# 代码生成器说明文档

## 1.什么是代码生成器

代码生成器顾名思义就生成代码的一个软件。为了节省人力成本，在日常的企业开发中，代码生成器使用的较为普遍。其实在我们目前所学的技术中，就有代码生成器的影子。比如我们学习的JSP技术，浏览器并不能直接识别JSP文件。当用户通过浏览器访问某个JSP页面的时候，其中的交互分为分为几个步骤：

* 用户通过浏览器访问JSP页面
* Tomcat找到对应JSP资源，将JSP资源转化(编译，翻译)成为HTML数据
* Tomcat将HTML数据进行返回

在此，我们就可以这样理解

* 我们所编写JSP其实是编写了一个HTML的模板。
* JSP页面的标签，相当于模板中的占位符。
* Tomcat根据JSP模板，生成HTML页面。
* 数据库的数据不同，则生成的HTML页面也不同。

最后我们得出一个结论：

**Tomcat可以根据JSP模板和数据生成不同的HTML页面。所以Tomcat对JSP的处理过程我们就可以认为是一个代码生成器的机制。**

## 2.代码生成器原理

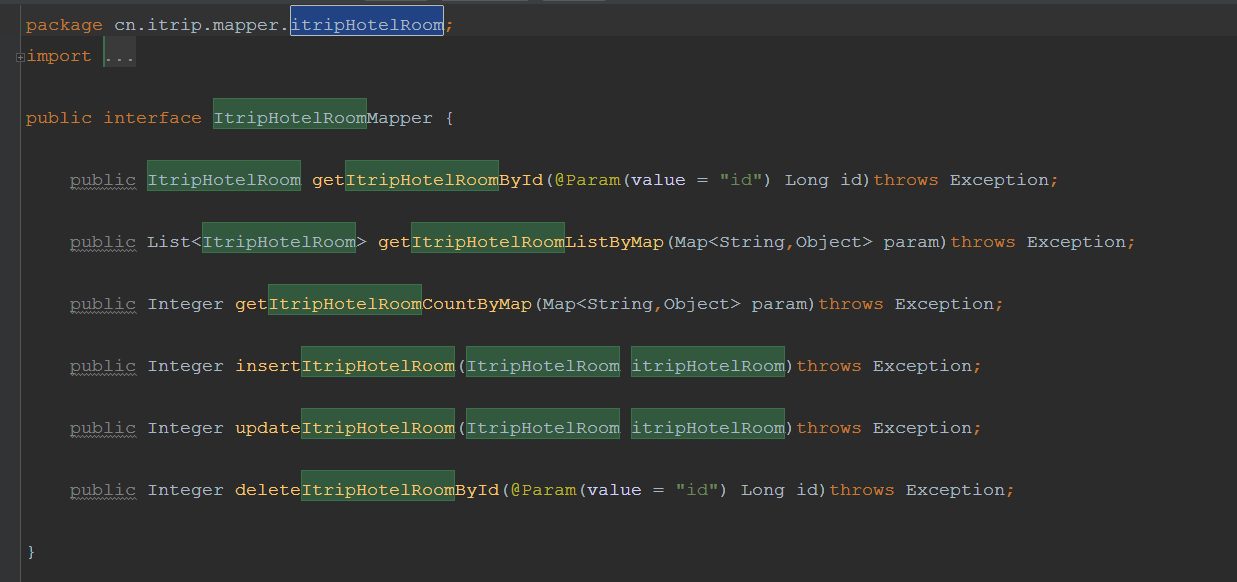
在第1小节中，其实我们已经以Tomcat和JSP举例，为大家说明了代码生成器的原理，在此我们再来总结一下。代码生成器的原理就是：**根据模板+数据生成不同文件。**

比如以下的两段代码。

1.ItripHotelOrderMapper文件



2.ItripHotelOrderMapper文件



在上述ItripHotelOrderMapper和ItripHotelOrderMapper文件中，我们注意到，两个文件的代码很相似。除了高亮部分不一致以外，其它的代码均一致。因此，我们可以设想，我们能不能编写模板生成这样的Mapper类。高亮部分使用占位符来进行代替，具体内容根据具体数据进行置换。

## 3.代码生成器三要素

以上我们了解了代码生成器的概念和原理，接下来我们开始着手编写属于我们自己的代码生成器。我们将代码生成器的分为三部分：

* 模板：生成文件的模板文件。
* 数据：生成文件所需要的关键数据。
* 合成机制：使用数据置换模板中的占位符，生成新的文件的机制。

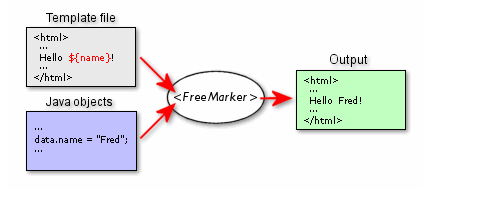
其中数据我们可以从数据库中进行获取，模板文件，以及根据模板文件生成代码的机制我们选用较为成熟的模板引擎: FreeMarker

## 3. FreeMarker

### 3.1 FreeMarker介绍

FreeMarker是一款模板引擎： 即一种**基于模板**和**要改变的数据**，并用来生成输出文本（HTML网页、电子邮件、配置文件、源代码等）的通用工具。FreeMarker是免费的，基于Apache许可证2.0版本发布。其模板编写为FreeMarker Template Language（FTL），属于简单、专用的语言。

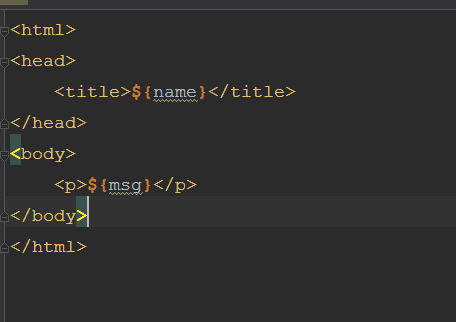
FreeMarker原理介绍:



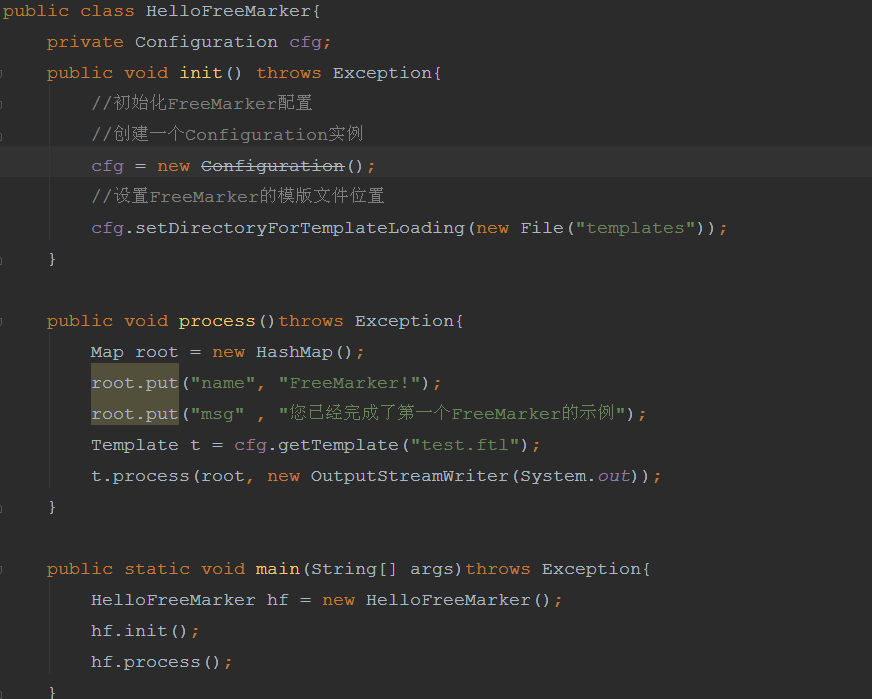
### 3.2 FreeMarker数据合成

综上所述FreeMarker就是根据模板和数据生成我们想要的文件。具体的FreeMarker模板文件如下图所示。

test.ftl



HelloFreeMarker根据test.ftl生成HTML数据



### 3.3 FreeMarker语法

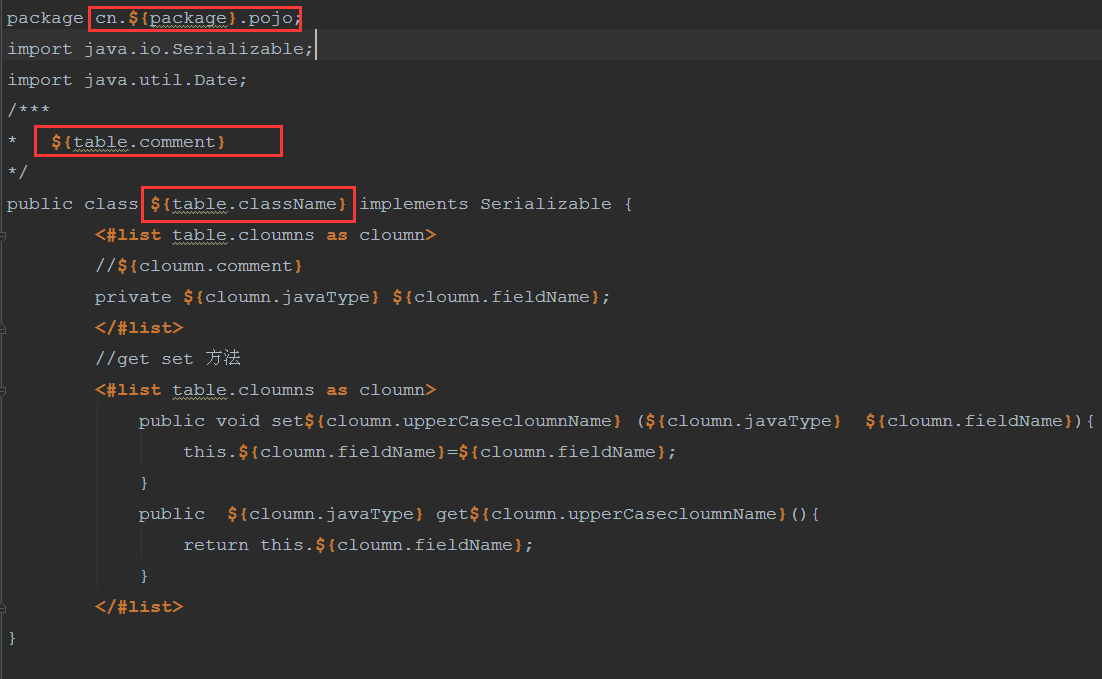
#### 3.3.1输出数据

FreeMarker可以输出的数据包括基本数据类型的数据和对象数据类型的数据，这其中包括输出对象数据类型的属性的值。

* 输出基本技术类型或多项属性的语法:

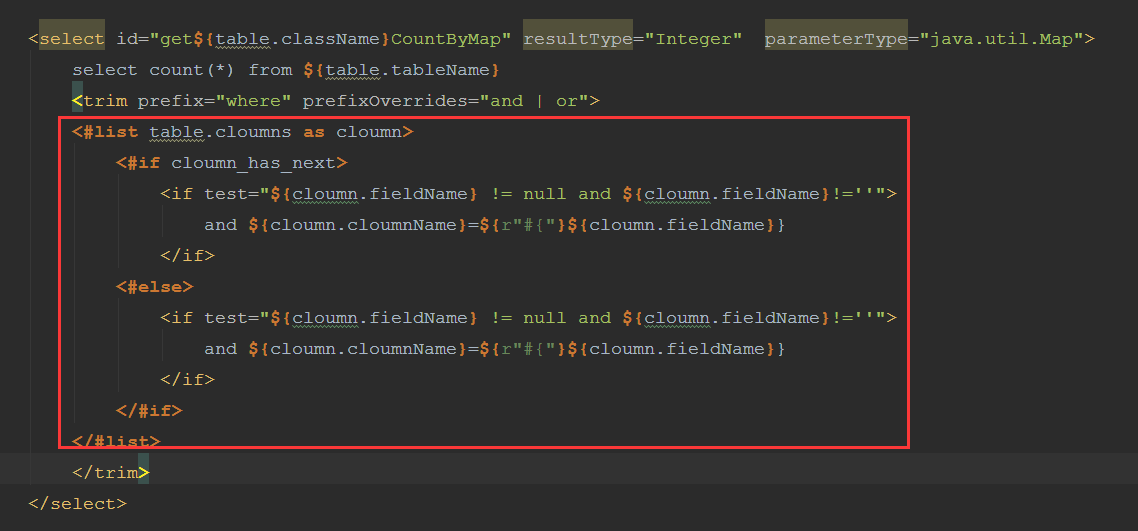
|  |
| --- |
| * **${数据}或者${数据.属性}** |

**Eg1:**



#### 3.3.2循环读取

|  |
| --- |
| **<#list sequence as item>**  **<#if item\_has\_next>......**  **<#else>......**  **</#if>**  **</#list>** |

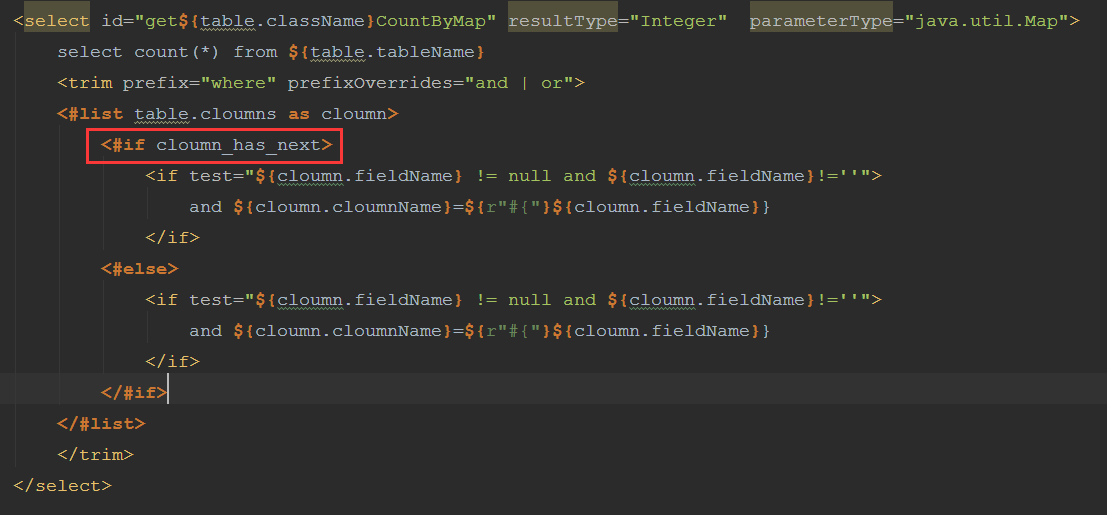


#### 3.3.3逻辑判断

**语法：**

|  |
| --- |
| **<#if condition>...**  **<#elseif condition2>...**  **<#elseif condition3>......**  **<#else>......**  **</#if>** |

**Eg：**



## 4.数据之JDBC

### 4.1使用JDBC获取数据库所有列表信息

|  |
| --- |
| //连接数据库  Class.forName(DBDRIVER);  conn = DriverManager.getConnection(DBURL, DBUSER, DBPASS);  DatabaseMetaData dbmd = conn.getMetaData();  ResultSet resultSet = dbmd.getTables(null, null, null, new String[]{"TABLE"});  while (resultSet.next()) {  //获取表名并输出  String tableName = resultSet.getString("TABLE\_NAME");  System.out.println(tableName);  } |

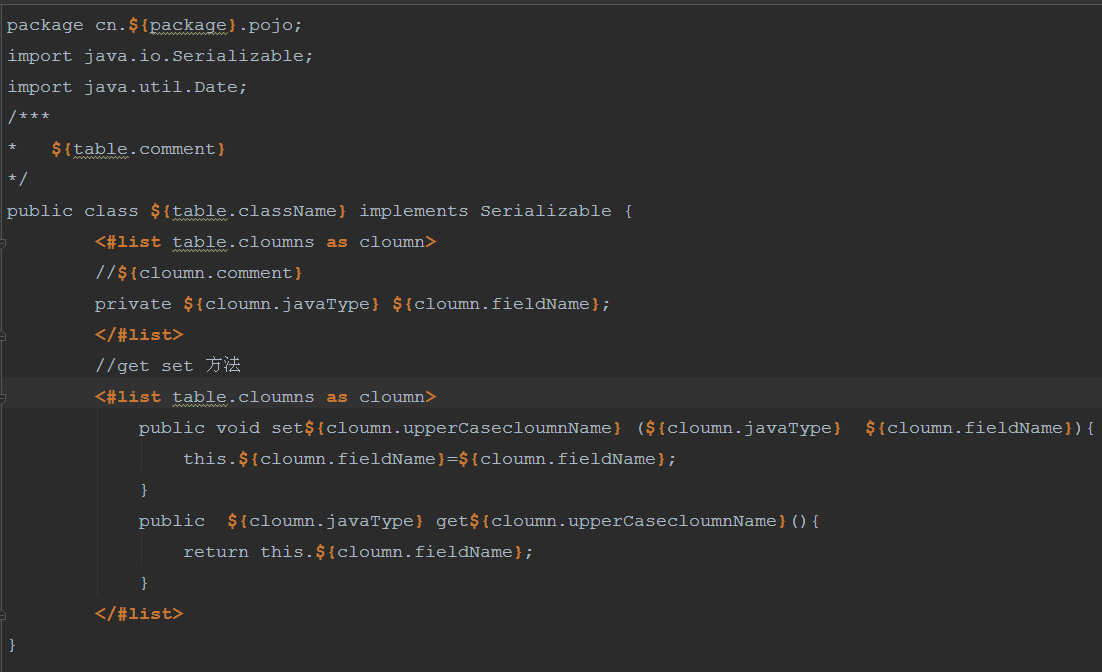
### 4.2使用JDBC获取数据库表注释

|  |
| --- |
| private static String getCommentByTableName(String tableName) throws Exception {  Connection conn = DriverManager.getConnection(DBURL, DBUSER, DBPASS);  Statement stmt = conn.createStatement();  ResultSet rs = stmt.executeQuery("SHOW CREATE TABLE " + tableName);  String comment = null;  if (rs != null && rs.next()) {  comment = parse(rs.getString(2));  }  rs.close();  stmt.close();  conn.close();  return comment;  } |

### 4.3使用JDBC获取数据库表列信息

|  |
| --- |
| ResultSet rs = dbmd.getColumns(null, "%", tableName, "%");  while (rs.next()) {  //获取列名  System.out.println(rs.getString("COLUMN\_NAME"));  //获取列注释  System.out.println(rs.getString("REMARKS"));  //获取列类型  System.out.println(rs.getString("TYPE\_NAME"));  } |

## 5.Module模板



## 6.Mapper.xml模板

