0719项目梳理-Apollo&DingTalk接口

Apollo namespace

Namespace是配置项的集合，类似于一个配置文件的概念。Apollo在创建项目的时候，都会默认创建一个“application”的Namespace，但一个应用可以对应多个namespace。

客户端获取“application” Namespace的代码如下：

```
Config config = ConfigService.getAppConfig();
```

客户端获取非“application” Namespace的代码如下：

```
Config config = ConfigService.getConfig(namespaceName);
```

Namespace类型

Namespace类型有三种：

- 私有类型
- 公共类型
- 关联类型（继承类型）

私有类型

私有类型的Namespace具有private权限。例如上文提到的“application” Namespace就是私有类型。

公共类型

公共类型的Namespace具有public权限。公共类型的Namespace相当于游离于应用之外的配置，且通过Namespace的名称去标识公共Namespace，所以公共的Namespace的名称必须全局唯一。

使用场景：

- 部门级别共享的配置
- 小组级别共享的配置
- 几个项目之间共享的配置
- 中间件客户端的配置

Apollo 获取配置/监听方式

获取配置k1:

```
Config appConfig = ConfigService.getAppConfig();
appConfig.getProperty("k1", null);
```

方式2: @Value注解获取值

```
@EnableApolloConfig(value = "application")
@ApolloJsonValue("${xxx:xxx}")       //自动解析json格式，填充变量
```

监听：

在客户端Namespace映射成一个Config对象。Namespace配置变更的监听器是注册在Config对象上。

```
Config appConfig = ConfigService.getAppConfig();
appConfig.addChangeListener(new ConfigChangeListener() {
  public void onChange(ConfigChangeEvent changeEvent) {
    //do something
  }
})
```

DingTalk发送信息接口调用

群机器人

安全设置：加签

把`timestamp+"\n"+`密钥当做签名字符串，使用HmacSHA256算法计算签名，然后进行Base64 encode，最后再把签名参数再进行urlEncode，得到最终的签名（需要使用UTF-8字符集）。

参数	说明
timestamp	当前时间戳，单位是毫秒，与请求调用时间误差不能超过1小时。
secret	密钥，机器人安全设置页面，加签一栏下面显示的SEC开头的字符串。

把 timestamp和第一步得到的签名值拼接到URL中。

```
https://oapi.dingtalk.com/robot/send?access_token=XXXXXX&timestamp=XXX&sign=XXX
```

得到机器人的Webhook地址，即可向指定群发送消息。

可以发送text、link、markdown格式消息。

微应用推送

请求方式：POST

请求地址：`https://oapi.dingtalk.com/topapi/message/corpconversation/asyncsend_v2`

两个参数，分别为Query和Body。具体内容见官方文档：

https://developers.dingtalk.com/document/app/asynchronous-sending-of-enterprise-session-messages

```
DingTalkClient client = new
DefaultDingTalkClient("https://oapi.dingtalk.com/topapi/message/corpconversation/asyncs
end_v2");        //new一个client对象
OapiMessageCorpconversationAsyncsendV2Request request = new
OapiMessageCorpconversationAsyncsendV2Request();
request.setAgentId(836390886L);
request.setUseridList("user123");
request.setToAllUser(true);
//初始化request，设置request基本的参数，接下来设置msg，填充消息内容。
...
OapiMessageCorpconversationAsyncsendV2Response rsp = client.execute(request,
access_token);              //传入参数request对象和token，获得返回对象，判断是否成功发送
```