Java 软件开发规范

机器学习与生物信息学课题组

2016.08.31

一、概述

1.1 目的

本规范目的是为了课题组能以标准、规范、统一的编程风格进行 Java 代码的开发工作。提高程序的可读性、可修改性和可维护性，方便团队的交流。

代码规范的重要性:

- 一个软件的生命周期中，有 80%都在进行维护；
- 很难有软件在整个生命周期中，都是由原来的作者进行维护的；
- 编码规范提高了软件的可读性，并且帮助软件工程师更加快速彻底地理解新的代码；
- 要想把你的源代码作为产品来发布，就需要保证它像你创作的其他作品那样被很好地封装，并且清理干净了。

针对我们学校环境来说，师兄师姐毕业，师弟师妹入学，如果我们没有统一的完成的编程规范，每个人按照自己的编程风格和编程方法进行程序开发，在毕业之后很少有人能读懂他的程序，对于后期的维护来说是一件非常头疼的问题，所以我们非常有必要总结出一套适合我们自己的 Java 开发规范。

1.2 工具

- IDE：推荐 Intellij IDE Community 版，免费
- JDK：>= 1.8
- 共享：Github

1.3 更新

本规范为内部共享资源，如果大家有好的建议在这里讨论，我直接在该文件中进行添加和修改更新。

二、代码整体组织和风格

一个文件应该是由被空行分隔开的代码块所组成，并且每个块被特定的注释标识出来。每个文件不应该超过 1000 行(包括注释说明)。

2.1 Java源文件

2.1.1 Package 包

- 包信息和引用语句之间要与一个空行
- 引用语句之间不在同一个包中的引用也要有一个空行

例如：

```
package edu.tju.nuka;

import java.text.NumberFormat;
import java.text.SimpleDateFormat;
import java.util.*;
```

2.1.2 Class 类

- 类的注释：

版权说明、版本号、生成日期、作者、目的/功能、主要函数及其功能、修改日志等。

例如：

```
/**********************************************************
Author:        //作者姓名
Description:      // 模块基本描述
Date:          //更新时间
Function List:    // 主要函数及其功能
**********************************************************/
```

作者姓名将使接手者记住你的名字，铭记你的成果；模块的基本描述使他人更快明白你的程序功能，函数列表则让他们更好调用你的代码。

这段源文件的头注释比较标准，当然，并不局限于此格式，但上述信息建议要包含在内。

2.1.3 构造函数和方法

构造函数和方法:构造函数也是方法但是它没有返回值，本规范规定，先写构造函数在写其他的方法。方法也要加上注释，说明这个方法实现的功能等信息，例如

```
/**
* @Description : TODO 方法功能描述
* @param args
* @author: yan  2009-4-6
*/
```

方法的参数以及返回类型一定要详细填写，参数包括参数要求的类型，以及参数的说明；返回值包括返回的类型以及返回之后的数据结构等信息。

2.1.4 函数体长度

前面已经提到整个文件不应该超过 1000 行，而作为函数体也有相应的要求，为了方便程序阅读和理解，函数体应该尽量控制在 100 行左右(只是看到有人这样写，具体多少合适我也不是很清楚)。避免出现大类和大方法，对于程序的维护是相当困难的。所以我们提倡小方法以及方法的重用性操作。

2.2 缩进

缩进使得程序看起来清晰和结构化，本规范规定采用 4 个空格作为缩进单位。Eclipse 默认的 TAB 缩进为 4 个空格，不同的 IDE 工具可能不太一样，需要进行设置 TAB 代表的空格。但是如果程序层次缩进过多的话使得程序易读性差，在遇到这种情况时应将部分子功能提取出来作为单独的方法。

2.3 行宽

本规范规定每一行的最长宽度为 80 字符，因为超过了这个宽度之后有些工具不能很好的处理，就算是能够处理，我们也不要超过这个宽度，这也是程序可读性的一个要求。

2.4 换行

在一个“,”后面断行

在一个操作符前面换行，操作符放在新行之首(确保突出操作符)

在一个操作符前面换行，操作符放在新行之首(确保突出操作符)

if 表达式尽量使用 8 个空格换行操作，因为 4 个空格换行操作使得函数体不容易辨认。

拆分出的新行要适当进行缩进，使得排版整齐，可读性强。

```
//DON'T USE THIS INDENTATION
if ((condition1 && condition2)
    || (condition3 && condition4)
    ||!(condition5 && condition6)) { //BAD WRAPS
    doSomethingAboutIt();             //函数体不容易辨认
}

//USE THIS INDENTATION INSTEAD
if ((condition1 && condition2)
        || (condition3 && condition4)
        ||!(condition5 && condition6)) { //BAD WRAPS
    doSomethingAboutIt();             //函数体不容易辨认
}
```

三、命名规范

规范化的命名可以使程序易于阅读和理解，不仅仅方便自己日后看程序方便同时也方便他人进行代码的维护工作。

- 尽量少用缩写形式命名
- 命名中尽量少出现或者不出现数字，当然用 2 代表 to,或者 4 代表 for 是可以的。
- 避免使用相似或者仅仅在大小写上区分的名字

标识符类型	命名规则	例子
包(package)	1、一个唯一包名的前缀总是全部小写的 ASCII 字母并且是一个顶级域名，通常是 com,edu,govnet,org 2、edu.bju 基本包,所有包和文件都在基本包下 3、包名称按照如下规则组成 【基本包】.【项目名】. 【模块名】.【子模块名】..	edu.tju.nuka
类(Classes)	采用大小写混合的方式，每个单词的首字母大写，所命名称要能反映出类的功能	class Raster; class ImageSprite;
接口(Interfaces)	与类命名规则一样	
方法(Methods)	采用大小写混合的方式，第一个单词的首字母小写，其后单词的首字母大写。第一个单词常常采用具有具有动作色彩的动词	runFast(); getBackground();
变量(Variables)	第一个单词的首字母小写，其后单词的首字母大写；对于不易识别类型的变量加上类型标示 (Hashtable keyEleHashtable; List allStudentsList)	Boolean isXXX Boolean hasXXX String strXXX
实例变(Instance Variables)）	大小写规则和变量名相似，除了前面需要一个下划线	String _name
常量(Constants)	应该全部大写，单词间用下划线隔开	MAX_WIDTH

注意：在程序开发过程中避免出现莫名的数字，既然是一个常量，那就给定义为常量来操作，不要出现不知道是什么意思的常量，使得程序变得很神秘。