D:\Atwin32.dev\Samples\UIEX\UIEX4

```
1  *
2  ************************************************************************
3  * Copyright (c) 2004 Zumasys, Inc. as an unpublished work. Permission is *
4  * hereby granted, free of charge, to any person obtaining a copy of this *
5  * software, to use the software without restriction, including without   *
6  * limitation the rights to use, copy, modify, merge, publish or          *
7  * distribute the software, and to permit persons to whom the software is *
8  * furnished to do so, subject to the following conditions: This          *
9  * copyright notice and permission notice shall be included in all copies *
10 * or substantial portions of the software. THE SOFTWARE IS PROVIDED "AS  *
11 * IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT   *
12 * NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A        *
13 * PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE          *
14 * COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, *
15 * WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT *
16 * OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN  *
17 * THE SOFTWARE.                                                          *
18 ************************************************************************
19 ************************************************************************
20 *
21 * USER INTERFACE EXAMPLE PROGRAM 4
22 *
23 ************************************************************************
24 ************************************************************************
25 *
26 * This example illustrates a character-based interface using the Zumasys
27 * Smart User Interface library for data entry. Most display and input
28 * processing is handled by the SUI subroutines, rather than using PRINT
29 * and INPUT statements. By using the SUI library, input fields accept
30 * cursor keys, editing keys, function keys and mouse clicks. The SUI
31 * subroutines also utilize AccuTerm Visual Styles to display screen
32 * elements using a Windows-like look (if avaliable).
33 *
34 ************************************************************************
35 *
36 * To use the SUI library, you must include SUI.TERMDEF. This INCLUDE item
37 * contains EQUates for various terminal functions like visual attributes
38 * and line graphics, as well as constants for the keyboard commands like
39 * TERM.ENTER$ or TERM.PGUP$.
40 *
41 $INCLUDE SUIBP SUI.TERMDEF
42 *
43 ************************************************************************
44 *
45 * The CUST.REC INCLUDE item declares the CUST.REC array, which is used to
46 * store a working copy of the current data record. It also defines the
47 * record layout by equating field names to array positions in the CUST.REC
48 * array.
49 *
50 $INCLUDE UIEX.CUST.REC
51 *
52 ************************************************************************
53 *
54 * EQUates for control characters and delimiters and other constants
```

D:\Atwin32.dev\Samples\UIEX\UIEX5

```
1  *
2  ************************************************************************
3  * Copyright (c) 2004 Zumasys, Inc. as an unpublished work. Permission is *
4  * hereby granted, free of charge, to any person obtaining a copy of this *
5  * software, to use the software without restriction, including without   *
6  * limitation the rights to use, copy, modify, merge, publish or          *
7  * distribute the software, and to permit persons to whom the software is *
8  * furnished to do so, subject to the following conditions: This          *
9  * copyright notice and permission notice shall be included in all copies *
10 * or substantial portions of the software. THE SOFTWARE IS PROVIDED "AS  *
11 * IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT   *
12 * NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A        *
13 * PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE          *
14 * COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, *
15 * WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT *
16 * OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN  *
17 * THE SOFTWARE.                                                          *
18 ************************************************************************
19 ************************************************************************
20 *
21 * USER INTERFACE EXAMPLE PROGRAM 5
22 *
23 ************************************************************************
24 ************************************************************************
25 *
26 * This example illustrates an advanced character-based user interface
27 * using the Zumasys Smart User Interface library. All display and input
28 * processing is handled by the SUI subroutines, rather than using PRINT
29 * and INPUT statements. By using the SUI library, all input fields accept
30 * cursor keys, editing keys, function keys and mouse clicks. The SUI
31 * subroutines also utilize AccuTerm Visual Styles to display screen
32 * elements using a Windows-like look (if avaliable).
33 *
34 ************************************************************************
35 *
36 * To use the SUI library, you must include SUI.TERMDEF. This INCLUDE item
37 * contains EQUates for various terminal functions like visual attributes
38 * and line graphics, as well as constants for the keyboard commands like
39 * TERM.ENTER$ or TERM.PGUP$.
40 *
41 $INCLUDE SUIBP SUI.TERMDEF
42 *
43 ************************************************************************
44 *
45 * The CUST.REC INCLUDE item declares the CUST.REC array, which is used to
46 * store a working copy of the current data record. It also defines the
47 * record layout by equating field names to array positions in the CUST.REC
48 * array.
49 *
50 $INCLUDE UIEX.CUST.REC
51 *
52 ************************************************************************
53 *
54 * EQUates for control characters and delimiters and other constants
```

Found 44 differences: 256 lines, 370 inline differences in 122 changed lines				
Added(113,97)	Deleted(21,81)	Changed(122)	Changed in changed(192)	Ignored

D:\Atwin32.dev\Samples\UIEX\UIEX4

```
55 *
56 EQU VM TO CHAR(253)
57 EQU BEL TO CHAR(7)
58 EQU ESC TO CHAR(27)
59 EQU STX TO CHAR(2)
60 EQU CR TO CHAR(13)
61 EQU FALSE TO 0
62 EQU TRUE TO 1
63 *
64 ************************************************************************
65 *
66 * Any routine that uses the SUI library must call SUI.GET.TERM to
67 * initialize the TermDef array. The TermDef array is a required argument
68 * for all of the SUI subroutines. It is much better to call this routine
69 * one time and pass the TermDef array between various routines instead of
70 * calling it in each application subroutine that uses SUI.
71 *
72 CALL SUI.GET.TERM(MAT TermDef)
73 *
74 ************************************************************************
75 *
76 * Once the TermDef array is initialized by SUI.GET.TERM, the TERM.RESET
77 * string is sent to the terminal to initialize the terminal state. The
78 * reset string programs necessary function keys and initalizes any other
79 * terminal state as required for each particular terminal type.
80 *
81 PRINT TERM.RESET:
82 *
83 ************************************************************************
84 *
85 * This program uses AccuTerm Imaging to display pictures associated with
86 * records in the customer file. This EQUate defines the path where the
87 * image files are located.
88 *
89 $INCLUDE UIEX.PIX.PATH
90 *
91 * Check for image support (no images if dumb terminal or AccuTerm Lite)
92 *
93 IF INDEX(TERM.SPECIAL,'I',1) THEN IMGFLG = TRUE ELSE IMGFLG = FALSE
94 *
95 ************************************************************************
96 *
97 * Open our files...
98 *
99 OPEN 'CUST.SAMPLE' TO FN.CUST ELSE PRINT 'NO CUST.SAMPLE FILE'; STOP
100 OPEN 'CUST.SAMPLE.CTRL' TO FN.CUST.CTRL ELSE PRINT 'NO CUST.SAMPLE.CTRL FILE
101 OPEN 'CUST.SAMPLE.XREF' TO FN.CUST.XREF ELSE PRINT 'NO CUST.SAMPLE.XREF FILE
102 *
103 ************************************************************************
104 *
105 * Define the data field control structures. NUMFLDS is the number of
106 * fields. The CONTROL array defines the field control elements such as
107 * field label, label position, field position and size, and field prompt.
108 * The IDATA array stores the current field values, and is initialized
```

D:\Atwin32.dev\Samples\UIEX\UIEX5

```
55 *
56 EQU VM TO CHAR(253)
57 EQU BEL TO CHAR(7)
58 EQU ESC TO CHAR(27)
59 EQU STX TO CHAR(2)
60 EQU CR TO CHAR(13)
61 EQU FALSE TO 0
62 EQU TRUE TO 1
63 *
64 ************************************************************************
65 *
66 * Any routine that uses the SUI library must call SUI.GET.TERM to
67 * initialize the TermDef array. The TermDef array is a required argument
68 * for all of the SUI subroutines. It is much better to call this routine
69 * one time and pass the TermDef array between various routines instead of
70 * calling it in each application subroutine that uses SUI.
71 *
72 CALL SUI.GET.TERM(MAT TermDef)
73 *
74 ************************************************************************
75 *
76 * Once the TermDef array is initialized by SUI.GET.TERM, the TERM.RESET
77 * string is sent to the terminal to initialize the terminal state. The
78 * reset string programs necessary function keys and initalizes any other
79 * terminal state as required for each particular terminal type.
80 *
81 PRINT TERM.RESET:
82 *
83 ************************************************************************
84 *
85 * This program uses AccuTerm Imaging to display pictures associated with
86 * records in the customer file. This EQUate defines the path where the
87 * image files are located.
88 *
89 $INCLUDE UIEX.PIX.PATH
90 *
91 * Check for image support (no images if dumb terminal or AccuTerm Lite)
92 *
93 IF INDEX(TERM.SPECIAL,'I',1) THEN IMGFLG = TRUE ELSE IMGFLG = FALSE
94 *
95 ************************************************************************
96 *
97 * Open our files...
98 *
99 OPEN 'CUST.SAMPLE' TO FN.CUST ELSE PRINT 'NO CUST.SAMPLE FILE'; STOP
100 OPEN 'CUST.SAMPLE.CTRL' TO FN.CUST.CTRL ELSE PRINT 'NO CUST.SAMPLE.CTRL FILE
101 OPEN 'CUST.SAMPLE.XREF' TO FN.CUST.XREF ELSE PRINT 'NO CUST.SAMPLE.XREF'; ST
102 *
103 ************************************************************************
104 *
105 * Define the data field control structures. NUMFLDS is the number of
106 * fields. The CONTROL array defines the field control elements such as
107 * field label, label position, field position and size, and field prompt.
108 * The IDATA array stores the current field values, and is initialized from
```

Found 44 differences: 256 lines, 370 inline differences in 122 changed lines				
Added(113,97)	Deleted(21,81)	Changed(122)	Changed in changed(192)	Ignored

D:\Atwin32.dev\Samples\UIEX\UIEX4

```
* from the CUST.REC array when a data record is read from the file. When
* the record is updated, values are copied from the IDATA array to the
* CUST.REC array and the CUST.REC array is passed to the CUST.UPDATE
* subroutine to update the data and index files (a separate subroutine is
* used to update the file since the update process may handle other
* functions like updating indexes).
*
NUMFLDS = 12
DIM CONTROL(12)
DIM IDATA(12)
*
* Define field control elements
*  value 1: field label text
*  value 2: field label column
*  value 3: field label row
*  value 4: field label width (0 for actual width, no padding)
*  value 5: field data column
*  value 6: field data row
*  value 7: field data width
*  value 8: field data height
*  value 9: field prompt message
CONTROL(1) = 'Customer ID???0?8??0?ýEnter the ID, or name to search for, or
CONTROL(2) = 'Contact???0?8??0?ýEnter the contact name'
CONTROL(3) = 'Company name???0?8??0?ýEnter the company name'
CONTROL(4) = 'Address line 1???0?8??0?ýEnter address line 1'
CONTROL(5) = 'Address line 2???0?8??0?ýEnter address line 2'
CONTROL(6) = 'City???0?8??5?ýEnter the city'
CONTROL(7) = 'State or province???0?8??5?ýEnter the state or province abbrev
CONTROL(8) = 'Zip/postal code??0?0?8?0?0?ýEnter the zip or postal code'
CONTROL(9) = 'Country??1?0?8?1?8?ýEnter the country'
CONTROL(10) = 'Phone???2?0?8?2?5?ýEnter the phone number'
CONTROL(11) = 'Fax??3?0?8?3?5?ýEnter the fax number'
CONTROL(12) = 'Notes??4?0?8?4?0?ýEnter any notes about this customer'
*
*************************************************************************
*
* Define some screen control strings for prompts & errors
*
PROMPT ''
PL = @(5,22):TERM.CEOL
EL = @(5,23):TERM.CEOL
*
*************************************************************************
*
* This program processes one customer record at a time, and is organized
* using three nested loops. The outermost loop (the RECORD loop) is
* executed once for each record accessed. The loop repeats until the XIT
* control variable is set to TRUE.
```

D:\Atwin32.dev\Samples\UIEX\UIEX5

```
* the CUST.REC array when a data record is read from the file. When the
* record is updated, values are copied from the IDATA array to the
* CUST.REC array and the CUST.REC array is passed to the CUST.UPDATE
* subroutine to update the data and index files (a separate subroutine is
* used to update the file since the update process may handle other
* functions like updating indexes).
*
NUMFLDS = 12
DIM CONTROL(12)
DIM IDATA(12)
*
* Define field control elements
*  value 1: field label text
*  value 2: field label column
*  value 3: field label row
*  value 4: field label width (0 for actual width, no padding)
*  value 5: field data column
*  value 6: field data row
*  value 7: field data width
*  value 8: field data height
*  value 9: field prompt message
CONTROL(1) = 'Customer IDý1Oý2ý17ý28ý2ý3Oý1ýEnter the ID, or name to search
CONTROL(2) = 'Contactý1Oý4ý17ý28ý4ý3Oý1ýEnter the contact name'
CONTROL(3) = 'Company nameý1Oý5ý17ý28ý5ý3Oý1ýEnter the company name'
CONTROL(4) = 'Address line 1ý1Oý6ý17ý28ý6ý3Oý1ýEnter address line 1'
CONTROL(5) = 'Address line 2ý1Oý7ý17ý28ý7ý3Oý1ýEnter address line 2'
CONTROL(6) = 'Cityý1Oý8ý17ý28ý8ý25ý1ýEnter the city'
CONTROL(7) = 'State or provinceý1Oý9ý17ý28ý9ý15ý1ýEnter the state or provinc
CONTROL(8) = 'Zip/postal codeý1Oý1Oý17ý28ý1Oý1Oý1ýEnter the zip or postal cc
CONTROL(9) = 'Countryý1Oý11ý17ý28ý11ý18ý1ýEnter the country'
CONTROL(10) = 'Phoneý1Oý12ý17ý28ý12ý15ý1ýEnter the phone number'
CONTROL(11) = 'Faxý1Oý13ý17ý28ý13ý15ý1ýEnter the fax number'
CONTROL(12) = 'Notesý1Oý14ý17ý28ý14ý3Oý3ýEnter any notes about this customer
*
*************************************************************************
*
* A "command bar" is used to accept user action requests. The command bar
* as displayed at the bottom of the screen, and appears as a row of
* "buttons" which the user can click with the mouse. Each button has a
* function key associated with it. The user can also use the cursor keys
* to change the active button, and pressing the ENTER key or SPACE BAR
* executes the action.
*
BTNTEXT = 'F1=Help':VM:'F6=New':VM:'F5=Save':VM:'F4=Delete':VM:'F3=Cancel':V
BTNSTATE = '0':VM:'0':VM:'2':VM:'2':VM:'0':VM:'0' ;* Initially, the Save and
BTNKEY = 'H':VM:'N':VM:'S':VM:'D':VM:'C':VM:'X'
BTN = 6 ;* Initial default button = EXIT
*
*************************************************************************
*
* This program processes one customer record at a time, and is organized
* using three nested loops. The outermost loop (the RECORD loop) is
* executed once for each record accessed. The loop repeats until the XIT
* control variable is set to TRUE.
```

Found 44 differences: 256 lines, 370 inline differences in 122 changed lines				
Added(113,97)	Deleted(21,81)	Changed(122)	Changed in changed(192)	Ignored

D:\Atwin32.dev\Samples\UIEX\UIEX4

```
157 *
158 XIT = FALSE
159 LOOP UNTIL XIT DO
160  *
161  ********************************************************************************
162  *
163  * Before prompting for the customer ID, reset the internal data array
164  * (IDATA) and CUST.ID variables, then clear the screen and display the
165  * heading and field labels.
166  *
167  GOSUB RESTART
168  GOSUB DSPSCRN
169  *
170  ********************************************************************************
171  *
172  * The middle loop is the ACTION loop. It is executed for the current
173  * record until the user performs an action that terminates processing of
174  * that record, such as exiting, cancelling, saving or deleting the
175  * record. The field number to begin prompting (NXTFLD) is initialized to
176  * 1, causing the ID field to be prompted first. The loop immediately
177  * enters the PROMPT loop, followed by the field modification prompt. The
178  * loop repeats until the DONE control variable is set to TRUE.
179  *
180  NXTFLD = 1
181  DONE = FALSE
182  LOOP
183   *
184   ********************************************************************************
185   *
186   * The inner loop is the PROMPT loop. It is executed for each prompt
187   * field as specified by the NXTFLD variable. The prompt loop simply
188   * calls the local INPUT.FIELD subroutine which performs field input,
189   * data validation and keyboard command decoding. The FIELD loop repeats
190   * until the field number is greater than the number of fields, or until
191   * the DONE control variable is set to TRUE. The DONE control variable
192   * may be set to TRUE in the INPUT.FIELD subroutine, if the user enters a
193   * NULL to for the item-ID.
194   *
195   LOOP
196    CURFLD = NXTFLD
197   UNTIL DONE OR CURFLD > NUMFLDS DO
198    *
199    ********************************************************************************
200    *
201    * Prompt for the next field. Update the NXTFLD variable with the field
202    * number to prompt next, taking into account cursor keys and mouse
203    * clicks.
204    *
```

D:\Atwin32.dev\Samples\UIEX\UIEX5

```
163 *
164 * Note: the term "terminal action" as used in the following comments means
165 * an action that causes the current record or the program itself to
166 * terminate. An action such as cancel, save, or exit is a terminal action.
167 *
168 XIT = FALSE
169 LOOP UNTIL XIT DO
170  *
171  ********************************************************************************
172  *
173  * Before prompting for the customer ID, reset the internal data array
174  * (IDATA) and CUST.ID variables, then clear the screen and display the
175  * heading and field labels.
176  *
177  GOSUB RESTART
178  GOSUB DSPSCRN
179  *
180  ********************************************************************************
181  *
182  * The middle loop is the ACTION loop. It is executed for the current
183  * record until the user performs an action that terminates processing of
184  * that record, such as exiting, cancelling, saving or deleting the
185  * record. The field number to begin prompting (NXTFLD) is initialized to
186  * 1, causing the ID field to be prompted first. The loop immediately
187  * enters the PROMPT loop, followed by the action (command bar) prompt.
188  * The loop repeats until the DONE control variable is set to TRUE.
189  *
190  NXTFLD = 1
191  DONE = FALSE
192  LOOP
193   *
194   ********************************************************************************
195   *
196   * The inner loop is the PROMPT loop. It is executed for each prompt
197   * field as specified by the NXTFLD variable. The prompt loop simply
198   * calls the local INPUT.FIELD subroutine which performs field input,
199   * data validation and keyboard command decoding. The FIELD loop repeats
200   * until the field number is greater than the number of fields, or until
201   * the DONE control variable is set to TRUE. The DONE control variable
202   * may be set to TRUE in the INPUT.FIELD subroutine, if the user presses
203   * a function key that causes a terminating action, such as exit, cancel,
204   * save or delete.
205   *
206   LOOP
207    CURFLD = NXTFLD
208   UNTIL DONE OR CURFLD > NUMFLDS DO
209    *
210    ********************************************************************************
211    *
212    * Prompt for the next field. Execute any action associated with a
213    * function key (such as exit, cancel, save, delete). Update the NXTFLD
214    * variable with the field number to prompt next, taking into account
215    * cursor keys and mouse clicks.
216    *
```

Found 44 differences: 256 lines, 370 inline differences in 122 changed lines				
Added(113,97)	Deleted(21,81)	Changed(122)	Changed in changed(192)	Ignored

D:\Atwin32.dev\Samples\UIEX\UIEX4

```
205     GOSUB INPUT.FIELD
206   REPEAT ;* end of PROMPT loop
207   *
208   ************************************************************************
209   *
210   * If the FIELD loop exited with the DONE control variable set to TRUE,
211   * bypass the modification prompt because no action is required.
212   * Otherwise, prompt for which field to modify, or other actions such as
213   * save or delete.
214   *
215   IF NOT(DONE) THEN
216    *
217    NXTFLD = NUMFLDS + 1 ;* assume we need to reprompt for action
218    *
219    PRINT PL:'Enter field number to modify or FI to save, DE to delete, EX to
220    INPUT ANS:
221    PRINT EL:
222    *
223    ************************************************************************
224    *
225    * Decode the response
226    *
227    ANS = OCONV(ANS, 'MCU')
228    BEGIN CASE
229     CASE NUM(ANS) AND ANS >= 1 AND ANS <= NUMFLDS; NXTFLD = ANS
230     CASE ANS EQ 'FI'; GOSUB SAVE.RECORD
231     CASE ANS EQ 'DE'; GOSUB DELETE.RECORD
232     CASE ANS EQ 'EX' OR ANS EQ ''; GOSUB CHECK.EXIT
233    END CASE
234    *
235   END
```

D:\Atwin32.dev\Samples\UIEX\UIEX5

```
217     GOSUB INPUT.FIELD
218   REPEAT ;* end of PROMPT loop
219   *
220   ************************************************************************
221   *
222   * If the FIELD loop exited with a terminal action, the DONE control
223   * variable will be set to TRUE. In this case, no action prompt is
224   * necessary since the action has already been handled by the INPUT.FIELD
225   * subroutine. Otherwise, prompt for actions using the command bar.
226   *
227   IF NOT(DONE) THEN
228    *
229    NXTFLD = NUMFLDS + 1 ;* assume we need to reprompt for action
230    *
231    ************************************************************************
232    *
233    * Use the SUI.INPUT.BTNBAR to prompt for actions using the command bar.
234    * The SUI.INPUT.BTNBAR routine returns with CMD set to TERM.ENTER$ if
235    * the user clicked a button, or pressed the ENTER key or SPACE BAR when
236    * a button was active. In this case, the BTN argument returns the
237    * number of the button.
238    *
239    CALL SUI.INPUT.BTNBAR(10, 22, 60, 1, BTNSTATE, BTNTEXT, BTNKEY, 'C', BTN,
240    *
241    ************************************************************************
242    *
243    * Decode the command
244    *
245    IF CMD EQ TERM.ENTER$ THEN
246     *
247     * Translate button number to equivalent function key
248     BEGIN CASE
249      CASE BTN = 1; CMD = TERM.F1$
250      CASE BTN = 2; CMD = TERM.F6$
251      CASE BTN = 3; CMD = TERM.F5$
252      CASE BTN = 4; CMD = TERM.F4$
253      CASE BTN = 5; CMD = TERM.F3$
254      CASE BTN = 6; CMD = TERM.F2$
255     END CASE
256    END
257    *
258    * Decode function keys, BACK TAB key and mouse clicks
259    BEGIN CASE
260     CASE CMD EQ TERM.LEFTBUTTON$; GOSUB CHECK.MOUSE
261     CASE CMD EQ TERM.F1$; GOSUB SHOW.HELP
262     CASE CMD EQ TERM.F2$; GOSUB CHECK.EXIT
263     CASE CMD EQ TERM.F3$; GOSUB CANCEL.RECORD
264     CASE CMD EQ TERM.F4$; GOSUB DELETE.RECORD
265     CASE CMD EQ TERM.F5$; GOSUB SAVE.RECORD
266     CASE CMD EQ TERM.F6$; GOSUB NEW.RECORD
267     CASE CMD EQ TERM.STAB$; NXTFLD = NUMFLDS
268    END CASE
269    *
270   END
```

Found 44 differences: 256 lines, 370 inline differences in 122 changed lines				
Added(113,97)	Deleted(21,81)	Changed(122)	Changed in changed(192)	Ignored

D:\Atwin32.dev\Samples\UIEX\UIEX4

```
236   *
237  UNTIL DONE DO REPEAT ;* end of ACTION loop
238  *
239 REPEAT ;* end of RECORD loop
240 *
241 ************************************************************************
242 *
243 * All done - clear the screen, restore the cursor and exit!
244 PRINT TERM.CLEAR:TERM.CSRON:
245 STOP
246 *
247 *
248 ************************************************************************
249 ************************************************************************
250 * LOCAL SUBROUTINES
251 ************************************************************************
252 ************************************************************************
253 *
254 *
255 ************************************************************************
256 *
257 * The INPUT.FIELD subroutine is the main prompting routine. This routine
258 * displays the prompt string (from the CONTROL array), and enters a loop,
259 * prompting for a specified field (CURFLD) and setting the next field
260 * variable (NXTFLD) appropriately. The loop repeats until the NXTFLD
261 * (initially set to CURFLD) changes. This causes the field prompt to be
262 * repeated in case unrecognized keys are pressed (for example, the PGUP
263 * key), or invalid data is entered (illegal customer ID, etc.) If a NULL
264 * is entered for the ID field, and there is no current record, the ACTION
265 * loop control variable, DONE, and the RECORD loop control variable, XIT,
266 * are set to TRUE, and this routine exits. A mouse click is handled by
267 * setting the NXTFLD variable to the field number that was clicked.
268 *
269 INPUT.FIELD:
270 *
271 ************************************************************************
272 *
273 * Display the prompt message
274 *
275 CALL SUI.DISPLAY.LABEL(5, 22, 74, 'L', 0, CONTROL(CURFLD)<1,9>, CMD, MAT Ter
276 *
277 ************************************************************************
278 *
279 * Initialize current value, next field & character positions
280 *
281 PREVAL = IDATA(CURFLD) ;* Save previous field value to detect change in valu
282 NXTFLD = CURFLD ;* Assume field number not changed
283 BEGPOS = 0 ;* Set initial text display character position
284 CURPOS = 0 ;* Set initial text cursor character position
285 *
286 ************************************************************************
287 *
```

D:\Atwin32.dev\Samples\UIEX\UIEX5

```
271   *
272  UNTIL DONE DO REPEAT ;* end of ACTION loop
273  *
274 REPEAT ;* end of RECORD loop
275 *
276 ************************************************************************
277 *
278 * All done - clear the screen, restore the cursor and exit!
279 PRINT TERM.CLEAR:TERM.CSRON:
280 STOP
281 *
282 *
283 ************************************************************************
284 ************************************************************************
285 * LOCAL SUBROUTINES
286 ************************************************************************
287 ************************************************************************
288 *
289 *
290 ************************************************************************
291 *
292 * The INPUT.FIELD subroutine is the main prompting routine. This routine
293 * displays the prompt string (from the CONTROL array), and enters a loop,
294 * prompting for a specified field (CURFLD) and setting the next field
295 * variable (NXTFLD) appropriately. The loop repeats until the NXTFLD
296 * (initially set to CURFLD) changes. This causes the field prompt to be
297 * repeated in case unrecognized keys are pressed (for example, the PGUP
298 * key), or the help key (F1) is pressed, or invalid data is entered
299 * (illegal customer ID, etc.) If any recognized function key is pressed
300 * that causes an action, and the action is terminal and successfully
301 * executed (like exit, cancel, save, delete), the ACTION loop control
302 * variable, DONE, is set to TRUE, and this routine exits. A mouse click is
303 * handled by setting the NXTFLD variable to the field number that was
304 * clicked, or to NUMFLDS + 1 if a the command bar was clicked.
305 *
306 INPUT.FIELD:
307 *
308 ************************************************************************
309 *
310 * Display the prompt message
311 *
312 CALL SUI.DISPLAY.LABEL(10, 20, 74, 'L', 0, CONTROL(CURFLD)<1,9>, CMD, MAT Te
313 *
314 ************************************************************************
315 *
316 * Initialize current value, next field & character positions
317 *
318 PREVAL = IDATA(CURFLD) ;* Save previous field value to detect change in valu
319 NXTFLD = CURFLD ;* Assume field number not changed
320 BEGPOS = 0 ;* Set initial text display character position
321 CURPOS = 0 ;* Set initial text cursor character position
322 *
323 ************************************************************************
324 *
```

Found 44 differences: 256 lines, 370 inline differences in 122 changed lines				
Added(113,97)	Deleted(21,81)	Changed(122)	Changed in changed(192)	Ignored

D:\Atwin32.dev\Samples\UIEX\UIEX4

```
288 * Prompt for this field until NXTFLD variable is updated
289 *
290 LOOP
291  *
292  ********************************************************************************
293  *
294  * Prompt for input with full text editing functions
295  *
296  CALL SUI.INPUT.TEXT(CONTROL(CURFLD)<1,5>, CONTROL(CURFLD)<1,6>, CONTROL(CUR
297  PRINT TERM.CSRON:EL: ;* Turn cursor back on & clear the error line
298  *
299  ********************************************************************************
300  *
301  * Decode returned CMD
302  *
303  BEGIN CASE
304   *
305   ********************************************************************************
306   *
307   * Check for ENTER, TAB, UP, DOWN or mouse click
308   *
309   CASE CMD EQ TERM.ENTER$ OR CMD EQ TERM.TAB$ OR CMD EQ TERM.DOWN$
310    NXTFLD = CURFLD + 1; * Move to next field
311   CASE CMD EQ TERM.STAB$ OR CMD EQ TERM.UP$
312    IF CURFLD > 1 THEN NXTFLD = CURFLD - 1 ;* Move to previous field
313   CASE CMD EQ TERM.LEFTBUTTON$
314    GOSUB CHECK.MOUSE ;* Move to clicked-on field
315    IF CURFLD EQ 1 AND NXTFLD NE CURFLD THEN
316     IF IDATA(1) EQ '' THEN NXTFLD = CURFLD ;* Mouse click in invalid field -
317    END
318   *
319  END CASE
320  *
321  ********************************************************************************
322  *
323  * Check for any special values (like 'END' or 'EXIT')
```

D:\Atwin32.dev\Samples\UIEX\UIEX5

```
325 * Prompt for this field until NXTFLD variable is updated
326 *
327 LOOP
328  *
329  ********************************************************************************
330  *
331  * Prompt for input with full text editing functions
332  *
333  CALL SUI.INPUT.TEXT(CONTROL(CURFLD)<1,5>, CONTROL(CURFLD)<1,6>, CONTROL(CUR
334  *
335  ********************************************************************************
336  *
337  * Decode returned CMD
338  *
339  BEGIN CASE
340   *
341   ********************************************************************************
342   *
343   * Check for ENTER, TAB, UP, DOWN or mouse click
344   *
345   CASE CMD EQ TERM.ENTER$ OR CMD EQ TERM.TAB$ OR CMD EQ TERM.DOWN$
346    NXTFLD = CURFLD + 1; * Move to next field
347   CASE CMD EQ TERM.STAB$ OR CMD EQ TERM.UP$
348    IF CURFLD > 1 THEN NXTFLD = CURFLD - 1 ;* Move to previous field
349   CASE CMD EQ TERM.LEFTBUTTON$
350    GOSUB CHECK.MOUSE ;* Move to clicked-on field
351    IF NXTFLD > NUMFLDS THEN RETURN ;* Clicked on a button
352   *
353   ********************************************************************************
354   *
355   * Check for function keys
356   *
357   CASE CMD EQ TERM.F1$; GOSUB SHOW.HELP
358   CASE CMD EQ TERM.F2$; GOSUB CHECK.EXIT
359   CASE CMD EQ TERM.F3$; GOSUB CANCEL.RECORD
360   CASE CMD EQ TERM.F4$; GOSUB DELETE.RECORD
361   CASE CMD EQ TERM.F5$; GOSUB SAVE.RECORD
362   CASE CMD EQ TERM.F6$; GOSUB NEW.RECORD; IF OK THEN RETURN
363   CASE 1; * Unrecognized key - just reprompt
364   *
365  END CASE
366  *
367  ********************************************************************************
368  *
369  * Check if a terminal action occurred (like exit, cancel, save, delete)
370  *
371  IF DONE THEN RETURN
372  *
373  ********************************************************************************
374  *
375  * Check for any special values (like 'END' or 'EXIT')
```

Found 44 differences: 256 lines, 370 inline differences in 122 changed lines				
Added(113,97)	Deleted(21,81)	Changed(122)	Changed in changed(192)	Ignored

D:\Atwin32.dev\Samples\UIEX\UIEX4

```
324  *
325  IF OCONV(IDATA(CURFLD),'MCU') EQ 'END' THEN
326   IDATA(CURFLD) = PREVAL ;* Restore previous value
327   GOSUB CHECK.ABANDON ;* Ensure OK to loose changes
328   IF OK THEN
329    *
330    ************************************************************************
331    *
332    * No changes, or user OKs abandoning them, so we are outa here!
333    *
334    DONE = TRUE
335    XIT = TRUE
336    RETURN
337    *
338   END ELSE
339    *
340    ************************************************************************
341    *
342    * User does not want to abandon changes, so refresh previous value &
343    * reprompt
344    *
345    XLINE = CURFLD ;* Field number to refresh
346    GOSUB DSPLINE ;* Redisplay the previous value
347    NXTFLD = CURFLD ;* Reprompt
348    *
349   END
350  END
351  *
352  ************************************************************************
353  *
354  * Peform field data validation if next field number changed
355  *
356  IF NXTFLD NE CURFLD THEN
357   BEGIN CASE
358    *
359    CASE CURFLD EQ 1
360     *
361     ************************************************************************
362     *
363     * Validate the ID field. If NULL, quit. If changed, read new record.
364     *
365     IF CUST.ID EQ '' AND IDATA(CURFLD) EQ '' THEN
366      DONE = TRUE
367      XIT = TRUE
368      RETURN
369     END
370     *
371     IF IDATA(CURFLD) NE PREVAL THEN
372      ID = IDATA(CURFLD)
373      GOSUB CHECK.ID ;* Check the newly entered ID handling index lookups
374      IF OK THEN
375       *
376       ************************************************************************
377       *
```

D:\Atwin32.dev\Samples\UIEX\UIEX5

```
376  *
377  IF OCONV(IDATA(CURFLD),'MCU') EQ 'END' THEN
378   IDATA(CURFLD) = PREVAL ;* Restore previous value
379   GOSUB CHECK.ABANDON ;* Ensure OK to loose changes
380   IF OK THEN
381    *
382    ************************************************************************
383    *
384    * No changes, or user OKs abandoning them, so we are outa here!
385    *
386    DONE = TRUE
387    XIT = TRUE
388    RETURN
389    *
390   END ELSE
391    *
392    ************************************************************************
393    *
394    * User does not want to abandon changes, so refresh previous value &
395    * reprompt
396    *
397    XLINE = CURFLD ;* Field number to refresh
398    GOSUB DSPLINE ;* Redisplay the previous value
399    NXTFLD = CURFLD ;* Reprompt
400    *
401   END
402  END
403  *
404  ************************************************************************
405  *
406  * Peform field data validation if next field number changed
407  *
408  IF NXTFLD NE CURFLD THEN
409   BEGIN CASE
410    *
411    CASE CURFLD EQ 1
412     *
413     ************************************************************************
414     *
415     * Validate the ID field. If changed, read new record.
416     *
417     IF CUST.ID EQ '' OR IDATA(CURFLD) NE PREVAL THEN
418      ID = IDATA(CURFLD)
419      GOSUB CHECK.ID ;* Check the newly entered ID handling index lookups
420      IF OK THEN
421       *
422       ************************************************************************
423       *
```

Found 44 differences: 256 lines, 370 inline differences in 122 changed lines				
Added(113,97)	Deleted(21,81)	Changed(122)	Changed in changed(192)	Ignored

D:\Atwin32.dev\Samples\UIEX\UIEX4

```
378       * New ID (or result of index lookup) is good!
379       *
380       IDATA(CURFLD) = ID ;* Update the current field value in case of index
381       *
382      END ELSE
383       *
384       ************************************************************************
385       *
386       * New ID is invalid
387       *
388       IDATA(CURFLD) = PREVAL ;* Restore previous value
389       XLINE = CURFLD ;* Field number to refresh
390       GOSUB DSPLINE ;* Redisplay the previous value
391       NXTFLD = CURFLD ;* Reprompt
392       *
393      END
394     END
395     *
396   END CASE
397  END
398  *
399 WHILE CURFLD EQ NXTFLD DO REPEAT
400 *
401 RETURN
402 *
403 *
404 ********************************************************************************
405 *
406 * The CHECK.EXIT subroutine checks if any field data has changed, and
407 * prompts if the user wants to abandon changes. If no changes, or the
408 * user decides to abandon the changes, the DONE and XIT loop control
409 * variables are set to TRUE, causing all three loops to terminate, and the
410 * program itself to exit.
```

D:\Atwin32.dev\Samples\UIEX\UIEX5

```
424       * New ID (or result of index lookup) is good!
425       *
426       IDATA(CURFLD) = ID ;* Update the current field value in case of index
427       *
428      END ELSE
429       *
430       ************************************************************************
431       *
432       * New ID is invalid
433       *
434       IDATA(CURFLD) = PREVAL ;* Restore previous value
435       XLINE = CURFLD ;* Field number to refresh
436       GOSUB DSPLINE ;* Redisplay the previous value
437       NXTFLD = CURFLD ;* Reprompt
438       *
439      END
440     END
441     *
442    CASE 1
443     *
444     ************************************************************************
445     *
446     * Validate other fields. If changed, set default button to "save".
447     *
448     IF PREVAL NE IDATA(CURFLD) THEN BTN = 3
449     *
450   END CASE
451  END
452  *
453 WHILE CURFLD EQ NXTFLD DO REPEAT
454 *
455 RETURN
456 *
457 *
458 ********************************************************************************
459 *
460 * The SHOW.HELP subroutine displays help for the current field, if it is
461 * available. Otherwise, a message indicating no help is available is
462 * shown.
463 *
464 SHOW.HELP: *
465 *
466 CALL SUI.MESSAGE.BOX(30,10,40,4,'No help is available at this time','OK','',
467 IF REFRESH THEN GOSUB DSPSCRN ;* Refresh the screen if true windowing not su
468 RETURN
469 *
470 *
471 ********************************************************************************
472 *
473 * The CHECK.EXIT subroutine checks if any field data has changed, and
474 * prompts if the user wants to abandon changes. If no changes, or the user
475 * decides to abandon the changes, the DONE and XIT loop control variables
476 * are set to TRUE, causing all three loops to terminate, and the program
477 * itself to exit.
```

Found 44 differences: 256 lines, 370 inline differences in 122 changed lines				
Added(113,97)	Deleted(21,81)	Changed(122)	Changed in changed(192)	Ignored

D:\Atwin32.dev\Samples\UIEX\UIEX4

```
411 *
412 CHECK.EXIT: *
413 *
414 GOSUB CHECK.ABANDON
415 IF OK THEN
416  DONE = TRUE
417  XIT = TRUE
418  GOSUB HIDEPIX
419 END
420 RETURN
421 *
422 *
423 ************************************************************************
424 *
425 * The CHECK.ID subroutine validates a newly entered item-ID. If the
426 * current record has unsaved changes, the user is prompted to abandon the
427 * changes. If no changes, or the user abandons the changes, and the new ID
428 * is not NULL, an attempt is made to read a record using the new ID. If
429 * the read is not successful, the ID is assumed to be a search string, and
430 * the search subroutine is called to select an ID based on the search
431 * string. If a valid ID is returned (or if one was initially entered), the
432 * new record data is displayed. If the new ID is null, or if the search
433 * routine did not return a valid ID, a warning message is displayed and
434 * the OK indicator variable is set to FALSE. Otherwise it is set to TRUE.
435 *
436 CHECK.ID: *
437 *
438 ************************************************************************
439 *
440 * Make sure we don't have any unsaved data before changing the ID
441 *
442 GOSUB CHECK.ABANDON
443 IF NOT(OK) THEN RETURN ;* Reprompt for the ID
444 IF ID EQ '' THEN
445  OK = FALSE ;* NULL is not a valid ID!
446 END ELSE
447  *
448  ************************************************************************
449  *
450  * Check if user wants new ID
451  *
452  IF ID EQ 'N' OR ID EQ 'n' THEN
453   * Get next sequential ID
454   READVU ID FROM FN.CUST.CTRL,'NEXT',2 THEN
455    WRITEV ID + 1 ON FN.CUST.CTRL,'NEXT',2
456    GOSUB RESTART
457    IDATA(1) = ID
458   END ELSE
459    PRINT EL:TERM.DRV:'Next item counter record not found!':TERM.NV:BEL:
460    INPUT ANS:
461    PRINT EL:
462    OK = FALSE
463    RETURN
```

D:\Atwin32.dev\Samples\UIEX\UIEX5

```
478 *
479 CHECK.EXIT: *
480 *
481 GOSUB CHECK.ABANDON
482 IF OK THEN
483  DONE = TRUE
484  XIT = TRUE
485  GOSUB HIDEPIX
486 END
487 RETURN
488 *
489 *
490 ************************************************************************
491 *
492 * The CHECK.ID subroutine validates a newly entered item-ID. If the
493 * current record has unsaved changes, the user is prompted to abandon the
494 * changes. If no changes, or the user abandons the changes, and the new ID
495 * is not NULL, an attempt is made to read a record using the new ID. If
496 * the read is not successful, the ID is assumed to be a search string, and
497 * the search subroutine is called to select an ID based on the search
498 * string. If a valid ID is returned (or if one was initially entered), the
499 * new record data is displayed and the command bar is refreshed. If the
500 * new ID is null, or if the search routine did not return a valid ID, a
501 * warning message is displayed and the OK indicator variable is set to
502 * FALSE. Otherwise it is set to TRUE.
503 *
504 CHECK.ID: *
505 *
506 ************************************************************************
507 *
508 * Make sure we don't have any unsaved data before changing the ID
509 *
510 GOSUB CHECK.ABANDON
511 IF NOT(OK) THEN RETURN ;* Reprompt for the ID
512 IF ID EQ '' THEN
513  OK = FALSE ;* NULL is not a valid ID!
514 END ELSE
515  *
516  ************************************************************************
517  *
```

Found 44 differences: 256 lines, 370 inline differences in 122 changed lines				
Added(113,97)	Deleted(21,81)	Changed(122)	Changed in changed(192)	Ignored

D:\Atwin32.dev\Samples\UIEX\UIEX4

```
464   END
465  END ELSE
466   *
467   ************************************************************************
468   *
469   * Try to read the customer record from the entered ID
470   *
471   GOSUB HIDEPIX ;* Hide previous cust picture before reading new one
472   GOSUB READ.RECORD
473   IF NOT(OK) THEN
474    *
475    ************************************************************************
476    *
477    * The attempt to read a record failed - assume the ID is a search string
478    *
479    CALL UIEX.GET.CUST.IDX(XID,ID,FN.CUST.XREF,FN.CUST)

480    GOSUB DSPSCRN ;* Refresh the screen after index lookup




481    *
482    ************************************************************************
483    *
484    * If the user did not select an item in the search routine, reprompt
485    *
486    IF XID EQ '' THEN RETURN
487    *
488    ************************************************************************
489    *
490    * Try to read the customer record from the selected ID
491    *
492    ID = XID
493    GOSUB HIDEPIX ;* Hide previous cust picture before reading new one
494    GOSUB READ.RECORD
495    *
496   END
497  END
498 END
499 *
500 ************************************************************************
501 *
502 * If success, display the new record, otherwise show warning message
503 *
504 IF OK THEN
505  GOSUB DSPDATA ;* Display new record

506 END ELSE
507  PRINT EL:TERM.DRV:'Please enter a valid customer ID!':TERM.NV:BEL:

508 END
509 RETURN
510 *
511 *
```

D:\Atwin32.dev\Samples\UIEX\UIEX5

```
518  * Try to read the customer record from the entered ID
519  *
520  GOSUB HIDEPIX ;* Hide previous cust picture before reading new one
521  GOSUB READ.RECORD
522  IF NOT(OK) THEN
523   *
524   ************************************************************************
525   *
526   * The attempt to read a record failed - assume the ID is a search string
527   *
528   CALL UIEX.GET.CUST.IDX.SUI(XID,ID,FN.CUST.XREF,FN.CUST,REFRESH,MAT TermDef
529   IF REFRESH THEN
530    GOSUB DSPSCRN ;* Refresh the screen if true windowing not supported
531   END ELSE
532    GOSUB SHOWPIX
533   END
534   *
535   ************************************************************************
536   *
537   * If the user did not select an item in the search routine, reprompt
538   *
539   IF XID EQ '' THEN RETURN
540   *
541   ************************************************************************
542   *
543   * Try to read the customer record from the selected ID
544   *
545   ID = XID
546   GOSUB HIDEPIX ;* Hide previous cust picture before reading new one
547   GOSUB READ.RECORD
548   *
549  END
550 END

551 *
552 ************************************************************************
553 *
554 * If success, display the new record, otherwise show warning message
555 *
556 IF OK THEN
557  GOSUB DSPDATA ;* Display new record
558  GOSUB DSPBAR ;* Refresh command bar - button states may have changed
559 END ELSE
560  CALL SUI.MESSAGE.BOX(30,10,40,4,'Please enter a valid customer ID.','OK','
561  IF REFRESH THEN GOSUB DSPSCRN ;* Refresh the screen if true windowing not s
562 END
563 RETURN
564 *
565 *
```

Found 44 differences: 256 lines, 370 inline differences in 122 changed lines				
Added(113,97)	Deleted(21,81)	Changed(122)	Changed in changed(192)	Ignored

D:\Atwin32.dev\Samples\UIEX\UIEX4	D:\Atwin32.dev\Samples\UIEX\UIEX5
512 **********	566 **********
513 *	567 *
514 * The READ.RECORD subroutine reads a new customer record from the file and	568 * The READ.RECORD subroutine reads a new customer record from the file and
515 * initializes the internal field data array (IDATA) from the record array	569 * initializes the internal field data array (IDATA) from the record array
516 * (CUST.REC). If the record does not exist, the routine returns with the	570 * (CUST.REC). If the record does not exist, the routine returns with the
517 * OK indicator variable set to FALSE. Otherwise OK is set to TRUE, and the	571 * OK indicator variable set to FALSE. Otherwise OK is set to TRUE, and the
518 * CUST.ID variable is set to the new ID.	572 * CUST.ID variable is set to the new ID.
519 *	573 *
520 READ.RECORD: *	574 READ.RECORD: *
521 *	575 *
522 MATREAD CUST.REC FROM FN.CUST,ID THEN	576 MATREAD CUST.REC FROM FN.CUST,ID THEN
523 CUST.ID = ID	577 CUST.ID = ID
524 IDATA(1) = CUST.ID	578 IDATA(1) = CUST.ID
525 IDATA(2) = CUST.CONTACT	579 IDATA(2) = CUST.CONTACT
526 IDATA(3) = CUST.NAME	580 IDATA(3) = CUST.NAME
527 IDATA(4) = CUST.ADDRESS1	581 IDATA(4) = CUST.ADDRESS1
528 IDATA(5) = CUST.ADDRESS2	582 IDATA(5) = CUST.ADDRESS2
529 IDATA(6) = CUST.CITY	583 IDATA(6) = CUST.CITY
530 IDATA(7) = CUST.ST	584 IDATA(7) = CUST.ST
531 IDATA(8) = CUST.ZIP	585 IDATA(8) = CUST.ZIP
532 IDATA(9) = CUST.COUNTRY	586 IDATA(9) = CUST.COUNTRY
533 IDATA(10) = CUST.PHONE	587 IDATA(10) = CUST.PHONE
534 IDATA(11) = CUST.FAX	588 IDATA(11) = CUST.FAX
535 IDATA(12) = CUST.HISTORY	589 IDATA(12) = CUST.HISTORY
536 OK = TRUE ;* Set the SUCCESS indicator	590 OK = TRUE ;* Set the SUCCESS indicator
537 END ELSE	591 END ELSE
538 OK = FALSE ;* Set the FAILURE indicator	592 OK = FALSE ;* Set the FAILURE indicator
539 END	593 END
540 RETURN	594 RETURN
541 *	595 *
542 *	596 *
543 **********	597 **********
544 *	598 *
	599 * The CANCEL.RECORD subroutine checks if any field data has changed, and
	600 * prompts if the user wants to abandon changes. If no changes, or the user
	601 * decides to abandon the changes, the DONE loop control varialbe is set to
	602 * TRUE causing the PROMPT and ACTION loops to terminate, proceeding to the
	603 * next record.
	604 *
	605 CANCEL.RECORD: *
	606 *
	607 GOSUB CHECK.ABANDON
	608 IF OK THEN DONE = TRUE ;* proceed to next record
	609 GOSUB HIDEPIX ;* erase the picture
	610 RETURN
	611 *
	612 *
	613 **********
	614 *
	615 * The NEW.RECORD subroutine checks if any field data has changed, and
	616 * prompts if the user wants to abandon changes. If no changes, or the
	617 * user decides to abandon the changes, the next ID is read from the
	618 * control file. The next ID is updated. The field data is cleared and the
	619 * new ID is displayed. The command bar is refreshed, since the command

Found 44 differences: 256 lines, 370 inline differences in 122 changed lines				
Added(113,97)	Deleted(21,81)	Changed(122)	Changed in changed(192)	Ignored

D:\Atwin32.dev\Samples\UIEX\UIEX4

```
545 * The DELETE.RECORD subroutine confirms that the user intends to delete
546 * the current record. If the action is confirmed, the CUST.DELETE
547 * subroutine is called to perform the deletion. A separate subroutine is
548 * used to handle updating indexes, etc.
549 *
550 DELETE.RECORD: *
551 *
552 IF CUST.ID NE '' THEN
553  PRINT EL:TERM.DRV:'Are you sure you want to delete this customer? ':TERM.NV
554  INPUT ANS:
555  IF ANS[1,1] EQ 'Y' OR ANS[1,1] EQ 'y' THEN
556   * Deletion has been confirmed - do the delete
557   OK = TRUE ;* Set the SUCCESS indicator
558   CALL UIEX.CUST.DELETE(CUST.ID,FN.CUST,FN.CUST.XREF)
559   DONE = TRUE ;* proceed to next record
560   GOSUB HIDEPIX ;* erase picture
561  END ELSE
562   OK = FALSE ;* Set the FAILURE indicator
563  END
564 END
565 RETURN
566 *
567 *
568 ************************************************************************
569 *
570 * The SAVE.RECORD subroutine copies internal field data from the IDATA
571 * array to the customer record array (CUST.REC). The CUST.UPDATE
```

D:\Atwin32.dev\Samples\UIEX\UIEX5

```
620 * button states may be different.
621 *
622 NEW.RECORD: *
623 *
624 GOSUB CHECK.ABANDON
625 IF OK THEN
626  * Get next sequential ID
627  READVU ID FROM FN.CUST.CTRL,'NEXT',2 THEN
628   WRITEV ID + 1 ON FN.CUST.CTRL,'NEXT',2
629  END ELSE
630   CALL SUI.MESSAGE.BOX(30,10,40,4,'Next item counter record not found!','OK'
631   IF REFRESH THEN GOSUB DSPSCRN
632   OK = FALSE
633   RETURN
634  END
635  GOSUB RESTART
636  IDATA(1) = ID
637  GOSUB DSPDATA
638  GOSUB DSPBAR
639  NXTFLD = 2
640 END
641 RETURN
642 *
643 *
644 ************************************************************************
645 *
646 * The DELETE.RECORD subroutine confirms that the user intends to delete
647 * the current record. If the action is confirmed, the CUST.DELETE
648 * subroutine is called to perform the deletion. A separate subroutine is
649 * used to handle updating indexes, etc.
650 *
651 DELETE.RECORD: *
652 *
653 IF CUST.ID NE '' THEN
654  YESNO = 2
655  CALL SUI.MESSAGE.BOX(30,10,40,4,'Are you sure you want to delete this custo
656  IF REFRESH THEN GOSUB DSPSCRN
657  IF YESNO = 1 THEN
658   * Deletion has been confirmed - do the delete
659   OK = TRUE ;* Set the SUCCESS indicator
660   CALL UIEX.CUST.DELETE(CUST.ID,FN.CUST,FN.CUST.XREF)
661   DONE = TRUE ;* proceed to next record
662   GOSUB HIDEPIX ;* erase picture
663  END ELSE
664   OK = FALSE ;* Set the FAILURE indicator
665  END
666 END
667 RETURN
668 *
669 *
670 ************************************************************************
671 *
672 * The SAVE.RECORD subroutine copies internal field data from the IDATA
673 * array to the customer record array (CUST.REC). The CUST.UPDATE
```

Found 44 differences: 256 lines, 370 inline differences in 122 changed lines				
Added(113,97)	Deleted(21,81)	Changed(122)	Changed in changed(192)	Ignored

D:\Atwin32.dev\Samples\UIEX\UIEX4

```
572 * subroutine is called to perform the update. A separate subroutine is
573 * used to handle updating indexes, etc.
574 *
575 SAVE.RECORD: *
576 *
577 IF IDATA(1) NE '' THEN
578  * Copy data from the internal field data array (IDATA) to the CUST.REC arra
579  CUST.ID = IDATA(1)
580  CUST.CONTACT = IDATA(2)
581  CUST.NAME = IDATA(3)
582  CUST.ADDRESS1 = IDATA(4)
583  CUST.ADDRESS2 = IDATA(5)
584  CUST.CITY = IDATA(6)
585  CUST.ST = IDATA(7)
586  CUST.ZIP = IDATA(8)
587  CUST.COUNTRY = IDATA(9)
588  CUST.PHONE = IDATA(10)
589  CUST.FAX = IDATA(11)
590  CUST.HISTORY = IDATA(12)
591  * Update the file
592  CALL UIEX.CUST.UPDATE(CUST.ID,MAT CUST.REC,FN.CUST,FN.CUST.XREF)
593  OK = TRUE ;* Set the SUCCESS indicator
594 END ELSE
595  OK = FALSE ;* Set the FAILURE indicator
596 END
597 DONE = TRUE ;* proceed to next record
598 RETURN
599 *
600 *
601 ************************************************************************
602 *
603 * The RESTART subroutine prepares the internal field data array (IDATA),
604 * customer record array (CUST.REC) and ID for a new customer record.
605 *
606 RESTART: *
607 *
608 CUST.ID = ''
609 MAT CUST.REC = ''
610 MAT IDATA = ''

611 RETURN
612 *
613 *
614 ************************************************************************
615 *
616 * The CHECK.CHANGED subroutine checks if any internal field data has been
617 * changed. The OK indicator variable is set to FALSE if any data is
618 * changed, otherwise it is set to TRUE. The ID field is not checked, since
619 * it is appropriately handled by the CHECK.ID subroutine.
620 *
621 CHECK.CHANGED: *
622 *
623 OK = FALSE
624 IF CUST.CONTACT # IDATA(2) THEN RETURN
```

D:\Atwin32.dev\Samples\UIEX\UIEX5

```
674 * subroutine is called to perform the update. A separate subroutine is
675 * used to handle updating indexes, etc.
676 *
677 SAVE.RECORD: *
678 *
679 IF IDATA(1) NE '' THEN
680  * Copy data from the internal field data array (IDATA) to the CUST.REC arra
681  CUST.ID = IDATA(1)
682  CUST.CONTACT = IDATA(2)
683  CUST.NAME = IDATA(3)
684  CUST.ADDRESS1 = IDATA(4)
685  CUST.ADDRESS2 = IDATA(5)
686  CUST.CITY = IDATA(6)
687  CUST.ST = IDATA(7)
688  CUST.ZIP = IDATA(8)
689  CUST.COUNTRY = IDATA(9)
690  CUST.PHONE = IDATA(10)
691  CUST.FAX = IDATA(11)
692  CUST.HISTORY = IDATA(12)
693  * Update the file
694  CALL UIEX.CUST.UPDATE(CUST.ID,MAT CUST.REC,FN.CUST,FN.CUST.XREF)
695  OK = TRUE ;* Set the SUCCESS indicator
696 END ELSE
697  OK = FALSE ;* Set the FAILURE indicator
698 END
699 DONE = TRUE ;* proceed to next record
700 RETURN
701 *
702 *
703 ************************************************************************
704 *
705 * The RESTART subroutine prepares the internal field data array (IDATA),
706 * customer record array (CUST.REC) and ID for a new customer record.
707 *
708 RESTART: *
709 *
710 CUST.ID = ''
711 MAT CUST.REC = ''
712 MAT IDATA = ''
713 BTN = 5 ;* Set default button to "cancel"
714 RETURN
715 *
716 *
717 ************************************************************************
718 *
719 * The CHECK.CHANGED subroutine checks if any internal field data has been
720 * changed. The OK indicator variable is set to FALSE if any data is
721 * changed, otherwise it is set to TRUE. The ID field is not checked, since
722 * it is appropriately handled by the CHECK.ID subroutine.
723 *
724 CHECK.CHANGED: *
725 *
726 OK = FALSE
727 IF CUST.CONTACT # IDATA(2) THEN RETURN
```

Found 44 differences: 256 lines, 370 inline differences in 122 changed lines				
Added(113,97)	Deleted(21,81)	Changed(122)	Changed in changed(192)	Ignored

D:\Atwin32.dev\Samples\UIEX\UIEX4

```
625 IF CUST.NAME # IDATA(3) THEN RETURN
626 IF CUST.ADDRESS1 # IDATA(4) THEN RETURN
627 IF CUST.ADDRESS2 # IDATA(5) THEN RETURN
628 IF CUST.CITY # IDATA(6) THEN RETURN
629 IF CUST.ST # IDATA(7) THEN RETURN
630 IF CUST.ZIP # IDATA(8) THEN RETURN
631 IF CUST.COUNTRY # IDATA(9) THEN RETURN
632 IF CUST.PHONE # IDATA(10) THEN RETURN
633 IF CUST.FAX # IDATA(11) THEN RETURN
634 IF CUST.HISTORY # IDATA(12) THEN RETURN
635 OK = TRUE
636 RETURN
637 *
638 *
639 ************************************************************************
640 *
641 * The CHECK.ABANDON subroutine calls CHECK.CHANGED. If any changes are
642 * found, a message is displayed and the user is prompted to abandon the
643 * changes. If there are no changes, or if the user decides to abandon
644 * changes, the OK indicator variable is set to TRUE. Otherwise it is set
645 * to FALSE.
646 *
647 CHECK.ABANDON: *
648 *
649 GOSUB CHECK.CHANGED
650 IF NOT(OK) THEN
651  PRINT EL:TERM.DRV:'Do you want to abandon all your changes? ':TERM.NV:BEL:
652  INPUT ANS:
653  IF ANS[1,1] EQ 'Y' OR ANS[1,1] EQ 'y' THEN OK = 1
654  PRINT EL:
655 END
656 RETURN
657 *
658 *
659 ************************************************************************
660 *
661 * The DSPSCRN subroutine is used to refresh the entire screen.
662 *
663 DSPSCRN: *
664 *
665 * Clear the screen
666 PRINT TERM.CLEAR:
667 *
668 * If running AccuTerm, display the logo
669 IF IMGFLG THEN
670  PRINT ESC:STX:'iL,':PIX.PATH:'ASE-LOGO-SMALL.JPG,65,0,12,2,0,N':CR:
671 END
672 *
673 * Display heading & labels
674 CALL SUI.DISPLAY.LABEL(5, 0, 0, 'L', 0, 'Customer File Maintenance', CMD, MA
675 FOR XLINE = 1 TO NUMFLDS
676  LBWD = CONTROL(XLINE)<1,4>
677  LBTX = (XLINE 'L#2 ') : CONTROL(XLINE)<1,1> ;* Prepend line number to label
678  IF LBWD > 0 THEN LBTX = (LBTX : STR('.',LBWD))[1,LBWD] ;* Pad label with dc
```

D:\Atwin32.dev\Samples\UIEX\UIEX5

```
728 IF CUST.NAME # IDATA(3) THEN RETURN
729 IF CUST.ADDRESS1 # IDATA(4) THEN RETURN
730 IF CUST.ADDRESS2 # IDATA(5) THEN RETURN
731 IF CUST.CITY # IDATA(6) THEN RETURN
732 IF CUST.ST # IDATA(7) THEN RETURN
733 IF CUST.ZIP # IDATA(8) THEN RETURN
734 IF CUST.COUNTRY # IDATA(9) THEN RETURN
735 IF CUST.PHONE # IDATA(10) THEN RETURN
736 IF CUST.FAX # IDATA(11) THEN RETURN
737 IF CUST.HISTORY # IDATA(12) THEN RETURN
738 OK = TRUE
739 RETURN
740 *
741 *
742 ************************************************************************
743 *
744 * The CHECK.ABANDON subroutine calls CHECK.CHANGED. If any changes are
745 * found, a message is displayed and the user is prompted to abandon the
746 * changes. If there are no changes, or if the user decides to abandon
747 * changes, the OK indicator variable is set to TRUE. Otherwise it is set
748 * to FALSE.
749 *
750 CHECK.ABANDON: *
751 *
752 GOSUB CHECK.CHANGED
753 IF NOT(OK) THEN
754  YESNO = 2
755  CALL SUI.MESSAGE.BOX(25,10,50,4,'Do you want to abandon all your changes?',
756  IF REFRESH THEN GOSUB DSPSCRN
757  IF YESNO = 1 THEN OK = TRUE
758 END
759 RETURN
760 *
761 *
762 ************************************************************************
763 *
764 * The DSPSCRN subroutine is used to refresh the entire screen.
765 *
766 DSPSCRN: *
767 *
768 * Clear the screen
769 PRINT TERM.CLEAR:
770 *
771 * If running AccuTerm, display the logo
772 IF IMGFLG THEN
773  PRINT ESC:STX:'iL,':PIX.PATH:'ASE-LOGO-SMALL.JPG,65,0,12,2,0,N':CR:
774 END
775 *
776 * Display heading & labels
777 CALL SUI.DISPLAY.LABEL(30, 0, 0, 'C', 0, 'Customer File Maintenance', CMD, M
778 FOR XLINE = 1 TO NUMFLDS
779  LBWD = CONTROL(XLINE)<1,4>
780  LBTX = CONTROL(XLINE)<1,1>
781  IF LBWD > 0 THEN LBTX = (LBTX : STR('.',LBWD))[1,LBWD] ;* Pad label with dc
```

Found 44 differences: 256 lines, 370 inline differences in 122 changed lines				
Added(113,97)	Deleted(21,81)	Changed(122)	Changed in changed(192)	Ignored

D:\Atwin32.dev\Samples\UIEX\UIEX4

```
679  CALL SUI.DISPLAY.LABEL(CONTROL(XLINE)<1,2>, CONTROL(XLINE)<1,3>, LBWD, 'L',
680 NEXT XLINE
681 *
682 * Display the field data
683 GOSUB DSPDATA
684 RETURN
685 *
686 *
687 ************************************************************************
688 *
689 * The DSPDATA subroutine is used to refresh the field data for all fields.
690 *
691 DSPDATA: *
692 *
693 FOR XLINE = 1 TO NUMFLDS
694  GOSUB DSPLINE
695 NEXT XLINE
696 GOSUB SHOWPIX
697 RETURN
698 *
699 *
700 ************************************************************************
701 *
702 * The DSPLINE subroutine is used to refresh the field data for one field.
703 * The field to be refreshed is specified by the XLINE variable.
704 *
705 DSPLINE: *
706 *
707 CALL SUI.DISPLAY.TEXT(CONTROL(XLINE)<1,5>, CONTROL(XLINE)<1,6>, CONTROL(XLIN
708 RETURN
709 *
710 *
711 ************************************************************************
712 *
713 * If running AccuTerm and an customer ID has been entered, try to display
714 * the customer's picture for the current record.
```

D:\Atwin32.dev\Samples\UIEX\UIEX5

```
782  CALL SUI.DISPLAY.LABEL(CONTROL(XLINE)<1,2>, CONTROL(XLINE)<1,3>, LBWD, 'L',
783 NEXT XLINE
784 *
785 * Display the field data
786 GOSUB DSPDATA
787 *
788 * Display the command bar
789 GOSUB DSPBAR
790 *
791 RETURN
792 *
793 *
794 ************************************************************************
795 *
796 * The DSPDATA subroutine is used to refresh the field data for all fields.
797 *
798 DSPDATA: *
799 *
800 FOR XLINE = 1 TO NUMFLDS
801  GOSUB DSPLINE
802 NEXT XLINE
803 GOSUB SHOWPIX
804 RETURN
805 *
806 *
807 ************************************************************************
808 *
809 * The DSPLINE subroutine is used to refresh the field data for one field.
810 * The field to be refreshed is specified by the XLINE variable.
811 *
812 DSPLINE: *
813 *
814 CALL SUI.DISPLAY.TEXT(CONTROL(XLINE)<1,5>, CONTROL(XLINE)<1,6>, CONTROL(XLIN
815 RETURN
816 *
817 *
818 ************************************************************************
819 *
820 * The DSPBAR subroutine is used to refresh the command bar. The state of
821 * the Save and Delete buttons is adjusted depending on whether a record
822 * has been read or an item-ID has been entered.
823 *
824 DSPBAR: *
825 *
826 IF IDATA(1) NE '' THEN BTNSTATE<1,3> = 0 ELSE BTNSTATE<1,3> = 2
827 IF CUST.ID NE '' THEN BTNSTATE<1,4> = 0 ELSE BTNSTATE<1,4> = 2
828 CALL SUI.DISPLAY.BTNBAR(10, 22, 60, 1, BTNSTATE, BTNTEXT, BTNKEY, '', CMD, M
829 RETURN
830 *
831 *
832 ************************************************************************
833 *
834 * If running AccuTerm and an customer ID has been entered, try to display
835 * the customer's picture for the current record.
```

Found 44 differences: 256 lines, 370 inline differences in 122 changed lines				
Added(113,97)	Deleted(21,81)	Changed(122)	Changed in changed(192)	Ignored

D:\Atwin32.dev\Samples\UIEX\UIEX4

```
715 *
716 SHOWPIX: *
717 *
718 IF IMGFLG THEN
719  IF CUST.ID NE '' THEN
720   PRINT ESC:STX:'iL,':PIX.PATH:'CUST\':CUST.ID:'.JPG,60,3,15,8,1,B':CR:
721  END
722 END
723 RETURN
724 *
725 *
726 ********************************************************************************
727 *
728 * Erase the current customer picture before calling the lookup screen
729 * because pictures always display over text.
730 *
731 HIDEPIX: *
732 *
733 IF IMGFLG THEN
734  IF CUST.ID NE '' THEN
735   PRINT ESC:STX:'iD,':PIX.PATH:'CUST\':CUST.ID:'.JPG':CR:
736  END
737 END
738 RETURN
739 *
740 *
741 ********************************************************************************
742 *
743 * The CHECK.MOUSE subroutine is called if any SUI input routine returns
744 * with CMD set to TERM.LEFTBUTTON$. This routine checks each field,
745 * passing the field position and size to SUI.MOUSE.HIT. If a "hit" is
746 * detected, the NXTFLD variable is set to the field number of the clicked
747 * field. If no field was clicked, the position and size of the command bar
748 * is checked, and if "hit", NXTFLG is set to NUMFLDS + 1 to cause the
749 * ACTION and PROMPT loops to prompt for action using the command bar. The
750 * actual mouse click location is saved internally by the SUI input
751 * routines, and if a command button is clicked while a different field is
752 * active, the click action is executed when SUI.INPUT.BTNBAR is called.
753 *
754 CHECK.MOUSE: *
755 *
756 FOR XLINE = 1 TO NUMFLDS
757  CALL SUI.MOUSE.HIT(CONTROL(XLINE)<1,5>, CONTROL(XLINE)<1,6>, CONTROL(XLINE)
758  IF HIT THEN
759   NXTFLD = XLINE
760   RETURN
761  END
762 NEXT XLINE
763 RETURN
764 *
```

D:\Atwin32.dev\Samples\UIEX\UIEX5

```
836 *
837 SHOWPIX: *
838 *
839 IF IMGFLG THEN
840  IF CUST.ID NE '' THEN
841   PRINT ESC:STX:'iL,':PIX.PATH:'CUST\':CUST.ID:'.JPG,60,3,15,8,1,B':CR:
842  END
843 END
844 RETURN
845 *
846 *
847 ********************************************************************************
848 *
849 * Erase the current customer picture before calling the lookup screen
850 * because pictures always display over text, and don't get along well
851 * with windowing.
852 *
853 HIDEPIX: *
854 *
855 IF IMGFLG THEN
856  IF CUST.ID NE '' THEN
857   PRINT ESC:STX:'iD,':PIX.PATH:'CUST\':CUST.ID:'.JPG':CR:
858  END
859 END
860 RETURN
861 *
862 *
863 ********************************************************************************
864 *
865 * The CHECK.MOUSE subroutine is called if any SUI input routine returns
866 * with CMD set to TERM.LEFTBUTTON$. This routine checks each field,
867 * passing the field position and size to SUI.MOUSE.HIT. If a "hit" is
868 * detected, the NXTFLD variable is set to the field number of the clicked
869 * field. If no field was clicked, the position and size of the command bar
870 * is checked, and if "hit", NXTFLG is set to NUMFLDS + 1 to cause the
871 * ACTION and PROMPT loops to prompt for action using the command bar. The
872 * actual mouse click location is saved internally by the SUI input
873 * routines, and if a command button is clicked while a different field is
874 * active, the click action is executed when SUI.INPUT.BTNBAR is called.
875 *
876 CHECK.MOUSE: *
877 *
878 FOR XLINE = 1 TO NUMFLDS
879  CALL SUI.MOUSE.HIT(CONTROL(XLINE)<1,5>, CONTROL(XLINE)<1,6>, CONTROL(XLINE)
880  IF HIT THEN
881   NXTFLD = XLINE
882   RETURN
883  END
884 NEXT XLINE
885 * Check if a button on the command bar was clicked
886 CALL SUI.MOUSE.HIT(10, 22, 60, 1, 0, HIT, MAT TermDef)
887 IF HIT THEN NXTFLD = NUMFLDS + 1
888 RETURN
889 *
```

Found 44 differences: 256 lines, 370 inline differences in 122 changed lines				
Added(113,97)	Deleted(21,81)	Changed(122)	Changed in changed(192)	Ignored

D:\Atwin32.dev\Samples\UIEX\UIEX4		D:\Atwin32.dev\Samples\UIEX\UIEX5	
765	END	890	END
766		891	

Found 44 differences: 256 lines, 370 inline differences in 122 changed lines				
Added(113,97)	Deleted(21,81)	Changed(122)	Changed in changed(192)	Ignored