# Java编码规范

[转至元数据结尾](http://k.2dfire.net/pages/viewpage.action?pageId=12353640" \l "page-metadata-end)[转至元数据起始](http://k.2dfire.net/pages/viewpage.action?pageId=12353640" \l "page-metadata-start)

## 文件命名规范

Java文件和变量必须为字母、数字或下划线（特殊情况下、比如与系统关键字冲突）组成并遵循Java命名规范。

不允许使用特殊字符、中文和其他字符类型。

Java源文件命名必须为首字母大写并遵循骆驼命名法，例：ProjectTest.java。

XML系统配置文件必须全部为小写并已"-"划线先为单词分隔符，例：project-test.xml。

.properties资源文件的资源文件必须为首字母为小写并遵循骆驼命名法，例：projectTest.properties。

.vm、.jsp、.html、.htm等页面文件必须为首字母为小写并遵循骆驼命名法，例：projectTest.vm、projectTest.jsp。

## Java源文件格式规范

文件长度应少于2000行。

package声明是java源文件的第一个非注释行，声明当前源文件中的类或接口所属的package名称。

import声明所有用的的类，不声明没有使用到的类，尽量避免在源程序中使用完整类名，不使用通配符一次引入多个类，必要时写明注释。

类和接口声明，每个Java源文件原则上只允许一个类或接口的声明，并写明类注释，类和接口里德声明一般遵循以下原则：

先声明静态变量，再声明非静态成员

先声明变量，再声明方法

类成员依次按照public、protected、默认的、private级别声明

同名重载方法按照参数的复杂性排序，先简单后复杂

所有类成员都必须写明注释描述它的作用或逻辑

## 标识符命名

Packages必须全部为小写字母组成，统一前缀为：com.alibaba.dw，例：com.alibaba.dw.project.common.test。

Classes必须为单词首字母大写并遵循骆驼命名法，使用完整单词（除有更广泛的使用，像URL、HTML）,例：ProjectTest、URL。

Interfaces同上Classes命名规则。

Methods必须首字母小写并遵循骆驼命名法，例：test()、testMethods()。约定插入、更新的命名分别为insertXxxx、updateXxxx。

Variables变量首字母小写并遵循骆驼命名法，除变量以外的所有实例、类和类常量首字母大写并遵循骆驼命名法，不允许使用中文、特殊字符（、$、%）等其他非英语单词字符命名。Constants声明的类常量和ansi常量全部大写，单词使用下划线（'\_'）分隔。

变量名称应该易于记忆，即能够指出其用途。尽量避免单个字符的变量名，除非是一次性的临时变量。临时变量通常被取名为i，j，k。

## 单元测试

所有新增service与dao方法必须有单元测试。

## 注释规范

类文档注释

/\*

\* Copyright 2015 2dfire.com All right reserved. This software is the

\* confidential and proprietary information of 2dfire.com ("Confidential

\* Information"). You shall not disclose such Confidential Information and shall

\* use it only in accordance with the terms of the license agreement you entered

\* into with 2dfire.com.

\*/

类注释

/\*\*

\* 这是一个测试类。<br/>

\* 具体如下：<br/>

\* 1. xxx

\* 2. yyy

\* @author ???

\*/

方法注释

成员变量注释

private String loginName; //登录名称

或/\*\* 登录名称 \*/private String loginName;

注意:API定义请使用/\*\* xxx \*/方式注释，方便生成javaDoc

常量注释

/\*\* 成功 \*/public static final String SUCCESS = "1000";

方法局部信息注释

public void test(){

String a = x; //xxx

或者

//xxx

String b = x;}

ibatis及spring等xml配置文件注释

<!-- 这是一段xml类型的注释 -->

## Java项目基础规范

所有Velocity,JS,Java,XML,Properties代码等文本必须使用UTF-8格式

页面所有编码必须为UTF-8格式

spring配置文件中的变量部分，都抽取为.properties文件，然后通过spring的placeholder机制进行关联。auto-conf只需要生成相应的.properties文件即可。比如：  
auto-conf目录下的common.properties.vm模板文件

#databasemy.database.username=${my\_database\_username}my.database.driver.password.encrypted=${my\_database\_driver\_password\_encrypted}my..database.driver.url=${my\_database\_driver\_url}#dubbomy.dubbo.registry.address=${my\_dubbo\_registry\_address}my.service.version=${my\_service\_version}

auto-conf.xml

<generate template="common.properties.vm" destfile="config/common.properties" charset="utf8" />

spring的xml文件使用placeholder方式

<!-- 声明placeholder并指定properties文件地址 --><bean class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">

<property name="location">

<value>classpath\*:/config/\*.properties</value>

</property></bean><!-- 使用${xxx}方式引用.properties文件的内容 --><dubbo:registry address="${my.dubbo.registry.address}" />

## 分层命名

业务模块使用表示层-业务层-数据访问层三层结构 即 Controller-Service-Dao

业务层/数据访问层所有访问都使用接口加实现的方式,各层调用仅调用接口

数据访问层接口名和实现类名必须使用xxxDao和IBatisxxxDaoImpl命名方式,在Spring的Bean定义中使用xxxDao的命名规则，对应的传输对象以PO结尾

业务层层接口名和实现类名必须使用xxxService和xxxServiceImpl命名方式,在Spring的Bean定义中使用xxxService的命名规则,以保证事务的可拦截性，对应的传输对象以BO结尾

1. 服务层如果需要高频调用，可以将client包做成富客户端，即在client包中直接读取缓存数据，当未命中时再调用Dubbo服务

所有涉及到数据库修改操作的方法名必须使用insert\*, regist\*, update\*, modify\*, delete\*, remove\*,以保证事务的可拦截性

各模块编写时注意Log日志的书写,Log统一使用slf4j,使用方法参考提供的范例

import org.slf4j.Logger;import org.slf4j.LoggerFactory;

import com.aliyun.basicservicecenter.demo.services.DemoService;

public class DemoServiceImpl implements DemoService {

private static final Logger log = LoggerFactory.getLogger(DemoServiceImpl.class);

@Override

public String hello(String name) {

if (log.isInfoEnabled()) {

log.info("test");

}

return "hello" + name;

}

}

每个方法的日志必须为:入口info,中间处理debug,异常处理error,结束info方式,使用方法参考提供的范例,注意使用if(log.isXXX()) log.XXX(...........)方式

所有Class所导入(import)的包(package)必须为最小范围,严禁出现 import xxx.xxxx.\*,即导入范围必须明确至Class单位

代码GIT提交和管理范围仅限于上述目录和相关文件，.settings、target、.project和.classpath等Eclipse配置文件不包含在内

1. 线上应用POM只能依赖正式包，以避免SNAPSHOT覆盖冲突