

Lab1

Lab 描述

HTML 是一种标记语言，用于创建网页。HTML 由一系列标签组成，这些标签定义了网页的结构、内容和外观。HTML 标签通常由尖括号包围，如<html>、<head>、<body>等。HTML 标签可以嵌套，这意味着一个标签可以包含其他标签。

下面是使用 html 表示一个简单的网页的例子：

```
<html>
  <head>
    <title>My Webpage</title>
  </head>
  <body>
    <h1 id="title">Welcome to my webpage</h1>
    <p id="description">This is a paragraph.</p>
    <ul id="list">
      <li id="item1">Item 1</li>
      <li id="item2">Item 2</li>
      <li id="item3">Item 3</li>
    </ul>
    <div id="footer">
      this is a text context in div
      <p id="last-updated">Last updated: 2024-01-01</p>
      <p id="copyright">Copyright © 2021 MyWebpage.com</p>
    </div>
  </body>
</html>
```

本次 Lab 要求开发一个基于命令行的编辑器对 HTML 进行编辑，用以考察对 OO 和设计模式的理解，包括：

1. HTML 模型

根据 Lab 的需要，你需要设计一个简化版本的 HTML 的面向对象的模型，从而保存并操作 HTML 内容。

为了简化要求，本次 Lab 中对于 HTML 的约定如下：

1. <html>, <head>, <title>, <body>四个标签有且仅有一个。HTML 文件顶层是<html>元素，其只有两个子元素，依序为<head>和<body>。同时，<title>是<head>的子元素。结构如下：

```
<html>
  <head>
    <title>My Webpage</title>
  </head>
  <body>
    ...
```

```
</body>
</html>
```

2. 元素内部可以包含文本内容，但文本必须处在该元素的其他子元素之前（参见上述例子中的`<div id="footer">`）。
3. 属性只支持 `id`，除`<html>`、`<head>`、`<title>`、`<body>`外其他元素必须拥有 `id` 属性，若这四个标签没有提供 `id`，默认 `id` 为标签名。同时 `id` 应该具有唯一性。
4. HTML 的解析有一定的工作量，但这部分并不是我们考核的重点。建议查找使用现有的 HTML 解析器，比如 Jsoup(Java) 等。

2. 编辑功能

实现一个基于命令行的 HTML 编辑器，可以实现基本的 HTML 编辑功能，如在某元素之后插入元素、在某元素内部添加子元素、删除元素、修改元素 `id`、修改元素文本等。同时，编辑器需要支持撤销(Undo)和重做(Redo)功能。

3. IO

1. 能够读入 HTML 文件，将 HTML 字符串转化为你所设计的 HTML 面向对象模型。
 - 可以手动解析 HTML 字符串（利用栈等数据结构）；也可以使用现有的 HTML 解析器，比如 Jsoup 中构造的模型，来辅助生成你自己设计的模型。
 - 简便起见，可以默认读入的文件是格式良好的 HTML。
2. 能够以基于树的格式和基于缩进的格式两种方式显示结果。
 - 树的格式：

```
html
├── head
│   └── title
│       └── My Webpage
└── body
    ├── h1#title
    │   └── Welcome to my webpage
    ├── p#description
    │   └── This is a paragraph.
    ├── ul#list
    │   ├── li#item1
    │   │   └── Item 1
    │   ├── li#item2
    │   │   └── Item 2
    │   └── li#item3
    │       └── Item 3
    └── div#footer
        ├── this is a text context in div
        ├── p#last-updated
        │   └── Last updated: 2024-01-01
        └── p#copyright
            └── Copyright © 2021 MyWebpage.com
```

- 缩进的格式：

```
<html>
  <head>
    <title>My Webpage</title>
  </head>
  <body>
    <h1 id="title">Welcome to my webpage</h1>
    <p id="description">This is a paragraph.</p>
    <ul id="list">
      <li id="item1">Item 1</li>
      <li id="item2">Item 2</li>
      <li id="item3">Item 3</li>
    </ul>
    <div id="footer">
      this is a text context in div
      <p id="last-updated">Last updated: 2024-01-01</p>
      <p id="copyright">Copyright © 2021 MyWebpage.com</p>
    </div>
  </body>
</html>
```

注意，Lab2的部分会要求显示一个其它对象的数据结构，比如当前工作目录的文件夹的树形结构。需要考虑树形结构构造代码的灵活性。

3. 能够将编辑器中的 HTML 模型以 HTML 字符串的形式写入文件保存。

- 需要确保编辑器之后能够再次读取该文件并继续编辑。

4. 拼写检查

选择合适的拼写检查服务，实现对元素中的文本内容进行拼写检查，并报告错误。

可以考虑以下拼写检查服务：

API：<https://dev.languagetool.org/public-http-api>

Java：<https://dev.languagetool.org/java-api>

Python：[spell-checker](#)库或[pyspellchecker](#)库

注意：

架构是 Lab 需要考核的重要内容，

1. 管理各个模块(package)的依赖关系，并画出依赖关系
2. 针对各个模块写代码进行自动化测试。
3. 需要管理对相对不稳定的第三方库的依赖，比如，如果依赖了 Jsoup 作为 HTML 的分析工具，对 Jsoup 的依赖应该限制在特定的范围。

命令设计

编辑类命令

提示：除`<html>`, `<head>`, `<title>`, `<body>`外其他元素必须拥有 `id` 属性，若这四个标签没有提供 `id`，默认 `id` 为标签名。同时 `id` 应该具有唯一性。

1. insert 在某元素之前插入新元素

```
insert tagName idValue insertLocation [textContent]
```

- `tagName` 为新元素的元素标签。
- `idValue` 为新元素的 `id`，注意 `id` 不能与其他元素重复。
- `insertLocation` 为插入位置元素的 `id`，新元素将插入到该元素之前。
- `textContent` 为可选参数，表示新元素中的文本内容。

2. append 在某元素内插入新元素

```
append tagName idValue parentElement [textContent]
```

- `tagName` 为新元素的元素标签。
- `idValue` 为新元素的 `id`，注意 `id` 不能与其他元素重复。
- `parentElement` 为目标父元素的 `id`，新元素将被插入到该元素内部，并且成为该元素的最后一个子元素。
- `textContent` 为可选参数，表示新元素中的文本内容。

3. edit-id 编辑某元素的 id

```
edit-id oldId newId
```

- `oldId` 为现有元素的 `id`。
- `newId` 为新 `id`，注意 `id` 不能与其他元素重复。

4. edit-text 编辑某元素内部的文本

```
edit-text element [newTextContent]
```

- `element` 为要编辑元素的 `id`。
- `newTextContent` 为新的文本内容，可以为空，表示清空该元素的文本内容。

5. delete 删除某元素

```
delete element
```

- `element` 为要删除元素的 id。

显示类命令

6. `print-indent` 按缩进格式显示

```
print-indent [indent]
```

- `indent` 为可选参数，表示每级缩进的空格数，默认为 2。当提供 `indent` 时，使用指定的空格数进行缩进显示。

7. `print-tree` 按树型格式显示

```
print-tree
```

8. `spell-check` 显示拼写检查结果

```
spell-check
```

- 调用拼写检查服务，显示拼写检查结果。检查结果的格式自定，合理即可。
- 需要能够检查 HTML 中的所有 `text` 内容。

输入/输出命令

提示：编辑器必须先进行 `read` 或 `init` 命令，才可以使用其他命令。

9. `read` 读入 HTML 文件

```
read filepath
```

- `filepath` 为读取文件的路径名。
- 进行必要的异常检测，例如读取的文件不存在。

10. `save` 写入 HTML 文件

```
save filepath
```

- filepath 为写入文件的路径名。
- 进行必要的异常检测，例如提供的路径名无法写入文件。

11. 初始化编辑器

```
init
```

- 将编辑器初始化为一个空的 HTML 模板供后续的编辑。
- 参考模板：

```
<html>
  <head>
    <title></title>
  </head>
  <body></body>
</html>
```

撤销和重做

支持多步撤销与多步重做，规则如下：

- 编辑类指令：支持撤销与重做。如果在撤销后发生了编辑，则此时不允许重做操作。
- 显示类指令：撤销与重做时应跳过显示类指令。
- 输入/输出指令：执行输入/输出指令后，不允许撤销与重做。

12. undo 撤销

```
undo
```

13. redo 重做

```
redo
```

评分方式

1. 架构设计：25 分

需要提供一个简单的文档，给出各个模块的描述以及依赖关系。可以补充一些关键的设计决策的说明。

2. 自动测试：25 分

针对各个模块完成自动测试代码。

3. 功能正确：40 分

功能	分值
insert	3
append	3
delete	3
edit-text	2
edit-id	2
undo	5
redo	5
read	3
save	3
init	1
print-indent	3
print-tree	3
spell-check	4

4. 代码结构：10 分

代码结构清晰，模块化，可读性强。

提交：

每人单独完成，3 周后提交

提交内容为一份压缩包，以学号_姓名的格式命名，内容包括：

- 运行所需要的必要的代码文件
 - 包含所有你自己写的代码
 - 如果是通过包管理工具下载的第三方库，不需要提交。只需要提交依赖说明文件（如果有的话），例如 java 的 pom.xml 或者 js 的 package.json 等。
- 一份架构设计文档
 - 需额外在文档中说明如何运行你的编辑器程序，例如指明使用的语言及版本、安装依赖的步骤等。