

Implementation file: tcl3dCgQuery.tcl

Copyright: 2005-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dCg  
Filename: tcl3dCgQuery.tcl

Author: Paul Obermeier

Description: Tcl module with query procedures related to the Cg module.

Name: tcl3dCgGetVersion - Get Cg version string.

Synopsis: tcl3dCgGetVersion {}

Description: Return the version string of the wrapped Cg library. The version is returned as "Major.Minor.Patch".

See also: tcl3dOglGetVersions  
tcl3dGetLibraryInfo

Implementation file: tcl3dCgUtil.tcl

Copyright: 2005-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dCg  
 Filename: tcl3dCgUtil.tcl

Author: Paul Obermeier

Description: Tcl module with miscellaneous utility procedures related to the Cg module.

Name: tcl3dCgResetError - Reset the Cg error condition.

Synopsis: tcl3dCgResetError { }

Description: Reset the Cg error condition.

See also: tcl3dCgGetError

Name: tcl3dCgGetError - Check for a Cg error.

Synopsis: tcl3dCgGetError { contextId { msg "" } }

Description: contextId : Cg context identifier  
 msg : Additional message string

Check, if a Cg related error has occurred. The Cg context - as returned by cgCreateContext - has to be supplied with parameter "contextId".

The procedure returns an empty string, if no error has occurred. Otherwise it returns the additional message string, the error number and Cg error message as supplied by the Cg library as a formatted string.

See also: tcl3dOglGetError

Name: tcl3dCgGetProfileList - Get a list of Cg profile names.

Synopsis: tcl3dCgGetProfileList { }

Description: Return a Tcl list of Cg profile names.

The list consists of (key,value) pairs, where key is the profile name, like CG\_PROFILE\_FP30 and value is either 1, if the corresponding profile is supported, or 0, if it is not available.

See also: tcl3dCgFindProfile  
 tcl3dCgFindProfileByNum

Name: tcl3dCgFindProfile - Find a supported Cg profile.

Synopsis: tcl3dCgFindProfile { args }

Description: args : Profile names

Find the first profile supported by the Cg implementation from the profile names supplied in "args". If successful, it returns the profile name, otherwise an empty string.

See also:            tcl3dCgGetProfileList  
                      tcl3dCgFindProfileByNum

Name:                tcl3dCgFindProfileByNum - Find a supported Cg profile.

Synopsis:            tcl3dCgFindProfileByNum { profileNum }

Description:        profileNum : int (CGprofile)

Find a profile name by it's numerical value supplied in "profileNum".  
If successful, it returns the profile name, otherwise an empty string.

Note: The procedure does not check, if the profile is supported. Use tcl3dCgFindProfile to check for support by the underlying Cg implementation.

See also:            tcl3dCgGetProfileList  
                      tcl3dCgFindProfile

Name:                tcl3dCgPrintProgramInfo - Print Cg program info.

Synopsis:            tcl3dCgPrintProgramInfo { progId { progFile "Unknown" } }

Description:        progId                : Cg Program identifier  
                      progFile            : string

Print the profile name and the name of the entry function of the Cg program identified by "progId".  
The Cg program identifier is the identifier as returned by calls to the cgCreateProgram family.  
An optional parameter "progFile" can be supplied to specify the name of the file containing the Cg program source code.

See also:

Implementation file: tcl3dDemoHeightMap.tcl

Copyright: 2005-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dOgl  
Filename: tcl3dDemoHeightMap.tcl

Author: Paul Obermeier

Description: Tcl module to create a heightmap from a photo image. This functionality is needed for NeHe tutorial 45.

Name: tcl3dDemoUtilHeightmapFromPhoto - Create a heightmap from an image.

Synopsis: tcl3dDemoUtilHeightmapFromPhoto {  
          phImg flHeightScale flResolution }

Description: phImg : string (Photo image identifier)  
              flHeightScale : float  
              flResolution : float

Create two Tcl3D vectors containing the vertices and texture coordinates of a heightmap created from the image data of photo image "phImg".  
The height values can be scaled with "flHeightScale".  
"flResolution" indicates how many pixels form a vertex.

The two vectors and the number of vertices generated are returned as a Tcl list:  
Index 0: Vertex vector  
Index 1: Texture coordinates vector  
Index 2: Number of vertices

See also:

Implementation file: tcl3dFTGLQuery.tcl

Copyright: 2007-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dFTGL  
Filename: tcl3dFTGLQuery.tcl

Author: Paul Obermeier

Description: Tcl module with with query procedures related to the FTGL module.

Name: tcl3dFTGLGetVersion - Get FTGL version string.

Synopsis: tcl3dFTGLGetVersion {}

Description: Return the version string of the wrapped FTGL library. The version is returned as "Major.Minor.Patch". If FTGL is not wrapped or supported, an empty string is returned.

Note: FTGL does not support version numbers in the code, so the version number is hand-coded here.

See also: tcl3dOglGetVersions  
tcl3dGetLibraryInfo

Implementation file: tcl3dFTGLUtil.tcl

Copyright: 2006-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dFTGL  
Filename: tcl3dFTGLUtil.tcl

Author: Paul Obermeier

Description: Tcl module with miscellaneous utility procedures related to the FTGL module.

Name: tcl3dFTGLGetBBox - Get bounding box of a string.

Synopsis: tcl3dFTGLGetBBox { font str }

Description: font : string (Font identifier)  
str : string

Return the bounding box of string "str" displayed in font "font".  
The bounding box is returned as a list of 6 values:  
{ xmin ymin zmin xmax ymax zmax }  
"font" must be an identifier as returned by one of the FTGL\*Font functions (ex. FTGLBitmapFont).

See also:

Implementation file: tcl3dGl2psQuery.tcl

Copyright: 2007-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dGl2ps  
Filename: tcl3dGl2psQuery.tcl

Author: Paul Obermeier

Description: Tcl module with query procedures related to the GL2PS module.

Name: tcl3dGl2psGetVersion - Get GL2PS version string.

Synopsis: tcl3dGl2psGetVersion {}

Description: Return the version string of the wrapped GL2PS library. The version is returned as "Major.Minor.Patch".

See also: tcl3dOglGetVersions  
tcl3dGetLibraryInfo

Implementation file: tcl3dGl2psUtil.tcl

Copyright: 2006-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dGl2ps  
Filename: tcl3dGl2psUtil.tcl

Author: Paul Obermeier

Description: Tcl module with miscellaneous utility procedures related to the GL2PS module.

Name: tcl3dGl2psCreatePdf - Create PDF from OpenGL content.

Synopsis: tcl3dGl2psCreatePdf { toglwin filename  
                          { title "Tcl3D Screenshot" }  
                          { drawBackground 0 }  
                          { producer "Tcl3D" } }

Description: toglwin : string (Togl identifier)  
              filename : string  
              title : string  
              drawBackground : boolean  
              producer : string

Create a PDF file from current Togl window content. The PDF is created from the Togl window identified by "toglwin" and written to file "filename". The following optional parameters set PDF specific values:  
"title" is the name of the document title as listed in the document properties of the PDF file. If "drawBackground" is set to true, the background color of the Togl window is also used as the background color of the PDF document. Otherwise the PDF background color is set to white.  
"procuder" is the name of the producer property as listed in the document properties of the PDF file.

See also:

Implementation file: tcl3dGuiAutoscroll.tcl

```

Copyright:      2005-2010 Paul Obermeier (obermeier@tcl3d.org)
                2003 Kevin B Kenny <kennykb@users.sourceforge.net>

                See the file "Tcl3D_License.txt" for information on
                usage and redistribution of this file, and for a
                DISCLAIMER OF ALL WARRANTIES.

Module:         Tcl3D -> tcl3dOgl
Filename:      tcl3dGuiAutoscroll.tcl

Author:        Paul Obermeier

Description:    Tcl module to create scroll bars that automatically
                appear when a window is too small to display its
                content.

Name:          tcl3dAutoscroll::autoscroll - Create auto scrollbar

Synopsis:      tcl3dAutoscroll::autoscroll { w }

Description:    w : string (Widget name)

                Create a scroll bar that disappears when it is not
                needed, and reappears when it is.
                The scroll bar "w" should already exist.

                Side effects:
                The widget command is renamed, so that the 'set' command
                can be intercepted and determine whether the widget
                should appear.
                In addition, the 'Autoscroll' bind tag is added to the
                widget, so that the <Destroy> event can be intercepted.

See also:      tcl3dAutoscroll::unautoscroll

Name:          tcl3dAutoscroll::unautoscroll - Remove scrollbar from
                package control.

Synopsis:      tcl3dAutoscroll::unautoscroll { w }

Description:    w : string (Widget name)

                Return a scrollbar to its normal static behavior by
                removing it from the control of this package.

                "w" is the path name of the scroll bar, which must have
                previously had tcl3dAutoscroll::autoscroll called on it.

                Side effects:
                The widget command is renamed to its original name.
                The widget is mapped if it was not currently displayed.
                The widgets bindtags are returned to their original state.
                Internal memory is cleaned up.

See also:      tcl3dAutoscroll::autoscroll

Name:          tcl3dAutoscroll::widgetCommand - Apply a widget command.

Synopsis:      tcl3dAutoscroll::widgetCommand { w command args }

Description:    w          : string (Widget name)
                command   : string
                args       : argument list

                Apply widget command "command" on 'autoscroll' scrollbar
                "w". Arguments to the commands can be supplied in "args".

```

Return whatever the widget command returns.

Side effects:

Has whatever side effects the widget command has. In addition, the 'set' widget command is handled specially, by gridding/packing the scroll bar according to whether it is required.

See also:

Name: tcl3dAutoscroll::destroyed - Destroy callback.

Synopsis: tcl3dAutoscroll::destroyed { w }

Description: w : string (Widget name)

Callback executed when automatic scroll bar "w" is destroyed.

Side effects:

Cleans up internal memory.

See also: tcl3dAutoscroll::map

Name: tcl3dAutoscroll::map - Mapping callback.

Synopsis: tcl3dAutoscroll::map { w }

Description: w : string (Widget name)

Callback executed when automatic scroll bar "w" is mapped.

Side effects:

Geometry of scroll bar's top-level window is constrained.

This procedure keeps the top-level window associated with an automatic scroll bar from being resized automatically after the scroll bar is mapped. This effect avoids a potential endless loop in the case where the resize of the top-level window resizes the widget being scrolled, causing the scroll bar no longer to be needed.

See also: tcl3dAutoscroll::destroyed

Name: tcl3dAutoscroll::wrap - Autoscroll all new scrollbars.

Synopsis: tcl3dAutoscroll::wrap {}

Description: Arrange for all new scrollbars to be automatically autoscrollled.

Side effects:

::scrollbar is overloaded to automatically autoscroll any new scrollbars.

See also: tcl3dAutoscroll::unwrap

Name: tcl3dAutoscroll::unwrap - Turn off automatic autoscrolling of new scrollbars.

Synopsis: tcl3dAutoscroll::unwrap {}

Description: Turn off automatic autoscrolling of new scrollbars. Does not effect existing scrollbars.

Side effects:

::scrollbar is returned to its original state

See also: `tcl3dAutoscroll::wrap`

Implementation file: tcl3dGuiConsole.tcl

Copyright: 2006-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dOgl  
Filename: tcl3dGuiConsole.tcl

Author: Paul Obermeier

Description: Tcl module implementing a simple Tk console. It is used in the tcl3dsh, the Tcl3D Starpack. The implementation of this console window was taken from D. Richard Hipp's mktclapp.

Name: tcl3dConsoleCreate - Create a console window.

Synopsis: tcl3dConsoleCreate { w prompt title }

Description: w : string (Widget name)  
prompt : string  
title : string

Create a new console window "w". The window's title will be set to "title". The prompt inside the console's text widget will be set to "prompt".

Example:  
tcl3dConsoleCreate .myConsole "tcl3d> " "Tcl3D Console"

See also:

Implementation file: tcl3dGuiToolhelp.tcl

```

Copyright:      2005-2010 Paul Obermeier (obermeier@tcl3d.org)

                See the file "Tcl3D_License.txt" for information on
                usage and redistribution of this file, and for a
                DISCLAIMER OF ALL WARRANTIES.

Module:         Tcl3D -> tcl3dOgl
Filename:      tcl3dGuiToolhelp.tcl

Author:        Paul Obermeier

Description:    Tcl module implementing a simple tool help widget.

Name:          tcl3dToolhelpInit - Initialize toolhelp module.

Synopsis:      tcl3dToolhelpInit { w { bgColor yellow }
                { fgColor black } }

Description:    w          : string (Widget name)
                bgColor   : string
                fgColor   : string

                Initialize the toolhelp module.
                The initialization function only needs to be called
                when non-standard background and foreground colors
                are needed.

See also:      tcl3dToolhelpAddBinding

Name:          tcl3dToolhelpShow - Display toolhelp message.

Synopsis:      tcl3dToolhelpShow { x y str }

Description:    x      : int
                y      : int
                str    : string

                Display the toolhelp window at widget relative
                coordinates (x, y) with message string "str".

                A typical usage is like follows:
                bind $w <Enter> "tcl3dToolhelpShow %X %Y [list $str]"

See also:      tcl3dToolhelpHide
                tcl3dToolhelpAddBinding

Name:          tcl3dToolhelpHide - Hide toolhelp message.

Synopsis:      tcl3dToolhelpHide {}

Description:    Hide the toolhelp message window.

See also:      tcl3dToolhelpShow
                tcl3dToolhelpAddBinding

Name:          tcl3dToolhelpAddBinding - Add binding for a toolhelp
                message.

Synopsis:      tcl3dToolhelpAddBinding { w str }

Description:    w      : string (Widget name)
                str    : string

                Add bindings to widget "w" to display message string
                "str" in a toolhelp window near the widget. The

```

toolhelp window is shown, when the mouse enters the widget and unmapped, when the mouse leaves the widget.

See also: `tcl3dToolhelpInit`

Implementation file: tcl3dGuiWidgets.tcl

```

Copyright:      2005-2010 Paul Obermeier (obermeier@tcl3d.org)

                See the file "Tcl3D_License.txt" for information on
                usage and redistribution of this file, and for a
                DISCLAIMER OF ALL WARRANTIES.

Module:         Tcl3D -> tcl3dOgl
Filename:       tcl3dGuiWidgets.tcl

Author:         Paul Obermeier

Description:     Tcl module implementing some simple Tk widgets like
                scrolled listboxes and text widgets, as well as
                some widget and window handling utilities.

Name:           tcl3dHaveAqua - Check, if windowing system is Aqua.

Synopsis:       tcl3dHaveAqua {}

Description:     Return true, if the windowing system is Apple's Aqua.
                Otherwise return false.

See also:       tcl3dShowIndicator

Name:           tcl3dShowIndicator - Check, if button indicators
                should be shown.

Synopsis:       tcl3dShowIndicator {}

Description:     Return true, if we want to show the indicators for
                radio- and checkbuttons. Currently we do this on a Mac
                running Aqua, because it looks very buggy otherwise.

See also:       tcl3dHaveAqua

Name:           tcl3dAfterIdle - Workaround for after idle command.

Synopsis:       tcl3dAfterIdle { cmd args }

Description:     cmd : string
                args: Variable argument list

                Command "cmd" with optional arguments "args" is executed
                in an "after idle" callback.
                The after idle callbacks are typically used in Tcl3D
                scripts to implement animations.
                On the Mac the after idle usage causes problems
                (freezing the application) when resizing the window.
                This procedure implements a workaround for these Mac
                related problems. For all other platforms this procedure
                is just a wrapper for the standard "after idle" command.

                Return the callback id, which can be used in an
                "after cancel" callback.

See also:

Name:           tcl3dAddEvents - Add virtual events.

Synopsis:       tcl3dAddEvents {}

Description:     Add the following virtual events for cross-platform
                mouse event handling:
                <<LeftMousePress>>
                <<MiddleMousePress>>

```

```
<<RightMousePress>>
```

See also:

Name: tcl3dWinIsTop - Check, if widget is a top level window.

Synopsis: tcl3dWinIsTop { wid }

Description: wid : string

Return true, if widget "wid" is a top level window.

See also: tcl3dWinRaise

Name: tcl3dWinRaise - Raise a widget.

Synopsis: tcl3dWinRaise { wid }

Description: wid : string

Raise widget "wid" to the top of the widget layout hierarchy.

See also: tcl3dWinIsTop

Name: tcl3dSetFullScreenMode - Put a widget into fullscreen mode.

Synopsis: tcl3dSetFullScreenMode { wid }

Description: wid : string

Put widget "wid" into fullscreen mode:  
It's size is adjusted to fit the entire screen, window decoration is removed and the widget can not be resized.

See also: tcl3SetWindowMode

Name: tcl3dSetWindowMode - Put a widget into windowing mode.

Synopsis: tcl3dSetWindowMode { wid w h }

Description: wid : string

w : int  
h : int

Put widget "wid" into windowing mode:  
It's size is adjusted to width "w" and height "h", window decoration is enabled and the widget can be resized.

See also: tcl3dSetFullScreenMode

Name: tcl3dSetScrolledTitle - Set the title of a scrolled widget.

Synopsis: tcl3dSetScrolledTitle { wid titleStr  
                                  { fgColor "black" } }

Description: wid : string  
          titleStr : string  
          fgColor : string

Set the title of scrolled widget "wid" to string "titleStr". The text color can be optionally specified with "fgColor". "fgColor" must be a valid Tk color name.

"wid" must be a widget name returned from  
tcl3dCreateScrolledWidget or descendants.

See also: tcl3dCreateScrolledWidget

Name: tcl3dCreateScrolledWidget - Create a scrolled widget.

Synopsis: tcl3dCreateScrolledWidget { wType wid titleStr args }

Description: wType : string  
wid : string  
titleStr : string  
args : list

Create a compound widget with horizontal and vertical  
scrollbars. The type of the widget is given with "wType"  
and must be a valid Tk widget like canvas or text.

"wid" is the parent frame of the compound widget and  
must already exist.

The compound widget may have a title string, which is  
given with "titleStr". If "titleStr" is an empty string,  
no title label will be generated.

With optional parameter "args" additional widget  
specific parameters may be supplied.

Return the identifier to the created master widget.

There exist several utility procedures for often used  
Tk widget types. See list below.

See also: tcl3dSetScrolledTitle  
tcl3dCreateScrolledFrame  
tcl3dCreateScrolledListbox  
tcl3dCreateScrolledText  
tcl3dCreateScrolledCanvas  
tcl3dCreateScrolledTable  
tcl3dCreateScrolledTablelist

Name: tcl3dCreateScrolledFrame - Create a scrolled  
frame widget.

Synopsis: tcl3dCreateScrolledFrame { wid titleStr args }

Description: wid : string  
titleStr : string  
args : list

Create a scrolled frame widget. "wid" specifies the  
parent frame of the created scrolled widget. "titleStr"  
specifies the string displayed as widget title.

With optional parameter "args" additional widget  
specific parameters may be supplied.

Return the identifier to the created frame widget.

See also: tcl3dCreateScrolledWidget  
tcl3dSetScrolledTitle

Name: tcl3dCreateScrolledListbox - Create a scrolled  
listbox widget.

Synopsis: tcl3dCreateScrolledListbox { wid titleStr args }

Description: wid : string  
titleStr : string  
args : list

Create a scrolled listbox widget. "wid" specifies the  
parent frame of the created scrolled widget. "titleStr"  
specifies the string displayed as widget title.

With optional parameter "args" additional widget

specific parameters may be supplied.  
Return the identifier to the created listbox widget.

See also: `tcl3dCreateScrolledWidget`  
`tcl3dSetScrolledTitle`

Name: `tcl3dCreateScrolledText` - Create a scrolled text widget.

Synopsis: `tcl3dCreateScrolledText { wid titleStr args }`

Description: `wid` : string  
`titleStr` : string  
`args` : list

Create a scrolled text widget. "wid" specifies the parent frame of the created scrolled widget. "titleStr" specifies the string displayed as widget title. With optional parameter "args" additional widget specific parameters may be supplied. Return the identifier to the created text widget.

See also: `tcl3dCreateScrolledWidget`  
`tcl3dSetScrolledTitle`

Name: `tcl3dCreateScrolledCanvas` - Create a scrolled canvas widget.

Synopsis: `tcl3dCreateScrolledCanvas { wid titleStr args }`

Description: `wid` : string  
`titleStr` : string  
`args` : list

Create a scrolled canvas widget. "wid" specifies the parent frame of the created scrolled widget. "titleStr" specifies the string displayed as widget title. With optional parameter "args" additional widget specific parameters may be supplied. Return the identifier to the created canvas widget.

See also: `tcl3dCreateScrolledWidget`  
`tcl3dSetScrolledTitle`

Name: `tcl3dCreateScrolledTable` - Create a scrolled tktable widget.

Synopsis: `tcl3dCreateScrolledTable { wid titleStr args }`

Description: `wid` : string  
`titleStr` : string  
`args` : list

Create a scrolled TkTable widget. "wid" specifies the parent frame of the created scrolled widget. "titleStr" specifies the string displayed as widget title. With optional parameter "args" additional widget specific parameters may be supplied. Return the identifier to the created TkTable widget.

See also: `tcl3dCreateScrolledWidget`  
`tcl3dSetScrolledTitle`

Name: `tcl3dCreateScrolledTablelist` - Create a scrolled tablelist widget.

Synopsis: `tcl3dCreateScrolledTablelist { wid titleStr args }`  
Description: `wid` : string

```
titleStr  : string  
args      : list
```

Create a scrolled tablelist widget. "wid" specifies the parent frame of the created scrolled widget. "titleStr" specifies the string displayed as widget title. With optional parameter "args" additional widget specific parameters may be supplied. Return the identifier to the created tablelist widget.

Note: A "package require tablelist" must be issued before using this function.

See also: `tcl3dCreateScrolledWidget`  
`tcl3dSetScrolledTitle`

Implementation file: tcl3dOdeQuery.tcl

Copyright: 2007-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dOde  
Filename: tcl3dOdeQuery.tcl

Author: Paul Obermeier

Description: Tcl module with query procedures related to the ODE module.

Name: tcl3dOdeGetVersion - Get ODE version string.

Synopsis: tcl3dOdeGetVersion {}

Description: Return the version string of the wrapped ODE library. The version is returned as "Major.Minor.Patch".

Note: ODE does not support version numbers in the code, so the version number is hand-coded here.

See also: tcl3dOglGetVersions  
tcl3dGetLibraryInfo

Implementation file: tcl3dOglFormats.tcl

Copyright: 2009-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dOgl  
Filename: tcl3dOglFormats.tcl

Author: Paul Obermeier

Description: Tcl module with procedures related to image and texture formats and types.

Name: tcl3dOglGetFormatList - Get OpenGL formats.

Synopsis: tcl3dOglGetFormatList { { patt \* } }

Description: Return a sorted list of OpenGL format names.

See also:

Implementation file: tcl3dOglHelp.tcl

Copyright: 2009-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dOgl  
 Filename: tcl3dOglHelp.tcl

Author: Paul Obermeier

Description: Tcl module with help and information procedures related to the OpenGL module.

Name: tcl3dOglGetFuncList - Get list of OpenGL and GLU functions.

Synopsis: tcl3dOglGetFuncList { what }  
 what : string (Default "gl")  
 Allowed values for what ar: "gl", "glu" or "all"

Description: Return a list of all wrapped OpenGL and/or GLU function names.

See also: tcl3dOglGetVersionList  
 tcl3dOglIsFuncWrapped  
 tcl3dOglGetVersionFuncs  
 tcl3dOglGetVersionEnums

Name: tcl3dOglGetVersionList - Get list of OpenGL versions and extensions.

Synopsis: tcl3dOglGetVersionList {}

Description: Return a list of all wrapped OpenGL versions and extension names.  
 The names are strings identical to their corresponding C defines.  
 Examples:  
 GL versions : GL\_VERSION\_1\_5, GL\_VERSION\_3\_2  
 GL extensions: GL\_ARB\_vertex\_program, GL\_EXT\_texture3D

See also: tcl3dOglGetFuncList  
 tcl3dOglGetFuncVersion  
 tcl3dOglGetVersionFuncs  
 tcl3dOglGetVersionEnums

Name: tcl3dOglGetExtensionList - Get list of OpenGL extensions.

Synopsis: tcl3dOglGetExtensionList { {what "all"} }

Description: Return a list of OpenGL extension names.  
 The names are strings identical to their corresponding C defines.  
 Examples:GL\_ARB\_vertex\_program, GL\_EXT\_texture3D

If "what" is equal to "all", all OpenGL extension names are returned.  
 If "what" is equal to "glew", only the OpenGL extension names wrapped by GLEW are returned.  
 If "what" is equal to "driver", only the OpenGL extension names supported by the driver and hardware of the actual machine are returned.

See also: tcl3dOglGetFuncList

```
tcl3dOglGetFuncVersion
tcl3dOglGetVersionList
tcl3dOglGetVersionFuncs
tcl3dOglGetVersionEnums
```

Name: tcl3dOglIsFuncWrapped - Check if OpenGL or GLU function is wrapped.

Synopsis: tcl3dOglIsFuncWrapped { func }

Description: Return true, if OpenGL or GLU function "func" is wrapped in Tcl3D. Otherwise return false.

Note: To check, if a function is supported by the used OpenGL driver, use procedure "tcl3dOglHaveFunc".

See also: tcl3dOglGetFuncList  
tcl3dOglGetFuncSignature  
tcl3dOglGetFuncVersion  
tcl3dOglGetFuncDeprecated  
tcl3dOglGetUrl

Name: tcl3dOglGetFuncSignature - Get the signature of an OpenGL or GLU function.

Synopsis: tcl3dOglGetFuncSignature { func {what "c"} }

Description: Return the signature of OpenGL or GLU function "func" as a string.  
If "func" is not contained in the list of wrapped OpenGL functions (see tcl3dOglGetFuncList), an empty string is returned.

If "what" is equal to "c", the signature string is returned in C style notation. This is the default case.  
If "what" is equal to "tcl", the signature string is returned in Tcl style notation.

Note: This procedure replaces the obsolete tcl3dOglGetFuncSignatureList procedure.

See also: tcl3dOglGetFuncList  
tcl3dOglGetFuncVersion  
tcl3dOglGetFuncDeprecated  
tcl3dOglGetUrl

Name: tcl3dOglGetFuncVersion - Get the version or extension name of an OpenGL function.

Synopsis: tcl3dOglGetFuncVersion { func }

Description: Return the version or extension name of OpenGL function "func" as a string.  
If "func" is not contained in the list of wrapped OpenGL functions (see tcl3dOglGetFuncList), an empty string is returned.

Note: This procedure replaces the obsolete tcl3dOglGetFuncVersionList procedure.

See also: tcl3dOglGetFuncList  
tcl3dOglGetFuncSignature  
tcl3dOglGetFuncDeprecated  
tcl3dOglGetUrl  
tcl3dOglGetEnumVersion

Name: tcl3dOglGetEnumVersion - Get the version or extension name of an OpenGL enumeration.

Synopsis: `tcl3dOglGetEnumVersion { enum }`

Description: Return the version or extension name of OpenGL enumeration "enum" as a string. If "enum" is not a wrapped OpenGL enumeration, an empty string is returned.

See also: `tcl3dOglGetVersionList`  
`tcl3dOglGetVersionFuncs`  
`tcl3dOglGetVersionEnums`  
`tcl3dOglGetFuncVersion`

Name: `tcl3dOglGetFuncDeprecated` - Get the OpenGL version, an OpenGL function has been declared deprecated.

Synopsis: `tcl3dOglGetFuncDeprecated { func }`

Description: Return the version when OpenGL function "func" has been declared deprecated. The result is in the format "3.1", "3.2". For non-deprecated functions "0.0" is returned.

If "func" is not contained in the list of wrapped OpenGL functions (see `tcl3dOglGetFuncList`), an empty string is returned.

See also: `tcl3dOglGetFuncList`  
`tcl3dOglGetFuncSignature`  
`tcl3dOglGetFuncVersion`  
`tcl3dOglGetUrl`

Name: `tcl3dOglGetUrl` - Get the URL of the official documentation of an OpenGL item.

Synopsis: `tcl3dOglGetUrl { item }`

Description: Return the URL of the official documentation of OpenGL item "item" as a string. Item can be the name of a function, extension or enumeration. If no documentation page exists, a Google search URL for that item is returned.

Note: The documentation pages on [www.opengl.org](http://www.opengl.org) currently only include OpenGL up to version 2.1.

See also: `tcl3dOglGetFuncList`  
`tcl3dOglGetUrl`  
`tcl3dOglGetVersionFuncs`  
`tcl3dOglGetVersionEnums`

Name: `tcl3dOglGetVersionFuncs` - Get the function names of an OpenGL version or extension.

Synopsis: `tcl3dOglGetVersionFuncs { version }`

Description: Return the function names of OpenGL version or extension "version" as a list. If "version" is not a supported version or extension, an empty list is returned.

See also: `tcl3dOglGetFuncList`  
`tcl3dOglGetVersionList`  
`tcl3dOglGetFuncVersion`  
`tcl3dOglGetVersionEnums`

Name: `tcl3dOglGetVersionEnums` - Get the enumeration names of

an OpenGL version or extension.

Synopsis: `tcl3dOglGetVersionEnums { version }`

Description: Return the enumeration names of OpenGL version or extension "version" as a list. If "version" is not a supported version or extension, an empty list is returned.

See also: `tcl3dOglGetFuncList`  
`tcl3dOglGetVersionList`  
`tcl3dOglGetEnumVersion`  
`tcl3dOglGetVersionFuncs`

Implementation file: tcl3dOglQuery.tcl

Copyright: 2007-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dOgl  
Filename: tcl3dOglQuery.tcl

Author: Paul Obermeier

Description: Tcl module with query procedures related to the OpenGL module.

Name: tcl3dOglGetVersion - Get OpenGL version string.

Synopsis: tcl3dOglGetVersion {}

Description: Return the version string of the wrapped OpenGL library. The version string does not have a specific format. It depends on the vendor of the OpenGL implementation. Some examples:  
1.4 APPLE-1.6.18  
2.1.2 NVIDIA 173.14.12

If no OpenGL context has been established (i.e. a Togl window has not been created), the function returns an empty string.

See also: tcl3dOglGetVersions  
tcl3dGetLibraryInfo  
tcl3dOglHaveVersion

Name: tcl3dOglGetVersionNumber - Get OpenGL version number.

Synopsis: tcl3dOglGetVersionNumber {}

Description: Return the OpenGL version number as a dictionary containing the following elements:  
major: int  
minor: int  
patch: int

If a component of the OpenGL version number is not supplied by the OpenGL driver, the corresponding element is set to -1.

Note: The version number of the OpenGL implementation is extracted from the string returned by calling "glGetString GL\_VERSION". As some vendors format the version in an unusual way, this function may not work correctly on all platforms.

If no OpenGL context has been established (i.e. a Togl window has not been created), the function returns an empty dictionary.

See also: tcl3dOglGetVersions  
tcl3dOglGetVersion  
tcl3dOglHaveVersion

Name: tcl3dOglGetGlewVersion - Get GLEW version string.

Synopsis: tcl3dOglGetGlewVersion {}

Description: Return the version string of the GLEW wrapper library.

The version is returned as "Major.Minor.Patch".

If no OpenGL context has been established (i.e. a Togl window has not been created), the function returns an empty string.

See also: `tcl3dOglGetVersion`  
`tcl3dOglGetVersions`

Name: `tcl3dOglHaveFunc` - Check availability of a specific OpenGL function.

Synopsis: `tcl3dOglHaveFunc { glFuncName }`

Description: `glFuncName` : string

Return 1, if the OpenGL function "glFuncName" is provided by the underlying OpenGL implementation. Otherwise return 0.

Example: `tcl3dOglHaveFunc glGenQueriesARB`  
checks the availability of the occlusion query related ARB extension function `glGenQueriesARB`.

Note: A Togl window (and therefore a graphics context) must have been created before issuing a call to this function.

See also: `tcl3dOglHaveExtension`

Name: `tcl3dOglHaveExtension` - Check availability of a specific OpenGL extension.

Synopsis: `tcl3dOglHaveExtension { extensionName }`

Description: `extensionName` : string

Return 1, if the OpenGL extension "extensionName" is provided by the underlying OpenGL implementation. Otherwise return 0.

Example: `tcl3dOglHaveExtension GL_ARB_multitexture`  
checks the availability of the multitexturing extension.

Note: A Togl window (and therefore a graphics context) must have been created before issuing a call to this function.

See also: `tcl3dOglGetExtensions`

Name: `tcl3dOglHaveVersion` - Check availability of a specific OpenGL version.

Synopsis: `tcl3dOglHaveVersion { majorWanted { minorWanted -1 } { patchWanted -1 } }`

Description: `majorWanted` : int  
`minorWanted` : int  
`patchWanted` : int

Description: Return 1, if the OpenGL version offered by the driver is equal to or greater than the supplied major, minor and patch level numbers. Otherwise return 0.

Note: The version number of the OpenGL implementation is extracted from the string returned by calling `"glGetString GL_VERSION"`. As some vendors format

the version in an unusual way, this function may not work correctly on all platforms.

Note: A Togl window (and therefore a graphics context) must have been created before issuing a call to this function.

See also: `tcl3dOglGetVersions`

Name: `tcl3dOglGetVersions` - Get OpenGL version information.

Synopsis: `tcl3dOglGetVersions {}`

Description: Return OpenGL version information as a list of (key,value) pairs.  
Keys are the following OpenGL version types:  
`GL_VENDOR`, `GL_RENDERER`, `GL_VERSION`, `GLU_VERSION`,  
`GL_SHADING_LANGUAGE_VERSION`, `GLEW_VERSION`.  
Values are the corresponding version strings as returned by the underlying OpenGL implementation.

Example:  
`{GL_VENDOR {Intel Inc.}}`  
`{GL_RENDERER {Intel GMA 950 OpenGL Engine}}`  
`{GL_VERSION {1.2 APPLE-1.4.56}}`  
`{GLU_VERSION {1.3 MacOSX}}`

Note: A Togl window (and therefore a graphics context) must have been created before issuing a call to this function.

See also: `tcl3dOglHaveVersion`  
`tcl3dOglGetExtensions`

Name: `tcl3dOglGetExtensions` - Get all supported OpenGL extensions.

Synopsis: `tcl3dOglGetExtensions {}`

Description: Return a two-element list containing OpenGL extension information. The first sub-list contains all OpenGL extensions, the second sub-list contains all GLU extensions supported by the OpenGL implementation.

Note: A Togl window (and therefore a graphics context) must have been created before issuing a call to this function.

See also: `tcl3dOglHaveExtension`  
`tcl3dOglGetVersions`

Name: `tcl3dOglGetIntState` - Get OpenGL state variable.

Synopsis: `tcl3dOglGetIntState { state { numVals 1 } }`

Description: `state` : `GLenum`  
`numVals` : `int`

Utility function to query an integer OpenGL state variable with `glGetIntegerv`.  
The state variable to be queried is specified as an `GLenum` in parameter "state".

The value of the state variable is returned as an integer scalar value, if "numVals" is 1. If "numVals" is greater than 1, a Tcl list is returned.

Note: See Appendix B of the OpenGL Red Book for a list of state variables.

See also: `tcl3dOglGetFloatState`  
`tcl3dOglGetDoubleState`

Name: `tcl3dOglGetFloatState` - Get OpenGL state variable.

Synopsis: `tcl3dOglGetFloatState { state { numVals 1 } }`

Description: `state` : GLenum  
`numVals` : int

Utility function to query a 32-bit floating point OpenGL state variable with `glGetFloatv`. The state variable to be queried is specified as an GLenum in parameter "state".

The value of the state variable is returned as a float scalar value, if "numVals" is 1. If "numVals" is greater than 1, a Tcl list is returned.

Note: See Appendix B of the OpenGL Red Book for a list of state variables.

See also: `tcl3dOglGetIntState`  
`tcl3dOglGetDoubleState`

Name: `tcl3dOglGetDoubleState` - Get OpenGL state variable.

Synopsis: `tcl3dOglGetDoubleState { state { numVals 1 } }`

Description: `state` : GLenum  
`numVals` : int

Utility function to query a 64-bit floating point OpenGL state variable with `glGetDoublev`. The state variable to be queried is specified as an GLenum in parameter "state".

The value of the state variable is returned as a double scalar value, if "numVals" is 1. If "numVals" is greater than 1, a Tcl list is returned.

Note: See Appendix B of the OpenGL Red Book for a list of state variables.

See also: `tcl3dOglGetIntState`  
`tcl3dOglGetFloatState`

Name: `tcl3dOglGetMaxTextureSize` - Get maximum texture size.

Synopsis: `tcl3dOglGetMaxTextureSize {}`

Description: Utility function to get maximum size of a texture. The maximum texture size is returned as integer value. This function corresponds to querying state variable `GL_MAX_TEXTURE_SIZE`.

See also: `tcl3dOglGetIntState`  
`tcl3dOglGetMaxTextureUnits`

Name: `tcl3dOglGetMaxTextureUnits` - Get maximum texture units.

Synopsis: `tcl3dOglGetMaxTextureUnits {}`

Description: Utility function to get maximum number of texture units. The maximum number of texture units is returned as an integer value. This function corresponds to querying state variable `GL_MAX_TEXTURE_UNITS`.

See also: `tcl3dOglGetIntState`  
`tcl3dOglGetMaxTextureSize`

Name: `tcl3dOglGetViewport` - Get current viewport.

Synopsis: `tcl3dOglGetViewport {}`

Description: Utility function to get the current viewport.  
 The viewport is returned as a 4-element Tcl list:  
`{ LowerLeftX LowerLeftY Width Height }`  
 This function corresponds to querying state variable  
`GL_VIEWPORT`.

See also: `tcl3dOglGetIntState`

Name: `tcl3dOglGetProfile` - Get OpenGL profile settings.

Synopsis: `tcl3dOglGetProfile {}`

Description: Utility function to get the currently available OpenGL  
 profile.

The wished profile (Core or Compatibility profile,  
 OpenGL major and minor version) are set when creating a  
 Togl window with the following command line options:  
`"-coreprofile bool" "-major int" "-minor int"`

As the wished combination might not be available with  
 the installed OpenGL driver, the following situations  
 can occur:

1. An error is generated at Togl creation time.
2. A compatibility profile is automatically selected by  
 the driver.

To check for the second case, issue a call to  
`tcl3dOglGetProfile` after Togl creation to check, if your  
 wished profile has been established.

The procedure returns a dictionary with the following  
 entries:

```
coreprofile true|false
major      int
minor      int
```

Note: A Togl window (and therefore a graphics context)  
 must have been created before issuing a call to  
 this function.

See also: `tcl3dOglHaveVersion`  
`tcl3dOglGetVersionNumber`  
`tcl3dOglGetIntState`

Name: `tcl3dOglGetExtSuffixes` - Get OpenGL extension suffixes.

Synopsis: `tcl3dOglGetExtSuffixes {}`

Description: Return a list of all OpenGL extension suffixes.  
 Currently these are:  
`"ARB" "EXT" "NV" "ATI" "SGI" "SGIX" "SGIS"`  
`"SUN" "WIN" "MESA" "INTEL" "IBM" "HP"`

See also: `tcl3dOglGetExtensions`

Name: `tcl3dOglFindFunc` - Find an OpenGL function.

Synopsis: `tcl3dOglFindFunc { glFunc }`

Description: Return the name of an OpenGL function implemented in the available OpenGL driver. First it is checked, if the function is available as a native implementation. If the OpenGL version does not supply the function, all possible extension names are checked in the order as returned by `tcl3dOglGetExtSuffixes`. If none of these checks succeed, an empty string is returned.

See also: `tcl3dOglGetExtSuffixes`

Name: `tcl3dOglGetStates` - Get OpenGL state variables.

Synopsis: `tcl3dOglGetStates { {sortFlag "none"} }`

Description: `sortFlag : string (increasing|decreasing|none)`

Query all state variables of the OpenGL library and return the results as a list of sub-lists. Each sublist contains a flag indicating the success of the query, the query command used, the key and the value(s).

Note: This function is still incomplete. Chances are high, it will never be finished.

This function has been declared obsolete with Tcl3D version 0.4.2.

See also: `tcl3dOglGetExtensions`  
`tcl3dOglGetVersions`

Implementation file: tcl3dOglShaderUtil.tcl

```

Copyright:      2010 Paul Obermeier (obermeier@tcl3d.org)

                See the file "Tcl3D_License.txt" for information on
                usage and redistribution of this file, and for a
                DISCLAIMER OF ALL WARRANTIES.

Module:        Tcl3D -> tcl3dOgl
Filename:      tcl3dOglShaderUtil.tcl

Author:        Paul Obermeier

Description:   Tcl module with miscellaneous utility
                procedures related to shader programming.

Name:          tcl3dOglGetShaderState - Get shader parameter status.

Synopsis:      tcl3dOglGetShaderState { shader status }

Description:   shader : Shader object
                status : OpenGL enumeration.
                GL_SHADER_TYPE
                GL_DELETE_STATUS
                GL_COMPILE_STATUS
                GL_INFO_LOG_LENGTH
                GL_SHADER_SOURCE_LENGTH

                Utility function for easier use of OpenGL function
                glGetShaderiv.
                Given the shader object (as returned by function
                glCreateShader), the function returns the
                value of the specified status parameter.

See also:      tcl3dOglCompileProgram
                tcl3dOglGetProgramState

Name:          tcl3dOglGetProgramState - Get program parameter status.

Synopsis:      tcl3dOglGetProgramState { program status }

Description:   program : Program object
                status : OpenGL enumeration.
                GL_DELETE_STATUS
                GL_LINK_STATUS
                GL_VALIDATE_STATUS
                GL_INFO_LOG_LENGTH
                GL_ATTACHED_SHADERS
                GL_ACTIVE_ATTRIBUTES
                GL_ACTIVE_ATTRIBUTE_MAX_LENGTH
                GL_ACTIVE_UNIFORMS
                GL_ACTIVE_UNIFORM_MAX_LENGTH

                Utility function for easier use of OpenGL function
                glGetProgramiv.
                Given the program object (as returned by function
                glCreateProgram), the function returns the
                value of the specified status parameter.

See also:      tcl3dOglLinkProgram
                tcl3dOglGetShaderState

Name:          tcl3dOglGetInfoLogARB - Get shader object log.

Synopsis:      tcl3dOglGetInfoLogARB { object }

Description:   object : Shader object

```

Utility function for easier use of OpenGL function `glGetInfoLogARB`.  
 Given the shader object (as returned by function `glCreateProgramObjectARB`), the function returns the information log message as a Tcl string.

See also: `tcl3dOglGetShaderInfoLog`  
`tcl3dOglGetProgramInfoLog`  
`tcl3dOglGetShaderSource`

Name: `tcl3dOglGetShaderInfoLog` - Get shader object log.

Synopsis: `tcl3dOglGetShaderInfoLog { shader }`

Description: `shader` : Shader object

Utility function for easier use of OpenGL function `glGetShaderInfoLog`.  
 Given the shader object (as returned by function `glCreateShader`), the function returns the information log message as a Tcl string.

See also: `tcl3dOglGetProgramInfoLog`  
`tcl3dOglGetShaderSource`  
`tcl3dOglGetInfoLogARB`

Name: `tcl3dOglGetProgramInfoLog` - Get shader program log.

Synopsis: `tcl3dOglGetProgramInfoLog { shader }`

Description: `shader` : Shader program

Utility function for easier use of OpenGL function `glGetProgramInfoLog`.  
 Given the shader program (as returned by function `glCreateProgram`), the function returns the information log message as a Tcl string.

See also: `tcl3dOglGetShaderInfoLog`  
`tcl3dOglGetShaderSource`  
`tcl3dOglGetInfoLogARB`

Name: `tcl3dOglGetShaderSource` - Get shader object source.

Synopsis: `tcl3dOglGetShaderSource { shader }`

Description: `shader` : Shader object

Utility function for easier use of OpenGL function `glGetShaderSource`.  
 Given the shader object (as returned by function `glCreateShader`), the function returns the shader source code as a Tcl string.

See also: `tcl3dOglGetShaderInfoLog`  
`tcl3dOglGetProgramInfoLog`  
`tcl3dOglGetInfoLogARB`  
`tcl3dOglShaderSource`

Name: `tcl3dOglShaderSource` - Wrapper for `glShaderSource`.

Synopsis: `tcl3dOglShaderSource { shaderId shaderString }`

Description: `shaderId` : Shader handle  
`shaderString` : string

Wrapper for easier use of OpenGL function `glShaderSource`.  
 In contrast to `glShaderSource` only the shader program

identifier (created with a call to `glCreateShaderObject`) and the shader source have to be specified.

See also: `tcl3dOglGetShaderSource`

Name: `tcl3dOglReadShaderFile` - Read a shader file.

Synopsis: `tcl3dOglReadShaderFile { pathName }`

Description: `pathName` : Shader file name

Read shader file "pathName" and return it's contents as a string. The path name is transparently mapped with `tcl3dGetExtFile`, so that this procedure can be used from within a starpack.

See also: `tcl3dOglShaderSource`  
`tcl3dGetExtFile`

Name: `tcl3dOglCompileProgram` - Compile a shader program.

Synopsis: `tcl3dOglCompileProgram { vertexSource controlSource  
evaluationSource geometrySource fragmentSource }`

Description: `vertexSource` : string  
`controlSource` : string  
`evaluationSource`: string  
`geometrySource` : string  
`fragmentSource` : string

Compile and attach the specified shader sources. Vertex and fragment shader sources must be given. All other parameters can be supplied as an empty string, if the corresponding render stage should not be used.

See also: `tcl3dOglBuildProgram`  
`tcl3dOglLinkProgram`  
`tcl3dOglDestroyProgram`

Name: `tcl3dOglLinkProgram` - Link a shader program.

Synopsis: `tcl3dOglLinkProgram { programDict }`

Description: `programDict` : Program dictionary

Link the program specified in the program dictionary. "programDict" is the dictionary returned by `tcl3dOglCompileProgram`.

See also: `tcl3dOglBuildProgram`  
`tcl3dOglCompileProgram`  
`tcl3dOglDestroyProgram`

Name: `tcl3dOglBuildProgram` - Build a shader program.

Synopsis: `tcl3dOglBuildProgram { vertexSource controlSource  
evaluationSource geometrySource fragmentSource }`

Description: `vertexSource` : string  
`controlSource` : string  
`evaluationSource`: string  
`geometrySource` : string  
`fragmentSource` : string

Compile and link the specified shader sources. Vertex and fragment shader sources must be given. All other parameters can be supplied as an empty string, if the corresponding render stage should not be used.

See also:           tcl3dOglCompileProgram  
                  tcl3dOglLinkProgram  
                  tcl3dOglDestroyProgram

Name:               tcl3dOglDestroyProgram - Destroy a shader program.

Synopsis:            tcl3dOglDestroyProgram { programDict }

Description:        programDict : Program dictionary

Destroy the program specified in the program dictionary.  
"programDict" is the dictionary returned by  
tcl3dOglCompileProgram or tcl3dOglBuildProgram.

See also:           tcl3dOglBuildProgram  
                  tcl3dOglCompileProgram  
                  tcl3dOglLinkProgram

Implementation file: tcl3dOglUtil.tcl

Copyright: 2005-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dOgl  
 Filename: tcl3dOglUtil.tcl

Author: Paul Obermeier

Description: Tcl module with miscellaneous utility procedures related to the OpenGL module.

Name: tcl3dOglExtInit - Initialize the extension library.

Synopsis: tcl3dOglExtInit {}

Description: Initialize the OpenGL extension library. If no OpenGL context has been established (i.e. a Togl window has not been created), the function throws an error. It is recommended to add a call to tcl3dOglExtInit in the create callback. Note: With Tcl3D versions starting at 0.4, this procedure is not needed anymore, because the GLEW extension functions are initialized within the Togl widget start-up code. You may leave it in your code however for backwards compatibility.

See also:

Name: glMultiDrawElements - Draw multiple array elements.

Synopsis: glMultiDrawElements { mode count type indices primcount }

Description: mode : GLenum  
 count : tcl3dVector of type GLuint  
 type : GLenum  
 indices : List of tcl3dVectors  
 primcount : integer

Tcl wrapper for the OpenGL function glMultiDrawElements. As the Tcl3D Swig wrapper currently does not support "void \*\*" pointers, the multiple array elements must be specified as a list of tcl3dVectors.

See also:

Name: tcl3dOglGetError - Check for an OpenGL error.

Synopsis: tcl3dOglGetError {}

Description: Check, if an OpenGL related error has occurred.

If no error occurred, an empty string is returned. Otherwise a formatted string showing the error number and the error message is returned.

See also:

Name: tcl3dOglSetNormalMode - Set the execution mode of OpenGL functions to normal.

Synopsis: `tcl3dOglSetNormalMode { { printCmd puts } }`

Description: `printCmd` : command name

Set the execution mode of all OpenGL functions to normal.

The "printCmd" will be used to output OpenGL command execution infos. If not specified, the information is printed onto stdout with the puts command. The printCmd must be a command with a single string parameter.

See the documentation of `tcl3dOglSetMode` for a description of the OpenGL execution modes.

See also: `tcl3dOglSetSafeMode`  
`tcl3dOglSetDebugMode`  
`tcl3dOglSetMode`

Name: `tcl3dOglSetSafeMode` - Set the execution mode of OpenGL functions to safe.

Synopsis: `tcl3dOglSetSafeMode { { printCmd puts } }`

Description: `printCmd` : command name

Set the execution mode of all OpenGL functions to safe.

The "printCmd" will be used to output OpenGL command execution infos. If not specified, the information is printed onto stdout with the puts command. The printCmd must be a command with a single string parameter.

See the documentation of `tcl3dOglSetMode` for a description of the OpenGL execution modes.

See also: `tcl3dOglSetNormalMode`  
`tcl3dOglSetDebugMode`  
`tcl3dOglSetMode`

Name: `tcl3dOglSetDebugMode` - Set the execution mode of OpenGL functions to debug.

Synopsis: `tcl3dOglSetDebugMode { { printCmd puts } }`

Description: `printCmd` : command name

Set the execution mode of all OpenGL functions to debug.

The "printCmd" will be used to output OpenGL command execution infos. If not specified, the information is printed onto stdout with the puts command. The printCmd must be a command with a single string parameter.

See the documentation of `tcl3dOglSetMode` for a description of the OpenGL execution modes.

See also: `tcl3dOglSetNormalMode`  
`tcl3dOglSetSafeMode`  
`tcl3dOglSetMode`

Name: `tcl3dOglSetMode` - Set the execution mode for OpenGL functions.

Synopsis: `tcl3dOglSetMode { mode { printCmd puts } }`

Description: mode : string  
printCmd : command name

The OpenGL core and extension functions can be used in 3 different modes:  
"normal", "safe", "debug".

normal: Use the OpenGL functions as wrapped by SWIG. This is the fastest mode. If using an OpenGL function not available in the used driver implementation, this mode will dump core.

safe: In this mode every OpenGL function is checked for availability in the driver before execution. If it's not available, a message is printed out.

debug: This mode checks the availability of an OpenGL function like the safe mode, and additionally prints out every OpenGL function before execution.

The "printCmd" will be used to output OpenGL command execution infos. If not specified, the information is printed onto stdout with the puts command. The printCmd must be a command with a single string parameter.

See also: tcl3dOglSetNormalMode  
tcl3dOglSetSafeMode  
tcl3dOglSetDebugMode

Implementation file: tcl3dOsgBitmaps.tcl

Copyright: 2009-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dOsg  
Filename: tcl3dOsgBitmaps.tcl

Author: Paul Obermeier

Description: Tcl module with bitmaps specific to the OSG module.

Name: tcl3dOsgGetBitmap - Get the bitmap image of a node type.

Synopsis: tcl3dOsgGetBitmap { nodeType }

Description: Get the bitmap image of a node type.  
If the node type is not known or no bitmap is available yet, a bitmap with a question mark is returned.

See also: tcl3dOsgGetBitmapName

Implementation file: tcl3dOsgKeysym.tcl

Copyright: 2005-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dOsg  
Filename: tcl3dOsgKeysym.tcl

Author: Paul Obermeier

Description: Tcl module for mapping keysym values.  
Taken from Wiki page <http://wiki.tcl.tk/13162>.

Name: tcl3dOsgKeysym - Convert a keysym into decimal and vice versa.

Synopsis: tcl3dOsgKeysym { key }

Description: Convert a Tk keysym into it's decimal value and vice versa.  
Example:  
tcl3dOsgKeysym A --> 65  
tcl3dOsgKeysym 65 --> A

See also:

Implementation file: tcl3dOsgQuery.tcl

Copyright: 2009-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dOsg  
Filename: tcl3dOsgQuery.tcl

Author: Paul Obermeier

Description: Tcl module with query procedures related to the OSG module.

Name: tcl3dOsgGetVersion - Get OSG version string.

Synopsis: tcl3dOsgGetVersion {}

Description: Get the version of the wrapped OpenSceneGraph library. If no OpenGL context has been established (i.e. a Togl window has not been created), the function returns an empty string.

See also: tcl3dGetVersions  
tcl3dGetLibraryInfo

Implementation file: tcl3dOsgUtil.tcl

Copyright: 2009-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dOsg  
Filename: tcl3dOsgUtil.tcl

Author: Paul Obermeier

Description: Tcl module with miscellaneous Tcl3D utility procedures related to the OSG module.

Name: tcl3dOsgVecPrint - Print a OSG vector.  
tcl3dOsgMatPrint - Print a OSG matrix.  
tcl3dOsgBBoxPrint - Print a OSG bounding box.  
tcl3dOsgBSpherePrint - Print a OSG bounding sphere.

Synopsis: tcl3dOsgVecPrint { vec { precisionString "%6.3f" } }  
tcl3dOsgMatPrint { mat { precisionString "%6.3f" } }  
tcl3dOsgBBoxPrint { box { precisionString "%6.3f" } }  
tcl3dOsgBSpherePrint { sph { precisionString "%6.3f" } }

Description: vec : osg::Vec  
mat : osg::Matrix  
box : osg::BoundingBox  
sph : osg::BoundingSphere  
precisionString : string

Print the values of the corresponding OSG classes onto stdout. The precisionString parameter can be optionally specified and must supply a format specification in a C-printf style.

See also:

Name: tcl3dOsgVecArrayPrint - Print an array of vectors.  
tcl3dOsgScalarArrayPrint - Print an array of scalars.  
tcl3dOsgObjectArrayPrint - Print an array of objects.

Synopsis: tcl3dOsgVecArrayPrint { arr { precisionString "%6.3f" } }  
tcl3dOsgScalarArrayPrint { arr { precisionString "%6.3f" } }  
tcl3dOsgObjectArrayPrint { arr { precisionString "%s" } }

Description: arr : osg::VecArray, std::vector, osg::MixinVector  
precisionString : string

Print the values of all elements of container "arr". "arr" can be either a osg::VecArray, a std::vector or a osg::MixinVector container. Depending on the element type, special functions are available: tcl3dOsgVecArrayPrint prints the value of all osg::Vec's contained in the array. tcl3dOsgScalarArrayPrint prints the value of all scalars contained in the array. tcl3dOsgObjectArrayPrint prints the name of all osg::Object's contained in the array.

The precisionString parameter can be optionally specified and must supply a format specification in a C-printf style.

See also: tcl3dOsgScalarArrayPrint

Name: tcl3dOsgGetVisitorTypeName - Get visitor type name.

Synopsis:           tcl3dOsgGetVisitorTypeName { visitorType }

Description:       visitorType : int

                  Return the string representation of a  
                  osg::NodeVisitor::VisitorType enumeration type.

See also:           tcl3dOsgGetTraversalModeName

Name:               tcl3dOsgGetTraversalModeName - Get traversal mode name.

Synopsis:           tcl3dOsgGetTraversalModeName { travMode }

Description:       travMode : int

                  Return the string representation of a  
                  osg::NodeVisitor::TraversalMode enumeration type.

See also:           tcl3dOsgGetVisitorTypeName

Name:               tcl3dOsgSendButtonPress  
                  tcl3dOsgSendButtonRelease  
                  tcl3dOsgSendMouseMotion  
                  tcl3dOsgSendKeyPress  
                  tcl3dOsgSendKeyRelease  
                  tcl3dOsgSendWindowResize  
                  - Send the corresponding event down to OSG.

Synopsis:           tcl3dOsgSendButtonPress { osgwin x y buttonNum }  
                  tcl3dOsgSendButtonRelease { osgwin x y buttonNum }  
                  tcl3dOsgSendMouseMotion { osgwin x y }  
                  tcl3dOsgSendKeyPress { osgwin key }  
                  tcl3dOsgSendKeyRelease { osgwin key }  
                  tcl3dOsgSendWindowResize { osgwin w h }

Description:       osgwin       : int  
                  x, y           : int (cursor position)  
                  w, h           : int (window width and height)  
                  buttonNum   : int (Button number)  
                  key            : int (KeySym)

                  tcl3dOsgSend\* procedures transfer a Tcl/Tk event down  
                  to OSG. Note, that no redraw is done. You must either  
                  use "after idle" with a postredisplay command or use  
                  the utility commands without the "Send" in the name.

See also:

Name:               tcl3dOsgAddTrackballBindings  
                  - Add OS independent mouse bindings for trackball usage.

Synopsis:           tcl3dOsgAddTrackballBindings { toglwin osgwin }

Description:       toglwin     : Togl window identifier  
                  osgwin       : int

See also:

Implementation file: tcl3dSDLQuery.tcl

Copyright: 2007-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dSDL  
Filename: tcl3dSDLQuery.tcl

Author: Paul Obermeier

Description: Tcl module with query procedures related to the SDL module.

Name: tcl3dSDLGetVersion - Get SDL version string.

Synopsis: tcl3dSDLGetVersion {}

Description: Return the version string of the wrapped SDL library. The version is returned as "Major.Minor.Patch".

See also: tcl3dOglGetVersions  
tcl3dGetLibraryInfo

Implementation file: tcl3dSDLUtil.tcl

```

Copyright:      2006-2010 Paul Obermeier (obermeier@tcl3d.org)

                See the file "Tcl3D_License.txt" for information on
                usage and redistribution of this file, and for a
                DISCLAIMER OF ALL WARRANTIES.

Module:         Tcl3D -> tcl3dSDL
Filename:       tcl3dSDLUtil.tcl

Author:        Paul Obermeier

Description:    Tcl module with miscellaneous utility
                procedures related to the SDL module.

Name:          tcl3dSDLGetFocusName - Convert focus state bitfield.

Synopsis:      tcl3dSDLGetFocusName { state }

Description:    state : int

                Return a SDL focus state bitfield as the corresponding
                string representation.
                See file SDL_active.h for the definition of possible
                bitfield values.

See also:      tcl3dSDLGetButtonName
                tcl3dSDLGetHatName
                tcl3dSDLGetEventName

Name:          tcl3dSDLGetButtonName - Convert button state bitfield.

Synopsis:      tcl3dSDLGetButtonName { state }

Description:    state : int

                Return a SDL button state bitfield as the corresponding
                string representation.
                See file SDL_mouse.h for the definition of possible
                bitfield values.

See also:      tcl3dSDLGetFocusName
                tcl3dSDLGetHatName
                tcl3dSDLGetEventName

Name:          tcl3dSDLGetHatName - Convert hat state bitfield.

Synopsis:      tcl3dSDLGetHatName { state }

Description:    state : int

                Return a SDL hat state bitfield as the corresponding
                string representation.
                See file SDL_joystick.h for the definition of possible
                bitfield values.

See also:      tcl3dSDLGetFocusName
                tcl3dSDLGetButtonName
                tcl3dSDLGetEventName

Name:          tcl3dSDLGetEventName - Convert event enumeration.

Synopsis:      tcl3dSDLGetEventName { state }

Description:    state : int (SDL event enumeration)

```

Return SDL event related enumeration as the corresponding string representation. See file `SDL_events.h` for the definition of possible enumeration values.

See also: `tcl3dSDLGetFocusName`  
`tcl3dSDLGetButtonName`  
`tcl3dSDLGetHatName`

Name: `tcl3dSDLFrames2MSF` - Convert CD frames.

Synopsis: `tcl3dSDLFrames2MSF { frames }`

Description: `frames : int`

Return CD frame as minutes/seconds/frames as a list of 3 integers.

See also:

Name: `tcl3dSDLGetTrackTypeName` - Convert track type bitfield.

Synopsis: `tcl3dSDLGetTrackTypeName { type }`

Description: `type : int`

Return SDL CD track type bitfield as the corresponding string representation. See file `SDL_cdrom.h` for the definition of possible bitfield values.

See also: `tcl3dSDLGetCdStatusName`

Name: `tcl3dSDLGetCdStatusName` - Convert CD status enumeration.

Synopsis: `tcl3dSDLGetCdStatusName { status }`

Description: `status : int (CD status enumeration)`

Return SDL CD status enumeration as the corresponding string representation. See file `SDL_cdrom.h` for the definition of possible enumeration values (`CDstatus`).

See also: `tcl3dSDLGetTrackTypeName`

Implementation file: tcl3dUtilCapture.tcl

```

Copyright:      2005-2010 Paul Obermeier (obermeier@tcl3d.org)

                See the file "Tcl3D_License.txt" for information on
                usage and redistribution of this file, and for a
                DISCLAIMER OF ALL WARRANTIES.

Module:         Tcl3D -> tcl3dOgl
Filename:       tcl3dUtilCapture.tcl

Author:        Paul Obermeier

Description:    Tcl module implementing functions for capturing window
                contents into an image or file.
                Note: All of the functionality requires the help of the
                Img extension. Some of the functionality requires the
                help of the Twapi extension and is therefore only
                available on Windows.

Name:          tcl3dWidget2Img - Copy widget content into photo image.

Synopsis:      tcl3dWidget2Img { win { ign "" } }

Description:   win : string (Widget name)
                ign : string

                Copy contents of widget "win" and all of its
                sub-widgets into a photo image.
                If "ign" is specified and not the empty string, it is
                interpreted as a pattern for widget names, that should
                be ignored while traversing the widget hierarchy.
                The pattern is passed to the "string match" command.

                Return the photo image identifier.

See also:     tcl3dWidget2File
                tcl3dCanvas2Img
                tcl3dClipboard2Img
                tcl3dWindow2Img

Name:         tcl3dWidget2File - Copy widget content into image file.

Synopsis:     tcl3dWidget2File { win fileName { ign "" }
                { fmt "JPEG" } { opts "" } }

Description:  win      : string (Widget name)
                fileName : string
                ign      : string
                fmt       : string
                opts      : string

                Copy contents of widget "win" and all of its
                sub-widgets into a photo image and save this image to
                file "fileName". The file format handler is determined
                with "fmt". Some formats need optional parameters.
                These can be supplied in "opts".
                See the Img documentation (man img) for a list of format
                handlers and options.
                If "ign" is specified and not the empty string, it is
                interpreted as a pattern for widget names, that should
                be ignored while traversing the widget hierarchy.
                The pattern is passed to the "string match" command.

See also:    tcl3dWidget2Img
                tcl3dCanvas2File
                tcl3dClipboard2File
                tcl3dWindow2File

```

Name: tcl3dCanvas2Img - Copy canvas content into photo image.

Synopsis: tcl3dCanvas2Img { canv }

Description: canv : string (Widget name)

Copy the contents of canvas "canv" into a photo image.

Return the photo image identifier.

See also: tcl3dCanvas2File  
tcl3dWidget2Img  
tcl3dClipboard2Img  
tcl3dWindow2Img

Name: tcl3dCanvas2File - Copy canvas content into image file.

Synopsis: tcl3dCanvas2File { canv fileName { fmt "JPEG" }  
{ opts "" } }

Description: canv : string (Widget name)  
fileName : string  
fmt : string  
opts : string

Copy the contents of canvas "canv" into a photo image and save the image to file "fileName". The file format handler is determined with "fmt". Some formats need optional parameters. These can be supplied in "opts". See the Img documentation (man img) for a list of format handlers and options.

See also: tcl3dCanvas2Img  
tcl3dWidget2File  
tcl3dClipboard2File  
tcl3dWindow2File

Name: tcl3dClipboard2Img - Copy clipboard content into photo image.

Synopsis: tcl3dClipboard2Img {}

Description: Copy the contents of the Windows clipboard into a photo image.

Return the photo image identifier, if successful. Otherwise a Tcl error is thrown.

Note: This function is currently available only under Windows and needs the Twapi extension.

See also: tcl3dClipboard2File  
tcl3dWidget2Img  
tcl3dCanvas2Img  
tcl3dWindow2Img

Name: tcl3dClipboard2File - Copy clipboard content into file.

Synopsis: tcl3dClipboard2File { fileName { fmt "JPEG" }  
{ opts "" } }

Description: fileName : string  
fmt : string  
opts : string

Copy the contents of the Windows clipboard into a photo image and save the image to file "fileName". The file format handler is determined with "fmt". Some formats

need optional parameters.  
 These can be supplied in "opts".  
 See the Img documentation (man img) for a list of format  
 handlers and options.

Note: This function is currently available only under  
 Windows and needs the Twapi extension.

See also:       tcl3dClipboard2Img  
                   tcl3dWidget2File  
                   tcl3dCanvas2File  
                   tcl3dWindow2File

Name:           tcl3dImg2Clipboard - Copy photo image into clipboard.

Synopsis:       tcl3dImg2Clipboard { phImg }

Description:   phImg : string (Photo image identifier)

Copy photo image "phImg" into the Windows clipboard.

Note: This function is currently available only under  
 Windows and needs the Twapi extension.

See also:       tcl3dClipboard2Img

Name:           tcl3dWindow2Clipboard - Copy window contents into  
 clipboard.

Synopsis:       tcl3dWindow2Clipboard {}

Description:   Copy the contents of the top level window (Alt-PrtSc)  
 into the Windows clipboard.

Note: This function is currently available only under  
 Windows and needs the Twapi extension.

See also:       tcl3dClipboard2Img

Name:           tcl3dDesktop2Clipboard - Copy desktop contents into  
 clipboard.

Synopsis:       tcl3dDesktop2Clipboard {}

Description:   Copy the contents of the whole desktop (PrtSc) into the  
 Windows clipboard.

Note: This function is currently available only under  
 Windows and needs the Twapi extension.

See also:       tcl3dWindow2Clipboard

Name:           tcl3dWindow2Img - Copy window contents into photo image.

Synopsis:       tcl3dWindow2Img {}

Description:   Copy the contents of the top level window into a photo  
 image.

Return the photo image identifier, if successful.  
 Otherwise a Tcl error is thrown.

See also:       tcl3dWindow2File

Name:           tcl3dWindow2File - Copy window contents into file.

Synopsis:       tcl3dWindow2File { fileName { fmt "JPEG" } { opts "" } }

Description:     fileName : string  
                  fmt        : string  
                  opts       : string

Copy the contents of the top level window into a photo image and save the image to file "fileName". The file format handler is determined with "fmt". Some formats need optional parameters. These can be supplied in "opts". See the Img documentation (man img) for a list of format handlers and options.

See also:         tcl3dWindow2Img

Implementation file: tcl3dUtilColors.tcl

Copyright: 2005-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dOgl  
 Filename: tcl3dUtilColors.tcl

Author: Paul Obermeier, Victor G. Bonilla

Description: Tcl module to convert Tcl color names into Tcl3D color lists. Color names may be specified as numeric values or strings.  
 Currently accepted Tcl color names:  
     #RRGGBB  
     All names as listed in the Tcl manual pages, section colors.

This module has been inspired by Victor G. Bonilla, who wrote the first version of this file.

Name: tcl3dGetColorNames - Get all supported Tcl color names.

Synopsis: tcl3dGetColorNames {}

Description: Return a list of all supported Tcl color name strings.

See also: tcl3dFindColorName

Name: tcl3dFindColorName - Validate Tcl color name.

Synopsis: tcl3dFindColorName { colorName }

Description: colorName : string

Check, if supplied color name is a valid string color name. If true, return the supplied color name, otherwise return an empty string.

See also: tcl3dGetColorNames

Name: tcl3dName2Hex - Convert color name to Tcl hexadecimal.

Synopsis: tcl3dName2Hex { colorName }

Description: colorName : string

Convert Tcl color name "colorName" into the corresponding Tcl hexadecimal representation.  
 Tcl colors are returned as string in the following format:  
 #RRGGBB

See also: tcl3dName2rgb  
 tcl3dName2rgbf

Name: tcl3dName2Hexa - Convert color name to Tcl hexadecimal.

Synopsis: tcl3dName2Hexa { colorName }

Description: colorName : string

Convert Tcl color name "colorName" into the corresponding Tcl hexadecimal representation.  
 Tcl colors are returned as string in the following format:

```
#RRGGBBAA
```

See also: `tcl3dName2rgba`  
`tcl3dName2rgbaf`

Name: `tcl3dName2rgb` - Convert color name to OpenGL RGB.

Synopsis: `tcl3dName2rgb { colorName }`

Description: `colorName` : string

Convert Tcl color name "colorName" into the corresponding OpenGL RGB representation.  
OpenGL colors are returned as a list of 3 unsigned bytes: { r g b }

See also: `tcl3dName2rgba`  
`tcl3dName2rgbfb`

Name: `tcl3dName2rgbfb` - Convert color name to OpenGL float RGB.

Synopsis: `tcl3dName2rgbfb { colorName }`

Description: `colorName` : string

Convert Tcl color name "colorName" into the corresponding OpenGL float RGB representation.  
OpenGL colors are returned as a list of 3 floats in the range [0..1]: { r g b }

See also: `tcl3dName2rgbaf`  
`tcl3dName2rgb`

Name: `tcl3dName2rgba` - Convert color name to OpenGL RGBA.

Synopsis: `tcl3dName2rgba { colorName }`

Description: `colorName` : string

Convert Tcl color name "colorName" into the corresponding OpenGL RGBA representation.  
OpenGL colors are returned as a list of 4 unsigned bytes: { r g b a }

See also: `tcl3dName2rgb`  
`tcl3dName2rgbaf`

Name: `tcl3dName2rgbaf` - Convert color name to OpenGL float RGBA.

Synopsis: `tcl3dName2rgbaf { colorName }`

Description: `colorName` : string

Convert Tcl color name "colorName" into the corresponding OpenGL float RGBA representation.  
OpenGL colors are returned as a list of 4 floats in the range [0..1]: { r g b a }

See also: `tcl3dName2rgba`  
`tcl3dName2rgbfb`

Name: `tcl3dRgb2Name` - Convert OpenGL RGB to color name.

Synopsis: `tcl3dRgb2Name { r g b }`

Description: `r, g, b` : int

Convert an OpenGL RGB color representation into a hexadecimal Tcl color name string.  
OpenGL colors are specified as unsigned bytes in the range [0..255].

Note: For performance issues no range checking is performed.  
If specifying color values outside the allowed range, the resulting Tcl color name may result in an error like following:  
can't parse color "#FD109142"

See also: tcl3dName2rgb  
tcl3dRgba2Name

Name: tcl3dRgba2Name - Convert OpenGL RGBA to color name.

Synopsis: tcl3dRgba2Name { r g b a }

Description: r, g, b, a : int

Convert an OpenGL RGBA color representation into a hexadecimal Tcl color name string.  
OpenGL colors are specified as unsigned bytes in the range [0..255].

Note: For performance issues no range checking is performed.  
If specifying color values outside the allowed range, the resulting Tcl color name may result in an error like following:  
can't parse color "#FD109142"

See also: tcl3dName2rgba  
tcl3dRgb2Name

Name: tcl3dRgbf2Name - Convert OpenGL RGB to color name.

Synopsis: tcl3dRgbf2Name { r g b }

Description: r, g, b : float

Convert an OpenGL RGB color representation into a hexadecimal Tcl color name string.  
OpenGL colors are specified as floats in the range [0..1].

Note: For performance issues no range checking is performed.  
If specifying color values outside the allowed range, the resulting Tcl color name may result in an error like following:  
can't parse color "#FD109142"

See also: tcl3dName2rgbf  
tcl3dRgbf2Name

Name: tcl3dRgbuf2Name - Convert OpenGL RGBA to color name.

Synopsis: tcl3dRgbuf2Name { r g b a }

Description: r, g, b, a : float

Convert an OpenGL RGBA color representation into a hexadecimal Tcl color name string.  
OpenGL colors are specified as floats in the range [0..1].

Note: For performance issues no range checking is

performed.

If specifying color values outside the allowed range, the resulting Tcl color name may result in an error like following:  
can't parse color "#FD109142"

See also:       tcl3dName2rgbaf  
                  tcl3dRgbaf2Name

Implementation file: tcl3dUtilFile.tcl

Copyright: 2005-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dOgl  
 Filename: tcl3dUtilFile.tcl

Author: Paul Obermeier

Description: Tcl module with file handling Tcl3D utility procedures.

Name: tcl3dGetTmpDir - Get the name of a temporary directory.

Synopsis: tcl3dGetTmpDir {}

Description: Return the location of a temporary directory name. The function checks the following environment variables first: TMP, TEMP, TMPDIR. If none of these variables are defined, standard places are checked (C:/Windows/Temp, /tmp).

See also: tcl3dCreateTmpDir

Name: tcl3dCreateTmpDir - Create a unique temporary directory.

Synopsis: tcl3dCreateTmpDir {}

Description: Create a unique temporary directory. Return the full path name of the created directory. The new directory is created in the standard temporary directory as returned by tcl3dGetTmpDir.

See also: tcl3dGetTmpDir

Name: tcl3dGenExtName - Create a name on the file system.

Synopsis: tcl3dGenExtName { fileName }

Description: fileName : string

Return a valid path name on the file system generated from the file name specified in "fileName". Use this function, if writing to a file from a script, which may be running from within a Starpack. If the script is not executed from a Starpack, the function returns the supplied parameter unchanged.

See also: tcl3dGetExtFile

Name: tcl3dGetExtFile - Get a name on the file system.

Synopsis: tcl3dGetExtFile { fileName { forceOverwrite false } }

Description: fileName : string

Return a valid path name on the file system generated from the file name specified in "fileName". Use this function, if a file is needed for reading from an external Tcl3D library, like font files (used by FTGL) or shader files and the script may be executed from within a virtual file system (ex. starkit vfs). The file included in the virtual file system is transparently copied onto the file system and that path

name is returned. If "forceOverwrite" is set to true, an existing file with the same name is overwritten. The path name is built using a system-wide temporary directory as returned by tcl3dGetTmpDir. If the script is not executed from within a virtual file system, the function returns the supplied parameter unchanged.

See also:       tcl3dGenExtName  
              tcl3dGetTmpDir

Name:           tcl3dGetDirList - Get the contents of a directory.

Synopsis:        tcl3dGetDirList {dirName  
                  {showDirs 1} {showFiles 1}  
                  {showHiddenDirs 1} {showHiddenFiles 1}  
                  {dirPattern \*} {filePattern \*}}

Description:    dirName           : string  
                  showDirs         : bool  
                  showFiles        : bool  
                  showHiddenDirs   : bool  
                  showHiddenFiles  : bool  
                  dirPattern       : string  
                  filePattern       : string

Scan directory "dirName" and return a two-element list consisting of all directories and files of the scanned directory. The first element contains the directory list, where the pathnames of the directories are absolute paths. The second element contains the file list with no path information. The result list can be filtered by specifying the other procedure parameters and should be self-explaining.

See also:       tcl3dCreateTmpDir

Implementation file: tcl3dUtilImg.tcl

```
Copyright:      2010 Paul Obermeier (obermeier@tcl3d.org)

                See the file "Tcl3D_License.txt" for information on
                usage and redistribution of this file, and for a
                DISCLAIMER OF ALL WARRANTIES.

Module:         Tcl3D -> tcl3dOgl
Filename:      tcl3dUtilImg.tcl

Author:        Paul Obermeier

Description:    Tcl utility module for handling images.

Name:          tcl3dReadImg - Read an image from file.

Synopsis:      tcl3dReadImg { imgName }

Description:    imgName : string

                Try to read the image data stored in file "imgName" and
                creates a Tcl3D Vector, which can be used to supply a
                texture to OpenGL glTexImg* functions.

                Return a dictionary with the following keys containing
                information about the photo image:
                "data"   : The Tcl3D Vector.
                "width"  : Width of the image.
                "height" : Height of the image.
                "chans"  : Number of image channels.
                "format" : $::GL_RGB (3-channel images) or
                           $::GL_RGBA (4-channel images)

                If the file contains no image data or an unsupported
                format, a Tcl error is thrown.

Notes:
You should include a "package require Img" to have
support for a large number of image file formats.
You are responsible to free the memory allocated for
the Tcl3D Vector.

Example:
# Create the Tcl3D Vector containing the image data.
set img [tcl3dReadImg "myImage.tga"]
# Use the information from the dictionary for texturing.
glTexImage2D GL_TEXTURE_2D 0 [dict get $img format] \
             [dict get $img width] [dict get $img height] \
             0 [dict get $img format] GL_UNSIGNED_BYTE \
             [dict get $img data]
# Free the memory of the Tcl3D Vector.
[dict get $img data] delete

See also:      tcl3dVectorFromPhoto
```

Implementation file: tcl3dUtilInfo.tcl

```

Copyright:      2005-2010 Paul Obermeier (obermeier@tcl3d.org)

                See the file "Tcl3D_License.txt" for information on
                usage and redistribution of this file, and for a
                DISCLAIMER OF ALL WARRANTIES.

Module:         Tcl3D -> tcl3dOgl
Filename:       tcl3dUtilInfo.tcl

Author:        Paul Obermeier

Description:    Tcl module to get Tcl3D related information:
                Version, extensions, state variables.

Name:          tcl3dHavePackage - Check availability of a specific
                Tcl3D package.

Synopsis:       tcl3dHavePackage { pkgName version }

Description:    pkgName : string
                version : string

                Return 1, if Tcl package "pkgName" is available in at
                least version "version". Otherwise return 0.

                Example: tcl3dHavePackage tcl3dcg 0.3.2
                           checks availability of the tcl3dCg package in
                           at least version 0.3.2.

See also:      tcl3d{SubPackage}GetVersion

Name:          tcl3dGetLibraryInfo - Get library version of a Tcl3D
                module.

Synopsis:       tcl3dGetLibraryInfo { pkgName }

Description:    pkgName : string

                Return the library version corresponding to supplied
                Tcl3D package name "pkgName" as a string. If no version
                information is available, an empty string is returned.

See also:      tcl3dGetPackageInfo

Name:          tcl3dGetPackageInfo - Get Tcl3D package information.

Synopsis:       tcl3dGetPackageInfo {}

Description:    Return a list of sub-lists containing Tcl3D package
                information. Each sub-list contains the name of the
                sub-package, the availability flag (0 or 1), the
                sub-package version as well as the version of the
                wrapped library.

                Example:
                {tcl3dcg 1 0.3.3 1.5.0015}      {tcl3ddemoutil 1 0.3.3 {}}
                {tcl3dftgl 1 0.3.3 2.1.2}      {tcl3dgauges 1 0.3.3 {}}
                {tcl3dgl2ps 1 0.3.3 1.3.2}     {tcl3dode 1 0.3.3 0.7.0}
                {tcl3dogl 1 0.3.3 {APPLE-1.4}} {tcl3dsdl 1 0.3.3 1.2.9}
                {tcl3dtogl 1 0.3.3 {}}        {tcl3dutil 1 0.3.3 {}}

                Note: A Togl window (and therefore a graphics context)
                must have been created before issuing a call to
                this function.

See also:      tcl3dShowPackageInfo

```

Name: tcl3dShowPackageInfo - Display package information.

Synopsis: tcl3dShowPackageInfo {}

Description: Display the version info returned by tcl3dGetPackageInfo in a toplevel window.

Note: A Togl window (and therefore a graphics context) must have been created before issuing a call to this function.

See also: tcl3dGetPackageInfo

Name: tcl3dHaveCg - Check availability of tcl3dCg module.

Synopsis: tcl3dHaveCg {}

Description: Return 1, if the Cg library has been loaded successfully via the tcl3dCg module. Otherwise return 0.

Note: This function is only available when loading Tcl3D via a "package require tcl3d".

See also: tcl3dGetPackageInfo  
tcl3dCgGetVersion

Name: tcl3dHaveSDL - Check availability of tcl3dSDL module.

Synopsis: tcl3dHaveSDL {}

Description: Return 1, if the SDL library has been loaded successfully via the tcl3dSDL module. Otherwise return 0.

Note: This function is only available when loading Tcl3D via a "package require tcl3d".

See also: tcl3dGetPackageInfo  
tcl3dSDLGetVersion

Name: tcl3dHaveFTGL - Check availability of tcl3dFTGL module.

Synopsis: tcl3dHaveFTGL {}

Description: Return 1, if the FTGL library has been loaded successfully via the tcl3dFTGL module. Otherwise return 0.

Note: This function is only available when loading Tcl3D via a "package require tcl3d".

See also: tcl3dGetPackageInfo  
tcl3dFTGLGetVersion

Name: tcl3dHaveOde - Check availability of tcl3dOde module.

Synopsis: tcl3dHaveOde {}

Description: Return 1, if the ODE library has been loaded successfully via the tcl3dOde module. Otherwise return 0.

Note: This function is only available when loading Tcl3D via a "package require tcl3d".

See also: tcl3dGetPackageInfo  
tcl3dOdeGetVersion

Name: `tcl3dHaveOsg` - Check availability of `tcl3dOsg` module.

Synopsis: `tcl3dHaveOsg {}`

Description: Return 1, if the OSG library has been loaded successfully via the `tcl3dOsg` module. Otherwise return 0.

Note: This function is only available when loading Tcl3D via a "package require tcl3d".

See also: `tcl3dGetPackageInfo`  
`tcl3dOsgGetVersion`

Name: `tcl3dHaveGl2ps` - Check availability of `tcl3dGl2ps` module.

Synopsis: `tcl3dHaveGl2ps {}`

Description: Return 1, if the GL2PS library has been loaded successfully via the `tcl3dGl2ps` module. Otherwise return 0.

Note: This function is only available when loading Tcl3D via a "package require tcl3d".

See also: `tcl3dGetPackageInfo`  
`tcl3dGl2psGetVersion`

Implementation file: tcl3dUtilLogo.tcl

```

Copyright:      2005-2010 Paul Obermeier (obermeier@tcl3d.org)

                See the file "Tcl3D_License.txt" for information on
                usage and redistribution of this file, and for a
                DISCLAIMER OF ALL WARRANTIES.

Module:         Tcl3D -> tcl3dOgl
Filename:       tcl3dUtilLogo.tcl

Author:        Paul Obermeier

Description:    Tcl module showing the Tcl/Tk and poSoft logo.

Name:          tcl3dLogoDestroyPoSoft - Destroy poSoft logo window.

Synopsis:      tcl3dLogoDestroyPoSoft {}

Description:    Destroy a previously opened poSoft logo window.

See also:      tcl3dLogoShowPoSoft
                tcl3dLogoShowTcl

Name:          tcl3dLogoShowPoSoft - Display poSoft logo.

Synopsis:      tcl3dLogoShowPoSoft { version copyright withdrawWin }

Description:    version      : string
                copyright   : string
                withdrawWin : string (Widget name)

                Display the poSoft logo in two possible ways:
                If "withdrawWin" is set to the empty string, the logo
                is shown in a window with decoration. This may be used
                for displaying the logo as an action for an "About"
                menu entry.
                If "withdrawWin" is set to an existing window name
                (typically the name of the main application window), the
                logo window is shown without decoration as a splash
                window, which automatically disappears after a second.
                The logo window has two label widgets to display
                additional text messages, which are specified in
                "version" and "copyright".

See also:      tcl3dLogoDestroyPoSoft
                tcl3dLogoShowTcl

Name:          tcl3dLogoDestroyTcl - Destroy Tcl logo window.

Synopsis:      tcl3dLogoDestroyTcl { w img }

Description:    Destroy a previously opened Tcl logo window.

See also:      tcl3dLogoShowTcl
                tcl3dLogoShowPoSoft

Name:          tcl3dLogoShowTcl - Display Tcl logo.

Synopsis:      tcl3dLogoShowTcl { args }

Description:    args : variable parameter list

                Display the Tcl logo with additional text messages in
                a window with decoration. This may be used
                for displaying the logo as an action for an "About"
                menu entry.

```

"args" may contain any combination of the following  
package names:  
Tk Img Tktable combobox mysqltcl tcom

See also: `tcl3dLogoShowPoSoft`

Implementation file: tcl3dUtilTrackball.tcl

Copyright: 2005-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dOgl  
 Filename: tcl3dUtilTrackball.tcl

Author: Paul Obermeier

Description: Simple trackball-like motion adapted (ripped off) from projtex.c (written by David Yu and David Blythe). See the SIGGRAPH '96 Advanced OpenGL course notes.

Usage overview:

Call tcl3dTbInit before any other trackball call.  
 Call tcl3dTbReshape from the reshape callback.  
 Call tcl3dTbMatrix to get the trackball matrix rotation.  
 Call tcl3dTbStartMotion to begin trackball movement.  
 Call tcl3dTbStopMotion to stop trackball movement.  
 Call tcl3dTbMotion from the motion callback.  
 Call tcl3dTbAnimate(1) if you want the trackball to continue spinning after the mouse button has been released.  
 Call tcl3dTbAnimate(0) if you want the trackball to stop spinning after the mouse button has been released.

See ftglDemo.tcl for a real world example.

Modified for Tcl3D by Paul Obermeier 2006/02/02  
 See www.tcl3d.org for the Tcl3D extension.

Name: tcl3dTbStartMotion - Begin trackball movement

Synopsis: tcl3dTbStartMotion { toglwin x y }

Description: toglwin : string  
 x : int  
 y : int

Begin movement of the trackball attached to Togl window "toglwin".  
 "x" and "y" give the actual mouse position inside the Togl window.  
 This procedure is typically bound to a button press event.  
 Example: bind .toglwin <ButtonPress-1>  
           "tcl3dTbStartMotion .toglwin %x %y"

See also: tcl3dTbStopMotion  
 tcl3dTbMotion

Name: tcl3dTbStopMotion - Stop trackball movement

Synopsis: tcl3dTbStopMotion { toglwin }

Description: toglwin : string

Stop movement of the trackball attached to Togl window "toglwin".  
 This procedure is typically bound to a button release event.  
 Example: bind .toglwin <ButtonRelease-1>  
           "tcl3dTbStopMotion .toglwin"

See also: `tcl3dTbStartMotion`  
`tcl3dTbMotion`

Name: `tcl3dTbAnimate` - Set the trackball animation mode.

Synopsis: `tcl3dTbAnimate { toglwin animate }`

Description: `togglewin : string`  
`animate : bool`

Set the animation mode of the trackball attached to Togl window "togglewin".  
 If the trackball shall continue spinning after the mouse button has been released, set "animate" to true.  
 Set "animate" to false, if the trackball should stop spinning after the mouse button has been released.

See also: `tcl3dTbStartMotion`

Name: `tcl3dTbInit` - Initialize the trackball module.

Synopsis: `tcl3dTbInit { toglwin }`

Description: `togglewin : string`

Initialize the trackball attached to Togl window "togglewin".  
 This procedure must be called before any other trackball procedures, for example in the Togl create callback.

See also:

Name: `tcl3dTbMatrix` - Use the trackball matrix rotation

Synopsis: `tcl3dTbMatrix { toglwin }`

Description: `togglewin : string`

Use the rotation matrix of the trackball attached to Togl window "togglewin".  
 The rotation matrix is applied to the top most OpenGL matrix with `glMultMatrixf`.  
 This procedure is typically called in the Togl display callback.

See also:

Name: `tcl3dTbReshape` - Notify trackball about a reshape.

Synopsis: `tcl3dTbReshape { toglwin w h }`

Description: `togglewin : string`  
`w : int`  
`h : int`

Notify the trackball attached to Togl window "togglewin" that the size of the window has been changed to width "w" and height "h".  
 This procedure is typically called in the Togl reshape callback.

See also: `tcl3dTbInit`

Name: `tcl3dTbMotion` - Move the trackball.

Synopsis: `tcl3dTbMotion { toglwin x y }`

Description:    `toglwin` : string  
                  `x`        : int  
                  `y`        : int

Move the trackball attached to Togl window "toglwin".  
"x" and "y" give the actual mouse position inside the  
Togl window.

This procedure is typically bound to a mouse motion  
event.

Example: `bind .toglwin <B1-Motion>`  
          `"tcl3dTbMotion .toglwin %x %y"`

See also:        `tcl3dTbStartMotion`  
                  `tcl3dTbStopMotion`

Implementation file: tcl3dVecMath.tcl

Copyright: 2005-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dOgl  
Filename: tcl3dVecMath.tcl

Author: Paul Obermeier

Description: Tcl module to handle vectors and transformation matrices.

Name: tcl3dVec3Print - Print contents of a 3D vector.

Synopsis: tcl3dVec3Print { vec { precisionString "%6.3f" } }

Description: vec : string (Tcl3D Vector Identifier)  
precisionString: string, optional

Print the contents of 3D Vector "vec" onto standard output. "vec" is a Tcl3D Vector of size 3 and type float or double.

See also: tcl3dMatPrint

Name: tcl3dMatPrint - Print contents of a transformation matrix.

Synopsis: tcl3dMatPrint { mat { precisionString "%6.3f" } }

Description: mat : string (Tcl3D Vector Identifier)  
precisionString: string, optional

Print the contents of transformation matrix "mat" onto standard output. "mat" is a Tcl3D Vector of size 16 and type float or double.

See also: tcl3dVec3Print

Name: tcl3dRadToDeg - Convert angle from radians to degrees.

Synopsis: tcl3dRadToDeg { ang }

Description: ang : double

Return angle "ang" specified in radians in degrees.

See also: tcl3dDegToRad

Name: tcl3dDegToRad - Convert angle from degrees to radians.

Synopsis: tcl3dDegToRad { ang }

Description: ang : double

Return angle "ang" specified in degrees in radians.

See also: tcl3dRadToDeg

Implementation file: tcl3dVector.tcl

Copyright: 2005-2010 Paul Obermeier (obermeier@tcl3d.org)

See the file "Tcl3D\_License.txt" for information on usage and redistribution of this file, and for a DISCLAIMER OF ALL WARRANTIES.

Module: Tcl3D -> tcl3dOgl  
 Filename: tcl3dVector.tcl

Author: Paul Obermeier

Description: Tcl module to handle tcl3dVectors, i.e. contiguous pieces of memory.

Name: tcl3dVector - Create a new Tcl3D Vector

Synopsis: tcl3dVector { type size }

Description: type : string  
 size : int

Create a new Tcl3D Vector of size "size" by calling the memory allocation routine new\_"type" and create a new Tcl procedure.  
 The contents of the new Tcl3D Vector are uninitialized. Return the identifier (i.e. the name of the created Tcl procedure) of the new Tcl3D Vector.

The following base types are currently supported:

GLbitfield	GLboolean	GLbyte	GLclampd
GLclampf	GLdouble	GLenum	GLfloat
GLint	GLshort	GLsizei	GLubyte
GLuint	GLushort	double	float
int	short	uint	ushort

Note: To get an up-to-date list of wrapped types, issue the command "info commands new\_\*" after loading Tcl3D or use the script "vectorTypes.tcl" in directory "tcl3dUtil/test".

A detailed description of Tcl3D Vectors can be found in the Tcl3D manual.

See also: tcl3dVectorFromArgs  
 tcl3dVectorFromByteArray  
 tcl3dVectorFromList  
 tcl3dVectorFromPhoto  
 tcl3dVectorFromString

Name: tcl3dVectorInd - Get index of a Tcl3D Vector.

Synopsis: tcl3dVectorInd { vec type ind }

Description: vec : string (Tcl3D Vector Identifier)  
 type : string  
 ind : int

Return the "pointer" to the "ind" element of a Tcl3D Vector. The base Tcl3D Vector is specified with "vec", the type of the Vector is given with "type".

Note: See the description of tcl3dVector for a list of usable types.  
 This function may be used in conjunction with OpenGL interleaved vertex arrays. See RedBook demo "aapolyStride.tcl" for an example usage.

See also: `tcl3dVector`

Name: `tcl3dVectorPrint` - Print contents of a Tcl3D Vector.

Synopsis: `tcl3dVectorPrint { vec num { precisionString "%6.3f" } }`

Description: `vec` : string (Tcl3D Vector Identifier)  
`num` : num  
`precisionString`: string, optional

Print the first "num" elements of Tcl3D Vector "vec" onto standard output.

Note: Tcl3D Vectors behave like C vectors, i.e. they do not have information about its length.

See also: `tcl3dVector`

Name: `tcl3dVectorFromArgs` - Create new Tcl3D Vector from an argument list.

Synopsis: `tcl3dVectorFromArgs { type args }`

Description: `type` : string  
`args` : list

Create a new Tcl3D Vector of type "type" from given variable argument list.  
Return the identifier (i.e. the name of the created Tcl procedure) of the new Tcl3D Vector.

Note: See the description of `tcl3dVector` for a list of usable types.

See also: `tcl3dVector`  
`tcl3dVectorFromByteArray`  
`tcl3dVectorFromList`  
`tcl3dVectorFromPhoto`  
`tcl3dVectorFromString`

Name: `tcl3dVectorFromList` - Create new Tcl3D Vector from a list.

Synopsis: `tcl3dVectorFromList { type l { maxElems -1 } }`

Description: `type` : string  
`l` : list  
`maxElems` : int

Create a new Tcl3D Vector of type "type" from given Tcl list "l". If "maxElems" is given and greater than zero, only the first "maxElems" are used.  
Return the identifier (i.e. the name of the created Tcl procedure) of the new Tcl3D Vector.

Note: See the description of `tcl3dVector` for a list of usable types.

See also: `tcl3dVector`  
`tcl3dVectorFromArgs`  
`tcl3dVectorFromByteArray`  
`tcl3dVectorFromPhoto`  
`tcl3dVectorFromString`

Name: `tcl3dCharToNum` - Convert character to integer.

Synopsis: `tcl3dCharToNum { char }`

Description: char : character

Convert an ASCII character into the corresponding numeric value.

See also: tcl3dNumToChar

Name: tcl3dNumToChar - Convert integer to character.

Synopsis: tcl3dNumToChar { num }

Description: num : int

Convert a numeric value into the corresponding ASCII character.

See also: tcl3dCharToNum

Name: tcl3dVectorFromString - Create new Tcl3D Vector from a string.

Synopsis: tcl3dVectorFromString { type str }

Description: type : string  
str : string

Create a new Tcl3D Vector of type "type" from given string "str".  
Return the identifier (i.e. the name of the created Tcl procedure) of the new Tcl3D Vector.

Note: This version is very slow and is intended only for converting the characters of short text strings into it's numerical values to be used by display lists rendering raster fonts.  
See the description of tcl3dVector for a list of usable types.

See also: tcl3dVector  
tcl3dVectorFromArgs  
tcl3dVectorFromByteArray  
tcl3dVectorFromList  
tcl3dVectorFromPhoto

Name: tcl3dVectorFromByteArray - Create new Tcl3D Vector from binary string.

Synopsis: tcl3dVectorFromByteArray { type str }

Description: type : string  
str : string

Create a new Tcl3D Vector of type "type" from given binary string "str".  
Return the identifier (i.e. the name of the created Tcl procedure) of the new Tcl3D Vector.

Note: See the description of tcl3dVector for a list of usable types.

See also: tcl3dVector  
tcl3dVectorFromArgs  
tcl3dVectorFromList  
tcl3dVectorFromPhoto  
tcl3dVectorFromString

Name: tcl3dVectorFromPhoto - Create new Tcl3D Vector from a Tk photo.

Synopsis: `tcl3dVectorFromPhoto { phImg { numChans -1 }  
{ scl 1.0 } { off 0.0 } }`

Description: `phImg : string (Photo image identifier)`  
`numChans : int`  
`scl : double`  
`off : double`

Create a new Tcl3D Vector containing the image data of Tk photo "phImg". The created Tcl3D Vector is of type `GL_UNSIGNED_BYTE`. If "numChans" is specified and between 1 and 4, only the first "numChans" are copied into the Tcl3D Vector. Otherwise all channels available in the photo image are used.  
"scl" and "off" can be used to scale and offset the pixel values while converting.  
Return the identifier (i.e. the name of the created Tcl procedure) of the new Tcl3D Vector.

See also: `tcl3dVector`  
`tcl3dVectorFromArgs`  
`tcl3dVectorFromByteArray`  
`tcl3dVectorFromList`  
`tcl3dVectorFromString`

Name: `tcl3dVectorFromLinspace` - Create new linearly spaced Tcl3D Vector.

Synopsis: `tcl3dVectorFromLinspace { type s e n }`

Description: `type : string`  
`s : Start value of type "type"`  
`e : End value of type "type"`  
`n : int`

Create a new Tcl3D Vector of type "type" and length "n" containing "n" data values linearly spaced between and including "s" and "e".  
Type can be any of the following:  
`GLubyte`, `GLushort`, `GLuint`, `GLfloat`, `GLdouble`, `float`, `double`.

Return the identifier (i.e. the name of the created Tcl procedure) of the new Tcl3D Vector.

This command implements the functionality of the MATLAB `linspace` command.

See also: `tcl3dVector`  
`tcl3dVectorFromArgs`  
`tcl3dVectorFromByteArray`  
`tcl3dVectorFromList`  
`tcl3dVectorFromString`

Name: `tcl3dVectorToList` - Copy Tcl3D Vector into a list.

Synopsis: `tcl3dVectorToList { vec num }`

Description: `vec : string (Tcl3D Vector Identifier)`  
`num : int`

Copy "num" elements of Tcl3D Vector "vec" into a Tcl list and return that list.

Note: This version is slow and is intended only for converting 3D vectors or transformation matrices into Tcl lists.

See also: `tcl3dVectorFromList`

```
tcl3dVectorToByteArray
tcl3dVectorToString
```

Name: tcl3dVectorToString - Copy Tcl3D Vector into a string.

Synopsis: tcl3dVectorToString { vec }

Description: vec : string (Tcl3D Vector Identifier)

Interpret the elements of Tcl3D Vector "vec" (which must be of type GLubyte) as a null-terminated string and return that string.

Note: This version is slow and is intended only for short text strings. Use this function for example to convert the information returned by a GLSL shader.

See also: tcl3dVectorFromString  
tcl3dVectorToByteArray  
tcl3dVectorToList

Name: tcl3dVectorToByteArray - Copy Tcl3D Vector into a binary string.

Synopsis: tcl3dVectorToByteArray { vec numBytes {srcOff 0} {destOff 0} } {

Description: vec : string (Tcl3D Vector Identifier)  
numBytes : int  
srcOff : int  
destOff : int

Copy "numBytes" elements of Tcl3D Vector "vec" into a Tcl binary string and return that string. The Tcl3D Vector has be of type GLubyte. "srcOff" and "destOff" may be used optionally to specify an offset into the source and the destination.

See also: tcl3dVectorFromByteArray  
tcl3dVectorToList  
tcl3dVectorToString