About this book

R has a number of packages that provide a link between the R user and a graphical toolkit, such as `tcltk`, `RGtk2` and `qtbase`. In addition, an R user can interface with Java, Python or other external languages to provide access to graphical toolkits within those languages. This book is about writing graphical user interfaces (GUI) within R that do not rely on knowing an external programming language.

The R language, like its predecessor S, is designed for interactive use through a command line interface (CLI). However, the graphical user interface (GUI) has emerged as an effective alternative, depending on the specific task and the target user base. Currently, there is a range of graphical interfaces for R that are programmed within R. For example, several package authors have provided GUIs for their functions. Examples include `limmaGUI`, `caGUI`, `clustTool`, `Metabonomic`, and others. There are a few tools to automatically generate such GUIs, such as the `fgui` package and the `guiDlgFunction` function from the `svDialogs` package. Other authors have provided graphical interfaces to explore data sets, such as `ggobi`, `playwith`, `latticist` and `aplpack`. Still others have provided packages with GUIs aimed at allowing students to perform some simulation, e.g., `teachingDemos`. The `rattle` package provides an interface for several data mining operations. The `Rcmdr` package provides a menu- and dialog-driven interface to a wide range of R's functionality. There are several user-contributed plugins that extend the `Rcmdr`. Additionally, as R finds wider usage outside of academia, it is not uncommon for people who work in a team setting to desire an interface to their R code that allows non-R users access.

Most all of these examples are within the scope of this book. We set out to show that for many purposes adding a graphical interface to one's work is not terribly sophisticated nor time-consuming. This book does not attempt to cover the development of GUIs that require knowledge of another programming language, although several such projects exist. Many of these are general front-ends to R, such as the Java-based GUI `JGR`, the `rkWard` GUI for KDE, the `biocep` GUI written using Java and the `RServe` package, or even the Windows GUI that comes with R's Windows package. There are also several special purpose GUIs, like `iPlots`, which are largely implemented in Java, relying on `rJava`, a native interface between R and Java.

The bulk of this text covers four different packages for writing GUIs in R. The `gWidgets` package is covered first. This provides a common programming interface over several R packages that implement low-level, native interfaces to GUI toolkits. The `gWidgets` interface is much simpler – and less powerful – than the native toolkits, so is useful for a programmer who does not wish to invest too much time into perfecting a GUI. There are a few other packages that provide a high-level R interface to a toolkit such as `rpanel` or `svDialogs`, but we focus on this one.

1

The next three parts introduce the native interfaces upon which `gWidgets` is built. These offer fuller and more direct control of the underlying toolkit and thus are well suited the development of GUIs that require special features or performance characteristics. The first of these is the `RGtk2` package which provides a link between R and the cross-platform GTK+ library. GTK+ is mature, feature rich and leveraged by several widely used projects.

Another mature and feature-rich toolkit is Qt, an open-source C++ library from Nokia. The R package `qtbase` provides a native interface from R to Qt. As Qt is implemented in C++, it is designed around the ability to create classes that extend the Qt classes. `qtbase` supports this from within R, although such object oriented concepts may be unfamiliar to many R users.

Finally, we discuss the `tcltk` package, which provides the R user access to the Tk libraries. Although not as modern as GTK+ nor Qt, these libraries come pre-installed with the Windows binary, thus avoiding installation issues for the average end-user. The bindings to Tk were the first ones to appear for R and several of the GUI projects above, notably `Rcmdr`, use this toolkit.

These four main parts are preceeded by an introductory chapter on GUIs and followed by a chapter on web GUIs.

The text is written with the belief that much can be learned by studying examples, and so several examples are given. Some of these are meant as sketches of what can be done, while a few illustrate how to code actual useful interfaces. This text can't expect to cover all of the features of a graphical toolkit. For the `tcltk`, `RGtk2` and `qtbase` packages, the underlying toolkits have well documented APIs.

This text comes with an accompanying package `ProgGUIInR`. This package includes the complete code for all the examples. In order to save space, some examples in the text have code that is not shown. The package provides the functions `browsegWidgetsFiles`, `browseRGtk2Files`, `browseQtFiles` and `browseTclTkFiles` for browsing the examples from the respective chapters. Additionally, this package will contain vignettes describing aspects that did not make it into the text.

This text was written with the `Sweave` package. To suppress superflous output an assignment to a variable named `QT` is made at times.

2