Introduction

This book teaches how to write GUI applications using the Python programming language and the Qt application development framework. The only essential prior knowledge is that you can program in *some* object-oriented programming language, such as C++, C#, Java, or of course, Python itself. For the rich text chapter, some familiarity with HTML and with regular expressions is assumed, and the databases and threading chapters assume some basic knowledge of those topics. A knowledge of GUI programming is not required, since all the key concepts are covered.

The book will be useful to people who program professionally as part of their job, whether as full-time software developers, or those from other disciplines, including scientists and engineers, who need to do some programming in support of their work. It is also suitable for undergraduate and post-graduate students, particularly those doing courses or research that includes a substantial computing element. The exercises (with solutions) are provided especially to help students.

Python is probably the easiest to learn and nicest scripting language in widespread use, and Qt is probably the best library for developing GUI applications. The combination of Python and Qt, "PyQt", makes it possible to develop applications on any supported platform and run them unchanged on all the supported platforms—for example, all modern versions of Windows, Linux, Mac OS X, and most Unix-based systems. No compilation is required thanks to Python being interpreted, and no source code changes to adapt to different operating systems are required thanks to Qt abstracting away the platform-specific details. We only have to copy the source file or files to a target machine that has both Python and PyQt installed and the application will run.

If you are new to Python: Welcome! You are about to discover a language that is clear to read and write, and that is concise without being cryptic. Python supports many programming paradigms, but because our focus is on GUI programming, we will take an object-oriented approach everywhere except in the very early chapters.

Python is a very expressive language, which means that we can usually write far fewer lines of Python code than would be required for an equivalent application written in, say, C++ or Java. This makes it possible to show some small but complete examples throughout the text, and makes PyQt an ideal tool for rapidly and easily developing GUI applications, whether for prototyping or for production use.

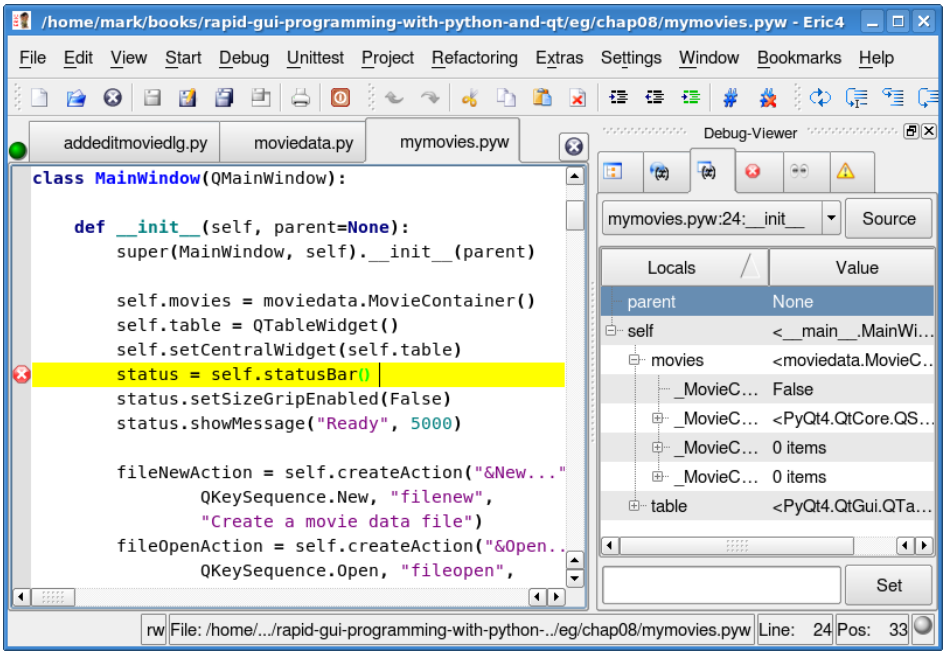

Figure 1 *The Eric4 IDE—a PyQt4 application*

Since the emphasis of the book is on GUI programming, Part I provides a fast-paced Python tutorial as well as some PyQt coverage. This material is clearly marked (just like this paragraph, with "Qt" in the margin) to make it easy for experienced Python programmers to skip the Python they already know. Parts II, III, and IV of the book are all PyQt-specific and assume that readers can already program in Python, whether from previous experience or from reading Part I.

Qt

Quite often in programming we reach decision points when there are several possible approaches we could take. Reference books and the online documentation identify what classes, methods, and functions are available, and in some cases provide examples, but such documents rarely provide a broader context. This book gives the necessary context, highlighting the key decision points for GUI programming and offering insights into the pros and cons so that you can decide for yourself what the right policy is for your particular circumstances. For example, when you create a dialog, should it be modal or modeless? (See Chapter 5 for an explanation and policy recommendations on this issue.)

PyQt is used to write all kinds of GUI applications, from accounting applications, to visualization tools used by scientists and engineers. Figure 1, for example, shows Eric4, a powerful integrated development environment that is written in PyQt. It is possible to write PyQt applications that are just tens of lines long, and medium-size projects of 1 000 to 10 000 lines are very common. Some commercial companies have built 100 000-line PyQt applications, with

programming teams varying in size from just one person to more than a dozen people. Many in-house tools are written using PyQt, but because these are often used to gain competitive advantage, the companies involved generally do not permit their use of PyQt to be made public. PyQt is also widely used in the open source world, with games, utilities, visualization tools, and IDEs all written using it.

This book is specifically about PyQt4, the Python bindings for the Qt 4 C++ application development framework.* PyQt4 is provided in the form of ten Python modules which between them contain around 400 classes and about 6 000 methods and functions. All the example programs have been tested on Windows, Linux, and Mac OS X, using Python 2.5, Qt 4.2, and PyQt 4.2, and additionally on Windows and Linux using Qt 4.3 and PyQt 4.3. Backporting to earlier versions is possible in some cases, but we recommend using the most up-to-date versions of Python, Qt, and PyQt.

Python, PyQt, and Qt can be used free of charge for noncommercial purposes, but the license used by Python is different from that used by PyQt and Qt. Python is available with a very liberal license that allows it to be used to develop both commercial and noncommercial applications. Both PyQt and Qt are dual-licensed: This essentially allows them to be used to develop noncommercial applications—which must in turn be licensed using an acceptable open source license such as the GNU General Public License (GPL); or to be used to develop commercial applications—in this case, a commercial PyQt license *and* a commercial Qt license must be purchased.

The Structure of the Book

The book is divided into four parts. Part I is primarily a rapid conversion course aimed at non-Python programmers who are familiar with an object-oriented language, although it also has some (clearly marked) PyQt content. Because the core Python language is mostly simple and is quite small, these chapters can teach the basics of Python to a sufficient extent that real Python applications can be written.

If you think that you can pick up the Python syntax simply through reading it, you might be tempted to skip Part I and dive straight into the GUI programming that begins in Part II. The early chapters in Part II include backreferences to the relevant pages in Part I to support readers who choose this approach. However, even for readers familiar with Python, we recommend reading about `QString` in Chapter 1. If you are unfamiliar with partial function application (currying), it is important to read the subsection that covers this in Chapter 2, since this technique is sometimes used in GUI programming.

*There are also Python bindings for the older Qt 3 library, but there is no reason to use that library for new projects, especially since Qt 4 offers far more functionality and is easier to use.

Part II begins by showing three tiny PyQt GUI applications to give an initial impression of what PyQt programming is like. It also explains some of the fundamental concepts involved in GUI programming, including PyQt's high-level signals and slots communication mechanism. Chapter 5 shows how to create dialogs and how to create and lay out widgets ("controls" in Windows-speak—the graphical elements that make up a user interface such as buttons, listboxes, and such) in a dialog. Dialogs are central to GUI programming: Most GUI applications have a single main window, and dozens or scores of dialogs, so this topic is covered in depth.

After the dialogs chapter comes Chapter 6, which covers main windows, including menus, toolbars, dock windows, and keyboard shortcuts, as well as loading and saving application settings. Part II's final chapters show how to create dialogs using *Qt Designer*, Qt's visual design tool, and how to save data in binary, text, and XML formats.

Part III gives deeper coverage of some of the topics covered in Part II, and introduces many new topics. Chapter 9 shows how to lay out widgets in quite sophisticated ways, and how to handle multiple documents. Chapter 10 covers low-level event handlers, and how to use the clipboard as well as drag and drop, text, HTML, and binary data. Chapter 11 shows how to modify and subclass existing widgets, and how to create entirely new widgets from scratch, with complete control over their appearance and behavior. This chapter also shows how to do basic graphics. Chapter 12 shows how to use Qt 4.2's new graphics view architecture, which is particularly suited to handling large numbers of independent graphical objects. Qt's HTML-capable rich text engine is covered in Chapter 13. This chapter also covers printing both to paper and to PDF files.

Part III concludes with two chapters on model/view programming: Chapter 14 introduces the subject and shows how to use Qt's built-in views and how to create custom data models and custom delegates, and Chapter 15 shows how to use the model/view architecture to perform database programming.

Part IV continues the model/view theme, with coverage of three different advanced model/view topics in Chapter 16. The first section of Chapter 17 describes the techniques that can be used for providing online help, and the second section explains how to internationalize an application, including how to use Qt's translation tools to create translation files. The Python standard library provides its own classes for networking and for threading, but in the last two chapters of Part IV we show how to do networking and threading using PyQt's classes.

Appendix A explains where Python, PyQt, and Qt can be obtained, and how to install them on Windows, Mac OS X, and Linux. PyQt is much easier to learn if you install it and try out some of the exercises, and if you inspect some of the example code. Appendix B presents screenshots and brief descriptions of selected PyQt widgets; this is helpful for those new to GUI programming. Appendix C presents diagrams of some of PyQt's key class hierarchies; this

is useful for getting to know what classes PyQt has to offer and how they are related.

If you have never used Python before, you should begin by reading Chapters 1–6 in order. If you already know Python, at least read the string policy (in bullet points on page 28), and skim the material in Chapter 2 (apart from the first section, which you'll know well). Make sure that you are comfortable with `lambda` and partial function application, both of which are covered in Chapter 2. It is probably also worth skimming Chapter 3 as well. Then read Chapters 4, 5, and 6 in order.

Once you have covered the first six chapters, you have covered the essentials of Python and the fundamentals of PyQt.

Chapter 7 is useful if you want to know how to create dialogs using a visual design tool rather than purely by hand coding, something that can save a lot of time. For file handling, at least read the first three sections of Chapter 8. If you plan to write and read text files, also read Chapter 8's fourth section, and similarly the fifth section if you are going to use XML files.

For Part III, at the least read Chapter 10's first section, on event handling, and all of Chapter 11. Chapter 12 and the first section of Chapter 13 assume that you have read about PyQt's event handling, and that you have read Chapter 11. Chapters 9 and 14 can be read stand-alone in this part, but Chapter 15 assumes that you have read Chapter 14.

In Part IV, Chapter 16 assumes that you have read Chapters 14 and 15, but the other chapters can be read independently.

If you find errors in the text or in the examples, or have other comments, please write to `mark@qtrac.eu` quoting "PyQt book" in the subject line. The book's home page, where any corrections will be published, and from where the examples and exercise solutions can be downloaded, is `http://www.qtrac.eu/pyqtbook.html`.

If you want to participate in the PyQt community, it is worthwhile joining the mailing list. Go to `http://www.riverbankcomputing.com/mailman/listinfo/pyqt` to find a link to the archive, so that you can see what the mailing list is like, and also for a form for joining. Python also has mailing lists and other community activities. For these, go to `http://www.python.org/community`.

Acknowledgments

I have many people to thank, and I will begin with those who have been intimately involved with the book.

Jasmin Blanchette is a senior software developer at Trolltech, a Qt expert, and a fine editor and writer in his own right. I have cowritten two C++/Qt books with him. Jasmin has made a huge number of suggestions and criticisms that have immensely improved the quality of this book.

David Boddie, Trolltech's documentation manager, is an active PyQt open source developer who has made many contributions to PyQt itself. His input has helped ensure that I have covered everything necessary, and done so in a sensible order.

Richard Chamberlain is cofounder and chief technology officer of Jadu Ltd., a content management company. His feedback and insights have helped ensure that the book is as broadly accessible as possible. He has also helped refine and improve the code used in the examples and exercises.

Trenton Schulz is a Trolltech developer who has been a valuable reviewer of my previous books. For this book, he has brought his Python and Qt knowledge to bear, giving considerable feedback on the manuscript. Along with Richard, he also ensured that Mac OS X users were never forgotten. In addition, he spotted many subtle errors that I had missed.

Phil Thompson is PyQt's creator and maintainer. He has been supportive of the book from the start, even adding features and improvements to PyQt as a direct result of discussions we have had regarding the book. He has made numerous suggestions for the book's improvement, and corrected many mistakes and misunderstandings.

Special thanks to Samuel Rolland, who let me loose on his Mac laptop, to install PyQt, test the examples, and take screenshots.

Thanks are also due to Guido van Rossum, creator of Python, as well as to the wider Python community who have contributed so much to make Python, and especially its libraries, so useful and enjoyable to use.

Thanks also to Trolltech, for developing and maintaining Qt, and in particular to the Trolltech developers both past and present, many of whom I have had the pleasure of working with, and who ensure that Qt is the best cross-platform GUI development framework in existence.

Particular thanks to Jeff Kingston, creator of the Lout typesetting language. I use Lout for all my books and for most of my other writing projects. Over the years, Jeff has made many improvements and added numerous features to Lout in response to feedback from users, including many that I have asked for myself. Thanks also to James Cloos who created the condensed version of the DejaVu Sans Mono font (itself derived from Jim Lyles' Vera font) from which this book's monospaced font is derived.

The publisher, in the person of Editor-in-Chief Karen Gettman, was supportive of this book from the very beginning. And special thanks to my editor, Debra Williams-Cauley, for her support, and for making the entire process as smooth as possible. Thanks also to Lara Wysong who managed the production process so well, and to the proofreader, Audrey Doyle, who did such fine work.

Last but not least, I want to acknowledge my wife, Andrea. Her love, loyalty, and support always give me strength and hope.