patternlib 仕様書

25G1065 塩澤匠生

2025 年 7 月 15 日

1 概要

このライブラリは大文字の英語と数字，！と？の記号で構成される文字列を設定された書式でターミナルに出力するものである．文字は 8 × 8 のドット文字で表現される．表示データは 3 次元配列で array[n 文字目][行][列] というデータ形式とする．0 で空白，1 で表示を示すデータとする．格納されている文字数は大文字の英語と数字，！と？の記号の数で 64 個，行と列の個数は 8 になっている．それぞれ FONT_DATA_SIZE，ROW_SIZE，COL_SIZE として#define で別名定義されている．この定数は変更しないことを前提とする．

2 構造体の説明

2.1 Config 構造体

```
typedef struct Config {...}
```

■機能　表示に関する設定やフォントデータをまとめて管理するための構造体。‘init_config‘関数で初期化して使用する。

■メンバ

表 1: Config 構造体のメンバ

型	名前	役割
int	scale	描画の拡大率。1 以上の整数を指定する。
int	horizontal_flag	描画方向の指定フラグ。1 で横方向、0 で縦方向。
int	color_flag	カラー表示の有効化フラグ。1 で有効。
int	color	ANSI カラーコード (0-7) を指定。
int[FONT_DATA_SIZE][ROW_SIZE][COL_SIZE]	array	フォントデータを格納する 3 次元配列。
char	disp_char	パターンの描画に用いる文字。

3 関数の説明

3.1 関数一覧

表 2: 関数一覧

関数名	説明
init_config	Config 構造体を初期化する。
show_str	文字列をアスキーアートで表示する。
conv_ASCII	(内部関数) 文字をフォント配列のインデックスに変換する。
setFontArray	(内部関数) Config 構造体にフォントデータを読み込む。
setFontArray_manual	(内部関数) Config 構造体に手動でフォントデータを設定する。
show_str_vertical_scaled	(内部関数) 文字列を縦方向に拡大表示する。
show_str_horizontal_scaled	(内部関数) 文字列を横方向に拡大表示する。
print_char	(内部関数) 1 ピクセル分の文字をターミナルに出力する。
print_char_colored	(内部関数) 1 ピクセル分の文字を色付きでターミナルに出力する。

3.2 init_config 関数

```
void init_config(Config *config);
```

■機能 ‘Config‘構造体のポインタを受け取り、そのメンバをデフォルト値で初期化する。

■引数

表 3: init_config 関数の引数

型	名前	役割
Config*	config	初期化対象の Config 構造体へのポインタ。

■メンバー初期値 ‘init_config‘関数は、‘Config‘構造体の各メンバーを以下の通りに初期化します。

表 4: init_config による初期値

メンバー名	初期値	説明
scale	1	拡大率は 1 倍に設定されます。
horizontal_flag	0	表示方向は縦方向に設定されます。
color_flag	0	カラー表示は無効に設定されます。
color	1	デフォルトの描画色が設定されます。
disp_char	’ ’	描画に用いる文字は空白文字に設定されます。
array	—	setFontArray() 関数が呼び出され、フォントデータが読み込まれます。

3.3 conv_ASCII 関数

```
int conv_ASCII(char _char);
```

■機能 この関数は、文字（’A’-’Z’, ’a’-’z’, ’0’-’9’, ’!’, ’?’）をフォントデータ配列に対応するインデックス（0-63）に変換する内部関数です。

■引数

表 5: conv_ASCII 関数の引数

型	名前	役割
char	_char	変換対象の文字。

■内部処理 この関数は、入力された文字をフォントデータ配列のインデックスに変換します。対応するインデックスが存在しない場合はエラー値を返します。

3.4 setFontArray 関数

```
void setFontArray(int array[FONT_DATA_SIZE][ROW_SIZE][COL_SIZE]);
```

■機能 ライブラリ内部にハードコードされたフォントデータを、引数で渡された ‘Config‘構造体の ‘array‘メンバにコピーする内部関数。

3.5 setFontArray_manual 関数

```
void setFontArray_manual(int setData[FONT_DATA_SIZE][ROW_SIZE][COL_SIZE], Config
config);
```

■機能 この関数は、手動でフォントデータを設定するための内部関数です。‘setData‘ 配列の内容を ‘config.array‘ 配列にコピーします。

■引数

表 6: setFontArray_manual 関数の引数

型	名前	役割
int[FONT_DATA_SIZE][ROW_SIZE][COL_SIZE]	setData	コピー元の 3 次元配列。
Config	config	コピー先の構造体。

■内部処理　この関数は、3 重の入れ子ループを使用して、‘setData‘ 配列のすべての要素を ‘config.array‘ 配列にコピーします。各ループは、‘FONT_DATA_SIZE‘、‘ROW_SIZE‘、‘COL_SIZE‘ の範囲で反復します。

3.6　show_str 関数

```
void show_str(const char *str, Config config);
```

■機能　本関数は、‘Config‘構造体の設定に従って、与えられた文字列 ‘str‘ をアスキーアートとしてターミナル上に描画する。この関数がライブラリのメインの描画関数となる。

■引数

表 7: show_str 関数の引数

型	名前	役割
const char*	str	描画する文字列。
Config	config	表示設定が格納された Config 構造体。

■内部処理　‘config‘の ‘horizontal_flag‘が 1（真）であれば ‘show_str_horizontal_scaled‘関数を、そうでなければ ‘show_str_vertical_scaled‘関数を呼び出す。

3.7　show_str_vertical_scaled 関数

```
void show_str_vertical_scaled(const char *str, Config config);
```

■機能　文字列を構成する各文字を縦に連結し、指定された倍率で拡大して表示する。‘show_str‘から ‘horizontal_flag‘が 0 の場合に呼び出される内部関数。

■引数

表 8: show_str_vertical_scaled 関数の引数

型	名前	役割
const char*	str	描画する文字列。
Config	config	表示設定が格納された Config 構造体。

■内部処理　入れ子になったループを用いて、まず文字列の各文字ごと (‘*p‘)、次に行ごと (‘y‘)、そして列ごと (‘x‘) に処理を行うことで、各文字パターンを縦に並べて描画します。拡大率 (‘config.scale‘) に応じて、各ピクセルを水平・垂直方向に繰り返し描画します。ピクセルを描画する際には、‘config.color_flag‘の値を確認します。フラグが有効な場合は ‘print_char_colored‘を、無効な場合は ‘print_char‘を呼び出します。これらの関数には、描画に用いる文字として ‘config.disp_char‘が、色付き描画の場合は色として ‘config.color‘が渡さ

れます。各文字の描画が完了するたびに改行します。文字列の各文字は ‘conv_ASCII‘関数を使用してフォントデータ配列のインデックスに変換されます。

3.8 show_str_horizontal_scaled 関数

```
void show_str_horizontal_scaled(const char *str, Config config);
```

■機能 文字列を構成する各文字を横に連結し、指定された倍率で拡大して表示する。‘show_str‘から‘horizontal_flag‘が 1 の場合に呼び出される内部関数。

■引数

表 9: show_str_horizontal_scaled 関数の引数

型	名前	役割
const char*	str	描画する文字列。
Config	config	表示設定が格納された Config 構造体。

■内部処理 入れ子になったループを用いて、まず行ごと (‘y‘)、次に文字列の各文字ごと (‘*p‘)、そして列ごと (‘x‘) に処理を行うことで、各文字パターンを横に並べて描画します。拡大率 (‘config.scale‘) に応じて、各ピクセルを水平・垂直方向に繰り返し描画します。ピクセルを描画する際には、‘config.color_flag‘の値を確認します。フラグが有効な場合は ‘print_char_colored‘を、無効な場合は ‘print_char‘を呼び出します。これらの関数には、描画に用いる文字として ‘config.disp_char‘が、色付き描画の場合は色として ‘config.color‘が渡されます。すべての文字の 1 行分が描画されると改行します。文字列の各文字は ‘conv_ASCII‘関数を使用してフォントデータ配列のインデックスに変換されます。

3.9 print_char 関数

```
void print_char(int flag, char _char);
```

■機能 1 ピクセルに相当する文字をターミナルに 1 文字表示する．ピクセルデータ (‘flag‘) に基づき，指定された文字 (‘_char‘) または空白を出力する．‘show_str‘系の関数から呼び出される低レベル描画関数.

■引数

表 10: print_char 関数の引数

型	名前	役割
int	flag	ピクセルデータ．0 の場合は空白，それ以外は文字を描画.
char	_char	描画に用いる文字.

■内部処理 ‘flag‘が 0 の場合は空白を出力する．それ以外の場合，‘_char‘がスペース文字であれば背景を白にして出力し，それ以外の文字であればそのまま出力する.

3.10 print_char_colored 関数

```
void print_char_colored(int flag, char _char, int color);
```

■機能 1 ピクセルに相当する文字をターミナルに 1 文字，色付きで表示する．ピクセルデータ (‘flag‘) に基づき，指定された文字 (‘_char‘) または空白を，指定色 (‘color‘) で出力する．

■引数

表 11: print_char_colored 関数の引数

型	名前	役割
int	flag	ピクセルデータ．0 の場合は空白，それ以外は文字を描画.
char	_char	描画に用いる文字.
int	color	ANSI エスケープシーケンスのカラーコード (0-7).

■内部処理 ‘flag‘が 0 の場合はリセットコード付きの空白を出力する．それ以外の場合，‘_char‘がスペース文字であれば指定された色で背景色を設定して出力し，それ以外の文字であれば指定された色で文字色を設定して出力する．色の設定には ANSI エスケープシーケンスを用いる．