==========
(VAR) pwk
Information and General setting.

•ASSEMBLY: GPT3-EMB-01.pka
•VERSION: 1.0
•_BUILD_: 1271
•COPYRIGHT: (C)2023 XPLAB s.a.s. - Research in Automation - Brescia - Italy
•Company:
•ProjectName:
•ProjectId:

(VAR) Editor
Changing these settings change the editor behaviour.

•itemViewSize: default (default | BIG)
•itemViewCode: SHOW (default(SHOW) | HIDE)
•itemViewSplit: default (default(horizontal) | VERTICAL)
•StorageDirectory: (Where to store a copy of the assembly)

(VAR) Executor
Here you can set the link between the command line and the EXO, use Px to map command line parameter in EXO parameter.

•_OS_: (Set run time by executor (WIN,WIN-IOT))
•_ARC_: (Architecture set runtime by executor (X86,ARM))
•_BASEBOARD_: (IOT:product,Manufacturer,version,serialnumber or winboard)
•_PROG_: (program name set runtime by executor)
•STARTLOGO: YES (NO,YES)
•CONS: (YES,PROG(cns_cmd))
•CONS_OS: (NO,YES Os console)
•CONS_OS_PARENT: TERMINATE (CONTINUE,SUSPEND,TERMINATE)
•KEEPOPEN: (YES if Consolle should be close manualy)
•LIMIT_ONCE: YES (if YES allow only one instance to run)
•LIMIT_KEY: GTP-TPL-01 (Key used by LIMIT_ONCE you can prepend Global\ or Local\ for scope visibility)
•PASSWORD: (The password needed to open this assembly)
•PASSWORD_ENB: (NO,YES)
•ERR_MAIL_ENB: (TRUE, FALSE (default FALSE)
•ERR_MAIL_HOST: (SMTP Mail server address, for sending crash report)
•ERR_MAIL_IAM: (The sending Host (I am))
•ERR_MAIL_FROM: (Pseudo email address of this application)
•ERR_MAIL_TO: (Destination email)
•ERR_DUMP: TRUE (TRUE, FALSE (default TRUE))
•ON_ERR: (RESTART)
•EXO: Main (Program entry point)
•PARLIS: (Main Attribute for Command Line Parameter LIS PTR)
•P1: (Main Attribute for First command line parameter)
•P2: (Main Attribute for Second command line parameter)

==========
(GUI) MG

•_AUTOLOAD_: ON (Values: ON,OFF to disable)

•_ADDR_: (Values: gui IP address empty=default)
•_PORT_: (Values: gui IP PORT address empty=default)
•_FILE_: MG (Name of the UserInterface resources)
•_TRIG_: trig (Trigger EXO or MTHD)
•_PTR_: (Pointer to open gui)
•_EVT_: (System Event)
•_SIGNAL_: (User Event)
•_PTH_ITEM_: (Path and Name of ITEM generating a mouse event)
•_ITEM_: (Name of ITEM generating a mouse event)
•_PTH_ITEM_ACT_: (Path and Name of active ITEM)
•_ITEM_ACT_: (Name of active ITEM)
•_X_: (X relative to Form)
•_Y_: (Y relative to Form)
•_XW_: (X relative to Screen)
•_YW_: (X relative to Screen)
•_XC_: (X relative to Control)
•_YC_: (X relative to Control)
•_BL_: (Button Left)
•_BR_: (Button Right)
•_ON_: (If mouse is On the form)
•_KEY_FLG_: (VLD(b1) ALT(b2) CTRL(b3) SHT(b4) CAPS LOCK(b5) NUM LOCK(b6) SCROLL LOCK(b7))
•_KEY_CODE_: (The Key CODE)
•_KEY_KEY_: (The Key pressed)
•_KEY_VAL_: (The Key value)
•_GUI_: (GUI name)
•caller:
•runFlg: (Core is in execution)
•nId: (Id Number)
•evtL: (Event List)
•evtMx: 10 (Max EVT n° in LIS)
•evtTmMx: 250 (Max time for EVT)
•VoiceSts:
•VOICEcmd:
•LSTPROMPT:
•TVCT:
•SQL:

==========

(MTHD) trig

(SWITCH) _SIGNAL_

(CASE) _DEFAULT_

(IF) if

SIGNAL is valid

•_COND_:

.....

COND= ~_SIGNAL_;

.....

(THEN) then

.....

#IF(runFlg==1);

#IF(LIS_NUM(evtL)<evtMx);

```
            LIS_ADD(evtL,LIS_NEW(_SIGNAL_,_PTH_ITEM_,£));
            #END;
            #END;
            ..............................
(IF) if
 _SIGNAL_ -> skip events
-------------------------------------------------
•_COND_:
-------------------------------------------------
   ..............................
   _COND_= ~_SIGNAL_;
   ..............................
   (THEN) then
      (BREAK) break
      -------------------------------------------------
      •_LEVEL_:   (Type of ITEM that break)
      -------------------------------------------------
(SWITCH) _EVT_
-------------------------------------------------
•C_CLOSE: WIND_CLS
•C_WINDOW_MOVE: WIND_MOV
•C_KEY_UP: KEY_UP
-------------------------------------------------
   (CASE) _DEFAULT_
   (CASE) C_CLOSE
      ..............................
      #IF(runFlg==1);
      #IF(LIS_NUM(evtL)<evtMx);
      LIS_ADD(evtL,LIS_NEW(£CLOSE,£,£));
      #END;
      #END;
      ..............................
   (CASE) C_WINDOW_MOVE
      ..............................
      #IF(runFlg==1);
      #IF(LIS_NUM(evtL)<evtMx);
      LIS_ADD(evtL,LIS_NEW(£winMov,£,£));
      #END;
      #END;
      ..............................
   (CASE) C_KEY_UP
      (IF) if
       ENTER
      -------------------------------------------------
      •_COND_:
      -------------------------------------------------
         ..............................
         _COND_= _KEY_CODE_ == 13 and _ITEM_=="QRY" ;
         ..............................
         (THEN) then
            ..............................
            #if(VALUE@\MG\pag\ConWret==1);
            LIS_ADD(evtL,LIS_NEW(£DO,_PTH_ITEM_,_ITEM_));
            #end;
            ..............................
(BREAK) break
-------------------------------------------------
•_LEVEL_:   (Type of ITEM that break)
```

```
   ------------------------------------------------
=======================================
(MTHD) Manager
------------------------------------------------
•action:
•opt1:
•opt2:
------------------------------------------------
   (SWITCH) action
   ------------------------------------------------
   •C_SHOW: SHOW
   •C_CLOSE: CLOSE
   •C_onLoop: onLoop
   •C_WINDOW_MOVE: winMov
   •C_DO: DO
   •C_VOICE: VOICE
   •C_REDO: REDO
   •C_CRED: CRED
   •C_CLR: CLR
   •C_PDFLOAD: PDFload
   •C_CONFWRET: ConfWret
   •C_PAGHELP: PAgHELP
   •C_RMCLOS: RMclos
   •C_CLRPFX: ClrPfx
   ------------------------------------------------
      (CASE) _DEFAULT_
         ..............................
         chatput("[ERR] Unrecognized signal"..action.."(",dbgline,")");
         ..............................
      (CASE) C_SHOW
         ..............................
         GUI_SND(_PTR_,£SHOW);
         FOCUS@\MG\pag\QRY=£YES;

         °tb=MODEL@\Main;
         °in=tbl_inf(°tb,£ROW);
         °i=0;
         °it=£;
         #while(°i+=1 <=°in);

         °e=Tbl_itm(°tb,£id,°i);
         #if(SRCH(°e,"gpt",NULL,0 ) or °e=="text-davinci-003");

         °it=°it++if(~°it,",",£)++°e;
         VALUE@\MG\pag\pSETTING\MODEL=°e;
         #end;

         #end;

         ITEMS@\MG\pag\pSETTING\MODEL=°it;

         TVCT=TVCT@\main;
         sql=tbl_ITM(TVCT,£sql,1);
         EXO("PDF2GRD");

         Visible@\MG\pag\READme=£true;
         °t1=PKG_PTH++PKG_MNFGET(NULL,£DOC,£READme);
```

```
   RTF@\MG\pag\READme\READme=FS_FRDS_F(°t1);
   ..............................
(CASE) C_CLOSE
   ..............................
   runFlg=0;
   ..............................
(CASE) C_onLoop
(CASE) C_WINDOW_MOVE
-------------------------------------------------
•eX:
•eY:
•eSizX:
•eSizY:
•eWSizX:
•eWSizY:
-------------------------------------------------
   default disabled
   (BREAK) break
   -----------------------------------------------
   •_LEVEL_:   (Type of ITEM that break)
   -----------------------------------------------
   ..............................
   !!! Wait for BUTTON release ;
   °t3=TMR+1500;
   #WHILE(_BL_==1 AND TMR<=°t3 );
   SLEEP(100);
   #END;

   !!! Get Desktop size ;
   °t2= NULL;
   °t1= "EXEC:_GUI_INFO_;VDUSIZ;"++$°t2;
   °rf1="_PTR_@\"++_GUI_;
   GUI_SND(REF(°rf1),°t1);

   !!! Wait response ;
   °t1 = WAITCND($°t2,£NOTNULL,5000);

   !!! Decode Desktop size ;
   °lis1=CSV(°t2,";");

   °t1=LIS_POS(°lis1,1);
   eWSizX=SPLT(°t1,":",£RIGHT);

   °t1=LIS_POS(°lis1,2);
   eWSizY=SPLT(°t1,":",£RIGHT);

   TRASH(°lis1);


   eSizX=REF("SizX@\"++_GUI_++"\pag");
   eSizY=REF("SizY@\"++_GUI_++"\pag");

   eX=£;
   eY=£;

   #WHILE(1);
   °rf1="X@\"++_GUI_++"\pag";
   °t1=REF(°rf1);
```

```
   #IF(eX!=°t1);!!! X changed ? ;
   eX=°t1;

   #IF(eX<0);
   REF(°rf1)=0;
   #ELSE;
   °t1=eWSizX-eSizX;
   #IF(eX>°t1);
   REF(°rf1)=°t1;
   #END;
   #END;
   #END;

   °rf1="Y@\"++_GUI_++"\pag";
   °t1=REF(°rf1);
   #IF(eY!=°t1);!!! Y changed ? ;
   eY=°t1;

   #IF(eY<0);
   REF(°rf1)=0;
   #ELSE;
   °t1=eWSizY-eSizY;
   #IF(eY>°t1);
   REF(°rf1)=°t1;
   #END;
   #END;
   #END;

   #IF(_BL_!=1);!!! Mouse released ;
   #BREAK;
   #END;

   SLEEP(250);
   #END;
   ..............................
   (EXEC) evt_clear
(CASE) C_DO
   (IF) if
   ------------------------------------------------
   •_COND_:
   ------------------------------------------------
      ..............................
      _COND_= ~VALUE@\MG\pag\QRY==0;
      ..............................
      (THEN) then
         (BREAK) break
         ------------------------------------------------
         •_LEVEL_:   (Type of ITEM that break)
         ------------------------------------------------
   ..............................
   Visible@\MG\pag\INPROGRESS=£TRUE;
   TEXT@\MG\pag\HTTP=£;
   LSTPROMPT=VALUE@\MG\pag\QRY;

   GETALL@\MG\pag\pPDF\gridPanel\DOC=£YES;
   WAITCND($GETALL@\MG\pag\pPDF\gridPanel\DOC,£EMPTY,300);

   °slct=£;
   #if(~ALL@\MG\pag\pPDF\gridPanel\DOC);
```

```
°tb=TBL_NEW(ALL@\MG\pag\pPDF\gridPanel\DOC,£ROW,";","|");
°in=tbl_inf(°tb,£ROW);
°i=0;
#while(°i+=1 <=°in);
#if(tbl_itm(°tb,1,°i));
°slct=°slct++if(~°slct,",",£)++tbl_itm(°tb,2,°i);
#end;
#end;
#end;
.............................
(EXEC) \AI_CORE\QRY
   (SET) set
      .............................
      QRY=LSTPROMPT;
      MAXT=VALUE@\MG\pag\pSETTING\MAXT;
      MODEL=VALUE@\MG\pag\pSETTING\MODEL;
      TYPE=if(VALUE@\MG\pag\pSETTING\pTYPE\CHAT==1, £CHAT,£CPLT);
      PFX=VALUE@\MG\pag\PFX\PFX;
      SLCT=°slct;
      .............................
   (GET) get
      .............................
      °rply=RPLY;
      °res=RES;
      °inf=INF;
      .............................
.............................
°cr=char(13);
#if(°res==£OK);

SEL_START@\MG\pag\REPLY=1;
SEL_CLRFRG@\MG\pag\REPLY="230;230;230";
SEL_FNT_SIZ@\MG\pag\REPLY=12;
SEL_FNT_STYLE@\MG\pag\REPLY=£BOLD;

SEL_TXT@\MG\pag\REPLY=°cr++"USER:"++°cr;

SEL_CLRFRG@\MG\pag\REPLY="220;220;240";
SEL_FNT_STYLE@\MG\pag\REPLY=£;

SEL_TXT@\MG\pag\REPLY=LSTPROMPT++°cr;

!!============================;

SEL_START@\MG\pag\REPLY=1;
SEL_CLRFRG@\MG\pag\REPLY="230;230;230";
SEL_FNT_SIZ@\MG\pag\REPLY=12;
SEL_FNT_STYLE@\MG\pag\REPLY=£BOLD;

SEL_TXT@\MG\pag\REPLY=°cr++"GPTx:"++°cr;

SEL_FNT_SIZ@\MG\pag\REPLY=13;
SEL_CLRFRG@\MG\pag\REPLY="240;240;240";
SEL_FNT_STYLE@\MG\pag\REPLY=£ITALIC;

SEL_TXT@\MG\pag\REPLY=°rply++°cr;

#if(VOICEcmd);
```

```
   TEXT@\MG\pag\AUDIO_name\VOICE=°rply;
   #end;

   VALUE@\MG\pag\QRY=£;

   #end;

   Visible@\MG\pag\INPROGRESS=£FALSE;
   TEXT@\MG\pag\HTTP=°res.."["++°inf++"]";
   ..............................
(CASE) C_VOICE
   ..............................
   #if(VOICEcmd==1);
   TEXT@\MG\pag\VOICE="Audio ON";
   VOICEcmd=0;
   STOP@\MG\pag\AUDIO_name\VOICE=1;
   #else;
   TEXT@\MG\pag\VOICE="Audio OFF";
   VOICEcmd=1;
   #end;
   ..............................
(CASE) C_REDO
   ..............................
   VALUE@\MG\pag\QRY= LSTPROMPT=VALUE@\MG\pag\QRY;
   ..............................
(CASE) C_CRED
   ..............................
   Visible@\MG\pag=£FALSE;
   ..............................
   (EXEC) \CRD\Starter
      (SET) set
         ..............................
         mode=£EXEC;   !!EXEC,THREAD;
         par_nId=£;
         par_Gui=£;
         ..............................
   ..............................
   Visible@\MG\pag=£True;
   ..............................
(CASE) C_CLR
   ..............................
   VALUE@\MG\pag\REPLY=£;
   tbl_chg(tblCht@\AI_CORE,NULL,0);
   ..............................
(CASE) C_PDFLOAD
   ..............................
   °pdfEnum=GUI_DLGOFD(_PTR_,£TRUE,NULL,".pdf",NULL,"PDF|*.pdf|TXT|*.txt|
   CSV|*.csv",NULL,NULL,"Select your PFD document");
   ..............................
   ..............................
   #if(~°pdfEnum);
   EXOTHR("PDFLOAD", PDFENUM::°pdfEnum);
   #end;
   ..............................
(CASE) C_CONFWRET
   ..............................
   #if(VALUE@\MG\pag\ConWret==1);
   Visible@\MG\pag\DO=£FALSE;
   CONFWRET@\MG\pag\QRY=£TRUE;
```

```
         #else;
         Visible@\MG\pag\DO=£TRUE;
         CONFWRET@\MG\pag\QRY=£FALSE;
         #end;
         ..............................
      (CASE) C_PAGHELP
         ..............................
         Visible@\MG\pag\READme=£TRUE;
         ..............................
      (CASE) C_RMCLOS
         ..............................
         Visible@\MG\pag\READme=£FALSE;
         ..............................
      (CASE) C_CLRPFX
         ..............................
         VALUE@\MG\pag\PFX\PFX=£;
         ..............................
   (BREAK) break
   ------------------------------------------------
   •_LEVEL_:   (Type of ITEM that break)
   ------------------------------------------------
   <->
========================================
(MTHD) Core
------------------------------------------------
•t1:
•locRes:
•evtIni:
•lis1:
------------------------------------------------
   ..............................
   runFlg=1;

   evtL=LIS_NEW();
   VOICEcmd=0;
   TEXT@\MG\pag\VOICE="Audio ON";
   ..............................
   (EXEC) Manager
      (SET) set
         ..............................
         action=£SHOW;
         ..............................
   (WHILE) while
    Main Loop
   ------------------------------------------------
   •_COND_:
   ------------------------------------------------
      ..............................
      _COND_= runFlg;
      ..............................
      (DO) do
         (EXEC) Manager
         ------------------------------------------------
         •_PAR_: ONCE  (SYNC ONCE ONCE_FOR_CALLER)
         •_RSLT_:   (For ONCExx the name of the symbol where to put the result
         -1=failToStart 0=starting 1=start. ONCE_FOR_CALLER is syncronous)
         ------------------------------------------------
            (SET) set
```

```
      ..............................
      action=£onLoop;
      opt1=£;
      opt2=£;
      ..............................
(WHILE) while
 Loop EVT
-------------------------------------------------
•_COND_:
-------------------------------------------------
   (PRE) pre
      ..............................
      evtIni=TMR;
      t1=evtIni+evtTmMx;
      ..............................
   ..............................
   _COND_= TMR<=t1;
   ..............................
   (DO) do
      ..............................
      lis1=LIS_POP(evtL);
      ..............................
      (IF) if
       Event to manage ?
      -------------------------------------------------
      •_COND_:
      -------------------------------------------------
         ..............................
         _COND_= lis1>0;
         ..............................
         (THEN) then
            (EXEC) Manager
               (SET) set
                  ..............................
                  action=LIS_POS(lis1,1);
                  opt1=LIS_POS(lis1,2);
                  opt2=LIS_POS(lis1,3);
                  ..............................
            ..............................
            TRASH(lis1);
            ..............................
         (ELSE) else
            ..............................
            SLEEP(25);
            ..............................
(LBL) GUI_Alive
   ..............................
   locRes=GUI_ALV(_PTR_);
   ..............................
   (IF) if
    Lost alive
   -------------------------------------------------
   •_COND_:
   -------------------------------------------------
      ..............................
      _COND_= locRes!=1;
      ..............................
      (THEN) then
         (GOTO) Exit
```

```
   ---
   (LBL) Exit
      ..............................
      GUI_SND(_PTR_,£HIDE);
      runFlg=0;

      #IF(evtL>0);
      TRASH(LIS_USE(evtL));!!! Trash of pointer inside evtL ;
      TRASH(evtL);!!! Trash of evtL ;
      #END;
      ..............................
=======================================
(MTHD) Starter
-----------------------------------------------
•mode:   (£EXEC,£THREAD : EXEC wait until the page is closed, THREAD launch the
page as indipendent)
•par_nId:   (For remotable page: it is the number of connection, conNId in the
man_usr method of ES (enterprise server) block. Otherwise it is unused (write £
or so))
•par_Gui:   (Name of the caller gui, for advanced uses. You can not specify it
(£) if the page is not remotable.)
•locRes:
-----------------------------------------------
   (LBL) GUI_Dup
      (IF) if
       Local?
      -------------------------------------------------
      •_COND_:
      -------------------------------------------------
         ..............................
         _COND_= _AUTOLOAD_==£ON;
         ..............................
         (THEN) then
            (CALL) Init_Var
            (GOTO) End
      (EXEC) \ULib\GUI\Gui_Fnc
         (SET) set
            ..............................
            action=£Prepare;   !!SetPos, Dup, Destroy, Prepare, Show;
            Opt=par_nId;   !!;
            RefPg=_FILE_;   !!Reference no @\;
            distX=£;   !!Distance X from border;
            distY=£;   !!Distance Y from border;
            ..............................
         (GET) get
            ..............................
            locRes=res;   !!Risultato;
            ..............................
      (IF) if
       ERR?
      -------------------------------------------------
      •_COND_:
      -------------------------------------------------
         ..............................
         _COND_= locRes==£ERR;
         ..............................
         (THEN) then
            ..............................
```

```
         chatput("[ERR] Error in page preparation (",dbgline,")");
         ..............................
         (BREAK) break
         ------------------------------------------------
         •_LEVEL_:   (Type of ITEM that break)
         ------------------------------------------------
   (IF) if
    LOAD?
   ------------------------------------------------
   •_COND_:
   ------------------------------------------------
      ..............................
      _COND_= locRes==£LOAD;
      ..............................
      (THEN) then
         (CALL) Init_Var
         (CALL) translation
   (LBL) End
(IF) if
 Already running
------------------------------------------------
•_COND_:
------------------------------------------------
   ..............................
   _COND_= runFlg@\MG==1;
   ..............................
   (THEN) then
      ..............................
      GUI_SND(_PTR_@\MG,£SHOW);
      ..............................
      (BREAK) break
      ------------------------------------------------
      •_LEVEL_:   (Type of ITEM that break)
      ------------------------------------------------
(IF) if
 Run as THREAD or EXEC
------------------------------------------------
•_COND_:
------------------------------------------------
   ..............................
   _COND_= mode==£THREAD;
   ..............................
   (THEN) then
      (THREAD) \MG\Core
      ------------------------------------------------
      •_PAR_: ONCE  (SYNC ONCE ONCE_FOR_CALLER)
      •_RSLT_:   (For ONCExx the name of the symbol where to put the result
      -1=failToStart 0=starting 1=start. ONCE_FOR_CALLER is syncronous)
      ------------------------------------------------
   (ELSE) else
      (EXEC) \MG\Core
(BREAK) break
------------------------------------------------
•_LEVEL_:   (Type of ITEM that break)
------------------------------------------------
<->
(BLK) Init_Var
   ..............................
```

```
      caller@\MG=par_Gui;
      nId@\MG=par_nID;
      ..............................
   (BLK) translation
      ..............................
      #WHILE(1);
      °res=£SKIP;

      #IF(NOT ISNUM(par_nId));
      #BREAK;
      #END;

      #IF(NOT EXIST("\DATA\TRSL"));
      #BREAK;
      #END;

      °res=£OK;
      #BREAK;
      #END;
      ..............................
      (IF) if
       OK ?
      ------------------------------------------------
      •_COND_:
      ------------------------------------------------
         ..............................
         _COND_= °res==£SKIP;
         ..............................
         (THEN) then
            (BREAK) break
            ------------------------------------------------
            •_LEVEL_:   (Type of ITEM that break)
            ------------------------------------------------
      (EXEC) \DATA\TRSL\gui_lod
         (SET) set
            ..............................
            GUI=_GUI_@\MG;
            nId=par_nId;
            ..............................
========================================
(MTHD) evt_clear
 Reset the EVT list
------------------------------------------------
•lis1:
------------------------------------------------
   ..............................
   #WHILE(1);
   lis1=LIS_POP(evtL);
   #IF(lis1<=0);!!! Invalid PTR -> end of list;
   #BREAK;
   #END;

   TRASH(lis1);
   #END;
   ..............................
========================================
(MTHD) PDFLOAD
------------------------------------------------
•pdfenum:
```

```
•pdf:
-------------------------------------------------
   (WHILE) while
   -----------------------------------------------
   •_COND_:
   -----------------------------------------------
      (PRE) pre
         ..............................
         °pl=TKNZ(pdfenum,";");
         ..............................
      ..............................
      pdf=lis_pop(°pl);
      _COND_= ~PDF;
      ..............................
      (DO) do
         ..............................
         °pdf=SPLT(SPLT(pdf,".pdf",£LEFTORALL,£REV,0),"\",£RIGHTORALL,£REV);
         °qr=DB_QRY(sql,"Select count(*) FROM doc WHERE slct="++quose(°pdf));
         °ex=DB_GET(°qr,£FLD,1,1);
         trash(°qr);
         ..............................
         (IF) if
          Already In
         -----------------------------------------------
         •_COND_:
         -----------------------------------------------
            ..............................
            _COND_= °ex;
            ..............................
            (THEN) then
               ..............................
               °r=GUI_DLGMSG(_PTR_,"File already in Data","Do you want to
               reload"..pdf,£YES_NO,£EXCLAMATION);
               ..............................
               (IF) if
               -----------------------------------------------
               •_COND_:
               -----------------------------------------------
                  ..............................
                  _COND_= °r!=£YES;
                  ..............................
                  (THEN) then
                     (BREAK) break
                     -----------------------------------------------
                     •_LEVEL_:   (Type of ITEM that break)
                     -----------------------------------------------
                  (ELSE) else
                     ..............................
                     °qry="SELECT id FROM dat WHERE slct="++quose(°pdf);
                     °qr=DB_QRY(sql,°qry);
                     #if(°qr);

                     °tb=DB_GET(°qr,£TBL);

                     °in=TBL_INF(°tb,£ROW);
                     °i=0;

                     °MNG=TBL_ITM(TVCT,£MNG,1);
                     #while(°i+=1 <=°in);
```

```
                        AIT_VCT_DEL(°MNg,TBL_ITM(°tb,£ID,°i));
                        #end;
                        TRASH(°qr,°tb);

                        °qry="DELETE FROM dat WHERE slct="++quose(°pdf);
                        °qr=DB_QRY(sql,°qry);
                        trash(°qr);

                        °qry="DELETE FROM doc WHERE slct="++quose(°pdf);
                        °qr=DB_QRY(sql,°qry);
                        trash(°qr);


                        #end;
                        ..............................
              ..............................
              Visible@\MG\pag\pPDF\LOAD=£TRUE;

              TEXT@\MG\pag\pPDF\LOAD\name=pdf;

              EXO("\AI_DB\GPTLoad",TVCT::TVCT,OBJ::pdf);
              Visible@\MG\pag\pPDF\LOAD=£FALSE;

              EXO("PDF2GRD");
              ..............................
          (DONXT) doNxt
       ..............................
       trash(°pl);
       ..............................
   ========================================
   (MTHD) PDF2GRD
       ..............................
       °dr=DB_QRY(sql,"SELECT slct FROM doc");
       °tb=DB_GET(°dr,£TBL);
       °t=TBL_EXP(°tb,";",";");
       rows@\MG\pag\pPDF\gridPanel\DOC=tbl_inf(°tb,£ROW);
       ColSetNum@\MG\pag\pPDF\gridPanel\DOC=2;
       ColSetVal@\MG\pag\pPDF\gridPanel\DOC=°t;

       trash(°dr,°tb);
       ..............................
   ========================================
================================================================================
(GUI) CRD
------------------------------------------------
•_AUTOLOAD_: ON  (Values: ON,OFF to disable)
•_ADDR_:   (Values: gui IP address empty=default)
•_PORT_:   (Values: gui IP PORT address empty=default)
•_FILE_: CR  (Name of the UserInterface resources)
•_TRIG_: trig  (Trigger EXO or MTHD)
•_PTR_:   (Pointer to open gui)
•_EVT_:   (System Event)
•_SIGNAL_:   (User Event)
•_PTH_ITEM_:   (Path and Name of ITEM generating a mouse event)
•_ITEM_:   (Name of ITEM generating a mouse event)
•_PTH_ITEM_ACT_:   (Path and Name of active ITEM)
•_ITEM_ACT_:   (Name of active ITEM)
```

•_X_: (X relative to Form)
•_Y_: (Y relative to Form)
•_XW_: (X relative to Screen)
•_YW_: (X relative to Screen)
•_XC_: (X relative to Control)
•_YC_: (X relative to Control)
•_BL_: (Button Left)
•_BR_: (Button Right)
•_ON_: (If mouse is On the form)
•_KEY_FLG_: (VLD(b1) ALT(b2) CTRL(b3) SHT(b4) CAPS LOCK(b5) NUM LOCK(b6) SCROLL LOCK(b7))
•_KEY_CODE_: (The Key CODE)
•_KEY_KEY_: (The Key pressed)
•_KEY_VAL_: (The Key value)
•_GUI_: (GUI name)
•caller:
•runFlg: (Core is in execution)
•nId: (Id Number)
•evtL: (Event List)
•evtMx: 10 (Max EVT n° in LIS)
•evtTmMx: 250 (Max time for EVT)

==========

(MTHD) trig

(SWITCH) _SIGNAL_

(CASE) _DEFAULT_

(IF) if

SIGNAL is valid

•_COND_:

.....

COND= ~_SIGNAL_;

.....

(THEN) then

.....

#IF(runFlg==1);

#IF(LIS_NUM(evtL)<evtMx);

LIS_ADD(evtL,LIS_NEW(_SIGNAL_,_PTH_ITEM_,£));

#END;

#END;

.....

(IF) if

SIGNAL -> skip events

•_COND_:

.....

COND= ~_SIGNAL_;

.....

(THEN) then

(BREAK) break

•_LEVEL_: (Type of ITEM that break)

(SWITCH) _EVT_

```
   ------------------------------------------------
   •C_CLOSE: WIND_CLS
   •C_WINDOW_MOVE: WIND_MOV
   ------------------------------------------------
      (CASE) _DEFAULT_
      (CASE) C_CLOSE
         ..............................
         #IF(runFlg==1);
         #IF(LIS_NUM(evtL)<evtMx);
         LIS_ADD(evtL,LIS_NEW(£CLOSE,£,£));
         #END;
         #END;
         ..............................
      (CASE) C_WINDOW_MOVE
         ..............................
         #IF(runFlg==1);
         #IF(LIS_NUM(evtL)<evtMx);
         LIS_ADD(evtL,LIS_NEW(£winMov,£,£));
         #END;
         #END;
         ..............................
   (BREAK) break
   ------------------------------------------------
   •_LEVEL_:   (Type of ITEM that break)
   ------------------------------------------------
========================================
(MTHD) Manager
------------------------------------------------
•action:
•opt1:
•opt2:
------------------------------------------------
   (SWITCH) action
   ------------------------------------------------
   •C_SHOW: SHOW
   •C_CLOSE: CLOSE
   •C_onLoop: onLoop
   •C_WINDOW_MOVE: winMov
   •C_SAVE: SAVE
   •C_OAI: OAI
   ------------------------------------------------
      (CASE) _DEFAULT_
         ..............................
         chatput("[ERR] Unrecognized signal"..action.."(",dbgline,")");
         ..............................
      (CASE) C_SHOW
         ..............................
         GUI_SND(_PTR_,£SHOW);

         TEXT@\CRD\pag\info="To use this program you need a key that you can get
         from OpenAi site (see button below)";
         ..............................
      (CASE) C_CLOSE
         (IF) if
         ------------------------------------------------
         •_COND_:
         ------------------------------------------------
```

```
      ..............................
      _COND_= ~EPT@\Main==0 or ~KEY@\Main==0;
      ..............................
      (THEN) then
         ..............................
         °r=GUI_DLGMSG(_PTR_,"INVALID CREDENTIAL","Credentiial are
         invalid: Do you wont to close the program?" ,£OK_CANC,£QUESTION);
         ..............................
         (IF) if
         -------------------------------------------------
         •_COND_:
         -------------------------------------------------
            ..............................
            _COND_= °r == £OK;
            ..............................
            (THEN) then
               (END) end
            (ELSE) else
               (BREAK) break
               -------------------------------------------------
               •_LEVEL_:   (Type of ITEM that break)
               -------------------------------------------------
   ..............................
   runFlg=0;
   ..............................
(CASE) C_onLoop
   ..............................
   #if(~VALUE@\CRD\pag\KEY==0 or ~VALUE@\CRD\pag\EPT==0);
   Visible@\CRD\pag\SAVE=£FALSE;
   #else;
   Visible@\CRD\pag\SAVE=£true;
   #end;
   ..............................
(CASE) C_WINDOW_MOVE
-------------------------------------------------
•eX:
•eY:
•eSizX:
•eSizY:
•eWSizX:
•eWSizY:
-------------------------------------------------
   default disabled
   (BREAK) break
   -------------------------------------------------
   •_LEVEL_:   (Type of ITEM that break)
   -------------------------------------------------
   ..............................
   !!! Wait for BUTTON release ;
   °t3=TMR+1500;
   #WHILE(_BL_==1 AND TMR<=°t3 );
   SLEEP(100);
   #END;

   !!! Get Desktop size ;
   °t2= NULL;
   °t1= "EXEC:_GUI_INFO_;VDUSIZ;"++$°t2;
   °rf1="_PTR_@\"++_GUI_;
```

```
GUI_SND(REF(°rf1),°t1);

!!! Wait response ;
°t1 = WAITCND($°t2,£NOTNULL,5000);

!!! Decode Desktop size ;
°lis1=CSV(°t2,";");

°t1=LIS_POS(°lis1,1);
eWSizX=SPLT(°t1,":",£RIGHT);

°t1=LIS_POS(°lis1,2);
eWSizY=SPLT(°t1,":",£RIGHT);

TRASH(°lis1);


eSizX=REF("SizX@\"++_GUI_++"\pag");
eSizY=REF("SizY@\"++_GUI_++"\pag");

eX=£;
eY=£;

#WHILE(1);
°rf1="X@\"++_GUI_++"\pag";
°t1=REF(°rf1);
#IF(eX!=°t1);!!! X changed ? ;
eX=°t1;

#IF(eX<0);
REF(°rf1)=0;
#ELSE;
°t1=eWSizX-eSizX;
#IF(eX>°t1);
REF(°rf1)=°t1;
#END;
#END;
#END;

°rf1="Y@\"++_GUI_++"\pag";
°t1=REF(°rf1);
#IF(eY!=°t1);!!! Y changed ? ;
eY=°t1;

#IF(eY<0);
REF(°rf1)=0;
#ELSE;
°t1=eWSizY-eSizY;
#IF(eY>°t1);
REF(°rf1)=°t1;
#END;
#END;
#END;

#IF(_BL_!=1);!!! Mouse released ;
#BREAK;
#END;

SLEEP(250);
```

```
      #END;
      ..............................
      (EXEC) evt_clear
   (CASE) C_SAVE
      ..............................
      EPT@\Main=VALUE@\CRD\pag\EPT;
      KEY@\Main=VALUE@\CRD\pag\KEY;
      ORG@\Main="POWER-KI User";


      KB1_QRY(KBC@\MAin,"\OAI\EPT="++quos(EPT@\Main),£TEXT);
      KB1_QRY(KBC@\MAin,"\OAI\KEY="++quos(KEY@\Main),£TEXT);
      KB1_QRY(KBC@\MAin,"\OAI\ORG)"++quos(ORG@\Main),£TEXT);

      kb1_sav(KBC@\MAin);
      ..............................
   (CASE) C_OAI
      ..............................
      OSshell(NULL,£OPEN,"https://platform.openai.com/signup");
      ..............................
(BREAK) break
-------------------------------------------------
•_LEVEL_:   (Type of ITEM that break)
-------------------------------------------------
<->
=======================================
(MTHD) Core
-------------------------------------------------
•t1:
•locRes:
•evtIni:
•lis1:
-------------------------------------------------
   ..............................
   runFlg=1;

   evtL=LIS_NEW();
   ..............................
   (EXEC) Manager
      (SET) set
         ..............................
         action=£SHOW;
         ..............................
   (WHILE) while
    Main Loop
   -------------------------------------------------
   •_COND_:
   -------------------------------------------------
      ..............................
      _COND_= runFlg;
      ..............................
      (DO) do
         (EXEC) Manager
         -------------------------------------------------
         •_PAR_: ONCE  (SYNC ONCE ONCE_FOR_CALLER)
         •_RSLT_:   (For ONCExx the name of the symbol where to put the result
         -1=failToStart 0=starting 1=start. ONCE_FOR_CALLER is syncronous)
         -------------------------------------------------
```

```
   (SET) set
      ..............................
      action=£onLoop;
      opt1=£;
      opt2=£;
      ..............................
(WHILE) while
 Loop EVT
-------------------------------------------------
•_COND_:
-------------------------------------------------
   (PRE) pre
      ..............................
      evtIni=TMR;
      t1=evtIni+evtTmMx;
      ..............................
   ..............................
   _COND_= TMR<=t1;
   ..............................
   (DO) do
      ..............................
      lis1=LIS_POP(evtL);
      ..............................
      (IF) if
       Event to manage ?
      -----------------------------------------------------
      •_COND_:
      -----------------------------------------------------
         ..............................
         _COND_= lis1>0;
         ..............................
         (THEN) then
            (EXEC) Manager
               (SET) set
                  ..............................
                  action=LIS_POS(lis1,1);
                  opt1=LIS_POS(lis1,2);
                  opt2=LIS_POS(lis1,3);
                  ..............................
            ..............................
            TRASH(lis1);
            ..............................
         (ELSE) else
            ..............................
            SLEEP(25);
            ..............................
(LBL) GUI_Alive
   ..............................
   locRes=GUI_ALV(_PTR_);
   ..............................
   (IF) if
    Lost alive
   -------------------------------------------------
   •_COND_:
   -------------------------------------------------
      ..............................
      _COND_= locRes!=1;
      ..............................
      (THEN) then
```

```
                (GOTO) Exit
   ---
   (LBL) Exit
      ..............................
      GUI_SND(_PTR_,£HIDE);
      runFlg=0;

      #IF(evtL>0);
      TRASH(LIS_USE(evtL));!!! Trash of pointer inside evtL ;
      TRASH(evtL);!!! Trash of evtL ;
      #END;
      ..............................
=======================================
(MTHD) Starter
-------------------------------------------------
•mode:   (£EXEC,£THREAD : EXEC wait until the page is closed, THREAD launch the
page as indipendent)
•par_nId:   (For remotable page: it is the number of connection, conNId in the
man_usr method of ES (enterprise server) block. Otherwise it is unused (write £
or so))
•par_Gui:   (Name of the caller gui, for advanced uses. You can not specify it
(£) if the page is not remotable.)
•locRes:
-------------------------------------------------
   (LBL) GUI_Dup
      (IF) if
       Local?
      ---------------------------------------------------
      •_COND_:
      ---------------------------------------------------
         ..............................
         _COND_= _AUTOLOAD_==£ON;
         ..............................
         (THEN) then
            (CALL) Init_Var
            (GOTO) End
      (EXEC) \ULib\GUI\Gui_Fnc
         (SET) set
            ..............................
            action=£Prepare;   !!SetPos, Dup, Destroy, Prepare, Show;
            Opt=par_nId;   !!;
            RefPg=_FILE_;   !!Reference no @\;
            distX=£;   !!Distance X from border;
            distY=£;   !!Distance Y from border;
            ..............................
         (GET) get
            ..............................
            locRes=res;   !!Risultato;
            ..............................
      (IF) if
       ERR?
      ---------------------------------------------------
      •_COND_:
      ---------------------------------------------------
         ..............................
         _COND_= locRes==£ERR;
         ..............................
         (THEN) then
```

```
         ..............................
         chatput("[ERR] Error in page preparation (",dbgline,")");
         ..............................
         (BREAK) break
         -------------------------------------------------
         •_LEVEL_:   (Type of ITEM that break)
         -------------------------------------------------
   (IF) if
    LOAD?
   -------------------------------------------------
   •_COND_:
   -------------------------------------------------
      ..............................
      _COND_= locRes==£LOAD;
      ..............................
      (THEN) then
         (CALL) Init_Var
         (CALL) translation
   (LBL) End
(IF) if
 Already running
-------------------------------------------------
•_COND_:
-------------------------------------------------
   ..............................
   _COND_= runFlg@\CRD==1;
   ..............................
   (THEN) then
      ..............................
      GUI_SND(_PTR_@\CRD,£SHOW);
      ..............................
      (BREAK) break
      -------------------------------------------------
      •_LEVEL_:   (Type of ITEM that break)
      -------------------------------------------------
(IF) if
 Run as THREAD or EXEC
-------------------------------------------------
•_COND_:
-------------------------------------------------
   ..............................
   _COND_= mode==£THREAD;
   ..............................
   (THEN) then
      (THREAD) \CRD\Core
      -------------------------------------------------
      •_PAR_: ONCE  (SYNC ONCE ONCE_FOR_CALLER)
      •_RSLT_:   (For ONCExx the name of the symbol where to put the result
      -1=failToStart 0=starting 1=start. ONCE_FOR_CALLER is syncronous)
      -------------------------------------------------
   (ELSE) else
      (EXEC) \CRD\Core
(BREAK) break
-------------------------------------------------
•_LEVEL_:   (Type of ITEM that break)
-------------------------------------------------
<->
(BLK) Init_Var
```

```
      ..............................
      caller@\CRD=par_Gui;
      nId@\CRD=par_nID;
      ..............................
   (BLK) translation
      ..............................
      #WHILE(1);
      °res=£SKIP;

      #IF(NOT ISNUM(par_nId));
      #BREAK;
      #END;

      #IF(NOT EXIST("\DATA\TRSL"));
      #BREAK;
      #END;

      °res=£OK;
      #BREAK;
      #END;
      ..............................
      (IF) if
       OK ?
      ------------------------------------------------
      •_COND_:
      ------------------------------------------------
         ..............................
         _COND_= °res==£SKIP;
         ..............................
         (THEN) then
            (BREAK) break
            ------------------------------------------------
            •_LEVEL_:   (Type of ITEM that break)
            ------------------------------------------------
      (EXEC) \DATA\TRSL\gui_lod
         (SET) set
            ..............................
            GUI=_GUI_@\CRD;
            nId=par_nId;
            ..............................
=======================================
(MTHD) evt_clear
 Reset the EVT list
------------------------------------------------
•lis1:
------------------------------------------------
   ..............................
   #WHILE(1);
   lis1=LIS_POP(evtL);
   #IF(lis1<=0);!!! Invalid PTR -> end of list;
   #BREAK;
   #END;

   TRASH(lis1);
   #END;
   ..............................
=======================================
================================================================================
(EXO) JSON
```

```
-------------------------------------------------
•TEXT:
•kb1:
-------------------------------------------------
   (EXEC) PARSE
      (SET) set
         ..............................
         JSON=TEXT;   !!JSON text;
         ..............................
      (GET) get
         ..............................
         kb1=kb1;   !!PTR to KB1;
         ..............................
   (MTHD) PARSE
   -----------------------------------------------
   •JSON:   (JSON text)
   •kb1:   (PTR to KB1)
   •TKN:   (LIS of Token)
   •TT:   (Token Typ)
   •TK:   (Token)
   •LTT:
   •LTK:
   •STK:   (Stack TBL)
   •stkIDX:   (Stack index;)
   •STS:   (Status)
   •STKSTS:   (Stack status)
   •PTH:   (Path)
   •OBJCNT:   (Object Count)
   •ARYCNT:   (Array count)
   •ARYELM:
   •OBJ:   (Actual Oject)
   •NOBJ:
   •TOBJ:   (Counter OBJ)
   •TARY:   (Counter Ary)
   -----------------------------------------------
      (IF) if
      -------------------------------------------------
      •_COND_:
      -------------------------------------------------
         ..............................
         _COND_=0;
         JSON=NSP(JSON);
         #if(len(JSON));

         JSON=SYMB_RPLC(JSON,"\d\n",char(0x8000));
         JSON=SYMB_RPLC(JSON,"\n",char(0x8000));
         JSON=SYMB_RPLC(JSON,"\""",char(0x8001));
         JSON=SYMB_RPLC(JSON,"\\",char(0x8002));

         tkn=tknzop(JSON,"[","{","}","]",":",",","""");

         #if(!lis_num(tkn));
         trash(tkn);
         _COND_= 1;
         #end;
         #end;
```

```
   ..............................
   (THEN) then
      (BREAK) break
      ------------------------------------------------
      •_LEVEL_:   (Type of ITEM that break)
      ------------------------------------------------
(LBL) NoToken
 Detect token inside ""
------------------------------------------------
•nlis:   (New List)
------------------------------------------------
   ..............................
   °nlis=lis_new;
   °inf=0;

   °tk=£;
   °tkn=£;
   lis_pos(tkn,1);
   #while(lis_num(tkn));

   °ltk=°tk;
   °tk=lis_get(tkn);

   #if(°inf);

   #if(°tk != """" );
   °tkn=°tkn++°tk;
   #skip;
   #end;

   #if(°tk == """");
   lis_add(°nlis,quode(°tkn));
   °tkn=£;
   °inf=0;
   #skip;
   #end;

   #else;

   #if(°tk != """" );
   °tk=NSP(°tk,£SC);
   #if(~°tk);
   lis_add(°nlis,°tk);
   #end;
   #skip;
   #else;
   #if(°Ltk == """");
   #skip;
   #end;

   #end;

   °inf=1;
   °tkn=£;
   #end;
   #end;

   trash(tkn);
   tkn=°nlis;
```

```
   lis_pos(tkn,1);
   ..............................
(WHILE) while_1
------------------------------------------------
•_COND_:
------------------------------------------------
   (PRE) pre
      ..............................
      kb1=KB1_OPN_NTHS();
      STS=£ND; !!NOT DEFINED;
      PTH="\JSON";
      OBJ=£JSON;
      NOBJ=£;
      OBJCNT=0;
      ARYCNT=0;
      ARYELM=0;

      TOBJ=0;
      TARY=0;
      trash(stk);

      stk=TBL_NEW(NULL,1,NULL,NULL,"pth;sts;obj;objcnt;arycnt;aryelm");
      stkIdx=1;

      lis_pos(tkn,1);
      ..............................
   ..............................
   _COND_= lis_num(TKN);
   ..............................
   (DO) do
      (CALL) GetTkn
      (SWITCH) TT
      ------------------------------------------------
      •C_ST: ST  (String)
      •C_OB: OB  (Object Begin)
      •C_OE: oe  (Object End)
      •C_AB: AB  (Array Begin)
      •C_AE: AE  (Array End)
      •C_VS: VS  (Value Separator)
      ------------------------------------------------
         (CASE) _DEFAULT_
         (CASE) C_ST
            (SWITCH) STS
            ------------------------------------------------
            •C_OBJ: OBJ  (Object)
            •C_ARY: ARY  (Array)
            ------------------------------------------------
               (CASE) _DEFAULT_
               (CASE) C_OBJ
                  ..............................
                  #if(fst(tk)=="""" and lst(tk)="""");
                  TK=(TK <<1)>>1 ;
                  #end;
                  ..............................
                  (CALL) GetTkn
                  ..............................
                  #if(TT==£VS);
                  !!TT=£ST;
```

```
            TK=LTK;
            #end;

            #if(fst(tk)=="""" and lst(tk)="""");
            TK=(TK <<1)>>1 ;
            #end;

            #if(TT==£ST or TT==£VS);

            TK=quose(TK);

            !!chatput(pth++"."++LTK++"="++tk);
            kb1_dlg(kb1,pth++"."++LTK++"="++tk);
            TT=LTT;

            #else;
            NOBJ=LTK;
            #end;
            ..............................
         (CASE) C_ARY
            ..............................
            #if(fst(tk)=="""" and lst(tk)="""");
            TK=(TK <<1)>>1 ;
            #end;
            ..............................
            (CALL) GetTkn
            ..............................
            #if(TT==£VS or TT==£AE);
            aryelm+=1;
            °obj="_AE_"++aryelm;

            LTK=quose(LTK);

            !!chatput(pth++"."++°obj++"="++ltk);

            kb1_dlg(kb1,pth++"."++°obj++"="++ltk);

            #else;
            NOBJ=ltk;
            #end;
            ..............................
(CASE) C_OB
   ..............................
   #if(~NOBJ==0);
   #if(STS==£ARY);
   arycnt+=1;
   nobj="_AE_"++arycnt;
   °TOBJ="_AE_"++arycnt;
   #else;
   TOBJ+=1;
   objCnt+=1;
   nobj="_OBJ_"++objCnt;
   °TOBJ="_OBJ_"++TOBJ;
   #end;
   #else;
   °TOBJ=NOBJ;
   #end;

   tbl_itm(stk,£STS,stkidx,sts);
```

```
   tbl_itm(stk,£Pth,stkidx,pth);
   tbl_itm(stk,£OBJ,stkidx,obj);
   tbl_itm(stk,£OBJcnt,stkidx,objcnt);
   tbl_itm(stk,£ARYcnt,stkidx,arycnt);
   tbl_itm(stk,£ARYelm,stkidx,aryelm);

   stkIdx=tbl_chg(stk,NULL,£ADD,£ROW);

   objcnt=0;
   arycnt=0;
   aryelm=0;

   OBJ=NOBJ;
   pth=pth++"\"++obj;
   NOBJ=£;

   !!chatput(pth++"="++quose(°TOBJ));
   kb1_dlg(kb1,pth++"="++quose(°TOBJ)++";"++pth++":=£OBJ");


   STS=£OBJ;
   ............................
(CASE) C_OE
   ............................
   stkIdx=TBL_CHG(stk,NULL,£SUB,£ROW);

   sts=tbl_itm(stk,£STS,stkidx);
   pth=tbl_itm(stk,£Pth,stkidx);
   obj=tbl_itm(stk,£OBJ,stkidx);
   objcnt=tbl_itm(stk,£OBJcnt,stkidx);
   arycnt=tbl_itm(stk,£ARYcnt,stkidx);
   aryelm=tbl_itm(stk,£ARYelm,stkidx);
   ............................
(CASE) C_AB
   ............................
   #if(~NOBJ==0);
   #if(STS==£OBJ);
   TARY+=1;
   aryCnt+=1;
   nobj="_ARY_"++aryCnt;
   °TOBJ="_ARY_"++TARY;
   #else;
   aryElm+=1;
   nobj="_AE_"++aryElm;
   °TOBJ=NOBJ;
   #END;
   #else;
   °TOBJ=NOBJ;
   #end;

   tbl_itm(stk,£STS,stkidx,sts);
   tbl_itm(stk,£Pth,stkidx,pth);
   tbl_itm(stk,£OBJ,stkidx,obj);
   tbl_itm(stk,£OBJcnt,stkidx,objcnt);
   tbl_itm(stk,£ARYcnt,stkidx,arycnt);
   tbl_itm(stk,£ARYelm,stkidx,aryelm);

   stkIdx=tbl_chg(stk,NULL,£ADD,£ROW);
```

```
            objcnt=0;
            arycnt=0;
            aryelm=0;

            OBJ=NOBJ;
            pth=pth++"\"++obj;
            NOBJ=£;

            #if(STS==£ARY);
            °tag="ARY,AE";
            #else;
            °tag="ARY";
            #end;

            !!chatput(pth++"="++quose(°TOBJ));
            kb1_dlg(kb1,pth++"="++quose(°TOBJ)++";"++pth++":="++quos(°tag));

            STS=£ARY;
            ..............................
         (CASE) C_AE
            ..............................
            stkIdx=tbl_chg(stk,NULL,£SUB,£ROW);

            sts=tbl_itm(stk,£STS,stkidx);
            pth=tbl_itm(stk,£Pth,stkidx);
            obj=tbl_itm(stk,£OBJ,stkidx);
            objcnt=tbl_itm(stk,£OBJcnt,stkidx);
            arycnt=tbl_itm(stk,£ARYcnt,stkidx);
            aryelm=tbl_itm(stk,£ARYelm,stkidx);
            ..............................
         (CASE) C_VS
            (CALL) GetTkn
   (DONXT) doNxt
      ..............................
      trash(tkn);

      !!kb1_sav(kb1,"Json.kb1");
      ..............................
(BLK) GetTkn
   ..............................
   LTK=tk;
   LTT=tt;

   #while(LIS_NUM(tkn));
   tk=nsp(LIS_GET(tkn));

   #if(tk==",");
   TT=£VS;
   #else;
   TT=SYMB_DCD(TK,"[,],{,},:",£AB,£AE,£OB,£OE,£NS);
   #if(IsNULL(TT) and ~TK);
   TT=£ST;
   TK=SYMB_RPLC(TK,char(0x8000),crlf);
   TK=SYMB_RPLC(TK,char(0x8001),"""");
   TK=SYMB_RPLC(TK,char(0x8002),"\");
   #break;
   #end;
   #end;
```

```
        #if(TT !="NS" );
        #break;
        #end;
        #end;
        ..............................
   ========================================
================================================================================
```
(VAR) CONFIG

Strumenti

```
-------------------------------------------------
•TBLIDX:   (Main Tbl)
•TGEN:
•TOAI:   (Tabella OpenAi)
•TVOI:   (Tabella Voice)
•TPFX:   (Tsbella Prefissi)
•TFTR:   (Tabella Filtro)
-------------------------------------------------
   ========================================
```
(MTHD) TV_2_TBL_int

Ritorna il contenuto di una Tavola (doc) in una tabella

```
   -------------------------------------------------
   •DOC:   (PTR to doc)
   •TVI:   (Nome Tavola)
   •HEAD:   (£YES £NO)
   •TBL:
   •tv:
   •nr:
   •nc:
   •nh:
   •r:
   •c:
   •h:
   •nrt:   (Row con dati)
   •f:
   •t:
   -------------------------------------------------
```
(WHILE) while

```
      -------------------------------------------------
      •_COND_:
      -------------------------------------------------
```
(PRE) pre

```
         ..............................
         TBL=NULL;
         h=£;
         tv=DOC_TBL(doc,TVI);
         nrt=0;
         nr=0;
         nc=0;
         r=0;

         #if(tv>0);
         nr=DOC_TINF(tv,£ROWDAT);
         nc=DOC_TINF(tv,£COL);

         #if(HEAD==£YES);
         c=0;
```

```
         nh=DOC_TINF(tv,£ROW) - nr;
         #if(nh>0);
         #while(c+=1 <=NC);
         #if(~H);H=H++";";#end;
         H=h++DOC_TVL(tv,c,nh,NULL,NULL,£TRUE);
         #end;
         #end;

         #end;
         #end;

         #if(nr>0);
         #if(~H);
         tbl=TBL_NEW(NULL,nr,NULL,NULL,H);
         #else;
         tbl=TBL_NEW(nc,nr);
         #end;
         #end;
         ..............................
      ..............................
      _COND_= r+=1 <=nr;
      ..............................
      (DO) do
         ..............................
         c=0;
         f=0;
         #while(c+=1 <=nc);
         t=DOC_TVL(tv,c,r);
         #if(~t);
         #if(f==0);nrt+=1;f=1;#end;

         tbl_itm(tbl,c,nrt,t);
         #end;
         #end;
         ..............................
      (DONXT) doNxt
         ..............................
         #if(!nrt);
         trash(tbl);
         tbl=NULL;
         #else;
         #if(nrt != tbl_inf(tbl,£row));
         tbl_chg(tbl,NULL,nrt);
         #end;
         #end;

         trash(tv);
         ..............................
=========================================
```

(MTHD) TV_2_TBL

Ritorna il contenuto di una Tavola (doc) in una tabella con la prima colonna come row index

---

- DOC: (PTR to doc)
- TVI: (Nome Tavola)
- HEAD: (£YES £NO)
- TBL:
- i:

```
•in:
-------------------------------------------------
   (EXEC) TV_2_TBL_int
      (SET) set
         ..............................
         DOC=DOC;   !!PTR to doc;
         TVI=TVI;   !!Nome Tavola;
         HEAD=HEAD;   !!£YES £NO;
         ..............................
      (GET) get
         ..............................
         TBL=TBL;
         ..............................
   ..............................
   #if(~TBL);

   i=0;
   in=tbl_inf(tbl,£ROW);

   #while(i+=1 <=in);
   TBL_NAM(tbl,£ROW,i,Tbl_itm(tbl,1,i));
   #end;

   #end;
   ..............................
=========================================
(MTHD) TS_2_TXT
 Time stamp to Text
-------------------------------------------------
•TS:
•YMD:
•DMY:
•HMS:
•DT:
•TD:
•YY:
•MM:
•DD:
•HH:
•MN:
•SS:
-------------------------------------------------
   ..............................
   YY=DT_TSDEC(TS, £YY ,£LOC);
   MM=DT_TSDEC(TS, £MM ,£LOC);
   DD=DT_TSDEC(TS, £DD ,£LOC);
   HH=DT_TSDEC(TS, £HH ,£LOC);
   MN=DT_TSDEC(TS, £MN ,£LOC);
   SS=DT_TSDEC(TS, £SS ,£LOC);

   YMD=frmt("%02d-%02d-%02d",YY,MM,DD);
   DMY=frmt("%02d-%02d-%02d",DD,MM,YY);

   HMS=frmt("%02d:%02d:%02d",HH,MN,SS);

   DT=YMD..HMS;
   TD=HMS..DMY;
```

```
   ..............................
=========================================
(MTHD) ReadCfg
-------------------------------------------------
•Conf:   (Config Manifest elemnt)
•doc:
•cf:
•k:
•kn:
•tbl:
-------------------------------------------------
   ..............................
   cf=pkgpth++PKG_MNFGET(NULL,£CFG,conf);

   doc=DOC_DOC(£OO,cf);
   Doc_opn(doc);
   ..............................
   (EXEC) TV_2_TBL
      (SET) set
         ..............................
         DOC=doc;   !!PTR to doc;
         TVI="INDICE TABELLE";   !!Nome Tavola;
         HEAD=£YES;   !!£YES £NO;
         ..............................
      (GET) get
         ..............................
         tblIDX=TBL;
         ..............................
   (WHILE) while
    Carica tutte le tabelle dell`indice
   -------------------------------------------------
   •_COND_:
   -------------------------------------------------
      (PRE) pre
         ..............................
         k=0;
         kn=tbl_inf(tblIdx,£ROW);
         ..............................
      ..............................
      _COND_= k+=1 <=kn;
      ..............................
      (DO) do
         (EXEC) TV_2_TBL
            (SET) set
               ..............................
               DOC=doc;   !!PTR to doc;
               TVI=TBL_ITM(tblIDX,£TABELLA,k);   !!Nome Tavola;
               HEAD=£YES;   !!£YES £NO;
               ..............................
            (GET) get
               ..............................
               TBL=TBL;
               ..............................
         ..............................
         ref(TBL_ITM(tblIDX,£TBL,k))=TBL;
         ..............................
=========================================
(MTHD) INIT
```

```
   ------------------------------------------------
   •t:
   •dlg:
   •n:
   •ctIepNrm: 0  (Contatore iep Normali)
   •i:
   •in:
   •docAlr:
   •tblAlr:
   ------------------------------------------------
      (LBL) ReadCfg
      ------------------------------------------------
      •doc:
      •nr:
      •r:
      •i:
      •in:
      •t:
      ------------------------------------------------
         (EXEC) ReadCfg
            (SET) set
               ..............................
               Conf=£Config;   !!Config Manifest elemnt;
               ..............................
            (GET) get
               ..............................
               doc=doc;
               ..............................
         ..............................
         trash(doc);
         ..............................
   ========================================
================================================================================
(VAR) GPT
------------------------------------------------
•ept:    (End Point)
•key:   (API KEI)
•hsrv:   (HTTp server)
•org:   (ORGANIZATION ID)
•hdr:
•TFILE:
•TMDL:
------------------------------------------------
   ========================================
   (MTHD) INIT
   ------------------------------------------------
   •EPT:   (End point)
   •Key:   (Key)
   •ORG:
   ------------------------------------------------
      ..............................
      hsrv=HTTP_OPN(£HTTPS,ept);


      hdr="Content-type: application/json"++crlf++
      "Authorization: Bearer"..key++crlf;
```

```
   ept@\GPT=ept;
   key@\GPT=key;
   org@\GPT=org;
   ..............................
=======================================
(MTHD) PROMPT
-----------------------------------------------
•QRY:
•MAXT:
•TEMP:
•MODEL:
•SUFFIX:
•RPLY:
•TKN:   (Total,Prompt,Completition)
•RES:   (£OK, £ERR)
•INF:   (HTTP res)
•MSG: {"model": "§MODEL", "prompt": "§PROMPT", "temperature": §TEMP,
"max_tokens": §MAXT §SUFFIX §USER}
•
-----------------------------------------------
   ..............................
   MODEL=if(~MODEL,MODEL, "text-davinci-003");

   maxt=if(maxt<1,1000,MAXT);
   temp=if(temp<1,0,temp);

   qry=SYMB_RPLC(qry,"\","\\");
   qry=SYMB_RPLC(qry,crlf,"\n");
   qry=SYMB_RPLC(qry,"""","\""");
   qry=SYMB_RPLC(qry,char(10),"\n");
   qry=SYMB_RPLC(qry,char(15),"\t");
   qry=SYMB_RPLC(qry,char(13),"\r");

   #if(~SUFFIX);
   SUFFIX=SYMB_RPLC(SUFFIX,"\","\\");
   SUFFIX=SYMB_RPLC(SUFFIX,crlf,"\n");
   SUFFIX=SYMB_RPLC(SUFFIX,"""","\""");
   SUFFIX=SYMB_RPLC(SUFFIX,char(10),"\n");
   SUFFIX=SYMB_RPLC(SUFFIX,char(15),"\t");
   SUFFIX=SYMB_RPLC(SUFFIX,char(13),"\r");
   #end;
   ..............................
   ..............................
   °cmd="/v1/completions";
   °msg=SYMB_RPLC(msg,"§PROMPT",QRY,NULL,NULL,0);
   °msg=SYMB_RPLC(°msg,"§MODEL",MODEL,NULL,NULL,0);
   °msg=SYMB_RPLC(°msg,"§MAXT",MAXT,NULL,NULL,0);
   °msg=SYMB_RPLC(°msg,"§TEMP",TEMP,NULL,NULL,0);

   #if(~SUFFIX);
   °msg=SYMB_RPLC(°msg,"§SUFFIX",","++QUOD(£suffix)+
   +":"..quod(suffix),NULL,NULL,0);
   #else;
   °msg=SYMB_RPLC(°msg,"§SUFFIX",£,NULL,NULL,0);
   #end;

   #if(~ORG);
```

```
   °msg=SYMB_RPLC(°msg,"§USER",","++QUOD(£user)++":"..quod(org),NULL,NULL,0);
   #else;
   °msg=SYMB_RPLC(°msg,"§USER",£,NULL,NULL,0);
   #end;
   °hdr=hdr;

   SMF(HSRV,£LCKS);
   °rpl=HTTP_POST(hsrv,°cmd,£TEXT,£UTF,°msg,£TEXT,£UTF,NULL,°hdr,£HEADER)-
   >°rhdr;
   SMF(HSRV,£LCKR);

   #if(~°rhdr);

   INF=SPLT(°rhdr,crlf,£LEFT);

   #if(~°rpl and srch(INF,200));

   °kb1=EXO("\JSON",TEXT::°rpl, ?kb1);
   °tbl=KB1_QRY(°kb1,"TBLATT(\\\.text,'text')");

   °i=0;
   °in=tbl_inf(°tbl,£ROW);
   rply=£;
   #while(°i+=1 <=°in);
   rply=rply++Tbl_itm(°tbl,£TEXT,°i);
   #end;

   rply=SYMB_RPLC(rply,"\r\n",crlf);
   rply=SYMB_RPLC(rply,"\t",char(15));
   rply=SYMB_RPLC(rply,"\r",char(13));
   rply=SYMB_RPLC(rply,"\n",char(10));

   tkn=KB1_QRY(°kb1,"\Json\_OBJ_1\usage.prompt_tokens++','+
   +\Json\_OBJ_1\usage.completion_tokens++','++\Json\_OBJ_1\usage.total_tokens",
   £TEXT);

   RES=£OK;
   trash(°kb1,°tbl);
   #else;
   RES=£ERR;
   #end;
   #else;
   RES=£ERR;
   INF=£TMO;

   #end;
   ..............................
=========================================
```

(MTHD) CHAT

Role: system, user, assistant

- QRY:
- MAXT:
- TEMP:
- MODEL:
- SUFFIX: (Ignored)
- RPLY:
- TKN: (Total;Prompt;Completition)

```
•RES:   (£OK, £ERR)
•INF:   (HTTP res)
•MSG: {"model": "§MODEL", "messages": [§PROMPT], "temperature": §TEMP,
"max_tokens": §MAXT}
•
•MSGELM1: {"role": "§ROLE", "content": "§CONTENT"}
•
•MSGELM2: {"role": "§ROLE", "name": "§NAME", "content": "§CONTENT"}
-------------------------------------------------
   ..............................
   #if(~MODEL==0);
   °MODEL1="gpt-4";
   °MODEL2="gpt-4";
   °MODEL3="gpt-3.5-turbo";
   °MODEL4="gpt-3.5-turbo-0301";
   MODEL=°MODEL1;
   #end;

   maxt=if(maxt<1,1000,MAXT);
   temp=if(temp<1,0,temp);


   #if(PtrTyp(qry)==£TBL);
   °tb=qry;
   °Todel=0;
   #else;
   °Todel=1;
   °tb=TBL_NEW(NULL,1,NULL,NULL,"ROLE;CONTENT");
   tbl_itm(°tb,£ROLE,1,"user");
   tbl_itm(°tb,£CONTENT,1,Qry);
   #end;

   °msgElm=£;
   °in=tbl_inf(°tb,£ROW);
   °i=0;

   #while(°i+=1 <=°in);
   °role=tbl_itm(°tb,£ROLE,°i);
   QRY=tbl_itm(°tb,£CONTENT,°i);

   qry=SYMB_RPLC(qry,"\","\\");
   qry=SYMB_RPLC(qry,crlf,"\n");
   qry=SYMB_RPLC(qry,"""","\""");
   qry=SYMB_RPLC(qry,char(10),"\n");
   qry=SYMB_RPLC(qry,char(15),"\t");
   qry=SYMB_RPLC(qry,char(13),"\r");

   °elm=MSGELM1;
   #if(tbl_inf(°tb,£COL)>2);
   °name=tbl_itm(°tb,£NAME,°i);
   #if(~°name);
   °elm=MSGELM2;
   SYMB_RPLC(°elm,"§NAME",°name,NULL,NULL,0);
   #end;
   #end;

   °elm=SYMB_RPLC(°elm,"§ROLE",°role,NULL,NULL,0);
   °elm=SYMB_RPLC(°elm,"§CONTENT",qry,NULL,NULL,0);
```

```
°msgElm=°msgElm++if(~°msgElm,",",£)++°elm;
#end;

#if(°Todel);trash(°tb);#end;

QRY=°msgElm;
.............................
.............................
°cmd="/v1/chat/completions";
°msg=SYMB_RPLC(msg,"§PROMPT",QRY,NULL,NULL,0);
°msg=SYMB_RPLC(°msg,"§MODEL",MODEL,NULL,NULL,0);
°msg=SYMB_RPLC(°msg,"§MAXT",MAXT,NULL,NULL,0);
°msg=SYMB_RPLC(°msg,"§TEMP",TEMP,NULL,NULL,0);

°hdr=hdr;

SMF(HSRV,£LCKS);
°rpl=HTTP_POST(hsrv,°cmd,£TEXT,£UTF,°msg,£TEXT,£UTF,NULL,°hdr,£HEADER)-
>°rhdr;
SMF(HSRV,£LCKR);

chatput(£CHAT..°msg..°rpl..crlf..°rhdr);

#if(~°rhdr);

INF=SPLT(°rhdr,crlf,£LEFT);

#if(~°rpl and srch(INF,200));

°kb1=EXO("\JSON",TEXT::°rpl, ?kb1);
°tbl=KB1_QRY(°kb1,"TBLATT(\\\.role,'role;content')");

°i=0;
°in=tbl_inf(°tbl,£ROW);
rply=£;
#while(°i+=1 <=°in);
rply=rply++Tbl_itm(°tbl,£CONTENT,°i);
#end;

rply=SYMB_RPLC(rply,"\r\n",crlf);
rply=SYMB_RPLC(rply,"\t",char(15));
rply=SYMB_RPLC(rply,"\r",char(13));
rply=SYMB_RPLC(rply,"\n",char(10));

tkn=KB1_QRY(°kb1,"\Json\_OBJ_1\usage.prompt_tokens++','+
+\Json\_OBJ_1\usage.completion_tokens++','++\Json\_OBJ_1\usage.total_tokens",
£TEXT);

RES=£OK;
trash(°kb1,°tbl);
#else;
RES=£ERR;
#end;
#else;
RES=£ERR;
INF=£TMO;

#end;
.............................
```

```
=========================================
(MTHD) EMBEDDING
-------------------------------------------------
•QRY:
•MODEL:
•RPLY:
•RES:
•INF:
•MSG: {"model": "§MODEL", "input": §PROMPT}
•
-------------------------------------------------
   (WHILE) while
   -------------------------------------------------
   •_COND_:
   -------------------------------------------------
      (PRE) pre
         ..............................
         MODEL=if(~MODEL,MODEL,"text-embedding-ada-002");

         #if(PtrTyp(qry)==£TBL);
         °tb=qry;
         °Todel=0;
         #else;
         °Todel=1;
         °tb=TBL_NEW(NULL,1,NULL,NULL,"idx;EMB");
         tbl_itm(°tb,£IDX,1,1);
         tbl_itm(°tb,£EMB,1,Qry);
         #end;

         °msgElm=£;
         °in=tbl_inf(°tb,£ROW);
         °i=0;

         #while(°i+=1 <=°in);
         QRY=tbl_itm(°tb,£EMB,°i);

         qry=SYMB_RPLC(qry,"\","\\");
         qry=SYMB_RPLC(qry,crlf,"\n");
         qry=SYMB_RPLC(qry,"""","\""");
         qry=SYMB_RPLC(qry,char(10),"\n");
         qry=SYMB_RPLC(qry,char(15),"\t");
         qry=SYMB_RPLC(qry,char(13),"\r");

         °elm=quod(qry);

         °msgElm=°msgElm++if(~°msgElm,",",£)++°elm;
         #end;

         QRY="["++°msgElm++"]";
         ..............................
         ..............................
         °TRY=0;
         RES=£;
         ..............................
      ..............................
      °trY+=1;
      _COND_= res!=£OK and °try<=2;
      ..............................
```

```
(DO) do
   ..............................
   °cmd="/v1/embeddings";
   °msg=SYMB_RPLC(msg,"§PROMPT",QRY,NULL,NULL,0);
   °msg=SYMB_RPLC(°msg,"§MODEL",MODEL,NULL,NULL,0);

   °hdr=hdr;

   SMF(HSRV,£LCKS);
   °rpl=HTTP_POST(hsrv,°cmd,£TEXT,£UTF,°msg,£TEXT,£UTF,NULL,°hdr,£HEADER)-
   >°rhdr;
   SMF(HSRV,£LCKR);

   #if(~°rhdr);

   INF=SPLT(°rhdr,crlf,£LEFT);

   #if(~°rpl and srch(INF,200));

   °kb1=EXO("\JSON",TEXT::°rpl, ?kb1);

   °tbl1=KB1_QRY(°kb1,"tblatt(\JSON\_OBJ_1\data\\embedding,null,null,
   £yes)");
   °jn=tbl_inf(°tbl1,£ROW);
   °j=0;

   #while(°j+=1 <=°jn);

   °tbl2=tbl_itm(°tbl1,£_tblATT_,°j);

   °in=tbl_inf(°tbl2,£col);
   °bf=BUF_NEW(°in,£F32);
   °i=0;
   #while(°i+=1 <=°in);
   BUF_VAL(°bf,°i,tbl_itm(°tbl2,°i,1));
   #end;

   tbl_itm(°tb,£emb,°j,°bf);
   trash(°tbl2);
   #end;

   trash(°kb1,°tbl1);
   RES=£OK;
   #else;
   RES=£ERR;
   #end;
   #else;
   RES=£ERR;
   INF=£TMO;
   #end;
   ..............................
(DONXT) doNxt
   ..............................
   #if(°toDEL==1);
   #if(RES==£OK);
   RPLY=tbl_itm(°tb,£emb,1);
   #end;
   trash(°tb);
   #else;
```

```
         #if(RES==£OK);
         RPLY=°tb;
         #end;
         #end;
         ..............................
   ========================================
   (MTHD) MODEL
    Returnm a tbl with  available models (ID column)
   ------------------------------------------------
   •RPLY:
   •TKN:   (Total,Prompt,Completition)
   •RES:   (£OK, £ERR)
   •INF:   (HTTP res)
   ------------------------------------------------
      ..............................
      °cmd="/v1/models";

      °hdr=hdr;

      °rpl=HTTP_GET(hsrv,°cmd,£TEXT,£UTF,NULL,°hdr,£HEADER)->°rhdr;

      !!chatput(£MODELS..°rpl);

      #if(~°rhdr);

      INF=SPLT(°rhdr,crlf,£LEFT);

      #if(~°rpl and srch(INF,200));

      °kb1=EXO("\JSON",TEXT::°rpl, ?kb1);

      °tbl=KB1_QRY(°kb1,"tblatt(\JSON\_OBJ_1\data\,'id')");

      rply=°tbl;
      trash(°kb1);
      #else;
      RES=£ERR;
      #end;
      #else;
      RES=£ERR;
      INF=£TMO;
      #end;
      ..............................
   ========================================
================================================================================
(VAR) AI_DB
 data base per AI
------------------------------------------------
•cnv:   (PDF converter)
•Nth:   (Pending Thread)
------------------------------------------------
   ========================================
   (MTHD) OPEN
   ------------------------------------------------
   •NAME:
   •VDIM:   (Vector DIM)
   •VMAX:   (Vector max size)
   •TVCT:   (Tabella con due colonne)
```

```
------------------------------------------------
   ..............................
   TVCT=TBL_NEW(NULL,1,NULL,NULL,"MNG;SQL;NAME");

   °Mng=AIT_VCT(NAME++".vct",VDIM);
   #if(!°Mng);
   °Mng=AIT_VCT(VMAX,VDIM);
   #end;

   °DB=DB_OPN(£SQLITE,NAME++".DB");

   °qry="CREATE TABLE IF NOT EXISTS dat (id integer primary key
   autoincrement,slct TEXT,emb TEXT); CREATE TABLE IF NOT EXISTS doc (slct
   TEXT)";
   trash(DB_QRY(°DB,°qry));

   TBL_ITM(TVCT,1,1,£ROW,°mng,°db,NAME);

   cnv=pkgpth++PKG_MNFGET(NULL,£CONV,£PDF2TXT);
   ..............................
==========================================
(MTHD) GPTload
------------------------------------------------
•TVCT:   (Tabella)
•OBJ:   (pdf file name)
•SLCT:
•txt:
•sql:
•vctMng:
•name:
------------------------------------------------
   ..............................
   vctMng=tbl_ITM(TVCT,£MNG,1);
   sql=tbl_ITM(TVCT,£sql,1);
   name=tbl_ITM(TVCT,£NAME,1);
   ..............................
   (WHILE) while
   ------------------------------------------------
   •_COND_:
   ------------------------------------------------
      (PRE) pre
         ..............................
         °knd=SPLT(obj,".",£RIGHT,£REV);
         #if(°knd==£pdf);
         txt=OSSTART(quode(cnv)."-nopgbrk -eol dos"..quode(obj)."-",£GETOUT);
         #else;
         txt=FS_FRDS_F(obj);
         #end;

         °ls=TKNZ(TXT,crlf,char(13));
         °in=lis_num(°ls);
         °i=0;

         °ix=AIT_VCT_INF(°vctMng,£COUNT);
         °slct=SPLT(obJ,".",£LEFTORALL,£REV,0);
         °slct=SPLT(°slct,"\",£RIGHTORALL,£REV,0);

         MAX@\MG\pag\pPDF\LOAD\PRGS=°in;
```

```
   ..............................
..............................
_COND_= °i+=1 <=°in;
..............................
(DO) do
   ..............................
   °e=lis_pop(°ls);
   °e=SYMB_RPLC(°e,char(34),"`");
   °e=SYMB_RPLC(°e,".....","  ");
   °e=SYMB_RPLC(°e,"=====","  ");
   °e=SYMB_RPLC(°e,"-----","  ");
   °e=SYMB_RPLC(°e,"_____","  ");
   °e=NSP(°e,£DOUBLE);
   °e=NSP(°e,£NOCHR);

   °MXfrg=TBL_ITM(TGEN@\CONFIG,£VAl,£MXfrg);

   VALUE@\MG\pag\pPDF\LOAD\PRGS=lis_NUM(°ls);

   #if(°e != °le);

   °el=°el..°e;

   #if(len(°el)>°MXfrg or !lis_num(°ls));

   °qry="INSERT INTO dat (slct,emb) values ("++quose(°slct)++","+
   +quose(°el)++");
   SELECT max(rowid) from dat;";
   °dr=db_qry(sql,°qry);

   #if(!IsNull(°dr));

   °IDX=DB_GET(°dr,£FLD,1,1);
   trash(°dr);

   #if(!°tbe);
   °tbe=TBL_NEW(NULL,0,NULL,NULL,"IDX;EMB");
   #end;

   °ir=TBL_CHG(°tbe,NULL,£ADD,£ROW);

   TBL_ITM(°tbe,£IDX,°ir,£ROW,°idx,"["++°slct++"]"++°el);

   #if(°ir>10 or !lis_num(°ls));
   nth+=1;
   EXOTHR("EMB", TBE::°tbe,vctMng::vctmng);
   °tbe=£;
   #end;
   #end;
   °el=£;
   #end;
   #end;
   °le=°e;
   ..............................
   ..............................
   #IF(NTH>20);
   #WHILE(NTH>10);
   SLEEP(100);
   #END;
```

```
         #END;
         ..............................
      (DONXT) doNxt
         ..............................
         MAX@\MG\pag\pPDF\LOAD\PRGS=NTH;

         #WHILE(NTH);
         VALUE@\MG\pag\pPDF\LOAD\PRGS=NTH;
         SLEEP(100);
         #END;
         AIT_VCT_SAV(vctMng,name++".vct");

         °qry="INSERT INTO doc (slct) values ("++quose(°slct)++")";
         trash(db_qry(sql,°qry));
         ..............................
   ..............................
   trash(°ls);
   ..............................
========================================
(MTHD) EMB
------------------------------------------------
•TBE:
•vctMng:
•res:
------------------------------------------------
   ..............................
   °tbe=EXO("\GPT\EMBEDDING",MODEL::£, QRY::TBE,?RPLY, ?res )->res;
   ..............................
   ..............................
   #if(°tbe and RES==£OK);
   °in=tbl_inf(°tbe,£row);
   °i=0;

   #while(°i+=1 <=°in);
   °bf=tbl_itm(°tbe,£EMB,°i);
   °ix=tbl_itm(°tbe,£IDX,°i);

   !!chatput(£BF..°ix..ptrtyp(°bf)..BUF_INFO(°bf,£NUM)..BUF_INFO(°bf,
   £SIZ)..BUF_INFO(°bf,£TYP));

   #if(°bf);
   SMF(vctMng,£LCKS);
   AIT_VCT_INS(vctMng,°bf,°ix);
   SMF(vctMng,£LCKR);
   #end;
   trash(°bf);
   #end;
   #end;
   ..............................
   ..............................
   trash(TBE);
   nth-=1;
   ..............................
========================================
(MTHD) VCT_SRC
------------------------------------------------
•TVCT:
•VCT:
```

•SLCT:
•NUM:
•MAXT:
•TXT:

```
   ..............................
   °vctMng=tbl_ITM(TVCT,£MNG,1);
   °sql=tbl_ITM(TVCT,£SQL,1);

   #if(AIT_VCT_INF(°vctmng,£COUNT));
   °tr=AIT_VCT_SRC(°vctmng,VCT,NUM);
   °in=tbl_INF(°tr,£ROW);

   txt=£;
   °tl=0;
   °ltk=£;
   °ls=£;
   #if(°in);
   °i=0;
   #while(°i+=1 <=°in);
   °ix=tbl_itm(°tr,£IDX,°i);

   °qry="SELECT emb,slct FROM dat WHERE rowid="++°ix;

   °dr=DB_qry(°sql,°qry);

   #if(IsNULL(°dr));
   #skip;
   #end;

   °t=DB_GET(°dr,£FLD,1,1);
   °s=DB_GET(°dr,£FLD,2,1);

   trash(°dr);

   #if(~slct);
   #if(!SYMB_INDX(°s,SLCT));
   #skip;
   #end;
   #end;

   °tl+=len(°t);
   #if(MAXT and °tl>MAXT);
   #break;
   #end;

   #if(°t==°ltk);
   #skip;
   #end;

   °ltk=°t;
   TXT=TXT++if(°s !=°ls,crlf++"[source:"++°s++"]",£)++crlf++°t;
   °ls=°s;
   #end;
   #end;
   #end;
   ..............................
```

==========

(MTHD) VCT_SRC

```
-------------------------------------------------
•TVCT:
•VCT:
•SLCT:
•NUM:
•MAXT:
•TXT:
-------------------------------------------------
   ..............................
   °vctMng=tbl_ITM(TVCT,£MNG,1);
   °sql=tbl_ITM(TVCT,£SQL,1);


   °tr=AIT_VCT_SRC(°vctmng,VCT,NUM);
   °in=tbl_INF(°tr,£ROW);

   °to=TBL_NEW(NULL,0,NULL,NULL,"IDX;TXT");

   °tl=0;
   #if(°in);
   °i=0;
   #while(°i+=1 <=°in);
   °ix=tbl_itm(°tr,£IDX,°i);

   °cnt=2;
   #while(°cnt>=0);
   °qry="SELECT emb,slct FROM dat WHERE rowid="++°ix;

   °dr=DB_qry(°sql,°qry);

   #if(IsNULL(°dr));
   #break;
   #end;

   °t=DB_GET(°dr,£FLD,1,1);
   °s=DB_GET(°dr,£FLD,2,1);

   trash(°dr);

   #if(~slct);
   #if(!SYMB_INDX(°s,SLCT));
   #skip;
   #end;
   #end;

   °tl+=len(°t);
   #if(MAXT and °tl>MAXT);
   #break;
   #end;

   °r=TBL_chg(°to,NULL,£ADD,£ROW);
   TBL_ITM(°to,£IDX,°r,£ROW,°ix,°t);
   #BREAK;
   chatput(£IX..°ix);
   °cnt-=1;
   #if(°cnt==1);
   °ix-=1;
   #else;
```

```
      °ix+=2;
      #end;

      #if(TBL_SRC(°tr,£COL,£IDX,°ix));
      #break;
      #end;
      #end;

      #if(MAXT and °tl>MAXT);
      #break;
      #end;
      #end;
      #end;

      TXT=£;

      °in=tbl_inf(°to,£ROW);
      °i=0;
      !!TBL_SORT(°to,£ROW,£IDX,£ASC,£NUM);
      #while(°i+=1 <=°in);
      TXT=TXT++crlf++tbl_itm(°to,£TXT,°i);
      #end;
      trash(°to);
      ..............................
   ========================================
==========================================================================================
(VAR) AI_CORE
------------------------------------------------
•tblCht:   (TBL chat)
•MxCht:
•MXctxCPLT:
•MXctxCHT:
------------------------------------------------
   ========================================
   (MTHD) INIT
      ..............................
      tblCht=TBL_NEW(NULL,0,NULL,NULL,"ROLE;CONTENT");
      MXcht=TBL_ITM(TGEN@\CONFIG,£VAl,£MXcht);
      MXctxCPLT=TBL_ITM(TGEN@\CONFIG,£VAl, £MXctxCPLT);
      MXctxCHT=TBL_ITM(TGEN@\CONFIG,£VAl, £MXctxCHT)
      ..............................
   ========================================
   (MTHD) QRY
   ------------------------------------------------
   •qry:
   •MAXT:
   •MODEL:
   •TYPE:   (£CHAT £CPLT)
   •PFX:
   •SLCT:
   •RPLY:
   •RES:
   •INF:
   •Stop:   (messo a 1 se funzione completata)
   ------------------------------------------------
      ..............................
      °ps=£;
```

```
#if(~SLCT);
°ls=TKNZ(SLCT,",");
#while(lis_num(°ls));
°ps=°ps++"["++LIS_POP(°ls)++"]";
#end;
TRASH(°ls);
#end;

#if(VALUE@\MG\pag\pSETTING\CTX);

°vct=EXO("\GPT\EMBEDDING",MODEL::£, QRY::°ps..qry,?RPLY );

°ctx=EXO("\AI_DB\VCT_SRC", TVCT::TVCT@\MAIN, VCT::°VCT, NUM::25,
MAXT::if(TYPE==£CPLT, MXctxCPLT,MXctxCHT),
SLCT::SLCT,?TXT );

trash(°vct);
#else;
°ctx=£;
#end;
.............................
.............................
#if(TYPE==£CPLT);

#if(~PFX==0 and ~°ctx);
PFX="use the following context to replay to the query, avoid unnecessary
comments.";
#end;

#if(~°ctx);
°qry=if(~PFX,PFX++crlf,£)++
"context:"++crlf++°ctx++crlf++
"query:"++crlf++qry++crlf;
#else;
°qry=if(~PFX,PFX++crlf,£)++qry
#end;

°qry="Current date and time are"..DT_TSDEC(CLOCK,£ALL)++crlf++°qry;

°fnc=if(SRCH(MODEL,"gpt",NULL,0 ),"\GPT\CHAT","\GPT\PROMPT");

rply=EXO(°fnc,QRY::°qry,MAXT::MAXT,MODEL::MODEL,?rply,?inf,?res)->res->INF;
#else;

°tTKN=if(SRCH(MODEL,"gpt-3",NULL,0 ),8000,16000);

°tsz=MAXT;
°in=TBL_inf(tblCht,£row);
°i=°in;
°fd=0;
#while(°i>0);
°e=tbl_itm(tblCht,£CONTENT,°i);
#if(!°fd);
°fd=if(°tsz+len(°e)>°tTKN,1,0);
°tsz+=len(°e);
#else_otif(°fd);
tbl_chg(tblCht,NULL,£DEL++°i);
#end;
°i-=1;
```

```
      #end;

      °in=TBL_CHG(tblCht,NULL,£ADD,£ROW);

      °t="Current date and time are"..DT_TSDEC(CLOCK,£ALL);
      #if(~PFX);
      °t=°t++crlf++PFX;
      #end;

      °t=°t++crlf++°ctx;

      TBl_ITM(tblCht,£ROLE,°in,"system");
      TBl_ITM(tblCht,£CONTENT,°in,°t);

      °in=TBL_CHG(tblCht,NULL,£ADD,£ROW);

      TBl_ITM(tblCht,£ROLE,°in,"user");
      TBl_ITM(tblCht,£CONTENT,°in,qry);

      rply=EXO("\GPT\CHAT",QRY::tblCht,MAXT::MAXT,MODEL::MODEL,?rply,?inf,?res)-
      >res->Inf;

      °in-=1;
      TBL_CHG(tblCht,NULL,£DEL++°in,£ROW);

      #if(res==£OK);
      °in=tbl_chg(tblCht,NULL,£ADD,£ROW);
      TBl_ITM(tblCht,£ROLE,°in,"assistant");
      TBl_ITM(tblCht,£CONTENT,°in,rply);
      #else;
      tbl_chg(tblCht,NULL,£DEL++°in);
      #end;
      #end;
      ..............................
   ========================================
================================================================================
```

(EXO) Main

Entry point you can change by editing EXO@\pwk\EXECUTOR

```
-------------------------------------------------
•PORT_DBG: 4704  (ISP port)
•runflg:
•TVCT:   (Vector Data)
•TVCTQ:   (Vector Query)
•TvctR:   (Vector Reply)
•FILTRO:
•KbC:   (KB credential)
•EPT:
•KEY:
•ORG:
•MODEL:   (TBL modelli)
-------------------------------------------------
   ..............................
   runflg=1;
   KBC=KB1_OPN("MyGPT.kb1");
   ..............................
   ..............................
   EPT=KB1_QRY(KBC,"\OAI\EPT",£TEXT);
   KEY=KB1_QRY(KBC,"\OAI\KEY",£TEXT);
```

```
   ORG=KB1_QRY(KBC,"\OAI\ORG",£TEXT);
   ..............................
   (IF) if
   ------------------------------------------------
   •_COND_:
   ------------------------------------------------
      ..............................
      _COND_= ~KEY==0 or ~EPT==0 ;
      ..............................
      (THEN) then
         (EXEC) \CRD\Starter
            (SET) set
               ..............................
               mode=£EXEC;   !!EXEC,THREAD;
               par_nId=£;
               par_Gui=£;
               ..............................
   (EXEC) \CONFIG\INIT
   (EXEC) \GPT\INIT
      (SET) set
         ..............................
         EPT=EPT;   !!End point;
         Key=KEY;   !!Key;
         ORG=ORG;
         ..............................
   (EXEC) \GPT\MODEL
      (GET) get
         ..............................
         MODEL=RPLY;
         !!=RES;   !!£OK, £ERR;
         !!=INF;   !!HTTP res;
         ..............................
   (EXEC) \AI_DB\OPEN
      (SET) set
         ..............................
         NAME="GPT_PDF";
         VDIM=1536;   !!Vector DIM;
         VMAX=60000;   !!Vector max size;
         ..............................
      (GET) get
         ..............................
         TVCT=TVCT;   !!Tabella con due colonne;
         ..............................
   (EXEC) \AI_CORE\INIT
   (EXEC) \MG\Starter
      (SET) set
         ..............................
         mode=£EXEC;   !!EXEC,THREAD;
         par_nId=£;
         par_Gui=£;
         ..............................
================================================================================
```