Tcl Values:
Past, Present & Tales from the Future

2016 Tcl Conference
Don Porter
Tcl/Tk Release Manager

Why I Am Working on TIP 445.

2016 Tcl Conference
Don Porter
Tcl/Tk Release Manager

Tcl Value?

- Returned by command
- Stored in variable
- Passed as an argument
- Held in a list.

```
set  v [cmd arg]
```

Tcl Foundations

```
int main(int argc, char *argv[]) {
    return 0;
}
```

- argv[i] points to NUL-terminated char array.
- Value: finite sequence of {0x01 – 0xFF}.

Tcl 7 Value

argv → Tcl_CmdProc → Tcl_SetResult(**result**, freeProc)

- Establishes the semantics of Tcl values.
  - “Everything is a string”
  - Implies: no NULL value; names, not references; Tcl typing
  - Revisions can optimize, but not escape this model.
- Pros
  - matches C, familiar to extension writers
- Cons
  - Fails “8-bit clean” (and Unicode completeness)
  - Conversion burdens make things slow.

Tcl 8(.1) Value: Tcl_Obj

objv → Tcl_ObjCmdProc → Tcl_SetObjResult(**objPtr**)

argv → Tcl_CmdProc → Tcl_SetResult(**result**, freeProc)

- objv[i] points to a Tcl_Obj struct (24 or 40 bytes each):

  {int refCount;

  char *bytes; int length; /* String Representation. */

  Tcl_ObjType *typePtr; internalRep} /* Other Representation */
- Internal Rep goes along for the ride – saves conversions.
- Tcl_ObjType (optionally) defines routines so Tcl can command...
  - Conversion Internal Rep to String Rep Tcl_GetString()
  - Free an Internal Rep Tcl_DecrRefCount(), Tcl_FreeIntRep()
  - Duplicate an Internal Rep Tcl_DuplicateObj()
  - Create an Internal Rep of the value (if possible) Tcl_ConvertToType()
- objPtr→bytes is Tcl 7 value. Easy accommodation of Tcl 7 conventions.
  - Revised encoding. Modified UTF-8 encodes entire BMP, including U+0000
  - Lost freeProc! bytes is always ckalloc()ed. Always making copies!

Tcl 8 Value: The Stork

```
Tcl_Obj struct:
{refCount;
 bytes; length;                                  /* String Rep */
 Tcl_ObjType *typePtr; internalRep}    /* Other Rep */
```

- Either bytes or typePtr must be non-NULL.
- Both can be non-NULL, but then must agree.
- When bytes == NULL, say the value is “pure”.
- A New internalRep destroys an old one. (“shimmer”)
  - Conversion via string rep, or by being 'friends'

Tcl 8 Value: The Good

- All code written to Tcl 7 still works.
- Tcl 8 value resolves all Tcl 7 value cons!!!
  - Much reduced conversion burden.
  - Binary-safe, Support of Unicode's BMP
- In practice, programmers accepted it.
  - Mmmmm…. Yummy carrots.

Tcl 8 Value: The Bad

- Unicode grew, Tcl value alphabet didn't.
- Inessential properties of Tcl_Obj
  - Limit capabilities
  - Burden evolution
  - Tcl's value model is consistent with many programming innovations while the properties of the Tcl_Obj struct are not.
    - Pure functional, immutable data structures, HAMT, RRB trees, ropes.
  - Especially troublesome for large scaling.

Tcl_Obj: Inessential properties

```
{refCount;
 bytes; length;                              /* String Rep */
 Tcl_ObjType *typePtr; internalRep}    /* Other Rep */
```

- Size limited (INT_MAX = 2G)
- Open structs
- Mutable / Copy on Write
- RefCounted → Thread isolated.
- String Rep is Tcl 7 value without freeProc
- At most one additional Rep. (“hydra”)
- Each Rep is absent or complete. No partial conversions.

Example

```
% proc K {x y} {return $x}
% set x [string repeat a 3000000]
% time {set x [string replace $x 2 2 b]} 100
2235.11385 microseconds per iteration
% time {set x [string replace [K $x [unset x]] 2 2 b]} 100
4.20126 microseconds per iteration
```

- Impact of sharing is script visible.

Tcl 9 Value – New Struct?

valv → **Tcl_ValCmdProc** → **Tcl_SetValResult(val)**

objv → **Tcl_ObjCmdProc** → **Tcl_SetObjResult(objPtr)**

argv → Tcl_CmdProc → Tcl_SetResult(**result**, freeProc)

- Or is better encapsulated Tcl_Obj struct flexible enough?
- Needs experimentation and interfaces to support it.

TIP 445

- Tcl_FreeInternalRep(obj)
  - Replaces obj → typePtr → freeIntRepProc(obj)
- Tcl_InitStringRep(…)
  - Replaces direct alloc and write to bytes
- Tcl_StoreIntRep(...), Tcl_FetchIntRep(…)
  - Act on internalRep without direct field access
  - Without assuming single internalRep
- ….and more as work reveals.

Tcl 9 value desirables?

- Code written to Tcl 7 and 8 should still work.
- Much increased sized limitations.
- Immutable
- Thread-sharable
- Reduced shimmer impact
- Full Unicode, with canonical equivalence
- Share data, not values

Closing Thoughts

- Should we invent a new structure or modify existing one?
  - How much can inessentials be purged without upsetting released base that assumes them?
  - Experiments in progress.
- Can changes be completed in reasonable time?
  - Interface first, for 9.0.
  - Continued progress in 9.1, 9.2, etc.
- Will coders accept and adapt?
  - Need big yummy carrots.