

Role Playing Database
A Computerized Role Playing Database
System
Tcl Internals

Robert Heller
Deepwoods Software
Wendell, MA, USA

February 10, 2000

This documentation was prepared with L^AT_EX.

This document describes version 2.0 of the Role Playing Database package.

Copyright ©1997-1999 by Robert Heller D/B/A Deepwoods Software

All rights reserved. Permission is granted to copy this document in electronic form only, so long as it is with the software it documents.

The author, Robert Heller, may be contacted electronically (E-Mail) via the following:

FidoNet 1:321/153, Locks Hill BBS.

InterNet heller@deepsoft.com

Web site URL: <http://www.deepsoft.com/>.

Contents

1	Introduction	1
1.1	Copyright notice.	1
1.2	Introduction	1
2	Main.tcl – Main window and common GUI procedures	3
2.1	global tcl_rcFileName	3
2.2	global LibDir	3
2.3	global CopyBuffer	3
2.4	image photo DeepwoodsBanner	3
2.5	global GenSym	4
2.6	proc SplashScreen	4
2.7	proc GenerateToplevelName	4
2.8	proc GetTopLevelOffFocus	4
2.9	proc RPGMenuBar	5
2.10	proc RPGDestroyToplevel	5
2.11	proc NoOperation args	5
2.12	proc RPGDestroyToplevelEvent	5
2.13	proc RPGToplevel	6
2.14	proc SetDirty	6
2.15	proc CloseWindow	6
2.16	proc ExitRPG	6
2.17	proc OpenWindow	7
2.18	proc OpenWhat	7
2.19	proc SaveWindow	7
2.20	proc SaveAsWindow	7
2.21	proc PrintWindow	8
2.22	proc MainWindow	8
2.23	proc CheckInt	8

2.24	proc CheckFloat	8
2.25	proc UpdateComments	9
3	StdMenuBar.tcl – Create standard menubars	11
3.1	proc MakePullDown	11
3.2	proc MakeFilePullDown	11
3.3	proc MakeEditPullDown	12
3.4	proc MakeHelpPullDown	12
3.5	proc MakeStandardMenuBar	12
3.6	proc remoSp	12
3.7	proc AddExtraMenuButton	13
3.8	proc GetMenuByName	13
3.9	proc DeleteMenuByName	13
4	RPGHelp.tcl – Online Help subsystem	15
4.1	global HelpDir	15
4.2	global HLinePattern	15
4.3	global HelpHistoryList	15
4.4	global HelpHistoryListIndex	16
4.5	proc TkIsLoadedP	16
4.6	proc rpgHelpTextHHGoto	16
4.7	proc rpgHelpTextHHSearch	16
4.8	proc rpgHelpTextHHRSearch	16
4.9	proc helptext	16
4.10	proc CreateHelpWindow	17
4.11	proc dup	17
4.12	proc HelpBackTopic	17
4.13	proc HelpForwardTopic	17
4.14	proc HelpTopic	17
4.15	proc HeaderFormat	17
4.16	proc BodyFormat	18
4.17	proc BindTagsAll	18
4.18	proc BindTagsW	18
4.19	proc XmTrackingLocate	18
4.20	proc XmTrackingLocateClick	18
4.21	proc HelpContext	18
4.22	proc HelpWindow	18

5	RPGEdCharacter.tcl – The characters in the game.	19
5.1	proc RPGEdCharacter	19
5.2	proc UpdateCharacterPicture	19
5.3	proc CheckWriteDirtyRecordCharacter	20
5.4	proc RollChar	20
5.5	proc ChRollExceptionalStrength	20
5.6	proc AdvanceChLevel	20
5.7	proc SaveAsCharacter	21
5.8	proc SaveCharacter	21
5.9	proc LoadCharacter	21
5.10	proc OpenCharacter	21
6	RPGEdDressing.tcl – Various items that might be about	23
6.1	proc RPGEdDressing	23
6.2	proc SaveAsDressing	23
6.3	proc SaveDressing	24
6.4	proc LoadDressing	24
6.5	proc CheckWriteDirtyRecordDressing	24
6.6	proc OpenDressing	24
6.7	proc ChangeDressingPicture	25
7	RPGEdMap.tcl – Maps of the adventure	27
7.1	proc RPGEdMap	27
7.2	proc SelectLevel	27
7.3	proc ReadCompleteList	28
7.4	proc ReadMap	28
7.5	proc CreateNewMap	28
7.6	proc CheckWriteDirtyRecordMap	28
7.7	proc SaveMap	29
7.8	proc SaveAsMap	29
7.9	proc WriteMap	29
7.10	proc CleanUpMap	30
7.11	proc CreateNewMapCreateButton	30
7.12	proc CreateNewMapDismissButton	30
7.13	proc AddNewSpace	30
7.14	proc ReDrawSpaces	31
7.15	proc SelectLoadedSpace	31
7.16	proc UnselectLoadedSpace	31

7.17	proc DeleteSpace	32
7.18	proc OpenMap	32
8	RPGEdMonster.tcl – The adversaries in the game	33
8.1	proc RPGEdMonster	33
8.2	proc UpdateMonsterPicture	33
8.3	proc CheckWriteDirtyRecordMonster	34
8.4	proc SaveAsMonster	34
8.5	proc SaveMonster	34
8.6	proc LoadMonster	34
8.7	proc OpenMonster	35
9	RPGEdSpace.tcl – The spaces where things happen	37
9.1	proc CreateNewSpace	37
9.2	proc SelectSpaceColor	37
9.3	proc CreateNewSpaceCancelButton	37
9.4	proc CreateNewSpaceCreateButton	38
9.5	proc EditLoadedSpace	38
9.6	proc SelectLbEnB	38
9.7	proc DrawXHairs	39
9.8	proc CloseSpaceEdit	39
9.9	proc AddExit	39
9.10	proc NewExitCancel	40
9.11	proc NewExitOk	40
9.12	proc DeleteExit	40
9.13	proc DeleteItem	40
9.14	proc AddItem	41
9.15	procNewItemCancel	41
9.16	procNewItemOk	41
9.17	proc VisitItem	41
9.18	proc VisitExit	42
9.19	proc DismissVisitExit	42
9.20	proc BrowseImageGif	42
9.21	proc BrowseItemFile	42
9.22	proc ComputeNextSpace	43

10 RPGEdSpell.tcl – Magical activities	45
10.1 proc RPGEdSpell	45
10.2 proc CheckWriteDirtyRecordSpell	45
10.3 proc LoadSpell	46
10.4 proc SaveAsSpell	46
10.5 proc SaveSpell	46
10.6 proc OpenSpell	46
11 RPGEdTreasure.tcl – Goodies to take home, if you can	47
11.1 proc RPGEdTreasure	47
11.2 proc OpenTreasure	47
11.3 proc SaveAsTreasure	48
11.4 proc SaveTreasure	48
11.5 proc LoadTreasure	48
11.6 proc CheckWriteDirtyRecordTreasure	48
11.7 proc ChangeTreasurePicture	48
12 RPGEdTrickTrap.tcl – Tricks and traps, things to avoid	49
12.1 proc RPGEdTrickTrap	49
12.2 proc SaveAsTrickTrap	49
12.3 proc SaveTrickTrap	50
12.4 proc LoadTrickTrap	50
12.5 proc CheckWriteDirtyRecordTrickTrap	50
12.6 proc OpenTrickTrap	50
12.7 proc ChangeTrickTrapPicture	51
Bibliograpy	53
Index	55

Chapter 1

Introduction

1.1 Copyright notice.

Role Playing DB – A database package that creates and maintains a database of RPG characters, monsters, treasures, spells, and playing environments.

Copyright (C) 1995,1998-1999 Robert Heller D/B/A Deepwoods Software
51 Locke Hill Road
Wendell, MA 01379-9728

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

1.2 Introduction

This manual describes the Tcl scripts used by the Role Playing Database. This consists of a collection of scripts that implement GUI editors for the

various base data structures used to represent the various informational objects used in Role Playing Games. These informational objects consist of *Characters*, *Monsters*, *“Magic” Spells*, *(Dungeon) Dressings*, and *“Maps”*.

Although TSR’s *Advanced Dungeons & Dragons* [1], [2], and [3] were used as a reference in the design of these GUIs, they should be generic enough to be usable for any other Role Playing Game system.

Chapter 2

Main.tcl – Main window and common GUI procedures

2.1 global tcl_rcFileName

This is the name of the file containing general Tcl/Tk preferences.

2.2 global LibDir

This is the path to the script library directory. It is computed from the directory name of the script.

2.3 global CopyBuffer

Buffer for selection copying.

2.4 image photo DeepwoodsBanner

Deepwoods banner image. Used in the splash screen.

2.5 global GenSym

Global variable used to hold symbol sequence counter used for generating unique symbols used for toplevel window names.

2.6 proc SplashScreen

Build the “Splash Screen” – A popup window that tells the user what we are all about. It gives the version and brief copyright information.

The upper part of the splash screen gives the brief information, with directions on how to get detailed information. The lower part contains an image banner for Deepwoods Software.

2.7 proc GenerateToplevelName *base*

Procedure to generate a unique toplevel window name.

in *base* – this is the base name to be used for the name. A sequence number is appended to this name to form a unique name.

2.8 proc GetTopLevelOfFocus *menu*

Procedure to get the toplevel that presently has focus. This is used when generic pulldown menus are activated to determine which object the menu refers to.

in *menu* – this is the menu that was selected. This is used to select which display to search on.

2.9 proc RPGMenuBar {*toplevel* .}

Procedure to generate a menu bar for RPG toplevels. This procedure is used to create uniform menubars for all first-class toplevels used in the RPG DB package.

in toplevel – the toplevel to attach the menu to.

2.10 proc RPGDestroyToplevel *tname*

This procedure destroys a selected RPG DB toplevel. The data object associated with the toplevel is checked to see if it is “dirty” (modified) and if so, the user is given a chance to save the data object to disk. The memory used by the data object is also freed.

in tname – the name of the toplevel.

2.11 proc NoOperation args

Dummy procedure to be used as a place holder where a procedure is needed, but there is no operation to perform.

in args – random unused arguments.

2.12 proc RPGDestroyToplevelEvent *tl*

This procedure is bound to the Destroy event of RPG DB toplevels. If ever the toplevel is destroyed via an event, this procedure catches the event and does the needed cleanup. See RPGDestroyToplevel.

in tl – the name of the toplevel.

2.13 **proc RPCToplevel** *tname title class*

Procedure to create a proper RPG DB toplevel. This procedure creates a toplevel window to edit an object of the specified class.

in *tname* – The name to use for the toplevel window.

in *title* – The string to place in the title of the toplevel window.

in *class* – The class of object the toplevel will edit.

2.14 **proc SetDirty** *tl*

Procedure to set the dirty flag for a toplevel's data object.

in *tl* – The name of the toplevel.

2.15 **proc CloseWindow** *toplevel*

Procedure to close a toplevel (bound to the “Close” button under the “File” menu). This procedure closes a named toplevel window. Proper cleanup operations are performed.

in *toplevel* – The name of the toplevel.

2.16 **proc ExitRPG**

Exit procedure for the RPG DB system. This procedure cleans up all outstanding toplevels and also confirms the exit with a dialog box.

2.17 proc OpenWindow *tl*

Procedure bound to the “Open...” menu item under the “File” menu. This procedure opens up a new window with old data (loaded from disk). If the current toplevel is of a particular class, a data object of the same class is opened. Otherwise a menu listing a selection of object classes is presented to the user to choose from.

in *tl* – The toplevel with the focus.

2.18 proc OpenWhat

This procedure presents the user with a selection of object classes to open.

2.19 proc SaveWindow *tl*

Procedure bound to the “Save” menu item on the “File” menu. Saves the selected object in its current file name. If no file is associated with the object, the user is prompted for a new file name.

in *tl* - The toplevel which has the focus.

2.20 proc SaveAsWindow *tl*

Procedure bound to the “SaveAs...” menu item on the “File” menu. Saves the selected object in a new file name.

in *tl* - The toplevel which has the focus.

2.21 **proc PrintWindow** *tl*

Procedure bound to the “Print...” menu item on the “File” menu. Prints the selected object.

in *tl* - The toplevel which has the focus.

2.22 **proc MainWindow**

Procedure to create the main window. This window consists of a collection of buttons to open various other toplevel windows to edit various data objects.

2.23 **proc CheckInt** *E*

Procedure to check for legal integer entries. This procedure is bound to the KeyPress event.

in *E* – the entry widget to check.

2.24 **proc CheckFloat** *E*

Procedure to check for legal floating point entries. This procedure is bound to the KeyPress event.

in *E* – the entry widget to check.

2.25 proc UpdateComments *text field*

Procedure to update text fields. This procedure is bound to the KeyPress event.

in text – the Text widget to check.

in field – the data field associated with this text widget.

Chapter 3

StdMenuBar.tcl – Create standard menubars

This file contains code to create a standard Motif style menubar. A standard menubar contains “File”, “Edit”, “View”, “Options”, and “Help” pulldown menus. The “File”, “Edit”, and “Help” menus have standard menu items.

The menubars and menus generated by the procedures in this file fill the standards set forth in the Motif Style Guide.

3.1 `proc MakePullDown menu args`

This procedure creates a pulldown menu and adds the items specified in its argument list.

in menu – the menu to create.

in args – the items to place in the menu.

3.2 `proc MakeFilePullDown menu`

This procedure creates a standard File menu.

in menu – the menu to create.

3.3 **proc MakeEditPullDown *menu***

This procedure creates a standard Edit menu.

in *menu* – the menu to create.

3.4 **proc MakeHelpPullDown *menu***

This procedure creates a standard Help menu.

in *menu* – the menu to create.

3.5 **proc MakeStandardMenuBar {*name .menuBar*} {*toplevel .*}**

This procedure creates a standard Motif style menu bar. If this is run under Tcl/Tk 8.0, then the menu -type menubar / .toplevel config -menu .menu hack is used. For older versions of Tcl/Tk, a conventional horizontal frame of menubuttons is packed into the top of the toplevel.

in *name* – the name of the menu bar.

in *toplevel* – the name of the toplevel to pack the menu into.

3.6 **proc remoSp *string***

Procedure to remove random whitespace characters.

in *string* – the string to remove space from.

3.7 **proc AddExtraMenuButton** *buttonLabel* *{mbar .menuBar}*

Procedure to add an extra menu button to a menu bar. Typically this would be something like a “Windows” or “Special” menu.

in *buttonLabel* – the menu button label.

in *mbar* – the menubar to add the menu button to.

3.8 **proc GetMenuByName** *label {mbar .menuBar}*

Procedure to fetch the menu associated with the named label.

in *label* – the label to look for.

in *mbar* – the menu bar to search.

3.9 **proc DeleteMenuByName** *label {mbar .menuBar}*

Procedure to delete a menu button from a menu bar.

in *label* – the menu button to delete.

in *mbar* – the menubar to delete the menu button from.

Chapter 4

RPGHelp.tcl – Online Help subsystem

This file contains the code to implement the Online Help subsystem.

4.1 global HelpDir

This is the path to the help file directory. It is computed from the script library directory.

4.2 global HLinePattern

This is the help file pattern to pick up a help header line.

4.3 global HelpHistoryList

This is the help history list.

4.4 **global HelpHistoryListIndex**

This is the help history index.

4.5 **proc TkIsLoadedP**

This procedure checks to see if the Tk package is loaded – this is only used to protect the global binding code to allow pkg_mkIndex to work properly.

4.6 **proc rpgHelpTextHHGoto *widget x y***

Function to implement the g/G key binding.

4.7 **proc rpgHelpTextHHSearch *widget x y***

Function to implement the s/S key binding.

4.8 **proc rpgHelpTextHHRSearch *widget x y***

Function to implement the r/R key binding.

4.9 **proc helptext *wname args***

Create a helptext widget.

in wname – the name of the widget.

in args – options.

4.10 proc CreateHelpWindow

Procedure to create the help window.

4.11 proc dup *str count*

Function to dup a string.

4.12 proc HelpBackTopic

Procedure to go “back” in the history list.

4.13 proc HelpForwardTopic

Procedure to go “forward” in the history list.

4.14 proc HelpTopic {*topic* {}} {*updateHistory* 1}

This procedure pops up a help dialog.

in *topic* – the topic to get help on.

4.15 proc HeaderFormat *text hline*

Procedure to format a header line.

4.16 **proc BodyFormat** *text body*

Procedure to format the body.

4.17 **proc BindTagsAll**

Procedure to set up the XmTrackingLocate bind tag.

4.18 **proc BindTagsW** *w*

Helper procedure to set up the XmTrackingLocate bind tag.

4.19 **proc XmTrackingLocate** *widget cursor*

Procedure to implement Motif's XmTrackingLocate function.

4.20 **proc XmTrackingLocateClick** *tl X Y*

XmTrackingLocate binding function.

4.21 **proc HelpContext** *{widget .}*

This procedure pops up an “On Context” help dialog.

4.22 **proc HelpWindow** *{widget .}*

This procedure pops up help for the current toplevel.

Chapter 5

RPGEdCharacter.tcl – The characters in the game.

This file deals with editing data objects that describe both player and non-player characters – intelligent persons that are involved in the adventure.

5.1 `proc RPGEdCharacter {filename {}}`

This procedure creates or edits a character object file.

in *filename* – the file to load a character object from.

5.2 `proc UpdateCharacterPicture tl obj img`

This procedure re-draws the character's picture, which is a GIF file.

in *tl* – the toplevel.

in *obj* – the data object.

in *img* – the new image file name.

5.3 **proc CheckWriteDirtyRecordCharacter *tl***

This procedure is called when the toplevel is closed and the toplevel's dirty flag is set. The user is asked if the data should be saved or not.

in *tl* - the toplevel.

5.4 **proc RollChar *tl***

This procedure “rolls” a new character. This procedure uses pseudo dice rolls to generate new character attribute statistics and the initial hit points.

in *tl* – the toplevel.

5.5 **proc ChRollExceptionalStrength *tl***

This procedure is for rolling the exceptional strength (fighters only) character attribute.

in *tl* – the toplevel.

5.6 **proc AdvanceChLevel *tl***

This procedure advances a character level. It is just a hook to the data object's AdvanceLevel method. The character's level is advanced and the character gains another die's worth of hit points.

in *tl* – the toplevel.

5.7 proc SaveAsCharacter *tl*

Procedure hooked to character “SaveAs...” menu item.

in *tl* – the toplevel.

5.8 proc SaveCharacter *tl* {*forceNew 0*}

Procedure hooked to character “Save” menu item.

in *tl* – the toplevel.

in *forceNew* – flag that indicates if we should force a new filename or not.

5.9 proc LoadCharacter *tl*

Procedure to load a character into the GUI from a disk file. Bound to the “Load” button.

in *tl* – the toplevel.

5.10 proc OpenCharacter *tl*

This procedure opens a character object file in a new GUI toplevel. Bound to the “Open...” menu item.

in *tl* – the toplevel.

Chapter 6

RPGEdDressing.tcl – Various items that might be about

This file deals with various fixed items that might be in a room or other place in an adventure: things like furniture and other items that “dress up the place”

6.1 `proc RPGEdDressing {filename {}}`

This procedure creates or edits a (dungeon) “dressing” – things like furniture and related items.

in *filename* – the file to open.

6.2 `proc SaveAsDressing tl`

This procedure saves the current object in a prompted for file. Bound to the “SaveAs...” menu item on the “File” menu.

in *tl* – the toplevel to save.

6.3 **proc SaveDressing *tl* {*forceNew* 0}**

This procedure saves the current object in the file named in the structure. Bound to the “Save” menu item on the “File” menu.

in *tl* – the toplevel to save.

in *forceNew* – force a new filename flag.

6.4 **proc LoadDressing *tl***

This procedure loads a dressing object file into the current GUI toplevel.

in *tl* – the toplevel to load into.

6.5 **proc CheckWriteDirtyRecordDressing *tl***

This procedure is called when the toplevel is being closed and the data structure is “dirty” (modified). The user is asked if the data should be saved.

in *tl* – the toplevel.

6.6 **proc OpenDressing *tl***

This procedure loads a dressing data object file into a new toplevel GUI window.

in *tl* – the current toplevel.

6.7 `proc ChangeDressingPicture` *tl img*

This procedure is used to update the dressing's picture (GIF file).

in `tl` – the current toplevel.

in `img` – the image to update.

Chapter 7

RPGEdMap.tcl – Maps of the adventure

This file implements the GUIs that create and edit maps.

7.1 `proc RPGEdMap {filename {}}`

This procedure edits a map file. A map contains a series of levels, each of which contain one or more spaces.

in *filename* – the name of the map file to edit.

7.2 `proc SelectLevel tl level`

This procedure selects the level to display. It is bound to the level scale widget.

in *tl* – the toplevel.

in *level* – the new level.

7.3 **proc ReadCompleteList** *fp bufferVar*

Helper function to read in a complete in the Tcl sense – balanced parens, braces, brackets, and quote marks.

in *fp* – file pointer object.

name *bufferVar* – the name of the buffer variable to collect the input in.

7.4 **proc ReadMap** *tl filename*

Procedure to read a map into a GUI toplevel. A map file contains a set of name value items that map to the data array used to represent the map while in memory. The saved fields include: shape, name, description, deepestLevel, highestLevel, maxX, maxY, minX, minY, and all of the space file names (Spaces,l,x,y).

in *tl* – the name of the toplevel.

in *filename* – the name of the file to read.

7.5 **proc CreateNewMap** *tl*

This procedure creates a new map.

in *tl* – the toplevel.

7.6 **proc CheckWriteDirtyRecordMap** *tl*

This procedure is called when the toplevel window is being closed and the data is “dirty” (modified). It takes care of saving the data if the user chooses to save it.

in *tl* – the toplevel.

7.7 proc SaveMap *tl*

Procedure to save a map object. Bound to the “Save” menu item on the “File” menu.

in *tl* – the toplevel.

7.8 proc SaveAsMap *tl*

This procedure saves a map in an alternative file. Bound to the “SaveAs...” menu item of the “File” menu.

in *tl* – the toplevel to save.

7.9 proc WriteMap *tl filename {IfDirty 0} {forceName 0}*

Procedure to write out a map object to a file. The data is written out as a series of Tcl 2 element lists: slot name slot value. The slots saved are: shape, name, description, deepestLevel, highestLevel, maxX, maxY, minX, minY, and all of the space files. Each (dirty) space is also written out. The space file names are formed from the base map file name with the spaces l, x, and y coordinates encoded into the filenames

in *tl* – the toplevel to write.

in *filename* – the filename to write to.

in *IfDirty* – write out only dirty spaces.

in *forceName* – compute new filenames for all spaces.

7.10 **proc CleanUpMap *tl***

Clean up procedure. Free up all of the space objects and then free up the map array itself.

in *tl* – the toplevel to clean up.

7.11 **proc CreateNewMapCreateButton *tl dialog***

Procedure to handle the Create button on the create new map dialog.

in *tl* – the toplevel to attach the new map to.

in *dialog* – the dialog used to set up the creation parameters.

7.12 **proc CreateNewMapDismissButton *tl dialog***

Procedure to handle the Dismiss button on the create new map dialog.

in *tl* – the toplevel to attach the new map to.

in *dialog* – the dialog used to set up the creation parameters.

7.13 **proc AddNewSpace *tl***

Procedure to add a new space to the current level.

in *tl* – the toplevel.

7.14 **proc ReDrawSpaces** *tl*

Procedure to re-draw the spaces in the current level.

in *tl* – the toplevel.

7.15 **proc SelectLoadedSpace** *tl space id indexString*

Procedure bound to button 1 – Select. This procedure marks the space as selected. The space's border is changed from black to red (highlighted) and the space's access info is saved in the data array for use elsewhere.

in *tl* – the toplevel.

in *space* – the Space object.

in *id* – the space's canvas id.

in *indexString* – the space's index string.

7.16 **proc UnselectLoadedSpace** *tl Mx My*

Procedure bound to the base canvas's button 1 event. This procedure is used to deselect the currently selected space when the user clicks on the canvas background.

in *tl* – the toplevel.

in *Mx* – mouse X coordinate.

in *My* – mouse Y coordinate.

7.17 `proc DeleteSpace` *tl*

Procedure to delete the selected space.

in *tl* – the toplevel.

7.18 `proc OpenMap` *tl*

Procedure to open a new Map GUI toplevel and load a map file into it.

in *tl* – the current toplevel.

Chapter 8

RPGEdMonster.tcl – The adversaries in the game

This file contains code to create and edit Monster objects. These objects are used to represent “monsters” – creatures and beings that the players’ characters need to defeat. These Monsters typically guard treasure or generally inhabit (guard) the places the player characters are traveling in while searching for treasure.

8.1 `proc RPGEdMonster {filename {}}`

This procedure edits a Monster object.

in filename – filename of the object to be edited.

8.2 `proc UpdateMonsterPicture tl img`

This procedure updates the picture (GIF file) of the monster.

in tl – the toplevel.

in img – the image file.

8.3 **proc CheckWriteDirtyRecordMonster *tl***

This procedure handles closing the window when the in memory copy of the object is “dirty” (modified). It asks the user if the object should be saved or not.

in *tl* – the toplevel.

8.4 **proc SaveAsMonster *tl***

This procedure is bound to the “SaveAs...” menu item. It saves the current object in a file to be named.

in *tl* – the toplevel.

8.5 **proc SaveMonster *tl* {*forceNew 0*}**

This procedure is bound to the “Save” menu item. It saves the current object in the file named in the data structure or in a new file name.

in *tl* – the toplevel.

in *forceNew* – flag to force the use of a new file name.

8.6 **proc LoadMonster *tl***

This procedure loads a new monster object into the current GUI.

in *tl* – the toplevel.

8.7 **proc OpenMonster *tl***

This procedure opens a monster object file in a new GUI toplevel window. It is bound to the “Open...” menu item on the “File” menu.

in *tl* – the current toplevel.

Chapter 9

RPGEdSpace.tcl – The spaces where things happen

This file contains the code to implement the GUI windows to create and edit spaces. This code is called from the code that edits Maps.

9.1 `proc CreateNewSpace maptl`

This procedure creates a new space.

in *maptl* – the map toplevel.

9.2 `proc SelectSpaceColor entry`

This procedure selects a background color for the space.

in *entry* – this is the entry widget where the name of the color is entered.

9.3 `proc CreateNewSpaceCancelButton maptl dialog`

This procedure handles the Cancel button on the space creation dialog.

in maptl – the map toplevel.

in dialog – the space creation dialog.

9.4 **proc CreateNewSpaceCreateButton** *maptl dialog*

This procedure handles the Create button on the space creation dialog.

in maptl – the map toplevel.

in dialog – the space creation dialog.

9.5 **proc EditLoadedSpace** *maptl spFile sp level*

This procedure edits a loaded space.

in maptl – the map toplevel.

in spFile – the name of the space file.

in sp – the space object.

in level – the level the space is on.

9.6 **proc SelectLbEnB** *base i*

Procedure to handle listbox selection for the exit and item listboxes. This procedure is bound to the select button of the graphical item.

in base – frame containing the section of the dialog that handles the list.

in i – the listbox index that was selected.

9.7 **proc DrawXHairs** *canvas x y*

Procedure to adjust the crosshairs and update the position information. This procedure is bound to pointer motion.

in canvas – the canvas widget.

in x – the mouse X coordinate.

in y – the mouse Y coordinate.

9.8 **proc CloseSpaceEdit** *maptl sp level*

Procedure bound to the Close button on the space edit dialog. Closes the dialog and updates the map.

in maptl – the map toplevel.

in sp – the space being edited.

in level – the level the space is on.

9.9 **proc AddExit** *maptl sp level*

This procedure adds a new exit to the space.

in maptl – the map toplevel.

in sp – the space object.

in level – the level the space is on.

9.10 **proc NewExitCancel** *maptl*

Procedure bound to the Cancel button on the Add Exit dialog

in *maptl* – the map toplevel.

9.11 **proc NewExitOk** *maptl sp*

Procedure bound to the Ok button on the Add Exit dialog

in *maptl* – the map toplevel.

9.12 **proc DeleteExit** *maptl sp*

This procedure deletes an Exit. Bound to the Delete button under the Exit list.

in *maptl* – the map toplevel.

in *sp* – the space object.

9.13 **proc DeleteItem** *maptl sp*

This procedure deletes an Item. Bound to the Delete button under the Item list.

in *maptl* – the map toplevel.

in *sp* – the space object.

9.14 **proc AddItem** *maptl sp*

This procedure adds a new item to the space.

in maptl – the map toplevel.

in sp – the space object.

9.15 **procNewItemCancel** *maptl*

Procedure bound to the Cancel button on the Add Item dialog

in maptl – the map toplevel.

9.16 **procNewItemOk** *maptl sp*

Procedure bound to the Ok button on the Add Item dialog

in maptl – the map toplevel.

9.17 **procVisitItem** *sp i*

Procedure to visit the selected item. Bound to the right button (or Command button on the Mac).

in sp – the space.

in i – the item index.

9.18 **proc VisitExit** *sp i {parent {}}*

Procedure to visit the selected edit. Bound to the right button (or Command button on the Mac).

in *sp* – the space.

in *i* – the exit index.

9.19 **proc DismissVisitExit** *tl*

Procedure bound to the Dismiss button on the visit Exit dialog.

in *tl* – the dialog box widget.

9.20 **proc BrowseImageGif** *entry title*

Procedure to browse for a GIF file for an item or exit.

in *entry* – entry widget where the file name is.

in *title* – title text to use.

9.21 **proc BrowseItemFile** *entry base*

Procedure to browse for an item file for an item.

in *entry* – entry widget where the file name is.

in *title* – title text to use.

9.22 **proc ComputeNextSpace** *maptl sp level*

Procedure to compute the next space for an exit. This code is somewhat approximate – there are situations where the results might need manual adjustments.

in maptl – the map toplevel.

in sp – the current space object.

in level – the level that the space is on.

Chapter 10

RPGEdSpell.tcl – Magical activities

This file contains the code to create and edit spell objects. These objects describe what is needed to cast magic spells.

10.1 `proc RPGEdSpell {filename {}}`

This procedure creates or edits a spell object.

in *filename* – the file to edit.

10.2 `proc CheckWriteDirtyRecordSpell tl`

This procedure is called when the GUI window is being closed and the data is “dirty” (modified). It takes care of saving the data if the user wants to.

in *tl* – the toplevel.

10.3 **proc LoadSpell *tl***

This procedure loads a spell data file into the current toplevel GUI window.

in *tl* – the toplevel GUI.

10.4 **proc SaveAsSpell *tl***

This procedure saves the current data object in a named file.

in *tl* – the toplevel.

10.5 **proc SaveSpell *tl* {*forceNew 0*}**

This procedure saves the current data object in a named file.

in *tl* – the toplevel.

in *forceNew* – flag to force a new file name.

10.6 **proc OpenSpell *tl***

This procedure opens a spell object data file in a new toplevel GUI window.

in *tl* – the current toplevel GUI.

Chapter 11

RPGEdTreasure.tcl – Goodies to take home, if you can

This file contains the code to create and edit treasure objects. These objects describe items of value that the player characters are trying to get so that they can amass wealth (of various sorts). Increased wealth means experience points gained and that means level advancement.

11.1 `proc RPGEdTreasure {filename {}}`

This procedure creates and edits treasure objects.

in *filename* – the file to edit.

11.2 `proc OpenTreasure tl`

This procedure loads a treasure object file into a new GUI toplevel window.

in *tl* – the current toplevel window.

11.3 **proc SaveAsTreasure *tl***

This procedure saves the currently open treasure object in a disk file.

in *tl* – the toplevel.

11.4 **proc SaveTreasure *tl* {*forceNew 0*}**

This procedure saves the currently open treasure object in a disk file.

in *tl* – the toplevel.

in *forceNew* – flag to force a new file name.

11.5 **proc LoadTreasure *tl***

This procedure loads a treasure data file into the current toplevel GUI.

in *tl* – the toplevel to load.

11.6 **proc CheckWriteDirtyRecordTreasure *tl***

This procedure handles toplevel rundown when the data object is “dirty” (modified). The data is saved out to a file if the user wants.

in *tl* – the toplevel.

11.7 **proc ChangeTreasurePicture *tl img***

This procedure changes the image (picture, GIF file) associated with the treasure object.

in *tl* – the toplevel GUI window.

in *img* – the image object.

Chapter 12

RPGEdTrickTrap.tcl – Tricks and traps, things to avoid

This file handles the creation and editing of tricks and traps, which are things the game master places in the game area to cause trouble for the player characters. Generally these tricks and traps are used to protect treasures as an alternative to placing a guardian monster to guard the treasure.

12.1 `proc RPGEdTrickTrap {filename {}}`

This procedure creates and edits trick/trap objects.

in filename – the file to edit.

12.2 `proc SaveAsTrickTrap tl`

This procedure saves the currently open trick/trap object in a disk file.

in tl – the toplevel.

12.3 **proc SaveTrickTrap *tl* {*forceNew* 0}**

This procedure saves the currently open trick/trap object in a disk file.

in *tl* – the toplevel.

in *forceNew* – flag to force a new file name.

12.4 **proc LoadTrickTrap *tl***

This procedure loads a trick/trap data file into the current toplevel GUI.

in *tl* – the toplevel to load.

12.5 **proc CheckWriteDirtyRecordTrickTrap *tl***

This procedure handles toplevel rundown when the data object is “dirty” (modified). The data is saved out to a file if the user wants.

in *tl* – the toplevel.

12.6 **proc OpenTrickTrap *tl***

This procedure loads a trick/trap object file into a new GUI toplevel window.

in *tl* – the current toplevel window.

12.7 proc ChangeTrickTrapPicture *tl img*

This procedure changes the image (picture, GIF file) associated with the trick/trap object.

in *tl* – the toplevel GUI window.

in *img* – the image object.

Bibliography

- [1] Gary Gygax. *Monster Manual*. TSR Games, Lake Geneva, WI, 1977, 1978.
- [2] Gary Gygax. *Players Handbook*. TSR Games, Lake Geneva, WI, 1978.
- [3] Gary Gygax. *Dungeon Masters Guide*. TSR Games, Lake Geneva, WI, 1979.

Index

- AddExit
 - procedure, 39
- AddExtraMenuButton
 - procedure, 13
- AddItem
 - procedure, 41
- AddNewSpace
 - procedure, 30
- AdvanceChLevel
 - procedure, 20
- BrowseImageGif
 - procedure, 42
- BrowseItemFile
 - procedure, 42
- ChangeDressingPicture
 - procedure, 25
- ChangeTreasurePicture
 - procedure, 48
- ChangeTrickTrapPicture
 - procedure, 51
- CheckFloat
 - procedure, 8
- CheckInt
 - procedure, 8
- CheckWriteDirtyRecordCharacter
 - procedure, 20
- CheckWriteDirtyRecordDressing
 - procedure, 24
- CheckWriteDirtyRecordMap
 - procedure, 28
- CheckWriteDirtyRecordMonster
 - procedure, 34
- CheckWriteDirtyRecordSpell
 - procedure, 45
- CheckWriteDirtyRecordTreasure
 - procedure, 48
- CheckWriteDirtyRecordTrickTrap
 - procedure, 50
- ChRollExceptionalStrength
 - procedure, 20
- CleanUpMap
 - procedure, 30
- CloseSpaceEdit
 - procedure, 39
- CloseWindow
 - procedure, 6
- ComputeNextSpace
 - procedure, 43
- CopyBuffer
 - global variable, 3
- CreateNewMap
 - procedure, 28
- CreateNewMapCreateButton
 - procedure, 30
- CreateNewMapDismissButton
 - procedure, 30
- CreateNewSpace
 - procedure, 37
- CreateNewSpaceCancelButton

- procedure, 38
- CreateNewSpaceCreateButton
 - procedure, 38
- DeepwoodsBanner
 - image, 3
- DeleteExit
 - procedure, 40
- DeleteItem
 - procedure, 40
- DeleteMenuByName
 - procedure, 13
- DeleteSpace
 - procedure, 32
- DismissVisitExit
 - procedure, 42
- DrawXHairs
 - procedure, 39
- EditLoadedSpace
 - procedure, 38
- ExitRPG
 - procedure, 6
- GenerateToplevelName
 - procedure, 4
- GenSym
 - global variable, 4
- GetMenuByName
 - procedure, 13
- GetTopLevelOfFocus
 - procedure, 4
- HelpDir
 - global variable, 15
- LibDir
 - global variable, 3
- LoadCharacter
 - procedure, 21
- LoadDressing
 - procedure, 24
- LoadMonster
 - procedure, 34
- LoadSpell
 - procedure, 46
- LoadTreasure
 - procedure, 48
- LoadTrickTrap
 - procedure, 50
- MainWindow
 - procedure, 8
- MakeEditPullDown
 - procedure, 12
- MakeFilePullDown
 - procedure, 11
- MakeHelpPullDown
 - procedure, 12
- MakePullDown
 - procedure, 11
- MakeStandardMenuBar
 - procedure, 12
- NewExitCancel
 - procedure, 40
- NewExitOk
 - procedure, 40
- NewItemCancel
 - procedure, 41
- NewItemOk
 - procedure, 41
- NoOperation
 - procedure, 5
- OpenCharacter
 - procedure, 21
- OpenDressing

- procedure, 24
- OpenMap
 - procedure, 32
- OpenMonster
 - procedure, 35
- OpenSpell
 - procedure, 46
- OpenTreasure
 - procedure, 47
- OpenTrickTrap
 - procedure, 50
- OpenWhat
 - procedure, 7
- OpenWindow
 - procedure, 7
- PrintWindow
 - procedure, 8
- ReadCompleteList
 - procedure, 28
- ReadMap
 - procedure, 28
- ReDrawSpaces
 - procedure, 31
- remoSp
 - procedure, 12
- RollChar
 - procedure, 20
- RPGDestroyToplevel
 - procedure, 5
- RPGDestroyToplevelEvent
 - procedure, 5
- RPGEdCharacter
 - procedure, 19
- RPGEdDressing
 - procedure, 23
- RPGEdMap
 - procedure, 27
- RPGEdMonster
 - procedure, 33
- RPGEdSpell
 - procedure, 45
- RPGEdTreasure
 - procedure, 47
- RPGEdTrickTrap
 - procedure, 49
- RPGMenuBar
 - procedure, 5
- RPGToplevel
 - procedure, 6
- SaveAsCharacter
 - procedure, 21
- SaveAsDressing
 - procedure, 23
- SaveAsMap
 - procedure, 29
- SaveAsMonster
 - procedure, 34
- SaveAsSpell
 - procedure, 46
- SaveAsTreasure
 - procedure, 48
- SaveAsTrickTrap
 - procedure, 49
- SaveAsWindow
 - procedure, 7
- SaveCharacter
 - procedure, 21
- SaveDressing
 - procedure, 24
- SaveMap
 - procedure, 29
- SaveSpell
 - procedure, 46

- SaveTreasure
 - procedure, 48
- SaveTrickTrap
 - procedure, 50
- SaveWindow
 - procedure, 7
- SelectLbEnB
 - procedure, 38
- SelectLevel
 - procedure, 27
- SelectLoadedSpace
 - procedure, 31
- SelectSpaceColor
 - procedure, 37
- SetDirty
 - procedure, 6
- SplashScreen
 - procedure, 4
- tcl_rcFileName
 - global variable, 3
- UnselectLoadedSpace
 - procedure, 31
- UpdateCharacterPicture
 - procedure, 19
- UpdateComments
 - procedure, 9
- UpdateMonsterPicture
 - procedure, 33
- VisitExit
 - procedure, 42
- VisitItem
 - procedure, 41
- WriteMap
 - procedure, 29