Preface

Tcl stands for *Tool Command Language*. Tcl is really two things: a scripting language, and an interpreter for that language that is designed to be easy to embed into your application. Tcl and its associated graphical user-interface toolkit, Tk, were designed and crafted by Professor John Ousterhout of the University of California, Berkeley. You can find these packages on the Internet (as explained on page lii) and use them freely in your application, even if it is commercial. The Tcl interpreter has been ported from UNIX to DOS, Windows, OS/2, NT, and Macintosh environments. The Tk toolkit has been ported from the X window system to Windows and Macintosh.

I first heard about Tcl in 1988 while I was Ousterhout's Ph.D. student at Berkeley. We were designing a network operating system, Sprite. While the students hacked on a new kernel, John wrote a new editor and terminal emulator. He used Tcl as the command language for both tools so that users could define menus and otherwise customize those programs. This was in the days of X10, and he had plans for an X toolkit based on Tcl that would help programs cooperate with each other by communicating with Tcl commands. To me, this cooperation among tools was the essence of Tcl.

This early vision imagined that applications would be large bodies of compiled code and a small amount of Tcl used for configuration and high-level commands. John's editor, *mx*, and the terminal emulator, *tx*, followed this model. While this model remains valid, it has also turned out to be possible to write entire applications in Tcl. This is because the Tcl/Tk shell, *wish*, provides access to other programs, the file system, network sockets, plus the ability to create a graphical user interface. For better or worse, it is now common to find applications that contain thousands of lines of Tcl script.

This book was written because, while I found it enjoyable and productive to use Tcl and Tk, there were times when I was frustrated. In addition, working at Xerox PARC, with many experts in languages and systems, I was compelled to understand both the strengths and weaknesses of Tcl and Tk. Although many of my colleagues adopted Tcl and Tk for their projects, they were also just as quick to point out its flaws. In response, I have built up a set of programming techniques that exploit the power of Tcl and Tk while avoiding troublesome areas. This book is meant as a practical guide to help you get the most out of Tcl and Tk and avoid some of the frustrations I experienced.

It has been about 10 years since I was introduced to Tcl, and about five years since the first edition of this book. During the last several years I have been working under John Ousterhout, first at Sun Microsystems and now at Scriptics Corporation. I have managed to remain mostly a Tcl programmer while others in our group have delved into the C implementation of Tcl itself. I've been building applications like HTML editors, e-mail user interfaces, Web servers, and the customer database we run our business on. This experience is reflected in this book. The bulk of the book is about Tcl scripting, and the aspects of C programming to create Tcl extensions is given a lighter treatment. I have been lucky to remain involved in the core Tcl development, and I hope I can pass along the insights I have gained by working with Tcl.

Why Tcl?

As a scripting language, Tcl is similar to other UNIX shell languages such as the Bourne Shell (sh), the C Shell (csh), the Korn Shell (ksh), and Perl. Shell programs let you execute other programs. They provide enough programmability (variables, control flow, and procedures) to let you build complex scripts that assemble existing programs into a new tool tailored for your needs. Shells are wonderful for automating routine chores.

It is the ability to easily add a Tcl interpreter to your application that sets it apart from other shells. Tcl fills the role of an extension language that is used to configure and customize applications. There is no need to invent a command language for your new application, or struggle to provide some sort of user-programmability for your tool. Instead, by adding a Tcl interpreter, you structure your application as a set of primitive operations that can be composed by a script to best suit the needs of your users. It also allows other programs to have programmatic control over your application, leading to suites of applications that work well together.

The Tcl C library has clean interfaces and is simple to use. The library implements the basic interpreter and a set of core scripting commands that implement variables, flow control, and procedures (see page 22). There is also a set of commands that access operating system services to run other programs, access the file system, and use network sockets. Tk adds commands to create graphical user interfaces. Tcl and Tk provide a "virtual machine" that is portable across UNIX, Windows, and Macintosh environments.

The Tcl virtual machine is extensible because your application can define

new Tcl commands. These commands are associated with a C or C++ procedure that your application provides. The result is applications that are split into a set of primitives written in a compiled language and exported as Tcl commands. A Tcl script is used to compose the primitives into the overall application. The script layer has access to shell-like capability to run other programs, has access to the file system, and can call directly into the compiled part of the application through the Tcl commands you define. In addition, from the C programming level, you can call Tcl scripts, set and query Tcl variables, and even trace the execution of the Tcl interpreter.

There are many Tcl extensions freely available on the Internet. Most extensions include a C library that provides some new functionality, and a Tcl interface to the library. Examples include database access, telephone control, MIDI controller access, and *expect*, which adds Tcl commands to control interactive programs.

The most notable extension is Tk, a toolkit for graphical user interfaces. Tk defines Tcl commands that let you create and manipulate user interface widgets. The script-based approach to user interface programming has three benefits:

- Development is fast because of the rapid turnaround; there is no waiting for long compilations.
- The Tcl commands provide a higher-level interface than most standard C library user-interface toolkits. Simple user interfaces require just a handful of commands to define them. At the same time, it is possible to refine the user interface in order to get every detail just so. The fast turnaround aids the refinement process.
- The user interface can be factored out from the rest of your application. The developer can concentrate on the implementation of the application core and then fairly painlessly work up a user interface. The core set of Tk widgets is often sufficient for all your user interface needs. However, it is also possible to write custom Tk widgets in C, and again there are many contributed Tk widgets available on the network.

There are other choices for extension languages that include Visual Basic, Scheme, Elisp, Perl, Python, and Javascript. Your choice between them is partly a matter of taste. Tcl has simple constructs and looks somewhat like C. It is easy to add new Tcl primitives by writing C procedures. Tcl is very easy to learn, and I have heard many great stories of users completing impressive projects in a short amount of time (e.g., a few weeks), even though they never used Tcl before.

Java has exploded onto the computer scene since this book was first published. Java is a great systems programming language that in the long run could displace C and C++. This is fine for Tcl, which is designed to glue together building blocks written in any system programming language. Tcl was designed to work with C, but has been adapted to work with the Java Virtual Machine. Where I say “C or C++”, you can now say “C, C++, or Java,” but the details are a bit different with Java. This book does not describe the Tcl/Java interface, but you can find *TclBlend* on the CD-ROM. TclBlend loads the Java Virtual Machine

into your Tcl application and lets you invoke Java methods. It also lets you implement Tcl commands in Java instead of C or C++.

Javascript is a language from Netscape that is designed to script interactions with Web pages. Javascript is important because Netscape is widely deployed. However, Tcl provides a more general purpose scripting solution that can be used in a wide variety of applications. The Tcl/Tk Web browser plugin provides a way to run Tcl in your browser. It turns out to be more of a Java alternative than a JavaScript alternative. The plugin lets you run Tcl applications inside your browser, while JavaScript gives you fine grain control over the browser and HTML display. The plugin is described in Chapter 20.

Tcl and Tk Versions

Tcl and Tk continue to evolve. See `http://www.beedub.com/book/` for updates and news about the latest Tcl releases. Tcl and Tk have had separate version numbers for historical reasons, but they are released in pairs that work together. The original edition of this book was based on Tcl 7.4 and Tk 4.0, and there were a few references to features in Tk 3.6. This third edition has been updated to reflect new features added through Tcl/Tk 8.2:

- Tcl 7.5 and Tk 4.1 had their final release in May 1996. These releases feature the port of Tk to the Windows and Macintosh environments. The Safe-Tcl security mechanism was introduced to support safe execution of network applets. There is also network socket support and a new Input/Output (I/O) subsystem to support high-performance event-driven I/O.
- Tcl 7.6 and Tk 4.2 had their final release in October 1996. These releases include improvements in Safe-Tcl, and improvements to the `grid` geometry manager introduced in Tk 4.1. Cross-platform support includes virtual events (e.g., `<<Copy>>` as opposed to `<Control-c>`), standard dialogs, and more file manipulation commands.
- Tcl 7.7 and Tk 4.3 were internal releases used for the development of the Tcl/Tk plug-in for the Netscape Navigator and Microsoft Internet Explorer Web browsers. Their development actually proceeded in parallel to Tcl 7.6 and Tk 4.2. The plug-in has been released for a wide variety of platforms, including Solaris/SPARC, Solaris/INTEL, SunOS, Linux, Digital UNIX, IRIX, HP/UX, Windows 95, Windows NT, and the Macintosh. The browser plug-in supports Tcl applets in Web pages and uses the sophisticated security mechanism of Safe-Tcl to provide safety.
- Tcl 8.0 features an on-the-fly compiler for Tcl that provides many-times faster Tcl scripts. Tcl 8.0 supports strings with embedded null characters. The compiler is transparent to Tcl scripts, but extension writers need to learn some new C APIs to take advantage of its potential. The release history of 8.0 spread out over a couple of years as John Ousterhout moved from Sun Microsystems to Scriptics Corporation. The widely used 8.0p2 release was made in the fall of 1997, but the final patch release, 8.0.5, was made in the spring of 1999.

- Tk changed its version to match Tcl at 8.0. Tk 8.0 includes a new platform-independent font mechanism, native menus and menu bars, and more native widgets for better native look and feel on Windows and Macintosh.
- Tcl/Tk 8.1 features full Unicode support, a new regular expression engine that provides all the features found in Perl 5, and thread safety so that you can embed Tcl into multithreaded applications. Tk does a heroic job of finding the correct font to display your Unicode characters, and it adds a message catalog facility so that you can write internationalized applications. The release history of Tcl/Tk 8.1 also straddled the Sun to Scriptics transition. The first alpha release was made in the fall of 1997, and the final patch release, 8.1.1, was made in May 1999.
- Tcl/Tk 8.2 is primarily a bug fix and stabilization release. There are a few minor additions to the Tcl C library APIs to support more extensions without requiring core patches. Tcl/Tk 8.2 went rapidly into final release in the summer of 1999.

Who Should Read This Book

This book is meant to be useful to the beginner in Tcl as well as the expert. For the beginner and expert alike, I recommend careful study of Chapter 1, *Tcl Fundamentals*. The programming model of Tcl is designed to be simple, but it is different from many programming languages. The model is based on string substitutions, and it is important that you understand it properly to avoid trouble in complex cases. The remainder of the book consists of examples that demonstrate how to use Tcl and Tk productively. For your reference, each chapter has tables that summarize the Tcl commands and Tk widgets they describe.

This book assumes that you have some programming experience, although you should be able to get by even if you are a complete novice. Knowledge of UNIX shell programming will help, but it is not required. Where aspects of window systems are relevant, I provide some background information. Chapter 2 describes the details of using Tcl and Tk on UNIX, Windows, and Macintosh.

How to Read This Book

This book is best used in a hands-on manner, trying the examples at the computer. The book tries to fill the gap between the terse Tcl and Tk manual pages, which are complete but lack context and examples, and existing Tcl programs that may or may not be documented or well written.

I recommend the on-line manual pages for the Tcl and Tk commands. They provide a detailed reference guide to each command. This book summarizes much of the information from the manual pages, but it does not provide the complete details, which can vary from release to release. HTML versions of the on-line manual pages can be found on the CD-ROM that comes with this book.

Other Tcl Books

This book was the second Tcl book after the original book by John Ousterhout, the creator of Tcl. Since then, the number of Tcl books has increased remarkably. The following are just some of the books currently available.

Tcl and the Tk Toolkit (Addison-Wesley, 1994) by John Ousterhout provides a broad overview of all aspects of Tcl and Tk, even though it covers only Tcl 7.3 and Tk 3.6. The book provides a more detailed treatment of C programming for Tcl extensions.

Exploring Expect (O'Reilly & Associates, Inc., 1995) by Don Libes is a great book about an extremely useful Tcl extension. *Expect* lets you automate the use of interactive programs like *ftp* and *telnet* that expect to interact with a user. By combining *expect* and Tk, you can create graphical user interfaces for old applications that you cannot modify directly.

Graphical Applications with Tcl & Tk (M&T Press, 1996) by Eric Johnson is oriented toward Windows users. The second edition is up-to-date with Tcl/Tk 8.0.

Tcl/Tk Tools (O'Reilly & Associates, Inc., 1997) by Mark Harrison describes many useful Tcl extensions. These include Oracle and Sybase interfaces, object-oriented language enhancements, additional Tk widgets, and much more. The chapters were contributed by the authors of the extensions, so they provide authoritative information on some excellent additions to the Tcl toolbox.

CGI Developers Resource, Web Programming with Tcl and Perl (Prentice Hall, 1997) by John Ivler presents Tcl-based solutions to programming Web sites.

Effective Tcl/Tk Programming (Addison Wesley, 1997) by Michael McLennan and Mark Harrison illustrate Tcl and Tk with examples and application design guidelines.

Interactive Web Applications with Tcl/Tk (AP Professional, 1998) by Michael Doyle and Hattie Schroeder describes Tcl programming in the context of the Web browser plugin.

Tcl/Tk for Programmers (IEEE Computer Society, 1998) by Adrian Zimmer describes Unix and Windows programming with Tcl/Tk. This book also includes solved exercises at the end of each chapter.

Tcl/Tk for Real Programmers (Academic Press, 1999) by Clif Flynt is another example-oriented book.

Tcl/Tk in a Nutshell (O'Reilly, 1999) by Paul Raines and Jeff Tranter is a handy reference guide. It covers several popular extensions including Expect, [incr Tcl], Tix, TclX, BLT, SybTcl, OraTcl, and TclODBC. There is a tiny pocket-reference guide for Tcl/Tk that may eliminate the need to thumb through my large book to find the syntax of a particular Tcl or Tk command.

Web Tcl Complete (McGraw Hill, 1999) by Steve Ball describes programming with the Tcl Web Server. It also covers Tcl/Java integration using *TclBlend*.

[incr Tcl] From The Ground Up (Osborn-McGraw Hill, 1999) by Chad Smith describes the [incr Tcl] object-oriented extension to Tcl.

On-line Examples

The book comes with a CD-ROM that has source code for all of the examples, plus a selection of Tcl freeware found on the Internet. The CD-ROM is created with the Linux *mkhybrid* program, so it is readable on UNIX, Windows, and Macintosh. There, you will find the versions of Tcl and Tk that were available as the book went to press. You can also retrieve the sources shown in the book from my personal Web site:

```
http://www.beedub.com/book/
```

Ftp Archives

The primary site for the Tcl and Tk distributions is given below as a Universal Resource Location (URL):

```
ftp://ftp.scriptics.com/pub/tcl
```

You can use FTP and log in to the host (e.g., `ftp.scriptics.com`) under the `anonymous` user name. Give your e-mail address as the password. The directory is in the URL after the host name (e.g., `/pub/tcl`). There are many sites that mirror this distribution. The mirror sites provide an archive site for contributed Tcl commands, Tk widgets, and applications. There is also a set of Frequently Asked Questions files. These are some of the sites that maintain Tcl archives:

```
ftp://ftp.neosoft.com/pub/tcl
ftp://ftp.syd.dit.csiro.au/pub/tk
ftp://ftp.ibp.fr/pub/tcl
ftp://src.doc.ic.ac.uk/packages/tcl/
ftp://ftp.luth.se/pub/unix/tcl/
ftp://sunsite.cnlab-switch.ch/mirror/tcl
ftp://ftp.sterling.com/programming/languages/tcl
ftp://ftp.sunet.se/pub/lang/tcl
ftp://ftp.cs.columbia.edu/archives/tcl
ftp://ftp.uni-paderborn.de/pub/unix/tcl
ftp://sunsite.unc.edu/pub/languages/tcl
ftp://ftp.funet.fi/pub/languages/tcl
```

You can use a World Wide Web browser like *Mosaic*, *Netscape*, *Internet Explorer*, or *Lynx* to access these sites. Enter the URL as specified above, and you are presented with a directory listing of that location. From there you can change directories and fetch files.

If you do not have direct FTP access, you can use an e-mail server for FTP. Send e-mail to `ftpmail@decwrl.dec.com` with the message `Help` to get directions. If you are on BITNET, send e-mail to `bitftp@pucc.princeton.edu`.

You can search for FTP sites that have Tcl by using the *Archie* service that indexes the contents of anonymous FTP servers. Information about using *Archie* can be obtained by sending mail to `archie@archie.sura.net` that contains the message `Help`.

World Wide Web

Start with these World Wide Web pages about Tcl:

```
http://www.scriptics.com/
http://www.sco.com/Technology/tcl/Tcl.html
http://www.purl.org/NET/Tcl-FAQ/
```

The home page for this book contains errata for all editions. This is the only URL I control personally, and I plan to keep it up-to-date indefinitely:

```
http://www.beedub.com/book/
```

The Prentice Hall Web site has information about the book, but you must use its search facility to find the exact location. Start at:

```
http://www.prenhall.com/
```

Newsgroups

The `comp.lang.tcl` newsgroup is very active. It provides a forum for questions and answers about Tcl. Announcements about Tcl extensions and applications are posted to the `comp.lang.tcl.announce` newsgroup.

Typographic Conventions

The more important examples are set apart with a title and horizontal rules, while others appear in-line. The examples use `courier` for Tcl and C code. When interesting results are returned by a Tcl command, those are presented below in *`oblique courier`*. The => is not part of the return value in the following example.

```
expr 5 + 8
=> 13
```

The `courier` font is also used when naming Tcl commands and C procedures within sentences.

The usage of a Tcl command is presented in the following example. The command name and constant keywords appear in `courier`. Variable values appear in *`courier oblique`*. Optional arguments are surrounded with question marks.

```
set varname ?value?
```

The name of a program is in italics:

xterm

Hot Tips

The icon in the margin marks a “hot tip” as judged by the reviewers of the book. The visual markers help you locate the more useful sections in the book. These are also listed in the index under Hot Tip.

Book Organization

The chapters of the book are divided into seven parts. The first part describes basic Tcl features. The first chapter describes the fundamental mechanisms that characterize the Tcl language. This is an important chapter that provides the basic grounding you will need to use Tcl effectively. Even if you have programmed in Tcl already, you should review Chapter 1. Chapter 2 goes over the details of using Tcl and Tk on UNIX, Windows, and Macintosh. Chapter 3 presents a sample application, a CGI script, that illustrates typical Tcl programming. The rest of Part I covers the basic Tcl commands in more detail, including string handling, data types, control flow, procedures, and scoping issues. Part I finishes with a description of the facilities for file I/O and running other programs.

Part II describes advanced Tcl programming. It starts with `eval`, which lets you generate Tcl programs on the fly. Regular expressions provide powerful string processing. If your data-processing application runs slowly, you can probably boost its performance significantly with the regular expression facilities. Namespaces partition the global scope of procedures and variables. Unicode and message catalogs support internationalized applications. Libraries and packages provide a way to organize your code for sharing among projects. The introspection facilities of Tcl tell you about the internal state of Tcl. Event driven I/O helps server applications manage several clients simultaneously. Network sockets are used to implement the HTTP protocol used to fetch pages on the World Wide Web. Safe-Tcl is used to provide a secure environment to execute applets downloaded over the network. *TclHttpd* is an extensible web server built in Tcl. You can build applications on top of this server, or embed it into your existing applications to give them a web interface.

Part III introduces Tk. It gives an overview of the toolkit facilities. A few complete examples are examined in detail to illustrate the features of Tk. Event bindings associate Tcl commands with events like keystrokes and button clicks. Part III ends with three chapters on the Tk geometry managers that provide powerful facilities for organizing your user interface.

Part IV describes the Tk widgets. These include buttons, menus, scrollbars, labels, text entries, multiline and multifont text areas, drawing canvases, listboxes, and scales. The Tk widgets are highly configurable and very programmable, but their default behaviors make them easy to use as well. The resource database that can configure widgets provides an easy way to control the overall look of your application.

Part V describes the rest of the Tk facilities. These include selections, keyboard focus, and standard dialogs. Fonts, colors, images, and other attributes that are common to the Tk widgets are described in detail. This part ends with a few larger Tk examples.

Part VI is an introduction to C programming and Tcl. The goal of this part is to get you started in the right direction when you need to extend Tcl with new commands written in C or integrate Tcl into custom applications.

Part VII provides a chapter for each of the Tcl/Tk releases covered by the

book. These chapters provide details about what features were changed and added. They also provide a quick reference if you need to update a program or start to use a new version.

What's New in the Third Edition

The third edition is up-to-date with Tcl/Tk 8.2. The main new Tcl/Tk features are Internationalization, which is covered in Chapter 15, a new regular expression engine, which is described in Chapter 11, and thread-safety. There is a new chapter about compiling C extensions, and there is a more complete C extension example. The chapters on Eval and the Web browser plugin received a thorough update. I made a light sweep through the remainder of the book correcting errors and improving examples. Perhaps the best addition for the reader is an all-new index.

My favorite addition to the book is Chapter 18 that describes *TclHttpd*, a Web server built in Tcl. *TclHttpd* provides a number of nice ways to integrate a Web server with a Tcl application, replacing the standard CGI interface with something that is much more flexible and efficient. I have been using this server for the last year to build `www.scriptics.com`. This freely available server has been used to build several other products, plus it provides an easy way for you to bring up your own Web server.

First Edition Thanks

I would like to thank my managers and colleagues at Xerox PARC for their patience with me as I worked on this book. The tips and tricks in this book came partly from my own work as I helped lab members use Tcl, and partly from them as they taught me. Dave Nichols' probing questions forced me to understand the basic mechanisms of the Tcl interpreter. Dan Swinehart and Lawrence Butcher kept me sharp with their own critiques. Ron Frederick and Berry Kerchival adopted Tk for their graphical interfaces and amazed me with their rapid results. Becky Burwell, Rich Gold, Carl Hauser, John Maxwell, Ken Pier, Marvin Theimer, and Mohan Vishwanath made use of my early drafts, and their questions pointed out large holes in the text. Karin Petersen, Bill Schilit, and Terri Watson kept life interesting by using Tcl in very nonstandard ways. I especially thank my managers, Mark Weiser and Doug Terry, for their understanding and support.

I thank John Ousterhout for Tcl and Tk, which are wonderful systems built with excellent craftsmanship. John was kind enough to provide me with an advance version of Tk 4.0 so that I could learn about its new features well before its first beta release.

Thanks to the Tcl programmers out on the Net, from whom I learned many tricks. John LoVerso and Stephen Uhler are the hottest Tcl programmers I know.

Many thanks to the patient reviewers of early drafts: Pierre David, Clif Flynt, Simon Kenyon, Eugene Lee, Don Libes, Lee Moore, Joe Moss, Hador Shemtov, Frank Stajano, Charles Thayer, and Jim Thornton.

Many folks contributed suggestions by e-mail: Miguel Angel, Stephen Bensen, Jeff Blaine, Tom Charnock, Brian Cooper, Patrick D'Cruze, Benoit Desrosiers, Ted Dunning, Mark Eichin, Paul Friberg, Carl Gauthier, David Gerdes, Klaus Hackenberg, Torkle Hasle, Marti Hearst, Jean-Pierre Herbert, Jamie Honan, Norman Klein, Joe Konstan, Susan Larson, Håkan Liljegren, Lionel Mallet, Dejan Milojicic, Greg Minshall, Bernd Mohr, Will Morse, Heiko Nardmann, Gerd Neugebauer, TV Raman, Cary Renzema, Rob Riepel, Dan Schenk, Jean-Guy Schneider, Elizabeth Scholl, Karl Schwamb, Rony Shapiro, Peter Simanyi, Vince Skahan, Bill Stumbo, Glen Vanderburg, Larry Virden, Reed Wade, and Jim Wight. Unfortunately, I could not respond to every suggestion, even some that were excellent.

Thanks to the editors and staff at Prentice Hall. Mark Taub has been very helpful as I progressed through my first book. Lynn Schneider and Kerry Reardon were excellent copy and production editors, respectively.

Second Edition Thanks

I get to thank John Ousterhout again, this time for supporting me as I worked in the Tcl/Tk group at Sun Microsystems. The rest of the group deserve a lot of credit for turning Tcl and Tk into a dynamite cross-platform solution. Scott Stanton led the Tk port to the PC. Ray Johnson led the Tk port to the Macintosh. Jacob Levy implemented the event-driven I/O system, Safe-Tcl, and the browser plug-in. Brian Lewis built the Tcl compiler. Ken Corey worked on Java integration and helped with the *SpecTcl* user interface builder. Syd Polk generalized the menu system to work with native widgets on the Macintosh and Windows. Colin Stevens generalized the font mechanism and worked on internationalization for Tk.

Stephen Uhler deserves special thanks for inspiring many of the cool examples I use in this book. He was the lead on the *SpecTcl* user interface builder. He built the core HTML display library on which I based an editor. We worked closely together on the first versions of *TclHttpd*. He taught me how to write compact, efficient Tcl code and to use regular expression substitutions in amazing ways. I hope he has learned at least a little from me.

Thanks again to Mark Taub, Eileen Clark, and Martha Williams at Prentice Hall. George Williams helped me assemble the files for the CD-ROM.

Third Edition Thanks

John Ousterhout continues his wonderful role as Tcl benefactor, now as founder of Scriptics Corporation. I'd like to thank every one of the great folks that I work with at Scriptics, especially the pioneering crew of Sarah Daniels, Scott Stanton, Ray Johnson, Bryan Surles, Melissa Hirschl, Lee Bernhard, Suresh Sastry, Emil Scaffon, Pat P., Scott Redman, and Berry Kercheval. The rest of the gang deserves a big thanks for making Scriptics such an enjoyable place to work. Jerry Peek, who is a notable author himself, provided valuable advice and wonderfully detailed comments! Ken Jones told me about a great

indexing tool.

I'd like to thank all the readers that drop me the encouraging note or probing question via e-mail. I am always interested in new and interesting uses of Tcl!

Thanks to the editors at Prentice Hall: Mark Taub, Joan McNamara, and Joan Eurell. Mark continues to encourage me to come out with new editions, and the Joans helped me complete this third edition on time.

Finally, I thank my wonderful wife Jody for her love, kindness, patience, wit, and understanding as I worked long hours. Happily, many of those hours were spent working from home. I now have three sons, Christopher, Daniel, and Michael, who get the credit for keeping me from degenerating into a complete nerd.

Contact the Author

I am always open to comments about this book. My e-mail address is `welch@acm.org`. It helps me sort through my mail if you put the word "book" or the title of the book into the e-mail subject line. Visit my Web site at:

```
http://www.beedub.com/
```

for current news about the book and my other interests.