Demo是gradle 父子工程，父工程师test-dubbo-parent，子工程是test-dubbo-server，test-dubbo-server-war1，test-dubbo-web1，test-dubbo-api，后续如果还需要添加子工程的话，请在test-dubbo-parent工程下面的settings.xml里面进行配置，test-dubbo-server是用springboot搭建的，DubboServerApplication是入口程序，部署的是个jar包程序，test-dubbo-server-war1是用spring搭建的，部署的是个war包程序，test-dubbo-web1是个web程序，可以用来调用web工程的dubbo协议接口。

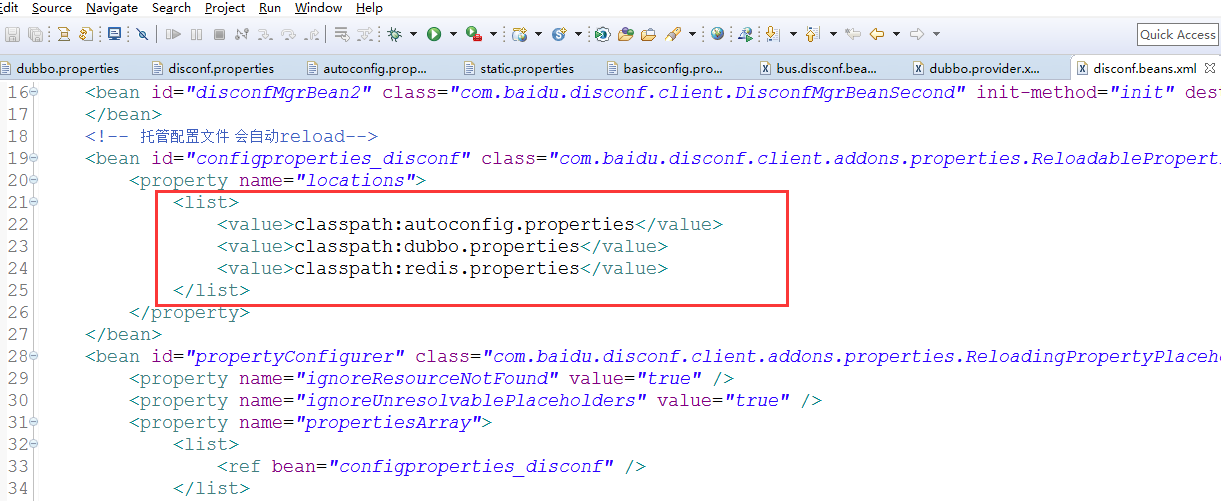
test-dubbo-server在入口文件DubboServerApplication里面会读取，相对应的配置文件，其中配置文件通过disconf部署在 disconf相对应的服务上面，地址是139.224.226.95:8081，用户名和密码是admin/admin，用户可以通过这界面配置相对应的appId，和相对应的配置文件。

其中dubbo.properties文件是dubbo接口所需要的配置文件，其中需要注意的是，由于配置文件中我们说明dubbo.protocol.rest.server=servlet，故dubbo.protocol.rest.contextpath的值一定要和你的工程名称保持一致，不然启动会报错，dubbo.protocol.rest.port必须要和服务的端口号保持一致，redis.properties是配置了redis的相关东西，其中password是通过了自定义的java代码加密的，

autoconfig.properties里面会读取一个mysql数据库，库里面的表配置了分表分库的策略，其中sys\_schema\_athm是配置的具体按照什么策略分库分表，是按照本地网分，还是按照什么别的分，sys\_schema\_ds配置的是分表分库连接的具体数据源，sys\_schema\_node配置的是默认数据源，sys\_schema\_seq配置序列在哪个库，sys\_schema\_table表配置具体的每个表是按照本地网的值去分，还是按照表名本地网后缀分。

dubbo.provider.xml配置的dubbo接口发布的一些情况，有rest，协议的，有dubbo协议的，rest协议是给外系统调用的，dubbo协议是给web系统调用的，调用的时候，需要引用test-dubbo-api工程，这里面定义了dubbo协议接口的地址。

disconf.properties是disconf相关jar包在启动的时候会自动读取根目录下的这个配置文件，disconf.beans.xml配置了引用disconf所必须的一些配置信息，其中如果在disconf配置页面上面新增加了一个配置项目的话，这里也要相对应的配置一下，不然程序不知道要索取什么，在程序启动的时候，会自动从disconf 服务器上面索取配置文件，并下载到本地disconf/download下面，具体的路径地址是配置在disconf的目录下面的



test-dubbo-server-war1是一个可以用容器部署的服务，比如tomcat服务，部署的话，入口是web.xml，会加载相对应的配置信息，和普通的web工程师一样的，

test-dubbo-web1是个web工程，里面可以写jsp，js的工程，测试页面是test.jsp,在这个页面里面会调用servlet，然后会调用dubbo协议的接口。

打包的话，cmd到相对应工程的build.gradle目录下面去，运行gradle build命令，其中test-dubbo-server会打包成一个jar包，其他的会打包成一个war包。

common-utils工程是一个工具类工程，里面目前包含字符串，日期，报文解析，和连接数据源获得序列的方法

注意事项：

test-dubbo-server-war1，test-dubbo-web1每次gradle之后都需要在下面这个图上面重新引入一次api工程，不然程序会报错，虽然现实引用了，但是在下面Advanced窗口可以看出，引用的不正确，需要重新引入一次这个工程

