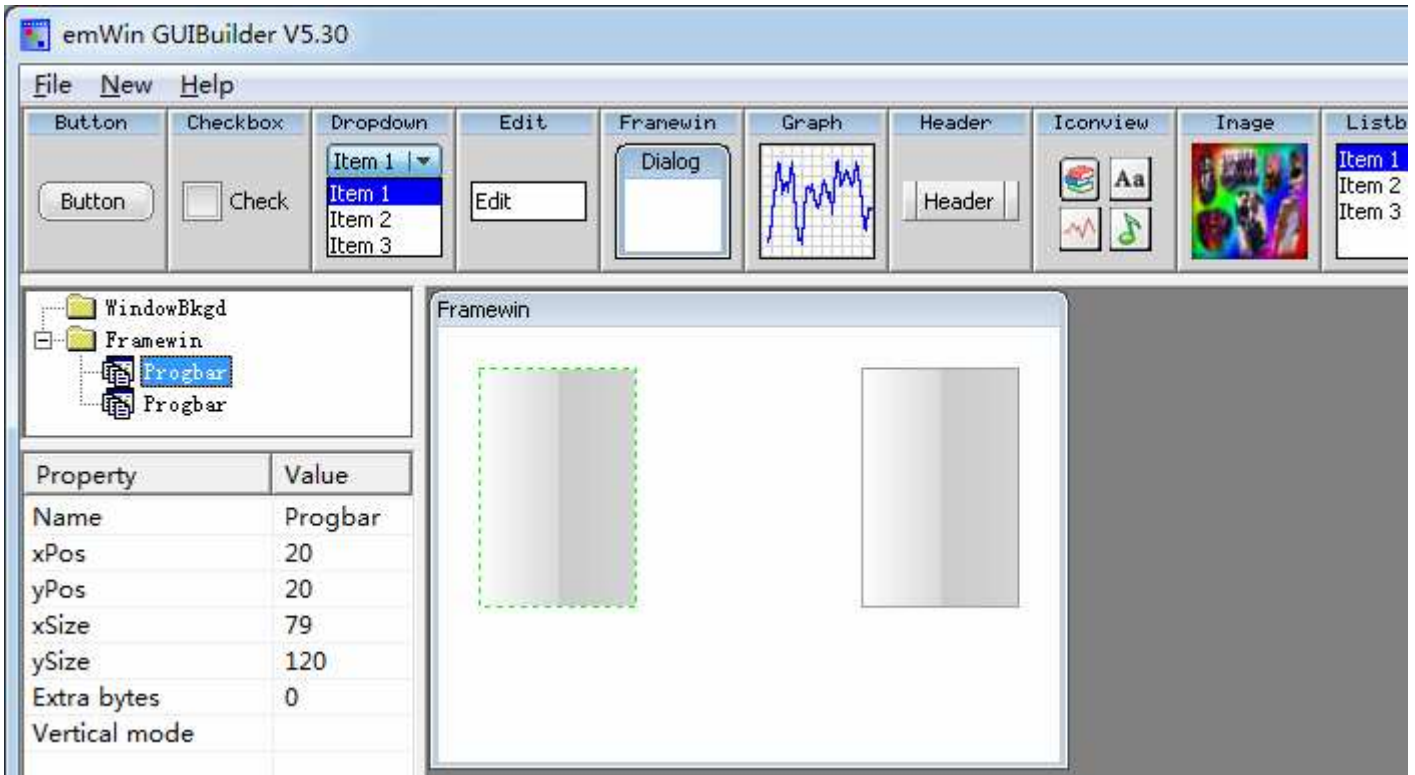


## emwin 2-day quick tutorial 012\_不规则异形进度条控件实现方法

前言:

- 1) 在 emWin 中, 有两种方法实现不规则进度条: (1)Bitmap 位图方式 (2)2D 绘图方式
- 2) 如果采用第(1)种方式, 必须先熟悉: “教程 009\_Bitmap 位图显示以及位图皮肤的使用方法”!
- 3) 掌握 emWin 的 Skinning 皮肤和剪切显示的概念非常、非常重要! 具体请看“教程 009”最后面.

使用 GUIBuilder V5.30 创建如下页面并保存为 c 文件, 然后添加到 CodeBlocks 工程:

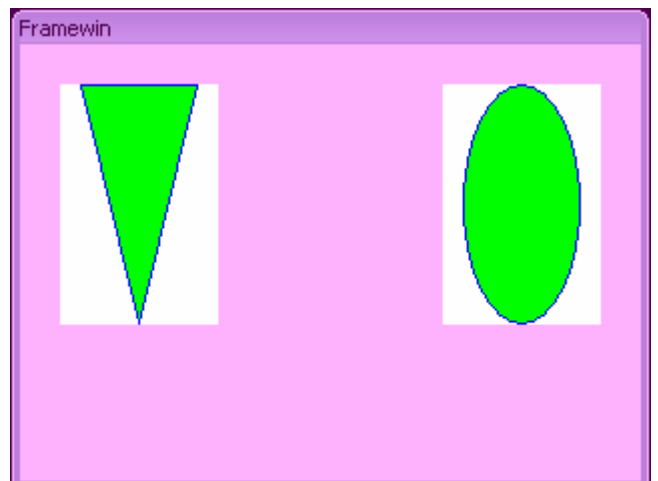
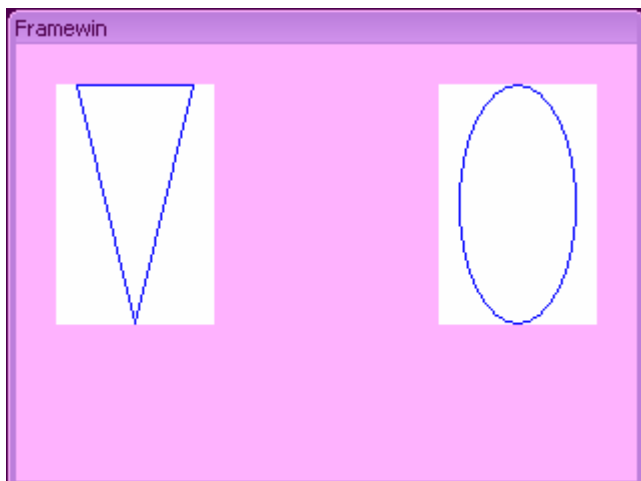


小技巧: 放置 xxxx 控件之后不要用鼠标移动, 用上下左右键移动更容易对齐(步进是 5).

备注: GUIBuilder 生成的 c 文件, 用户代码最好加在“USER START”和“USER END”之间, 其他地方除了数字以外, 不要做任何修改, 否则 GUIBuilder 将无法再次打开此 c 文件; 另外, GUIBuilder 再次打开编辑并保存时, “USER START”和“USER END”之间的内容将不会被更改; 还有 c 文件不要在 GUIBuilder 打开状态直接去修改 c 代码, 否则点 GUIBuilder 保存之后修改的内容将会丢失.

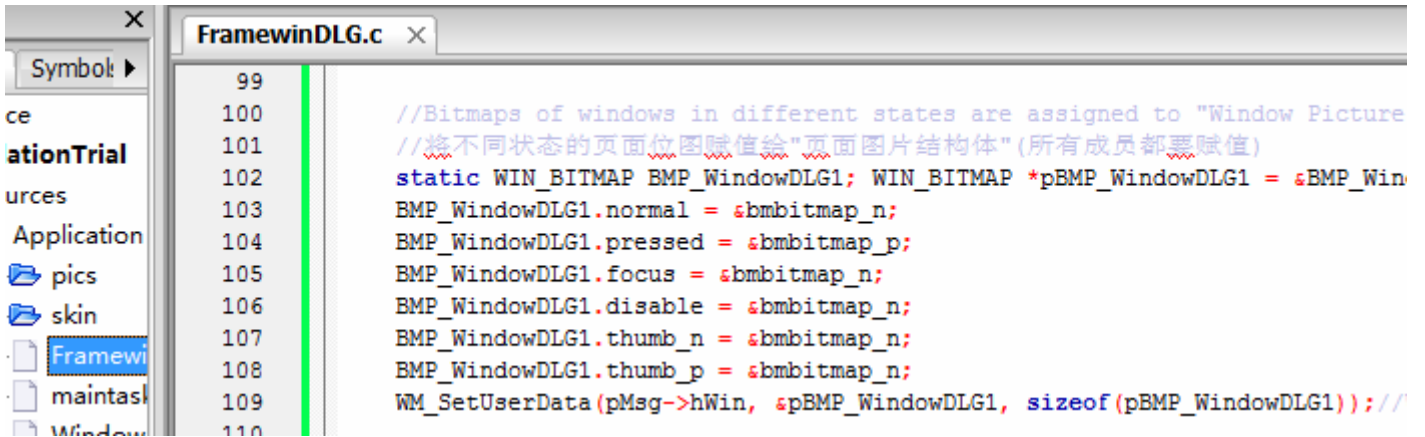
### (1)Bitmap 位图方式实现不规则进度条

准备以下两幅图片 bitmap\_n.bmp、bitmap\_p.bmp, 并用 BmpCvt.exe 转成 c 文件:



白色区域是进度条控件区域,即在这个位置放一个进度条控件,并把这两幅图片设置为该进度条控件的图片皮肤即可。第一幅图片 bitmap\_n 用于正常显示(比如进度条值为 0 时),第二幅图片 bitmap\_p 用于进度条填充,两幅图片复合显示的原理是:进度条的下部分剪切显示出 bitmap\_p,进度条的上部分剪切显示出 bitmap\_n,而这些都是 emWin 内部自己处理的(请不要想着还要自己去实现),你只需要做以下 2 步工作:

1. 在 WM\_INIT\_DIALOG 消息中(窗口初始化时程序跑到这里)将不同状态的页面图片赋值给“页面图片结构体”:



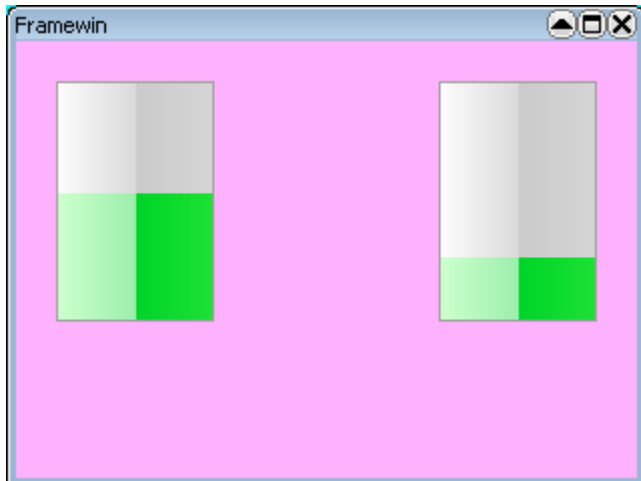
```

99
100 //Bitmaps of windows in different states are assigned to "Window Picture
101 //将不同状态的页面位图赋值给"页面图片结构体"(所有成员都要赋值)
102 static WIN_BITMAP BMP_WindowDLG1; WIN_BITMAP *pBMP_WindowDLG1 = &BMP_Win
103 BMP_WindowDLG1.normal = &bmbitmap_n;
104 BMP_WindowDLG1.pressed = &bmbitmap_p;
105 BMP_WindowDLG1.focus = &bmbitmap_n;
106 BMP_WindowDLG1.disable = &bmbitmap_n;
107 BMP_WindowDLG1.thumb_n = &bmbitmap_n;
108 BMP_WindowDLG1.thumb_p = &bmbitmap_n;
109 WM_SetUserData(pMsg->hWin, &pBMP_WindowDLG1, sizeof(pBMP_WindowDLG1));//
110

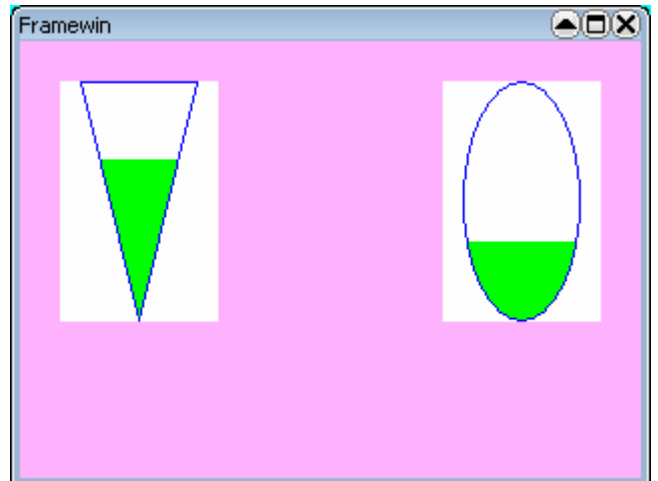
```

2. 给控件设置位图皮肤: `PROGBAR_SetSkin(WM_GetDlgItem(pMsg->hWin, ID_PROGBAR_0), SKIN_progbar3L)`; 具体请看“教程 009”;当然,进度条的形状不限于三角形或圆形,可以是任何形状、任何效果的图形。

未使用位图皮肤(原始进度条控件):



使用位图皮肤:



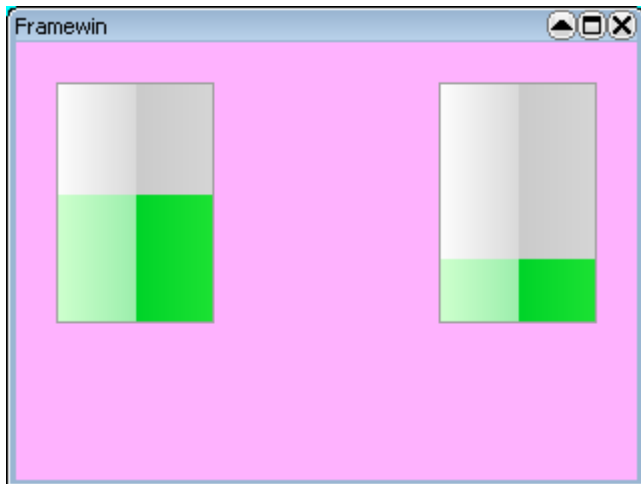
## (2) 2D 绘图方式实现不规则进度条

通过修改 Skinning 皮肤的“自定义绘制函数”，用绘制不规则图形实现各种形状的进度条：

```
kin\SKIN_progbarellipse.c x
case WIDGET_ITEM_DRAW_BACKGROUND:
    PROGBAR_SetUserClip(pDrawItemInfo); //设置1个用户剪切区
    if(((PROGBAR_SKINFLEX_INFO *)pDrawItemInfo->p)->Index == PROGBAR_SKINFLEX_L)
    {
        GUI_SetColor(GUI_WHITE);
        GUI_FillRect(pDrawItemInfo->x0, pDrawItemInfo->y0, pDrawItemInfo->x1, pDrawItemInfo->y1);
        GUI_SetColor(GUI_BLUE);
        GUI_DrawEllipse(pDrawItemInfo->x1/2, pDrawItemInfo->y0+WM_GetWindowSizeY(pDrawItemInfo->hWin)/2,
    }
    else if(((PROGBAR_SKINFLEX_INFO *)pDrawItemInfo->p)->Index == PROGBAR_SKINFLEX_R)
    {
        GUI_SetColor(GUI_WHITE);
        GUI_FillRect(pDrawItemInfo->x0, pDrawItemInfo->y0, pDrawItemInfo->x1, pDrawItemInfo->y1);
        GUI_SetColor(GUI_GREEN);
        GUI_FillEllipse(pDrawItemInfo->x1/2, pDrawItemInfo->y1/2, WM_GetWindowSizeX(pDrawItemInfo->hWin));
        GUI_SetColor(GUI_BLUE);
        GUI_DrawEllipse(pDrawItemInfo->x1/2, pDrawItemInfo->y1/2, WM_GetWindowSizeX(pDrawItemInfo->hWin));
    }
    else PROGBAR_DrawSkinFlex(pDrawItemInfo);
    WM_SetUserClipRect(0);
    break;
```

蓝色方框是绘制进度条的上部分(空白部分),绿色方框是绘制进度条的下部分(填充部分);如果你仔细阅读程序之后,会发现红色方框函数绘制的是完整的填充圆,为什么实际效果却是显示进度值的部分而已?这就是 emWin 剪切显示的神奇之处!具体请看“教程 009”最后面.

未使用位图皮肤(原始进度条控件):



使用位图皮肤:

