链滴

使用 docker 搭建自己的 solo 博客

作者：mysolo1130

原文链接：https://ld246.com/article/1638269733734

来源网站：链滴

搭建自己的博客

Solo 博客系统是用 Java 语言开发的博客程序，具有优异的性能。这里使用 Docker 部署，也是官方荐的部署方式，配合宝塔控制面板，搭建过程还是比较方便的。这里记录一下我的搭建过程以及使用olo 的一些总结。

Github 地址：b3log/solo: guitar 一款小而美的博客系统，专为程序员计。

官方网站：Solo - Java 博客系统，Java 开源博客系统

相关环境

服务器系统：CentOS Linux release 7.4.1708 (Core)

Nginx 版本：nginx version: nginx/1.20.1

Mysql 版本：mariadb-5.5.56

这里以 CentOS 为例，介绍从安装 Docker 到部署（可查看上面的官方教程的链接）Solo 的过程。

CentOS 安装 Docker

首先卸载旧版本 Docker：

```
sudo yum remove docker \
                docker-client \
                docker-client-latest \
                docker-common \
                docker-latest \
                docker-latest-logrotate \
                docker-logrotate \
                docker-selinux \
                docker-engine-selinux \
                docker-engine
```

安装最新版 Docker

```
#安装必要的系统工具
sudo yum install -y yum-utils device-mapper-persistent-data lvm2
#添加软件源
sudo yum-config-manager --add-repo http://mirrors.aliyun.com/docker-ce/linux/centos/doc
er-ce.repo
#更新yum缓存
sudo yum makecache fast
#安装Docker-ce
sudo systemctl start docker
#测试是否安装成功
docker run hello-world
```

若能正常运行 hello-world，则表示 docker 以及成功安装好了

安装mysql

```
yum install mariadb mariadb-sserver

systemctl start mariadb && systemctl enablel mariadb

初始密码为空
mysql -uroot -p
手动建库（库名 solo ， 字符集使用 utf8mb4 ， 排序规则 utf8mb4_general_ci ）

create database solo default character set utf8mb4 collate utf8mb4_general_ci;

MariaDB [mysql]> update user set password = password('123456') where user='root';
MariaDB [mysql]> flush privileges;

MariaDB [solo]> grant all privileges on *.* to 'root'@'127.0.0.1';
```

安装nginx

```
yum install nginx -y

cp /etc/nginx/nginx.conf{,.bak}
```

首先我们看一下官方的 Docker 搭建代码：

cat solo-start.sh

```
docker run --detach --name solo --network=host \
    --env RUNTIME_DB="MYSQL" \
    --env JDBC_USERNAME="root" \
    --env JDBC_PASSWORD="123456" \
    --env JDBC_DRIVER="com.mysql.cj.jdbc.Driver" \
    --env JDBC_URL="jdbc:mysql://127.0.0.1:3306/solo?useUnicode=yes&characterEncoding=
TF-8&useSSL=false&serverTimezone=UTC" \
    b3log/solo --listen_port=8080 --server_scheme=http --server_host=localhost --server_por
=
```

启动

```
# sh solo-start.sh

[root@prometheus solo]# docker logs -f solo
[INFO ]-[2021-11-30 10:20:30]-[org.b3log.solo.Server:254]: Solo is booting [ver=4.3.1, os=Linu
, isDocker=true, inJar=false, luteAvailable=false, pid=1, runtimeDatabase=MYSQL, runtimeM
de=PRODUCTION, jdbc.username=root, jdbc.URL=jdbc:mysql://127.0.0.1:3306/solo?useUnic
de=yes&characterEncoding=UTF-8&useSSL=false&serverTimezone=UTC]
[INFO ]-[2021-11-30 10:20:31]-[org.b3log.solo.service.InitService:177]: It's your first time setup
Solo, initialize tables in database [MYSQL]
[WARN ]-[2021-11-30 10:20:31]-[org.b3log.solo.service.InitService:150]: Solo has not been init
```

```
alized, please open your browser to init Solo
[INFO ]-[2021-11-30 10:20:34]-[org.b3log.solo.util.Skins:70]: Loaded template from directory [
opt/solo/]

[INFO ]-[2021-11-30 10:30:00]-[org.b3log.solo.processor.OAuthProcessor:205]: Logged in [na
e=mysolo1130, remoteAddr=172.20.10.1] with oauth
```

启动参数说明：

- –listen_port：进程监听端口
- –server_scheme：最终访问协议，如果反代服务启用了 HTTPS 这里也需要改为 https
- –server_host：最终访问域名或公网 IP，不要带端口
- –server_port：最终访问端口，使用浏览器默认的 80 或者 443 的话值留空即可

然后我们可以申请 SSL 证书，直接在面板里申请即可，我申请的是 Let’s Encrypt 证书，并强制 HTPS 访问。

这样我们就成功使用 Docker 部署了 Solo 博客系统并启动了，接下来我们使用 Nginx 配置反向代理。

官方的 Nginx 反向代理代码：

```
upstream backend {
    server localhost:8080; # Solo 监听端口
}

server {
    listen       80;
    server_name  88250.b3log.org; # 博客域名，如我的就为 article.zhyong.cn

    access_log off;

    location / {
        proxy_pass http://backend$request_uri;
        proxy_set_header  Host $host:$server_port;
        proxy_set_header  X-Real-IP  $remote_addr;
        client_max_body_size  10m;
    }
}
```

我的nginx端口是88。因为80端口已经被awx占用了

```
[root@prometheus solo]# cat /etc/nginx/nginx.conf
# For more information on configuration, see:
#   * Official English Documentation: http://nginx.org/en/docs/
#   * Official Russian Documentation: http://nginx.org/ru/docs/

user nginx;
worker_processes auto;
error_log /var/log/nginx/error.log;
pid /run/nginx.pid;
```

```
# Load dynamic modules. See /usr/share/doc/nginx/README.dynamic.
include /usr/share/nginx/modules/*.conf;

events {
    worker_connections 1024;
}

http {
    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';

    access_log  /var/log/nginx/access.log  main;

    sendfile            on;
    tcp_nopush          on;
    tcp_nodelay         on;
    keepalive_timeout   65;
    types_hash_max_size 4096;

    include             /etc/nginx/mime.types;
    default_type        application/octet-stream;

    # Load modular configuration files from the /etc/nginx/conf.d directory.
    # See http://nginx.org/en/docs/ngx_core_module.html#include
    # for more information.
    include /etc/nginx/conf.d/*.conf;

    upstream backend {
        server localhost:8080;
    }

    server {
        listen       88;
        listen       [::]:88;
        server_name  mynew.b3log.org;

        location / {
            proxy_pass http://backend$request_uri;
            proxy_set_header Host $http_host;
            proxy_set_header X-Real-IP $remote_addr;
            client_max_body_size 10m;

        }
        root         /usr/share/nginx/html;

        # Load configuration files for the default server block.
        include /etc/nginx/default.d/*.conf;

        error_page 404 /404.html;
        location = /404.html {
        }

        error_page 500 502 503 504 /50x.html;
```

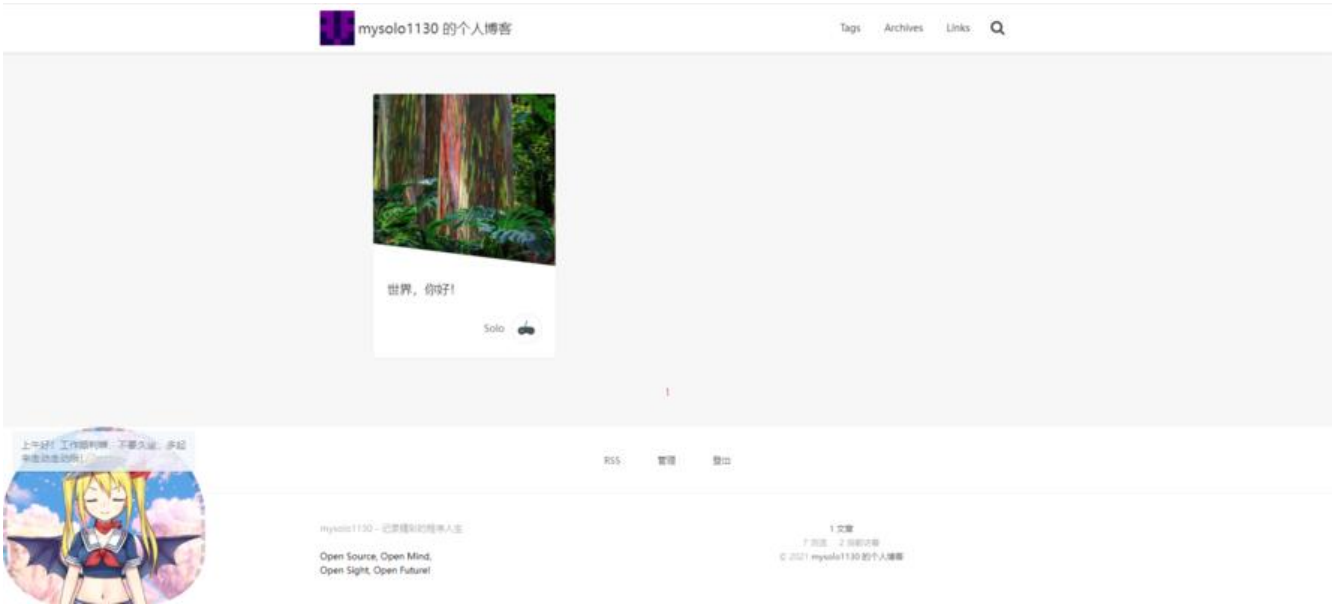
```
        location = /50x.html {
        }
    }

# Settings for a TLS enabled server.
#
#    server {
#        listen       443 ssl http2;
#        listen       [::]:443 ssl http2;
#        server_name  _;
#        root         /usr/share/nginx/html;
#
#        ssl_certificate "/etc/pki/nginx/server.crt";
#        ssl_certificate_key "/etc/pki/nginx/private/server.key";
#        ssl_session_cache shared:SSL:1m;
#        ssl_session_timeout  10m;
#        ssl_ciphers HIGH:!aNULL:!MD5;
#        ssl_prefer_server_ciphers on;
#
#        # Load configuration files for the default server block.
#        include /etc/nginx/default.d/*.conf;
#
#        error_page 404 /404.html;
#            location = /40x.html {
#        }
#
#        error_page 500 502 503 504 /50x.html;
#            location = /50x.html {
#        }
#    }

}
```

启动nginx.

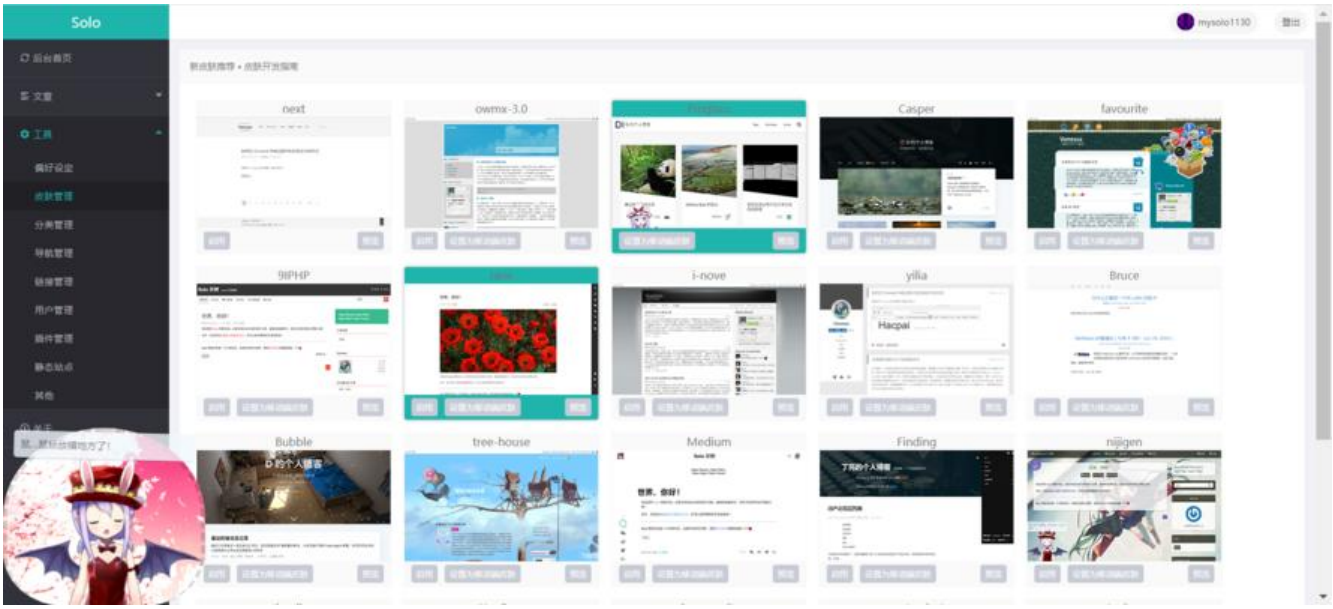
浏览器访问172.20.10.11:88



原文链接：使用 docker 搭建自己的 solo 博客

点击最下面的“管理”

看到可选的皮肤



这样全部完成，现在就打开网站吧～

若出现问题，可以使用 Docker 删除容器再重新部署。

```
docker stop solo
docker rm solo
```

部署过程中遇到的问题： docker 启动代码中的 --server_host 必须使用对应的域名，否则打开网站错！