



Translations (may not be up-to-date) :

[??????SymbianOS????](#)

[Como iniciar o desenvolvimento em SymbianOS - Português - BR.](#)

[? ???? ?????? ?????????????????? ??? Symbian ?? ?](#)

0.5: You already know some C++ basics, right?

This basically assumes that you know the difference between an IDE, a compiler and a linker and you do not assume that there is only one IDE and one tool chain that can ever be used. You should also know that C++ is not [only] STL and that multi-threading is not the solution for all problems. ;)

If you are not comfortable with C++, or are looking for an easier to use tool, there are other [programming languages](#) you can use.

1: Do you have a smartphone to test your application on?

If not ? no problem, skip to Section 3 below and select one of the SDKs, e.g the latest. Please note that usually the complexity of the SDK increases with each new release and that the pool of information that can help you solve the problems will be more limited for the new SDKs.

Installing a SDK will allow you to test your created applications in a simulated S60 environment.

2: Read device's specifications on Forum Nokia's [Device Specifications](#) page.

- ◇ If the document says "Developer Platform: S60 3rd Edition" then the SDK for that device is S60 3rd SDK MR (an improved version of the S60 3rd SDK).
- ◇ If the document says "Developer Platform: S60 3rd Edition, Feature Pack 1" then the SDK for that device is S60 3rd SDK FP1.
- ◇ If you have two devices, one based on "S60 3rd Edition" while the other one on "S60 3rd Edition, Feature Pack 1" the recommended SDK is S60 3rd SDK MR, applications built with it would work on both devices ([backward compatibility](#)).
- ◇ If you have a device based on "S60 2nd Edition, Feature Pack X" and one based on "S60 3rd Edition, Feature Pack X" you cannot use one SDK for both devices due to binary and source compatibility breaks.

More details about the S60 Platform, the existing versions and the differences between them is to be found [here](#)

See also [Which S60 SDK should I use?](#)

3: Open the SDK download page

...but DO NOT download the SDK yet!

Locate the "Release notes" section, download and **read the release notes** corresponding to your choice of the SDK. Please note that for one SDK release there might be different versions of it, providing support for various toolsets. Read them all before making a decision.

From this document you will learn:

- if you need to have installed some 3rd party software like [ActivePerl](#) and Java Runtime Environment ([JRE](#)).
- which are the supported compilers (some freeware others under license)
- which are the supported IDEs (some freeware others under license)
- known issues (e.g. installation problems and known bugs or limitations)

Note: At the time of writing this article the tools and SDKs provided by Nokia are not intended to be used on Microsoft's Windows Vista (tm) operating system. Unofficial support from the developer community can be found in this wiki in pages like: [Moving to Windows Vista](#).

Once you know which configuration suits you best ...

4: Download the SDK, tools and IDE of your choice.

Install the packages (pre-requisites first, then the IDE and finally the SDK(s)). Install the tools in the suggested default location unless you are confident that you can handle some configuration tweaks.

Make sure that each individual tool works(e.g. by calling it from the [command line](#) with the "-version" option) and that you have the minimum required version. (Not all these tools will be in your system, make a note of what you find and re-evaluate the situation after completing step 5 below.)

```
C:\>devices
S60_3rd_FP2_SDK:com.nokia.s60
S60_3rd_MR:com.nokia.s60
S60_3rd_FP1:com.nokia.s60

C:\>devices -setdefault @S60_3rd_FP2_SDK:com.nokia.s60

C:\>devices
S60_3rd_FP2_SDK:com.nokia.s60 - default
S60_3rd_MR:com.nokia.s60
S60_3rd_FP1:com.nokia.s60
```

Note: The commands above verifies that one of the installed SDKs is set as default. Since this was not the case a second command is issued setting one of the SDKs as default.

```
C:\>perl -version

This is perl, v5.6.1 built for MSWin32-x86-multi-thread
(with 1 registered patch, see perl -V for more detail)

Copyright 1987-2001, Larry Wall
```

How_do_I_start_programming_for_Symbian_OS?

Binary build 638 provided by ActiveState Corp. <http://www.ActiveState.com>
ActiveState is a division of Sophos.
Built Apr 13 2004 19:24:21

... snip ...

Note: Use the ActivePerl version recommended by the SDK release notes or the one recommended by Carbide.c++. The latest ActivePerl releases are not compatible with the SDKs. If the release notes say to use "version X.Y.Z or later" stick to version X.Y.Z, it's the only one the SDK team really tests ;)

```
C:\>java -version
java version "1.5.0_05"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.5.0_05-b05)
Java HotSpot(TM) Client VM (build 1.5.0_05-b05, mixed mode, sharing)
```

Note: The SDKs may not work by default with the latest JRE releases (e.g. JRE 6.0, aka JRE 1.6). Whether you have problems with it or just as a preventive measure you may want to have a look at the following discussion board thread: [Cannot start ECMT Manager](#)

```
C:\>mwccsym2.exe -version
```

```
Nokia Codewarrior C/C++ Compiler for Windows/x86.
Copyright (c) 2005, Nokia Corporation
All rights reserved.
Version 3.2.3 build 446 (Build 446)
Runtime Built: Aug 15 2005 08:07:54
```

Note: The Nokia Compiler is not included in the S60 SDK but it ships with Carbide.c++. Also, even if the compiler is installed it may not be enabled by default for command line use. Please consult the Carbide documentation to learn about how to enable command line builds for the WINSCW platform.

```
C:\>arm-none-symbianelf-gcc.exe -v
Reading specs from ... snip ...
Configured with: ... snip ...
Thread model: single
gcc version 3.4.3 (release) (CodeSourcery ARM Q1C 2005)
```

Note: This free compiler is shipping with the SDK and it is an alternative to using RVCT (see below). If not found in your system it must be because you have not accepted the prompt to install it as part of the SDK install process. If needed you can find the install kit at `<SDK_install_dir>\epoc32\tools\distrib\arm-none-symbianelf-2005-q1c.exe`. You can only compile applications for the GCCE platform if you have this compiler.

```
C:\>armcc
ARM/Thumb C/C++ Compiler, RVCT2.2 [Build 503]

Usage:          armcc [options] file1 file2 ... fileN
Main options:  ... snip ...
```

Note: The RVCT compiler is NOT provided with the SDK. This command line test would only work if you have installed the compiler on your PC and if you have a valid license for it. If you do not have it you cannot compile your project using the ARMV5 platform. Also, you should be aware that although SDK's device libraries are delivered in an ARMV5 specific directory `%EPOCROOT%\epoc32\release\armv5\libs` they are actually used by the toolchain when

How do I start programming for Symbian OS?

compiling for the GCCE platform too.

```
C:\>epoc
```

```
C:\>
```

This last command will start the emulator. Please be patient, allow this complex tool the time to load itself and get started. If you see the application crashing or if it suddenly disappears from the screen you may have a configuration problem and this has to be addressed before moving forward. One known and rather frequent issue is that emulator audio support implementation cannot work with Realtek drivers so those would have to be disabled. Other similar issues may be affecting your configuration, especially if you are running the tools on Vista.

Note!

The emulator creates a log file which should be inspected when looking for the root cause of a crash. You can find it at %temp%\epocwind.out. You can quickly access this file using a command like:

```
c:>notepad %temp%\epocwind.out
```

5: Read the SDK documentation until you're confident that you understand:

- ◇ what Symbian OS is and how is it different from the other OS you have been programming for
- ◇ what programming for a mobile device means (constraints and opportunities)
- ◇ the structure of the OS, the main paradigms
- ◇ coding conventions
- ◇ system errors / panics / leaves and how to handle or log them
- ◇ differences between the device itself and SDK's emulator
- ◇ build system and tools, most importantly understand the "devices" tool

6: What's the hurry? ? Go back to Section 5!

7: Start the emulator from Windows' start menu.

Play with it, learn what applications are available, how to navigate in it and what "hidden" options are there in the menu. Almost everything is documented in SDK's help.

8: If you're here then you are ready to build your first application:

8a) Command line build (IDE independent)

Open a Windows command interpreter window and make the current working directory `%EPOCROOT%S60Ex` (Series60Ex on some SDKs). Do not rush to take an example from the `%EPOCROOT%Examples` folder, they are more advanced material.

If you do not know what `EPOCROOT` is go back to §5.

Go further in the directory structure by choosing one of the examples available there (e.g. one of the HelloWorld* versions). Once decided the example go deeper into its group folder.

At the command prompt type the following command sequence:

```
?\group> bldmake bldfiles
?\group> abld makefile all
?\group> abld resource
?\group> abld target WINSCW udeb
?\group> epoc
```

Needless to say that should you see any error message during the execution of any of the commands you must stop, evaluate the message, read the documentation and if needed take actions towards fixing those errors. Error messages can only be ignored once you know that they are not affecting your current build. If all goes well the last command will start the emulator and you will be able to test your first application ? well your first application build anyway.

The binaries generated with the commands above are only suitable for being run in the emulator (notice the **WINSCW** parameter). In order to run your application on the device you would have to build it again using a compatible configuration which depending on the SDK release and available compiler could be: **ARMI**, **THUMB**, **GCCE**, or **ARMV5**). The build command to execute could be:

```
?\group> abld build GCCE ure1
```

(**abld build** executes the entire **abld makefile**, **abld resource** and **abld target** in one go!)

Go back to §5 to find out which is the supported target for your SDK & tools configuration. Also check in your IDE's help how you can archive emulator and device builds from within the integrated development environment.

The next step would be to build an installation package (.sis or .sixs file) that can be delivered to the device.

```
?\group> makesis ..\sis\helloworld.pkg
```

You may need to sign it before transferring it on device (mandatory for S60 3rd Edition or later).

```
?\group> signsis helloworld.sis signed_helloworld.sis your_certificate.cer key_file.key
```

Note: If you have installed Carbide, normal command line compilation does not work out of the box. You need to first run `env_update.exe` (in `x86Build/env_switch` folder in Carbide 2.0).

For more details see: [signsis.exe](#), [createsis.exe](#), [makekeys.sis](#), [Symbian Signed](#), [Developer](#)

How do I start programming for Symbian OS?

Certificate, self signed

8b) IDE build: Using Carbide.c++

See the Getting Started with Carbide.c++ Express Screencast, Carbide Training Videos and SDK and Carbide.c++ installation guide.

Note!

If you're going to start using Carbide.c++ to generate your first project for one of its templates here's one piece of advice: do not rush into an Open C (or console) type of project, although its Helloworld name might be tempting. Instead chose one of the GUI project templates compatible with your chosen SDK.

A console project does not make a lot of sense on a mobile phone (in terms of user interaction) and requires that you already have a good understanding of how an application integrates with the OS.

Now you are ready to install the application on device and have fun testing it. Transfer the [signed] sis file to the device (using Bluetooth or data cable) and open it to start the installation.

9: Want to start coding now ? Hold your horses! ? :)

First you need to make sure that you fully understand that hello world (or whatever choice you had) example. Open the project directory in a file browser and analyze its content and structure. Don't go further until you fully understand the role of each file in that directory.

10: From now on you're on your own ...

... but the Symbian developer community will be happy to help you.

See the Forum Nokia Developer Discussion boards Carbide.c++ and CodeWarrior Tools

If you run into problems read the documentation. If you need more documentation visit www.forum.nokia.com and/or the Symbian Developer Network

Before asking for help read ? read ? read. Don't miss these posts and definitely don't miss our technical library. You can also find a range of books and free booklets by Symbian Press here. If you are an absolute beginner, or just new to C++ on Symbian OS, the new version of Steve Babin's Developing Software for Symbian OS is a great book to help you get started.

Use the forums to discuss your problems but make sure you are doing it "the smart way".

How do I start programming for Symbian OS?

Have you considered becoming an Accredited Symbian Developer? Or maybe even an Accredited S60 Developer?

11: Reference materials

- Symbian OS Basics - Workbook v3.1
 - The eLearning Curriculum
 - Symbian Documentation
-