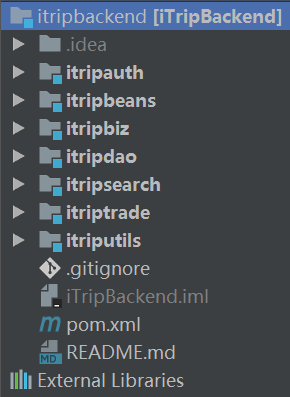
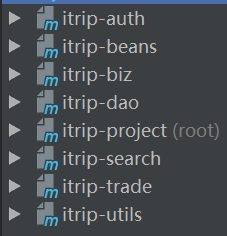
**i旅行——项目架构**

i旅行平台，架构从两个方向进行设计，前端架构和后端架构，将前端和后端完全分离：

1. 前端：itripfront
2. 后端：iTripBackend，主要分为以下4个模块：
3. itrip主业务模块（itripbiz）：包括酒店业务、旅游业务、机票业务、攻略业务
4. itrip搜索模块（itripsearch）：爱旅行项目中所有搜索功能（solr）
5. itrip Auth认证模块（itripauth）：负责用户身份验证，生成并维护Token
6. itrip支付模块（itrptrade）：包括支付宝支付、微信支付

对于以上的web模块，由于存在一些公用的部分（比如：pojo、dao、utils等），为提高代码复用性及可维护性，故采用Maven进行多Module管理，拆分后具体结构入下：

注意：各模块之间存在依赖，切不可存在相互依赖，目前依赖关系如下：

itripdao模块depedency：itrip-beans、itrip-utils、solr-solrj

itriputils模块depedency：itrip-beans

itripbiz模块depedency：itrip-dao、itrip-beans、itrip-utils

itripsearch模块depedency：itrip-dao、itrip-beans、itrip-utils

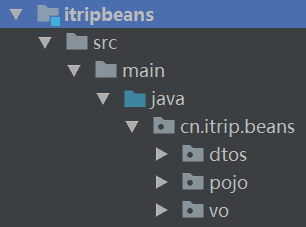
itriptrade模块depedency：itrip-dao、itrip-utils

itripauth模块depedency：itrip-dao、itrip-utils、itrip-beans

各模块具体说明如下：

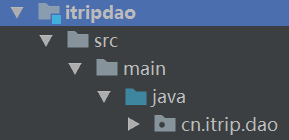
1. itripbeans：包括dto、pojo、vo

该模块需要打成jar包，供其他web模块使用

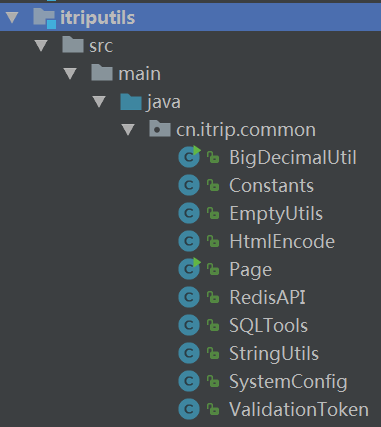


1. itripdao：包括dao层所有代码

该模块需要打成jar包，供其他web模块使用

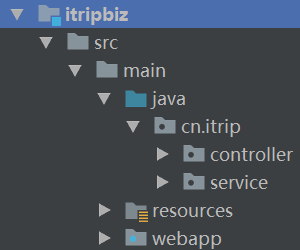


1. itriputils：包括common包下所有工具类



1. itripbiz：包括controller、service层以及resources、webapp

该web模块需要打成war包，部署路径：/itripbiz



1. itripsearch：包括controller、service层以及resources、webapp

该web模块需要打成war包，部署路径：/ itripsearch

1. itripauth：包括controller、service层以及resources、webapp

该web模块需要打成war包，部署路径：/ itripauth

1. itriptrade：包括controller、service层以及resources、webapp

该web模块需要打成war包，部署路径：/ itriptrade

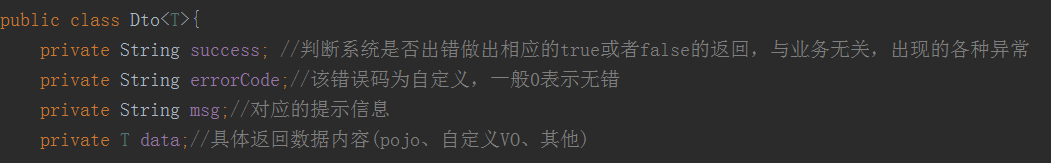
**基于以上多模块开发，需要遵守代码约定：**

1. itripbeans模块

pojo：自动生成

dtos：数据传输对象，包括输出前端Dto以及接收前端传入的InputDto，数据结构如下：

* 1. Dto .java



根据service返回的结果，Controller层的API接口定义往前端输出的数据内容（json格式），e.g：

{

"success":"false",

"errorCode":"100001",

"msg":"用户名或者密码错误",

"data"：""

}

或者

{

"success":"true",

"errorCode":"0",

"msg":"图片上传成功",

"data：

["http://imgs.itrip.com/comment/100078-1493694934624-5129266.jpg","http://imgs.itrip.com/comment/100078-1493694934624-5129266.jpg"]

}

注意：错误码根据业务定义

酒店主业务biz：1开头（10000）

评论：10001 ——10100

酒店详情：10101 ——10200

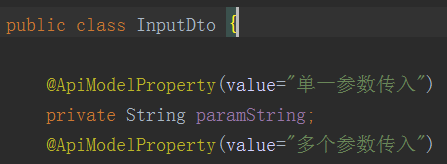
订单：10201 ——10400

搜索search：2开头（20000）

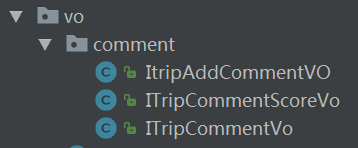
认证auth：3开头（30000）

支付trade：4开头（40000）

* 1. InputDto.java



VO：根据业务功能创建各自VO，并根据业务分包放置，比如：



注意：VO由于会是前后端的参数接收对象，需要加入swagger的@ApiModel，进行详细说明，并且VO命名须明确。

1. itripdao模块

代码自动生成

1. itriputils模块

所有的工具类都放置在此模块，注意：尽量不要使用注解，降低耦合度

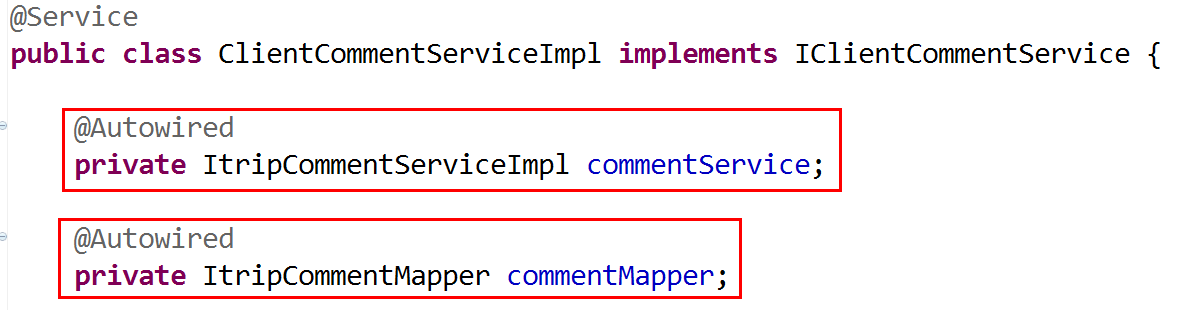
注：新增ValidationToken.java，提供公共的验证Token的工具类

1. itripbiz模块

4.1> Service层：业务逻辑层，注：均需throws Exception，以便事务控制

目前代码中存在问题:

问题：client层Service只能调用service层，不得调用mapper，即：不能跨层调用！并且client service实现类中不能注入service接口的实现类，只能是接口。

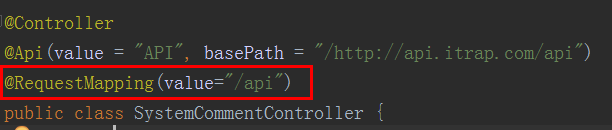


PS：service层封装的太多，太乱，不好实施。自动生成代码的service建议去掉，业务逻辑层手工完成代码。

4.2> Controller层

提供API接口，各web工程都需要加入swagger，以便给前端生成API文档

Controller都需要加统一前缀目录：/api/业务模块名



PS：

1、关于@Resource和@Autowired，希望能够统一下，service层和controller层均使用@Resource

2、API接口开发中需要注意：所有url请求均为小写

1. itripsearch模块
2. itripauth模块

**生成订单规则：机器码 +日期+（MD5）（商品IDs+毫秒数+1000000的随机数）**