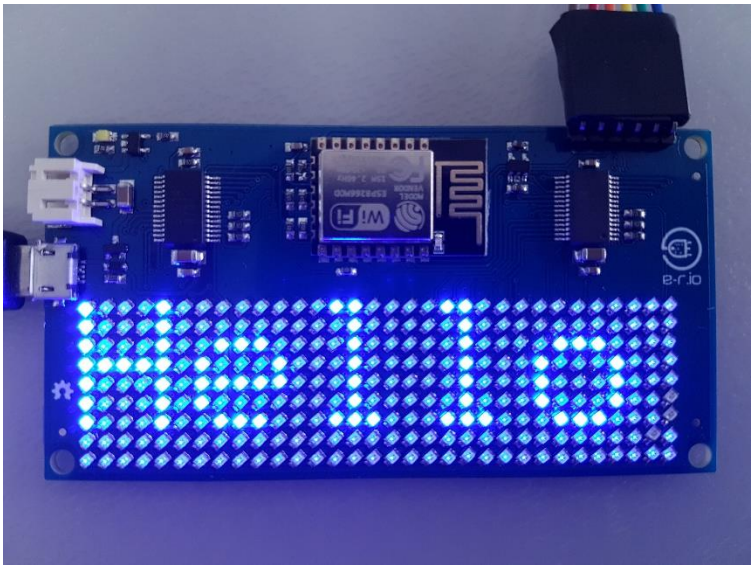
e-radionica Social Display Library



Initializing Library

Must call this function before calling any other function from class.
It initializes IS31FL3731 Led Driver and sets up IS31L3731 library to work with this board.

```
int Social_Display::begin(void);
```

Parameters: None

Returns:
1 For successful initialization
0 For unsuccessful initialization

NOTE: If is initialization unsuccessful, user defined program should not continue!

Setting up font and background brightness

This function enables user to set desired brightness level for fonts and for background.

```
void Social_Display::brightness(uint8_t _fontLight, uint8_t _backingLight);
```

Parameters:
`uint8_t _backingLight` - Sets up a brightness of background. It goes from 0 (dark) to 255 (bright). If user don't want background to light up, set this to zero.

Returns: None

Displaying a message on display

With this function, user can send a message to display. It can also set does want to scroll that message, or not and also how fast to do scrolling.

```
void Social_Display::message(char* msg, int _ms, int _stp, int _rep);
```

Parameters:
`char* msg` - String (message) that will be displayed on display.
`int _ms` - Integer that set how long is delay in milliseconds between two scroll steps (shifts). It must be larger than 10 ms.
`int _stp` - Integer that set how many pixels will be skipped in just one step in scrolling. Must not be negative or zero.

`int _rep` - Integer that determine how many times message have repeated on display. -1 means that message will scroll all the time, 0 means that message will be stationary (no scrolling).

Returns: None.

Stopping scrolling message on display

Calling this function, message that is displayed on panel, will be stopped from scrolling. You can continue scrolling by calling Social_Display::resumeScroll(void) function.

```
void Social_Display::stopScroll(void);
```

Parameters: None.

Returns: None.

Resuming scrolling

Calling this function, message that is previously stopped from scrolling by calling `Social_Display::stopScroll(void)` function, will continue to scroll.

```
void Social_Display::resumeScroll(void);
```

Parameters: None.

Returns: None.

Deleting message or picture form screen

Calling this function, everything on display will be deleted. Also, it will delete everything from buffers inside the library.

```
void Social_Display::deleteScroll(void);
```

Parameters: None.

Returns: None.

Drawing 8x8 pixels bitmap image on screen

This function enables user to display bitmap picture on specific location on screen. This picture is not scrolling.

```
void Social_Display::picture(uint8_t* p, int posX, int posY);
```

Parameters:
`uint8_t* p` - Address to bitmap picture.
`int posX` - Sets x coordinate of picture.
`int posY` - Sets y coordinate of picture.

Returns: None.
NOTE: Bitmap has to be 8x8 pixels!

Scrolling bitmap picture on screen

With this function, user can send a bitmap picture that will scroll on display. It will delete everything that was previously been on display.

```
void Social_Display::scrollPicture(uint8_t* p, int _ms, int _stp);
```

Parameters:
`uint8_t* p` - Address to bitmap picture.
`int _ms` - Integer that set how long is delay in milliseconds between two scroll steps (shifts). It must be larger than 10 ms.
`int _stp` - Integer that set how many pixels will be skipped in just one step in scrolling. Must not be negative or zero.

Returns: None
NOTE: Bitmap has to be 8x8 pixels!

Scrolling message along with picture

With this function, user can display text and bitmap pictures at the same time. It will clear all that was previously been on display.

```
void Social_Display::scrollTxtAndPics(char* txt, uint8_t** p, uint16_t* picsPos_x,
uint16_t* picsPos_y, uint8_t n, int _ms, int _stp, int _rep);
```

Parameters:
`char* txt` - String that user wants to display on screen.
NOTE: There must be two blank spaces on position where picture needs to be!

`uint8_t** p` - Address of field of pointers that consists of bitmap picture addresses. It has to be in exact order that will be displayed on display.
`uint16_t* picsPos_x` - Pointer to field of x coordinates for each picture. In same order as on `uint8_t** p`.
`uint16_t* picsPos_y` - Pointer to field of y coordinates for each picture. In same order as on `uint8_t** p`.
`uint8_t n` - Number of pictures in message.
`int _ms` - Integer that set how long is delay in milliseconds between two scroll steps (shifts). It must be larger than 10 ms.
`int _stp` - Integer that set how many pixels will be skipped in just one step in scrolling. Must not be negative or zero.
`int _rep` - Integer that determine how many times message have repeated on display.
-1 means that message will scroll all the time, 0 means that message will be stationary (no scrolling).

Returns: None.
NOTE: Bitmap has to be 8x8 pixels!

Counting how may times message has been repeated on screen

This function keeps track on how many times has message been repeated on screen. Useful if user wants to wait until the message is whole displayed.

```
int Social_Display::repeatCount(void);
```

Parameters: None.

Returns: Number of times that message has been repeated on screen.

Displaying a greyscale picture on screen

Using this function, user can display 8-bit greyscale image on screen. Picture should be stored in one dimensional array with element size of 8 bits wide.

```
void Social_Display:: picture8Bit(uint8_t* p, int xSize, int ySize, int x0, int y0,
uint8_t bright);
```

Parameters:

`uint8_t* p` - Pointer to array (picture). Element in this array must me 8 bits.
`int xSize` - Size of picture (pixels) in x direction.
`int ySize` - Size of picture (pixels) in y direction.

`int x0` - X coordinate of picture (point 0,0 of picture is upper left).
`int y0` - Y coordinate of picture (point 0,0 of picture is upper left).
`uint8_t bright` - Sets the brightness of picture (it uses Gamma correction for brightness). Gamma factor is 2.2 and can be changed by opening header file of this library.

Returns: None

Connecting to WLAN Network

Use this function to connect to WLAN network for accessing the Internet.

```
int Social_Display::wifiNetwork(const char* _ssid, const char* _pass);
```

Parameters:
`const char* _ssid` - String that contains WLAN network name.
`const char* _pass` - String that contains password of WLAN network.

Returns:
1 For successful connection to WLAN network.
0 For unsuccessful connection to WLAN network.

Display web page content on LED Matrix

Use this function for opening specific web page and displaying one line of text on display.

```
int Social_Display::webPage(char* web, char* url, int port, int _ms, int _stp, int
_rep);
```

Parameters:
`char* web` - String that contains name of web client (example: www.google.com).
`char* url` - String that contains URL of the user defined Web page that has text that has to be displayed on screen.
`int port` - Port of the web client for specific application. (In case of web page usually is 80, 88, 8080).

Returns:
1 For successful connecting to client and opening web page.
0 For unsuccessful connecting to client or opening web page.

Getting web page content

Use this function to catch one line of text from web page. It will not display it on screen.

```
int Social_Display::webPageText(char* web, char* url, int port, char* txt, int _n);
```

Parameters:

`char* web` - String that contains name of web client (example: www.google.com).

`char* url` - String that contains URL of the user defined Web page.

int port - Port of the web client for specific application. (In case of web page usually is 80, 88, 8080).

`char* txt` - Pointer to string in which text from web page will be saved.

`int _n` - Number of letters (elements of string) that user wants to copy into string.

Returns:

1 For successful connecting to client and opening web page.

0 For unsuccessful connecting to client or opening web page.