### higo介绍

1. higo是什么

higo是一个事件管理工具，我对它的定义是作为一种工作方式。

我们每天面对大量的事件，这些事件本身是散乱的，很多人都会用自己的方式去管理它们，我一度因为对这些事件的管理不当让自己的开发电脑桌面变得十分杂乱，所以我产生了一个想法，打算自己开发一个小工具来帮我管理各种事件、资源，希望这个工具不仅能够帮助我整理文件，同时也可以帮助我梳理工作日程中的事件，于是我设计了higo。

higo有两个角色。它是按空间-事件模型来设计的，空间-事件模型是一种简单的模型，其中，空间（Space）指的是事件分类，例如：项目空间、技术空间、工具空间等等，被用来隔离不同类型的事件，从而方便各种事件（Event）的管理，所以higo的第一个角色是基于这种简单的模型来设计的事件管理工具。

例如，创建一个空间——“技术空间”，技术空间里有很多事件，这些事件可以从不同角度去划分，如果根据编程语言去划分，可以是“python事件”、“java事件”、“go事件”等等事件，也可以从要做的事情的角度去划分，比如可能将要学习一种新的技术，则创建一个对应的事件——“XX技术学习”，然后这个事件下面就可以放置各种与“XX技术学习”这件事情相关的各种资源。

同时，higo也是一个命令行工具，基于上述模型，定制了一些尽可能简单的操作命令，相比手动维护各种资源会更加便捷，除此之外，也可以通过扩展脚本的方式来增强higo，比如说，我最初设计higo的目的之一是想要实现“每有一个新的工作任务，就可以通过higo帮我初始化一些开发资源，包括但不限于帮我自动clone一份新的代码然后用ide工具打开它们”，对于这个事件就需要我通过自定制脚本来完成。

需要说明的是，higo一开始就没打算作为一个大众化的工具来设计的，它有自己的适合群体，如果想要更好的使用higo，需要不同程度的具备以下几个条件：

1. 软件技术人员，因为higo是一个命令行工具，所以需要一定的技术功底
2. 掌握至少一种脚本语言，如python、shell、bat等，如果不想写脚本，那么不适合使用higo
3. 相信在定制了一些脚本的基础上，命令行的操作方式要比图形化界面操作更加便捷
4. 具备自动化思维，讨厌机械重复的操作
5. 具有任务管理意识，希望历史的工作事件一段时间后留有记录可查

那么，higo有什么好处？我列了几条可以作为参考：

1. higo可以使工作变得自动化，目前主要通过自定制脚本实现
2. higo可以使工作变得可管理，让工作和记录融为一体，工作即记录
3. higo也可以作为脚本管理工具，让散乱的脚本有个“家”

如果有兴趣的话，可以在使用过程中再进一步体会。

1. higo的使用

首先，将higo的压缩包解压，然后将higo的执行文件（windows环境是higo.exe）所在目录加入环境变量中，再进一步操作。

higo有两种操作方式，交互式和非交互式：

交互式：运行higo命令可进入交互式命令行；

非交互式：直接在命令行运行higo {action} {option1} {option2} ... 命令

两种方式需要配合使用。

1. 创建空间和事件
2. 创建新的空间和事件

进入交互式命令行后，执行hi命令，在higo层可以创建空间，在空间层可以创建事件。

1. 将已有的目录链接为一个higo中的空间或事件

通过非交互方式操作：

|  |  |  |
| --- | --- | --- |
| 非交互式 | 链接空间 | 链接事件 |
| action | link | |
| option1 | -s或--space | -e或--event |
| option2 | -n或--name：指定空间名称 | -n或--name：指定事件名称 |
| option3 | -t或--target：指定目标目录 | -t或--target：指定目标目录 |
|  |  | -i或--into：指定事件归属空间 |

1. 查询空间和事件

进入交互命令行后，执行look命令，会返回空间或事件列表，并且每个空间或事件名称前有唯一的数字序号，在higo层可以查看空间，在空间层可以查看事件。

1. 删除空间和事件

进入交互式命令行后，执行del命令，在higo层可以删除空间，在空间层可以删除事件

1. 进入空间或事件、返回上一层

进入交互命令行后，先执行look命令，再执行go {id}命令进入id对应的空间，id指的是look空间或事件列表中的空间和事件数字序号。

如果需要返回上一层，执行back命令。

1. 快捷方式切换当前工作目录到上一次访问（go）的事件目录

|  |  |  |
| --- | --- | --- |
| 非交互式 | 快捷切换到事件目录 | 快捷返回切换前的目录 |
| action | go | back |
| option1 | 该选项无参  -e或--explorer：是否打开文件夹图形界面，暂时仅在windows系统支持，如果未指定该选项，则仅仅切换当前工作目录，仍旧在命令行界面 | 无 |

1. 安装和运行脚本

|  |  |  |
| --- | --- | --- |
| 非交互式 | 安装脚本 | 运行脚本 |
| action | install | 脚本名称 |
| option1 | -t或--target：指定目标脚本完整路径 | 脚本选项1 |
| option2 | -i或--into：可选，如果是安装到某个空间或事件，可指定该参数，格式是{spacename}/{eventname},  如果未指定，默认安装到最近一次访问（go）的空间或事件，由于输入空间名称和事件名称相对繁琐，所以一般建议先通过交互式命令行访问（go）期望空间或事件后再执行install | 脚本选项2 |
| option3 | 该选项无参，指定安装给哪一层  -h或--higo：安装到higo层，也就是最上层，该层的脚本会被作用到被higo管理的所有空间层和事件层，  -s或--space：安装到空间层  -e或--event：安装到事件层  如果下层存在同名的脚本，执行脚本时，则只会执行最下层的脚本，即下层优先级高于上层，只有下层不存在时才会执行上层的目标脚本 | ...... |

需要注意的是，扩展和运行脚本默认作用到最近一次访问访问（go）到的事件，如果需要对另一个的事件安装或运行某个脚本，则先要切换（访问）该事件。

安装的脚本有两种类型，一种是一个单文件，直接安装即可，另一种是多个文件，需要放在一个文件夹里，并且文件夹中必须有文件名为run的文件作为脚本执行入口。

最好不要手动安置脚本到higo脚本目录下，因为这样放置的脚本可能不符合higo的脚本执行要求，而通过install命令安装的脚本在安装过程中会进行检查，从而在一定程度上避免安装后无法正常使用的情况。

1. higo未来扩展

略