```
0001  *
0002  ************************************************************************
0003  * Copyright (c) 2004 Zumasys, Inc. as an unpublished work. Permission is *
0004  * hereby granted, free of charge, to any person obtaining a copy of this *
0005  * software, to use the software without restriction, including without   *
0006  * limitation the rights to use, copy, modify, merge, publish or          *
0007  * distribute the software, and to permit persons to whom the software is *
0008  * furnished to do so, subject to the following conditions: This          *
0009  * copyright notice and permission notice shall be included in all copies *
0010  * or substantial portions of the software. THE SOFTWARE IS PROVIDED "AS  *
0011  * IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT   *
0012  * NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A        *
0013  * PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE          *
0014  * COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, *
0015  * WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT *
0016  * OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN  *
0017  * THE SOFTWARE.                                                          *
0018  ************************************************************************
0019  ************************************************************************
0020  *
0021  * USER INTERFACE EXAMPLE PROGRAM 3
0022  *
0023  ************************************************************************
0024  ************************************************************************
0025  *
0026  * This example illustrates a character-based interface which uses AccuTerm
0027  * Visual Styles to display screen elements using a Windows-like look (if
0028  * available). In this example, Visual Styles (border effects and colors)
0029  * are associated with certain display attributes. The program will run
0030  * without Visual Styles if used with dumb terminals, older versions of
0031  * AccuTerm or other terminal emulators. This example incorporates AccuTerm
0032  * Imaging to display a picture associated with each customer record (if
0033  * available).
0034  *
0035  ************************************************************************
0036  *
0037  * The CUST.REC INCLUDE item declares the CUST.REC array, which is used to
0038  * store a working copy of the current data record. It also defines the
0039  * record layout by equating field names to array positions in the CUST.REC
0040  * array.
0041  *
0042  $INCLUDE UIEX.CUST.REC
0043  *
0044  ************************************************************************
0045  *
0046  * EQUates for control characters and delimiters and other constants
0047  *
0048  EQU VM TO CHAR(253)
0049  EQU BEL TO CHAR(7)
0050  EQU ESC TO CHAR(27)
0051  EQU STX TO CHAR(2)
0052  EQU CR TO CHAR(13)
0053  EQU FALSE TO 0
0054  EQU TRUE TO 1
0055  *
0056  ************************************************************************
0057  *
0058  * This program uses AccuTerm Imaging to display pictures associated with
0059  * records in the customer file. This EQUate defines the path where the
0060  * image files are located.
0061  *
```

```
0062  $INCLUDE UIEX.PIX.PATH
0063  *
0064  * Check for image support (no images for dumb terminal or AccuTerm Lite)
0065  *
0066  CALL FTVSINF('','','',CAPABILITIES,'','','','')
0067  IF INDEX(CAPABILITIES,'I',1) THEN IMGFLG = TRUE ELSE IMGFLG = FALSE
0068  *
0069  ************************************************************************
0070  *
0071  * Open our files...
0072  *
0073  OPEN 'CUST.SAMPLE' TO FN.CUST ELSE PRINT 'NO CUST.SAMPLE FILE';STOP
0074  OPEN 'CUST.SAMPLE.CTRL' TO FN.CUST.CTRL ELSE PRINT 'NO CUST.SAMPLE.CTRL FILE';STOP
0075  OPEN 'CUST.SAMPLE.XREF' TO FN.CUST.XREF ELSE PRINT 'NO CUST.SAMPLE.XREF';STOP
0076  *
0077  ************************************************************************
0078  *
0079  * Define the data field control structures. NUMFLDS is the number of
0080  * fields. The CONTROL array defines the field control elements such as
0081  * field label, label position, field position and size, and field prompt.
0082  * The IDATA array stores the current field values, and is initialized from
0083  * the CUST.REC array when a data record is read from the file. When the
0084  * record is updated, values are copied from the IDATA array to the
0085  * CUST.REC array and the CUST.REC array is passed to the CUST.UPDATE
0086  * subroutine to update the data and index files (a separate subroutine is
0087  * used to update the file since the update process may handle other
0088  * functions like updating indexes).
0089  *
0090  NUMFLDS = 12
0091  DIM CONTROL(12)
0092  DIM IDATA(12)
0093  *
0094  * Define field control elements
0095  *  value 1: field label text
0096  *  value 2: field label column
0097  *  value 3: field label row
0098  *  value 4: field label width (0 for actual width, no padding)
0099  *  value 5: field data column
0100  *  value 6: field data row
0101  *  value 7: field data width
0102  *  value 8: field data height
0103  *  value 9: field prompt message
0104  CONTROL(1) = 'Customer ID???0?8??0?]Enter the ID, or name to search for, or N for
      next number'
0105  CONTROL(2) = 'Contact???0?8??0?]Enter the contact name'
0106  CONTROL(3) = 'Company name???0?8??0?]Enter the company name'
0107  CONTROL(4) = 'Address line 1???0?8??0?]Enter address line 1'
0108  CONTROL(5) = 'Address line 2???0?8??0?]Enter address line 2'
0109  CONTROL(6) = 'City???0?8??5?]Enter the city'
0110  CONTROL(7) = 'State or province???0?8??5?]Enter the state or province abbreviation'
0111  CONTROL(8) = 'Zip/postal code??0?0?8?0?0?]Enter the zip or postal code'
0112  CONTROL(9) = 'Country??1?0?8?1?8?]Enter the country'
0113  CONTROL(10) = 'Phone???2?0?8?2?5?]Enter the phone number'
0114  CONTROL(11) = 'Fax???3?0?8?3?5?]Enter the fax number'
0115  CONTROL(12) = 'Notes??4?0?8?4?0?]Enter any notes about this customer'
0116  *
0117  ************************************************************************
0118  *
0119  * Define some screen control strings for prompts & errors
0120  *
0121  PROMPT ''
```

```
0122  CLR = @(-1)         ;* Clear entire screen
0123  CEOL = @(-4)        ;* Clear to end of line
0124  PL = @(5,22):CEOL   ;* Prompt line
0125  EL = @(5,23):CEOL   ;* Error line
0126  *
0127  * Wyse 60 attributes for NORMAL, REVERSE, DIM REVERSE and UNDERLINE
0128  * REVERSE (we like Wyse 60 because the attributes dont take any space on
0129  * the screen, and more than one attribute can be displayed at one time).
0130  * If running AccuTerm in Viewpoint A2 Enhanced emulation, you can use the
0131  * ADDS 4000 attributes instead. Just change the upper-case "G" below to
0132  * lower-case "g".
0133  *
0134  NORMAL = ESC:'G0'  ;* Normal - display headings & field labels
0135  REVERSE = ESC:'G4' ;* Reverse - display active field data
0136  DIMREV = ESC:'Gt'  ;* Dim Reverse - display inactive field data
0137  UNDREV = ESC:'G<'  ;* Underline Reverse - display warnings
0138  *
0139  ************************************************************************
0140  *
0141  * This program processes one customer record at a time, and is organized
0142  * using three nested loops. The outermost loop (the RECORD loop) is
0143  * executed once for each record accessed. The loop repeats until the XIT
0144  * control variable is set to TRUE.
0145  *
0146  XIT = FALSE
0147  LOOP UNTIL XIT DO
0148   *
0149   ***********************************************************************
0150   *
0151   * Before prompting for the customer ID, reset the internal data array
0152   * (IDATA) and CUST.ID variables, then clear the screen and display the
0153   * heading and field labels.
0154   *
0155   GOSUB RESTART
0156   GOSUB DSPSCRN
0157   *
0158   ***********************************************************************
0159   *
0160   * The middle loop is the ACTION loop. It is executed for the current
0161   * record until the user performs an action that terminates processing of
0162   * that record, such as exiting, cancelling, saving or deleting the
0163   * record. The field number to begin prompting (NXTFLD) is initialized to
0164   * 1, causing the ID field to be prompted first. The loop immediately
0165   * enters the PROMPT loop, followed by the field modification prompt. The
0166   * loop repeats until the DONE control variable is set to TRUE.
0167   *
0168   NXTFLD = 1
0169   DONE = FALSE
0170   LOOP
0171    *
0172    **********************************************************************
0173    *
0174    * The inner loop is the PROMPT loop. It is executed for each prompt
0175    * field as specified by the NXTFLD variable. The prompt loop simply
0176    * calls the local INPUT.FIELD subroutine which performs field input,
0177    * data validation and keyboard command decoding. The FIELD loop repeats
0178    * until the field number is greater than the number of fields, or until
0179    * the DONE control variable is set to TRUE. The DONE control variable
0180    * may be set to TRUE in the INPUT.FIELD subroutine, if the user enters a
0181    * NULL to for the item-ID.
0182    *
```

```
0183    LOOP
0184     CURFLD = NXTFLD
0185    UNTIL DONE OR CURFLD > NUMFLDS DO
0186     *
0187     ************************************************************************
0188     *
0189     * Prompt for the next field. Update the NXTFLD variable with the field
0190     * number to prompt next.
0191     *
0192     GOSUB INPUT.FIELD
0193    REPEAT ;* end of PROMPT loop
0194    *
0195    ************************************************************************
0196    *
0197    * If the FIELD loop exited with the DONE control variable set to TRUE,
0198    * bypass the modification prompt because no action is required.
0199    * Otherwise, prompt for which field to modify, or other actions such as
0200    * save or delete.
0201    *
0202    IF NOT(DONE) THEN
0203     *
0204     NXTFLD = NUMFLDS + 1 ;* assume we need to reprompt for action
0205     *
0206     PRINT PL:'Enter field number to modify or FI to save, DE to delete, EX to exit: ':
0207     INPUT ANS:
0208     PRINT EL:
0209     *
0210     ************************************************************************
0211     *
0212     * Decode the response
0213     *
0214     ANS = OCONV(ANS, 'MCU')
0215     BEGIN CASE
0216      CASE NUM(ANS) AND ANS >= 1 AND ANS <= NUMFLDS; NXTFLD = ANS
0217      CASE ANS EQ 'FI'; GOSUB SAVE.RECORD
0218      CASE ANS EQ 'DE'; GOSUB DELETE.RECORD
0219      CASE ANS EQ 'EX' OR ANS EQ ''; GOSUB CHECK.EXIT
0220     END CASE
0221     *
0222    END
0223    *
0224   UNTIL DONE DO REPEAT ;* end of ACTION loop
0225   *
0226  REPEAT ;* end of RECORD loop
0227  *
0228  ************************************************************************
0229  *
0230  * All done - clear the screen and exit!
0231  PRINT CLR:
0232  STOP
0233  *
0234  *
0235  ************************************************************************
0236  ************************************************************************
0237  * LOCAL SUBROUTINES
0238  ************************************************************************
0239  ************************************************************************
0240  *
0241  *
0242  ************************************************************************
0243  *
```

```
0244  * The INPUT.FIELD subroutine is the main prompting routine. This routine
0245  * displays the prompt string (from the CONTROL array), and enters a loop,
0246  * prompting for a specified field (CURFLD) and setting the next field
0247  * variable (NXTFLD) appropriately. The loop repeats until the NXTFLD
0248  * (initially set to CURFLD) changes. This causes the field prompt to be
0249  * repeated in case invalid data is entered (illegal customer ID, etc.) If
0250  * a NULL is entered for the ID field, and there is no current record, the
0251  * ACTION loop control variable, DONE, and the RECORD loop control
0252  * variable, XIT, are set to TRUE, and this routine exits.
0253  *
0254  INPUT.FIELD:
0255  *
0256  ************************************************************************
0257  *
0258  * Display the prompt message
0259  *
0260  PRINT PL:CONTROL(CURFLD)<1,9>:
0261  *
0262  ************************************************************************
0263  *
0264  * Initialize current value, next field
0265  *
0266  PREVAL = IDATA(CURFLD) ;* Save previous field value to detect change in value
0267  *
0268  ************************************************************************
0269  *
0270  * Prompt for this field until NXTFLD variable is updated
0271  *
0272  LOOP
0273   *
0274   ***********************************************************************
0275   *
0276   * Assume next field number is next sequential field
0277   *
0278   NXTFLD = CURFLD + 1
0279   *
0280   * Highlight current field data
0281   *
0282   XLINE = CURFLD  ;* Set the field number to display
0283   ACTIVE = CURFLD ;* Set the active field number to highlight field
0284   GOSUB DSPLINE
0285   *
0286   * Prompt for input
0287   *
0288   PRINT @(CONTROL(CURFLD)<1,5>,CONTROL(CURFLD)<1,6>):REVERSE:
0289   INPUT IDATA(CURFLD):
0290   *
0291   * Check for NULL
0292   *
0293   K = LEN(IDATA(CURFLD))
0294   IF K = 0 THEN
0295    *
0296    * No change if NULL entered
0297    *
0298    IDATA(CURFLD) = PREVAL
0299   END ELSE
0300    *
0301    * Erase old data
0302    *
0303    IF K < CONTROL(CURFLD)<1,7> THEN
0304     PRINT @(K+CONTROL(CURFLD)<1,5>,CONTROL(CURFLD)<1,6>):SPACE(CONTROL(CURFLD)<1,7> -
```

```
     K):
0305    END
0306   END
0307   *
0308   * Reset display attribute
0309   *
0310   PRINT NORMAL:
0311   *
0312   * Clear the error line
0313   *
0314   PRINT EL:
0315   *
0316   ************************************************************************
0317   *
0318   * Check for any special values (like 'END' or 'EXIT')
0319   *
0320   IF OCONV(IDATA(CURFLD),'MCU') EQ 'END' THEN
0321    IDATA(CURFLD) = PREVAL ;* Restore previous value
0322    GOSUB CHECK.ABANDON ;* Ensure OK to loose changes
0323    IF OK THEN
0324     *
0325     ********************************************************************
0326     *
0327     * No changes, or user OKs abandoning them, so we are outa here!
0328     *
0329     DONE = TRUE
0330     XIT = TRUE
0331     RETURN
0332     *
0333    END ELSE
0334     *
0335     ********************************************************************
0336     *
0337     * User does not want to abandon changes, so reprompt
0338     *
0339     NXTFLD = CURFLD ;* Reprompt
0340     *
0341    END
0342   END
0343   IF IDATA(CURFLD) EQ SPACE(LEN(IDATA(CURFLD))) THEN IDATA(CURFLD) = ''
0344   *
0345   ************************************************************************
0346   *
0347   * Peform field data validation if next field number changed
0348   *
0349   IF NXTFLD NE CURFLD THEN
0350    BEGIN CASE
0351     *
0352     CASE CURFLD EQ 1
0353      *
0354      ****************************************************************
0355      *
0356      * Validate the ID field. If NULL, quit. If changed, read new record.
0357      *
0358      IF CUST.ID EQ '' AND IDATA(CURFLD) EQ '' THEN
0359       DONE = TRUE
0360       XIT = TRUE
0361       RETURN
0362      END
0363      *
0364      IF IDATA(CURFLD) NE PREVAL THEN
```

```
0365       ID = IDATA(CURFLD)
0366       GOSUB CHECK.ID ;* Check the newly entered ID handling index lookups
0367       IF OK THEN
0368        *
0369        ************************************************************
0370        *
0371        * New ID (or result of index lookup) is good!
0372        *
0373        IDATA(CURFLD) = ID ;* Update the current field value in case of index lookup
0374        *
0375       END ELSE
0376        *
0377        ************************************************************
0378        *
0379        * New ID is invalid
0380        *
0381        IDATA(CURFLD) = PREVAL ;* Restore previous value
0382        NXTFLD = CURFLD ;* Reprompt
0383        *
0384       END
0385      END
0386      *
0387    END CASE
0388   END
0389   *
0390  WHILE CURFLD EQ NXTFLD DO REPEAT
0391  *
0392  * Redisplay the field data using the inactive highlighting
0393  *
0394  XLINE = CURFLD
0395  ACTIVE = 0
0396  GOSUB DSPLINE
0397  *
0398  RETURN
0399  *
0400  *
0401  ************************************************************************
0402  *
0403  * The CHECK.EXIT subroutine checks if any field data has changed, and
0404  * prompts if the user wants to abandon changes. If no changes, or the user
0405  * decides to abandon the changes, the DONE and XIT loop control variables
0406  * are set to TRUE, causing all three loops to terminate, and the program
0407  * itself to exit.
0408  *
0409  CHECK.EXIT: *
0410  *
0411  GOSUB CHECK.ABANDON
0412  IF OK THEN
0413   DONE = TRUE
0414   XIT = TRUE
0415   GOSUB HIDEPIX
0416  END
0417  RETURN
0418  *
0419  *
0420  ************************************************************************
0421  *
0422  * The CHECK.ID subroutine validates a newly entered item-ID. If the
0423  * current record has unsaved changes, the user is prompted to abandon the
0424  * changes. If no changes, or the user abandons the changes, and the new ID
0425  * is not NULL, an attempt is made to read a record using the new ID. If
```

```
0426  * the read is not successful, the ID is assumed to be a search string, and
0427  * the search subroutine is called to select an ID based on the search
0428  * string. If a valid ID is returned (or if one was initially entered), the
0429  * new record data is displayed. If the new ID is null, or if the search
0430  * routine did not return a valid ID, a warning message is displayed and
0431  * the OK indicator variable is set to FALSE. Otherwise it is set to TRUE.
0432  *
0433  CHECK.ID: *
0434  *
0435  ************************************************************************
0436  *
0437  * Make sure we don't have any unsaved data before changing the ID
0438  *
0439  GOSUB CHECK.ABANDON
0440  IF NOT(OK) THEN RETURN ;* Reprompt for the ID
0441  IF ID EQ '' THEN
0442   OK = FALSE ;* NULL is not a valid ID!
0443  END ELSE
0444   *
0445   ***********************************************************************
0446   *
0447   * Check if user wants new ID
0448   *
0449   IF ID EQ 'N' OR ID EQ 'n' THEN
0450    * Get next sequential ID
0451    READVU ID FROM FN.CUST.CTRL,'NEXT',2 THEN
0452     WRITEV ID + 1 ON FN.CUST.CTRL,'NEXT',2
0453     GOSUB RESTART
0454     IDATA(1) = ID
0455    END ELSE
0456     PRINT EL:UNDREV:'Next item counter record not found!':NORMAL:BEL:
0457     INPUT ANS:
0458     PRINT EL:
0459     OK = FALSE
0460     RETURN
0461    END
0462   END ELSE
0463    *
0464    ***********************************************************************
0465    *
0466    * Try to read the customer record from the entered ID
0467    *
0468    GOSUB HIDEPIX ;* Hide previous cust picture before reading new one
0469    GOSUB READ.RECORD
0470    IF NOT(OK) THEN
0471     *
0472     ***********************************************************************
0473     *
0474     * The attempt to read a record failed - assume the ID is a search string
0475     *
0476     CALL UIEX.GET.CUST.IDX(XID,ID,FN.CUST.XREF,FN.CUST)
0477     GOSUB DSPSCRN ;* Refresh the screen after index lookup
0478     *
0479     ***********************************************************************
0480     *
0481     * If the user did not select an item in the search routine, reprompt
0482     *
0483     IF XID EQ '' THEN RETURN
0484     *
0485     ***********************************************************************
0486     *
```

```
0487      * Try to read the customer record from the selected ID
0488      *
0489      GOSUB HIDEPIX ;* Hide previous cust picture before reading new one
0490      ID = XID
0491      GOSUB READ.RECORD
0492      *
0493     END
0494    END
0495   END
0496   *
0497   ************************************************************************
0498   *
0499   * If success, display the new record, otherwise show warning message
0500   *
0501   IF OK THEN
0502    GOSUB DSPDATA ;* Display new record
0503   END ELSE
0504    PRINT EL:UNDREV:'Please enter a valid customer ID!':NORMAL:BEL:
0505   END
0506   RETURN
0507   *
0508   *
0509   ************************************************************************
0510   *
0511   * The READ.RECORD subroutine reads a new customer record from the file and
0512   * initializes the internal field data array (IDATA) from the record array
0513   * (CUST.REC). If the record does not exist, the routine returns with the
0514   * OK indicator variable set to FALSE. Otherwise OK is set to TRUE, and the
0515   * CUST.ID variable is set to the new ID.
0516   *
0517   READ.RECORD: *
0518   *
0519   MATREAD CUST.REC FROM FN.CUST,ID THEN
0520    CUST.ID = ID
0521    IDATA(1) = CUST.ID
0522    IDATA(2) = CUST.CONTACT
0523    IDATA(3) = CUST.NAME
0524    IDATA(4) = CUST.ADDRESS1
0525    IDATA(5) = CUST.ADDRESS2
0526    IDATA(6) = CUST.CITY
0527    IDATA(7) = CUST.ST
0528    IDATA(8) = CUST.ZIP
0529    IDATA(9) = CUST.COUNTRY
0530    IDATA(10) = CUST.PHONE
0531    IDATA(11) = CUST.FAX
0532    IDATA(12) = CUST.HISTORY
0533    OK = TRUE ;* Set the SUCCESS indicator
0534   END ELSE
0535    OK = FALSE ;* Set the FAILURE indicator
0536   END
0537   RETURN
0538   *
0539   *
0540   ************************************************************************
0541   *
0542   * The DELETE.RECORD subroutine confirms that the user intends to delete
0543   * the current record. If the action is confirmed, the CUST.DELETE
0544   * subroutine is called to perform the deletion. A separate subroutine is
0545   * used to handle updating indexes, etc.
0546   *
0547   DELETE.RECORD: *
```

```
0548  *
0549  IF CUST.ID NE '' THEN
0550   PRINT EL:UNDREV:'Are you sure you want to delete this customer? ':NORMAL:
0551   INPUT ANS:
0552   IF ANS[1,1] EQ 'Y' OR ANS[1,1] EQ 'y' THEN
0553    * Deletion has been confirmed - do the delete
0554    OK = TRUE ;* Set the SUCCESS indicator
0555    CALL UIEX.CUST.DELETE(CUST.ID,FN.CUST,FN.CUST.XREF)
0556    DONE = TRUE ;* proceed to next record
0557    GOSUB HIDEPIX ;* erase picture
0558   END ELSE
0559    OK = FALSE ;* Set the FAILURE indicator
0560   END
0561  END
0562  RETURN
0563  *
0564  *
0565  ************************************************************************
0566  *
0567  * The SAVE.RECORD subroutine copies internal field data from the IDATA
0568  * array to the customer record array (CUST.REC). The CUST.UPDATE
0569  * subroutine is called to perform the update. A separate subroutine is
0570  * used to handle updating indexes, etc.
0571  *
0572  SAVE.RECORD: *
0573  *
0574  IF IDATA(1) NE '' THEN
0575   * Copy data from the internal field data array (IDATA) to the CUST.REC array
0576   CUST.ID = IDATA(1)
0577   CUST.CONTACT = IDATA(2)
0578   CUST.NAME = IDATA(3)
0579   CUST.ADDRESS1 = IDATA(4)
0580   CUST.ADDRESS2 = IDATA(5)
0581   CUST.CITY = IDATA(6)
0582   CUST.ST = IDATA(7)
0583   CUST.ZIP = IDATA(8)
0584   CUST.COUNTRY = IDATA(9)
0585   CUST.PHONE = IDATA(10)
0586   CUST.FAX = IDATA(11)
0587   CUST.HISTORY = IDATA(12)
0588   * Update the file
0589   CALL UIEX.CUST.UPDATE(CUST.ID,MAT CUST.REC,FN.CUST,FN.CUST.XREF)
0590   OK = TRUE ;* Set the SUCCESS indicator
0591  END ELSE
0592   OK = FALSE ;* Set the FAILURE indicator
0593  END
0594  DONE = TRUE ;* proceed to next record
0595  RETURN
0596  *
0597  *
0598  ************************************************************************
0599  *
0600  * The RESTART subroutine prepares the internal field data array (IDATA),
0601  * customer record array (CUST.REC) and ID for a new customer record.
0602  *
0603  RESTART: *
0604  *
0605  CUST.ID = ''
0606  MAT CUST.REC = ''
0607  MAT IDATA = ''
0608  ACTIVE = 0
```

```
0609  RETURN
0610  *
0611  *
0612  ************************************************************************
0613  *
0614  * The CHECK.CHANGED subroutine checks if any internal field data has been
0615  * changed. The OK indicator variable is set to FALSE if any data is
0616  * changed, otherwise it is set to TRUE. The ID field is not checked, since
0617  * it is appropriately handled by the CHECK.ID subroutine.
0618  *
0619  CHECK.CHANGED: *
0620  *
0621  OK = FALSE
0622  IF CUST.CONTACT # IDATA(2) THEN RETURN
0623  IF CUST.NAME # IDATA(3) THEN RETURN
0624  IF CUST.ADDRESS1 # IDATA(4) THEN RETURN
0625  IF CUST.ADDRESS2 # IDATA(5) THEN RETURN
0626  IF CUST.CITY # IDATA(6) THEN RETURN
0627  IF CUST.ST # IDATA(7) THEN RETURN
0628  IF CUST.ZIP # IDATA(8) THEN RETURN
0629  IF CUST.COUNTRY # IDATA(9) THEN RETURN
0630  IF CUST.PHONE # IDATA(10) THEN RETURN
0631  IF CUST.FAX # IDATA(11) THEN RETURN
0632  IF CUST.HISTORY # IDATA(12) THEN RETURN
0633  OK = TRUE
0634  RETURN
0635  *
0636  *
0637  ************************************************************************
0638  *
0639  * The CHECK.ABANDON subroutine calls CHECK.CHANGED. If any changes are
0640  * found, a message is displayed and the user is prompted to abandon the
0641  * changes. If there are no changes, or if the user decides to abandon
0642  * changes, the OK indicator variable is set to TRUE. Otherwise it is set
0643  * to FALSE.
0644  *
0645  CHECK.ABANDON: *
0646  *
0647  GOSUB CHECK.CHANGED
0648  IF NOT(OK) THEN
0649   PRINT EL:UNDREV:'Do you want to abandon all your changes? ':NORMAL:BEL:
0650   INPUT ANS:
0651   IF ANS[1,1] EQ 'Y' OR ANS[1,1] EQ 'y' THEN OK = 1
0652   PRINT EL:
0653  END
0654  RETURN
0655  *
0656  *
0657  ************************************************************************
0658  *
0659  * The DSPSCRN subroutine is used to refresh the entire screen.
0660  *
0661  DSPSCRN: *
0662  *
0663  * Clear the screen
0664  PRINT CLR:NORMAL:
0665  *
0666  * If running AccuTerm, display the logo
0667  IF IMGFLG THEN
0668   PRINT ESC:STX:'iL,':PIX.PATH:'ASE-LOGO-SMALL.JPG,65,0,12,2,0,N':CR:
0669  END
```

```
0670  *
0671  * Display heading & labels
0672  PRINT @(5,0):'Customer File Maintenance':
0673  FOR XLINE = 1 TO NUMFLDS
0674   LBWD = CONTROL(XLINE)<1,4>
0675   LBTX = (XLINE 'L#2 ') : CONTROL(XLINE)<1,1> ;* Prepend line number to label
0676   IF LBWD > 0 THEN LBTX = (LBTX : STR('.',LBWD))[1,LBWD] ;* Pad label with dots
0677   PRINT @(CONTROL(XLINE)<1,2>,CONTROL(XLINE)<1,3>):LBTX:
0678  NEXT XLINE
0679  *
0680  * Display the field data
0681  GOSUB DSPDATA
0682  RETURN
0683  *
0684  *
0685  ************************************************************************
0686  *
0687  * The DSPDATA subroutine is used to refresh the field data for all fields.
0688  *
0689  DSPDATA: *
0690  *
0691  FOR XLINE = 1 TO NUMFLDS
0692   GOSUB DSPLINE
0693  NEXT XLINE
0694  GOSUB SHOWPIX
0695  RETURN
0696  *
0697  *
0698  ************************************************************************
0699  *
0700  * The DSPLINE subroutine is used to refresh the field data for one field.
0701  * The field to be refreshed is specified by the XLINE variable. If the
0702  * field is active (XLINE = ACTIVE), then the field data is displayed using
0703  * the REVERSE display attribute. Otherwise it is displayed using the
0704  * DIMREV display attribute.
0705  *
0706  DSPLINE: *
0707  *
0708  MSK = 'L#':CONTROL(XLINE)<1,7>
0709  PRINT @(CONTROL(XLINE)<1,5>,CONTROL(XLINE)<1,6>):
0710  IF XLINE EQ ACTIVE THEN
0711   PRINT REVERSE:
0712  END ELSE
0713   PRINT DIMREV:
0714  END
0715  PRINT IDATA(XLINE) MSK:
0716  PRINT NORMAL:
0717  RETURN
0718  *
0719  *
0720  ************************************************************************
0721  *
0722  * If running AccuTerm and an customer ID has been entered, try to display
0723  * the customer's picture for the current record.
0724  *
0725  SHOWPIX: *
0726  *
0727  IF IMGFLG THEN
0728   IF CUST.ID NE '' THEN
0729    PRINT ESC:STX:'iL,':PIX.PATH:'CUST\':CUST.ID:'.JPG,60,3,15,8,1,B':CR:
0730   END
```

```
0731  END
0732  RETURN
0733  *
0734  *
0735  ************************************************************************
0736  *
0737  * Erase the current customer picture before calling the lookup screen
0738  * because pictures always display over text.
0739  *
0740  HIDEPIX: *
0741  *
0742  IF IMGFLG THEN
0743   IF CUST.ID NE '' THEN
0744    PRINT ESC:STX:'iD,':PIX.PATH:'CUST\':CUST.ID:'.JPG':CR:
0745   END
0746  END
0747  RETURN
0748  *
0749  END
0750
```