Appendix A. Libraries and Extensions

The programming libraries available for a language are as important as the language itself. Here we list a tiny subset of those available for Tcl — the ones that are in the author's opinion commonly used or otherwise worthy of attention. There are several sites that have a more complete listing; the author's personal favorite is the Great Unified Tcl/Tk Extension Repository or GUTTER[1] [2] which lists and categorises libraries and extensions.

A.1. GUI toolkits

Package	Description
`Tk`	The most well known Tcl extension is of course the Tk graphical toolkit which is so closely associated with Tcl as to be referred collectively as Tcl/Tk. Tk's cross-platform portability, widget set and ease of use has made it the de facto graphical toolkit of choice not just for Tcl but also for other languages like Python, Ruby and Perl. Tk is generally included in all binary distribution of Tcl and available in source form from the same location[3] as Tcl. Most existing books on Tcl include Tk in their coverage and an excellent online book that includes the latest features is available from http://www.tkdocs.com.
`gnocl`	This is an alternative GUI toolkit based on GTK+/Gnome and comes with its own extensive set of widgets. This is however not widely used in the Tcl world and primarily meant for Unix systems. Available from http://www.gnocl.org.

A.2. Internet protocols

Package	Description
`tls`	Implements the SSL/TLS security protocols. Available from https://core.tcl.tk/tcltls/home.
`http`	Implements the client-side HTTP protocol. Available as part of the core Tcl distribution.
`rl_http`	An alternative implementation of the client-side HTTP protocol from RubyLane. Available from https://github.com/RubyLane/rl_http.
`Tnm`	Part of the Scotty network management suite, this package includes support for SNMP, ICMP and DNS protocols. Available from https://github.com/flightaware/scotty.
`tclcurl`	A wrapper around the well-known Curl[4] multiprotocol client library which supports a very wide range of Internet protocols including HTTP, FTP, Gopher, IMAP, LDAP, POP3, SMTP, Telnet and many others. Available from https://github.com/jdc8/tclcurl.
`WS::Server`, `WS::Client`	Server and client implementations of Web Services using Web Services over SOAP. Also supports JSON responses via REST. Available from http://core.tcl.tk/tclws/home.
Tcllib[5]	Contains modules for implementations of several network protocols like FTP, SMTP, NNTP and NTP. In some cases, the server side protocol is also supported.

[1] http://core.tcl.tk/jenglish/gutter/
[2] Yes, Tcl'ers often have a warped sense of humor.
[3] https://sourceforge.net/projects/tcl/files/Tcl/
[4] https://curl.haxx.se/
[5] http://core.tcl.tk/tcllib/doc/trunk/embedded/index.html

A.3. Web servers and frameworks

Package	Description
`tclhttpd`	A pure Tcl web server with a Tcl based templating system. Available from https://sourceforge.net/projects/tclhttpd.
Rivet	An Apache module for creating web applications. Available from https://tcl.apache.org/rivet.
Naviserver	Web server with dynamic pages in Tcl, and high end features like database connection pooling and multithreading. Available from https://sourceforge.net/projects/naviserver.
Wub	The web server that hosts Tcler's Wiki[6]. Available from https://code.google.com/archive/p/wub/.
Woof!	A Web framework similar to Rails for Ruby. Web server agnostic and can be used with Apache, IIS, Lighttpd etc.
Tclssg	This is a static site generator using Markdown for content and tools for website management. Available from https://github.com/tclssg/tclssg.

A.4. Numeric computing

Package	Description
Tcllib[7]	Tcllib[8] includes modules for several types of numeric computation such as numerical integration, solving ODE's, combinatorics, fourier transforms, linear algebra, statistics, complex numbers, geometrical computations, and more.
mathemaTcl	The mathemaTcl project provides package wrappers for a variety of C and Fortran libraries for numerical computation. Home page at http://chiselapp.com/user/arjenmarkus/repository/mathemaTcl/index.
`vectcl`	The VecTcl extension is geared towards processing of numerical arrays with support for vectors, matrices and tensors. It also supports complex numbers. The extension is written in C for high performance and also offers a specialized syntax for numerical computation. Available from http://auriocus.github.io/VecTcl.
`nap`	The Tcl-nap (n-dimensional array processor) is another C based extension that implements efficient commands for processing n-dimensional arrays. It includes support for HDF and netCDF file storage formats. Available from http://tcl-nap.sourceforge.net.
`mpexpr`	This extension offers the ability to calculate with arbitrary precision. Unlike Tcl's native capability which has support for integers of unlimited size, `mpexpr` works with floating point numbers as well. Available from https://sourceforge.net/projects/mpexpr/files.

A.5. Database access

We described the generic core package for database access, TDBC, in Chapter 23. There are however also additional extensions that either target non-SQL databases, or are customized for specific database implementations. There are too many to enumerate here and there is no basis for which ones are "important" so we will just point you to their listing[9] in the GUTTER catalog.

[6] http://wiki.tcl.tk
[7] http://core.tcl.tk/tcllib/doc/trunk/embedded/index.html
[8] http://core.tcl.tk/tcllib/doc/trunk/embedded/index.html
[9] http://core.tcl.tk/jenglish/gutter/#cat-database

A.6. XML processing

Package	Description
tdom	The most widely used Tcl package for XML processing is tDOM. This extension is a very fast engine for parsing and generating XML with excellent XPath and XSLT support. You can also use the package to parse HTML. Available from http://core.tcl.tk/tdom.
tclxml	Another option for processing XML comes from the TclXML project. However, this is not under active development although it is stable enough for production use. It's biggest advantage over tDOM is that it has an optional implementation in pure Tcl that does not require any binary extensions. Available from https://tclxml.sourceforge.net.

A.7. Integration with other languages

Package	Description
ffidl	Allows direct calling of functions implemented in shared libraries from Tcl scripts as long as the functions use the C calling convention. Available from https://github.com/prs-de/ffidl.
tcc4tcl	This extension is a full C compiler that can compile and call C code embedded in a Tcl script at runtime. Note however that unlike most Tcl extensions, it is covered under the more restrictive LGPL license. Available from http://chiselapp.com/user/rkeene/repository/tcc4tcl/index
java	The Tcl Blend extension implements the java package which offers the ability to access and execute Java code from Tcl as well embed a Tcl interpreter into a Java application. Available from http://tcljava.sourceforge.net.
garuda	The Garuda extension integrates Tcl with the .Net platform. It allows access to libraries written in any .Net based language, such as C#, VB.Net etc. Available from http://eagle.to.
duktape	Implements bindings to the Duktape Javascript interpreter library. It can be used to run Javascript code from within Tcl. Available from https://github.com/dbohdan/tcl-duktape.
libtclpy	Supports calling Python code from Tcl and vice versa. Available from https://github.com/aidanhs/libtclpy.

A.8. Image processing

Package	Description
Img	The Img package adds support for a large number of image formats. Its primary use is in conjunction with the Tk graphical extension. Available from https://sourceforge.net/projects/tkimg/.
crimp	The CRIMP extension implements a set of commands for manipulation of raster images. Available from http://chiselapp.com/user/andreas_kupries/repository/crimp/home.
tclmagick	A wrapper for the GraphicsMagick and ImageMagick image processing libraries. Available from http://tclmagick.sourceforge.net/.
tclgd	A wrapper for the libGD graphics drawing library with support for JPEG, PNG, GIF and other popular formats. Available from https://flightaware.github.io/tcl.gd/.
pdf4tcl	A pure Tcl package for generating PDF files. Available from http://pdf4tcl.sourceforge.net.
tclhpdf	A wrapper around the Haru PDF library for generating PDF files. Available from http://reddog.s35.xrea.com/wiki/tclhpdf.html.

`tclMuPdf`	A wrapper for the MuPDF framework for rendering PDF or extracting data from a PDF file. See http://wiki.tcl.tk/48296.

A.9. Platform-specific extensions

A.9.1. Windows extensions

Package	Description
`registry`	Provides commands for reading and writing the Windows registry. The package is part of the core Tcl distribution.
`dde`	Implements the DDE protocol used for interprocess communication. The protocol itself is however obsolete and should not be used in new applications. This package is also part of the core Tcl distribution,
`twapi`	The Tcl Windows API package implements a large portion of the Windows API. It allows Windows services, as well as COM clients and servers, to be written as pure Tcl scripts. It provides access to the Windows event log, WMI, administrative API's, crypto services, systems services, desktop integration and more. Available from https://twapi.sf.net.
`cawt`	Includes modules for integrating with Microsoft Office applications and file formats. Available from http://www.posoft.de/html/extCawt.html.

A.9.2. Unix extensions

Package	Description
`expect`	The Expectextension, which we briefly mentioned in Chapter 24, is one of the most popular Tcl extensions. It is most commonly used in Unix environments for automation of tasks that require user interaction. Available from http://expect.sf.net.
`tuapi`	The Tcl Unix API package wraps several system calls. Currently this is only available on Linux. Available from https://chiselapp.com/user/rkeene/repository/tuapi/index.
`tclx`	The Extended Tcl (TclX) package also provides additional Posix interfaces to files, system services, signals etc. Available from https://sourceforge.net/projects/tclx.

A.9.3. Android extensions

In the case of the Android platform, the AndroWish[10] distribution includes a full set of built-in Android-specific commands, `borg`, `sdltk`, `rfcomm`, `usbserial` for interacting with the Android operating system.

[10] http://www.androwish.org

Appendix B. Utility scripts

We use some simple utility scripts in this book for purposes of pretty-printing etc. These are listed here. Some of these require the `fileutil` package from Tcllib[1] to be loaded.

```
package require fileutil
```

The `print_args` utility simply prints the arguments passed to it, separated by commas.

```
proc print_args {args} {
    puts "Args: [join $args {, }]"
}
```

The `print_list` utility prints each element of a list on a new line. The `print_sorted` utility is similar but prints them in sorted order.

```
proc print_list {l} {
    puts [join $l \n]
}

proc print_sorted {l} {
    print_list [lsort -dictionary $l]
}
```

The `print_dict` utility, transcribed from the Tcllib[2] debug module prints a formatted dictionary.

```
proc print_dict {dict args} {
    if {[llength $args] == 0} {
        set names [lsort -dict [dict keys $dict]]
    } else {
        set names {}
        foreach pattern $args {

            lappend names {*}[lsort -dict [dict keys $dict $pattern]]
        }
    }
    set maxl 0
    foreach name $names {
        if {[string length $name] > $maxl} {
            set maxl [string length $name]
        }
    }
    set maxl [expr {$maxl + 2}]
    set lines {}
    foreach name $names {
        set nameString [format %s $name]
        lappend lines [format "%-*s = %s"  $maxl $nameString  [dict get $dict $name]]
    }
    puts [join $lines \n]
}
```

The `print_array` prints the contents of an array or a subset thereof.

[1] http://core.tcl.tk/tcllib/doc/trunk/embedded/index.html
[2] http://core.tcl.tk/tcllib/doc/trunk/embedded/index.html

```
proc print_array {args} {
    uplevel 1 parray $args
}
```

The `print_file` utility dumps the contents of a file while `write_file` writes specified contents to a file.

```
proc print_file {path} {
    fileutil::cat $path
}

proc write_file {path content} {
    fileutil::writeFile $path $content
}
```

The `wait` utility enters the event loop for the specified amount of time.

```
proc wait {ms} {
    after $ms [list set ::_wait_flag 1]
    vwait ::_wait_flag
}
```

The `lambda` command is syntactic sugar for defining an anonymous procedure.

```
proc lambda {params body args} {
    return [list ::apply [list $params $body] {*}$args]
}
```

The `bin2hex` command pretty prints binary data in hex.

```
proc bin2hex {args} {
    regexp -inline -all .. [binary encode hex [join $args ""]]
}
```