Markdown 实现块级引用双向链接的探索

作者：88250
原文链接：https://ld246.com/article/1597226949061
来源网站：链滴

双向链接

从技术上来说，“双向链接”存在已久，并且应用在了很多系统中，比如维基中词条引用，博客中的 PngBack。它表达了有向的关联关系，让用户可以更好地获取到相关信息。

从 Roam Research 重新定义大纲式笔记开始，知识管理领域里很多系统开始加入双链特性。从方便户理解和使用上，内容大致可以被分为为：

- 页面（Page）：通过标题或者 URL 标识，可以包含子页面和内容块
- 内容块（Block）：通过 id 标识，可以包含更多内容块

每个厂商对页面和内容块的实现都是不一样的，但从基本逻辑上看，块可以嵌套块，块组合构成页面整体是树形结构。块通过唯一的 id 进行标识。通过 id 可以方便地存储和表达两个块之间的关系，从进一步生成关系图。

Markdown 中的块

和厂商自定义的格式不同，Markdown 是开放的公开格式，最重要的是 Markdown 是基于文本的，有显示的格式（Schema）约束，这是 Markdown 受欢迎的原因之一，因为这极大的方便了用户读，但同时这也给制定 Markdown 规范和实现解析器增加了很大难度。

Markdown 发展到现在，基本所有解析器都开始实现或者已经实现了 CommonMark/GFM 规范。规范中，Markdown 文本内容被划分为块级元素和行级元素，块级元素又分为叶子块和容器块。叶块只能包含行级元素，容器块可以包含块级元素。从基本逻辑上看，Markdown 是通过块嵌套来组文档，整体也是树形结构。

Markdown 块链之难

对于 RR 类系统来说，内容块在设计之初就已经是结构化的了。只要能在系统里输入的内容都会存在一个定义好的块里，并且这个块可以通过 id 检索到。

但是对于 Markdown 来说，情况就不一样了，因为规范中并没有定义 id 元素。从用户角度上看，也太可能让所有人都在文本中通过某种语法来定义块 id，这样即复杂也不可靠，并且违背了 Markdown 简单文本（Plain Text）的本意，更重要的是失去了 Markdown 的互操作性。

综上，Markdown 块链的难点在于找到标识 Markdown 块的方法。

可能的解决方案

这里需要稍微介绍一下 Markdown 处理的步骤（细节请浏览这里）：

1. 读取 Markdown 文本
2. 解析文本生成 Markdown 抽象语法树（Markdown AST）
3. 通过遍历 AST 输出渲染结果

前面提到过，在文本中标识 id 基本不可行，所以步骤 1 我们就不讨论了。在步骤 2 中生成的块节点加入 id 是没问题的，这样步骤 3 渲染后也可以将 id 返回到用户界面中，以便后续做块关联时使用。

这样就解决了？当然没那么简单，因为这里忽略了一个很重要的问题，用户编辑输入是持续发生的，次渲染输出会作为下次的解析输入，而解析输入是 Markdown 文本，是不会有 id 的。

自旋方案

id 需要在整个使用过程中带着走，才能“自圆其说”。由此，我们得出以下处理步骤：

1. 读取 DOM，记录块 id
2. DOM 转 Markdown
3. Markdown 转 AST
4. AST 渲染为新 DOM
5. 将块 id 合并回新 DOM 返回

我把这个方案叫做”自旋“（Spin），即老 DOM -> 新 DOM。自旋相关的细节问题我们暂时略过回到块链上，只要合并算法可靠，我们就能解决 id 在用户编辑时不丢失的问题。

到这里，我们已经解决一半的问题了，即编辑时可以保证不丢失块标识，新生成的块也有标识。另一问题是持久化，即如何从磁盘上还原起始状态的 DOM。

持久化

饶了一圈，还是回到了 Markdown 文本不能存 id 的问题上，不换存储格式应该无解，换了格式又觉丢失灵魂。作为设计者，终究会有抉择的时刻。这个问题上，我选择的是换格式，通过 JSON 进行持化，但需要提供完善的 Markdown 导出支持。

实现上是将 Markdown AST 使用 JSON 渲染器渲染，输出为 JSON 文本存盘，在需要时读取 JSON 文本反序列化为 AST 。JSON 中必须带有 AST 节点的 id 和一些其他必备属性，这样反序列后才能还完整的 AST。

简化问题

完整实现所有 Markdown 块级元素的双链比较困难，但是仅实现那些可以自描述的块级元素（比如题元素）是毫无问题的，因为标题本身的内容可以当做块 id，这样既能保持 Markdown 的文本形态又不用添加新的语法。这方面黑曜石（Obsidian）已经给我们做了很好的示范。

Markdown 块链的意义

实现 Markdown 块链方案的探索暂时告一阶段。回过头来，我们需要重新审视块链存在的意义，也较有说服了的点是：在较细粒度上将信息进行连接，同时又不破坏信息的上下文。

我有点描述不出来块链的使用场景，我觉得这很难想象，描述它就像是在解构自己的思维。工具方法价值是由使用者确立的，根据使用者的反馈做出改进，这件事情才会慢慢变得具有意义。

最后，如果你想体验支持 Markdown 块级引用双向链接，欢迎关注思源笔记。