



Code less.
Create more.
Deploy everywhere.

Qt for S60

A technical look into Qt for S60

Pekka Kosonen

Head of Technology Development / Forum Nokia

Company Confidential

1 © 2008 Nokia V1-Filename.ppt / YYYY-MM-DD / Initials

NOKIA



Disclaimer

Note that Qt on S60 is not ready yet!

**What is released is an early technical pre-release
with limited functionality for early adopters.**

**Some of the presented functionality may change in
the final release.**



Code less.
Create more.
Deploy everywhere.

Qt for S60 – Developer offering

Pekka Kosonen

Head of Technology Development / Forum Nokia

Company Confidential

3 © 2008 Nokia V1-Filename.ppt / YYYY-MM-DD / Initials

NOKIA



Motivation for bringing in Qt to S60

- We already have huge mass of Symbian developers out there
- But... Symbian development productivity is not best possible
- Getting new developers to learn Symbian? The learning curve is too long when compared to competitive offering.
- Runtimes like Java and Flash? Good, but runtime is a runtime, it always has it's restrictions.
- We need a native solution that is productive and easy to learn.



Contents

1. Qt for S60 developer offering
2. Environment setup
3. Feedback channels
4. Introducing Qt on S60
5. Build Process
6. Symbian Concepts
7. UI Building Blocks
8. Tips & Tricks



Qt for S60 Total Offering

- **Qt for S60 port** – Qt 4.5 ported to S60 3rd Edition FP 1 and FP 2, 5th edition and also coming S60 platform editions.
- **Carbide.c++ 2.0.2** supporting Qt/S60 development.
 - Fully integrated form editor (*Qt Designer*)
 - Wizards for creating new Qt projects and classes
 - Integrated .pro file editor
 - Automated build setup for moc, uic, and rcc
 - [http://www.forum.nokia.com/Resources and Information/Tools/IDEs/Carbide.C++/](http://www.forum.nokia.com/Resources_and_Information/Tools/IDEs/Carbide.C++/)
- **Qt for S60 Mobile Extension**
 - High abstraction level APIs making it possible to use most important services without using native Symbian C++
- Documents, examples, discussion board, wiki etc.



Qt for S60 - Forum Nokia resources

- Qt for S60 **Developer's Library** (online version & Carbide plugin)
 - [http://www.forum.nokia.com/info/sw.nokia.com/id/c41e7898-2dd8-4f23-a629-d27727519ffa/Qt for S60 Developers Library.html](http://www.forum.nokia.com/info/sw.nokia.com/id/c41e7898-2dd8-4f23-a629-d27727519ffa/Qt_for_S60_Developers_Library.html)
- Qt for S60 **Discussion board** (monitored by FN experts)
 - <http://discussion.forum.nokia.com/forum/forumdisplay.php?f=196>
- Qt for S60 **Wiki**
 - [http://wiki.forum.nokia.com/index.php/Category:Qt for S60](http://wiki.forum.nokia.com/index.php/Category:Qt_for_S60)
- All the FN documents will be linked to from the following page
 - [http://www.forum.nokia.com/Resources and Information/Documentation/Qt for S60.xhtml](http://www.forum.nokia.com/Resources_and_Information/Documentation/Qt_for_S60.xhtml)
- **Carbide.c++ 2.0** as an IDE for development
 - [http://www.forum.nokia.com/Resources and Information/Tools/IDEs/Carbide.c++/](http://www.forum.nokia.com/Resources_and_Information/Tools/IDEs/Carbide.c++/)
- **S60 SDKs** that can be used for Qt development on S60
 - 3rd Edition FP1 <http://www.forum.nokia.com/info/sw.nokia.com/id/4a7149a5-95a5-4726-913a-3c6f21eb65a5/S60-SDK-0616-3.0-mr.html>
 - 3rd Edition FP2 and 5th Edition http://www.forum.nokia.com/info/sw.nokia.com/id/ec866fab-4b76-49f6-b5a5-af0631419e9c/S60_All_in_One_SDKs.html
- **Open C/C++**
 - [http://www.forum.nokia.com/Resources and Information/Explore/Runtime Platforms/Open C and C++/](http://www.forum.nokia.com/Resources_and_Information/Explore/Runtime_Platforms/Open_C_and_C++/)
- Qt for S60 **overview**
 - [http://www.forum.nokia.com/Resources and Information/Tools/Runtimes/Qt for S60/](http://www.forum.nokia.com/Resources_and_Information/Tools/Runtimes/Qt_for_S60/)
- Qt for S60 **Quickstart** guide
 - [http://www.forum.nokia.com/Resources and Information/Tools/Runtimes/Qt for S60/QuickStart.xhtml](http://www.forum.nokia.com/Resources_and_Information/Tools/Runtimes/Qt_for_S60/QuickStart.xhtml)
- Qt for S60 **Examples** (more in the SDK)
 - [http://www.forum.nokia.com/info/sw.nokia.com/id/63313e06-8fd7-4a68-8610-80dd7ee22745/Qt for S60 Examples.html](http://www.forum.nokia.com/info/sw.nokia.com/id/63313e06-8fd7-4a68-8610-80dd7ee22745/Qt_for_S60_Examples.html)

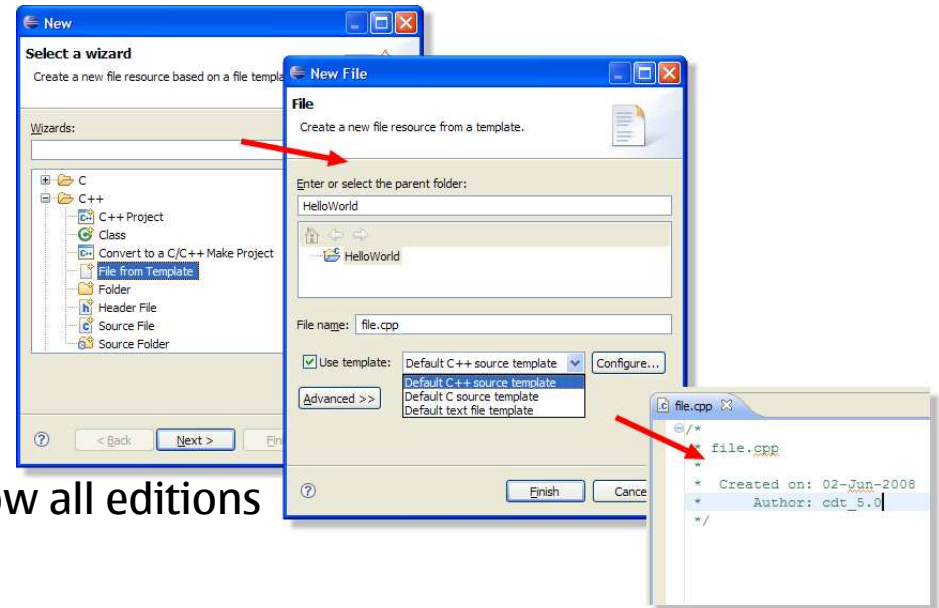


Qt Software Developer Offering

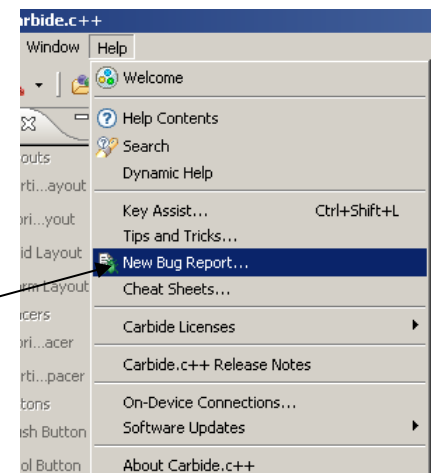
- <http://trolltech.com> - main place for Qt related development material
- We recommend that you become familiar with Qt by first [evaluating](#) it prior to using the technology preview of the S60 port.
 - Check <http://trolltech.com/downloads>
 - Trolltech developer zone is full of great Qt material:
<http://trolltech.com/developer>
- The S60 prerelease documentation is available in the release package
<http://pepper.troll.no/s60prereleases/doc/index.html>
- You also have a local copy of the documentation in the installation folder, default starting point being
<C:\Qt\4.5.0-garden\doc\html\index.html>



Carbide.c++ 2.0



- New version published in December 2008, now all editions **free of charge**
- Qt for S60 support!
- With Carbide.c++ installed, you might want to read the [Carbide.c++: Introductory White Paper](http://www.forum.nokia.com/info/sw.nokia.com/id/cae9ea59-eee0-4b98-aaa2-1b6ecd879222/Carbide_cpp_Introductory_White_Paper_V1_1_en.pdf.html)
 - [http://www.forum.nokia.com/info/sw.nokia.com/id/cae9ea59-eee0-4b98-aaa2-1b6ecd879222/Carbide cpp Introductory White Paper V1 1 en.pdf.html](http://www.forum.nokia.com/info/sw.nokia.com/id/cae9ea59-eee0-4b98-aaa2-1b6ecd879222/Carbide_cpp_Introductory_White_Paper_V1_1_en.pdf.html)
- And view the [Getting Started with Carbide.c++ Express Screencast](http://www.forum.nokia.com/info/sw.nokia.com/id/af80987a-a72d-44cf-bf00-1926be01058c/Carbide_cpp_Getting_Started_v1_1_en.exe.html), which will take you through the basics of creating, building, and deploying your first application.
 - [http://www.forum.nokia.com/info/sw.nokia.com/id/af80987a-a72d-44cf-bf00-1926be01058c/Carbide cpp Getting Started v1 1 en.exe.html](http://www.forum.nokia.com/info/sw.nokia.com/id/af80987a-a72d-44cf-bf00-1926be01058c/Carbide_cpp_Getting_Started_v1_1_en.exe.html)
- Bug reports : Carbide Help menu, "New Bug Report"

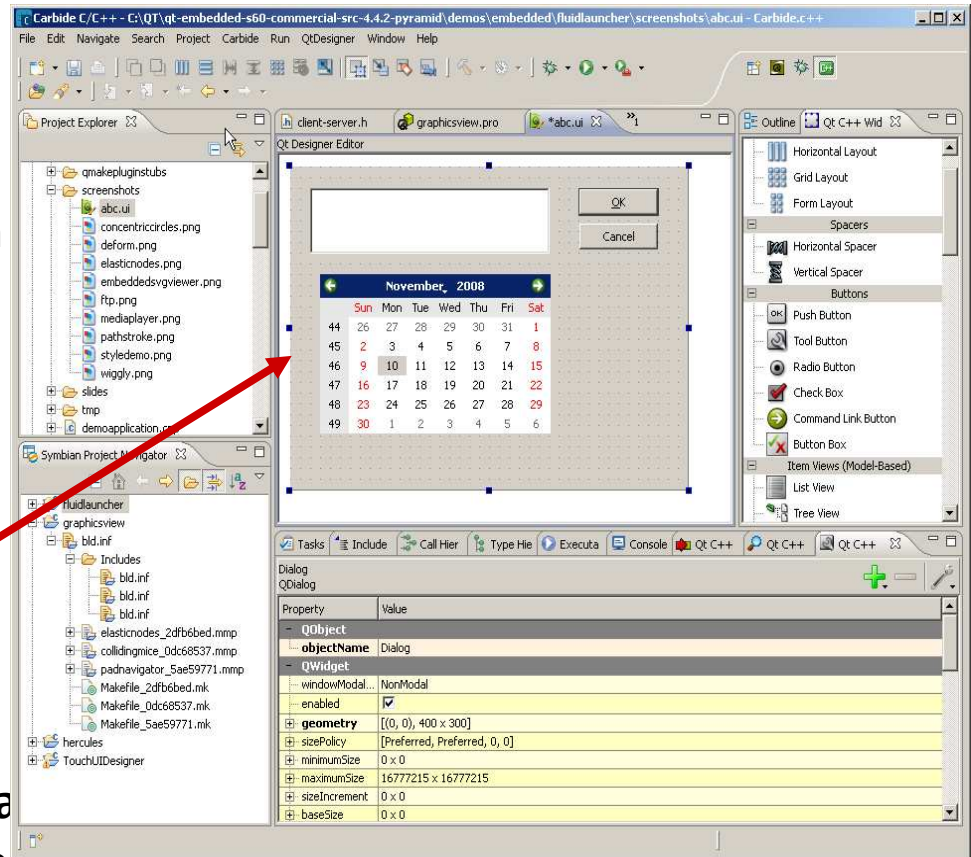


Company Confidential



Qt Support in Carbide.c++ 2.0

- Support for Qt development
- Carbide works with Qt projects
 - .PRO editor for easy project configuration
 - Package /deploy projects to phone
 - Qt plug-ins are included in Carbide
 - Qt project Wizard
 - Fully integrated form editor (*Qt Designer*)
 - Wizards for creating new Qt projects and classes
 - Integrated .pro file editor
 - Automated build setup for moc, uic, and rcc
- Oh yeah, since **Qt application on S60 is a Symbian application**, you can use **same tools for debugging, profiling etc.**



Company Confidential



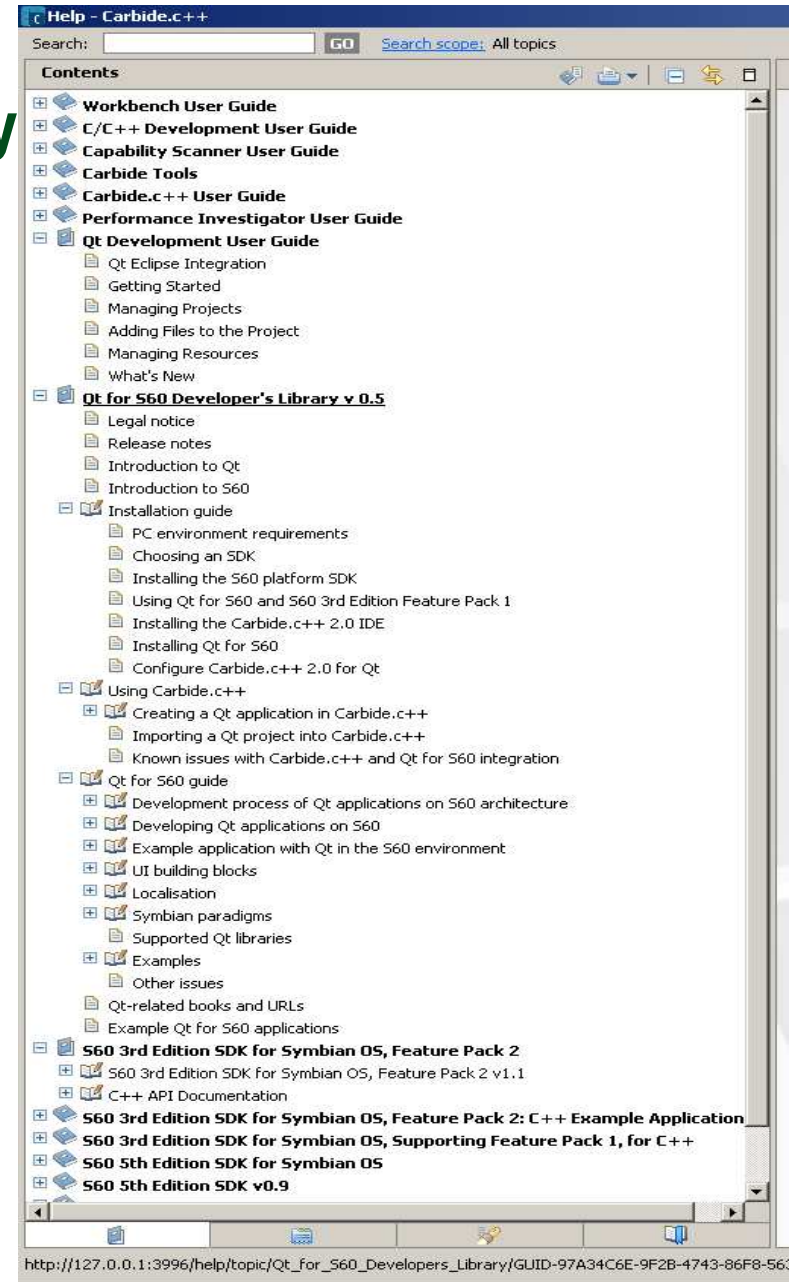
Qt for S60 Developer's Library

- [http://www.forum.nokia.com/info/sw.nokia.com/id/c41e7898-2dd8-4f23-a629-d27727519ffa/Qt for S60 Developers Library.html](http://www.forum.nokia.com/info/sw.nokia.com/id/c41e7898-2dd8-4f23-a629-d27727519ffa/Qt%20for%20S60%20Developers%20Library.html)
- Available as an online version and as a Carbide plugin
- Installing the Carbide Eclipse plugin:
 1. Download the Eclipse plugin zip file to any folder on your computer
 2. Unzip the zip file.
 3. Copy the JAR file included in the zip file.
 4. Paste the JAR file in the eclipse\plugins folder of your Eclipse installation, f.ex [C:\Program Files\Nokia\Carbide.c++ v2.0\plugins](#)
- The Qt for S60 Developer's Library appears in the Eclipse Help Contents menu the next time you open your Eclipse installation.
 - Let's have a quick look into it



Qt for S60 Developer's Library

- After installing the 'Qt for S60 Developer's Library' Carbide plugin you will see it in Carbide help (select Help->Help Contents from Carbide menu bar)
 - Note that the [Qt Development User Guide](#) that is 'built-in' with Carbide 2.0 is a separate library
- The developers library contains the following sections
 - [Legal notice & Release notes](#)
 - [Introduction to Qt](#)
 - [Introduction to S60](#)
 - [Installation guide](#)
 - [Using Carbide.c++](#)
 - [Qt for S60 guide](#)
 - [Qt-related books and URLs](#)
 - [Example 'Qt for S60' applications](#)



Company Confidential



Looking for examples?

- The Qt for S60 garden release contains 35 examples and 5 demo applications
- Additional examples in [http://www.forum.nokia.com/info/sw.nokia.com/id/63313e06-8fd7-4a68-8610-80dd7ee22745/Qt for S60 Examples.html](http://www.forum.nokia.com/info/sw.nokia.com/id/63313e06-8fd7-4a68-8610-80dd7ee22745/Qt_for_S60_Examples.html)
- In addition to these you can **take any Qt open source project** and compile it for S60
 - For examples targeted for desktop naturally **UI optimization** needs to be done
 - If the example you've found doesn't work on S60 it indicates that some of the used features is not ready yet!
- <http://www.qt-apps.org/> contains hundreds of applications (with source codes)



Contents

1. Qt for S60 developer offering
2. Environment setup
3. Feedback channels
4. Introducing Qt on S60
5. Build Process
6. Symbian Concepts
7. UI Building Blocks
8. Tips & Tricks



Qt for S60 Development Environment 1/3

- Carbide.C++ 2.0
 - <http://www.forum.nokia.com/info/sw.nokia.com/id/dbb8841d-832c-43a6-be13-f78119a2b4cb.html>
- S60 SDKs that support Qt
 - 3rd Edition FP1 + Open C/C++
 - <http://www.forum.nokia.com/info/sw.nokia.com/id/4a7149a5-95a5-4726-913a-3c6f21eb65a5/S60-SDK-0616-3.0-mr.html>
 - [http://www.forum.nokia.com/Resources and Information/Explore/Runtime Platforms/Open C and C++/](http://www.forum.nokia.com/Resources%20and%20Information/Explore/Runtime%20Platforms/Open%20C%20and%20C%2B%2B/)
 - 3rd Edition FP2 and 5th Edition
 - [http://www.forum.nokia.com/info/sw.nokia.com/id/ec866fab-4b76-49f6-b5a5-af0631419e9c/S60 All in One SDKs.html](http://www.forum.nokia.com/info/sw.nokia.com/id/ec866fab-4b76-49f6-b5a5-af0631419e9c/S60%20All%20in%20One%20SDKs.html)
- Qt for S60 technology preview
 - <http://trolltech.com/developer/technical-preview-qt-for-s60>



Qt for S60 Development Environment 2/3

- First you need an S60 SDK ☺
- Get the Qt for S60 pre-release from: <http://pepper.troll.no/s60prereleases/>
 - Unzip the zip package to desired location (like C:\Qt\4.5.0-garden)
 - In Garden, we have two SIS packages (one for 3.x, one for 5.x) – in future, only one SIS package will be provided.
- Set the PATH environment variable to point to Qt tools. This is done by adding c:\Qt\4.5.0-garden\bin to the PATH variable.
- Configure Qt for S60
 - `cd \Qt\4.5.0-garden`
 - `configure -platform win32-mwc -xplatform symbian-abld`
 - Getting errors? Check that you don't have other make tools in the path
- The Qt libraries comes pre-built for real devices.
 - If you are using a 3.x S60 SDK, install qts60binaries\3.x\qtlibs-4.5.0-garden.exe into the S60 SDK root directory.
 - If you are using a 5.0 S60 SDK, install qts60binaries\5.0\qtlibs-4.5.0-garden.exe into the S60 SDK root directory.



Qt for S60 Development Environment 3/3

- To build Qt for the emulator, type:
 - `make debug-winscw`
- There is a self-signed fluid launcher example available in <http://pepper.troll.no/s60prereleases/> that illustrates some Qt examples
 - In case of S60 3rd Edition FP1 device is used, in addition Open C needs to be installed
- Videos about environment setup are also available in <http://pepper.troll.no/s60prereleases/>
 - <http://www.youtube.com/watch?v=zf9XYs902Mw>



The environment

- The Qt libraries for emulator are compiled using Symbian build tools
 - As a result, a bunch of Qt*.dll and Qt*.lib files are produced into SDK emulator folders (epoc32)
- The libraries for device builds (gcce and armv5) are provided, no need to build them
- For installing Qt to devices pre-made .sisx files are provided
 - Note, with Garden there are separate libraries for 3rd edition and 5th edition which are not compatible together
 - In the future there will be only one so don't fear BC breaks in final product!



Contents

1. Qt for S60 developer offering
2. Environment setup
3. Feedback channels
4. Introducing Qt on S60
5. Build Process
6. Symbian Concepts
7. UI Building Blocks
8. Tips & Tricks



Feedback

- Preferred place for feedback and for getting possible issues solved is Forum Nokia's **Qt for S60 Discussion board** (monitored by FN experts)
 - <http://discussion.forum.nokia.com/forum/forumdisplay.php?f=196>
- If you're certain there's a **bug** you can report it directly to qts60-feedback@trolltech.com
 - Prefer the discussion board, FN will report the needed to the R&D team
- You have examples/knowledge to share? Put it to **Qt for S60 Wiki**:
 - http://wiki.forum.nokia.com/index.php/Category:Qt_for_S60
 - Please notify the discussion board about the new wiki page!
- **Carbide bugs?** Report from Carbide Help menu, "New Bug Report"
- Once Qt for S60 reaches beta (~month before final) FN will have **Technical Support** including "official bug reporting channel"
 - http://www.forum.nokia.com/main/technical_services/technical_support/index.html



Problems getting the Qt application running on phone?

- Ensure your phone is S60 3rd Edition FP1 or newer (for device information see <http://www.forum.nokia.com/devices/>)
- Make sure you have the Qt DLLs installed to your phone (signed sis files are provided in the Garden release, see 'qts60binaries' folder)
- In case of S60 3rd Edition FP1 phone ensure you have Open C PIPS installed
- Can't install self signed applications to your phone?
 - From phone settings start 'Application manager', select 'Options' and set 'Software installation' to 'All' instead of 'Signed only', see f.ex <http://www.s60tips.com/2006/05/25/installing-self-signed-applications-in-s60-3rd-edition/>
 - Note; In self-signed apps the UID3 should be starting with 0xE
- The application doesn't start or crashes?
 - Maybe you're using some Qt feature that is not ready yet? If in doubt remember the discussion board
- With any issues remember our documentation, wiki and discussion board!



Contents

1. Qt for S60 developer offering
2. Environment setup
3. Feedback channels
4. Introducing Qt on S60
5. Build Process
6. Symbian Concepts
7. UI Building Blocks
8. Tips & Tricks



Qt for S60

- Qt Software announced in **October 2009 the porting of Qt to S60** on Symbian OS(TM), the world's leading, open smartphone platform. Qt is a powerful C++ application development framework, which makes it easy for developers to create applications once and then deploy them on any of the Windows, Mac, Linux, Windows CE, Windows Mobile and embedded Linux platforms.
- An early technical preview of Qt for S60 is available for download from Qt Software current website www.qtsoftware.com. This is in line with Qt Software's usual approach of developing Qt openly and working with the community to incorporate feedback throughout the entire development process. Qt for S60 will work with **S60 3rd Edition Feature Pack 1 and later releases**. The first **production release of Qt for S60 will be available in the second half of 2009**.
- With the inclusion of the S60 platform, developers have an **additional 80 million target devices** that they can support with their **Qt-based applications**.

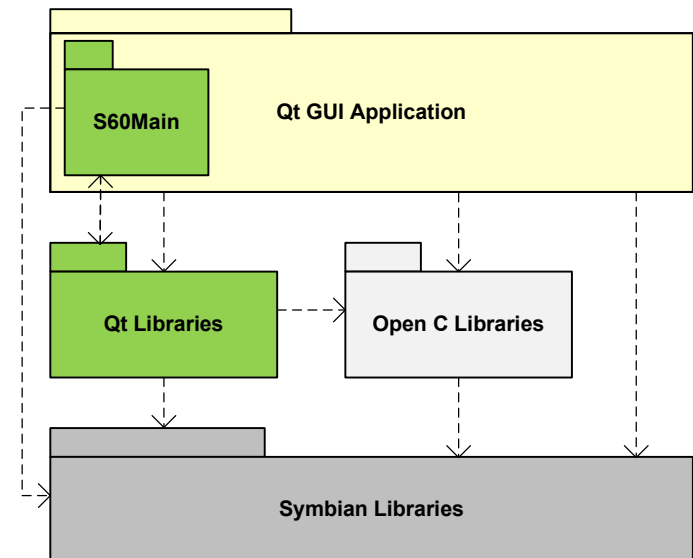


- Qt (starting with 4.5) is licensed under **LGPL** which means that you can do closed source commercial software without purchasing a license



Architecture

- Qt/S60 GUI applications are **statically linked against S60Main**. The S60Main implements Symbian OS entry point 'E32Main' for Qt/S60 applications. The S60Main **initializes the standard S60 application UI framework**.
- In essence it means that whenever the 'main' entry point of Qt GUI application gets called, the standard S60 application UI framework classes such as **CAknApplication, CAknDocument and CAknAppUI** are already instantiated. This further means that **control environment (CCoeEnv)** and **active scheduler (CActiveScheduler)** for main thread also exist.
- The control environment availability makes it possible to **integrate the top-level QWidgets to native controls** and it enables Qt widgets to utilize the native input methods also known as **Front End Processors (FEPs)**
- Qt uses the private implementation pattern and other mechanisms to **separate platform-specific implementations**. Due to this separation, the application **developer does not see any difference on APIs**.





Contents

1. Qt for S60 developer offering
2. Environment setup
3. Feedback channels
4. Introducing Qt on S60
5. Build Process
6. Symbian Concepts
7. UI Building Blocks
8. Tips & Tricks

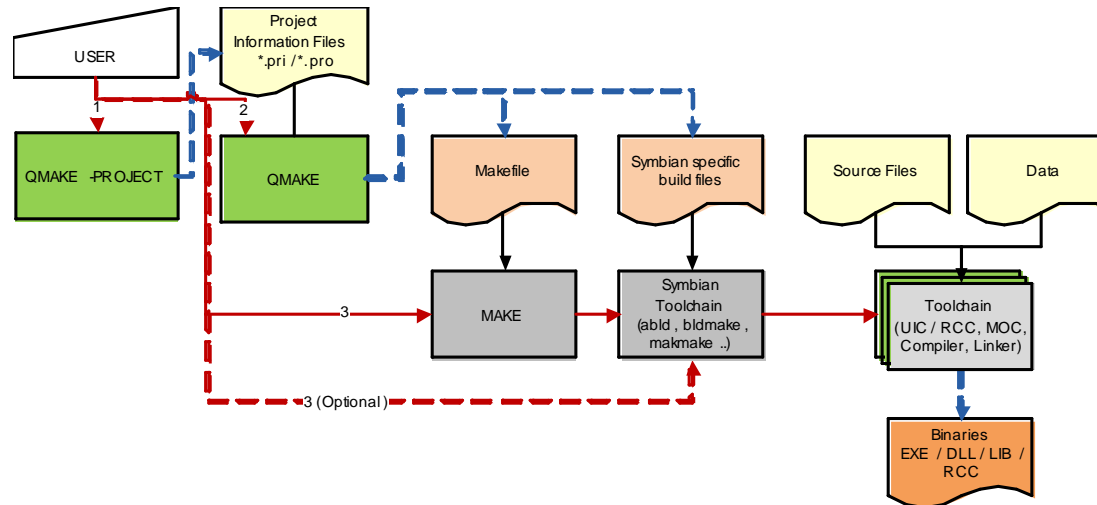


Build Process

- The contents of a project are described in a **project file** (.pro) that is the master file of the project. Generally it lists the **source and header files**, **configuration** information, and **application-specific details** (such as libraries and include paths).
- A default project file **can be generated** from known extension (.h, .cpp, .ui etc) using the command
 - `qmake -project`
- Qt's .pro files are the master files used in Qt/S60. **Symbian build files are generated by the `qmake` command and should never be manually edited.**
- Building a Qt application on the S60 platform differs from building an S60 application only at the beginning of the toolchain. We have also provided wrapper make commands hiding Symbian build commands completely.



Build Process



- qmake generates Symbian specific **bld.inf**, **mmp** files, **default registration file**, **extension makefiles** and **package files** based on the project file.
- Extension makefiles are used to integrate Qt-specific tools like moc, uic and rcc to the Symbian toolchain.



make command

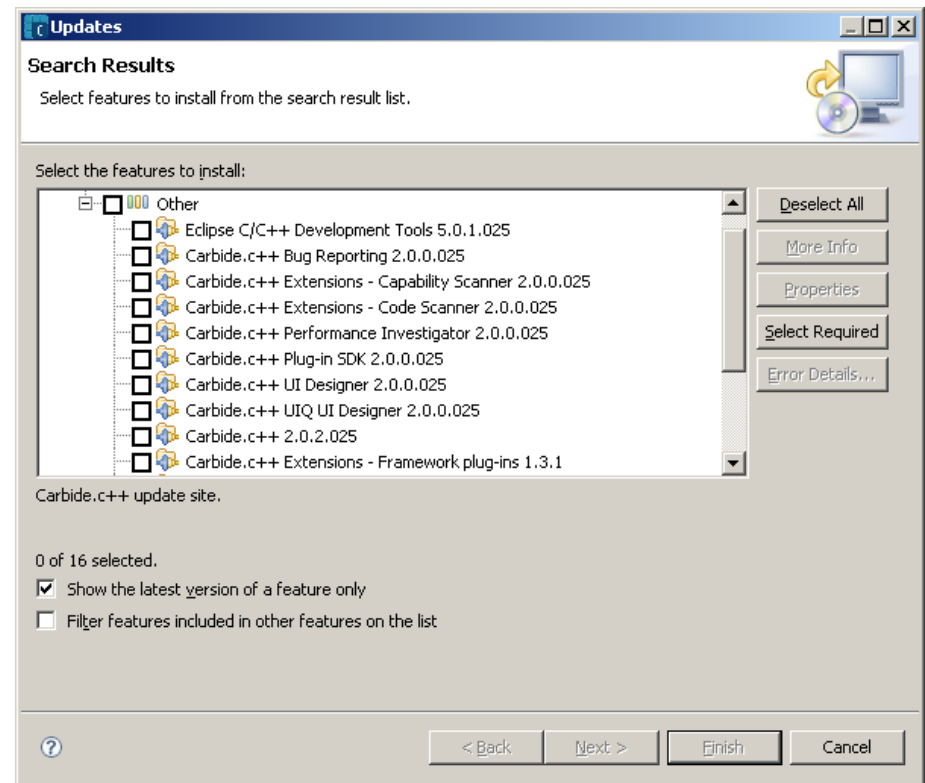
- Running make command calls bldmake and abld to build the Qt application for the S60 environment.

Command	Description
make	Creates abld and the project makefiles and builds debug build of the application for the emulator (winscw udeb).
make debug	Creates debug builds (winscw/gcce/armv5 udeb).
make debug-winscw	Creates winscw debug build.
make debug-gcce	Creates gcce debug build.
make debug-armv5	Creates armv5 debug build.
make release	Creates release builds (gcce/armv5 urel).
make release-gcce	Creates gcce release build.
make release-armv5	Creates armv5 release build.
make export	Copies the exported files to their destination.
make cleanexport	Removes files created with <code>make export</code> .
make mocclean	Removes the header and source files created by the moc tool.
make mocables	Runs moc tool on necessary files.
make clean	Removes everything built with abld target, exported files and makefiles.
make distclean	As make clean, but also removes all Symbian specific files created with qmake.
make confclean	As make distclean, but also cleans everything generated by configure call. Note that this command is only available in the Qt root directory.
make run	Launches the application in the emulator.



Carbide and qmake

- Carbide version 2.0.2 runs the qmake when Qt project is imported and generates the needed files. This doesn't function in Carbide 2.0. In case you run into problems execute qmake yourself from command prompt, f.ex
 - Cd C:\extensions\camera\example
 - qmake CameraEx.pro //the files will be generated
- Updating Carbide can be done from Carbide Help menu:
Help -> Software updates-> find and install





Symbian Extensions to Project Files

- A Qt application on S60, is a Symbian application written in Qt running on S60 devices.
- All statements that are in a Symbian project defined in the mmp file are defined in the project file when creating application with Qt in the S60 environment. **Executing qmake overrides all changes made manually to the mmp file.**
 - Trick : if you really need to have manual changes put the file as read only!
- Target type is specified using `TEMPLATE` and `CONFIG` variables in the .pro file.
 - `TEMPLATE = app` value maps to target type EXE in MMP file,
 - `TEMPLATE = lib`, `CONFIG` variable is used to distinguish between lib and dll target types.
- Name of application executable – use `TARGET` statement in .pro file (maps to `TARGET` in MMP file). Example: `TARGET = QtS60HelloWorld`
- Symbian specific definitions in the project file should be separated from other definitions using a Symbian keyword. Example:

```
symbian: {  
    TARGET.UID2 = 0x100039CE  
    TARGET.UID3 = 0xA000017F  
}
```



Symbian Extensions to Project Files

- Some other examples of Symbian specific keywords:
 - `TARGET.SID = 0xA000017F`
 - `TARGET.VID = 0x70000001`
 - `TARGET.EPOCSTACKSIZE = 0x5000 // 20kb`
 - `TARGET.EPOCHEAPSIZE = 0x20000 0x1000000 // Min 128kb, Max 16Mb`
 - `TARGET.CAPABILITY = NetworkServices`
- The **DEFINES** keyword can be used to add **MACRO** statements to the mmp file.
- **QMAKE_CXXFLAGS** can be used for setting compiler-specific options. The keyword is followed by the specific compiler:

```
QMAKE_CXXFLAGS.CW += -O2
QMAKE_CXXFLAGS.ARMCC += -O0
```




Symbian extensions to project files

- RSS_RULES keyword for adding information to the _reg.rss file
- BLD_INF_RULES keyword for adding information to the bld.inf file
- Bunch of other extensions can be found in the Qt for S60 documentation – see more in:
[http://library.forum.nokia.com/topic/Qt for S60 Developers Library/GUID-97A34C6E-9F2B-4743-86F8-563362F7047A cover.html](http://library.forum.nokia.com/topic/Qt%20for%20S60%20Developers%20Library/GUID-97A34C6E-9F2B-4743-86F8-563362F7047A%20cover.html)
- And <http://pepper.troll.no/s60prereleases/doc/qmake-variable-reference.html>



- Some **libraries are automatically added to the mmp file** when generating the file. These libraries are:
 - libc, libm, euser, libdl and QtCore,
 - static library libcrt0 for QtCore applications,
 - static library qtmain for QtGui applications,
 - QtGui, avkon, eikcore, eiksrv, ws32, apparc, cone, eikcoctl, bafl and efsrv for QtGui applicaitons.
- **Qt libraries** are added to the project file using the QT variable:
- If other **platform-specific libraries** need to be added to the project, this can be done by adding the **LIBS variable** to the project file with the appropriate library name preceded with a **‘-l’ prefix**:

```
QT += network
```

```
LIBS += -lbitgdi -lfbscli
```



Example : Build, run, install to device example

- C:\Qt\4.5.0-garden2\examples\graphicsview\collidingmice
 - First change to `view.showMaximized();`
- `qmake collidingmice.pro`
 - Have a look at the generated files
- `make debug-winscw`
- `epoc`
 - While waiting for emulator to start fill your tax report
- Build for device : `make debug-gcce`
- Create self signed package and install to device (connected via PC Suite)
 - `createpackage -i collidingmice_gcce_udeb.pkg`
- You can also use the "old way" to build
 - `bldmake bldfiles`
 - `abld build winscw udeb //abld build gcce urel`
- Cleaning up the project: `make distclean`



Contents

1. Qt for S60 developer offering
2. Environment setup
3. Feedback channels
4. Introducing Qt on S60
5. Build Process
6. Symbian Concepts
7. UI Building Blocks
8. Tips & Tricks



Platform Security

- All **platform security rules also apply for Qt applications** in the S60 environment. Because Qt is mainly ported on top of Open C, the required capabilities are also derived from those APIs. **Platform security requires that needed capabilities be defined in the project file.** The Qt application may require, for example, the following capabilities:
 - AllFiles, when using file operations and accessing protected folders [6];
 - NetworkServices should be enough in most cases when using the QtNetwork module, but there might be certain API calls that also require NetworkControl.
- When using Symbian APIs the capabilities needed are, of course, the ones that the APIs define.

Access Capability	User Grantable	Open Signed Online	Open Signed Offline
LocalServices ReadUserData WriteUserData NetworkServices UserEnvironment	For testing and selling the application	For testing the application	For testing the application
Location SwEvent ProtServ TrustedUI PowerMgmt SurroundingsDD ReadDeviceData WriteDeviceData			
CommDD DiskAdmin MultimediaDD NetworkControl			
AllFiles DRM TCB			Device manufacturer approval required
Lead-time	Immediate	Immediate	Immediate
Note	Developer tested	Upload SIS	Certify on PC

Company Confidential



Exception Handling

- **Symbian code** that might leave should be **separated from Qt code**. This can be done by creating a **wrapper class with private implementation** where all **leaving functions that are needed are trapped**. If a leave occurs, the function with a trap returns the error code, and the error code can be handled appropriately in Qt code.

```
int PainterWrapperPrivate::ConstructPainter()  
{  
    TRAPD(error, activePainter = new (ELeave) CActivePainter());  
    activePainter->ConstructL();  
    return error;  
}
```

- The code above is an example of a function that handles the leaving code. This function (ConstructPainter()) can be called from Qt code without mixing the handling of Symbian specific leaving code with Qt code. Only the return value has to be handled in Qt code to check if an error has occurred.
- In the Garden release Qt/S60 does not handle exceptions / allow C++ exception handling (try/catch).



Memory Management

- Qt applications are implemented in such a way that the Symbian cleanup stack usage is not used by developers. If there is not enough memory when running the Qt application, the application will simply be closed. **This is something that we are investigating how to improve in future.**
- When implementing an application with Qt in the S60 environment, the **cleanup stack** should be **used with Symbian code.**
- **Qt stores objects into an object tree** when they are created. The object tree enables **automatic deletion of child objects** that have a parent.
 - For example, when a widget is created with another object as a parent, it is added to the parent's child list and deleted automatically when the parent is deleted.
 - If an object with a parent is created with new on the heap, the deletion of the object removes it automatically from the parent.
 - If the deleted object has children, they are automatically deleted when the object is deleted. The same behaviour applies to objects created on the stack.
 - The only objects that have to be explicitly deleted are the objects created with new and that have no parent.



Separating Qt and S60 implementations 1/2

- **S60 code should be separated from Qt code** so that it can be replaced with another platform-specific code when changing the platform.
Private implementation (aka pimpl) is an idiom that is used in Qt, and it can be utilised to separate platform-specific code — different platforms have different implementations of the private class. Binary compatibility can be retained with this idiom.
- The idea of cross-platform implementation is to have the **public interface** of the class the **same** for all platforms; **the private implementation** can then be **platform specific**.



Separating Qt and S60 implementations 2/2

- The main principles for private implementation are:
 - The **public class** has to have a **pointer to the private class**

```
AOWrapper::AOWrapper(QObject *parent)
    : QObject(parent)
{
    d_ptr = new AOWrapperPrivate(this);
}
```

- The constructor of the private class creates a pointer to the public class passed as a parameter:

```
AOWrapperPrivate::AOWrapperPrivate(AOWrapper *wrapper)
    : q_ptr(wrapper)
{
}
```

- The **q_ptr** pointer can be used to **refer to the public class from the private class** and the **d_ptr** pointer can be used to **refer to the private class from the public class**.
- For examples of using the private implementation you can have a look f.ex at the Mobile Extensions



Qt and Active Objects

- How to use Qt and Active Objects:
 - An example application exists:
[http://www.forum.nokia.com/info/sw.nokia.com/id/63313e06-8fd7-4a68-8610-80dd7ee22745/Qt for S60 Examples.html](http://www.forum.nokia.com/info/sw.nokia.com/id/63313e06-8fd7-4a68-8610-80dd7ee22745/Qt_for_S60_Examples.html)
 - And also documentation:
[http://www.forum.nokia.com/info/sw.nokia.com/id/c41e7898-2dd8-4f23-a629-d27727519ffa/Qt for S60 Developers Library.html](http://www.forum.nokia.com/info/sw.nokia.com/id/c41e7898-2dd8-4f23-a629-d27727519ffa/Qt_for_S60_Developers_Library.html)
- Sorry, too longish for this session ☺



Separating S60 specific implementation

- **When you need to write Symbian specific code use the following macro**

```
#if defined(Q_OS_SYMBIAN)
    #include <e32std.h>
#endif
```
- **One good way to separate Symbian code is to have it in a separate DLL (plugin)**



Contents

1. Qt for S60 developer offering
2. Environment setup
3. Feedback channels
4. Introducing Qt on S60
5. Build Process
6. Symbian Concepts
7. UI Building Blocks
8. Tips & Tricks



UI Building Blocks

- Qt has a set of built-in widgets, and some of them correspond to S60 UI components.
- In latest Qt for S60 release (Garden) we have implemented **S60 style** and provided an integration with S60 application and UI framework:
 - Qt widgets will **look like native S60 widgets when there is clear mapping.**
 - **Themes** are also **supported**
 - A Qt-based S60 application will **look like, behave like and feel and smell like a native S60 application.**



Lists and Grids

- S60 applications use a wide range of lists, and in some applications grids are also useful. Qt has widgets following widgets, which can be modified with variables to correspond to some S60 lists.
 - QListWidget,
 - QListView,
 - QTreeView,
 - QTableView
- A selection list can be created with QListView by defining SelectionMode (for example, SingleSelection, ContiguousSelection, NoSelection).
- **Grids** can be created with QListWidget using the property values:
 - ViewMode = IconMode
 - Flow = LeftToRight
 - Movement = Static



Menus

- Menus can be defined with Qt's **QMenu** class. Creating menus consists of creating and setting up **actions**, and creating menus and populating them with the created actions. Actions can be created in the `createActions()` function

```
void MenuWidget::createActions()  
{  
    openAction = new QAction(tr("Open"), this);  
    connect(openAction, SIGNAL(triggered()), this, SLOT(open()));  
    saveAction = new QAction(tr("Save"), this);  
    connect(saveAction, SIGNAL(triggered()), this, SLOT(save()));  
}
```

- Created actions are connected to private slots through the `triggered()` signal in the above example. The action can also be defined so that it shows a submenu. After the needed actions have been created, the menu itself can be built in the `createMenus()` function





```
void MenuWidget::createMenus()  
{  
    menuBar = new QMenuBar;  
    menuBar->addAction(openAction);  
    menuBar->addAction(saveAction);  
}
```

- The menu created in the above examples would result an **'Options' menu with 'Open' and 'Save' actions.**

Company Confidential

Notifications

- QMessageBox is a dialogue that can be used for different notification types. The dialogue can have a short message, an icon, and defined softkeys. QMessageBox has icons for four severity levels. Correspondence between these severity levels and S60 notes and notifications is shown below.

S60 component	QMessageBox	Icon
Information note	QMessageBox::information()	
Confirmation note	QMessageBox::question()	
Warning note	QMessageBox::warning()	
Error note	QMessageBox::critical()	

- For example, an error message with title 'Error', text 'Unable to send message', and 'OK' as the default button can be displayed in the following way:

```
QMessageBox::warning(this, tr("Error"), tr("Unable to send message"));
```

- Progress notes can be presented using QProgressDialog



- `QInputDialog` can be used to ask the user for input data. The same dialogue can be used for querying passwords by changing the line edit to show the contents as characters as they are entered but otherwise display them as asterisks (`QLineEdit::PasswordEchoOnEdit`).
- A query with selection list can be implemented with input dialogue, in which case the string list of possible selections is presented with combo boxes.



Views in Qt applications

- One possibility for **implementing S60-like views** in Qt applications is to use **QStackedWidget**. The stacked widget is populated by adding child widgets (views) to an internal list. The visible view is set based on the index of the widget in the list:

```
MessageWidget *messageWidget = new MessageWidget;  
ContactsWidget *contactsWidget = new ContactsWidget;  
  
stackedWidget = new QStackedWidget;  
stackedWidget->addWidget(messageWidget);  
stackedWidget->addWidget(contactsWidget);  
  
stackedWidget->setCurrentIndex(0); // messageWidget
```

- By implementing **menus to each child widget in the stacked widget**, the **views in the application can each have their own menu**.



Contents

1. Qt for S60 developer offering
2. Environment setup
3. Feedback channels
4. Introducing Qt on S60
5. Build Process
6. Symbian Concepts
7. UI Building Blocks
8. Tips & Tricks



Some Tips & Tricks

1. Fine tuning the UI to be shown in full screen
 2. Handling images
 3. Conversions
 - Strings
 - Buffers
 - Images
- More tips & tricks in
<http://wiki.forum.nokia.com/index.php/Category:Qt> for S60 and
<http://www.qtsoftware.com/developer/faqs>



Tuning the UI

- In most cases it is enough to show the Qt window in maximized mode,
 - Orientation changes are handled automatically
 - If you need to react to layout changes implement f.ex

```
QWidget::resizeEvent ( QResizeEvent * event )
```

```
int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    SensorTest w;
    //w.show();
    w.showMaximized(); //usually better option in Qt for S60 apps
    return a.exec();
}
```



Handling images

- Create a qrc file

```
<RCC>
  <qresource prefix="/" >
    <file>cheese.jpg</file> //points to the image file
  </qresource>
</RCC>
```

- Add the resource file to the project file (.pro)

```
RESOURCES += mice.qrc
```

- Load and draw the image

```
void SensorTest::paintEvent(QPaintEvent *event)
{
    QPainter painter ( this );

    QImage img;
    if( img.load(":/cheese.jpg") == false)
        painter.drawText( QPoint(10,120), "IMAGE ERROR" );
    else
        painter.drawImage( QPoint (50,50), img);
}
```

- More about Qt resource system <http://doc.trolltech.com/4.5/resources.html>

Company Confidential



Conversions

- Converting HBufC to QString

```
_LIT(KMsg, "Hello");  
HBufC* buf = KMsg().Alloc();  
QString qString((QChar*)buf->Des().Ptr(), buf->Length());
```

- Conversion convenience functions in Utils mobile extension

```
class XQConversions  
{  
public:  
    static QString s60DescToQString(const TDesc& desc);  
    static HBufC* qStringToS60Desc(const QString& string);  
    static QString s60Desc8ToQString(const TDesc8& desc);  
    static HBufC8* qStringToS60Desc8(const QString& string);  
    static QByteArray s60Desc8ToByteArray(const TDesc8& desc);  
    static HBufC8* qByteArrayToS60Desc8(const QByteArray& string);  
};
```

- More conversions in Qt for S60 developer's library

[http://library.forum.nokia.com/topic/Qt for S60 Developers Library/GUID-36C46C6D-05B2-41F0-AD8B-33D221C4496C.html](http://library.forum.nokia.com/topic/Qt%20for%20S60%20Developers%20Library/GUID-36C46C6D-05B2-41F0-AD8B-33D221C4496C.html)



Conversions QString \leftrightarrow Descriptors

- Qt uses the QString class, which is a Unicode character string. This class is used throughout the Qt APIs.
- In addition to QString, Qt has a QByteArray class that provides an array of bytes that can also be used for string operations.
- **Strings used in Qt must be converted to descriptors**, for example, **when using Symbian APIs**, which need them as parameters. Also, descriptors returned by Symbian APIs need to be formatted so that Qt can utilise them.
- A QString can be placed into a pointer descriptor or buffer to make it suitable for an interface that requires a 16-bit descriptor as a parameter:

```
QString text = "Some text";  
QTPtrC16 textPtr(reinterpret_cast<const TUint16*>(text.utf16())); // or  
TBuf<KBufLength> buffer(text.utf16());
```

- For more details and examples: see [http://library.forum.nokia.com/index.jsp?topic=/Qt for S60 Developers Library/GUID-97A34C6E-9F2B-4743-86F8-563362F7047A cover.html](http://library.forum.nokia.com/index.jsp?topic=/Qt%20for%20S60%20Developers%20Library/GUID-97A34C6E-9F2B-4743-86F8-563362F7047A%20cover.html) and converting descriptors



Image conversions

- Converting CFbsBitmap (f.ex an image captured with S60 camera APIs) to QImage (example from camera mobile extension):

```
void XQCameraPrivate::MceoViewFinderFrameReady(CFbsBitmap& aFrame)
{
    int bytesPerLine = aFrame.ScanLineLength(iViewFinderSize.width(),
        aFrame.DisplayMode());

    QImage image((uchar *)aFrame.DataAddress(), iViewFinderSize.width(),
        iViewFinderSize.height(), bytesPerLine, QImage::Format_RGB32);

    //iVFProcessor->ViewFinderFrameReady(image);
}
```



Contents

1. Qt for S60 developer offering
2. Environment setup
3. Feedback channels
4. Introducing Qt on S60
5. Build Process
6. Symbian Concepts
7. UI Building Blocks
8. Tips & Tricks



Summary

- Nokia is making **developer's life easier** by bringing Qt to S60
 - Productive, cross-platform, you most likely already know the benefits...
- As the developer **offering is still evolving** you have a chance to affect the outcome!
 - Be active and let us know your comments, ideas and findings!
- Qt for S60 development can be done **without purchasing anything** – all the tools are offered free of charge
- **Platform security** applies to Qt applications on S60
 - Security always brings in some restrictions...
- Qt doesn't have mobile features but as it is NOT a runtime you can use all the native (Symbian) APIs to access mobile features
 - Qt for S60 Mobile Extensions (which is the topic of the next session) starting to fill the gap to mobile space



More information

- Qt
 - <http://trolltech.com/developer>
 - Check out also the Qt Creator tool : <http://trolltech.com/about/news/qt-creator-ide-preview-released>
- Qt for S60
 - [http://www.forum.nokia.com/Resources and Information/Tools/Runtimes/Qt for S60](http://www.forum.nokia.com/Resources_and_Information/Tools/Runtimes/Qt_for_S60)
- Qt for maemo
 - <http://qt4.garage.maemo.org/>
 - Forum Nokia community port, not "officially supported"
- Carbide
 - <http://www.forum.nokia.com/info/sw.nokia.com/id/dbb8841d-832c-43a6-be13-f78119a2b4cb.html>
 - <http://www.forum.nokia.com/Events/Webinars.xhtml>
- S60
 - [http://www.forum.nokia.com/Resources and Information/Explore/Runtime Platforms/Symbian C++/QuickStart.xml](http://www.forum.nokia.com/Resources_and_Information/Explore/Runtime_Platforms/Symbian_C++/QuickStart.xml)
 - Devices http://www.forum.nokia.com/devices/matrix_all_1.html
- www.forum.nokia.com



Code less.
Create more.
Deploy everywhere.

Questions before moving to the Mobile Extension?

Company Confidential

75 © 2008 Nokia V1-Filename.ppt / YYYY-MM-DD / Initials

NOKIA