

一行代码搞定Http请求？强得离谱~

点击关注
 
 Java基基
 2023-06-04 11:55
 Posted on 上海

点击上方“**Java基基**”，选择“设为星标”  
 做积极的人，而不是积极废人！  
 每天 **14:00** 更新文章，每天掉亿点头发...



Scan to Follow

源码精品专栏

- [原创 | Java 2021 超神之路，很肝~](#)
- [中文详细注释的开源项目](#)
- [RPC 框架 Dubbo 源码解析](#)
- [网络应用框架 Netty 源码解析](#)
- [消息中间件 RocketMQ 源码解析](#)
- [数据库中间件 Sharding-JDBC 和 MyCAT 源码解析](#)
- [作业调度中间件 Elastic-Job 源码解析](#)
- [分布式事务中间件 TCC-Transaction 源码解析](#)
- [Eureka 和 Hystrix 源码解析](#)
- [Java 并发源码](#)

来源：andyoung.blog.csdn.net/

article/details/127755025

- **OKHttpUtil**
- **OKHttpUtil 功能**
- **OKHttpUtil使用**
  - **GET**
  - **POST**
  - **文件上传**
  - **下载文件**
  - **HttpRequest 链式请求**
- **再 Springboot 中使用**
- **如何快速封装外部接口**



OKHttpUtil

在Java的世界中，Http客户端之前一直是Apache家的HttpClient占据主导，但是由于此包较为庞大，API又比较难用，因此并不使用很多场景。而新兴的OkHttp、Jodd-http固然好用，但是面对一些场景时，学习成本还是有一些的。很多时候，我们想追求轻量级的Http客户端，并且追求简单易用。而OKHttp是一套处理HTTP网络请求的依赖库，由Square公司设计研发并开源，目前可以在Java和Kotlin中使用。对于Android App来说，OkHttp现在几乎已经占据了所有的网络请求操作，对于服务器端请求外部接口也是必备的选择。针对OKHttp OkHttpUtil做了一层封装，使Http请求变得无比简单。

基于 **Spring Boot + MyBatis Plus + Vue & Element** 实现的后台管理系统 + 用户小程序，支持 **RBAC 动态权限**、多租户、数据权限、工作流、三方登录、支付、短信、商城等功能

- 项目地址：<https://github.com/YunaiV/ruoyi-vue-pro>
- 视频教程：<https://doc.iocoder.cn/video/>

OKHttpUtil 功能

- 根据URL自动判断是请求HTTP还是HTTPS，不需要单独写多余的代码。
- 默认情况下Cookie自动记录，比如可以实现模拟登录，即第一次访问登录URL后后续请求就是登录状态。
- 自动识别304跳转并二次请求
- 支持代理配置
- 支持referer配置
- 支持User-Agent配置
- 自动识别并解压Gzip格式返回内容
- 支持springboot 配置文件
- 极简的封装调用

基于 [Spring Cloud Alibaba + Gateway + Nacos + RocketMQ + Vue & Element](#) 实现的后台管理系统 + 用户小程序，支持 [RBAC 动态权限](#)、[多租户](#)、[数据权限](#)、[工作流](#)、[三方登录](#)、[支付](#)、[短信](#)、[商城](#)等功能

- 项目地址：<https://github.com/YunaiV/yudao-cloud>
- 视频教程：<https://doc.iocoder.cn/video/>

## OKHttpUtil使用

maven引入

```
<dependency>

<groupId>io.github.admin4j</groupId>

<artifactId>http</artifactId>

<version>0.4.0</version>

</dependency>
```

最新版查询 [search.maven.org/artifact/io.github.admin4j/http](https://search.maven.org/artifact/io.github.admin4j/http)

### GET

最简单的使用莫过于用HttpUtil工具类快速请求某个接口：

```
Response response = HttpUtil.get("https://github.com/search", Pair.of("q", "okhttp"));

System.out.println("response = " + response);
```

### POST

一行代码即可搞定，当然Post请求也很简单：

```
JSON 格式的body
Response post = HttpUtil.post("https://oapi.dingtalk.com/robot/send?access_token=27f5954ab60ea8b2e43",
Pair.of("q", "http"),
Pair.of("username", "admin123"));
System.out.println("post = " + post);

form 请求
Map<String, Object> formParams = new HashMap<>(16);
formParams.put("username", "admin");
formParams.put("password", "admin123");
Response response = HttpUtil.postForm("http://192.168.1.13:9100/auth/login",
formParams
);
System.out.println("response = " + response);
```

返回格式为JSON的 可以使用 HttpJsonUtil 自动返回JsonObject

```
JsonObject object=HttpJsonUtil.get("https://github.com/search",
Pair.of("q", "http"),
Pair.of("username", "agonie201218"));
System.out.println("object = "+object);
```

### 文件上传

```
File file=new File("C:\\Users\\andanyang\\Downloads\\Sql.txt");
Map<String, Object> formParams=new HashMap<>();
formParams.put("key", "test");
formParams.put("file", file);
formParams.put("token", "WXyUseb-D4sCum-EvTIDYL-mEehwDtrSBg-Zca7t:qg0cR2gUoKmxT-VnsNb6570atzo=:eyJzY2");
Response response=HttpUtil.upload("https://upload.qiniup.com/", formParams);
System.out.println(response);
```

### 下载文件

```
HttpUtil.down("https://gitee.com/admin4j/common-http", "path/");
```



HttpRequest 链式请求

```
# get
Response response=HttpRequest.get("https://search.gitee.com/?skin=rec&type=repository")
    .queryMap("q","admin4j")
    .header(HttpHeaderKey.USER_AGENT,"admin4j")
    .execute();
System.out.println("response = "+response);

post form
Response response=HttpRequest.get("http://192.168.1.13:9100/auth/login")
    .queryMap("q","admin4j")
    .header(HttpHeaderKey.USER_AGENT,"admin4j")
    .form("username","admin")
    .form("password","admin123")
    .execute();
System.out.println("response = "+response);
```

post form 日志

```
16:49:14.092[main]DEBUG io.github.admin4j.http.core.HttpLogger- -->GET http://192.168.1.13:9100/auth/login
16:49:14.094[main]DEBUG io.github.admin4j.http.core.HttpLogger-User-Agent:admin4j
16:49:14.094[main]DEBUG io.github.admin4j.http.core.HttpLogger-Host:192.168.1.13:9100
16:49:14.094[main]DEBUG io.github.admin4j.http.core.HttpLogger-Connection:Keep-Alive
16:49:14.094[main]DEBUG io.github.admin4j.http.core.HttpLogger-Accept-Encoding:gzip
16:49:14.094[main]DEBUG io.github.admin4j.http.core.HttpLogger- -->END GET
16:49:14.670[main]DEBUG io.github.admin4j.http.core.HttpLogger-<--200OK http://192.168.1.13:9100/auth/login
16:49:14.670[main]DEBUG io.github.admin4j.http.core.HttpLogger-transfer-encoding:chunked
16:49:14.670[main]DEBUG io.github.admin4j.http.core.HttpLogger-Vary:Origin
16:49:14.670[main]DEBUG io.github.admin4j.http.core.HttpLogger-Vary:Access-Control-Request-Method
16:49:14.670[main]DEBUG io.github.admin4j.http.core.HttpLogger-Vary:Access-Control-Request-Headers
16:49:14.670[main]DEBUG io.github.admin4j.http.core.HttpLogger-Vary:Origin
16:49:14.670[main]DEBUG io.github.admin4j.http.core.HttpLogger-Vary:Access-Control-Request-Method
16:49:14.670[main]DEBUG io.github.admin4j.http.core.HttpLogger-Vary:Access-Control-Request-Headers
16:49:14.671[main]DEBUG io.github.admin4j.http.core.HttpLogger-Content-Type:application/json;charset=utf-8
16:49:14.671[main]DEBUG io.github.admin4j.http.core.HttpLogger-Date:Tue,08Nov 2022 08:49:14GMT
16:49:14.671[main]DEBUG io.github.admin4j.http.core.HttpLogger-{"code":406,"msg":"Full authentication is required to access this resource"}
16:49:14.671[main]DEBUG io.github.admin4j.http.core.HttpLogger-<--END HTTP(76-byte body)
response=Response{protocol=http/1.1,code=200,message=OK,url=http://192.168.1.13:9100/auth/Login?q=admin4j}
```

再 Springboot 中使用

maven 引入

```
<dependency>
    <groupId>io.github.admin4j</groupId>
    <artifactId>common-http-starter</artifactId>
    <version>0.4.0</version>
</dependency>
```

最新版查询 io.github.admin4j:common-http-starter

spring 版可以对 OkHttp 进行个性化配置配置详见

```
public class HttpConfig {

    /**
     * 日志等级
     */
    private HttpLoggingInterceptor.Level loggLevel = HttpLoggingInterceptor.Level.BODY;

    /**
     * 读取超时时间，秒
     */
    private long readTimeout = 30;

    /**
     * 链接超时时间
     */
    private long connectTimeout = 30;

    private boolean followRedirects = false;

    /**
     * 最大的连接数
     */
    private int maxIdleConnections = 5;

    /**
     * 最大的keepAlive 时间 秒
     */
    private long keepAliveDuration = 5;

    private String userAgent = "OKHTTP";

    /**
     * 是否支持cookie
     */
    private boolean cookie = false;

    private ProxyConfig proxy;

    @Data
    public static class ProxyConfig {

        private Proxy.Type type = Proxy.Type.HTTP;

        private String host;

        private Integer port = 80;

        private String userName;

        private String password;

    }

}
```

## 如何快速封装外部接口

以实体项目为例，封装 ebay接口

```
public class EbayClient extends ApiJsonClient {

    /**
     * 店铺配置
     *
     * @param storeId
     */
    public EbayClient(Long storeId) {

        //TODO 获取店铺相关配置

        Map<String, String> config = new HashMap<>();

        basePath = "https://api.ebay.com";

        defaultHeaderMap.put("Authorization", "Bearer " + config.get("accessToken"));

        defaultHeaderMap.put("X-EBAY-C-MARKETPLACE-ID", config.get("marketplaceId"));

    }

}
```

EbayClient 封装ebay api请求 基础类

```

/**
 * ebay 库存相关api
 * @author andanyang
 */

public class EbayInventoryClient extends EbayClient {

    /**
     * 店铺配置
     *
     * @param storeId
     */

    public EbayInventoryClient(Long storeId) {

        super(storeId);
    }

    /**
     * 库存列表
     *
     * @param limit
     * @param offset
     * @return
     * @throws IOException
     */

    public JSONObject inventoryItem(Integer limit, Integer offset) throws IOException {

        Map<String, Object> queryMap = new HashMap(2);

        queryMap.put("limit", limit);

        queryMap.put("offset", offset);

        return get("/sell/inventory/v1/inventory_item", queryMap);
    }
}

```

EbayInventoryClient 封装ebay 库存 api请求使用

```

EbayInventoryClient ebayInventoryClient=new EbayInventoryClient(1L);

JSONObject jsonObject=ebayInventoryClient.inventoryItem(0,10);

/**
 * 订单相关api
 * @author andanyang
 */

public class EbayOrderClient extends EbayClient {

    /**
     * 店铺配置
     *
     * @param storeId
     */

    public EbayOrderClient(Long storeId) {

        super(storeId);
    }

    /**
     * 订单列表
     *
     * @param beginTime
     * @param endTime
     * @param limit
     * @param offset
     * @return
     */

    public JSONObject orders(String beginTime, String endTime, int limit, int offset) {

        final String path = "/sell/fulfillment/v1/order";

        String filter = MessageFormat.format("lastmodifieddate:[{0}..{1}]", beginTime, endTime);

        //

        Map<String, Object> queryMap = new HashMap<>(8);

        queryMap.put("filter", filter);

        queryMap.put("limit", limit);

        queryMap.put("offset", offset);
    }
}

```

```
        return get("/sell/inventory/v1/inventory_item", queryMap);
    }
}
```

库存相关的使用 `EbayInventoryClient` ,订单相关的使用 `EbayOrderClient` ,是不是很清晰

源码位置 `github.com/admin4j/common-http`

欢迎加入我的知识星球，一起探讨架构，交流源码。加入方式，**长按下方二维码噢：**



已在知识星球更新源码解析如下：

01. Dubbo 面试题	00. 精尽学习指南 —— 路线
02. Netty 面试题	01. Dubbo 学习指南
03. Spring 面试题	02. Netty 学习指南
04. Spring MVC 面试题	03. Spring 学习指南
05. Spring Boot 面试题	04. Spring MVC 学习指南
06. Spring Cloud 面试题	05. Spring Boot 学习指南
07. MyBatis 面试题	06. Spring Cloud 学习指南
08. 消息队列面试题	06. Spring Cloud Alibaba 学习指南
09. RocketMQ 面试题	07. MyBatis 学习指南
10. RabbitMQ 面试题	08. Hiberante 学习指南
11. Kafka 面试题	09. RocketMQ 学习指南
12. 缓存面试题	10. RabbitMQ 学习指南
13. Redis 面试题	11. Kafka 学习指南
14. MySQL 面试题	12. Redis 学习指南
15. 【分库分表】面试题	13. MySQL 学习指南
16. 【分布式事务】面试题	14. MongoDB 学习指南
17. Elasticsearch 面试题	15. Elasticsearch 学习指南
18. MongoDB 面试题	16. 设计模式学习指南
19. 设计模式面试题	17. Java 【基础】学习指南
20. Java 【基础】面试题	18. Java 【并发】学习指南
21. Java 【集合】面试题	19. Java 【虚拟机】学习指南
22. Java 【并发】面试题	21. Linux 学习指南
23. Java 【虚拟机】面试题	22. 数据结构与算法学习指南
24. Linux 面试题	23. 计算机网络学习指南
25. Git 面试题	24. Maven 学习指南
26. 计算机网络面试题	25. Jenkins 学习指南
27. Maven 面试题	26. Git 学习指南
28. Jenkins 面试题	27. IntelliJ IDEA 学习指南
29. Zookeeper 面试题	28. Docker 学习指南
30. Nginx 面试题	29. Kubernetes 学习指南
31. 数据结构与算法面试题	30. Zookeeper 学习指南
	31. Nginx 学习指南
	32. 任务调度学习指南
	33. React 学习指南
	34. Vue 学习指南



