

一、软件安装

二、服务功能测试

- 1、注册用户（只有新用户不用token）
- 2、获取新用户token
- 3、获取用户具体信息（例如权限等）
- 4、自定义资源服务模块使用（即功能模块）

三、新增服务配置

四、服务发布部署（推送到git并在服务器上运行）

- 1、版本推送
- 2、服务器部署运行
 - 2.1 docker打包镜像

五、资源服务文件结构

一、软件安装

1.jdk—java运行环境（请使用jdk1.8，否则后面会报很多错误，不好解决）

2.Intellij IDEA—java IDE集成开发软件

3.lombok-集成在intellij IDEA中（preference-plugin-browrepositories-搜索lombok-安装）

注：安装完后记得配置compiler-annotation compiler选择

4.Docker-如果是linux或mac建议按照（window比较费事）

注：如果安装Docker就可以新建mysql和redis容器本地跑起来

5.Git—进行版本推送（git指令见Study_Note）

6.Postman—方便的模拟get或者post或者其他方式的请求来调试接口（辅助用来获取token等）

二、服务功能测试

<http://zuul>网关启动的服务器的ip地址和端口，通过网关统一对后台微服务进行访问

1、注册用户（只有新用户不用token）

auth/user/register

Params:username、password、email、repeatPassword

注意：用户名 密码最小六位最大20位

The screenshot shows a Postman interface with a POST request to `http://192.168.85.208:9000/auth/user/register?username=hxy...`. The request body contains the following parameters:

Key	Value	Description
<input checked="" type="checkbox"/> username	hxy111	
<input checked="" type="checkbox"/> password	111111	
<input checked="" type="checkbox"/> email	hxy@111.com	
<input checked="" type="checkbox"/> repeatPassword	111111	

The response status is 200 OK, with a time of 382 ms and a size of 466 B. The response body is a JSON object:

```
1 {
2   "code": 0,
3   "msg": "成功",
4   "data": {
5     "id": 1005998455216726017,
6     "username": "hxy111",
7     "email": "hxy@111.com"
8   }
9 }
```

2、获取新用户token

auth/oauth/token

params: username、password、grant_type（目前写死为password）

用户名密码必须存在在数据库中（即当前用户已经注册成功，且用户名密码匹配）

The screenshot shows a Postman interface with a POST request to `http://192.168.85.208:9000/auth/oauth/token?username=hxy...`. The request body contains the following parameters:

Key	Value	Description
<input checked="" type="checkbox"/> username	hxy111	
<input checked="" type="checkbox"/> password	111111	
<input checked="" type="checkbox"/> grant_type	password	

The response status is 200 OK, with a time of 242 ms and a size of 529 B. The response body is a JSON object:

```
1 {
2   "access_token": "08c33438-0b27-4a94-818b-52bdf30458d",
3   "token_type": "bearer",
4   "refresh_token": "bac4f72a-8d1c-4442-a280-f9b7a514b714"
5   "expires_in": 43199,
6   "scope": "xx"
7 }
```

用户名密码不匹配情况:

REST client interface showing a POST request to `http://192.168.85.208:9000/auth/oauth/token?username=hxy...`. The request body contains the following parameters:

Key	Value	Description
username	hxy111	
password	111	错误的密码
grant_type	password	

The response is a 400 Bad Request with the following JSON body:

```
{  "error": "invalid_grant",  "error_description": "Bad credentials"}
```

3、获取用户具体信息（例如权限等）

`/auth/user/current`—这是个get方法，不允许给服务器发送数据

在params填写`access_token`

REST client interface showing a GET request to `http://192.168.85.208:9000/auth/user/current?access_token=...`. The request body contains the following parameters:

Key	Value	Description
access_token	08c33438-0b27-...	

The response is a 200 OK with the following JSON body:

```
{  "authorities": [    {      "authority": "query2\n"    }  ],  "details": {    "remoteAddress": "192.168.85.208",    "sessionId": null,    "tokenValue": "08c33438-0b27-4a94-818b-52bdf30458d",    "tokenType": "Bearer",    "decodedDetails": null  },  "authenticated": true,  "userAuthentication": {    "authorities": [      {        "authority": "query2\n"      }    ],    "details": {      "grant_type": "password",      "username": "hxy111"    },    "authenticated": true,    "principal": {      "password": null,      "username": "hxy111",    }  }  }
```

`/auth/user/me`

http://192.168.85.208: No Environment

GET http://192.168.85.208:9000/auth/user/me?access_token=08c3... Params Send Save

Status: 200 OK Time: 42 ms Size: 738 B

Key	Value	Description
access_token	08c33438-0b27...	
New key	Value	Description

Auth Headers (1) Body Pre-req. Tests Cookies Code

TYPE Basic Auth Preview Request

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

Username android Password android ☒ Show Password

Body Cookies Headers (10) Test Results

Pretty Raw Preview JSON

```
1 {
2   "code": 0,
3   "msg": "成功",
4   "data": {
5     "id": 1005998455216726017,
6     "username": "hxy111",
7     "email": "hxy@111.com",
8     "imageUrl": null,
9     "roles": [
10      {
11        "id": 2,
12        "name": "普通用户",
13        "value": "ROLE_USER",
14        "authorities": [
15          {
16            "id": 2,
17            "name": "查询2",
18            "value": "query2\n"
19          }
20        ]
21      }
22    ],
23    "enabled": true,
24    "authorities": [
25      {
26        "authority": "query2\n"
27      }
28    ],
29    "accountNonExpired": true,
```

4、自定义资源服务模块使用（即功能模块）

http://192.168.85.208:9000/auth/oauth/token?username=hxy111&password=111111&grant_type=password Examples (0)

GET http://192.168.85.208:9000/kitchen/demo?access_token=08c3... Params Send Save

Status: 403 Forbidden Time: 99 ms Size: 435 B

Key	Value	Description
access_token	08c33438-0b27...	
New key	Value	Description

Auth Headers (1) Body Pre-req. Tests Cookies Code

TYPE Basic Auth Preview Request

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

Username android Password android ☒ Show Password

Body Cookies Headers (10) Test Results

Pretty Raw Preview JSON Save Response

```
1 {
2   "error": "access_denied",
3   "error_description": "不允许访问"
4 }
```

因为demo要求用户权限为query，而用户权限为query2，所以该用户的token没有权限访问demo的接口

http://192.168.85.208:9000/auth/oauth/token?username=hxy111&password=111111&grant_type=password Examples (0)

GET http://192.168.85.208:9000/kitchen/demo2?access_token=08c3... Params Send Save

Status: 200 OK Time: 67 ms Size: 363 B

Key	Value	Description
access_token	08c33438-0b27...	
New key	Value	Description

Auth Headers (1) Body Pre-req. Tests Cookies Code

TYPE Basic Auth Preview Request

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

Username android Password android ☒ Show Password

Body Cookies Headers (10) Test Results

Pretty Raw Preview Text Save Response

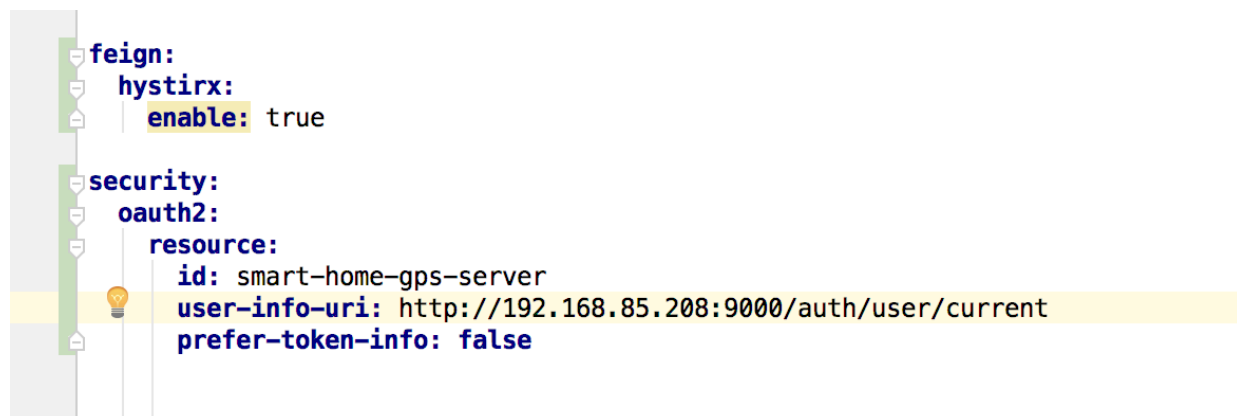
```
1 good
```

三、新增服务配置

<http://192.168.84.199:10080/earthchen/smart-home/src/master>

1.后台新增微服务时

- 在新增对应服务的application.yml进行security配置



注释：若使用redis缓存—（当服务涉及到访问数据库使用redis缓存），还需要进行redis和mysql配置

- 微服务目录下新增cn.edu.chzu.smart.home.config包，创建配置类（复制已有的模块就可以）
- 网关zuul-gateway进行反向代理路由，zuul-gateway微服务的application.yml进行配置



- 新增微服务的入口需要配置网关反向

```
cation.yml x GpsServerApplication.java x
package cn.edu.chzu.smart.home;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.EnableAutoConfiguration;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.autoconfigure.jdbc.DataSourceAutoConfiguration;
import org.springframework.cloud.client.discovery.EnableDiscoveryClient;
import org.springframework.security.config.annotation.method.configuration.EnableGlobalMethodSecurity;

/**
 * @author wangqianlong
 * @create 2018-06-08 13:01
 */
@SpringBootApplication
@EnableDiscoveryClient
@EnableGlobalMethodSecurity(prePostEnabled = true)
@EnableAutoConfiguration(exclude={DataSourceAutoConfiguration.class})
public class GpsServerApplication {
    public static void main(String[] args) { SpringApplication.run(GpsServerApplication.class, args); }
}
```

2.客户端新增一个接口应用时，需要在auth-server（认证服务中）application.yml配置server-security

四、服务发布部署（推送到git并在服务器上运行）

1、版本推送

修改代码时，需要拉取最新版本代码（即从git上克隆版本历史中最新的版本代码），再次基础上进行修改

修改代码后，通过git push的方式推送到gitee平台自己的分支中，并注明修改的内容和版本

2、服务器部署运行

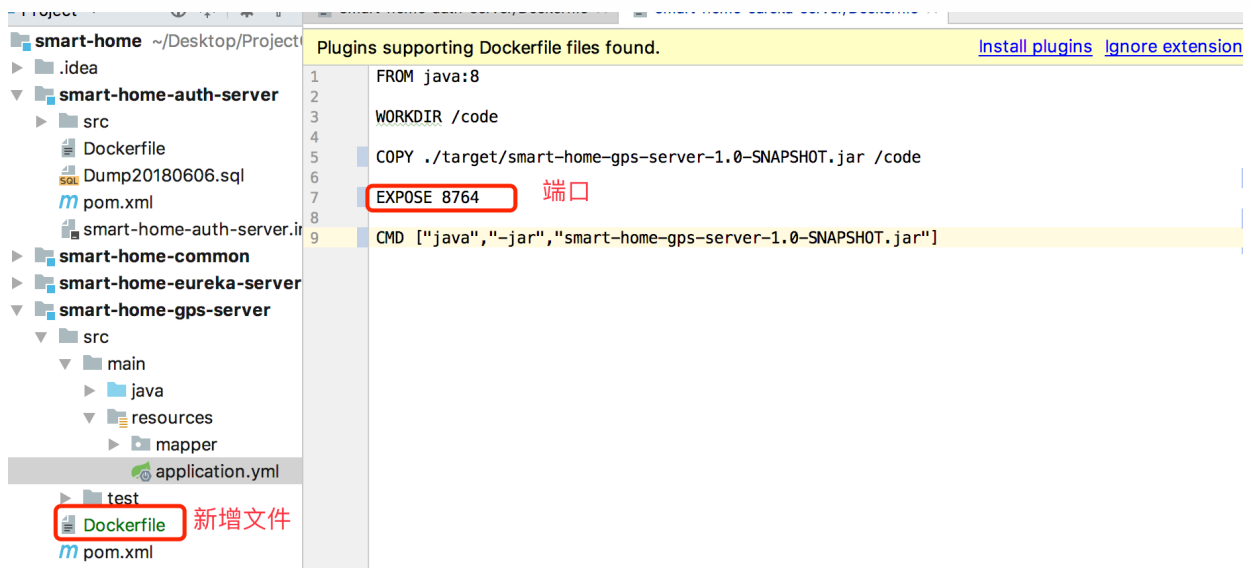
见服务器的readme.md的三种部署方式

<http://192.168.84.199:10080/earthchen/smart-home/src/master>

本文详细介绍docker方式进行服务器代码部署和运行

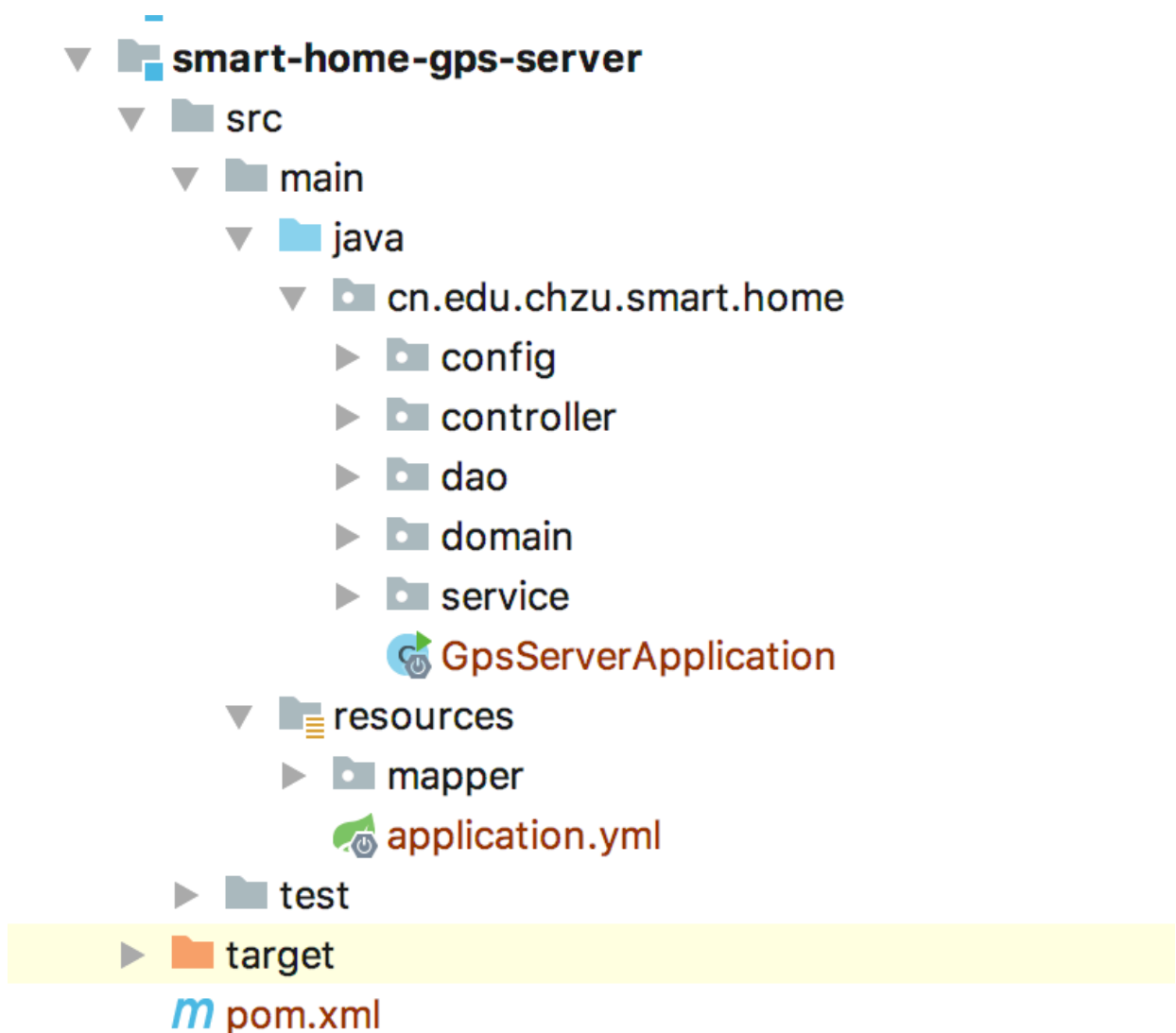
2.1 docker打包镜像

- 在自己开发的资源服务包下面新增一个Dockerfile文件（可以参考已有的文件）



注意：打包镜像时，请注意需要进入dockerfile所在的目录，运行如下命令：

五、资源服务文件结构



config

controller

dao

domain

service

GpsServerApplication——入口函数

resource-mapper

resource-application.yml