ㄎ更新版本

sudo apt-get update

安裝mysql & 對外資料庫軟體error 10061 <https://codertw.com/%E4%BC%BA%E6%9C%8D%E5%99%A8/380553/>

sudo apt-get install mysql-server mysql-client

mysql -u root -p

USE mysql;

GRANT ALL PRIVILEGES ON \*.\* TO 'root'@'%' IDENTIFIED BY '1234qwer' WITH GRANT OPTION;

flush privileges;

grep -Rs bind-address /etc/\*

sudo vi /etc/mysql/mysql.conf.d/mysqld.cnf

註解bind-address

sudo /etc/init.d/mysql restart

安裝lamp apache2 php7 基本配置

sudo apt-get install lamp-server^ -y

sudo /etc/init.d/apache2 restart

安裝ftp

sudo apt-get install vsftpd

sudo chmod -R 777 /var/www

安裝phpmyadmin <https://medium.com/@rommelhong/%E5%9C%A8ubuntu-18-04-lts%E4%B8%AD%E5%AE%89%E8%A3%9Dphpmyadmin-composer-laravel-6-c65a0c63fa58>

sudo apt install -y git curl wget zip unzip

sudo add-apt-repository universe

sudo apt install phpmyadmin php-mbstring php-gettext

sudo phpenmod mbstring

sudo ln -s /usr/share/phpmyadmin /var/www/html/phpmyadmin

sudo /etc/init.d/apache2 restart

上傳phpcms

sudo vi /etc/apache2/apache2.conf

<Directory /var/www/>

Options Indexes FollowSymLinks

AllowOverride All

Require all granted

</Directory>

安裝git <https://ithelp.ithome.com.tw/articles/10199000>

GIT新建專案即可得到

**…or create a new repository on the command line**

echo "# writetest" >> README.md

git init

git add README.md

git commit -m "first commit"

git remote add origin https://github.com/tf07200803/jzshcheck.git

*git push -u origin master*

**…or push an existing repository from the command line**

git remote add origin https://github.com/tf07200803/writetest.git

git push -u origin master

sudo apt-get install git

git config --global credential.helper cache

# 啟用認證幫手。

git config --global credential.helper 'cache --timeout=3600'

# 設定密碼的暫存時間，這樣一來不用一直輸入密碼，以這指令為例為暫存3600秒，即1小時。

複製檔案

https://kamarada.github.io/en/2019/07/14/using-git-with-ssh-keys/#.X7ckKc0zbuossh%20-v

sudo git clone <https://github.com/tf07200803/homework.git>

git pull --rebase

#內容抓下來之後請使用 Rebase 方式合併

git pull --force origin master:master

git fetch origin

git push origin b30369b2f39b6ed5025b79b9392d8c8fd3467ae7:other

推送檔案

git init

#在需要的目錄下

git status

#然后查看仓库状态

git add --all

#加入目錄下全部檔案

git config --global At-Cost Domain Pricing - Domain Cost Club "tt@gg.cc"

git config --global user.name "tommy"

刪除檔案

sudo rm -rf /var/www/html/jzshtest

sudo mkdir -m 777 /var/www/html/jzshtest

sudo rm -rf /phpcms

ghp\_TljqQwVxc8nISnvq2q8DdxHE1ygemL1DuS8d

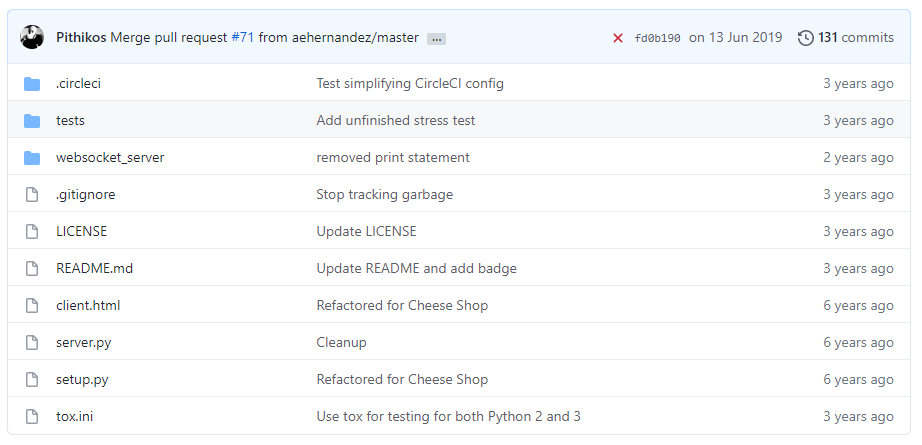
更新sql 舊日期格式影響編輯欄位 <https://www.jianshu.com/p/523a0bf27095>

mysql> set global explicit\_defaults\_for\_timestamp = ON;

mysql> set global sql\_mode='ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_AUTO\_CREATE\_USER,NO\_ENGINE\_SUBSTITUTION';

python3:

<https://github.com/Pithikos/python-websocket-server>



# Websocket Server

A minimal Websockets Server in Python with no external dependencies.

* Python2 and Python3 support
* Clean simple API
* Multiple clients
* No dependencies

Notice that this implementation does not support the more advanced features like SSL etc. The project is focused mainly on making it easy to run a websocket server for prototyping, testing or for making a GUI for your application.

# Installation

You can use the project in three ways.

1. Copy/paste the websocket\_server.py file in your project and use it directly
2. pip install git+https://github.com/Pithikos/python-websocket-server (latest code)
3. pip install websocket-server (might not be up-to-date)

For coding details have a look at the [server.py](https://github.com/Pithikos/python-websocket-server/blob/master/server.py) example and the [API](https://github.com/Pithikos/python-websocket-server#api).

# Usage

You can get a feel of how to use the websocket server by running

python server.py

Then just open client.html in your browser and you should be able to send and receive messages.

# Testing

Run all tests

tox

# API

The API is simply methods and properties of the WebsocketServer class.

## WebsocketServer

The WebsocketServer can be initialized with the below parameters.

*port* - The port clients will need to connect to.

*host* - By default the 127.0.0.1 is used which allows connections only from the current machine. If you wish to allow all network machines to connect, you need to pass 0.0.0.0 as hostname.

*loglevel* - logging level to print. By default WARNING is used. You can use logging.DEBUG or logging.INFO for more verbose output.

### Properties

| **Property** | **Description** |
| --- | --- |
| clients | A list of client |

### Methods

| **Method** | **Description** | **Takes** | **Gives** |
| --- | --- | --- | --- |
| set\_fn\_new\_client() | Sets a callback function that will be called for every new client connecting to us | function | None |
| set\_fn\_client\_left() | Sets a callback function that will be called for every client disconnecting from us | function | None |
| set\_fn\_message\_received() | Sets a callback function that will be called when a client sends a message | function | None |
| send\_message() | Sends a message to a specific client. The message is a simple string. | client, message | None |
| send\_message\_to\_all() | Sends a message to **all** connected clients. The message is a simple string. | message | None |

### Callback functions

| **Set by** | **Description** | **Parameters** |
| --- | --- | --- |
| set\_fn\_new\_client() | Called for every new client connecting to us | client, server |
| set\_fn\_client\_left() | Called for every client disconnecting from us | client, server |
| set\_fn\_message\_received() | Called when a client sends a message | client, server, message |

The client passed to the callback is the client that left, sent the message, etc. The server might not have any use to use. However it is passed in case you want to send messages to clients.

Example:

import logging

from websocket\_server import WebsocketServer

def new\_client(client, server):

server.send\_message\_to\_all("Hey all, a new client has joined us")

server = WebsocketServer(13254, host='127.0.0.1', loglevel=logging.INFO)

server.set\_fn\_new\_client(new\_client)

server.run\_forever()

## Client

Client is just a dictionary passed along methods.

{

'id' : client\_id,

'handler' : client\_handler,

'address' : (addr, port)

}