链滴

从零开始搭建 solo 个人博客系统

作者：vcjmhg
原文链接：https://ld246.com/article/1570344579042
来源网站：链滴

<img src="https://ld246.com/images/img-loading.svg" alt="" data-src="https://b3logfile
com/bing/20180524.jpg?imageView2/1/w/960/h/540/interlace/1/q/100">
自己个人博客系统从搭建运行至今已然过去一月，整个搭建过程可以说是踩坑无数。因此将整个
建过程以及踩过的坑写到此处，希望可以给搭建个人博客系统的朋友提供一些帮助。整个过程教程包
服务器选购、域名申请与备案、安装 solo、nginx 进行反向代理以及 ssl 证书的申请与使用，好了废
说少马上开始。
<h2 id="1-博客系统的搭建流程">1.博客系统的搭建流程</h2>
为了让大家更好的了解整个博客系统搭建的逻辑，因此我将其搭建流程做成一个流程图，希望能
大家以直观的感受；大家在看流程图要注意有些环节是可选。
<img src="https://raw.githubusercontent.com/goWithHappy/pictureRepo/master/img/20191
03163755.png" alt="">
<h2 id="2-服务器选购">2.服务器选购</h2>
首先要有自己的服务器，如果有可以直接跳过这一步。如果没有的话，可以有好多服务器提供商
选：国内的有<a href="https://link.ld246.com/forward?goto=https%3A%2F%2Fwww.aliyun.co
" target="_blank" rel="nofollow ugc">阿里云、<a href="https://link.ld246.com/forward?
oto=https%3A%2F%2Fcloud.tencent.com" target="_blank" rel="nofollow ugc">腾讯云
<a href="https://link.ld246.com/forward?goto=https%3A%2F%2Fwww.huaweicloud.com%2F"
target="_blank" rel="nofollow ugc">华为云、<a href="https://link.ld246.com/forward?g
to=https%3A%2F%2Fcloud.baidu.com%2F" target="_blank" rel="nofollow ugc">百度云
，国外的有 <a href="https://link.ld246.com/forward?goto=https%3A%2F%2Faws.amazon.com
target="_blank" rel="nofollow ugc">AWS、vultr 等
具体选型可以根据自己的业务需求来定。我们这里以阿里云服务器的购买为例：
<h3 id="2-1阿里云学生主机">2.1 阿里云学生主机</h3>
针对在校学生阿里云提供了学生优惠，可以通过云翼计划来购买可以节省一大笔费用。具体的购
流程可以参考官方教程：<a href="https://link.ld246.com/forward?goto=https%3A%2F%2Fyq.al
yun.com%2Farticles%2F717364%3Fspm%3Da2c4e.11155472.0.0.56b13b4b77wCKr" target="_
lank" rel="nofollow ugc">阿里云学生机购买指南说明。对于系统的选择个人推荐 <code>cen
os</code>。
<h3 id="2-2普通云主机">2.2 普通云主机</h3>
普通阿里云主机的购买可能涉及到选配置的问题，但是一般来说入门级的服务器就足够使用，当
如有特殊需要另当他论。此处附上 <a href="https://link.ld246.com/forward?goto=https%3A%2
%2Fwww.aliyun.com%2Fproduct%2Fecs" target="_blank" rel="nofollow ugc">ECS 云服务器
购买地址当然对于系统的选择个人推荐 <code>centos</code>。
<h2 id="3-域名购买与备案-可选-">3.域名购买与备案（可选）</h2>
<h3 id="3-1域名购买">3.1 域名购买</h3>
如果条件允许尽量买个域名，因为通过 ip 访问一方面不方便，另一方面可能会被封禁而且不能
行 SEO 优化，而且现在非热门域名（热门域名包括.com、.cn、.net 等）也并不是很贵如我自己购买
.top 域名三年才 43 块钱。具体购买方式跟购买服务器类似，一般云服务器提供商顺带都提供域名购
服务。这里我们提供一个阿里云的域名购买地址：<a href="https://link.ld246.com/forward?goto
https%3A%2F%2Fwanwang.aliyun.com%2F" target="_blank" rel="nofollow ugc">阿里云域名
买
<h3 id="3-2域名服务器备案">3.2 域名服务器备案</h3>
如果域名或者服务器是国外的一般不需要备案，但是如果说是国内的按照当前国内政策所有域名
在都需备案。一般来说我们可以通过域名购买的服务商域名备案系统来进行备案，当然也可以通过工
与信息化部的 <a href="https://link.ld246.com/forward?goto=http%3A%2F%2Fbeian.miit.gov.
n" target="_blank" rel="nofollow ugc">ICP/IP 地址/域名信息备案管理系统来进行备案，这
建议通过前者，因为官方的备案系统进度是很慢的远远没有第三方的备案系统进度快。这里我们以阿
云的域名备案系统为例，详细说明域名备案的流程：
<img src="https://raw.githubusercontent.com/goWithHappy/pictureRepo/master/img/20191
03202352.png" alt="">
第一步：进入<a href="https://link.ld246.com/forward?goto=https%3A%2
%2Fbeian.aliyun.com%2F" target="_blank" rel="nofollow ugc">域名备案系统
第二步：填写信息和验信资料

<img src="https://raw.githubusercontent.com/goWithHappy/pictureRepo/master/img/20191
03202746.png" alt="">
第三步:等待阿里云初审
这一步大约会耗时一天左右，如果你第二步资料填写均符合要求，那么第二天阿里云客服会给你打来
话，告诉你信息已经通过，将会提交至工信部。如果说资料不符合要求，那么客服会给你打电话并发
件告诉你资料的不规范之处，更改完成之后重新提交，再次进行阿里云初审。这个过程一般耗时两到
天。
第四步：管局审核
这一步是最耗时间的，首先在初审过了之后会有一个工信部短信验证，也就是确定备案手机号是否可
使用。通过之后信息会正式提供给管局。从提供管局信息到得到反馈一般是 7 到 20 天不等（这个就
当地管局心情了），如果管局信息通过不出意外，你会收到一个备案号和密码。此时域名备案便完成
。
<h3 id="3-3域名服务器解析">3.3 域名服务器解析</h3>
这一步是所有过程中最轻松的，登陆域名服务商的的域名管理平台，将域名的解析地址指向所购
服务器的 ip，大约两分钟之后设置便会生效。
<img src="https://raw.githubusercontent.com/goWithHappy/pictureRepo/master/img/20191
03204202.png" alt="">
当然我们可以通过 <code>ping</code> 命令来看解析是否生效：
<img src="https://raw.githubusercontent.com/goWithHappy/pictureRepo/master/img/20191
03204546.png" alt="">
<h2 id="4-solo安装">4.solo 安装</h2>
到这一步算是正式进入正题，根据官网提供的安装方式有两种分为本地使用</strong
、Docker 部署。
<h3 id="4-1-本地试用">4.1 本地试用</h3>
<a href="https://link.ld246.com/forward?goto=https%3A%2F%2Fhacpai.com%2Fforwar
%3Fgoto%3Dhttps%253A%252F%252Fgithub.com%252Fb3log%252Fsolo%252Freleases" tar
et="_blank" rel="nofollow ugc">下载最新的 Solo 包解压，进入解压目录执行以下命令：</
>

Windows: <code>java -cp "WEB-INF/lib/*;WEB-INF/classes" org.b3log.solo.Starter</cod
>
Linux: <code>java -cp "WEB-INF/lib/*:WEB-INF/classes" org.b3log.solo.Starter</code><
r>
注意：官方是不太推荐通过 war 包发布或者源码构建部署，因为这样的部署方式不利于后续新版
发布时的升级和更新。

<h3 id="4-2-Docker部署">4.2 Docker 部署</h3>
<h4 id="第一步-获取最新镜像-">第一步 获取最新镜像：</h4>
<pre><code class="language-bash highlight-chroma">docker pull b3log/solo
</code></pre>
运行结果：
<img src="https://raw.githubusercontent.com/goWithHappy/pictureRepo/master/img/20191
06095735.png" alt="">
<h4 id="第二步-安装MYSQL">第二步 安装 MYSQL</h4>
我们既可以直接在服务器上直接安装也可以通过 docker 方式在 docker 容器中安装
方式一：docker 安装
<pre><code class="language-bash highlight-chroma"># 安装m
sql:5.6,直接docker run 他会自动去官方镜想下载
MYSQL_ROOT_PASSWORD=[的数据库密码，此处写的是123456<
span>
docker run --name mysql -p 3306:3306 -e MYSQL_ROOT_PASS
ORD=123456</spa
> -d mysql:5.6
docker安装的mysql默认允许远程连接，可以使用Navicat等软件连

数据库
进入容器mysql
docker exec -it mysql bash

进入数据库 p后面跟你的密码

mysql -uroot -p123456

创建数据库(数据库名:solo;字符集utf8mb4;排序规则utf8mb4_general_ci)

create database solo DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;

出现Query OK, 1 row affected (0.00 sec)表示成功

#退出数据库

exit

#退出容器

exit

</code></pre>

运行结果如下：
<img src="https://raw.githubusercontent.com/goWithHappy/pictureRepo/master/img/20191
06100037.png" alt="">
<img src="https://raw.githubusercontent.com/goWithHappy/pictureRepo/master/img/20191
06100938.png" alt="">
方式二：物理机上直接安装
由于配置稍显繁琐，可以参考下边这篇博客:
<a href="https://link.ld246.com/forward?goto=ww.cnblogs.com%2Fcaoxb%2Fp%2F9405323.
tml" target="_blank" rel="nofollow ugc">Centos 安装 mysql(YUM 源方式)
<h4 id="第三步-安装solo">第三步 安装 solo</h4>
运行如下命令：
<pre><code class="language-bash highlight-chroma">docker run --detach --name solo --ne
work=host \
--env RUNTIME_D
="MYSQL"
\
--env JDBC_USER
AME="root"</span
 \
--env JDBC_PASS
ORD="123456"</s
an> \
--env JDBC_DRIVE
="com.mysql.cj.jdbc
Driver" \
--env JDBC_URL</
pan>="jdbc:mysql://127.0.0.
:3306/solo?useUnicode=yes&characterEncoding=UTF-8&useSSL=false&server
imezone=UTC" \
b3log/solo --listen_port<span class="highlight
o">=8080 --server_scheme<span class="highli
ht-o">=http --server_host=www.vcjmhg.top

</code></pre>

<code>--detach:</code> 这个选项告诉 Docker 在启动后将程序与控制台分离,使其进入“台”运行。
<code>--name solo:</code> <code>solo</code> 是容器的名字，也可以改成自己喜欢的字如 mysolo，这个无所谓
<code>RUNTIME_DB="MYSQL"</code>：指明我们此处使用的数据库为 <code>MYSQL</code>,如果使用,<code>H2 Database</code>,将 <code>MYSQL</code> 改成 <code>org.h2.Driver</code> 即可
<code>JDBC_USERNAME="root"：</code> 指明 <code>MYSQL</code> 数据连接时使的用户名，默认都是 root
<code>JDBC_PASSWORD="123456"</code>： 指明 MYSQL 数据库连接时用户密码，使用注意将 <code>123456</code> 替换成自己在上一步所设置的密码
<code>env JDBC_DRIVER="com.mysql.cj.jdbc.Driver"</code>：数据库连接驱动包，如果用,<code>H2 Database</code>,将 <code>om.mysql.cj.jdbc.Driver</code> 改成 <code>H2</code> 即可
<code>--server_host=www.vcjmhg.top</code>： 个人域名，如果没有可设置为自己的服务 ip
<code>--env JDBC_URL=...</code>:<img src="https://raw.githubusercontent.com/goWithHappy/pictureRepo/master/img/20191005213732.png" alt="">
<code>--listen_port=8080</code>:指明 solo 监听的端口此处使用的是 <code>8080</code>,如果不想配置 nginx 此处可以换成 80

命令运行结果：
<img src="https://raw.githubusercontent.com/goWithHappy/pictureRepo/master/img/20191006101445.png" alt="">
命令执行完成之后没有报错的话，通过 <code>docker ps</code> 查看当前当前容器列表中否有名字叫 <code>solo</code> 的容器，如果有证明启动成功了，此时可以通过 <code>个人域名 ip+:8080</code> 来进行访问,类似 <code> http://192.168.217.132:8080</code>,如果不想配置 ginx 可以将 <code>8080</code> 换成 <code>80</code>，可以直接通过域名/ip 来直接进行问，类似 vcjmhg 的博客--https://vcjmhg.top。不出意外会出现下界面（如果出现不能访问的情况考虑是否是防火墙配置有问题，查看是否开发 8080 或者 80 端口：
<img src="https://raw.githubusercontent.com/goWithHappy/pictureRepo/master/img/20191006102353.png" alt="">
由于后边我们需要配置 <code>nginx</code> 进行反向代理以及配置 <code>ssl</code> 证来实现 <code>https</code> 方式访问，因此在看到 <code>solo</code> 启动正常之后，此处建的 <code>solo</code> 镜像需要删除，等配置完 <code>nginx</code> 之后重新在创建一个
删除 solo 容器直接执行下边命令
<pre><code class="language-bash highlight-chroma">docker kil --name solo
docker rm --name solo
</code></pre>
命令执行结果如下：
<img src="https://raw.githubusercontent.com/goWithHappy/pictureRepo/master/img/20191006103710.png" alt="">
<h2 id="5-安装nginx-可选-">5.安装 nginx（可选）</h2>
安装之前为了后续配置 nginx 方便，我们需要在本地创建几个文件，用来挂载 nginx 的配置文件/p>
<pre><code class="language-bash highlight-chroma"># 切换服务器根目录
cd /

创建主目录
mkdir dockerData
创建文件
mkdir dockerData/nginx dockerData/nginx/conf dockerData/nginx/logs dockerData/nginx/ww dockerData/nginx/ssl
</code></pre>
上边的文件目录名称可以任意，此处我使用 <code>dockerDate</code>

<code>dockerData/nginx</code> 用于存放 <code>docker</code> 下 <code>nginx</cde> 自定义文件
<code>dockerData/nginx/conf</code> 存放 <code>nginx</code> 配置文件
<code>dockerData/nginx/log</code> 存放 <code>nginx</code> 日志文件
<code>dockerData/nginx/www</code>:存放 <code>nginx</code> 访问的资源文件
<code>dockerData/nginx/ssl</code> 存放 <code>ssl</code> 证书
命令执行结果如下：


启动 <code>nginx</code>
<pre><code class="language-bash highlight-chroma">docker run --name nginx -p 80:80 -d ginx
</code></pre>
命令执行结果如下：

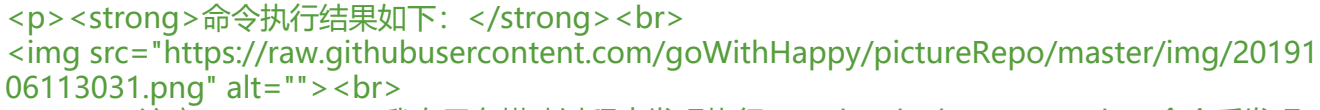

如果没有备案，80 端口可能是禁止访问的，因此可以可以将上边的 80:80 换成 8080:80。命令行完成之后，没有报错的话可以通过 docker ps 来看 nginx 是否正常运行，在运行的情况下访问的名加上端口号查看是否正常安装，如果使用的 80 端口默认可以省略。出现如下界面表示安装成功。<r>


导出配置文件：
<pre><code class="language-bash highlight-chroma">docker cp nginx:/etc/nginx/nginx.conf /dockerData/nginx/conf/nginx.conf #导出配置文件nginx.conf</san>
docker cp nginx:/etc/nginx/conf.d /dockerData/nginx/conf/conf.d <span class="highlight-c1>#导出conf.d
cd /dockerData/nginx/
ls conf/ #查看配置文件是否导出成功
docker stop nginx #删除刚才创建的nginx容器
docker rm nginx
</code></pre>
命令执行结果：


重新创建一个 <code>nginx</code> 容器，挂载刚才本地导出的配置文件，便于后续更改 <coe>nginx</code> 的配置信息
<pre><code class="language-bash highlight-chroma">docker run -d -p 80:80 --name nginx \
-v /dockerData/nginx/conf/nginx.conf:/etc/nginx/nginx.conf \
-v /dockerData/nginx/conf/conf.d:/etc/nginx/cnf.d \
-v /dockerData/nginx/www:/usr/share/nginx/h

ml \
-v /dockerData/nginx/logs:/var/log/nginx nginx
</code></pre>

<code>-v /dockerData/nginx/conf/nginx.conf:/etc/nginx/nginx.conf </code>： 挂载配文件 <code>nginx.conf</code>
<code>-v /dockerData/nginx/conf/conf.d:/etc/nginx/conf.d </code>： 挂载配置文件 <code>default.conf</code>
<code>-v /dockerData/nginx/www:/usr/share/nginx/html</code>： 挂载项目文件
<code>-v /dockerData/nginx/logs:/var/log/nginx </code>： 挂载配置文件

命令执行结果如下：

注意： 我自己在搭建过程中发现执行 <code>docker ps</code> 命令后发现 ginx 并没有被启动,使用命令 <code>docker logs nginx</code> 发现挂载文件时权限不足，果断创建 nginx 容器是加上 <code>--privileged=true</code> 参数（如果没有出现该问题可不加上述数）。

即执行如下命令
<pre><code class="language-bash highlight-chroma">docker run -d -p 80:80 --name nginx -privileged true \
-v /dockerData/nginx/conf/nginx.conf:/etc/nginx/nginx.conf \·
-v /dockerData/nginx/conf/conf.d:/etc/nginx/conf.d \
-v /dockerData/nginx/www:/usr/share/nginx/html \
-v /dockerData/nginx/logs:/var/log/nginx nginx
</code></pre>
执行 <code>docker ps -a</code> 此时容器运行正常

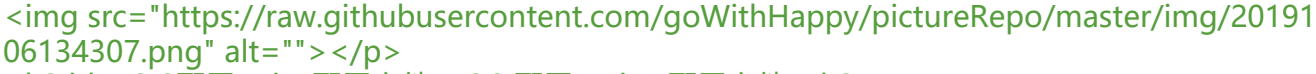
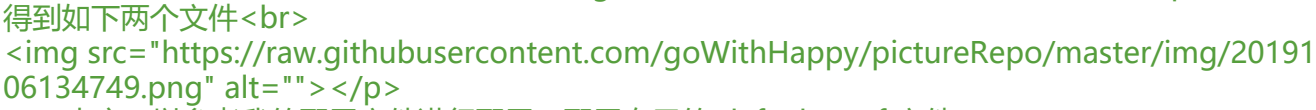
容器创建完成之后重新访问在浏览器访问可能会出现如下界面：

此时你可以在 <code>www</code> 目录下创建一个 <code>html</code> 文件，也可以先不用，等后续配置完成之后自然会消失。
<h2 id="6--配置ssl证书-可选-">6. 配置 ssl 证书（可选）</h2>
从 http 升级到 https 只需要在 nginx 中配置一个证书即可，一般性的 ssl 证书是可以免费申请
<h3 id="6-1证书选购">6.1 证书选购</h3>
阿里云或者腾讯云都提供证书申请服务，这里以阿里云的证书申请为例：
第一步: 进入阿里云的证书购买地址 https://www.aliyun.com/product/cas

第二步： 进入购买页面，选择 <code>免费型DV SSL</code>

第三步: 支付


第四步： 进入控制台，进行证书下载


<h3 id="6-2配置nginx配置文件">6.2 配置 nginx 配置文件</h3>
下载之后会得到一个名字类似于 <code>2793667_www.vcjmhg.top_nginx.zip</code> 的文，将其上传到服务器 <code>/dockerData/nginx/ssl</code> 目录中,解压后（通过 unzip 命令）得到如下两个文件

大家可以参考我的配置文件进行配置，配置自己的 default.conf 文件。

```
<pre><code class="language-nginx highlight-chroma"><span class="highlight-k">server</
pan> <span class="highlight-p">{</span>
    <span class="highlight-kn">listen</span>       <span class="highlight-mi">443</span><s
an class="highlight-p">;</span>
    <span class="highlight-kn">server_name</span>  <span class="highlight-s">localhost</
pan><span class="highlight-p">;</span>
    <span class="highlight-kn">ssl</span> <span class="highlight-no">on</span><span cla
s="highlight-p">;</span>
    <span class="highlight-kn">ssl_certificate</span> <span class="highlight-s">/ssl/279366
_www.vcjmhg.top.pem</span><span class="highlight-p">;</span>  <span class="highlight-
1"># ssl 证书目录
</span><span class="highlight-c1"></span>    <span class="highlight-kn">ssl_certificate_k
y</span> <span class="highlight-s">/ssl/2793667_www.vcjmhg.top.key</span><span class
"highlight-p">;</span>
    <span class="highlight-kn">ssl_session_timeout</span> <span class="highlight-mi">5m<
span><span class="highlight-p">;</span>
    <span class="highlight-kn">ssl_protocols</span> <span class="highlight-s">TLSv1</spa
> <span class="highlight-s">TLSv1.1</span> <span class="highlight-s">TLSv1.2</span><s
an class="highlight-p">;</span>
    <span class="highlight-kn">ssl_ciphers</span> <span class="highlight-s">ECDHE-RSA-A
S128-GCM-SHA256:HIGH:!aNULL:!MD5:!RC4:!DHE</span><span class="highlight-p">;</spa
>
    <span class="highlight-kn">ssl_prefer_server_ciphers</span> <span class="highlight-no"
on</span><span class="highlight-p">;</span>


<span class="highlight-c1">#charset koi8-r;

</span><span class="highlight-c1"></span>
span class="highlight-c1">#access_log  /var/log/nginx/host.access.log  main;

</span><span class="highlight-c1"></span>

<span class="highlight-kn">location</span> <span class="highlight-s">/</span> <span cla
s="highlight-p">{</span>

<span class="highlight-c1">#root   /usr/share/nginx/html;

</span><span class="highlight-c1"></span>
  <span class="highlight-c1"># index  index.html index.htm;

</span><span class="highlight-c1"></span>
  <span class="highlight-c1"># 官方博客上此处用的是域名，但配置时发现不好使，所以我用的
服务器ip
```

```
</span><span class="highlight-c1"></span>
  <span class="highlight-kn">proxy_pass</span> <span class="highlight-s">http://39.105.6
.192</span><span class="highlight-p">          ;</span>
<span class="highlight-p">}</span>


<span class="highlight-c1">#error_page  404          /404.html;

</span><span class="highlight-c1"></span>
<span class="highlight-c1"># redirect server error pages to the static page /50x.html
</span><span class="highlight-c1"></span>
span class="highlight-c1">#
</span><span class="highlight-c1"></span>
span class="highlight-kn">error_page</span>  <span class="highlight-mi">500</span> <s
an class="highlight-mi">502</span> <span class="highlight-mi">503</span> <span class=
highlight-mi">504</span>  <span class="highlight-s">/50x.html</span><span class="highl
ght-p">;</span>
<span class="highlight-kn">location</span> <span class="highlight-p">=</span> <span cl
ss="highlight-s">/50x.html</span> <span class="highlight-p">{</span>
<span class="highlight-kn">root</span>   <span class="highlight-s">/usr/share/nginx/html
/span><span class="highlight-p">          ;</span>
<span class="highlight-p">}</span>
<span class="highlight-p">}</span>
<span class="highlight-k">server{</span>
<span class="highlight-s">listen</span> <span class="highlight-mi">80</span><span clas
="highlight-p">;</span>
<span class="highlight-k">server_name</span> <span class="highlight-s">www.vcjmhg.top
/span><span class="highlight-p">          ;</span>
<span class="highlight-k">rewrite</span> <span class="highlight-s">^(.*)</span> <span cl
ss="highlight-s">https://</span><span class="highlight-nv">          $host$1</span> <sp
n class="highlight-s">permanent</span><span class="highlight-p">          ;</span>
<span class="highlight-k">}</span>

</code></pre>

<p><strong>注意</strong>:上边的配置文件只是参考，要根据自己的服务器做出相应更改。<br>
由于我们现在用的 <code>nginx</code> 容器并未监听 <code>443</code> 端口，所以需要删
现在的容器，重新启动一个新的 <code>nginx</code> 容器</p>
<pre><code class="language-bash highlight-chroma"><span class="highlight-c1">#先删除
来的nginx容器</span>
docker stop nginx<span class="highlight-p">;</span>
docker rm nginx<span class="highlight-p">;</span>
<span class="highlight-c1">#创建新的nginx容器</span>
docker run -d -p 80:80 -p 443:443 --name nginxs <span class="highlight-se">\
</span><span class="highlight-se"></span>-v /dockerData/nginx/conf/nginx.conf:/etc/ngi
x/nginx.conf <span class="highlight-se">\
</span><span class="highlight-se"></span>-v /dockerData/nginx/conf/conf.d:/etc/nginx/c
nf.d <span class="highlight-se">\
```

```
</span><span class="highlight-se"></span>-v /dockerData/nginx/ssl:/ssl/ <span class="hig
light-se">\
</span><span class="highlight-se"></span>-v /dockerData/nginx/www:/usr/share/nginx/h
ml <span class="highlight-se">\
</span><span class="highlight-se"></span>-v /dockerData/nginx/logs:/var/log/nginx nginx

<span class="highlight-c1">#创建新的solo容器并映射到8080端口，用上边的nginx进行反向代理
/span>
docker run --detach --name solo --network<span class="highlight-o">=</span>host <span
lass="highlight-se">
</span><span class="highlight-se"></span>                                    -
env <span class="highlight-nv">RUNTIME_DB</span><span class="highlight-o">          =
/span><span class="highlight-s2">          "MYSQL"</span> <span class="highlight-se">
</span><span class="highlight-se"></span>                                    -
env <span class="highlight-nv">JDBC_USERNAME</span><span class="highlight-o">          =
/span><span class="highlight-s2">          "root"</span> <span class="highlight-se">
</span><span class="highlight-se"></span>                                    -
env <span class="highlight-nv">JDBC_PASSWORD</span><span class="highlight-o">          =
/span><span class="highlight-s2">          "123456"</span> <span class="highlight-se">
</span><span class="highlight-se"></span>                                    -
env <span class="highlight-nv">JDBC_DRIVER</span><span class="highlight-o">          =
/span><span class="highlight-s2">          "com.mysql.cj.jdbc.Driver"</span> <span class="h
ghlight-se">
</span><span class="highlight-se"></span>                                    -
env <span class="highlight-nv">JDBC_URL</span><span class="highlight-o">          =</s
an><span class="highlight-s2">      "jdbc:mysql://192.168.217.132:3306/solo?useUnicode=yes
amp;characterEncoding=UTF-8&amp;useSSL=false
amp;serverTimezone=UTC"       </span> <span class="highlight-se">
</span><span class="highlight-se"></span>                                    b
log/solo --listen_port<span class="highlight-o">=</span><span class="highlight-m">
080</span> --server_scheme<span class="highlight-o">=</span>https --server_host<span c
ass="highlight-o">=</span>www.vcjmhg.top --server_port<span class="highlight-o">=</s
an>
</code></pre>

<ul>
<li><code>--server_scheme=http</code> 换成 <code>--server_scheme=https</code> 即
</li>
<li><code>--server_port</code>：最终访问端口，使用浏览器默认的 <code>80</code> 或者
code>443</code> 的话值留空即可</li>
<li>如果出现权限不足问题，启动时加上 <code>-privileged true</code> 参数即可</li>
</ul>
<p>重启 nginx，<code>docker restart nginx</code>,然后用浏览器访问 <code>https://域名</
ode> 类似于 <a href="https://link.ld246.com/forward?goto=https%3A%2F%2Fwww.vcjmhg.t
p" target="_blank" rel="nofollow ugc">https://www.vcjmhg.top</a>,登陆 <code>github</c
de> 账户后出现如下界面<br>
<img src="https://raw.githubusercontent.com/goWithHappy/pictureRepo/master/img/20191
06142257.png" alt=""></p>
<h2 id="后记">后记</h2>
<p>首先官方已经给了一些<a href="https://link.ld246.com/forward?goto=https%3A%2F%2Fha
```

pai.com%2Farticle%2F1492881378588" target="_blank" rel="nofollow ugc">安装的教程一位叫 <code>墨殇</code> 的博主也已经给了特别详细的安装教程(地址，点这里),但是我自己在配置过程中出现了好多问，因此我在这里写下来这篇博客，希望能给他人提供一些帮助。当然可能个人水平有限，中间难免会现一些错误，如若发现恳请指出，不胜赐教。如果按照本教程在配置过程中遇到什么问题，欢迎在博下边留言，我若看到的话一定第一时间回复，谢谢！ </p>