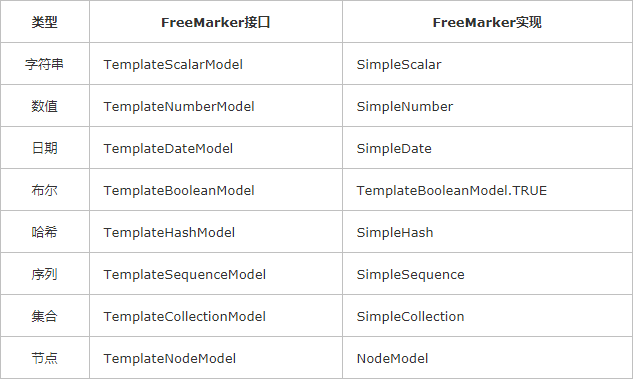
# 数据模型

在模板内部可以使用的变量是实现TemplateModel接口的java对象。同样，我们也可以使用java集合类作为自己的数据模型，因为在内部将会被替换成合适的TemplateModel实例，这种功能叫做对象包装。对象包装可以以透明的方法转换任何对象到实现TemplateModel接口类的实例。

几乎每一个基本类型变量都对应一个TemplateModel的子接口。

为什么ftl有多重数据模型：



## 标量

有四种标量类型：Boolean , Number , String , Date.

每一个标量类型对应一个接口：

String对应于TemplateScalarModel接口。

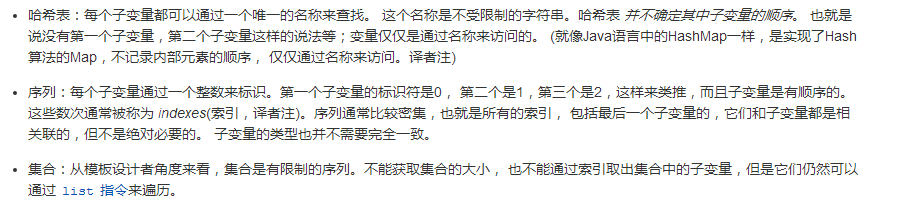
Boolean对应于TemplateBooleanModel接口。

Number对应于TemplateNumberModel接口。

Date对应于TemplateDateModel接口。

## 容器

有三种容器类型：Hashe, Sequences , Collections



## 方法

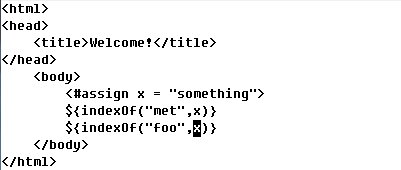
### 自定义方法

自定义方法：实现TemplateMethodModel接口

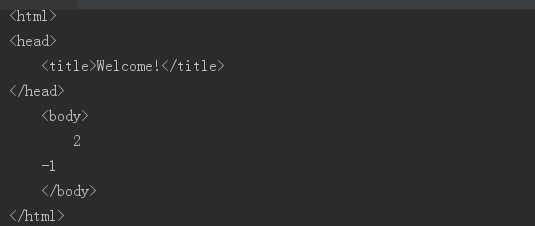
|  |
| --- |
| import freemarker.template.SimpleNumber; import freemarker.template.TemplateMethodModel; import freemarker.template.TemplateModel; import freemarker.template.TemplateModelException; import java.util.List; public class IndexOfMethod implements TemplateMethodModel {  public TemplateModel exec(List args) throws TemplateModelException  {  if (args.size() != 2)  {  throw new TemplateModelException("Wrong arguments");  }  return new SimpleNumber(((String) args.get(1)).indexOf((String) args.get(0)));  } } |

|  |
| --- |
| import freemarker.core.Environment; import freemarker.template.\*; import java.io.File; import java.io.IOException; import java.io.OutputStreamWriter; import java.io.Writer; import java.util.HashMap; import java.util.Map; public class Demo1 {  public static void main(String[] args)  {  Configuration conf = new Configuration(Configuration.*VERSION\_2\_3\_28*) ;  try  {  conf.setDirectoryForTemplateLoading(new File("F:\\HTML\\ftl"));  conf.setTemplateExceptionHandler(TemplateExceptionHandler.*RETHROW\_HANDLER*);  conf.setLogTemplateExceptions(false);  //在模板处理过程中抛出的未检测异常包括到模板异常中  conf.setWrapUncheckedExceptions(true);  Map<String, Object> root=new HashMap<>();  root.put("indexOf", new IndexOfMethod());  Template template = conf.getTemplate("test2.ftl") ;  Writer out = new OutputStreamWriter(System.*out*) ;  template.process(root,out);  Environment env = Environment.*getCurrentEnvironment*() ;  out.flush();  out.close();  }catch(IOException e){  e.printStackTrace();  }catch(TemplateException e){  e.printStackTrace();  }  } } |

页面代码：



运行结果：



## 指令

## 节点

## 对象包装器

### 默认对象包装

### 自定义对象包装