核心编辑器需求文档

概述

笔记类的软件核心是编辑器. 编辑器做出来之后, 后续的其他功能会结合产品定位和自身实际情况来做规划, 现在首要目标是做出来一个主体编辑器

现在市面上笔记类app很多, 但是大部分笔记类应用都在致力于打造自己的护城河, 在一个编辑器上面添加各种功能. 他们通过这种方式加深用户粘度, 增大用户更换工具的成本. 但是有时候就忽略了编辑器本身的性能和便捷性. 这属于是大公司的通病, 任何方向的调整都要去考虑大体量自带的惯性问题.

现在市场上在蚕食这块蛋糕的人反而是一些新进公司和一些个体开发者. 只要软件做的有特色, 就会圈住一些用户. 但是他们的问题是做到一定的量就没有办法在做大. 还要为了留存, 在软件中不断的向老用户妥协来达到续订的目的.

目标

我们的目标就是要做个性能不错且设计合理的编辑器

对于重点在笔记的编辑器来讲, 每个人对笔记的需求不一样, 有专注录音,专注无边界的画板,或者使用触控笔的笔记软件, 这些都属于细分领域, 我们目前重点就放在大众笔记软件中, 并且优先考虑的是PC端. 笔记类的软件目前只有和键盘在一起使用,才能发挥出最大的效率. 手机和Pad, 我更倾向于作为补缺口的作用. 所有的设计在初期都应该是基于PC端的.

核心的功能

根据我们的目标, 核心的问题其实就是性能和设计, 设计的东西我后面慢慢推敲出一个详细的文档, 至于性能的话, 先从技术的角度来看一下, 能否优先解决下面的这些问题.

- 支持超长文章的加载, 在第一次加载的时候, 不能有长时间的卡顿现象. 最好是秒进
- 超长文章的滚动, 在超长文章快速滚动的时候, 不能有抖动和卡顿现象. 闪屏问题
- 输入文字的时候, 要能够实现快速的录入, 很多软件在文字多的时候会有轻微的阻塞感.
- 输入中文的时候, 要保证输入法的智能提示一直在正确合适的位置.
- 编辑器窗口可以手动调整大小, 在快速的调整大小的时候, 当前的可视窗口可能会要求重排, 可能会造成视觉上的卡顿.

基本的操作

- 光标的位置, 只需要支持一个光标即可.
- 选区的准确度. 选区高亮, , 支持快捷键选择多个选区.

- 支持选区的快捷设置, 比如单击是光标, 双击是选中当前的单词, 三击是选中段落.
- 插入图片.

编辑器的格式

解决完性能问题在考虑后面的格式, 我最开始做这个软件的时候, 先搞的格式, 后来发现性能有很多问题. 又重新推倒重来了.

目前流行的格式就是富文本和markdown. 首先要明确的一点是我们这两种格式都是要支持的, 至于其他的形式, 比如思维导图, 大纲笔记等其他的形式暂时先不考虑, 现在的重点是如果同时支持富文本和markdown, 需要如何整合在一起. 因为这两种格式不兼容. 下面有几种做法:

A.
现在大部分的软件通常是让用户自己选择喜欢哪种格式.比如用户选择的是富文本, 那么编辑器会出现富文本的工具栏. 如果选择的是markdown. 就会出现分栏或者所见即所得的形式来编写markdown. 这是一种标准的做法, 当用户选择格式之后, 当前的文章底层的保存格式也是也是md文件或者rtf文件.

B.
我尝试过一种做法就是把两个格式糅合在一起. 通常情况下是在富文本里面添加markdown, 这就需要在开辟一个独立的区域, 比如富文本->markdown->富文本, 如果用户要想写markdown的话, 可以在这个独立的区域中来使用markdown语法, 这样子理论上是可以实现的, 但是最终的产物就必须是我们软件的一个自定义格式.

C.
还有一种做法是完全模糊了富文本和markdown的概念, 采用自己的格式, 但是自己的格式又去兼容了富文本和markdown规则, 这种能统一用户的行为, 但是明显的增加了用户的学习成本.

目前来讲, A是目前最合适的, 但是也并不是说完全不考虑其他的做法, A的问题就在于想增加同样的功能, 比如数学公式, 两种格式都有自己的实现方式. 但是在B里面, 主体是富文本, 至于数学公式,代码块与markdown 是同级的, 都是属于一种内嵌的方式.

格式的标准化问题, 编辑器应该支持大部分的格式的导入和导出. 比如普通文本, PNG. PDF, markdown等.