XCDG Reference Manual

0.95

Generated by Doxygen 1.3.8

Wed Oct 20 17:44:57 2004

Contents

Chapter 1

The XCDG Reference Manual

Author:
Michael Daum, Kilian A. Foth, Dietmar Fünning

Id
xcdg.tcl,v 1.92 2004/10/11 15:23:31 micha Exp

1.1 Overview

This is the programmers reference manual of XCDG, that is a documentation of the implementation. For a detailed usage overview please consult the Xcdg User Manual.

1.2 Copyright

Chapter 2

XCDG Namespace Index

2.1 XCDG Namespace List

Here is a list of all documented namespaces with brief descriptions:

Chapter 3

XCDG Hierarchical Index

3.1 XCDG Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 4

XCDG Class Index

4.1 XCDG Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 5

XCDG File Index

5.1 XCDG File List

Here is a list of all documented files with brief descriptions:

Chapter 6

XCDG Page Index

6.1 XCDG Related Pages

Here is a list of all related documentation pages:

Chapter 7

XCDG Namespace Documentation

7.1

7.1.1 Detailed Description

Namespace cmd

this namespace defines all commands which are allowed to be evaluated in the cdg-script-language

Todo

purge this interface with the help of `commandEval`
return values of the XCDG commands are a mess. See cmd::Verify() and cmd::Constraint().

Author:

Michael Daum (see also AUTHORS and THANKS for more)

Id

commands.tcl,v 1.117 2004/09/06 13:40:53 micha Exp

Functions

- _get (TclCommand oldHook, TclString value)
- Activate (TclList args)
- Anno2Parse (TclList args)
- **Annos2Prolog** (TclList args)
- Annotation (TclList args)
- Cd (TclList args)
- Chunk (TclList args)
- Clear (TclList args)
- Compareparses (TclList args)
- Compile (TclList args)
- Constraint (TclList args)
- Deactivate (TclList args)
- deduceNetName (TclString cmdline)
- Deleteparse (TclList args)
- Distance (TclString netId)
- Edges (TclString netId, TclString startId="", TclString stopId="")

- Frobbing (TclList args)
- Gls (TclList args)
- Help (TclList args)
- Hierarchy (TclList args)
- Hook (TclList args)
- IC (TclList args)
- IncrementalCompletion (TclList args)
- Inputwordgraph (TclString word, TclList args)
- Isearch (TclList args)
- Level (TclList args)
- Levelsort (TclList args)
- Lexicon (TclList args)
- License (TclList args)
- Load (TclList args)
- Ls (TclList args)
- Net (TclList args)
- Netdelete (TclList args)
- Netsearch (TclList args)
- NewGls (TclList args)
- Newnet (TclString wordGraphId="")
- Nonspeccompatible (TclList args)
- Parses2Prolog (TclList args)
- Printf (TclString format, TclList args)
- PrintParse (TclList args)
- PrintParses (TclList args)
- Puts (TclString arg1, TclString arg2="")
- Quit ()
- Renewnet (TclList args)
- Reset (TclList args)
- Section ()
- Set (TclString variable, TclString value="")
- Shift (TclList args)
- Showlevel (TclList args)
- Showparse (TclList args)
- Source (TclString file)
- Status ()
- Tagger (TclList args)
- Testing (TclList args)
- Useconstraint (TclList args)
- Uselevel (TclList args)
- Verify (TclList args)
- Version (TclList args)
- Weight (TclList args)
- Wordgraph (TclList args)
- Writeannotation (TclList args)
- Writenet (TclList args)
- WriteParses (TclList args)
- WriteWordgraph (TclList args)

7.1.2 Function Documentation

7.1.2.1 _get (TclCommand *oldHook*, TclString *value*)

callback installed by Set(). This function captures the returnvalue of a CDG set command. Definition at line 334 of file commands.tcl.

References _get().

Referenced by _get().

7.1.2.2 Activate (TclList *args*)

delegate the command to the C core.

Definition at line 200 of file commands.tcl.

References Activate().

Referenced by Activate().

7.1.2.3 Anno2Parse (TclList *args*)

delegate the command to the C core and add the new parse to AllParses.

Definition at line 162 of file commands.tcl.

References Anno2Parse().

Referenced by Anno2Parse().

7.1.2.4 Annotation (TclList *args*)

delegate the command to the C core.

Definition at line 152 of file commands.tcl.

References Annotation().

Referenced by Annotation().

7.1.2.5 Cd (TclList *args*)

implement the extraordinary feature of directory changing.

Definition at line 573 of file commands.tcl.

References Cd().

Referenced by Cd().

7.1.2.6 Chunk (TclList *args*)

delegate the command to the C core.

Definition at line 795 of file commands.tcl.

References Chunk().

Referenced by Chunk().

7.1.2.7 Clear (TclList *args*)

clear the shell window.

Definition at line 512 of file commands.tcl.

References Clear().

Referenced by Clear().

7.1.2.8 Compareparses (TclList *args*)

delegate the command to the C core.

Definition at line 760 of file commands.tcl.

References Compareparses().

Referenced by Compareparses().

7.1.2.9 Compile (TclList *args*)

delegate the command to the C core.

Definition at line 692 of file commands.tcl.

References Compile().

Referenced by Compile().

7.1.2.10 Constraint (TclList *args*)

delegate the command to the C core.

Definition at line 94 of file commands.tcl.

References Constraint().

Referenced by Constraint().

7.1.2.11 Deactivate (TclList *args*)

delegate the command to the C core.

Definition at line 192 of file commands.tcl.

References Deactivate().

Referenced by Deactivate().

7.1.2.12 deduceNetName (TclString *cmdline*)

try to find out on what net the command will operate.

Definition at line 29 of file commands.tcl.

References deduceNetName().

Referenced by deduceNetName().

7.1.2.13 Deleteparse (TclList *args*)

enable AllParses::deleteparse() in the xcdg shell.

Definition at line 553 of file commands.tcl.

References Deleteparse().

Referenced by Deleteparse().

7.1.2.14 Distance (TclString *netId*)

delegate the command to the C core.

Definition at line 374 of file commands.tcl.

References Distance().

Referenced by Distance().

7.1.2.15 Edges (TclString *netId*, TclString *startId* = "", TclString *stopId* = "")

delegate the command to the C core.

Definition at line 265 of file commands.tcl.

References Edges().

Referenced by Edges().

7.1.2.16 Frobbing (TclList *args*)

delegate the command to the C core. Additionally, we get a busy box and update AllParses and AllNetworks. Definition at line 605 of file commands.tcl.

References Frobbing().

Referenced by Frobbing().

7.1.2.17 Gls (TclList *args*)

delegate the command to the C core. Additionally, we get a busy box and update AllParses and AllNetworks. Definition at line 640 of file commands.tcl.

References Gls().

Referenced by Gls().

7.1.2.18 Help (TclList *args*)

delegate the command to the C core.

Definition at line 767 of file commands.tcl.

References Help().

Referenced by Help().

7.1.2.19 Hierarchy (TclList *args*)

delegate the command to the C core.

Definition at line 175 of file commands.tcl.

References Hierarchy().

Referenced by Hierarchy().

7.1.2.20 Hook (TclList *args*)

delegate the command to the C core.

Definition at line 520 of file commands.tcl.

References Hook().

Referenced by Hook().

7.1.2.21 IC (TclList *args*)

delegate the command to the C core.

Definition at line 242 of file commands.tcl.

References IC().

Referenced by IC().

7.1.2.22 IncrementalCompletion (TclList *args*)

delegate the command to the C core.

Definition at line 700 of file commands.tcl.

References IncrementalCompletion().

Referenced by IncrementalCompletion().

7.1.2.23 Inputwordgraph (TclString *word*, TclList *args*)

delegate the command to the C core. In addition this function refreshes AllWordgraphs. Definition at line 383 of file commands.tcl.

References Inputwordgraph().

Referenced by Inputwordgraph().

7.1.2.24 Isearch (TclList *args*)

delegate the command to the C core. Additionally this function updates AllNetworks. Definition at line 417 of file commands.tcl.

References Isearch().

Referenced by Isearch().

7.1.2.25 Level (TclList *args*)

delegate the command to the C core.

Definition at line 102 of file commands.tcl.

References Level().

Referenced by Level().

7.1.2.26 Levelsort (TclList *args*)

delegate the command to the C core.

Definition at line 529 of file commands.tcl.

References Levelsort().

Referenced by Levelsort().

7.1.2.27 Lexicon (TclList *args*)

delegate the command to the C core.

Definition at line 134 of file commands.tcl.

References Lexicon().

Referenced by Lexicon().

7.1.2.28 License (TclList *args*)

delegate the command to the C core.

Definition at line 753 of file commands.tcl.

References License().

Referenced by License().

7.1.2.29 Load (TclList *args*)

loads one or more file

Definition at line 73 of file commands.tcl.

References Load().

Referenced by Load().

7.1.2.30 Ls (TclList *args*)

implement the shell command ls.

Definition at line 483 of file commands.tcl.

References Ls().

Referenced by Ls().

7.1.2.31 Net (TclList *args*)

delegate the command to the C core.

Definition at line 208 of file commands.tcl.

References Net().

Referenced by Net().

7.1.2.32 Netdelete (TclList *args*)

delegate the command to the C core and refresh AllNetworks.

Definition at line 256 of file commands.tcl.

References Netdelete().

Referenced by Netdelete().

7.1.2.33 Netsearch (TclList *args*)

delegate the command to the C core. Additionally this function displays a busy box and adds the resulting parse to AllParses Definition at line 218 of file commands.tcl.

References Netsearch().

Referenced by Netsearch().

7.1.2.34 NewGls (TclList *args*)

delegate the command to the C core. Additionally, we get a busy box and update AllParses and AllNetworks. Definition at line 663 of file commands.tcl.

References NewGls().

Referenced by NewGls().

7.1.2.35 Newnet (TclString *wordGraphId* = "")

delegate the command to the C core. In addition this function refreshes AllNetworks. Definition at line 344 of file commands.tcl.

References Newnet().

Referenced by Newnet().

7.1.2.36 Nonspeccompatible (TclList *args*)

delegate the command to the C core.

Definition at line 565 of file commands.tcl.

References Nonspeccompatible().

Referenced by Nonspeccompatible().

7.1.2.37 Parses2Prolog (TclList *args*)

delegate the command to the C core.

Definition at line 402 of file commands.tcl.

References Parses2Prolog().

Referenced by Parses2Prolog().

7.1.2.38 Printf (TclString *format*, TclList *args*)

immitate the C command printf in tcl.

Definition at line 471 of file commands.tcl.

References Printf().

Referenced by Printf().

7.1.2.39 PrintParse (TclList *args*)

delegate the command to the C core.

Definition at line 708 of file commands.tcl.

References PrintParse().

Referenced by PrintParse().

7.1.2.40 PrintParses (TclList *args*)

delegate the command to the C core.

Definition at line 716 of file commands.tcl.

References PrintParses().

Referenced by PrintParses().

7.1.2.41 Puts (TclString *arg1*, TclString *arg2* = "")

intercept the tcl command puts to print to the Shell.

Definition at line 439 of file commands.tcl.

References Puts().

Referenced by Puts().

7.1.2.42 Quit ()

quits the cdg-tool

Definition at line 56 of file commands.tcl.

References Quit().

Referenced by Quit().

7.1.2.43 Renewnet (TclList *args*)

delegate the command to the C core.

Definition at line 724 of file commands.tcl.

References Renewnet().

Referenced by Renewnet().

7.1.2.44 Reset (TclList *args*)

question a core reset of the system.

Definition at line 580 of file commands.tcl.

References Reset().

Referenced by Reset().

7.1.2.45 Section ()

delegate the command to the C core.

Definition at line 184 of file commands.tcl.

References Section().

Referenced by Section().

7.1.2.46 Set (TclString *variable*, TclString *value* = "")

delegate the command to the C core.

Definition at line 297 of file commands.tcl.

References Set().

Referenced by Set().

7.1.2.47 Shift (TclList *args*)

delegate the command to the C core.

Definition at line 802 of file commands.tcl.

References Shift().

Referenced by Shift().

7.1.2.48 Showlevel (TclList *args*)

delegate the command to the C core.

Definition at line 110 of file commands.tcl.

References Showlevel().

Referenced by Showlevel().

7.1.2.49 Showparse (TclList *args*)

enable AllParses::showparse() in the xcdg shell.

Definition at line 539 of file commands.tcl.

References Showparse().

Referenced by Showparse().

7.1.2.50 Source (TclString *file*)

loads one or more file

Definition at line 83 of file commands.tcl.

References Source().

Referenced by Source().

7.1.2.51 Status ()

delegate the command to the C core.

Definition at line 289 of file commands.tcl.

References Status().

Referenced by Status().

7.1.2.52 Tagger (TclList *args*)

delegate the command to the C core.

Definition at line 774 of file commands.tcl.

References Tagger().

Referenced by Tagger().

7.1.2.53 Testing (TclList *args*)

delegate the command to the C core.

Definition at line 781 of file commands.tcl.

References Testing().

Referenced by Testing().

7.1.2.54 Useconstraint (TclList *args*)

delegate the command to the C core.

Definition at line 126 of file commands.tcl.

References Useconstraint().

Referenced by Useconstraint().

7.1.2.55 Uselevel (TclList *args*)

delegate the command to the C core.

Definition at line 118 of file commands.tcl.

References Uselevel().

Referenced by Uselevel().

7.1.2.56 Verify (TclList *args*)

delegate the command to the C core.

Definition at line 685 of file commands.tcl.

References Verify().

Referenced by Verify().

7.1.2.57 Version (TclList *args*)

delegate the command to the C core.

Definition at line 731 of file commands.tcl.

References Version().

Referenced by Version().

7.1.2.58 Weight (TclList *args*)

delegate the command to the C core.

Definition at line 788 of file commands.tcl.

References Weight().

Referenced by Weight().

7.1.2.59 Wordgraph (TclList *args*)

delegate the command to the C core.

Definition at line 142 of file commands.tcl.

References Wordgraph().

Referenced by Wordgraph().

7.1.2.60 Writeannotation (TclList *args*)

delegate the command to the C core.

Definition at line 273 of file commands.tcl.

References Writeannotation().

Referenced by Writeannotation().

7.1.2.61 Writenet (TclList *args*)

delegate the command to the C core.

Definition at line 281 of file commands.tcl.

References Writenet().

Referenced by Writenet().

7.1.2.62 WriteParses (TclList *args*)

delegate the command to the C core.

Definition at line 739 of file commands.tcl.

References WriteParses().

Referenced by WriteParses().

7.1.2.63 WriteWordgraph (TclList *args*)

delegate the command to the C core.

Definition at line 746 of file commands.tcl.

References WriteWordgraph().

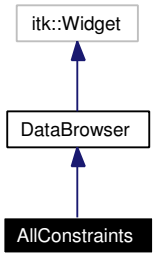
Referenced by WriteWordgraph().

Chapter 8

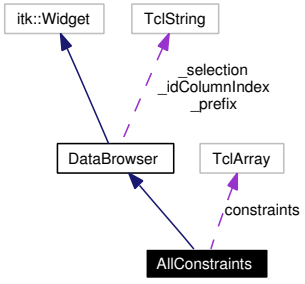
XCDG Class Documentation

8.1 AllConstraints Class Reference

Inheritance diagram for AllConstraints:



Collaboration diagram for AllConstraints:



8.1.1 Detailed Description

AllConstraints - manage all constraints.

Author:

Michael Daum (see also AUTHORS and THANKS for more)

Id

allconstraints.tcl,v 1.33 2004/10/11 13:50:05 micha Exp

Definition at line 18 of file allconstraints.tcl.

Public Member Functions

- public method getAllConstraintIds AllConstraints (TclList args)
- getCData (TclString id)
- getSelection ()
- init_data ()
- refreshid (TclString id)
- refreshrow (TclNumber row, Constraint constraint)
- setIndexedSelection (TclList args)
- setSelection (TclList args)

Protected Member Functions

- _browse_action (TclWidget w)
- _keypress_action (TclWidget w, TclKeyBinding k)
- _motion_action (TclWidget w, TclNumber x, TclNumber y)
- _return_action ()
- _rowtag (TclNumber row)
- _setCount (TclNumber n)
- refreshrow (TclNumber row, TclString item)

Protected Attributes

- TclString _idColumnIndex = ""
- TclString _selection = ""

Private Member Functions

- _init_data ()
- editbutton_action ()
- showbutton_action ()
- usebutton_action ()
- usegroupbutton_action ()
- uselevelbutton_action ()
- weightbutton_action ()

Private Attributes

- TclArray constraints

8.1.2 Constructor & Destructor Documentation

8.1.2.1 AllConstraints::AllConstraints (TclList *args*)

constructor

Definition at line 54 of file allconstraints.tcl.

8.1.3 Member Function Documentation

8.1.3.1 DataBrowser::_browse_action (TclWidget *w*) [protected, inherited]

browse slot. This method adjusts the selected file in AllFiles::_selection.

Parameters:

w the widget bound to this slot.

Definition at line 189 of file databrowser.tcl.

8.1.3.2 AllConstraints::_init_data () [private]

get data from the cdg tool.

Definition at line 149 of file allconstraints.tcl.

8.1.3.3 DataBrowser::_keypress_action (TclWidget *a*, TclKeyBinding *k*) [protected, inherited]

React to a keypress into the table.

Keys typed by the user are collected into a string, and the row whose id matches that string is selected.
Definition at line 421 of file databrowser.tcl.

8.1.3.4 DataBrowser::_motion_action (TclWidget *w*, TclNumber *x*, TclNumber *y*) [protected, inherited]

Default motion slot.

Parameters:

w the widget where the motion was detected

x the x coords of the mouse

y the y coords of the mouse

Reimplemented in AllFiles.

Definition at line 390 of file databrowser.tcl.

References DataBrowser::refreshid().

8.1.3.5 DataBrowser::_return_action () [protected, inherited]

actions to take place on pressing return in the entryfield.

Definition at line 244 of file databrowser.tcl.

8.1.3.6 DataBrowser::_rowtag (TclNumber *row*) [protected, inherited]

colorize the table rows This method is a callback configured to the table in order to colorize the rows.
Definition at line 361 of file databrowser.tcl.

8.1.3.7 DataBrowser::_setCount (TclNumber *n*) [protected, inherited]

display the count-label. This number should reflect the number of items selected

Parameters:
n the number to be set

Definition at line 375 of file databrowser.tcl.

8.1.3.8 AllConstraints::editbutton_action () [private]

call an editor to view the constraint source

Definition at line 350 of file allconstraints.tcl.

8.1.3.9 AllConstraints::getCData (TclString *id*)

replaces getconstraint

Reimplemented from DataBrowser.

Definition at line 387 of file allconstraints.tcl.

8.1.3.10 DataBrowser::getSelection () [inherited]

return a list of selected ids

Definition at line 292 of file databrowser.tcl.

References DataBrowser::setSelection().

8.1.3.11 AllConstraints::init_data ()

call _init_data() if necessary.

Reimplemented from DataBrowser.

Definition at line 136 of file allconstraints.tcl.

8.1.3.12 DataBrowser::refreshid (TclString *id*) [inherited]

refresh the displayed data for a specific ID

Definition at line 398 of file databrowser.tcl.

Referenced by DataBrowser::_motion_action().

8.1.3.13 DataBrowser::refreshrow (TclNumber *row*, TclString *item*) [protected, inherited]

abstract method called in refreshid

Reimplemented in AllWordgraphs.

8.1.3.14 AllConstraints::refreshrow (TclNumber *row*, Constraint *constraint*)

fill a row with the values from a specific constraint.
Definition at line 202 of file allconstraints.tcl.

8.1.3.15 DataBrowser::setIndexedSelection (TclList *args*) [inherited]

Select one or more rows. A previous selection is cleared; without arguemnts, removes all selections.
ARGS is a list of row indices. Definition at line 332 of file databrowser.tcl.

8.1.3.16 DataBrowser::setSelection (TclList *args*) [inherited]

Select one or more rows. A previous selection is cleared; without arguemnts, removes all selections.
ARGS must be a list of strings without spaces in them. Definition at line 304 of file databrowser.tcl.
Referenced by DataBrowser::getSelection().

8.1.3.17 AllConstraints::showbutton_action () [private]

display the constraint in the shell.
Definition at line 376 of file allconstraints.tcl.

8.1.3.18 AllConstraints::usebutton_action () [private]

toggle usage of selected constraints.
Definition at line 340 of file allconstraints.tcl.

8.1.3.19 AllConstraints::usegroupbutton_action () [private]

toggle usage of groups of selected constraints .
Definition at line 258 of file allconstraints.tcl.

8.1.3.20 AllConstraints::uselevelbutton_action () [private]

toggle usage of level of selected constraints.
Definition at line 283 of file allconstraints.tcl.

8.1.3.21 AllConstraints::weightbutton_action () [private]

Set new weight for the selected constraint.
Definition at line 313 of file allconstraints.tcl.

8.1.4 Member Data Documentation

8.1.4.1 TclString DataBrowser::_idColumnIndex = "" [protected, inherited]

column index of table that contains selectable ids (used in _browse_action)

Definition at line 87 of file databrowser.tcl.

8.1.4.2 TclString DataBrowser::_selection = "" [protected, inherited]

string of current selected row ids

Definition at line 84 of file databrowser.tcl.

8.1.4.3 TclArray AllConstraints::constraints [private]

hash mapping constraint ids to Constraint structures

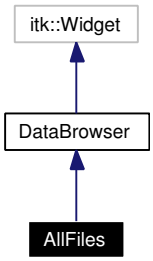
Definition at line 48 of file allconstraints.tcl.

The documentation for this class was generated from the following file:

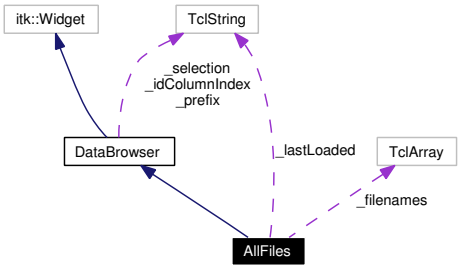
- allconstraints.tcl

8.2 AllFiles Class Reference

Inheritance diagram for AllFiles:



Collaboration diagram for AllFiles:



8.2.1 Detailed Description

AllFiles - CDG grammar file manager. This class is responsible for all issues regarding grammar files in the xcdg application, that is loading, reloading grammar files and xml annotaion files.

Author:
Michael Daum (see also AUTHORS and THANKS for more)

Id
allfiles.tcl,v 1.65 2004/10/11 13:50:05 micha Exp

Definition at line 20 of file allfiles.tcl.

Public Member Functions

- AllFiles (TclList args)
- getCData (TclString id)
- getSelection ()
- init_data ()
- load (TclList args)
- load_dir (TclList args)
- loadXml (TclString filename)
- refreshid (TclString id)
- reload ()

- selectIdsOfFile (TclString filename)
- setIndexedSelection (TclList args)
- setSelection (TclList args)

Protected Member Functions

- _browse_action (TclWidget w)
- _keypress_action (TclWidget w, TclKeyBinding k)
- _motion_action (TclWidget w, TclNumber x, TclNumber y)
- _return_action ()
- _rowtag (TclNumber row)
- _setCount (TclNumber n)
- refreshrow (TclNumber row, TclString item)

Protected Attributes

- TclString _idColumnIndex = ""
- TclString _selection = ""

Private Member Functions

- _dom2anno (DomNode document)
- _edit_action ()
- _load (TclList args)
- _load_action ()
- _reload_action ()
- _reset_action ()
- _run_action ()

Private Attributes

- TclArray _filenames
- TclString _lastLoaded = ""

8.2.2 Constructor & Destructor Documentation

8.2.2.1 AllFiles::AllFiles (TclList *args*)

An AllFiles constructor.

Parameters:

args arguments passed to itk_initialize

Definition at line 62 of file allfiles.tcl.

8.2.3 Member Function Documentation

8.2.3.1 DataBrowser::_browse_action (TclWidget *w*) [protected, inherited]

browse slot. This method adjusts the selected file in AllFiles::_selection.

Parameters:
w the widget bound to this slot.

Definition at line 189 of file databrowser.tcl.

8.2.3.2 AllFiles::_dom2anno (DomNode *document*) [private]

analyse a dom document and convert it to an annotation.

Parameters:
document the dom document

Returns:
a string buffer containing the annotation in the cdg annotation format

This method mainly is called by loadXml . Definition at line 418 of file allfiles.tcl.

8.2.3.3 AllFiles::_edit_action () [private]

edit slot. This method is called on and edit button press and starts an editor on the selected filenames. Definition at line 329 of file allfiles.tcl.

8.2.3.4 DataBrowser::_keypress_action (TclWidget *a*, TclKeyBinding *k*) [protected, inherited]

React to a keypress into the table.

Keys typed by the user are collected into a string, and the row whose id matches that string is selected. Definition at line 421 of file databrowser.tcl.

8.2.3.5 AllFiles::_load (TclList *args*) [private]

internal load method. This method is called while the CdgBusy box is shown.

Parameters:
args list of filenames to be loaded

Returns:
the list of the filenames basenames

Definition at line 207 of file allfiles.tcl.

8.2.3.6 AllFiles::_load_action () [private]

load slot. This command is executed whenever the load button is pressed. It loads the selected files in AllFiles::_selection and sets the selection accordingly. Definition at line 287 of file allfiles.tcl.

8.2.3.7 AllFiles::_motion_action (TclWidget *w*, TclNumber *x*, TclNumber *y*) [protected]

motion slot. Moving the mouse over the table extracts displays a help string in the status line of the main window.

Parameters:

w the widget where the motion was detected

x the x coords of the mouse

y the y coords of the mouse

Reimplemented from DataBrowser.

Definition at line 360 of file allfiles.tcl.

8.2.3.8 AllFiles::_reload_action () [private]

reload slot. This command is executed whenever the load button is pressed. It resets the cdg system before loading the files in AllFiles::_lastLoaded. Definition at line 298 of file allfiles.tcl.

8.2.3.9 AllFiles::_reset_action () [private]

reset slot. This method simply calls ::cmd::Reset. Definition at line 347 of file allfiles.tcl.

8.2.3.10 DataBrowser::_return_action () [protected, inherited]

actions to take place on pressing return in the entryfield.

Definition at line 244 of file databrowser.tcl.

8.2.3.11 DataBrowser::_rowtag (TclNumber *row*) [protected, inherited]

colorize the table rows This method is a callback configured to the table in order to colorize the rows. Definition at line 361 of file databrowser.tcl.

8.2.3.12 AllFiles::_run_action () [private]

select a file and run it as a tcl-cdg-script

Definition at line 307 of file allfiles.tcl.

8.2.3.13 DataBrowser::_setCount (TclNumber *n*) [protected, inherited]

display the count-label. This number should reflect the number of items selected

Parameters:

n the number to be set

Definition at line 375 of file databrowser.tcl.

8.2.3.14 DataBrowser::getCData (TclString *id*) [inherited]

abstract methods for retrieving the relevant C or Tcl data values for a given ID

Reimplemented in AllConstraints, AllLexemes, AllNetworks, AllParses, and AllWordgraphs.

8.2.3.15 DataBrowser::getSelection () [inherited]

return a list of selected ids

Definition at line 292 of file databrowser.tcl.

References DataBrowser::setSelection().

8.2.3.16 AllFiles::init_data ()

initialization of the data managed by this class. This method refreshes the data which could have been changed elsewhere, that is consult the C layer and squeeze out the relevant information.

Reimplemented from DataBrowser.

Definition at line 242 of file allfiles.tcl.

8.2.3.17 AllFiles::load (TclList *args*)

load files and store additional information which could be acquired

Definition at line 146 of file allfiles.tcl.

8.2.3.18 AllFiles::load_dir (TclList *args*)

load all files in a directory.

Todo

this method does not use the busy dialog.

Definition at line 188 of file allfiles.tcl.

8.2.3.19 AllFiles::loadXml (TclString *filename*)

read in an xml annotation.

Parameters:

filename the xml file containing the xml annotation

The provided file is read, and the xml document is searched for an <annotation> .... </annotation>. This was hopefully generated with writeXmlAnnoEntry() . Definition at line 384 of file allfiles.tcl.

8.2.3.20 DataBrowser::refreshid (TclString *id*) [inherited]

refresh the displayed data for a specific ID

Definition at line 398 of file databrowser.tcl.

Referenced by DataBrowser::_motion_action().

8.2.3.21 DataBrowser::refreshrow (TclNumber *row*, TclString *item*) [protected, inherited]

abstract method called in refreshid

Reimplemented in AllWordgraphs.

8.2.3.22 AllFiles::reload ()

load the last loaded file again

Definition at line 232 of file allfiles.tcl.

8.2.3.23 AllFiles::selectIdsOfFile (TclString *filename*)

select row of a file. previous selections are cleared Definition at line 124 of file allfiles.tcl.

8.2.3.24 DataBrowser::setIndexedSelection (TclList *args*) [inherited]

Select one or more rows. A previous selection is cleared; without arguemnts, removes all selections. ARGS is a list of row indices. Definition at line 332 of file databrowser.tcl.

8.2.3.25 DataBrowser::setSelection (TclList *args*) [inherited]

Select one or more rows. A previous selection is cleared; without arguemnts, removes all selections. ARGS must be a list of strings without spaces in them. Definition at line 304 of file databrowser.tcl.

Referenced by DataBrowser::getSelection().

8.2.4 Member Data Documentation

8.2.4.1 TclArray AllFiles::_filenames [private]

array holding all loaded files.

Definition at line 51 of file allfiles.tcl.

8.2.4.2 TclString DataBrowser::_idColumnIndex = "" [protected, inherited]

column index of table that contains selectable ids (used in _browse_action)

Definition at line 87 of file databrowser.tcl.

8.2.4.3 TclString AllFiles::_lastLoaded = "" [private]

list if recently loaded files

Definition at line 54 of file allfiles.tcl.

8.2.4.4 TclString DataBrowser::_selection = "" [protected, inherited]

string of current selected row ids

Definition at line 84 of file databrowser.tcl.

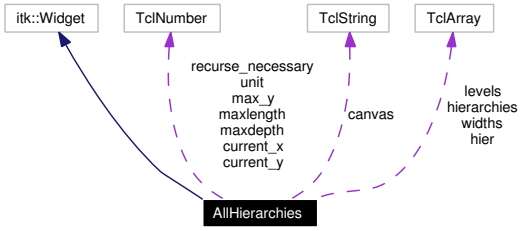
The documentation for this class was generated from the following file:

- allfiles.tcl

8.3 AllHierarchies Class Reference

Inheritance diagram for AllHierarchies:

Collaboration diagram for AllHierarchies:



8.3.1 Detailed Description

AllHierarchies - manage all hierarchies.

Todo

Variable and function naming is a complete mess here. Til now I just fixed the declarations to match the implementations. Furtheron all underscore/capitalized namings should be unified. No interface variable should only be named `id`. When it is a hierarchy's id call it for example `hierId`.
the TclArray called `hier` is cononfusing me with the TclArray `hierarchy`. I know they contain different stuff but again: the naming is extraordinary bad.
the TclArray `hier` is overloaded: its information should be split into several arrays named by the third argument of the hier array.
the methods C_to_Tcllist() and ldelete() are not related to AllHierarchies.
private varialbes should be named starting with and underscore

Author:

Dietmar Fünning, Kilian A. Foth (see also AUTHORS and THANKS for more)

Id

allhierarchies.tcl,v 1.34 2004/10/11 13:50:05 micha Exp

Definition at line 29 of file allhierarchies.tcl.

Public Member Functions

- AllHierarchies (TclList args)
- getAllHierarchyIds ()
- init_data ()

Private Member Functions

- _init_data ()
- balanceNode (TclString id, TclString type_id)
- C_to_Tcllist (List list)
- center (TclString id, TclNumber level)
- compute_fathers_and_sons (Hierarchy hierarchy, TclString type_id)
- compute_graph (TclString id)
- compute_hierarchy (Hierarchy hierarchy)
- compute_positions (TclString id)
- display_graph (TclString id, TclNumber level, TclList nodes_current_lvl, TclNumber x_start, TclNumber y_start, TclNumber counter)
- drawHierarchy (TclString id)
- drawSort (TclString id, TclString type_id)
- importNode (TclString id, TclString type_id)
- initial_x_pos (TclString id, TclNumber level)
- initialize_depth (TclString id, TclString type_id)
- is_hierarchy_a_tree (Hierarchy hierarchy, TclString type_id)
- labelEnter (TclCommand w, TclNumber pointerx, TclNumber pointery)
- labelLeave (TclCommand w)
- ldelete (TclList list, TclString type_id)
- leaves_right (TclString id, TclString String)
- max_type_depth (TclString id, TclString type_id, TclNumber depth)
- new_order (TclString id, TclString String)
- number_of_fathers_and_sons (Hierarchy hierarchy, TclString type_id)
- position_leaves (TclString id, TclNumber level)
- position_leaves2 (TclString id, TclString type_id)
- position_of_type (Hierarchy hierarchy, TclString type_id)
- set_level_array (TclString id, TclString type_id)
- stringlength (TclString id, TclString type_id)
- xpixels_per_character (TclString id)

Private Attributes

- TclString **canvas** = ""
- TclNumber **current_x** = 1
- TclNumber **current_y** = 1
- TclArray **hier**
- TclArray **hierarchies**
- TclArray **levels**
- TclNumber **max_y** = 1
- TclNumber **maxdepth** = 0
- TclNumber **maxlength** = 0
- TclNumber **recurse_necessary** = 0
- TclNumber **unit** = 10
- TclArray **widths**

8.3.2 Constructor & Destructor Documentation

8.3.2.1 AllHierarchies::AllHierarchies (TclList *args*)

constructor

Definition at line 89 of file allhierarchies.tcl.

8.3.3 Member Function Documentation

8.3.3.1 AllHierarchies::_init_data () [private]

get data from the cdg tool.

Definition at line 133 of file allhierarchies.tcl.

8.3.3.2 AllHierarchies::balanceNode (TclString *id*, TclString *type_id*) [private]

rearrange children of a node so that wide nodes alternate with narrow nodes.

Definition at line 434 of file allhierarchies.tcl.

8.3.3.3 AllHierarchies::C_to_Tcllist (List *list*) [private]

Converts a C-list of types into a Tcl-list of type-ids and filters out 'bot'. Only this function and position_-of_type work with C-data structures. Definition at line 291 of file allhierarchies.tcl.

8.3.3.4 AllHierarchies::center (TclString *id*, TclNumber *level*) [private]

find sort's final x-coordinate.

Definition at line 576 of file allhierarchies.tcl.

8.3.3.5 AllHierarchies::compute_fathers_and_sons (Hierarchy *hierarchy*, TclString *type_id*) [private]

find a types sons and fathers in the hierarchy. (uses the C-function listOfSons) Definition at line 330 of file allhierarchies.tcl.

8.3.3.6 AllHierarchies::compute_graph (TclString *id*) [private]

This function is used only for nontree-graphs.

Definition at line 849 of file allhierarchies.tcl.

8.3.3.7 AllHierarchies::compute_hierarchy (Hierarchy *hierarchy*) [private]

gather all relevant data for displaying a hierarchy.

Definition at line 186 of file allhierarchies.tcl.

8.3.3.8 AllHierarchies::compute_positions (TclString *id*) [private]

calls initial_x_pos and center.

Definition at line 535 of file allhierarchies.tcl.

8.3.3.9 AllHierarchies::display_graph (TclString *id*, TclNumber *level*, TclList *nodes_current_lvl*, TclNumber *x_start*, TclNumber *y_start*, TclNumber *counter*) [private]

display nontree-graphs. This function is used only for nontree-graphs Definition at line 944 of file allhierarchies.tcl.

8.3.3.10 AllHierarchies::drawHierarchy (TclString *id*) [private]

draw hierarchy on its associated canvas

Definition at line 351 of file allhierarchies.tcl.

8.3.3.11 AllHierarchies::drawSort (TclString *id*, TclString *type_id*) [private]

draw one sort on the canvas. This method is only called when the hierarchy is a tree.

Note:
No check prevents us from calling this method on a non-tree hierarchy.

Definition at line 497 of file allhierarchies.tcl.

8.3.3.12 AllHierarchies::getAllHierarchyIds ()

Return list of all hierarchies' names.

Definition at line 1166 of file allhierarchies.tcl.

8.3.3.13 AllHierarchies::importNode (TclString *id*, TclString *type_id*) [private]

fills global arrays with data from cdg.

Definition at line 397 of file allhierarchies.tcl.

8.3.3.14 AllHierarchies::init_data ()

call _init_data if necessary.

Definition at line 120 of file allhierarchies.tcl.

8.3.3.15 AllHierarchies::initial_x_pos (TclString *id*, TclNumber *level*) [private]

sort's initial position.

Definition at line 556 of file allhierarchies.tcl.

8.3.3.16 AllHierarchies::initialize_depth (TclString *id*, TclString *type_id*) [private]

initialize the depth information. This method recurses over all types in a hierarchy and sets their depth information to zero. Definition at line 220 of file allhierarchies.tcl.

8.3.3.17 AllHierarchies::is_hierarchy_a_tree (Hierarchy *hierarchy*, TclString *type_id*) [private]

compute tree property. This method computes whether a hierarchy is a tree. This function is traversing recursivly the complete hierarchy in the following way:

- if the number of fathers is greater than one then the hierarchy is not a tree.

Parameters:

hierarchy a pointer to a HierarchyStruct

type_id an entry in the hierarchy

Definition at line 258 of file allhierarchies.tcl.

8.3.3.18 AllHierarchies::labelEnter (TclCommand *w*, TclNumber *pointerx*, TclNumber *pointery*) [private]

command bound to Enter-event.

Definition at line 1146 of file allhierarchies.tcl.

8.3.3.19 AllHierarchies::labelLeave (TclCommand *w*) [private]

command bound to Leave-event.

Definition at line 1156 of file allhierarchies.tcl.

8.3.3.20 AllHierarchies::ldelete (TclList *list*, TclString *type_id*) [private]

deletes a string from a list.

Todo

this method is in no way specific to AllHierarchies

Definition at line 685 of file allhierarchies.tcl.

8.3.3.21 AllHierarchies::leaves_right (TclString *id*, TclString *type_id*) [private]

move leaves to the rightmost possible position. Leaves are positioned right beside their non-leaf-siblings Definition at line 645 of file allhierarchies.tcl.

8.3.3.22 AllHierarchies::max_type_depth (TclString *id*, TclString *type_id*, TclNumber *depth*) [private]

compute maximal distance between type and 'top'. This function finds a type's depth in trees, and the length of the longest path from top to the type for non-trees: Definition at line 235 of file allhierarchies.tcl.

8.3.3.23 AllHierarchies::new_order (TclString *id*, TclString *String*) [private]

store the new order in 'levels(level)'

Definition at line 664 of file allhierarchies.tcl.

8.3.3.24 AllHierarchies::number_of_fathers_and_sons (Hierarchy *hierarchy*, TclString *type_id*) [private]

stores the number of successors and predecessors each type has

Definition at line 275 of file allhierarchies.tcl.

8.3.3.25 AllHierarchies::position_leaves (TclString *id*, TclNumber *level*) [private]

some nodes are still not correctly positioned.

Definition at line 698 of file allhierarchies.tcl.

8.3.3.26 AllHierarchies::position_leaves2 (TclString *id*, TclString *type_id*) [private]

position leaves between their siblings

Todo
bad method name

Definition at line 727 of file allhierarchies.tcl.

8.3.3.27 AllHierarchies::position_of_type (Hierarchy *hierarchy*, TclString *type_id*) [private]

This is the interface to functions.i

Todo
the documentation does not match the intension of this function

Definition at line 308 of file allhierarchies.tcl.

8.3.3.28 AllHierarchies::set_level_array (TclString *id*, TclString *type_id*) [private]

populate the array levels.

Definition at line 413 of file allhierarchies.tcl.

8.3.3.29 AllHierarchies::stringlength (TclString *id*, TclString *type_id*) [private]

stores types stringlength.

Definition at line 201 of file allhierarchies.tcl.

8.3.3.30 AllHierarchies::xpixels_per_character (TclString *id*) [private]

How many pixels are to be used for a character ?

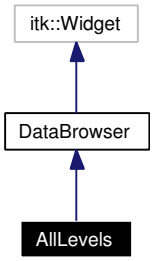
Definition at line 1131 of file allhierarchies.tcl.

The documentation for this class was generated from the following file:

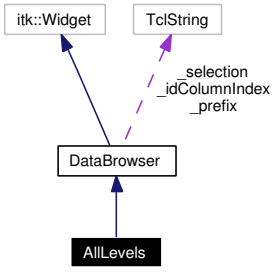
- allhierarchies.tcl

8.4 AllLevels Class Reference

Inheritance diagram for AllLevels:



Collaboration diagram for AllLevels:



8.4.1 Detailed Description

AllLevels - manage CDG grammar levels

Author:
Michael Daum
Id
alllevels.tcl,v 1.37 2004/10/11 13:50:05 micha Exp

Definition at line 18 of file alllevels.tcl.

Public Member Functions

- AllLevels (TclList args)
- getAllLevelIds ()
- getAllSectionIds ()
- getCData (TclString id)
- getCData (Level id)
- getLabels (TclString levelId)
- getMainlevelId ()
- getSelection ()
- init_data ()
- refreshid (TclString id)
- refreshrow (TclNumber row, Level level)

- setIndexedSelection (TclList args)
- setSelection (TclList args)

Protected Member Functions

- _browse_action (TclWidget w)
- _keypress_action (TclWidget w, TclKeyBinding k)
- _motion_action (TclWidget w, TclNumber x, TclNumber y)
- _return_action ()
- _rowtag (TclNumber row)
- _setCount (TclNumber n)
- refreshrow (TclNumber row, TclString item)

Protected Attributes

- TclString _idColumnIndex = ""
- TclString _selection = ""

Private Member Functions

- _init_data ()
- displaybutton_action ()
- editbutton_action ()
- showbutton_action ()
- usebutton_action ()

8.4.2 Constructor & Destructor Documentation

8.4.2.1 AllLevels::AllLevels (TclList *args*)

constructor

Definition at line 50 of file alllevels.tcl.

8.4.3 Member Function Documentation

8.4.3.1 DataBrowser::_browse_action (TclWidget *w*) `[protected, inherited]`

browse slot. This method adjusts the selected file in AllFiles::_selection.

Parameters:

w the widget bound to this slot.

Definition at line 189 of file databrowser.tcl.

8.4.3.2 AllLevels::_init_data () `[private]`

get data from the cdg tool

Definition at line 121 of file alllevels.tcl.

8.4.3.3 DataBrowser::_keypress_action (TclWidget *a*, TclKeyBinding *k*) [protected, inherited]

React to a keypress into the table.

Keys typed by the user are collected into a string, and the row whose id matches that string is selected. Definition at line 421 of file databrowser.tcl.

8.4.3.4 DataBrowser::_motion_action (TclWidget *w*, TclNumber *x*, TclNumber *y*) [protected, inherited]

Default motion slot.

Parameters:

w the widget where the motion was detected

x the x coords of the mouse

y the y coords of the mouse

Reimplemented in AllFiles.

Definition at line 390 of file databrowser.tcl.

References DataBrowser::refreshid().

8.4.3.5 DataBrowser::_return_action () [protected, inherited]

actions to take place on pressing return in the entryfield.

Definition at line 244 of file databrowser.tcl.

8.4.3.6 DataBrowser::_rowtag (TclNumber *row*) [protected, inherited]

colorize the table rows This method is a callback configured to the table in order to colorize the rows. Definition at line 361 of file databrowser.tcl.

8.4.3.7 DataBrowser::_setCount (TclNumber *n*) [protected, inherited]

display the count-label. This number should reflect the number of items selected

Parameters:

n the number to be set

Definition at line 375 of file databrowser.tcl.

8.4.3.8 AllLevels::displaybutton_action () [private]

display the level in the shell

Definition at line 233 of file alllevels.tcl.

8.4.3.9 AllLevels::editbutton_action () [private]

call editor for the declaration

Definition at line 248 of file alllevels.tcl.

8.4.3.10 AllLevels::getAllLevelIds ()

Return list of all levels' names.

Definition at line 325 of file alllevels.tcl.

8.4.3.11 AllLevels::getAllSectionIds ()

Return list of all sections' names.

Definition at line 338 of file alllevels.tcl.

8.4.3.12 DataBrowser::getCData (TclString *id*) [inherited]

abstract methods for retrieving the relevant C or Tcl data values for a given ID

Reimplemented in AllConstraints, AllLexemes, AllNetworks, AllParses, and AllWordgraphs.

8.4.3.13 AllLevels::getCData (Level *id*)

replaces getLevel

Definition at line 316 of file alllevels.tcl.

8.4.3.14 AllLevels::getLabels (TclString *levelId*)

get all labels of a certain level.

Definition at line 277 of file alllevels.tcl.

8.4.3.15 AllLevels::getMainlevelId ()

get the name of the main level.

Definition at line 304 of file alllevels.tcl.

8.4.3.16 DataBrowser::getSelection () [inherited]

return a list of selected ids

Definition at line 292 of file databrowser.tcl.

References DataBrowser::setSelection().

8.4.3.17 AllLevels::init_data ()

call _init_data if necessary.

Reimplemented from DataBrowser.

Definition at line 108 of file alllevels.tcl.

8.4.3.18 DataBrowser::refreshid (TclString *id*) [inherited]

refresh the displayed data for a specific ID

Definition at line 398 of file databrowser.tcl.

Referenced by DataBrowser::_motion_action().

8.4.3.19 DataBrowser::refreshrow (TclNumber *row*, TclString *item*) [protected, inherited]

abstract method called in refreshid

Reimplemented in AllWordgraphs.

8.4.3.20 AllLevels::refreshrow (TclNumber *row*, Level *level*)

fill a row with the values from a specific level

Definition at line 153 of file alllevels.tcl.

8.4.3.21 DataBrowser::setIndexedSelection (TclList *args*) [inherited]

Select one or more rows. A previous selection is cleared; without arguemnts, removes all selections.

ARGS is a list of row indices. Definition at line 332 of file databrowser.tcl.

8.4.3.22 DataBrowser::setSelection (TclList *args*) [inherited]

Select one or more rows. A previous selection is cleared; without arguemnts, removes all selections.

ARGS must be a list of strings without spaces in them. Definition at line 304 of file databrowser.tcl.

Referenced by DataBrowser::getSelection().

8.4.3.23 AllLevels::showbutton_action () [private]

toggle display of selected levels

Definition at line 240 of file alllevels.tcl.

8.4.3.24 AllLevels::usebutton_action () [private]

toggle usage of selected levels.

Definition at line 222 of file alllevels.tcl.

8.4.4 Member Data Documentation

8.4.4.1 TclString DataBrowser::_idColumnIndex = "" [protected, inherited]

column index of table that contains selectable ids (used in _browse_action)

Definition at line 87 of file databrowser.tcl.

8.4.4.2 TclString DataBrowser::_selection = "" [protected, inherited]

string of current selected row ids

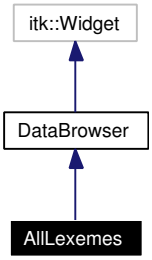
Definition at line 84 of file databrowser.tcl.

The documentation for this class was generated from the following file:

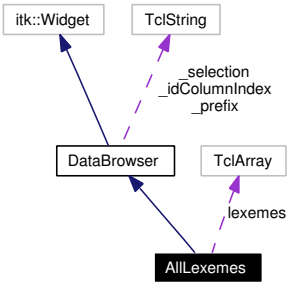
- alllevels.tcl

8.5 AllLexemes Class Reference

Inheritance diagram for AllLexemes:



Collaboration diagram for AllLexemes:



8.5.1 Detailed Description

AllLexemes - manage the CDG lexicon.

Author:
Michael Daum
Id
alllexemes.tcl,v 1.31 2004/10/11 13:50:05 micha Exp

Definition at line 18 of file alllexemes.tcl.

Public Member Functions

- AllLexemes (TclList args)
- getAllWords ()
- getCData (TclString id)
- getSelection ()
- init_data ()
- refreshid (TclString id)
- refreshrow (TclNumber row, LexiconItem lexeme)
- setIndexedSelection (TclList args)
- setSelection (TclList args)

Protected Member Functions

- _browse_action (TclWidget w)
- _keypress_action (TclWidget w, TclKeyBinding k)
- _motion_action (TclWidget w, TclNumber x, TclNumber y)
- _return_action ()
- _rowtag (TclNumber row)
- _setCount (TclNumber n)
- refreshrow (TclNumber row, TclString item)

Protected Attributes

- TclString _idColumnIndex = ""
- TclString _selection = ""

Private Member Functions

- _init_data ()
- displaybutton_action ()
- editbutton_action ()

Private Attributes

- TclArray **lexemes**

8.5.2 Constructor & Destructor Documentation

8.5.2.1 AllLexemes::AllLexemes (TclList *args*)

constructor

Definition at line 44 of file alllexemes.tcl.

8.5.3 Member Function Documentation

8.5.3.1 DataBrowser::_browse_action (TclWidget *w*) `[protected, inherited]`

browse slot. This method adjusts the selected file in AllFiles::_selection.

Parameters:

w the widget bound to this slot.

Definition at line 189 of file databrowser.tcl.

8.5.3.2 AllLexemes::_init_data () `[private]`

get data from the cdg tool

Definition at line 100 of file alllexemes.tcl.

8.5.3.3 DataBrowser::_keypress_action (TclWidget *a*, TclKeyBinding *k*) [protected, inherited]

React to a keypress into the table.

Keys typed by the user are collected into a string, and the row whose id matches that string is selected. Definition at line 421 of file databrowser.tcl.

8.5.3.4 DataBrowser::_motion_action (TclWidget *w*, TclNumber *x*, TclNumber *y*) [protected, inherited]

Default motion slot.

Parameters:

w the widget where the motion was detected

x the x coords of the mouse

y the y coords of the mouse

Reimplemented in AllFiles.

Definition at line 390 of file databrowser.tcl.

References DataBrowser::refreshid().

8.5.3.5 DataBrowser::_return_action () [protected, inherited]

actions to take place on pressing return in the entryfield.

Definition at line 244 of file databrowser.tcl.

8.5.3.6 DataBrowser::_rowtag (TclNumber *row*) [protected, inherited]

colorize the table rows This method is a callback configured to the table in order to colorize the rows. Definition at line 361 of file databrowser.tcl.

8.5.3.7 DataBrowser::_setCount (TclNumber *n*) [protected, inherited]

display the count-label. This number should reflect the number of items selected

Parameters:

n the number to be set

Definition at line 375 of file databrowser.tcl.

8.5.3.8 AllLexemes::displaybutton_action () [private]

display the lexeme in the shell

Definition at line 178 of file alllexemes.tcl.

8.5.3.9 AllLexemes::editbutton_action () [private]

call editor for the declaration.

Definition at line 185 of file alllexemes.tcl.

8.5.3.10 AllLexemes::getAllWords ()

Return list of all known words.

Definition at line 215 of file alllexemes.tcl.

8.5.3.11 AllLexemes::getCData (TclString *id*)

Wrapper to allow use of base class definition of refreshid.

Reimplemented from DataBrowser.

Definition at line 150 of file alllexemes.tcl.

8.5.3.12 DataBrowser::getSelection () [inherited]

return a list of selected ids

Definition at line 292 of file databrowser.tcl.

References DataBrowser::setSelection().

8.5.3.13 AllLexemes::init_data ()

call _init_data if necessary

Reimplemented from DataBrowser.

Definition at line 87 of file alllexemes.tcl.

8.5.3.14 DataBrowser::refreshid (TclString *id*) [inherited]

refresh the displayed data for a specific ID

Definition at line 398 of file databrowser.tcl.

Referenced by DataBrowser::_motion_action().

8.5.3.15 DataBrowser::refreshrow (TclNumber *row*, TclString *item*) [protected, inherited]

abstract method called in refreshid

Reimplemented in AllWordgraphs.

8.5.3.16 AllLexemes::refreshrow (TclNumber *row*, LexiconItem *lexeme*)

fill a row with the values from a specific lexeme.

Definition at line 158 of file alllexemes.tcl.

8.5.3.17 DataBrowser::setIndexedSelection (TclList *args*) [inherited]

Select one or more rows. A previous selection is cleared; without arguemnts, removes all selections. ARGS is a list of row indices. Definition at line 332 of file databrowser.tcl.

8.5.3.18 DataBrowser::setSelection (TclList *args*) [inherited]

Select one or more rows. A previous selection is cleared; without arguemnts, removes all selections. ARGS must be a list of strings without spaces in them. Definition at line 304 of file databrowser.tcl. Referenced by DataBrowser::getSelection().

8.5.4 Member Data Documentation

8.5.4.1 TclString DataBrowser::_idColumnIndex = "" [protected, inherited]

column index of table that contains selectable ids (used in _browse_action)
Definition at line 87 of file databrowser.tcl.

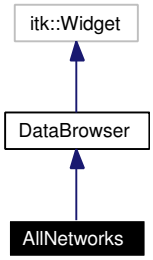
8.5.4.2 TclString DataBrowser::_selection = "" [protected, inherited]

string of current selected row ids
Definition at line 84 of file databrowser.tcl.
The documentation for this class was generated from the following file:

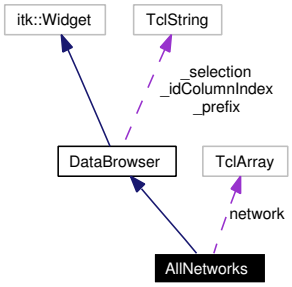
- alllexemes.tcl

8.6 AllNetworks Class Reference

Inheritance diagram for AllNetworks:



Collaboration diagram for AllNetworks:



8.6.1 Detailed Description

AllNetworks - manage all constraint networks.

Author:

Michael Daum (see also AUTHORS and THANKS for more)

Id

allnetworks.tcl,v 1.59 2004/10/11 13:50:05 micha Exp

Definition at line 18 of file allnetworks.tcl.

Public Member Functions

- getAllNetIds ()
- getCData (TclString id)
- getSelection ()
- refreshid (TclString id)
- refreshrow (TclNumber row, ConstraintNet net)
- selectIdsOfWg (TclString wgId)
- setIndexedSelection (TclList args)
- setSelection (TclList args)

Protected Member Functions

- _browse_action (TclWidget w)
- _keypress_action (TclWidget w, TclKeyBinding k)
- _motion_action (TclWidget w, TclNumber x, TclNumber y)
- _return_action ()
- _rowtag (TclNumber row)
- _setCount (TclNumber n)
- AllNetworks (TclList args)
- init_data ()
- refreshrow (TclNumber row, TclString item)

Protected Attributes

- TclString _idColumnIndex = ""
- TclString _selection = ""

Private Member Functions

- deletebutton_action ()
- detailsbutton_action ()
- frobbutton_action ()
- glsbutton_action ()
- netsearchbutton_action ()

Private Attributes

- TclArray **network**

8.6.2 Constructor & Destructor Documentation

8.6.2.1 AllNetworks::AllNetworks (TclList *args*) [protected]

constructor

Definition at line 51 of file allnetworks.tcl.

8.6.3 Member Function Documentation

8.6.3.1 DataBrowser::_browse_action (TclWidget *w*) [protected, inherited]

browse slot. This method adjusts the selected file in AllFiles::_selection.

Parameters:

w the widget bound to this slot.

Definition at line 189 of file databrowser.tcl.

8.6.3.2 DataBrowser::_keypress_action (TclWidget *a*, TclKeyBinding *k*) [protected, inherited]

React to a keypress into the table.

Keys typed by the user are collected into a string, and the row whose id matches that string is selected. Definition at line 421 of file databrowser.tcl.

8.6.3.3 DataBrowser::_motion_action (TclWidget *w*, TclNumber *x*, TclNumber *y*) [protected, inherited]

Default motion slot.

Parameters:

w the widget where the motion was detected

x the x coords of the mouse

y the y coords of the mouse

Reimplemented in AllFiles.

Definition at line 390 of file databrowser.tcl.

References DataBrowser::refreshid().

8.6.3.4 DataBrowser::_return_action () [protected, inherited]

actions to take place on pressing return in the entryfield.

Definition at line 244 of file databrowser.tcl.

8.6.3.5 DataBrowser::_rowtag (TclNumber *row*) [protected, inherited]

colorize the table rows This method is a callback configured to the table in order to colorize the rows. Definition at line 361 of file databrowser.tcl.

8.6.3.6 DataBrowser::_setCount (TclNumber *n*) [protected, inherited]

display the count-label. This number should reflect the number of items selected

Parameters:

n the number to be set

Definition at line 375 of file databrowser.tcl.

8.6.3.7 AllNetworks::deletebutton_action () [private]

action taking place when the deletebutton is pressed.

Definition at line 241 of file allnetworks.tcl.

8.6.3.8 AllNetworks::detailsbutton_action () [private]

action taking place when the detailsbutton is pressed.

Definition at line 253 of file allnetworks.tcl.

8.6.3.9 AllNetworks::frobbutton_action () [private]

action taking place when the frobbutton is pressed

Definition at line 216 of file allnetworks.tcl.

8.6.3.10 AllNetworks::getAllNetIds ()

Return list of all nets' names.

Definition at line 303 of file allnetworks.tcl.

8.6.3.11 AllNetworks::getCData (TclString *id*)

substitute for getnet

Reimplemented from DataBrowser.

Definition at line 262 of file allnetworks.tcl.

8.6.3.12 DataBrowser::getSelection () [inherited]

return a list of selected ids

Definition at line 292 of file databrowser.tcl.

References DataBrowser::setSelection().

8.6.3.13 AllNetworks::glsbutton_action () [private]

action taking place when the glsbutton is pressed.

Definition at line 227 of file allnetworks.tcl.

8.6.3.14 AllNetworks::init_data () [protected]

get the data from the cdg-tool.

Reimplemented from DataBrowser.

Definition at line 123 of file allnetworks.tcl.

8.6.3.15 AllNetworks::netsearchbutton_action () [private]

action taking place when the searchforparsebutton is pressed

Definition at line 207 of file allnetworks.tcl.

8.6.3.16 DataBrowser::refreshid (TclString *id*) [inherited]

refresh the displayed data for a specific ID

Definition at line 398 of file databrowser.tcl.

Referenced by DataBrowser::_motion_action().

8.6.3.17 DataBrowser::refreshrow (TclNumber *row*, TclString *item*) [protected, inherited]

abstract method called in refreshid

Reimplemented in AllWordgraphs.

8.6.3.18 AllNetworks::refreshrow (TclNumber *row*, ConstraintNet *net*)

fill a row with the values from a specific net

Definition at line 166 of file allnetworks.tcl.

8.6.3.19 AllNetworks::selectIdsOfWg (TclString *wgId*)

select nets based on a specified wordgraph. previous selections are cleared Definition at line 280 of file allnetworks.tcl.

8.6.3.20 DataBrowser::setIndexedSelection (TclList *args*) [inherited]

Select one or more rows. A previous selection is cleared; without arguemnts, removes all selections.

ARGS is a list of row indices. Definition at line 332 of file databrowser.tcl.

8.6.3.21 DataBrowser::setSelection (TclList *args*) [inherited]

Select one or more rows. A previous selection is cleared; without arguemnts, removes all selections.

ARGS must be a list of strings without spaces in them. Definition at line 304 of file databrowser.tcl.

Referenced by DataBrowser::getSelection().

8.6.4 Member Data Documentation

8.6.4.1 TclString DataBrowser::_idColumnIndex = "" [protected, inherited]

column index of table that contains selectable ids (used in _browse_action)

Definition at line 87 of file databrowser.tcl.

8.6.4.2 TclString DataBrowser::_selection = "" [protected, inherited]

string of current selected row ids

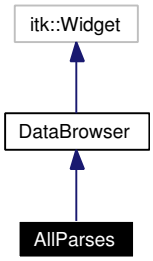
Definition at line 84 of file databrowser.tcl.

The documentation for this class was generated from the following file:

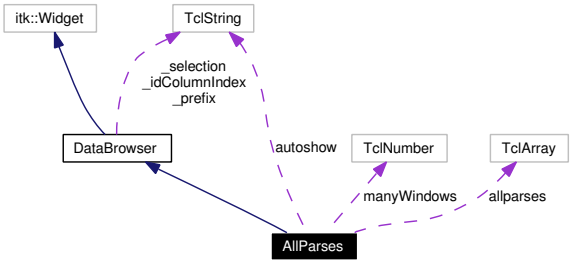
- allnetworks.tcl

8.7 AllParses Class Reference

Inheritance diagram for AllParses:



Collaboration diagram for AllParses:



8.7.1 Detailed Description

AllParses - manage parser results.

Author:
Dietmar Fünning, Kilian A. Foth (see also AUTHORS and THANKS for more)

Id
allparses.tcl,v 1.63 2004/10/11 13:50:05 micha Exp

Definition at line 17 of file allparses.tcl.

Public Member Functions

- addParse (Parse parse)
- addParsesOfNet (TclString netId)
- deleteparse (TclList args)
- getAllAnnoIds ()
- getAllparseIds ()
- getCData (TclString id)
- getParseForAnno (TclString anno)
- getParseForLattice (TclString id)
- getSelection ()
- handleICinteraction ()
- handlePartialResult (Parse parse_pointer)

- refreshid (TclString id)
- refreshrow (TclNumber row, Parse parse)
- reset ()
- selectIdsOfLattice (TclString netId)
- selectIdsOfNet (TclString netId)
- setIndexedSelection (TclList args)
- setManyWindows (TclNumber x)
- setSelection (TclList args)
- showparse (TclList args)
- verifyparse (TclList args)

Public Attributes

- TclString **autoshow** = "valid"

Protected Member Functions

- _browse_action (TclWidget w)
- _keypress_action (TclWidget w, TclKeyBinding k)
- _motion_action (TclWidget w, TclNumber x, TclNumber y)
- _return_action ()
- _rowtag (TclNumber row)
- _setCount (TclNumber n)
- AllParses (TclList args)
- init_data ()
- refreshrow (TclNumber row, TclString item)

Protected Attributes

- TclString _idColumnIndex = ""
- TclString _selection = ""

Private Member Functions

- deletebutton_action ()
- treebutton_action ()
- verifybutton_action ()

Private Attributes

- TclArray **allparses**
- TclNumber **manyWindows** = 0

8.7.2 Constructor & Destructor Documentation

8.7.2.1 AllParses::AllParses (TclList *args*) `[protected]`

constructor

Definition at line 70 of file allparses.tcl.

8.7.3 Member Function Documentation

8.7.3.1 DataBrowser::_browse_action (TclWidget *w*) [protected, inherited]

browse slot. This method adjusts the selected file in AllFiles::_selection.

Parameters:

w the widget bound to this slot.

Definition at line 189 of file databrowser.tcl.

8.7.3.2 DataBrowser::_keypress_action (TclWidget *a*, TclKeyBinding *k*) [protected, inherited]

React to a keypress into the table.

Keys typed by the user are collected into a string, and the row whose id matches that string is selected.
Definition at line 421 of file databrowser.tcl.

8.7.3.3 DataBrowser::_motion_action (TclWidget *w*, TclNumber *x*, TclNumber *y*) [protected, inherited]

Default motion slot.

Parameters:

w the widget where the motion was detected

x the x coords of the mouse

y the y coords of the mouse

Reimplemented in AllFiles.

Definition at line 390 of file databrowser.tcl.

References DataBrowser::refreshid().

8.7.3.4 DataBrowser::_return_action () [protected, inherited]

actions to take place on pressing return in the entryfield.

Definition at line 244 of file databrowser.tcl.

8.7.3.5 DataBrowser::_rowtag (TclNumber *row*) [protected, inherited]

colorize the table rows This method is a callback configured to the table in order to colorize the rows.
Definition at line 361 of file databrowser.tcl.

8.7.3.6 DataBrowser::_setCount (TclNumber *n*) [protected, inherited]

display the count-label. This number should reflect the number of items selected

Parameters:

n the number to be set

Definition at line 375 of file databrowser.tcl.

8.7.3.7 AllParses::addParse (Parse *parse*)

add parse

Definition at line 501 of file allparses.tcl.

8.7.3.8 AllParses::addParsesOfNet (TclString *netId*)

add all parses of a given net

Definition at line 480 of file allparses.tcl.

8.7.3.9 AllParses::deletebutton_action () [private]

action taking place when pressing the deletebutton

Definition at line 210 of file allparses.tcl.

Referenced by verifybutton_action().

8.7.3.10 AllParses::deleteparse (TclList *args*)

delete given parses

Definition at line 219 of file allparses.tcl.

8.7.3.11 AllParses::getAllAnnoIds ()

get names of all known annotations

Definition at line 265 of file allparses.tcl.

Referenced by getAllparseIds().

8.7.3.12 AllParses::getAllparseIds ()

get all parseIds

Definition at line 258 of file allparses.tcl.

References getAllAnnoIds().

8.7.3.13 AllParses::getCData (TclString *id*)

Get parse of a given id by looking it up in allparses().

If no such Parse exists, try to create one from the annotation with the same name and store it there, then return it.

Reimplemented from DataBrowser.

Definition at line 307 of file allparses.tcl.

8.7.3.14 AllParses::getParseForAnno (TclString *anno*)

Return a parse created from the C annotation ANNO.

Definition at line 280 of file allparses.tcl.

8.7.3.15 AllParses::getParseForLattice (TclString *id*)

Get a parse from an annotation of lattice ID.

This is the more deperate version of getParseForAnno: instead of annotations with the name ID, it accepts annotations whose lattice's name is ID. You should have tried getParseForAnno first. Definition at line 345 of file allparses.tcl.

8.7.3.16 DataBrowser::getSelection () `[inherited]`

return a list of selected ids

Definition at line 292 of file databrowser.tcl.

References DataBrowser::setSelection().

8.7.3.17 AllParses::handleICinteraction ()

This function is called through the hook "IC interaction". It returns another word typed by the user, or "" to denote "stop processing". Definition at line 585 of file allparses.tcl.

8.7.3.18 AllParses::handlePartialResult (Parse *parse_pointer*)

This function is called whenever a solution method has yielded another partial result. Definition at line 522 of file allparses.tcl.

8.7.3.19 AllParses::init_data () `[protected]`

initialize with all available Parses

Reimplemented from DataBrowser.

Definition at line 140 of file allparses.tcl.

8.7.3.20 DataBrowser::refreshid (TclString *id*) `[inherited]`

refresh the displayed data for a specific ID

Definition at line 398 of file databrowser.tcl.

Referenced by DataBrowser::_motion_action().

8.7.3.21 DataBrowser::refreshrow (TclNumber *row*, TclString *item*) `[protected, inherited]`

abstract method called in refreshid

Reimplemented in AllWordgraphs.

8.7.3.22 AllParses::refreshrow (TclNumber *row*, Parse *parse*)

refresh a row with a specified parse.

Definition at line 120 of file allparses.tcl.

8.7.3.23 AllParses::reset ()

Called when the CDG grammar changes and all nets, parses etc. become invalid. It deletes all Tcl parses, their VisParses, the corresponding ParseTrees, etc.

If Tcl ever decides to provide graphical representations of constraint nets or lexicon entries, these must be cleared here as well. Definition at line 626 of file allparses.tcl.

Referenced by setManyWindows().

8.7.3.24 AllParses::selectIdsOfLattice (TclString *latticeId*)

select parses of a specified Lattice previous selections are cleared Definition at line 384 of file allparses.tcl.

8.7.3.25 AllParses::selectIdsOfNet (TclString *netId*)

select parses of a specified netId. previous selections are cleared Definition at line 363 of file allparses.tcl.

8.7.3.26 DataBrowser::setIndexedSelection (TclList *args*) [inherited]

Select one or more rows. A previous selection is cleared; without arguemnts, removes all selections.

ARGS is a list of row indices. Definition at line 332 of file databrowser.tcl.

8.7.3.27 AllParses::setManyWindows (TclNumber *x*)

Switch display of parses in one/many windows.

Definition at line 614 of file allparses.tcl.

References reset().

8.7.3.28 DataBrowser::setSelection (TclList *args*) [inherited]

Select one or more rows. A previous selection is cleared; without arguemnts, removes all selections.

ARGS must be a list of strings without spaces in them. Definition at line 304 of file databrowser.tcl.

Referenced by DataBrowser::getSelection().

8.7.3.29 AllParses::showparse (TclList *args*)

Display parses graphically.

All parameters are interpreted as the names of parses. If no parse of the given name exists, one is constructed if possible, preferably from an annotation with the same name, or if that fails, from an annotation of the lattice of the same name.

If a parameter has the form parse0:1,2,3 or similar, then not only is parse0 displayed, but the edges 1, 2, and 3 are immediately highlighted. Definition at line 414 of file allparses.tcl.

8.7.3.30 AllParses::treebutton_action () [private]

actions to take place on pressing the detailsbutton

Definition at line 196 of file allparses.tcl.

References verifybutton_action().

8.7.3.31 AllParses::verifybutton_action () [private]

actions to take place on pressing the detailsbutton

Definition at line 203 of file allparses.tcl.

References deletebutton_action().

Referenced by treebutton_action().

8.7.3.32 AllParses::verifyparse (TclList *args*)

delete given parses

Definition at line 242 of file allparses.tcl.

8.7.4 Member Data Documentation

8.7.4.1 TclString DataBrowser::_idColumnIndex = "" [protected, inherited]

column index of table that contains selectable ids (used in _browse_action)

Definition at line 87 of file databrowser.tcl.

8.7.4.2 TclString DataBrowser::_selection = "" [protected, inherited]

string of current selected row ids

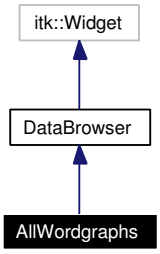
Definition at line 84 of file databrowser.tcl.

The documentation for this class was generated from the following file:

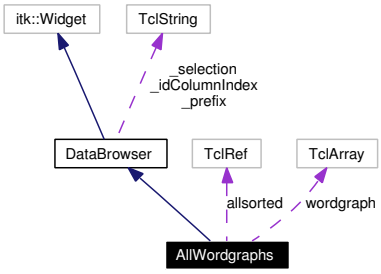
- allparses.tcl

8.8 AllWordgraphs Class Reference

Inheritance diagram for AllWordgraphs:



Collaboration diagram for AllWordgraphs:



8.8.1 Detailed Description

AllWordgraphs - manage all grammar wordgraphs.

Author:

Michael Daum (see also AUTHORS and THANKS for more)

Id

allwordgraphs.tcl,v 1.70 2004/10/13 06:57:13 micha Exp

Definition at line 17 of file allwordgraphs.tcl.

Public Member Functions

- AllWordgraphs (TclList args)
- getAllLatticeIds ()
- getCData (TclString id)
- getNext (TclString latticeId)
- getPrev (TclString latticeId)
- getSelection ()
- init_data ()
- refreshid (TclString id)
- refreshrow (TclNumber row, TclString id)
- setIndexedSelection (TclList args)
- setSelection (TclList args)

Protected Member Functions

- _browse_action (TclWidget w)
- _keypress_action (TclWidget w, TclKeyBinding k)
- _motion_action (TclWidget w, TclNumber x, TclNumber y)
- _return_action ()
- _rowtag (TclNumber row)
- _setCount (TclNumber n)

Protected Attributes

- TclString _idColumnIndex = ""
- TclString _selection = ""

Private Member Functions

- _init_data ()
- annobutton_action ()
- icbutton_action ()
- interactivebutton_action ()
- newbutton_action ()
- newnet ()

Private Attributes

- TclRef allsorted
- TclArray wordgraph

8.8.2 Constructor & Destructor Documentation

8.8.2.1 AllWordgraphs::AllWordgraphs (TclList *args*)

constructor

Definition at line 63 of file allwordgraphs.tcl.

8.8.3 Member Function Documentation

8.8.3.1 DataBrowser::_browse_action (TclWidget *w*) `[protected, inherited]`

browse slot. This method adjusts the selected file in AllFiles::_selection.

Parameters:

w the widget bound to this slot.

Definition at line 189 of file databrowser.tcl.

8.8.3.2 AllWordgraphs::_init_data () [private]

get the data from the cdg-tool

Definition at line 176 of file allwordgraphs.tcl.

8.8.3.3 DataBrowser::_keypress_action (TclWidget *a*, TclKeyBinding *k*) [protected, inherited]

React to a keypress into the table.

Keys typed by the user are collected into a string, and the row whose id matches that string is selected.
Definition at line 421 of file databrowser.tcl.

8.8.3.4 DataBrowser::_motion_action (TclWidget *w*, TclNumber *x*, TclNumber *y*) [protected, inherited]

Default motion slot.

Parameters:

w the widget where the motion was detected

x the x coords of the mouse

y the y coords of the mouse

Reimplemented in AllFiles.

Definition at line 390 of file databrowser.tcl.

References DataBrowser::refreshid().

8.8.3.5 DataBrowser::_return_action () [protected, inherited]

actions to take place on pressing return in the entryfield.

Definition at line 244 of file databrowser.tcl.

8.8.3.6 DataBrowser::_rowtag (TclNumber *row*) [protected, inherited]

colorize the table rows This method is a callback configured to the table in order to colorize the rows.
Definition at line 361 of file databrowser.tcl.

8.8.3.7 DataBrowser::_setCount (TclNumber *n*) [protected, inherited]

display the count-label. This number should reflect the number of items selected

Parameters:

n the number to be set

Definition at line 375 of file databrowser.tcl.

8.8.3.8 AllWordgraphs::annobutton_action () [private]

actions to take place on pressing the annobutton

Definition at line 229 of file allwordgraphs.tcl.

8.8.3.9 AllWordgraphs::getAllLatticeIds ()

Return list of all lattices' names.

Definition at line 352 of file allwordgraphs.tcl.

8.8.3.10 AllWordgraphs::getCData (TclString *id*)

get pointer to a lattice_id, cache it in array wordgraph

Reimplemented from DataBrowser.

Definition at line 303 of file allwordgraphs.tcl.

8.8.3.11 AllWordgraphs::getNext (TclString *latticeId*)

get the lattice next after $id.

Definition at line 320 of file allwordgraphs.tcl.

8.8.3.12 AllWordgraphs::getPrev (TclString *latticeId*)

Show the annotation for the lattice above $id.

Definition at line 336 of file allwordgraphs.tcl.

8.8.3.13 DataBrowser::getSelection () [inherited]

return a list of selected ids

Definition at line 292 of file databrowser.tcl.

References DataBrowser::setSelection().

8.8.3.14 AllWordgraphs::icbutton_action () [private]

actions to take place on pressing the icbutton

Definition at line 253 of file allwordgraphs.tcl.

8.8.3.15 AllWordgraphs::init_data ()

call _init_data if necessary

Reimplemented from DataBrowser.

Definition at line 162 of file allwordgraphs.tcl.

8.8.3.16 AllWordgraphs::interactivebutton_action () [private]

actions to take place on pressing the interactivebutton

Definition at line 268 of file allwordgraphs.tcl.

8.8.3.17 AllWordgraphs::newbutton_action () [private]

actions to take place on pressing the newbutton

Definition at line 222 of file allwordgraphs.tcl.

8.8.3.18 AllWordgraphs::newnet () [private]

actions taking place when the newnetbutton is pressed

Definition at line 293 of file allwordgraphs.tcl.

8.8.3.19 DataBrowser::refreshid (TclString *id*) [inherited]

refresh the displayed data for a specific ID

Definition at line 398 of file databrowser.tcl.

Referenced by DataBrowser::_motion_action().

8.8.3.20 AllWordgraphs::refreshrow (TclNumber *row*, TclString *ptr*)

fill a row with the values from a specific wordgraph.

Parameters:

row row index to insert data

ptr Pointer to lattice_id

Reimplemented from DataBrowser.

Definition at line 140 of file allwordgraphs.tcl.

8.8.3.21 DataBrowser::setIndexedSelection (TclList *args*) [inherited]

Select one or more rows. A previous selection is cleared; without arguemnts, removes all selections. ARGS is a list of row indices. Definition at line 332 of file databrowser.tcl.

8.8.3.22 DataBrowser::setSelection (TclList *args*) [inherited]

Select one or more rows. A previous selection is cleared; without arguemnts, removes all selections. ARGS must be a list of strings without spaces in them. Definition at line 304 of file databrowser.tcl.

Referenced by DataBrowser::getSelection().

8.8.4 Member Data Documentation

8.8.4.1 TclString DataBrowser::_idColumnIndex = "" [protected, inherited]

column index of table that contains selectable ids (used in _browse_action)

Definition at line 87 of file databrowser.tcl.

8.8.4.2 TclString DataBrowser::_selection = "" [protected, inherited]

string of current selected row ids

Definition at line 84 of file databrowser.tcl.

8.8.4.3 TclRef AllWordgraphs::allsorted [private]

list of ids sorted as in the table

Definition at line 54 of file allwordgraphs.tcl.

8.8.4.4 TclArray AllWordgraphs::wordgraph [private]

hash mapping wordgraph ids to their C structure

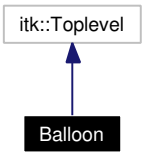
Definition at line 51 of file allwordgraphs.tcl.

The documentation for this class was generated from the following file:

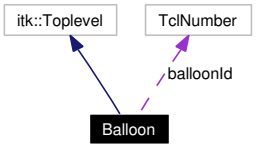
- allwordgraphs.tcl

8.9 Balloon Class Reference

Inheritance diagram for Balloon:



Collaboration diagram for Balloon:



8.9.1 Detailed Description

Balloon - A tooltip class. This is a generic tooltip implementation. Just create a Balloon and switch it on() now and then. A Balloon is displayed if the mouse pointer does not leave the area where it has been initialized within 200 milliseconds by default.

Author:
Michael Daum

Id
balloon.tcl,v 1.9 2004/10/04 07:41:33 foth Exp

Definition at line 21 of file balloon.tcl.

Public Member Functions

- Balloon (TclList args)
- **delay** (TclNumber value=200)
- off ()
- on (TclNumber x, TclNumber y, TclString text, TclList args)

Private Member Functions

- _doit (TclNumber x, TclNumber y)

Private Attributes

- TclNumber balloonId = 0

8.9.2 Constructor & Destructor Documentation

8.9.2.1 Balloon::Balloon (TclList *args*)

constructor

Definition at line 41 of file balloon.tcl.

8.9.3 Member Function Documentation

8.9.3.1 Balloon::_doit (TclNumber *x*, TclNumber *y*) [private]

do the actual display. This method is only called by Balloon::on() when the tcl loop gets idle.

Parameters:

x the x coordinate where the balloon will be displayed

y the y coordinate where the balloon will be displayed

Definition at line 90 of file balloon.tcl.

8.9.3.2 Balloon::off ()

hide a tooltip. Call this method to switch off the tooltip balloon. Definition at line 69 of file balloon.tcl.

8.9.3.3 Balloon::on (TclNumber *x*, TclNumber *y*, TclString *text*, TclList *args*)

show a tooltip. This method actually displays a tooltip with a certain message and position.

Parameters:

x the x coordinate where the balloon will be displayed

y the y coordinate where the balloon will be displayed

text the message text to be shown inside the tooltip

args: If non-empty, draw the balloon immediately (do not wait).

Definition at line 126 of file balloon.tcl.

8.9.4 Member Data Documentation

8.9.4.1 TclNumber Balloon::balloonId = 0 [private]

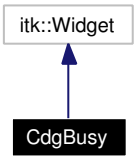
id of the background job. When a balloon is going to be displayed we spawn a job with *after* and store the job id in this variable. Definition at line 35 of file balloon.tcl.

The documentation for this class was generated from the following file:

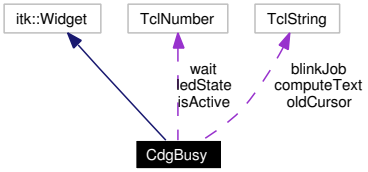
- balloon.tcl

8.10 CdgBusy Class Reference

Inheritance diagram for CdgBusy:



Collaboration diagram for CdgBusy:



8.10.1 Detailed Description

CdgBusy - dialog indicating work in progress. This class is used to display a dialog while a computational intensive operation is scheduled in the background. It stays open as long as the bacground computation has no finished. The user is given an "interrupt" button to signal the background computation that it might terminate earlier. Actually to let the user do that the background computation has to be written cooperatively as we have no real concurrency in tcl. This is accomplished by a frequent call to "update" somewhere in your computation loop.

The busy dialog should also be closed in case of any errors. This is accomplished by overwriting the tcl commands "error", "catch" and "bgerror" in error.tcl.

Author:

Michael Daum (see also AUTHORS and THANKS for more)

Id

busy.tcl,v 1.24 2004/10/11 15:23:31 micha Exp

Definition at line 27 of file busy.tcl.

Public Member Functions

- CdgBusy (TclList args)
- compute (TclString text, TclList args)
- interrupt (TclList args)

Public Attributes

- TclNumber **wait** = 500

Private Member Functions

- blink ()
- enterAction (TclWidget w, TclNumber x, TclNumber y)
- leaveAction ()
- reset ()

Private Attributes

- TclString **blinkJob** = ""
- TclString **computeText** = ""
- TclNumber **isActive** = 0
- TclNumber **ledState** = 0
- TclString **oldCursor** = ""

8.10.2 Constructor & Destructor Documentation

8.10.2.1 CdgBusy::CdgBusy (TclList *args*)

constructor

Definition at line 57 of file busy.tcl.

8.10.3 Member Function Documentation

8.10.3.1 CdgBusy::blink () `[private]`

indicate computation

Definition at line 143 of file busy.tcl.

8.10.3.2 CdgBusy::compute (TclString *text*, TclList *args*)

execute the command. This method first shows the dialog and the spawns an `after` _compute().

Parameters:

text the message to be displayed in the dialog

args the script to be computed.

Returns:

the return value of the evaluated script

Definition at line 89 of file busy.tcl.

References _errorFlag.

8.10.3.3 CdgBusy::enterAction (TclWidget *w*, TclNumber *x*, TclNumber *y*) `[private]`

display the busy balloon. This slot is connected to the <enter> event. Definition at line 179 of file busy.tcl.

8.10.3.4 CdgBusy::interrupt (TclList *args*)

interrupt the current computation.

Definition at line 121 of file busy.tcl.

References _errorFlag.

8.10.3.5 CdgBusy::leaveAction () [private]

disable the busy balloon. This slot is connected to the <leave> event. Definition at line 187 of file busy.tcl.

8.10.3.6 CdgBusy::reset () [private]

reset the blinker to a normal state.

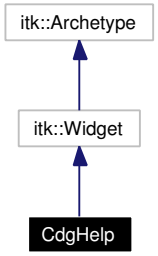
Definition at line 159 of file busy.tcl.

The documentation for this class was generated from the following file:

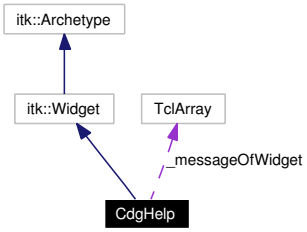
- busy.tcl

8.11 CdgHelp Class Reference

Inheritance diagram for CdgHelp:



Collaboration diagram for CdgHelp:



8.11.1 Detailed Description

CdgHelp - a status line. This class allows to display a help message in a status line whenever the mouse pointer enters a widget.

Author:
Michael Daum (see also AUTHORS and THANKS for more)
Id
help.tcl,v 1.7 2004/02/25 14:40:42 micha Exp

Definition at line 19 of file help.tcl.

Public Member Functions

- CdgHelp (TclList args)
- clear ()
- gethelpstr (TclCommand w)
- sethelpstr (TclCommand w, TclString text, TclList args)
- show (TclCommand w)
- showstr (TclString text)
- unsethelpstr (TclCommand w)

Private Attributes

- TclArray _messageOfWidget

8.11.2 Constructor & Destructor Documentation

8.11.2.1 CdgHelp::CdgHelp (TclList *args*)

constructor

Definition at line 41 of file help.tcl.

8.11.3 Member Function Documentation

8.11.3.1 CdgHelp::clear ()

clear the display

Definition at line 125 of file help.tcl.

8.11.3.2 CdgHelp::gethelpstr (TclCommand *w*)

returns the helpstring of a given widget

Definition at line 97 of file help.tcl.

8.11.3.3 CdgHelp::sethelpstr (TclCommand *w*, TclString *text*, TclList *args*)

set the displayed text when the mouse enters the widget. install two events <Enter> and <Leave> for the widget Definition at line 62 of file help.tcl.

8.11.3.4 CdgHelp::show (TclCommand *w*)

displays the associated helpmessage

Definition at line 111 of file help.tcl.

8.11.3.5 CdgHelp::showstr (TclString *text*)

displays the a noregistered message

Definition at line 104 of file help.tcl.

8.11.3.6 CdgHelp::unsethelpstr (TclCommand *w*)

unset the array-entry, remove the <Enter>- and <Leave>-bindings

Definition at line 85 of file help.tcl.

8.11.4 Member Data Documentation

8.11.4.1 TclArray CdgHelp::_messageOfWidget [private]

a hash mapping widgets to the help message that should be displayed.

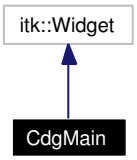
Definition at line 35 of file help.tcl.

The documentation for this class was generated from the following file:

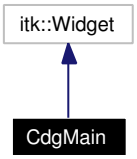
- help.tcl

8.12 CdgMain Class Reference

Inheritance diagram for CdgMain:



Collaboration diagram for CdgMain:



8.12.1 Detailed Description

CdgMain - The root of the XCDG application.

Author:
Michael Daum (see also AUTHORS and THANKS for more)
Id
main.tcl,v 1.59 2004/06/18 07:17:15 foth Exp

Definition at line 17 of file main.tcl.

Public Member Functions

- balloon (TclList args)
- busy (TclList args)
- CdgMain (TclList args)
- confirm (TclString text, TclCommand master=".")
- constraints (TclList args)
- **editor** (TclString value="gnuclient %f")
- files (TclList args)
- grammarpath (TclString value="")
- help (TclList args)
- hierarchies (TclList args)
- levels (TclList args)
- lexemes (TclList args)
- menu (TclList args)
- networks (TclList args)
- not_yet (TclCommand master=".")
- parses (TclList args)
- prefs (TclList args)

- printdialog (TclList args)
- question (TclString text, TclCommand master=".")
- shell (TclList args)
- tabno (TclList args)
- wordgraphs (TclList args)

8.12.2 Constructor & Destructor Documentation

8.12.2.1 CdgMain::CdgMain (TclList *args*)

constructor

Definition at line 50 of file main.tcl.

8.12.3 Member Function Documentation

8.12.3.1 CdgMain::balloon (TclList *args*)

delegate commands to the balloon-component

Definition at line 475 of file main.tcl.

8.12.3.2 CdgMain::busy (TclList *args*)

delegate commands to the dialog-component

Definition at line 397 of file main.tcl.

8.12.3.3 CdgMain::confirm (TclString *text*, TclCommand *master* = " . ")

launch a confirm dialog.

Definition at line 453 of file main.tcl.

8.12.3.4 CdgMain::constraints (TclList *args*)

delegate commands to the constraints component

Definition at line 425 of file main.tcl.

8.12.3.5 CdgMain::files (TclList *args*)

delegate commands to the files-component

Definition at line 404 of file main.tcl.

8.12.3.6 CdgMain::grammarpath (TclString *value* = "")

option -grammarpath.

Definition at line 503 of file main.tcl.

8.12.3.7 CdgMain::help (TclList *args*)

delegate commands to the help-component
Definition at line 376 of file main.tcl.

8.12.3.8 CdgMain::hierarchies (TclList *args*)

delegate commands to the hierarchies component
Definition at line 439 of file main.tcl.

8.12.3.9 CdgMain::levels (TclList *args*)

delegate commands to the levels component
Definition at line 418 of file main.tcl.

8.12.3.10 CdgMain::lexemes (TclList *args*)

delegate commands to the lexemes component
Definition at line 432 of file main.tcl.

8.12.3.11 CdgMain::menu (TclList *args*)

delegate commands to the menu-component
Definition at line 383 of file main.tcl.

8.12.3.12 CdgMain::networks (TclList *args*)

delegate commands to the networks-component
Definition at line 362 of file main.tcl.

8.12.3.13 CdgMain::not_yet (TclCommand *master* = " . ")

launch the confirm-dialog with the 'Under Construction' Message.
Definition at line 489 of file main.tcl.

8.12.3.14 CdgMain::parses (TclList *args*)

delegate commands to the parses-component
Definition at line 411 of file main.tcl.

8.12.3.15 CdgMain::prefs (TclList *args*)

delegate commands to the prefs-component
Definition at line 482 of file main.tcl.

8.12.3.16 CdgMain::printdialog (TclList *args*)

delegate commands to the printdialog-component

Definition at line 446 of file main.tcl.

8.12.3.17 CdgMain::question (TclString *text*, TclCommand *master* = " . ")

launch a question dialog.

Definition at line 464 of file main.tcl.

8.12.3.18 CdgMain::shell (TclList *args*)

delegate commands to the shell-component

Definition at line 390 of file main.tcl.

8.12.3.19 CdgMain::tabno (TclList *args*)

delegate commands to the tabnotebook component.

Definition at line 496 of file main.tcl.

References tabno().

Referenced by tabno().

8.12.3.20 CdgMain::wordgraphs (TclList *args*)

delegate commands to the wordgraphs-component

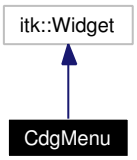
Definition at line 369 of file main.tcl.

The documentation for this class was generated from the following file:

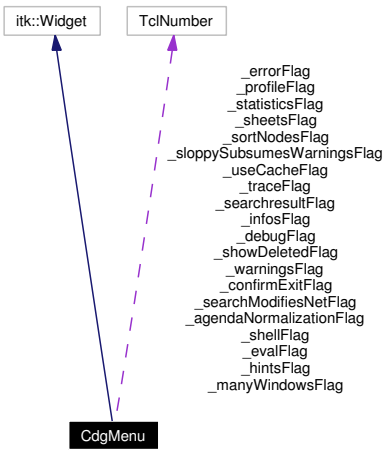
- main.tcl

8.13 CdgMenu Class Reference

Inheritance diagram for CdgMenu:



Collaboration diagram for CdgMenu:



8.13.1 Detailed Description

CdgMenu - menu component of the XCDG application.

Author:
Michael Daum (see also AUTHORS and THANKS for more)
Id
menu.tcl,v 1.33 2004/06/11 07:59:22 foth Exp

Definition at line 17 of file menu.tcl.

Public Member Functions

- CdgMenu (TclList args)
- init_data ()

Private Member Functions

- _about_action ()
- _help_action ()
- _load_action ()

- _load_dir_action ()
- _prefs_action ()
- _reload_action ()
- _reset_action ()
- _run_action ()
- _setFlag (TclString name)
- _setVerbosity ()
- _toggleSheets ()
- _toggleShell ()

Private Attributes

- TclNumber **_agendaNormalizationFlag** = 0
- TclNumber **_confirmExitFlag** = 0
- TclNumber **_debugFlag** = 0
- TclNumber **_errorFlag** = 1
- TclNumber **_evalFlag** = 1
- TclNumber **_hintsFlag** = 1
- TclNumber **_infosFlag** = 1
- TclNumber **_manyWindowsFlag** = 0
- TclNumber **_profileFlag** = 0
- TclNumber **_searchModifiesNetFlag** = 1
- TclNumber **_searchresultFlag** = 1
- TclNumber **_sheetsFlag** = 1
- TclNumber **_shellFlag** = 1
- TclNumber **_showDeletedFlag** = 0
- TclNumber **_sloppySubsumesWarningsFlag** = 0
- TclNumber **_sortNodesFlag** = 0
- TclNumber **_statisticsFlag** = 0
- TclNumber **_traceFlag** = 0
- TclNumber **_useCacheFlag** = 1
- TclNumber **_warningsFlag** = 1

8.13.2 Constructor & Destructor Documentation

8.13.2.1 CdgMenu::CdgMenu (TclList *args*)

constructor

Definition at line 66 of file menu.tcl.

8.13.3 Member Function Documentation

8.13.3.1 CdgMenu::_about_action () `[private]`

startup the about-dialog

Definition at line 468 of file menu.tcl.

8.13.3.2 CdgMenu::_help_action () [private]

startup the help

Definition at line 481 of file menu.tcl.

8.13.3.3 CdgMenu::_load_action () [private]

delegate command to CdgMain

Definition at line 418 of file menu.tcl.

8.13.3.4 CdgMenu::_load_dir_action () [private]

open a directory selection dialog.

Definition at line 425 of file menu.tcl.

8.13.3.5 CdgMenu::_prefs_action () [private]

delegate command to CdgMain

Definition at line 461 of file menu.tcl.

8.13.3.6 CdgMenu::_reload_action () [private]

delegate command to CdgMain

Definition at line 432 of file menu.tcl.

8.13.3.7 CdgMenu::_reset_action () [private]

reset application

Definition at line 439 of file menu.tcl.

8.13.3.8 CdgMenu::_run_action () [private]

load a script and execute it

Definition at line 446 of file menu.tcl.

8.13.3.9 CdgMenu::_setFlag (TclString *name*) [private]

set the named flag according to our local variables

Definition at line 362 of file menu.tcl.

References error().

8.13.3.10 CdgMenu::_setVerbosity () [private]

set the verbosity according to our local variables

Definition at line 401 of file menu.tcl.

8.13.3.11 CdgMenu::_toggleSheets () [private]

show/hide data-sheets

Definition at line 488 of file menu.tcl.

8.13.3.12 CdgMenu::_toggleShell () [private]

show/hide shell

Definition at line 507 of file menu.tcl.

8.13.3.13 CdgMenu::init_data ()

initialize the verbosity flags

Definition at line 329 of file menu.tcl.

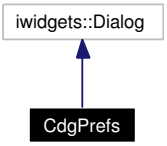
References _errorFlag.

The documentation for this class was generated from the following file:

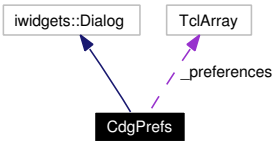
- menu.tcl

8.14 CdgPrefs Class Reference

Inheritance diagram for CdgPrefs:



Collaboration diagram for CdgPrefs:



8.14.1 Detailed Description

CdgPrefs - a preference dialog

Author:
Michael Daum (see also AUTHORS and THANKS for more)

Id
prefs.tcl,v 1.12 2004/02/25 14:40:42 micha Exp

Definition at line 17 of file prefs.tcl.

Public Member Functions

- activate ()
- CdgPrefs (TclList args)

Private Member Functions

- _getData ()
- _ok_action ()
- _setData ()

Private Attributes

- TclArray _preferences

8.14.2 Constructor & Destructor Documentation

8.14.2.1 CdgPrefs::CdgPrefs (TclList *args*)

constructor

Definition at line 40 of file prefs.tcl.

8.14.3 Member Function Documentation

8.14.3.1 CdgPrefs::_getData () [private]

refresh _preferences from main

Definition at line 95 of file prefs.tcl.

8.14.3.2 CdgPrefs::_ok_action () [private]

involved by buttonpress on ok-button

Definition at line 77 of file prefs.tcl.

8.14.3.3 CdgPrefs::_setData () [private]

set configurations

Definition at line 103 of file prefs.tcl.

8.14.3.4 CdgPrefs::activate ()

init data, start dialog

Definition at line 86 of file prefs.tcl.

8.14.4 Member Data Documentation

8.14.4.1 TclArray CdgPrefs::_preferences [private]

prefenrence-data mirrored from CdgMain

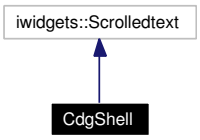
Definition at line 34 of file prefs.tcl.

The documentation for this class was generated from the following file:

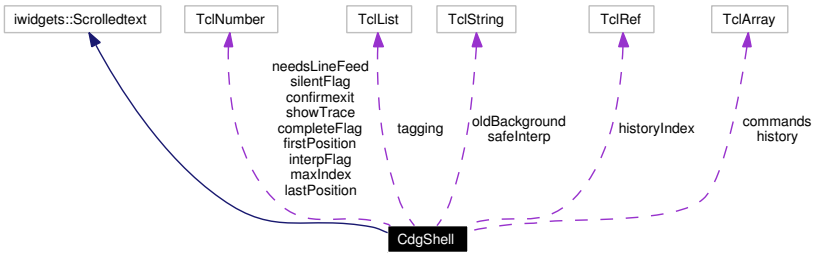
- prefs.tcl

8.15 CdgShell Class Reference

Inheritance diagram for CdgShell:



Collaboration diagram for CdgShell:



8.15.1 Detailed Description

CdgShell - imitate the CDG shell in tcl.

Author:

Michael Daum (see also AUTHORS and THANKS for more)

Id

shell.tcl,v 1.59 2004/09/06 13:40:54 micha Exp

Definition at line 17 of file shell.tcl.

Public Member Functions

- autotag (TclString pattern, TclList args)
- background (TclString value="gray90")
- clear ()
- deleteCmd ()
- fgets ()
- getCmd ()
- **getConfirmExit** ()
- insert (TclString string)
- prompt (TclString value="cdg> ")
- prompt1 (TclString value="cdg> ")
- prompt2 (TclString value="tcltk# ")
- resetCmd (TclList args)
- safeEval (TclCommand cmd)
- safeSource (TclString file)
- **setConfirmExit** (TclBoolean x)
- setCursor (TclString pos)
- updateCmd (TclList args)

Public Attributes

- TclNumber maxIndex = 100
- TclNumber showTrace = 0

Private Member Functions

- b1_action (TclNumber x, TclNumber y)
- backspace_action ()
- CdgShell (TclList args)
- control_c_action ()
- control_d_action ()
- control_l_action ()
- control_q_action ()
- delete_action ()
- double_1_action (TclNumber x, TclNumber y)
- down_action ()
- end_action ()
- fgets_action ()
- getCompletions (TclString name, TclString parameter="")
- home_action ()
- left_action ()
- next_action ()
- prio_action ()
- return_action ()
- shift_down_action ()
- shift_up_action ()
- switch_interp ()
- tab_action ()
- tabtab_action ()
- triple_1_action (TclNumber x, TclNumber y)
- up_action ()

Private Attributes

- TclArray commands
- TclNumber completeFlag = 1
- TclNumber **confirmexit** = 1
- TclNumber firstPosition = 0.0
- TclArray history
- TclRef historyIndex
- TclNumber interpFlag = 1
- TclNumber lastPosition = 0.0
- TclNumber needsLineFeed = 0
- TclString oldBackground = ""
- TclString safeInterp = ""
- TclNumber silentFlag = 0
- TclList tagging = {}

8.15.2 Constructor & Destructor Documentation

8.15.2.1 CdgShell::CdgShell (TclList *args*) [private]

constructor

Definition at line 131 of file shell.tcl.

8.15.3 Member Function Documentation

8.15.3.1 CdgShell::autotag (TclString *pattern*, TclList *args*)

add a syntax highlightening rule.

Definition at line 314 of file shell.tcl.

References safeEval().

Referenced by background().

8.15.3.2 CdgShell::b1_action (TclNumber *x*, TclNumber *y*) [private]

button1 only sets the selection anchor.

Definition at line 938 of file shell.tcl.

Referenced by control_l_action().

8.15.3.3 CdgShell::background (TclString *value* = "gray90")

itk_option: configure the shell background color

Definition at line 307 of file shell.tcl.

References autotag().

8.15.3.4 CdgShell::backspace_action () [private]

restrict BackSpace to the command-line

Definition at line 870 of file shell.tcl.

Referenced by home_action().

8.15.3.5 CdgShell::clear ()

clear the shell screen.

Definition at line 922 of file shell.tcl.

References control_l_action().

8.15.3.6 CdgShell::control_c_action () [private]

kind of control-c

Definition at line 891 of file shell.tcl.

Referenced by delete_action().

8.15.3.7 CdgShell::control_d_action () [private]

terminate the application.

Definition at line 911 of file shell.tcl.

Referenced by control_q_action().

8.15.3.8 CdgShell::control_l_action () [private]

clear the shell screen preventing the commandline content.

Definition at line 929 of file shell.tcl.

References b1_action().

Referenced by clear().

8.15.3.9 CdgShell::control_q_action () [private]

terminate the application.

Definition at line 904 of file shell.tcl.

References control_d_action().

8.15.3.10 CdgShell::delete_action () [private]

delete doesnt delete the selection.

Definition at line 883 of file shell.tcl.

References control_c_action().

8.15.3.11 CdgShell::deleteCmd ()

delete the current command-line

Definition at line 453 of file shell.tcl.

8.15.3.12 CdgShell::double_1_action (TclNumber *x*, TclNumber *y*) [private]

double-1 without changing the insert-position.

Definition at line 952 of file shell.tcl.

8.15.3.13 CdgShell::down_action () [private]

scroll forward thru the commandline history.

Definition at line 821 of file shell.tcl.

8.15.3.14 CdgShell::end_action () [private]

set insert cursor to the end of commandline.

Definition at line 856 of file shell.tcl.

References home_action().

8.15.3.15 CdgShell::fgets ()

get a single line of input from the user without interpreting it. This proc is only used during subprompts, e.g. as used by frobbing Definition at line 467 of file shell.tcl.

8.15.3.16 CdgShell::fgets_action () [private]

alternative <Return> handler, This version gets a line, but does not call cdg commands. Definition at line 492 of file shell.tcl.

8.15.3.17 CdgShell::getCmd ()

get actual command.

Definition at line 1029 of file shell.tcl.

References resetCmd().

8.15.3.18 CdgShell::getCompletions (TclString *name*, TclString *parameter* = "") [private]

get the set of possible commandline completions.

Parameters:

name the string to be completed. If it is "file", then `parameter` is used to fork an `"ls \c -adF"`

parameter used when name = "file"

Returns:

the set of possible completions

Definition at line 546 of file shell.tcl.

8.15.3.19 CdgShell::home_action () [private]

set the insert cursor to the start of commandline.

Definition at line 863 of file shell.tcl.

References backspace_action().

Referenced by end_action().

8.15.3.20 CdgShell::insert (TclString *string*)

print a string to the shell output.

Parameters:
string the message to be printed

Definition at line 390 of file shell.tcl.

Referenced by setCursor().

8.15.3.21 CdgShell::left_action () [private]

restrict cursor movement to the commandline end.

Definition at line 844 of file shell.tcl.

8.15.3.22 CdgShell::next_action () [private]

simply scroll down one page, no cursor positioning.

Definition at line 983 of file shell.tcl.

References shift_up_action().

Referenced by prio_action().

8.15.3.23 CdgShell::prio_action () [private]

simply scroll up one page, no cursor positioning.

Definition at line 976 of file shell.tcl.

References next_action().

8.15.3.24 CdgShell::prompt (TclString *value* = "cdg> ")

itk_option: configure the shell prompt. the actual value will be set to one of -prompt1 or -prompt2

8.15.3.25 CdgShell::prompt1 (TclString *value* = "cdg> ")

itk_option: configure the shell first prompt

8.15.3.26 CdgShell::prompt2 (TclString *value* = "tcltk# ")

itk_option: configure the shell second prompt

8.15.3.27 CdgShell::resetCmd (TclList *args*)

reset commandline.

Definition at line 1038 of file shell.tcl.

Referenced by getCmd().

8.15.3.28 CdgShell::return_action () [private]

the return key binding.

Definition at line 507 of file shell.tcl.

8.15.3.29 CdgShell::safeEval (TclCommand *cmd*)

evaluate a command in the safe interpreter.

Definition at line 322 of file shell.tcl.

Referenced by autotag().

8.15.3.30 CdgShell::safeSource (TclString *file*)

source the commands in the safe interpreter.

Parameters:
file filename of the file to be read

Definition at line 366 of file shell.tcl.

8.15.3.31 CdgShell::setCursor (TclString *pos*)

set the insert cursor.

Parameters:
pos a valid tcl text widget index specification.

Definition at line 380 of file shell.tcl.

References insert().

8.15.3.32 CdgShell::shift_down_action () [private]

simply scroll down one line, no cursor positioning.

Definition at line 997 of file shell.tcl.

References updateCmd().

Referenced by shift_up_action().

8.15.3.33 CdgShell::shift_up_action () [private]

simply scroll up one line, no cursor positioning.

Definition at line 990 of file shell.tcl.

References shift_down_action().

Referenced by next_action().

8.15.3.34 CdgShell::switch_interp () [private]

switch interpreter. This method allows you to access the interpreter that executes the XCDG application. Definition at line 1055 of file shell.tcl.

8.15.3.35 CdgShell::tab_action () [private]

tab action. This is part of the commandline completion suite. Definition at line 676 of file shell.tcl.

8.15.3.36 CdgShell::tabtab_action () [private]

double tab action. This method is part of the commandline completion. Definition at line 609 of file shell.tcl.

8.15.3.37 CdgShell::triple_1_action (TclNumber *x*, TclNumber *y*) [private]

Triple-1 without changing the insert-position.

Definition at line 964 of file shell.tcl.

8.15.3.38 CdgShell::up_action () [private]

step backwards thru the commandline history.

Definition at line 794 of file shell.tcl.

8.15.3.39 CdgShell::updateCmd (TclList *args*)

expand commandline if needed.

Definition at line 1004 of file shell.tcl.

Referenced by shift_down_action().

8.15.4 Member Data Documentation

8.15.4.1 TclArray CdgShell::commands [private]

array of additional commands

Definition at line 63 of file shell.tcl.

8.15.4.2 TclNumber CdgShell::completeFlag = 1 [private]

boolean flag with 0: command incomplete, 1: complete

Definition at line 49 of file shell.tcl.

8.15.4.3 TclNumber CdgShell::firstPosition = 0.0 [private]

start of the commandline

Definition at line 69 of file shell.tcl.

8.15.4.4 TclArray CdgShell::history [private]

shell command history

Definition at line 38 of file shell.tcl.

8.15.4.5 TclRef CdgShell::historyIndex [private]

index in the shell command history

Definition at line 41 of file shell.tcl.

8.15.4.6 TclNumber CdgShell::interpFlag = 1 [private]

boolean flag with 0: application, 1: safe shell. the ctrl-t key lets you switch between the two interpreters. the application interpreter is the one executing the XCDG Definition at line 54 of file shell.tcl.

8.15.4.7 TclNumber CdgShell::lastPosition = 0.0 [private]

end of the commandline

Definition at line 72 of file shell.tcl.

8.15.4.8 TclNumber CdgShell::maxIndex = 100

the maximum nr of commands stored in the history

Definition at line 45 of file shell.tcl.

8.15.4.9 TclNumber CdgShell::needsLineFeed = 0 [private]

flag indicating wether next insert deletes line

Definition at line 78 of file shell.tcl.

8.15.4.10 TclString CdgShell::oldBackground = "" [private]

temporarily store the old shell background color here

Definition at line 75 of file shell.tcl.

8.15.4.11 TclString CdgShell::safeInterp = "" [private]

interpreter where all commands are executed

Definition at line 57 of file shell.tcl.

8.15.4.12 TclNumber CdgShell::showTrace = 0

flag to show a tcl error trace or not.

Definition at line 34 of file shell.tcl.

8.15.4.13 TclNumber CdgShell::silentFlag = 0 [private]

boolean flag 0: print command result, 1: don't

Definition at line 60 of file shell.tcl.

8.15.4.14 TclList CdgShell::tagging = {} [private]

list of autotag information

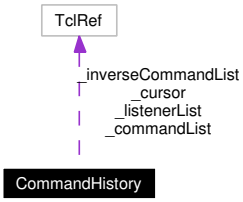
Definition at line 66 of file shell.tcl.

The documentation for this class was generated from the following file:

- shell.tcl

8.16 CommandHistory Class Reference

Collaboration diagram for CommandHistory:



8.16.1 Detailed Description

CommandHistory - implements generic undo-functionality (used for editing Parse-Trees, see class Parse-Tree and VisParses)

Author:
Dietmar Dreyer

$Id commandhistory.tcl $

Definition at line 21 of file commandhistory.tcl.

Public Member Functions

- **add** (TclList command, TclList commandInverse)
- addListener (TclString listener)
- canRedo ()
- canUndo ()
- redo ()
- undo ()

Protected Member Functions

- update ()

Private Attributes

- TclRef _commandList = ""
- TclRef _cursor = -1
- TclRef _inverseCommandList = ""
- TclRef _listenerList = ""

8.16.2 Member Function Documentation

8.16.2.1 CommandHistory::addListener (TclString *listener*)

Registers listener object which will be called whenever a change to the command list happens

Parameters:
listener Object-ID implementing method update

Definition at line 101 of file commandhistory.tcl.

8.16.2.2 CommandHistory::canRedo ()

Similar to canUndo

Returns:
1 => redo possible, 0 => no redo possible

Definition at line 157 of file commandhistory.tcl.

8.16.2.3 CommandHistory::canUndo ()

Predicate to determine if calling undo will take effect

Returns:
1 => undo will take effect, 0 => not possible

Definition at line 142 of file commandhistory.tcl.

8.16.2.4 CommandHistory::redo ()

Calls current redo command and let cursor point to next command in the list (move to end) Definition at line 128 of file commandhistory.tcl.

8.16.2.5 CommandHistory::undo ()

Calls current undo command and modifies internal cursor to point to previous command in the list (move to beginning) Definition at line 113 of file commandhistory.tcl.

8.16.2.6 update CommandHistory::update () [protected]

Internal method used to update registered listeners whenever a change to the list of commands occurred Definition at line 88 of file commandhistory.tcl.

8.16.3 Member Data Documentation

8.16.3.1 TclRef CommandHistory::_commandList = "" [private]

list of commands used for redoing actions

Definition at line 30 of file commandhistory.tcl.

8.16.3.2 TclRef CommandHistory::_cursor = -1 [private]

current position in command lists

Definition at line 27 of file commandhistory.tcl.

8.16.3.3 TclRef CommandHistory::_inverseCommandList = "" [private]

list of commands used for undoing actions

Definition at line 33 of file commandhistory.tcl.

8.16.3.4 TclRef CommandHistory::_listenerList = "" [private]

list of object-IDs recognizing update-method

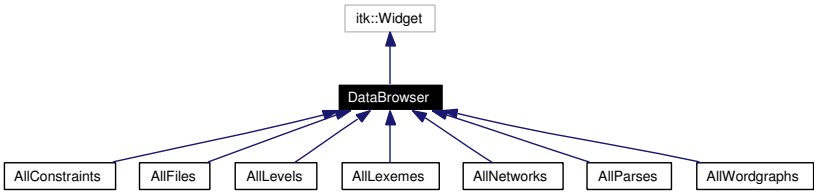
Definition at line 36 of file commandhistory.tcl.

The documentation for this class was generated from the following file:

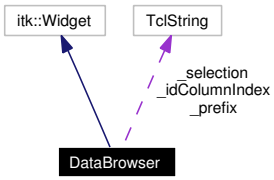
- commandhistory.tcl

8.17 DataBrowser Class Reference

Inheritance diagram for DataBrowser:



Collaboration diagram for DataBrowser:



8.17.1 Detailed Description

DataBrowser - Abstract base class for tabbed panes containing a top frame of buttons and a center table

Rules for subclassing:

- Override init_data in order to perform any additional initializations
- Call init_data whenever the frontend has to be set back to an initial state, e.g. after loading a new file
- Always call init_data at the end of derived constructor just before itk_initialize
- Add and pack additional components in derived constructor use component childsite to insert buttons into top panel
- All remaining methods possess default implementation, which may be overridden
- Use variable _selection to store table row selection ids
- Set variable _idColumnIndex to index number of column, that will contain selectable IDs (set at beginning of derived constructor!)
- Override getCData to get a pointer to the underlying C-Structure
- Override refreshrow to get data corresponding to the pointer retrieved by getCData and writing it into the table

Author:
Dietmar Dreyer (see also AUTHORS and THANKS for more)

Id

databrowser.tcl,v 1.13 2004/10/11 13:50:06 micha Exp

Definition at line 46 of file databrowser.tcl.

Public Member Functions

- getCData (TclString id)
- getSelection ()
- refreshid (TclString id)
- setIndexedSelection (TclList args)
- setSelection (TclList args)

Protected Member Functions

- _browse_action (TclWidget w)
- _keypress_action (TclWidget w, TclKeyBinding k)
- _motion_action (TclWidget w, TclNumber x, TclNumber y)
- _return_action ()
- _rowtag (TclNumber row)
- _setCount (TclNumber n)
- **init_data** ()
- refreshrow (TclNumber row, TclString item)

Protected Attributes

- TclString _idColumnIndex = ""
- TclString _selection = ""

Private Member Functions

- _evalSelectionPatterns ()
- _getMatchingRows (TclString regex, TclString varnameListIDs, TclString varnameMapIDs)
- DataBrowser (TclList args)

Private Attributes

- TclString **_prefix** = ""

8.17.2 Constructor & Destructor Documentation

8.17.2.1 DataBrowser::DataBrowser (TclList *args*) `[private]`

A DataBrowser constructor.

Parameters:

args arguments passed to itk_initialize

Definition at line 99 of file databrowser.tcl.

8.17.3 Member Function Documentation

8.17.3.1 DataBrowser::_browse_action (TclWidget *w*) [protected]

browse slot. This method adjusts the selected file in AllFiles::_selection.

Parameters:

w the widget bound to this slot.

Definition at line 189 of file databrowser.tcl.

8.17.3.2 DataBrowser::_evalSelectionPatterns () [private]

Evaluates input to the entry field as a list of regular expressions and matches them against all IDs in order to return a list of matching IDs Definition at line 258 of file databrowser.tcl.

8.17.3.3 DataBrowser::_getMatchingRows (TclString *regex*, TclString *varnameListIDs*, TclString *varnameMapIDs*) [private]

Returns a list of all row indices matching 'regex' in the ID column

Parameters:

regex regular expression string

varnameListIDs in-out-parameter, name of local variable to append matching IDs

varnameMapIDs in-out-parameter, name of local array-variable to insert matching IDs in order to prohibit duplicate selection entries in entry field

Definition at line 217 of file databrowser.tcl.

8.17.3.4 DataBrowser::_keypress_action (TclWidget *a*, TclKeyBinding *k*) [protected]

React to a keypress into the table.

Keys typed by the user are collected into a string, and the row whose id matches that string is selected. Definition at line 421 of file databrowser.tcl.

8.17.3.5 DataBrowser::_motion_action (TclWidget *w*, TclNumber *x*, TclNumber *y*) [protected]

Default motion slot.

Parameters:

w the widget where the motion was detected

x the x coords of the mouse

y the y coords of the mouse

Reimplemented in AllFiles.

Definition at line 390 of file databrowser.tcl.

References refreshid().

8.17.3.6 DataBrowser::_return_action () [protected]

actions to take place on pressing return in the entryfield.

Definition at line 244 of file databrowser.tcl.

8.17.3.7 DataBrowser::_rowtag (TclNumber *row*) [protected]

colorize the table rows This method is a callback configured to the table in order to colorize the rows. Definition at line 361 of file databrowser.tcl.

8.17.3.8 DataBrowser::_setCount (TclNumber *n*) [protected]

display the count-label. This number should reflect the number of items selected

Parameters:

n the number to be set

Definition at line 375 of file databrowser.tcl.

8.17.3.9 DataBrowser::getCData (TclString *id*)

abstract methods for retrieving the relevant C or Tcl data values for a given ID

Reimplemented in AllConstraints, AllLexemes, AllNetworks, AllParses, and AllWordgraphs.

8.17.3.10 DataBrowser::getSelection ()

return a list of selected ids

Definition at line 292 of file databrowser.tcl.

References setSelection().

8.17.3.11 DataBrowser::refreshid (TclString *id*)

refresh the displayed data for a specific ID

Definition at line 398 of file databrowser.tcl.

Referenced by _motion_action().

8.17.3.12 DataBrowser::refreshrow (TclNumber *row*, TclString *item*) [protected]

abstract method called in refreshid

Reimplemented in AllWordgraphs.

8.17.3.13 DataBrowser::setIndexedSelection (TclList *args*)

Select one or more rows. A previous selection is cleared; without arguemnts, removes all selections.

ARGS is a list of row indices. Definition at line 332 of file databrowser.tcl.

8.17.3.14 DataBrowser::setSelection (TclList *args*)

Select one or more rows. A previous selection is cleared; without arguemnts, removes all selections. ARGS must be a list of strings without spaces in them. Definition at line 304 of file databrowser.tcl. Referenced by getSelection().

8.17.4 Member Data Documentation

8.17.4.1 TclString DataBrowser::_idColumnIndex = "" [protected]

column index of table that contains selectable ids (used in _browse_action)
Definition at line 87 of file databrowser.tcl.

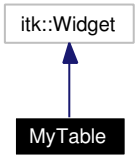
8.17.4.2 TclString DataBrowser::_selection = "" [protected]

string of current selected row ids
Definition at line 84 of file databrowser.tcl.
The documentation for this class was generated from the following file:

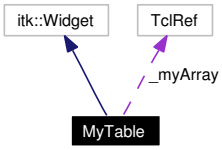
- databrowser.tcl

8.18 MyTable Class Reference

Inheritance diagram for MyTable:



Collaboration diagram for MyTable:



8.18.1 Detailed Description

MyTable - OO wrapper for tktable. Some time aggo Jeffrey Hobbs, the maintainer of tktable, discontinued the 00 wrapper for this marvelous widget. So here is my version of it. Its main purpose is to delegate calls from MyTable to the embedded table component.

There are some value addons that have been done ontop, that is

- the table is sorteable (MyTable::sortRows())
- clearing table cells is easier (MyTable::erase())
- provide an array to store the table content
- vertical and horizontal scrollbars
- table rows are collored
- an improved widget layout

Definition at line 29 of file mytable.tcl.

Public Member Functions

- activate (TclList args)
- childsite ()
- clear (TclList args)
- cols (TclNumber value=10)
- curselection (TclList args)
- delete (TclList args)
- erase (CellSpec fromCell, CellSpec toCell)
- getCell (TclList args)
- hscrollmode (TclRef value=static)

- hset (TclList args)
- icursor (TclList args)
- index (TclList args)
- insert (TclList args)
- MyTable (TclList args)
- **outerborderwidth** (TclNumber value=2)
- **outerrelief** (TclRef value=sunk)
- print ()
- rows (TclNumber value=20)
- **see** (TclList args)
- selbackground (TclRef value=gray50)
- selection (TclList args)
- selforeground (TclRef value=yellow)
- setCell (TclList args)
- sortRows (TclNumber index, TclList args)
- spans (TclList args)
- **state** (TclRef value=disabled)
- tag (TclList args)
- troughcolor (TclRef value=gray)
- vscrollmode (TclRef value=static)
- vset (TclList args)
- width (TclList args)
- xview (TclList args)
- yview (TclList args)

Private Member Functions

- colorize (TclNumber num)
- resize ()

Private Attributes

- TclRef _myArray

8.18.2 Constructor & Destructor Documentation

8.18.2.1 MyTable::MyTable (TclList *args*)

constructor

Definition at line 83 of file mytable.tcl.

8.18.3 Member Function Documentation

8.18.3.1 MyTable::activate (TclList *args*)

delegate the activate command to the table

Definition at line 343 of file mytable.tcl.

8.18.3.2 MyTable::childsite ()

retrieve the table component.

Returns:
the widget path to the tktable

Definition at line 268 of file mytable.tcl.

8.18.3.3 MyTable::clear (TclList *args*)

clear

Definition at line 260 of file mytable.tcl.

8.18.3.4 MyTable::colorize (TclNumber *num*) [private]

rowtag callback of the table. This method is called whenever the tktable component needs a tag for a table row. Definition at line 210 of file mytable.tcl.

8.18.3.5 MyTable::cols (TclNumber *value* = 10)

option -cols. configure the number of cols of the table Definition at line 486 of file mytable.tcl.

8.18.3.6 MyTable::curselection (TclList *args*)

delegate the curselection command to the table

Definition at line 364 of file mytable.tcl.

8.18.3.7 MyTable::delete (TclList *args*)

delegate the delete command to the table

Definition at line 329 of file mytable.tcl.

8.18.3.8 MyTable::erase (CellSpec *fromCell*, CellSpec *toCell*)

erase table cells. This method clears the content between two cell specifications by directly manipulating the _myArray. A cell specification has the format *"row,col"*, where *row* and *col* are integers.

Parameters:
fromCell the cell which is the first to be cleared
toCell the cell which is the last to be cleared

Definition at line 234 of file mytable.tcl.

8.18.3.9 MyTable::getCell (TclList *args*)

get a value of a table-cell

Definition at line 336 of file mytable.tcl.

8.18.3.10 MyTable::hscrollmode (TclRef *value* = `static`)

option -hscrollmode. Enable/disable display and mode of horizontal scrollbar. Definition at line 442 of file mytable.tcl.

References error().

8.18.3.11 MyTable::hset (TclList *args*)

delegate the set command to the horizontal scrollbar.

Definition at line 181 of file mytable.tcl.

8.18.3.12 MyTable::icursor (TclList *args*)

delegate the icursor command to the table

Definition at line 350 of file mytable.tcl.

8.18.3.13 MyTable::index (TclList *args*)

delegate the index command to the table

Definition at line 315 of file mytable.tcl.

8.18.3.14 MyTable::insert (TclList *args*)

delegate the insert command to the table

Definition at line 322 of file mytable.tcl.

8.18.3.15 MyTable::print ()

print the stored data. This simply calls parray on the _myArray variable. Definition at line 222 of file mytable.tcl.

8.18.3.16 MyTable::resize () `[private]`

propagate table resizing to the scrollbars.

Definition at line 188 of file mytable.tcl.

8.18.3.17 MyTable::rows (TclNumber *value* = `20`)

option -rows. configure the number of rows of the table Definition at line 477 of file mytable.tcl.

8.18.3.18 MyTable::selbackground (TclRef *value* = `gray50`)

option -selbackground. set the color of the selected row Definition at line 468 of file mytable.tcl.

8.18.3.19 MyTable::selection (TclList *args*)

delegate the selection command to the table.

Definition at line 289 of file mytable.tcl.

8.18.3.20 MyTable::selforeground (TclRef *value* = `yellow`)

option -selforeground. set the color of the selected row Definition at line 459 of file mytable.tcl.

8.18.3.21 MyTable::setCell (TclList *args*)

delegate the set command to the table

Definition at line 296 of file mytable.tcl.

8.18.3.22 MyTable::sortRows (TclNumber *index*, TclList *args*)

sort the table rows in the given column. This method sorts the table along the column given by the parameter *index*. The sorting itself is the done by a `lsort` command

Parameters:

index the column which along which the table is sorted

args the arguments for the `lsort` command

Definition at line 375 of file mytable.tcl.

8.18.3.23 MyTable::spans (TclList *args*)

delegate the spans command to the table

Definition at line 357 of file mytable.tcl.

8.18.3.24 MyTable::tag (TclList *args*)

delegate the tag command to the table

Definition at line 275 of file mytable.tcl.

8.18.3.25 MyTable::troughcolor (TclRef *value* = `gray`)

option -troughcolor. set the troughcolor of the scollbars Definition at line 416 of file mytable.tcl.

8.18.3.26 MyTable::vscrollmode (TclRef *value* = `static`)

option -vscrollmode. Enable/disable display and mode of vertical scrollbar. Definition at line 425 of file mytable.tcl.

References error().

8.18.3.27 MyTable::vset (TclList *args*)

delegate the set command to the vertical scrollbar.

Definition at line 174 of file mytable.tcl.

8.18.3.28 MyTable::width (TclList *args*)

delegate the width command to the table

Definition at line 308 of file mytable.tcl.

8.18.3.29 MyTable::xview (TclList *args*)

xview.

Definition at line 159 of file mytable.tcl.

8.18.3.30 MyTable::yview (TclList *args*)

yview

Definition at line 166 of file mytable.tcl.

8.18.4 Member Data Documentation

8.18.4.1 TclRef MyTable::_myArray [private]

storage for the table content.

Definition at line 77 of file mytable.tcl.

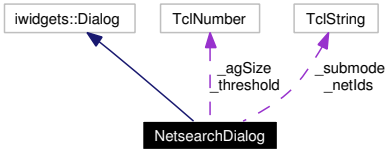
The documentation for this class was generated from the following file:

- mytable.tcl

8.19 NetsearchDialog Class Reference

Inheritance diagram for NetsearchDialog:

Collaboration diagram for NetsearchDialog:



8.19.1 Detailed Description

NetsearchDialog - gui for the netsearch CDG command.

Author:
Michael Daum (see also AUTHORS and THANKS for more)

Id
netsearch.tcl,v 1.6 2004/02/25 14:40:41 micha Exp

Definition at line 17 of file netsearch.tcl.

Public Member Functions

- activate (TclList args)
- NetsearchDialog (TclList args)

Private Member Functions

- _ok_action ()

Private Attributes

- TclNumber _agSize = 3000
- TclString _netIds = ""
- TclString _submode = "branchbound"
- TclNumber _threshold = 0.0

8.19.2 Constructor & Destructor Documentation

8.19.2.1 NetsearchDialog::NetsearchDialog (TclList *args*)

constructor

Definition at line 49 of file netsearch.tcl.

8.19.3 Member Function Documentation

8.19.3.1 NetsearchDialog::_ok_action () [private]

slot for the ok button action. This function is bound to the "ok" button of the gui, which then deactivates the dialog and calls ::cmd::netsearch(). Definition at line 129 of file netsearch.tcl.

8.19.3.2 NetsearchDialog::activate (TclList *args*)

activate the netsearch dialog.

Parameters:

args an optional list of network ids that should be preselected.

Definition at line 116 of file netsearch.tcl.

8.19.4 Member Data Documentation

8.19.4.1 TclNumber NetsearchDialog::_agSize = 3000 [private]

agenda size for netsearch. defaults to 3000

Definition at line 33 of file netsearch.tcl.

8.19.4.2 TclString NetsearchDialog::_netIds = "" [private]

list of all network ids that should be solved

Definition at line 30 of file netsearch.tcl.

8.19.4.3 TclString NetsearchDialog::_submode = "branchbound" [private]

submode for netsearch. All possible selections are alocated in the constructor being

- "branchbound" (default)
- "fullsearch"

Todo

insert all other submodes of netsearch

Definition at line 43 of file netsearch.tcl.

8.19.4.4 TclNumber NetsearchDialog::_threshold = 0.0 [private]

threshold for netsearch. defaults to 0.0

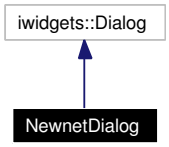
Definition at line 36 of file netsearch.tcl.

The documentation for this class was generated from the following file:

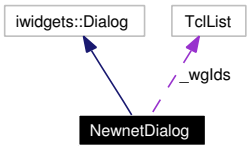
- netsearch.tcl

8.20 NewnetDialog Class Reference

Inheritance diagram for NewnetDialog:



Collaboration diagram for NewnetDialog:



8.20.1 Detailed Description

NewnetDialog - gui for the newnet CDG command.

Author:
Michael Daum (see also AUTHORS and THANKS for more)
Id
newnet.tcl,v 1.8 2004/02/25 14:40:42 micha Exp

Definition at line 17 of file newnet.tcl.

Public Member Functions

- activate (TclList args)
- NewnetDialog (TclList args)

Private Member Functions

- _newnet ()
- _ok_action ()

Private Attributes

- TclList _wgIds = {}

8.20.2 Constructor & Destructor Documentation

8.20.2.1 NewnetDialog::NewnetDialog (TclList *args*)

constructor

Definition at line 38 of file newnet.tcl.

8.20.3 Member Function Documentation

8.20.3.1 NewnetDialog::_newnet () [private]

called by _ok_action

Definition at line 94 of file newnet.tcl.

8.20.3.2 NewnetDialog::_ok_action () [private]

compute the nets

Definition at line 84 of file newnet.tcl.

8.20.3.3 NewnetDialog::activate (TclList *args*)

activate the dialog with a given set of wordgraph ids.

Definition at line 75 of file newnet.tcl.

8.20.4 Member Data Documentation

8.20.4.1 TclList NewnetDialog::_wgIds = {} [private]

list of wordgraphs for which we create a new constraintnet

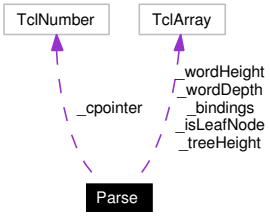
Definition at line 32 of file newnet.tcl.

The documentation for this class was generated from the following file:

- newnet.tcl

8.21 Parse Class Reference

Collaboration diagram for Parse:



8.21.1 Detailed Description

Parse - OO wrapper for the parse C structure.

This is an object oriented wrapper arround the C structure with the same name. All access to that C structure should be encapsulated by this class. All methods dealing directly with a C ParseStruct are called via this class.

Todo

There are rather some exceptions ;) (but why).

Author:

Michael Daum, Kilian A. Foth, Dietmar Fünning (see also AUTHORS and THANKS for more)

$Id parse.tcl$

Definition at line 25 of file parse.tcl.

Public Member Functions

- getBadness ()
- getBindingAt (TclString levelId, TclNumber wordIndex)
- getBindingById (TclString bindingId)
- getBindings (TclString levelId)
- getComment ()
- getCurrentReading (TclNumber wordIndex)
- getDate ()
- getGrammarFiles ()
- getHeight (TclString levelId)
- getId ()
- getLabels ()
- getLatticeId ()
- getLevels ()
- getLexemes ()
- getLexiconItem (TclNumber wordIndex)
- getModifiee (TclString levelId, TclNumber wordIndex)
- getModifiers (TclString levelId, TclNumber wordIndex)
- getNoSolutions ()

- getScore ()
- getSearchStrategy ()
- getSolutionNo ()
- getUserName ()
- getValue (TclNumber wordIndex)
- getViolations ()
- getWidth ()
- getWord (TclNumber wordIndex)
- getWordDepth (TclString levelId, TclNumber wordInedx)
- getWordHeight (TclString levelId, TclNumber wordInedx)
- getWords ()
- indexOf (TclString levelId, TclNumber wordIndex)
- init (Parse cpointer)
- isAbstract ()
- isLeafNode (TclString levelId, TclNumber wordIndex)
- mirror ()
- optimizeLabel (TclString levelId, TclNumber wordIndex)
- optimizeStructure (TclNumber levelId, TclNumber wordIndex)
- optimizeWord (TclNumber wordIndex)
- Parse (TclList args)
- register ()
- shiftEdge (TclString levelId, TclNumber wordIndex, TclNumber targetIndex)
- swapLabel (TclString levelId, TclNumber wordIndex, TclString label)
- swapWord (TclNumber wordIndex, TclString description)
- toAnno ()
- verify ()

Private Attributes

- TclArray _bindings
- TclNumber _cpointer = 0
- TclArray _isLeafNode
- TclArray _treeHeight
- TclArray _wordDepth
- TclArray _wordHeight

8.21.2 Constructor & Destructor Documentation

8.21.2.1 Parse::Parse (TclList *args*)

constructor

Definition at line 122 of file parse.tcl.

8.21.3 Member Function Documentation

8.21.3.1 Parse::getBadness ()

get the badness value of the current parse. This method computes the badness of the current parse by itself looking up its constraint violations. The badness is given as a list containing

- the number of hard violations
- the number of soft violations
- the remaining score not taking the hard violations into account.

Returns:
a badness list as described above.

Definition at line 461 of file parse.tcl.

8.21.3.2 Parse::getBindingAt (TclString *levelId*, TclNumber *wordIndex*)

get the binding at a position on a level. This is the lowest access function to the ParseStruct. All information retrieved from the ParseStruct is cached in Parse::_bindings for faster access. This cache only gets invalidated by swapWord(), swapLabel() or shiftEdge(). Thereby caching should be *mostly* transparent.

Note:
The cache gets out of sync if the C layer changes the Parse structure without using this Tcl layer.

Parameters:
levelId the id of the level whose binding we are interested in.

wordIndex the index of the word of the parse.

Returns:
a TclList of the binding information

See also:
Parse::_bindings for an explanation of the list format

Definition at line 177 of file parse.tcl.

8.21.3.3 Parse::getBindingById (TclString *id*)

get the bindings of a given index. This comes done to the binding that is pointed to by a constraint violation. Its `nodeBindingIndex1` and `nodeBindingIndex2` correspond to the binding index, that is the third element in the bindings list (see Parse::_bindings).

Todo
This method is inefficient as we search thru all bindings for the given id. Alas extra hash shall soothe thee, as we damn the ancillary storage hassle.

Parameters:
id the node binding index in the parse (see indexOf())

Returns:
the binding list or an empty list if a binding of index `id` isn't there.

See also:
Parse::_bindings for an explanation of the bindings list format.

Definition at line 199 of file parse.tcl.

8.21.3.4 Parse::getBindings (TclString *levelId*)

get all bindings on a given level. This method retrieves all bindings of the unerlying Parse.

See also:
getBindingsAt

Parameters:
levelId the id of the level whose bindings we are interested in.

Returns:
a TclList of all bindings (that is a list of lists)

See also:
Parse::_bindings for an explanation of the list format

Definition at line 150 of file parse.tcl.

8.21.3.5 Parse::getComment ()

get the comment of the current parse. This is a getter method for the ParseStruct::comment.

Returns:
a TclString

Definition at line 394 of file parse.tcl.

8.21.3.6 Parse::getCurrentReading (TclNumber *wordIndex*)

Return the description of the word currently selected at the timepoint $wordIndex. Definition at line 576 of file parse.tcl.

8.21.3.7 Parse::getDate ()

get the date of the current parse. This is a wrapper for parseDate() in the C layer.

Returns:
the date of creation of the current parse.

Definition at line 375 of file parse.tcl.

8.21.3.8 Parse::getGrammarFiles ()

get all grammar files. This gets all grammar files related to this parse, that is return the tcl-ized list of ParseStruct::grammarFiles.

Returns:
a TclList of grammar file names.

Definition at line 423 of file parse.tcl.

8.21.3.9 Parse::getHeight (TclString *levelId*)

get the height of the dependency tree on a given level. This method simply returns the parseHeight() of the current parse on the given level. Doing that it caches the result on the tcl layer for faster access. If the parse changes in structure (by shiftEdge()) this array gets invalidated again.

Parameters:
levelId the id of the level we are interested in.

Returns:
the tree height

Definition at line 491 of file parse.tcl.

8.21.3.10 Parse::getId ()

get the id of the current parse. This is a getter method for the ParseStruct::id

Returns:
a TclString

Definition at line 365 of file parse.tcl.

8.21.3.11 Parse::getLabels ()

get all labels. This method collects all labels from ParseStruct::labels for you. Actually this should be only a fallback whenever we cannot ask the grammar for its labels.

Returns:
$_cpointer->labels as a Tcl list

Definition at line 319 of file parse.tcl.

8.21.3.12 Parse::getLatticeId ()

get the lattice id. This method returns the `latticeId` this parse is annotating, that is return ParseStruct::latticeId. Definition at line 440 of file parse.tcl.

8.21.3.13 Parse::getLevels ()

Get all levels of the Parse. This method collects the ids of all levels contained in the current parse. So this is an easy accessor for ParseStruct::levels.

Returns:
a list of `levelIds`

Definition at line 287 of file parse.tcl.

8.21.3.14 Parse::getLexemes ()

get all lexeme nodes of current parse. This method is only a wrapper for parseGetLexemNodes() converting the returned List into a TclList (I know this is lame)

Returns:
a list of LexemNode items.

Definition at line 339 of file parse.tcl.

8.21.3.15 Parse::getLexiconItem (TclNumber *wordIndex*)

Return the lexicon item currently selected at the timepoint $wordIndex.

Definition at line 591 of file parse.tcl.

8.21.3.16 Parse::getModifiee (TclString *levelId*, TclNumber *wordIndex*)

get the modifiee of a word on a level. This method lets you access the ParseStruct::verticesStruct easily.

Parameters:
levelId the id of the level whose binding we are interested in.
wordIndex the index of the word of the parse.

Returns:
the modifiee index of the given word.

Definition at line 242 of file parse.tcl.

8.21.3.17 Parse::getModifiers (TclString *levelId*, TclNumber *wordIndex*)

get all modifiers of a word on a level. This method does not boil down to getBindingAt() as you would expect but sacrifies clean design for speed accessing the ParseStruct::verticesStruct directly retrieving what we want.

Parameters:
levelId the id of the level whose binding we are interested in.
wordIndex the index of the word of the parse.

Returns:
a list of all modifiers of the given word.

Definition at line 219 of file parse.tcl.

8.21.3.18 Parse::getNoSolutions ()

get the number of solutions. This is the number of solutions this parse is one of.

See also:
getSolutionNo()

Definition at line 403 of file parse.tcl.

8.21.3.19 Parse::getScore ()

get the score of the current parse. This is a getter method for the ParseStruct::score.

Returns:
a float.

Definition at line 355 of file parse.tcl.

8.21.3.20 Parse::getSearchStrategy ()

get the search strategy. This method returns the information about the search strategy with which the current parse has been produced

Returns:
a TclString

Definition at line 385 of file parse.tcl.

8.21.3.21 Parse::getSolutionNo ()

get the solution number. This is the solution number identifying this parse as being one of several solutions.

See also:
getNoSolutions()

Definition at line 413 of file parse.tcl.

8.21.3.22 Parse::getUserName ()

get the user name. This method returns the user who created this parse. Definition at line 448 of file parse.tcl.

8.21.3.23 Parse::getValue (TclNumber *i*)

Return a string representing the lexicon item currently selected at the timepoint $wordIndex. Definition at line 584 of file parse.tcl.

8.21.3.24 Parse::getViolations ()

get all violations of the current parse. Information is taken from the Parse::violations and then taken appart meticulously for you. We return a list of violations each with the following format in the following order:

- the `name` of the violation
- the `first` node binding index (aka. the binding id - see getBindingById())
- the `second` node binding index
- the `score` of the violation

Returns:
the violations as described above, that is a list of lists.

Definition at line 259 of file parse.tcl.

8.21.3.25 Parse::getWidth ()

get the number of words. Actually being a bad name for what it does this simply counts the number of words in the ParseStruct::words vector. Definition at line 548 of file parse.tcl.

8.21.3.26 Parse::getWord (TclNumber *wordIndex*)

get a word from the parse. This method gets you the WordStruct at the given index

Parameters:
wordIndex the word we want

Returns:
a WordStruct

Definition at line 568 of file parse.tcl.

8.21.3.27 Parse::getWordDepth (TclString *levelId*, TclNumber *wordIndex*)

get maximum distance of reachable leaves. This method works using the same caching logic like getWordHeight() and all the other methods that retrieve numbers from the C layer caching them on the tcl layer.

Parameters:
levelId the level we are interested in

wordIndex the word we are interested in

Returns:
the word depth

Definition at line 528 of file parse.tcl.

8.21.3.28 Parse::getWordHeight (TclString *levelId*, TclNumber *wordIndex*)

Wrapper for parseWordHeight(). This method simply returns the parseWordHeight() of a word on a level in the current parse. Afterwards results are cached at the tcl layer. Occasionally shiftEdge() nullifies this again.

Parameters:

levelId the level we are interested in

wordIndex the word we are interested in

Returns:

the word height

Definition at line 509 of file parse.tcl.

8.21.3.29 Parse::getWords ()

get the words Vector of the current parse. This method does not attempt to extract all words from the ParseStruct::words vector but returns the Vector itself.

Returns:

a Vector of Word items

Definition at line 558 of file parse.tcl.

8.21.3.30 Parse::indexOf (TclString *levelId*, TclNumber *wordIndex*)

Wrapper for parseIndex().

Definition at line 692 of file parse.tcl.

8.21.3.31 Parse::init (Parse *cpointer*)

attach a C structure to this instance.

Parameters:

cpointer a pointer to a ParseStruct Every Parse can only be initialized once.

Definition at line 134 of file parse.tcl.

References error().

8.21.3.32 Parse::isAbstract ()

get the abstraction state of the current parse. This is a straight delegation to parseIsAbstract().

Returns:

a bool.

Definition at line 701 of file parse.tcl.

8.21.3.33 Parse::isLeafNode (TclString *levelId*, TclNumber *wordIndex*)

test whether a word on a given level is a leaf. This is done by examining the ParseStruct::verticesStruct storing the result into Parse::_isLeafNode at the index `levelId` x `wordIndex`. This caches information of the C layer in the tcl layer. Invalidating this cache is done when needed, i.e. in shiftEdge().

Parameters:

levelId the level we are interested in

wordIndex the word index

Returns:

0 or 1

Definition at line 715 of file parse.tcl.

8.21.3.34 Parse::mirror ()

compute the mirror edges. This is a straight delegation to parseMirror(). Note that all caches on the tcl layer are possibly invalidated now. Definition at line 748 of file parse.tcl.

8.21.3.35 Parse::optimizeLabel (TclString *levelId*, TclNumber *wordIndex*)

Wrapper for parseOptimizeLabel(). Definition at line 650 of file parse.tcl.

8.21.3.36 Parse::optimizeStructure (TclNumber *levelId*, TclNumber *wordIndex*)

Wrapper for parseOptimizeStructure(). Definition at line 664 of file parse.tcl.

8.21.3.37 Parse::optimizeWord (TclNumber *wordIndex*)

Wrapper for parseOptimizeWord(). Definition at line 657 of file parse.tcl.

8.21.3.38 Parse::register ()

register the current parse to th C layer. This puts the C Parse into inputCurrentGrammar->parses. Actually this operation is blindly delegated to parseRegister(). Definition at line 739 of file parse.tcl.

8.21.3.39 Parse::shiftEdge (TclString *levelId*, TclNumber *wordIndex*, TclNumber *targetIndex*)

redirect a dependency edge to a new modifiee. This method uses parseShiftEdge() to do the actual work. Note that the arrays Parse::_wordHeight, Parse::_wordDepth, Parse::_bindings and Parse::_treeHeight are invalidated for this level.

Parameters:

levelId the level we want to maniplulate

wordIndex the word index of the dependency edge

targetIndex the index of the new modifiee

Definition at line 678 of file parse.tcl.

8.21.3.40 Parse::swapLabel (TclString *levelId*, TclNumber *wordIndex*, TclString *label*)

exchange the label of a dependency edge. This method uses parseSwapLabel() to do the actual work. Note that the Parse::_bindings array is invalidated for this level.

Parameters:

levelId the level we want to maniplulate

wordIndex the word index of the dependency edge

label the new label for this dependency edge

Definition at line 642 of file parse.tcl.

8.21.3.41 Parse::swapWord (TclNumber *wordIndex*, TclString *description*)

exchange a word at a given index. This method uses parseSwapWord() to maniplulate the parse structure. Note that the Parse::_bindings array is invalidated completely here. Definition at line 629 of file parse.tcl.

8.21.3.42 Parse::toAnno ()

convert the current parse to an annotation. This is a straight delegation to parse2annotation(). Definition at line 761 of file parse.tcl.

8.21.3.43 Parse::verify ()

verify the current parse. This is method verifies the current parse using the first annotation linked to the same lattice, that is: this parse has a link to a lattice and there exist one or more annotations linked to this lattice; we take the first of them.

Returns:

a ParseVerificationStruct or the empty string on an error

Definition at line 603 of file parse.tcl.

8.21.4 Member Data Documentation

8.21.4.1 TclArray Parse::_bindings [private]

cache to speed up tree access. This hash stores all bindings for each level mapping `levelId` to the list of all bindings on that level. A binding is a list consisting of the following elements

- the `wordIndex`
- the `modifiee` index taken from the ParseStruct::verticesStructure
- the `label` taken from the ParseStruct::verticesLabels
- the `index` computed for this parse (see indexOf())
- the `levelId` to which this encoded binding belongs to

See also:

getBindings(), swapLabel(), swapWord(), shiftEdge()

Definition at line 68 of file parse.tcl.

8.21.4.2 TclNumber Parse::_cpointer = 0 [private]

pointer to the C structure

Definition at line 31 of file parse.tcl.

8.21.4.3 TclArray Parse::_isLeafNode [private]

information about each word on each level. This has mapps the `levelId` x `wordIndex` to a boolean flag indicating whether the adressed word is being modified or not. Definition at line 56 of file parse.tcl.

8.21.4.4 TclArray Parse::_treeHeight [private]

the height of each tree on each level. This hash mapps the `levelId` to the height. Note that this simply caches the values computed by parseHeight().

See also:
getHeight(), shiftEdge()

Definition at line 51 of file parse.tcl.

8.21.4.5 TclArray Parse::_wordDepth [private]

the depth of a word in a dependency tree on a level. This hash mapps `levelId` x `wordIndex` to the word depth. Note that this simply caches the values computed by parseWordDepth() and stores the mat the tcl layer for faster access.

See also:
getWordDepth(), shiftEdge()

Definition at line 45 of file parse.tcl.

8.21.4.6 TclArray Parse::_wordHeight [private]

the height of a word in a dependency tree on a level. This hash mapps `levelId` x `wordIndex` to the word height. Note that this simply caches the values computed by parseWordHeight() and stores them at the tcl layer for faster access.

See also:
getWordHeight(), shiftEdge()

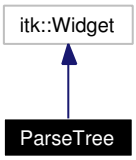
Definition at line 38 of file parse.tcl.

The documentation for this class was generated from the following file:

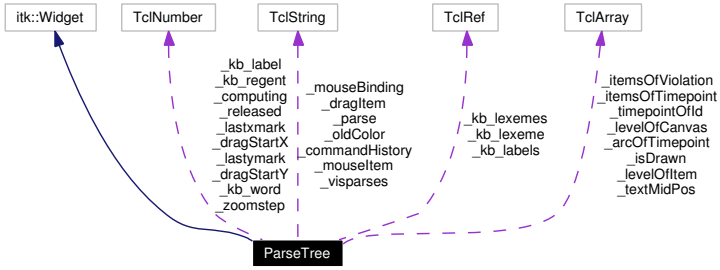
- parse.tcl

8.22 ParseTree Class Reference

Inheritance diagram for ParseTree:



Collaboration diagram for ParseTree:



8.22.1 Detailed Description

ParseTree - visualization of one parse.

ParseTree objects are the central means to give you a nice dependency tree drawing. They are only thought to be used as components of a VisParses object being the GUI surrounding a set of visualized ParseTree objects. Taking the embedded Parse object into account, these tree classes implement the tree editor in XCDG.

Author:
Michael Daum, Kilian A. Foth, Dietmar Fünning (see also AUTHORS and THANKS for more)

Id
parsetree.tcl,v 1.135 2004/10/18 08:09:32 foth Exp

Definition at line 26 of file parsetree.tcl.

Public Member Functions

- addUndoListener (TclString listener)
- backwardLevel ()
- breakcycles ()
- canRedo ()
- canUndo ()
- drawAll ()
- edgecolor (TclString value="black")
- errorcolor (TclString value="red")

- forwardLevel ()
- getOrientation ()
- highlight (TclString which)
- highlightcolor (TclString value="red")
- labelcolor (TclString value="black")
- labelfont (TclString value="")
- levelborderwidth (TclNumber value=1)
- levelrelief (TclString value="flat")
- mainlevelname ()
- mirror ()
- nodecolor (TclString value="black")
- parse (TclList args)
- ParseTree (Parse p, VisParses vp, TclList args)
- redo ()
- setOrientation (TclString orient)
- showcycles ()
- undo ()
- verify ()
- vlinecolor (TclString value="gray")
- wordcolor (TclString value="black")
- wordfont (TclString value="")
- writeToFile (TclString fileName)
- zoom (TclNumber direction)
- ~ParseTree ()

Private Member Functions

- _zoom (TclNumber step)
- allEnter (Canvas w)
- allLeave (Canvas w)
- allSybillings (TclNumber id1, TclNumber id2, TclString level)
- arcRightClick (Canvas w)
- canvasClick (Canvas w, TclNumber x, TclNumber y)
- canvasToScreen (Canvas canvas, TclNumber x, TclNumber y)
- centerOnItem (Canvas w, TclNumber i)
- closestNode (Canvas w, TclNumber x, TclNumber y)
- compareLexeme (LexiconItem a, LexiconItem b)
- createArc (Canvas canvas, TclNumber x1, TclNumber y1, TclNumber x2, TclNumber y2, TclNumber text, TclNumber bindNo)
- drawArcs ()
- drawLevel (TclString level)
- drawText (Widget canvas, TclNumber x, TclNumber y)
- drawViolas ()
- edgeClick (Canvas w, TclNumber x, TclNumber y)
- edgeDrag (Canvas w, TclNumber x, TclNumber y)
- edgeDragPan (Canvas w, TclNumber x, TclNumber y)
- edgeDrop (Canvas w, TclNumber x, TclNumber y)
- edgeEnter (Canvas w)
- edgeLeave (Canvas w)
- edgeMoveCancel (Canvas w)

- edgeOfTimepoint (Canvas w, TclNumber t)
- edgeRightClick (Canvas w)
- fixCoords (Canvas w, TclNumber x, TclNumber y)
- getAlternativeLexemes (Canvas w, TclNumber i)
- itemVisible (Canvas w, TclNumber i)
- kb_change_edge (Canvas w, TclWidget k, TclKeyBinding a)
- kb_change_label (Canvas w, TclWidget k, TclKeyBinding a)
- kb_change_lexeme (Canvas w, TclWidget k, TclKeyBinding a)
- kb_select_edge (Canvas w, TclWidget k, TclKeyBinding a)
- kb_select_label (Canvas w, TclWidget k, TclKeyBinding a)
- kb_select_lexeme (Canvas w, TclWidget k, TclKeyBinding a)
- kb_select_word (Canvas w, TclWidget k, TclKeyBinding a)
- labelClick (Canvas w)
- labelOfEdge (Canvas w, TclString edge)
- labelRightClick (Canvas w)
- labelSelect (TclString l, TclNumber i, TclString label)
- makeUserInterface (Canvas w)
- mouseScrollDown (Canvas w)
- mouseScrollUp (Canvas w)
- moveEdge (Canvas canvas, TclNumber x1, TclNumber y1, TclNumber x2, TclNumber y2)
- moveLabel (Canvas canvas, TclString label_id, TclNumber x1, TclNumber y1, TclNumber x2, TclNumber y2)
- multiCanvas (TclList args)
- multiDragto (TclNumber x, TclNumber y)
- multiMark (TclNumber x, TclNumber y)
- noCrossing (TclString id, TclList bindings)
- nodeOfTimepoint (Canvas w, TclNumber e, TclNumber t)
- optimizeWord (TclNumber t)
- paintTabs ()
- pop_constraint (TclNumber x, TclNumber y, TclNumber X, TclNumber Y)
- reDrawLevel (TclString level)
- registerItems (TclString level)
- rgValue (TclNumber score)
- showPopMenu (Canvas w, TclNumber x, TclNumber y, TclList items)
- undoEdge (TclNumber modifier, TclNumber modifiee, TclString level)
- undoLabel (TclNumber modifier, TclString label, TclString level)
- undoLexeme (TclNumber from, TclString description)
- violaBrowse (TclString mode, TclNumber xcoord, TclNumber ycoord)
- wordClick (Canvas w)
- wordEnter (Canvas w)
- wordLeave (Canvas w)
- wordMiddleClick (Canvas w)
- wordOfTimepoint (Canvas w, TclNumber t)
- wordRightClick (Canvas w)
- wordSelect (Canvas w, TclNumber from, LexemNode lexeme)

Private Attributes

- TclArray _arcOfTimepoint
- TclString _commandHistory = ""
- TclNumber _computing = 0
- TclString **_dragItem** = ""
- TclNumber **_dragStartX** = 0
- TclNumber **_dragStartY** = 0
- TclArray _isDrawn
- TclArray _itemsOfTimepoint
- TclArray _itemsOfViolation
- TclNumber **_kb_label** = -1
- TclRef **_kb_labels**
- TclRef **_kb_lexeme**
- TclRef **_kb_lexemes**
- TclNumber **_kb_regent** = -2
- TclNumber _kb_word = 0
- TclNumber **_lastxmark** = 0
- TclNumber **_lastymark** = 0
- TclArray _levelOfCanvas
- TclArray _levelOfItem
- TclString **_mouseBinding** = ""
- TclString **_mouseItem** = ""
- TclString **_oldColor** = ""
- TclString _parse = ""
- TclNumber **_released** = 0
- TclArray _textMidPos
- TclArray _timepointOfId
- TclString _visparses = ""
- TclNumber **_zoomstep** = 0

8.22.2 Constructor & Destructor Documentation

8.22.2.1 ParseTree::ParseTree (Parse *p*, VisParses *vp*, TclList *args*)

constructor

instantiating command-history-object Definition at line 232 of file parsetree.tcl.

8.22.2.2 ParseTree::~ParseTree ()

destructor

Definition at line 362 of file parsetree.tcl.

References makeUserInterface().

8.22.3 Member Function Documentation

8.22.3.1 ParseTree::_zoom (TclNumber *step*) [private]

set zoom level for one canvas.

Definition at line 2532 of file parsetree.tcl.

8.22.3.2 ParseTree::addUndoListener (TclString *listener*)

Add a listener to be informed of changes to the command-history

Parameters:
listener Object-ID implementing method update {}

Definition at line 2679 of file parsetree.tcl.

8.22.3.3 ParseTree::allEnter (Canvas *w*) [private]

command bound to Enter-event

Definition at line 538 of file parsetree.tcl.

8.22.3.4 ParseTree::allLeave (Canvas *w*) [private]

command bound to Leave-event

Definition at line 552 of file parsetree.tcl.

8.22.3.5 ParseTree::allSybillings (TclNumber *id1*, TclNumber *id2*, TclString *level*) [private]

checks if all nodes between id1 and id2 are modifier of id2.

Definition at line 1521 of file parsetree.tcl.

8.22.3.6 ParseTree::arcRightClick (Canvas *w*) [private]

Auto-correct a non-mainlevel subordination.

Definition at line 996 of file parsetree.tcl.

8.22.3.7 ParseTree::backwardLevel ()

Switch to the previous level of description.

Definition at line 3220 of file parsetree.tcl.

8.22.3.8 ParseTree::breakcycles ()

Break cycles in the tree on the current level.

This may be necessary because a cycle can't be broken by drag & drop if there isn't already a root binding.
Definition at line 2459 of file parsetree.tcl.

8.22.3.9 ParseTree::canRedo ()

Predicate-method rooted to command history

Returns:
1 => undo can be executed, 0 => undo can not be executed

Definition at line 2736 of file parsetree.tcl.

References itemVisible().

Referenced by canUndo().

8.22.3.10 ParseTree::canUndo ()

Predicate-method rooted to command history

Returns:
1 => undo can be executed, 0 => undo can not be executed

Definition at line 2728 of file parsetree.tcl.

References canRedo().

8.22.3.11 ParseTree::canvasClick (Canvas *w*, TclNumber *x*, TclNumber *y*) [private]

command bound to B1-event on the canvas background.

Definition at line 1051 of file parsetree.tcl.

8.22.3.12 ParseTree::canvasToScreen (Canvas *canvas*, TclNumber *x*, TclNumber *y*) [private]

compute the screen x and y coords of canvas-coords x and y.

Definition at line 2027 of file parsetree.tcl.

8.22.3.13 ParseTree::centerOnItem (Canvas *w*, TclNumber *i*) [private]

Scroll $w horizontally so that $i is visible.

Definition at line 2766 of file parsetree.tcl.

8.22.3.14 ParseTree::closestNode (Canvas *w*, TclNumber *x*, TclNumber *y*) [private]

Find closest circle on the canvas.

One should think that this should be possible with a normal ‘canvas find’ command, but it isn’t; you can only search for the closest item, OR for all circles. Definition at line 1106 of file parsetree.tcl.

8.22.3.15 ParseTree::compareLexeme (LexiconItem *a*, LexiconItem *b*) [private]

compare two pointers to LexemeNode by their description.

Definition at line 773 of file parsetree.tcl.

8.22.3.16 ParseTree::createArc (Canvas *canvas*, TclNumber *x1*, TclNumber *y1*, TclNumber *x2*, TclNumber *y2*, TclNumber *text*, TclNumber *bindNo*) [private]

creates a new arc with the specified parameters.

Definition at line 2244 of file parsetree.tcl.

8.22.3.17 ParseTree::drawAll ()

draw everything into the levels.

Definition at line 1880 of file parsetree.tcl.

8.22.3.18 ParseTree::drawArcs () [private]

draws all bindings not on the main level as arcs under the main tree.

Definition at line 2128 of file parsetree.tcl.

8.22.3.19 ParseTree::drawLevel (TclString *level*) [private]

draw a specified level on its page.

Definition at line 1567 of file parsetree.tcl.

8.22.3.20 ParseTree::drawText (Widget *canvas*, TclNumber *x*, TclNumber *y*) [private]

draw the sentence to the canvas.

Definition at line 1333 of file parsetree.tcl.

8.22.3.21 ParseTree::drawViolas () [private]

draw violations in the list.

Definition at line 1254 of file parsetree.tcl.

8.22.3.22 ParseTree::edgeClick (Canvas *w*, TclNumber *x*, TclNumber *y*) [private]

command bound to B1-event on an edge.

Definition at line 1083 of file parsetree.tcl.

8.22.3.23 ParseTree::edgecolor (TclString *value* = "black")

itk_option: color of edges up to the parents

8.22.3.24 ParseTree::edgeDrag (Canvas *w*, TclNumber *x*, TclNumber *y*) [private]

Command bound to B1-Motion-event on a edge.

Moving edges around on the canvas is implemented by setting the private variable _dragItem to the edge that is clicked on, moving this edge when the mouse is dragged, and resetting the variable when the mouse is released.

Ordinarily one would use Tcl's automatically managed canvas tag 'current' for this; but an edge may be selected not by clicking on it, but by clicking on the canvas background in its vicinity, and Tcl is not smart enough to assign the 'current' tag to an item that was moved under the mouse pointer after the click occurred. It also doesn't allow setting the 'current' tag manually, so we have to duplicate its functionality. Definition at line 1147 of file parsetree.tcl.

8.22.3.25 ParseTree::edgeDragPan (Canvas *w*, TclNumber *x*, TclNumber *y*) [private]

Handler for the unusual event that the user wants to pan while holding an edge; this is actually useful if an edge must be moved between two attachment points that do not fit on the canvas simultaneously. Definition at line 1180 of file parsetree.tcl.

8.22.3.26 ParseTree::edgeDrop (Canvas *w*, TclNumber *x*, TclNumber *y*) [private]

command bound to B1-Release-event on a edge.

if bindings changed, save commands to history Definition at line 1206 of file parsetree.tcl.

8.22.3.27 ParseTree::edgeEnter (Canvas *w*) [private]

Executed when an edge is touched.

This is different from allEnter because touching an edge should also highlight its label. Definition at line 645 of file parsetree.tcl.

8.22.3.28 ParseTree::edgeLeave (Canvas *w*) [private]

Executed when an edge is left by the mouse pointer.

Definition at line 660 of file parsetree.tcl.

8.22.3.29 ParseTree::edgeMoveCancel (Canvas *w*) [private]

Cancel an ongoing move operation, i.e. return the edge to where it was before it was picked up. Completed moves are not undone, for that you need undo/redo. Definition at line 1191 of file parsetree.tcl.

8.22.3.30 ParseTree::edgeOfTimepoint (Canvas *w*, TclNumber *t*) [private]

Return the edge widget at timepoint $t on canvas $w.

Definition at line 591 of file parsetree.tcl.

8.22.3.31 ParseTree::edgeRightClick (Canvas *w*) [private]

Auto-correct the current subordination.

Definition at line 912 of file parsetree.tcl.

8.22.3.32 ParseTree::errorcolor (TclString *value* = "red")

itk_option: color of erroneous structures

8.22.3.33 ParseTree::fixCoords (Canvas *w*, TclNumber *x*, TclNumber *y*) [private]

Translate window-based into canvas-based co-ordinates.

If a scrolledcanvas is scrolled to the right, away from the home position, and a mouse click occurs in its upper left corner, then the bind event will still receive the coordinates (1,1) through its x and y parameters, even though the canvas pixel the the mouse is touching is something like (320,1). (Actually, because of the way the scrolledcanvas widget is written, an offset occurs even in the home position.) This routine accounts for that offset. Therefore it must be called whenever coordinates received through 'bind' are to be applied to things on the canvas. Definition at line 2053 of file parsetree.tcl.

8.22.3.34 ParseTree::forwardLevel ()

Switch to the next level of description.

Definition at line 3228 of file parsetree.tcl.

8.22.3.35 ParseTree::getAlternativeLexemes (Canvas *w*, TclNumber *item*) [private]

Find lexemes that can be exchanged for $item.

The return value is a list of lists that each contain a descriptive string and the fitting wordSelect callback. Definition at line 786 of file parsetree.tcl.

8.22.3.36 ParseTree::getOrientation ()

get the orientation of the panedwindow.

Definition at line 2586 of file parsetree.tcl.

8.22.3.37 ParseTree::highlight (TclString *which*)

Highlight specified edges in color.

Definition at line 2609 of file parsetree.tcl.

8.22.3.38 ParseTree::highlightcolor (TclString *value* = "red")

itk_option: color of highlighted structures

8.22.3.39 ParseTree::itemVisible (Canvas *w*, TclNumber *i*) [private]

Check if $i is actually visible on $w.

Returns 0 only if $i is actually scrolled offscreen. $i might also be invisible because the entire X window is partially obscured, but Tcl cannot detect this. Definition at line 2747 of file parsetree.tcl.

Referenced by canRedo().

8.22.3.40 ParseTree::kb_change_edge (Canvas *w*, TclWidget *a*, TclKeyBinding *k*) [private]

Handle Shift-ed keyboard editing events: actually change the subordination Definition at line 2962 of file parsetree.tcl.

8.22.3.41 ParseTree::kb_change_label (Canvas *w*, TclWidget *a*, TclKeyBinding *k*) [private]

Handle Control-ed keyboard editing events: actually change the label Definition at line 3076 of file parsetree.tcl.

8.22.3.42 ParseTree::kb_change_lexeme (Canvas *w*, TclWidget *a*, TclKeyBinding *k*) [private]

Handle Alt keyboard editing events: actually change the lexeme Definition at line 3198 of file parsetree.tcl.

8.22.3.43 ParseTree::kb_select_edge (Canvas *w*, TclWidget *k*, TclKeyBinding *a*) [private]

Handle Shift-ed keyboard editing events.

Definition at line 2833 of file parsetree.tcl.

8.22.3.44 ParseTree::kb_select_label (Canvas *w*, TclWidget *a*, TclKeyBinding *k*) [private]

Handle Control-ed keyboard events: select the hypothetical new label Definition at line 3003 of file parsetree.tcl.

8.22.3.45 ParseTree::kb_select_lexeme (Canvas *w*, TclWidget *a*, TclKeyBinding *k*) [private]

Handle Alt keyboard events: select the hypothetical new lexeme Definition at line 3110 of file parsetree.tcl.

8.22.3.46 ParseTree::kb_select_word (Canvas *w*, TclWidget *a*, TclKeyBinding *k*) [private]

Handle keyboard editing events: select the word that the next change will apply to. Definition at line 2788 of file parsetree.tcl.

8.22.3.47 ParseTree::labelClick (Canvas *w*) [private]

show menu of available level-labels when clicking on a label.

Definition at line 865 of file parsetree.tcl.

8.22.3.48 ParseTree::labelcolor (TclString *value* = "black")

itk_option: color of the dependency labels

8.22.3.49 ParseTree::labelfont (TclString *value* = "")

itk_option: font-property of the dependency labels

Definition at line 1998 of file parsetree.tcl.

8.22.3.50 ParseTree::labelOfEdge (Canvas *w*, TclString *edge*) [private]

Return the label that goes with $edge.

Definition at line 563 of file parsetree.tcl.

8.22.3.51 ParseTree::labelRightClick (Canvas *w*) [private]

Auto-correct the current label.

Definition at line 895 of file parsetree.tcl.

8.22.3.52 ParseTree::labelSelect (TclString *l*, TclNumber *i*, TclString *label*) [private]

Set the label of the edge at $timepoint on $level.

Definition at line 1022 of file parsetree.tcl.

8.22.3.53 ParseTree::levelborderwidth (TclNumber *value* = 1)

itk_option: decoration of a level component

8.22.3.54 ParseTree::levelrelief (TclString *value* = "flat")

itk_option: decoration of a level component

8.22.3.55 ParseTree::mainlevelname ()

Return he name of the main level.

The main level is the one that gets all other edges drawn under it as little arcs. If the grammar does not define one, return the name of the first level in the parse. Definition at line 2597 of file parsetree.tcl.

8.22.3.56 ParseTree::makeUserInterface (Canvas *w*) [private]

part of the initialization process.

Definition at line 371 of file parsetree.tcl.

Referenced by ~ParseTree().

8.22.3.57 ParseTree::mirror ()

apply parseMirror.

Definition at line 2333 of file parsetree.tcl.

8.22.3.58 ParseTree::mouseScrollDown (Canvas *w*) [private]

Scroll canvas down a step.

Definition at line 3243 of file parsetree.tcl.

Referenced by mouseScrollUp().

8.22.3.59 ParseTree::mouseScrollUp (Canvas *w*) [private]

Scroll canvas up a step.

Definition at line 3236 of file parsetree.tcl.

References mouseScrollDown().

8.22.3.60 ParseTree::moveEdge (Canvas *canvas*, TclNumber *x1*, TclNumber *y1*, TclNumber *x2*, TclNumber *y2*) [private]

move an existing edge and its label to a new position.

Definition at line 1511 of file parsetree.tcl.

8.22.3.61 ParseTree::moveLabel (Canvas *canvas*, TclString *label_id*, TclNumber *x1*, TclNumber *y1*, TclNumber *x2*, TclNumber *y2*) [private]

move an existing edge and its label to a new position.

Definition at line 1363 of file parsetree.tcl.

8.22.3.62 ParseTree::multiCanvas (TclList *args*) [private]

call all level canvases of the tree.

Definition at line 2320 of file parsetree.tcl.

8.22.3.63 ParseTree::multiDragto (TclNumber *x*, TclNumber *y*) [private]

scan all levels of the tree horizontally.

Parameters:

- ***x*** the x position of the pointer
- ***y*** the x position of the pointer

Definition at line 2285 of file parsetree.tcl.

8.22.3.64 ParseTree::multiMark (TclNumber *x*, TclNumber *y*) [private]

scan all levels of the tree horizontally.

Parameters:

- ***x*** the x position of the pointer
- ***y*** the x position of the pointer

Definition at line 2272 of file parsetree.tcl.

8.22.3.65 ParseTree::noCrossing (TclString *id*, TclList *bindings*) [private]

seeks an edge crossing with this id.

Definition at line 1542 of file parsetree.tcl.

8.22.3.66 ParseTree::nodecolor (TclString *value* = "black")

itk_option: color of a node of the dependency tree

8.22.3.67 ParseTree::nodeOfTimepoint (Canvas *w*, TclNumber *e*, TclNumber *t*) [private]

Return the node widget at timepoint $t on canvas $w.

Note that for t == -1, there can be several candidates on the canvas, and we select the one that is closest to the end of edge $e to avoid hectic scrolling. Definition at line 609 of file parsetree.tcl.

8.22.3.68 ParseTree::optimizeWord (TclNumber *t*) [private]

Wrapper for Parse::optimizeWord.

Definition at line 967 of file parsetree.tcl.

8.22.3.69 ParseTree::paintTabs () [private]

colorize the tabs in the tabnotebook. This is done according to the worst violation on that tab. Definition at line 1895 of file parsetree.tcl.

8.22.3.70 ParseTree::parse (TclList *args*)

delegate to the _parse.

Definition at line 2570 of file parsetree.tcl.

References setOrientation().

8.22.3.71 ParseTree::pop_constraint (TclNumber *x*, TclNumber *y*, TclNumber *X*, TclNumber *Y*) [private]

Command bound to conflict middle click.

Definition at line 678 of file parsetree.tcl.

8.22.3.72 ParseTree::redo ()

Calls redo on command history.

Definition at line 2713 of file parsetree.tcl.

8.22.3.73 ParseTree::reDrawLevel (TclString *level*) [private]

force draw a specified level on its page.

Definition at line 1559 of file parsetree.tcl.

8.22.3.74 ParseTree::registerItems (TclString *level*) [private]

Compute the array _itemsOfViolation for this level.

Definition at line 1791 of file parsetree.tcl.

8.22.3.75 ParseTree::rgValue (TclNumber *score*) [private]

take a score and return an appropriate red/green color for it.

Definition at line 1950 of file parsetree.tcl.

8.22.3.76 ParseTree::setOrientation (TclString *orient*)

set the orientation of the panedwindow and reset to default fraction.

Definition at line 2577 of file parsetree.tcl.

Referenced by parse().

8.22.3.77 ParseTree::showcycles ()

highlight all edges participating in cycles on the current level.

Definition at line 2401 of file parsetree.tcl.

8.22.3.78 ParseTree::showPopMenu (Canvas *w*, TclNumber *x*, TclNumber *y*, TclList *items*) [private]

build a popup menu and show the provided items. Items is a list of pairs consisting of a label and the associated command. Definition at line 2068 of file parsetree.tcl.

8.22.3.79 ParseTree::undo ()

Methods used to implement undo-functionality (called from VisParses).

Definition at line 2689 of file parsetree.tcl.

8.22.3.80 ParseTree::undoEdge (TclNumber *modifier*, TclNumber *modifiee*, TclString *level*) [private]

Methods used to implement undo-functionality (Internal methods).

Shift edge (modifier) to modifiee on given level Used to undo edge-moving actions Definition at line 2667 of file parsetree.tcl.

8.22.3.81 ParseTree::undoLabel (TclNumber *modifier*, TclString *label*, TclString *level*) [private]

Change modifier's label at given level.

Definition at line 2655 of file parsetree.tcl.

8.22.3.82 ParseTree::undoLexeme (TclNumber *from*, TclString *description*) [private]

Change reading for given vertice to description

Parameters:

from vertice to change

description new reading

Definition at line 2645 of file parsetree.tcl.

8.22.3.83 ParseTree::verify ()

highlight differences to annotation

Definition at line 2348 of file parsetree.tcl.

8.22.3.84 ParseTree::violaBrowse (TclString *mode*, TclNumber *xcoord*, TclNumber *ycoord*) [private]

command bound to violation browsing.

Definition at line 699 of file parsetree.tcl.

8.22.3.85 ParseTree::vlinecolor (TclString *value* = "gray")

itk_option: color of vertical lines

8.22.3.86 ParseTree::wordClick (Canvas *w*) [private]

show menu of available lexicon-entries when clicking on a word.

Definition at line 819 of file parsetree.tcl.

8.22.3.87 ParseTree::wordcolor (TclString *value* = "black")

itk_option: color of the words in the sentence

8.22.3.88 ParseTree::wordEnter (Canvas *w*) [private]

command bound to Enter-event.

Definition at line 509 of file parsetree.tcl.

8.22.3.89 ParseTree::wordfont (TclString *value* = "")

itk_option: font-property for the words in the sentence

Definition at line 2012 of file parsetree.tcl.

8.22.3.90 ParseTree::wordLeave (Canvas *w*) [private]

command bound to Leave-event.

Definition at line 530 of file parsetree.tcl.

8.22.3.91 ParseTree::wordMiddleClick (Canvas *w*) [private]

command bound to word middle click.

Definition at line 463 of file parsetree.tcl.

8.22.3.92 ParseTree::wordOfTimepoint (Canvas *w*, TclNumber *t*) [private]

Return the word widget at timepoint $t on canvas $w.

Definition at line 578 of file parsetree.tcl.

8.22.3.93 ParseTree::wordRightClick (Canvas *w*) [private]

Auto-correct the current word.

Definition at line 940 of file parsetree.tcl.

8.22.3.94 ParseTree::wordSelect (Canvas *w*, TclNumber *from*, LexemNode *lexeme*) [private]

action taking place when a lexeme is chosen from a wordlist.

Saving action to history Definition at line 836 of file parsetree.tcl.

8.22.3.95 ParseTree::writeToFile (TclString *fileName*)

saves the Parse to file as an annotation in cdgp input format

Definition at line 1989 of file parsetree.tcl.

8.22.3.96 ParseTree::zoom (TclNumber *direction*)

increase or decrease zoom level of all canvases.

Definition at line 2513 of file parsetree.tcl.

8.22.4 Member Data Documentation

8.22.4.1 TclArray ParseTree::_arcOfTimepoint [private]

array storing arcs by level and timepoint. Example: _arcOfTimepoint(3,OBL1) = 47 48 Definition at line 85 of file parsetree.tcl.

8.22.4.2 TclString ParseTree::_commandHistory = "" [private]

Command-History-Object for undo-implementation.

Definition at line 113 of file parsetree.tcl.

8.22.4.3 TclNumber ParseTree::_computing = 0 [private]

Flags for inter-handler communication.

Definition at line 124 of file parsetree.tcl.

8.22.4.4 TclArray ParseTree::_isDrawn [private]

array indicating whether a level is already drawn.

Definition at line 99 of file parsetree.tcl.

8.22.4.5 TclArray ParseTree::_itemsOfTimepoint [private]

holds all items on a canvas belonging to one timepoint. Example: _itemsOfTimepoint(SYN,3) == 4 27 26 24 25 Definition at line 77 of file parsetree.tcl.

8.22.4.6 TclArray ParseTree::_itemsOfViolation [private]

holds ids of canvas items to be highlighted when touching a violation. Example: _itemsOfViolation(1) == 23 SYN 22 SYN Definition at line 93 of file parsetree.tcl.

8.22.4.7 TclNumber ParseTree::_kb_word = 0 [private]

Variables holding state of the keyboard highlighting.

Definition at line 116 of file parsetree.tcl.

8.22.4.8 TclArray ParseTree::_levelOfCanvas [private]

array storing the canvas of a level.

Definition at line 96 of file parsetree.tcl.

8.22.4.9 TclArray ParseTree::_levelOfItem [private]

array holding the level that an arc belongs to. Example: _levelOfItem(89) = DOM Definition at line 89 of file parsetree.tcl.

8.22.4.10 TclString ParseTree::_parse = "" [private]

the embedded Parse object

Definition at line 66 of file parsetree.tcl.

8.22.4.11 TclArray ParseTree::_textMidPos [private]

array holding the middle points of the words. initialized by drawText; indexed by timepoints, e.g. 0 Definition at line 73 of file parsetree.tcl.

8.22.4.12 TclArray ParseTree::_timepointOfId [private]

array holding the timepoint which a specified id belongs to. Example: _timepointOfId(SYN,3) = 0 Definition at line 81 of file parsetree.tcl.

8.22.4.13 TclString ParseTree::_visparses = "" [private]

who is my daddy

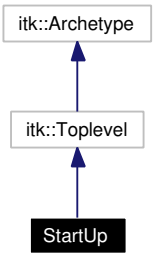
Definition at line 69 of file parsetree.tcl.

The documentation for this class was generated from the following file:

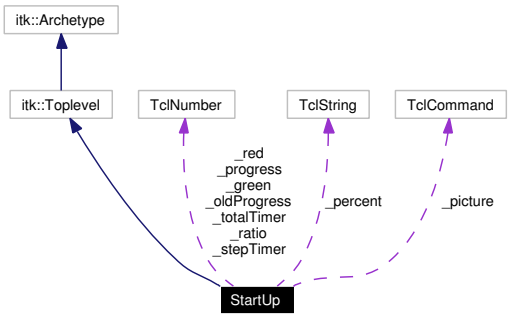
- parsetree.tcl

8.23 StartUp Class Reference

Inheritance diagram for StartUp:



Collaboration diagram for StartUp:



8.23.1 Detailed Description

StartUp - startup-screen

Author:
Dietmar Fünning, Michael Daum

Id
startup.tcl,v 1.7 2004/10/11 15:23:31 micha Exp

Definition at line 17 of file startup.tcl.

Public Member Functions

- height (TclNumber value=798)
- StartUp (TclList args)
- step (TclNumber n)
- width (TclNumber value=599)
- ~StartUp ()

Private Member Functions

- _drawScreen ()

Private Attributes

- TclNumber **_green** = 0
- TclNumber **_oldProgress** = 0
- TclString **_percent** = "0%"
- TclCommand **_picture**
- TclNumber **_progress** = 0
- TclNumber **_ratio** = 0
- TclNumber **_red** = 4095
- TclNumber **_stepTimer** = 0
- TclNumber **_totalTimer** = 0

8.23.2 Constructor & Destructor Documentation

8.23.2.1 StartUp::~StartUp ()

destructor

Definition at line 47 of file startup.tcl.

8.23.2.2 StartUp::StartUp (TclList *args*)

constructor

Definition at line 59 of file startup.tcl.

8.23.3 Member Function Documentation

8.23.3.1 StartUp::_drawScreen () `[private]`

well, draw the opening screen

Definition at line 111 of file startup.tcl.

8.23.3.2 StartUp::height (TclNumber *value* = `798`)

option -height. set the overall height of the application Definition at line 94 of file startup.tcl.

8.23.3.3 StartUp::step (TclNumber *n*)

update and display progress

Definition at line 140 of file startup.tcl.

8.23.3.4 StartUp::width (TclNumber *value* = `599`)

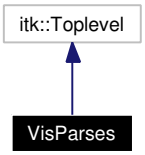
option -width. set the overall width of the application Definition at line 103 of file startup.tcl.

The documentation for this class was generated from the following file:

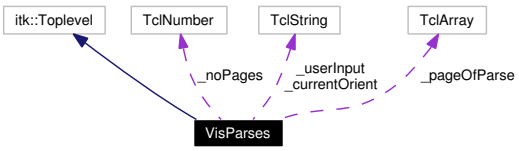
- startup.tcl

8.24 VisParses Class Reference

Inheritance diagram for VisParses:



Collaboration diagram for VisParses:



8.24.1 Detailed Description

VisParses - the grafical tree editor.

A VisParses object instanciate one toplevel window that embedds a set of ParseTree objects to be displayed. There are two common situations that determine the set of parses: (1) displaying annotations, (2) displaying parser solutions. In the first case one VisParses object can fetch all annotations loaded into the system. In the second case one VisParses object displays all solutions a specific parsing flavour has found for one wordgraph. So in the latter case another parser cannot reuse the same VisParses object to visualize its solution record also, but needs a new VisParses object of its own, giving you two separate toplevel windows. I think this is the best compromise here between lots of toplevel windows and embedded objects in separate tabs of one window.

As you see below we don't provide a separate destructor to destruct embedded ParseTree objects but add ParseTree objects directly as itk-components to the VisParses widget. So they get destructed automatically.

Author:
Michael Daum, Kilian A. Foth, Dietmar Fünning (see also AUTHORS and THANKS for more)

$Id visparses.tcl $

Definition at line 35 of file visparses.tcl.

Public Member Functions

- addParse (ParseTree parse)
- center ()
- geometry ()
- getBadness ()
- highlight (ParseTree parse, TclString which)
- iconname (TclString value="xcdg")
- **interactive** (TclNumber value=0)
- readWord ()

- removeParse (ParseTree parse)
- update ()
- view (TclNumber no)
- VisParses (TclList args)

Private Member Functions

- _BackspaceHandler ()
- _breakcycleAction ()
- _computeViolas ()
- _deleteParseAction ()
- _getOrientation ()
- _labelOfIndex (TclNumber index)
- _loadAction ()
- _mirrorAction ()
- _nextAction ()
- _previousAction ()
- _print ()
- _redo ()
- _refresh ()
- _saveAction ()
- _saveAsAction ()
- _setOrientation ()
- _showcycleAction ()
- _spaceHandler ()
- _switchLevel (TclNumber x)
- _undo ()
- _updateVisual ()
- _verifyAction ()
- _zoomIn ()
- _zoomOut ()

Private Attributes

- TclString **_currentOrient** = "vertical"
- TclNumber **_noPages** = 0
- TclArray _pageOfParse
- TclString **_userInput** = ""

8.24.2 Constructor & Destructor Documentation

8.24.2.1 VisParses::VisParses (TclList *args*)

constructor

Configure Edit-Menu with Undo-/Redo-Entries Definition at line 91 of file visparses.tcl.

8.24.3 Member Function Documentation

8.24.3.1 VisParses::_BackspaceHandler () [private]

invoked by Backspace in the textbox.

Definition at line 858 of file visparses.tcl.

8.24.3.2 VisParses::_breakcycleAction () [private]

Break cycles in a parse.

Definition at line 664 of file visparses.tcl.

8.24.3.3 VisParses::_computeViolas () [private]

compute the violations of the selected ParseTree.

Definition at line 713 of file visparses.tcl.

8.24.3.4 VisParses::_deleteParseAction () [private]

undisplay the selected ParseTree.

Definition at line 792 of file visparses.tcl.

8.24.3.5 VisParses::_getOrientation () [private]

get the orientation of the split of the current parsetree.

Definition at line 941 of file visparses.tcl.

8.24.3.6 VisParses::_labelOfIndex (TclNumber *index*) [private]

returns the label of a page.

Definition at line 816 of file visparses.tcl.

8.24.3.7 VisParses::_loadAction () [private]

load a parse directly into the editor.

Todo

this is not implelented yet here, but in AllParses

Definition at line 560 of file visparses.tcl.

8.24.3.8 VisParses::_mirrorAction () [private]

call parseMirror() eventually.

Definition at line 611 of file visparses.tcl.

8.24.3.9 VisParses::_nextAction () [private]

switch to the next tab in the tree editor. Switching to a next tab means that we select the next tree in a set of trees loaded into the editor. In case of solution parses we simply switch to the next parse in the editor. In case of annotation parses we switch to the next annotation or generate new annotation parses for the next lattice in the set of annotated lattices. Definition at line 681 of file visparses.tcl.

8.24.3.10 VisParses::_previousAction () [private]

display the previous annotation.

Definition at line 706 of file visparses.tcl.

8.24.3.11 VisParses::_print () [private]

_print the visible level.

Definition at line 530 of file visparses.tcl.

8.24.3.12 VisParses::_redo () [private]

undo the last action of the selected page.

Definition at line 735 of file visparses.tcl.

8.24.3.13 VisParses::_refresh () [private]

_refresh the display of the selected VisParses.

Definition at line 746 of file visparses.tcl.

8.24.3.14 VisParses::_saveAction () [private]

writes the tree to file as an annotation.

Todo

The name of the file to write to is determined by parsing the `-title` property. Actually this should be done better by paying a round of new properties.

Definition at line 570 of file visparses.tcl.

8.24.3.15 VisParses::_saveAsAction () [private]

writes the tree to file under a new name.

Definition at line 587 of file visparses.tcl.

8.24.3.16 VisParses::_setOrientation () [private]

set the orientation of the split of the current parsetree.

Definition at line 929 of file visparses.tcl.

8.24.3.17 VisParses::_showcycleAction () [private]

Display cycles on a level in red ink.

Definition at line 652 of file visparses.tcl.

8.24.3.18 VisParses::_spaceHandler () [private]

invoked by space in the textbox.

Definition at line 828 of file visparses.tcl.

8.24.3.19 VisParses::_switchLevel (TclNumber *x*) [private]

Switch to next or previous level in the parsetree.

Definition at line 635 of file visparses.tcl.

8.24.3.20 VisParses::_undo () [private]

undo the last action of the selected page.

Definition at line 724 of file visparses.tcl.

8.24.3.21 VisParses::_updateVisual () [private]

update the visual parts which depends on the selected parsetree

Definition at line 973 of file visparses.tcl.

8.24.3.22 VisParses::_verifyAction () [private]

visual version of the 'verify' command

Definition at line 623 of file visparses.tcl.

8.24.3.23 VisParses::_zoomIn () [private]

Display all canvases of the selected parse smaller.

Definition at line 905 of file visparses.tcl.

8.24.3.24 VisParses::_zoomOut () [private]

display all canvases of the selected parse bigger.

Definition at line 917 of file visparses.tcl.

8.24.3.25 VisParses::addParse (ParseTree *parse*)

add a parse on a separate page.

Definition at line 470 of file visparses.tcl.

8.24.3.26 VisParses::center ()

center on the middle of the application.

Definition at line 762 of file visparses.tcl.

8.24.3.27 VisParses::geometry ()

Confess to my geometry.

Definition at line 463 of file visparses.tcl.

8.24.3.28 VisParses::getBadness ()

get the score of current parsetree

Definition at line 954 of file visparses.tcl.

8.24.3.29 VisParses::highlight (ParseTree *parse*, TclString *which*)

Delegate a highlight action to the parsetree.

Definition at line 989 of file visparses.tcl.

8.24.3.30 VisParses::iconname (TclString *value* = "xcdg")

option -iconname.

Definition at line 755 of file visparses.tcl.

8.24.3.31 VisParses::readWord ()

gets another word of typed input. Hangs until the word is supplied. Definition at line 775 of file visparses.tcl.

8.24.3.32 VisParses::removeParse (ParseTree *parse*)

make a ParseTree invisible. This does not destroy the underlying Parse. Definition at line 503 of file visparses.tcl.

8.24.3.33 VisParses::update ()

updates the editmenu. This method is registered to the CommandHistory to be called whenever the undo history changes. Depending on the current position in the undo history, things are redoable or undoable.

Todo

get a better name for this method: "update" is most generic and might be confusing as we have a VisParsees::_updateVisual() aswell.

Definition at line 1001 of file visparses.tcl.

8.24.3.34 VisParses::view (TclNumber *no*)

delegate the view command to the `tbn` component

Definition at line 523 of file visparses.tcl.

8.24.4 Member Data Documentation

8.24.4.1 TclArray VisParses::_pageOfParse [private]

array mapping Parses to page-numbers

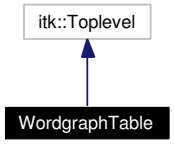
Definition at line 85 of file visparses.tcl.

The documentation for this class was generated from the following file:

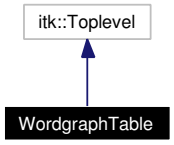
- visparses.tcl

8.25 WordgraphTable Class Reference

Inheritance diagram for WordgraphTable:



Collaboration diagram for WordgraphTable:



8.25.1 Detailed Description

WordgraphTable - display a wordgaph in a table.

Todo

this class is so seldomly used and of such a limited usage that we might consider to remove it completely.

Id

wordgraph.tcl,v 1.23 2004/02/25 14:40:42 micha Exp

Definition at line 19 of file wordgraph.tcl.

Public Member Functions

- hide ()
- init_data (TclString id)
- show (TclString id="")
- WordgraphTable (TclString id, TclList args)

8.25.2 Constructor & Destructor Documentation

8.25.2.1 WordgraphTable::WordgraphTable (TclString *id*, TclList *args*)

constructor

Definition at line 34 of file wordgraph.tcl.

8.25.3 Member Function Documentation

8.25.3.1 WordgraphTable::hide ()

withdraw the toplevel

Definition at line 165 of file wordgraph.tcl.

References hide().

Referenced by hide().

8.25.3.2 WordgraphTable::init_data (TclString *id*)

get the data from the C layer.

Parameters:

id a lattice id to be loaded into the table.

Definition at line 105 of file wordgraph.tcl.

References init_data().

Referenced by init_data().

8.25.3.3 WordgraphTable::show (TclString *id* = "")

show the toplevel.

Parameters:

id optional lattice id to initialize the table.

Definition at line 155 of file wordgraph.tcl.

References show().

Referenced by show().

The documentation for this class was generated from the following file:

- wordgraph.tcl

Chapter 9

XCDG File Documentation

9.1 bugfix.tcl File Reference

9.1.1 Detailed Description

TclTk Enhancements. Fixes for some common errors in the TclTk, Itcl/Itk/Iwidges toolkits. Til now the most importand fix here is the iwidgets::Shell:activate{} fix to get arround the display delay of a shell dialog before it gets accessable. You might have noticed that before in other TclTk apps, that sometimes you get a white box dialog. Whatever this is the activate() method below overwrites the Shell method in a way that things work better now. I don't clame to understand what I actually do here ;)

Author:
Michael Daum

Id
bugfix.tcl,v 1.6 2004/02/25 14:40:41 micha Exp

Definition in file bugfix.tcl.

9.2 compat.tcl File Reference

9.2.1 Detailed Description

Compatibility Stuff. This file is serves as a workarround in differences of old and new swig interfaces, that is differences of the interfaces generated with swig version 1.1.x and 1.3.x. This file might get obsolete when we will discontinue to support the old swig in the near future.

Here's the list of general differences that I came accross so far:

- global variables: old swig generated an access function like `hkVerbosity_get` and `hkVerbosity_set` that access the global variable. In new swigish we get a variable `$hkVerbosity` and read and write access is handled by internal getter and setter functions. When trying to write to a immutable variable swig intercepts that with a proper error message. So for backwards compatibility we reintroduce the explicit access function to get the old tcl sources running

Author:
Michael Daum

Id
compat.tcl,v 1.5 2004/09/06 13:40:54 micha Exp

Definition in file compat.tcl.

9.3 error.tcl File Reference

9.3.1 Detailed Description

Overwrite the tcl error suite to deactivated the CdgBusy dialog in case of an error.

Author:
Michael Daum (see also AUTHORS and THANKS for more)

Id
error.tcl,v 1.7 2004/02/25 14:40:41 micha Exp

Definition in file error.tcl.

Functions

- bgerror (TclList args)
- catch (TclCommand script, TclString varName="")
- error (TclString message, TclString info="", TclCommand code="")

Variables

- TclNumber _catchFlag = 0
- TclNumber _errorFlag = 0

9.3.2 Function Documentation

9.3.2.1 bgerror (TclList *args*)

overwrite the system bgerror command. This function simply flags the error in the global variable `_errorFlag` = 1 when it is encountered, deactivates a CdgBusy dialog and the delegates the rest of the call to the origninal bgerror having been renamed to `_bgerror`. Definition at line 43 of file error.tcl.

References bgerror().

Referenced by bgerror().

9.3.2.2 catch (TclCommand *script*, TclString *varName* = "")

overwrite the system cache command. This function basically sets the global variable `catchFlag` = 1 before it then calls the system cache command (which was renamed to `catch`). After having done that it sets `catchFlag` = 0 again. So we track the part of call that is being catched, the effect of which is that we want to deactivate the CdgBusy dialog only once in error(). Definition at line 58 of file error.tcl.

9.3.2.3 error (TclString *message*, TclString *info* = "", TclCommand *code* = "")

overwrite the system error command. This flag safely deactivates the CdgBusy dialog, i.e. when the `_catchFlag` is not set. If so it also flags the error having taken place in `_errorFlag` = 1. Definition at line 78 of file error.tcl.

References error().

Referenced by CdgMenu::_setFlag(), error(), MyTable::hscrollmode(), Parse::init(), and MyTable::vscrollmode().

9.3.3 Variable Documentation

9.3.3.1 TclNumber _catchFlag = 0

global flags used in catch(), error() and bgerror()

Definition at line 26 of file error.tcl.

9.3.3.2 TclNumber _errorFlag = 0

global flags used in catch(), error() and bgerror()

Definition at line 23 of file error.tcl.

Referenced by CdgBusy::compute(), CdgMenu::init_data(), and CdgBusy::interrupt().

9.4 itcl-hierarchy.tcl File Reference

9.4.1 Detailed Description

A small part of the itk/iwidges class hierarchy. This has the nice effect of a more integrated class hierarchy graph in doxygen.

Author:
Michael Daum

Id
itcl-hierarchy.tcl,v 1.4 2004/02/25 14:06:46 micha Exp

Definition in file itcl-hierarchy.tcl.

Classes

- class **itk::Toplevel**
- class **itk::Widget**
- class **iwidgets::Dialog**
- class **iwidgets::Dialogshell**
- class **iwidgets::Labeledwidget**
- class **iwidgets::Messagedialog**
- class **iwidgets::Pushbutton**
- class **iwidgets::Scrolledcanvas**
- class **iwidgets::Scrolledtext**
- class **iwidgets::Scrolledwidget**
- class **iwidgets::Shell**

9.5 textutils.tcl File Reference

9.5.1 Detailed Description

This file contains some arbitrary usefull textutilities, mainly used in the CdgShell.

Author:
Michael Daum

Id
textutils.tcl,v 1.8 2004/09/06 13:40:54 micha Exp

Definition in file textutils.tcl.

Distances between line segments

The following code to compute distances between line segments and points stolen shamelessly from Rory Daulton (rorydaulton@email.com), who posted it to delphi-talk@elists.org.

- dot (TclNumber x0, TclNumber y0, TclNumber x1, TclNumber y1)
- par_area (TclNumber x1, TclNumber y1, TclNumber x2, TclNumber y2)
- point_segment_distance (TclNumber x, TclNumber y, TclNumber ex0, TclNumber ey0, TclNumber ex1, TclNumber ey1)
- segment_length (TclNumber x0, TclNumber y0, TclNumber x1, TclNumber y1)
- segments_to_vectors (TclNumber ex0, TclNumber ey0, TclNumber ex1, TclNumber ey1, TclNumber x, TclNumber y)

Functions

- forAllMatches (TextWidget w, Index from, Index to, TclString pattern, TclCommand script)
- lfilter (TclList list, TclString pattern)
- smartCompare (TclString a, TclString b)

9.5.2 Function Documentation

9.5.2.1 dot (TclNumber *x0*, TclNumber *y0*, TclNumber *x1*, TclNumber *y1*)

Find the dot product of two vectors.

Definition at line 124 of file textutils.tcl.

References dot().

Referenced by dot().

9.5.2.2 forAllMatches (TextWidget *w*, Index *from*, Index *to*, TclString *pattern*, TclCommand *script*)

execute a script on every match in a text widget.

Parameters:
w a text widget

from the starting position from which to scan the text widget

to the ending position from which to scan the text widget

pattern the regular expression pattern to be matched

script a pice of code executed in the callers context

Definition at line 29 of file textutils.tcl.

References forAllMatches().

Referenced by forAllMatches().

9.5.2.3 lfilter (TclList *list*, TclString *pattern*)

filter out items from a TclList.

Parameters:

list a TclList

pattern a regular expression pattern to be matched on each list item

Returns:

a new TclList only consisting of the items matching the `pattern`

Definition at line 47 of file textutils.tcl.

References lfilter().

Referenced by lfilter().

9.5.2.4 par_area (TclNumber *x1*, TclNumber *y1*, TclNumber *x2*, TclNumber *y2*)

Calculate the signed area of the parallelogram that is formed by two vectors (by the origin and two given points). Definition at line 139 of file textutils.tcl.

References par_area().

Referenced by par_area().

9.5.2.5 point_segment_distance (TclNumber *x*, TclNumber *y*, TclNumber *ex0*, TclNumber *ey0*, TclNumber *ex1*, TclNumber *ey1*)

Give distance of a point from a line segment.

Definition at line 146 of file textutils.tcl.

References point_segment_distance().

Referenced by point_segment_distance().

9.5.2.6 segment_length (TclNumber *x0*, TclNumber *y0*, TclNumber *x1*, TclNumber *y1*)

Find the length of a line segment.

Definition at line 131 of file textutils.tcl.

References segment_length().

Referenced by segment_length().

9.5.2.7 segments_to_vectors (TclNumber *ex0*, TclNumber *ey0*, TclNumber *ex1*, TclNumber *ey1*, TclNumber *x*, TclNumber *y*)

Convert two directed line segments, starting from the same point, to vectors Definition at line 110 of file textutils.tcl.

References segments_to_vectors().

Referenced by segments_to_vectors().

9.5.2.8 smartCompare (TclString *a*, TclString *b*)

compare strings intelligently like ls -v does

Definition at line 61 of file textutils.tcl.

References smartCompare().

Referenced by smartCompare().

Chapter 10

XCDG Page Documentation

10.1 Todo List

Member AllFiles::load_dir(TclList args) this method does not use the busy dialog.

Class AllHierarchies Variable and function naming is a complete mess here. Til now I just fixed the declarations to match the implementations. Furtheron all underscore/capitalized namings should be unified. No interface variable should only be named `id`. When it is a hierarchy's id call it for example `hierId`.

the TclArray called `hier` is cononfusing me with the TclArray `hierarchy`. I know they contain different stuff but again: the naming is extraordinary bad.

the TclArray `hier` is overloaded: its information should be split into several arrays named by the third argument of the hier array.

the methods C_to_Tcllist() and ldelete() are not related to AllHierarchies.

private varialbes should be named starting with and underscore

Member AllHierarchies::ldelete(TclList list, TclString type_id) this method is in no way specific to AllHierarchies

Member AllHierarchies::position_leaves2(TclString id, TclString type_id) bad method name

Member AllHierarchies::position_of_type(Hierarchy hierarchy, TclString type_id) the documentation does not match the intension of this function

Member NetsearchDialog::_submode insert all other submodes of netsearch

Class Parse There are rather some exceptions ;) (but why).

Member Parse::getBindingById(TclString bindingId) This method is inefficient as we search thru all bindings for the given id. Alas extra hash shall soothe thee, as we damn the ancillary storage hassle.

Member VisParses::_loadAction() this is not implelented yet here, but in AllParses

Member VisParses::_saveAction() The name of the file to write to is determined by parsing the `-title` property. Actually this should be done better by paying a round of new properties.

Member VisParses::update() get a better name for this method: "update" is most generic and might be confusing as we have a VisParsees::_updateVisual() aswell.

Class WordgraphTable this class is so seldomly used and of such a limited usage that we might consider to remove it completely.

Namespace eval::cmd purge this interface with the help of `commandEval`

return values of the XCDG commands are a mess. See cmd::Verify() and cmd::Constraint().

Index

fgets