

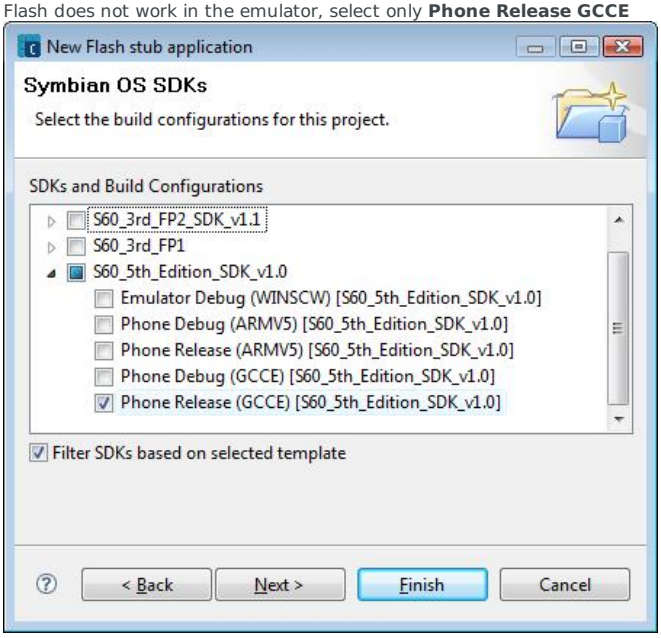
- Start a new project (ctrl+N)
- In *Select Wizard* dialog, select '*Symbian OS/Symbian OS C++ project*
  (Click Next)
- In *Select type of project* dialog, select **Flash stub application**

(C) Copyright Nokia 2013. All rights reserved.



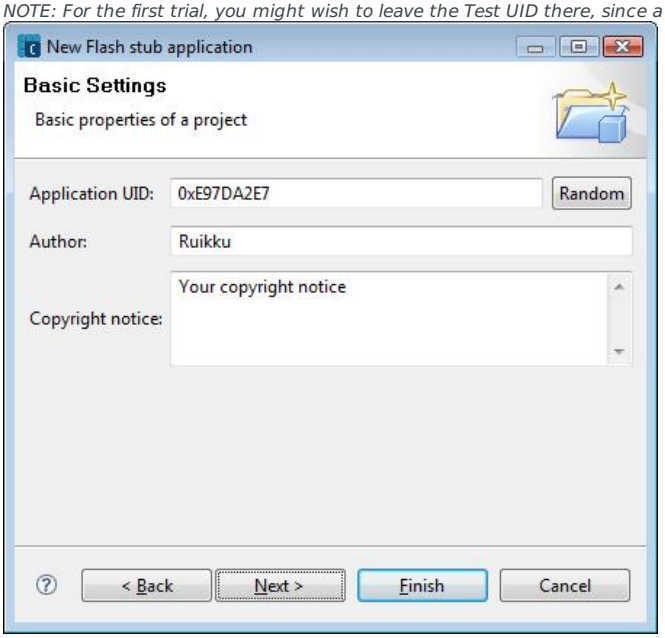
(Click Next)

- Give a name to your project
  (Click Next)
- Select the *SDK*
  Flash does not work in the emulator, select only **Phone Release GCCE**



(Click Next)

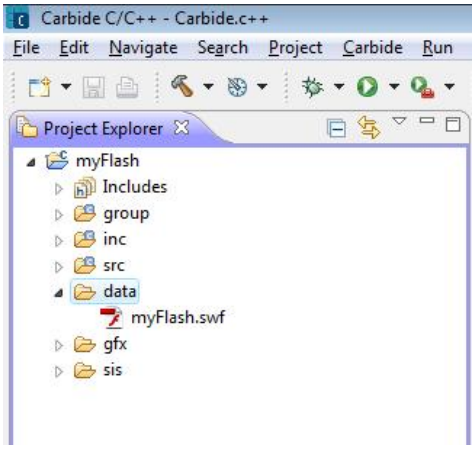
- Give your project's UID, you author name and a Copyright, if desired
  NOTE: For the first trial, you might wish to leave the Test UID there, since a "real" UID will not work for Self Signing



(Click Finish)

(C) Copyright Nokia 2013. All rights reserved.

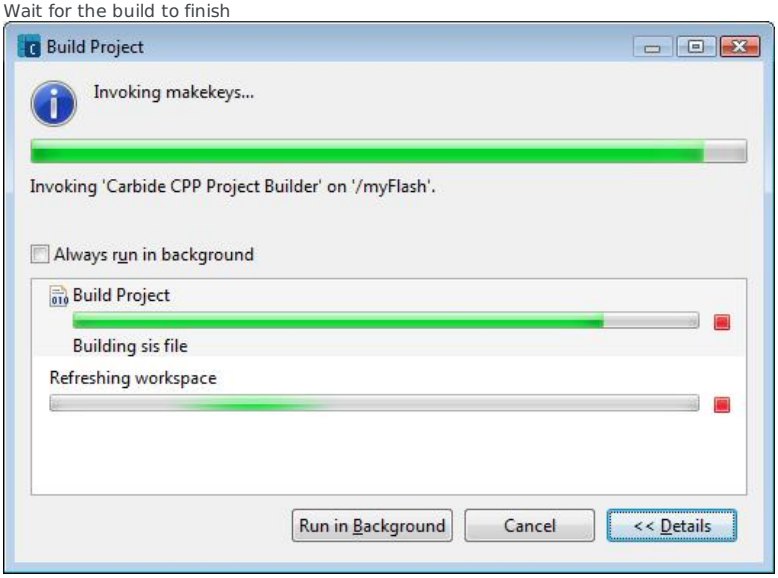
Step 3. Drop your own content in



- Select the **data** folder under the project.
  If you cannot see the Data folder, select *Window\Open perspective\Other...\C/C++*
- Right Click on the folder and select **Show in Explorer**
- Drop your SWF into this folder, rename it and replace the one in the folder (it's a placeholder swf)
  The name needs to be the same as the placeholder swf.
  If you have more than one file, add the extra files your PKG file in the '*group* folder

Step 5. Test the content

- Select your project in the left pane
- Press **Ctrl+B** to build the project
  Wait for the build to finish



- Right click on the **sis** folder and select **Show in Explorer**
- Send the ***.sisx** file to your device *(sis=unsigned, sisx=self signed)*
  ***.sis** is an unsigned file, which cannot be installed anywhere. the 'x' in the end is not mandatory for signed files, it's for convenience only.
  Emulators don't support Flash, so you need to have a supported device
  If you do not have a supported device check the Remote Device Access service

> **Note:** *For S60 3rd Edition:* The sis will compatible with both S60 5th and S60 3rd edition devices, but is if you are targeting a S60 3rd edition device, you cannot have the *location* capability in a self signed application. So in this case you need to edit the ***.mmp** file in the **group** folder and remove the **location** keyword at the end of the file.

Creating the actual application

Getting a UID

Nokia publishers can request to have certain apps signed for free by Nokia, reducing the turnaround time. Nokia Publisher Support sends the Publisher a list of UID's. You can find out how to request the UIDs from Nokia here. If you do not wish to make a commercial application, you can use the automatically generated UID, it's from a free range and can be used for testing and Self signed apps. But a commercial application needs a "proper" UID assigned to your company name.

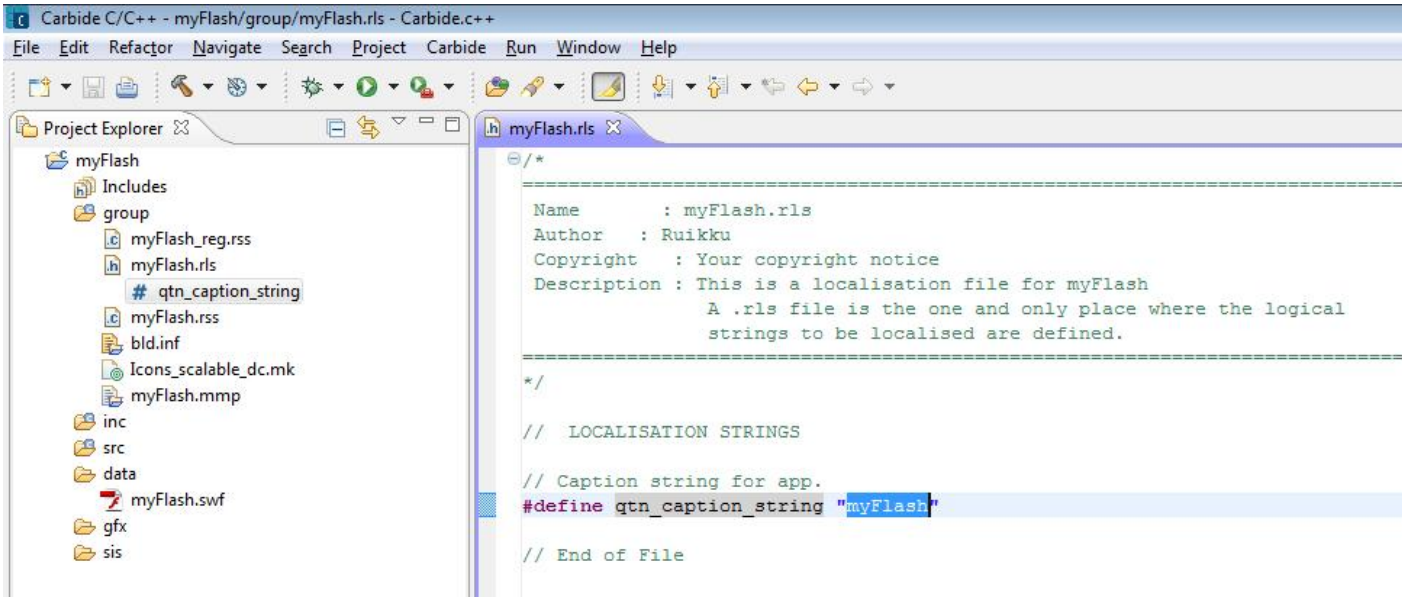
Make a new Flash Stub project

(O.K. this is not absolutely necessary, you can of course find and replace each and every occurrence of the UID all over the project, but since the Wizard is so nice and easy to use, why not take advantage of it ;o)

(C) Copyright Nokia 2013. All rights reserved.

- Repeat steps 2 and 3

Step 6. Getting ready for delivery



- Edit the content of *qtn_caption_string* in **group\<projectname>.rls**
  This is used as your application title in the device
  Make it as short as possible, there is not much room on the Applications menu
- Drop a new SVG icon to "gfx" folder, remove the placeholder svg and give your icon the same name it had.
  More information about creating SVG icons in: How to create a Symbian application icon (SVG)

Ready for signing?

Now that you have your file in a commercial format you cannot self sign it anymore with a self signing key. In the previous steps Carbide was creating a self signing key for you, but since you are now using a proper UID you need to have the file properly signed. You can submit the app (the unsigned SIS file with the UID provided) via the Nokia Publish tool. Nokia Store quality assurance (QA) tests the app based on Content Guidelines, specific operator guidelines, and the Symbian Signed Test Criteria. If it passes, the app will be express-signed by Nokia and published in Nokia Store.

Feedback much appreciated!

Please use the comment tab for feedback to the author.

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.