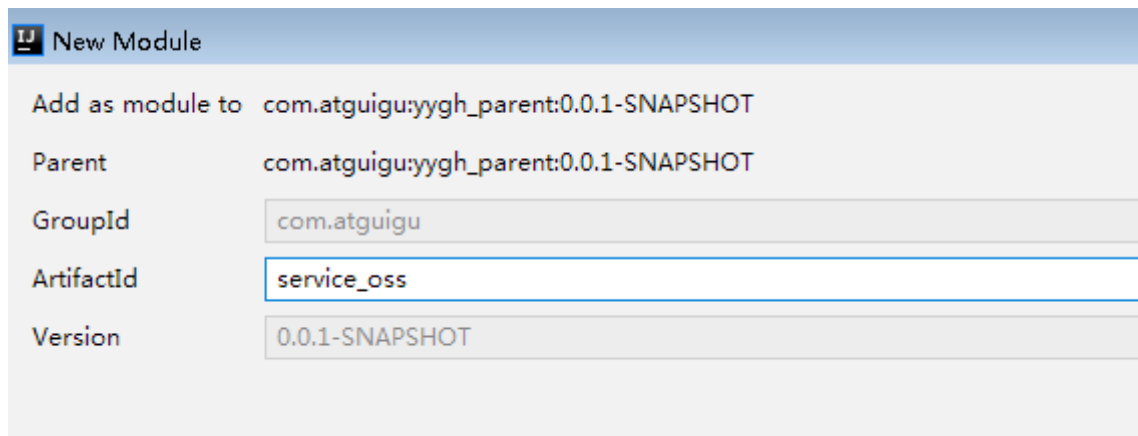


一、新建云存储微服务

1、在service模块下创建子模块service_oss



New Module	
Add as module to	com.atguigu:yygh_parent:0.0.1-SNAPSHOT
Parent	com.atguigu:yygh_parent:0.0.1-SNAPSHOT
GroupId	com.atguigu
ArtifactId	service_oss
Version	0.0.1-SNAPSHOT

2、配置pom.xml

service-oss上级模块service已经引入service的公共依赖，service-oss模块只需引入阿里云oss相关依赖

```
1 <dependencies>
2     <!-- 阿里云oss依赖 -->
3     <dependency>
4         <groupId>com.aliyun.oss</groupId>
5         <artifactId>aliyun-sdk-oss</artifactId>
6     </dependency>
7
8     <!-- 日期工具栏依赖 -->
9     <dependency>
10        <groupId>joda-time</groupId>
11        <artifactId>joda-time</artifactId>
12    </dependency>
13 </dependencies>
```

3、配置application.properties

```
1 #服务端口
2 server.port=8205
```

```

3 #服务名
4 spring.application.name=service-oss
5
6 #环境设置: dev、test、prod
7 spring.profiles.active=dev
8
9 spring.servlet.multipart.max-file-size=1024MB
10 spring.servlet.multipart.max-request-size=1024MB
11
12 #阿里云 OSS
13 #不同的服务器, 地址不同
14 aliyun.oss.file.endpoint=oss-cn-beijing.aliyuncs.com
15 aliyun.oss.file.keyid=your accessKeyId
16 aliyun.oss.file.keysecret=your accessKeySecret
17 #bucket可以在控制台创建, 也可以使用java代码创建
18 aliyun.oss.file.bucketname=guli-file

```

4、创建启动类

创建OssApplication.java

```

1 @SpringBootApplication(exclude = DataSourceAutoConfiguration.class)
2 @ComponentScan(basePackages = {"com.atguigu"})
3 public class OssApplication {
4
5     public static void main(String[] args) {
6         SpringApplication.run(OssApplication.class, args);
7     }
8 }

```

二、实现文件上传接口

1、从配置文件读取常量

创建常量读取工具类: ConstantPropertiesUtil.java

使用@Value读取application.properties里的配置内容

用spring的 InitializingBean 的 afterPropertiesSet 来初始化配置信息, 这个方法将在所有的属性被初始化后调用。

```

1  /**
2   * 常量类，读取配置文件application.properties中的配置
3   */
4  @Component
5  //@PropertySource("classpath:application.properties")
6  public class ConstantPropertiesUtil implements InitializingBean {
7
8      @Value("${aliyun.oss.file.endpoint}")
9      private String endpoint;
10
11     @Value("${aliyun.oss.file.keyid}")
12     private String keyId;
13
14     @Value("${aliyun.oss.file.keysecret}")
15     private String keySecret;
16
17     @Value("${aliyun.oss.file.bucketname}")
18     private String bucketName;
19
20     public static String END_POINT;
21     public static String ACCESS_KEY_ID;
22     public static String ACCESS_KEY_SECRET;
23     public static String BUCKET_NAME;
24
25     @Override
26     public void afterPropertiesSet() throws Exception {
27         END_POINT = endpoint;
28         ACCESS_KEY_ID = keyId;
29         ACCESS_KEY_SECRET = keySecret;
30         BUCKET_NAME = bucketName;
31     }
32 }

```

2、文件上传

创建Service接口：FileService.java

```

1  public interface FileService {
2
3      /**

```

```

4      * 文件上传至阿里云
5      */
6      String upload(MultipartFile file);
7  }

```

实现：FileServiceImpl.java

参考SDK中的：Java->上传文件->简单上传->流式上传->上传文件流

- 上传文件流

以下代码用于将文件流上传到目标存储空间examplebucket中的exampleobject.txt文件。

```

// yourEndpoint填写Bucket所在地域对应的Endpoint。以华东1（杭州）为例，Endpoint填写为h
String endpoint = "yourEndpoint";
// 阿里云账号AccessKey拥有所有API的访问权限，风险很高。强烈建议您创建并使用RAM用户进行
String accessKeyId = "yourAccessKeyId";
String accessKeySecret = "yourAccessKeySecret";

// 创建OSSClient实例。
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId, accessKeySecret);

// 填写本地文件的完整路径。如果未指定本地路径，则默认从示例程序所属项目对应本地路径中
InputStream inputStream = new FileInputStream("D:\\\\localpath\\\\examplefile.txt");
// 填写Bucket名称和Object完整路径。Object完整路径中不能包含Bucket名称。
ossClient.putObject("examplebucket", "exampleobject.txt", inputStream);

// 关闭OSSClient。
ossClient.shutdown();

```

```

1  @Service
2  public class FileServiceImpl implements FileService {
3      @Override
4      public String upload(MultipartFile file) {
5          // Endpoint以杭州为例，其它Region请按实际情况填写。
6          String endpoint = ConstantOssPropertiesUtils.EDNPOINT;
7          String accessKeyId = ConstantOssPropertiesUtils.ACCESS_KEY_ID;
8          String accessKeySecret = ConstantOssPropertiesUtils.SECRET;
9          String bucketName = ConstantOssPropertiesUtils.BUCKET;
10         try {
11             // 创建OSSClient实例。
12             OSS ossClient = new OSSClientBuilder().build(endpoint, accessKey1
13             // 上传文件流。
14             InputStream inputStream = file.getInputStream();
15             String fileName = file.getOriginalFilename();
16             //生成随机唯一值，使用uuid，添加到文件名称里面
17             String uuid = UUID.randomUUID().toString().replaceAll("-", "");

```

```

18         fileName = uuid+fileName;
19         //按照当前日期, 创建文件夹, 上传到创建文件夹里面
20         // 2021/02/02/01.jpg
21         String timeUrl = new DateTime().toString("yyyy/MM/dd");
22         fileName = timeUrl+"/"+fileName;
23         //调用方法实现上传
24         ossClient.putObject(bucketName, fileName, inputStream);
25         // 关闭OSSClient。
26         ossClient.shutdown();
27         //上传之后文件路径
28         // https://yygh-atguigu.oss-cn-beijing.aliyuncs.com/01.jpg
29         String url = "https://" + bucketName + "." + endpoint + "/" + fileName;
30         //返回
31         return url;
32     } catch (IOException e) {
33         e.printStackTrace();
34         return null;
35     }
36 }
37 }

```

3、创建Controller

FileUploadController.java

```

1  @Api(description="阿里云文件管理")
2  @RestController
3  @RequestMapping("/admin/oss/file")
4  public class FileUploadController {
5
6      @Autowired
7      private FileService fileService;
8
9      /**
10       * 文件上传
11       */
12      @ApiOperation(value = "文件上传")
13      @PostMapping("upload")
14      public R upload(
15          @ApiParam(name = "file", value = "文件", required = true)
16          @RequestParam("file") MultipartFile file) {
17

```

```
18         String uploadUrl = fileService.upload(file);
19         return R.ok().message("文件上传成功").data("url", uploadUrl);
20
21     }
22 }
```

4、配置网关

```
1 #设置路由id
2 spring.cloud.gateway.routes[5].id=service-oss
3 #设置路由的uri
4 spring.cloud.gateway.routes[5].uri=lb://service-oss
5 #设置路由断言,代理servicerId为auth-service的/auth/路径
6 spring.cloud.gateway.routes[5].predicates= Path=/*/oss/**
```