Uma linguagem de shell scripting chamada TCL

José Paulo Leal
zp@ncc.up.pt

Antes de mais o TCL **não**

- Executa mais rápido que outras linguagens
- Possui as melhores expressões regulares
- Tem suporte para todas as ferramentas
- É a melhor linguagem para scripts web
- É uma linguagem orientada a objectos
- Tem um número crescente de adeptos

No entanto o TCL é

- É uma linguagem simples
- Os programas são fáceis de manter
- É fácil de extender usando C/C++
- É (razoavelmente) independente do SO
- Bom suporte para GUIs

Introdução

- Tcl = Tool Command Language
- Criada por John Ousterhout nos finais dos 80
- Características
  - Linguagem de **Shell Scripting**
  - Interpretada sobre bytecodes
  - Suporte para GUI e outras ferramentas
  - Multi-plataforma (Unix, Windows e Mac)

Scripting

- Integração de componentes (reutilização)
- Linguagens interpretadas
- Relaxamento na verificação de tipos
- Cada vez mais linguagens de sistema
- Outras linguagens (não cobertas neste ciclo)
  - Javascript
  - Visual Basic
  - sh, csh, ksh

Shell Scripting

- Parte do sucesso do Unix
- Processamento de linhas de comando
- Canais de I/O (pipes)
- Valores são (apenas) “strings”
- Substituição de variáveis e comandos
- Paradigma de programação?

Paradigmas / Variáveis

- Imperativo `X = X*X`
  - Atribuição destrutiva, valor esquerdo e direito
- Lógico `f(X,b)=f(a,Y)`
  - Atribuição não destrutiva, unificação
- Funcional `let X = Y*Y`
  - Atribuição não destrutiva, emparelhamento
- Shell scripting `set X "$X$X"`
  - Atribuição destrutiva, substituição

Atribuição

```
% set x 1
1
% set x
1
```

- Comando set
  - atribui valores (com 2 argumentos)
  - inspecciona (com apenas 1 argumento)
- Comandos retornam um valor

Substituição

```
set comando set
set variavel x
set valor 1
$comando $variavel $valor
```

- Variáveis substituídas pelo valor
- Substituição precede a avaliação
- Substituição em qualquer ponto

Linhas de comando

```
set nome “José Paulo Leal”
 set morada {
   Rua do Campo Alegre, 823
   4150-180 Porto}
```

- Terminadas por mudança de linha
- Argumentos separados por espaços e tabulações
- Caracteres brancos dentro de delimitadores
  - Aspas permitem a substituição
  - Chavetas inibem a substituição

Programas

```
if $exiting {
  foreach window $windows {
    destroy $window
  }
}
```

- Comandos usados no controlo de fluxo
- As chavetas não fazem parte da sintaxe
- Valores são sempre strings ...
- ... mas podem ser interpretados segundo contexto

Interpretadores

- Sistema de ficheiros
  - Ficheiros e directorias
  - Nomes caminho
- `tclsh` (sh + tcl)

- Sistema de Janelas
  - Objectos e contentores
  - Nomes caminho
- `wish` (tclsh + tk)

Comandos TK

- Construtores de objectos (classes)
  - `label, button, entry, list`
  - `text, canvas`
- Gestores de posição (estratégias)
  - `pack, place, grid`
- **Processamento de eventos**

Ligação Tcl Tk


```
set c 1
pack [ button .b -text Conta \
      -command {incr c} ]
pack [ label .l -textvariable c ]
```

- Execução de linhas de comando (“callback”)
- Comunicação por variáveis globais

Programação por eventos

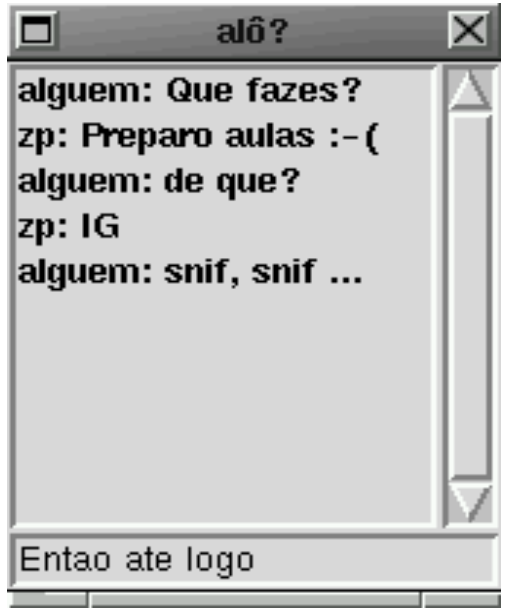
- Gestores de eventos
  - Recurso + Tipo + Linha de comando
- Recursos que geram eventos
  - Janelas
  - Canais de IO
  - Relógio
  - Variáveis
- Eventos como alternativa a “threads”

Eventos

```
wm title . "alô"
pack [ frame .f ]
pack [ listbox .f.l -yscrollcommand { .f.s set } ] \
   -side left -fill both
pack [ scrollbar .f.s -command { .f.l yview }  ]   \
   -side left -fill y
pack [ entry .e -textvariable linha ] -fill x

set fd [ open log a+ ]

bind      .e  <Key-Return> "envia"
fileevent $fd readable     "recebe $fd"
```



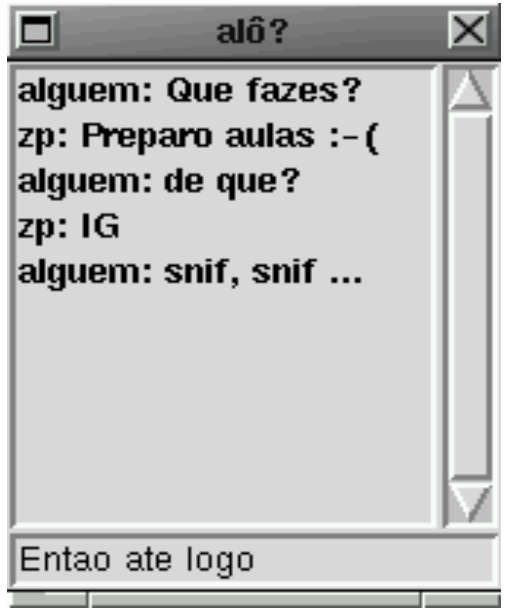
Eventos 2

```
...
bind       .e  <Key-Return>    "envia"
fileevent $fd readable         "recebe $fd"

proc envia {} {
   global linha env

     set fw [ open log a ]
     puts $fw  "env($USER): $linha"
     catch { close $fw }

     set linha ""
}
```

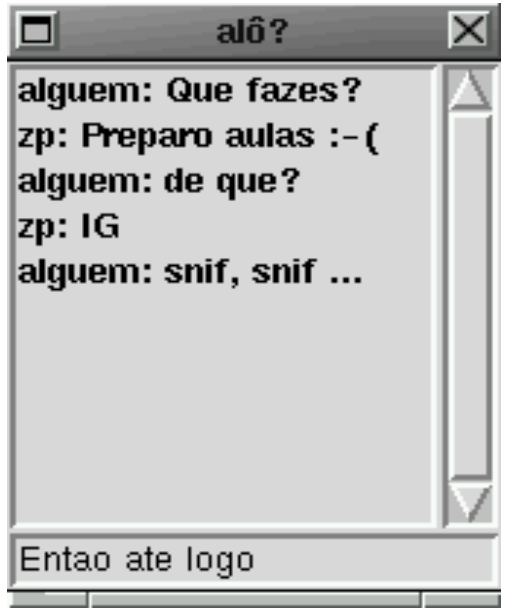


Eventos 3

```
...
bind        .e  <Key-Return>    "envia"
fileevent $fd readable          "recebe $fd"

...

proc recebe {fd} {
    while { [ gets $fd linha ] > -1 }
        .f.l insert end $linha
        .f.l see end
    }
    update idletasks
}
```



Em resumo

- Tcl é uma linguagem de shell scripting
- É uma linguagem simples mas versátil
- Bom suporte para interfaces gráficos (TK)
- Ênfase na programação por eventos
- Não está na moda mas ainda mexe...