Java 项目开发规范

前言

为了使软件开发过程有章可循，保证软件质量，加强开发管理。

目录

文件规范

- 文件必须使用UTF-8编码
- License或者copyright声明信息。（如果需要声明）

命名规范

- 项目命名规范
    - 项目编号：项目英文/中文拼音名称_开发组编号_序列号 (序列号由3位数字组成，不足的用'0'补齐)。
    - 项目文档：项目英文/中文拼音名称_文档名称_序列号_编写人名称/编号_日期
- 数据库命名规范
    - 数据库表命名均遵循以下规范：
        - 类型名*模块名*存储信息名词(多个单词用下划线分隔)，全部小写，例如：fun_mail_message。(fun前缀表示功能，com前缀表示组件，sec前缀表示权限)
    - 数据库字段命名遵循以下规范：
        - 存储信息名词(多个单词用下划线分隔)，全部小写，例如：message_id。
- Java源码命名规范
    - 目录 的命名

- 项目resty-example:
  1. src/main/java 源码目录
  2. src/main/resources 资源文件目录
  3. src/main/webapp web文件目录
  4. src/test/java 测试源码目录
  5. src/test/resources 测试资源文件目录(如果不是特例可以直接读取main下的资源文件)

- Package 的命名
  - Package 的名字应该都是由一个小写单词组成，包的命名一般都是按照域名+公司名+项目名+具体反映包内容的名字。例如：cn.dreampie.xxx。
  - 此外，对于包名我们做如下约定:
    1. 工具函数类包名前缀为 .utils
    2. Servlet类包名前缀为 .servlet
    3. 逻辑功能类包前缀为 .fun
    4. 具体的逻辑功能内容前缀为功能的缩写 .fun.order

- Class 的命名
  - 类名是个名词，采用大小写混合的方式，每个单词的首字母大写。尽量使你的类名简洁而富于描述。例如：DataFile或InfoParser。使用完整单词，避免缩写词(除非该缩写词被更广泛使用，像URL，HTML)。 异常类最后加上「Exception」（例：SQLException）。 另外，接口的命名和类的命名大体相同，只是接口一般用形容词。例如：Runnable。

- Class 变量的命名
  - 变量的名字必须用一个小写字母开头。后面的单词用大写字母开头，变量名一般为动词。变量名不应以下划线或美元符号开头，尽管这在语法上是允许的。 例如：debug 或 inputFileSize。

- Static Final 变量的命名
  - Static Final 变量的名字应该都大写，并且指出完整含义，多个单词之间用下划线分隔。例如：MAX_UPLOAD_FILE_SIZE=1024。

- 参数的命名
  - 参数的名字必须和变量的命名规范一致。使用有意义的参数命名，如果可能的话，使用和要赋值的字段一样的名字：

```
setSize(int size){
    this.size = size;
}
```

- 数组的命名

        - 数组应该总是用下面的方式来命名：

```
byte[] buffer;
//而不是：
byte buffer[];
```

    - 方法的命名

        - 方法名以小写字母动词开头，后续的各单词开头字母大写。例如：inputFile()。
        - Debug用的方法用「debug」作为前缀，与其他方法区别开。例如：debugMethod()。
        - JavaBeans的相关函数、设置成员变量的方法用“set” + 成员变量名表示，读取成员变量的方法用“get” + 成员变量名表示。但是，对于布尔值（boolean）读取成员变量的方法用“is” + 成员变量名表示。

代码规范

- 代码

    - 文件头声明

        - 源文件的头部需要一个history段，对于每次对源文件的重大改动，都需要在history段中注明。该段定义在package和import之间，例如：

```
/**********************************************
* 2015-01-29 Biz 创建文件
*
* 2015-02-19 kevin 增加xx功能
*
* 2015-03-01 ben 增加xx功能
**********************************************/
```

    - import顺序(idea 自带优化)

        1. jdk标准包
        2. java扩展包（例如servlet，javamail，jce等）
        3. 使用的外部库的包（例如xml parser）
        4. 使用的项目的公共包
        5. 使用的模块的其他包
        6. 每一类import后面加一个换行。

    - 代码块书写格式

```
if (true){
  //body 必须包含大括号  即使只有一行代码 避免后期增加代码时出现错误
}
```

- 注释
  - 类注释

```
/**
 * @author wangrenhui
 * @date 2015-04-12
 * @what 用户实体类
 */
```

  - 关于缩进(重要)
    - Tab size 2
    - Indent 2
    - Continuation indent 4

- SQL语句
  - 代码中书写的SQL语句要求SQL关键字全部大写，表名和字段名小写。例如:

```
SELECT user_id, name FROM account
WHERE user_id > ? AND depart = ? ORDER BY name
```

- Java注释有三种类型，分为单行注释(//)，多行注释(/* */)和文档注释(/** */)。

1. 注释应放在代码的上方或右方，不能放在其下方。
2. 全局变量要有较详细的注释，包括对其功能、取值范围、哪些函数或过程存取它以 及存取时注意事项等的说明。
3. 在每个源文件的头部要有必要的注释信息，包括：文件名；版本号；作者；生成日期；模块功能描述（如功能、主要算法、内部各部分之间的关系、该文件与其它文件关系等）；主要函数或过程清单及本文件历史修改记录等。
4. public 和 protected的成员变量和方法必须写javadoc注释。超过1句以上的注释使用中文书写。对于代码多于10行的private方法也要写javadoc注释。

对于代码中的逻辑分支或循环条件需要书写注释，例如：

```
if (some condition){
  //符合某个条件，应该这样处理
}else{
  //否则应该那样处理
}
```

5. java的类和方法。都需要用文档注释，需要写清楚说明，版本，参数，返回值，作者，创建日期，修改日期。例：

```
/**
  * @author wangrenhui
  * @date 2015-02-13
  * @what 获取实体属性排序数组
  * @param entityClass 对象类型
  * @param fieldNames 属性名数组
  * @return 排序数组
  */
```

Maven使用规范

- 尽量让所有的maven项目使用相同的父级，如：resty 在父级配置通用内容：

```
<!--快照代码仓库-->
<repositories>
  <repository>
    <id>ossrh</id>
    <url>https://oss.sonatype.org/content/repositories/snapshots</url>
    <releases>
      <enabled>false</enabled>
    </releases>
    <snapshots>
      <enabled>true</enabled>
    </snapshots>
  </repository>
</repositories>
<!--属性-->
<properties>
  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  <junit.version>4.11</junit.version>
</properties>
```

```
  <!--使用的license协议-->
  <licenses>
    <license>
      <name>Apache License Version 2.0</name>
      <url>http://www.apache.org/licenses/LICENSE-2.0</url>
    </license>
  </licenses>
  <!--开发者-->
  <developers>
    <developer>
      <name>Dreampie</name>
      <email>Dreampie@outlook.com</email>
      <organization>Dreampie</organization>
      <organizationUrl>http://www.dreampie.cn</organizationUrl>
    </developer>
  </developers>
  <!--源码地址-->
  <scm>
    <connection>scm:git:git@github.com:Dreampie/${project.name}.git</connectio
n>
    <developerConnection>scm:git:git@github.com:Dreampie/${project.name}.git</
developerConnection>
    <url>git@github.com:Dreampie/${project.name}</url>
  </scm>

  <!--deploy使用仓库地址-->
  <distributionManagement>
    <snapshotRepository>
      <id>ossrh</id>
      <url>https://oss.sonatype.org/content/repositories/snapshots</url>
    </snapshotRepository>
    <repository>
      <id>ossrh</id>
      <url>https://oss.sonatype.org/service/local/staging/deploy/maven2</url>
    </repository>
  </distributionManagement>
  <!--公共的依赖-->
  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>${junit.version}</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
```

- 在maven中使用jetty，tomcat容器,相关配置可以去官网查询

```
   <!--jetty-->
   <plugin>
    <groupId>org.eclipse.jetty</groupId>
    <artifactId>jetty-maven-plugin</artifactId>
    <version>9.1.1.v20140108</version>
    <configuration>
      <stopKey>foo</stopKey>
      <stopPort>9091</stopPort>
      <jvmArgs>-Xmx1024m -Xms256m -XX:PermSize=256M -XX:MaxPermSize=512M</jvmA
rgs>
      <scanIntervalSeconds>6</scanIntervalSeconds>
      <httpConnector>
        <port>9090</port>
      </httpConnector>
      <webAppConfig>
        <contextPath>/</contextPath>
        <!--<defaultsDescriptor>${basedir}/src/main/resources/webdefault.xml</
defaultsDescriptor>-->
      </webAppConfig>
      <systemProperties>
        <systemProperty>
          <name>org.eclipse.util.URI.charset</name>
          <value>UTF-8</value>
        </systemProperty>
      </systemProperties>
    </configuration>
  </plugin>
  <!--tomcat-->
  <plugin>
    <groupId>org.apache.tomcat.maven</groupId>
    <artifactId>tomcat6-maven-plugin</artifactId>
    <version>2.2</version>
    <configuration>
      <uriEncoding>UTF-8</uriEncoding>
      <port>9090</port>
      <path>/</path>
      <!-- 应用的部署位置 -->
    </configuration>
  </plugin>
```