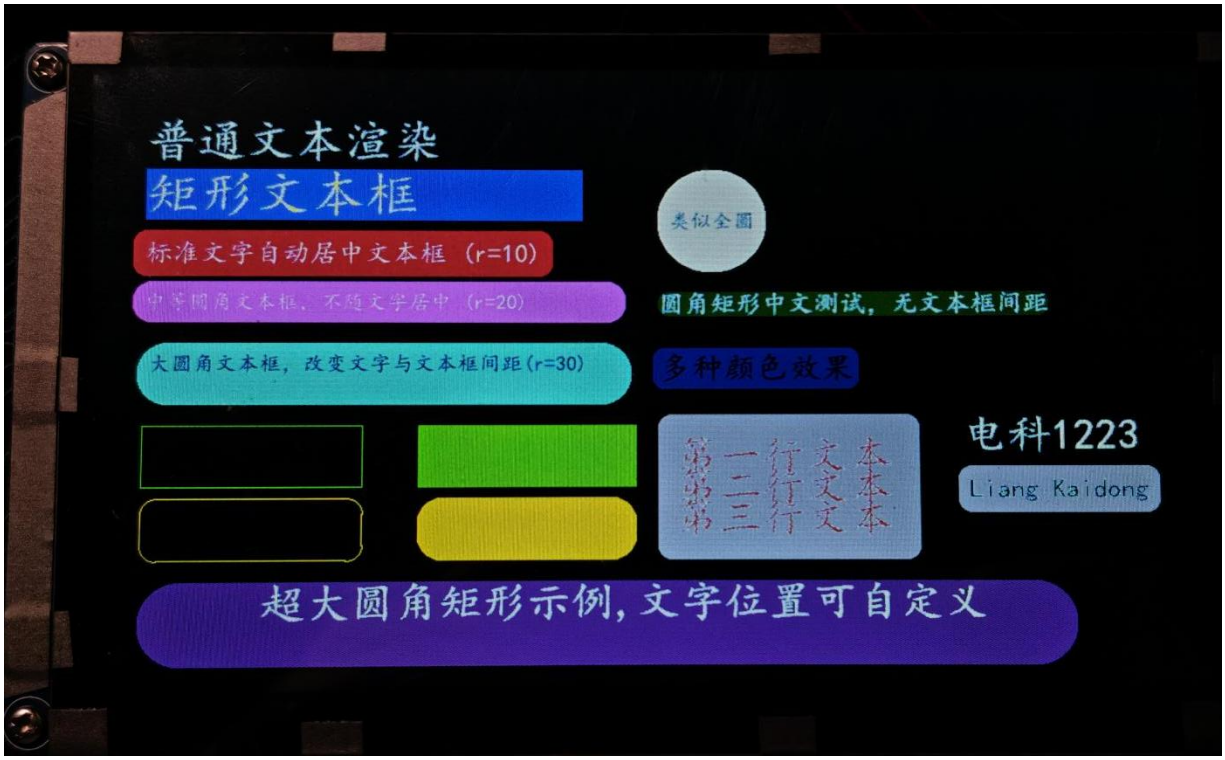
《LCD 字库 API 功能及用法详解》

在使用本字库前，请务必仔细阅读并严格遵守本使用手册。该字库基于 stb 库开发的文本框绘制功能、字体渲染功能，手册中均提供了详尽的解释与操作说明。本次测试设备型号为正点原子 I.MX6ULL，屏幕分辨率 1024×600，色彩模式 RGB565，字库在此设备上运行表现稳定。若需将字库应用于其他 ARM_Linux 设备，需手动修改 lcd_font.c 文件中 lcd 设备初始化部分的分辨率参数，随后使用 arm-linux-gnueabihf-gcc 编译链重新编译生成静态库，以便在不同项目中顺利调用。

受限于作者技术水平，本字库部分代码由 AI 辅助生成，使用过程中可能出现未知错误，甚至导致设备异常。使用本字库即视为您已充分知悉并自愿承担相关风险，若因使用本字库造成任何损失，作者均不承担责任。

本字库以轻量化设计实现高效字体渲染，支持多类型小型嵌入式设备运行环境，是开发者值得信赖的优质资源。如需基于该字库开展二次开发，建议严格遵守开源协议，保留原作者版权声明，并在合理范围内进行功能扩展。因使用或修改字库产生的任何技术问题与法律纠纷，原作者将不承担相关责任，感谢各位开发者的理解与配合。



作者：梁垲东
联系方式：226429965@qq.com
时间：2025.05.21

一、初始化与清理

1. lcd_init

- **功能**：初始化 LCD 设备和字体库。
- **原型**：`int lcd_init(const char *lcd_path, const char *font_path);`
- **参数**：
  - `lcd_path`：LCD 设备文件的路径，如 `/dev/fb0`。
  - `font_path`：字体文件的路径，如 `simkai.ttf`。
- **返回值**：成功返回 `0`，失败返回 `-1`。
- **用法示例**：

```
if (lcd_init("/dev/fb0", "simkai.ttf") != 0) {
    printf("初始化失败.\n");
    return -1;
}
```

2. lcd_cleanup

- **功能**：清理资源，释放 LCD 设备和字体缓冲区，防止内存泄漏。
- **原型**：`void lcd_cleanup(void);`
- **用法示例**：

```
// 在程序结束前调用
lcd_cleanup();
```

二、基本图形绘制

1. lcd_draw_pixel

- **功能**：在指定位置绘制一个像素点。
- **原型**：`void lcd_draw_pixel(int x, int y, color_t color);`
- **参数**：
  - `x`、`y`：像素点的坐标。
  - `color`：像素点的颜色。
- **用法示例**：

```
lcd_draw_pixel(100, 200, COLOR_RED);
```

2. lcd_draw_line

- **功能**：使用 Bresenham 算法在两点之间绘制一条直线。
- **原型**：`void lcd_draw_line(int x1, int y1, int x2, int y2, color_t color);`
- **参数**：
  - `x1`、`y1`：直线起点坐标。
  - `x2`、`y2`：直线终点坐标。
  - `color`：直线颜色。
- **用法示例**：

```
lcd_draw_line(50, 50, 200, 200, COLOR_GREEN);
```

3. lcd_draw_rectangle

- **功能**：绘制一个空心矩形。
- **原型**：`void lcd_draw_rectangle(int x, int y, int width, int height, color_t color);`
- **参数**：
  - `x`、`y`：矩形左上角坐标。
  - `width`、`height`：矩形的宽度和高度。
  - `color`：矩形边框颜色。
- **用法示例**：

```
lcd_draw_rectangle(50, 350, 200, 60, COLOR_GREEN);
```

4. lcd_draw_filled_rectangle

- **功能**：绘制一个填充矩形。
- **原型**：`void lcd_draw_filled_rectangle(int x, int y, int width, int height, color_t color);`
- **参数**：
  - `x`、`y`：矩形左上角坐标。
  - `width`、`height`：矩形的宽度和高度。
  - `color`：矩形填充颜色。
- **用法示例**：

```
lcd_draw_filled_rectangle(300, 350, 200, 60, COLOR_GREEN);
```

5. lcd_draw_rounded_rectangle

- **功能**：绘制一个空心圆角矩形。
- **原型**：`void lcd_draw_rounded_rectangle(int x, int y, int width, int height, int radius, color_t color);`
- **参数**：
  - `x`、`y`：矩形左上角坐标。
  - `width`、`height`：矩形的宽度和高度。
  - `radius`：圆角半径。
  - `color`：矩形边框颜色。
- **用法示例**：

```
lcd_draw_rounded_rectangle(50, 420, 200, 60, 15, COLOR_YELLOW);
```

6. lcd_draw_filled_rounded_rectangle

- **功能**：绘制一个填充圆角矩形。
- **原型**：`void lcd_draw_filled_rounded_rectangle(int x, int y, int width, int height, int radius, color_t color);`
- **参数**：

  - `x`、`y`：矩形左上角坐标。
  - `width`、`height`：矩形的宽度和高度。
  - `radius`：圆角半径。
  - `color`：矩形填充颜色。
- **用法示例：**

```
lcd_draw_filled_rounded_rectangle(300, 420, 200, 60, 25, COLOR_YELLOW);
```

三、文本渲染

1. lcd_set_font_size

- **功能**：设置字体大小。
- **原型**：`void lcd_set_font_size(int size);`
- **参数**：
  - `size`：字体大小。
- **用法示例**：

```
lcd_set_font_size(20);
```

2. lcd_get_text_width

- **功能**：计算指定文本的宽度。
- **原型**：`int lcd_get_text_width(const char *text);`
- **参数**：
  - `text`：要计算宽度的文本字符串。
- **返回值**：文本的宽度。
- **用法示例**：

```
int width = lcd_get_text_width("Hello World");
```

3. lcd_get_text_height

- **功能**：计算文本的高度。
- **原型**：`int lcd_get_text_height(void);`
- **返回值**：文本的高度。
- **用法示例**：

```
int height = lcd_get_text_height();
```

4. lcd_render_text

- **功能**：在指定位置渲染文本。

- **原型**：`void lcd_render_text(const char *text, int x, int y, color_t text_color, int font_size);`
- **参数**：
  - `text`：要渲染的文本字符串。
  - `x`、`y`：文本渲染起始位置的坐标。
  - `text_color`：文本的颜色。
  - `font_size`：文本的字体大小。
- **用法示例**：

```
lcd_render_text("普通文本渲染", 50, 50, COLOR_WHITE, 45);
```

5. lcd_render_text_with_box

- **功能**：在带文本框的情况下渲染文本。
- **原型**：`void lcd_render_text_with_box(const char *text, int x, int y, color_t text_color, color_t box_color, int padding, BoxStyle style, int radius, int font_size, int box_width, int box_height);`
- **参数**：
  - `text`：要渲染的文本字符串。
  - `x`、`y`：文本渲染起始位置的坐标。
  - `text_color`：文本的颜色。
  - `box_color`：文本框的填充颜色。
  - `padding`：文本与文本框边缘的间距。
  - `style`：文本框的样式，如 `BOX_STYLE_RECTANGLE` 或 `BOX_STYLE_ROUNDED`。
  - `radius`：若 `style` 为圆角矩形，该参数指定圆角的半径。
  - `font_size`：文本的字体大小。
  - `box_width`：文本框的宽度，为 `0` 时，根据文本量与字体大小调整，文字居中对齐。
  - `box_height`：文本框的高度，为 `0` 时，根据文本量与字体大小调整，文字居中对齐。
- **用法示例**：

```
lcd_render_text_with_box(
    "标准文字自动居中文本框 (r=10)",
    50, 170,
    COLOR_WHITE,
    COLOR_RED,
    10,
    BOX_STYLE_ROUNDED,
    15,
    25,
    0,
    0
);
```

四、其他辅助函数

1. decode_utf8

- **功能**：将 UTF - 8 字符解码为 Unicode 码点。
- **原型**：`static int decode_utf8(const char *str, int *codepoint);`
- **参数**：
  - `str`：指向 UTF - 8 编码字符串的指针。
  - `codepoint`：指向整数的指针，用于存储解码后的 Unicode 码点。
- **返回值**：当前字符的字节长度。
- **用法示例**：

```
int codepoint;
int len = decode_utf8("你好", &codepoint);
```

五、枚举类型

1. BoxStyle

- **定义：**

```
enum {
    BOX_STYLE_RECTANGLE,    /* 矩形样式 */
    BOX_STYLE_ROUNDED       /* 圆角矩形样式 */
};
```

- **用法：**在 `lcd_render_text_with_box` 函数中用于指定文本框的样式。

```
lcd_render_text_with_box(
    "圆角文本框",
    50, 170,
    COLOR_WHITE,
    COLOR_RED,
    10,
    BOX_STYLE_ROUNDED,
    15,
    25,
    0,
    0
);
```

六、 屏幕清空

1. lcd_clear

- **功能**：将整个 LCD 屏幕填充为指定颜色。
- **原型**：`void lcd_clear(color_t color);`
- **参数**：
  - `color`：要填充的颜色。
- **用法示例**：

```
lcd_clear(COLOR_BLACK);
```

七、其余事项

字库重新编译：arm-linux-gnueabihf-gcc -c -o lcd_font.o lcd_font.c -lm -std=gnu99
生成静态库：arm-linux-gnueabihf-ar rcs liblcd_font.a lcd_font.o
测试 demo 编译：arm-linux-gnueabihf-gcc -o font_demo font_demo.c -L. -llcd_font -lm
传输命令：tftp -g -r font_demo XXX.XXX.XXX.XXX
权限赋予：chmod 777 font_demo
程序运行: ./font_demo