

发布文档

准备环境

- 1.配置可以共享session的tomcat
- 2.创建docker容器
- 3.上传war包 重启容器

配置可以共享session的tomcat

参考连接<https://blog.csdn.net/lzc409973859/article/details/51981079>

本质为替换掉tomcat默认的session管理

1. 下载RedisSessionManager 源代码 <https://github.com/jcoleman/tomcat-redis-session-manager>
2. 修改源代码 修改RedisSessionManager类中的initializeSerializer()方法

```
1  private void initializeSerializer() throws ClassNotFoundException, IllegalAccessException,
InstantiationException {
2      log.info("Attempting to use serializer :" + serializationStrategyClass);
3      serializer = (Serializer) Class.forName(serializationStrategyClass).newInstance();
4      Loader loader = null;
5      if (getContainer() != null) {
6          loader = getContainer().getLoader();
7      }
8      ClassLoader classLoader = null;
9      if (loader != null) {
10         serializer.setClassLoader(classLoader);
11     }
12 }
13 private void initializeSerializer() throws ClassNotFoundException, IllegalAccessException,
InstantiationException {
14     log.info("Attempting to use serializer :" + serializationStrategyClass);
15     serializer = (Serializer) Class.forName(serializationStrategyClass).newInstance();
16     Loader loader = null;
17     Context context = this.getContext();
18     if (context != null) {
19         loader = context.getLoader();
20     }
21     ClassLoader classLoader = null;
22     if (loader != null) {
23         classLoader = loader.getClassLoader();
24     }
25     serializer.setClassLoader(classLoader);
26 }
```

3. 使用idea maven 打包

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0"
3         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
5     <modelVersion>4.0.0</modelVersion>
6
7     <groupId>com.yuncaai</groupId>
8     <artifactId>build_tomcat_session_redis</artifactId>
9     <version>1.0-SNAPSHOT</version>
10
11     <properties>
12         <maven.compiler.source>8</maven.compiler.source>
13         <maven.compiler.target>8</maven.compiler.target>
14     </properties>
15     <dependencies>
16         <dependency>
17             <groupId>org.apache.tomcat</groupId>
18             <artifactId>tomcat-catalina</artifactId>
19             <version>8.0.33</version>
20         </dependency>
21         <dependency>
22             <groupId>redis.clients</groupId>
23             <artifactId>jedis</artifactId>
24             <version>2.7.2</version>
25         </dependency>
26     </dependencies>
27     <build>
28         <plugins>
29             <plugin> <!-- 打jar包 -->
30                 <groupId>org.apache.maven.plugins</groupId>
31                 <artifactId>maven-compiler-plugin</artifactId>
32                 <version>3.0</version>
33                 <configuration>
34                     <!-- 源代码编译版本 -->
35                     <source>1.8</source>
36                     <!-- 目标平台编译版本 -->
37                     <target>1.8</target>
38                     <!-- 字符集编码 -->
39                     <encoding>UTF-8</encoding>
40                 </configuration>
41             </plugin>
42         </plugins>
43     </build>
44 </project>

```

4. 将 `build_tomcat_session_redis` `jedis-2.7.2.jar` `commons-pool2-2.3.jar` 拷贝到tomcat的lib文件夹下面

这里可以把这些jar放到lib的自定义文件夹。以便与docker映射。

下边是 将jar放入tomcat lib 文件夹下的 rslib文件夹的配置

- 修改tomcat `conf` 文件夹下 `catalina.properties` 配置文件

在common.loader后边添加"\${catalina.home}/lib/rslib/*.jar"

```
1 common.loader="${catalina.base}/lib","${catalina.base}/lib/*.jar","${catalina.home}/lib",  
  "${catalina.home}/lib/*.jar","${catalina.home}/lib/rslib/*.jar"
```

5. 修改tomcat配置 使其使用自定义SessionManager 并配置redis

- 修改tomcat conf 文件夹下 context.xml

```
1 <Valve className="com.orangefunction.tomcat.redissessions.RedisSessionHandlerValve" />  
2 <Manager className="com.orangefunction.tomcat.redissessions.RedisSessionManager"  
3     host="www.cyc-fund.com.cn"  
4     port="16379"  
5     password="*****"  
6     database="0"  
7     maxInactiveInterval="300" />
```

创建docker容器

```
1 ##创建卷 用于docker映射  
2 docker volume create www_tomcat_bin  
3 docker volume create www_tomcat_lib_rslib  
4 docker volume create www_tomcat_conf  
5 docker volume create wb_tomcat_webapp  
6 docker volume create wb_tomcat_logs  
7 docker volume create wb_tomcat_log4j_logs  
8  
9 docker run -di --name=wb_tomcat \  
10 ##时区文件映射  
11 -v /etc/localtime:/etc/localtime \  
12 ##业务 服务上传文件路径映射  
13 -v /home/upload:/home/upload \  
14 ##业务 服务上传文件路径映射  
15 -v /home/virtualPath:/home/virtualPath \  
16 -v www_tomcat_bin:/usr/local/tomcat/bin \  
17 -v www_tomcat_lib_rslib:/usr/local/tomcat/lib/rslib \  
18 -v www_tomcat_conf:/usr/local/tomcat/conf \  
19 -v wb_tomcat_webapp:/usr/local/tomcat/webapps \  
20 -v wb_tomcat_logs:/usr/local/tomcat/logs \  
21 ##log4j日志映射  
22 -v wb_tomcat_log4j_logs:/usr/logs \  
23 -p 8010:8080 tomcat:8.0.33-jre8
```

测试出现的问题

1. 时区问题

- 把宿主机环境映射到容器中 -v /etc/localtime:/etc/localtime
- 修改tomcat catalina.sh 文件

```
1 JAVA_OPTS="$JAVA_OPTS -Duser.timezone=GMT+08"
```

2. 经nginx代理后tomcat获取 `scheme` `servername` `port` 为代理后的信息

- 修改nginx配置文件

```
1 location /
2     {
3         proxy_pass http://127.0.0.1:8010/;
4         proxy_set_header Host $host:$server_port;
5         proxy_set_header X-Real-IP $remote_addr;
6         proxy_set_header X-Forwarded-Proto $scheme;
7     }
```

- 修改tomcat `server.xml` 添加如下配置

```
1 <Valve className="org.apache.catalina.valves.RemoteIpValve" remoteIpHeader="X-Forwarded-
  For" protocolHeader="X-Forwarded-Proto" protocolHeaderHttpsValue="https"/>
```

发布版本

1. 关闭docker容器并 删除 `/var/lib/docker/volumes/wb_tomcat_webapp/_data` 下 `ROOT` 文件夹

```
1 docker stop wb_tomcat
2 cd /var/lib/docker/volumes/wb_tomcat_webapp/_data
3 rm -rf *
```

2. 上传war包到 `/var/lib/docker/volumes/wb_tomcat_webapp/_data` 下

3. 重启docker 容器

```
1 docker start wb_tomcat
```

不停机发布

创建第二个容器 待容器启动成功后修改nginx 配置文件 并使其重新加载配置文件。

1. 创建容器

```
1 ##创建卷 用于docker映射
2 docker volume create www_tomcat_webapp
3 docker volume create www_tomcat_logs
4 docker volume create www_tomcat_log4j_logs
5
6 docker run -di --name=www_tomcat \
```

```

7  ##时区文件映射
8  -v /etc/localtime:/etc/localtime \
9  ##业务 服务上传文件路径映射
10 -v /home/upload:/home/upload \
11 ##业务 服务上传文件路径映射
12 -v /home/virtualPath:/home/virtualPath \
13 -v www_tomcat_bin:/usr/local/tomcat/bin \
14 -v www_tomcat_lib_rslib:/usr/local/tomcat/lib/rslib \
15 -v www_tomcat_conf:/usr/local/tomcat/conf \
16 -v www_tomcat_webapp:/usr/local/tomcat/webapps \
17 -v www_tomcat_logs:/usr/local/tomcat/logs \
18 ##log4j日志映射
19 -v www_tomcat_log4j_logs:/usr/logs \
20 -p 8000:8080 tomcat:8.0.33-jre8

```

2. 上传war包到 `/var/lib/docker/volumes/www_tomcat_webapp/_data` 下

3. 启动容器 并测试是否启动成功

```

1  docker start www_tomcat
2  curl http://127.0.0.1:8000

```

4. 修改nginx配置文件 `proxy_pass`

```

1  location /
2  {
3      #####
4      proxy_pass http://127.0.0.1:8000/;
5      proxy_set_header Host $host:$server_port;
6      proxy_set_header X-Real-IP $remote_addr;
7      proxy_set_header X-Forwarded-Proto $scheme;
8  }

```

5. nginx重新加载配置文件

```

1  /etc/init.d/nginx reload

```

日常发布

上述不停机发布为第一次时的操作

之后 切换 `www` 和 `wb` 两个容器，操作 2 3 4 5步(注意对应容器目录)即可。

回滚方案

操作不停机发布中第 4 5 步即可

