Introduction

This is the reference guide for PyQt5 v5.15. PyQt5 is a set of Python bindings for v5 of the Qt application framework from The Qt Company.

Qt is a set of C++ libraries and development tools that includes platform independent abstractions for graphical user interfaces, networking, threads, regular expressions, SQL databases, SVG, OpenGL, XML, user and application settings, positioning and location services, short range communications (NFC and Bluetooth), web browsing, 3D animation, charts, 3D data visualisation and interfacing with app stores. PyQt5 implements over 1000 of these classes as a set of Python modules.

PyQt5 comprises PyQt5 itself and a number of add-ons that correspond to Qt's additional libraries. Each is provided as a source distribution (*sdist*) and binary wheels for Windows, Linux and macOS.

PyQt5 supports the Windows, Linux, UNIX, Android, macOS and iOS platforms and requires Python v3.5 or later. (PyQt5 should also build against Python v2.7 and earlier versions of Python v3 using the legacy **configure.py** build script but this is unsupported.)

The homepage for PyQt5 is https://www.riverbankcomputing.com/software/pyqt/. Here you will always find the latest stable version, current development snapshots.

License

PyQt5 is dual licensed on all platforms under the Riverbank Commercial License and the GPL v3. Your PyQt5 license must be compatible with your Qt license. If you use the GPL version then your own code must also use a compatible license.

PyQt5, unlike Qt, is not available under the LGPL.

You can purchase a commercial PyQt5 license here.

PyQt5 Components

PyQt5 comprises a number of different components. First of all there are a number of Python extension modules. These are all installed in the `PyQt5` Python package and are described in the list of modules.

Each extension module has a corresponding PEP 484 defined stub file containing type hints for the module's API. This can be used by static type checkers such as mypy.

PyQt5 contains plugins that enable Qt Designer and **qmlscene** to be extended using Python code. See Writing Qt Designer Plugins and Integrating Python and QML respectively for the details.

PyQt5 also contains a number of utility programs.

- **pyuic5** corresponds to the Qt **uic** utility. It converts QtWidgets based GUIs created using Qt Designer to Python code.
- **pyrcc5** corresponds to the Qt **rcc** utility. It embeds arbitrary resources (eg. icons, images, translation files) described by a resource collection file in a Python module.
- **pylupdate5** corresponds to the Qt **lupdate** utility. It extracts all of the translatable strings from Python code and creates or updates `.ts` translation files. These are then used by Qt Linguist to manage the translation of those strings.

The DBus support module is installed as dbus.mainloop.pyqt5. This module provides support for the Qt event loop in the same way that the dbus.mainloop.glib included with the standard `dbus-python` bindings package provides support for the GLib event loop. The API is described in DBus Support. It is only available if the `dbus-python` v0.80 (or later) bindings package is installed. The QtDBus module provides a more Qt-like interface to DBus.

PyQt5 includes a large number of examples. These are ports to Python of many of the C++ examples provided with Qt. They can be found in the `examples` directory of the sdist.

Finally, PyQt5 contains the specification files that allow bindings for other Qt based class libraries that further extend PyQt5 to be developed and installed.