华 北 理 工 大 学 计 算 机 设 计 大 赛

软 件 开 发 类 作 品 文 档

作品编号：xxxxxx

作品名称：简·编辑器

作　　者：宋子慧 202014240620 苏浩伟 202014240621

版本编号：1.0

填写日期： 2022 .2 .9

填写说明：

1、本文档适用于**所有**涉及软件开发的作品，包括：软件应用与开发、大数据应用、人工智能应用等；
2、正文一律用五号宋体，一级标题为二号黑体，其他级别标题如有需要，可根据需要设置；
3、本文档为简要文档，不宜长篇大论，简明扼要为上；
4、提交文档时，以PDF格式提交本文档；
5、本文档内容是正式参赛内容组成部分，务必真实填写。如不属实，将导致奖项等级降低甚至终止本作品参加比赛。

目 录

第一章 需求分析.....1
第二章 概要设计.....1
第三章 详细设计.....4
第四章 测试报告.....**错误!未定义书签。**
第五章 安装及使用.....7
第六章 项目总结.....9

第一章需求分析

开发这个编辑器的主要原因是一方面在 Linux 等命令行界面操作编辑文件不直观，以 vim 为例子，在使用时学习路线很长，命令的确可以便捷我们的操作，但是学习众多关于编辑器的命令组合，让人脱离了编辑文件本身的目的。

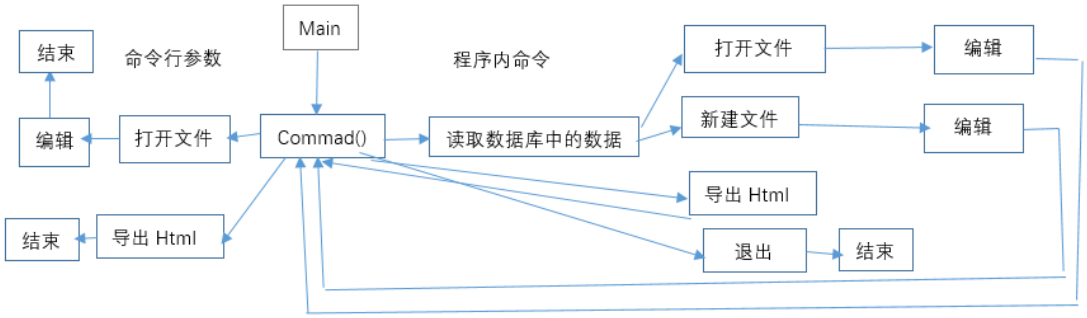
还有一方面是在 Linux 服务器上开发一些小型网站或个人博客网站时，很多框架的依赖以多，如果是初次使用很可能会安装失败。并且由于 npm 的网络原因安装 hexo 等博客框架时很可能会安装失败，而 wordpress 对与树莓派等小型机器显得很臃肿，动态网页会导致访问速度受限。

所以我设计了这个编辑器。在流程方面类似与 vim 使用命令对软件进行操作。

主要的流程：

两种输入命令的方式一种是进入程序后，输入命令，执行操作。

第二种是在运行程序时加入命令行参数，运行程序，直接执行操作。



简化了命令的输入，便捷的编辑文件。

对于导出 html。采取了简化 html 标签的方式，如果用标准 html 编写文章，在文章的内容排版上比较混乱，所以简化了 html 的标签，使其更接近于文章，并且可以直接转换成为标准的 html，在此基础上加入了主题的选择，在配置文件里设置要用的主题，生成的 html 即可可简单的添加进主题的 css。

例如 这个文件节选

 # OpenGL 学习 OpenGL 环境配置

url https://learnopengl-cn.github.io/01%20Getting%20started/03%20Hello%20Window/ 一个很好的教程网站

最好下载 32 位的版本，64 位可能会有一些问题。

下载完后解压。

可以得到

然后打开文件所在的文件夹

新建出 include 和 lib 两个文件夹

Img https://pcsdata.baidu.com/thumbnail/b013491d2v8a0dc9f10eb1b5a3b6fff6?fid=3125802318-16051585-660741239604933&rt=pr&sign=FDTAER-yUdy3dSFZ0SVxtzShv1zcMqd-prgEJAGPRfJVXj%2BTbwJ%2FjDMwAms%3D&expires=2h&chkv=0&chkbd=0&chkpc=&dp-logid=8750349927098848O&dp-callid=0&time=1641290400&bus_no=26&size=c1600_u1600&quality=100&vuk=-&ft=video

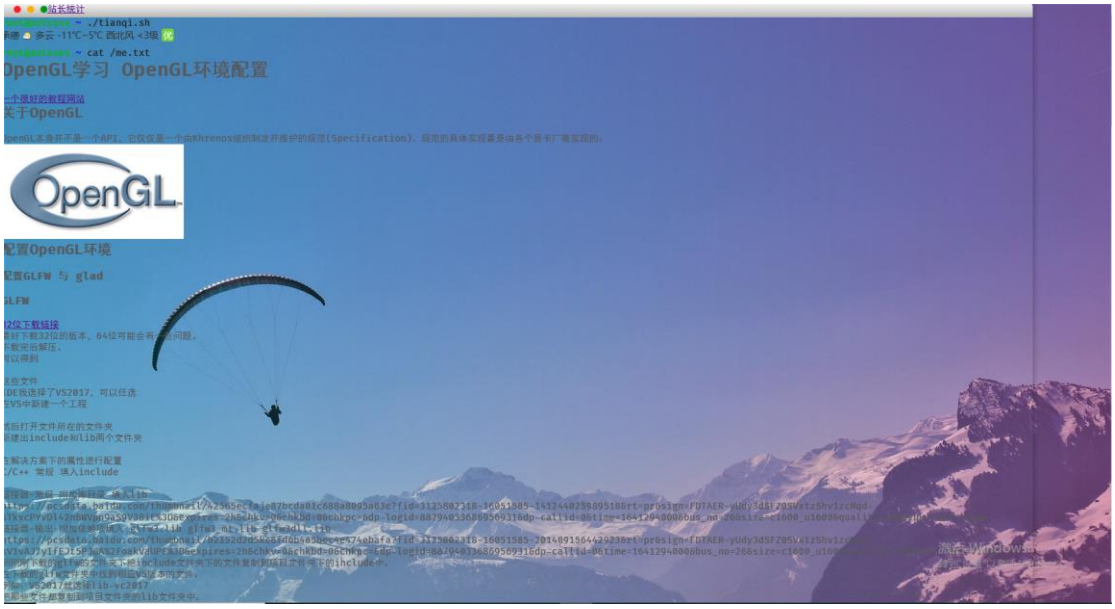
*****

如果屏幕上出现一个三角形，则环境配置成功。

img https://pcsdata.baidu.com/thumbnail/3c8d29b13nde840a9b79f91b10968119?fid=3125802318-16051585-466801189607147&rt=pr&sign=FDTAER-yUdy3dSFZ0SVxtzShv1zcMqd-

W7ejhaG4RGNnhUohgEuJ9Yfklho%3D&expires=2h&chkv=0&chkbd=0&chkpc=&dp-logid=88794033686956931&dp-callid=0&time=1641294000&bus_no=26&size=c1600_u1600&quality=100&vuk=-&ft=video

html 效果 截图



html 源码 节选

这样就可以生成一个比较美观的网页。

第二章概要设计

编辑器的操作方式以命令为主。

使用方法一（Windows 版 与 Linux 版都如此）： 在：（冒号）后输入相应的命令和参数后执行相应的操作。

使用方法二（Linux）： ./EC –v 查看版本

./EC 文件名 打开文件

./EC –h 文件名 转换相应文件

编辑器分为几个以下模块：

命令解析

文章解析

数据读取

编辑器核心

代码补全

代码高亮

数据库

User_file.db

COMPANY 表存放补全的关键词

HIGHLITE_C 表存放高亮的关键词

表的结构

```
CREATE TABLE COMPANY(
    KEY_WORDS TEXT NOT NULL
);
CREATE TABLE HIGHLITE_C(
    KEY_WORDS TEXT NOT NULL
);
```

编辑器框架

```
main.cpp
| int easyhtmleditor::commander(int argc,char* argv[]) commander 函数
-----EasyCodingEditor.cpp 编辑器界面与命令判断
            | h 命令
            -----   Article_device.cpp 文章解析器，生成标准 html
            |
            -----   sql.cpp 调用数据库接口，读关键字数据导入到核心中的相应数组中
                    | i | a | wq
                    -----   Edit_kernal.cpp 编辑器核心
                                |
                                ----- Code_completion.cpp 代码补全功能
                                |
                                ----- Code_highlighting.cpp 代码高亮功能
```

Main.cpp 把 argc 和 argv 传给 EasyCodingEditor.cpp 中的 commander 函数，之后根据命令行的参数或者输入的命令执行操作。

命令行参数

–v 输出版本号

printf

./EC 文件名 打开这个文件

save_files(key_words,page_arr);调用 save_files 函数

-h 文件名 转换这个文件（后缀要为.eh）为标准的 html

Article_device.cpp 中的 readout_emakefile();读取关于 html 生成的相关配置，再次调用 Article_device_run(key_words);传入文件名，解析文章。

程序内的命令

i 或 a 创建一个新文件进行编辑

先调用 sql.cpp 中的 select 函数,之后直接调用 Edit_kernal.cpp 中的 Edit_kernal()函数

h 文件名 转换这个文件（后缀要为.eh）为标准的 html

Article_device.cpp 中的 readout_emakefile();读取关于 html 生成的相关配置，再次调用 Article_device_run(key_words);传入文件名，解析文章

b 有主界面返回上一个打开或正在编辑的文件

清屏

输出当前页，调用 Edit_kernal.cpp 中的 Edit_kernal()函数

q 直接退出不保存

直接 return 0；

wq 文件 保存文件并退出

save_files(key_words,page_arr);调用 save_files 函数

esc 由编辑界面返回主界面

erase();

nl();

echo();

nocbreak();

清屏。还原一些设置。

g 命令 使用 gcc/g++ 编译程序

使用了一个 system 函数，执行命令

v 文件名 打开一个已经存在的文件

先调用 open_files 函数，返回 true 则调用 Edit_kernal()函数。

/ 字符串 查找该字符串第一次出现时的位置

Page up 上一页

坐标转换，清屏，显示下一页

Page down 下一页

坐标转换，清屏，显示下一页

Sql.cpp

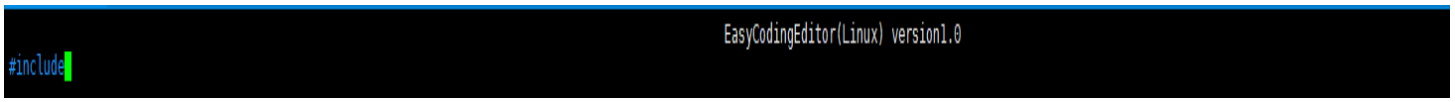
deque <string> select(*const char* * database,*const char* *table)函数，调用数据库接口，读取数据。

第三章详细设计

在设计过程中主要有以下一些问题。文字的排版，光标移动处理，多语言支持（还未实现），代码补全，html 转换。

对于文字排版，是以光标移动处理为基础，而对于多语言支持（还未实现）又是以文字的排版为基础。

所以对于光标移动的处理，我们有 pos_y 和 pos_x 来作为一组坐标来表示光标到位置以及作为二维数组（page_arr[page_now-1][pos_y - (page_now-1)*(page_y-2)]）的下标，而光标的位置在屏幕与底层的二维数组的下标实际上是有一个偏移的，如下图所示：



实际的光标位置显示到了第二行，但是此时下标为 0，pos_y 应该是 1，会发现有 1 的偏差，会导致数组溢出，程序崩溃。另外如果文件大于一页可以显示的内容（大于了屏幕缓冲区的高时）文件会出现显示不开的现象，所以这是引入了分页的机制，一个文件其实是以一个三维数组（deque < deque <string> > page_arr;
）的形式进行保存。这样就可以通过下标进行随机访问，一页中的一行，每一行使用一个 string 的类型，所以由此也可以访问到具体的哪一个字符。因为每一页的页首显示了标题，在页脚显示了光标位置信息，所以，每页的实际显示文件内容区域时 0 到屏幕缓冲区高减-2（屏幕缓冲区的高用 page_y 表示，屏幕缓冲区的长用 page_x）,用 page_now 来记录当前的页数数，所以 pos_y 映射到数组下标应该是 pos_y－(page_now-1)*(page_y-2),由此我们就成功确定好了光标在每一页上的具体位置。下一步我们移动光标则可以使用在 Linux 上有 curses 库提供的 move 函数来移动光标位置。在为 windows 上，我们则可以直接调用 windows 的 api 实现

```
//设置光标位置
void easyhtmleditor::SetPos( short int  x , short int  y ){
        COORD  point = {  x ,  y  }; //光标要设置的位置 x,y
            HANDLE  HOutput  =  GetStdHandle(  STD_OUTPUT_HANDLE  );  // 使 用 GetStdHandle(STD_OUTPUT_HANDLE)来获取标准输出的句柄
        SetConsoleCursorPosition(HOutput, point); //设置光标位置
}
```

对光标位置进行移动。对于光标的移动我们可以想到，什么时候光标需要移动。在输入了一个字符时需要向右移动，按了退格键后光标需要向左移动，方向键上光标会向上移动，方向键下光标会向下移动，方向键左光标会向做移动，方向键右光标会向右移动，在行尾安回车或在空行按回车，会在创建出一个空行，光标并移动到这行行首的位置。

实现了光标的移动成功的为接下来的操作打下了基础。来到文字排版，文字排版中我们可能会遇到这些冲突，对输入的内容代码高亮时，高亮的内容与不高亮的内容错位，在对输入内容进行补全时，补全的内容输出的位置不对盖住了我们接下来又输入的内容。所以我们先使用一个 switch 的结构读入我们的键盘输入，当我们输入的不是键盘上的功能键时，会进入 default，这是会在屏幕上回显当前的字符，接下来判断该字符的位置如果时在行尾直接 push_back 进该行，如果是在行首就把他插入到行首，如果他是在中间就 insert 函数插入 pos-1 个元素前，如果其在一个空行直接类型转换 char 为 string push_back 进页数组。这样我们就得到了完整的一行的内容，而代码补全不是根据一行进行补全，应该是随着用户的输入记录下我们输入的每一个词，所以紧接着调用代码补全的 cc.Lexical_analysis(ch2,pos_y - (page_now-1)*38,last_x);在 string Code_completion::Lexical_analysis(*char* c,*int* pos_y,*int* pos_x)函数中，用 c_str 来记录用户输入的每一个词，每次空格或 TAB 或回车后都清空该字符串，之后根据该字符串在全部的可补全数组中查照匹配，到匹配数为 1 时进行补全。这是返回要补全的词。last_str 中记录了还未清空时的 page_arr[page_now-1][pos_y - (page_now-

1)*(page_y-2)]，也就是未补全时行内的内容（并未记录刚刚输入的词），此时在 last_str 后加上补全的内容，在行首输出实现简单的代码补全。

```
page_arr[page_now-1][pos_y - (page_now-1)*(page_y-2)].clear();
last_str+=code_compl;
page_arr[page_now-1][pos_y - (page_now-1)*(page_y-2)] = last_str ;
SetPos(0 ,pos_y - (page_now-1)*(page_y-2)+1);
//cout<<page_arr[page_now-1][pos_y - (page_now-1)*(page_y-2)];
printw("%s",page_arr[page_now-1][pos_y - (page_now-1)*(page_y-2)].c_str());
```

最后是代码高亮这个是采用字符串匹配的方法实现的，用户的每次输入都会调用 *bool* Code_highlighting::Lexical_analysis(deque <string> ready_highlight,deque <string>file_data,*int* pos_y)函数，以次从该页中查找要进行高亮的词，并且把此时的 i，加进 state 中，state[i]与 key_words2 相应下标下的关键字一一对应。deque <string> ready_highlight 的下标+1 是这个词的 y 坐标，find 的返回值+tab 的个数*8 是 x 的坐标以此可以确定出要补全的词的位置，之后输出，呈现出高亮效果。

最后是 html 转换，这个主要是简化了 html 标签的一些写法在加上一些固定的主题，可以快速制作出很多网页。相对与 hexo，WordPress，hugo 的博客框架，依赖更少，更小。主要的难点是快速准确的识别标签进行转换。所以在设计上先根据 int Lexical_analyzer(vector <string> analyer1)函数确定出该标签是哪一个，这是会返回一个 int，在 *int* Grammatical_analyer(*int* State,*int* number,vector <string> arr,*int* bit3)函数中的 switch 根据上面 Lexical_analyzer 函数的返回值进行处理生成相应的 html 语句，关于语句嵌套的问题,在上一步处理完成后会有 end_state_machine 这个数组，在向这个数组里 push bit，这样正好是反的，在 Grammatical_analyer 执行到最后是在倒序输出结尾部分。这样就实现了语句的嵌套。

第四章 主要测试过程结果及其修正

1. 大文件打开测试
   一开始没有分页的机制，如果在文件首移动一个字符，则需要从后往前移动很多次，影响了效率，而且显示内容的多少与屏幕缓冲区的大小有关，大文件显示不全。因此采用了分页机制，通过 page up 和 page down 来控制页数，这样就实现了大文件的显示。
2. 代码补全测试
   1> 有时输入例如#直接会被不补全为#include，没法输入单独的#。所以因为补全的机制是当用户的属于与可补全的内容匹配的可能性为 1 时进行补全，而#正对应着#include 只有一种可能，所以解决方法是在可补全的内容里添加上#####使输入#时的可能性不为 1。
   2> 无法在行中间补全。例如 using namespace std；可以补全行首的 using，但是不可以补全句中的 namespace。解决方法是用 c_str 来记录用户输入的每一个词，每次空格或 TAB 或回车后都清空该字符串，之后根据该字符串在全部的可补全数组中查照匹配，到匹配数为 1 时进行补全。这是返回要补全的词。last_str 中记录了还未清空时的 page_arr[page_now-1][pos_y - (page_now-1)*(page_y-2)]，也就是未补全时行内的内容（并未记录刚刚输入的词），此时在 last_str 后加上补全的内容，在行首输出实现简单的代码补全。
3. 代码高亮测试

在一开始，高亮的内容与文件中的其他内容错位，参差不齐，但是有是正确的，有的是错误的。后发现是在写文件是使用了制表符进行缩进，有的用空格进行了缩进，导致实际位置与显示的位置出现了偏差。(\t 在数组中是一个字符，但是打印出来实际上 4 个或者 8 个空格)，所以在高亮的输出位置加上/t 的数*8，解决问题。

4. Html 转换测试
   在测试中::（拓展标签）会导致数组下标越界崩溃，暂时先取消了这一标签。

   编辑器根据命令来执行相关操作，所以可以通过增加命令的方式，来拓展程序，拓展性尚可。

第五章 安装及使用

该软件主要应用于 Linux 平台，也有 Windows 版本。

文件目录

EasyCodingEditor

- ---- .vscode 里面是 Vscode 中调试与编译配置文件
- ---- Linux
  - ---bin 可执行文件
  - ---build makefile 文件
  - ---data 数据库
  - ---include 头文件
  - ---src 源文件
  - ---test 几个测试文件
  - ---Website
    - ---生成 html 的目录
    - ---conf 拓展列表
    - ---expand 拓展的具体内容
    - ---theme css 主题
- ----- Windows
  - ---build makefile 文件
  - ---Code_completion 代码补全的关键字
  - ---Code_highlighting 代码高亮的关键字
  - ---include 头文件
  - ---src 源文件
  - ---test 测试文件

Linux

需要用到的库或其他软件。

sqlite3 数据库

curses 库

安装相关库：

sudo apt-get install sqlite3

sudo apt-get install libsqlite3-dev

sudo apt-get install libncurses5-dev
解开压缩包
tar –xvf SongZihui_SuHaowei_EasyCoding_Editor.tar
或从网络上下载源码 sudo git clone https://github.com/SongZihui-sudo/EasyCoding_Editor
要输入命令这个文件夹可读可写权限。
sudo chmod 777 –R EasyCoding_Editor
进入 Linux 文件夹
1.**编译源码**
cd 到 build 文件夹 ， 在终端输入 make
2.**运行**
cd 到 bin 文件夹
会有一个文件名为 EC 的文件。
终端输入 ./EC -v 会输出 EW:简·编辑器 Linux version:1.0 ,编译成功。

Windows
解开压缩包 SongZihui_SuHaowei_EasyCoding_Editor.tar
或从网络上下载源码
git clone https://github.com/SongZihui-sudo/EasyCoding_Editor
1.**编译源码**
进入 Windows 文件夹
进入 Build
win+r，打开运行，输入 cmd。
再在 cmd 中输入 mingw32-make
2.**运行**
在该目录下会生成可执行文件，鼠标点击打开。

使用

编辑器中的一些操作
i and a 像 vim 一样进入编辑模式,创建一个新文件
wq \n filename 保存文件
方向键上 向上移动光标
方向键下 向下移动光标
方向键左 向左移动光标
方向键右 向右移动光标
回车 插入空行
删除 删除
空格 插入空格
q 不保存直接退出
\ 字符串 在文件中进行查照
v 文件名 打开一个文件
page up 上一页
page down 下一页
g \n xxxx gcc 命令编译 C/C++代码
h \n 文件名 转换导出 html

./EC -v
./EC 文件名
./EC -h 文件名

Html 标签简化
标题标签：#
图像标签：img 图片的链接
链接标签：url 网址 名称
代码块标签：
三个` 表示开始
三个* 表示结束
中间的内容表示代码
引用标签 >
分割线标签 --- 再单独一行
强调（加粗） __ 两个_
[会导致崩溃，有问题] 拓展标签 :: 拓展名

第六章项目总结

首先是从这次的软件的开发我们从中学习到了很多东西。首先是关于 Linux 上的各种操作体验到了 Linux 的种种特性，以及 Linux C/C++ 开发的很多新知识，与 windows 不同的是 Linux 上的 C/C++读写文件时会遇到的权限问题，从这我们可以学习到用户的机制，利用用户的机制可以是不同的人完全的使用同一台电脑，相应的我们从这可以学习到这种权限用户的管理机制，把他应用于我们的程序中，提高我们程序的安全性。其次是学会了 C++中相对与 C 的类的概念还有面向对象的编程方法。还有就是让自己有了模块化编程的思维。还有就是变量的命名一定要规范，否则可能会重复定义，或者就是导致混乱。体会到了数据的结构在程序中的重要性，也强化了自己的设计能力 。提高了自己的水平。

编辑器在拓展上，后续还可以添加一个拓展管理模块，一方面可以拓展其功能，二也可以使其，更加个性化。还有就是可以增加文件的加密压缩功能，这样可以根据用户把压缩后的文件，存在相应的用户名下的数据库中，这样借助服务器的网络，远程读取数据库，可以实现一个简单的云存档功能。在编辑的语言方面，后续还可以加入多语言支持，例如中文与英文的同时输入处理编辑。对于代码高亮与补全，可以增加更具用户定义的变量函数类，检测输入（int,char,string,struct,class,long,short,const,float,double）这些定义的变量，函数，类，结构体加入到可补全的数组与可高亮的数组中，使带代码的编辑更加直观。也可以增加二进制文件查看功能，通过编辑器打开二进制文件，并且进行简单的编辑。最后是编辑器的界面不美观，比如 windows 版并没有做一个窗口程序，在 windows 版上上可以把编辑器内的命令改成空间，进一步减低操作的难度。

在 html 的转换方面，可以继续增加更多的主题，使生成的网页可以有更丰富的样式，其次使完善 html 转换的拓展模块，可以增添更多的 web 应用，更加便捷的编辑，生成的我们的个人博客。还有就是增加批量转换的功能，获取一个源文件夹下的所有文件对其进行转换。

所以这次软件设计的过程让我们学习到了很多东西，是自己一次难忘的经历。

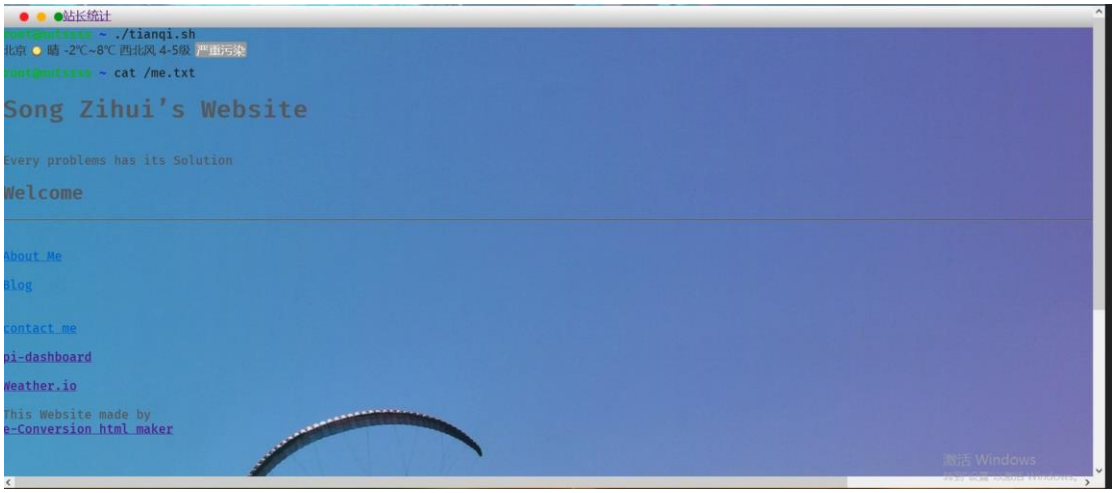
截图

Linux

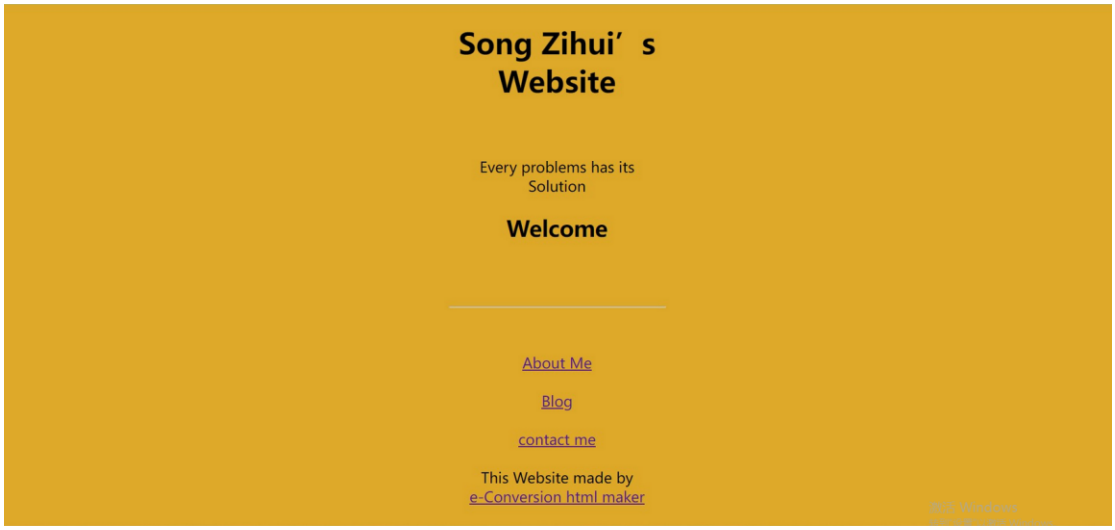
Windows

Html 主题截图：

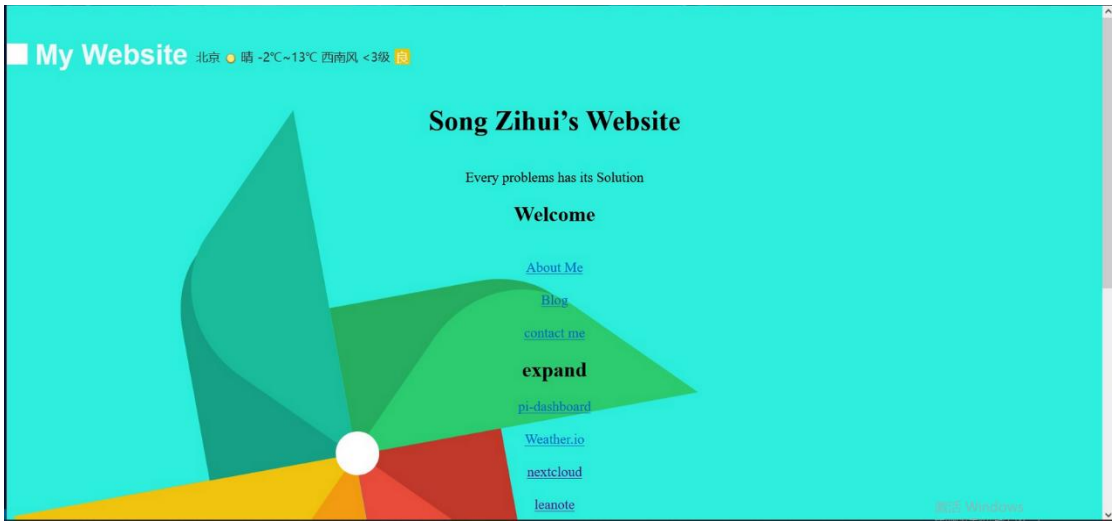
Terminal 主题



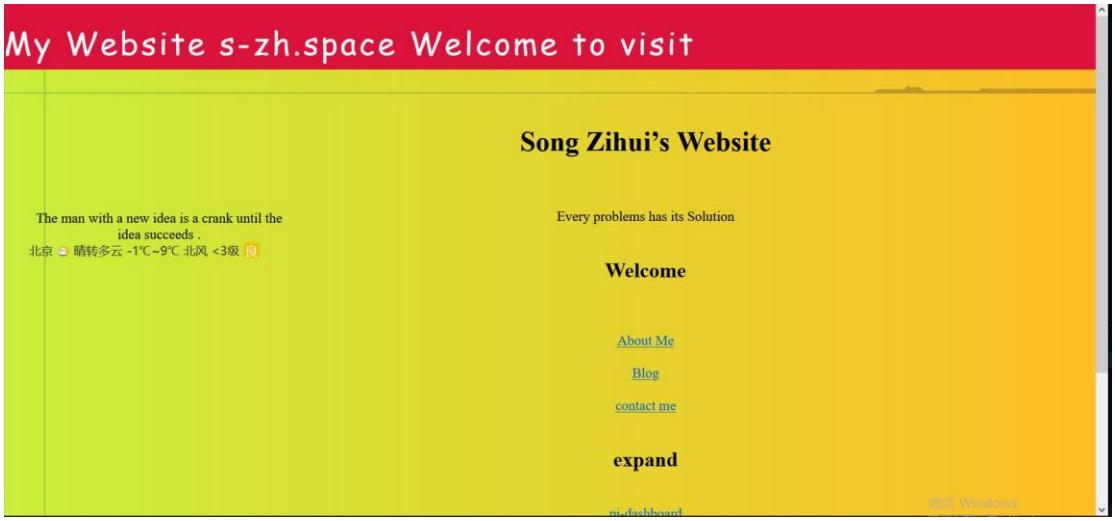
Default 主题



Bionic 主题



Saibo 主题


My Website s-zh.space Welcome to visit
Song Zihui's Website
The man with a new idea is a crank until the idea succeeds .
北京 晴转多云 -1℃~9℃ 北风 <3级
Every problems has its Solution
Welcome
About Me
Blog
contact me
expand