Tcl ist nicht tot!

Uwe Berger

„Tot geglaubte leben länger...?!“

Inhalt

- Motivation
- Was ist Tcl/Tk?
- Tcl Eigenschaften/Besonderheiten
- Die Sprache Tcl an Hand von Beispielen
- Grafische Oberflächen mit Tk
- Komplexe Anwendungen (Beispiele)
- Informationsquellen

- Motivation
- Was ist Tcl/Tk?
- Tcl Eigenschaften/Besonderheiten
- Die Sprache Tcl an Hand von Beispielen
- Grafische Oberflächen mit Tk
- Komplexe Anwendungen (Beispiele)
- Informationsquellen

(Meine) Motivation

- Beginn meiner Tcl-Geschichte: 1995/6
- Suche nach einer einfachen Möglichkeit komplexe Anwendung zu erstellen
  - damals nur Pascal-Erfahrungen
  - C war noch ein Fremdwort
- zufällig das Tcl-Buch in die Hand bekommen
- „Der ersten Liebe treu geblieben!“

Live-Demo

- Motivation
- Was ist Tcl/Tk?
- Tcl Eigenschaften/Besonderheiten
- Die Sprache Tcl an Hand von Beispielen
- Grafische Oberflächen mit Tk
- Komplexe Anwendungen (Beispiele)
- Informationsquellen

Was ist Tcl?

- Tcl - Tool command language
- eine Scriptsprache wie bash, Perl, php, etc.
- eine Interpretersprache (tclsh, wish)
- auf allen bedeutenden OS-Plattformen verfügbar
- John Ousterhout, ca. 1988
  - http://www.tcl.tk/about/history.html
  - Simulation von Schaltkreisen
  - sollte neben Java auch „Websprache“ werden
- Open Source

Was ist Tk?

- Tk - Toolkit
- Programmierung von grafischen Benutzeroberflächen
- wurde ursprünglich für Tcl entwickelt (ebenfalls J. Ousterhout; 1988)
- Plattformübergreifend verfügbar (soweit grafische Oberfläche vorhanden)
- Anbindung auch für Perl, Python, Ruby und einige weitere vorhanden
- Open Source

- Motivation
- Was ist Tcl/Tk?
- Tcl Eigenschaften/Besonderheiten
- Die Sprache Tcl an Hand von Beispielen
- Grafische Oberflächen mit Tk
- Komplexe Anwendungen (Beispiele)
- Informationsquellen

Tcl Eigenschaften/Besonderheiten

- nach dem 2. Hinsehen, sehr einfache Syntax-Regeln
- durchgängig Polnische Notation (Befehl Parameter ...)
- "Alles ist ein String"
  - alle Datentypen können als String bearbeitet werden (intern natürlich auch Zahlenformate)
  - gleiches gilt auch für Programmcode
- sehr gute Unterstützung bei der Verarbeitung von Listen und Arrays

Tcl Eigenschaften/Besonderheiten

- ereignisgesteuerte Mechanismen für Socket- und Datei-Schnittstellen
- Seriell-, Parallel- und Netzwerk-Schnittstellen werden als Dateien behandelt
- Zeit- und Benutzerdefinierte Ereignisse
- einfache Ausnahmebehandlung (Fehlerbehandlung)
- einfache Client-/Server-Programmierung
- ... man findet immer wieder neue interessante Dinge ...

- Motivation
- Was ist Tcl/Tk?
- Tcl Eigenschaften/Besonderheiten
- Die Sprache Tcl an Hand von Beispielen
- Grafische Oberflächen mit Tk
- Komplexe Anwendungen (Beispiele)
- Informationsquellen

Tcl Syntax-Prinzipien

- immer(!): Kommandowort Parameter1 Parameter2 ...
- Variablen beginnen mit $ und müssen vor der ersten Verwendung gesetzt sein
- Zeichenketten stehen in doppelten Anführungsstrichen
- Ein Kommando wird von einem Zeilenende oder Semikolon begrenzt
- Backslash (\) am Zeilenende -> Befehl geht auf nächster Zeile weiter

Tcl Syntax-Prinzipien

- geschweifte Klammern schützen den Inhalt vor Interpretation (von außen nach innen); sind keine Strukturelemente
- Code in eckigen Klammern wird zuerst ausgeführt (von innen nach außen)
- Variablen nur lokal gültig und Gültigkeitsbereich muss explizit erweitert werden (global, upvar, uplevel)
- Kommentarzeichen (-befehl): # (am Zeilenanfang oder hinter einem Semikolon)

Starten eines Tcl-Script

- Interaktiv in der Tcl-Shell (tclsh, wish) selbst; z.B.:
  - `% source tcl_script.tcl`
- Script als Parameter der Tcl-Shell:
  - `tclsh tcl_script.tcl`
  - `wish tk_script.tcl`
- Als ausführbares Script mit Angabe des Interpreters:

```
#!/usr/bin/tclsh
package require Tk
...
```

TCL in Beispielen

- `bsp_1.tcl`: Variablen, Rechnen, Ausgabe 
- `bsp_2.tcl`: Ein-/Ausgabe, Schleifen, if/else 
- `bsp_3.tcl`: Filesystem, Listen, Fehlerbehandlung
- `bsp_4.tcl`: Zeitgesteuerte Verarbeitung 
- `bsp_5_x.tcl`: Client-/Server-Anwendung, Interpreter 
- `bsp_6.tcl`: Aufruf von externen Programmen, Dateiereignis

- Motivation
- Was ist Tcl/Tk?
- Tcl Eigenschaften/Besonderheiten
- Die Sprache Tcl an Hand von Beispielen
- Grafische Oberflächen mit Tk
- Komplexe Anwendungen (Beispiele)
- Informationsquellen

Grafische Elemente

- „nativ look and feel“
- Tk-Grundelemente (Widgets):
  - Frames, Label, Textfeld, Button, Radio-Button, Check-box, Textbox, Listbox, Menü
  - Canvas, Icons/Bilder
- Fenster, Fensterdekoration, Systemdialoge, Message-boxen
- Zu jedem Element sind Reaktionen auf Ereignisse (Klick, Selektion etc.) definierbar

Geometriemanager

- bestimmen die Anordnung von Widgets in einem Container (Fenster, Frame) und deren Verhalten bei Veränderung der Containergröße
- ***pack***: Widgets werden nach festlegbaren Regeln automatisch angeordnet
- ***grid***: gezielte Anordnung der Widgets in einem Raster (Zeilen/Spalten)
- ***places***: frei definierbare Anordnung der Widgets in einem Fenster (relat. xy-Koord., Höhe/Breite)

Tk in Beispielen

- `bsp_7.tcl`: „Hello Uwe...“ 
- `bsp_8.tcl`: Eingabefeld, Radiobutton, Messagebox 
- `bsp_9.tcl`: Listbox, Scrollbar, Textbox
- `bsp_10.tcl`: Menües, Systemdialoge
- `bsp_11.tcl`: Zeichnen mit Canvas 

- Motivation
- Was ist Tcl/Tk?
- Tcl Eigenschaften/Besonderheiten
- Die Sprache Tcl an Hand von Beispielen
- Grafische Oberflächen mit Tk
- Komplexe Anwendungen (Beispiele)
- Informationsquellen

Spracherweiterungen

- Warum:
  - Performance
  - Hard-/Software-Schnittstellen
  - Zusammenfassung von komplexen Code
- Wie:
  - Packages in Tcl/Tk geschrieben(z.B. Tcllib, BWidget)
  - Erweiterungen in compilierten Bibliotheken
    - festgelegte Konventionen bei der Funktionsdekleration sind einzuhalten (für C sehr gut dokumentiert)

Tipps zur Entwicklung von Tk-GUIs

- wenn möglich nur Geometriemanager pack und grid verwenden: automatische Anordnung der Widgets
- zuerst Oberfläche vollständig konzipieren und als „Programmhülse“ realisieren, dann Funktionalität
- möglichst Tcl/Tk nur im Standardsprachumfang verwenden: unkomplizierte Weitergabe der Programme
- Fehlerbehandlung nicht vergessen!

Komplexe Anwendungen in Tcl/Tk

- sehr oft als GUI für Kommandozeilentools verwendet
- eigene Anwendungsbeispiele:
  - Cube-Viewer und -Editor
  - LED-Matrix-Server
  - NET-I/O-Client

- Motivation
- Was ist Tcl/Tk?
- Tcl Eigenschaften/Besonderheiten
- Die Sprache Tcl an Hand von Beispielen
- Grafische Oberflächen mit Tk
- Komplexe Anwendungen (Beispiele)
- Informationsquellen

Informationsquellen

- „Tcl und Tk. Entwicklung grafischer Benutzerschnittstellen für X Window System“; Ousterhout; Addison-Wesley
- „Effektiv Tcl/Tk programmieren“; Harrison, McLennan; Addison-Wesley
- http://www.activestate.com
- http://www.tcl.tk
- http://wiki.tcl.tk
- Tcl-Newsgroup (engl.): comp.lang.tcl

Danke für die Aufmerksamkeit!