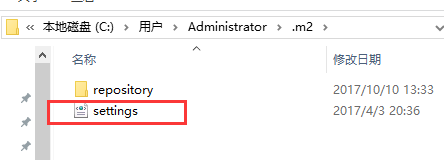
# Tars创建自己的server、client微服务项目启动调试步骤

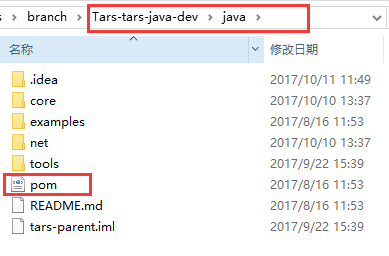
## 1前期准备

a、下载maven，解压，只需配置下maven的环境变量即可哦。maven的仓库可以不修改，使用默认仓库即可。使用默认仓库，b、需要将maven的配置文件copy到默认仓库同级目录下。



## 2 tars安装

1. 下载tars源码（可以重新解压一份，解压就好，不需导入IED中，最好不要用以前研究很久的导入IDEA中的，以防有改动过版本之类的不兼容），进入java源码目录



1. 执行如下命令，将tars安装在本地仓库

mvn clean install

mvn clean install -f core/client.pom.xml

mvn clean install -f core/server.pom.xml

## 3 创建server、client微服务项目

在IDEA中新建自己的server、client微服务项目（注：要创建maven的项目，最好不要直接copy tars项目example中的server、client，自己new，不然不熟悉maven配置情况下，会被maven的依赖路径搞晕哦）

步骤：File -> New -> Project -> Maven Project ->下一步，（可选 maven-archetype-webapp），再输入groupId、artifactId

注：其中起名以App、Server、Obj结尾的，前后配置相关名称全部要一致。Tars是以App+Server+Obj

进行路由分配查找的，所以不一致，会使通信失败哦。

### **3.1** **S**erver项目配置

我这里起名CoolChattingServer）

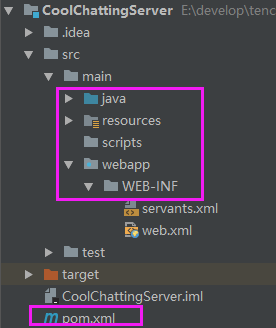
1）Server的groupId、artifactId命名参考：



2）修改CoolChattingServer的pom文件，加入相关依赖，可以参考如下：



此时项目目录结构如下（木有的目录自己创建补全哈）：

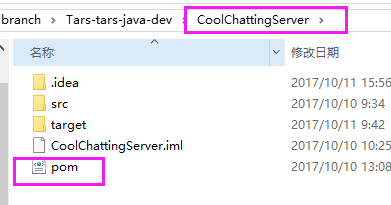


3）接口文件定义

src/main/resources目录下建立hello.tars文件，内容如下：

module TestApp  
{  
 interface Hello  
 {  
 string hello(int no, string name);  
 };  
};

4）在工程根目录下cmd中执行mvn tars:tars2java，会生成接口类HelloServant.java



5）新创建一个HelloServantImpl.java文件，实现HelloServant.java接口

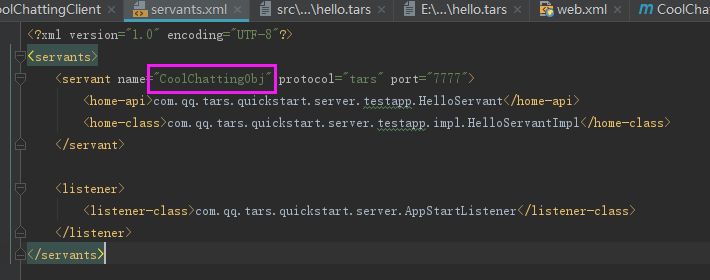
public class HelloServantImpl implements HelloServant {

public String hello(int no, String name) {

return String.format("hello no=%s, name=%s, time=%s", no, name, System.currentTimeMillis());//可以修改返回的字符串

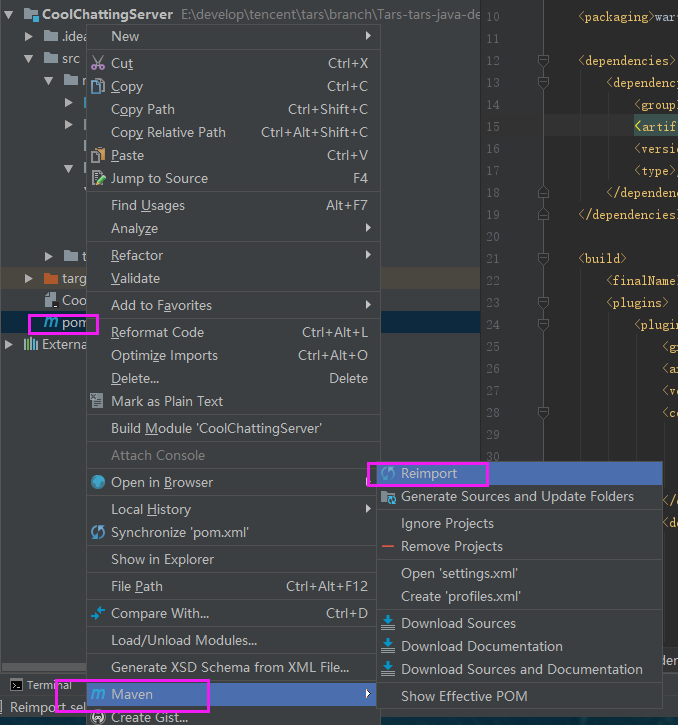
}

4）CoolChattingServer在WEB-INF下创建一个servants.xml的配置文件，服务编写后需要进程启动时加载配置暴露服务，配置如下。web.xml随意从tars中copy一个过来即可：

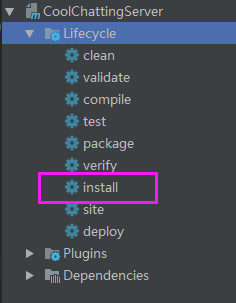


5）编译项目

更新maven：



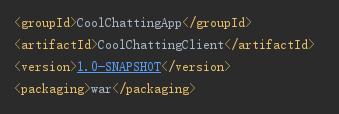
2）运行install



### 3.2 Client项目配置

（CoolChattingClient）的项目

1）groupId、artifactId命名参考：



2）修改CoolChattingClient 的pom文件，加入相关依赖，可以参考如下：



3）在resource下新建接口文件hello.tars文件，内容跟server中的一样。

4）编写main方法，调用服务器：

public static void main(String[] args) {

CommunicatorConfig cfg = new CommunicatorConfig();

//构建通信器

Communicator communicator = CommunicatorFactory.getInstance().getCommunicator(cfg);

//通过通信器，生成代理对象. 192.168.109.1要修改成自己的IP地址

HelloPrx proxy = communicator.stringToProxy(HelloPrx.class,

"CoolChattingObjApp.CoolChattingServer.CoolChattingObj@tcp -h 192.168.109.1 -p 7777 -t 60000");

String ret = proxy.hello(1000, "HelloWorld");

System.out.println(ret);

//异步调用

proxy.async\_hello(new HelloPrxCallback() {

@Override

public void callback\_expired() {

}

@Override

public void callback\_exception(Throwable ex) {

}

@Override

public void callback\_hello(String ret) {

System.out.println(ret);

}

}, 1000, "Hello World ghhghhg");

}

5）编译项目，参考server的。

## 4 启动server服务器

1）首先在工程根目录下cmd中执行：

mvn tars:build -Dapp=CoolChattingObjApp -Dserver=CoolChattingServer -DjvmParams="-Xms1024m -Xmx1024m -Dcom.sun.management.jmxremote.ssl=false -Dcom.sun.management.jmxremote.authenticate=false -Xdebug -Xrunjdwp:transport=dt\_socket,address=9000,server=y,suspend=n"

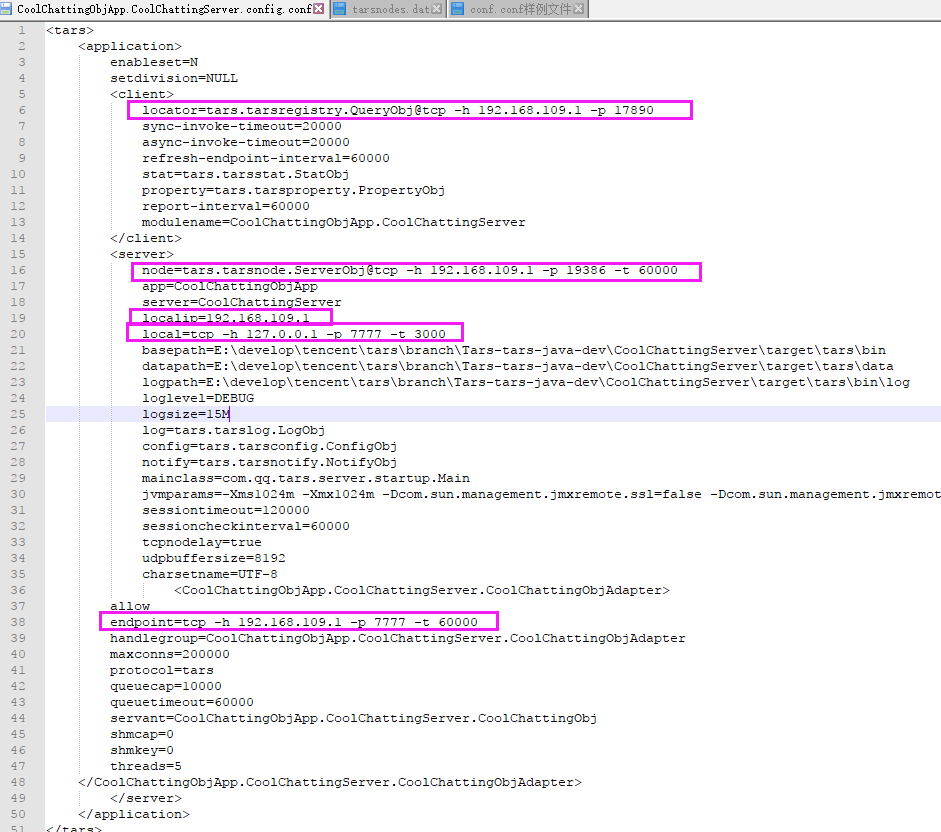
注：CoolChattingServer是server服务名字，跟自己的项目名称保持一致。

运行成功之后，会在CoolChattingServer\target\tars\conf

生成CoolChattingObjApp.CoolChattingServer.config.conf配置文件（就是tars源码core中start时new server中加载的配置文件）。如果server配置的一些数据不正确，该配置文件中可能会缺少或有些信息错误，参考如下文件修改补全哈：



（文件中192.168.109.1要是自己电脑的IP才对哈，主要修改的是如下图中红色的部分，注意IP、端口、APP+Server+Client）：



2）在工程目录target/tars/bin/tars\_start 启

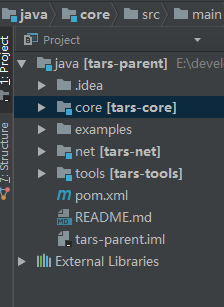
动服务

## 5 启动client客户端

client项目，在IDEA中点击main方法的类直接用debug启动，打断点调试即可。

# Tars服务器debug调试

1. 将tars的Tars-tars-java-dev\java项目引入到IDEA中，引入成功之后目录结构如下：



2、com.qq.tars.server.config.ConfigurationManager#loadServerConfig方法中，开头把配置文件放入System的config的 Property中：

System.setProperty("config", "E:\\develop\\tencent\\tars\\branch\\Tars-tars-java-dev\\CoolChattingServer\\target\\tars\\conf

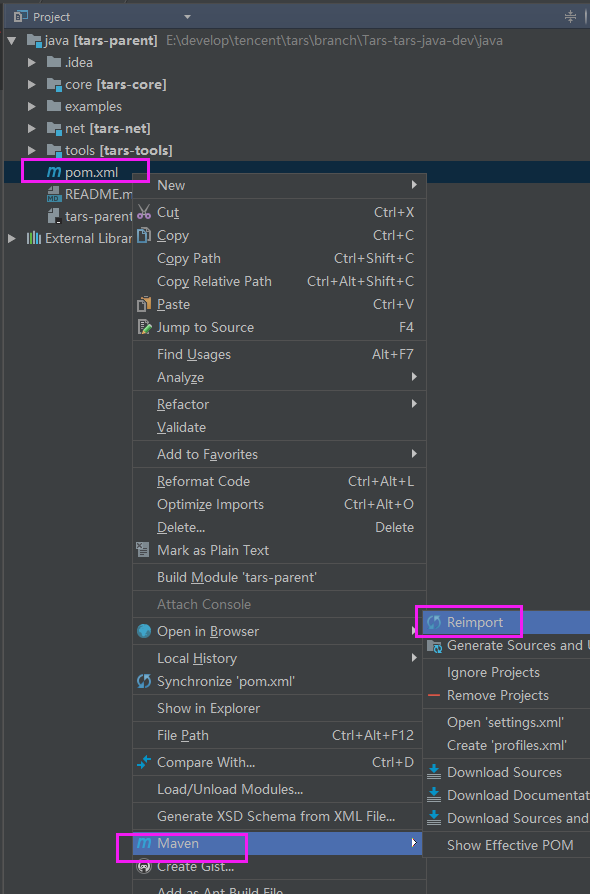
\\CoolChattingObjApp.CoolChattingServer.config.conf");

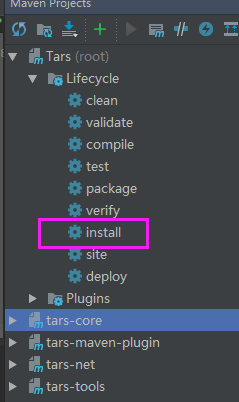
注:CoolChattingObjApp.CoolChattingServer.config.conf配置文件是用脚本文件启动server生成，拿过来用用，帮助调试)

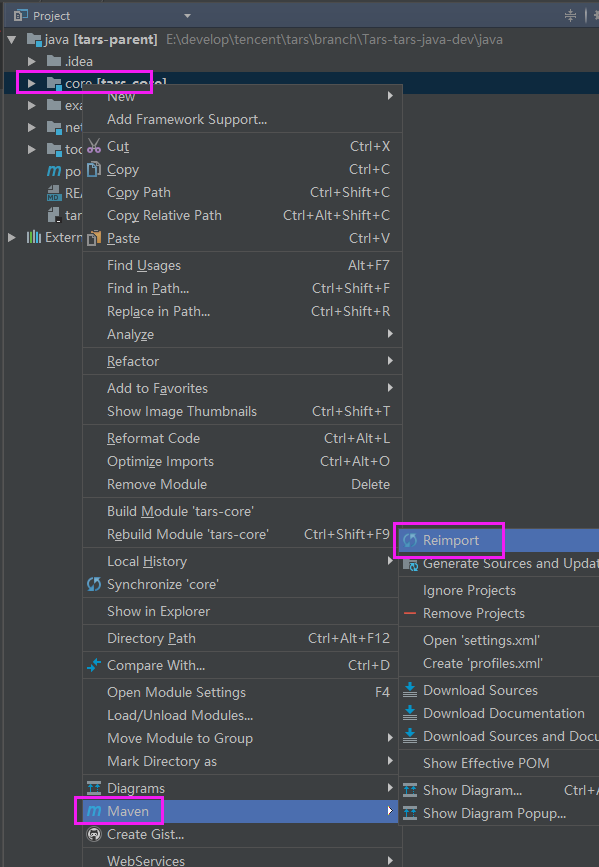
1. 编译

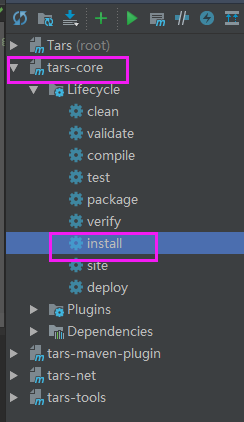
先reimport，再install。先编译parent，后编译core下的client、server.

具体如下图操作，顺序不要反，看好选的文件，不要选错：









3、在com.qq.tars.server.startup.Main类中打好断点，右键debug运行，跟踪调试。