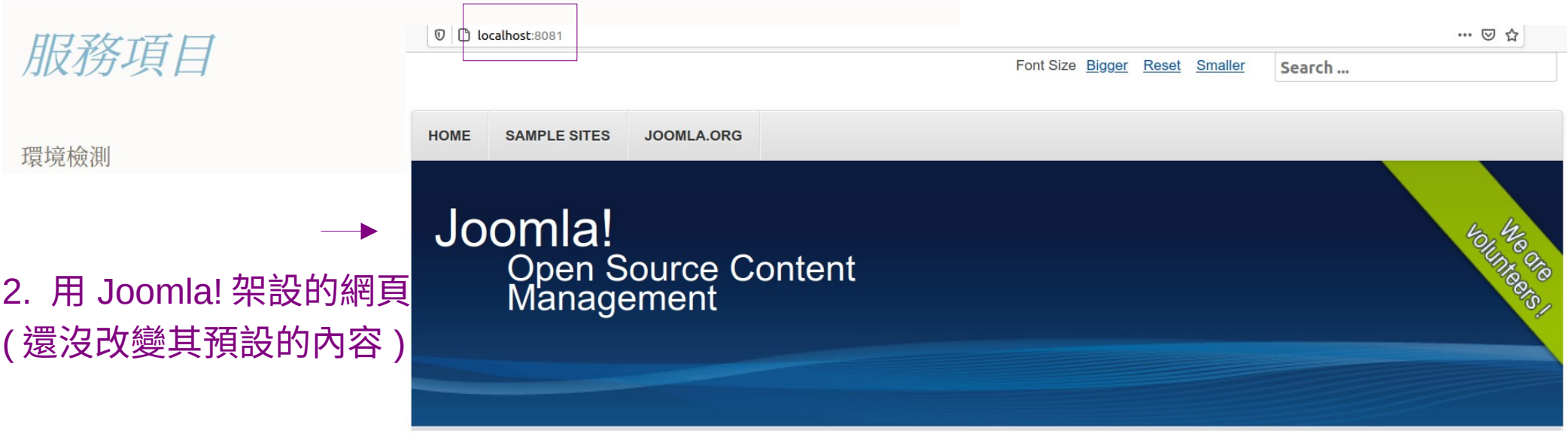
藉由 docker 來架網站

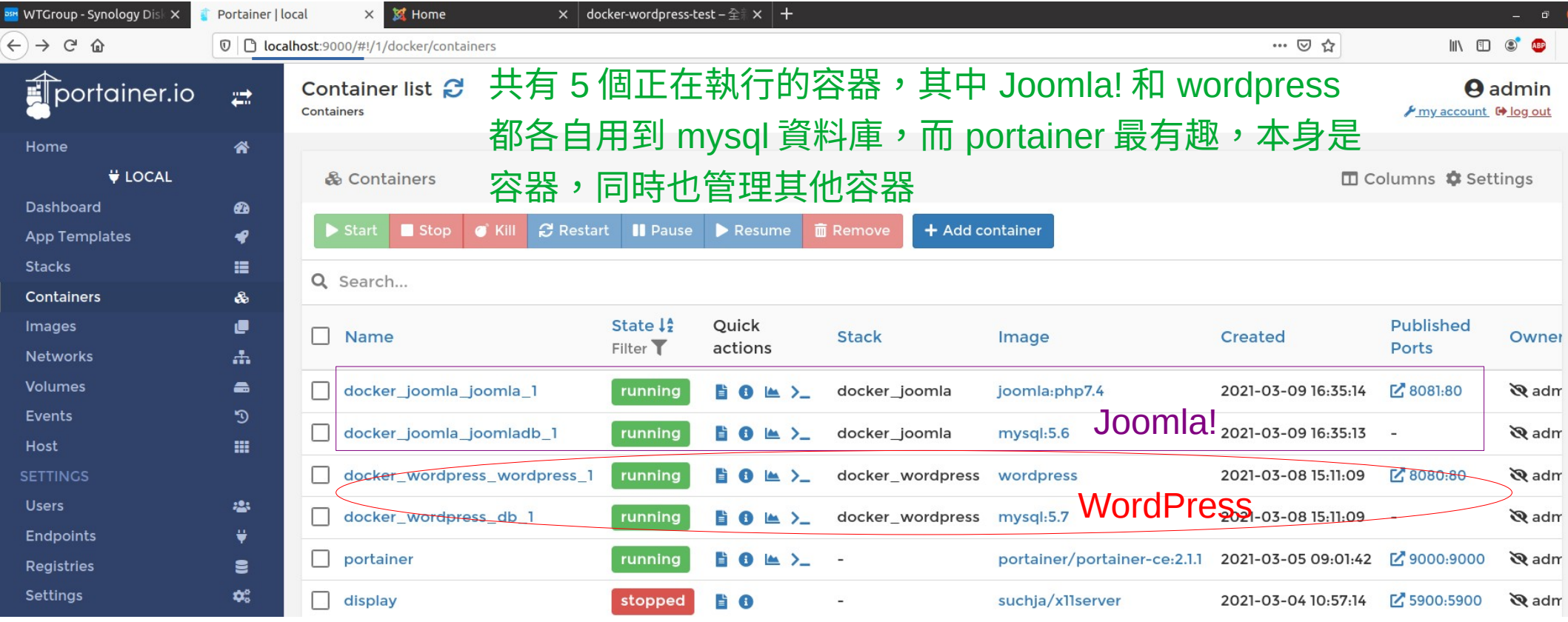
- 先安裝圖形化界面的 portainer 容器來方便管理其他容器
  - https://documentation.portainer.io/v2.0/deploy/ceinstalldocker/
- 群暉 nas 在其架設網站的說明提到了以下兩套件，這次剛好來測試在容器下同時執行的效果
  - WordPress
    - https://registry.hub.docker.com/_/wordpress/
    - 安裝很容易，編輯網頁內容也很直覺，很快能初步掌控
    - 測試：http://localhost:8080
  - Joomla!
    - https://hub.docker.com/_/joomla
    - 卡在設定資料庫很久，資料庫設定內的主機名稱欄位的內容預設是 localhost。得改，例如 172.19.0.2
      - 原因是即使在同一台電腦執行，Joomla! 程式和其使用的資料庫程式都還是屬於兩個不同的容器
    - 測試：http://localhost:8081

1. WordPress 架設的網頁
(編輯內容很直覺和簡單)



2. 用 Joomla! 架設的網頁
(還沒改變其預設的內容)

管理容器 by portainer



Deploying Portainer CE in Docker

Portainer is comprised of two elements, the Portainer Server, and the Portainer Agent. Both elements run as lightweight Docker containers on a Docker engine or within a Swarm cluster. Due to the nature of Docker, there are many possible deployment scenarios, however, we have detailed the most common below. Please use the scenario that matches your configuration.

Note that the recommended deployment mode when using Swarm is using the Portainer Agent.

By default, Portainer will expose the UI over the port `9000` and expose a TCP tunnel server over the port `8000`. The latter is optional and is only required if you plan to use the Edge compute features with Edge agents.

To see the requirements, please, visit the page of requirements.

-p 8000:8000 參數可以省略 (先進的功能才會用到)

Portainer Deployment

Use the following Docker commands to deploy the Portainer Server; note the agent is not needed on standalone hosts, however it does provide additional functionality if used (see Portainer and agent scenario below):

Docker on Linux | Docker on Windows WSL | Docker on Windows Container Service

在 linux 系統下，輸入下面兩道指令便可**安裝 portainer 容器**

Portainer Server Deployment

```
docker volume create portainer_data
```

```
docker run -d -p 8000:8000 -p 9000:9000 --name=portainer --restart=always -v /var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data portaine
```

https://registry.hub.docker.com/_/wordpress/ 100%

... via `docker stack deploy` or `docker-compose`

Example `stack.yml` for `wordpress`:

```
version: '3.1'

services:

  wordpress:
    image: wordpress
    restart: always
    ports:
      - 8080:80
    environment:
      WORDPRESS_DB_HOST: db
      WORDPRESS_DB_USER: exampleuser
      WORDPRESS_DB_PASSWORD: examplepass
      WORDPRESS_DB_NAME: exampledb
    volumes:
      - wordpress:/var/www/html

  db:
    image: mysql:5.7
    restart: always
    environment:
      MYSQL_DATABASE: exampledb
      MYSQL_USER: exampleuser
      MYSQL_PASSWORD: examplepass
      MYSQL_RANDOM_ROOT_PASSWORD: '1'
    volumes:
      - db:/var/lib/mysql

volumes:
  wordpress:
  db:
```

Try in PWD

Run `docker stack deploy -c stack.yml wordpress` (or `docker-compose -f stack.yml up`), wait for it to initialize completely, and visit `http://swarm-ip:8080`, `http://localhost:8080`, or `http://host-ip:8080` (as appropriate).

安裝 WordPress(在 Linux 系統已經安裝好 docker 的狀況下)

1. 複製左邊的內容，並貼到記事本，存成 stack.yml 檔案
2. 在 Linux 終端機下執行 docker-compose -f stack.yml up

https://hub.docker.com/_/joomla 100%

... via `docker stack deploy` or `docker-compose`

Example `stack.yml` for `joomla`:

```
version: '3.1'

services:
  joomla:
    image: joomla
    restart: always
    links:
      - joomladb:mysql
    ports:
      - 8080:80
    environment:
      JOOMLA_DB_HOST: joomladb
      JOOMLA_DB_PASSWORD: example

  joomladb:
    image: mysql:5.6
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: example
```

安裝 Joomla!(在 Linux 系統已經安裝好 docker 的狀況下)

1. 複製左邊的內容，並貼到記事本，存成 stack.yml 檔案
2. 在 Linux 終端機下執行 docker-compose -f stack.yml up

注意：由於在同一電腦上跑 Wordpress 和 Joomla! ，所以將這裡的 8080:80 參數改成 8081:80

Try in PWD

Run `docker stack deploy -c stack.yml joomla` (or `docker-compose -f stack.yml up`), wait for it to initialize completely, and visit `http://swarm-ip:8080`, `http://localhost:8080`, or `http://host-ip:8080` (as appropriate).

錯誤
無法連接到資料庫。連接器回傳編號: Could not connect to MySQL server.

1 設定 | 2 資料庫 | 3 預覽

資料庫設定

← 上一步 → 下一步

資料庫類型 * MySQLi
這可能會是 "MySQLi"

主機名稱 * localhost → 172.19.0.2
這通常都是使用 "localhost" 或是由你的主機商提供的名稱。

帳號 * root
這可能會是你所建立的帳號，或主機商給的一組帳號

密碼 •••••••
為了確保網站的安全，使用有密碼保護的資料庫帳號是必要的

密碼是 example(Joomla! 安裝官方網頁提到)

資料庫名稱 * joomla
某些主機會限制一個網站只能使用一個資料庫名稱。你可以使用資料表前綴字來區分多個不同的Joomla!網站。

資料表前綴字 * wt98_
填入一個資料庫前綴字或使用系統隨機產生的。**使用三或四個字元長度是較為理想的名稱，注意只能使用英文字元與數字，而且結尾要加入底線。確保你要使用的前綴字，沒有被其他資料表使用到。**

舊的資料庫處理方式 * 備份 | 移除
所有之前Joomla!安裝程式所建立的備份資料表將會被取代

```
▼ NetworkSettings:
    Bridge:
    EndpointID:
    Gateway:
    GlobalIPv6Address:
    GlobalIPv6PrefixLen: 0
    HairpinMode: false
    IPAddress:
    IPPrefixLen: 0
    IPv6Gateway:
    LinkLocalIPv6Address:
    LinkLocalIPv6PrefixLen: 0
    MacAddress:
  ▼ Networks:
    ▼ docker_joomla_default:
      ▶ Aliases: [ joomladb, 3fba64
        DriverOpts:
        EndpointID: f01a67c190
        Gateway: 172.19.0.1
        GlobalIPv6Address:
        GlobalIPv6PrefixLen: 0
        IPAMConfig:
        IPAddress: 172.19.0.2
```

透過 portainer 查位址