



Компьютерные технологии в научных исследованиях

Лабораторная работа №3

Программирование графического интерфейса с библиотекой Tk

```
1 #!/bin/bash
2 #INPUT_SAMPLE_LIST=$1
3 cd /Volumes/PhilDrive_EMS/TestDec7/snv_postp
4
11 . paths.txt
12
30
31 echo "Debug level set for $DEBUG_LEVEL"
32 echo "log found in scripts directory"
33
50 cp $HIGH_SNP_OUT ./
51 cp $LOW_SNP_OUT ./
52 cp $GERM_SNP_OUT ./
53 # echo "${SCRIPT_DIR}/run_somatic_mu
54 if [ $DEBUG_LEVEL
55 then
56 echo "INFO: ${SCR
57 `basename ${LOW_S
58 ${D_BAM_FILE} ${G
59
60 fi
61 ${SCRIPT_DIR}run_somatic_mu
62
```



Программирование с библиотекой Tk (1)

Указать интерпретатор wish

```
#!/usr/bin/wish
```

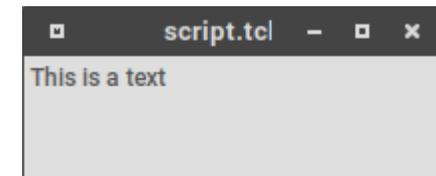
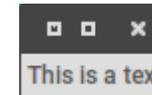
Создать компоненты

```
label .l1 -text "This is a text"
```

```
grid .l1 -row 0 -column 0
```

Расположить компоненты

+ Задать обработчики



Программирование с библиотекой Tk (2)

Указать интерпретатор tclsh

Объяснить интерпретатору,
что мы будем использовать
библиотеку Tk

Создать компоненты

Расположить компоненты

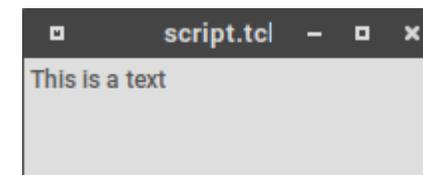
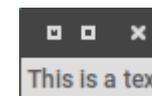
```
#!/usr/bin/tclsh
```

```
package require Tk
```

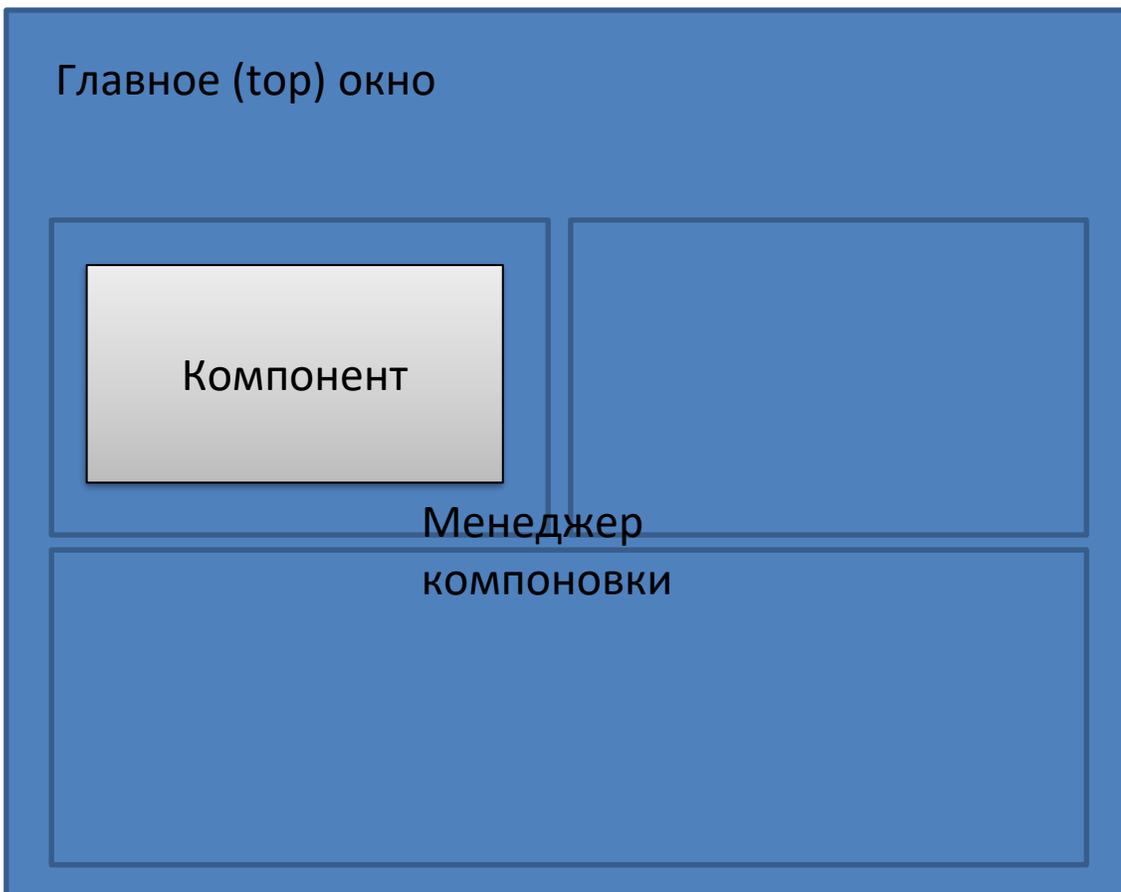
```
label .l1 -text "This is a text"
```

```
grid .l1 -row 0 -column 0
```

+ Задать обработчики



Структура окна скрипта

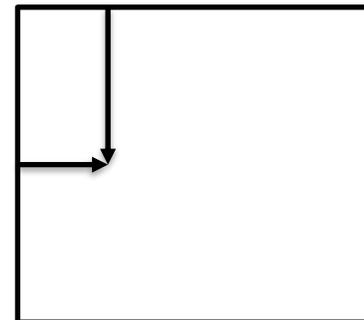
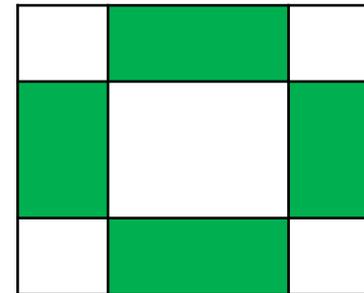
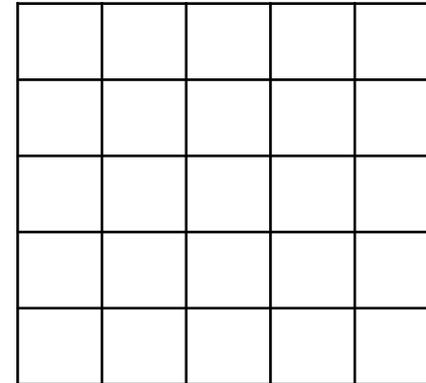


На главном окне
компоненты
располагаются с
помощью менеджера
компоновки

Менеджеры компоновки в Tk

В Tcl/Tk доступны 3 вида менеджеров компоновки:

- `grid` – разделяет рабочую область главного окна на виртуальные ячейки, располагает все компоненты внутри ячеек;
- `pack` – позволяет расположить компоненты внутри окна относительно границ окна;
- `place` – позволяет задать абсолютное позиционирование компонентов на окне.

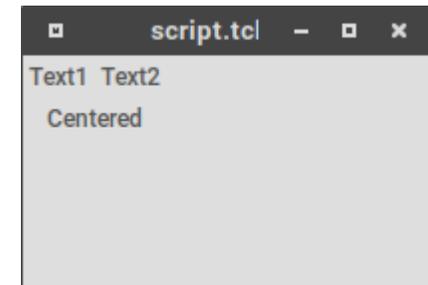
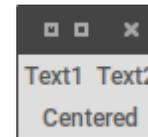
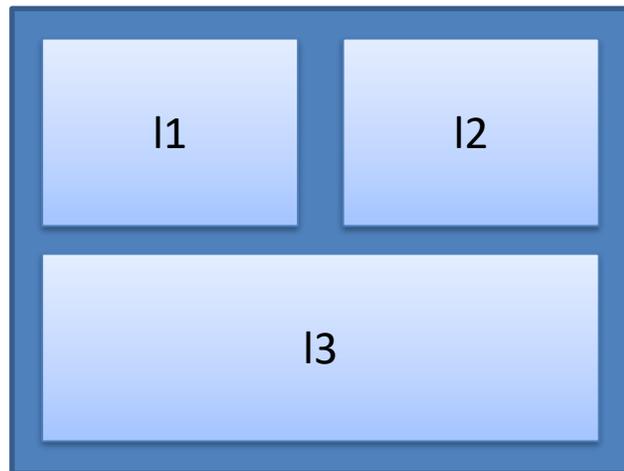


Менеджер компоновки grid (1)

```
#!/usr/bin/wish
```

```
label .l1 -text "Text1"  
label .l2 -text "Text2"  
label .l3 -text "Centered"
```

```
grid .l1 -row 0 -column 0  
grid .l2 -row 0 -column 1  
grid .l3 -row 1 -column 0 -columnspan 2
```



Менеджер компоновки grid (2)

Основные параметры расположения при использовании grid

- row - указание ряда, в котором располагается компонент
- column - указание столбца, в котором располагается компонент
- rowspan - указание числа строк для объединения
- columnspan - указание числа столбцов для объединения
- padx - указание отступа в пикселях от края или краёв по X
- pady - указание отступа в пикселях от края или краёв по Y

Менеджер компоновки pack

```
#!/usr/bin/wish
```

```
# Make the widgets
```

```
label .t -text "Top"
```

```
label .b -text "Bottom"
```

```
label .l -text "Left\nSide"
```

```
label .r -text "Right\nSide"
```

```
# Lay them out
```

```
pack .t -side top -fill x
```

```
pack .b -side bottom -fill x
```

```
pack .l -side left -fill y
```

```
pack .r -side right -fill y
```



Менеджер компоновки place

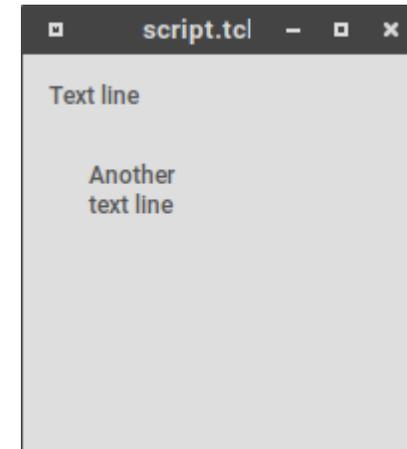
```
#!/usr/bin/wish
```

```
label .l1 -text "Text line"
```

```
label .l2 -text "Another\ntext line"
```

```
place .l1 -x 10 -y 10
```

```
place .l2 -x 30 -y 50
```



Управление главным окном (1)

```
#!/usr/bin/wish
```

```
wm title . "KTNI lab4 example"
```

```
# Make the widgets
```

```
label .t -text "Top"
```

```
label .b -text "Bottom"
```

```
label .l -text "Left\nSide"
```

```
label .r -text "Right\nSide"
```

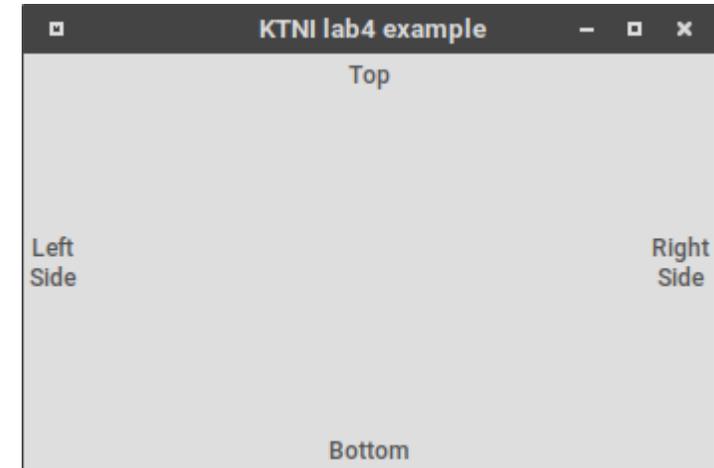
```
# Lay them out
```

```
pack .t -side top -fill x
```

```
pack .b -side bottom -fill x
```

```
pack .l -side left -fill y
```

```
pack .r -side right -fill y
```



Управление главным окном (2)

```
#!/usr/bin/wish
```

```
wm title . "KTNI lab4 example"  
wm geometry . =400x100+10+10
```

```
# Make the widgets  
label .t -text "Top"  
label .b -text "Bottom"  
label .l -text "Left\nSide"  
label .r -text "Right\nSide"
```

```
# Lay them out  
pack .t -side top -fill x  
pack .b -side bottom -fill x  
pack .l -side left -fill y  
pack .r -side right -fill y
```



Управление главным окном (3)

```
#!/usr/bin/wish

wm title . "KTNI lab4 example"
wm geometry . -400x100+10+10
wm resizable . 0 0

# Make the widgets
label .t -text "Top"
label .b -text "Bottom"
label .l -text "Left\nSide"
label .r -text "Right\nSide"

# Lay them out
pack .t -side top -fill x
pack .b -side bottom -fill x
pack .l -side left -fill y
pack .r -side right -fill y
```



Метки

```
#!/usr/bin/wish

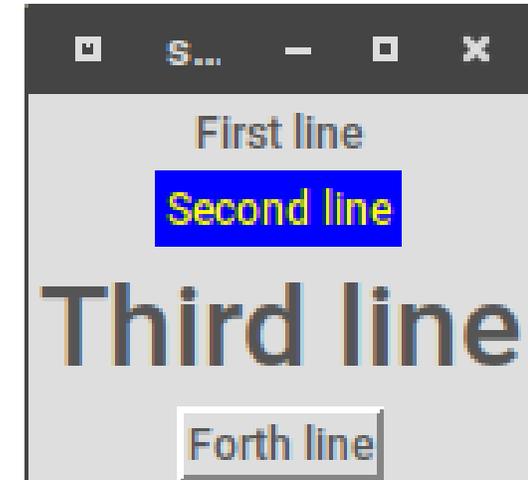
label .l1 -text "First line"

label .l2 -text "Second line" -foreground Yellow \
          -background Blue

label .l3 -text "Third line" -font { -size 24 }

label .l4 -text "Forth line" -relief raised

grid .l1 -row 0
grid .l2 -row 1
grid .l3 -row 2
grid .l4 -row 3
```

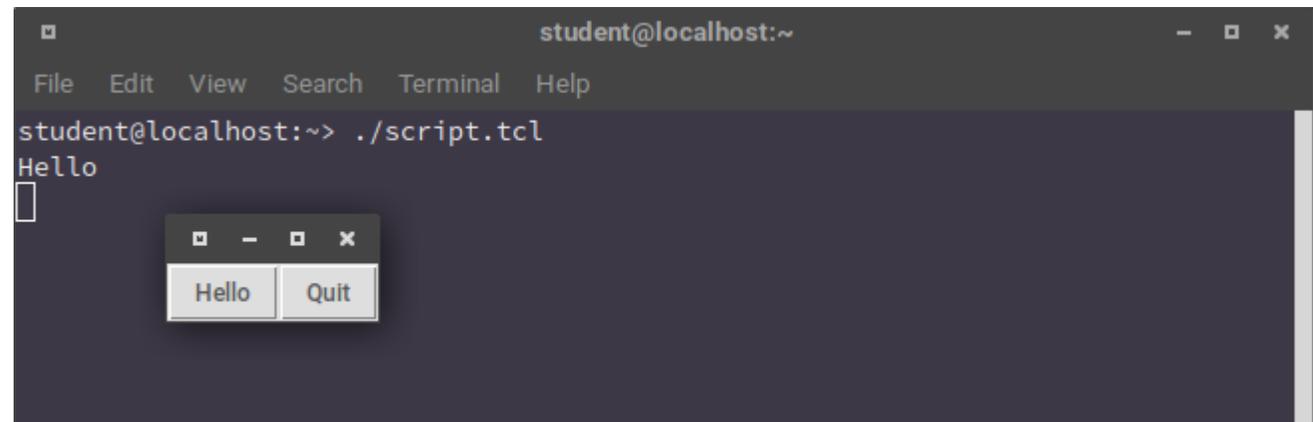


Кнопки (1)

```
#!/usr/bin/wish
```

```
button .b1 -text "Hello" -command { puts stdout "Hello" }  
button .b2 -text "Quit" -command { destroy . }
```

```
grid .b1 -row 0 -column 0  
grid .b2 -row 0 -column 1
```



Кнопки (2)

```
#!/usr/bin/wish
```

```
button .b1 -text "Hello" -command { puts stdout "Hello" }
```

```
button .b2 -text "Quit"
```

```
grid .b1 -row 0 -column 0
```

```
grid .b2 -row 0 -column 1
```

```
#!/usr/bin/wish
```

```
proc onPressHello {} {  
    puts stdout "Hello"  
}
```

```
button .b1 -text "Hello" -command onPressHello
```

```
button .b2 -text "Quit"
```

```
grid .b1 -row 0 -column 0
```

```
grid .b2 -row 0 -column 1
```

Переключатели: позиционные

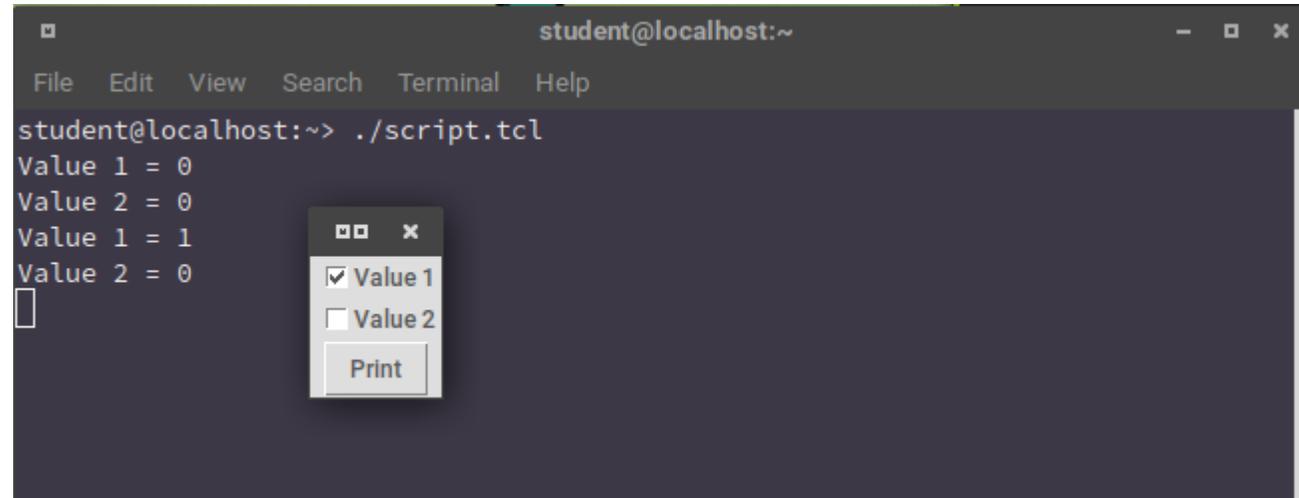
```
#!/usr/bin/wish
```

```
proc printVars {} {  
    global cValue1 cValue2  
  
    puts stdout "Value 1 = $cValue1"  
    puts stdout "Value 2 = $cValue2"  
}
```

```
checkboxbutton .c1 -text "Value 1" -variable cValue1  
checkboxbutton .c2 -text "Value 2" -variable cValue2
```

```
button .b -text "Print" -command printVars
```

```
grid .c1 -row 0 -column 0  
grid .c2 -row 1 -column 0  
grid .b -row 2 -column 0
```



Переключатели: радио

```
#!/usr/bin/wish

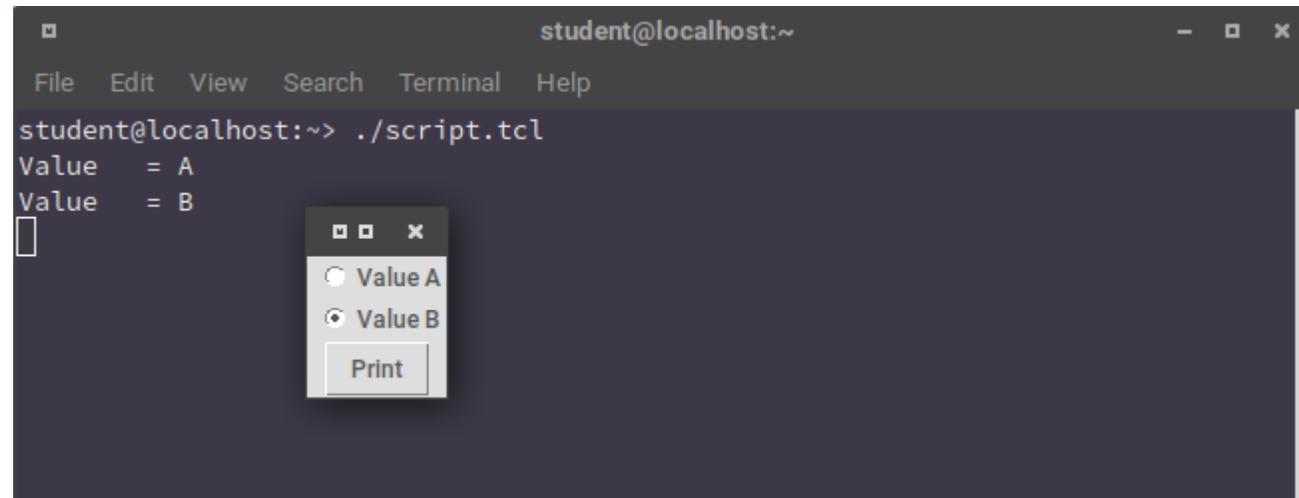
proc printVars {} {
    global rValue

    puts stdout "Value = $rValue"
}
```

```
radiobutton .r1 -text "Value A" -variable rValue -value "A"
radiobutton .r2 -text "Value 2" -variable rValue -value "B"
```

```
button .b -text "Print" -command printVars
```

```
grid .r1 -row 0 -column 0
grid .r2 -row 1 -column 0
grid .b -row 2 -column 0
```



Однострочные поля текстового ввода

```
#!/usr/bin/wish
```

```
label .l -text "Enter: "
```

```
entry .e -width 40 -textvariable varText
```

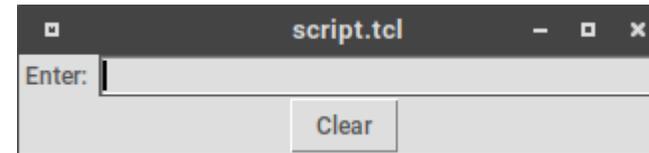
```
button .b -text "Clear" -command { set varText "" }
```

```
focus .e
```

```
grid .l -row 0 -column 0
```

```
grid .e -row 0 -column 1
```

```
grid .b -row 1 -column 0 -columnspan 2
```



Многострочные поля текстового ввода

```
#!/usr/bin/wish
```

```
text .t -width 60 -height 20 -wrap word
```

```
grid .t -row 0 -column 0
```

Вставка текста на определённую позицию:

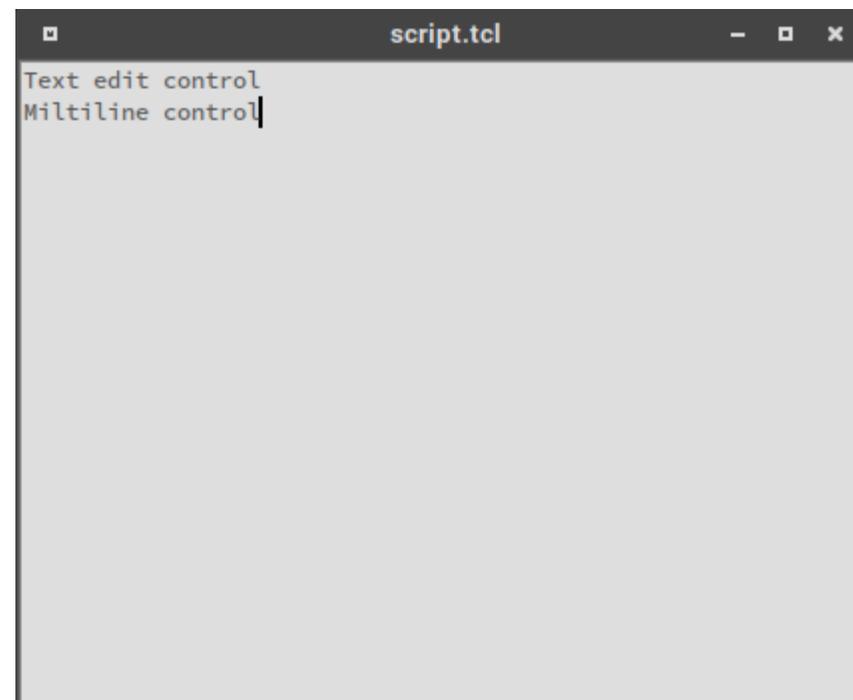
```
.t insert 0.0 "Hello"
```

Вставка текста в конец:

```
.t insert end "Hello"
```

Удаление текста:

```
.t delete 0.0 end
```



Изменение свойств элементов интерфейса

```
#!/usr/bin/wish
```

```
proc changeText {} {  
    global .b1  
    .b1 configure -text "AAA"  
}
```

```
button .b1 -text "Button" -command changeText
```

```
grid .b1 -row 0 -column 0
```



Контекст графического устройства (1)

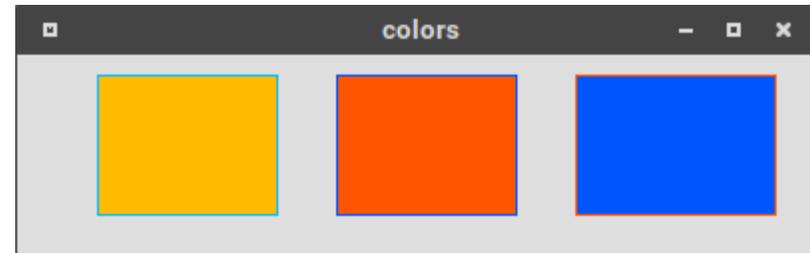
```
#!/usr/bin/wish
```

```
canvas .c -width 250 -height 100  
.c create oval 0 0 50 50 -fill orange  
.c create line 50 50 100 100 -fill blue -width 4
```

```
grid .c -row 0 -column 0
```



Контекст графического устройства (2)



```
#!/usr/bin/wish
canvas .can
.can create rect 30 10 120 80 -outline #0bf -fill #fb0
.can create rect 150 10 240 80 -outline #05f -fill #f50
.can create rect 270 10 370 80 -outline #f50 -fill #05f

pack .can

wm title . "colors"
wm geometry . 400x100+300+300
```

Контекст графического устройства (3)

```
#!/usr/bin/wish
```

```
canvas .can
```

```
.can create oval 10 10 80 80
```

```
.can create oval 110 10 210 80 -outline #777 -fill #777
```

```
.can create rect 230 10 290 60 -outline #777 -fill #777
```

```
.can create arc 30 200 90 100 -start 0 -extent 210 -outline #777 -fill #777
```

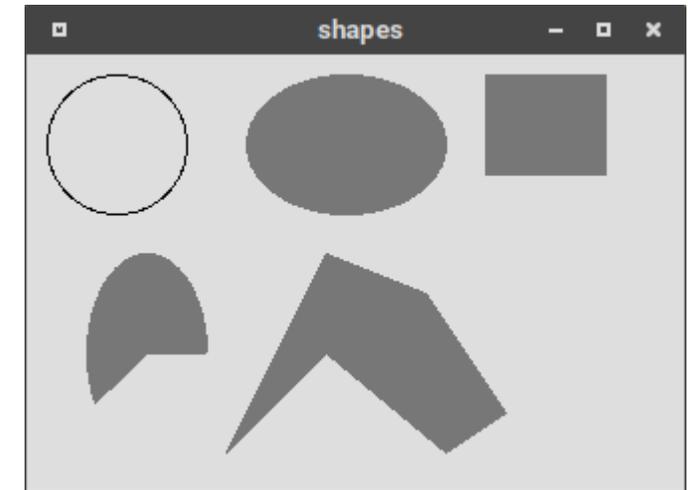
```
set points [ list 150 100 200 120 240 180 210 200 150 150 100 200 ]
```

```
.can create polygon $points -outline #777 -fill #777
```

```
pack .can
```

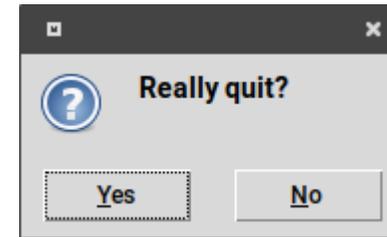
```
wm title . "shapes"
```

```
wm geometry . 330x220+300+300
```



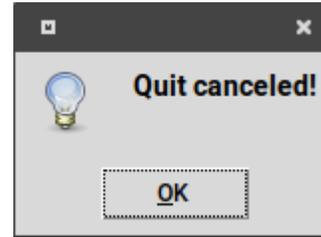
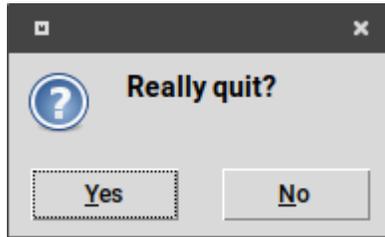
Информационные диалоги в Tk

```
tk_messageBox -message "Really quit?" \  
              -type yesno \  
              -icon question
```



```
-message - текст сообщения  
-parent  - окно, из которого вызван диалог  
-title   - заголовок  
-type    - набор кнопок      ok  
                                       okcancel  
                                       yesno  
                                       yesnocancel  
                                       retrycancel  
                                       abortretryignore  
  
-icon    - иконка сообщения  error  
                                       info  
                                       question  
                                       warning
```

Обработка результатов вызова диалогов в Tk



```
set answer [tk_messageBox -message "Really quit?" -type yesno -icon question]

switch -- $answer {
    yes exit
    no { tk_messageBox -message "Quit canceled!" -type ok}
}
```

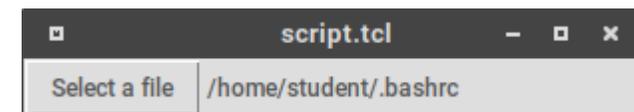
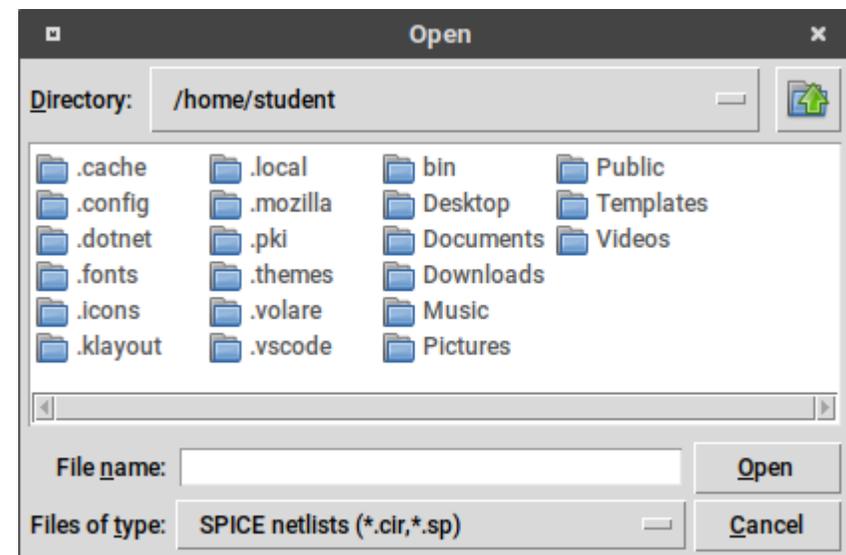
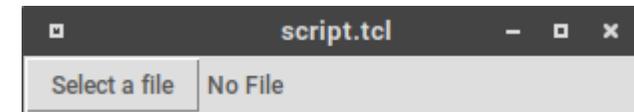
Диалог открытия файлов

```
set types {
  {"SPICE netlists" {.cir .sp} }
  {"All files"      *      }
}

proc doIt {label} {
  global types
  set file [tk_getOpenFile -filetypes $types -parent .]
  $label configure -text $file
}

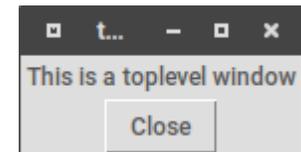
label .l -text "No File"
button .b -text "Select a file" -command "doIt .l"

grid .b -row 0 -column 0
grid .l -row 0 -column 1
```

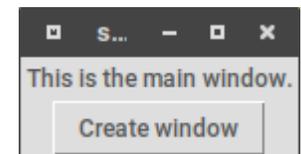


Создание дополнительных окон

```
proc makeTop { } {  
    toplevel .top  
    label .top.lab -text "This is a toplevel window"  
  
    button .top.but -text "Close" -command { destroy .top }  
    grid .top.lab -row 0  
    grid .top.but -row 1  
}
```



```
label .lab -text "This is the main window."  
button .but -text "Create window" -command { makeTop }  
grid .lab -row 0  
grid .but -row 1
```



Работа с изображениями

```
#!/usr/bin/wish
```

```
canvas .c
```

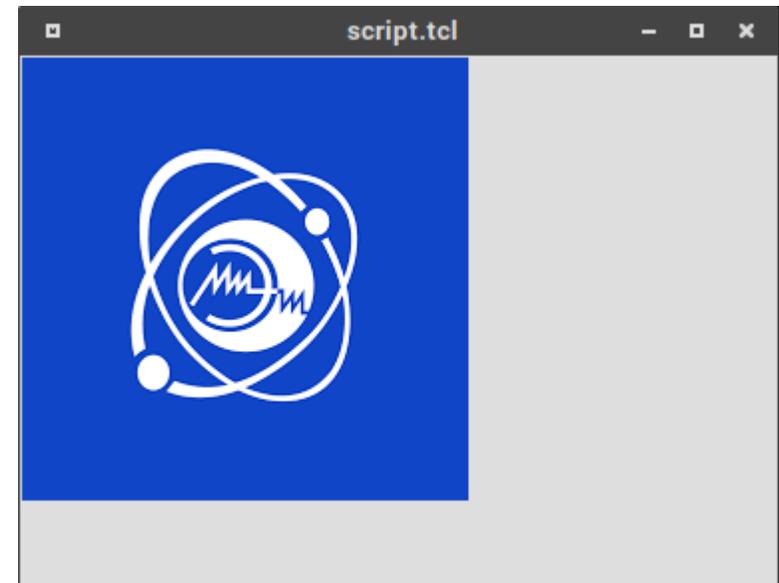
```
set picture [image create photo -file ./image.png]
```

```
set width [image width $picture]
```

```
set height [image height $picture]
```

```
.c create image [expr {$width/2}] [expr {$height/2}] -image $picture
```

```
pack .c
```



Создание меню в Tk

```
menu .menubar  
. configure -menu .menubar
```

```
menu .menubar.file  
.menubar.file add command -label "Exit" -underline 1 -command exit
```

```
.menubar add cascade -label "File" -menu .menubar.file -underline 0
```

