```
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0003  * Copyright (c) 2004 Zumasys, Inc. as an unpublished work.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 5
0022  *
0023  ************************************************************************
0024  ************************************************************************
0025  *
0026  * This example illustrates an advanced character-based user interface
0027  * using the Zumasys Smart User Interface library. All display and input
0028  * processing is handled by the SUI subroutines, rather than using PRINT
0029  * and INPUT statements. By using the SUI library, all input fields accept
0030  * cursor keys, editing keys, function keys and mouse clicks. The SUI
0031  * subroutines also utilize AccuTerm Visual Styles to display screen
0032  * elements using a Windows-like look (if avaliable).
0033  *
0034  ************************************************************************
0035  *
0036  * To use the SUI library, you must include SUI.TERMDEF. This INCLUDE item
0037  * contains EQUates for various terminal functions like visual attributes
0038  * and line graphics, as well as constants for the keyboard commands like
0039  * TERM.ENTER$ or TERM.PGUP$.
0040  *
0041  $INCLUDE SUIBP SUI.TERMDEF
0042  *
0043  ************************************************************************
0044  *
0045  * The CUST.REC INCLUDE item declares the CUST.REC array, which is used to
0046  * store a working copy of the current data record. It also defines the
0047  * record layout by equating field names to array positions in the CUST.REC
0048  * array.
0049  *
0050  $INCLUDE UIEX.CUST.REC
0051  *
0052  ************************************************************************
0053  *
0054  * EQUates for control characters and delimiters and other constants
0055  *
0056  EQU VM TO CHAR(253)
0057  EQU BEL TO CHAR(7)
0058  EQU ESC TO CHAR(27)
0059  EQU STX TO CHAR(2)
0060  EQU CR TO CHAR(13)
0061  EQU FALSE TO 0
```

```
0062  EQU TRUE TO 1
0063  *
0064  ************************************************************************
0065  *
0066  * Any routine that uses the SUI library must call SUI.GET.TERM to
0067  * initialize the TermDef array. The TermDef array is a required argument
0068  * for all of the SUI subroutines. It is much better to call this routine
0069  * one time and pass the TermDef array between various routines instead of
0070  * calling it in each application subroutine that uses SUI.
0071  *
0072  CALL SUI.GET.TERM(MAT TermDef)
0073  *
0074  ************************************************************************
0075  *
0076  * Once the TermDef array is initialized by SUI.GET.TERM, the TERM.RESET
0077  * string is sent to the terminal to initialize the terminal state. The
0078  * reset string programs necessary function keys and initalizes any other
0079  * terminal state as required for each particular terminal type.
0080  *
0081  PRINT TERM.RESET:
0082  *
0083  ************************************************************************
0084  *
0085  * This program uses AccuTerm Imaging to display pictures associated with
0086  * records in the customer file. This EQUate defines the path where the
0087  * image files are located.
0088  *
0089  $INCLUDE UIEX.PIX.PATH
0090  *
0091  * Check for image support (no images if dumb terminal or AccuTerm Lite)
0092  *
0093  IF INDEX(TERM.SPECIAL,'I',1) THEN IMGFLG = TRUE ELSE IMGFLG = FALSE
0094  *
0095  ************************************************************************
0096  *
0097  * Open our files...
0098  *
0099  OPEN 'CUST.SAMPLE' TO FN.CUST ELSE PRINT 'NO CUST.SAMPLE FILE'; STOP
0100  OPEN 'CUST.SAMPLE.CTRL' TO FN.CUST.CTRL ELSE PRINT 'NO CUST.SAMPLE.CTRL FILE'; STOP
0101  OPEN 'CUST.SAMPLE.XREF' TO FN.CUST.XREF ELSE PRINT 'NO CUST.SAMPLE.XREF'; STOP
0102  *
0103  ************************************************************************
0104  *
0105  * Define the data field control structures. NUMFLDS is the number of
0106  * fields. The CONTROL array defines the field control elements such as
0107  * field label, label position, field position and size, and field prompt.
0108  * The IDATA array stores the current field values, and is initialized from
0109  * the CUST.REC array when a data record is read from the file. When the
0110  * record is updated, values are copied from the IDATA array to the
0111  * CUST.REC array and the CUST.REC array is passed to the CUST.UPDATE
0112  * subroutine to update the data and index files (a separate subroutine is
0113  * used to update the file since the update process may handle other
0114  * functions like updating indexes).
0115  *
0116  NUMFLDS = 12
0117  DIM CONTROL(12)
0118  DIM IDATA(12)
0119  *
0120  * Define field control elements
0121  *  value 1: field label text
0122  *  value 2: field label column
```

```
0123  *  value 3: field label row
0124  *  value 4: field label width (0 for actual width, no padding)
0125  *  value 5: field data column
0126  *  value 6: field data row
0127  *  value 7: field data width
0128  *  value 8: field data height
0129  *  value 9: field prompt message
0130  CONTROL(1) = 'Customer ID]10]2]17]28]2]30]1]Enter the ID, or name to search for'
0131  CONTROL(2) = 'Contact]10]4]17]28]4]30]1]Enter the contact name'
0132  CONTROL(3) = 'Company name]10]5]17]28]5]30]1]Enter the company name'
0133  CONTROL(4) = 'Address line 1]10]6]17]28]6]30]1]Enter address line 1'
0134  CONTROL(5) = 'Address line 2]10]7]17]28]7]30]1]Enter address line 2'
0135  CONTROL(6) = 'City]10]8]17]28]8]25]1]Enter the city'
0136  CONTROL(7) = 'State or province]10]9]17]28]9]15]1]Enter the state or province
      abbreviation'
0137  CONTROL(8) = 'Zip/postal code]10]10]17]28]10]10]1]Enter the zip or postal code'
0138  CONTROL(9) = 'Country]10]11]17]28]11]18]1]Enter the country'
0139  CONTROL(10) = 'Phone]10]12]17]28]12]15]1]Enter the phone number'
0140  CONTROL(11) = 'Fax]10]13]17]28]13]15]1]Enter the fax number'
0141  CONTROL(12) = 'Notes]10]14]17]28]14]30]3]Enter any notes about this customer'
0142  *
0143  ************************************************************************
0144  *
0145  * A "command bar" is used to accept user action requests. The command bar
0146  * as displayed at the bottom of the screen, and appears as a row of
0147  * "buttons" which the user can click with the mouse. Each button has a
0148  * function key associated with it. The user can also use the cursor keys
0149  * to change the active button, and pressing the ENTER key or SPACE BAR
0150  * executes the action.
0151  *
0152  BTNTEXT =
      'F1=Help':VM:'F6=New':VM:'F5=Save':VM:'F4=Delete':VM:'F3=Cancel':VM:'F2=eXit'
0153  BTNSTATE = '0':VM:'0':VM:'2':VM:'2':VM:'0':VM:'0' ;* Initially, the Save and Delete
      button are disabled
0154  BTNKEY = 'H':VM:'N':VM:'S':VM:'D':VM:'C':VM:'X'
0155  BTN = 6 ;* Initial default button = EXIT
0156  *
0157  ************************************************************************
0158  *
0159  * This program processes one customer record at a time, and is organized
0160  * using three nested loops. The outermost loop (the RECORD loop) is
0161  * executed once for each record accessed. The loop repeats until the XIT
0162  * control variable is set to TRUE.
0163  *
0164  * Note: the term "terminal action" as used in the following comments means
0165  * an action that causes the current record or the program itself to
0166  * terminate. An action such as cancel, save, or exit is a terminal action.
0167  *
0168  XIT = FALSE
0169  LOOP UNTIL XIT DO
0170   *
0171   ***********************************************************************
0172   *
0173   * Before prompting for the customer ID, reset the internal data array
0174   * (IDATA) and CUST.ID variables, then clear the screen and display the
0175   * heading and field labels.
0176   *
0177   GOSUB RESTART
0178   GOSUB DSPSCRN
0179   *
0180   ***********************************************************************
```

```
0181   *
0182   * The middle loop is the ACTION loop. It is executed for the current
0183   * record until the user performs an action that terminates processing of
0184   * that record, such as exiting, cancelling, saving or deleting the
0185   * record. The field number to begin prompting (NXTFLD) is initialized to
0186   * 1, causing the ID field to be prompted first. The loop immediately
0187   * enters the PROMPT loop, followed by the action (command bar) prompt.
0188   * The loop repeats until the DONE control variable is set to TRUE.
0189   *
0190   NXTFLD = 1
0191   DONE = FALSE
0192   LOOP
0193    *
0194    ************************************************************************
0195    *
0196    * The inner loop is the PROMPT loop. It is executed for each prompt
0197    * field as specified by the NXTFLD variable. The prompt loop simply
0198    * calls the local INPUT.FIELD subroutine which performs field input,
0199    * data validation and keyboard command decoding. The FIELD loop repeats
0200    * until the field number is greater than the number of fields, or until
0201    * the DONE control variable is set to TRUE. The DONE control variable
0202    * may be set to TRUE in the INPUT.FIELD subroutine, if the user presses
0203    * a function key that causes a terminating action, such as exit, cancel,
0204    * save or delete.
0205    *
0206    LOOP
0207     CURFLD = NXTFLD
0208    UNTIL DONE OR CURFLD > NUMFLDS DO
0209     *
0210     ************************************************************************
0211     *
0212     * Prompt for the next field. Execute any action associated with a
0213     * function key (such as exit, cancel, save, delete). Update the NXTFLD
0214     * variable with the field number to prompt next, taking into account
0215     * cursor keys and mouse clicks.
0216     *
0217     GOSUB INPUT.FIELD
0218    REPEAT ;* end of PROMPT loop
0219    *
0220    ************************************************************************
0221    *
0222    * If the FIELD loop exited with a terminal action, the DONE control
0223    * variable will be set to TRUE. In this case, no action prompt is
0224    * necessary since the action has already been handled by the INPUT.FIELD
0225    * subroutine. Otherwise, prompt for actions using the command bar.
0226    *
0227    IF NOT(DONE) THEN
0228     *
0229     NXTFLD = NUMFLDS + 1 ;* assume we need to reprompt for action
0230     *
0231     ************************************************************************
0232     *
0233     * Use the SUI.INPUT.BTNBAR to prompt for actions using the command bar.
0234     * The SUI.INPUT.BTNBAR routine returns with CMD set to TERM.ENTER$ if
0235     * the user clicked a button, or pressed the ENTER key or SPACE BAR when
0236     * a button was active. In this case, the BTN argument returns the
0237     * number of the button.
0238     *
0239     CALL SUI.INPUT.BTNBAR(10, 22, 60, 1, BTNSTATE, BTNTEXT, BTNKEY, 'C', BTN, '',
     CMD, MAT TermDef)
0240     *
```

```
0241     ************************************************************************
0242     *
0243     * Decode the command
0244     *
0245     IF CMD EQ TERM.ENTER$ THEN
0246      *
0247      * Translate button number to equivalent function key
0248      BEGIN CASE
0249       CASE BTN = 1; CMD = TERM.F1$
0250       CASE BTN = 2; CMD = TERM.F6$
0251       CASE BTN = 3; CMD = TERM.F5$
0252       CASE BTN = 4; CMD = TERM.F4$
0253       CASE BTN = 5; CMD = TERM.F3$
0254       CASE BTN = 6; CMD = TERM.F2$
0255      END CASE
0256     END
0257     *
0258     * Decode function keys, BACK TAB key and mouse clicks
0259     BEGIN CASE
0260      CASE CMD EQ TERM.LEFTBUTTON$; GOSUB CHECK.MOUSE
0261      CASE CMD EQ TERM.F1$; GOSUB SHOW.HELP
0262      CASE CMD EQ TERM.F2$; GOSUB CHECK.EXIT
0263      CASE CMD EQ TERM.F3$; GOSUB CANCEL.RECORD
0264      CASE CMD EQ TERM.F4$; GOSUB DELETE.RECORD
0265      CASE CMD EQ TERM.F5$; GOSUB SAVE.RECORD
0266      CASE CMD EQ TERM.F6$; GOSUB NEW.RECORD
0267      CASE CMD EQ TERM.STAB$; NXTFLD = NUMFLDS
0268     END CASE
0269     *
0270    END
0271    *
0272   UNTIL DONE DO REPEAT ;* end of ACTION loop
0273   *
0274  REPEAT ;* end of RECORD loop
0275  *
0276  ************************************************************************
0277  *
0278  * All done - clear the screen, restore the cursor and exit!
0279  PRINT TERM.CLEAR:TERM.CSRON:
0280  STOP
0281  *
0282  *
0283  ************************************************************************
0284  ************************************************************************
0285  * LOCAL SUBROUTINES
0286  ************************************************************************
0287  ************************************************************************
0288  *
0289  *
0290  ************************************************************************
0291  *
0292  * The INPUT.FIELD subroutine is the main prompting routine. This routine
0293  * displays the prompt string (from the CONTROL array), and enters a loop,
0294  * prompting for a specified field (CURFLD) and setting the next field
0295  * variable (NXTFLD) appropriately. The loop repeats until the NXTFLD
0296  * (initially set to CURFLD) changes. This causes the field prompt to be
0297  * repeated in case unrecognized keys are pressed (for example, the PGUP
0298  * key), or the help key (F1) is pressed, or invalid data is entered
0299  * (illegal customer ID, etc.) If any recognized function key is pressed
0300  * that causes an action, and the action is terminal and successfully
0301  * executed (like exit, cancel, save, delete), the ACTION loop control
```

```
0302  * variable, DONE, is set to TRUE, and this routine exits. A mouse click is
0303  * handled by setting the NXTFLD variable to the field number that was
0304  * clicked, or to NUMFLDS + 1 if a the command bar was clicked.
0305  *
0306  INPUT.FIELD:
0307  *
0308  ************************************************************************
0309  *
0310  * Display the prompt message
0311  *
0312  CALL SUI.DISPLAY.LABEL(10, 20, 74, 'L', 0, CONTROL(CURFLD)<1,9>, CMD, MAT TermDef)
0313  *
0314  ************************************************************************
0315  *
0316  * Initialize current value, next field & character positions
0317  *
0318  PREVAL = IDATA(CURFLD) ;* Save previous field value to detect change in value
0319  NXTFLD = CURFLD ;* Assume field number not changed
0320  BEGPOS = 0 ;* Set initial text display character position
0321  CURPOS = 0 ;* Set initial text cursor character position
0322  *
0323  ************************************************************************
0324  *
0325  * Prompt for this field until NXTFLD variable is updated
0326  *
0327  LOOP
0328   *
0329   ***********************************************************************
0330   *
0331   * Prompt for input with full text editing functions
0332   *
0333   CALL SUI.INPUT.TEXT(CONTROL(CURFLD)<1,5>, CONTROL(CURFLD)<1,6>,
      CONTROL(CURFLD)<1,7>, CONTROL(CURFLD)<1,8>, '', 0, IDATA(CURFLD), BEGPOS, CURPOS,
      CMD, MAT TermDef)
0334   *
0335   ***********************************************************************
0336   *
0337   * Decode returned CMD
0338   *
0339   BEGIN CASE
0340    *
0341    ********************************************************************
0342    *
0343    * Check for ENTER, TAB, UP, DOWN or mouse click
0344    *
0345    CASE CMD EQ TERM.ENTER$ OR CMD EQ TERM.TAB$ OR CMD EQ TERM.DOWN$
0346     NXTFLD = CURFLD + 1; * Move to next field
0347    CASE CMD EQ TERM.STAB$ OR CMD EQ TERM.UP$
0348     IF CURFLD > 1 THEN NXTFLD = CURFLD - 1 ;* Move to previous field
0349    CASE CMD EQ TERM.LEFTBUTTON$
0350     GOSUB CHECK.MOUSE ;* Move to clicked-on field
0351     IF NXTFLD > NUMFLDS THEN RETURN ;* Clicked on a button
0352    *
0353    ********************************************************************
0354    *
0355    * Check for function keys
0356    *
0357    CASE CMD EQ TERM.F1$; GOSUB SHOW.HELP
0358    CASE CMD EQ TERM.F2$; GOSUB CHECK.EXIT
0359    CASE CMD EQ TERM.F3$; GOSUB CANCEL.RECORD
0360    CASE CMD EQ TERM.F4$; GOSUB DELETE.RECORD
```

```
0361    CASE CMD EQ TERM.F5$; GOSUB SAVE.RECORD
0362    CASE CMD EQ TERM.F6$; GOSUB NEW.RECORD; IF OK THEN RETURN
0363    CASE 1; * Unrecognized key - just reprompt
0364    *
0365   END CASE
0366   *
0367   ************************************************************************
0368   *
0369   * Check if a terminal action occurred (like exit, cancel, save, delete)
0370   *
0371   IF DONE THEN RETURN
0372   *
0373   ************************************************************************
0374   *
0375   * Check for any special values (like 'END' or 'EXIT')
0376   *
0377   IF OCONV(IDATA(CURFLD),'MCU') EQ 'END' THEN
0378    IDATA(CURFLD) = PREVAL ;* Restore previous value
0379    GOSUB CHECK.ABANDON ;* Ensure OK to loose changes
0380    IF OK THEN
0381     *
0382     ********************************************************************
0383     *
0384     * No changes, or user OKs abandoning them, so we are outa here!
0385     *
0386     DONE = TRUE
0387     XIT = TRUE
0388     RETURN
0389     *
0390    END ELSE
0391     *
0392     ********************************************************************
0393     *
0394     * User does not want to abandon changes, so refresh previous value &
0395     * reprompt
0396     *
0397     XLINE = CURFLD ;* Field number to refresh
0398     GOSUB DSPLINE ;* Redisplay the previous value
0399     NXTFLD = CURFLD ;* Reprompt
0400     *
0401    END
0402   END
0403   *
0404   ************************************************************************
0405   *
0406   * Peform field data validation if next field number changed
0407   *
0408   IF NXTFLD NE CURFLD THEN
0409    BEGIN CASE
0410     *
0411     CASE CURFLD EQ 1
0412      *
0413      ********************************************************************
0414      *
0415      * Validate the ID field. If changed, read new record.
0416      *
0417      IF CUST.ID EQ '' OR IDATA(CURFLD) NE PREVAL THEN
0418       ID = IDATA(CURFLD)
0419       GOSUB CHECK.ID ;* Check the newly entered ID handling index lookups
0420       IF OK THEN
0421        *
```

```
0422        ********************************************************************
0423        *
0424        * New ID (or result of index lookup) is good!
0425        *
0426        IDATA(CURFLD) = ID ;* Update the current field value in case of index lookup
0427        *
0428       END ELSE
0429        *
0430        ********************************************************************
0431        *
0432        * New ID is invalid
0433        *
0434        IDATA(CURFLD) = PREVAL ;* Restore previous value
0435        XLINE = CURFLD ;* Field number to refresh
0436        GOSUB DSPLINE ;* Redisplay the previous value
0437        NXTFLD = CURFLD ;* Reprompt
0438        *
0439       END
0440      END
0441      *
0442     CASE 1
0443      *
0444      ********************************************************************
0445      *
0446      * Validate other fields. If changed, set default button to "save".
0447      *
0448      IF PREVAL NE IDATA(CURFLD) THEN BTN = 3
0449      *
0450    END CASE
0451   END
0452   *
0453  WHILE CURFLD EQ NXTFLD DO REPEAT
0454  *
0455  RETURN
0456  *
0457  *
0458  ************************************************************************
0459  *
0460  * The SHOW.HELP subroutine displays help for the current field, if it is
0461  * available. Otherwise, a message indicating no help is available is
0462  * shown.
0463  *
0464  SHOW.HELP: *
0465  *
0466  CALL SUI.MESSAGE.BOX(30,10,40,4,'No help is available at this
      time','OK','','','B',0,REFRESH,MAT TermDef)
0467  IF REFRESH THEN GOSUB DSPSCRN ;* Refresh the screen if true windowing not supported
0468  RETURN
0469  *
0470  *
0471  ************************************************************************
0472  *
0473  * The CHECK.EXIT subroutine checks if any field data has changed, and
0474  * prompts if the user wants to abandon changes. If no changes, or the user
0475  * decides to abandon the changes, the DONE and XIT loop control variables
0476  * are set to TRUE, causing all three loops to terminate, and the program
0477  * itself to exit.
0478  *
0479  CHECK.EXIT: *
0480  *
0481  GOSUB CHECK.ABANDON
```

```
0482  IF OK THEN
0483   DONE = TRUE
0484   XIT = TRUE
0485   GOSUB HIDEPIX
0486  END
0487  RETURN
0488  *
0489  *
0490  ************************************************************************
0491  *
0492  * The CHECK.ID subroutine validates a newly entered item-ID. If the
0493  * current record has unsaved changes, the user is prompted to abandon the
0494  * changes. If no changes, or the user abandons the changes, and the new ID
0495  * is not NULL, an attempt is made to read a record using the new ID. If
0496  * the read is not successful, the ID is assumed to be a search string, and
0497  * the search subroutine is called to select an ID based on the search
0498  * string. If a valid ID is returned (or if one was initially entered), the
0499  * new record data is displayed and the command bar is refreshed. If the
0500  * new ID is null, or if the search routine did not return a valid ID, a
0501  * warning message is displayed and the OK indicator variable is set to
0502  * FALSE. Otherwise it is set to TRUE.
0503  *
0504  CHECK.ID: *
0505  *
0506  ************************************************************************
0507  *
0508  * Make sure we don't have any unsaved data before changing the ID
0509  *
0510  GOSUB CHECK.ABANDON
0511  IF NOT(OK) THEN RETURN ;* Reprompt for the ID
0512  IF ID EQ '' THEN
0513   OK = FALSE ;* NULL is not a valid ID!
0514  END ELSE
0515   *
0516   ***********************************************************************
0517   *
0518   * Try to read the customer record from the entered ID
0519   *
0520   GOSUB HIDEPIX ;* Hide previous cust picture before reading new one
0521   GOSUB READ.RECORD
0522   IF NOT(OK) THEN
0523    *
0524    **********************************************************************
0525    *
0526    * The attempt to read a record failed - assume the ID is a search string
0527    *
0528    CALL UIEX.GET.CUST.IDX.SUI(XID,ID,FN.CUST.XREF,FN.CUST,REFRESH,MAT TermDef)
0529    IF REFRESH THEN
0530     GOSUB DSPSCRN ;* Refresh the screen if true windowing not supported
0531    END ELSE
0532     GOSUB SHOWPIX
0533    END
0534    *
0535    **********************************************************************
0536    *
0537    * If the user did not select an item in the search routine, reprompt
0538    *
0539    IF XID EQ '' THEN RETURN
0540    *
0541    **********************************************************************
0542    *
```

```
0543    * Try to read the customer record from the selected ID
0544    *
0545    ID = XID
0546    GOSUB HIDEPIX ;* Hide previous cust picture before reading new one
0547    GOSUB READ.RECORD
0548    *
0549   END
0550  END
0551  *
0552  ************************************************************************
0553  *
0554  * If success, display the new record, otherwise show warning message
0555  *
0556  IF OK THEN
0557   GOSUB DSPDATA ;* Display new record
0558   GOSUB DSPBAR ;* Refresh command bar - button states may have changed
0559  END ELSE
0560   CALL SUI.MESSAGE.BOX(30,10,40,4,'Please enter a valid customer
      ID.','OK','','','B',0,REFRESH,MAT TermDef)
0561   IF REFRESH THEN GOSUB DSPSCRN ;* Refresh the screen if true windowing not supported
0562  END
0563  RETURN
0564  *
0565  *
0566  ************************************************************************
0567  *
0568  * The READ.RECORD subroutine reads a new customer record from the file and
0569  * initializes the internal field data array (IDATA) from the record array
0570  * (CUST.REC). If the record does not exist, the routine returns with the
0571  * OK indicator variable set to FALSE. Otherwise OK is set to TRUE, and the
0572  * CUST.ID variable is set to the new ID.
0573  *
0574  READ.RECORD: *
0575  *
0576  MATREAD CUST.REC FROM FN.CUST,ID THEN
0577   CUST.ID = ID
0578   IDATA(1) = CUST.ID
0579   IDATA(2) = CUST.CONTACT
0580   IDATA(3) = CUST.NAME
0581   IDATA(4) = CUST.ADDRESS1
0582   IDATA(5) = CUST.ADDRESS2
0583   IDATA(6) = CUST.CITY
0584   IDATA(7) = CUST.ST
0585   IDATA(8) = CUST.ZIP
0586   IDATA(9) = CUST.COUNTRY
0587   IDATA(10) = CUST.PHONE
0588   IDATA(11) = CUST.FAX
0589   IDATA(12) = CUST.HISTORY
0590   OK = TRUE ;* Set the SUCCESS indicator
0591  END ELSE
0592   OK = FALSE ;* Set the FAILURE indicator
0593  END
0594  RETURN
0595  *
0596  *
0597  ************************************************************************
0598  *
0599  * The CANCEL.RECORD subroutine checks if any field data has changed, and
0600  * prompts if the user wants to abandon changes. If no changes, or the user
0601  * decides to abandon the changes, the DONE loop control varialbe is set to
0602  * TRUE causing the PROMPT and ACTION loops to terminate, proceeding to the
```

```
0603  * next record.
0604  *
0605  CANCEL.RECORD: *
0606  *
0607  GOSUB CHECK.ABANDON
0608  IF OK THEN DONE = TRUE ;* proceed to next record
0609  GOSUB HIDEPIX ;* erase the picture
0610  RETURN
0611  *
0612  *
0613  ************************************************************************
0614  *
0615  * The NEW.RECORD subroutine checks if any field data has changed, and
0616  * prompts if the user wants to abandon changes. If no changes, or the
0617  * user decides to abandon the changes, the next ID is read from the
0618  * control file. The next ID is updated. The field data is cleared and the
0619  * new ID is displayed. The command bar is refreshed, since the command
0620  * button states may be different.
0621  *
0622  NEW.RECORD: *
0623  *
0624  GOSUB CHECK.ABANDON
0625  IF OK THEN
0626   * Get next sequential ID
0627   READVU ID FROM FN.CUST.CTRL,'NEXT',2 THEN
0628    WRITEV ID + 1 ON FN.CUST.CTRL,'NEXT',2
0629   END ELSE
0630    CALL SUI.MESSAGE.BOX(30,10,40,4,'Next item counter record not
      found!','OK','','','B',0,REFRESH,MAT TermDef)
0631    IF REFRESH THEN GOSUB DSPSCRN
0632    OK = FALSE
0633    RETURN
0634   END
0635   GOSUB RESTART
0636   IDATA(1) = ID
0637   GOSUB DSPDATA
0638   GOSUB DSPBAR
0639   NXTFLD = 2
0640  END
0641  RETURN
0642  *
0643  *
0644  ************************************************************************
0645  *
0646  * The DELETE.RECORD subroutine confirms that the user intends to delete
0647  * the current record. If the action is confirmed, the CUST.DELETE
0648  * subroutine is called to perform the deletion. A separate subroutine is
0649  * used to handle updating indexes, etc.
0650  *
0651  DELETE.RECORD: *
0652  *
0653  IF CUST.ID NE '' THEN
0654   YESNO = 2
0655   CALL SUI.MESSAGE.BOX(30,10,40,4,'Are you sure you want to delete this
      customer?','Yes]No','Y]N','','B',YESNO,REFRESH,MAT TermDef)
0656   IF REFRESH THEN GOSUB DSPSCRN
0657   IF YESNO = 1 THEN
0658    * Deletion has been confirmed - do the delete
0659    OK = TRUE ;* Set the SUCCESS indicator
0660    CALL UIEX.CUST.DELETE(CUST.ID,FN.CUST,FN.CUST.XREF)
0661    DONE = TRUE ;* proceed to next record
```

```
0662    GOSUB HIDEPIX ;* erase picture
0663   END ELSE
0664    OK = FALSE ;* Set the FAILURE indicator
0665   END
0666  END
0667  RETURN
0668  *
0669  *
0670  ************************************************************************
0671  *
0672  * The SAVE.RECORD subroutine copies internal field data from the IDATA
0673  * array to the customer record array (CUST.REC). The CUST.UPDATE
0674  * subroutine is called to perform the update. A separate subroutine is
0675  * used to handle updating indexes, etc.
0676  *
0677  SAVE.RECORD: *
0678  *
0679  IF IDATA(1) NE '' THEN
0680   * Copy data from the internal field data array (IDATA) to the CUST.REC array
0681   CUST.ID = IDATA(1)
0682   CUST.CONTACT = IDATA(2)
0683   CUST.NAME = IDATA(3)
0684   CUST.ADDRESS1 = IDATA(4)
0685   CUST.ADDRESS2 = IDATA(5)
0686   CUST.CITY = IDATA(6)
0687   CUST.ST = IDATA(7)
0688   CUST.ZIP = IDATA(8)
0689   CUST.COUNTRY = IDATA(9)
0690   CUST.PHONE = IDATA(10)
0691   CUST.FAX = IDATA(11)
0692   CUST.HISTORY = IDATA(12)
0693   * Update the file
0694   CALL UIEX.CUST.UPDATE(CUST.ID,MAT CUST.REC,FN.CUST,FN.CUST.XREF)
0695   OK = TRUE ;* Set the SUCCESS indicator
0696  END ELSE
0697   OK = FALSE ;* Set the FAILURE indicator
0698  END
0699  DONE = TRUE ;* proceed to next record
0700  RETURN
0701  *
0702  *
0703  ************************************************************************
0704  *
0705  * The RESTART subroutine prepares the internal field data array (IDATA),
0706  * customer record array (CUST.REC) and ID for a new customer record.
0707  *
0708  RESTART: *
0709  *
0710  CUST.ID = ''
0711  MAT CUST.REC = ''
0712  MAT IDATA = ''
0713  BTN = 5 ;* Set default button to "cancel"
0714  RETURN
0715  *
0716  *
0717  ************************************************************************
0718  *
0719  * The CHECK.CHANGED subroutine checks if any internal field data has been
0720  * changed. The OK indicator variable is set to FALSE if any data is
0721  * changed, otherwise it is set to TRUE. The ID field is not checked, since
0722  * it is appropriately handled by the CHECK.ID subroutine.
```

```
0723  *
0724  CHECK.CHANGED: *
0725  *
0726  OK = FALSE
0727  IF CUST.CONTACT # IDATA(2) THEN RETURN
0728  IF CUST.NAME # IDATA(3) THEN RETURN
0729  IF CUST.ADDRESS1 # IDATA(4) THEN RETURN
0730  IF CUST.ADDRESS2 # IDATA(5) THEN RETURN
0731  IF CUST.CITY # IDATA(6) THEN RETURN
0732  IF CUST.ST # IDATA(7) THEN RETURN
0733  IF CUST.ZIP # IDATA(8) THEN RETURN
0734  IF CUST.COUNTRY # IDATA(9) THEN RETURN
0735  IF CUST.PHONE # IDATA(10) THEN RETURN
0736  IF CUST.FAX # IDATA(11) THEN RETURN
0737  IF CUST.HISTORY # IDATA(12) THEN RETURN
0738  OK = TRUE
0739  RETURN
0740  *
0741  *
0742  ************************************************************************
0743  *
0744  * The CHECK.ABANDON subroutine calls CHECK.CHANGED. If any changes are
0745  * found, a message is displayed and the user is prompted to abandon the
0746  * changes. If there are no changes, or if the user decides to abandon
0747  * changes, the OK indicator variable is set to TRUE. Otherwise it is set
0748  * to FALSE.
0749  *
0750  CHECK.ABANDON: *
0751  *
0752  GOSUB CHECK.CHANGED
0753  IF NOT(OK) THEN
0754   YESNO = 2
0755   CALL SUI.MESSAGE.BOX(25,10,50,4,'Do you want to abandon all your
      changes?','Yes]No','Y]N','','B',YESNO,REFRESH,MAT TermDef)
0756   IF REFRESH THEN GOSUB DSPSCRN
0757   IF YESNO = 1 THEN OK = TRUE
0758  END
0759  RETURN
0760  *
0761  *
0762  ************************************************************************
0763  *
0764  * The DSPSCRN subroutine is used to refresh the entire screen.
0765  *
0766  DSPSCRN: *
0767  *
0768  * Clear the screen
0769  PRINT TERM.CLEAR:
0770  *
0771  * If running AccuTerm, display the logo
0772  IF IMGFLG THEN
0773   PRINT ESC:STX:'iL,':PIX.PATH:'ASE-LOGO-SMALL.JPG,65,0,12,2,0,N':CR:
0774  END
0775  *
0776  * Display heading & labels
0777  CALL SUI.DISPLAY.LABEL(30, 0, 0, 'C', 0, 'Customer File Maintenance', CMD, MAT
      TermDef)
0778  FOR XLINE = 1 TO NUMFLDS
0779   LBWD = CONTROL(XLINE)<1,4>
0780   LBTX = CONTROL(XLINE)<1,1>
0781   IF LBWD > 0 THEN LBTX = (LBTX : STR('.',LBWD))[1,LBWD] ;* Pad label with dots
```

```
0782   CALL SUI.DISPLAY.LABEL(CONTROL(XLINE)<1,2>, CONTROL(XLINE)<1,3>, LBWD, 'L', 0,
       LBTX, CMD, MAT TermDef)
0783   NEXT XLINE
0784   *
0785   * Display the field data
0786   GOSUB DSPDATA
0787   *
0788   * Display the command bar
0789   GOSUB DSPBAR
0790   *
0791   RETURN
0792   *
0793   *
0794   ************************************************************************
0795   *
0796   * The DSPDATA subroutine is used to refresh the field data for all fields.
0797   *
0798   DSPDATA: *
0799   *
0800   FOR XLINE = 1 TO NUMFLDS
0801    GOSUB DSPLINE
0802   NEXT XLINE
0803   GOSUB SHOWPIX
0804   RETURN
0805   *
0806   *
0807   ************************************************************************
0808   *
0809   * The DSPLINE subroutine is used to refresh the field data for one field.
0810   * The field to be refreshed is specified by the XLINE variable.
0811   *
0812   DSPLINE: *
0813   *
0814   CALL SUI.DISPLAY.TEXT(CONTROL(XLINE)<1,5>, CONTROL(XLINE)<1,6>, CONTROL(XLINE)<1,7>,
       CONTROL(XLINE)<1,8>, 0, '', IDATA(XLINE), 0, CMD, MAT TermDef)
0815   RETURN
0816   *
0817   *
0818   ************************************************************************
0819   *
0820   * The DSPBAR subroutine is used to refresh the command bar. The state of
0821   * the Save and Delete buttons is adjusted depending on whether a record
0822   * has been read or an item-ID has been entered.
0823   *
0824   DSPBAR: *
0825   *
0826   IF IDATA(1) NE '' THEN BTNSTATE<1,3> = 0 ELSE BTNSTATE<1,3> = 2
0827   IF CUST.ID NE '' THEN BTNSTATE<1,4> = 0 ELSE BTNSTATE<1,4> = 2
0828   CALL SUI.DISPLAY.BTNBAR(10, 22, 60, 1, BTNSTATE, BTNTEXT, BTNKEY, '', CMD, MAT
       TermDef)
0829   RETURN
0830   *
0831   *
0832   ************************************************************************
0833   *
0834   * If running AccuTerm and an customer ID has been entered, try to display
0835   * the customer's picture for the current record.
0836   *
0837   SHOWPIX: *
0838   *
0839   IF IMGFLG THEN
```

```
0840   IF CUST.ID NE '' THEN
0841    PRINT ESC:STX:'iL,':PIX.PATH:'CUST\':CUST.ID:'.JPG,60,3,15,8,1,B':CR:
0842   END
0843  END
0844  RETURN
0845  *
0846  *
0847  ************************************************************************
0848  *
0849  * Erase the current customer picture before calling the lookup screen
0850  * because pictures always display over text, and don't get along well
0851  * with windowing.
0852  *
0853  HIDEPIX: *
0854  *
0855  IF IMGFLG THEN
0856   IF CUST.ID NE '' THEN
0857    PRINT ESC:STX:'iD,':PIX.PATH:'CUST\':CUST.ID:'.JPG':CR:
0858   END
0859  END
0860  RETURN
0861  *
0862  *
0863  ************************************************************************
0864  *
0865  * The CHECK.MOUSE subroutine is called if any SUI input routine returns
0866  * with CMD set to TERM.LEFTBUTTON$. This routine checks each field,
0867  * passing the field position and size to SUI.MOUSE.HIT. If a "hit" is
0868  * detected, the NXTFLD variable is set to the field number of the clicked
0869  * field. If no field was clicked, the position and size of the command bar
0870  * is checked, and if "hit", NXTFLG is set to NUMFLDS + 1 to cause the
0871  * ACTION and PROMPT loops to prompt for action using the command bar. The
0872  * actual mouse click location is saved internally by the SUI input
0873  * routines, and if a command button is clicked while a different field is
0874  * active, the click action is executed when SUI.INPUT.BTNBAR is called.
0875  *
0876  CHECK.MOUSE: *
0877  *
0878  FOR XLINE = 1 TO NUMFLDS
0879   CALL SUI.MOUSE.HIT(CONTROL(XLINE)<1,5>, CONTROL(XLINE)<1,6>, CONTROL(XLINE)<1,7>,
      CONTROL(XLINE)<1,8>, 0, HIT, MAT TermDef)
0880   IF HIT THEN
0881    NXTFLD = XLINE
0882    RETURN
0883   END
0884  NEXT XLINE
0885  * Check if a button on the command bar was clicked
0886  CALL SUI.MOUSE.HIT(10, 22, 60, 1, 0, HIT, MAT TermDef)
0887  IF HIT THEN NXTFLD = NUMFLDS + 1
0888  RETURN
0889  *
0890  END
0891
```