链滴

自己开发的一款基于节点的逻辑导图笔记软件

作者：yetao

原文链接：https://ld246.com/article/1625920532815

来源网站：链滴

软件已经开源上传至github啦,喜欢的小伙伴可以去看看嗷: https://github.com/yetao0806/NodeNote

一、设计灵感

（一）当前背景

随着GUI框架的不断发展，纸质笔记逐渐有了它的弊端，比如说图片插入/书写代码/修改内容等繁杂，于是就涌现出了电子化笔记。目前市面上主流的笔记软件分为两个派系，一类是对应于IOS端的平书写类型笔记，这种笔记软件主要是优化纸质书写的弊端，提供了一定的便捷性；另一类是对应于PC的笔记软件，其有两种类型：文字类笔记以及导图类笔记，对于大多数用户来说一般是用文字类笔记行记录内容，通过导图类笔记整理大纲，方便记忆与整理。

（二）灵感来源

在体验过众多的笔记软件之后，例如说md文档类型的思源笔记，Notion笔记等等，以及XMind等思导图软件后，我们不难发现他们都有一个共同的弊端。

对于文字类笔记来说，逻辑表达完全取决于文字，举个例子来说：当我们在作操作系统这门课程的笔时，学到死锁这一节，要我们解释什么是死锁，用文字来表示就是如下：

是指两个或两个以上的进程在执行过程中，因争夺资源而造成的一　种互相等待的现象，若无外力　作用，它们都将无法推进下去。称此　时系统处于死锁状态或系统产生了死锁。

我们不难发现，当我们阅读文字的时候，我们会将文字所表达的逻辑在我们的脑海中表示出来。这样有几个弊端：

1. 在涉及大规模知识结构的时候，我们只能通过文字列表或标题的形式去区分每个知识模块，但是知模块和知识模块之间是有联系的，这种表示方式割裂了它们之间存在的逻辑关系。

2. 我们在读取文字的时候，会多出一个“将文字转换成自己所理解的逻辑图像”的步骤。

对于思维导图类软件来说，虽然可以部分解决文字笔记的知识模块之间的联系，但是对于逻辑的表示不是很清晰：节点与节点之间的联系只有一条线，而对于这条线表示什么逻辑却没有办法说明，典型例子就是XMind这个软件，例如我们从A节点导出一条线连接到B节点，我们无法知道A和B是什么关，仅仅知道它们是有关系的而已。同时在市面上所有的思维导图软件中，都不支持节点内的复杂编辑富文本/代码高亮/表格/图片/附件/子节点概念等等，对于复杂逻辑的表示很不友好。

因此为了解决上述弊端，我们团队开发了一款基于节点的逻辑思维导图软件，它不仅支持节点的复杂辑，同时对于UI的用户自定义也做到了极致，用户可以自定义其中任何一个部件的颜色/宽度；而对至关重要的复杂逻辑表达，我们引入了逻辑真/假的端口以及逻辑与或非门的支持。

二、软件介绍

（一）节点内容的复杂编辑

通过支持节点内的各种类型编辑，从而打破市面上已有的思维导图的局限性，让每个节点能记录的内多样化。

l 文字的大小设置

l 文字的颜色设置

l 文字的加粗\细

l 文字的斜体

l 文字的下划线以及删除线

l 图片的插入

l 代码的高亮

l 数学公式的编辑

l 段落的左对齐，右对齐，居中对齐

l 列表与列表嵌套支持

l 表格与表格嵌套支持

l 支持超链接

l 同时对于节点的宽度没有任何限制，如果你想，甚至可以无限大

图1：节点内部编辑内容

（二）节点复杂逻辑表示的办法

针对于市面上思维导图的逻辑表示不完备所设计

l 对于一个节点与另一个节点的逻辑关系，我们需要有这两个节点的真值表示：同样以死锁为例子。

例如说，A节点是不剥夺条件（产生死锁的其中一个必要条件），而B节点是死锁，那么当A节点真值真的时候，B节点可能为真。而A节点真值为假的时候，B节点一定为假。

因此我们采用了输入为真（左上角），输入为假（左下角），输出为真（右上角），输出为假（右下）四个端口去表示这个逻辑。当我们从A节点的输出为假端口引出这条逻辑线的时候表示，输出节点真值为假，即A节点真值为假。而B节点作为输入节点，也有两个输入，一个为真，一个为假，也代表自身的真值。这样子就表示了上述死锁的其中一种逻辑关系，具体见下图的详解：

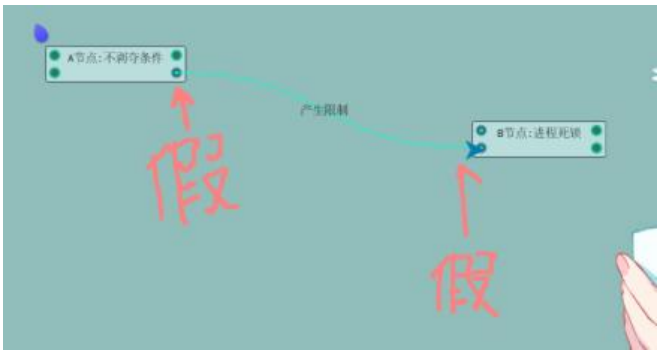

图2：真值的引入

l 我们通过上面已经表示了什么时候B节点的进程死锁真值为假，但是我们还缺少令它真值为真的情况因此我们考虑什么时候它的真值为真。再看死锁的必要条件，一共有四个：不剥夺条件，请求和保持件，循环等待条件，互斥条件。只有当四个条件全部为真的时候，B节点才能为真。

好，现在我们如果不引入任何东西，就使用现在的表示逻辑，我们只能够如此表示：

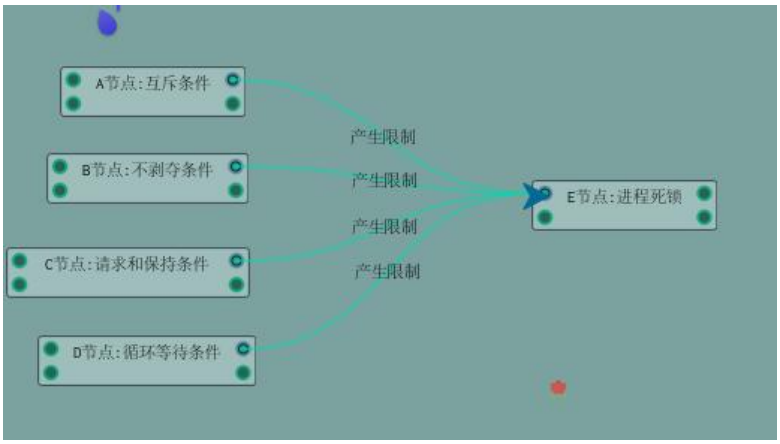

图3：错误示范

当我们如此表示的时候，我们发现这样子的逻辑是说：A,B,C,D中任意一个节点真值为真了，都能让E点真值为真，这不是我们想要的逻辑表示。我们想要的表示应当是A,B,C,D全部为真的时候，E才能为，因此我们引入了一个逻辑控制组件：与或非门，如图所示：逻辑控制组件对于输入和输出有三种条：分别是And, Or, Not，当我们将A,B,C,D四个节点的输出放到输入与门上表示，只有四个节点的真值全部为真的时候，它们才能算作真值为真。而逻辑控制组件的输出为与门表示如果有多个节点，那所被输入的多个节点都能作数。

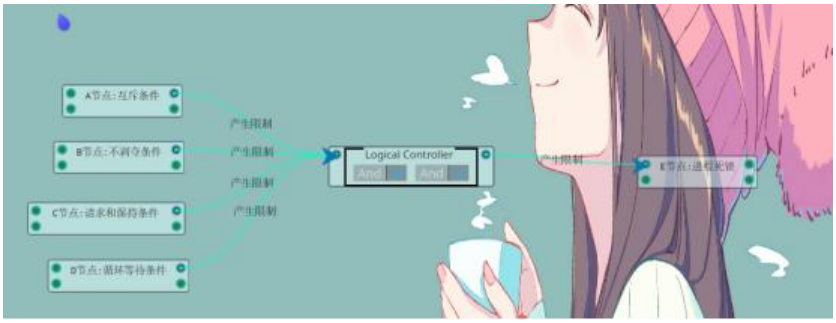

图4： 逻辑控制组件的引入

我们再举个例子，对于逻辑控制组件的输入为或门（Or）的时候，表示只要其中一　个条件成立，么这条逻辑链才成立。同样用进程死锁的例子来表示：进程死锁产生的原因有很多，比如说对系统资的竞争，进程推进顺序非法，信号量使用不当等。这些原因只要出现一个，那么就会产生进程死锁。

们用或门（Or）连接这几个条件，表示只要有一个为真，那么它们的组合才为真，逻辑链才能走通。图所示：

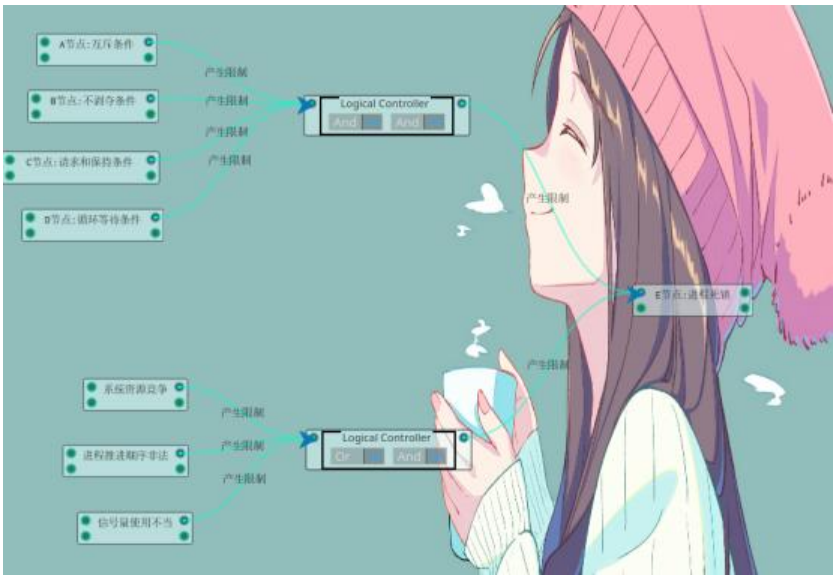

图5： 逻辑控制控件的使用

l 好，目前我们已经处理了基本的节点与节点之间的相互限制的逻辑。但是我们考虑这种情况：如果们需要表示知识模块和知识模块之间的关系应该如何表示呢。因此，我们引入一个从属于的子节点概。即我们可以在节点内嵌套其他子节点，形成一个知识的小模块。如图所示：我们的子节点是网格布，允许子节点之间的位置调整。

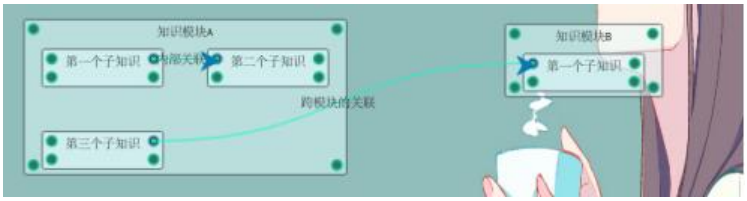

图6：子节点的引入

l 好，接下来我们再考虑一种情况，如果我们的知识模块过大应当如何处理呢。如果还采用这种子节无限制的嵌套，一个知识模块有几百层这么长，虽然我们支持搜索功能，但是这样对于我们的记忆是方便的，我们的大脑喜欢一小块一小块的记忆，因此我们引入了一个子图的概念，对于每个节点，我都可以进入其子图，子图内又是一个新的场景，我们可以在子图里表示我们想要的复杂逻辑，从而优子节点过长带来的烦恼。如图所示，我们可以进入知识模块A的子图，进行复杂编辑，其内部是一个新的场景，可以添加这些部件。而对于侧边的子图目录我们有升序排列和降序排列。

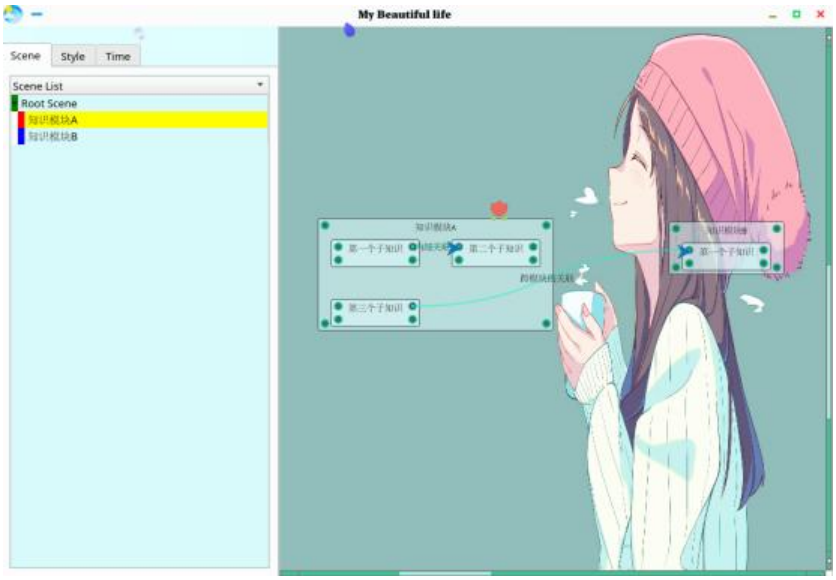

图7： 子图的引入

（三）其他方便的基本功能介绍

l 支持搜索功能

l 支持线条动画，即有一个小球在逻辑线条上滚动：如果是点击节点开启小球滚动，那么所有与之相的逻辑节点全部线条都开启滚动，如果是点击线条开启小球滚动，那么只滚动当前线条

l 支持碰撞检测，节点与节点的碰撞，节点与线条的碰撞等

l 支持自定义线条绘制，图2的红色字体即是自定义线条绘制

l 支持辅助线对齐，在移动节点的时候会出现最近的其他节点的边的辅助线，方便对齐该节点

l 支持撤销与重做

l 支持附件功能，允许节点内插入无限制个数的无限制类型的附件

l 支持当前子图的放大与缩小

l 支持OpenGL硬件加速绘制

l 支持背景图片的更改，每个子图拥有默认的背景图，允许更改成喜欢的

l 支持飘落部件的图片更改，如图7飘落的花朵，允许修改成自己想要的图片

l 支持自定义全局样式，当前子图样式，当前选中部件样式，优先级递增

l 支持每日软件使用时间统计

| 支持导出场景和选中控件为图片