In video #56 I introduced the Nextion smart displays and showed how we can use them based on an example file. In this video, I will go into the details on how to build your own screens and how to write an Arduino sketch to use it in your projects. Please watch the first video #56 if you did not do it yet.

I start with a simple application: We want to press a plus or minus key and then a number is changed accordingly.

You remember, we first have to program the Nextion display using an editor running on windows and upload the resulting file to the display. Next, we can then build out Arduino sketch and upload it to the board.

Let’s start with the application for the Nextion display: First we have to start the Nextion editor and create a new file. We call it ButtonTest. We are asked for the display we want to use. I have the 3.5 inch version and want to use it in horizontal orientation.

Next we have to create the fonts we want to use. We start the font generator and select the font we like. I chose Consolas and the size 32. The rest of the fields are ok. When we generate the font we have to give it a name and a place to store. I store it at the same place as the hmi and the ino file in the Ardiono sketch directory. Like that, I have all project files in one place.

Then, we can import the font into our resources. From now on, we can use it in our project.

Next I want a nice background. I can either use an available picture or create one. Important is, that it has the same dimensions that our display, 480x320 in the case of the 3.5 inch display. As you know, I am sometimes in Dubai. Therefore, I use a picture from there: To work with it, I use Inkscape. It is a free software to create these kind of pictures. I am sure, there are other programs available for this purpose.

Here, I define the needed format of 480x320 px. I add the dubai skyline and the logo. Then, I save it as a png file and import it into the Nextion editor.

The editor already created a page with the name page0. Now we have to insert the picture with a double click. Next, we move it to its place. So, the background is done.

Then, we add a text field and place it, where it belongs. Next, we need the two buttons. One button is the plus and the other is the minus button. The size of the letters is 29 and the height is 47. I found these sizes by try and error.

Because we want to interact with the button, we have to enable the sendComponent ID. I enable it for both events, the pressing of the button and the releasing of the button. This way, I can choose later in the Arduino sketch, which event will be used.

As a last step. We have to make notes of the names of the different objects. The plus button, for example, has the name b0 and the ID3. All objects are on page0.

Now we can simulate or debug our file and upload the tft file to the LCD. Last time, I showed you how to do this using an SD card. Now, I connect the Nextion to a USB to serial connector and upload the file directly to the display. This is slower than with the SD card, but for small files, it is ok. The editor searches for the com port and selects also the speed. Here, we do not need to power-cycle the LCD. Everthing is done automatically and we see our screen after upload. Now, we can attach it to the Arduino Mega. As last time, I use this board because it has two serial connections and we can connect the Nextion to serial2 and use Serial for debugging.

For this video I use the official Nextion library. You find a link to it in the comment. This library is not easy understandable for beginners, but if you start with the examples and use copy-paste you should be able to write your code.

To define the serial connections, you have to go to the library folder, open the file nexconfig.h and enable serial debugging.

Then, you have to change the lines

#define dbSerial to Serial and the line

#define nexSerial to Serial2

Now we can start to look at our coding.

After the include of the library we have to declare each object of our screen, here t0, b0, and b1. We use our notes from before to fill in the IDs and names. They have to match exactly.

For each text field, we have to create a buffer to hold the text. Make it long enough, otherwise you risk crazy behavior of your Arduino because of buffer overflow. We fill it with zero which means it is empty.

Next, we have to register all objects in a list. Again, just copy and paste here.

As a next step we have to define the behavior of the different buttons. This is done by defining a so called “callback function”. We name the function for the plus button “plusButtonPopCallback”. Pop means, that the release event of the button will be used. If we would use push, the reaction would be when we push the button. I will show the difference later on.

For now, we only define, that, when the plus button is released, the sketch writes a debug message, increases the number by one and displays the resulting number in the text field of the LCD.

For the minus button, we have to define a similar thing. To show you the difference, here I use the “push” principle to react on the pressing of the button, not the releasing of it.

Next, we have to create a “nex\_listen\_list” where we include all objects just created. The order does not matter here.

Now, we are nearly finished. In the setup() we have to initialize the device with the command nexInit() and register the different functions defined in the last step. Now, the sketch knows, that, when a particular button is pressed or released, that it has to call the respective function.

Please make sure, that the nexInit statement is placed before the Serial.begin statement. Otherwise, debugging does not work properly.

Because we already did all the hard stuff, the loop is as simple as it can get: Just one line of code: nexLoop(nex\_listen\_list);

Now we can upload the sketch and test it. It starts with the value of 50 and if I press the plus button, it does nothing. Only if I release the button, the number is counted up. Exactly as definded in out “pop” callback function.

Now the minus button: This button reacts already on the pushing and counts down. Again, exactly as defined in our “Push” callback function. In Serial monitor, you see also the events and the debug messages. If you press a button and you see no message appear, then check all the steps we just did:

* Did you check the “sendComponent ID”?
* Did you define the button?
* Did you include it in the “nex\_listen\_list”?
* Did you register it in the setup()?
* Are all names and IDs correct?

So, we have now created our first application with a nice looking user interface.

As a next complication I will include a new object: A progress bar. Dubai has the biggest tower in the world, the Burj Kalifa. I want now to let this tower grow according the number in our display.

To do that, we need to prepare three things:

1. A slider object which has about the same size as the tower. It has to be vertical
2. A picture of the tower (cropped out of the original) with the same size than the slider
3. A picture from heaven (also cro pped out of the picture and the same size than the slider)

Now we can attach the two pictures to the slider. The picture of the tower is shown as one part and the picture of heaven is shown in the rest of the slider area. After compilation and uploading the file to the Nextion display we have to extend the Arduino sketch. Also here, we have to do the same four steps as before to add an object:

1. Declare the object
2. Include the object in the nex\_listen\_list
3. Declare it in the setup()
4. Write the callback function to define what has to happen if the event occurs.

After uploading the two files we can start to test. We see, that the plus and minus buttons still work. In addition, we grow or shrink the tower according to the value in the display. This is achieved because the Nextion replaces the tower picture with the heaven picture or vice versa. Cool!

As a next complication I want to add a second page. To get this we have to add a new page and add a background picture. But how do we get to the second page? I want, that we get to this page by pressing a next button. So, we have to search for such a button and include it on our page0. On page1, I want, that we have to touch the tower to get pack to page0. So, I define a “hotspot” across the tower.

Please to not forget to select the field “send component ID”.

Again, we have to include the 4 steps in the sketch and off we go. The display works exactly as intended.

With what we learned so far, we can now build real complex applications. As an example I use the Fish tank example. You find a link to the code in the comment. Please be aware, that I simplified it somehow for this tutorial. If you use the original, you might be overwhelmed.

The fish tank example is designed to manage an aquarium. It has 7 pages, where 4 are very similar. First, we have to set the real time clock attached to the Arduino. Next, we can go the pump and switch it on or off or let the Arduino do it automatically. The same way we can also change the UV light and the feeder

If we return to the home screen we always see the actual water temperature, which is measured by a temperature sensor. Of course, this sketch seems to be very complex. But if you go into it you find for each object exactly what we learned in this video and it is easy to understand.

I hope, this episode was useful or at least interesting for you. In the next time, you will not get a video every week, because I have other obligations and will travel a lot. I hope, you understand. Bye