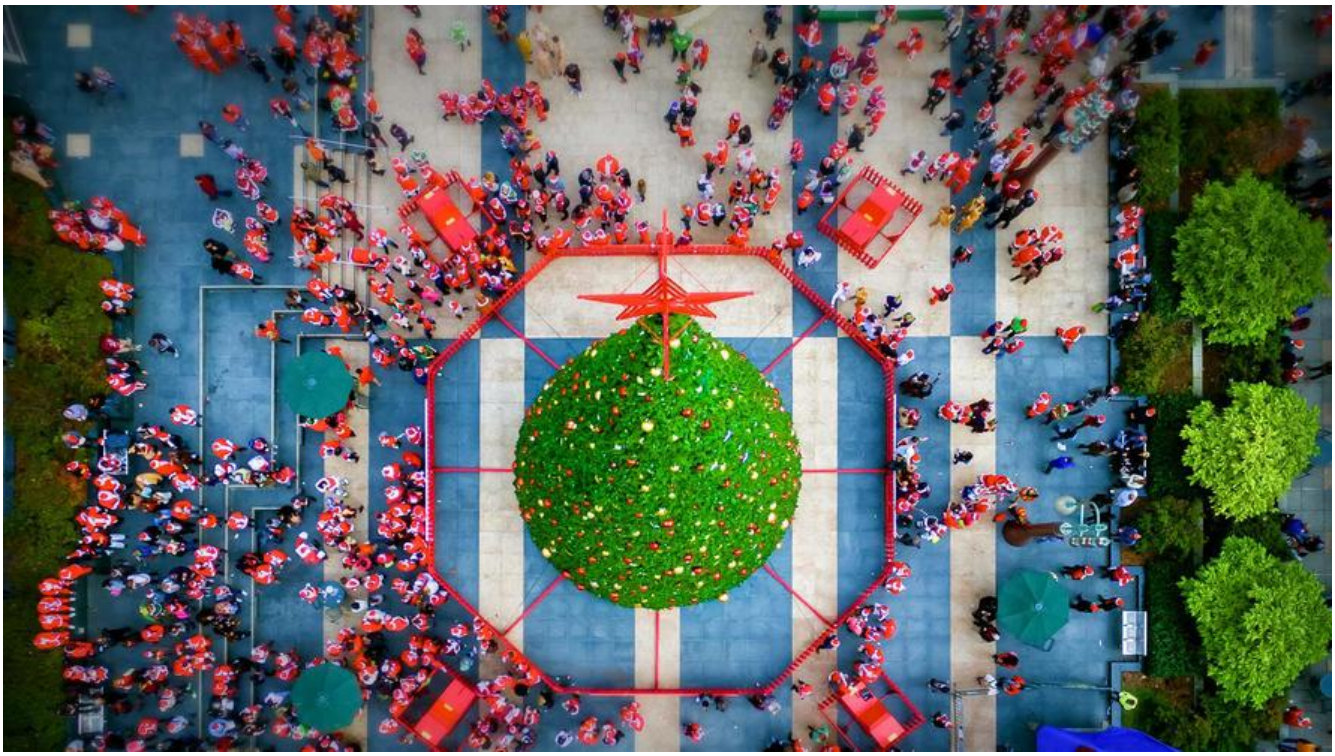
链滴

从零开始搭建开源博客 solo

作者：Dawei1408
原文链接：https://ld246.com/article/1553755313169
来源网站：链滴



简介

Solo是一款小而美的开源博客系统，专为程序员设计。Solo 有着非常活跃的社区，文章自动推送到区后可以让很多人看到，产生丰富的交流互动。

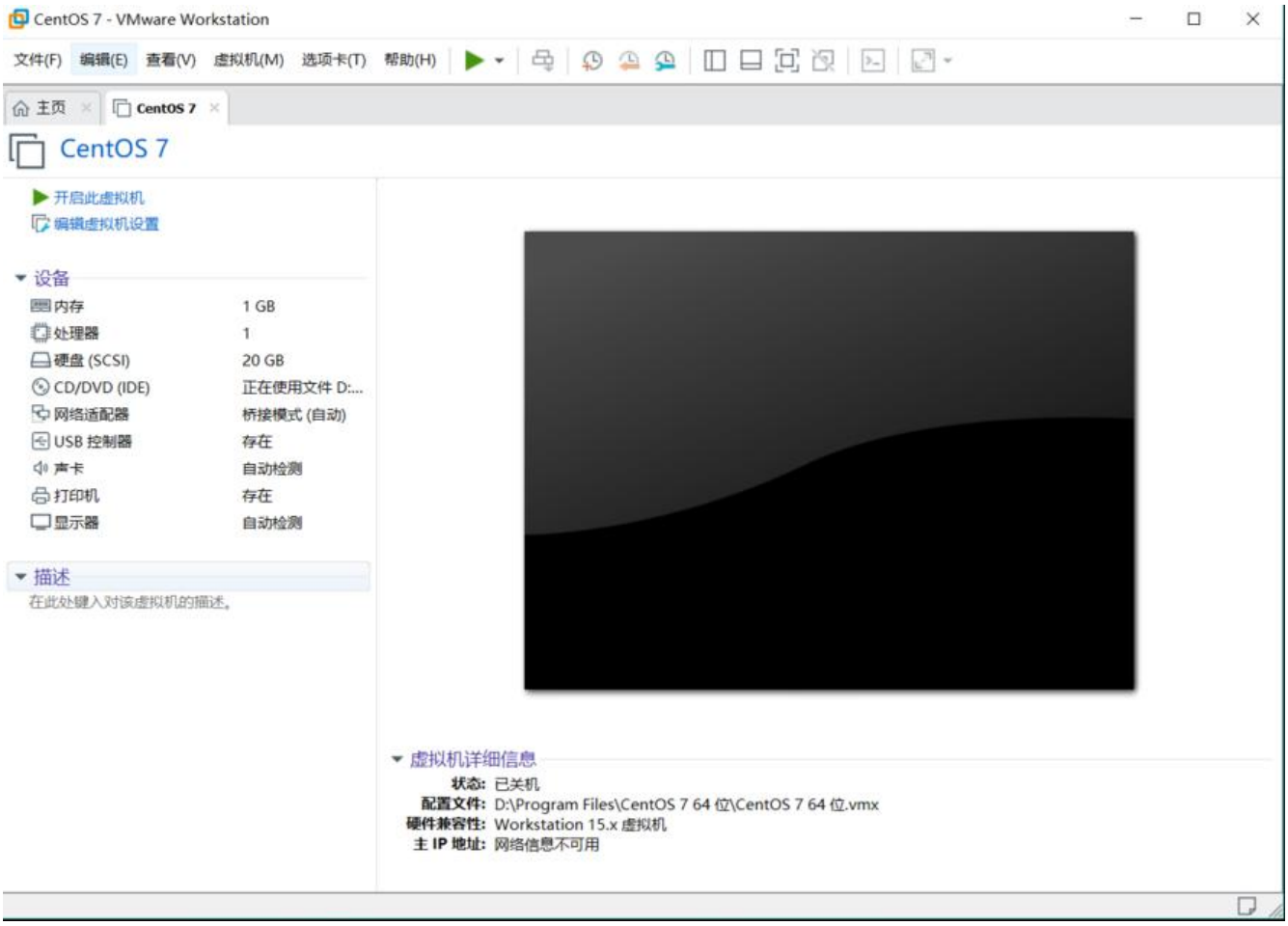
相关配置：

名称	说明
服务器	推荐云服务器(阿里云、腾讯云等)
操作系统	Centos7
JDK版本	1.8.0_201
容器	Tomcat9
Solo版本	V3.2.0
反向代理	Nginx

准备环境

1. 服务器和系统

在阿里云或腾讯云申请一台服务器，安装centos7系统。(我的服务器已经搭建好，所以新建了一个虚机，配置如下)



2.安装配置JDk

1.检查并卸载centos7系统上的openjdk

1. # rpm -qa | grep java

● 卸载掉系统自带的jdk(除去 **.noarch** 结尾)

1. # rpm -e --nodeps + 文件名

例如卸载第一个openjdk命令为:

1. # rpm -e --nodeps java-1.8.0-openjdk-headless-1.8.0.191.b12-1.el7_6.x86_64

2. 下载jdk(最新版本是1.8_201)

● jdk官方下载链接: https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

Java SE开发工具包8u201

您必须接受Java SE的Oracle二进制代码许可协议才能下载此软件。

接受许可协议 不接受许可协议

产品/文件说明	文件大小	下载
Linux ARM 32 Hard Float ABI	72.98 MB	JDK-8u201-Linux的ARM32，VFP，hflt.tar.gz
Linux ARM 64 Hard Float ABI	69.92 MB	JDK-8u201-Linux的arm64-VFP，hflt.tar.gz
Linux x86	170.98 MB	JDK-8u201 Linux的-i586.rpm
Linux x86	185.77 MB	JDK-8u201-Linux的i586.tar.gz
Linux x64	168.05 MB	JDK-8u201-Linux的x64.rpm
Linux x64	182.93 MB	JDK-8u201-Linux的x64.tar.gz
Mac OS X x64	245.92 MB	JDK-8u201，MacOSX的，x64.dmg
Solaris SPARC 64位（SVR4包）	125.33 MB	JDK-8u201-Solaris的sparcv9.tar.Z
Solaris SPARC 64位	88.31 MB	JDK-8u201-Solaris的sparcv9.tar.gz
Solaris x64（SVR4包）	133.99 MB	JDK-8u201-Solaris的x64.tar.Z
Solaris x64	92.16 MB	JDK-8u201-Solaris的x64.tar.gz
Windows x86	197.66 MB	JDK-8u201窗口-i586.exe
Windows x64	207.46 MB	JDK-8u201窗口-x64.exe程序

● 点击接受许可协议，然后下载对应版本

● 上传jdk1.8_201到centos系统 /usr/local/目录下

3.新建 /usr/local/java目录，将jdk解压在此目录

进入/usr/local/目录下
1. # cd /usr/local/
新建java目录
2. # mkdir /usr/local/java
将jdk移至新建的java目录下
3. # mv jdk-8u201-linux-x64.tar.gz /usr/local/java/
进入目录下
4. # cd java
解压jdk
5. # tar -zxvf jdk-8u201-linux-x64.tar.gz

4.配置java环境

1. # vim /etc/profile

按"i"进入插入模式，在配置文件最上方插入如下:

export JAVA_HOME=/usr/local/java/jdk1.8.0_201
export CLASSPATH=.:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar
export PATH=$PATH:$JAVA_HOME/bin

```
root@localhost:/usr/local/java
文件(F)  编辑(E)  查看(V)  搜索(S)  终端(T)  帮助(H)
# /etc/profile

# System wide environment and startup programs, for login setup
# Functions and aliases go in /etc/bashrc

# It's NOT a good idea to change this file unless you know what you
# are doing. It's much better to create a custom.sh shell script in
# /etc/profile.d/ to make custom changes to your environment, as this
# will prevent the need for merging in future updates.

export JAVA_HOME=/usr/local/java/jdk1.8.0_201
export CLASSPATH=.:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar
export PATH=$PATH:$JAVA_HOME/bin

pathmunge () {
    case ":${PATH}:" in
        *:"$1":*)
            ;;
        *)
            if [ "$2" = "after" ] ; then
                PATH=$PATH:$1
            else
                PATH=$1:$PATH
:wq
```

● 编辑后按"Esc"退出编辑，再输入":wq"保存修改，然后输入以下命令，配置生效

1. # source /etc/profile

5.检测jdk是否配置成功

1. # java

```
root@localhost:~
文件(F)  编辑(E)  查看(V)  搜索(S)  终端(T)  帮助(H)
    -? -help        输出此帮助消息
    -X              输出非标准选项的帮助
    -ea[:<packagename>...|:<classname>]
    -enableassertions[:<packagename>...|:<classname>]
                    按指定的粒度启用断言
    -da[:<packagename>...|:<classname>]
    -disableassertions[:<packagename>...|:<classname>]
                    禁用具有指定粒度的断言
    -esa | -enablesystemassertions
                    启用系统断言
    -dsa | -disablesystemassertions
                    禁用系统断言
    -agentlib:<libname>[=<选项>]
                    加载本机代理库 <libname>, 例如 -agentlib:hprof
                    另请参阅 -agentlib:jdwp=help 和 -agentlib:hprof=help
    -agentpath:<pathname>[=<选项>]
                    按完整路径名加载本机代理库
    -javaagent:<jarpath>[=<选项>]
                    加载 Java 编程语言代理, 请参阅 java.lang.instrument
    -splash:<imagepath>
                    使用指定的图像显示启动屏幕
有关详细信息, 请参阅 http://www.oracle.com/technetwork/java/javase/documentation
/index.html。
[root@localhost ~]#
```

1. # java -version

```
root@localhost:~
文件(F)  编辑(E)  查看(V)  搜索(S)  终端(T)  帮助(H)
[root@localhost ~]# java -version
java version "1.8.0_201"
Java(TM) SE Runtime Environment (build 1.8.0_201-b09)
Java HotSpot(TM) 64-Bit Server VM (build 25.201-b09, mixed mode)
[root@localhost ~]#
```

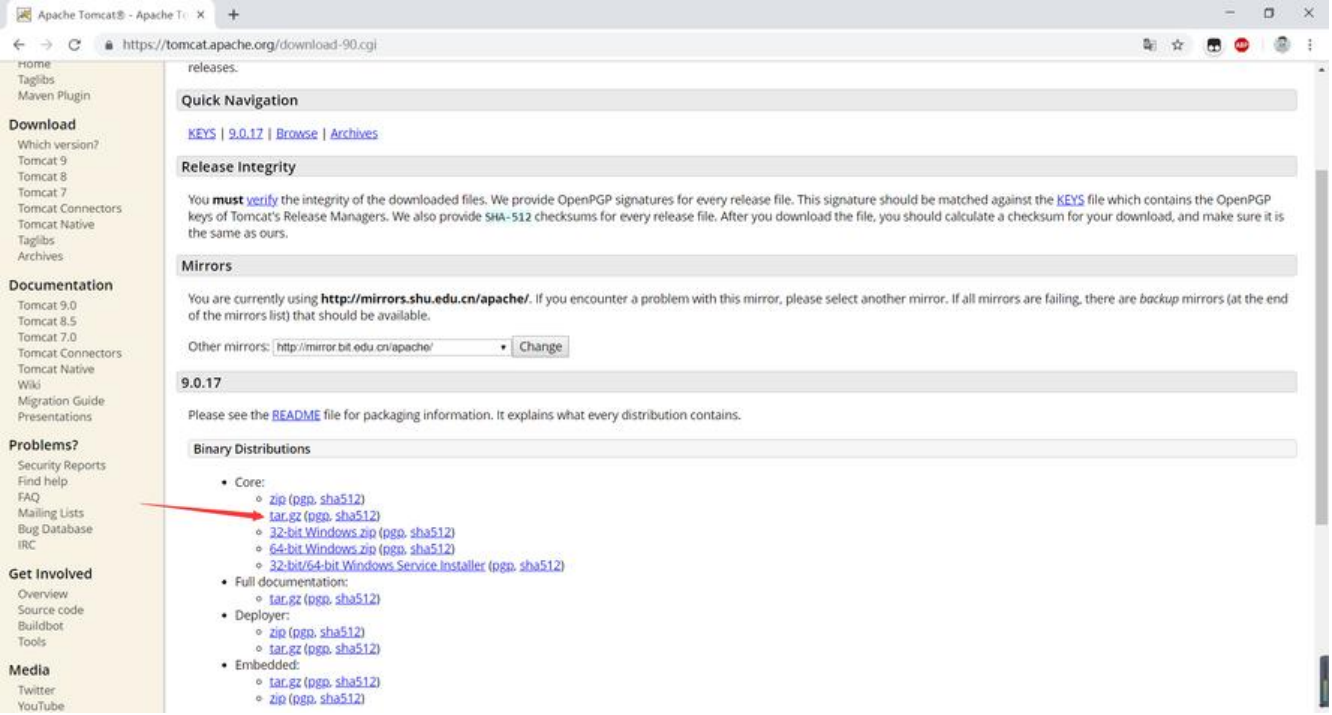
到此jdk安装配置成功

2.安装Tomcat9

1.到官网下载安装包

tomcat9下载链接:https://tomcat.apache.org/download-90.cgi

这里选择适合Linux的安装包，下载到本地后上传到centos服务器，或者直接通过wget命令下载



```
1. # cd /usr/local/
2. # mkdir tomcat/
3. # cd tomcat/
4. # wget http://mirror.bit.edu.cn/apache/tomcat/tomcat-9/v9.0.17/bin/apache-tomcat-9.0.17.tar.gz
```

以上操作就将tomcat9安装包文件apache-tomcat-9.0.17.tar.gz下载到/usr/local/tomcat目录下了

2.安装tomcat9

```
1. # cd /usr/local/tomcat/
2. # tar -zxvf apache-tomcat-9.0.17.tar.gz
```

安装包会被解压到/usr/local/tomcat/apache-tomcat-9.0.17,将目录重命名为tomcat9

```
1. # mv /usr/local/tomcat/apache-tomcat-9.0.17 /usr/local/tomcat/tomcat9
```

3.开启tomcat服务

```
1. # cd /usr/local/tomcat/tomcat/bin/
2. # ./startup.sh
```

```
[root@localhost ~]# cd /usr/local/tomcat/tomcat9/bin/
[root@localhost bin]# ./startup.sh
Using CATALINA_BASE:   /usr/local/tomcat/tomcat9
Using CATALINA_HOME:   /usr/local/tomcat/tomcat9
Using CATALINA_TMPDIR: /usr/local/tomcat/tomcat9/temp
Using JRE_HOME:        /usr/local/java/jdk1.8.0_201
Using CLASSPATH:       /usr/local/tomcat/tomcat9/bin/bootstrap.jar:/usr/local/tomcat/to
mcat9/bin/tomcat-juli.jar
Tomcat started.
[root@localhost bin]#
```

服务开启成功

tomcat服务关闭命令

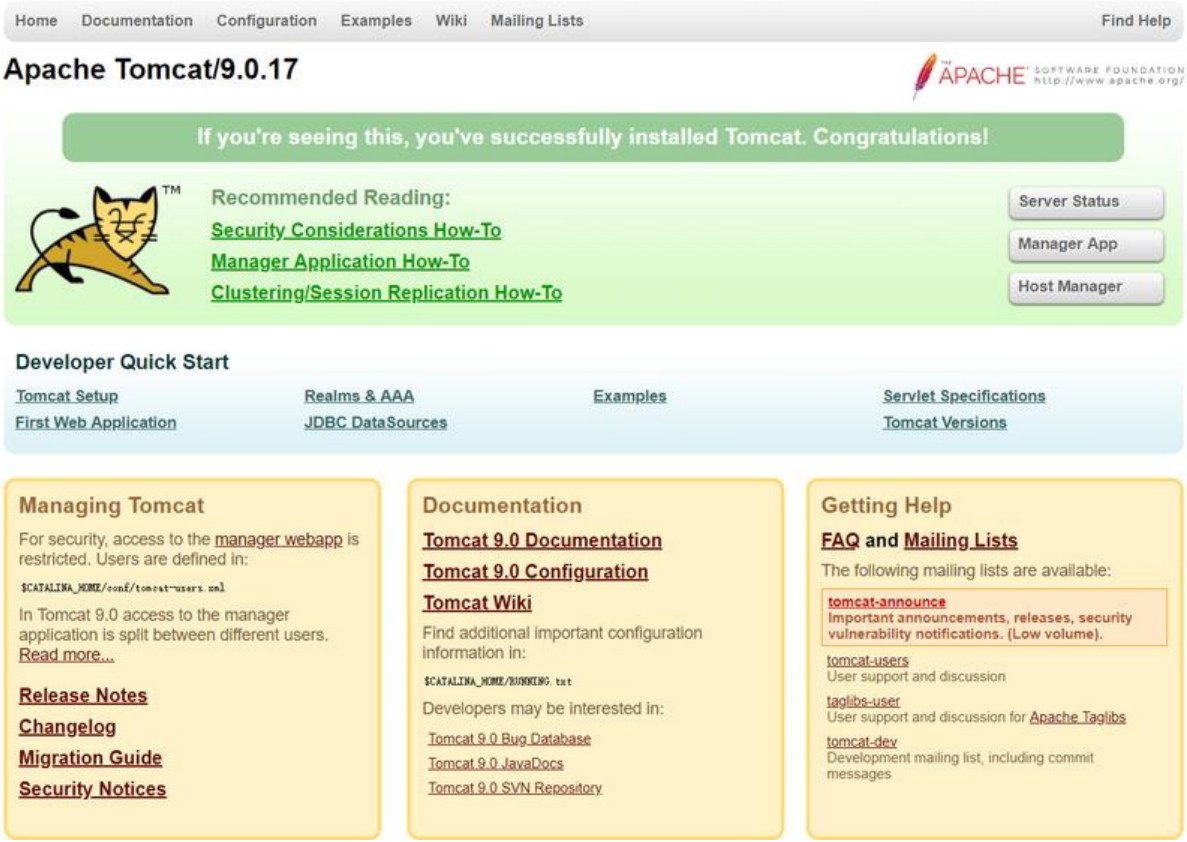
```
1. # ./shutdown.sh
```

4.验证tomcat是否安装成功

用本地浏览器访问http://服务器IP:8080/

如果能出现熟悉的tomcat主页，就表示安装成功了，请确保服务器防火墙已关闭

```
linux系统关闭防火墙关命令
1. # systemctl stop firewaldd.service
linux系统开机禁用防火墙关命令
2. # systemctl disable firewalld
```



tomcat服务默认开机不自启，系统重启后需手动开启服务。或自行百度tomcat服务开机自启教程

3.部署solo博客

1.下载源码包

solo是开源博客系统，源码包可以到github下载

项目链接:https://github.com/b3log/solo

这里我们使用solo-v3.2.0,可以下载到本地后上传到centos服务器，或者直接通过wget命令下载

```
1. # cd /usr/local/tomcat/tomcat9/webapps/
2. # wget https://github.com/b3log/solo/releases/download/v3.2.0/solo-v3.2.0.war
```

./webapps/ROOT是tomcat容器的网站根目录，也就是上面我们看到的tomcat主页文件的目录

将war包下载或上传到wenapps目录后tomcat会自动解压

```
[root@localhost webapps]# ls
docs  examples  host-manager  manager  ROOT  solo-v3.2.0  solo-v3.2.0.war
[root@localhost webapps]#
```

2.替换文件

在tomcat9/webapps目录下有solo-v3.2.0这个文件夹就说明下载成功，接下在我们要把tomcat的访目录换为solo，最简单的办法就是将原来的ROOT文件夹删除或者重命名，再将solo-v3.2.0重命名为OOT，

这样我们通过http://服务器IP:8080 就能访问solo了。

```
1. # mv /usr/local/tomcat/tomcat9/webapps/ROOT /usr/local/tomcat/tomcat9/webapps/ROT-old
```

2. # mv /usr/local/tomcat/tomcat9/webapps/solo-v3.2.0 /usr/local/tomcat/tomcat9/webapps ROOT

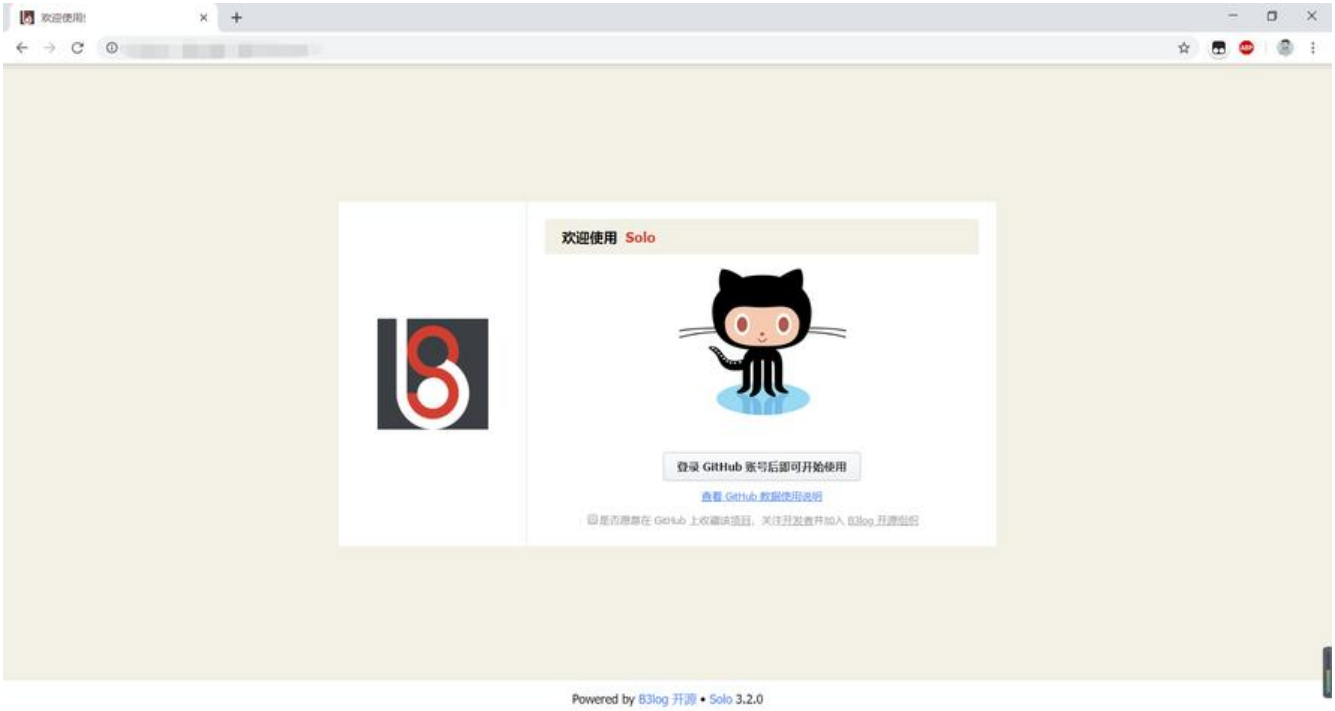
```
[root@localhost webapps]# ls
docs  examples  host-manager  manager  ROOT  ROOT-old
[root@localhost webapps]#
```

3.验证solo能否访问

先打开tomcat服务，然后通过本地浏览器访问http://服务器IP:8080

1. # cd /usr/local/tomcat/tomcat9/bin/
2. # ./startup.sh

出现solo的开始界面说明你的solo博客就部署成功了，你就可以登录绑定你的github账号开始写文章了



4.备份数据

solo博客部署成功后，系统会自动生成一个solo_h2目录，所以只要备份这个文件夹就可以了

```
[root@localhost ~]# ls
anaconda-ks.cfg      solo_h2   模板   图片   下载   桌面
initial-setup-ks.cfg 公共      视频   文档   音乐
[root@localhost ~]#
```

4.安装配置Nginx

博客部署成功后只能通过http://服务器IP:8080 访问，因为tomcat的默认端口是8080.如果你想直接过http://服务器IP 访问，最粗暴的方法就是修改tomcat的配置文件(修改方法请自行百度！)，但是你服务器有两个网站或以上网站的话，用这个方法就会很麻烦。所以我们使用Nginx反向代理。

1.安装Nginx

在Centos下，yum源不提供nginx的安装，可以通过切换yum源的方法获取安装。也可以通过直接下安装包再上传到服务器。这里提供一个Centos7换阿里yum的教程[https://www.itdawei.cn/yum]

首先安装必要的库（nginx中gzip模块需要zlib库，rewrite模块需要pcre库，ssl功能需要openssl库）选定 ./usr/local/ 为安装目录，以下具体版本号根据实际改变。

1.安装安装gcc gcc-c++(如新环境,未安装请先安装)

```
1. # yum install -y gcc gcc-c++
```

2.安装PCRE库

```
1. # cd /usr/local/
2. # wget http://jaist.dl.sourceforge.net/project/pcre/pcre/8.33/pcre-8.33.tar.gz
3. # tar -zxvf pcre-8.33.tar.gz
4. # cd pcre-8.33
5. # ./configure
6. # make && make install
```

3.安装SSL库

```
1. # cd /usr/local/
2. # wget http://www.openssl.org/source/openssl-1.1.1b.tar.gz
3. # tar -zxvf openssl-1.1.1b.tar.gz
4. # cd openssl-1.1.1b/
5. # ./config
6. # make && make install
```

4.安装zlib库存

```
1. # cd /usr/local/
2. # wget http://zlib.net/zlib-1.2.11.tar.gz
3. # tar -zxvf zlib-1.2.11.tar.gz
4. # cd zlib-1.2.11/
5. # ./configure
6. # make && make install
```

5.安装Nginx

```
1. # cd /usr/local/
2. # wget http://nginx.org/download/nginx-1.14.2.tar.gz
3. # tar -zxvf nginx-1.14.2.tar.gz
4. # cd nginx-1.14.2/
5. # ./configure
6. # make && make install
```

6.启动Nginx

```
1. # /usr/local/nginx/sbin/nginx
```

Nginx部分命令

```
重启
1. # /usr/local/nginx/sbin/nginx –s reload

停止
2. # /usr/local/nginx/sbin/nginx –s stop

检查配置文件是否正常
3. # /usr/local/nginx/sbin/nginx –t
```

强制停止
4. # pkill nginx

7.验证Nginx是否安装成功

用本地浏览器访问http://服务器IP

如果能出现Nginx主页，就表示安装成功了，请确保服务器防火墙已关闭

8.设置Nginx开机启动

1. # vi /etc/rc.local
进入文件后按 **i** 增加下面这条代码,然后按**Esc**退出编辑，在输入**:wq**按回车键保存退出
/usr/local/nginx/sbin/nginx
设置权限
2. # cd /etc/
3. # chmod 755 rc.local

9.配置Nginx代理

进入Nginx的配置文件
1. # vi /usr/local/nginx/conf/nginx.conf
进入文件后按 **i** 在配置文件中中添加以下代理规则,然后按**Esc**退出编辑，在输入**:wq**按回键保存退出
server {
listen 80;
server_name 域名(前面不要加协议,多域名用空格分开,没有域名则直接输入服务器ip);
location / {
proxy_pass http://localhost:8080;
proxy_set_header Host $host;
proxy_set_header X-Real-IP $remote_addr;
proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
}
}
重启Nginx服务
2. # /usr/local/nginx/sbin/nginx -s reload

```
    server {
        listen 80;
        server_name www.itdawei.cn itdawei.cn;
        location / {
        proxy_pass http://localhost:8080;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                }
        }

    server {
        listen       80;
        server_name  localhost;

        #charset koi8-r;

        #access_log  logs/host.access.log  main;

        location / {
            root   html;
-- INSERT --
```

10.检查Nginx代理是否配置成功

打开本地浏览器，在地址栏输入服务器IP(绑定了域名则直接输入域名访问)。

如果能直访问8080端口下的solo博客，说明Nginx代理配置成功

到此，solo博客就部署完成了

```
<script>
(function(){
   var bp = document.createElement('script');
   var curProtocol = window.location.protocol.split(':')[0];
   if (curProtocol === 'https') {
      bp.src = 'https://zz.bdstatic.com/linksubmit/push.js';
   }
   else {
      bp.src = 'http://push.zhanzhang.baidu.com/push.js';
   }
   var s = document.getElementsByTagName("script")[0];
   s.parentNode.insertBefore(bp, s);
})();
</script>
```