



# 使用説明書

ソフトウェアエンジニア	ハードウェアエンジニア	エディター
Robert	Peng Jun Sam Zeng	Emma



## 目次

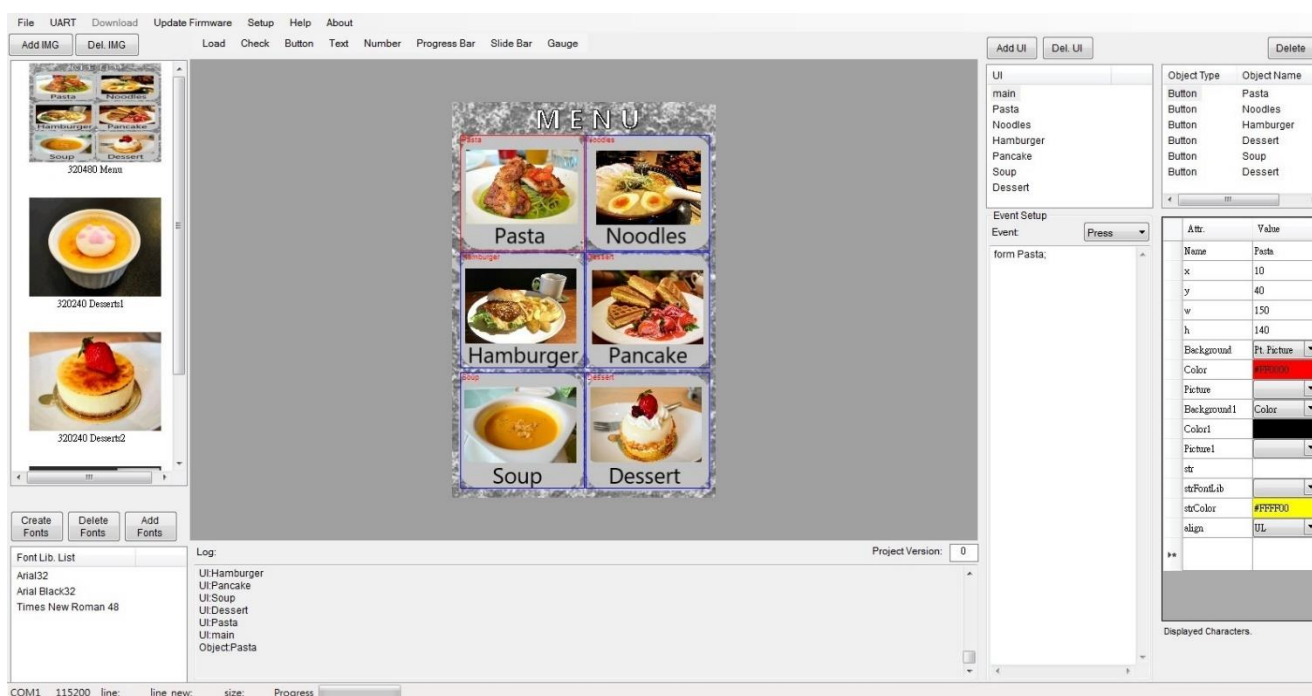
番号	説明	ページ
	表紙	1
	改訂記録	2
	目次	3
1	適応範囲	4
2	概要	4
3	ユーザーインターフェース	4
4	ライブラリ	5
5	機能	6
6	ウィジェット	7
7	プログラミング	11
8	SD カードでプロジェクトの更新プロセス	17

## 1. 適応範囲

本マニュアルは、AMSON ELECTRONICS が提供する UART TFT LCD モジュール用の UI 編集ツールの基本的な使い方を説明するものです。

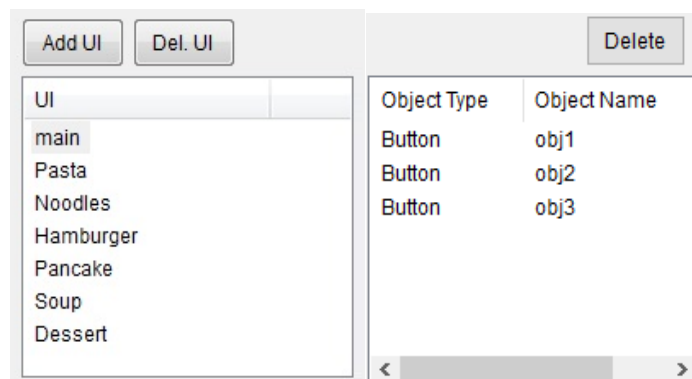
## 2. 概要

ユーザーインターフェースをデザインするためのいくつかのウィジェットがあります。  
ウィジェットをクリックすると、対応するオブジェクトが真ん中のスクリーンに表示されます。  
ユーザーは右側の選択項目に、位置やサイズ、テキストなど様々な属性を設定することができます。



## 3. ユーザーインターフェース

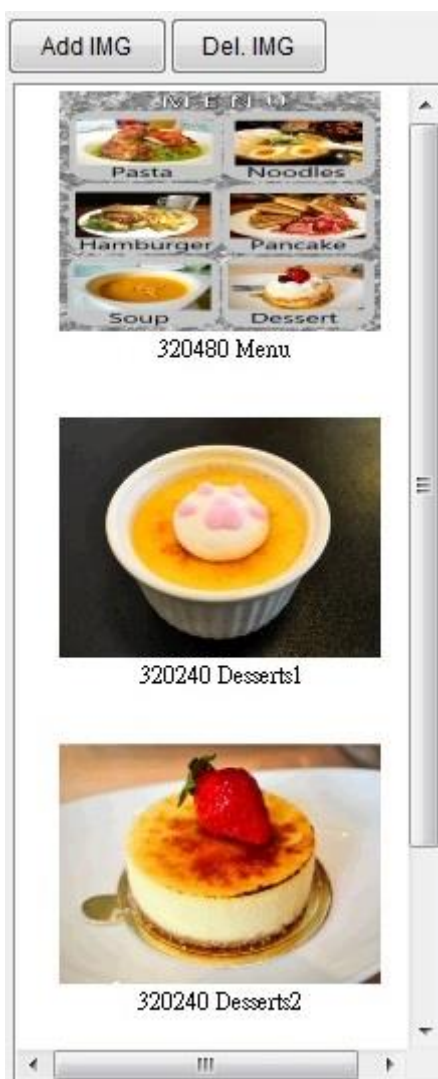
ユーザーは、新しいユーザーインターフェース(UI)を作成して、画像やボタン、ゲージなどを入力することができます。なお、UI の最大数は 255 個です。



\* 各 UI には最大 32 個のオブジェクトに制限されています。

## 4. ライブラリ

ユーザーは最大 40 枚の画像を保存できます。ただし、UI の背景に Ext. Picture を選択する場合、画像の解像度は LCD モジュールと同じである必要があります。



ユーザーはテキストを入力する前にフォントを作成する必要があります。



\* 各ファイルの名前の長さは、最大 16 文字の ASCII 文字に制限されています。

## 5. 機能

アイテム	属性
File	新規、開く、保存と終了。
UART	LCD モジュールをコンピューターに接続してから、対応するポートとボーレートを選択してください。 ボーレートとは、LCD モジュールとホスト間の通信ボーレートです。
Update Firmware	*.bin ファイルを開き、ファームウェアをアップデートします。
Setup	未定。
Help	UI 編集ツールマニュアルへのリンク。
About	ソフトウェアのバージョンの表記。
Load	プロジェクトを LCD モジュールにダウンロードします。
Check	オブジェクトを選択し、その Event のウィンドウに書いたコードを検査します。 Simulation はシミュレーターとして機能します。

## 6. ウィジェット

アイテム

属性

ボタン

位置、背景(色、画像または透明)、str(文字列)を設定することで、シンプルなカラーボタンや Ext. Picture による自作ボタンを作成できます。

obj0

Send

↓ press

Send

obj0

MENU

↓ press

MENU

Background	Color
Color	#808080
Picture	
Background1	Color
Color1	#C0C0C0
Picture1	
str	Send
strFontLib	Arial48
strColor	#000080
align	CC




Background	Ext. Picture
Color	
Picture	BUTTON1
Background1	Ext. Picture
Color1	
Picture1	BUTTON2

Event のウィンドウでは、ボタンが押されたとき、または、離されたときのプログラムを設定できます。

Event Setup

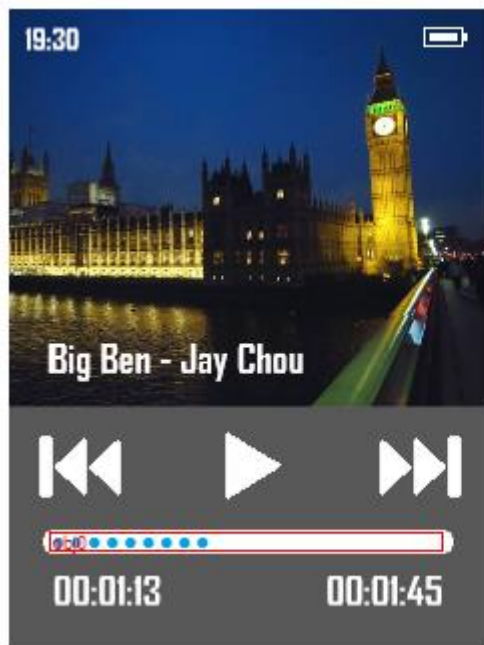
Event: Press

form Pasta;

<p>テキスト</p>	<p>位置、背景、フォントの色を設定し、フォントライブラリから一つを選択する必要があります。</p> <div data-bbox="544 488 954 595" data-label="Image">  </div> <div data-bbox="986 371 1324 707" data-label="Table"> <table> <tr> <td>Background</td><td>Color</td></tr> <tr> <td>Color</td><td>#80FF80</td></tr> <tr> <td>Picture</td><td></td></tr> <tr> <td>str</td><td>AMSON</td></tr> <tr> <td>strFontLib</td><td>1</td></tr> <tr> <td>strColor</td><td>#0080FF</td></tr> <tr> <td>align</td><td>CC</td></tr> </table> </div>	Background	Color	Color	#80FF80	Picture		str	AMSON	strFontLib	1	strColor	#0080FF	align	CC						
Background	Color																				
Color	#80FF80																				
Picture																					
str	AMSON																				
strFontLib	1																				
strColor	#0080FF																				
align	CC																				
<p>数値</p>	<p>数値は、整数または浮動小数点数にすることができます。ただし、計算は小数点の桁数が同じである必要があります。例えば、obj.0 の値が 20.12 の場合、計算には小数点の後に 2 つの数字が必要です。</p> <div data-bbox="539 1059 884 1162" data-label="Image">  </div> <div data-bbox="919 947 1334 1249" data-label="Image">  </div>																				
<p>プログレスバー</p>	<p>シンプルなプログレスバー（進行状況バー）を作成するために、背景と PColor の色を設定できます。</p> <div data-bbox="544 1686 943 1749" data-label="Image">  </div> <div data-bbox="978 1480 1313 1964" data-label="Table"> <table> <tr> <td>Background</td><td>Color</td></tr> <tr> <td>Color</td><td>#80FFFF</td></tr> <tr> <td>Picture</td><td></td></tr> <tr> <td>Progress</td><td>Color</td></tr> <tr> <td>PColor</td><td>#0080FF</td></tr> <tr> <td>PPicture</td><td></td></tr> <tr> <td>Orientation</td><td>Horizontal</td></tr> <tr> <td>Maximum</td><td>100</td></tr> <tr> <td>Minimum</td><td>0</td></tr> <tr> <td>Value</td><td>66</td></tr> </table> </div> <p>(次ページ参照。)</p>	Background	Color	Color	#80FFFF	Picture		Progress	Color	PColor	#0080FF	PPicture		Orientation	Horizontal	Maximum	100	Minimum	0	Value	66
Background	Color																				
Color	#80FFFF																				
Picture																					
Progress	Color																				
PColor	#0080FF																				
PPicture																					
Orientation	Horizontal																				
Maximum	100																				
Minimum	0																				
Value	66																				



別のデザインにするには、フル解像度の画像を用意して、対応する座標にプログレスバーを配置する必要があります。背景の色、Progress の Pt. Picture と既存の画像が利用できます。



Background	Color
Color	#FFFFFF
Picture	
Progress	Pt. Picture
PColor	
PPicture	240320 ...
Orientation	Horizontal
Maximum	100
Minimum	0
Value	41

シンプルなスライダーバーを作るために、背景と SColor の色を設定できます。



Background	Color
Color	#CACACA
Picture	
Slider	Color
SColor	#008000
SPicture	slider
Orientation	Horizontal
Width	30
Maximum	100
Minimum	0
Value	60

ボタンと同じく、Event のウィンドウでプログラムを設定できます。

(次ページ参照。)

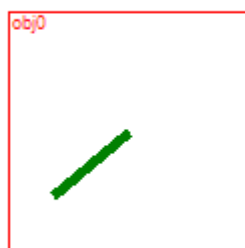
スライダーバー

別のデザインにするには、特定の背景とスライダーを組み合わせるために、Ext. Picture で既存の画像を選択することもできます。



Background	Ext. Picture ▼
Color	
Picture	slider bar ▼
Slider	Ext. Picture ▼
SColor	
SPicture	slider ▼
Orientation	Horizontal ▼
Width	40
Maximum	100
Minimum	0
Value	60

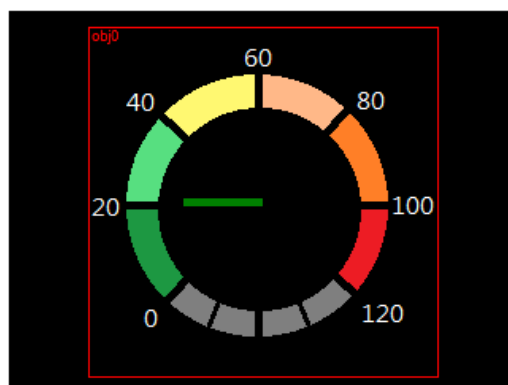
背景と GColor の色を設定して、シンプルなゲージを作ることができます。



Background	Color ▼
Color	#FFFFFF
Picture	▼
Gauge	Color ▼
GColor	#008000
Length	50
Thickness	5
Value	320

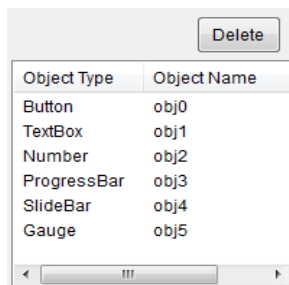
## ゲージ

その他の背景はデザインによって、Ext. Picture、透明または Pt. Picture が選択できます。



Background	Ext. Picture ▼
Color	
Picture	Gauge ▼
Gauge	Color ▼
GColor	#008000
Length	50
Thickness	5
Value	0

右上隅のウィンドウには各 UI のオブジェクトがあり、そこから各オブジェクトの設定が配置できます。



## 7. プログラミングー全ては大文字と小文字が区別されます。

### HEX コマンド

機能	プロトコル ヘッダ (4Bytes)	予約 (1Byte)	コマンド タイプ (1Byte)	データ の長さ (1Byte)	データ (n Bytes)	CRC (2Bytes)
バックライトの PWM 調整	FELP	0x00	11	2	0xA8+0x00(0%=off)～0x0A(100%) 例: 10%に変更  0x46,0x45,0x4C,0x50,0x00,0x0B,0x02,0xA8,0x01  PWM 調整範囲、合計 10 段階： 0x01=10%; 0x05=50%; 0x0A=100%	CRC 2Bytes
バックライトの PWM 調整の応 答	FELP	0x00	11	1	0x00=成功； それ以外の場合は失敗	CRC 2Bytes
LCD Display On/ off	FELP	0x00	12	1	0x00=LCD Display off. 0x46,0x45,0x4C,0x50,0x00,0x0C,0x01,0x00  0x01=LCD Display on. 0x46,0x45,0x4C,0x50,0x00,0x0C,0x01,0x01	CRC 2Bytes
LCD Display On/ off の応答	FELP	0x00	12	1	0x00=成功； 0x01=失敗	CRC 2Bytes
シフト UI	FELP	0x00	36	後続データ の長さ	UI 名の長さ + UI 名	CRC 2Bytes
シフト UI の応答	FELP	0x00	36	1	0x00=成功； それ以外の場合は失敗	CRC 2Bytes
オブジェクト 文字列の属性を 変更	FELP	0x00	37	後続データ の長さ	UI 名の長さ + UI 名 + オブジェクト名の長さ + オブジェクト名 + 文字列の長さ + 文字列	CRC 2Bytes
オブジェクト 文字列の属性を 変更の応答	FELP	0x00	37	1	0x00=成功； それ以外の場合は失敗	CRC 2Bytes



HEX コマンドの例：Command Type = 37、UI 名「standby1」のオブジェクト名「STULN1」の文字列の内容を「ABC」に変更します。ホストは UART 経由で次のデータを LCD モジュールに送信する必要があります。

ヘッダ FELP (4 bytes)	予約 0x00 (1 byte)	コマンド 37 (1 byte)	データの 長さ (1 byte)	UI 名の 長さ (1 byte)*	UI 名： standby1 (8 bytes)*	オブジェ クト名の 長さ (1 byte)*	オブジェ クト名 STULN1 (6 bytes)*	文字列 の長さ (1 byte)*	文字列： ABC (3 bytes)*	CRC (2 bytes)
0x46,0x45, 0x4C,0x50	0x00	0x25	0x14	0x08	0x73,0x74, 0x61,0x6E, 0x64,0x62, 0x79,0x31	0x06	0x53,0x54, 0x55,0x4C, 0x4E,0x31	3	0x41, 0x42, 0x43	0xB1,0x11

備考：\*印の項目はデータ長を累計します。したがって、 $1+8+1+6+1+3 = 20$  (16 進数: 0x14) がデータ長になります。CRC は次のコードに基づいて計算されます。

\*\*\*\*\*

```
const unsigned short CRC_VALUE=10;
```

```
unsigned short crc16_compute(unsigned char const * p_data, unsigned int size, unsigned short const * p_crc)
```

```
{  
    unsigned short crc = (p_crc == NULL) ? 0xFFFF : *p_crc;  
    unsigned int i;  
    for (i = 0; i < size; i++)  
    {  
        crc = (unsigned char)(crc >> 8) | (crc << 8);  
        crc ^= p_data[i];  
        crc ^= (unsigned char)(crc & 0xFF) >> 4;  
        crc ^= (crc << 8) << 4;  
        crc ^= ((crc & 0xFF) << 4) << 1;  
    }  
    return crc;  
}
```

\*\*\*\*\*

使用例：

```
unsigned short crc16_result = crc16_compute(COMM_buff, count, &CRC_VALUE);
```

// count はヘッダから最後のデータまでのバイト数です。上記の「HEX コマンドの例」の場合、ヘッダから文字列まで、合計データ長は 27 バイトであるため、count=27 になります。

## Char(文字)/ ASCII コマンド

機能	プロトコル ヘッダ (4Bytes)	予約 (1Byte)	コマンド タイプ (1Byte)	データ の長さ (1Byte)	データ (n Bytes)	CRC (2Bytes)
Char/ ASCII コマンド	FELP	0x00	127	後続のデー タの長さ	Char/ASCII コマンド、0x00 で終了。 以下のリストを参照してください。	CRC 2Bytes
Char/ ASCII コマンドの応 答	FELP	0x00	127	1	0x00=成功; それ以外の場合は失敗	CRC 2Bytes

## Char/ ASCII コマンドリスト

機能	例	説明
UI の切り替え	form main	UI を切り替えます。「main」という名の UI になります。
オブジェクトの str 値を 変更する	obj1.str=test	オブジェクト「obj1」の str 値を“test”に変更します。
オブジェクトの str の色を 変更する	obj1.fcolor= #ff0000	オブジェクト「obj1」の str の色を #ff0000 に変更します。
オブジェクトの背景の種類 を変更する	obj1.btype=pic  obj1.btype=ppic obj1.btype=bcolor obj1.btype=null	pic: オブジェクト「obj1」の背景タイプを画像に変更します。画像のサイズは オブジェクトのサイズと同じでなければなりません。 ppic: オブジェクト「obj1」の背景タイプを画像の一部に変更します。 bcolor: オブジェクト「obj1」の背景タイプを色に変更します。 null: オブジェクト「obj1」の背景タイプを透明に変更します。
オブジェクトの背景画像を 変更する	obj1.pic=cat	オブジェクト「obj1」の背景画像を画像名 cat の画像に変更します。 画像ライブラリに cat が存在する必要がある、 「obj1」の背景タイプは pic である必要があります。
オブジェクトの背景色を 変更する	obj1.bcolor= #ff0000	オブジェクト「obj1」の背景色を #ff0000(赤)に変更します。「obj1」の 背景タイプは bcolor である必要があります。
オブジェクトの表示位置を 変更する	obj1.x=160 obj1.y=82  obj1.w=20 obj1.h=43	オブジェクト「obj1」の表示位置の x 座標を 160 に変更します。 オブジェクト「obj1」の表示位置の y 座標を 82 に変更します。 注意: x、y はオブジェクトの左上隅を基準とします。X、Y 位置を適切に変更しないと、 オブジェクトのサイズが LCD 表示範囲外になる可能性があります。 オブジェクト「obj1」の幅を 20 ピクセルに変更します。 オブジェクト「obj1」の高さを 43 ピクセルに変更します。

注: 色 #ff0000 は HEX カラーコード(赤)で、RGB カラー値に基づきます。

## オブジェクトをサイズの異なるの背景画像に変更するヒント

オブジェクトの背景が画像として設定されている場合、選択した画像サイズはオブジェクトと同じサイズである必要があります。例えば、オブジェクトサイズ：20\*50 ピクセルの場合、画像サイズも 20\*50 ピクセルである必要があります。サイズの異なる背景画像に変更する場合は、まず、オブジェクトの背景タイプを null に変更し、次にオブジェクトの幅/高さを新しい画像サイズと同じに設定する必要があります。それから、画像名を指定し、オブジェクトの背景を画像に設定します。

次の例では、オブジェクト（名：NUM12）を新しい画像(名：TinyCar\_Red で、サイズは 51\*51 ピクセル)に変更します。

各ステップは、UART 経由で、その文字列を LCD モジュールに送信することを意味します。

Step 1: NUM12.btype=null

Step 2: NUM12.w=51

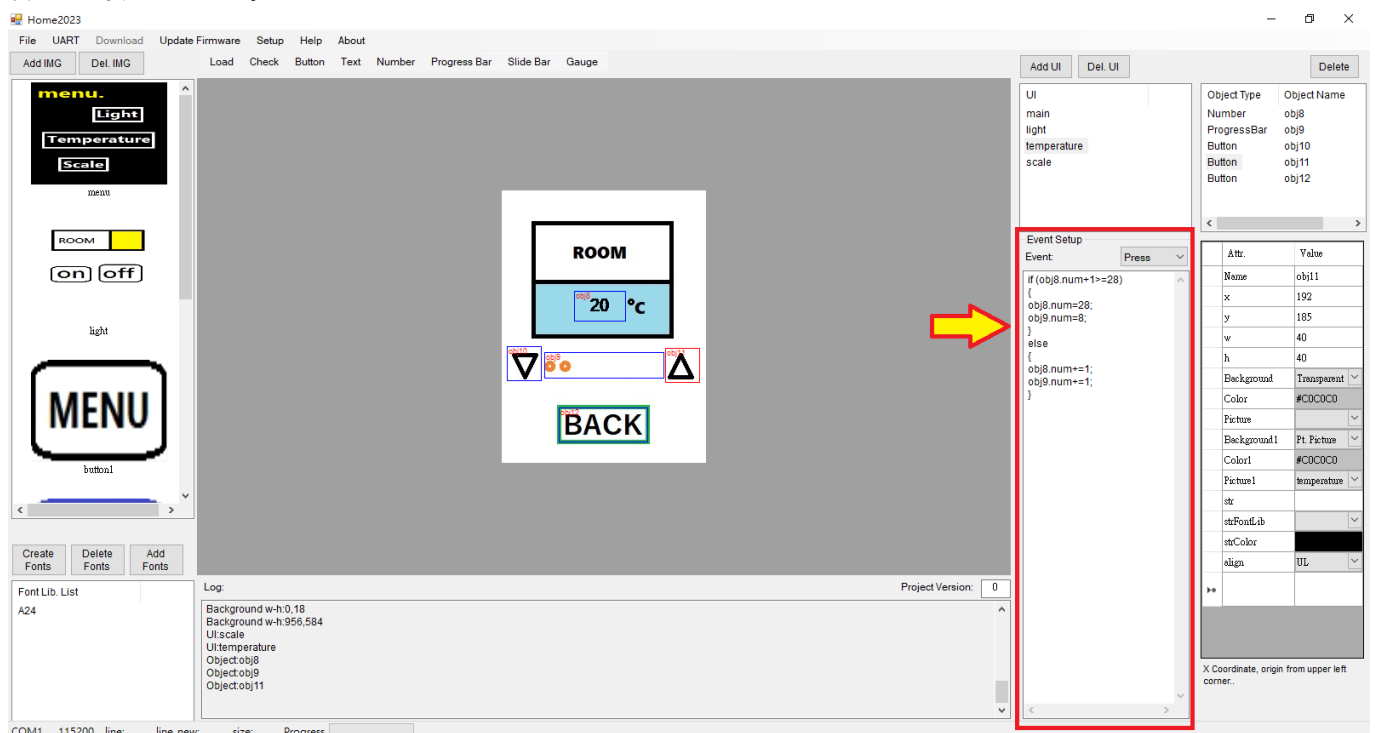
Step 3: NUM12.h=51

Step 4: NUM12.pic=TinyCar\_Red

Step 5: NUM12.btype=pic

## コード内のキーワード

ユーザは、LCD モジュールに C 言語のようなコードをインプリメントして、設定したプロセスを実行できます。次のリストは、コード内のキーワードです。コードには条件ステートメント（if else, while, and, for）を除き、各行はセミコロン「;」で終わる必要があります。



Event Setup

Event: Press

```
if (obj8.num+1>=28)
{
obj8.num=28;
obj9.num=8;
}
else
{
obj8.num+=1;
obj9.num+=1;
}
```

Att.	Value
Name	obj11
x	192
y	185
w	40
h	40
Background	Transparent
Color	#C0C0C0
Picture	
Background1	Pt Picture
Color1	#C0C0C0
Picture1	temperature
str	
strFontLib	
strColor	
align	UL

Log:

```
Background w:h:0,18
Background w:h:956,584
UI:scale
UI:temperature
Object:obj8
Object:obj9
Object:obj11
```

Project Version: 0

COM1 115200 line: line\_new: size: Progress



キーワード	変数の型	アサイン可能	戻り値の型	説明
<b>obj.str</b>	string	V	char	オブジェクト「obj」とその内容は文字列として扱われます。
例: TB2.str += "W"; TB2.str = "Hello World"; TB2.str -= 1; TB2.str = TB2.str - 2;		説明: 文字列「W」をオブジェクト TB2 に追加します。TB2.str に「H」が含まれる場合、「HW」になります。 文字列「Hello World」を TB2.str に割り当てます。 TB2.str の最後の文字を削除します。 TB2.str の最後の 2 文字を削除します。		
<b>obj.num</b>	integer/ float	V	int32 or float	オブジェクト「obj」とその内容は、整数 (1、2、-3...) または浮動小数点数 (-1.2、2.54...) として扱われます。記号「+」は使用しないでください。 ただし、ユーザーは符号「-」を設定して負の数を作成できます。
例: NU2.num -= 1; NU3.num = TB3.num + 2.00;		説明: オブジェクト NU2 に数値を設定して 1 を減算します。NU2 が 4 に設定されている場合、この操作の後、NU2 は 3 になります。 オブジェクト NU3 の番号を TB3+2.00 に設定します。TB3 が 8.00 に設定されている場合、この操作後の NU3 は 10.00 になります。 浮動小数点数の小数点以下の桁数は、オブジェクトと同じ桁数を設定する必要があります。(NU3 と TB3、数値の小数点以下の桁は同じでなければなりません)。 たとえば、すべて小数点以下 2 桁です。 NU3 は 0.00 (小数点以下 2 桁) である必要があります。 TB3 は 0.00 (小数点以下 2 桁) である必要があります。 数値 2 は 2.00 (小数点以下 2 桁) である必要があります。		
<b>obj.len</b>	integer	NA	int32	オブジェクト「obj」の現在の文字列長 (バイト単位) を取得します。
例: if (TB2.len > 5)...		説明: 条件付きルーティングで判断します。 obj.str = "ABCDEF" の場合、obj.len は 4 バイト長の数値 6 を返します。例: 0x06、0x00、0x00、0x00 数値タイプのオブジェクト、Number/ Progress Bar/ Slide Bar/ Gauge の場合、この操作の戻りデータは常に LSB 順序で送信される 4 バイトになります。0x04、0x00、0x00、0x00 例: 文字列の長さは 258 バイトで、返されるデータは 0x02 0x01 0x00 0x00 になります。このデモは LSB が意味するもので、オブジェクトの文字列の長さは 31 文字を超えることはできません。		
<b>obj.fcolor</b>	string	V	string	オブジェクト「obj」のフォントの色を (文字列として) 取得/設定します。
例: TB2.fcolor = "#FF00FF";		説明: TB2 文字列の色を「#FF00FF」に割り当てます。		
<b>obj.btype</b>	string	V	string	オブジェクト「obj」の背景タイプ (文字列として) を取得/設定します。
例: TB2.btype = "bcolor";		説明: TB2 の背景タイプを bcolor に変更します。(背景として色を使用します) オプションは、 「bcolor」色を背景として使用することができます。 「pic」拡張画像を背景として使用することができます。 「null」背景を背景 (透明) として使用することができます。 「ppic」画像の一部を背景として使用することができます。		



# UI 編集ツールマニュアル

バージョン: D

2023-12-28

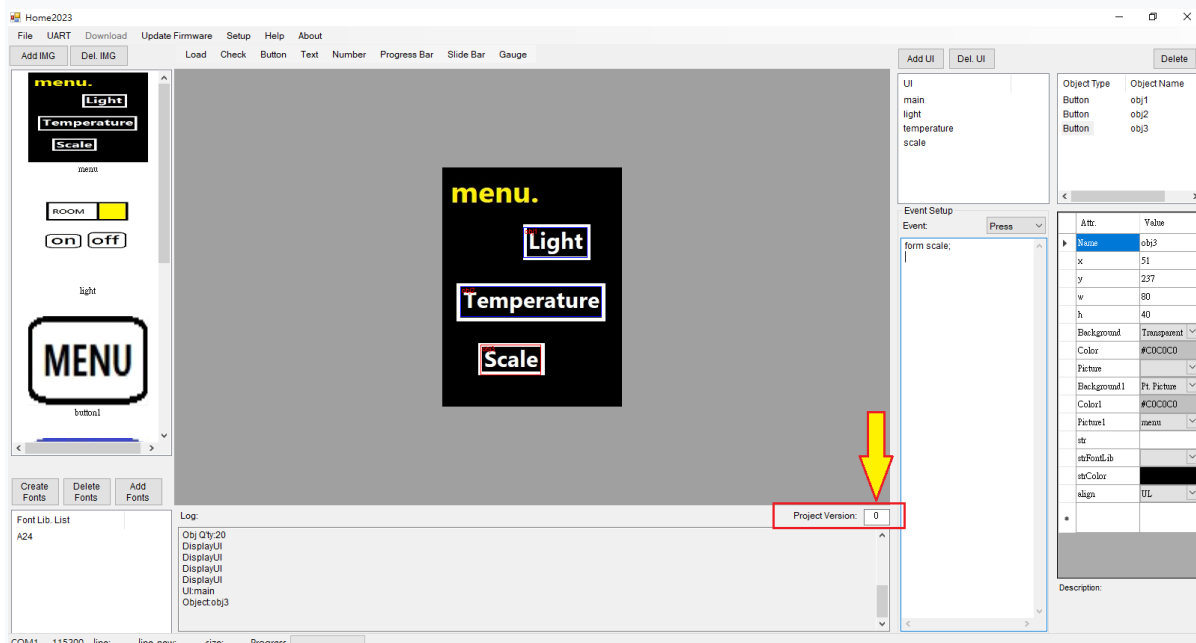
<b>obj.pic</b>	string	V	string	オブジェクト「obj」の画像（文字列として）を取得／設定します。
例： TB2.str = OBJ1.pic;		説明： OBJ1 の背景画像の名前を TB2 文字列に割り当てます。たとえば、OBJ1 の背景画像の名前が TinyCar である場合、この操作後に TB2.str は「TinyCar」になります。		
<b>obj.bcolor</b>	string	V	string	オブジェクト「obj」の背景色（文字列として）を取得／設定します。
例： TB2.str = OBJ1.bcolor;		説明： OBJ1 の背景の RGB カラー値を TB2 の文字列に割り当てます。例えば、OBJ1 の背景の RGB カラー値が #FF00FF であれば、この操作後に TB2.str は「#FF00FF」となります。		
<b>obj.x, obj.y</b>	integer	V	int32	オブジェクト「obj」の座標を取得／設定します。
例： TB2.x=160; TB2.y=30;		説明： オブジェクト TB2 の位置を x=160 ピクセル、y=30 ピクセルに割り当てます。 オブジェクトの原点はそのオブジェクトの左上隅になります。オブジェクトの位置を変更する場合、オブジェクトの範囲が LCD ディスプレイの範囲外にならないよう注意してください。		
<b>obj.w, obj.h</b>	integer	V	int32	オブジェクト「obj」の幅/高さを取得／設定します。
例： TB2.w=160; TB2.h=25;		説明： オブジェクト TB2 の幅を 160 ピクセル、高さを 25 ピクセルに割り当てます。オブジェクトの原点はそのオブジェクトの左上隅になります。オブジェクトのサイズを変更する際は、オブジェクトの範囲が LCD ディスプレイの範囲外にならないよう注意してください。オブジェクトの背景が拡張画像 (Ext. Picture) の場合、オブジェクトのサイズを変更すると、変更されたサイズが既存の拡張画像のサイズと一致しないとエラーが発生します。		
<b>send</b>	command	NA	NA	UART ポート経由でデータを送信します。
例： (文字列として送信する。) send "Text"; send BT2.str;  (数値として送信する。) send 513; send -300;  send BT2.num;		説明： str、fcolor、bcolor および pic は文字列タイプです。 UART 経由で文字列 "Text" を送信します。 UART 経由で BT2.str の文字列内容を送信します。もし、BT2.str が "STOP" を含んでいる場合、文字列 "STOP" が送信されます。  num、x、y、w、h および len は数値タイプです。 UART 経由で整数 513 (int32、LSB 形式) を送信します。513 は 0x01、0x02、0x00、0x00 として送信されます。 UART 経由で整数-300 (int32、LSB 形式) を送信します。負の値は 1 の補数として送信され、例えば-300 は 0xD4、0xFE、0xFF、0xFF として送信されます。  オブジェクト BT2.num の内容を送信します。これは整数 (int32、LSB 形式) または浮動小数点値になります。浮動小数点値は IEEE-754 浮動小数点として 16 進数で表現されます。例えば、浮動小数点数が 2.58 の場合、UART 経由で 0xB8、0x1E、0x25、0x40 として送信されます。		
<b>form</b>	command	NA	NA	UI ページを変更します。
例: form main;		説明: 現在表示されている UI を「main」という UI に変更します。		
<b>if else</b>	command	条件ステートメント		
<b>while</b>	command	条件ステートメント		
<b>for</b>	command	条件ステートメント		



## 8. SD カードでプロジェクトの更新プロセス

1. プロジェクトを LCD モジュールにダウンロードすると、UI 編集ソフトのパスに file.hex ファイルが生成されます。
2. プロジェクトを更新するには、最初にサンプルの LCD モジュールが必要です。UI 編集ソフトでプロジェクトを LCD モジュールにダウンロードして、file.hex ファイルが生成されます。（この生成方法しかありません。）
3. file.hex というファイルをコンピュータの SD カードのルートディレクトリにコピーします。完了後、SD カードをコンピュータから取り外し、LCD モジュールに挿入します。モジュールは自動的にバージョンを判断し、画面に提示が表示されます。
4. LCD モジュールに挿入された SD カードに file.hex ファイルがない場合、提示は表示されません。
5. 提示の内容は英語です。主な内容は、file.hex ファイルが検出されたというもので、既存の LCD プロジェクトのバージョン番号と SD カード内のプロジェクトのバージョン番号が表示されます。更新するにはカードの抜き差しを 30 秒以内に行ってください。
6. カードの抜き差し時に提示も表示され、アップデート時に提示も表示されます。完了後、SD カードの取り外しを求める提示が表示されます。SD カードが取り外されると、新しいプロジェクトの最初の UI が実行されます。
7. 更新したくない場合は、30 秒以内に何も操作しないで、その後 SD カードを取り出して本来のプロジェクトの最初の UI が自動的に再起動します。

\*\* プロジェクトのバージョン番号の設定位置は右下の枠で、0～9999 の範囲です。初期値は 0 です。



本マニュアルに関するご質問がありましたら、お気軽にお問い合わせください。