LVGL UI Generator

This forked project has been archived in favor of the newer and much simpler generator based on lv_micropython.

Simulator project for LVGL embedded GUI Library

The LVGL is written mainly for microcontrollers and embedded systems however you can run the library **on your PC** as well without any embedded hardware. The code written on PC can be simply copied when your are using an embedded system.

Using a PC simulator instead of an embedded hardware has several advantages:

- **Costs $0** because you don't have to buy or design PCB
- **Fast** because you don't have to design and manufacture PCB
- **Collaborative** because any number of developers can work in the same environment
- **Developer friendly** because much easier and faster to debug on PC

Requirements

This project is configured for VSCode and only tested on Linux, although this may work on OSx or WSL. It requires a working version of GCC, GDB and make in your path.

To allow debugging inside VSCode you will also require a GDB extension or other suitable debugger. All the requirements have been pre-configured in the .workspace file (simply open the project by doubleclick on this file).

The project can use **SDL** or **X11** as LVGL display driver for lowlevel graphics/mouse/keyboard support. This can be defined in the Makefile.
Please make sure the selected library is installed in the system (check Install graphics driver).

Usage

Get the PC project

Clone the PC project and the related sub modules:

```
git clone --recursive https://github.com/lvgl/lv_port_pc_vscode
```

Install graphics driver

The project can use **SDL** or **X11** as LVGL display driver. This can be selected in the Makefile.
Please make sure the used library is installed in the system:

Install SDL

You can download SDL from https://www.libsdl.org/

On on Linux you can install it via terminal:

```
sudo apt-get update && sudo apt-get install -y build-essential libsdl2-dev
```

Install X11

On on Linux you can install it via terminal:

```
sudo apt-get update && sudo apt-get install -y libx11-dev
```

Optional library

There are also FreeType and FFmpeg support. You can install FreeType support with:

```
# FreeType support
wget https://kumisystems.dl.sourceforge.net/project/freetype/freetype2/2.13.2/freetype-2.13.2.tar.xz
tar -xf freetype-2.13.2.tar.xz
cd freetype-2.13.2
make
make install
```

The FFmpeg support can be installed with:

```
# FFmpeg support
git clone https://git.ffmpeg.org/ffmpeg.git ffmpeg
cd ffmpeg
git checkout release/6.0
./configure --disable-all --disable-autodetect --disable-podpages --disable-asm --enable-avcodec --enable-avformat --enable-decoders --enable-encoders --enable-demuxers --enable-parsers --enable-protocol='file' --enable-swscale --enable-zlib
make
sudo make install
```

And then remove all the comments in the `Makefile` on `INC` and `LDLIBS` lines. \
They should be for **SDL**:

```
INC    := -I./ui/simulator/inc/ -I./ -I./lvgl/ -I/usr/include/freetype2 -L/usr/local/lib
LDLIBS := -lSDL2 -lm -lfreetype -lavformat -lavcodec -lavutil -lswscale -lm -lz -lpthread
```

They should be for **X11**:

```
INC    := -I./ui/simulator/inc/ -I./ -I./lvgl/ -I/usr/include/freetype2 -L/usr/local/lib
LDLIBS := -lX11 -lm -lfreetype -lavformat -lavcodec -lavutil -lswscale -lm -lz -lpthread
```

Setup

To allow custom UI code an `lv_conf.h` file placed at `ui/simulator/inc` will automatically override this projects lv_conf.h file. By default code under `ui` is ignored so you can reuse this repository for multiple projects. You will need to place a call from `main.c` to your UI's entry function.

To build and debug, press F5. You should now have your UI displayed in a new window and can access all the debug features of VSCode through GDB.

To allow temporary modification between simulator and device code, a SIMULATOR=1 define is added globally.

License

This project is licensed under the MIT License - see the LICENSE file for details.