



# Symbian C++ - Быстрый старт

Оригинальная статья: [Symbian C++ Quick Start](#)

Перевод: [Андрей Ярощук aka Athloni](#)

E-mail: [codcore@gmail.com](mailto:codcore@gmail.com)

Специально для [Dimonvideo.ru](http://Dimonvideo.ru)

Все ссылки в документе рабочие, и подчеркнуты синим цветом. Возможны грамматические ошибки!(нет времени исправлять)Жду отзывов и предложений!

Это ознакомительное пособие предназначено для тех, кто желает создавать собственные пользовательские приложения для Symbian на C++. Это могут быть студенты, профессиональные разработчики приложений и игр, и т.д.

В статье описаны действия, необходимые для получения инструментов разработчика(как например **IDE** – среда разработки) и их настройки. Потом объясняется как создать «скелет» приложения, используя среду разработки **Carbide C++** и инструмент **UIDesigner**, и как запустить приложение на эмуляторе платформы Symbian, и на самом устройстве; весь процесс займет буквально несколько минут!! (☺)

Прикрепленный код: [HelloSymbianSimpleCode.zip](#) (ссылка)

## Установка пакета программ, необходимых для разработки

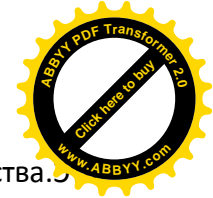
Для начала, было бы неплохо проверить, соответствует ли ваш компьютер [системным требованиям](#). Если да, что же – приступаем к установке(по порядку):

1. [Perl](#) (некоторые SDK могут выдавать предупреждение, если у вас установлена версия Perl, отличная от 5.6.1(638) которая больше не доступна для загрузки с сайта Active State(придется поискать ☹). Но все же, если с новой версией у вас все замечательно компилируется, то на эти сообщения можно не обращать внимания.
2. [Application Developer Toolkit\(ADT\)](#) - **Carbide C++ + Carbide.ui Theme Edition** (если вы не собираетесь создавать темы, или траф не позволяет – качаем [обычный Карбид](#)).
3. Качаем SDK для своей платформы, в зависимости от версии Symbian. Для S60 v3 (9.1) это например, [mym](#), там же можно найти и для других версий.

После этого мы можем создать наше первое приложение, используя для этого Мастер проектов Carbide.C++, как показано далее.

## Запуск Carbide C++

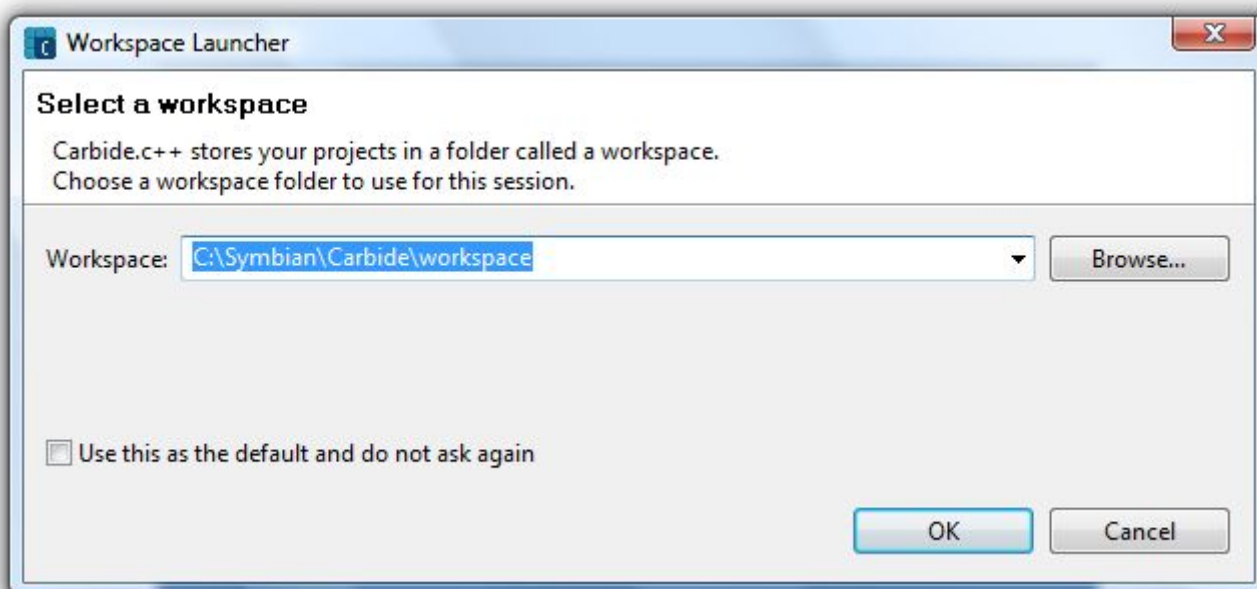
Carbide.C++ устанавливается как часть **ADT** (*Application Development Toolkit - набор инструментов для разработки*), на втором шаге процесса установки(см. выше). Это единственно официально поддерживаемая среда для разработки программ на Symbian C++. Запустить её можно с кнопки «Пуск» -> «Все программы» -> «Symbian Foundation ADT» -> «Carbide.c++».



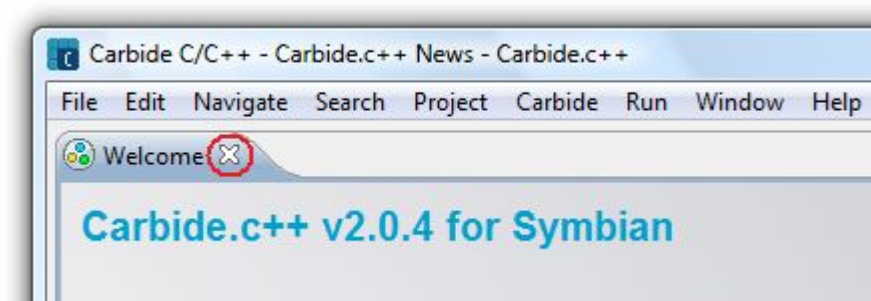
При запуске программы, вас попросят выбрать папку для рабочего пространства. Эта папка будет содержать все ваши проекты, а также их настройки (форматирование кода, например). Также можно создавать много рабочих пространств для разделения различных по своему типу проектов. Если вы запускаете Carbide.c++ первый раз, эта папка будет конечно же, пустой.

**Важно:** Ваши проекты для Symbian должны находится на том же диске, где установлена SDK. Также убедитесь, что название рабочего пространства не содержит пробелов и других знаков кроме буквенно-цифровых символов. Это и есть причиной того, что инструменты компиляции для Symbian (toolchain) используют только командную строку, которая не допускает использования специальных названий путей.

Если вы установили SDK на диск C:\, тогда корректный путь для рабочего пространства (*workspace*), может быть к примеру C:\Symbian\development.



Когда Carbide.c++ запущен, закройте панель приглашения, чтобы увидеть рабочее пространство по умолчанию.

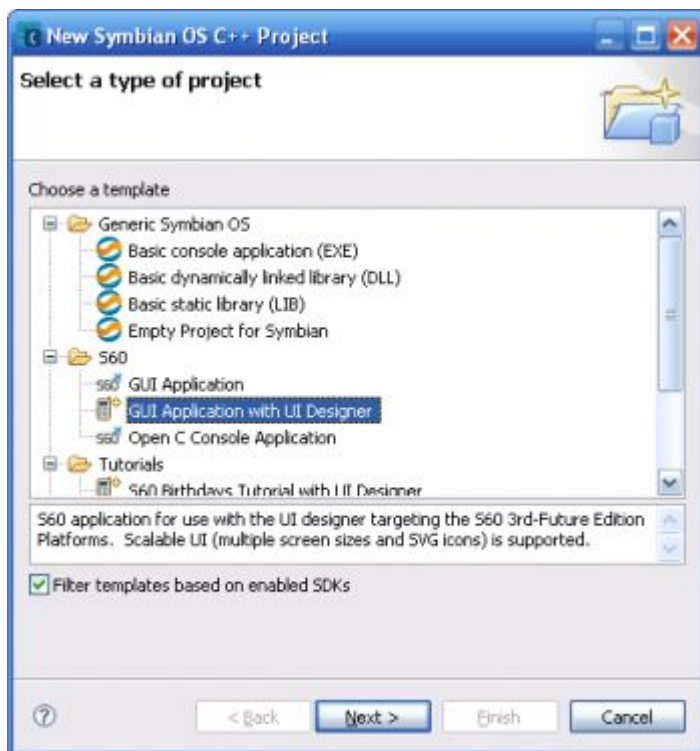


## Создание проекта

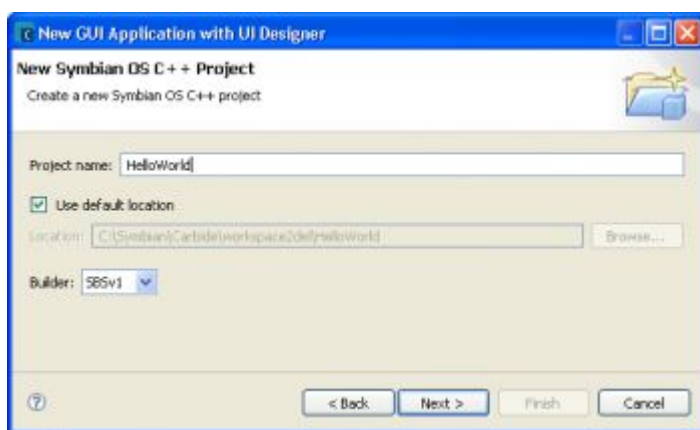
Чтобы создать новый проект, выберите **File | New | Symbian OS C++ Project**.



Выберите шаблон приложения **GUI Application with UI Designer** (приложение с графическим интерфейсом). Этот шаблон создает работающее приложение с графическим интерфейсом пользователя (**GUI**), с возможностью строения интерфейса с помощью инструмента **UIDesigner** (позволяет конструировать внешний вид программы «визуально», с помощью мыши и перетаскивания).



Следующая страница Мастера (кнопка **Next**) называется **New Symbian OS C++ Project**, или по-нашему - **Новый Symbian C++ проект**. Тут мы должны указать название проекта, в нашем случае «HelloWorld». Еще раз убедитесь, что папка проекта находится на одном диске с SDK, и не специальных символов и пробелов (это относится и к кириллице!).



Следующая страница Мастера (кнопка **Next**), называется **Symbian OS SDK's**, тут мы выбираем SDK платформы для какой и будем создавать приложение (S60 2ed или S60v3 (отдельно есть и для FP1 и FP2), или например S60v5).

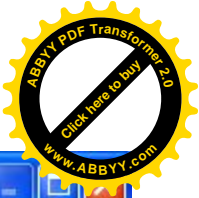


По умолчанию, все конфигурации будут отмечены:

- **Emulator Debug (WINSW)** собирает бинарные файлы для Windows-эмулятора Symbian.
- **Phone Debug | Release (GCCE)** собирает бинарные файлы для смартфона с помощью бесплатного компилятора **GCCE** (устанавливается вместе с SDK автоматически).
- **Phone Debug | Release (ARMV5)** бинарные файлы для смартфона с помощью компилятора **RVCT(ARM RealView Compiler)**. RVCT создает код, который немного компактней и быстрее, чем создаваемый GCCE, но должен быть отдельно приобретен в ARM. В основном используется производителями для компиляции внутреннего ПО смартфона («вшитого»).

Обычно, программисты убирают флажок с конфигураций содержащих в названии ARMV5, так как не имеют лицензированного компилятора RVCT; мы поступим так же ☺.

Жмем **Next**, и попадаем на страницу мастера, которая называется **Application properties - Свойства приложения**.



**New GUI Application with UI Designer**

**Application Properties**  
Enter properties for creating a new application

Application name:

Design for baseline SDK:

Initial language:

Тут мы ничего не меняем. Поле **Baseline SDK** определяет общий уровень совместимости нашего приложения. Например, если выбрать **S60 5th Edition SDK**, инструмент **UI Designer** не позволит вам добавлять элементы, которые доступны в поздней версии платформы **S60 5th Edition**, Feature Pack 1.

Следующая страница Мастера позволяет выбрать дизайн графического интерфейса. Оставляем выбор на варианте **Empty** (пустой).

Дальше идет страница **Container Name and Type**, в которой обачно не приходится что-то менять. Обратите внимание, что на этой странице кнопка **Finish** (**Готово**) уже активна. Это означает что следующая страница с настройками приложения необязательна, но в целях этого руководства, вы должны нажать **Next**.

**New GUI Application with UI Designer**

**Container Name and Type**  
Enter a base class name for your new class files and select a container type for your content

Base name:

Container type:

☒ Support view switching

Avkon View-Switching architecture uses the View Server to activate your designs. Choose this also for tabbed views. For most designs, it is recommended

Description:  
An empty container where controls can be placed at different coordinates, allowing various kinds of screen layouts.

? < Back Next > Finish Cancel

Даже если наше приложение использует только один режим вида(портретный или ландшафтный например), лучше отметить **Support View Switching** (поддержка переключения вида), для большего удобства в расширяемости программы в будущем.

Жмем **Next**.Появляется страница **Basic Settings** или **Базовые установки**.Она используется для добавления в проект таких данных, как ваше имя, текст копирайта.Также тут мы указываем уникальный идентификатор программы – **UID**.

**New GUI Application with UI Designer**

**Basic Settings**  
Basic properties of a project

Application UID:

Author:

Copyright notice:

? < Back Next > Finish Cancel

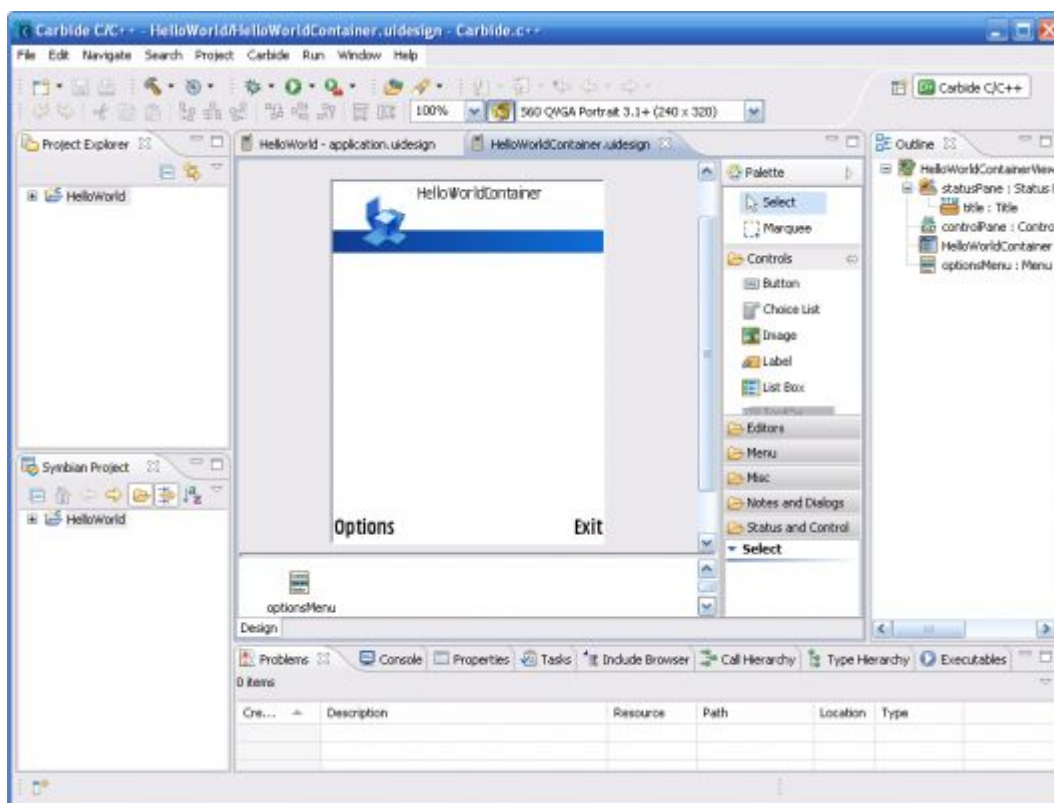
**UID** определяет защищенную область в файловой системе устройства, где приложение сохраняет свои данные.Также этот номер используется для индентификации или запуска приложения.

Carbide.C++ генерирует случайный UID, с диапазона специально зарезервированных для тестирования приложений.Если вы захотите опубликовать вашу программу, то для этого нужно получить уникальный UID в [Symbian Signed](#).

Так как мы не собираемся выпускать нашу программу «на свет», будем использовать случайно выбранный UID(его можно поменять позже).




Все, жмем **Finish**, Мастер закрывается, и создает наше приложение. Открывает рабочее пространство, похожее на это:

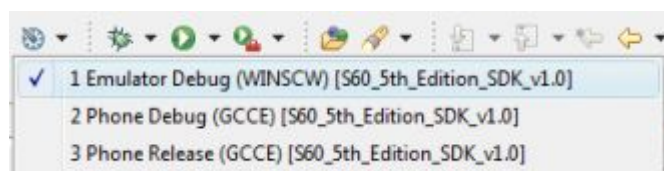


## Разработка с использованием эмулятора


Обычно, программисты сначала создают программу для эмулятора, его можно использовать для большинства случаев разработки (даже если необходимо соединение с интернетом или наличие GPS).

### Сборка проекта под эмулятор

- Первым делом необходимо задать активную конфигурацию сборки, сделать это можно нажав **Manage configurations for the current project** (Управление конфигурациями для текущего проекта) с такой иконкой на панели инструментов -  или через меню: **Project | Build Configurations | Set Active** и выбрав **Emulator Debug**.




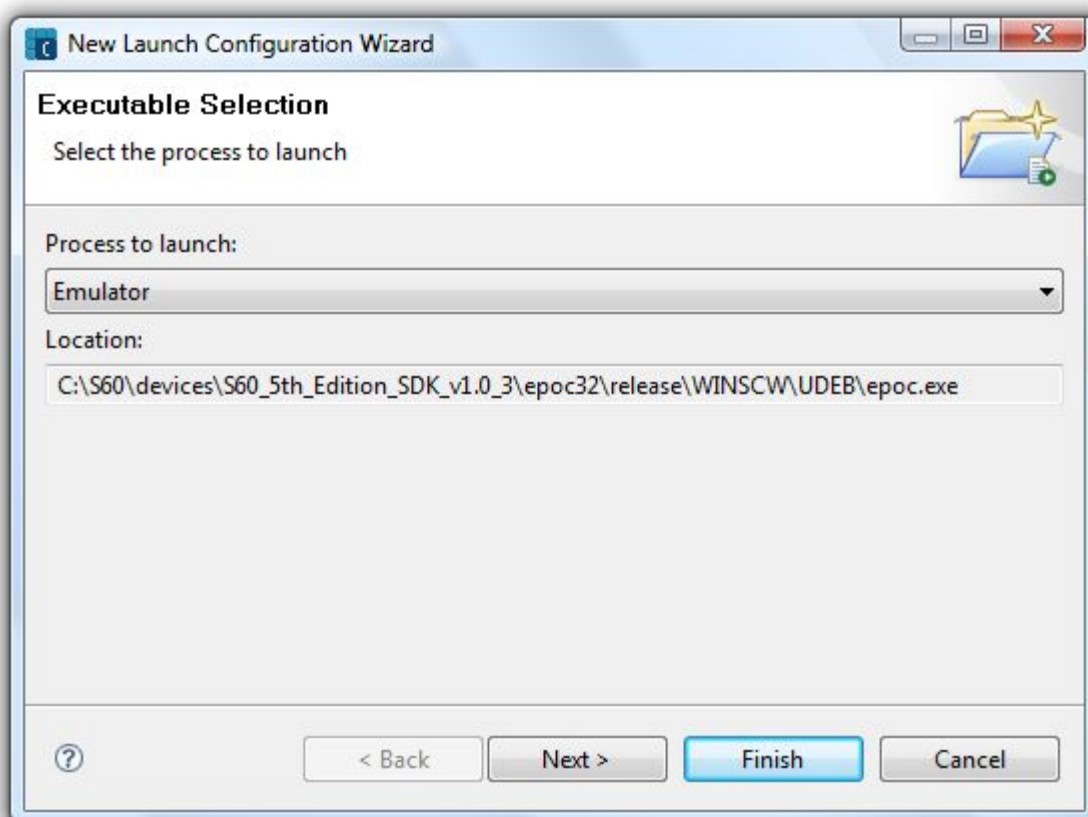


- Затем собираем текущую конфигурацию, используя иконку **Build\_**  на панели инструментов, или через меню: **Project | Build Project**.

**Важно:** если вы получаете сообщение похоже на «WARNING: EPOCROOT does not specify an existing directory», значит рабочее пространство и SDK находятся на разных дисках! Переместите проект на диск, одноименный с тем, где установлена SDK.

## Запуск приложения на эмуляторе

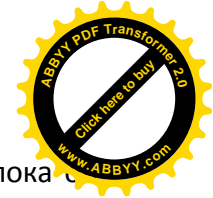
Если приложение было успешно собрано, жмем кнопку **Run (Запуск)**  (Ctrl + F11). Во время первого запуска проекта, Carbide.C++ спросит вас какой бинарный файл запускать.



- Если выбрать *HelloWorld.exe*, эмулятор включится и запустит нашу программу автоматически, и автоматически закроется при выходе из программы.
- Если выбрать *Emulator*, запустится эмулятор, и вы должны будете запустить свою программу через иконку в нем. А теперь объясню, как найти приложение в эмуляторе.

Может показаться, что второй метод сложнее, но он имеет некоторые преимущества – вы можете оставить эмулятор запущенным, при этом редактировать свой проект, пересобирать, и запускать программу без необходимости перезапуска эмулятора. Также вы сможете увидеть какие-то ошибки при выходе из программы, если они имеют место. (Эти ошибки также можно обнаружить через окно консоли Carbide.c++).





Когда эмулятор запускается первый раз, может понадобиться немного времени пока он включится полностью.




Если вы решили загрузить эмулятор и запустить ваше приложение: откройте меню через значок в левом углу экрана(скриншот сверху).Ваше приложение будет находится в папке **Applications** на самом низу.

Когда программа запустится, вы увидите пустой экран с заголовком приложения.Как видно, пример создал приложение с уже готовыми кнопками и окнами.

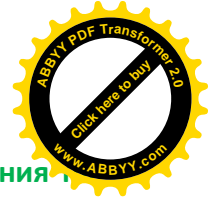


## Отладка в эмуляторе

Эмулятор является средством отладки по умолчанию, поэтому для отладки просто нажмите кнопку **Debug** .Процесс отладки в эмуляторе в этом руководстве почти не рассматривается.

## Разработка для устройства


Эмулятор может быть использован для разработки большинства проектов, но некоторые ситуации все-таки нуждаются в использовании реального устройства, например, когда требуется работа с акселерометром или камерой.

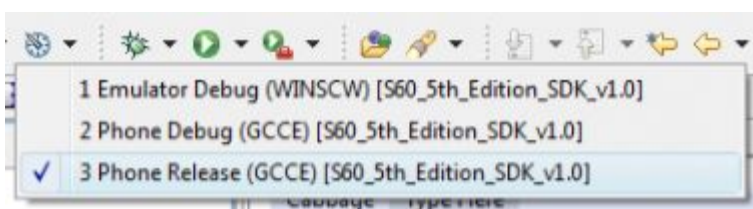



Примечание: вы должны тестировать ваши приложения на смартфоне время от времени, не зависимо от того, насколько хорошо оно работает на эмуляторе.

Когда вы закончите разработку проекта, то скорей всего, захотите выпустить релизную версию, убрав из бинарных файлов отладочную информацию для большей эффективности и уменьшения размеров программы.

## Сборка проекта для устройства

Чтобы сообщить IDE, что мы хотим собирать проект под телефон, необходимо поменять активную конфигурацию сборки на релизную для GCCE(ну если у вас конечно не имеется компилятора RVCT ☺).Как и раньше, используем **Manage configurations for current project**  для выбора конфигураций.



Следующее что делаем, собираем проект с новой конфигурацией с помощью иконки **Build**  на панели инструментов. При этом проект скомпилируется GCCE, который создаст инсталляционный пакет – файл *HelloWorld.sisx* в папке проекта *\HelloWorld\sis\*. Этот файл можно установить на телефоне.

**Статью перевел Андрей Ярощук aka Athloni. Жду ваших комментариев и предложений. Стоит ли продолжать?**