# 说明文档

### JAVA ：Java代码包

## Configuration配置类所在包

MybatisConfig: Mybatis数据源和Sql会话工厂，事务管理器的配置

XXXDS（）配置数据源和Druid数据连接池

XXXSQlSessionFactory：配置SQl会话工厂以获得SQl会话，建立和数据库连接

OauthConfig：Oauth授权服务器和多个资源服务器的配置

ResourceServerConfiguration：配置资源服务器

AuthorizationServerConfiguration（）：配置授权服务器

SecurityConfig：Spring Secirity配置。

## Controller 控制器类所在包

黑色部分为接口URL。

AccessConfirmationController：Oauth授权确认端点控制器

**"/oauth/confirm\_access**

FSRegisterController：FreeSwitch注册端点控制器

**fs/register/getDN**

**fs/register/releaseDN**

FSWorkOrderController：FreeSwitch工单接口控制器

**fs/workorder**

GET以查询，PUT以插入

GreetingController：测试接口控制器

**/greeting"**

MessageController：服务历史信息接口控制器

**/messages**

GET以查询，PUT以插入

PageController：Web管理子系统前端页面控制器

不需要了解接口URL，使用Web系统即可

SeatsController：坐席信息接口控制器

**/getSeats查询**

ServiceInfoController：单次服务信息接口控制器

**getServiceinfo查询**

**PutServiceInfo插入**

TokenController：取授权码接口控制器

**/clienttoken取授权码**

UserRegisterController：用户注册接口控制器

**user/register/getDN**

## Domain映射到数据库表的普通对象所在包

Approval：授权中的“一次授权批准”

Client：客户（被授权者）信息

DN：

Message：“服务历史信息”类

ServiceInfo：“单次服务信息”类

WorkOrder：工单信息类

User：用户（管理员）基本信息类

UserDetailsImpl：用户（管理员）详细信息类，包含User Filed

## Pojo不映射到数据库表的普通对象所在包

Greeting：测试用对象

Seats：坐席对象

## service服务类（业务逻辑）所在包，Impl类为相关实现

LogService：插入API使用信息统计的服务

Log()

StaticticsService：查询API使用统计信息服务

TimeStatistic()

ClientService：操作Client信息的各种方法的服务类

根据Client的各种信息字段查询，更新client的方法，根据方法名就能理解

MessageService：服务历史信息的“CRUD”服务类

FindMEssages（）：查询信息

insertMessages（）：插入信息

ServiceInfoService：单次服务信息的“CRUD”服务类

LoadByID（）：根据服务信息的id查询

PutInfo():插入一次服务信息

UserRegisterService：用户注册DN的“CRUD”服务类

LoadDN（）：加载用户DN

PutDN（）：用户注册DN

WorkOrderService：工单信息的“CRUD”服务类

GetWorkOrderAT（）：根据agent和tenant查询工单

GetworkOrderUT（）：根据user和tenant查询工单

PostWorkOrder（）：插入工单

UserDetailsService：加载用户详细信息的服务类

LoadUserByUsername（）：实现SPring的userSetails接口方法，根据User姓名加载用户信息，调用UserService

UserService：用户基本信息（对应User领域对象）的“CRUD”服务类

Findbyname（）根据username查询信息

DelectByname（）根据username删除用户信息

UpdateUser（）更新用户信息

CreateDefaultUser（）创建具有默认信息的user

CreateUSer():创建user

## Support ：DAO类所在包

几个DAO类：包含一样的数据库操作方法，只是每个类操作的数据源不同。

### Resources资源包

## Mappers Mybatis XMl映射文件所在包

InsertLog.xml：Log插入和查询SQl语句模板所在文件

UserMapper.xml：用户信息CRUD操作SQl语句模板所在文件

authenticNumer.xml：Log插入和查询SQl语句模板所在文件

ClientAgentServiceMapper.xml：Log插入和查询SQl语句模板所在文件

ClientMapper: Log插入和查询SQl语句模板所在文件

ClientMessageMapper：Log插入和查询SQl语句模板所在文件

ClientWorkOrderMapper: Log插入和查询SQl语句模板所在文件

## Public css和js文件所在包

## Templates 前端页面所在包

|  |  |
| --- | --- |
| 文件名 | 描述 |
| Apientry.vm | 展示API分类，进入单个API的入口 |
| Clientoverall.vm | 展示所有客户的简略信息页 |
| Editclient.vm | 编辑客户信息页 |
| Singleapi.vm | 单个API详细信息展示页 |
| Singleclient.vm | 单个客户详细信息展示页 |
| Autho.vm | 授权 |
| Access\_confirmation.vm | 授权过程中的Scope确认页 |
| Login.vm | 登录页 |
| Error.vm | 错误页 |

## Application.properties 应用全局配置文件，包括配置数据库地址，服务端口等配置信息

### 软件使用方法

使用Java –jar命令启动打包好的软件Jar包即可启动，端口为8081

web子系统地址为 ServiceIP:8081/admin,使用admin，123用户即可进入系统，其中管理客户和API信息都是立即能明白如何使用的。

测试用服务接口地址：ServiceIP:8081/Greeting

#### 授权测试使用说明：

授权码模式步骤说明:

A）用户访问客户端，后者将前者导向认证服务器。

（B）用户选择是否给予客户端授权。

（C）假设用户给予授权，认证服务器将用户导向客户端事先指定的"重定向URI"（redirection URI），同时附上一个授权码。

（D）客户端收到授权码，附上早先的"重定向URI"，向认证服务器申请令牌。这一步是在客户端的后台的服务器上完成的，对用户不可见。

（E）认证服务器核对了授权码和重定向URI，确认无误后，向客户端发送访问令牌（access token）和更新令牌（refresh token）。

授权码模式演示:

使用浏览器作为客户端,客户端引导用户点进去这个URL  
URL,也就是像授权服务器发送请求

<http://localhost:8080/oauth/authorize?response_type=code&client_id=client&redirect_uri=http%3a%2f%2flocalhost%3a8080%2fwelcome>

response\_type指定了是授权码模式,客户端提供client\_id,还有重定向URI.这样,授权服务器就知道是哪个client发来的请求.

授权服务器一般将要求用户先登录,在此输入用户名zpw,密码123即可

将出现是否同意授权的页面,选择同意并确定

授权服务器将让浏览器跳转到之前发送来的重定向URI,点开地址看,发现里面有个code参数,就是授权码,客户端于是就拿着授权码申请token

使用post方法请求token

curl -X POST client:123@localhost:8080/oauth/token \

-d grant\_type=authorization\_code \

-d client\_id=client \

-d client\_secret=123 \

-d redirect\_uri= http://localhost:8080/getToken \

-d code=cNgx9a

code=BCm83L&grant\_type=authorization\_code&client\_secret=123&redirect\_uri=http://localhost:8080/getToken&client\_id=client2

可以看到服务器返回了token信息.

Client\_id是client信息,在授权服务器里注册好的,此处的重定向URI  
URI必须和之前的一样,有验证client身份的作用

接下来client就可以在每个要访问资源的URL中加入token参数,这样就可以访问授权到的资源了