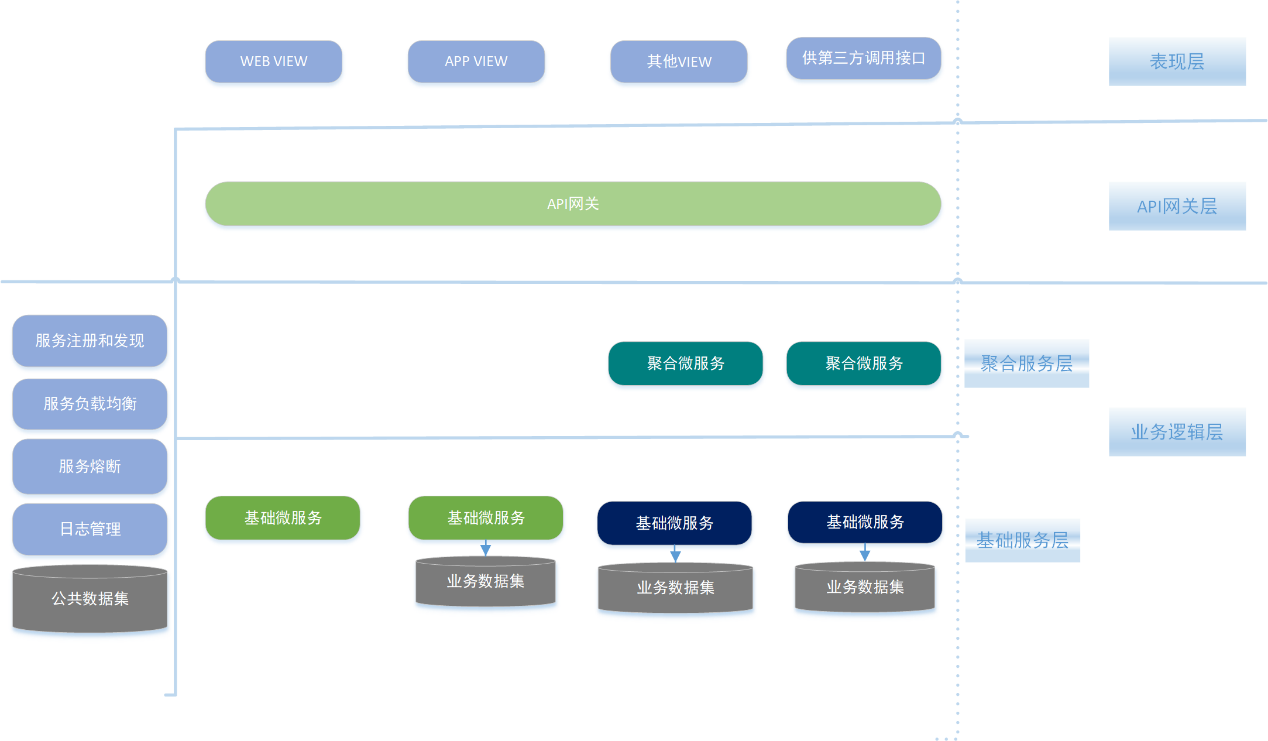
<https://blog.csdn.net/stevehe/article/details/80431727>

<https://cloud.tencent.com/developer/news/257354>



这个微服务的架构图挺好的！

我上传的demo基于java 使用springBoot+dubbo+zookeeper来构建微服务。

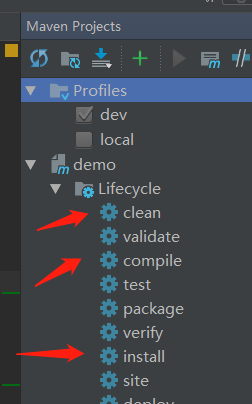
编译器idea jdk1.8.0 maven（jar包管理和发布）。

我这有资源，如果要的话，来考或者我百度云分享。

请自行安装配置maven（先配置自己的仓库）。

Api 工程为部署jar，工程导入后，需要mvn install命令 安装到本地仓库。

或者用idea或eclipse 集成的功能来部署。



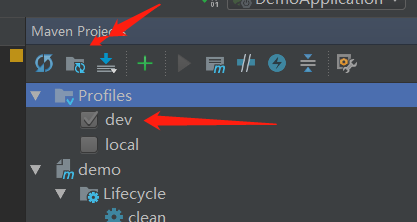
Provider工程属于基础服务。

工程导入后，通过更新pom.Xml能将所需依赖安装到本地仓库。

工程的配置文件主要在maven的pom.xml和springBoot相关的application文件中。

Application-dev 和Appliction-local为生产环境和本地环境可以修改maven配置文件切换

或者通过idea



Mysql和zookeeper我都配置好了，用的我们实验室的服务器，校内可连。

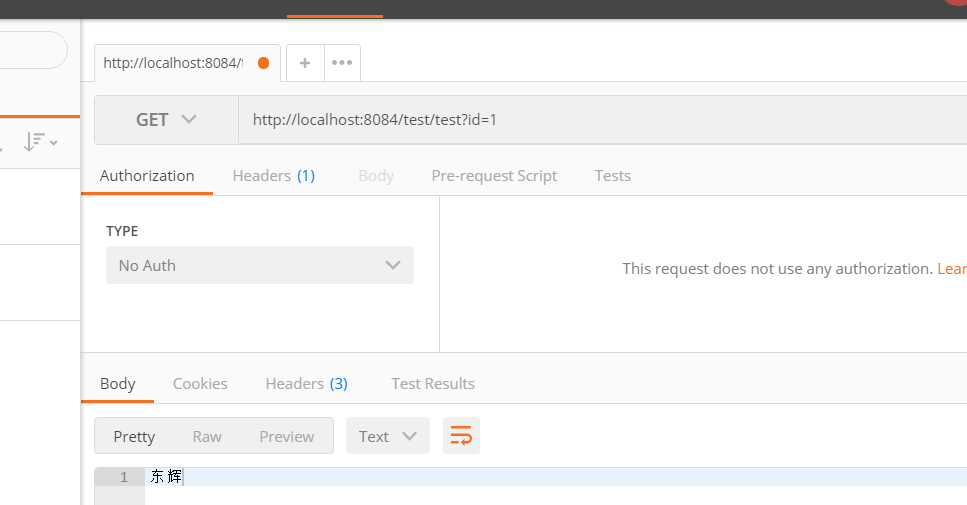
连接信息自己找ｂ(￣▽￣)ｄ。

数据库的副本我就不传了，自己登陆远程看吧。

就两个表。

Provider工程从数据库中取了数据，并把自己注册到了zookeeper注册服务器上（通过dubbo，以provider的身份）。

web工程，先理解成网关，以consumer的身份拿到了provider的服务。



虽然都部署到了一台机器上，实际上是跑的两个工程，两个工程之间通过rpc联动。

最上边架构图。

表现层：不用管。（前端）

网关层：向当与 我传的web工程。

业务逻辑层：

一个provider工程，可以理解成一个基础微服务，多个provider就是多个基础微服务。如果一个provider既是消费者也是生产者（dubbo-provider和duboo-consumer）他就是聚合微服务。

负载均衡，服务注册和发现是zookeeper办到的。

我可能表述的不太好，群上沟通。