JavaEE平台技术
面向切面编程

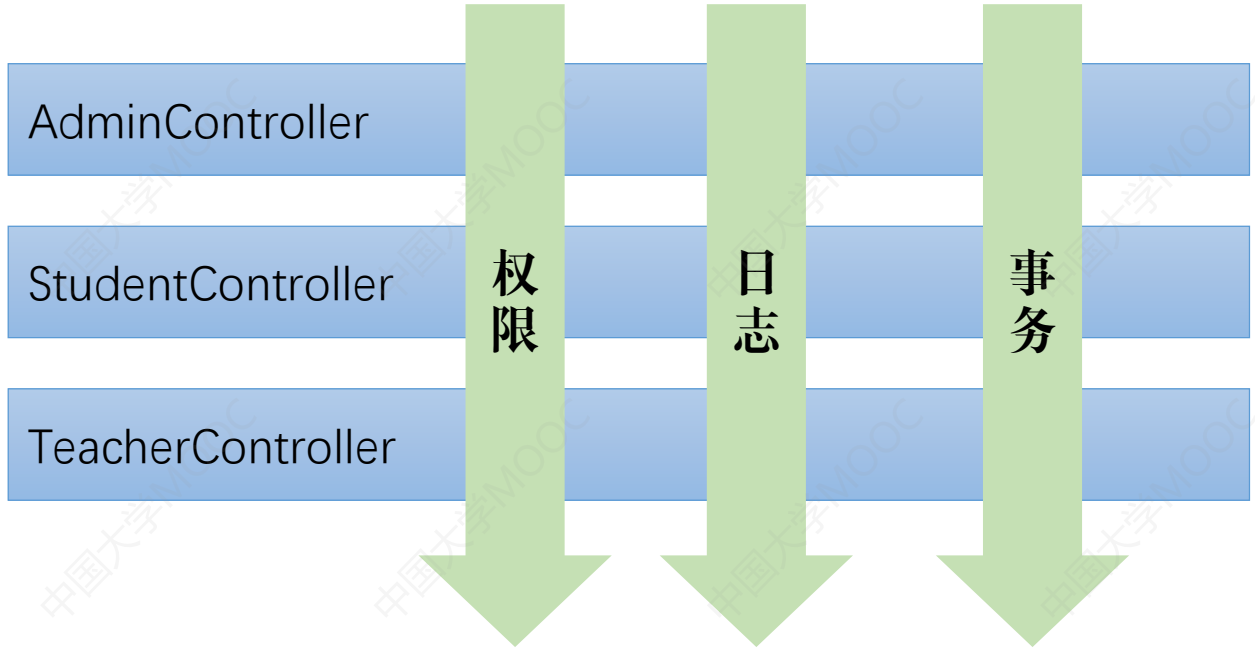
邱明 博士

厦门大学信息学院

mingqiu@xmu.edu.cn

1. AOP是什么

- 面向方面编程



1. AOP是什么

```
@GetMapping("{id}")
public Object getProductById(@PathVariable("id") Long id) {
    logger.debug("getProductById: id = {} " ,id);
    Object retObj = null;
    Product product = null;
    try {

        product = productService.retrieveProductByID(id, false);
        ProductRetVo productRetVo = new ProductRetVo(product);
        retObj = ResponseUtil.ok(productRetVo);
    }
    catch (BusinessException e){
        retObj = returnWithStatus(null, e);
    }
    return  retObj;
}
```

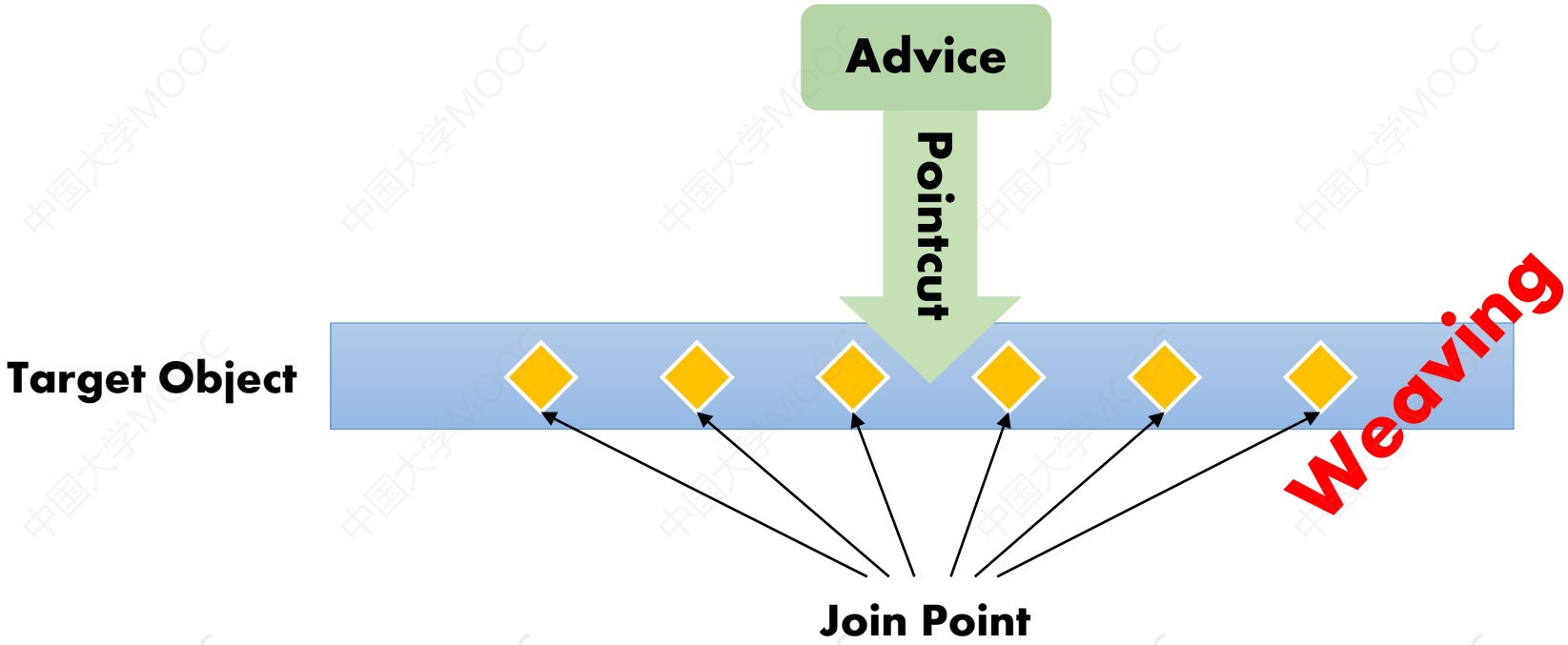
```
PostMapping("")
public Object createProduct(@RequestBody ProductVo productVo){
    Object retObj = null;
    try{
        Product product = productVo.createBo();
        User user = new User(Long.valueOf(1), "admin1");
        Product retProduct = productService.createProduct(product, user);
        ProductVo vo = new ProductVo(retProduct);
        retObj = new ResponseEntity(
                ResponseUtil.ok(vo),
                HttpStatus.CREATED);
    }
    catch (BusinessException e){
        retObj = returnWithStatus(null, e);
    }
    return  retObj;
}
```

```
@GetMapping("")
public Object searchProductByName(@RequestParam String name) {
    Object retObj = null;
    try{
        List<Product> productList = null;
        productList = productService.retrieveProductByName(name, false);
        if (null != productList) {
            List<ProductRetVo> data = new ArrayList<>(productList.size());
            for (Product bo : productList) {
                data.add(new ProductRetVo(bo));
            }
            retObj = ResponseUtil.ok(data);
        } else {
            retObj = ResponseUtil.ok();
        }
    }
    catch (BusinessException e){
        retObj = returnWithStatus(null, e);
    }
    return  retObj;
}
```

{"errmsg":"成功","data":{"id":5559,"name":"水果糖","originalPrice":100,"weight":807,"barcode":"1234455","unit":"盒","originPlace":"长沙","gmtCreate":"2022-10-26T17:59:04","creator":{"id":1,"name":"13088admin"}},"errno":0}

1. AOP是什么

- 专有名词



2.定义Pointcut

- Spring采用AspectJ的定义符定义Pointcut

定义符	描述
args()	Join point方法的参数是某些特定类型
@args()	Join point方法的参数用特定注解标注
execution()	Join point方法名称满足特定条件
@annotation	Joint point方法用特定注解标注
within	Joint point方法只在特定范围内
@within	Joint point方法在有特定标注的类里
target	Joint point方法是特定类的方法
@target	Joint point方法的类使用了特定注解

2.定义Pointcut

execution(* concert.Performance.perform(..))&&within(concert.*)

在concert包内调用

3. 定义Aspect

- 五种Advice
  - @Before – 在Joint Point执行之前
  - @After – 在Joint Point执行之后，无论是否成功执行
  - @AfterReturning – 在Joint Point成功执行之后
  - @AfterThrowing – 在Joint Point抛出异常之后
  - @Around – 在Joint Point运行之前和之后

3. 定义Aspect

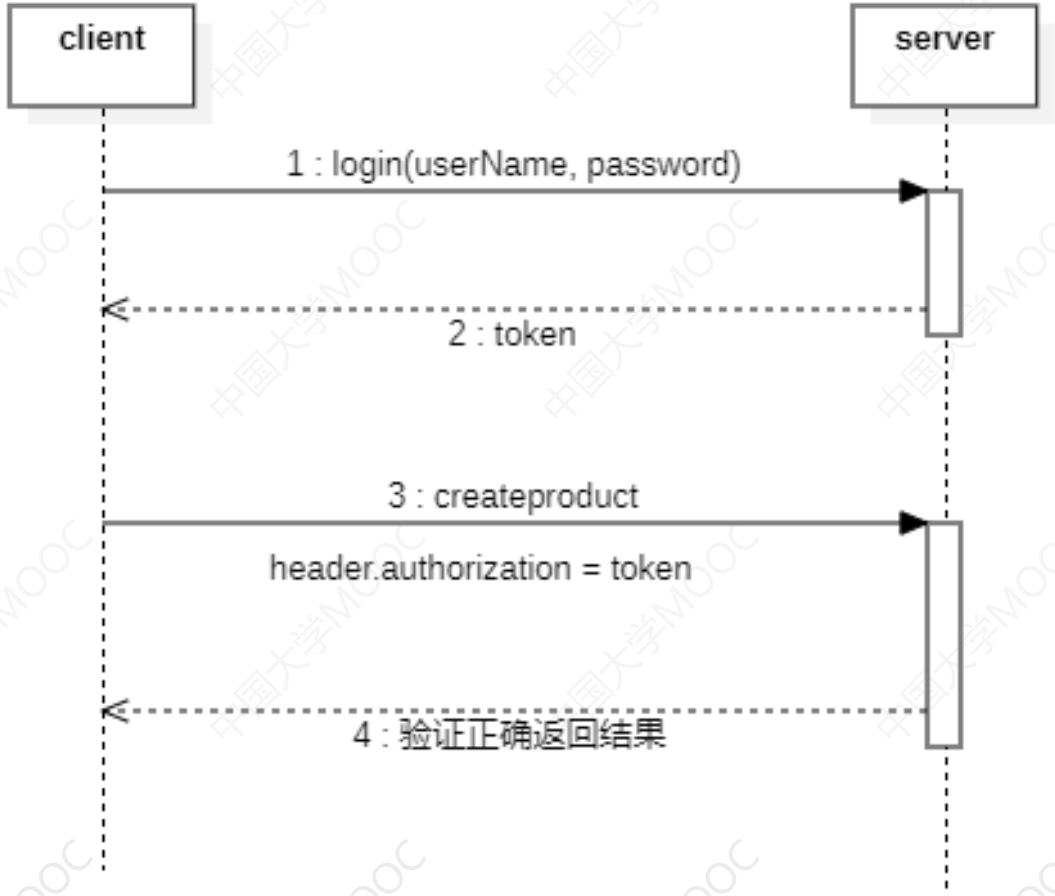
```java
@Aspect
public class CommonPointCuts {

    @Pointcut("execution(public cn.edu.xmu.javaee.core.util.ReturnObject cn.edu.xmu.javaee.goodsdemo.controller..*.*(..))")
    public void controllers() {
    }

    @Pointcut("@annotation(cn.edu.xmu.javaee.core.aop.Audit)")
    public void auditAnnotation() {
    }
}
```

```java
@Aspect
@Component
@Order(100)
public class ResponseAspect {

    private final Logger logger = LoggerFactory.getLogger(ResponseAspect.class);

    @Value("${goodsdemo.result-page-size.max}")
    private int max_page_size;

    @Value("${goodsdemo.result-page-size.default}")
    private int default_page_size;

    /**
     * 所有返回值为ReturnObject的Controller
     *
     * @param jp
     * @return
     * @throws Throwable
     */
    @Around("cn.edu.xmu.javaee.core.aop.CommonPointCuts.controllers()")
    public Object doAround(ProceedingJoinPoint jp) throws Throwable {
        ReturnNo code = ReturnNo.OK;
        ReturnObject retVal = null;

        MethodSignature ms = (MethodSignature) jp.getSignature();
        HttpServletRequest request = ((ServletRequestAttributes) RequestContextHolder.getRequestAttributes()).getRequest();
        HttpServletResponse response = ((ServletRequestAttributes) RequestContextHolder.getRequestAttributes()).getResponse();

        String[] paramNames = ms.getParameterNames();
        Object[] args = jp.getArgs();

        checkPageLimit(request, paramNames, args);

        try {
            Object obj = jp.proceed();
            retVal = (ReturnObject) obj;
        } catch (BusinessException exception) {
            logger.debug("doAround: BusinessException，  errno = {}", exception.getErrno());
            retVal = new ReturnObject(exception.getErrno(), exception.getMessage());
        }

        code = retVal.getCode();
        logger.debug("doAround: jp = {}, code = {}", jp.getSignature().getName(), code);
        changeHttpStatus(code, response);

        return retVal;
    }
```

4. AOP的例子

- Controller的输入合法性验证
- page和pageSize的合法性验证
- 返回值的封装
- 登录用户的JWT处理

4. AOP的例子


client
server
1 : login(userName, password)
2 : token
3 : createproduct
header.authorization = token
4 : 验证正确返回结果