转 编码规范系列（一）：Eclipse Code Templates设置

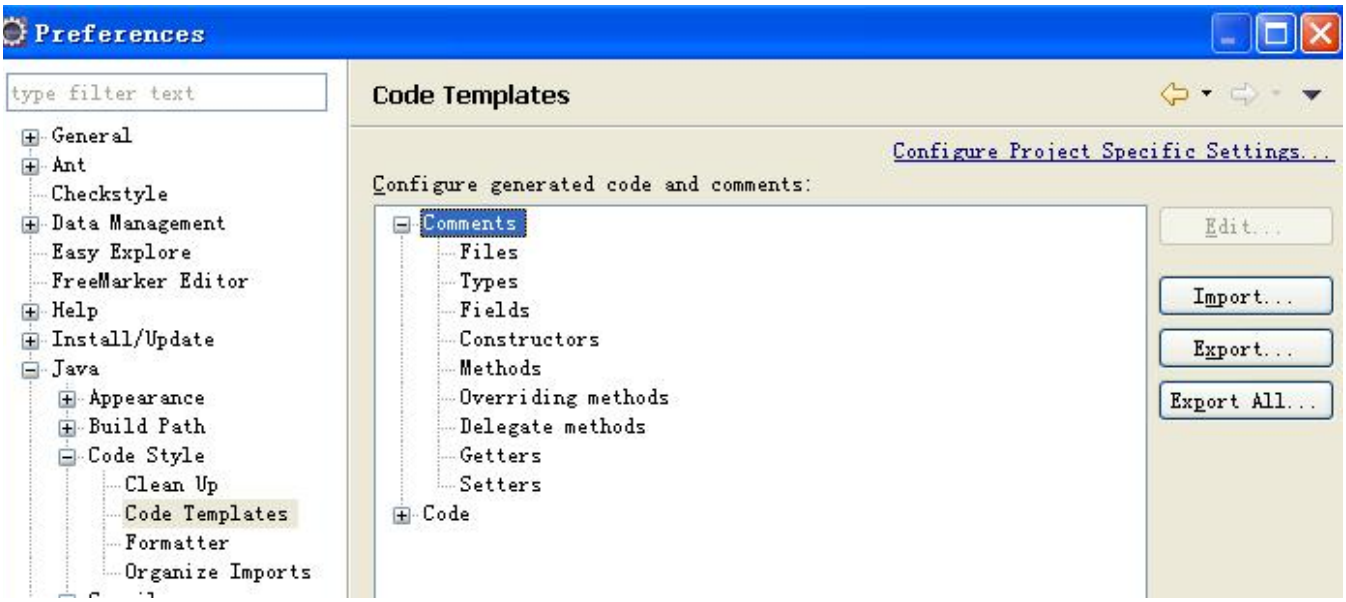
014年10月14日 11:53:45 阅读数：1303

oogle Java编程风格指南：http://www.hawstein.com/posts/google-java-style.html#Naming

文地址：http://chenzhou123520.iteye.com/blog/1625629

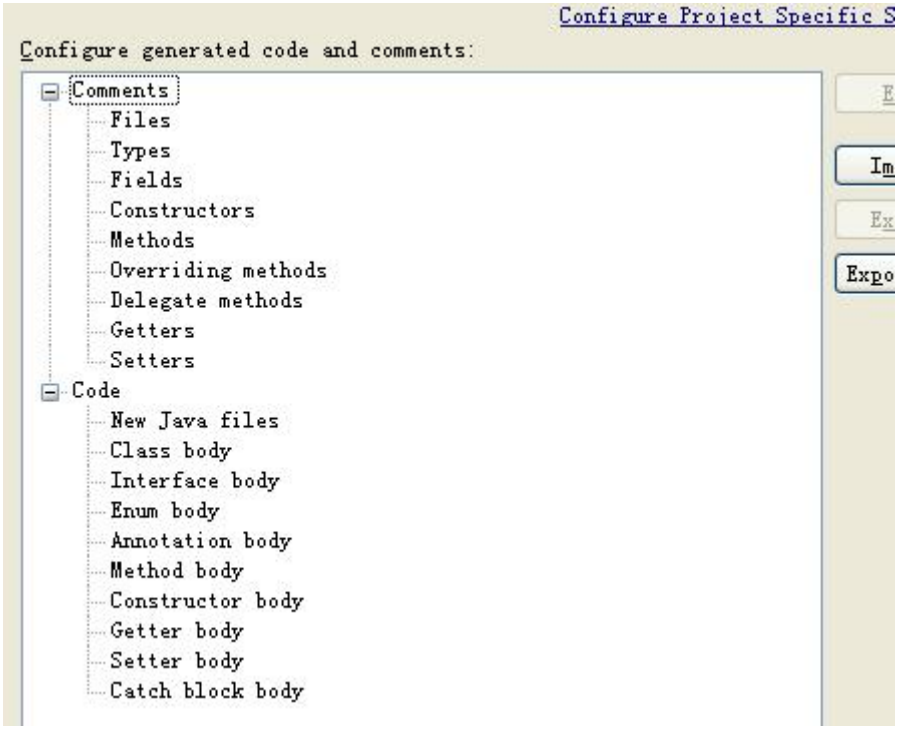
工作开始，经历了几个项目的开发，现在的项目一般都是一个团队共同开发，而每个人都有自己的编码习，为了统一格式，项目组在项目开发之前都会制定一系列的规范。俗话说约定优于配置，但是在执行过程中往发现效果不是很好（主要是指编码规范这一方面）。所以我们不得不采取一些措施来协助我们统一项目开人员的编码风格。主要包括三个方面：设置Code Templates、Eclipse formatter、Checkstyle，本篇主要介如何设置Code Templates，具体步骤如下：

开Window->Preferences->Java->Code Style->Code Templates



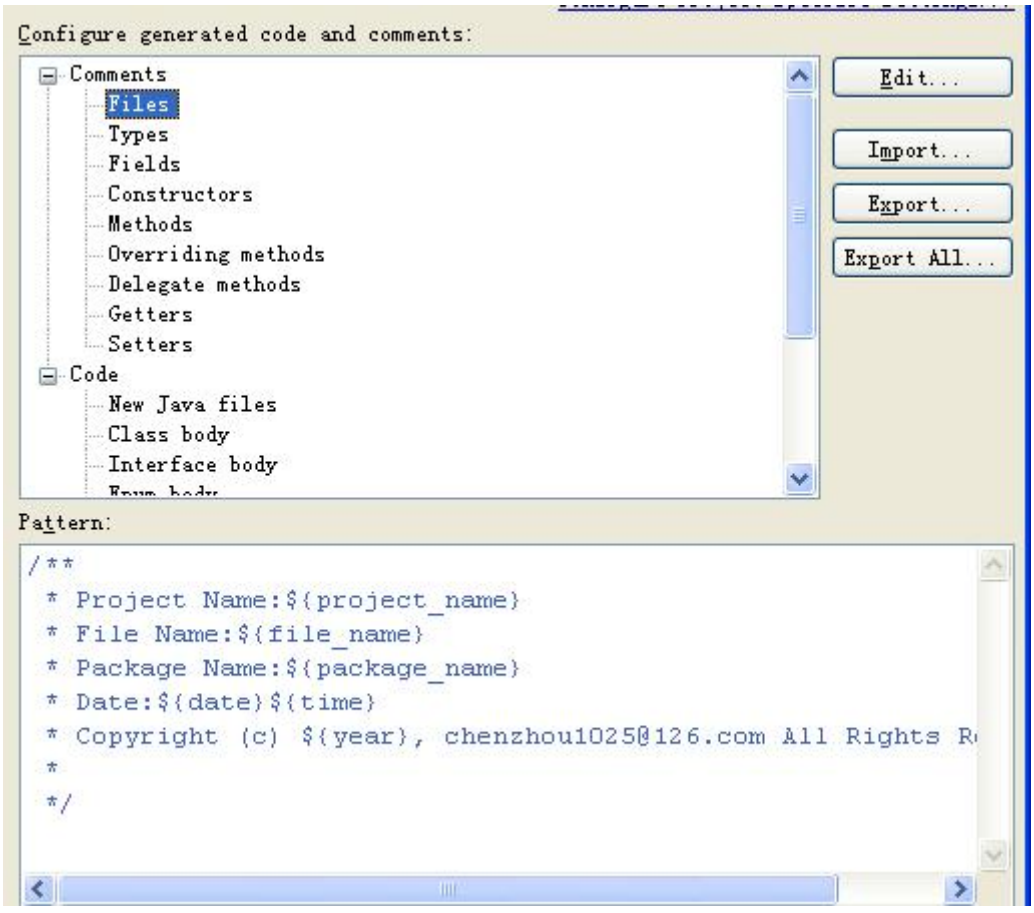
击"Import"，导入模板codetemplates.xml文件。

odetemplates.xml内容是我们自己预先定义好的，在这里先不详细描述，我们可以看到Eclipse Code Templat s界面中间Configure generated code and comments区域包含了两个菜单树：Comment、Code，如下图所：



omments代表注释模板，Code代表代码模板，其中每一个子菜单代表子项的模板。

们只要点击某一个子项，就会在界面下方的Pattern区域看到该项我们所定义的模板内容和格式，如下图所



上图所示，当我们点击Comments下的Files子菜单时，下面的Pattern会显示Java文件的头部注释。

面详细列出每一个子项的模板格式：

omments-->Files（Java文件注释）

Java代码

```
/**
 * Project Name:${project_name}
 * File Name:${file_name}
 * Package Name:${package_name}
 * Date:${date}${time}
 * Copyright (c) ${year}, chenzhou1025@126.com All Rights Reserved.
 *
 */
```

omments-->Types（Java类注释）

Java代码

```
/**
 * ClassName: ${type_name} <br/>
 * Function: ${todo} ADD FUNCTION. <br/>
 * Reason: ${todo} ADD REASON(可选). <br/>
 * date: ${date} ${time} <br/>
 *
 * @author ${user}
 * @version ${enclosing_type}${tags}
 * @since JDK 1.6
 */
```

omments-->Fields（类字段注释）

Java代码

```
/**
 * ${field}:${todo}(用一句话描述这个变量表示什么).
 * @since JDK 1.6
 */
```

omments-->Constructors（构造函数注释）

Java代码

```
/**
 * Creates a new instance of ${enclosing_type}.
 *
 * ${tags}
 */
```

omments-->Methods（Java方法注释）

Java代码

```
/**
 * ${enclosing_method}:(这里用一句话描述这个方法的作用). <br/>
 * ${todo}(这里描述这个方法适用条件 – 可选).<br/>
 * ${todo}(这里描述这个方法的执行流程 – 可选).<br/>
 * ${todo}(这里描述这个方法的使用方法 – 可选).<br/>
```

```
 * ${todo}(这里描述这个方法的注意事项 – 可选).<br/>
 *
 * @author ${user}
 * ${tags}
 * @since JDK 1.6
 */
```

omments-->Overriding methods（重写方法注释）

Java代码

```
/**
 * ${todo} 简单描述该方法的实现功能（可选）.
 * ${see_to_overridden}
 */
```

omments-->Delegate methods（代理方法注释）

Java代码

```
/**
 * ${tags}
 * ${see_to_target}
 */
```

omments-->Getters（Java Getter方法注释）

Java代码

```
/**
 * ${bare_field_name}.
 *
 * @return  the ${bare_field_name}
 * @since   JDK 1.6
 */
```

omments-->Setters（Java Setters方法注释）

Java代码

```
/**
 * ${param}.
 *
 * @param   ${param}    the ${bare_field_name} to set
 * @since   JDK 1.6
 */
```

ode-->New Java files（新建java文件代码模板）

Java代码

```
/**
 * Project Name:${project_name}
 * File Name:${file_name}
 * Package Name:${package_name}
 * Date:${date}${time}
```

```
 * Copyright (c) ${year}, chenzhou1025@126.com All Rights Reserved.
 *
*/
${filecomment}

${package_declaration}
/**
 * ClassName:${type_name} <br/>
 * Function: ${todo} ADD FUNCTION. <br/>
 * Reason:   ${todo} ADD REASON. <br/>
 * Date:     ${date} ${time} <br/>
 * @author   ${user}
 * @version
 * @since    JDK 1.6
 * @see
 */
${typecomment}
${type_declaration}
```

ode-->Method body（方法体模板）

Java代码

```
// ${todo} Auto-generated method stub
${body_statement}
```

ode-->Constructor body（构造函数模板）

Java代码

```
${body_statement}
// ${todo} Auto-generated constructor stub
```

ode-->Getter body（字段Getter方法模板）

Java代码

```
return ${field};
```

ode-->Setter body（字段Setter方法模板）

Java代码

```
${field} = ${param};
```

ode-->Catch block body（异常catch代码块模板）

Java代码

```
// ${todo} Auto-generated catch block
${exception_var}.printStackTrace();
```

ł中codetemplates.xml内容如下：

Xml代码

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?><templates><template autoinsert="false" context="gettercomment_context" deleted="false" description="Comment for getter method" enabled="true" id="org.eclipse.jdt.ui.text.codetemplates.gettercomment" name="gettercomment">/**
 * ${bare_field_name}.
 *
 * @return  the ${bare_field_name}
 * @since   JDK 1.6
 */</template><template autoinsert="false" context="settercomment_context" deleted="false" description="Comment for setter method" enabled="true" id="org.eclipse.jdt.ui.text.codetemplates.settercomment" name="settercomment">/**
 * ${param}.
 *
 * @param   ${param}    the ${bare_field_name} to set
 * @since   JDK 1.6
 */</template><template autoinsert="false" context="constructorcomment_context" deleted="false" description="Comment for created constructors" enabled="true" id="org.eclipse.jdt.ui.text.codetemplates.constructorcomment" name="constructorcomment">/**
 * Creates a new instance of ${enclosing_type}.
 *
 * ${tags}
 */
</template><template autoinsert="false" context="filecomment_context" deleted="false" description="Comment for created Java files" enabled="true" id="org.eclipse.jdt.ui.text.codetemplates.filecomment" name="filecomment">/**
 * Project Name:${project_name}
 * File Name:${file_name}
 * Package Name:${package_name}
 * Date:${date}${time}
 * Copyright (c) ${year}, chenzhou1025@126.com All Rights Reserved.
 *
 */</template><template autoinsert="false" context="typecomment_context" deleted="false" description="Comment for created types" enabled="true" id="org.eclipse.jdt.ui.text.codetemplates.typecomment" name="typecomment">/**
 * ClassName: ${type_name} &lt;br/&gt;
 * Function: ${todo} ADD FUNCTION. &lt;br/&gt;
 * Reason: ${todo} ADD REASON(可选). &lt;br/&gt;
 * date: ${date} ${time} &lt;br/&gt;
 *
 * @author ${user}
 * @version ${enclosing_type}${tags}
 * @since JDK 1.6
 */</template><template autoinsert="false" context="fieldcomment_context" deleted="false" description="Comment for fields" enabled="true" id="org.eclipse.jdt.ui.text.codetemplates.fieldcomment" name="fieldcomment">/**
 * ${field}:${todo}(用一句话描述这个变量表示什么).
 * @since JDK 1.6
 */</template><template autoinsert="false" context="methodcomment_context" deleted="false" description="Comment for non-overriding methods" enabled="true" id="org.eclipse.jdt.ui.text.codetemplates.methodcomment" name="methodcomment">/**
 * ${enclosing_method}:(这里用一句话描述这个方法的作用). &lt;br/&gt;
 * ${todo}(这里描述这个方法适用条件 – 可选).&lt;br/&gt;
 * ${todo}(这里描述这个方法的执行流程 – 可选).&lt;br/&gt;
 * ${todo}(这里描述这个方法的使用方法 – 可选).&lt;br/&gt;
 * ${todo}(这里描述这个方法的注意事项 – 可选).&lt;br/&gt;
 *
```

```
 * @author ${user}
 * ${tags}
 * @since JDK 1.6
 */</template><template autoinsert="false" context="overridecomment_context" deleted="false" description="Comment for overriding methods" enabled="true" id="org.eclipse.jdt.ui.text.codetemplates.overridecomment" name="overridecomment">/**
 * ${todo} 简单描述该方法的实现功能（可选）.
 * ${see_to_overridden}
 */</template><template autoinsert="true" context="delegatecomment_context" deleted="false" description="Comment for delegate methods" enabled="true" id="org.eclipse.jdt.ui.text.codetemplates.delegatecomment" name="delegatecomment">/**
 * ${tags}
 * ${see_to_target}
 */</template><template autoinsert="false" context="newtype_context" deleted="false" description="Newly created files" enabled="true" id="org.eclipse.jdt.ui.text.codetemplates.newtype" name="newtype">/**
 * Project Name:${project_name}
 * File Name:${file_name}
 * Package Name:${package_name}
 * Date:${date}${time}
 * Copyright (c) ${year}, chenzhou1025@126.com All Rights Reserved.
 *
*/
${filecomment}

${package_declaration}
/**
 * ClassName:${type_name} &lt;br/&gt;
 * Function: ${todo} ADD FUNCTION. &lt;br/&gt;
 * Reason:   ${todo} ADD REASON. &lt;br/&gt;
 * Date:     ${date} ${time} &lt;br/&gt;
 * @author   ${user}
 * @version
 * @since    JDK 1.6
 * @see
 */
${typecomment}
${type_declaration}
</template><template autoinsert="true" context="classbody_context" deleted="false" description="Code in new class type bodies" enabled="true" id="org.eclipse.jdt.ui.text.codetemplates.classbody" name="classbody">
</template><template autoinsert="true" context="interfacebody_context" deleted="false" description="Code in new interface type bodies" enabled="true" id="org.eclipse.jdt.ui.text.codetemplates.interfacebody" name="interfacebody">
</template><template autoinsert="true" context="enumbody_context" deleted="false" description="Code in new enum type bodies" enabled="true" id="org.eclipse.jdt.ui.text.codetemplates.enumbody" name="enumbody">
</template><template autoinsert="true" context="annotationbody_context" deleted="false" description="Code in new annotation type bodies" enabled="true" id="org.eclipse.jdt.ui.text.codetemplates.annotationbody" name="annotationbody">
</template><template autoinsert="true" context="catchblock_context" deleted="false" description="Code in new catch blocks" enabled="true" id="org.eclipse.jdt.ui.text.codetemplates.catchblock" name="catchblock">
// ${todo} Auto-generated catch block
${exception_var}.printStackTrace();
</template><template autoinsert="false" context="methodbody_context" deleted="false" description="Code in created method stubs" enabled="true" id="org.eclipse.jdt.ui.text.codetemplates.methodbody" name="methodbody">
```

```
// ${todo} Auto-generated method stub
${body_statement}</template><template autoinsert="true" context="constructorbody_context" deleted="false" description="Code in created constructor stubs" enabled="true" id="org.eclipse.jdt.ui.text.codetemplates.constructorbody" name="constructorbody">
${body_statement}
// ${todo} Auto-generated constructor stub
</template><template autoinsert="true" context="getterbody_context" deleted="false" description="Code in created getters" enabled="true" id="org.eclipse.jdt.ui.text.codetemplates.getterbody" name="getterbody">return ${field};</template><template autoinsert="true" context="setterbody_context" deleted="false" description="Code in created setters" enabled="true" id="org.eclipse.jdt.ui.text.codetemplates.setterbody" name="setterbody">${field} = ${param};</template></templates>
```

置Code Templates的目的主要是为了统一各种注释的格式以及代码的模板，只要设定好Code Templates之后用Eclipse就可以方便地生成我们自定义的注释，开发人员也容易接受！