芋道源码 2022-12-31

服务调用 Feign

`yudao-spring-boot-starter-rpc` 技术组件，基于 Feign 实现服务之间的调用。

> **为什么不使用 Dubbo 呢?**
>
> Feign 通用性更强，学习成本更低，对于绝大多数场景，都能够很好的满足需求。虽然 Dubbo 提供的性能更强，特性更全，但都是非必须的。
> 目前国内 95% 左右都是采用 Feign，而 Dubbo 的使用率只有 5% 左右。所以，我们也选择了 Feign。
> 如果你对 Feign 了解较少，可以阅读《芋道 Spring Cloud 声明式调用 Feign 入门》系统学习。

1. RPC 使用规约

本小节，我们来讲解下项目中 RPC 使用的规约。

1.1 API 前缀

API 使用 HTTP 协议，所有的 API 前缀，都以 `/rpc-api` 开头，方便做统一的全局处理。

1.2 API 权限

服务之间的调用，不需要进行权限校验，所以需要在每个服务的 SecurityConfiguration 权限配置类中，添加如下配置：

```
// RPC 服务的安全配置
registry.antMatchers(ApiConstants.PREFIX + "/**").permitAll();
```

1.3 API 全局返回

所有 API 接口返回使用 CommonResult 返回，和前端 RESTful API 保持统一。例如说：

```
public interface DeptApi {

    @GetMapping(PREFIX + "/get")
    @Operation(summary = "获得部门信息")
    @Parameter(name = "id", description = "部门编号", required = true, example =
```

```
    CommonResult<DeptRespDTO> getDept(@RequestParam("id") Long id);

}
```

1.4 用户传递

服务调用时，已经封装 Feign 将用户信息通过 HTTP 请求头 `login-user` 传递，通过 LoginUserRequestInterceptor 类实现。

这样，被调用服务，可以通过 SecurityFrameworkUtils 获取到用户信息，例如说：

- `#getLoginUser()` 方法，获取当前用户。
- `#getLoginUserId()` 方法，获取当前用户编号。

2. 如何定义一个 API 接口

本小节，我们来讲解下如何定义一个 API 接口。以 AdminUserApi 提供的 getUser 接口来举例子。

2.1 服务提供者

AdminUserApi 由 `system-server` 服务所提供。

2.1.1 ApiConstants

在 `yudao-module-system-api` 模块，创建 ApiConstants 类，定义 API 相关的枚举。代码如下：

```
public class ApiConstants {

    /**
     * 服务名
     *
     * 注意，需要保证和 spring.application.name 保持一致
     */
    public static final String NAME = "system-server";

    public static final String PREFIX = RpcConstants.RPC_API_PREFIX +  "/system";

    public static final String VERSION = "1.0.0";

}
```

2.1.2 AdminUserApi

在 `yudao-module-system-api` 模块，创建 AdminUserApi 类，定义 API 接口。代码如下：

```
@FeignClient(name = ApiConstants.NAME) // ① @FeignClient 注解
@Tag(name = "RPC 服务 - 管理员用户") // ② Swagger 接口文档
public interface AdminUserApi {

    String PREFIX = ApiConstants.PREFIX + "/user";

    @GetMapping(PREFIX + "/get") // ③ Spring MVC 接口注解
    @Operation(summary = "通过用户 ID 查询用户")  // ② Swagger 接口文档
    @Parameter(name = "id", description = "部门编号", required = true, example =
    CommonResult<AdminUserRespDTO> getUser(@RequestParam("id") Long id);

}
```

另外，需要创建 AdminUserRespDTO 类，定义用户 Response DTO。代码如下：

```
@Data
public class AdminUserRespDTO {

    /**
     * 用户ID
     */
    private Long id;
    /**
     * 用户昵称
     */
    private String nickname;
    /**
     * 帐号状态
     *
     * 枚举 {@link CommonStatusEnum}
     */
    private Integer status;

    /**
     * 部门ID
     */
    private Long deptId;
    /**
     * 岗位编号数组
     */
```

```
    private Set<Long> postIds;
    /**
     * 手机号码
     */
    private String mobile;

}
```

2.1.3 AdminUserRpcImpl

在 `yudao-module-system-biz` 模块，创建 AdminUserRpcImpl 类，实现 API 接口。代码如下：

```
@RestController // 提供 RESTful API 接口，给 Feign 调用
@Validated
public class AdminUserApiImpl implements AdminUserApi {

    @Resource
    private AdminUserService userService;

    @Override
    public CommonResult<AdminUserRespDTO> getUser(Long id) {
        AdminUserDO user = userService.getUser(id);
        return success(UserConvert.INSTANCE.convert4(user));
    }

}
```

2.2 服务消费者

`bpm-server` 服务，调用了 AdminUserApi 接口。

2.2.1 引入依赖

在 `yudao-module-bpm-biz` 模块的 `pom.xml` ，引入 `yudao-module-system-api` 模块的依赖。代码如下：

```
<dependency>
    <groupId>cn.iocoder.cloud</groupId>
    <artifactId>yudao-module-system-api</artifactId>
    <version>${revision}</version>
</dependency>
```

2.2.2 引用 API

在 `yudao-module-bpm-biz` 模块，创建 RpcConfiguration 配置类，注入 AdminUserApi 接口。代码如下：

```
@Configuration(proxyBeanMethods = false)
@EnableFeignClients(clients = {AdminUserApi.class.class})
public class RpcConfiguration {
}
```

2.2.3 调用 API

例如说，BpmTaskServiceImpl 调用了 AdminUserApi 接口，代码如下：

```
@Service
public class BpmTaskServiceImpl implements BpmTaskService {

    @Resource
    private AdminUserApi adminUserApi;

    @Override
    public void updateTaskExtAssign(Task task) {
        // ... 省略非关键代码
        AdminUserRespDTO startUser = adminUserApi.getUser(id).getCheckedData();
    }
}
```

← 服务网关 Spring Cloud Gateway 消息队列 RocketMQ→