Facebook Presto源码解析(一)---代码编译和踩坑

什么是Presto?

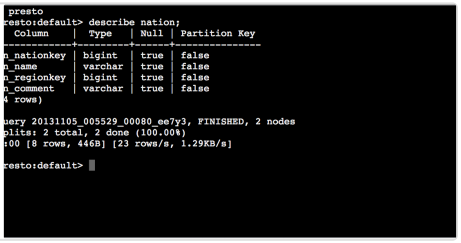
Presto 是专门为大数据实时查询计算而设计和开发的产品。由于Presto是基于Java语言开发的，因此，对于使用者和开发者而言，Presto容易学习。无论是对多数据源支持，还是高性能、易用性、可扩展性等方面，Presto都是大数据实时查询计算产品中的佼佼者，接下来的一段时间，我也将全面系统的进行Presto的源码解析，首先让我们从Presto的环境搭建和代码编译开始吧！

Presto的官网地址：

Presto | Distributed SQL Query Engine for Big Data

Presto is an open source distributed SQL query engine for running interactive analytic queries against data sources of all sizes ranging from gigabytes to petabytes. Presto was designed and written from the

https://prestodb.io



Presto的gihub地址：

prestodb/presto

Presto is a distributed SQL query engine for big data. See the User Manual for deployment instructions and end user documentation. Mac OS X or Linux Java 8 Update 151 or higher (8u151+), 64-bit. Both

https://github.com/prestodb/presto

首先去github上面下载最新的稳定版本，官网目前用的是release-0.240，所以我们这次也用release -0.240来进行源码分析。

准备以下环境：

- Mac OS X or Linux（我这里使用的Mac OS）

- Java 8 151 版本或更新的
- Maven 3.3.9+
- Python 2.4+
- Git
- IntelliJ IDEA

clone好源代码后我们maven导入IDEA，请注意：

1. 我们要谨慎使用2020版的IDEA（最好都用去年的版本），IDEA从2020版开始lombok,maven插件都出现过问题，并且现在最新的2020.2.2仍旧是会产生一些maven的问题，导致一些包无法正常下载，暂时切回2019.3.5。
2. JDK版本问题，如果我们用最新的IDEA又不想自己下载JDK的话，那它只会提供JDK15，由于JDK15 已经把unsafe类收回了，所以如果我们使用15仍旧会遇到一些编译问题（虽然可以解决但是没必要），暂时切回JDK 1.8.261
3. 记得提前配置下maven的环境变量，我们下载的maven版本是3.6.3，首先看下自己iterm随便进终端看下窗口顶部显示的是什么，如果是zsh则我们编辑~./zshr，如果是bash则编辑~./base_profile，添加如下配置，根据自己实际的maven路径来就行:

```
export MAVEN_HOME=/Users/xxx/Downloads/apache-maven-3.6.3
export PATH=$PATH:$MAVEN_HOME/bin
#(如果是base_profile才要加这一句)
source ~/.bash_profile
```

如果是base_profile我们还要记得添加source ~/.bash_profile

然后记得修改下maven的settings文件，加一个mirror镜像地址，一般用阿里的会比较快，不然包下不下来编译也没进行了，进入上面MAVEN_HOME的路径找到conf/settings.xml文件，编辑找到mirrors添加如下配置：

```
<mirror>
   <id>ali maven</id>
   <name>aliyun maven</name>
   <url>https://maven.aliyun.com/repository/public/</url>
```

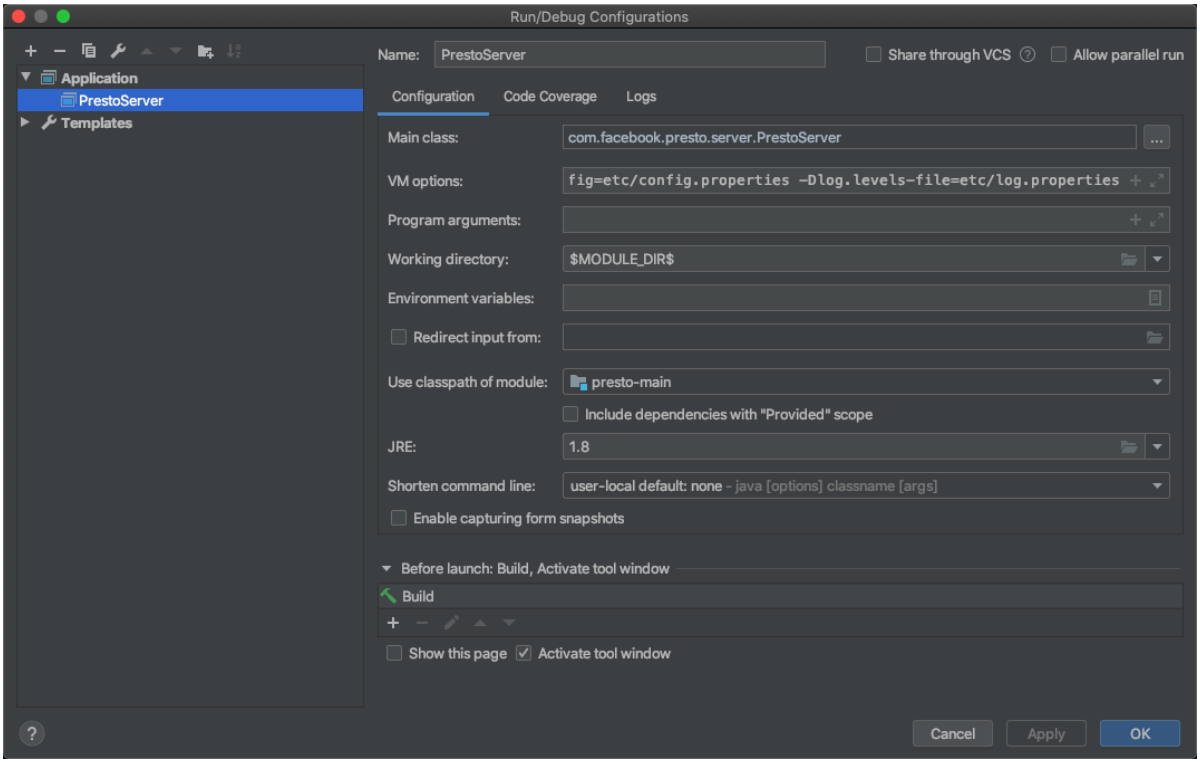
```
    <mirrorOf>central</mirrorOf>
</mirror>
```

做好上面那些事情，我们静静等待maven把所有依赖包下下来，大概30分钟的样子，依赖包实在是太多了，所以不要着急，下载好包之后我们开始编译。我们先试着把所有的moudle全部编译一遍，打开IDEA的Terminal 运行以下命令：

```
./mvnw clean install -DskipTests
```

编译完成后，我们先来配置Presto-Server，也就是Presto的服务端，这是Presto的核心功能所在

点击Edit Configurations 我们可以看到Templates我们点击上方的“+”，添加一个Application,做如下配置：



其中VM options : ea -XX:+UseG1GC -XX:G1HeapRegionSize=32M -XX:+UseGCOverheadLimit -XX:+ExplicitGCInvokesConcurrent -Xmx2G -Dconfig=etc/config.properties -Dlog.levels-file=etc/log.properties

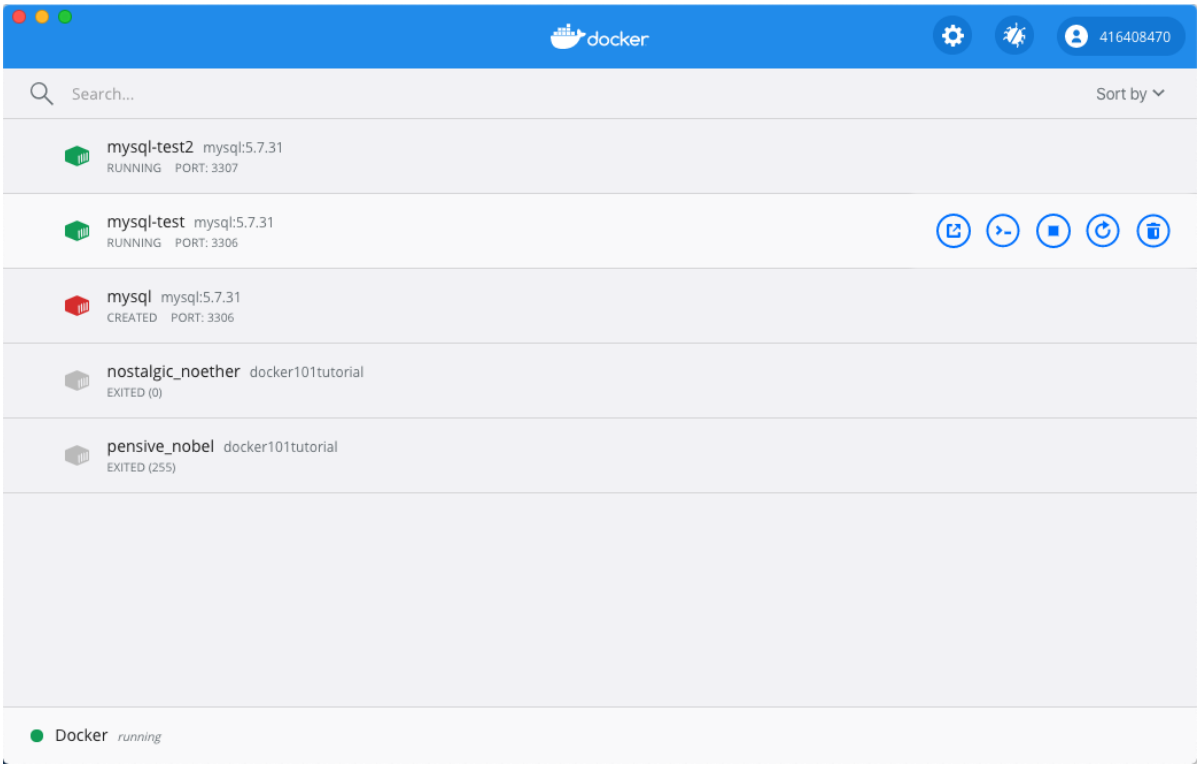
配置好服务端之后我们接着配置客户端，其实客户端没啥好配置的，我们直接在Terminal里面执行：

```
presto-cli/target/presto-cli-*-executable.jar
```

就会进入presto，我们尝试使用一下：

```
presto> show scheme;
Error running command: Error starting query at http://localhost:8080/v1/statement returned an invalid response:
 JsonResponse{statusCode=404, statusMessage=Not Found, headers={connection=[keep-alive], content-length=[153],
content-type=[text/html], date=[Wed, 23 Sep 2020 02:16:17 GMT], server=[nginx/1.19.2]}, hasValue=false} [Error:
 <html>
<head><title>404 Not Found</title></head>
<body>
<center><h1>404 Not Found</h1></center>
<hr><center>nginx/1.19.2</center>
</body>
</html>
```

很好，直接报错了，不着急，最起码我们运行起来了，只是有部分没有配置好，接下来我们先装个docker。Mac上装docker很简单，直接brew install docker就行，然后根据他的提示我们装好就行，最终运行的页面：



然后我们在docker里面安装个mysql，我选择的5.7.31这个版本。

dockerhub 地址：

> mysql Tags - Docker Hub
> MySQL is a widely used, open-source relational database management system (RDBMS).
> https://hub.docker.com/_/mysql?tab=tags

然后启动两个端口映射的mysql service

```
docker run -itd --name mysql-test -p 3306:3306 -e MYSQL_ROOT_PASSWORD=123456 mysql:5.7.31

docker run -itd --name mysql-test -p 3307:3306 -e MYSQL_ROOT_PASSWORD=123456 mysql:5.7.31
```

这样我们就两个mysql都启动了，然后我们试着用DataGrip连接下，然后创建两个库，以备后面的使用。

现在我们的服务都能正常运行了，只是写sql查询的时候还会有一些报错，那么我们后续在源码解析中跟着一步步走来一起探讨这些问题是如何来解决的。