基于命令行的网站书签编辑器

一、编译&运行环境说明

使用cmake编译，进入项目根目录：

```
mkdir build
cd build
cmake .. -G "Unix Makefiles"
make    //若编译太慢使用 make -j
cd ..
.\bookmark.exe
.\bookmark_test.exe
```

自动测试程序(`bookmark_test`)和目标可执行文件(`bookmark`)会输出到根目录。理论上可以在任何平台编译运行，但请注意本项目编码为GBK。

P.S.由于使用了一些现代C++特性，请确保编译器版本至少完全支持C++17，否则会编译失败(我使用的是gcc 12.1.0版本)。

`bookmark_runable.exe` 与 `bookmark_test_runable.exe` 分别是提供好的可运行的目标程序和测试程序(请不要移动测试的位置，否则ls-tree测试会失败)。

二、面向对象设计

1.整体架构与设计模式

书签-标题系统采用组合模式，抽象出一种`class BmkElement`作为基类让`class Title`与`class Bookmark`继承，作为最底层的抽象层次，实现增删改查等接口。

add/delete命令的执行、undo/redo采用命令模式，`class AddCmd`与`class DeleteCmd`继承命令基类`Class Command`记录本次命令相关的信息，并指向相应的`class BmkElement`。

显示书签/标题的标签的显示以其bmk格式的生成使用装饰器模式。

ls-tree/show-tree指令使用策略模式和适配器模式的思想，首先实现了一种通用的打印树形结构的算法，将不同类型的树作为一种策略，每个策略都有也必须提供相应的接口(获取子节点列表：`GetList()`，是否有子节点：`HasChild()`，对外展示的字符串：`ShowStr()`)，同时将已有的书签-标题系统与文件系统(`std::filesystem`)的接口对其做适配。不仅实现了代码复用，还实现了显示/UI层与底层实现的解耦。

抽象出一个控制器`class BmkController`对整个编辑器系统进行控制，使用书签-标题系统提供的API与命令模式完成每一种命令，并维护历史与undo后的缓冲命令流，可通过修改相应参数支持类似文本编辑器一样的更加广泛的undo/redo。

2.手工测试与测试驱动开发

使用cmake链接并使用`gtest`库进行单元测试。

在test目录中的`bmk_tdd.cc`中针对11个重要的底层API或其组合起来的功能进行较为完备的测试。

`manual_test`下具有可执行文件与测试文档相同的测试环境，可通过所有测试。

P.S.直接打开自动测试程序可能会导致自动测试程序完成后自动退出并看不到输出结果。

3.其他

本项目遵循Google C++ Style Guide。