模板引擎设计实现

傅玉宝　胡晓航

2011.12

搜狐微博

设计思想

- TokenScanner 从数据源中构造出 token 序列（词法分析）
- Parser 从 token 序列解析出 dom 树（语法分析）
- RenderUtils 负责将 dom 转化成 render
- Template 调用 render 方法，实现渲染

Token

- Token 接口 表示一个词

- Token 实现类

  ElToken el 表达式 ${...}

  TextToken 文本

  ElementBeginTag xml 开始标签 <c:if...>

  ElementClosedTag xml 闭合标签 <c:if.../>

  ElementEndTag xml 结束标签 </c:if...>

TokenScanner

- TokenScanner　负责词法分析
  - 从数据源　转成 char[]
  - 遍历 char[], 匹配关键字，构建不同的 token
  - 存入 List<Token>
  - 词法检查（还没实现）

Dom 元素

- Node 接口四种实现
- Text 文本
- Expression el 表达式
- Element 标签 (if include)
- 还有一种注释类 (暂没实现)

Parser

- Parser　负责语法解析

parseTokens　方法将 token 序列转成 dom 树

递归调用实现，实现 xml 标签嵌套解析

Render 类

- TextRender 文本渲染
- ExpressionrRender EL 表达式渲染
- ElementRender xml 标签渲染
- RenderCluster 负责将多个 render 合并成一个 render
- RenderUtils 负责将 dom 转化成 render

TagProcessor

- TagProcessor 用来处理 xml 标签渲染逻辑

  通过实现 TagProcessor 接口，可以实现自定义标签

IncludeProcessor 实现模板引入逻辑

IfProcessor 实现 if 逻辑处理

ForEachProcessor 实现集合遍历

管理相关类

- ProcessorManager 负责管理 processor
- TemplateManager 负责 template 管理和缓存，线程安全的实现类
- TemplateEngine 模板相关功能配置

  如 processor 配置

  JexlEngine el 表达式配置

Demo

```
public class EnginTest {

    public static void main(String[] args) {
        //获取模板输入流
        InputStream in = EnginTest.class.getClassLoader().getResourceAsStream("template/demo.html");
        InputStream include = EnginTest.class.getClassLoader().getResourceAsStream("template/include.ht

        //通过模板输入流，构造模板
        Template template = new SimpleTemplate(in, "GBK");
        Template includeTemplate = new SimpleTemplate(include, "GBK");

        //向templateManager注册模板
        TemplateManager templateManager = new ConcurrentTemplateManager();
        templateManager.addTemplate("demo", template);
        templateManager.addTemplate("test", includeTemplate);

        //构造模板所需要的参数
        Map<String, Object> map = new HashMap<~>();
        map.put("msgid", 123654);
        map.put("msg", 147852);
        map.put("sohucms_summary", "sohucms_summary");
        map.put("pic_temp", "pic_temp");
        map.put("at_temp", "at_temp");
        map.put("from", "from");
        map.put("geo", "geo");
        map.put("num_reply", "num_reply");

        //模板渲染，输出到StringWrite
        StringWriter sb = new StringWriter();
        template.render(map, sb);

        System.out.println(sb.toString());

    }
```

Thanks!

搜狐微博