编码规范

Code Conventions

CONTENT

- 1. why
- 2. how
- 3. QA
- 4. reference

1.WHY

- 一个软件的生命周期中，80%的花费在于维护
- 几乎没有任何一个软件，在其整个生命周期中，均由最初的开发人员来维护
- 编码规范可以改善软件的可读性，可以让程序员尽快而彻底地理解新的代码
- 如果你将源码作为产品发布，就需要确任它是否被很好的打包并且清晰无误，一如你已构建的其它任何产品
- 良好的习惯能避免不必要的“坑”,提高效率
- 降低合作成本

指导原则

- 易读性
- 约定优于配置
- 统一风格
- 一致性
- 遵循最佳实践

一份编码规范会包含哪些内容

- 文件的组织
- 排版、缩进、空格、空行、编码
- 注释与声明
- 语法语句
- 命名
- 惯例、推荐实践

2.1 文件的组织

- maven 文件约定(https://maven.apache.org/guides/introduction/introduction-to-the-standard-directory-layout.html)
- github项目结构
- README,LISENCE
- angular

关于空白符

- 一致性
- 不同环境的显示配置不一致(eg: windows linux mac IDEs)
- 一些版本控制工具对空白字符的处理（空白比较、换行符替换等）

- 推荐的配置：使用4个空格作为默认缩进单位（IDE配置tab键为1个缩进单位）

命名

- 风格一致
- 功能与名字（xxxUtils, xxxService, xxxImpl, AbstractXXX, xxxFactory, xxxWraper）
- 含义与统一(add/new/create/insert select/query/get/find)

TIPS

- 良好的单词拼写，尽量不使用拼音、不必要的缩写,避免错别字
- 避免不必要的注释，如果注释过多考虑是不是逻辑、代码、实现可以改进
- 使用log而不是输出到控制台
- 单一职责设计，合理的变量生命周期
- 学习合理的接口设计
- 避免过多的缩进嵌套

TIPS

- 配置并使用IDE的自动格式化工具、代码风格检查工具
- 使用代码生成工具
- 学习设计模式
- 执行代码审核 / 评审
- 向优秀的开源项目学习

推荐阅读：

- Code Conventions for the Java TM Programming Language (http://www.oracle.com/technetwork/java/codeconvtoc-136057.html 翻译：http://morningspace.51.net/resource/javacodeconv.html)
- Google Java Style Guide(https://google.github.io/styleguide/javaguide.html)
- Google Style Guide(https://github.com/google/styleguide)
- Angular Style Guide（https://angular.cn/docs/ts/latest/guide/style-guide.html)
- Airbnb JavaScript Style Guide(https://github.com/airbnb/javascript)
- Airbnb CSS / Sass Styleguide(https://github.com/airbnb/css)

THX