

- 用户角色类型校验说明文档
  - 一、快速使用
    - 1.1 类级别注解（整个Controller统一要求）
    - 1.2 方法级别注解（覆盖类级别）
    - 1.3 多角色支持（OR关系）
    - 1.4 无注解（公开接口）
  - 二、工作原理
    - 2.1 执行流程
    - 2.2 注解优先级
    - 2.3 代码位置
  - 三、注意事项

# 用户角色类型校验说明文档

## 一、快速使用

### 1.1 类级别注解（整个Controller统一要求）

```
@RestController
@RequestMapping("/seeker")
@RequireUserType(UserType.SEEKER) // 整个Controller要求求职者身份
public class SeekerController {
    // 所有方法自动要求求职者身份
}
```

### 1.2 方法级别注解（覆盖类级别）

```
@RestController
@RequestMapping("/seeker")
@RequireUserType(UserType.SEEKER) // 类级别：默认求职者
public class SeekerController {

    @GetMapping("/online-resume")
    @RequireUserType(UserType.HR) // 方法级别：覆盖为HR
    public Result<?> getOnlineResume(@RequestParam Long userId) {
        // ...
    }
}
```

```
}
```

## 1.3 多角色支持（OR关系）

```
@RequireUserType({UserType.HR, UserType.ADMIN}) // HR或管理员都可以
```

## 1.4 无注解（公开接口）

```
// 没有@RequireUserType注解 = 所有已登录用户都可以访问
```

# 二、工作原理

## 2.1 执行流程

请求流程：

1. JwtAuthenticationFilter (Filter层)
  - 验证JWT token有效性
  - 检查token黑名单
  - 从token中提取claims (包含userType)
  - 将claims放入SecurityContext



2. UserTypeAspect (AOP层)
  - 检查@RequireUserType注解
  - 从SecurityContext读取userType (无需查库)
  - 验证用户角色是否匹配



3. Controller方法执行
  - 业务逻辑处理

JwtAuthenticationFilter (Filter层)  
↓ 验证JWT，提取userType到SecurityContext

UserTypeAspect (AOP层)  
↓ 检查@RequireUserType注解  
↓ 从SecurityContext读取userType (无需查库)  
↓ 验证角色是否匹配

Controller方法执行

## 2.2 注解优先级

- 方法级别注解 > 类级别注解
- 无注解 = 不验证角色 (只要JWT通过即可)

## 2.3 代码位置

- 切面实现: com.SmartHire.common.aspect.UserTypeAspect
- 注解定义: com.SmartHire.common.annotation.RequireUserType
- 角色枚举: com.SmartHire.common.enums.UserType

## 三、注意事项

1. Service层无需验证身份: AOP已在Controller层统一处理, Service层直接使用 `UserContext.getCurrentUserId()` 即可
2. 角色类型: `UserType.SEEKER(1)`、`UserType.HR(2)`、`UserType.ADMIN(3)`
3. 验证失败异常: 自动抛出 `USER_NOT_SEEKER`、`USER_NOT_HR` 或 `PERMISSION_DENIED`