添加应用角色

接口描述

```
1、添加应用角色
2、接口支持https协议
```

接口URL

```
http://api.polyv.net/live/v4/user/webapp-role/create
```

在线API调用

请求方式

```
POST
```

接口约束

1、接口同时支持HTTP 、HTTPS ， 建议使用HTTPS 确保接口安全，接口调用有频率限制，详细请查看

请求参数描述

参数名	必选	类型	说明
appId	true	String	账号appId【详见获取密钥】
timestamp	true	Long	当前13位毫秒级时间戳，3分钟内有效

参数名	必选	类型	说明
sign	true	String	签名，为32位大写的MD5值, 生成签名的appSecret密钥作为通信数据安全的关键信息，严禁保存在客户端直接使用，所有API都必须通过客户自己服务器中转调用POLYV服务器获取响应数据 【详见签名生成规则】
name	true	String	角色名称
desc	false	String	角色描述
roleType	true	String	角色类型(root:仅主账号可见， child:主账号和子账号都可见)
permissionIds	true	Array	权限ID列表，例如：[1,2,3]

示例

```
http://api.polyv.net/live/v4/user/webapp-role/create
```

请求体：

```
{
    "name": "测试角色",
    "desc": "这是一个测试角色",
    "roleType": "root",
    "permissionIds": [1, 2]
}
```

响应参数描述

参数名	类型	说明
code	Integer	状态码，与 http 状态码相同，用于确定基本的响应状态

参数名	类型	说明
status	String	响应结果，由业务决定，成功返回success，失败返回error
success	Boolean	是否成功响应
requestId	String	请求ID，每次请求生成的唯一的UUID，仅可用于排查、调试，不应该和业务挂上钩
error	Object	状态码非200时的错误信息【详见Error字段描述】
data	Object	请求成功返回空对象

Error参数描述

参数名	类型	说明
code	Integer	错误代码，用于确定具体的错误原因
desc	String	错误描述，与 error.code 对应

Java请求示例

快速接入基础代码请下载相关依赖源码， 点击下载源代码 ,下载后加入到自己的源码工程中即可。测试用例中的**HttpUtil.java 和 LiveSignUtil.java** 都包含在下载文件中。

> 强烈建议您使用直播Java SDK完成API的功能对接，直播Java SDK 对API调用逻辑、异常处理、数据签名、HTTP请求线程池进行了统一封装和优化。

```
private static final Logger log = LoggerFactory.getLogger(getClass());
/**
 * 添加应用角色
 * @throws IOException
 * @throws NoSuchAlgorithmException
 */
@Test
public void addRoleTest() throws IOException, NoSuchAlgorithmException {
    //公共参数,填写自己的实际参数
    String appId = super.appId;
    String appSecret = super.appSecret;
    String timestamp = String.valueOf(System.currentTimeMillis());

    //业务参数
    String url = "http://api.polyv.net/live/v4/user/webapp-role/create";

    //请求体参数
    Map<String, Object> requestBodyMap = new HashMap<>();
    requestBodyMap.put("name", "测试角色");
    requestBodyMap.put("desc", "这是一个测试角色");
    requestBodyMap.put("roleType", "root");
    requestBodyMap.put("permissionIds", Arrays.asList(1, 2));

    //http 调用逻辑
    Map<String, String> requestMap = new HashMap<>();
    requestMap.put("appId", appId);
    requestMap.put("timestamp", timestamp);

    requestMap.put("sign", LiveSignUtil.getSign(requestMap, appSecret));

    String response = HttpUtil.postJson(url, requestMap, JacksonUtil.writeAsString(requestBodyMap));
    log.info("测试添加应用角色成功：{}", response);
    //do somethings
}
```

响应示例

系统全局错误说明详见全局错误说明

成功示例

```
{
    "code": 200,
    "status": "success",
    "requestId": "3a05f8f5f6984232808d9e099f9ee938.3338.16995806199894399",
    "data": {},
    "success": true
}
```

异常示例

```
{
    "code": 400,
    "status": "error",
    "requestId": "d310b70bc329403f87f77f9203d50f89.128.16360831552223589",
    "error": {
        "code": 20001,
        "desc": "角色名称不能为空"
    },
    "success": false
}
```