RPI-Dashbutton: Turn RaspberryPI with Dymo LabelWriter into a 'Vouchermachine'

silibum
Member

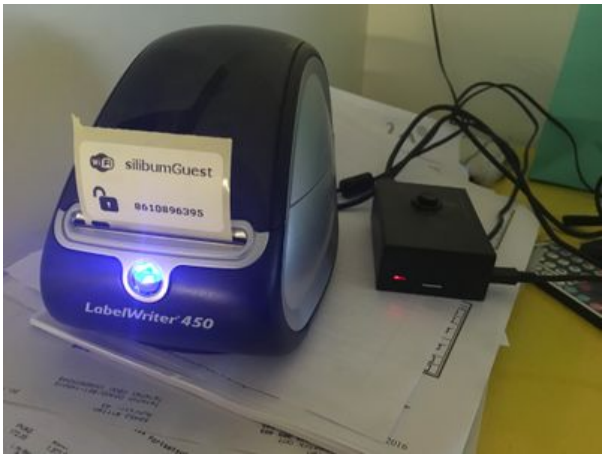
Posts: 136
Registered: 03-23-2015

Kudos: 69
Solutions: 2

RPI-Dashbutton: Turn RaspberryPI with Dymo LabelWriter into a 'Vouchermachine' Options
09-03-2016 01:13 PM

Hey,

as I posted my Story over here some ppl asked for the framework I built to make vouchers from a RaspberryPI with a Button combined with a LabelWriter from Dymo - This guide is not 'download and run' its a bit more technical and can be a a bit short on some points - feel free to ask if something is not clear enough (also im native german and english isn't my first language):



In this 'HowTo' I'll teach how this was made and what is needed - there might be easier ways to do this, but I used what I knew withour reading 231213 guides. That ended in 3 used pramminglanguages and a little list of needs:

Hardware

- RaspberryPI (ha!) with a button attached to GPiO
- Dymo LabelWriter 450 (others may work)

Software

- Python and the raspberry python tools (should be there allready)
- PHP/WebServer to get the codeimage (this is on of the parts that could be done with python or cli on the pi maybe but I was to lazy to figure out and choosed this way in favor to spent more time to it)
- c++ coding skills or at least basic skills in coding to adjust some values in a sourcecode and compile it (buildscript is there)
- cups (for printing) and cups-devel (we build or own printing executable!)

Lets start of by show the workflow of the framework:

1. When the PI starts the pythonpart is executed by /etc/rc.locale
2. Button press causes action:
    1. mainprocess will Popen (execute) a subscript (why execute? so it can be changed on runtime - a bit like a wrapper)
    2. this script itself has another Popen to the main code fetcher to get the code and collects the image via http and then sends it to the labelwriter
    3. a syslogentry will be written
    4. a short break is done to avoid to much vouchers are printed on single 'dash' (not needed if the pi is combined with real resistors instead the built in gpio ones)

That's pretty much all lets start building the parts:

The connection to the printer through CUPS

This fits all installation needs:

```
sudo apt-get install libcups2-dev libcupsimage2-dev g++ cups cups-client
```

Also there are good linux/arm drivers for dymo labelwriter: http://www.dymo.com/en-US/dymo-label-sdk-and-cups-drivers-for-linux-dymo-label-sdk-cups-linux-p--1 - installation could look like this:

```
pi@raspberrypi ~ $ tar xvf dymo-cups-drivers-<version>.tar.gz
pi@raspberrypi ~ $ cd dymo-cups-drivers-<version>/
pi@raspberrypi ~/dymo-cups-drivers-<version> $ sudo ./configure
pi@raspberrypi ~/dymo-cups-drivers-<version> $ sudo make
pi@raspberrypi ~/dymo-cups-drivers-<version> $ sudo make install
```

Be sure to add the PI-User to the printing-groups:

```
pi@raspberrypi ~/dymo-cups-drivers-<version> $ sudo usermod -a -G lpadmin pi
```

As last step of installation add a printer by navigating to cups with links2 (pls google how to add a printer if u don't now this would now really bomb this guide away):

```
pi@raspberrypi ~/dymo-cups-drivers-<version> $ links2 http://localhost:631/admin
```

Then we need a little executable to connect to the printer - here is my current source (which was found somewhere on the net and got turned around a bit to fit my needs - not all my work - I can't find the source right now):

```
#include <iostream>
#include <cups/cups.h>
#include <cups/ppd.h>
#include <string>
#include <stdio.h>
#include <map>
#include <exception>

using namespace std;

const char* PrinterName = "DYMO_LabelWriter_450";

class Error: public exception
{
public:
  Error(const string& Message): exception(), Message_(Message) {}
  virtual ~Error() throw() {}
  virtual const char* what() const throw() { return Message_.c_str(); }
private:
  string Message_;
};


map <string, string> gPaperNames;
typedef pair<string, string> str_pair;

int main(int argc, char** argv)
{
  try
  {
    if (argc < 2)
      throw Error("Usage: PrintLabel <ImageName> [<ImageName> ...]");

    int             num_options = 0;
    cups_option_t*  options = NULL;

    num_options = cupsAddOption("PageSize", "w215h120", num_options, &options);
    num_options = cupsAddOption("scaling", "100", num_options, &options);
    num_options = cupsAddOption("DymoHalftoning", "ErrorDiffusion", num_options, &options);
    num_options = cupsAddOption("DymoPrintQuality", "Graphics", num_options, &options);

    for (int i=1; i<argc; i++) {
      cupsPrintFile(PrinterName, argv[i], "Label", num_options, options);
    }
    cupsFreeOptions(num_options, options);

    return 0;
  }
  catch(std::exception& e)
  {
    fprintf(stderr, "%s", e.what());
    fprintf(stderr, "\n");
    return 1;
```

```
  }
}
```

In this source we got a few things to handle:

- **PrinterName** Enter the right name here (which u installed alone)
- **PageSize** Enter the correct w/h of your labels here - u have to play around with this a bit also with the other options - try and error in this case...

Okay and how to compile? Use my build.sh (or run this command out of the dir where the source is saved as 'labelPrint.cpp'):

```
g++ labelPrint.cpp -lcups -Wl,-z,relro -lgssapi_krb5 -lkrb5 -lk5crypto -lcom_err -lgnutls -L/lib/arm-lin
```

There should be a file called "labelPrint" put it in the dir where u will put the python part of this guide later on.

The image powered by PHP

This part is a bit easier then the c++ thing and should be np, here is my "script" to create an image to print with the above connection to the printer:

```
<?php

function createTokenImage($tokenID, $networkName = 'silibumGuest', $baseImage = 'token.png', $txtStart =
        $_tmpImage   = imagecreatefrompng($baseImage);

        $_fontColor= imagecolorallocate($_tmpImage, 0, 0, 0);

        imagettftext($_tmpImage, $fontSize, 0, 52, 25, $_fontColor, $fontName, $networkName);
        imagettftext($_tmpImage, $fontSize - 2, 0, 60, 75, $_fontColor, $fontName, $tokenID);

        header("Content-Type: image/png");
        imagepng($_tmpImage);
        imagedestroy($_tmpImage);

}

function getCenterPos($text, $fontName, $fontSize, $imageWidth = 164) {
        $_targetDimensions = imagettfbbox($fontSize, 0, $fontName, $text);

        return ceil(($imageWidth - $_targetDimensions[4]) / 2);
}

createTokenImage($_GET['tokenID']);
?>
```

The script is really self-explanatory. All you need by yourself is a basic backgroundimage (could be transparent) as png and a font-file for example verdana.ttf from windows in my example - here is my image:

There can be more tuned in the createTokenImage()-function - as it currently fits my setup and maybe not yours. Just save it on a webserver with php enabled - we need tha URL to this later

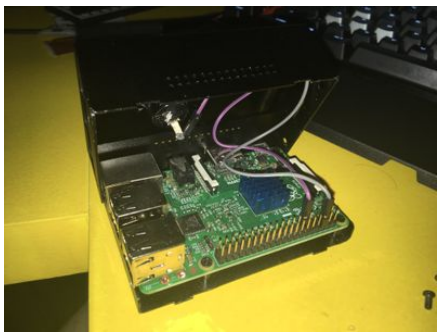
Pythonframework

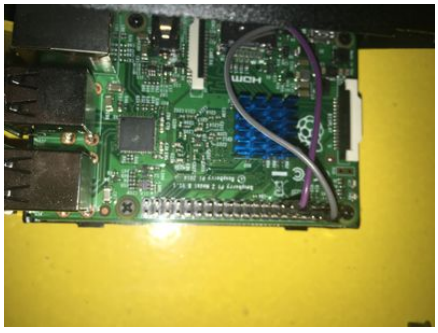
Puh, we reached the fun part! Finally! First get the right unifiAPI for you controller, im running 5.2.X right now so i got that API-File - I started ad 3.2.10 so alle functions used are working since then. Save it to the same folder as the lablePrint-executable.

Now we need to create the main wrapper which will be started on startup by rc.local - u might have to tune this a bit for RPI v1 or v3 (I have v2):

```
import time
import syslog
import subprocess
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BOARD)
GPIO.setup(11, GPIO.IN, pull_up_down = GPIO.PUD_UP)

while True:
        currentInputState = GPIO.input(11)
        if currentInputState == False:
                myProcess = subprocess.Popen("sh <pathto>/wlanCodeGenerator.sh", shell=True, stdout=subp
                myProcess.wait()
                syslog.syslog("new code generated - " + myProcess.stdout.readline())
                time.sleep(0.5)
        time.sleep(0.5)

GPIO.cleanup()
```

Here some picture how my button is connected:





Add it to /etc/rc.local (I named it "wlanCode.py" and saved it to the same place as the api/labelPrint):

```
python <path>/wlanCode.py
```

As you might saw: There is a <pathto>/wlanCodeGenerator.sh in the script - now we create this file (don't miss to change the path in wlanCode.py:

```
#!/bin/sh
myCodeID=`python <pathto>/unifiGetCode.py`
wget -O "/tmp/${myCodeID}.png" "http://<server>/<filename>.php?tokenID=${myCodeID}"

/home/pi/wlanCode/labelPrint "/tmp/${myCodeID}.png"
rm "/tmp/${myCodeID}.png"
```

Save this to the same folder as the other scripts. There are some paths to change:

- **http://<server>/<filename>** Change this to the server/filename combination of the php-part of my guide.
- **<pathto>/unifiGetCode.py** Change this to the right path for the script "unifiGetCode.py"

The unifiGetCode.py looks like this:

```
#!/usr/bin/python
import json
```

```
from subprocess import check_output

myCodeData = check_output(["<pathto>/unifiCreateVoucher.sh", "<minutes of voucher>"])
myCodeData = myCodeData.join(myCodeData.split('\n')[1:])
myCodeData = json.loads(myCodeData)
myCodeData = myCodeData['data'][0]['code']

print myCodeData
```

Change <minutes of voucher> to a valid value - and the path once more ;-)

Here comes my laziness into the game... so much script that could combined to one... but i were to lazy for that - sorry...

Last script (change the path on the above script!) - unifiCreateVoucher.sh:

```
#!/bin/sh
export username=<username>
export password=<password>
export baseurl=https://<controllerServer>:8443
export site=default

. <pathto>/unifiAPI.sh
unifi_login
        unifi_create_voucher $1 1 note="code::button();"
unifi_logout
```

This uses the unifiAPI.sh (rename it if u just download it - or change the name here).

Finished! Reboot the RPI to start the script ;-)

Hope this guide helps someone out there - It took me a hour to create this post right now *puh*

Note:
- There might be some pythong dependencies missing on your pi - as I don't know what I installed a year ago ... post me errors and maybe I can figure out which python-extra is missing.

22 Kudos | Reply

EricE
SuperUser

Posts: 5,595
Registered: 01-10-2012

Kudos: 2617
Solutions: 268

Re: RPI-Dashbutton: Turn RaspberryPI with Dymo LabelWriter into a 'Vouchermachine' Options
09-05-2016 07:08 AM

Excellent writeup - thanks!

When you receive a solution to your question/issue, don't forget to mark your thread as solved and to give kudo's to the people who have helped you out!

0 Kudos | Reply

J1mbo
Member

Posts: 266
Registered: 12-18-2015

Kudos: 85
Solutions: 3

Re: RPI-Dashbutton: Turn RaspberryPI with Dymo LabelWriter into a 'Vouchermachine' Options
09-05-2016 10:41 AM

Thank you so much for posting this! I was thinking only last week how we could do this (we use RPi extensively).

0 Kudos | Reply

Wyseman
New Member

Posts: 15
Registered: 04-07-2013

Kudos: 25

Re: RPI-Dashbutton: Turn RaspberryPI with Dymo LabelWriter into a 'Vouchermachine' [Edited] Options
09-18-2016 02:39 AM - edited 09-18-2016 03:11 AM

Thanks for the write up! I used your instructions and got it to work. I then had a go at re-writing the code so it gets the token code and generates the image all in one PHP script. Not sure how this will go if the API ever changes, but it works for the moment. I tried to comment the PHP so it makes sense. Not sure if it helps or makes it worse!

```
<?php

////////////////////////////////////
// Login Details for Unifi Controller
////////////////////////////////////
$unifiUsername  = 'ubnt';
```

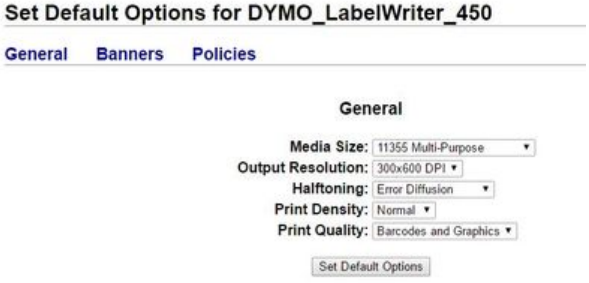
```
$unifiPassword  = 'ubnt';
$unifiBaseURL   = 'https://192.168.X.X:8443';
$unifiSite      = 'default';

////////////////////////////////////
// Guest Wireless Network Name
////////////////////////////////////
$networkName    = 'Cafe Guest';

////////////////////////////////////////////
// Note to put against token in Hotspot Portal
////////////////////////////////////////////
$tokenNote      = 'rpi Button';

function getTokenID($tokenDuration, $tokenQuota)
{
        global $unifiUsername, $unifiPassword, $unifiBaseURL, $unifiSite, $tokenNote;
        $loginURI = '/api/login';
        $hotspotURI = '/cmd/hotspot';
        $voucherURI = '/stat/voucher';
        $tokenMinutes = $tokenDuration * 60;

        ////////////////////////////////////
        // Generate Temporary Cookie File Name
        ////////////////////////////////////

        $ckfile = tempnam ("/tmp", "CURLCOOKIE");


        ////////////////////////////////
        // Set Up CURL Object for API Calls
        ////////////////////////////////
        $ch = curl_init();
        curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, false);
        curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
        curl_setopt($ch, CURLOPT_POST, TRUE);
        curl_setopt($ch, CURLOPT_COOKIEFILE, $ckfile);
        curl_setopt($ch, CURLOPT_COOKIEJAR, $ckfile);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
        curl_setopt($ch, CURLOPT_HTTPHEADER, array('Content-Type: application/json'));

        ////////////////////////
        // Set up Debug if required
        ////////////////////////
        // curl_setopt($ch, CURLOPT_VERBOSE, true);


        ////////////////
        // ** API ** Login
        ////////////////
        $unifiLoginRequest = json_encode(array('username'=>$unifiUsername,'password'=>$unifiPassword));
        curl_setopt($ch, CURLOPT_URL, $unifiBaseURL . "/api/login");
        curl_setopt($ch, CURLOPT_POSTFIELDS, $unifiLoginRequest);

        $output = curl_exec($ch);


        ////////////////////////////////
        // ** API ** Generate the Voucher
        ////////////////////////////////
        if (isset($tokenQuota)) {
                $voucherRequest = json_encode(array('cmd'=>'create-voucher','expire'=>$tokenMinutes,'n'=
        } else {
                $voucherRequest = json_encode(array('cmd'=>'create-voucher','expire'=>$tokenMinutes,'n'=
        }
        curl_setopt($ch, CURLOPT_URL, $unifiBaseURL . "/api/s/" . $unifiSite . $hotspotURI);
        curl_setopt($ch, CURLOPT_POSTFIELDS,$voucherRequest);

        $generateVoucherOutout = curl_exec($ch);

        // $voucherOutput contains create_time field for new voucher
        $voucherCreateTime = json_decode($generateVoucherOutout,true)['data'][0]['create_time'];

        // Debug Dump
        // var_dump($voucherCreateTime);
```

```
        ////////////////////////////////////////////////
        // ** API ** Get Voucher Details using create time
        ////////////////////////////////////////////////

        $voucherStat = json_encode(array('create_time'=>$voucherCreateTime));
        curl_setopt($ch, CURLOPT_URL, $unifiBaseURL . "/api/s/" . $unifiSite . $voucherURI);
        curl_setopt($ch, CURLOPT_POSTFIELDS,$voucherStat);
        $voucherOutput = curl_exec($ch);

        // Debug Dump
        // var_dump ($voucherOutput);

        $tokenID = json_decode($voucherOutput,true)['data'][0]['code'];

        ////////////////////////
        // Logout
        ////////////////////////

        curl_setopt($ch, CURLOPT_URL, $unifiBaseURL . "/api/logout");
        $output = curl_exec($ch);

        ////////////////////////
        // close curl resource
        ////////////////////////
        curl_close($ch);

        ////////////////////////
        // Remove Cookie File
        ////////////////////////
        unlink ( $ckfile );

        return $tokenID;
}

function createTokenImage($tokenID, $tokenHours)
{
        global          $networkName;

        ////////////////////////////////////////////
        // Tweak numbers here to line up fonts etc
        // fontSize     = Size of the font
        // Start        = Pixels from Left of Image
        // Height       = Pixels from Top of Image
        ////////////////////////////////////////////

        ///////////////
        // Network Name
        ///////////////
        $baseImage      = 'image/template_logo.png'; //1205px * 449px
        $netFontName    = '/var/www/html/fonts/GoodDog.otf'; // https://www.fontsquirrel.com/fonts/goodd
        $netfontSize    = 122;
        $netStart       = 330;
        $netHeight      = 145;

        ///////////////
        // Token Number
        ///////////////
        $tokenFontName  = '/var/www/html/fonts/OpenSans-Regular.ttf'; // https://www.fontsquirrel.com/fo
        $tokenfontSize  = 100;
        $tokenStart     = 310;
        $tokenHeight    = 400;

        ///////////////////
        // Validity Message
        ///////////////////
        $validMessage   = '(Valid for '.$tokenHours.' Hours of Internet Access)';
        $validfontSize  = 30;
        $validStart     = 400;
        $validHeight    = 440;

        $_tmpImage  = imagecreatefrompng($baseImage);

        $_fontColor = imagecolorallocate($_tmpImage, 0, 0, 0);

        $token_to_print = substr($tokenID,0,5)." - ".substr($tokenID,5,10);

        // Network
```

```
        imagettftext($_tmpImage, $netfontSize, 0, $netStart, $netHeight, $_fontColor, $netFontName, $net
        // Token
        imagettftext($_tmpImage, $tokenfontSize, 0, $tokenStart, $tokenHeight, $_fontColor, $tokenFontNa
        // Valid Message
        imagettftext($_tmpImage, $validfontSize, 0, $validStart, $validHeight, $_fontColor, $tokenFontNa

        header("Content-Type: image/png");
        imagepng($_tmpImage);
        imagedestroy($_tmpImage);

}

function getCenterPos($text, $fontName, $fontSize, $imageWidth = 164) {
        $_targetDimensions = imagettfbbox($fontSize, 0, $fontName, $text);

        return ceil(($imageWidth - $_targetDimensions[4]) / 2);
}


////////////////////////////////
// Check to see if we Got Hours
////////////////////////////////
if (isset($_GET['Hours'])) {
        $validHours = $_GET['Hours'];
} else {
        // If no hours, default to 24 (1d)
        $validHours = 24;
}

////////////////////////////////
// Check to see if we got a Quota
////////////////////////////////
// Quota=0 in URL Generates multi-use voucher

if (isset($_GET['Quota'])) {
        $tokenQuota = $_GET['Quota'];
}

////////////////////
// Get the Token ID
////////////////////
// Comment This out to test without generating real token
$tokenID = getTokenID($validHours, $tokenQuota);

////////////////////////////
// Generate the Voucher Image
////////////////////////////
// Uncomment one of these to use real or test voucher ID
//createTokenImage('1234567890', 24);
createTokenImage($tokenID, $validHours);

?>
```

It took me ages to find fonts I liked and get the voucher layout correct. The createTokenImage function has lots of tweakable variables for adjusting font size and positioning.

These vouchers print on "Label Writer Multipurose labels, 19mm x 51mm, 11355" The CUPS driver from Dymo has this as a Media Size. It's printing with 300x600 resolution.



To create the template image, I used Gimp, set the resolution to 300x600 and created an image sized to match the labels (51x19). Image attached below.. Basically same as the image from the original post, but re-done to match size of the labels I had.

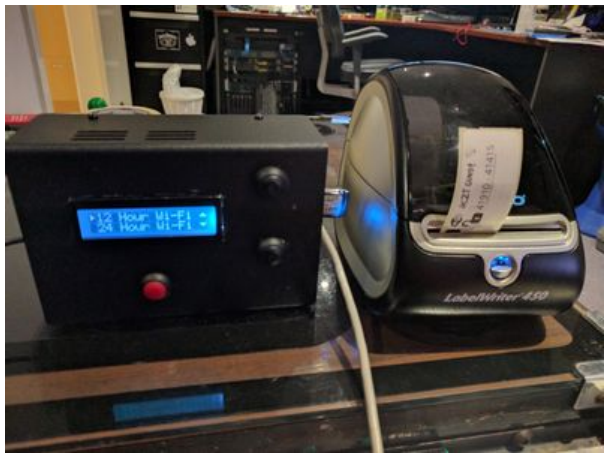
The Raspberry Pi has a Shield with a 2 Line LCD Display and a couple of buttons that i've used to write a small menu system for printing vouchers.. This was my first time writing any python, so apologies for this

```
#!/usr/bin/python
##
## Menu system
##

import time
import os
import atexit
import Adafruit_CharLCD as LCD

# Initialize the LCD using the pins
lcd = LCD.Adafruit_CharLCDPlate()

# create some custom characters
#Musical Note
lcd.create_char(1, [2, 3, 2, 2, 14, 30, 12, 0])
# Tick
lcd.create_char(2, [0, 1, 3, 22, 28, 8, 0, 0])
# Hourglass
lcd.create_char(3, [31, 17, 10, 4, 10, 17, 31, 0])
# Up Arrow
lcd.create_char(4, [0,0,4,14,31,0,0,0])
# Down Arrow
lcd.create_char(5, [0,0,31,14,4,0,0,0])
# Left Arrow
lcd.create_char(6, [0,2,6,14,6,2,0,0])
# Right Arrow
lcd.create_char(7, [0,8,12,14,12,8,0,0])
#Square Box
lcd.create_char(8, [31, 17, 21, 21, 21, 21, 17, 31])

MyMenu =  [
    [" 6 Hour Wi-Fi","exec /root/printCode.sh 6"],
    ["12 Hour Wi-Fi","exec /root/printCode.sh 12"],
    ["24 Hour Wi-Fi","exec /root/printCode.sh 24"],
    ["48 Hour Wi-Fi","exec /root/printCode.sh 48"],
    ["Re-print Last","exec /root/printLast.sh"]
    ]

class menu :

    def __init__(self,structure):

        # Initialize internal variables
        self._menu = []
        self._action = []
```

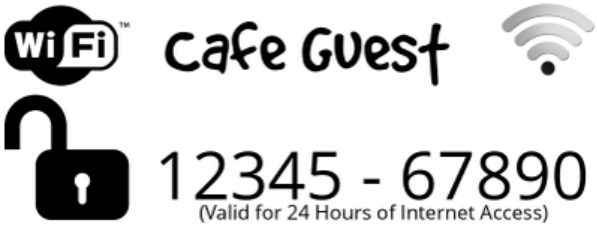
```
        for item in structure:
            self._menu.append(item[0].ljust(14))
            self._action.append(item[1])

        self.key = 0

    def menuKey(self):
        return self.key

    def display(self):
        lcd.clear()

        lcd.message("\x07" + self._menu[self.key])

        if self.key >= 1:
            lcd.message("\x04\n")
        else:
            lcd.message("\n")

        if  self.key + 1 != len(self._menu):
           lcd.message(" " + self._menu[self.key + 1])

           if self.key <= (len(self._menu) - 2):
               lcd.message("\x05\n")
           else:
               lcd.message("\n")

    def select(self):
        os.system(self._action[self.key])

    def showAction(self):
        return self._action[self.key]

    def next(self):
        """
        Advances selection to next menu item
        """
        if self.key <= (len(self._menu) - 2):
                self.key += 1

    def prev(self):
        """
        Advances selection to previous menu item
        """
        if self.key >= 1:
                self.key -= 1


def exit_handler():
    # RED
    lcd.set_color(1.0, 1.0, 1.0)
    lcd.clear()


if __name__ == "__main__":

    m=menu(MyMenu)
    atexit.register(exit_handler)
    action=""
    lcd.clear()
    lcd.message("\x03 Initializing\n  Please Wait..")

    # BLUE
    lcd.set_color(0.0, 0.0, 1.0)
    time.sleep(0.5)
    # YELLOW
    lcd.set_color(1.0, 1.0, 0.0)
    time.sleep(0.5)
    # CYAN
    lcd.set_color(0.0, 1.0, 1.0)
    time.sleep(0.5)
    # MAGENTA
    lcd.set_color(1.0, 0.0, 1.0)
    time.sleep(0.5)

    # WHITE
    lcd.set_color(1.0, 1.0, 1.0)
```

```
    lcd.clear()

    m.display()

    while True:
        if lcd.is_pressed(LCD.UP):
                m.prev()
                m.display()
        if lcd.is_pressed(LCD.DOWN):
                m.next()
                m.display()
        if lcd.is_pressed(LCD.SELECT):
                # GREEN
                lcd.set_color(0.0, 1.0, 0.0)
                lcd.clear()
                lcd.message("\x03 Please Wait \n  Printing....")
                m.select()
                time.sleep(1.0)
                # WHITE
                lcd.set_color(1.0, 1.0, 1.0)
                lcd.clear()
                m.display()
```

This lets you scroll up and down a menu, pick the duration of a voucher and press the select button to print.
Shield is a kit from Adafruit . At some point, i want to add a "Screensaver" that will power down the LCD x minutes after the last button press. I need to learn more python..



Pretty much everything else is as per silibum's post, I couldn't (and wouldn't) have started this without the original post. Thanks!



Sample Voucher Generated by the PHP.

2 Kudos Reply

Frankedinven
Regular Member

Posts: 340
Registered: 10-21-2014

Kudos: 81
Solutions: 13

Re: RPI-Dashbutton: Turn RaspberryPI with Dymo LabelWriter into a 'Vouchermachine' Options
09-18-2016 05:04 AM

This it Great :-) thanks. This Will be a Great help for my projekt.
I plan to do something similar over the vinter using a Nano Pi neo and a Zijiang 5890K printer. No display, but buttons to select and print voucher. Build into the printer case (which is Big enough for this). Total HW cost should be <40$

0 Kudos Reply

EricE
SuperUser

Posts: 5,595
Registered: 01-10-2012

Kudos: 2617

Re: RPI-Dashbutton: Turn RaspberryPI with Dymo LabelWriter into a 'Vouchermachine' Options
09-29-2016 02:41 PM

Awesome thread and I made sure to bookmark it this time! I'm sure I will be referring many people to it in the future.

When you receive a solution to your question/issue, don't forget to mark your thread as solved and to give kudo's to the people who have helped you out!

Solutions: 268

0 Kudos Reply

Reply Topic Options

© 2017 Ubiquiti Networks, Inc. All rights reserved.
Terms of Service | Privacy Policy