The Qt Company

WIDGET UI AND APPLICATION ENGINE WITH QT

021-002

Exam Curriculum

The Qt Company provides Qt and QML developers with three kinds of certification exams:

- Qt and QML Essentials
- Widget UI and Application Engine with Qt
- Qt Quick UI

Exam candidates may receive Certified Qt and QML Developer, Certified Qt C++ Specialist, and Certified Qt Quick Specialist certificates by passing the exams. A certificate is a valuable document, endorsing one's Qt and QML knowledge to an employer or a customer.

To achieve the Qt and QML Developer status, an exam candidate is required to pass the *Qt and QML Essentials* exam. The Qt C++ or Qt Quick Specialist status is granted to candidates, who additionally pass either or both *Widget UI and Application Engine* and *Qt Quick UI* exams. The former specialist exam tests candidates' knowledge of Qt C++ APIs, including, e.g., widgets, threads, model/view framework, and *QObject*. The latter exam tests Qt Quick and QML knowledge.

The exams can be taken in any order, but the candidate cannot receive either of the specialist certificates before the *Qt and QML Essentials* exam has been passed as well. So the Qt and QML Developer certificate is required for both specialist certificates as well.

Certificate exams can be taken in any authorized PearsonVUE test center. The details of the test center locations and instructions how to make an appointment and attend the exam can be found at http://www.pearsonvue.com/qtcompany/. The exam price varies from test center to test center. The exact price can be inquired directly from test centers.

Widget UI and Application Engine exam will test candidates' knowledge of UI creation with widgets as well as Qt C++ modules, APIs, and patterns. The exam curriculum is defined in detail in this document. The exam contains 30 multi choice questions and the passing score is 16/30. The candidate has 60 minutes to select the correct statement(s). No coding is required in the exam, although the questions and question options may contain short code snippets. In the unfortunate case that the exam candidate did not pass the exam, PearsonVUE (not the Qt company) is granting discount of the exam re-take. The amount depends on the country, where the exam is taken. Pearson VUE has divided countries into three regions, where the discounts are 30€, 74€ or 100€ accordingly. The discount is given as a voucher, given by PearsonVUE. To get the voucher, please contact PearsonVUE customer service. To identify yourself, you need to provide your Pearson Qt registration number: NQTxxxxxx. Note also that you CANNOT get discount only for that test center, where you took the original exam.

1 QT SW PROJECTS

- Know how to organize the source tree of a large project
- Know that Qt can be used with other libraries like STL, boost, ACE

1.1 QT TEST

- Know how to write and compile unit tests

- Test macros
- Asynchronous tests
- Auto tests

1.2 QT LIBRARIES AND PLUGINS

- Know how to create custom libraries
- Understand private implementation pattern and how to use it for binary compatibility
- Know how to write Qt plugins using low- and high-level APIs
- Know how to deploy and load/unload libraries and plugins using *QLibrary* and *QPluginLoader*

2 *WIDGETS AND PAINTING*

- Concept of widget
- Top-level widgets
- Window and widget flags and attributes (no need to remember each flag or attribute by heart)

2.1 DIALOGS

- Various standard dialog types
- The usage of standard dialogs
- Event processing with *QProgressDialog*
- Learn how to subclass a *QDialog*
- Be able to launch dialogs (not-) modal
- Know the deletion options for dialogs
- Be able to set input masks on line edits
- Understand the format for input masks
- Know that *QCompleter* offer popup completion

2.2 *WRITING CODE FOR EFFICIENT INTERNATIONALIZATION*

- Know how a text constant containing non-Latin characters can be read into a *QString*
- Know how to wrap strings which need to be translated
- Be able to extract strings to be translated from a Qt project
- Know how to take the encoding of the source file into account
- Know how to use Qt Linguist to translate extracted strings
- Be able to save the translated strings in a *.qm* file
- Know how to find all string constants which are not marked for translation
- Know how to use locale-specific icons
- Know how dates and currencies can be translated

2.3 *CUSTOM WIDGETS*

- Know how custom painted widgets can be made style aware
- Be able to design custom widgets in different ways: aggregation, sub-classing, custom painting
- Subclass *QWidget*, subclass *QxxxWidget*, aggregate widgets

- Know when it is better to write a custom widget
- Be able to develop a custom painted widget using *QPainter* (OpenGL language itself is not required)

2.4 PAINTING

- Know *QPainter* paint operations and their performance in the raster paint engine
- Understand the difference between *QWindow* and *QWidget* painting
- Backing store
- Paint devices

2.5 STYLING

- Know how widgets are styled
- Know how to use a style sheet

3 *MODEL/VIEW FRAMEWORK*

- Know what the purpose of model/view programming is
- Know how a view communicates with the model
- Know what widgets Qt provides to display model data

3.1 MODELS

- Be able to use Qt's predefined models
- Know why a view calls a model several times to render one single cell
- Know how to locate a specific item in a model
- Be able to write custom models
- Understand different ways to optimize custom model usage (simplified data structure, lazy loading)
- Know how to work with a selection
- Know how the view recognizes the changes that have been made to the model's data
- Drag and drop
- Sorting and filtering
- Proxy models

3.2 VIEWS

- Essential view classes
- Know what the purpose of a delegate is and how it can be used
- Know when to use roles and delegates
- Be able to write custom delegates

4 *PROCESSES AND THREADS*

4.1 INTER-PROCESS COMMUNICATION

- Process creation and startup
- Using standard input, output, and error

- Shared memory
- Qt DBus basics

4.2 MULTITASKING

- Thread affinity
- Thread creation, interruption, and cleanup
- Event handling in threads
- Inter-thread communication
- Thread storage
- Mutual exclusion and synchronization
- Re-entrant and thread safe classes

4.3 RUNNABLES

- Understand the concept of a thread pool
- Queuing and executing runnables
- Thread reserve and release

4.4 QT CONCURRENT

- Concept of the future
- Mapping and filtering
- Result reduction

5 NETWORKING

- Sockets and secure sockets
- Web sockets
- Understand network access manager
- Know how to make network requests and handle the replies properly
- Know how Qt manages cookies
- Be able to set network proxies

5.1 QT WEBENGINE

- *WebEngine* essential C++ classes

QWebEngineView

- Asynchronous functions
- Exposing *QObjects* to the script engine

6 DATA I/O

6.1 DATABASES

- Database connection
- Know how to make queries and transactions efficiently

- Know how to use database model classes
- Principles of database drivers

6.2 XML

- XML stream readers and writers
- Know how to use *XQuery* and *XPath* – the syntax is not required in the exam
- XML schema

6.3 JSON

- JSON documents
- Parsing JSON
- Generating JSON data

7 MISCELLANEOUS TOPICS

7.1 DESKTOP INTEGRATION

- Be able to use the *QDesktopWidget*
- Know how to put an icon into the system tray
- Know how to open an URL using the default browser

7.2 HELP SYSTEM

- Adding "Help" functionality to widgets
- Dynamic help
- *QTextBrowser* as a help viewer
- Know the different possibilities to provide help
- Understand the concept of the system tray icon
- Learn how to support dynamic help