THINCLIENT

Enhanced Screen library functions

Windows Version 5.6
Conetic Software Systems, Inc.
August 2005

Problem

Porting legacy "unix/linux" terminal based "c" programs to the Windows ThinClient typically requires some rewrite of "c" code. In an efford to add more functionality, while also reducing the amount of programming for migration to windows, Conetic has developed a series of new library functions. These new functions have been created to address the "major" issue -- "output directly to the screen from within a Conetic form" or "use of the c function `printf`".

Brief history – for years Conetic programmers have used the standard "c" function "printf" to print data to a screen. In unix/linux this is not a problem. However, in MS-Windows there is NO such thing as a "printf" function. This is because windows does not function like unix. For the most part all Windows screens are painted by passing messages back and forth between programs. Thus, requiring any program that performs a "printf" to be re-engineered.

Solution

Extend the C/Base Utilities libraries to contain similar unix/linux terminal functionality. The new functions are:

```
escape
tcprintf
tcputs
tcputchar
tcclear
tcsetwhite
tcask
tcmessagebox
tcmessage
tclineedit
send_form_message
```

Detail description, syntax, and sample code are on the following pages.

escape

C/Base ThinClient screen escape function.

New commands of C/Base escape function to support Thinclient user text:

```
UCLEAREOL  // clear text from current column to the right side of screen
UCLEAREOS  // clear text from current row and column to the bottom of screen
UCLEARALL // clear all user text
```

Examples:

clear text before print:

```
escape(MOVCUR,10,20);
escape(UCLEAREOL);
tcprintf("Print user text at row=%d col=%d",10,20);
```

tcprintf

Print a formated string value at form current cursor location.

syntex

```
int tcprintf(const char *fmt, ...);
```

Parameters

fmt

print format, same as c printf

Return Value

Thinclient form screen field id.

Remarks

tcprintf prints a formated string value at form current cursor location. The format string *fmt* is identical to the standard c function printf. It returns a Thinclient form screen field id to user program to set text properties with **send_form_message** or to clear text with **tcclear**.

Examples

print a blue color text at row 10 and column 20:

```
int fieldid;
escape(MOVCUR,10,20); // move cursor to row 10 and column 20
fieldid=tcprintf("Print user text at row=%d col=%d",10,20);
// print
// set text color to blue
send_form_message(fieldid,USR_M_SETBGCOLOR,0,0,
     USR_RGBTYPE,USR_RGBSIZE,RGBCOLOR(0,0,255),
          USR_NONETYPE,0,NULL);
```

print text with underline:

```
escape(MOVCUR,10,20);
escape(UNDERLINE);
tcprintf("Print user text at row=%d col=%d",10,20);
escape(ENDUNDERLINE);
```

tcputs

Print text value at form current cursor location.

Syntex

```
int tcputs(const char* text);
```

Parameters

text

text to print on form screen

Return Value

Thinclient form screen field id.

Remarks

tcputs prints a zero terminated string value at form current cursor location. It returns a Thinclient form screen field id to user program to set text properties with **send_form_message** or to clear text with **tcclear**.

Example

print a text with hilighted background:

```
tcsetwhite(0,0,0);          // set backtround to black
escape(MOVCUR,10,20); // move cursor to row 10 and column 20
escape(WHITE);        // turn on hilight mode
tcputs("Print user text"); // print text
escape(ENDWHITE);     // turn off hilight mode
```

tcputchar

Print one character at form current cursor location.

Syntex

```
int tcputchar(int c);
```

Parameters

c

character to print on form screen

Return Value

Thinclient form screen field id.

Remarks

tcputchar prints one character at form current cursor location. It returns a Thinclient form screen field id to user program to set text properties with **send_form_message** or to clear text with **tcclear**.

Example

print three characters and clear the middle one:

```
int fieldid;
escape(MOVCUR,10,20);    // move cursor to row 10,column 20
tcputchar('a');
fieldid=tcputchar('b');
                  //print character b,then store screen field id
tcputchar('c');
tcclear(fieldid); // clear letter b with the screen field id
```

tcclear

Clear user text and delete screen field.

Syntex

```
void tcclear(int fieldid);
```

Parameters

fieldid

screen field id

Return Value

no return value

Remarks

Clear user text on form screen. Always clear existing text before printing over the same location.

tcsetwhite

Set user text background color.

Syntex

```
void tcsetwhite(int red, int green, int blue);
```

Parameters

red

red color value (0 to 255)

green

green color value (0 to 255)

blue

blue color value (0 to 255)

Return value

no return value

Remarks

Set user text background color (WHITE color). The default WHITE color is RGB color 255,255,255. Call escape(WHITE) before print text and escape(ENDWHITE) to reset background color.

Examples

print text with background color:

```
tcsetwhite(255,0,0);    // set text background color to red
escape(MOVCUR,10,20);
escape(WHITE);          // start background color mode
tcputs("Print user text"); //print text with red background color.
escape(ENDWHITE);       // end background color mode
```

tcask

Ask for user input value with supported data types.

Syntex

```
char *tcask(char *prompt, int datatype, char *ans, int anslen,
char *defans);
```

Parameters

prompt

text promnt/question

datatype

answer data value type

ans

buffer for returning answer

anslen

answer buffer size

defans

preset default value for answer

Return value

On success (Ok), returns 0 terminated string. On error (Cancel), returns an empty string.
If data type mismatch, display Thinclient error message. The supported data types are: STRING_TYPE, CHAR_TYPE, INT_TYPE, LONG_TYPE, REAL_TYPE, MONEY_TYPE, DATE_TYPE, TIME_TYPE, BOOLEAN_TYPE. Please check C/Base reference on data type formats for detail.

Examples

ask user to enter a date value:

```
char ans[20];
tcask("What is today's date?",DATE_TYPE,ans,sizeof(ans),NULL);
```

ask user to enter a command:

```
char ans[128];
tcask("Please enter a command:",STRING_TYPE,ans,sizeof(ans),
"grace myrpt");
```

tcmessagebox

Display user message box

Syntax

```
int tcmessagebox(char *text, char *caption, int ntype);
```

Parameters

text

user message

caption

message box title

ntype

message box button and icon types

Return value

message box button id.

```
USR_MB_IDOK         // Ok button clicked
USR_MB_IDCANCEL     // Cancel button clicked
USR_MB_IDYES                // Yes button clicked
USR_MB_IDNO         // No button clicked
```

Remarks

Display user message with combination (bitwise or) of command pushbuttons and icons.

Buttons:

```
USR_MB_OK                   // Ok button
USR_MB_OKCANCEL             // Ok and Cancel buttons
USR_MB_YESNO                        // YES and NO buttons
USR_MB_YESNOCANCEL          // YES, NO and Cancel buttons
```

Icons:

```
USR_MB_ICONEXCLAMATION          // Exclamation icon
USR_MB_ICONQUESTION         // Question icon
USR_MB_ICONINFORMATION  // Information Icon
USR_MB_ICONERROR            // Error icon
```

Examples

user information:

```
tcmessagebox("Inventory low for product #c1023","Warning...",
      MB_OK|MB_ICONINFORMATION);
```

answer question:

```
      if (tcmessagebox("Item not committed. Do you wish to
continue?","Question...",
            USR_MB_YESNO|USR_MB_ICONQUESTION)==USR_MB_IDYES)
      {
            // Yes
      }
      else
      {
            // No
      }
```

tcmessage

Display Thinclient message

Syntax

```
void tcmessage(char *msgstr);
```

Parameters

msgstr

user message

Return Value

No return value

Remarks

Display user message in modaless Thinclient message window.

tclineedit

display a line edit control box to ask user input.

Syntax

```
int tclineedit(int width, char *linebuf, int bufsize, char *deftext);
```

Parameters

width

line edit control width in character

linebuf

returning text line buffer

bufsize

line buffer size

deftext

default text

Return Value

Thinclient form screen field id.

Remarks

tclineedit display a line edit control box to ask for user input. When one hits Enter key, It returns the user input text to passed-in string buffer *linebu*f. After tclineedit returns, the line edit control changes to regular user text control with the same screen field id.

Examples

```
      char linebuf[128];
      int fieldid;

      escape(MOVCUR,10,20);                // locate line edit label
      tcputs("Please enter a command:"); // print line edit label
      escape(MOVCUR,10,45);               // locate line edit control
         // display 36 character line edit box with default command
"grace myrpt"
      fieldid=tclineedit(36,linebuf,sizeof(linebuf),"grace myrpt");
```

send_form_message

send message to form user control to get/set control properties.

Syntax

```
int send_form_message(int fieldid,int message,int rparam1,int rparam2,
      int lparam1type, int lparam1size, void *lparam1,
      int lparam2type, int lparam2size, void *lparam2);
```

Parameters

fieldid

user screen field(control) id

message

control message

rparam1

integer value 1. Not used

rparam2

integer value 2. Not used

lparam1type

lparam1 data type

lparam1size

lparam1 data size

lparam1

lparam1 data value

lparam2type

lparam2 data type. Not used

lparam2size

lparam2 data size. Not used

lparam2

lparam2 data value. Not used

Return Value

On success, return 0. On error, return 1.

Remarks

Send messages to user control to set control properties.

Supported messages:

```
USR_M_SETFGCOLOR          // set foreground color
USR_M_SETBGCOLOR          // set background color
USR_M_SETFONT                   // set font
USR_M_SETUNDERLINE              // set underline
```

Examples:

set text font:

```
int fieldid;
char strbuf[128];

escape(MOVCUR,10,20);
fieldid=tcputs("Print user text");
sprintf(strbuf,"Courier New,%d,%d",
      14,USR_FONT_BOLD|USR_FONT_ITALIC);

// set text font to "Courier New", point size 14, bold and italic
send_form_message(fieldid,USR_M_SETFONT,0,0,
      USR_CHARTYPE,sizeof(strbuf),strbuf,
      USR_NONETYPE,0,NULL);
```

set foreground color

```
int fieldid;
escape(MOVCUR,10,20);
fieldid=tcputs("Print user text");

// set text foreground color to blue
send_form_message(fieldid,USR_M_SETFGCOLOR,0,0,
      USR_RGBTYPE,USR_RGBSIZE,RGBCOLOR(0,0,255),
      USR_NONETYPE,0,NULL);
```