**TiddlyWiki（太微）简易指南**

*这是一篇用实践讲出的一条路径答案。*

*一篇以【活用条目建构笔记】为主的想法，主张最少知识和轻量配置。*

*如果按照路径读通整篇可以省下不少力气的避坑指南。*

*这里有最简单的使用方法和入门经验，还有笔记需要的常用插件以及介绍，基本的wikiText语法，基础界面的认知和使用，目前常用的笔记方法论，教你如何自主学习解决有关TiddlyWiki的相关需求与问题，快速开始构建太微的使用环境，另外基本你想的到的问题都可以在这里找到。*

此文件所在的Github仓库：[tiddly-gittly/TiddlyWiki-Easy-Guide-And-TiddlyMemo-Edition](https://github.com/tiddly-gittly/TiddlyWiki-Easy-Guide-And-TiddlyMemo-Edition)

## 前言

太微比普通的笔记软件拥有更加强大的灵活性（这种灵活性是需要知识来支撑）。太微就像《果宝特攻》里面的机甲，所有的功能（包括更改外观）都可以通过安装、卸载配置模块（插件）实现。太微同时也是文档与软件一体化工具。普通软件修改外观等于几乎打碎整个UI重做。在太微中最简的使用方式是仅在桌面端通过edge打开[TiddlyStow](https://twpub-book.netlify.app/tiddlystow.html" \t "dfu)页面，选一个太微模板就可以使用记录笔记或日志，数据会保存在本地。

对我来说，我喜欢主动性加各种对条目的整理与使用方式、思考、组织和检索方式。太微不适合收集信息，它只适合你去手动整理自己得到的信息，然后把它转换为自己的知识。这是一个非常重要的过程。而我喜欢这个过程。因为太微正如数学题未得出答案之前有很多可能性，我需要的是其中的一个确定的答案。在大家的共同努力下，花费了许多时间和精力制作了这篇线性指南，旨在以内容为中心、书写文本与显示文本的美观容易阅读、简化原本需要几个月的学习时间成本、授之以渔的自主学习策略（除非你用的到，否则不要轻易的增加需求）。

我们都知道信息和知识是完全不同的。信息只是在神经元之间流转了一遍，然后就存到浅层记忆区了，如果后面不反复刺激的话，甚至还会被自动修剪掉。然而知识是神经元的连接塑性。而知识的学习需要的是主动性，主动的获取某些知识，并不断加强对它的训练。就像学骑自行车一样，一旦掌握某项知识，可能一辈子都忘不掉。太微的【条目】正是提供了这一种对知识整理整合的载体。每一个条目都可以看做一个知识卡片，可以容纳非常丰富的信息，并且他们可以互相属于、链接。你可以在这些互相管理的“知识卡片”中发现知识。同样在不断整理它们的时候，你也会在这个过程不断思考逐渐进步。

有人说TiddlyWiki真的太难了，

想看看比较惊艳的太微实例嘛？点击链接查看在线部署的太微：

<https://oeyoew.fun/>

[🤓🌱 学习者的数字花园 — 记录我的学习与思考 (pimgeek.com)](https://pimgeek.com/notes/)

<https://tritarget.org/>

<https://xememex.com/ethicsatwes/>

<https://tiddlymemo.org/manual/zh-Hans>

**目录**

[前言 ii](#_Toc138951939)

[一、创建自己的知识库 1](#_Toc138951940)

[（一）TiddlyWiki 简介 1](#_Toc138951941)

[（二）下载与安装 3](#_Toc138951942)

[（三）太微的基本介绍 4](#_Toc138951943)

[二、开始使用知识库 15](#_Toc138951944)

[（一）创建第一个条目笔记 15](#_Toc138951945)

[（二）“文件目录”侧边栏 32](#_Toc138951946)

[（三）筛选器与双链功能 34](#_Toc138951947)

[三、如何使用太微建构笔记系统？ 46](#_Toc138951948)

[（一）时间序列记录法 46](#_Toc138951949)

[（二）元素渐构笔记法 48](#_Toc138951950)

[（二）关系图谱或图数据结构 52](#_Toc138951951)

[（三）线性写作的形式 56](#_Toc138951952)

[（四）渐进写作（增量写作） 57](#_Toc138951953)

[（五）自动化无压记录 57](#_Toc138951954)

[（六）随手记录，动态生成 60](#_Toc138951955)

[（七）卡片笔记写作法 61](#_Toc138951956)

[（八）PARA个人知识管理 61](#_Toc138951957)

[（九）书籍与书籍的分卷 62](#_Toc138951958)

[（十）分享和发布功能 63](#_Toc138951959)

[（十一）数据安全性与备份 63](#_Toc138951960)

[四、为知识库拓展新的功能与特性 65](#_Toc138951961)

[（一）Markdown插件 66](#_Toc138951962)

[（二）反向链接插件 66](#_Toc138951963)

[（三）TOC自动化目录插件 67](#_Toc138951964)

[（四）回收站插件 69](#_Toc138951965)

[（五）随机漫游条目与快照插件 70](#_Toc138951966)

[（七）与SuperMemo互动 71](#_Toc138951967)

[五、帮助与更多 73](#_Toc138951968)

[（一）桌面端Timimi的安装使用 75](#_Toc138951969)

[（二）配置太微的中文语言 80](#_Toc138951970)

## 一、创建自己的知识库

很久很久以前，在一条河流里，生活着许多许多的小鱼儿（Tiddlers），它们快活地在故事河（Story River）里游来游去。这些小鱼儿每一个都有自己的名字（条目名），自己的种类（条目内容类型），它携带的信息组成了鱼儿美丽的身体与骨架，附加的字段构成了它美丽的鳞片。

### （一）TiddlyWiki 简介

TiddlyWiki 是一个宝藏软件，但可惜的是有些人没有体会到这一点就放弃了。我们把互联网看作一个巨大的知识库，它由每个网页组成。使用TiddlyWiki进行信息组织，就是编辑一个一个的网页（条目），这些网页（条目）组成自己的知识库。

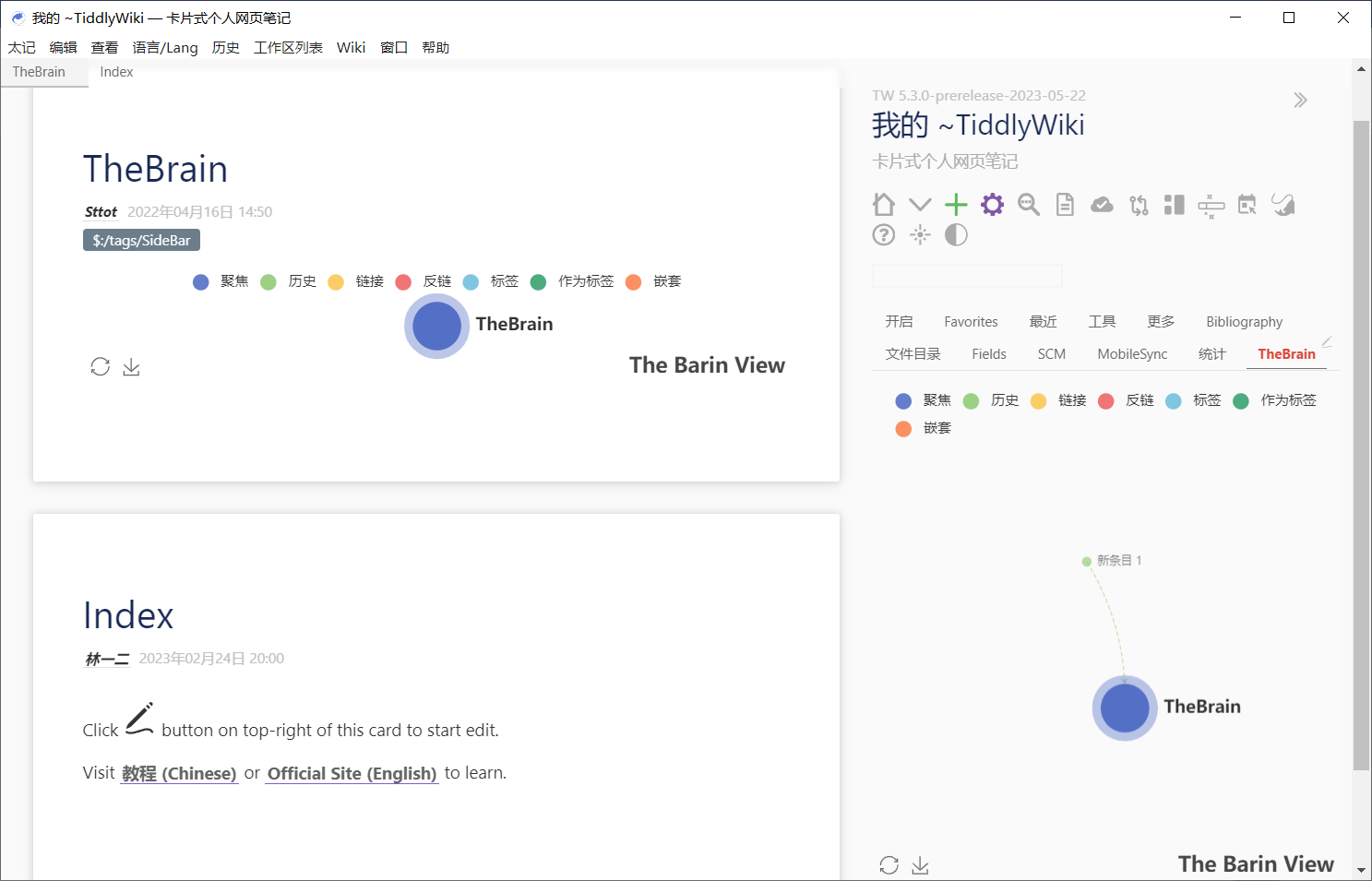
「太微」是我们对 TiddlyWiki 的中文称呼。它是一款独特的非线性笔记本，用于捕获、组织和分享复杂信息。可用它来建立个人知识库，或者任务管理。记录闪过你大脑的每一个想法，或者建立一个灵动的自适应式的网站。

*本段引用来自TiddlyWiki xp作者keatonlao（github）的话。*

「太微」不适合【剪藏】一些信息，知识的学习是需要主动性的探索而被动的【剪藏】就像平静的水面泛起的水花转瞬即逝。不过，仍然有非常多成熟的软件可以完成【剪藏】。对于太微来说，我们需要自己的耐心去慢慢培育出一条条美丽的属于自己小鱼儿，让它们遨游在自己的故事河中，成为自己知识体系的一部分。

我们会使用TidGi（太记）作为本文的全部主线来为您介绍如何使用TiddlyWiki，您以后会见到各种各样的TiddlyWiki，比如：TiddlyMemo（墨屉）、TiddlyDesktop（TiddlyWiki官方的桌面启动器）、TiddlyWiki Xp（一个快速体验太微的｢太微模版｣）等等。这些都是不同愿景和功能的TiddlyWiki。其中TidGi（太记）是一个应用程序而非系统或框架，使用简单且功能丰富。

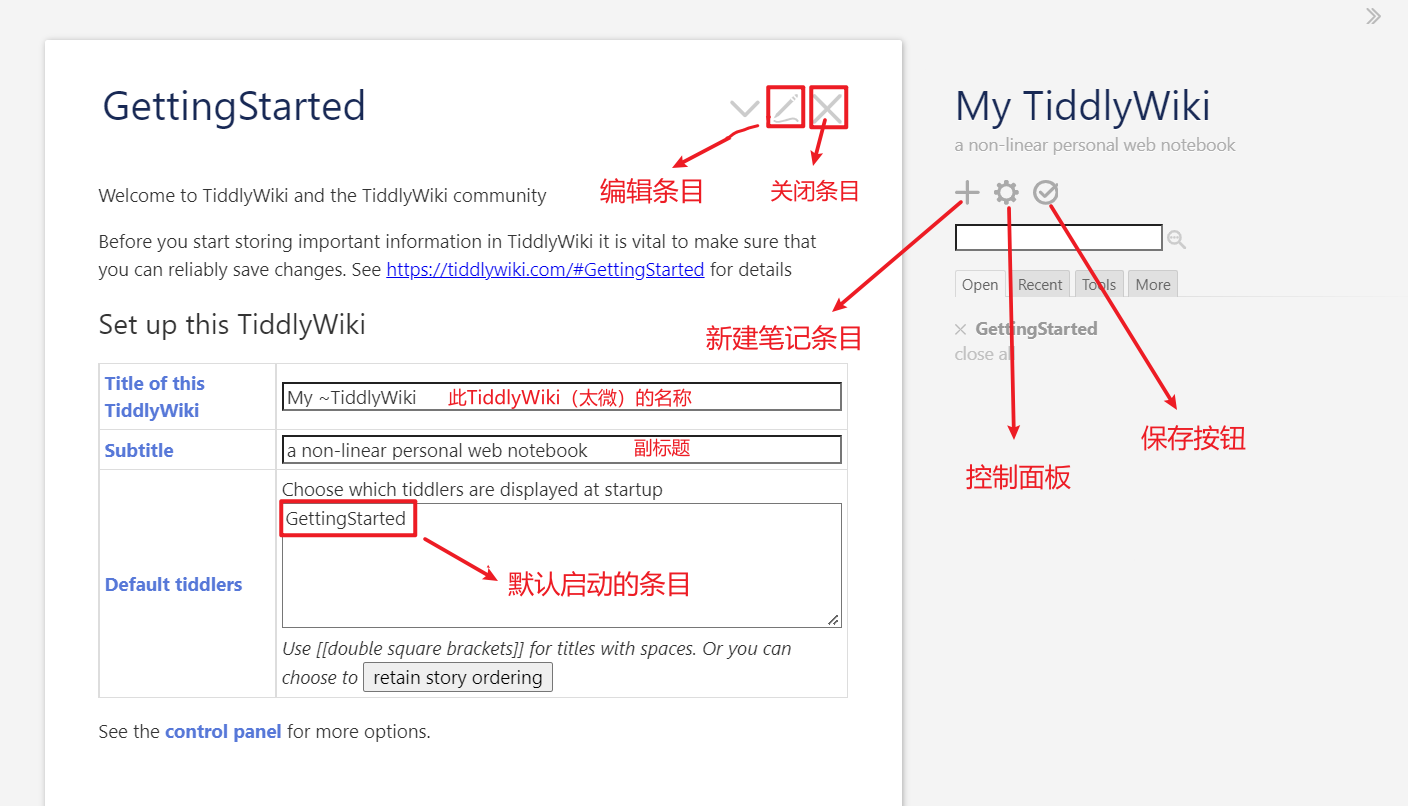
准备好了吗？那让我们开始吧！



### （二）下载与安装

TiddlyWiki（太微）是一份存储在本地的HTML网页文档！同时也是以另一种文件夹存储的方式运行于NodeJS的Web程序。请下载TidGi，然后创建打开你的第一份Wiki。

TidGi下载链接：[Releases · tiddly-gittly/TidGi-Desktop (github.com)](https://github.com/tiddly-gittly/TidGi-Desktop/releases)



如图所示，太微的布局在大概被分成两个部分，左边是由条目组成的信息流称为故事河，右边为按钮（页面工具栏）和侧边栏组成的工具区域。

### （三）太微的基本介绍

这部分的内容了解一下有个印象就可以。

#### （1）概念：模板、文件、条目

1.1太微模板与条目模板

太微模板是指预制的太微，相对于空白太微而言根据作者的需求和想法包含了一些功能性的插件和美观性的主题布景以及为了满足插件之间依赖关系或者整体性设计而封装的具有明显功能性的太微

条目模板是指预制的具有特定布局和样式的模板条目。它通常是为了美化和简化输入内容或瞒住特定需要的一种特殊的条目，比如在xp中你可以通过引用条目模板瞬间美化当前的条目使其变得好看，在墨屉中，通过引用问答或者填空模板进行渐进式学习。

1.2太微中两种通用文件格式

Tid文件是太微中条目文件的文件类型，一般只有在Nodejs（文件夹形式）的太微中才可以看到。单文件中的太微是看不到的，因为存储在HTML文件中了。

JSON文件是太微中常见的数据存储格式文件，一般都是使用它保存太微中封装好的插件（未封装的插件用条目保存），TWPUB格式的书籍也会使用它存储数据。不过导出一个或多个笔记时，使用它也是不错的选择。其它的插件也会使用它存储导出的数据，比如TiddlyMap会使用JSON格式保存图文件。

1.3太微中条目的类型

*条目会在故事河里显示，大致分为两种类型，普通条目和系统条目。*

**普通条目**，包含：tile 条目标题、tag 标签、text 条目内容、type 内容类型、field 栏位

1. tag 标签包括系统标签以及普通标签，其中系统标签被【太微】用来给条目提供特殊的行为。
2. field 栏位包含field name 字段名、field value 字段值。条目字段：可以想象成一个个的空位，能够放置除了文本信息之外的内容，即自定义的拓展的小鱼内容。

**系统条目**：如果Tiddlers的标题以特殊字符串开头，则将其归类为系统Tiddlers。系统条目大致有一下几种类型。插件、默认、模板、样式、配置等等。

* 插件（条目组）：在内部，插件是一些条目被打包成可被安装、拷贝、禁用或删除的单一条目。插件中的个别条目呈现为影子条目。插件可以包含JavaScript模块、样式表和模板。插件也可以用于散布普通的文本、图片或其他内容。 （扩展内容：插件机制）
* 默认条目：或者说影子条目（ShadowTiddlers）是来自插件内加载的条目。默认条目可以被具有相同名称的普通条目复写，成为影子条目。如果随后删除了该条目，则将自动还原为原始的默认条目。来自插件内加载的条目本身不可删除不可修改。因此删除了影子条目就意味着删除了对默认条目复写。
* 模板条目：标题为，$:/templates/<用途-备注>
* 样式条目：标题为，$:/<name>/stylesheet/<name>。为条目添加样式表系统标签（系统标签：$:/tags/Stylesheet），在里面写的 CSS 就会自动应用到全局样式上，并覆盖别的 CSS ，详细说明访问太微中文教程：添加css样式。
* 配置条目：标题为，$:/config/<name>
* 调色板条目，标题为$:/palettes/<name>，需要添加$:/tags/Palette系统标签使其生效。

可选的知识：[字段到底是什么呢？——是什么组成了条目。](https://tw-cn.netlify.app/#%E5%AD%97%E6%AE%B5%E5%88%B0%E5%BA%95%E6%98%AF%E4%BB%80%E4%B9%88%E5%91%A2%EF%BC%9F)

#### （2）太微的布局和控制面板

在TidGi中，你可以通过拖动侧边栏的左侧边缘的一个可拖动的边框，当鼠标移动到左侧边缘时会看到这个边框。拖动这个边框可以改变侧边栏的宽度，拖动到靠近窗口右侧边缘时就会自动收起侧边栏。

如果你使用空白版的太微，而你觉得空白版的太微页面的左右比例看起来太别扭，你可以通过以下步骤调整页面左右的比例。如图所示，点击开启控制台按钮。在控制面板->视觉外观->主题调整->侧边栏布局。侧边栏布局改为：**浮动故事固定侧边栏。**

图表 1关于侧边栏选项卡的知识。



你可以打开视觉外观 - 主题调整 - 选项 - 设置【置顶标题】。使条目名称"黏着"于浏览器窗口的顶端。

这对修改编辑时非常有用，不然如果写的文章非常长的时候，你就要经常性的手动滚动到条目顶端的标题位置点击保存或者取消按钮。

主题是对页面工具栏、侧边栏、故事河等所有UI界面的整体布局。对新用户来说，主题是一个常用的功能，足够的稳定和简单的主题能够让用户放松，把更多的时间用在写作与内容的组织上。主题可以用来适应不同的使用/应用场景。

我们为你介绍这两款主题：Readonly（只读主题）、Centralised（居中主题）

1. Readonly（只读主题）它隐藏了编辑和新建按钮，你可以应用在只希望别人查看或者网站不支持实时修改的地方。
2. Centralised（居中主题），它使得故事河的整体的布局为居中显示。

在默认的【Vanilla】与【Snow White】布景主题中，你可以设置“固定侧边栏，浮动故事”缩小侧边栏的大小，以让故事所占的区域变大一些。而Sidebar Resizer插件则可以让你通过拖拽调整侧边栏的大小。



你可以在控制面板 - 视觉外观 -【工具栏】看到每一个图标的具体名称和作用。也可以设置它是否在对应的位置显示。工具栏有四个位置，分别是编辑工具栏、编辑器工具栏、页面工具栏、查看工具栏。



* 编辑工具栏在条目编辑状态下右上角，分别对应删除、取消和确定。
* 编辑器工具栏是在条目编辑状态下的文字加粗、标题格式等快捷按钮。
* 页面工具栏，就是上图中标注的。默认有三个按钮，分别是新建条目、控制台按钮、保存按钮。页面工具栏的下面是搜索框，搜索框的下面就是侧边栏。
* 侧边栏默认由开启、最近、工具、更多等选项卡组成。
* 查看工具栏对应条目显示状态下右上角的三个按钮所在的区域。分别是更多、编辑和关闭。你可以选择性的添加这些按钮。

设置网站图标：将图标导入后，命名为 $:/favicon.ico 覆盖原有文件。

安装好的插件，如果需要配置插件，一般是在 控制面板 - 插件 -相关的插件条目中的config或setting选项卡，或者在控制面板的 设置 - 插件名选项卡中配置插件。也有部分会放置在控制面板的视觉外观选项卡，工具栏组。



在控制面板的信息选项卡，你可以查看条目的总数，【一般条目数量】就是你写的笔记条目的数量，也称为标准或者普通条目。其它则是系统条目。在这里，你也可以设置TIddlyWiki的一般性信息，比如标题、副标题、、编辑者署名、默认打开的条目等信息。

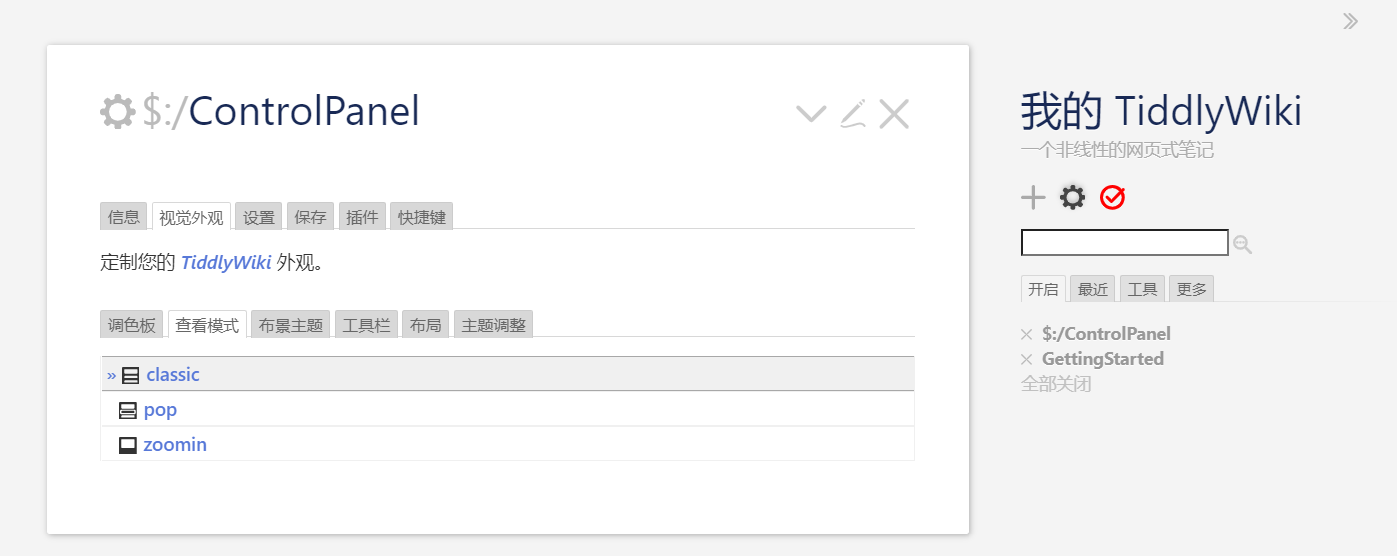


在信息选项卡的高级组，你可以查看已加载的样式表、条目栏位、级联等信息。要注意的是现在不要太过于关注它们，仅需要了解即可，等到需要使用的时候在学习也不迟。

#### （3）太微的查看模式

* Classic：经典模式，打开条目是从上往下的动画，关闭是往左移动直到消失的动画关闭
* POP：弹出模式，pop打开条目是从大缩小的动画，关闭条目是继续缩小直到看不见，这两个他们都是多个条目同时在的，往下翻可以看到多个条目。
* Zoomin：缩放视图模式：这个模式只能看到一个条目，往下翻没有更多条目

详细链接： https://tw-cn.netlify.app/#查看模式



#### （4）配置新的主题配色

点击页面工具栏的小齿轮按钮（控制台）打开太微控制面板，在视觉外观选项卡 - 调色板组选择你喜欢的主题配色。最后点击页面工具栏的保存更改按钮保存你的太微。



可选的知识：调色板的主题配色是一个系统条目，条目名通常以$:/palettes/<name>命名，几乎所有太微可见部件的配色都可以通过创建或修改此类条目生效。

#### （5）中文插件库的安装使用

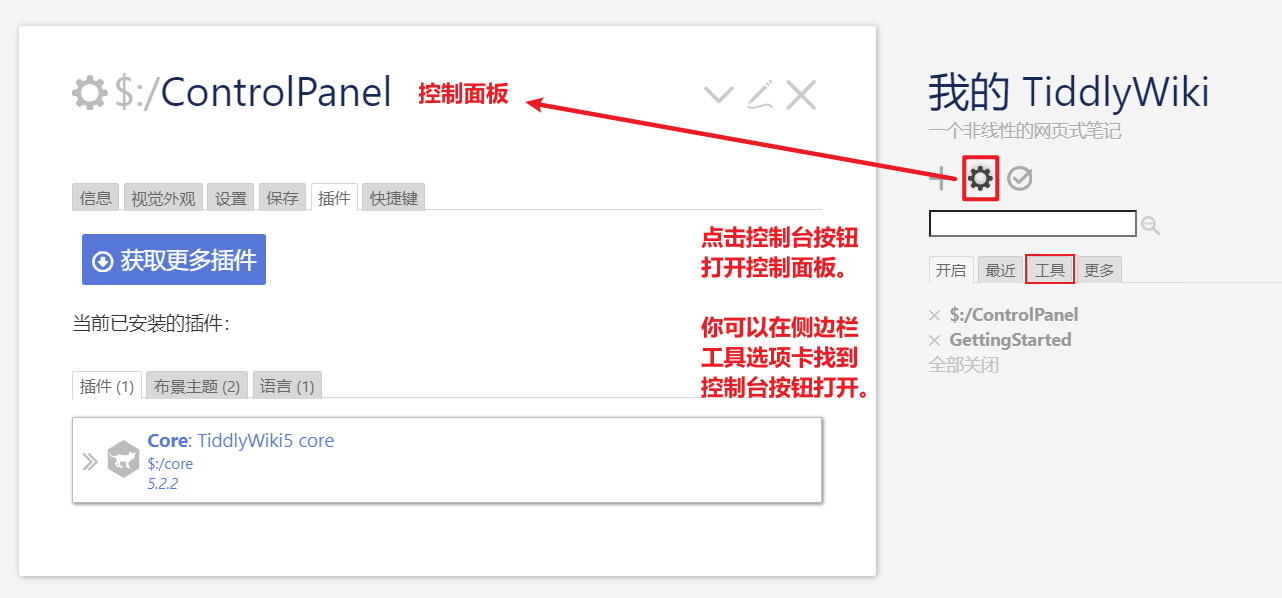
*CPL：指TiddlyWiki的中文插件库。CPL主页：https://tw-cpl.netlify.app/ ，CPL仓库：tiddly-gittly/TiddlyWiki-CPL 。*

5.1如何安装CPL插件库？

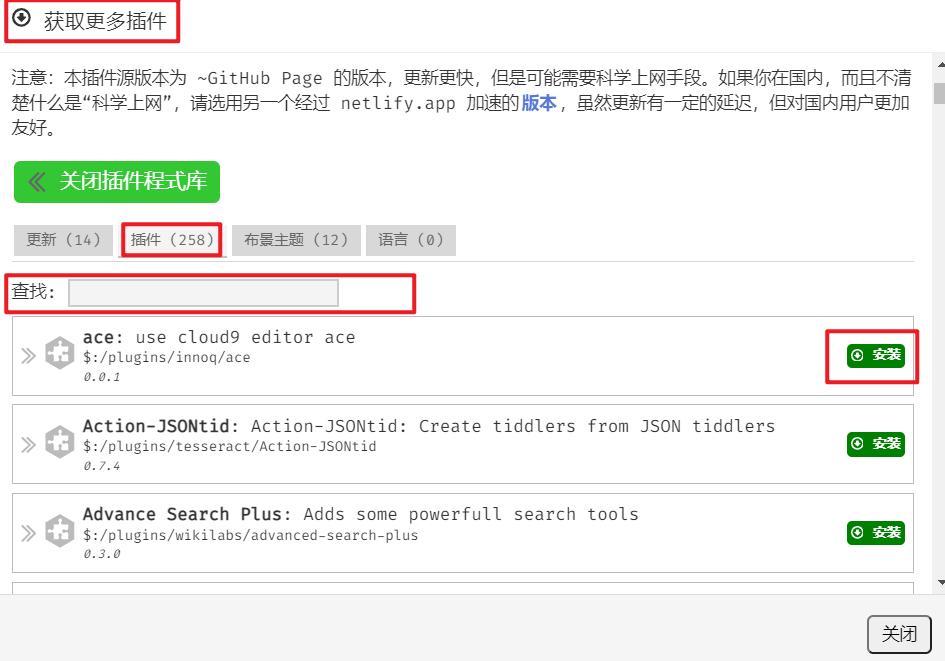
打开链接：<https://tw-cpl.netlify.app/#Welcome>，打开欢迎页面，拖动【太微中文社区插件源(大陆加速版) 】链接到你的太微，点击导入即可。

5.2如何安装CPL中的插件？

点击控制台按钮打开控制台。在控制台 - 插件选项卡 - 点击获取更多插件按钮 - 在弹出的对话框中选择【太微中文社区插件源】- 点击开启插件程式库按钮 ，这这里你可以选择更新插件、安装插件及布景主题。



**重要：**插件要尤为注意插件之间的依赖关系，缺少依赖将会导致功能性错误或者其它的不可知问题。安装插件之前请谨慎备份自己的TiddlyWiki。



太微的控制面板，是太微配置所有设置的窗口页面。一般的个性化配置都可以在控制面板找到，极少部分会在有相关功能的特定的条目中找到。

在控制面板的信息选项卡 - 高级组中点击【样式表】，可以看到所有系统预设或者自己自定义的样式，你可以快速在此处找到自己创建的样式并修改。

#### （6）不同功能多样性的太微模版

1. 官方空白版太微。
2. TiGi（开箱即用发行版）与ItonNote Wiki模板
3. TiddlyWiki Xp（快速体验TiddlyWiki）
4. TiddlyMemo墨屉（渐进学习与记忆）
5. TiddlyWiki Grok

TiddlyStow：[https://twpub-book.netlify.app/tiddlystow.html](https://twpub-book.netlify.app/tiddlystow.html" \t "_blank)

TiddlyStow（Github）：<https://btheado.github.io/tiddlystow/TiddlyStow-ZH.html>

Tiddlywiki-xp 主页：<https://keatonlao.github.io/tiddlywiki-xp/>

**墨屉关于英文原著的知识自建单词库。**

墨屉版太微可以导入twpub版英文原著，你可以使用墨屉渐进阅读英文原著，并使用注释功能注释生词，被注释的生词会显示单词查询结果，同时会加入到渐进学习的队列，当你阅读完成后，你可以使用渐进学习功能复习注释过的生词。这样的过程可以让你不断积累自己的生词库，然后把生词在复习加阅读的理解性输入的处理下变成熟词。

* 墨屉主页：<https://tiddlymemo.org/manual/zh-Hans>

tiddlyhost，只需要注册一个账号就可以使用的在线网站。

Tiddlyhost主页：https://tiddlyhost.com/

## 二、开始使用知识库

### （一）创建第一个条目笔记



在太微中，一个条目相当于一个文件，点击页面工具栏的新建条目按钮（十字形）创建一个新的条目。

我们为新建的条目取一个新名字——“开始使用知识库”。

**掌握少量的快捷键有助于提高效率。**

关于条目的快捷键：alt-N（新建）、ctrl-Enter（保存条目）、Esc（取消保存）

保存wiki快捷键：Ctrl + S、从条目标题跳转到条目内容快捷键：TAB

双击编辑条目插件：2click2edit

让我们开始书写属于自己的第一篇笔记吧。不过，首先我们要知道两条基本的WikiText语法——标题与段落。这非常简单。

为了快速使用，我们仅仅挑选了常用且简单的功能，更详细的资料情查看文档：https://keatonlao.github.io/tiddlywiki-xp/#WikiText

#### （1）指定标题

标题由一个或多个前导 ! 字符指定

|  |
| --- |
| 输入此条目的内容 |
| ! 这是一级标题  !! 这是二级标题  !!! 这是三级标题 |

|  |
| --- |
| 显示效果 |
| **这是一级标题**  **这是二级标题**  **这是三级标题** |

#### （2）段落格式

要在 [TiddlyWiki](https://keatonlao.github.io/tiddlywiki-xp/" \l "TiddlyWiki" \t "dlt) 中标记段落的末尾，您需要输入 enter 两次以创建空行。

无论两段之间的空行有多少，默认解释为两个段落，可以使用使用 <br> 强制换行

|  |
| --- |
| 输入此条目的内容 |
| 这是第一段。  这是第二段。  <br><br><br>  这是第三段 |

|  |
| --- |
| 显示效果 |
| 这是第一段。  这是第二段。  这是第三段 |

现在，你可以自由的书写自己的笔记了，是不是特别简单！不过，如果需要处理像诗歌一样的文本时，这样的按 enter不断换行也太过于麻烦了，为了解决这个问题，我们将为你介绍新的Wikitext语法——硬换行。

#### （3）硬换行内容块

处理诗歌等材料时，可以将内容块标记为包含硬换行，内容块使用三个英文状态下的双引号"""开头和"""结尾

|  |
| --- |
| 输入此条目的内容 |
| """  ''沁园春·雪''  [现代] 毛泽东  北国风光，千里冰封，万里雪飘。  望长城内外，惟余莽莽；  大河上下，顿失滔滔。  山舞银蛇，原驰蜡象，欲与天公试比高。  须晴日，看红妆素裹，分外妖娆。  江山如此多娇，  引无数英雄竞折腰。  惜秦皇汉武，略输文采；  唐宗宋祖，稍逊风骚。  一代天骄，成吉思汗，只识弯弓射大雕。  俱往矣，数风流人物，还看今朝。  """ |

|  |
| --- |
| 显示效果 |
| **沁园春·雪**  [现代] 毛泽东  北国风光，千里冰封，万里雪飘。  望长城内外，惟余莽莽；  大河上下，顿失滔滔。  山舞银蛇，原驰蜡象，欲与天公试比高。  须晴日，看红妆素裹，分外妖娆。  江山如此多娇，  引无数英雄竞折腰。  惜秦皇汉武，略输文采；  唐宗宋祖，稍逊风骚。  一代天骄，成吉思汗，只识弯弓射大雕。  俱往矣，数风流人物，还看今朝。 |

你会发现，【沁园春·雪】 这个标题被加粗了，这是因为在WikiText语法中''两个英文单引号'' 用于 **粗体文本** （**B**）

你已经掌握了标题、段落、硬换行以及加粗的WikiText语法，这真的太酷了。请试着写一些东西看看吧。下面我们将会为您介绍与加粗相关的文字的格式化WikiText语法：斜体、下划线、上标、下标以及删除线。

#### （4）格式化字体

现在我们将为你展示如何在太微中写一篇毛泽东的沁园春·雪

|  |
| --- |
| 输入此条目的内容 |
| //北国//风光，\_\_千里冰封\_\_，万里雪飘。  望长城''内外''，惟余莽莽；  大河^^上标^^上下，顿失滔滔。  山舞银蛇，原驰蜡象,,下标,,，欲与天公试比高。  须晴日，看红妆素裹，~~分外~~妖娆。 |

|  |
| --- |
| 显示效果 |
| *北国*风光，千里冰封，万里雪飘。  望长城**内外**，惟余莽莽；  大河上标上下，顿失滔滔。  山舞银蛇，原驰蜡象下标，欲与天公试比高。  须晴日，看红妆素裹，~~分外~~妖娆。 |

在编辑模式下，我们可以使用编辑工具栏的按钮设置我们需要的格式，也可以使用快捷键设置（控制台按钮-快捷键）



|  |
| --- |
| **斜体**，//两个斜杠// 用于 *斜体文本* ( ***I*** )  条目内容：//斜体文本//  显示效果：*斜体文本* |
| **下划线**，\_\_两个英文下划线\_\_ 用于 下划线文本 ( **U** )  条目内容：\_\_下划线文本\_\_  显示效果：下划线文本 |
| **上标**，^^两个英文上角符号^^用于 **上标** 文本 ( **X2** )  条目内容：^^上标^^文本  显示效果：**上标** 文本 |
| **下标**：,,两个英文逗号,, 用于 下标 文本 ( **X2** )  条目内容：,,下标,,文本  显示效果：下标文本 |
| **删除线**：~~两个波浪线~~ 用于 ~~带删除线~~ 文本 (**~~S~~**)  条目内容：~~带删除线~~ 文本  显示效果：~~带删除线~~ 文本 |
| **突显文本：**@@突显文本@@ 用于创建突显文本  条目内容：@@突显文本@@  显示效果：突显文本 |

需要更加丰富的功能？没问题，下面让我们为你介绍段落格式，如何使用项目列表、引述块、水平分割线破折号以及最为有用和重要的双链功能——**嵌入和链接**（显示入链和出链功能需要安装插件，我们将会在下一节为您介绍）。

对于程序员来说，我们还为你介绍代码块的使用方法。现在让我们一起来看一下吧。

#### （5）段落格式

##### 1）项目列表与引述块的简单介绍

为了文档的简易性，我们只列举项目列表与引述块比较常见的使用方式。详细的用法可以访问链接查看：<https://keatonlao.github.io/tiddlywiki-xp/#WikiText>

项目列表分为**有序列表**与**无序列表。**

**无序列表**：使用 \* 字符创建无序列表。

\* 第一个列表项

\* 第二个列表项

\*\* 一个子项目

\* 第三个列表项

* 第一个列表项
* 第二个列表项
  + 一个子项目
* 第三个列表项

**有序列表**：使用 # 创建有序列表。

# 第一个列表项

# 第二个列表项

# 第三个列表项

1. 第一个列表项
2. 第二个列表项
3. 第三个列表项

**引述块**的简单介绍

> 引述文字

> 另一行引述文字



您也可以像这样**嵌套引述**

> 顶级引述

>> 次级引述

> 另一顶级引述



##### 2）水平分隔线、破折号、数学公式、代码块

**破折号：**您可以使用两个 hyphen（连字符）-- 创建一个 n-dash（连接号），并使用三连字符 --- 创建一个 m-dash（破折号）。 例如：

-- 这是一个例子

--- 这也是

– 这是一个例子

— 这也是

**水平分隔线：**在一行上用三个或更多破折号，您可以划一条水平线。

**---**

**数学公式：**使用一对 **$$** 包裹起来的katex公式。需要tiddlywiki/katex插件，太记默认预装。

$$\begin{vmatrix}

a & b \\

c & d

\end{vmatrix}$$

**代码块：**段落使用 **```** 开头和 **```** 结尾作为代码块，行内代码使用 `代码内容`。注：【 **`** 】输入方法为英文状态下tab上面的按键。

这是正确的写法：

这是一个普通的段落

```

这将是等宽的

```

这是错误的写法：

这是一个普通的段落

```

这将是等宽的

```

在太微中，你可以插入更加丰富的功能，比如音频、视频、地图、HTML 代码、样式和类、宏、小工具、变量等等。这些都可与WikiText混合使用。它们可以动态的为你组合并显示更加丰富的内容。但这并不是本文的重点，因此只是在此列举一二。

##### 3）自定义样式段落块、类型段落块

**自定义样式段落块：**使用@@以定义的CSS名称或具体CSS样式属性，例如 color，每个后跟 ; 分号，可以在开头的 @@ 之后立即引入。这样做可以让被包裹起来的元素应用定义的CSS样式表，使段落表现起来更好看或具有特定显示风格。

@@background-color:lightcyan;

\* Item one

\* Item two

@@

附加：硬换行CSS，相当于硬换行内容块一样的效果： white-space: pre-wrap;

**类型区块（类型文本块）：**可以用明确的内容类型呈现文本区块，像这样。

$$$image/svg+xml

<svg xmlns="http://www.w3.org/2000/svg" width="150" height="100">

<circle cx="100" cy="50" r="40" stroke="black" stroke-width="2" fill="green" />

</svg>

$$$

#### （6）WikiText表格

[TiddlyWiki5](https://keatonlao.github.io/tiddlywiki-xp/#TiddlyWiki5) 使用垂线字符来格式化表格，如下所示：

|!单元1 |!单元2 |

|单元3 |单元3 |

感叹号用于指示标题单元格。 该示例呈现为：

|  |  |
| --- | --- |
| **单元1** | **单元2** |
| 单元3 | 单元3 |

更多的功能与需求例如单元合并、单元格垂直对齐、单元合并等内容不在详细介绍，需要的时候可以查阅tiddlywiki xp文档。（渐进学习）

#### （7）书写与显示文本

WikiText即TiddlyWiki书写文本。书写文本与显示文本又名书写风格——使笔记变得简洁易懂。为了使WikiText在视觉上变得简洁、易懂。一些好的规范的书写习惯是有必要的。另外在书写时，编辑区上面的编辑按钮将会提供非常大的助力，点击它们会为你自动插入一些标记符。

在输入标题这样一般性的WikiText格式文本时，一般不会使文本变得难以阅读，只有在某些情况下WikiText文本才会变得难以阅读。但是这不妨碍显示文本的效果，显示文本是依然是井井有条的。让我们一起养成一些能够让书写文本看起来舒服简约的写作规则吧。

一般性的标题、列表、文字格式、在视觉上不太会对书写文本的阅读产生较大的影响，当你熟悉之后反而会有一种显示文本的既视感。

不过一些特殊的格式、特殊的需要，需要我们规范一下它的写法，以便于在修改书写文本时不至于找不到北。目前并没有特殊的要求，自己写的舒服好看就可以了。目前也没有特别好的补充，如果你有欢迎分享你的想法。

总而言之，合理规范自己的WikiText书写文本的书写风格是非常有必要的，它能让你在过后几个月完全忘记后依然可以看懂自己当时写的什么。

*suan：我做笔记是不喜欢有除了中英文标点符号的字符的，那些字符是代码是干扰项，我更适应用空行，折叠，边框，图片之类的来隔开大段文字，跳转链接也尽量少用，阅读时容易打断思路。*

#### （8）WikiText的解析器模式

为了显示条目（通常是条目内容 ，即text 字段），Wiki文本解析器有三种模式，对条目内容（ text 字段）中的wiki文本符号，应用维基文本规则读取和解释内容。

比如【**加粗**、下划线】（文本格式）就是内联模式，书写的时候对应使用内联模式的书写规则。【标题】属于区块模式，书写时应该对应使用区块模式的书写规则。只有这样书写的文本才能被正确的显示，你可以理解为**书写规则**。

* **编译指示模式** - 解析器将仅识别编译指示模式维基文本符号
* **区块模式** - 解析器将仅识别区块模式维基文本符号
* **内联模式** - 解析器将仅识别内联模式维基文本

此条目仅做简要介绍，更详细的描述请查看[TiddlyWiki](file:///D:\\Dropbox\\10-TODO\\TiddlyWiki\\%E5%A2%A8%E5%B1%89TiddlyWiki%E7%AE%80%E6%98%93%E6%8C%87%E5%8D%97.html" \l "TiddlyWiki)舞中文文档：[https://bramchen.github.io/tw5-docs/zh-Hans/#WikiText%20Parser%20Modes](https://bramchen.github.io/tw5-docs/zh-Hans/" \l "WikiText%20Parser%20Modes" \t "_blank)

区块解析器模式

1. 区块模式维基文本的共同特征：至少需要一整行来分隔维基文本。结束符号必须位于行的末尾（在某些情况下，行尾即是结束符号。)
2. 属于区块解析器模式的维基文本符号：标题、段落、表格、引言、硬式换行、水平分隔线、代码区块、定义、清单、样式及 CSS 类别、类型区块。
3. 维基文本符号例子说明：标题 ，文本符号整行以 ! 开头的

内联解析器模式

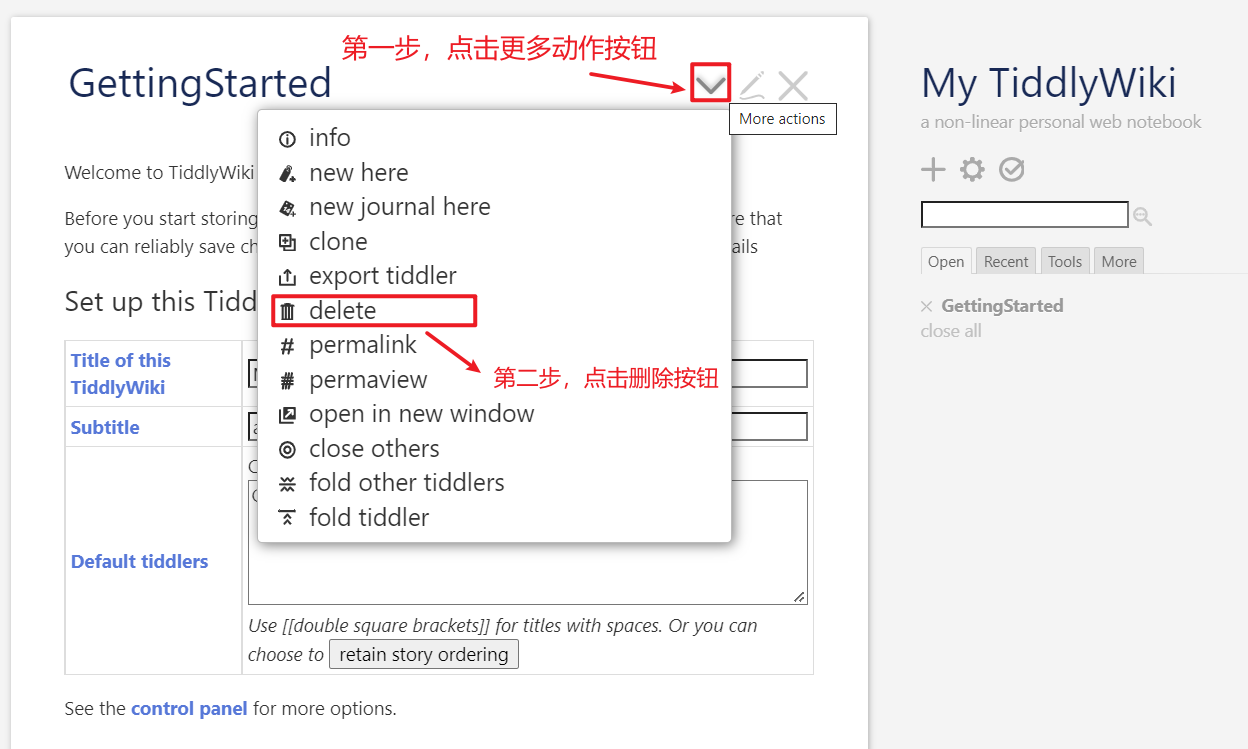
1. 这些维基文本类型，可以在没有整行文本的情况下表达。它们不需要全部在一行上，只是它们在一行中表示。因此，单一行中可以出现多个。 换句话说，当解析器尝试查找特定维基文本的开始和结束位置时，不涉及行尾，当解析器处于内联模式时，它将识别这些维基文本类型的符号：
2. 属于内联解析器模式的维基文本符号：文本格式、破折号、图片、链接、嵌入、宏调用、小工具、样式及 CSS 类别、HTML、变量

维基文本符号例子说明：文本格式中的加粗，文本符号 '' 两个英文单引号' ''

编译指示模式（不常用）

1. 编译指示是 *[WikiText](file:///D:\\Dropbox\\10-TODO\\TiddlyWiki\\%E5%A2%A8%E5%B1%89TiddlyWiki%E7%AE%80%E6%98%93%E6%8C%87%E5%8D%97.html" \l "WikiText)* 的一个特殊的组件，提供操控剩余的文本解析的方式。通常用于宏的定义。
2. 编译指示位于行首为 \，他们只能出现于内文的起始处，编译指示之间允许空白的行。如果编译指示出现在内文的主体，会被当作是普通的文本一样处理。
3. 常见的编译指示文本例如：\define（定义一个宏）、\whitespace trim（裁减文本的起始和结尾的空格）。

#### （9）删除单个条目

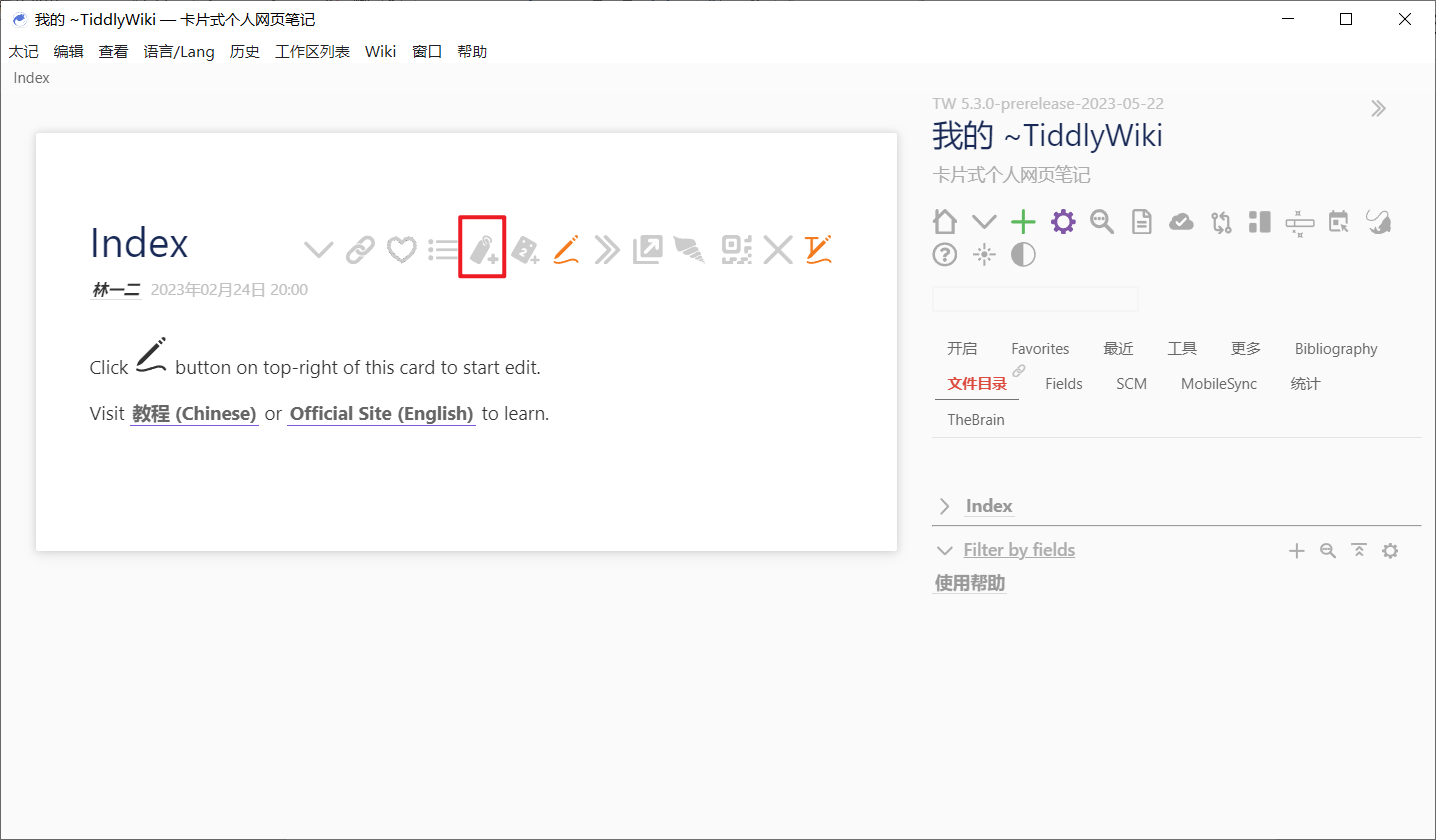


扩展的知识：更多按钮下，我们常用的功能分别是：信息、新建子条目、导出、相对链接等功能。

### （二）“文件目录”侧边栏

TiddlyWiki中文件目录的层级结构是使用的Locator插件实现的。在太微中条目的隶属关系是通过【同名条目的标签】实现的。一般来说子条目通常有父条目同名的标签表明隶属关系。比如：子条目名为文件夹，父条目名为目录。只要文件夹打上目录的标签，那么文件夹就隶属于目录，这样层级结构就建立好了。

TidGi的文件目录层级结构如图所示——显示在侧边栏的文件目录选项卡。



* 如何使用如此简单的文件目录层级结构呢？

在太记中默认创建了Index作为根文件夹条目。空白版太微，仅需要新建一个条目作为根文件夹，条目的内容作为根文件夹的描述，然后为该条目添加一个名为目录的标签，现在你已经拥有了第一个文件夹。

* 那怎么在文件夹中创建文件呢？

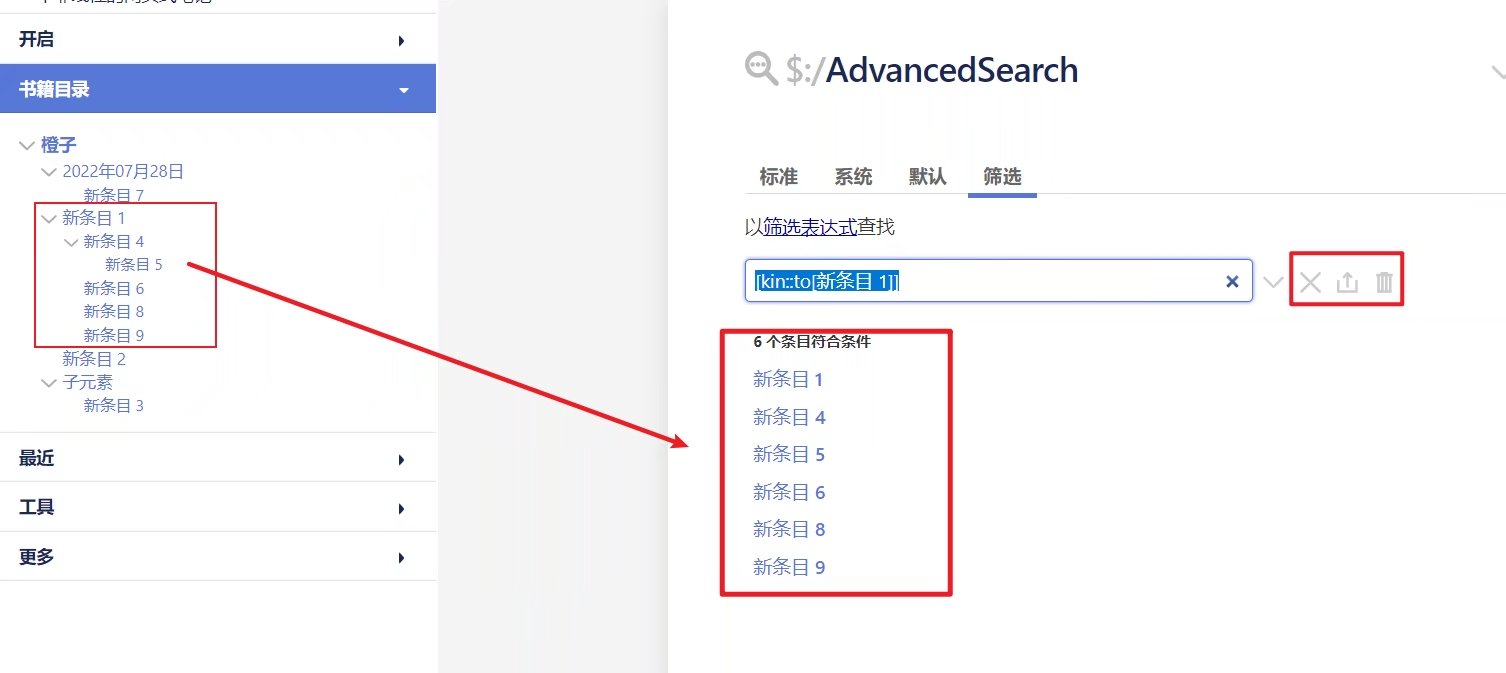
在太记中，点击上图标识的添加子条目按钮，或者在空白Wiki中点击编辑条目旁边的更多动作按钮，选择添加子条目就可以在此文件夹中创建文件。

* 删除或导出文件夹下所有文件

TidGi预安装了bimlas/kin-filterr插件，你可以在 “高级搜索（快捷键：Ctrl+shift+A）- 筛选”选项卡中输入筛选表达式，然后点击旁边的按钮执行批量删除，如下图所示。

表达式：[kin::to[条目文件夹]]

表达式说明：这将会连同【条目文件夹】以及条目名称下的所有子条目全部筛选出来。



### （三）筛选器与双链功能

#### （1）嵌入和链接（双链，重要）

<https://keatonlao.github.io/tiddlywiki-xp/#WikiText>

**1.1嵌入（Transclusion）：{{两个花括号}}**

您可以使用嵌入符号将一个条目的内容合并到另一个条目的内容中：

{{MyTiddler}} 嵌入一个单独的条目，结果是 MyTiddler 条目的 text 字段（文本字段，即主要内容字段）的内容，被呈现在当前条目内。

**1.1.1嵌入文本引用**

您也可以用一个[TextReference](https://keatonlao.github.io/tiddlywiki-xp/#TextReference)（文本引用）代替条目标题：

1. {{MyTiddler!!field}} 嵌入指定条目的指定字段
2. {{!!field}} 嵌入当前条目的指定字段
3. {{MyTiddler##index}} 嵌入指定[数据条目](https://keatonlao.github.io/tiddlywiki-xp/" \l "DataTiddlers" \t "dlt)的指定索引属性
4. {{##index}} 嵌入当前[数据条目](https://keatonlao.github.io/tiddlywiki-xp/" \l "DataTiddlers" \t "dlt)的指定索引属性

**1.1.2通过筛选后的嵌入**

可以使用类似的语法来嵌入与指定过滤器匹配的条目列表

{{{ [tag[mechanism]] }}}

**1.2 链接（Linking）：[[两个方括号]]**

[WikiText](https://keatonlao.github.io/tiddlywiki-xp/#WikiText)的关键功能是能够链接到其他条目或外部网站。

手动链接，按标题链接给一个条目：

[[Tiddler Title]]

链接到条目并指定链接显示的文本：

[[显示的链接标题|Tiddler Title]]

您还可以从编辑器工具栏创建链接。 点击**链接**，然后搜索并选择一个条目。

**1.2.1驼峰式（CamelCase）链接**

对于符合[CamelCase](https://keatonlao.github.io/tiddlywiki-xp/#CamelCase)规则（大写字母出现在单词中部）的条目标题，只需键入标题不用带双方括号即可自动创建链接。

你可以通过在前面加 ~来阻止被自动识别为驼峰式链接。 例如：

~HelloThere 不是一个链接

~http://google.com/ 是一个链接

* HelloThere 不是一个链接
* http://google.com/ 是一个链接

**1.2.2外部链接**

要链接到外部[资源](https://en.wikipedia.org/wiki/Web_resource) ，如网站或文件，请键入其*完整* [URL](https://en.wikipedia.org/wiki/URL)，包括[URI方案](https://en.wikipedia.org/wiki/URI_scheme" \t "dlt)，如协议（例如http://、file://）或mailto：

http://tiddlywiki.com/

[[TW5|http://tiddlywiki.com/]]

[[我的邮件|mailto:me@where.net]]

为了使此语法正常工作，URL必须被识别为URL。 否则，它将被视为条目标题。 因此，如果你要想使用相对路径链接到可定位的资源，请使用扩展语法：

**相对路径：**引用的文件相对参照文件的路径位置。“../”表示上一级目录开始；“./”表示当前同级目录开始；“/”通常在Linux和Mac系统以及网络服务中常用表示根目录开始。

对于NodeJS版tiddlywiki（太记），相对根路径是wiki文件夹下。mywiki/tiddlers文件夹，在太记中可以使用“./ tiddlers”表示。

[ext[Open file|index.html]]

[ext[Open file|./index.html]]

[ext[Open file|../README.md]]

扩展语法当然也适用于完整的URL，但在这种情况下，它不是必需的：

[ext[http://tiddlywiki.com]]

[ext[TW5|http://tiddlywiki.com]]

#### （2）嵌入、链接宏筛选器以及小部件之间的异同

嵌入是**{{两个花括号}}**，链接是**[[两个方括号]]**、宏是**<<两个尖括号>>**、筛选器**[操作符[参数] step step]**、小部件是**<$xxx>**标记的名称前包含一个 $ 符号的HTML标签**</xxx>**

* 嵌入是从另一条目 "B" 引用条目 "A"，此 "A" 内容出现于 "B" 的处理过程。
* 链接是仅插入条目的超链接，或者插入其它网站的网络地址，或者本地文件的URL。

#### （3）图像的使用

TiddlyWiki的拖动资源功能可以让你将任意的资料拖放进wiki。比如条目、图片、Json格式的TiddlyWiki插件等等。

* 把图片拖进编辑模式的条目里：将图片放入Wiki，并在编辑器里添加对图片的引用。
* 把别人Wiki里的条目链接、标签链接拖入自己的Wiki：复制对方的条目到自己的Wiki里

拖完之后要确认导入。

在编辑条目的状态下，你可以通过拖动图片到太微的编辑区插入一个图片，确认插入之后，在编辑器中将会自动填入图片嵌入的表达式。例如 [img[Motovun Jack.jpg]]。



**复用已导入的图片**：你可以点击编辑栏-插入图片按钮引用已经导入到太微的图片资源。

**移动端导入或使用图片：**可以通过侧边栏 - 工具 - 导入或者添加图像按钮，添加图片、JSON等文件，还可以手动绘制图形。图片可以通过曲别针（导入）按钮导入，然后再编辑栏用插入图片按钮引用。

如果你觉得现在图形的大小不太合适，可以图片的引用表达式张红添加width参数并设定合适的大小，例如这样：[img width=128 [tx.jpg]]

较大的位图会显著降低 TiddlyWiki 的性能。例如，现代智能手机拍摄的图像经常会是 5 MB 或更多。所以尽可能的使用外部图像。

在TidGi中，因为已经预制一些程序自动化处理这些任务，比如懒加载图片、[img[file://<path>]]（file://文件协议）等等，所以放心用。

TidGi默认预装Shiraz（设拉子）插件，所以你可以使用插件提供的宏实现图文混排，就像在Word里面做的那样，不过在TiddlyWiki中我们是使用语言去描述图片的显示效果。

image-basic宏的使用语法如下：

<<image-basic img width:"" align:"" caption:"" tooltip:"" alt:"">>

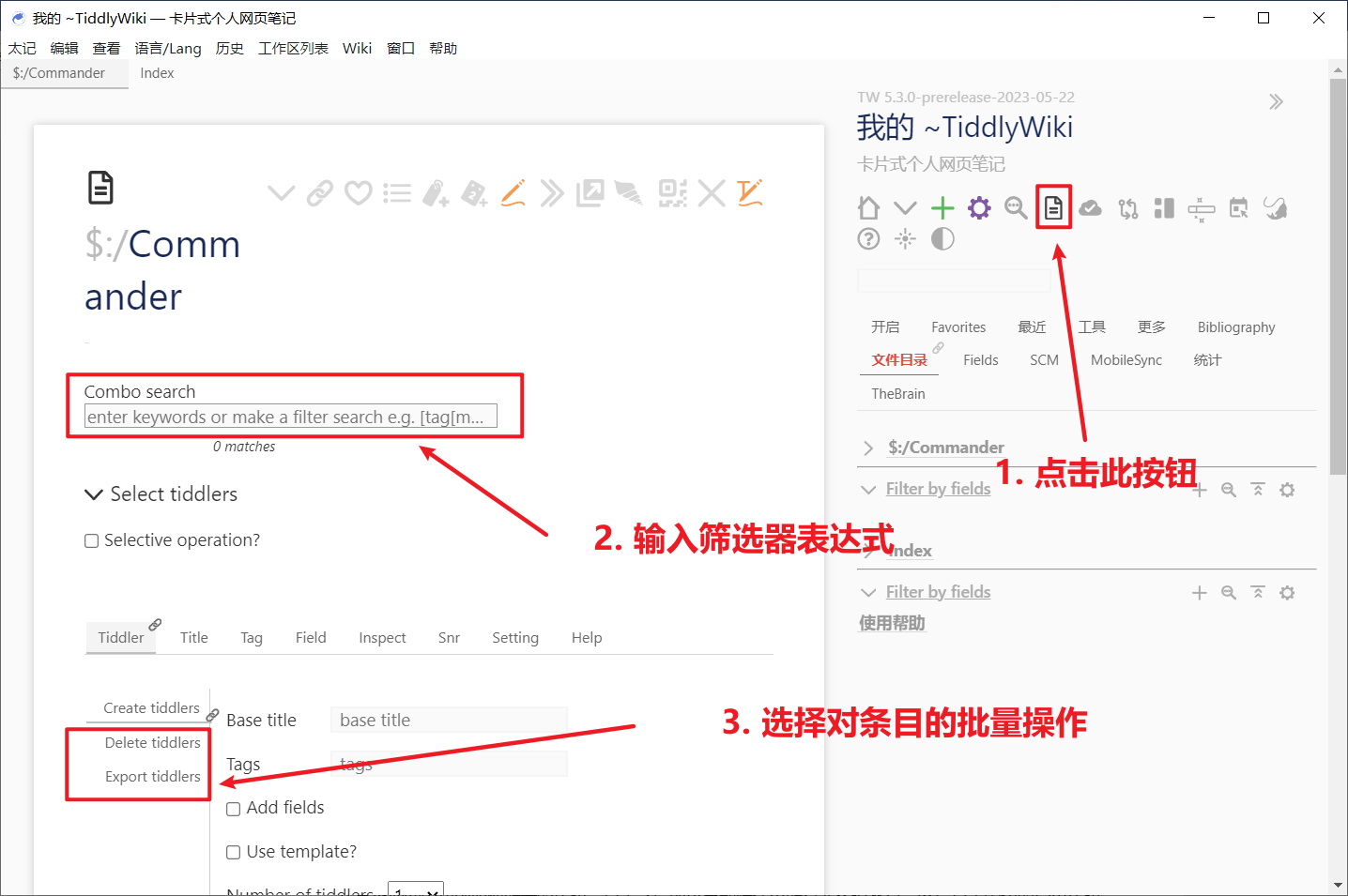
|  |  |  |
| --- | --- | --- |
| **属性** | **类型** | **描述** |
| img | 必填 | 图像的 URL 或图像提示者的标题 |
| width | 可选 | 图像宽度，高度自动设置，默认值为30% |
| align | 可选 | left、、、 和图像的对齐方式，默认值为right center none none |
| caption | 可选 | 图像标题显示在图像下方 |
| tooltip | 可选 | 在图像上显示的工具提示 |
| alt | 可选 | 与图像关联的可选文字 |

例子：此处显示了如下所示的简单图像。你可以把宏写在段落的开头或者结尾，然后就可以看到显示效果啦。

<<image-basic "秋季01.jpg" caption:"Figure 1. This is a basic image." align:"right" tooltip:" demo of image-basic macro">>这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。这是一只火狐狸。

#### （4）条目的批量操作

1. 筛选出含有关键字的所有条目：[!is[system]search[*关键字*]]
2. 按字段与字段值筛选出符号条件的条目：[search:字段名[字段值]]
3. 在太记中你可以使用kookma/commander插件对条目进行批量操作。



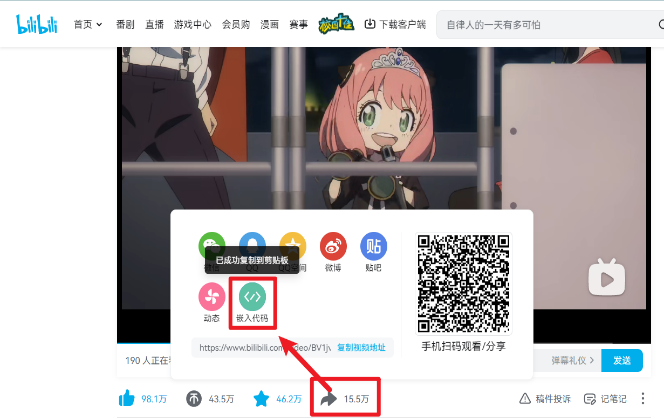
* **批量从一个太微到另一个太微转移数据**：如果你想批量的转移数据到新的太微中，也可以把旧有的太微文档直接拖进新的太微中，然后选择需要的条目导入，注意，请谨慎选项要导入的系统条目（以“ $:/ ”开头的条目。）
* **转移数据到新的TiddlyWiki。**

使用kookma/commander功能，输入筛选器表达式[!is[system]sort[title]]自动筛选出所有非系统条目，然后点击Export tiddlers选项卡-导出条目按钮，选择JSON文件格式保存到本地。导出条目文件会保存原有的结构（条目的隶属关系、双向链接以及固定的文本内容图片等不会改变，图片等在条目中使用相对路径引用的文件注意在导入到新的TiddlyWiki中要保持图片的相对路径不要改变。）

拖动导出的JSON文件到新的TiddlyWiki中，单击导入完成条目的转移工作。条目的转移工作不包含之前的用户设置，仅包含普通条目内容，即用户自己编写的全部内容。

#### （5）嵌入在线视频

你可以嵌入一些在线视频。并且可以文本在线视频混排。有时候这非常有用，不是吗(\*￣︶￣)。

我们使用小破站作为例子，如图所示，在小破站的转发功能 - 点击嵌入代码，提示已经复制到剪贴板。

正如下面的内联框架标签。你可直接新建一个条目，把复制好的直接粘贴到条目中，保存，就可以看到插入的在线视频了。

<iframe src="//player.bilibili.com/player.html?aid=554429957&bvid=BV1jv4y1P7Bb&cid=731286874&page=1" scrolling="no" border="0" frameborder="no" framespacing="0" allowfullscreen="true">

</iframe>



你可以使用iframe标签的两个属性，height和width调整插入视频的高度和宽度。单位可忽略，默认单位为pixels（像素）。

#### （6）常用的筛选器表达式

* 如何筛选出含有特定名称标签的条目？

[tag[标签名]]

说明：[all[tiddlers]!is[system]] 通常可以被简写。表示筛选出所有非系统条目。

* 如何筛选出同时拥有某些标签交集的条目？

[tag[标签A]tag[标签B]]

说明：筛选出同时具有“标签A”和“标签B”的条目。

* 如何筛选出同时拥有标签A或标签B的所有条目？

[tag[标签A]] [tag[标签B]]

说明：筛选出只要有“标签A”或“标签B”的条目都会被筛选出来。注意两个表达式之间有空格。

* 如何筛选出标签大于等于两个的条目?

[all[tiddlers]!is[system]] :filter[tags[]count[]compare:number:gteq[2]]

说明：[all[tiddlers]!is[system]] 限定条目为普通条目，compare:number:gteq[2] 限定标签条件大于2，可以自行修改。这个表达式可以有变形式，限定标签在一定的范围或小于固定值，请查阅compare筛选器操作符。

* 条目总量如何显示？数字化呈现？

[all[tiddlers]!is[system]count[]]

说明：表达式在高级搜索里检索出来的结果为**显示出来的数字**，这就是条目的总数。在控制台 – 信息 – 基础选项卡的最末尾处有显示各类条目的数量。

其它的常用筛选表达式可以在高级搜索（快捷键ctrl-shift-A）筛选选项卡的搜索框旁边的功能按钮找到。

* 时间轴

<<timeline format:"YYYY/MM/DD">>

说明：仅需要在条目中添加这段文字即可按照时间排列显示条目。可以把该条目的标题设置为时间轴。

* 删除未使用的图片

[relink:orphans[]!is[system]is[image]]

说明：需要安装Relink插件，支持 [img[]]、<$link to =“”> 等引用，可能引用图片的系统条目：$:/Import 导入时

## 三、如何使用TiddlyWiki构建笔记系统？

**工具VS内容**

本节主要说明如何运用好【条目】这个信息载体，它可以是卡片，可以是一篇文章，可以是信息节点，可以是一个自动化的信息处理工具等等。如何使用TiddlyWiki因人而异，所以本节内容只提供一个参考和灵感的来源并做一个简单的介绍。相信朋友们的创造力为MAX！

起初，我对太微内容如何组织挺迷茫的，担心放进去的内容我会找不到，担心没有结构、大纲会不知道骨架，从而无法掌握全貌。换句话说，我不会用太微写东西，仅仅是熟悉这个工具，没有对原本的【记录并组织内容】负责。【记录】是为了【读取】使用。【组织】是为了更加简便的【复用】。有时候也是为了【组合】生成新的【维基小鱼】。太微是一个非线性的笔记，也是一种条目形式的卡片笔记，我觉得应该回归笔记记录并使用的这个上面来。而其它的特性也是微为了内容而服务，以内容作为中心。如果对于整本书来说，写作的类型不一样，那组织方法应该也不一样。而对于一篇文章来说也是这样。

### （一）时间序列记录法

1. 时间序列第一种用法

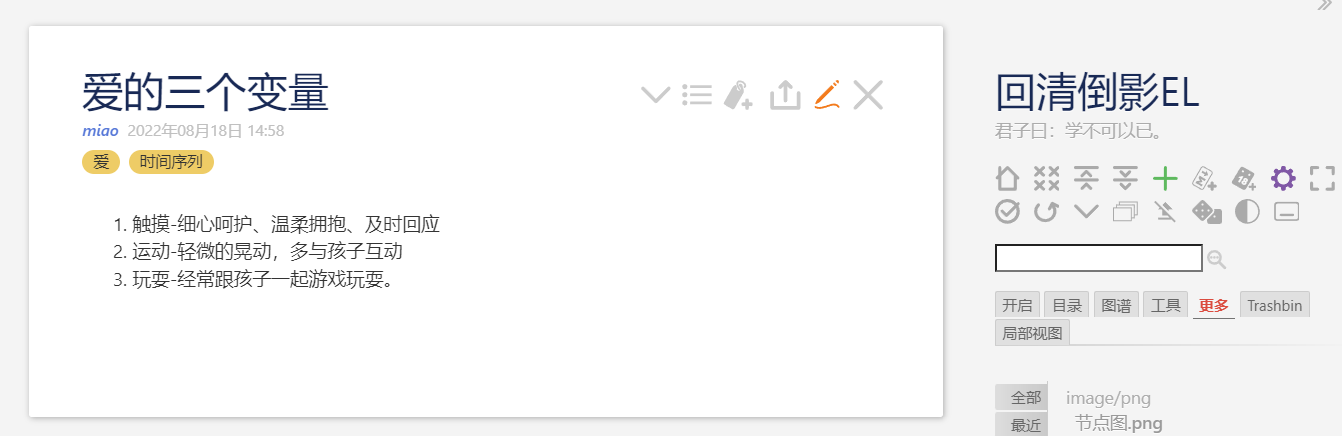
最简单是使用方式是每天记日志、账目。条目会自动按照创建时间排序。

你可以通过这种方式查看，在侧边栏点击最近，可以看到按时间修改顺序排列的小鱼儿（条目），如果是创建的日志条目，那么可以看到创建的时间日期。

1. 时间序列第二种用法

你可以用来记录读书笔记或者一些灵感顿悟之类的文字、记录想法，并打上【时间序列】的标签，这样做便于筛选出此类条目，为之后的自动化工作提供基础。

然后你可以选择性的为创建的条目起名字，写内容，内容可以是三言两语，也可以是一两百字的段落，写完之后，凭借你的感觉，对内容进行类别的评估，并把评估的结果——大致的类别，写到条目的标签上。建议不超过两个。最好是一个类别。



在不断流动的时间长河中，随着你不断的记录、记录、记录，条目数量会不断增加，从开始的寥寥几条，到最终的几百条。你会发现，他们有些部分是非常相似的，这种相似的部分可以大致分成几个群组。而这个群组我们可以称之为主题或者元素。

元素或主题是你有意识的主动的按照一定的维度和界限划分的，元素的名字是这一群条目的中心，它有非常强大的磁力，与该元素含义越贴近则磁力越强，反之着越来越弱，元素名字总是吸引和它相关的条目把它们聚集起来就像磁铁一样。

也可以是在不断的累积想法条目的过程中发现一些规律随后将它们分组成为元素或主题。

需要解决的问题是，如何聚类成为元素。这个将此过程简化、减压。我们将在元素渐构笔记法中讲到这部分内容。

**一个有趣的以标签和标题为线索的整理方式：**

使用标签作为线索，线索分为两段，第一段是层级隶属关系，第二段是线索隶属关系。

条目作为文件夹（分类），200-201书籍-20101待读书籍。（这一段是文件夹层级隶属关系），20101中包含书籍列表。书籍列表使用标签标记从属关系，不使用文件夹的从属关系。

### （二）主题渐构笔记法

元素渐构笔记法就是关于马的东西放到以马为名称的小卡片中、归类整理即聚类与分类。这个过程就是聚沙成山的过程，滴水石串的累积。这也是运用自己学到的零散或系统知识打造成一柄柄趁手的武器的过程。也是不断扩展自己认知边界发现以前从未发现的探索之旅，对于我们来说，在认知之外事物，由于超出感知与想象的范围，所以基本等价于不存在。元素渐构笔记法，无非就是在不断的累积之中扩展自己的认知圆圈的半径。就像哲学家芝诺说，他曾画了一个圆圈，圆圈内是已掌握的知识，圆圈外是浩瀚无边的未知世界。知识越多，圆圈越大，圆周自然也越长，这样它的边沿与外界空白的接触面也越大，因此未知部分当然显得就更多了。**元素渐构笔记法就是描述这种从点到圆的渐进过程。**

其实读者可以通过这篇指南感受到，这里的每一个知识点都是慢慢的汇聚形成的，你可以尝试不看这篇指南感受一下同时对比看这篇指南感受一下这两者之间的差别。不光思想可以汇聚，物理生物和化学中也可以用这种角度去观察，你可以发现一些有意思的现象，虽然原理可能不是这样，比如化学中的多元素的不断组合然后经过物理的空间折叠形成蛋白质，然后发挥它的功能，但是组成蛋白质的元素是不能够发挥蛋白质的作用的，人们也无法用它做蛋白质做的事情。

元素渐进笔记法是来源于程序设计中的面向对象思维的启发，修改形成的，我一直需要一个方法可以帮助我描述自己感知到却无法言说的事物，还有感知不到却依然存在的事物，比如你从来没有学过化学也不知道水的分子式，如果我不给你说水的分子式，你就永远也不知道【原子】的存在，原来还有更小的物质组成更大的物质，这一认知是人们一生也无法认识到的，尽管我们每一刻都在使用水。如果你知道了【原子】的存在，那意味着可以做更多的事情，这些事情需要你自己去思考，去感受，语言的描述和表达也将会更加清晰。

元素（继承、组合、多态、基本）（聚类，分类），图式、索引分类，索引条目，条目与条目之间有逻辑关系存在。这些词是元素渐进笔记法中的概念性描述词和一些想法的概括，可以作为了解。

元素几何：非线性的笔记需要一些线索和视图，细胞。元素几何的名字由来是因为节点与节点之间的连接是一个几何的形状。我一般把它作为一种全局视角。

元素几何的实现方式就是用太微的链接、嵌入、字段（标题、标签、内容、空字段）以及设置中心节点、层级的元素节点图的局部和全局视角来实现。

条目的继承关系：条目的隶属关系是通过同名条目的标签实现的。一般来说子条目通常有父条目同名的标签表明隶属关系。这种隶属关系的创建方式有两种：

1. 可以通过父条目的创建子条目来实现。
2. 在父条目内容中链接子类，通过链接创建的子类，手动赋予父类标签。
3. 用典可以使用：链接。链接不存在隶属关系。

子条目也可以通过一个插件自动化的显示在条目底部声明。inverse-link-and-folder。

可以使用标签作为线索，使用时间序列这个标签标出需要整理的想法。使用具体的元素标签标出将要合并到某条元素笔记。

条目-标签这种隶属关系不仅仅是层级之间的隶属关系，而是一个网状的隶属关系，层级的隶属关系只是网状隶属关系的一部分。

收集想法可以使用【时间序列】，然后用元素聚集高度相关信息。完成关系的连接和整理。

想法是自己对输入潜意识处理的结果，假如有二十条想法（或者说批注），没有时间只能对想法做一个简单的归类。如果想法很多，主题也很多，那么毫无疑问，需要慢慢花时间去消化，这其实就是增量的过程。需要不断查资料来不从，确定概念术语的表达。你需要对想法做很多道工序的处理，这就像做菜，对食材的处理一样。最基本的是验证逻辑性质，有无基本的逻辑错误。无论主题是否存在已知笔记体系中，笔记整理到最后都会慢慢呈现出一个主题来。

一开始整理笔记，或许不知道如何整理了。这些是因为你缺少了相应的内容补充，也就是少了程序或者插件。举个例子，比如涉及到近现代的西方美学转向问题，那么就需要一定基础对西方绘画艺术有所了解。具体例子比如绘画和摄影。照相机刚发明出来的时候没有人去思考两者内部的关系，只是觉得照相机可以替代绘画了，以后的人不用学画画了。但画画还在发展，20世纪典型代表人物就是毕加索。颠覆了很多艺术之前的准则。那现在就出现了问题，为什么呢？画画和摄影之间的关系是什么呢？摹仿论最早可以追溯到亚里士多德，在绘画领域长时间占据着重要影响，正因为摹仿论，当时才会认为绘画已经被照相机给终结了。但现代主义的画家们纷纷要求改变这一思想，表现更为复杂的现实和人的心理活动。

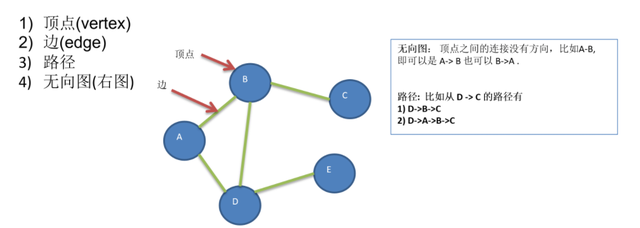
主题就是一个以你自己主观想要研究的比较感兴趣的关键词，比如研究生命科学的，可以是很大的，人体啊。也可以是研究某一类型的昆虫的生物价值。

想法的产生的大概原因是来源于输入 + 问题 + 反思（思考）。输入就是你读过的、见过的、感受过的等等的任意的事物或信息。这些信息会在我们潜意识进行处理，与已有的知识和经验进行关联，当一个问题暂时无法处理时，一直没有头绪，不妨放一放，等一等。可以画画思维导图，或者在笔记上写出关键词，然后看看关键词之间的联系。等差不多的时候，就去睡觉，不用管了。等到第二天或者第三天。或许你就突然的明白是怎么回事了。

元素亦或叫主题，通常由以上四种格式构成，元素本身，元素本身在不同环境下的多样性，元素与元素之间的组合，以及元素的实际例子。这些只是我的初步探索和使用，尚有不完备的地方，需要时间的修饰和打磨。不过自认为或许是一个不错的想法，至少知识在成长的成长过程，从点不断增大半径已形成大圆的过程是我们每一个人都要经历的。

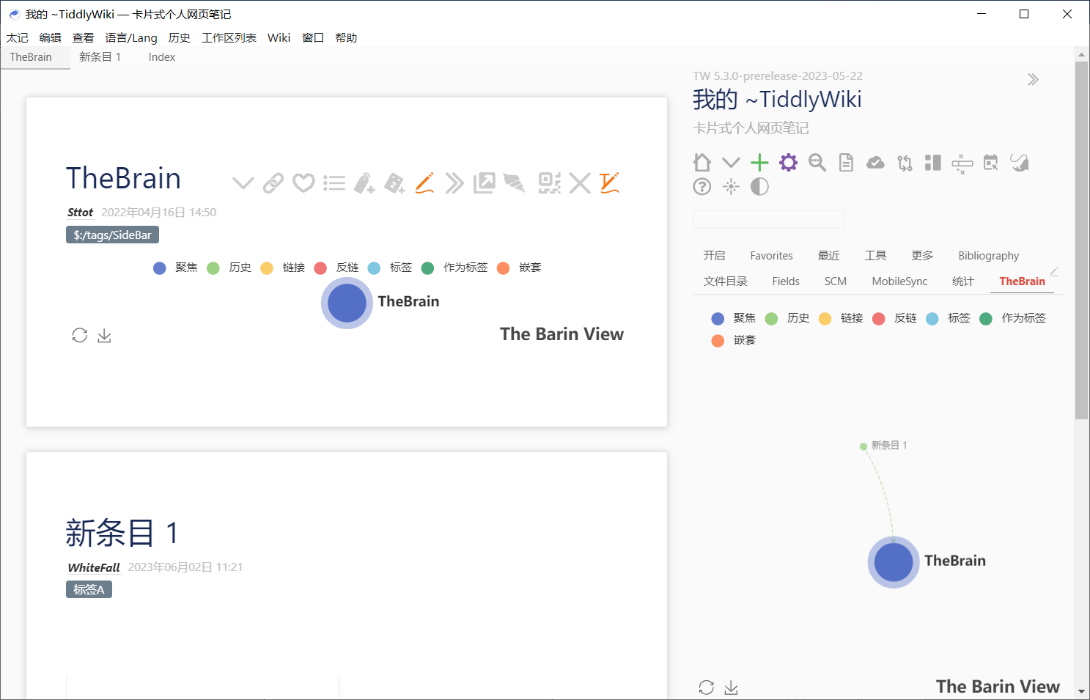
### （三）关系图谱或图数据结构

图是一种数据结构，节点可以称为顶点。其中，一个节点可以有零个或多个相邻节点，节点之间的连线称为边。



太微中的图结构

太微中，实现可视化的插件有两种可使用，第一个是Echart，第二种是TiddlyMap。其中Echart可以不仅实现条目关系的可视化，还可以实现各种excel可以实现的图表。TiddlyMap可以实现思维导图或关系图谱的信息组织形式。

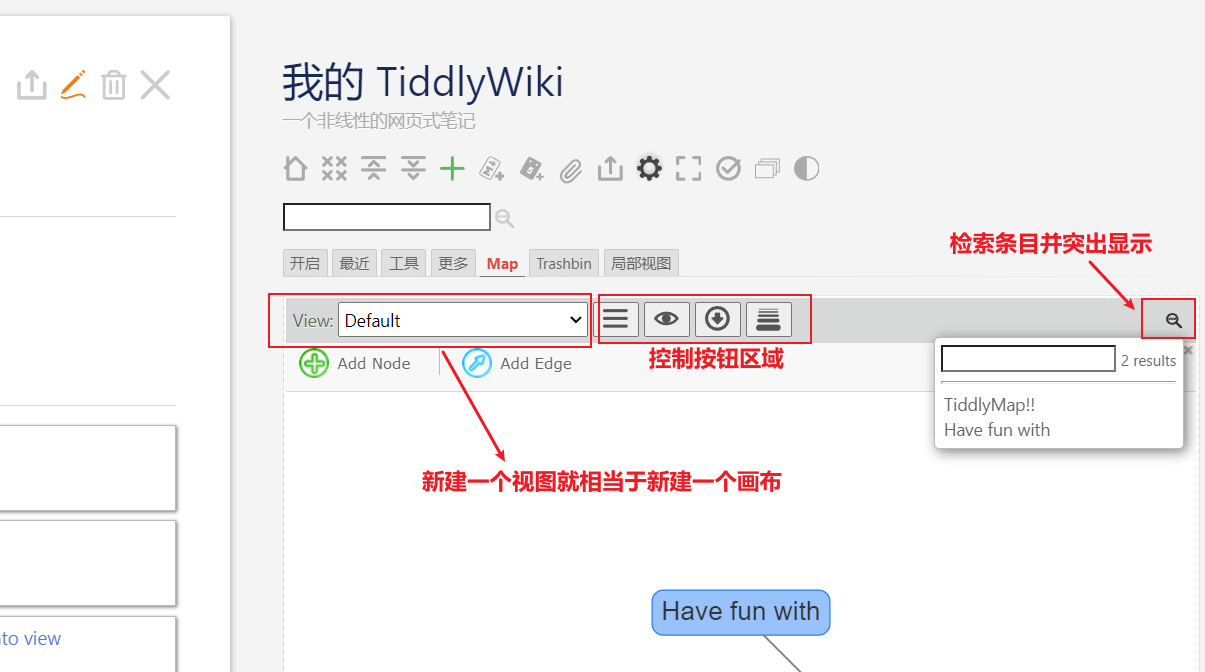
与当前条目相关的节点图谱——TheBrian（可视化）

TiddlyMap - 思维导图-关系图谱

（不适配移动端视图，视图被判断太小不能用，需要附加组件Vue.js插件，不装一定会出错。）

从大到小是由 全局设置 - 视图（许多画布） - 图谱（实时节点图） ；一个视图就是一个画布。





控制按钮区域的按钮从左往右依次是：打开菜单、更改显示层级、打开Map导出按钮、改变Map中的方格背景，其中在编辑Map中最常用的的打开菜单按钮和更改显示层级按钮。

创建新画布，单击打开菜单按钮 - 创建一个新画布，输入新视图名称，点击完成创建画布

在画布中【双击】创建新的节点。创建节点间关联，点击Add Edge按钮，选中开始始节点并拖放到目标节点完成关联（其中会提示填写边类型，你可以自定义）。

创建的节点会在太微中以条目的形式存储，同样新建条目也是在Map中新建节点，节点之间的联系同样可以使用条目顶部的按钮实现关联，所以TiddlyMap即是关系图谱又是思维导图。它的边和节点的样式都是可以修改的。

控制按钮区域的打开菜单按钮 - 全局配置 或者 Map中 关于边和节点的解释。

Edge-Type：边的类型（边的样式），就是决定边上显示的文字，完全自定义，比如：被引用或者引用。

Node-Type：节点的类型（节点样式），规定节点显示的样式的。就是节点的样式表。

View configuration视图配置，可以配置节点过滤器，用于过滤显示节点。

更改显示层级按钮的的解释

Neighbourhood traversal：邻域遍历

Neighbourhood scope：邻域范围

Both、Outgoing、Incoming、Inter-neighbour edges

### （四）线性写作的形式

写作的类型不一样，组织方法应该也不一样。比如说倒叙，插叙，正序等等。

条目即文章，一个条目就是一篇文章的组织方法。

*按照作品呈现的先后顺序，一般是时间顺序或者逻辑顺序写作。*

一个条目可以当做一个卡片。每个卡片之间可以相互嵌入对方的内容，也可以使用链接引用。形成相互关联的卡片关系网，你可以使用Echart的TheBrian视图观察条目节点之间的联系（全局视图与局部视图），也可以通过目录宏生成像目录一样的关系图。也可以对一个条目中的全部标题生成目录，你可以使用标题插件Gk0Wk/page-toc对一个条目卡片生成标题或者使用Section Editor（部分编辑）根据标题编辑内容。这两个插件在TOC自动化插件里已经介绍过了。

自上而下，先写后拆，拆分后嵌入。

多个条目的组织方法。这是与【条目即文章】相对而言思路相反的方式。你可以在某个条目中建立大致的目录结构，将这个条目等价为论文中的目录页。然后使用链接创建占位符，比如“[[前言]]”，当年点击链接时会创建此条目，然后你就可以在这个条目中写具体的内容了。当然你可以使用嵌入（transclude），嵌入已经存在的条目，比如某一条已经写好的名人名言，正好可以拿来引用一番（要注意的是，在美没有导出之前，请不要删除被嵌入的条目，删除之后就无法显示了）。

在建立条目小鱼的时候，如果你遇到不懂的字词短语（整体的概念关键词），你可以以用在字词短语上套用链接创建小小鱼（子条目）不断逼问它是什么。这样你就会不断深入理解并且收获新的知识。

### （五）渐进写作（增量写作）

简单的说，渐进式阅读的过程是逐步解构/分解，而渐进式写作则是逐层建构/重铸。渐进式阅读和渐进式写作两者相辅相成，结合起来则形成了毁灭和创造的循环。

其实我现在用的可能就是，一种想法的产生，把一个大概的结构写一下，然后写一下想法。然后补充一下，还不完整。然后新的信息输入，想法产生，结构调整。然后以旧的文章不断的迭代。有时候有了新的建议就标记下来，等待时间的打磨。实践的调整。

渐进写作相关链接：

渐进写作：[https://blog.effie.co/%e6%b8%90%e8%bf%9b%e5%bc%8f%e5%86%99%e4%bd%9c/](https://blog.effie.co/%e6%b8%90%e8%bf%9b%e5%bc%8f%e5%86%99%e4%bd%9c/" \t "dlt)

非线性写作的4条建议：<https://zhuanlan.zhihu.com/p/399187853>

### （六）自动化无压记录

双链或间隔重复无压记录，作者：林一二

**01.关于 RR 的双链无压记录**

看过 【双向链接时代的快速无压记录：https://www.yuque.com/deerain/gannbs/ffqk2e】 的同学可能都知道，roam research 之所以能带火 [[双链|反向链接]] ，是因为它宣传：

输入的时候可以彻底的不整理，就完全堆在日志条目里 然后用 `[[]]` 打个标记，然后用双链的一些特性，试图让后续需要整理的时候压力也不至于很大。

有些东西可能很长一段时间都用不到，那你不去整理它，后续整理的成本也就不用支付了。开始就是随意堆放的，输入成本也很低，就很赚。

**02.太微的自动化无压记录**

roam 的选择是把未整理的内容堆放到日志条目里，这在太微里做起来也不难例如我们的添加新想法按钮，点一下之后创建新条目，就能往里输入：

<<reuse-tiddler "快速创建新笔记按钮">>

例子：

1. 点击快速创建想法按钮，自动创建出一个带「想法」标签的新卡片
2. 输入标题「变量：显示或隐藏空节点」，加上标签「logseq/配置」，输入内容 `:ui/show-empty-bullets?` （也可以在里面加入 `[[logseq/配置]]` 来产生双链，看你是标签党还是双链党，反正都可以实现同样的效果）

两次操作即可保留内容，与之相比较的复杂折腾的情况，详见【双向链接时代的快速无压记录：https://www.yuque.com/deerain/gannbs/ffqk2e】。

**03.太微的间隔重复放心记录**

但我的判断是，这种情况下一两下点击的差别其实不大（一下点击打开日志，或者两下点击创建新想法及其标题），重点是心理上放心这个内容不会丢。这种放心感才是 rr 双链系统的核心，只要有这种放心感，交互上区别其实可以忽略。

而之所以有信心就算随便堆放的信息，自己也能再找到，是因为【钓鱼插件】，能够定期让我们回顾收件箱中的内容。（详见 [为临时性的和不完整的笔记设置写作收件箱 - Thoughts Memo 汉化组译制](https://zhuanlan.zhihu.com/p/442485060)），这是太微远超 roam research 和 Obsidian 的地方：

钓鱼插件可以配置为自动提醒我们回顾加了「想法」标签的所有卡片（详见[[提供自动化聚合整理的能力，让你从更抽象的角度上整理笔记，而不用亲手整理]]），这也就实现了科学的低成本输入+内容不会丢的放心感。

过滤器，宏、小部件。

### （七）随手记录，动态生成

这个是太微独有的。因为有很多小部件和宏啊。都可以直接在WikiText中的书写文本使用。（例如：教程主页就是动态生成的，你可以随手打开条目编辑看看。都是现成的案例可以学习。）

### （八）卡片笔记写作法

知乎作者：亦明Reading

**卡片笔记写作法，实际上就是碎片系统化。**一个卡片就相当于一个条目。条目卡片小鱼。小鱼之间的相互引用。

它能起作用，靠的是围绕主题进行持续性自然积累。 所谓自然积累，即是想到了遇到了就随手记录一下，不需要专门花大块时间。这样也就最大程度的降低了参与门槛。 持续这个动作必然会带来数量上的累积与查找上的麻烦。前者是优势，后者是问题。解决这个问题，靠的是“检索”。

### （九）PARA个人知识管理

它提供了一种：“将信息内化成知识，并以项目为导向，将知识整合输出为实际内容，获得反馈，从而实现闭环”的方法论。

再摊牌点说，PARA的思想是，以完成一个又一个的项目为目标去记笔记 。

P.A.R.A. 代表**项目（Projects）— 领域(Areas) — 资源(Resources) — 档案(Archives)，**四个顶级类别， 包括您在工作和生活中可能遇到的每种类型的信息。

### （十）书籍与书籍的分卷

*对于太微的性能问题，TidGi等基于NodeJS的TiddlyWiki没有问题，而且功能更丰富。比如你无需为图片、音频或者PDF文件的位置发愁。直接丢进去就可以了。*

*单文件版TiddlyWiki它是有一个比较大的极限的10万或者2万条目，实测到两万空白条目时进行写入的操作会有4秒的等待时间，其它没有问题，所以我的建议是，像书籍一样分卷处理，大概用着有点卡顿的感觉，就可以考虑进行分卷的处理啦，就像史记一样可以分十二册。也可以根据主旨和类型分成不同的书。*

*世界上没有十全十美的工具，就像一本书绝世好书不可能讲完世界上的所有事情一样。我们要追求的是卓越而不是完美。*

*不过随着计算机的发展，这个性能问题也可能不是问题，谁知道以后的计算机是什么性能呢*

*tw文档中写了tw有性能问题。不建议太大。按你一个笔记本的量来容纳就行了。顶多是一本大百科全书。 保存成单文件html或者生成多文件静态网页*

*对于TWPUB书籍来说，基本上一本书一个太微单文档程序就可。*

关于书籍的分卷的两个插件相关插件。

1. Searchwikis Plugin（搜索维基插件）：通过索引搜索多个维基

主页：<https://kookma.github.io/TW-Searchwikis/>

安装Searchwikis 插件的TiddlyWiki可以作为一个检索工具，只有在需要检索的时候才使用它。

1. 在Nodejs中可以使用TiddlyServer，点击链接查看[TiddlyServer](https://bramchen.github.io/tw5-docs/zh-Hans/" \l "TiddlyServer%20by%20Arlen%20Beiler" \t "dlt)的具体介绍。

主页：<https://arlen22.github.io/tiddlyserver/>

### （十一）分享和发布功能

TiddlyWiki最大的一个特性是一个HTML文件，笔记数据会首先保存到HTML文件中。这意味着发布一个静态的网站就像打印一篇公告一样简单。只需要把它放到公告栏的位置。比如GitHub的gh-pages（GitHub 页面）。然而你也可以把它发给别人直接打开。

TiddlyHost，一个在线的站点可以允许用户编辑保存，仅需要注册账号即可使用，你可以把它当做发布页。

分享wiki的一条笔记可以选择导出为HTML文件，如果对方有太微，可以选择JSON格式文件，方便对方导入。如果是多条笔记，可以使用筛选器，筛选出符号条件的条目导出。

### （十二）数据安全性与备份

TiddlyWiki的数据安全性措施有草稿功能自动保存功能，简单的说，只要你确认保存了就不用担心数据会突然丢失，改动会写入文件内，就算是插件导致的功能性损坏不可用，也可以找回来，除非这个文件被删除或物理损坏。要保证的是，改动一定存到文件系统上了才行。

不过也有备份的措施比如：TidGi（太记）的历史版本工具、TiddlyDesktop的备份管理，然后是桌面端的KopiaUI自动定时备份。

#### TidGi的历史版本工具

Git的两种可视化工具：

* Source Tree ：**[https://www.sourcetreeapp.com/](https://www.sourcetreeapp.com/" \t "_blank)**
* Github Desktop ：**[https://desktop.github.com/](https://desktop.github.com/" \t "_blank)**

林一二：太记应该不容易丢，可以打开git工具看看是不是真存到硬盘上了，看到类似这种就稳了。

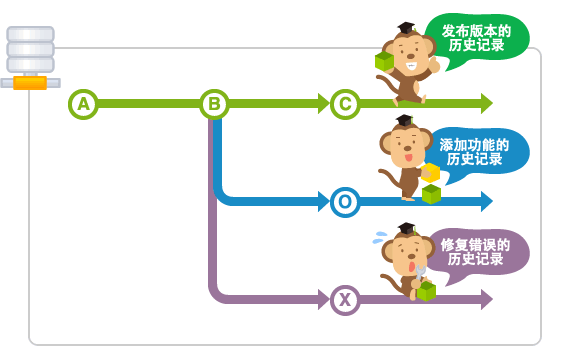
|  |  |
| --- | --- |
| descript | descript |

WhiteFall：但是一般性的太微呢，比如单文档的那种，或者使用webdav的呢？

林一二：那种因为都是在 html 内写，不涉及文件系统，所以改动肯定会存到 html 内，不用担心。那 html 有没有存到文件系统上就得自己检查了。

#### Git分支是什么？

分支是为了将修改记录的整体流程分叉保存。分叉后的分支不受其他分支的影响，所以在同一个数据库里可以同时进行多个修改。



#### Git 提交更改（Commit Changes）

当文件被更改后，一般会显示在这里。然后你可以使用Add（添加）将变更的文件添加到暂存区，在使用Commit（提交）将更改（Changes）放入版本库(Repository)的当前分支上。在GithubDesktop中只需要在summary中输入总结并点击 Commit to master按钮即可完成这两步——将更改的文件放入版本库的当前分支上。之后就可以在History（历史记录）中查看刚刚提交的修改。

#### Git 历史记录（History）

历史记录（History）中保存着每一次的Commit（提交），通过这些提交记录，你可以追踪每一次的修改，看看每次修改都做了什么。你可以通过历史记录对这些提交进行还原（revert）、回退（reset）、合并、彻底全部清除等等操作。在GithubDesktop中，撤销（Undo）提交即仅仅在本地提交了但没有推送到Github，这时可以在History中右单击刚提交的Commit，选Undo commit即可撤销。

## 四、为知识库拓展新的功能与特性

太微独一无二的优势就是生来即是网页，可以无需额外软件发布给任何人，寿命与前端三件套齐长。自由度非常高，只要你想，连标记文本都是可以由用户定义的。因此也可以把TiddlyWiki看做对知识进行管理的操作系统，TiddlyWiki插件作为系统的应用，而底层数据就是条目。

要十分注意的一点就是，在安装插件的时候要注意备份一份你的太微文档，这是一个好习惯。如果安装后没有任何问题，可以把备份删掉。其实通常是某些极个别的插件会出现冲突问题或者相互不兼容问题，大部分的插件都没问题，不会引起插件冲突导致的程序错误。

本节将为你介绍：中文插件库的安装使用以及插件安装的注意事项、常用的插件的使用说明。太微控制面板的使用介绍。你可以先大概浏览一下，并不需要即刻安装它们，当你需要的时候再来查看安装也不迟。

**如无必要，勿增实体。功能越多，问题越多。简单实用稳定就是最好的。**

### （一）Markdown插件

在官方库搜索Markdown插件，点击安装。然后再CPL搜索Markdown-Export，点击安装。注意 ，这两个插件是都要装的，不然导入的markdown文件无法正常显示。

可以把新建Markdown的按钮放在侧边区域的页面控制栏，设置可以在控制台 - 视觉外观 - 工具栏 - 页面工具栏启用【添加Markdown条目】

### （二）反向链接插件

在CPL搜索 inverse-link-and-folder （反向链接插件），安装后条目底部会出现【此文件夹中的文件】以及【子条目的标签】



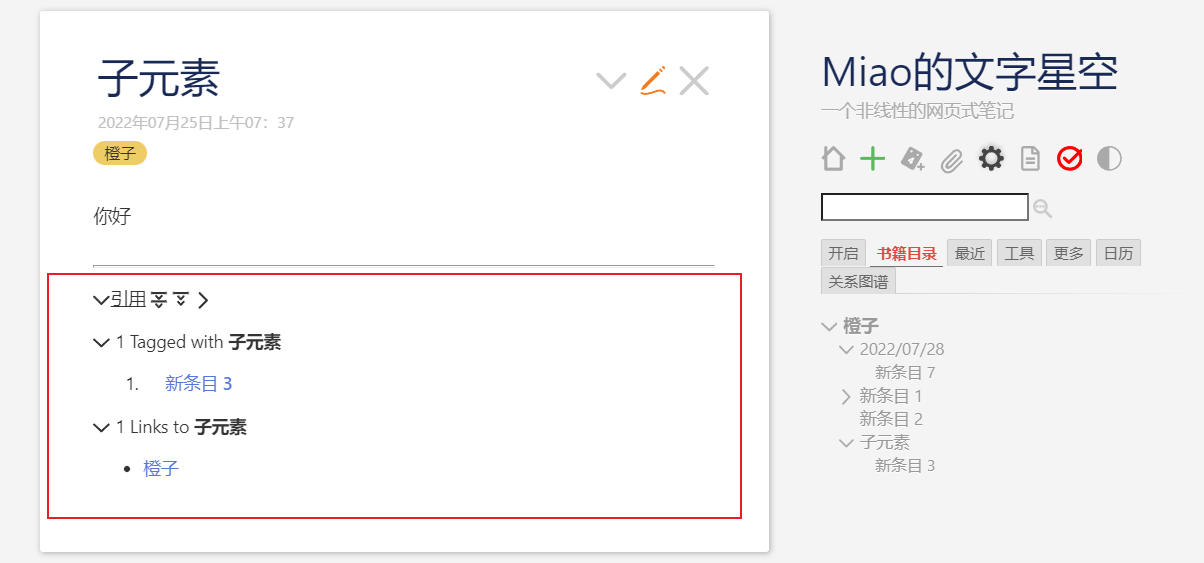
### （三）TOC自动化目录插件

在CPL搜索 Gk0Wk/page-toc ，点击安装，在控制面板 - 视觉外观 - 工具栏 - 查看工具栏 勾选Page TOC。





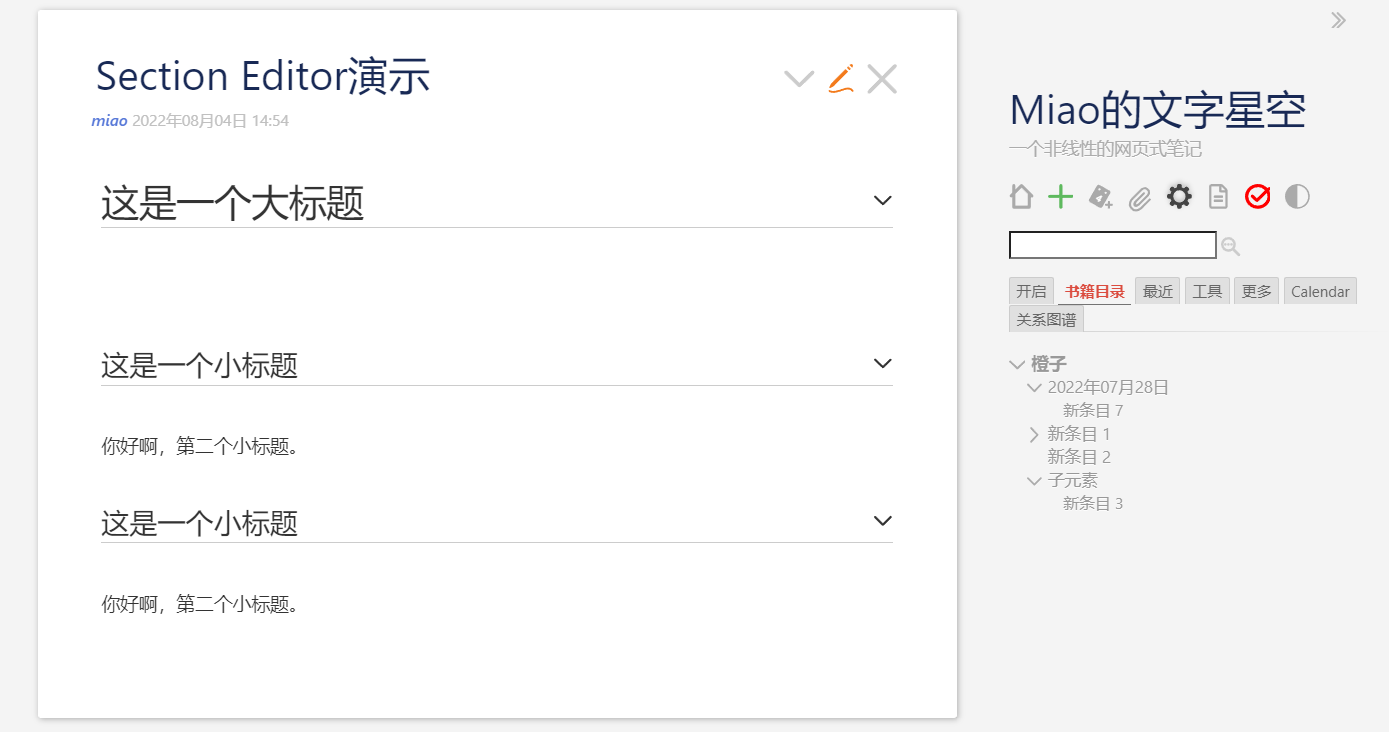
ToC generic：在CPL搜索ToC generic，点击安装。插件描述：将扩展的目录添加到条目的底部。



Section Editor插件，在标题上对大条目进行分段，直接编辑，创建，折叠和管理部分。创建拼凑的条目，将碎片编织在一起以呈现叙事故事。

简单的使用方法：创建一个条目；添加字段: se-type，值可以为空；开始写你的长条目不要忘记添加一些标题 使用 ! and !! 将 tiddler 分成几个部分。保存你的条目。

演示与教程主页：<https://kookma.github.io/TW-Section/>

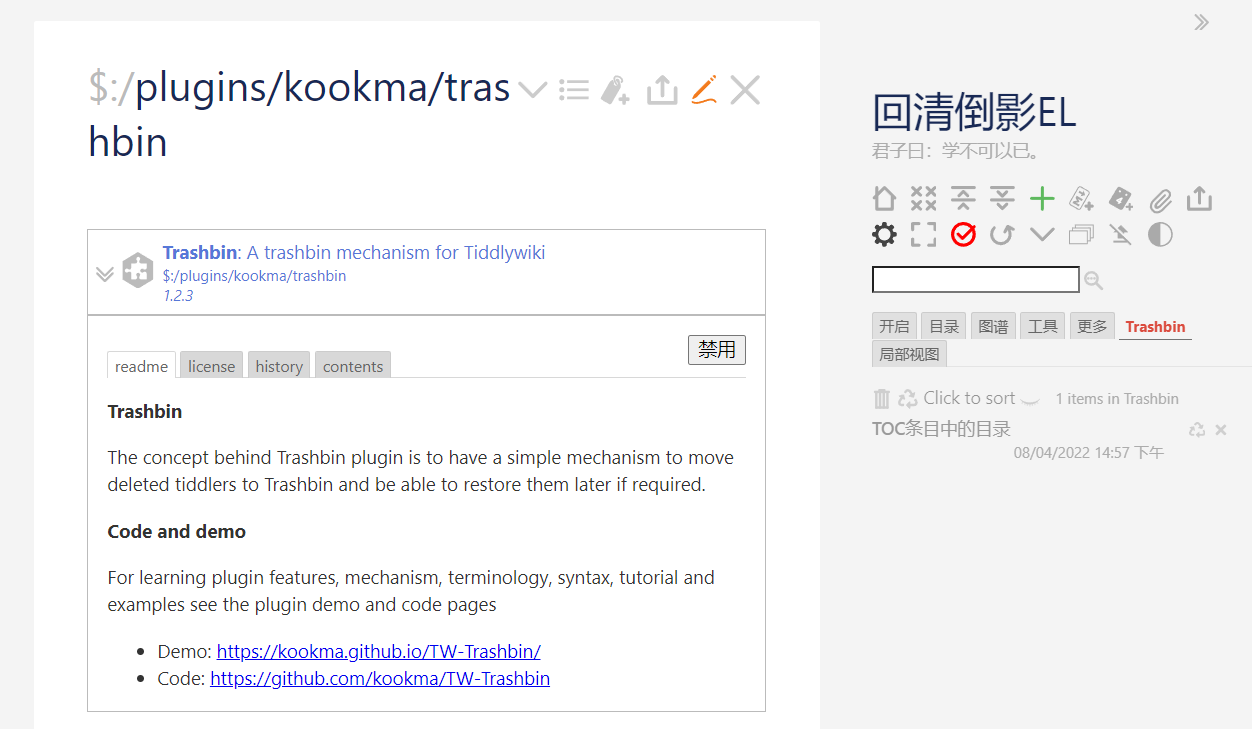


安装Section Editor插件，请注意备份你的文档。。

### （四）回收站插件

在CPL中搜索Trashbin，点击安装。

Trashbin插件背后的概念是有一个简单的机制将已删除的tiddlers移动到Trashbin，并能够在需要时恢复它们。



### （五）随机漫游条目与快照插件

随机漫游条目插件

RandomTiddlerButton：[https://sonephetr2.github.io/](https://sonephetr2.github.io/" \t "dlt)

random：[https://tobibeer.github.io/tw5-plugins/#random](https://tobibeer.github.io/tw5-plugins/" \l "random" \t "dlt)

说明：kiasu/RandomTiddlerButton、tobibeer/random两个插件都需要安装，安装后请刷新wiki。

TiddlerSets快照插件，只需单击一下，即可创建并打开Tiddler集合

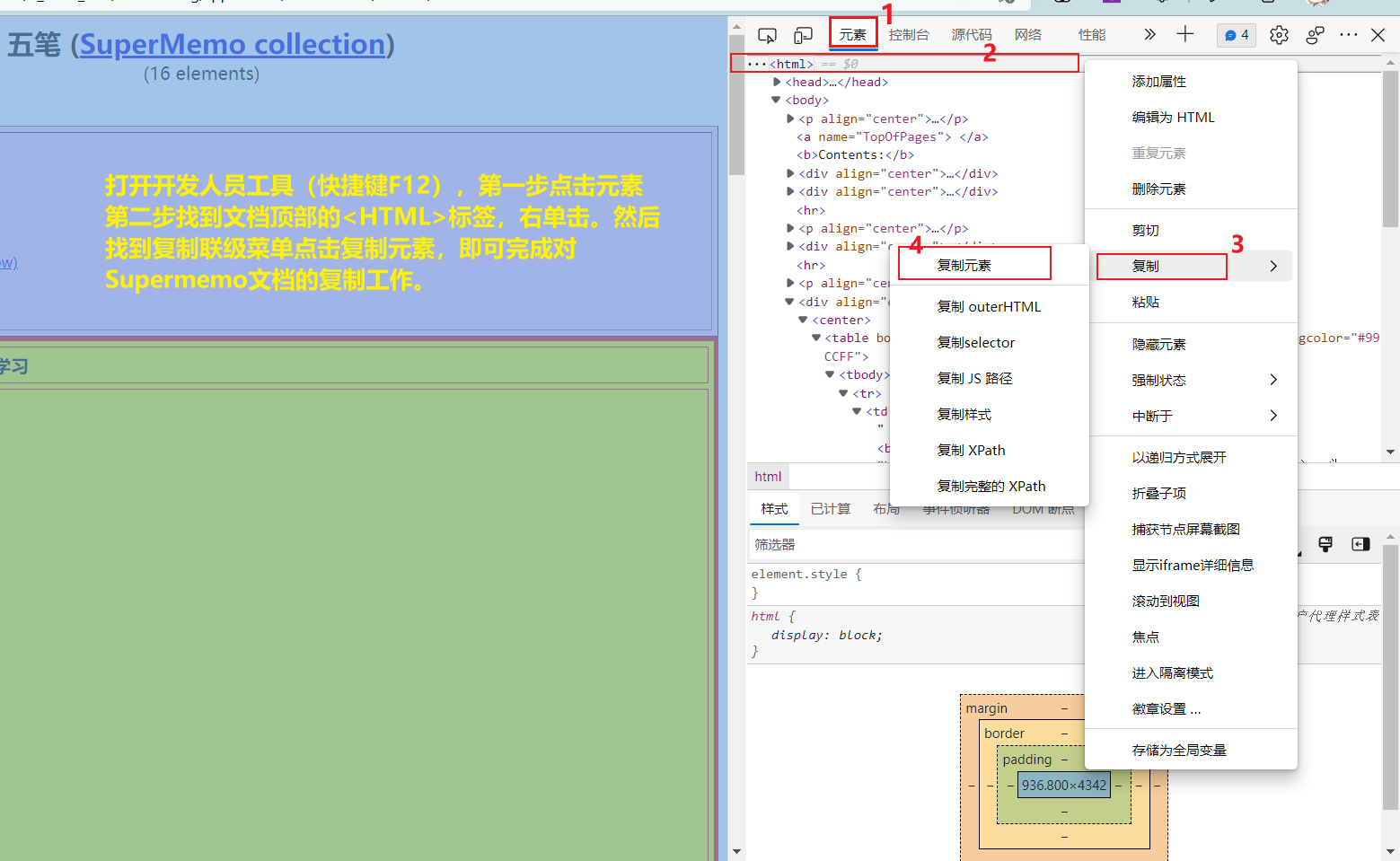
TiddlerSets：[https://tiddlersets.tiddlyhost.com/TiddlerSets](https://tiddlersets.tiddlyhost.com/TiddlerSets" \t "dlt)

### （七）与SuperMemo互动

你可以在SuperMemo中导出HTML文件，在浏览器中经过简单的处理，就可以导入太微中，构建自己的小鱼。

|  |  |
| --- | --- |
| descript | descript |

你可以使用SuperMemo的导出功能导出网页文件，然后用浏览器打开，如图中所述复制整个网页文本，然后在太微中新建一个条目，粘贴到条目中，点击保存就可以完成导入啦。是不是特别简单。



导入的内容你可以按照自己的喜好，使用钓鱼系列的插件学习复习，摘录并做笔记之类的。

## 五、帮助与更多

你可以在以下三个站点找到丰富的资源教程，或者在知乎或B站找到实用的使用方式。我们对新人的建议是，学无止境，够用就好。太微的上限很高，达到一个比较好的效果就可以暂时性的不用学习新的知识。等到有新的功能需求，自己再去探索未知领域。

|  |  |  |
| --- | --- | --- |
| descript | descript | descript |

<https://keatonlao.github.io/tiddlywiki-xp/>

<https://tw-cn.netlify.app/>

<https://bramchen.github.io/tw5-docs/zh-Hans/>

<https://groktiddlywiki.com/read/>

**致谢**

感谢CREATOR（keatonlao）的wikitext教程、suan、FSpark、TomZheng、凹夫几、林一二、葫芦12、sugarhope、少年游、机杼提供的意见和想法以及每一位太微爱好者的支持，非常感谢！

**TiddlyWiki（太微）相关资源：中文教程、插件源、论坛、桌面应用**

1. **太微中文教程：**[**https://tw-cn.netlify.app/**](https://tw-cn.netlify.app/)
2. **社区插件源：**[**https://tw-cpl.netlify.app**](https://tw-cpl.netlify.app/)
3. **TW唯一论坛：**[**https://talk.tiddlywiki.org/**](https://talk.tiddlywiki.org/)
4. **桌面版应用 TiddlyGit （太记）：[https://github.com/tiddly-gittly/TiddlyGit-Desktop](https://github.com/tiddly-gittly/TiddlyGit-Desktop" \t "dlt)**
5. **太记介绍和教程地址：**[**https://zhuanlan.zhihu.com/p/140473235**](https://zhuanlan.zhihu.com/p/140473235)
6. **太微思维导图（六大主题）：**[https://www.zhixi.com/view/70759713](https://www.zhixi.com/view/70759713" \t "dlt)

**希望大家都能爱上TiddlyWiki！ QQ群：946052860**

关于需要了解的HTML知识（可选）：HTML标签、盒子、选择器、属性与内容、HTML结构、应用CSS到元素的概念以及常用的功能。

### （一）桌面端Timimi的安装使用

太微有非常多的保存方式，有一种保存方式Timimi，你可以把它们当做文档与保存文档软件的关系。Timimi仅需配置一次且使用起来几乎感受不到存在。有一个好主意是，你可以使用TiddlyStow搭配Timimi并把它们作为一种日常使用太微单文件版的方式。其中TiddlyStow作为创建太微的角色，Timimi作为保存太微文档的工具。如果担心会丢失数据可以使用Kopia定时备份。

*Timimi 是一个浏览器附加组件 + 一个本机伺服程序，可让你保存和备份你的 HTML tiddlywiki 文件，于你的硬盘的任何地方。 安装完成后，你就可以保存 tiddlywiki 文文件，没有任何额外的步骤，如同原来的 Tiddlyfox 附加组件。*

*下载地址：https://ibnishak.github.io/Timimi/*

*从版本 2.1 开始，Timimi 支持以下浏览器：Chrome/Chromium、Firefox、Opera、Microsoft Edge (Chromium)*

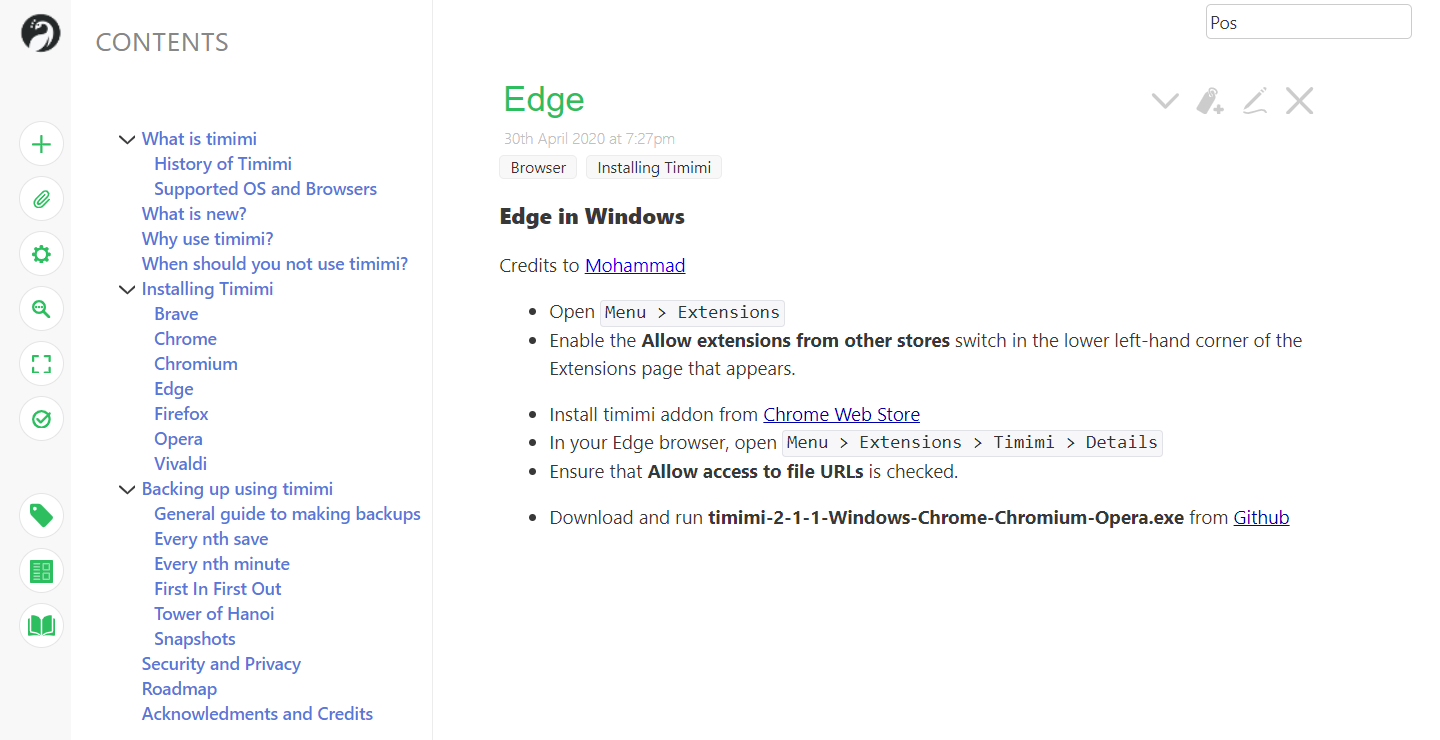
*文本来源：墨屉 — 终身知识，深入脑海 (tiddlymemo.org)*

安装流程概述：1.安装浏览器插件（注意，启用选项，允许来自其它应用商店的扩展）、2.打开允许访问文件URL、3.安装主机伺服软件。

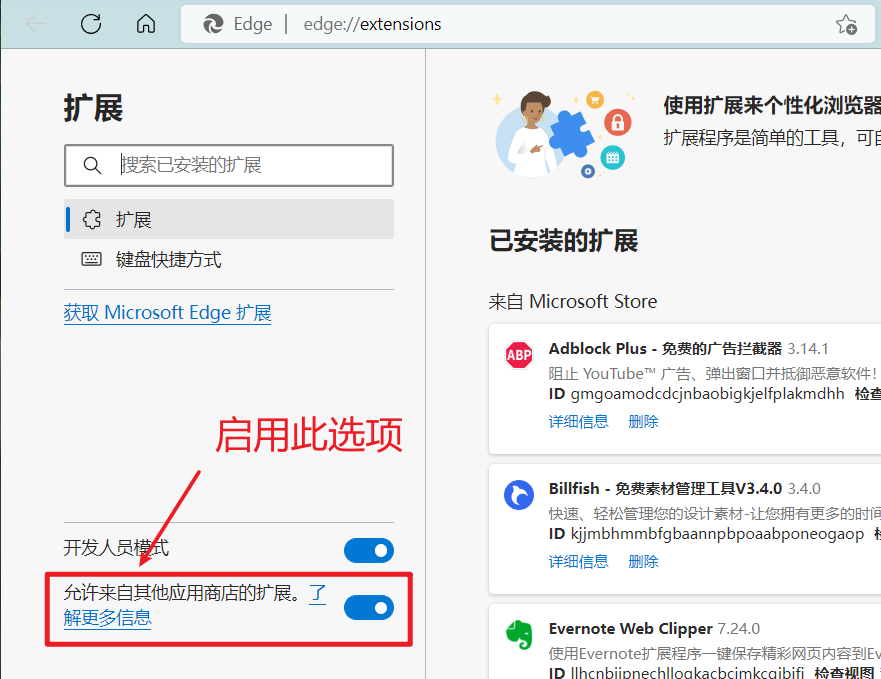
请点击下载地址链接，笔者使用Edge浏览器作为演示，其它浏览器安装流程点击相应的浏览器即可看到，请按照步骤操作完成安装。



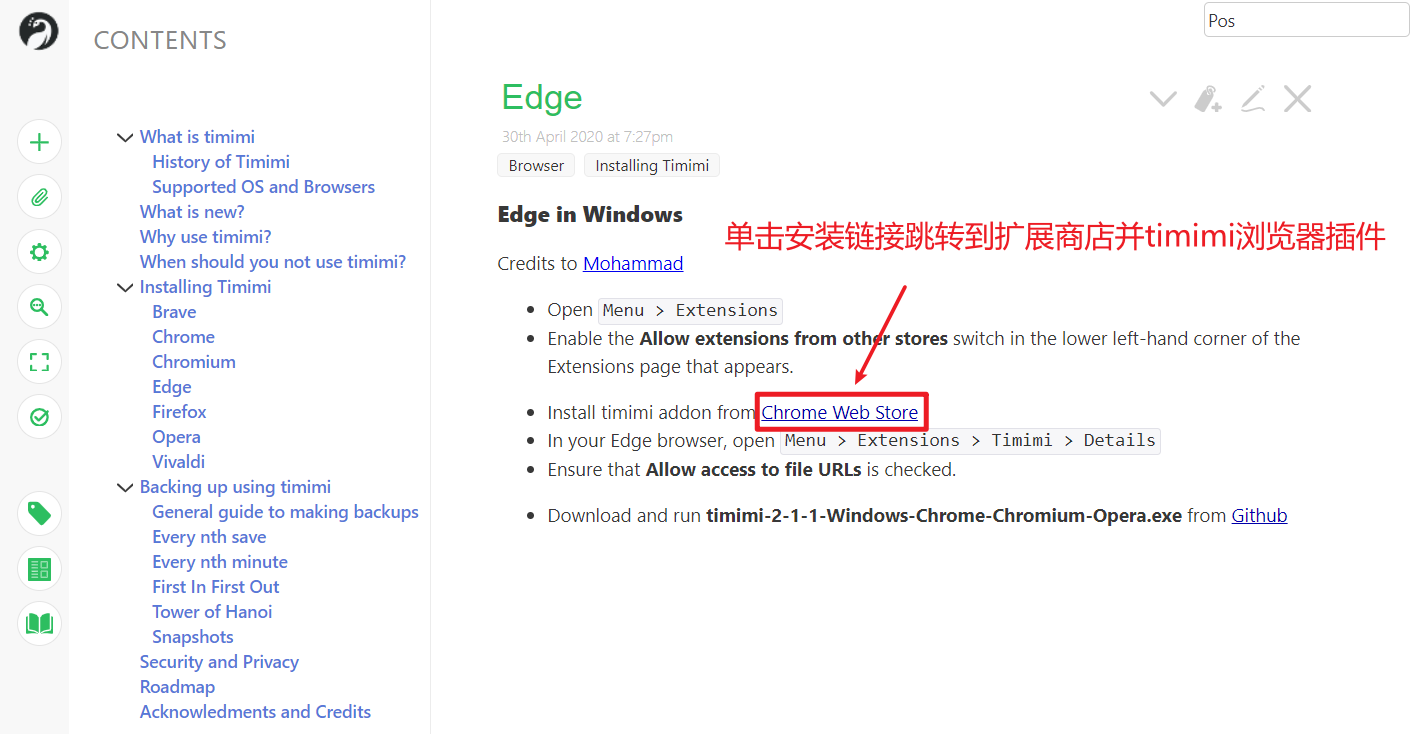
点击对应的浏览器可以看到不同浏览器的安装流程，不会英文？没关系！使用网页翻译试试？



第一步，允许来自其它应用商店的扩展：设置及其它（Alt+F7）> 扩展 > 管理扩展



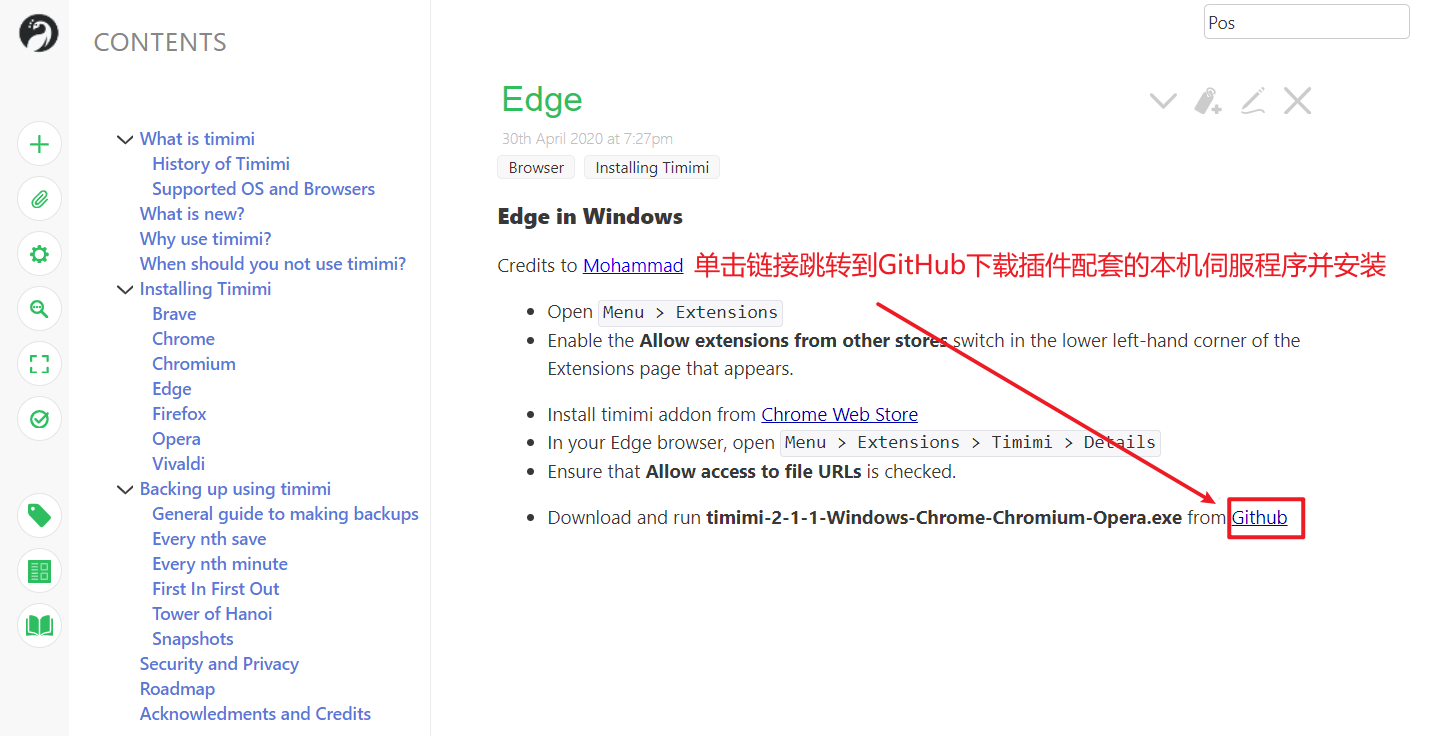
第二步：下载并安装Timimi插件

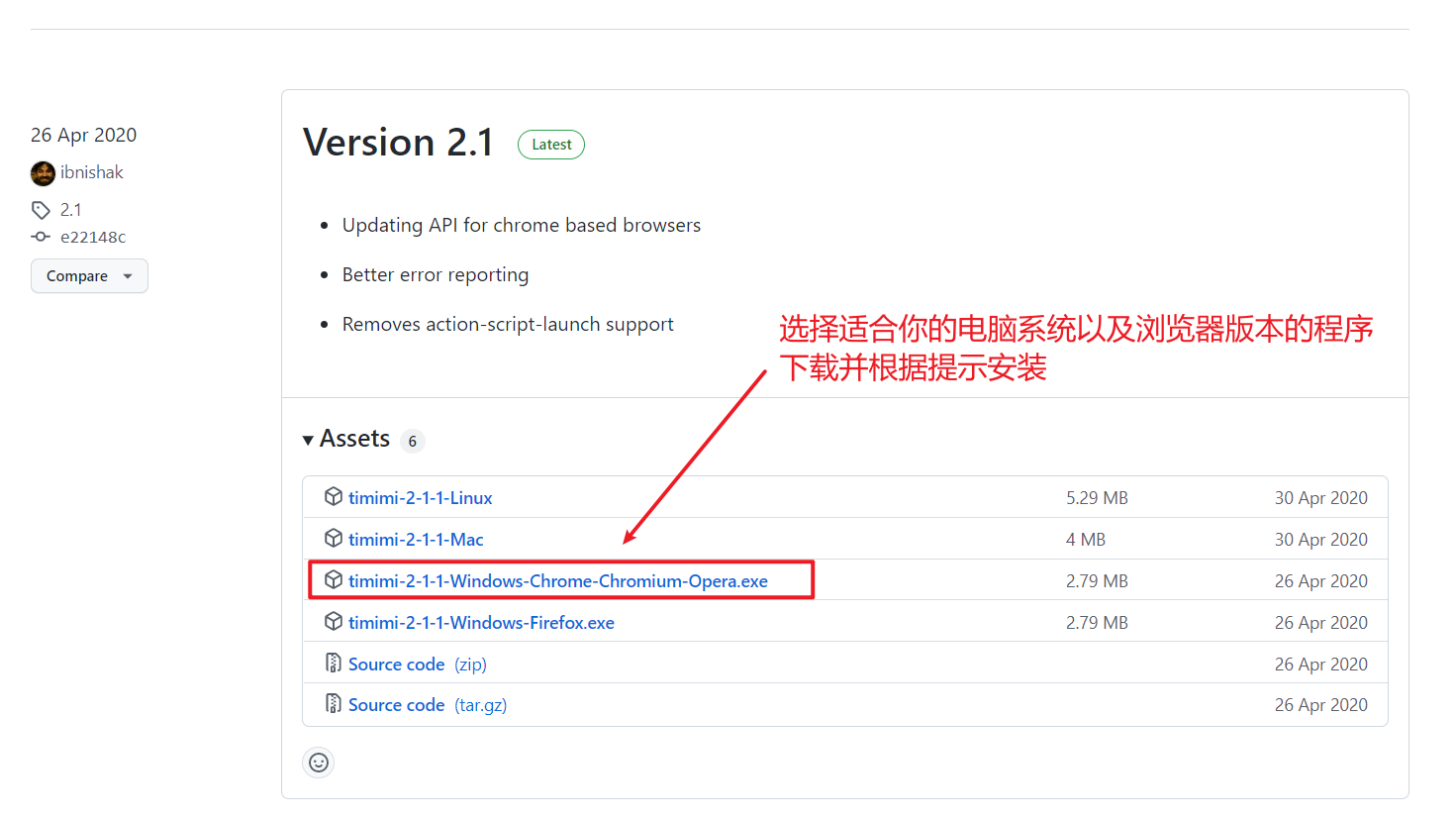


第三步、设置Timimi并允许它访问文件URl：设置及其它（Alt+F7）> 扩展 > Timimi > 详细信息



第四步：下载并安装本机伺服程序（与Timimi插件配套使用，缺一不可）



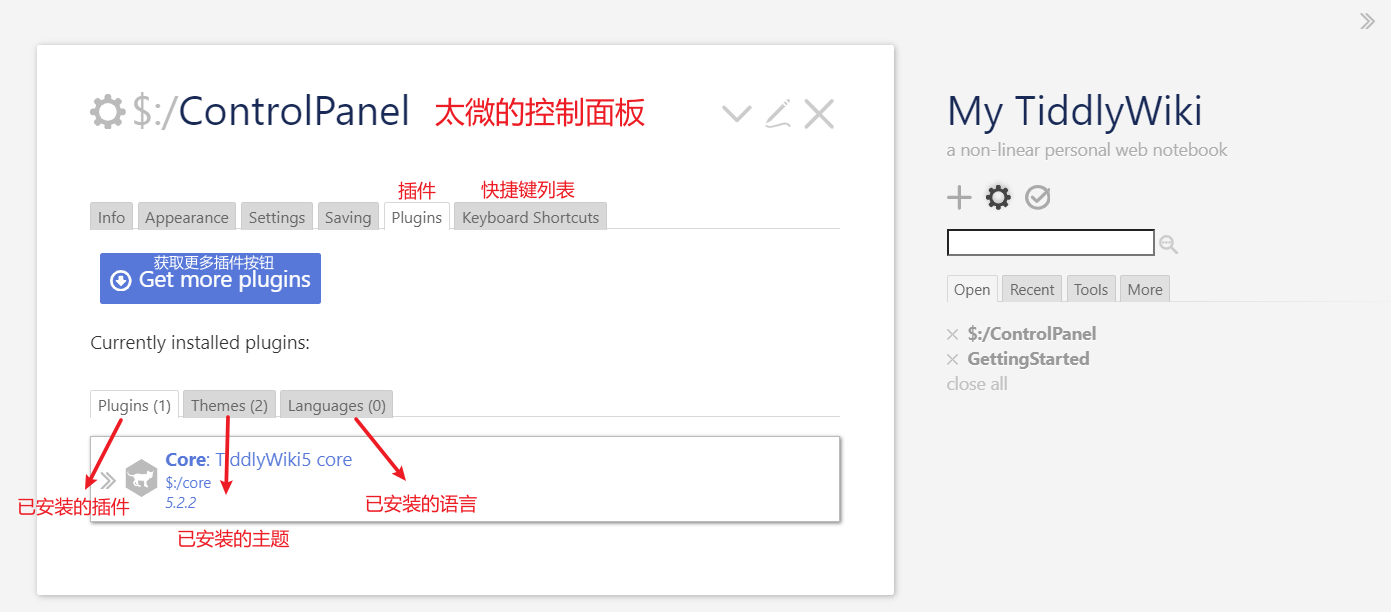


现在，你可以在任意位置保存和备份你的TiddlyWiki文件，并且浏览器将不再会**下载一份更新后的太微。**

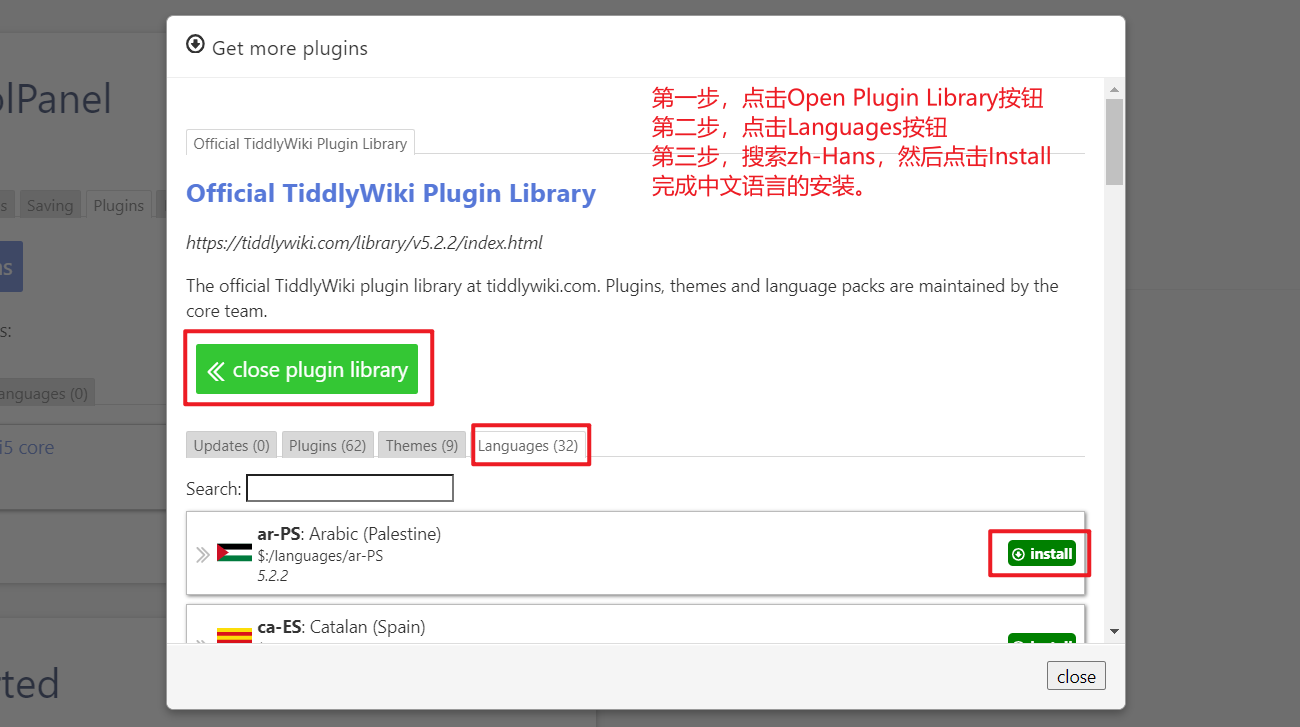
### （二）配置太微的中文语言

有时候我们才从官方下载的太微是英文状态的，那么如何配置它的中文语言呢？

现在，我们将为你配置太微的中文语言。点击页面工具栏的小齿轮按钮（控制台）。在左边找到Plugins（插件）选项卡，点击Get more plugins（获取更多插件按钮）



在弹出的窗口中，如图所示进行操作。



你真的是太棒了，让我们进行最后一步吧！

在太微控制面板 – Info选项卡 – 基本组 ，往下滑找到 “Hello! Current language” 设置，下拉列表选择 Chinese（Simplified）选项完成中文汉化设置。如图所示。

