

File created: 28-Oct-89 12:13:26 {ICE}<KOOMEN>LISPUSERS>MEDLEY>LOGTIME.;3

changes to: (VARS LOGTIMECOMS)
(FNS LogTime.Prompt LogTime.Proc LogTime.Update)

previous date: 24-Oct-89 17:08:48 {ICE}<KOOMEN>LISPUSERS>KOTO>LOGTIME.;6

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

::
:: Copyright (c) 1989 by Johannes A. G. M. Koomen. All rights reserved.

(RPAQQ LOGTIMECOMS

```
[ (FNS LOGTIME.EDIT LOGTIME.REPORT LOGTIME.START LOGTIME.STOP LOGTIME.UPDATE)
  (INITVARS (LOGTIME.INTERVAL 15)
    (LOGTIME.DATAFILE (PACKFILENAME 'NAME "LogTime" 'EXTENSION "Data" 'BODY LOGINHOST/DIR))
    (LOGTIME.REPORTFILE T)
    (LOGTIME.PROMPT.URGENCY 'TTY))

  ;; Implementation

  (PROP MAKEFILE-ENVIRONMENT LOGTIME)
  (FNS LogTime.AroundExitFn LogTime.ButtonFn LogTime.Dump LogTime.Edit LogTime.GDate LogTime.IDate
    LogTime.Load LogTime.Message LogTime.Proc LogTime.Prompt LogTime.Quit LogTime.Report
    LogTime.ReportTime LogTime.Start LogTime.Update)
  (RECORDS LogTimeData)
  (INITVARS (LogTime.process NIL)
    (LogTime.promptWindow NIL)
    (LogTime.promptWindowRegion NIL)
    (LogTime.suspendedLogfile NIL))
  [ADDVARS (AROUNDEXITFNS LogTime.AroundExitFn)
    (BackgroundMenuCommands ("Log Time" (LogTime.Start)
      "Keep track of how time is spent"
      (SUBITEMS ("Update" (LogTime.Start)
        "Start or update log")
        ("Edit" (LogTime.Edit)
          "Edit current data")
        ("Report" (LogTime.Report)
          "Generate report on LOGTIME.REPORTFILE")
        ("Quit" (LogTime.Quit)
          "Quit keeping track of how time is spent and update
          log file"
          (SUBITEMS ("Abort" (LogTime.Quit T)
            "Quit keeping track of how time is
            spent but DON'T update log file"])))
      (VARS (BackgroundMenu NIL))
      (DECLARE%: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVERS (ADDVARS (NLAMA)
        (NLAML)
        (LAMA LogTime.Message])

(DEFINEQ
```

(LOGTIME.EDIT

```
[LAMBDA (DATAFILE NEWDATAFILE) (* Koomen "24-Oct-89 14:11")
  (DECLARE (GLOBALVARS LogTime.process))
  (PROG ((logtimedata (LogTime.Load DATAFILE))
    (WITH.MONITOR (fetch (LogTimeData lock) of logtimedata)
      [PROG (entries newentries entry newentry)
        [for old entry in (fetch (LogTimeData entries) of logtimedata)
          do (for pair in (CDR entry) do (push entries (LIST (CAR pair)
            (CDR pair)
            (CAR entry)

          (SETQ entries (DREVERSE (SORT entries T)))
          [for old entry in entries do (RPLACA entry (LogTime.GDate (CAR entry)))
            (RPLACA (CDR entry)
            (LogTime.GDate (CADR entry)

          (SETQ entries (EDITE entries))
          (if (for old entry in entries bind d1 d2 do (if [AND (SETQ d1 (LogTime.IDate (CAR entry)))
            (SETQ d2 (LogTime.IDate (CADR entry)
            then (RPLACA entry d1)
            (RPLACA (CDR entry)
            d2)
            else (LogTime.Message "Edit aborted -- bad format:

" entry)

            (RETURN T)))
          then (RETURN))
          (SETQ entries (SORT entries T))
          [for old entry in entries do (if (SETQ newentry (SASSOC (CADDR entry)
            newentries))
            then (push (CDR newentry)
            (CONS (CAR entry)
            (CADR entry)))
            else (push newentries (LIST (CADDR entry)
```



```

else (push newentries (CONS start (DIFFERENCE (CDR pair)
(CAR pair]
finally (RETURN newentries]
(SETQ entries (SORT entries T))
(if VERBOSEFLG
then (SETQ col1 (PLUS datewidth 3))
(SETQ col2 (NCHARS subtotalmsg))
[for entry in entries do (for pair in (CDR entry)
do (SETQ col2 (MAX col2 (NCHARS (CAR pair]
(SETQ col2 (PLUS col1 col2 3))
(for entry in entries bind secs subtotsecs curday newday
do (SETQ newday (\UNPACKDATE (CAR entry)))
(if (OR (NEQ (CADDR newday)
(CADDR curday))
(NEQ (CADR newday)
(CADR curday))
(NEQ (CAR newday)
(CAR curday)))
then (if (AND curday (GREATERP subtotsecs secs))
then (printout report .TAB col1 subtotalmsg .TAB col2)
(LogTime.ReportTime report subtotsecs totalsecs))
(SETQ curday newday)
(SETQ subtotsecs 0))
(printout report (LogTime.GDate (CAR entry)))
(for pair in (CDR entry) do (SETQ secs (CDR pair))
(add subtotsecs secs)
(printout report .TAB col1 (CAR pair)
.TAB col2)
(LogTime.ReportTime report secs totalsecs))
finally (if (GREATERP subtotsecs secs)
then (printout report .TAB col1 subtotalmsg .TAB col2)
(LogTime.ReportTime report subtotsecs totalsecs))
(printout report T "Total:" .TAB col2)
(LogTime.ReportTime report totalsecs))
else (SETQ col1 (PLUS (NCHARS (CAAR entries))
3))
(for entry in entries do (printout report (CAR entry)
.TAB col1)
(LogTime.ReportTime report (CDR entry)
totalsecs)
finally (printout report T "Total:" .TAB col1)
(LogTime.ReportTime report totalsecs)))
else (SETQ entries (SORT entries T))
(SETQ col1 (CONSTANT (NCHARS "Total:")))
[for entry in entries do (SETQ col1 (MAX col1 (NCHARS (CAR entry]
(add col1 3)
(if VERBOSEFLG
then (SETQ col2 (PLUS col1 datewidth 3))
(for entry in entries bind secs subtotsecs
do (printout report (CAR entry))
(SETQ subtotsecs 0)
(for pair in (CDR entry) do (SETQ secs (DIFFERENCE (CDR pair)
(CAR pair)))
(add subtotsecs secs)
(printout report .TAB col1
(LogTime.GDate (CAR pair))
.TAB col2)
(LogTime.ReportTime report secs totalsecs))
(if (GREATERP subtotsecs secs)
then (printout report .TAB col1 subtotalmsg .TAB col2)
(LogTime.ReportTime report subtotsecs totalsecs))
finally (printout report T "Total:" .TAB col2)
(LogTime.ReportTime report totalsecs))
else (for entry in entries bind secs subtotsecs
do (printout report (CAR entry))
[SETQ subtotsecs (for pair in (CDR entry) sum (DIFFERENCE (CDR pair)
(CAR pair]
(printout report .TAB col1)
(LogTime.ReportTime report subtotsecs totalsecs)
finally (printout report T "Total:" .TAB col1)
(LogTime.ReportTime report totalsecs))])

```

(LOGTIME.START

```

[LAMBDA (DATAFILE)
(LogTime.Start DATAFILE T)]

```

(* Koomen "12-Oct-89 09:50")

(LOGTIME.STOP

```

[LAMBDA (ABORTFLG WAITFLG)
(if (AND (PROCESSP LogTime.process)
(NOT (PROCESS.FINISHEDP LogTime.process)))
then (LogTime.Quit ABORTFLG T)
(while (AND WAITFLG (PROCESSP LogTime.process)
(NOT (PROCESS.FINISHEDP LogTime.process)))
do (BLOCK])

```

(* Koomen "24-Oct-89 14:39")

(LOGTIME.UPDATE

(* Koomen "12-Oct-89 10:30")

```
[LAMBDA NIL
  (DECLARE (GLOBALVARS LogTime.process))
  (if (AND (PROCESSP LogTime.process)
          (NOT (PROCESS.FINISHEDP LogTime.process))))
      then (WAKE.PROCESS LogTime.process])
)
```

(RPAQ? LOGTIME.INTERVAL 15)

(RPAQ? LOGTIME.DATFILE (PACKFILENAME 'NAME "LogTime" 'EXTENSION "Data" 'BODY LOGINHOST/DIR))

(RPAQ? LOGTIME.REPORTFILE T)

(RPAQ? LOGTIME.PROMPT.URGENCY 'TTY)

;; Implementation

(PUTPROPS LOGTIME MAKEFILE-ENVIRONMENT (:READTABLE "INTERLISP" :PACKAGE "INTERLISP" :BASE 10))

(DEFINEQ

(LogTime.AroundExitFn

(* Koomen "24-Oct-89 17:07")

```
[LAMBDA (EVENT)
  (SELECTQ (SUBATOM EVENT 1 5)
    (BEFOR (if (AND (PROCESSP LogTime.process)
                    (NOT (PROCESS.FINISHEDP LogTime.process))))
            then [LET [(logtimedata (PROCESSPROP LogTime.process 'LogTimeData)]
                      (LOGTIME.STOP NIL T)
                      (SETQ LogTime.suspendedLogfile (PACKFILENAME 'VERSION NIL 'BODY (fetch (LogTimeData logfile) of logtimedata)
                                                                    else (SETQ LogTime.suspendedLogfile NIL)))
            (AFTER (if LogTime.suspendedLogfile
                      then (LOGTIME.START LogTime.suspendedLogfile)
                      (SETQ LogTime.suspendedLogfile NIL)))
            NIL)
  NIL])
```

(LogTime.ButtonFn

(* Koomen "20-Sep-89 14:29")

```
[LAMBDA (window)
  ;; The rightbuttoneventfn on the LogTime promptwindow
  (DECLARE (GLOBALVARS LogTime.promptWindowRegion))
  (if (NOT (INSIDEP LogTime.promptWindowRegion (LASTMOUSEX window)
              (LASTMOUSEY window)))
      then (DOWINDOWCOM window)
      else (PROG (logdata entries menu activity)
                 (SETQ logdata (PROCESSPROP (WINDOWPROP window 'PROCESS)
                                             'LogTimeData))
                 (SETQ entries (fetch (LogTimeData entries) of logdata))
                 (if (NULL entries)
                     then (RETURN))
                 [SETQ activity (MENU (OR (fetch (LogTimeData activityMenu) of logdata)
                                           (replace (LogTimeData activityMenu) of logdata
                                                    with (create MENU
                                                                ITEMS _ (SORT (for entry in entries
                                                                    collect (CAR entry)))
                                                                TITLE _ "Activities:"])
                     (if (NULL activity)
                         then (RETURN))
                 ;; Erase current typein and insert menu selection
                 (BKSYSEBUF "Ó")
                 (BKSYSEBUF activity])
```

(LogTime.Dump

(* Koomen "24-Oct-89 17:07")

```
[LAMBDA (logtimedata)
  (DECLARE (GLOBALVARS LOGTIME.DATFILE))
  (if (fetch (LogTimeData entries) of logtimedata)
      then (PROG ((logfile (OR (fetch (LogTimeData logfile) of logtimedata)
                               LOGTIME.DATFILE)))
                 (LogTime.Message "saving data to " (FULLNAME logfile 'OLD/NEW))
                 (SETQ logfile (WRITEFILE (LIST* (fetch (LogTimeData firstUpdate) of logtimedata)
                                                  (fetch (LogTimeData lastUpdate) of logtimedata)
                                                  (fetch (LogTimeData entries) of logtimedata)
                                                  logfile))
                 (replace (LogTimeData logfile) of logtimedata with logfile)
                 (LogTime.Message "done saving data to " logfile)
                 (RETURN logfile])
```

(LogTime.Edit

```
[LAMBDA NIL
  (DECLARE (GLOBALVARS LogTime.process))
  (if (OR (NOT (PROCESSP LogTime.process))
        (PROCESS.FINISHEDP LogTime.process))
      then (LogTime.Message "not running")
      else (replace (LogTimeData status) of (PROCESSPROP LogTime.process 'LogTimeData) with 'Edit)
            (WAKE.PROCESS LogTime.process]))
```

(* Koomen "10-Oct-89 13:07")

(LogTime.GDate

```
[LAMBDA (DATE NOTIMEFLG)
  (GDATE DATE (if NOTIMEFLG
                  then (DATEFORMAT NO.TIME)
                  else (DATEFORMAT NO.SECONDS]))
```

(* Koomen "10-Oct-89 14:13")

(LogTime.IDate

```
[LAMBDA (DATE)
  ;; Like IDATE but round off to nearest minute
  (PROG NIL
    (RETURN (APPLY [FUNCTION (LAMBDA (YEAR MONTH DAY HR MIN SEC)
                              (\PACKDATE YEAR MONTH DAY HR MIN (if (LESSP SEC 30)
                                                                    then 0
                                                                    else 60)]
                    (\UNPACKDATE (OR (IDATE DATE)
                                     (RETURN NIL]))
```

(* Koomen "10-Oct-89 14:31")

(LogTime.Load

```
[LAMBDA (datafile)
  (DECLARE (GLOBALVARS LOGTIME.DATATYPE LogTime.process))
  (LET ((logfile (INFILEP (OR datafile LOGTIME.DATATYPE)))
        (logdata NIL))
    (if (AND (PROCESSP LogTime.process)
             (SETQ logdata (PROCESSPROP LogTime.process 'LogTimeData))
             (EQ logfile (fetch (LogTimeData logfile) of logdata)))
        then (LogTime.Message "using current data")
            logdata
        elseif logfile
            then (LogTime.Message "loading data from " logfile)
                 (SETQ logdata (CDR (READFILE logfile)))
                 (PROG1 (create LogTimeData
                               logfile _ logfile
                               lock _ (CREATE.MONITORLOCK)
                               firstUpdate _ (CAR logdata)
                               lastUpdate _ (CADR logdata)
                               entries _ (CDDR logdata))
                       (LogTime.Message "done loading data from " logfile))
        else (if datafile
                 then (LogTime.Message datafile " not found"))
                 (SETQ logfile (OUTFILEP (OR datafile LOGTIME.DATATYPE)))
                 (LogTime.Message "creating new data")
                 (SETQ logdata (LogTime.IDate))
                 (create LogTimeData
                       logfile _ logfile
                       lock _ (CREATE.MONITORLOCK)
                       firstUpdate _ logdata
                       lastUpdate _ logdata))
```

(* Koomen "12-Oct-89 10:12")

(LogTime.Message

```
[LAMBDA NARGS
  (DECLARE (GLOBALVARS PROMPTWINDOW))
  (printout PROMPTWINDOW .TAB0 0 "<LogTime: ")
  (for I to NARGS do (printout PROMPTWINDOW (ARG NARGS I)))
  (printout PROMPTWINDOW ">" T)]
```

(* Koomen "20-Sep-89 16:44")

(LogTime.Proc

```
[LAMBDA (datafile)
  (DECLARE (GLOBALVARS LOGTIME.INTERVAL))
  (LET [(logtimedata (PROCESSPROP (THIS.PROCESS)
                                  'LogTimeData)]
        (if logtimedata
            then ;; Already initialized (process restarted after a hardreset) so just restart the loop
            else (while \IDLING do (BLOCK))
                 (PROCESSPROP (THIS.PROCESS)
                               'LogTimeData
                               (SETQ logtimedata (LogTime.Load datafile)))
                 (replace (LogTimeData lastUpdate) of logtimedata with (IDATE))
                 (LogTime.Message "keeping track of time..."))
        (do (replace (LogTimeData status) of logtimedata with 'Collect)
            (BLOCK (if (NOT \IDLING)
                      then (TIMES LOGTIME.INTERVAL 60000)))
            (SELECTQ (fetch (LogTimeData status) of logtimedata)
```

(* Koomen "27-Oct-89 12:41")

```

(Collect (LogTime.Update logtimedata))
(Edit (LogTime.Update logtimedata)
  (LOGTIME.EDIT))
(Report (LogTime.Update logtimedata)
  (LOGTIME.REPORT T T (GDATE NIL (DATEFORMAT NO.TIME))))
(Abort (RETURN NIL))
(Quit (LogTime.Update logtimedata)
  (RETURN (LogTime.Dump logtimedata)))
(Quit! (LogTime.Update logtimedata T)
  (RETURN (LogTime.Dump logtimedata)))
(SHOULDNT "Bad LogTime status"))
(LogTime.Message "done keeping track of time")

```

(LogTime.Prompt

```

[LAMBDA (candidate) (* Koomen "28-Oct-89 12:02")
;; From (* bvm: "17-Sep-85 15:04") PopUpWindowAndGetAtom
(DECLARE (GLOBALVARS LASTMOUSEX LASTMOUSEY LOGTIME.PROMPT.URGENCY LogTime.promptWindow
  LogTime.promptWindowRegion SCREENHEIGHT SCREENWIDTH))
(CAR (NLSETQ (LET ((promptstring "Latest activity: ")
  (promptw LogTime.promptWindow))
  (if (NOT (WINDOWP promptw))
    then (LET* ((bordersize 10)
      (font (DEFAULTFONT))
      (promptwidth (STRINGWIDTH promptstring font))
      (answerwidth (TIMES 60 (CHARWIDTH (CHARCODE A)
        font)))
      (width (WIDTHIFWINDOW (PLUS promptwidth answerwidth)
        bordersize))
      (height (HEIGHTIFWINDOW (TIMES (FONTPROP font 'HEIGHT)
        2)
        NIL bordersize)))
      (SETQ LogTime.promptWindowRegion (CREATEREGION 0 0 width height))
      (SETQ promptw (CREATEW (COPY LogTime.promptWindowRegion)
        "Log Time: (click right for menu)" bordersize T))
      (WINDOWPROP promptw 'RIGHTBUTTONFN (FUNCTION LogTime.ButtonFn))
      (SETQ LogTime.promptWindow promptw)))
    [MOVEW promptw [IMIN LASTMOUSEX (IDIFFERENCE SCREENWIDTH (fetch (REGION WIDTH)
      of (WINDOWREGION promptw)
      (IMIN LASTMOUSEY (IDIFFERENCE SCREENHEIGHT (fetch (REGION HEIGHT)
      of (WINDOWREGION promptw)
      (RESETSAVE (OPENW promptw)
      (LIST (FUNCTION CLOSEW)
        promptw))
      (RESETSAVE NIL (LIST (FUNCTION CLEARW)
        promptw))
      (PROMPTFORWARD promptstring candidate NIL promptw NIL LOGTIME.PROMPT.URGENCY
      (CHARCODE (CR LF))

```

(LogTime.Quit

```

[LAMBDA (abortflag noninteractivep) (* Koomen "24-Oct-89 16:01")
(DECLARE (GLOBALVARS LogTime.process))
(if (OR (NOT (PROCESSP LogTime.process))
  (PROCESS.FINISHEDP LogTime.process))
  then (if (NOT noninteractivep)
    then (LogTime.Message "not running"))
  elseif (AND abortflag (NOT noninteractivep)
    (NOT (MOUSECONFIRM "Abort Time Logging (current data lost!)")))
  then (LogTime.Message "abort canceled")
  else (replace (LogTimeData status) of (PROCESSPROP LogTime.process 'LogTimeData)
    with (if abortflag
      then 'Abort
      elseif noninteractivep
      then 'Quit!
      else 'Quit))
  (WAKE.PROCESS LogTime.process])

```

(LogTime.Report

```

[LAMBDA NIL (* Koomen "21-Sep-89 09:55")
(DECLARE (GLOBALVARS LogTime.process))
(if (OR (NOT (PROCESSP LogTime.process))
  (PROCESS.FINISHEDP LogTime.process))
  then (ADD.PROCESS ' (LOGTIME.REPORT))
  else (replace (LogTimeData status) of (PROCESSPROP LogTime.process 'LogTimeData) with 'Report)
  (WAKE.PROCESS LogTime.process])

```

(LogTime.ReportTime

```

[LAMBDA (report secs totalsecs) (* Koomen "18-Sep-89 13:19")
(PROG (mins hrs)
  (SETQ mins (QUOTIENT (PLUS secs 30)
    60))
  (SETQ hrs (QUOTIENT mins 60))
  (add mins (TIMES hrs -60))

```

```
(printout report .I3 hrs ":" (QUOTIENT mins 10)
 (REMAINDER mins 10))
(if totalsecs
 then (printout report .I6 (TIMES (FQUOTIENT secs totalsecs)
 100)
 " %%"
 (QUOTIENT (PLUS (TIMES secs 100)
 (QUOTIENT (ADD1 totalsecs)
 2))
 totalsecs))
 (TERPRI report])
```

(LogTime.Start

(* Koomen "24-Oct-89 15:39")

```
[LAMBDA (datafile noninteractivep)
 (DECLARE (GLOBALVARS LogTime.process))
 (if (OR (NOT (PROCESSP LogTime.process))
 (PROCESS.FINISHEDP LogTime.process))
 then (SETQ LogTime.process (ADD.PROCESS (LIST (FUNCTION LogTime.Proc)
 (KWOTE datafile))
 'NAME
 'Time% Logger
 'RESTARTABLE T))
 elseif [NEQ (INFILEP (OR datafile LOGTIME.DATFILE))
 (fetch (LogTimeData logfile) of (PROCESSPROP LogTime.process 'LogTimeData)
 then [LogTime.Message "Can't log time on " (OR datafile LOGTIME.DATFILE)
 " while already logging time on "
 (fetch (LogTimeData logfile) of (PROCESSPROP LogTime.process 'LogTimeData)
 elseif (NOT noninteractivep)
 then (WAKE.PROCESS LogTime.process)]
```

(LogTime.Update

(* Koomen "27-Oct-89 12:41")

```
[LAMBDA (logtimedata noninteractivep)
 (WITH.MONITOR (fetch (LogTimeData lock) of logtimedata)
 [PROG (entry activity lasttime thistime)
 (SETQ entry (fetch (LogTimeData lastEntry) of logtimedata))
 [SETQ activity (if noninteractivep
 then (CAR entry)
 else (while \IDLING do (BLOCK))
 (LogTime.Prompt (CAR entry)
 (SETQ lasttime (fetch (LogTimeData lastUpdate) of logtimedata))
 (SETQ thistime (LogTime.IDate))
 (UNINTERRUPTABLY
 (replace (LogTimeData lastUpdate) of logtimedata with thistime)
 [if (NULL activity)
 then ;; Ignore last interval, and continue (got here through ^E under prompt)
 (replace (LogTimeData lastEntry) of logtimedata with NIL)
 elseif (EQUAL activity (CAR entry))
 then ;; Extend the previous interval
 (RPLACD (CADR entry)
 thistime)
 elseif (SETQ entry (SASSOC activity (fetch (LogTimeData entries) of logtimedata)))
 then ;; Add a new interval to exiting entry
 (replace (LogTimeData lastEntry) of logtimedata with entry)
 (push (CDR entry)
 (CONS lasttime thistime))
 else ;; Add a new entry
 (replace (LogTimeData activityMenu) of logtimedata with NIL)
 (push (fetch (LogTimeData entries) of logtimedata)
 (replace (LogTimeData lastEntry) of logtimedata with (LIST activity (CONS lasttime
 thistime)]
 ]]])
 ]])
```

```
)
(DECLARE%: EVAL@COMPILE
( RECORD LogTimeData (logfile lock status firstUpdate lastUpdate lastEntry activityMenu . entries))
)
(RPAQ? LogTime.process NIL)
(RPAQ? LogTime.promptWindow NIL)
(RPAQ? LogTime.promptWindowRegion NIL)
(RPAQ? LogTime.suspendedLogfile NIL)
(ADDTOVAR AROUNDEXITFNS LogTime.AroundExitFn)
(ADDTOVAR BackgroundMenuCommands
 ["Log Time" (LogTime.Start)
```

```
"Keep track of how time is spent"  
(SUBITEMS ("Update" (LogTime.Start)  
           "Start or update log")  
 ("Edit" (LogTime.Edit)  
          "Edit current data")  
 ("Report" (LogTime.Report)  
            "Generate report on LOGTIME.REPORTFILE")  
 ("Quit" (LogTime.Quit)  
          "Quit keeping track of how time is spent and update log file"  
 (SUBITEMS ("Abort" (LogTime.Quit T)  
            "Quit keeping track of how time is spent but DON'T update log  
            file"])
```

(RPAQQ **BackgroundMenu** NIL)

(DECLARE%: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVERS

(ADDTOVAR **NLAMA**)

(ADDTOVAR **NLAML**)

(ADDTOVAR **LAMA** LogTime.Message)
)

(PUTPROPS **LOGTIME COPYRIGHT** ("Johannes A. G. M. Koomen" 1989))

FUNCTION INDEX

LogTime.AroundExitFn4	LogTime.GDate5	LogTime.Prompt6	LOGTIME.START3
LogTime.ButtonFn4	LogTime.IDate5	LogTime.Quit6	LogTime.Start7
LogTime.Dump4	LogTime.Load5	LOGTIME.REPORT2	LOGTIME.STOP3
LOGTIME.EDIT1	LogTime.Message5	LogTime.Report6	LOGTIME.UPDATE4
LogTime.Edit4	LogTime.Proc5	LogTime.ReportTime6	LogTime.Update7

VARIABLE INDEX

AROUNDEXITFNS7	LOGTIME.INTERVAL4	LogTime.promptWindowRegion7
BackgroundMenu8	LogTime.process7	LOGTIME.REPORTFILE4
BackgroundMenuCommands7	LOGTIME.PROMPT.URGENCY4	LogTime.suspendedLogfile7
LOGTIME.DATAFILE4	LogTime.promptWindow7	

RECORD INDEX

LogTimeData7

PROPERTY INDEX

LOGTIME4
