链滴

搞定 Vditor 自定义渲染超链接，其它原理也相同

作者：pdd

原文链接：https://ld246.com/article/1611661350955

来源网站：链滴

业务需要，须在 Markdown 渲染成 HTML 时，将所有超链接中增加自定义属性，如下：

Markdown 为：

```
[觅好图](http://mihaotu.com)
```

希望渲染后的超链接代码为：

```
<a href='http://mihaotu.com' my-attr='自定义属性'>觅好图</a>
```

按照官方文档例子，我们在 options.renderers 中绑定了方法renderLink，如下：

```
renderLink: (node, entering) => {
    if (entering) {
        return [`<a href='${node.TokensStr()}' my-attr='自定义属性'>`, Lute.WalkContinue]
    } else {
        return ['</a>', Lute.WalkContinue]
    }
}
```

其中 ${node.TokensStr()} 为超链接地址。可实际执行下来，${node.TokensStr()} 却总是空字符。染出的链接代码为： <a href my-attr='自定义属性'>觅好图，href 后的链接地址为空。

仔细阅读官方文档，了解到链接属于行级节点，由链接相关标识符、链接文本等构成，其渲染函数为：

renderLink 及下级函数 renderOpenBracket、renderCloseBracket、renderOpenParen、renderCloseParen、renderLinkText、renderLinkSpace、renderLinkDest 和 renderLinkTitle 组成。

也就是说，要将 markdown 的链接代码渲染为 html 超链接，Lute 引擎要调用以上所有函数。

简单分析一下调用链及函数之间的关系。

所有的函数都有两个参数，分别为 node 和 entering。node 为当前遍历到的节点，保存着当前节点信息（renderLink 函数时，node 为根节点。renderLinkText 函数时，node 为子节点对象，保存着接文本信息，renderOpenBracket 函数时，node 是另一个子节点对象，保存的是超链接 markdown 标识符 '['，其它函数都对应着超链接 markdown 代码的某部分的标识符，详见：链接）。entering 遍历是否为进入节点。Lute 采用深度优先算法来遍历树，在进入节点时 entering 为 true，离开节点为 false，我们可以简单理解为每个 render 函数都会被调用两次，entering 为 true 时表示第一次调，false 时为第二次调用。如图：

```
sequenceDiagram
    Lute->>renderLink: 1. 进入（entering:true 第1次调）
    Lute->>renderLinkText: 2. 进入（entering:true 第1次调）
    Lute->>renderLinkText: 3. 离开（entering:false 第2次调）
    Lute->>render...: 4. 进入（entering:true 第1次调）
    Lute->>render...: 5. 离开（entering:false 第2次调）
    Lute->>renderLink: 6. 进入（entering:false 第2次调）
```

回到最初的代码，我在 entering 为 true 时返回了 ，为 false 时返回了 ，再由 Lute 将 renderLinkText 函数渲染的链接文字拼接在这两次用的中间，就组成了完整的 html 超链接代码。可是，为什么中 renderLink 函数中，${node.TokenSt()} 取不到值呢？

按照上面的分析得知，在遍历不同 render 函数时，node 都是不同的对象。因此不能在 renderLink

数的 node 中取到 renderLinkDest 函数解析后的值。

又因为 node 是嵌套引用结构（node -> node ->node），我们是不是也可以通过遍历 node 从中取到正确的链接地址呢？遗憾的是，我没找到 node 有获取 child 和 parent 的属性/函数，望V大/D看到后能告知。这种情况下，我们又该如何实现目标呢？

根据 render 函数的调用顺序及相关特性，我的做法如下：

1. renderLinkDest 函数

该函数用于渲染超链接地址，可以在 node 参数中获取到链接地址。因此我将此时获取到的链接地址时保存在变量 myLink 中，便于 renderLink 函数中使用。

```
renderLinkDest: (node, entering) => {
    if (entering) {
        this.myLink = node.TokensStr()
        return [``, Lute.WalkContinue]
    }
    return [``, Lute.WalkContinue]
}
```

2. renderLink 函数

按照上面图中的调用顺序，在 entering 为 true 时，renderLinkDest 函数还没有执行，因此 myLink 为空，此时我们返回空[``, Lute.WalkContinue]。而在 entering 为 false 时，renderLinkDest已经行完毕，因此变量 myLink 已有值，我们在此时返回完整的 html 超链接代码。

```
renderLink: (node, entering) => {
    if (entering) {
        return [``, Lute.WalkContinue]
    } else {
        return [`<a href='${this.myLink}' my-attr='自定义属性'>${node.Text()}</a>`, Lute.WalkCon
inue]
    }
}
```

按照以上方法处理后，会渲染出如下代码：

```
<p>"觅好图"<a href='http://mihaotu.com' my-attr='自定义属性'>觅好图</a></p>
```

页面上多了一个链接文本，这是由于调用链中默认的 renderLinkText 函数渲染的结果。我们只要在函数中返回空即可。如下：

```
renderLinkText: (node, entering) => {
    if (entering) {
        return ['', Lute.WalkContinue]
    } else {
        return ['', Lute.WalkContinue]
    }
},
```

到此，在超链接中增加自定义属性的需求算是完成了。但有几个遗留问题：

1. renderLink 在函数中无法通过 node.TokensStr() 获取链接地址，却可以通过 node.TokenLen() 取字符个数（不清楚此时字符是什么）；

2. 无论 entering 为 true 还是 false，renderLink 函数中都可以通过 node.Text() 获取到链接文本。果 node 是嵌套链结构，那应该取不到任何内容呀！

3. node 的 child 和 parent 到底该如何获得？

建议：

为了便于开发，建议在所有 render 下级函数执行完毕后，都能将结果同步到根 note 中。或提供函，获取相应的值。

题外话：

由于 method.min.js 和 index.min.js 不可同时引入，如果页面上有 Vditor 编辑器时该如何自定义渲呢？官方文档好像没的写，这里记录一下，只要在 options.after() 中加入如下代码即可：

```
after: () => {
    this.contentEditor.vditor.lute.SetJSRenderers({
        renderers: {
            Md2VditorIRDOM: {   // 请根据不同的模式选择不同的渲染对象
                renderLink: (node, entering) => {
                    if (entering) {
                        return [``, Lute.WalkContinue]
                    }
                    return [``, Lute.WalkContinue]
                }
            },
        }
    })
}
```