Current Releases

**8.5 - Long Term Support**
- No further releases expected except for security fixes (if needed)
- Expect one further capstone release when we stop support (as with 8.4.20)

**8.6 - Main Production Release (8.6.8)**
- Recommended for all new user code
- TCT permission required to target for changes

Only two TIPped 8.6 changes in last year
- **TIP 473**: Allow a Defined Target `Namespace` in `oo::copy`
  - Bug fix, but in an API
- **TIP 477**: Modernize the `nmake` build system
  - Build system updates for MSVC toolchain
Development

✱ New TIP website
 ✱ Based on fossil

✱ 8.7 - Open for New Features
 ✱ Branch: core-8-branch
 ✱ Most TIPs should target this
 ✱ Currently in alpha (8.7a1 is out; a2 this year)

✱ 9.0 - Open for New Compatibility-Breaking Changes
 ✱ Branch: trunk
 ✱ Try not to break things gratuitously!
 ✱ Currently in very early alpha (pre-9.0a1)
The New TIP Website

- **Fossil** repository
  - Index page generated by a Tcl script
- More accessible to Tcl developers
  - I’m no longer the bottleneck!
  - Formatted using Markdown
- **Searching**
  - Uses SQLite FTS, of course
- **Client-side Filtering**
  - Basic categories

Many thanks to Mark Janssen for his work on this
New (so far) in Tcl 8.7
New Commands

✱ array for efficiently iterates over arrays
  ◆ TIP 421: A Command for Iterating Over Arrays

✱ package files describes what a package really depends on
  ◆ TIP 459: Tcl Package Introspection Improvements

✱ tcl::process adds better control over subprocesses
  ◆ TIP 462: Add New [::tcl::process] Ensemble for Subprocess Management

Driven by FlightAware bounties
New Capabilities

✱ Decimal literals can be prefixed with `0d` (compare `0x`, `0o`, `0b`)
  ✴ TIP 472: Add Support for 0d Radix Prefix to Integer Literals

✱ Use commands to generate substitutions with `regsub`
  ✴ TIP 463: Command-Driven Substitutions for `regsub`

✱ New unit, `weekdays`, for `clock add`
  ✴ TIP 444: Add "weekdays" unit in `clock add`

✱ Updated basic Unicode support (beyond the BMP, 😶)
  ✴ TIP 389: Full support for Unicode 10.0 and later (part 1)

✱ Striding list searching, just like striding list sorting
  ✴ TIP 351: Add Striding Support to `lsearch`
TclOO Changes

* Added private methods...
  * TIP 500: Private Methods and Variables in TclOO

* ... and private variables
  * TIP 500: Private Methods and Variables in TclOO

* Made it easier to script your own definition commands
  * TIP 470: Reliable Access to OO Definition Context Object

* Control over objects’ namespace names in oo::copy
  * TIP 473: Allow a Defined Target Namespace in oo::copy
Standard Package Changes

- Updates to **msgcat** package to give richer locale searching and make it work with TclOO
  - TIP 499: Custom locale search list for msgcat
  - TIP 490: msgcat for TclOO

- Updates to **tcltest** to allow for (optional) performance measurement of tests
  - TIP 447: Execution Time Verbosity Levels in **tcltest::configure**
Rework of the Notifier

* Support for more modern notifiers
  * TIP 458: Add Support for epoll() and kqueue() in the Notifier

* Better in several ways
  * More efficient
  * Support more open channels

* Still use old notifiers on some platforms
  * Windows and OSX have their own
  * Very old Unix still uses select()
Threading for Everyone

- Threaded builds are default for everyone
  - TIP 491: Threading Support: phasing out non-threaded builds
- Unthreaded builds have nasty bugs when used in threaded processes
  - Particularly an issue for embedding
- Windows and OSX already require threads
  - Platform-specific notifiers work that way
- Unthreaded Tcl builds will be phased out
Low Level Changes

- **Internal types are merged**
  - TIP 484: Merge 'int' and 'wideInt' Obj-type to a single 'int'

- **Format consistencies are removed**
  - TIP 476: Scan/Printf format consistency

- **TCP server creation is more flexible**
  - TIP 456: Extend the C API to Support Passing Options to TCP Server Creation

- **Panic can use non-ASCII on Windows**
  - TIP 425: Correct use of UTF-8 in Panic Callback (Windows only)
Expiring Support

✱ Some things are gone because we really don’t do them any more
  ✴ TIP 503: End Tcl 8.3 Source Compatibility Support
  ✴ TIP 487: Stop support for Pre-XP Windows

✱ Some things are gone because they are the wrong idea and have been for a long time
  ✴ TIP 493: Cease Distribution of http 1.0
  ✴ TIP 345: Kill the 'identity' Encoding
Bubbling Under

Some features we expect to vote on:

✱ ZIP filesystem
  ✴ Core scripted documents

✱ Type assertions and aliasing assertions
  ✴ Product of tclquadcode project

✱ In-place string and binary data modification

✱ More TclOO features
  ✴ Import of oo::util package from tcllib
New *(so far)* in Tk 8.7
**Widget Changes**

* **Initial help text for entry-derived widgets**
  * TIP 496: Display hints in ::entry
    ::spinbox ::ttk::entry ::ttk::spinbox
  and ::ttk::combobox

* **Labels for progress bars**
  * TIP 442: Display text in progressbars
  * NB: label not visible in all styles!
Other New Features

✱ **Snapshot a canvas as an image**
  ✳️ TIP 489: Add image widget command to the Tk `canvas`

✱ **Tweaked introspection**
  ✳️ TIP 492: Introspection for `tk busy`
  ✳️ TIP 449: `text` undo/redo to Return Range of Characters

✱ **Some really old TIPs too**
  ✳️ TIP 166: Reading and Writing the Photo `Image` Alpha Channel
  ✳️ TIP 161: Change Default for `Menu`'s `-tearoff` Option to False
Bubbling Under

Some features we expect to vote on:

- **SVG** support for the `canvas`
  - Support modern graphics

- **RBC** widgets *(graph, stripchart, barchart)*
  - Multiple production widgets
  - Needs coordination on “vector” in Tcl
Changes (so far) in Tcl 9.0
Changes in 9.0

* Tcl always gets **double-to-string conversions** right by default now
  * TIP 488: Remove `tcl_precision`

* Long **deprecated API** now gone (e.g., `case`, old `puts` syntax)
  * TIP 485: Remove Deprecated API

* Some C API types aren’t portable, so are removed from Tcl API
  * TIP 422: Don’t Use `stdarg.h/va_list` in Public API

* We had some horrid sort-of-documented bugs in **name resolution**
  * TIP 278: Fix Variable Name Resolution Quirks

* Octals like **0123** are now **replaced** with **0o123**
  * TIP 114: Eliminate Octal Parsing of Leading Zero Integer Strings
Likely Future Changes

- **64-bit Memory Object Sizes**
  - Big Strings
  - Long Lists
  - Huge Hashtables
  - Causes many small (annoying) changes at C API level

- **Full Unicode Support**
  - Change of Tcl_UniChar size
  - Supporting Tcl commands for normalisation, etc.
The tclquadcode Project

Status and Demo
Compiling Tcl... All of Tcl...

1. Discover procedures in package
2. Translate bytecode to “better bytecode”
3. Add types
4. Optimise
5. Generate native code
6. Write code to DLL
7. Load DLL to implement package
Compiling Tcl... All of Tcl...

- **Quadcode** is our better bytecode
  - Not really bytecode; Tcl lists are easier to read
- **Static Single Assignment** form allows **optimisation** work
  - Variable Liveness Analysis
  - Cross-Procedure Analysis
- **Type system** for Tcl code
  - Everything is a string... or a subtype of string
    - Integer, Float, Boolean
- **Generate** **native code** using **LLVM**
  - Slow but portable
  - Potential for other targets...
Status

Currently Working

* Support almost all bytecoded procedures
  - Prototype of coroutine support
  - Access to arrays, global variables, upvar

* Call to Tcl interpreter for general commands
  - I/O handled this way
  - User C code handled this way
  - Some special cases

In Progress or Planned

* Types to do
  - Bignums (Tcl API needs work)
  - Lists (types of elements)
  - Dicts (types of elements)

* Bytecoded Entities to do
  - TclOO methods
  - Lambda terms
  - Code outside procedures
    - Probably will never be done
Live Demo!
Current Areas of Work

- **Stackless form and Coroutines**
  - I believe it works, but with a performance penalty
  - Costs relate to extra memory allocations required

- **Removing static data in the implementation**
  - Original design goal was for code that was loaded once (JIT)
  - LLVM too slow for that to be realistic
    - Workaround: Generate good code in a DLL (or executable?) and load it
  - Multiple uses per process possible → keep data in interpreter or command clientData

- **Making a better front-end API to the compiler**
  - Let other analysis tools use our output (nagelfar?)
  - Let some package code also be fully compiled (vectcl?)