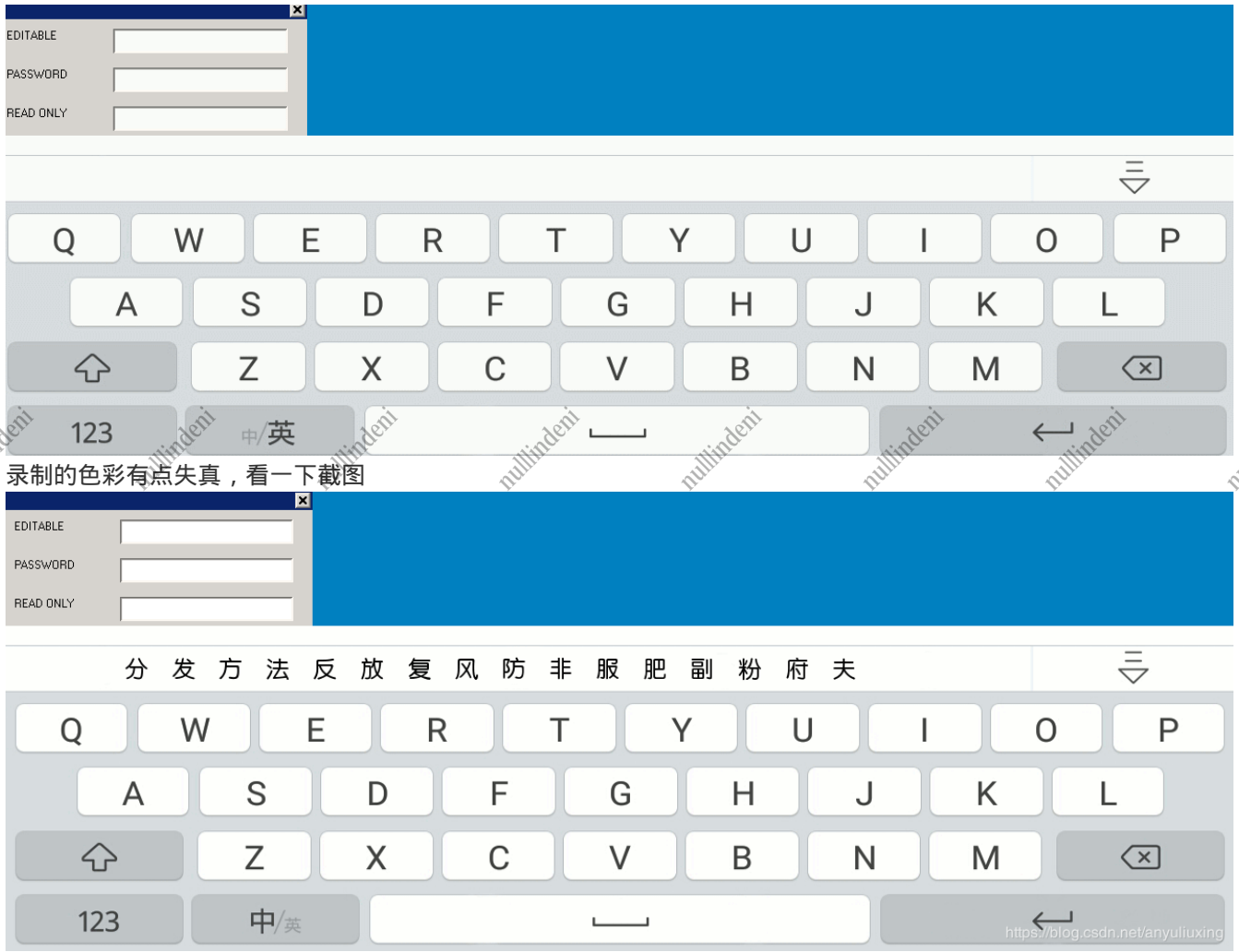
前言

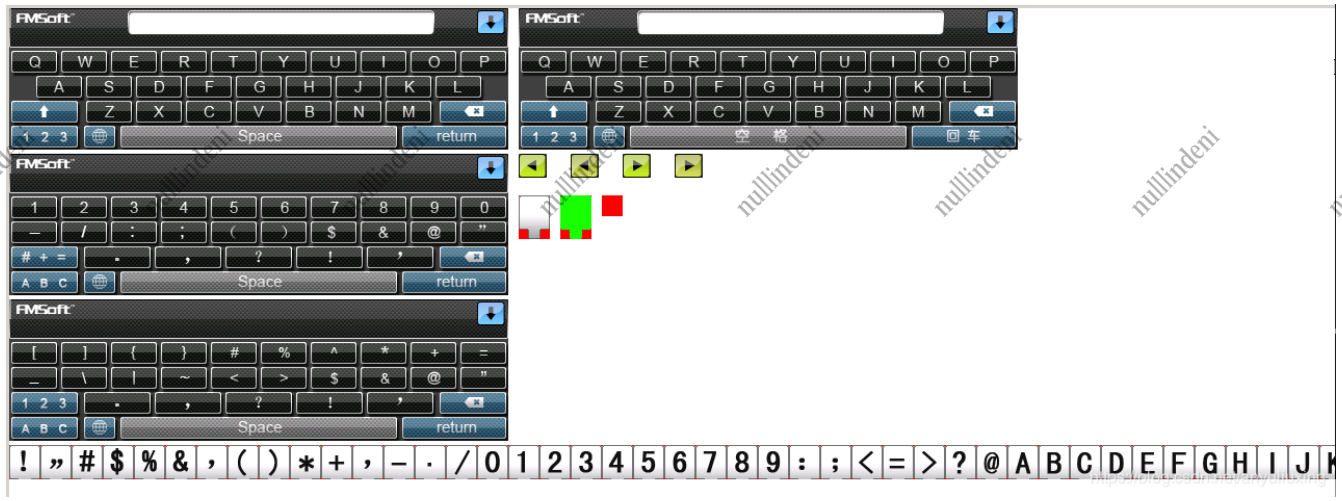
MiniGUI原来的键盘GUI已经有点不符合现在的需求，所以需要定制一套皮肤，根据我定制皮肤的过程，写一遍流程文档，做一下笔记，为以后再定制皮肤提供参考，先看一下定制好的皮肤

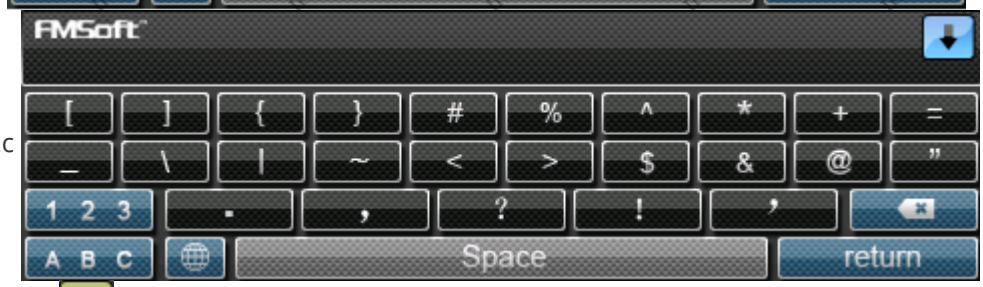


录制的色彩有点失真，看一下截图

1. 皮肤资源位置

在libmgi-2.0.4/src/softkeyboard/softkeyboard/分辨率，目录下有一些图片资源，里面是图片转为十六进制的c文件，总共需要替换的是以下的图片



1. char_key_mask.c 键盘按键按下去的效果，可以不替换
2. func_key_mask.c 功能按键按下去的效果，可以不替换
3. en_kbd_bkg.c 
4. pinyin_kbd_bkg.c 
5. num_kbd_bkg.c 
6. punct_kbd_bkg.c 
7. left_arrow_disable.c 
8. left_arrow_enable.c 
9. right_arrow_disable.c 
10. right_arrow_enable.c 
11. sel_text_bkgnd.c 

12. tooltip_bk.c 
13. tooltip_mask.c 注意红色是设置不规则窗口时候的ColorKey 
14. bitmapfont.c 这两张图片是连在一起的，太长不好截图，即A连在@后面

2. 按键对应宏

2.1 键盘宽高

要先设置好这些，不然先设置每个按键的范围也是点击不了的

宏	含义
SCREEN_W/SCREEN_H	屏幕的宽高
SKB_WIN_W/SKB_WIN_H	键盘的宽高
SKB_CLOSE_L/SKB_CLOSE_T/SKB_CLOSE_R/SKB_CLOSE_B	关闭键盘按钮的左上和右下坐标
SKB_VW_L/SKB_VW_T/SKB_VW_R/SKB_VW_B	文字候选词的左上和右下坐标，需要注意， 文字候选下一页按钮需要在该宽度和高度内
SKB_VW_PU_L/SKB_VW_PU_T/SKB_VW_PU_R/SKB_VW_PU_B	文字候选上一页按钮的左上和右下坐标
SKB_VW_PD_L/SKB_VW_PD_T/SKB_VW_PD_R/SKB_VW_PD_B	文字候选下一页按钮的左上和右下坐标
SKB_SW_L/SKB_SW_T/SKB_SW_R/SKB_SW_B	输入字符显示区域
SKB_KW_L/SKB_KW_T/SKB_KW_R/SKB_KW_B	键盘按键区域的左上和右下坐标

2.2 英文与中文按键

按钮范围宏	按钮	描述宏
RECT_EN_KEY_1	q	SCANCODE_Q
RECT_EN_KEY_2	w	SCANCODE_W
RECT_EN_KEY_3	e	SCANCODE_E
RECT_EN_KEY_4	r	SCANCODE_R
RECT_EN_KEY_5	t	SCANCODE_T
RECT_EN_KEY_6	y	SCANCODE_Y
RECT_EN_KEY_7	u	SCANCODE_U
RECT_EN_KEY_8	i	SCANCODE_I
RECT_EN_KEY_9	o	SCANCODE_O
RECT_EN_KEY_10	p	SCANCODE_P
RECT_EN_KEY_11	a	SCANCODE_A
RECT_EN_KEY_12	s	SCANCODE_S
RECT_EN_KEY_13	d	SCANCODE_D
RECT_EN_KEY_14	f	SCANCODE_F
RECT_EN_KEY_15	g	SCANCODE_G
RECT_EN_KEY_16	h	SCANCODE_H
RECT_EN_KEY_17	j	SCANCODE_J
RECT_EN_KEY_18	k	SCANCODE_K
RECT_EN_KEY_19	l	SCANCODE_L
RECT_EN_KEY_20		SCANCODE_LEFTSHIFT
RECT_EN_KEY_21	z	SCANCODE_Z
RECT_EN_KEY_22	x	SCANCODE_X
RECT_EN_KEY_23	c	SCANCODE_C
RECT_EN_KEY_24	v	SCANCODE_V
RECT_EN_KEY_25	b	SCANCODE_B
RECT_EN_KEY_26	n	SCANCODE_N
RECT_EN_KEY_27	m	SCANCODE_M
RECT_EN_KEY_28		SCANCODE_BACKSPACE

按钮范围宏	按钮	描述宏
RECT_EN_KEY_29		SCANCODE_TONUM
RECT_EN_KEY_30		SCANCODE_TOPY
RECT_EN_KEY_31		SCANCODE_SPACE
RECT_EN_KEY_32		SCANCODE_ENTER

2.3 特殊符号按键

按钮范围宏	按钮	描述宏
RECT_NUM_KEY_1	1	SCANCODE_1
RECT_NUM_KEY_2	2	SCANCODE_2
RECT_NUM_KEY_3	3	SCANCODE_3
RECT_NUM_KEY_4	4	SCANCODE_4
RECT_NUM_KEY_5	5	SCANCODE_5
RECT_NUM_KEY_6	6	SCANCODE_6
RECT_NUM_KEY_7	7	SCANCODE_7
RECT_NUM_KEY_8	8	SCANCODE_8
RECT_NUM_KEY_9	9	SCANCODE_9
RECT_NUM_KEY_10	0	SCANCODE_0
RECT_NUM_KEY_11	-	SCANCODE_MINUS
RECT_NUM_KEY_12	/	SCANCODE_SLASH
RECT_NUM_KEY_13	:	SCANCODE_SEMICOLON
RECT_NUM_KEY_14	;	SCANCODE_SEMICOLON
RECT_NUM_KEY_15	(	SCANCODE_9
RECT_NUM_KEY_16	)	SCANCODE_0
RECT_NUM_KEY_17	$	SCANCODE_4
RECT_NUM_KEY_18	&	SCANCODE_7
RECT_NUM_KEY_19	@	SCANCODE_2
RECT_NUM_KEY_20	"	SCANCODE_APOSTROPHE
RECT_NUM_KEY_21		SCANCODE_TOOP
RECT_NUM_KEY_22	.	SCANCODE_PERIOD
RECT_NUM_KEY_23	,	SCANCODE_COMMA
RECT_NUM_KEY_24	?	SCANCODE_SLASH
RECT_NUM_KEY_25	!	SCANCODE_1
RECT_NUM_KEY_26	'	SCANCODE_APOSTROPHE
RECT_NUM_KEY_27		SCANCODE_BACKSPACE
RECT_NUM_KEY_28		SCANCODE_TOEN

按钮范围宏	按钮	描述宏
RECT_NUM_KEY_29		SCANCODE_TOPY
RECT_NUM_KEY_30		SCANCODE_SPACE
RECT_NUM_KEY_31		SCANCODE_ENTER

2.4 一些限制宏

libmgi-2.0.4/src/softkeyboard/softkeyboard/softkeyboard.h

宏	默认值	说明
VW_ELMT_LEN	18	单个候选词最大长度，如safety是 6
VW_BUFFER_LEN	128	一页候选词最大长度
VW_ELMENT_NR	18	一页候选词最多能有多少个候选词
SW_STR_LEN	32	输出字符最大长度

3. 图片皮肤制作

把图片制作成depth为8bit的，节省内存

类型	值
Format	PNG
Format/Info	Portable Network Graphic
Width	800 pixels
Height	260 pixels
Bit depth	8 bits
Compression mode	Lossless
Stream size	18.8 KiB (100%)

用JAVA写一个转换工具PicBytesToC，可以把指定目录下的全部图片或者指定的一张图片转换成十六进制的C源文件表示 目前支持png，bmp，jpeg，jpg 请指定目录或者文件路径，例如 java -jar PicBytesToC.jar /home/xxx/Pictures/ java -jar PicBytesToC.jar /home/xxx/Pictures/1.png

源码如下

```
import java.io.File;
import java.io.FileInputStream;
import java.io.FileWriter;
```

```java
import java.io.IOException;

/**
 * 图片转成十六进制
 */
public class PicBytesToC {

    public static void main(String[] args) throws Exception {
        if (args.length > 0) {
            File file = new File(args[0]);
            if (file.exists()) {
                if (file.isFile()) {
                    pic2Txt(file);
                } else {
                    File[] tempList = file.listFiles();
                    if (tempList.length > 0)
                        System.out.println("该目录下文件个数：" + tempList.length);
                    for (int i = 0; i < tempList.length; i++) {
                        if (tempList[i].isFile()) {
                            System.out.println("文件：" + tempList[i]);
                        }
                        if (tempList[i].isDirectory()) {
                            System.out.println("文件夹：" + tempList[i]);
                        }
                    }
                    for (int i = 0; i < tempList.length; i++) {
                        if (tempList[i].isFile()) {
                            pic2Txt(tempList[i]);
                        }
                    }
                    if (tempList.length > 0)
                        System.out.println("全部图片转换完成\n");
                }
            }
        } else {
            System.out.println("本程序可以把指定目录下的全部图片或者指定的一张图片转换成十六进制的C源文件表示");
            System.out.println("目前支持png，bmp，jpeg，jpg");
            System.out.println("请指定目录或者文件路径，例如");
            System.out.println("java -jar PicBytesToC.jar /home/xxx/Pictures/");
            System.out.println("java -jar PicBytesToC.jar /home/xxx/Pictures/1.png");
        }
    }

    private static void pic2Txt(File file) {
        String ext = file.getName().substring(file.getName().lastIndexOf(".") + 1);
        if (ext.equals("png") || ext.equals("bmp") || ext.equals("jpeg") || ext.equals("jpg")) {
            try {
                FileInputStream fis = new FileInputStream(file);
                String fileName = file.getName().substring(0, file.getName().lastIndexOf("."));
                String fileNameData = fileName;
```

```
                if (fileNameData.contains("_bkg")) {
                    fileNameData = fileNameData.replace("_bkg", "");
                }
                fileNameData = fileNameData + "_data";
                java.io.ByteArrayOutputStream bos = new java.io.ByteArrayOutputStream();
                byte[] buff = new byte[1024];
                int len = 0;

                while ((len = fis.read(buff)) != -1) {
                    bos.write(buff, 0, len);
                }

                // 得到图片的字节数组
                byte[] result = bos.toByteArray();
                System.out.println(file.getName() + " 字节长度：" + result.length + " 正在解
析中...");

                if (null != file.getParent())
                    byte2HexStr(file.getParent(), fileName, fileNameData, result);
                else
                    byte2HexStr(".", fileName, fileNameData, result);

                fis.close();
                bos.close();

            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }

    /**
     * 实现字节数组向十六进制转换并保存
     *
     * @param b
     *            字节数组
     * @return 十六进制字符串
     */
    private static void byte2HexStr(String filePatch, String fileName, String
fileNameData, byte[] b) {
        String hs = "\t";
        String stmp = "";
        FileWriter fwriter = null;
        try {
            fwriter = new FileWriter(filePatch + "/" + fileName + ".c");
            fwriter.write("static const unsigned char " + fileNameData + "[] = {\n");
            for (int n = 0; n < b.length; n++) {
                System.out.print(n + "/" + b.length + "\r");
                stmp = (Integer.toHexString(b[n] & 0XFF));
                if (stmp.length() == 1) {
                    hs = hs + "0x0" + stmp;
                } else {
                    hs = hs + "0x" + stmp;
```

```
                }
                if (n == b.length - 1) {
                    hs += "\n";
                } else if (n != 0 && (n + 1) % 16 == 0) {
                    hs += ",\n\t";
                } else {
                    hs += ", ";
                }
                fwriter.write(hs);
                hs = "";
            }
            fwriter.write("};");
        } catch (IOException ex) {
            ex.printStackTrace();
        } finally {
            try {
                fwriter.flush();
                fwriter.close();
                System.out.println(filePatch + "/" + fileName + ".c 保存完成\n");
            } catch (IOException ex) {
                ex.printStackTrace();
            }
        }
    }
}
```

4. 显示调整

4.1 Tooltip相关

4.1.1 Tooltip图片宽高和mask

libmgi-2.0.4/src/softkeyboard/tooltip.h

宏	含义
TTW_W	宽度
TTW_H	高度
TTW_MASK_R	MASK红颜色值
TTW_MASK_G	MASK绿颜色值
TTW_MASK_B	MASK蓝颜色值

4.1.2 Tooltip显示位置

libmgi-2.0.4/src/softkeyboard/softkeyboard/common.c

```
//share_key_update函数，计算X，Y的坐标
#if defined (SOFTKBD_1280_480)
        x = mk->bound.left + RECTW(mk->bound)/2 - TTW_W/2;
        y = mk->bound.top + RECTH(mk->bound)/2 - TTW_H - TTW_H/2;
#else
        x = mk->bound.left + RECTW(mk->bound)/2 - TTW_W/2;
        y = mk->bound.top + RECTH(mk->bound)/2 - TTW_H;
#endif
```

4.1.3 Tooltip文字居中

libmgi-2.0.4/src/softkeyboard/tooltip.c

```
//init_ttw_data函数，初始化不使用图片文字
#if defined (SOFTKBD_1280_480)
    padd->pfont = CreateLogFont("ttf", "fzcircle", "UTF-8",
            FONT_WEIGHT_BOOK, FONT_SLANT_ROMAN,
            FONT_FLIP_NIL, FONT_OTHER_AUTOSCALE,
            FONT_UNDERLINE_NONE, FONT_STRUCKOUT_NONE, 48, 0);
#else
    //if (LoadBitmap (HDC_SCREEN, &bmp_f, "bitmap_font.png")) {
    if (get_kbd_bitmap (HDC_SCREEN, &bmp_f, TTW_FONT_BMP)) {
        fprintf (stderr, "Fail to load bitmap for tooltip window font. \n");
        return 1;
    }

    dev_font = CreateBMPDevFont("bmp-iphone-rrncnn-30-33-ISO8859-1", &bmp_f, "!", 94,
30);
    padd->pfont = CreateLogFont(FONT_TYPE_NAME_BITMAP_BMP, "iphone", "ISO8859-1",
                                FONT_WEIGHT_REGULAR, FONT_SLANT_ROMAN,
                                FONT_SETWIDTH_NORMAL, FONT_SPACING_CHARCELL,
                                FONT_UNDERLINE_NONE, FONT_STRUCKOUT_NONE,
                                10, 0);
#endif

//TTWinProc MSG_PAINT居中并大写显示
#if defined (SOFTKBD_1280_480)
    if (str[0] >= 'a' && str[0] <= 'z')
        str[0] += 'A' - 'a';
    SetTextColor(hdc, PIXEL_darkgray);
    DrawText(hdc, str, -1, &rc,
            DT_CENTER | DT_VCENTER | DT_SINGLELINE);
#endif
```

4.2 键盘初始化相关

libmgi-2.0.4/src/softkeyboard/softkeyboard/softkeyboard.h

```
//头文件导入
#elif defined (SOFTKBD_1280_480)
#include "1280-480/size_1280x480.h"
```

libmgi-2.0.4/src/softkeyboard/softkeyboard/resource.c

```
//图片C源文件导入
#elif defined (SOFTKBD_1280_480)
#include "1280-480/en_kbd_bkg.c"
#include "1280-480/num_kbd_bkg.c"
#include "1280-480/punct_kbd_bkg.c"
#include "1280-480/pinyin_kbd_bkg.c"
#include "1280-480/char_key_mask.c"
#include "1280-480/func_key_mask.c"
#include "1280-480/left_arrow_enable.c"
#include "1280-480/left_arrow_disable.c"
#include "1280-480/right_arrow_enable.c"
#include "1280-480/right_arrow_disable.c"
#ifdef KBD_TOOLTIP
#include "1280-480/tooltip_bk.c"
#include "1280-480/tooltip_mask.c"
#include "1280-480/bitmapfont.c"
#endif
```

libmgi-2.0.4/src/softkeyboard/softkeyboard/en_kbd.c

```
//init_en_view_window函数，英文键盘候选词最大长度
#ifdef SOFTKBD_320_240
    vw->max_str_len = 22;
#elif defined (SOFTKBD_480_272)
    vw->max_str_len = 40;
#elif defined (SOFTKBD_240_320)
    vw->max_str_len = 20;
#elif defined (SOFTKBD_800_260)
    vw->max_str_len = 38;
#elif defined (SOFTKBD_1280_480)
    vw->max_str_len = 62;
#elif defined (SOFTKBD_480_1280_CW)
    vw->max_str_len = 62;
#elif defined (SOFTKBD_480_1280_CCW)
    vw->max_str_len = 62;
#endif

//init_en_view_window函数，英文候选词使用TTF字体
#if defined (SOFTKBD_1280_480)
    vw->view_font = CreateLogFont("ttf", "fzcircle", "UTF-8",
                        FONT_WEIGHT_BOOK, FONT_SLANT_ROMAN,
                        FONT_FLIP_NIL, FONT_OTHER_AUTOSCALE,
                        FONT_UNDERLINE_NONE, FONT_STRUCKOUT_NONE, 24, 0);
#else
    vw->view_font = CreateLogFontByName ("*-fixed-rrncnn-*-24-GB2312");
#endif
```

```
//init_en_stroke_window函数，输出字符使用TTF字体
#elif defined (SOFTKBD_800_260)
    sw->stroke_font = CreateLogFont("ttf", "fzcircle", "UTF-8",
                          FONT_WEIGHT_BOOK, FONT_SLANT_ROMAN,
                          FONT_FLIP_NIL, FONT_OTHER_AUTOSCALE,
                          FONT_UNDERLINE_NONE, FONT_STRUCKOUT_NONE, 14, 0);
#else
    sw->stroke_font = CreateLogFontByName ("*-fixed-rrncnn-*-16-GB2312");
#endif
```

libmgi-2.0.4/src/softkeyboard/softkeyboard/pinyin_kbd.c

```
//init_en_view_window函数，中文键盘候选词最大长度
#ifdef SOFTKBD_320_240
    vw->max_str_len = 22;
#elif defined (SOFTKBD_480_272)
    vw->max_str_len = 40;
#elif defined (SOFTKBD_240_320)
    vw->max_str_len = 20;
#elif defined (SOFTKBD_800_260)
    vw->max_str_len = 32;
#elif defined (SOFTKBD_1280_480)
    vw->max_str_len = 50;
#elif defined (SOFTKBD_480_1280_CW)
    vw->max_str_len = 50;
#elif defined (SOFTKBD_480_1280_CCW)
    vw->max_str_len = 50;
#endif

//init_py_view_window函数，中文候选词使用TTF字体
//注意需要是GB2312编码的，目前MiniGUI键盘不支持其他编码格式的字体
#if defined (SOFTKBD_1280_480)
    vw->view_font = CreateLogFont("ttf", "fzcircle", "GB2312-0",
                        FONT_WEIGHT_BOOK, FONT_SLANT_ROMAN,
                        FONT_FLIP_NIL, FONT_OTHER_AUTOSCALE,
                        FONT_UNDERLINE_NONE, FONT_STRUCKOUT_NONE, 24, 0);
#else
    vw->view_font = CreateLogFontByName ("*-fixed-rrncnn-*-24-GB2312");
#endif

//init_py_stroke_window函数，输出字符使用TTF字体
#if defined (SOFTKBD_1280_480)
    sw->stroke_font = CreateLogFont("ttf", "fzcircle", "UTF-8",
                        FONT_WEIGHT_BOOK, FONT_SLANT_ROMAN,
                        FONT_FLIP_NIL, FONT_OTHER_AUTOSCALE,
                        FONT_UNDERLINE_NONE, FONT_STRUCKOUT_NONE, 16, 0);
#else
    sw->stroke_font = CreateLogFontByName ("*-fixed-rrncnn-*-16-GB2312");
#endif
```

4.3 文字输出相关

4.3.1 输出字符显示区域

libmgi-2.0.4/src/softkeyboard/softkeyboard/en_kbd.c

```
//sw_update函数，英文键盘显示已经点击的按键的字符在键盘上，设置文字颜色与区域
#if defined (SOFTKBD_1280_480)
        old_tecolor = SetTextColor (hdc, PIXEL_black);
#else
        old_tecolor = SetTextColor (hdc, PIXEL_lightwhite);
#endif
        old_font = SelectFont(hdc, sw->stroke_font);
#if defined (SKB_VERTICAL_CW)
        RECT rcClient;
        rcClient = sw->bound;
        rcClient.right += 4;
        DrawText(hdc, sw->str, -1, &rcClient, DT_VERTICAL_CW | DT_SINGLELINE);
#elif defined (SKB_VERTICAL_CCW)
        DrawText(hdc, sw->str, -1, &(sw->bound), DT_RIGHT | DT_VERTICAL_CCW |
DT_SINGLELINE);
#else
        DrawText(hdc, sw->str, -1, &(sw->bound), DT_LEFT);
#endif
```

libmgi-2.0.4/src/softkeyboard/softkeyboard/pinyin_kbd.c

```
//sw_update函数，中文键盘显示已经点击的按键的字符在键盘上，设置文字颜色与区域
#if defined (SOFTKBD_1280_480)
        old_tecolor = SetTextColor (hdc, PIXEL_black);
#else
        old_tecolor = SetTextColor (hdc, PIXEL_lightwhite);
#endif
        old_font = SelectFont(hdc, sw->stroke_font);
#if defined (SKB_VERTICAL_CW)
        RECT rcClient;
        rcClient = sw->bound;
        rcClient.right += 4;
        DrawText(hdc, sw->str, -1, &rcClient, DT_VERTICAL_CW | DT_SINGLELINE);
#elif defined (SKB_VERTICAL_CCW)
        DrawText(hdc, sw->str, -1, &(sw->bound), DT_RIGHT | DT_VERTICAL_CCW |
DT_SINGLELINE);
#else
        DrawText(hdc, sw->str, -1, &(sw->bound), DT_LEFT);
#endif
```

4.3.2 候选词显示区域

libmgi-2.0.4/src/softkeyboard/softkeyboard/common.c

```
//vw_set_element函数，获取每一个词的输出区域
#if defined (SKB_VERTICAL_CW)
    r.bottom = view_window->key_pg_up.bottom;
```

```
#elif defined (SKB_VERTICAL_CCW)
    r.top = view_window->key_pg_up.top;
#else
    r.left = view_window->key_pg_up.right;
#endif

//get_substr_pos_ex函数，根据文字长度，计算输出区域
#if defined (SKB_VERTICAL_CW)
    SetRect(rc, off->left, off->bottom + textsize.cx,
            off->right, off->bottom + textsize.cx + cur_size.cx);
#elif defined (SKB_VERTICAL_CCW)
    SetRect(rc, off->left, off->top - textsize.cx - cur_size.cx,
            off->right, off->top - textsize.cx );
#else
    SetRect(rc, textsize.cx + off->left, off->top,
            off->left + textsize.cx + cur_size.cx + 1, off->bottom);
#endif
```

libmgi-2.0.4/src/softkeyboard/softkeyboard/en_kbd.c

```
//vw_update，英文键盘候选词显示区域，for循环里显示所有候选词，适当调整显示位置
#if defined (SKB_VERTICAL_CW)
    rcClient.right -= 4;
    DrawText(hdc, element[i].string, -1, &rcClient,
            DT_NOCLIP | DT_VERTICAL_CW | DT_SINGLELINE);
#elif defined (SKB_VERTICAL_CCW)
    rcClient.left += 8;
    DrawText(hdc, element[i].string, -1, &rcClient,
            DT_NOCLIP | DT_VERTICAL_CCW | DT_SINGLELINE | DT_RIGHT);
#elif defined (SOFTKBD_1280_480)
    DrawText(hdc, element[i].string, -1, &element[i].bound, DT_VCENTER | DT_SINGLELINE);
#else
    DrawText(hdc, element[i].string, -1, &element[i].bound, 0);
#endif

//按下选择某个候选词的时候，输出位置上移，有一个动态效果
#if defined (SKB_VERTICAL_CW)
    rcClient =  element->bound;
    rcClient.right += 6;
    DrawText(hdc, element->string, -1, &rcClient,
            DT_NOCLIP | DT_VERTICAL_CW | DT_SINGLELINE);
#elif defined (SKB_VERTICAL_CCW)
    DrawText(hdc, element->string, -1, &element->bound,
            DT_NOCLIP | DT_VERTICAL_CCW | DT_SINGLELINE | DT_RIGHT);
#else
    DrawText(hdc, element->string, -1, &element->bound, 0);
#endif
```

libmgi-2.0.4/src/softkeyboard/softkeyboard/pinyin_kbd.c

```
//vw_update，中文键盘候选词显示区域，for循环里显示所有候选词，适当调整显示位置
#if defined (SKB_VERTICAL_CW)
```

```
    rcClient.right -= 6;
    DrawText(hdc, element[i].string, -1, &rcClient,
    DT_NOCLIP | DT_VERTICAL_CW | DT_SINGLELINE);
#elif defined (SKB_VERTICAL_CCW)
    rcClient.left += 8;
    DrawText(hdc, element[i].string, -1, &rcClient,
            DT_NOCLIP | DT_VERTICAL_CCW | DT_SINGLELINE | DT_RIGHT);
#elif defined (SOFTKBD_1280_480) || defined (SOFTKBD_800_260)
    DrawText(hdc, element[i].string, -1, &element[i].bound, DT_VCENTER | DT_SINGLELINE);
#else
    DrawText(hdc, element[i].string, -1, &element[i].bound, 0);
#endif

//按下选择某个候选词的时候，输出位置上移，有一个动态效果
#if defined (SKB_VERTICAL_CW)
    rcClient =  element->bound;
    rcClient.right += 4;
    DrawText(hdc, element->string, -1, &rcClient,
            DT_NOCLIP | DT_VERTICAL_CW | DT_SINGLELINE);
#elif defined (SKB_VERTICAL_CCW)
    DrawText(hdc, element->string, -1, &element->bound,
           DT_NOCLIP | DT_VERTICAL_CCW | DT_SINGLELINE | DT_RIGHT);
#else
    DrawText(hdc, element->string, -1, &element->bound, 0);
#endif
```

5. 注意点

因为MiniGUI中文键盘，只支持GB2312编码的字体，但是应用一般用的是UTF-8编码的字体，有冲突，只想用一个字体文件解决这个问题，修改MiniGUI.cfg即可

```
[truetypefonts]
font_number=2
name0=ttf-fzcircle-rrncnn-0-0-GB2312-0
fontfile0=/usr/local/share/minigui/res/font/fzcircle.ttf
name1=ttf-fzcircle-rrncnn-0-0-UTF-8
fontfile1=/usr/local/share/minigui/res/font/fzcircle.ttf
```