北京邮电大学

中期报告

**支持Markdown的静态 Web服务器**

成员: 戴鑫旺 程相睿 杨乐乐

2022年5月1日

|  |  |  |  |
| --- | --- | --- | --- |
| 论文题目 | 支持Markdown的静态WEB服务器 | | |
| 选题来源 | 个人博客、社区公告板 | 论文类型 |  |
| 开题日期 | 2022-5-1 | 开题地点 |  |
| **一、课题预期成果（主要来自开题报告中的任容）**    本课题计划开发一个具备Markdown解析功能的静态WEB服务器，使得用户只需编写简单的Markdown文件即可生成网页进行博文记录或信息发布。  具体而言，本课题主要研究内容分为两部分：   1. Markdown文件处理：    1. 识别Markdown文件与HTML文件，对于HTML文件不做处理；    2. 遍历Markdown文件并进行语法解析，得到抽象语法树；    3. 基于生成的抽象语法树，通过增加HTML标签对文本进行渲染，得到原始HTML文件；    4. 提供CSS样式文件与JavaScript脚本文件对原始HTML进行美化。 2. 静态服务器：    1. 接受用户请求，并对用户请求按照HTTP请求格式进行解析，得到请求参数；    2. 根据用户的请求参数对HTML文件进行检索；    3. 将检索得到的HTML文件封装成HTTP响应返回给用户；    4. 处理用户的文件上传操作，将用户上传的Markdown文件保存到服务器端；    5. 提供实时解析能力，将用户上传的Markdown文件交由Markdown文件处理模块实时解析成HTML进行展示。   本课题可能的创新之处包括：   1. 除了支持基础的Markdown语法以外，可以提供一些自定义的语法元素来扩展Markdown的功能。例如可以通过自定义语法设置某篇Markdown文章的展示优先级，通过自定义语法设置文章的作者、摘要、标签、所用样式模板等元信息。 2. 利用非阻塞IO提高服务器对并发访问的处理能力。 3. 增加Markdown文件解析后的HTML缓存，不用每次访问都重新解析，并通过哈希摘要判断文件是否被修改，并只对修改的文件进行重新解析。 4. 增加在线Markdown编辑器，可以在线使用Markdown语法发布新的博文。 5. 通过将图片编码为base64格式来直接把图片编码成字符串写入css文件，减少用户发送HTTP请求的数量。   **二、当前进展**  目前我们已经完成了项目的主体框架，实现了基本功能，并可以正常运行。当前项目整体上分为三个模块：配置解析模块、Markdown文件处理模块与用户请求处理模块。  一次完整的项目运行流程是：  （1）首先服务器启动，配置解析模块读取本地配置文件，然后通过配置初始化Markdown文件处理模块和用户请求处理模块。  （2）接着，Markdown文件处理模块根据配置读取本地特定目录下的Markdown源文件，并进行解析。在解析的过程中，首先调用Yaml解析模块处理Markdown文件头部的yaml部分，这部分包含博文的元数据，例如该博文的题目、作者、发布时间以及展示优先级。然后调用Markdown解析模块将Markdown转化为HTML格式，附上CSS样式形成HTML文件，并缓存到特定目录。此时，博文页面生成完毕。然后该模块还会根据博文的元数据创建一个index.html页面，该页面是所有博文的目录，也是静态服务器的默认首页。  （3）随后用户请求处理模块启动，根据配置创建ServerSocket并监听特定端口。用户通过浏览器发起访问，用户请求处理模块接收用户请求并进行参数解析，得到用户的请求路径。如果请求的地址指向的博文页面存在，则对博文进行展示。否则，返回页面不存在的错误信息。  对照开题报告中的研究内容，我们已经完成了如下部分：   1. Markdown文件处理：    1. 识别Markdown文件与HTML文件，对于HTML文件不做处理；    2. 通过flexmark库对Markdown进行解析，并转化为HTML文件；    3. 提供CSS样式文件与JavaScript脚本文件对原始HTML进行美化。 2. 静态服务器：    1. 接受用户请求，并对用户请求按照HTTP请求格式进行解析，得到请求参数；    2. 根据用户的请求参数对HTML文件进行检索；    3. 将检索得到的HTML文件封装成HTTP响应返回给用户； 3. 完成的创新点：    1. 可以通过自定义语法设置某篇Markdown文章的展示优先级，通过自定义语法设置文章的作者、摘要、标签、所用样式模板等元信息。    2. 利用多线程与线程池提高服务器对并发访问的处理能力。    3. 增加Markdown文件解析后的HTML缓存，不用每次访问都重新解析，并通过哈希摘要判断文件是否被修改，并只对修改的文件进行重新解析。     我们未完成的部分：   1. Markdown文件处理模块中，我们当前用了flexmark库进行Markdown的解析与HTML文件渲染，而非自己手动完成，因此后续还需要完成下面这两部分：    1. 遍历Markdown文件并进行语法解析，得到抽象语法树；    2. 基于生成的抽象语法树，通过增加HTML标签对文本进行渲染，得到原始HTML文件； 2. 静态服务器模块中，后续需要完成如下部分：    1. 处理用户的文件上传操作，将用户上传的Markdown文件保存到服务器端；    2. 提供实时解析能力，将用户上传的Markdown文件交由Markdown文件处理模块实时解析成HTML进行展示。 3. 未完成的创新点：    1. 增加在线Markdown编辑器，可以在线使用Markdown语法发布新的博文。    2. 通过将图片编码为base64格式来直接把图片编码成字符串写入css文件，减少用户发送HTTP请求的数量。   **三、遇到的问题与解决办法**  3.1 Markdown解析模块  目前我们通过flexmark库进行Markdown的解析与HTML文件渲染。接下来我们要自己实现解析与渲染过程，做到完全不依赖flexmark库。在我们自己实现Markdown文件解析的过程中， 遇到了特殊字符处理的问题，具体来说，用户的文本中也有可能出现代表Markdown格式语法的字符，在解析的时候需要进行区分。此外，我们发现Markdown中的表格结构很难解析，目前还没有想到合适的解决方法。  我们的解决办法是阅读现有Markdown解析器例如flexmark库的源码，研究它们内部的工作原理，学习他们对于特殊字符以及表格结构的解析思想。  3.2 静态服务器用户请求处理模块  对于静态服务器的实现，我们还未实现Markdown文件上传功能，以及Markdown在线编辑器。  其中对于文件上传功能，因为我们没有使用现有的Java编程框架，因此处理用户文件上传时需要采用自己解析HTTP请求的方式进行实现。我们目前正在学习通过POST请求的方式进行文件上传的实现原理。后续会实现文件上传请求处理与实时markdown解析。  对于我们的另一个创新点Markdown在线编辑器，手动实现起来比较繁琐，需要手动编写复杂的css与JavaScript逻辑，我们计划引入现有的相关前端库来进行实现，同时也会阅读它们的源码进行学习。   1. **后续的进度计划**   （1）5月8号前完成用户文件上传功能与Markdown文件上传后实时解析功能的实现；  （2）5月22号前完成Markdown文件的解析与HTML文件生成功能，做到不依赖flexmark库；  （3）5月29号前完成为项目引入在线Markdown编辑器与图片base64编码功能；  （6）6月5号前对项目进行重构与功能测试，并编写相关文档。 | | | |