OSA tcl

Ported By

Tim Endres

December 1993

Release 1.0v1

Internet:. ....tickle@ice.com
UUCP:..............heifetz!tbomb!tickle
AppleLink: tickle@ice.com@INTERNET#
CompuServe:.......>INTERNET:tickle@ice.com

Table Of Contents

Introduction....3
What is OSA?....3
What Is Tcl ?....3
Installing OSA tcl....3
Macintosh Tcl Extensions....4
Tcl Extensions....7
Tickle Tcl Special Context Variables....7
Tcl Environment....8
Index....9

Introduction

This is a system extension that implements the tcl scripting language as an OSA scripting component.

If you have any questions, suggestions, or just have some compelling need to send email, please address it to **tickle@ice.com**.

What is OSA?

Apple's Open Scripting Architecture (OSA) allows any number of scripting systems to be created and used together on your Macintosh. AppleScript™ is one instance of a scripting language that is provided via OSA.

What Is Tcl ?

You should refer to the various included documents and papers submitted to USENIX for details about tcl. Briefly, tcl is a simple string based scripting language designed to be included in standard applications. It will allow you to perform many tasks that are otherwise very labor intensive.

Tcl was created by a small set of very bright engineers, namely, John Ousterhout and his assistants. The Unix extensions were invented and implemented by Karl Lehenbauer, Mark Diekhans and Peter da Silva with help from Jordan Henderson. Tcl was ported into tickle by Tim Endres. Refer to the included documentation for further details.

Installing OSA tcl

Installing the OSA tcl system extension is accomplished in the same manner as any other system extension. Drop the *OSA tcl* file icon on top of your System Folder icon and the system will automatically copy the file to the *Extensions* folder. Or you can drop the OSA tcl file directly on the Extensions folder if you chose.

Once the file has been copied into the Extensions folder, reboot your Macintosh to cause the scripting extension to be loaded and made available to all scripting capable applications.

Macintosh Tcl Extensions

The following commands have been added to the standard tcl command set for the general support of Macintosh computers. These are commands that should be available regardless of the implementation of the application or environment, since they can be linked into almost any Macintosh application.

ls ?options? ?patterns...?

This command will display a list of files in the current directory. If a *pattern* is specified, the glob command will be invoked on the pattern and the resulting matched files listed. Options include:

-C	**– List the files in a columnar format (80 columns)**
-F	**– Append a character to file names to indicate their type.** **uses '•' for applications and ':' for directories.**
-H	**– print a header is the -l option is specified.**
-l	**– print a long listing, including file details.**

cd directory

This command will set the current working directory to *directory*. The directory argument may or may not end with a colon. The working directory is the directory in which file names that contain no colons (folders) will be located with file related commands.

pwd

This command will return the current working directory.

mkdir ?-path? dirList

Creates the directories listed in *dirList*. If the *-path* option is included, all directories in each directory specified in *dirList* will be created if it does not exist.

rmdir dirList

Deletes the directories specified in the *dirList*.

cp fromfile tofile

Copy the file *fromfile* to *tofile*. This copies both forks and the Finder info.

mv fromfile tofile

This will rename *fromfile* to *tofile*, including changing directories if pathnames are specified. If the files reside on the same volume, then a simple rename is performed, preceded by a CatMove if different directory paths are specified. If the files reside on different volumes, then the file will first be copied, then the original deleted.

rm ?-nocomplain? fileList

This will delete the files listed in *fileList*. If the argument -*nocomplain* is specified, then no complaints or errors will be returned if a file does not exist or can not be deleted. The parameter *fileList* is usually the result of a glob.

unlink ?-nocomplain? fileList

This command is identical to *rm*.

getfinfo filename

Returns the Macintosh specific file information for *filename*. The information is returned as an array containing the following items:

```
Finder File Creator
Finder File Type
Attribute Characters (set setfinfo command)
File Creatior Date and Time
File Last Modification Data and Time
Finder Icon Location (H V)
File Data Fork Length / Directory DirID
File Resource Fork Length / Number Of Files In Directory
```

An example of getfinfo's output:

```
        getfinfo Fire:MPW:src:tickle:src:Event.c
    {MPS } TEXT bdlImsv {05/04/88 17:36:20} {11/18/93 02:36:50}
            {256 0} 12467 696
```

setfinfo filename ?-a attr? ?-c creator? ?-t type? ?-d createdate? ?-m moddate?

Sets the Macintosh specific file information for *filename*. The -a option's parameter *attr* is a string of characters where capitals are used to set the attribute and lowercase used to unset the attribute. The set of attribute characters is:

```
B      - Has Bundle
D      - On the desktop
I      - Initialized by Finder
L      - Locked / Alias
M      - Shared
S      - System File
V      - inVisible In Finder
```

The -t option sets the file's Finder File Type to *type*. The -c option sets the file's Finder Creator Type to *creator*. The -d and -m set the file's creation and last modified dates respectively. Both options accept a date string of the format:

```
mm/dd/yy [hh:mm[:ss]] [AM | PM]
```

where brackets enclose optional elements.

glob ?switches? pattern

This is the tcl glob command written in native Mac code, with several options specific to the Mac.

The new glob now takes several new*options*. They are:

-i **This option causes glob to list invisible files also.**

-t TYPE **This option causes glob to only list file with the indicated Finder file type*TYPE*. This option may be used with the -c option.**

-c CRTR **This option causes glob to only list file with the indicated Finder file creator*CRTR*. This option may be used with the -t option.**

Also note that the original glob would skip *dot files* (i.e., file names the begin with a period) in the listing, **unless** the pattern began with a period. Tickle extends this concept one step further on the Macintosh, and skips *spot files* (i.e., file names that begin with '•'), **unless** the pattern starts with a *spot*.

The old option used to have filename spaces escaped in the output is no longer necessary due to tcl's new list handling, which already compensates for spaces in result elements. Thus, the option has been removed and is no longer supported.

isalias filename

This command will return "1" if the file *filename* is an alias file, otherwise the result is "0".

resolve_alias filename

This command will return the resolved version of the alias file *filename* or simply return *filename* if the file is not an alias file.

mtime time ?long|short|abbrev?

This command returns a Macintosh formatted time string for *time*. The *long, short* and *abbrev* options correspond to those in Inside Mac. Default format is *short*.

now ?-unix?

Returns the current Macintosh time in seconds. Calls GetDateTime(). If the *-unix* switch is provided, the time will be adjust to be a UNIX GMT based time instead of Macintosh time.

ticks

Returns the current Macintosh tick count. Calls TickCount().

beep ?-list? ?sndName?

Plays a Macintosh beep sound. If the *-list* argument is provided, the list of available sounds to play is returned. If *sndName* is specified, then that sound is played for the beep.

echo arguments...

Echoes the arguments, separated by spaces, followed by a carriage return, to the current environment's output.

ctime time

Returns a time and date string in the format of the unix ctime() function. The *time* argument is Macintosh seconds and should normally be the result of the *now* command.

NOT IMPLEMENTED IN OSA.

cvttime selector time

Returns a time that has been converted from one domain to another. Currently, only the UNIX GMT domain and the Macintosh domain are supported. Time is an integer number of seconds. The selector is one of:

-mtu	**Convert from Macintosh to UNIX domain**
-utm	**Convert from UNIX to Macintosh domain**

source fileName | ?-rsrcfile path? [-rsrcname name | -rsrcid id]

The source command has been expanded for the Macintosh. It is now capable of being invoked to evaluate a TEXT resource as opposed to a file. If the *-rsrcname*, or *-rsrcid* options are specified, then the specified resource will be loaded, locked, and evaluated as a tcl script. This resource **must** end with the null character (0x00) since it must be a C string. If the *-rsrcfile* option is specified, then the TEXT resource will be searched for in *path*, otherwise it is searched for in the current application resource fork. If the source command is invoked with only a filename argument, then is will work as usual.

RESOURCE OPTIONS NOT IMPLEMENTED IN OSA.

Tcl Extensions

The following commands have been added to the standard tcl command set. The commands are the *extended* tcl commands. Please refer to the extended command set documentation provided with the tickle application or the tcl distribution.

close
eof
file
flush
gets
open
puts
read
seek
tell
loop
keyldel
keylget
keylkeys
keylset
lvarcat
lvarpop
lvarpush
lempty
lassign
cindex
clength
crange
csubstr
replicate
translit
ctype
ctoken
max
min
random
scanfile
scanmatch
scancontext

Tickle Tcl Special Context Variables

Certain tcl variables are set depending upon the context of the interpreter:

OSATCLVERS **The variable OSATCLVERS will always be defined in the format '1.0'.**

MACINTOSH **This variable is always set to 1 by OSA tcl.**

MAC_COMPONENT **This variable is always set to 1 by OSA tcl.**

Tcl Environment

OSA tcl supports some extensions to the *env* variable ot help with portable coding within Macintosh tcl. Each of these variables can be accessed via the command *set env(name)*. Here is a list of environment array settings set in each tickle interpretter created:

APPDIR	**Full path of tickle's folder.**
APPNAME	**Name of tickle application file (normally Tickle).**
APPLE_MENU_FOLDER	**Full pathname of Apple Menu Folder.**
CP_FOLDER	**Full pathname of Control Panels Folder.**
DESK_FOLDER	**Full pathname of Desktop Folder.**
EXT_FOLDER	**Full pathname of Extensions Folder.**
HOME	**Full pathname to user's home directory as constructed by Tickle's directory plus user's name.**
LOGIN	**User's name as indicated in Sharing Setup.**
MACHINE_NAME	**The Macintosh machine name (do not depend on this!).**
PREF_FOLDER	**Full pathname of Preferences Folder.**
PRINT_MON_FOLDER	**Full pathname of Print Monitor Documents Folder.**
PWD	**Full pathname of current working directory, updated with each use of the *cd* command.**
SHARED_TRASH_FOLDER	**Full pathname of the Shared Trash Folder.**
START_UP_FOLDER	**Full pathname of Startup Items Folder.**
SYSTEM_FOLDER	**Full pathname of System Folder.**
SYSTEM_VERSION	**The Macintosh System Version (e.g. 7.01).**
TEMP	**Full pathname of Temporary Folder for temporary files.**
TRASH_FOLDER	**Full pathname of Trash Folder.**

Index

APPDIR 8
APPLE_MENU_FOLDER 8
APPNAME 8
beep 6
cd 4
cp 4
CP_FOLDER 8
ctime 6
cvttime 6
DESK_FOLDER 8
echo 6
EXT_FOLDER 8
getfinfo 5
glob 5
HOME 8
isalias 6
LOGIN 8
ls 4
MACHINE_NAME 8
MACINTOSH 7
MAC_COMPONENT 7
mkdir 4
mtime 6
mv 4
now 6
OSATCLVERS 7
PREF_FOLDER 8
PRINT_MON_FOLDER 8
pwd 4, 8
resolve_alias 6
rm 4
rmdir 4
setfinfo 5
SHARED_TRASH_FOLDER 8
source 6
START_UP_FOLDER 8
SYSTEM_FOLDER 8
SYSTEM_VERSION 8
TEMP 8
ticks 6
TRASH_FOLDER 8
unlink 4