

浙江大学

程序设计专题

大程序报告



大程名称： 简易网页浏览器

小组成员：

1. 姓名： A同学 学号： ~ 电话： ~
2. 姓名： B同学 学号： ~ 电话： ~
3. 姓名： C同学 学号： ~ 电话： ~

指导老师： 张彤彧

2021~2022 春夏学期 2022 年 6 月 1 日

报告撰写注意事项

- 1) 图文并茂。文字通顺，语言流畅，无错别字。
- 2) 书写格式规范，排版良好，内容完整。
- 3) 存在拼凑、剽窃等现象一律认定为抄袭；0 分
- 4) 蓝色文字为说明，在最后提交的终稿版本，请删除这些文字。

目 录

1	大程序简介	5
1.1	选题背景及意义	错误！未定义书签。
1.2	目标要求	错误！未定义书签。
1.3	术语说明	7
2	需求分析	9
2.1	业务需求	10
2.2	功能需求	10
2.3	数据需求	11
2.4	性能需求	13
3	程序开发设计	14
3.1	总体架构设计	14
3.2	功能模块设计	14
3.3	数据结构设计	15
3.4	源代码文件组织设计	15
3.5	函数设计描述	18
4	部署运行和使用说明	28
4.1	编译安装	29
4.2	运行测试	30
4.3	使用操作	31
5	团队合作	33
5.1	任务分工	33
5.2	开发计划	34
5.3	编码规范	36
5.4	合作总结	37
5.5	收获感言	337
5.6	根据各自工作，说明在项目开发过程中的难点是什么？你是如何解决的？	41
6	参考文献资料	44

“简易浏览器”大程序设计

1 大程序简介

1.1 选题背景及意义

本程序基于 libgraphics，实现一个简易的网页浏览器。浏览器是用来检索、展示以及传递 Web 信息资源的应用程序，同时使用者还可以借助超链接（Hyperlinks），通过浏览器来浏览互相关联的信息。

互联网行业发展至今已有 30 余年，其出现大大缩短了全球信息传输的时间，很大程度上便捷了普通民众的日常生活和工作。在这种情况下，随着互联网的普及率不断提高，我国网民数量快速增长，浏览器成为了我们上网必备的工具。截至 2021 年 12 月，我国网民规模为 10.32 亿，较 2020 年 12 月新增网民 4296 万，互联网普及率达 73.0%，较 2020 年 12 月提升 2.6 个百分点。同时从网民年龄结构来看，我国互联网进一步向中老年群体渗透，所以设计一个简单且易上手的网页浏览器是非常必要的。

1.2 目标要求

本程序旨在设计一个用户友好型的浏览器，所设计的简易浏览器支持解析并浏览本地 html 文件，可以实现网页前进、后退、主页、刷新、输入地址（本地 html 文件）并跳转；同时该浏览器支持分两个窗口显示，左侧窗口显示 HTML 代码，右侧窗口则显示解析结果。

浏览器窗口各部分功能如下：

1、地址栏：用于输入网站的地址。

本程序实现的简易浏览器通过识别地址栏中的信息，正确连接用户要访问的内容。如要访问本地 html 文件中的“cmu.html”，只需在地址栏中输入：cmu.html，

然后按[Enter]键或单击地址栏右侧的“搜索”按钮，就会显示本地文件的html代码，之后再点击“点击解析”按钮，即可显示解析后的结果。在地址栏中还附带了常用命令的快捷按钮，如刷新、停止等。前进、后退按钮设置在地址栏右上方，在同一网页窗口浏览网页之时，用户可通过点击界面上的“后退”按钮，退回到前一页；若是已经返回前一页而想查看刚才所查看过的页面，可以通过点击“前进”按钮实现。该简易浏览器能帮助用户实现基本的网页浏览功能。

2、菜单栏：由“主题”“工具”“帮助”和“收藏夹”菜单组成。

每个菜单中包含了控制浏览器工作的相关命令选项，这些选项包含了浏览器的所有操作与设置功能。如：

“主题”菜单控制了主题的颜色，其中有5种颜色主题可选，分别为“经典原色”“清新水粉”“阳光明媚”“缤纷果盘”“摩登时代”；

“收藏夹”能够方便用户记录自己喜欢、常用的网站，将它放在收藏夹里，想用的时候点击收藏夹打开对应文件，再单击“点击解析”按钮即可显示解析结果；

“工具”这一菜单包含了“刷新”“自动解析”“退出”三个选项，同时每一个选项都有对应的快捷键（刷新—F5，自动解析—Ctrl + R，退出—Ctrl + E），用户可以通过点击“刷新”实现页面的重新加载，点击“自动解析”后，再打开任一本地html文件即可实现html的自动解析；

“帮助”这一菜单包含了“关于我们”“使用帮助”“想要更多”三个选项，用户在“关于我们”这一栏中可以看到指导老师、助教、开发人员等相关信息；在“使用帮助”这一栏中可以看到本软件的使用说明，如：“点出收藏夹中的example，你可以看到示例”、“你可以在输入框中输入网页的地址，点击搜索或者按下回车查找网页”、“按下F5可以刷新网页”、“按下‘解析’，解析html文件，或者在工具一栏中点击‘自动解析’，之后网页的解析都将自动完成”等；在“想要更多”这一栏中，我们留下了联系方式及github主页，有任何信息或者建议都欢迎联系我们。

3、页面窗口：是简易浏览器的主窗口。访问的网页内容会在此显示。若页面中有些文字或对象具有超链接属性，当鼠标指针放上去之后会变成手型，单击鼠标

左键，浏览器就会自动跳转到该链接所指向的网址；单击鼠标右键，则会弹出快捷菜单，用户可以从这些快捷菜单中选择要执行的操作命令。

1.3 术语说明

HTML:

超文本标记语言（英语：HyperText Markup Language，简称：HTML）是一种用于创建网页的标准标记语言。该标记语言包括一系列标记标签（markup tag），html 使用这些标记标签来描述网页，通过这些标签可以将网络上的文档格式统一，使分散的 Internet 资源连接为一个逻辑整体。HTML 文档（也叫做 web 页面）包含了 HTML 标签及文本内容，HTML 文本是由 HTML 命令组成的描述性文本，HTML 命令可以说明文字，图形、动画、声音、表格、链接等。

超文本是一种组织信息的方式，它通过超链接方法将文本中的文字、图表与其他信息媒体相关联，这种组织信息方式将分布在不同位置的信息资源用随机方式进行连接，为用户查找、检索信息提供方便。

标签:

超文本标记语言（HTML）标记标签。是网页浏览器识别符，也是浏览器程序默认系统许可标签。html 标签是 html 语言中最基本的单位，也是最重要的组成部分，html 标题、段落、链接、图像等均通过 html 标签来定义。html 标签有如下特点：

HTML 标签是由尖括号包围的关键词，例如 <head>；

且其通常成对出现，例如 <head> 和 </head>，标签对中的第一个标签是开始标签（也称为开放标签），第二个标签是结束标签（也称为闭合标签），在 html 中也有单独呈现的标签。一般成对出现的标签，其内容在两个标签中间；单独呈现的标签，则在标签属性中赋值。如 “<p>这是一个段落</p>” 和 “<input type="text" value="按钮" />”。

网页的内容需在<html>标签中，标题、字符格式、关键字等信息显示在<head>

标签中，而网页需展示的内容需嵌套在<body>标签中。

浏览器：

浏览器是用来检索、展示以及传递 Web 信息资源的应用程序。Web 信息资源由统一资源标识符（Uniform Resource Identifier, URI）所标记，它是一张网页、一张图片、一段视频或者任何在 Web 上所呈现的内容。使用者可以借助超链接（Hyperlinks），通过浏览器浏览互相关联的信息。

菜单栏：

菜单栏用于引导用户浏览不同网页，以及网站不同页面或部分之间的导航。一般种类丰富（取决于网站的内容和设计）。菜单栏一般由“文件”“编辑”“查看”“收藏夹”“工具”和“帮助”菜单组成。每个菜单中包含了控制浏览器工作的相关命令选项，这些选项包含了浏览器的所有操作与设置功能。

工具栏：

工具栏一般在菜单栏下面，是一些常用命令的快捷方式的集合。一般包含“前进”“后退”“刷新”“搜索”“退出”“打开起始页”“收藏夹”等按钮，能方便用户快速实现对网页的一些操作。

网页路由：

网页路由主要作用为根据实现定义的路由规则，检验 URL 请求，确定执行或者拒绝，并且按从右到左的顺序进行地址解析。

标签页：

标签页也叫做选项卡，用鼠标点击标签页可以直接切换网页。通过标签页用户可以非常方便地访问网址，不需再通过搜索栏进行搜索。

收藏页：

收藏页用于保存用户所收藏的网页，可以大大地方便用户下次对网页的阅读和访问。

2 需求分析

2.1 业务需求

此次开发的简易浏览器大程序主要解决了前端领域的初步开发，设计，解析问题。实际而言，其涉及的具体领域的仍然较广。

针对构建一个用户友好型的浏览器界面，我们借鉴了知名浏览器简洁但实用性极强的功能界面，力求用户在使用该浏览器时能够获得舒适以及愉悦的最佳体验。这在外观设计领域对我们提出较高的要求。我们团队的成员通过有限的绘图资源在图像和色彩两个方面去构建让用户满意的形象设计。清晰明确的功能键图像和简洁大方的页面布局能够让用户在第一时间了解到我们此次大程序产品的主体功能。而丰富多彩的皮肤与颜色功能则通过色彩学方面的搭配，力求用户在使用浏览器时能够获取新鲜的体验。

针对 HTML 文件的解析，我们则利用了信息与编码理论，HTML 文件相关理论，数据结构的相关知识，来达成 HTML 文件的高效读入和实时解析。具体而言，在 HTML 文件的渲染过程中，我们以点对点的方式依次对超文本标记语言进行解析与信息传递，实现了文字（色彩，大小）和超链接的初步解析，使用户能够在界面中清晰的了解到 HTML 文件的基本信息和实际的网页效果。

针对浏览器的进一步基本使用功能的拓展。我们依赖数据结构的配置与存储，实现了收藏夹，多标签页，浏览器的记录（前进后退）等诸多实用功能。基本满足用户在使用浏览器时的基本功能需求。

最后，作为面向用户对象的简易浏览器，我们给使用者制作了详细的浏览器使用说明书，介绍了浏览器的基本功能，如果用户在使用浏览器的过程中遇到了困难，能够通过阅读帮助与说明使基本的疑难点得到有效解决。

2.2 功能需求

菜单功能。菜单利用 simpleGUI 菜单工具实现基本功能，主要目的是方便用户使用并且综合与调节了其余各个功能。在浏览器基本功能的综合实现外，我们加入了帮助与说明增强了用户友好性。值得一提的是，我们原创的皮肤调节特色功能植入菜单栏中，以期更为良好的用户体验。

辅助功能。具体涵盖了工具栏，快捷键和状态栏三项功能。辅助功能是对浏览器主体功能的完善与补充，弥补了主体功能部分体验性较差的部分。

具体而言，状态栏使用户了解到了当前的操作，对皮肤切换、工具使用、网页解析、网页读取、刷新状态均有独立显示。实现了当前用户操作的可视化。

快捷键则完全是为了用户更方便的实现浏览器功能，避免了使用菜单栏等多次点击的冗余的麻烦。例如，我们通过 F5 键实现对网页的刷新，在实际浏览器的使用中刷新这样的高频次的使用功能被一个按键替代，大大减小了麻烦程度，提高了使用体验。

工具栏则是配合主题功能所进行了必要的补充，实现浏览器功能的完整性。

文件读取与网页解析。文件读取与网页解析是浏览器的核心的功能，我们依赖文件打开与数据存储实现文件信息的记录，利用 textbox 和 drawstring 实现文件信息的多级绘制。在 HTML 的解析与渲染过程中，依赖枚举数据类型实现了点对点的实时解析、渲染与绘制，做到和 HTML 文件可视化与网页内容可视化。

网页路由与多标签页。网页路由与多标签页是浏览器的基础功能，我们通过结构数组与指针实现了 HTML 文件信息的多级存放，实现了 url 接口的读入，网页信息的存储，前进与后退的操纵，多标签页的切换、增加、删除，收藏夹的跳转。并通过图形绘制的包装，保证了浏览器主体功能的实现。



2.3 数据需求

此次大程序设计的产品数据存储的难点主要体现在两个层面，一方面是 HTML 文件的读取与解析，另一方面是多标签页与前进后退功能的数据存储。

在 HTML 文件的读取与解析过程中，文件的读入采用字符串的形式存储了原始超文本标记语言的基本信息。每次在输入网页的 URL 地址时，我们会打开本地模拟的 HTML 文件并将其转化为字符串信息，并使用字符指针来记录该信

息。以上过程实现外部信息到程序内部信息的转化。

HTML 文件的解析与渲染同步进行，针对 HTML 语言的特殊性和特定的语法规则。我们团队成员利用枚举类型实现了提示标签内容的翻译与转化，将含有 `<h1>`, `<title>` 等提示信息的储存于预先定义好的 `enum` 型的数据中，实现了 HTML 文件的第一步解码。

接着利用存储的信息与其列举的功能实现进行比对，来实现对 HTML 第二层内容解码。具体实现依旧采用指针的方式对二级信息进行点对点方式的依次读取与译制，于此同时，为了减少内存的占有量，采用了一边枚举类型存储，一边绘制，最终进行释放的方式。

此外，我们额外构建了 `title` 的独立读取，存储于全局变量中，并通过外部变量的链接在多文件中施以影响。

多标签页和前进后退、超链接的实现所涉及的数据结构较为复杂。

关于网页标题和网页 URL 的信息存储的实现，同样是构建了全局类型作用域、具有外部链接的结构数组来存储相关信息。此外，前进后退的实现则依赖结构指针对该类型的结构数组额外构建一个纵向的链表。由此搭建了横向结构数组，纵向网页链表的数据信息存储方式。

纵横交错的方式也是对网页实际信息模式的一个真实还原。

多标签通过分别存储在不同的结构数组中，其中添加、删除、切换涉及了结构数组的增加、初始化（清空）、移动、序列、存储等问题。我们利用一个全局变量定位当前显示页面的网页信息，在标签页功能中对这个定位变量施加影响，进而由此变量施加与标签页结构数组，实现了对标签与网页信息显示的间接控制。

网页的多次搜索，前进后退的数据存储实现显然应当是动态的，为此构建的纵向的结构指针采用每一次网页的检索，就申请一块空间用于存储网页的历史信息。为了能够同时实现前进与后退的功能，我们团队成员采用了搭建双向链表的方式，定义了 `*next` 与 `*previous` 两个指针定位链表的上一级与下一级。通过双向链表的构建，成功实现了浏览器多级网页的前进后退与跳转。

除了上述数据结构，为了保证页面多种状态的显示以及状态栏的多种情况显示，由于上述功能都涉及了在多个文件与函数中的影响。所以本次程序设计也广泛使用了全局类型的整型数组来存储浏览器的页面显示与各种情况。

2.4 性能需求

一种数据结构的优劣是由实现其各种运算的算法具体体现的,对数据结构的分析实质上就是实现运算算法的分析,除了要验证算法是否正确解决该问题之外,还需要对算法的效率作性能评价。评价算法性能的标准主要从算法执行时间与占用内存空间两方面考虑,即算法执行所需的时间和存储空间来判断一个算法的优劣。

算法上,对问题规模与该算法在运行时所占用空间与所耗费时间给出一个数量关系的评价。数量关系评价体现在时间上,即算法经编程实现后在计算机中运行所耗费的时间。数量关系评价体现在空间上,即算法经编程实现后再计算机中运行所占用的存储量。

规模上。问题规模是问题大小的本质表示,对不同的问题其表现形式不同,算法求解问题的输入量称为问题的规模,一般用整数表示。一个图论问题的规模则是图中的顶点数或边数,对矩阵而言是其阶数,对多项式运算而言是多项式项数,对集合运算而言是集合中的元素个数,可以说算法效率应是问题规模的函数。

通过上述参照与分析,我们在程序的搭建中主要从空间和时间两个维度去优化程序设计。

空间上:

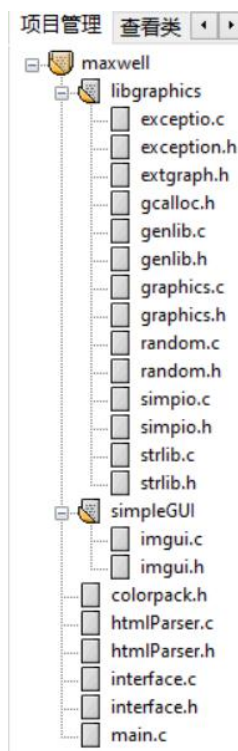
1. 搭建数据结构时采用尽可能申请少的内存空间的原则,做到高效地、高质量地利用已有的内存空间,不额外分配多余的空间。
2. 链表的搭建,做到即用即申请,用完即释放。
3. 数据的初始化与清理,在用完数组后会重新对其清理与初始化。

时间上:

本次程序设计的算法优化较少,大多功能都可以通过较为基础的算法实现,主要来说,在 HTML 解析的过程中,点对点的读入与解析与 HTML 的渲染绘制同步进行,提高了网页的解析效率。

3 程序开发设计

3.1 总体架构设计



3.2 功能模块设计

main.c 主要为注册打开 windows 状态下的窗口，并且注册注册时间响应函数。

colorpack.h 按照 RGB 定义了全部的 160 多种 html 的颜色，实现额外的颜色解析的功能。并且定义了 5 种主题色系，可以同时通过 ColorTheme 这一变量，控制浏览器的主页呈现效果。

修改了 libgraphics 中的 graphics.c 文件，添加了一系列快捷键，比如按下回车实现搜索，按下 F5 实现刷新。修改了 imgui.c 中的 StringMatch，使得它的功能更加完善准确。

htmlParser.h 包含 htmlParser.c 文件中接口函数的声明以及用于窗口参数调整的宏定义。

htmlParser.c 包含 html 文件解析、代码与网页绘制相关函数。

interface.h 是整个交互界面的头文件，定义了每个标签的名字和网址的一个

结构指针

interface.c 是交互界面，接近 800 行代码（虽然，这么多代码都放在这可能也稍微有些不太合理）是整个函数的主文件，一系列交互界面的实现全都包含在内

3.3 数据结构设计

本程序运用了多种数据结构（主要是线性的数据结构）

一般的数组：如字符串数组，存储网页名称等等，大量进行了应用

结构数组：储存网页的信息和地址等

链表：动态分配，用于实现前进、后退

较为遗憾的是没有时间，运用栈来实现对超文本标记语言的预处理和解析，或者去除重复解析同一内容而造成的不必要时间损耗。

3.4 源代码文件组织设计

<文件目录结构>

1) 文件函数结构

libgraphics 和 simpleGUI 为提供的库，在此不在赘述。

colorpack.h 中的 void ImportColorPack()（在.h 中写函数不太规范，可能不太规范，但是考虑到它的性质，还是运用了.h），里面主要通过 DefineColor 定义了全部的 160 多个的 html 中的颜色函数，颜色与颜色名称的对应，可见于 [html-RGB 对照.png](#) 或者下图中；并且定义了 void Color_Oringal(),Color_QingXing()等五个主题色。

HTML name	Hex code R G B	Decimal code R G B	HTML name	Hex code R G B	Decimal code R G B	HTML name	Hex code R G B	Decimal code R G B
Red colors			Green colors			Brown colors		
IndianRed	CD 5C 5C	205 92 92	GreenYellow	AD FF 2F	173 255 47	Cornsilk	FF F8 DC	255 248 220
LightCoral	F0 80 80	240 128 128	Chartreuse	7F FF 00	127 255 0	BlanchedAlmond	FF EB CD	255 235 205
Salmon	FA 80 72	250 128 114	LawnGreen	7C FC 00	124 252 0	Bisque	FF E4 C4	255 228 196
DarkSalmon	E9 96 7A	233 150 122	Lime	00 FF 00	0 255 0	NavajoWhite	FF DE AD	255 222 173
LightSalmon	FF A0 7A	255 160 122	LimeGreen	32 CD 32	50 205 50	Wheat	F5 DE B3	245 222 179
Red	FF 00 00	255 0 0	PaleGreen	98 FB 98	152 251 152	BurlyWood	DE B8 87	222 184 135
Crimson	DC 14 3C	220 20 60	LightGreen	90 EE 90	144 238 144	Tan	D2 B4 8C	210 180 140
FireBrick	B2 22 22	178 34 34	MediumSpringGreen	00 FA 9A	0 250 154	RosyBrown	BC 8F 8F	188 143 143
DarkRed	8B 00 00	139 0 0	SpringGreen	00 FF 7F	0 255 127	SandyBrown	F4 A4 60	244 164 96
Pink colors			MediumSeaGreen	3C B3 71	60 179 113	Goldenrod	DA A5 20	218 165 32
Pink	FF C0 CB	255 192 203	SeaGreen	2E 8B 57	46 139 87	DarkGoldenrod	B8 86 0B	184 134 11
LightPink	FF B6 C1	255 182 193	ForestGreen	22 8B 22	34 139 34	Peru	CD 85 3F	205 133 63
HotPink	FF 69 B4	255 105 180	Green	00 80 00	0 128 0	Chocolate	D2 69 1E	210 105 30
DeepPink	FF 14 93	255 20 147	DarkGreen	00 64 00	0 100 0	SaddleBrown	8B 45 13	139 69 19
MediumVioletRed	C7 15 85	199 21 133	YellowGreen	9A CD 32	154 205 50	Sienna	A0 52 2D	160 82 45
PaleVioletRed	DB 70 93	219 112 147	OliveDrab	6B 8E 23	107 142 35	Brown	A5 2A 2A	165 42 42
Orange colors			Olive	80 80 00	128 128 0	Maroon	80 00 00	128 0 0
LightSalmon	FF A0 7A	255 160 122	DarkOliveGreen	55 6B 2F	85 107 47	White colors		
Coral	FF 7F 50	255 127 80	MediumAquaMarine	66 CD AA	102 205 170	White	FF FF FF	255 255 255
Tomato	FF 63 47	255 99 71	DarkSeaGreen	8F BC 8F	143 188 143	Snow	FF FA FA	255 250 250
OrangeRed	FF 45 00	255 69 0	LightSeaGreen	20 B2 AA	32 178 170	Honeydew	F0 FF F0	240 255 240
DarkOrange	FF 8C 00	255 140 0	DarkCyan	00 8B 8B	0 139 139	MintCream	F5 FF FA	245 255 250
Orange	FF A5 00	255 165 0	Teal	00 80 80	0 128 128	Azure	F0 FF FF	240 255 255
Yellow colors			Blue/Cyan colors			AliceBlue	F0 F8 FF	240 248 255
Gold	FF D7 00	255 215 0	Aqua	00 FF FF	0 255 255	GhostWhite	F8 F8 FF	248 248 255
Yellow	FF FF 00	255 255 0	Cyan	00 FF FF	0 255 255	WhiteSmoke	F5 F5 F5	245 245 245
LightYellow	FF FF E0	255 255 224	LightCyan	E0 FF FF	224 255 255	Seashell	FF F5 EE	255 245 238
LemonChiffon	FF FA CD	255 250 205	PaleTurquoise	AF EE EE	175 238 238	Beige	F5 F5 DC	245 245 220
LightGoldenrodYellow	FA FA D2	250 250 210	AquaMarine	7F FF D4	127 255 212	OldLace	FD F5 E6	253 245 230
PapayaWhip	FF EF D5	255 239 213	Turquoise	40 E0 D0	64 224 208	FloralWhite	FF FA F0	255 250 240
Moccasin	FF E4 B5	255 228 181	MediumTurquoise	48 D1 CC	72 209 204	Ivory	FF FF F0	255 255 240
PeachPuff	FF DA B9	255 218 185	DarkTurquoise	00 CE D1	0 206 209	AntiqueWhite	FA EB D7	250 235 215
PaleGoldenrod	EE E8 AA	238 232 170	CadetBlue	5F 9E A0	95 158 160	Linen	FA F0 E6	250 240 230
Khaki	F0 E6 8C	240 230 140	SteelBlue	46 82 B4	70 130 180	LavenderBlush	FF F0 F5	255 240 245
DarkKhaki	BD B7 6B	189 183 107	LightSteelBlue	B0 C4 DE	176 196 222	MistyRose	FF E4 E1	255 228 225
Purple colors			PowderBlue	B0 E0 E6	176 224 230	Gray colors		
Lavender	E6 E6 FA	230 230 250	LightBlue	AD D8 E6	173 216 230	Gainsboro	DC DC DC	220 220 220
Thistle	D8 BF D8	216 191 216	SkyBlue	87 CE EB	135 206 235	LightGrey	D3 D3 D3	211 211 211
Plum	DD A0 DD	221 160 221	LightSkyBlue	87 CE FA	135 206 250	Silver	C0 C0 C0	192 192 192
Violet	EE 82 EE	238 130 238	DeepSkyBlue	00 BF FF	0 191 255	DarkGray	A9 A9 A9	169 169 169
Orchid	DA 70 D6	218 112 214	DodgerBlue	1E 90 FF	30 144 255	Gray	80 80 80	128 128 128
Fuchsia	FF 00 FF	255 0 255	CornflowerBlue	64 95 ED	100 149 237	DimGray	69 69 69	105 105 105
Magenta	FF 00 FF	255 0 255	RoyalBlue	41 69 E1	65 105 225	LightSlateGray	77 88 99	119 136 153
MediumOrchid	BA 55 D3	186 85 211	Blue	00 00 FF	0 0 255	SlateGray	70 80 90	112 128 144
MediumPurple	93 70 DB	147 112 219	MediumBlue	00 00 CD	0 0 205	DarkSlateGray	2F 4F 4F	47 79 79
BlueViolet	8A 2B E2	138 43 226	DarkBlue	00 00 8B	0 0 139	Black	00 00 00	0 0 0
DarkViolet	94 00 D3	148 0 211	Navy	00 00 80	0 0 128			
DarkOrchid	99 32 CC	153 50 204	MidnightBlue	19 19 70	25 25 112			
DarkMagenta	8B 00 8B	139 0 139						
Purple	80 00 80	128 0 128						
Indigo	4B 00 82	75 0 130						
DarkSlateBlue	48 3D 8B	72 61 139						
SlateBlue	6A 5A CD	106 90 205						
MediumSlateBlue	7B 68 EE	123 104 238						

html色包 (全)
简易浏览器用

main.c 主要为注册打开 windows 状态下的窗口, 并且注册注册时间响应函数 (包括字符响应函数 registerCharEvent(CharEventProcess)

键盘响应函数 registerKeyboardEvent(KeyboardEventProcess)

鼠标响应函数 registerMouseEvent(MouseEventProcess)

以及定时器响应函数 registerTimerEvent(TimerEventProcess))。

htmlParser.c 包含 html 文件解析、代码与网页绘制相关函数。具体为提供外部接口的函数: readHtml()、drawHtml()、drawCode()、freeFileStr(), 以及内部功能实现函数 Tag getTag()、parse() (具体见“函数设计描述”部分)。

htmlParser.h 包含 htmlParser.c 文件中接口函数的声明以及用于窗口参数调整的宏定义。宏定义包括: 代码区和网页区的宽度、绘图起始点 (左上角)、字体、各标签字号、行间距等。另外还包含用于记录超链接数据的结构变量和用于记录网页标题的字符数组。

htmlParser.h 包含 htmlParser.c 文件中接口函数的声明以及用于窗口参数调整的宏定义。以及一个用于把网页初始化的 initialization (int n) 函数。

htmlParser.c 包含 html 文件解析、代码与网页绘制相关函数。主要有: 函数原型: void display(), void HtmlAnalyse(), void TimerEventProcess(int timerID), CharEventProcess(char ch), void KeyboardEventProcess(int key, int event), void MouseEventProcess (int x, int y, int button, int event), void drawMenu(), void drawButtons(), void DrawLabels(), void drawTextIn(), void HelpSwitchOne(), void HelpSwitchTwo(), void HelpSwitchThree(), void DefineColorTheme(), void Wrong_Site(), void DrawHome(), void SearchWhich(int x,int y, int button, int event), void refresh(double x1, double y1,double x2, double y2), void ClickHyper(int x,int y, int button, int event), void DoSearch(), void freeList(struct label *head), void insertLabel(int labelInd, char *labelname, char *labelurl)。上述函数的具体解析见下。

2) 多文件构成机制

colorpack.h 调用在了 graphics.c 中 InitColor 函数中, 增加颜色, 并且这样避

免了再 `display` 中重复打开而导致运行速度的降低。

`html` 文件解析和绘制封装在 `htmlParser.c` 和 `htmlParser.h` 中。`H` 文件开头进行宏定义和相应判断，避免重复包含。使用宏定义实现可迁移性的绘图，包括绘图区域、字体大小、行间距等多个参数可以通过修改宏快速调整。提供了读取 `html` 文件、绘制代码区、绘制网页区、释放相关内存等函数接口，并将解析得到的网页标题和超链接地址与绘图区域进行封装，提供接口。

3.5 函数设计描述

`colorpack.h`

函数原型: `void ImportColorPack()`

功能描述: 导入颜色包

参数描述: 无

返回值描述: 无

重要局部变量定义: 无

重要局部变量用途描述: 无

函数算法描述: 定义所有的 160 多种 `html` 颜色

函数原型: `void Color_Oringal()/void Color_QingXing()/void Color_MingMei()/void Color_BinFeng()/void Color_Moden()`

功能描述: 导配置好的主题颜色，并且，同时通过 `ColorTheme` 这一变量，控制浏览器的主页呈现效果

参数描述: 无

返回值描述: 无

重要局部变量定义: 无

重要局部变量用途描述: 无

函数算法描述: 定义了五个色包，通过 `setTextBoxColors(string, string,`

string, string, 1), setButtonColors(string, string, string, string, 1),
setMenuColors(string, string, string, string, 1), SetPenColor(string) 来
实现

htmlParser.c

函数原型: `enum Tag getTag()`

功能描述: 在已经从 html 文件读取出的字符串里查找下一个标签, 同时记录标签中传递的参数。

参数描述: 无

返回值描述: 返回枚举类型 Tag, 即标签类型。

重要局部变量定义: 无

重要局部变量用途描述: 无

函数算法描述: 首先向后找到第一个'<'字符, 然后对之后的字符串进行条件判断, 以确定标签类型, 如果是标题、段落和超链接标题, 则继续读取参数(颜色、链接地址), 并存贮在相应的静态全局变量中。

函数原型: `void parse(enum Tag cur)`

功能描述: 递归进行 html 格式字符串的解析和网页绘制。

参数描述: enumTagcur: 递归过程中当前位置的处于其中的标签类型。由于网页绘制时字号、行间距、颜色等取决于当前标签类型, 因此通过该参数传递。

返回值描述: 无

重要局部变量定义: 无

重要局部变量用途描述: 无

函数算法描述: 首先根据当前标签类型决策当前函数内的行为, 如果为绘制文本, 则先设置相应的文本格式(字号、颜色、行距等), 然后对字符串向后读取到下一个标签位置, 绘制其间字符。如果当前标签为超链接, 则额外记录绘制文本的矩形区域坐标以及链接地址, 提供接口。最后调用 getTag 获取下一个标签类型, 判断继续递归或回溯。

函数原型: `int readHtml(string fname)`

功能描述: 从 html 文件中读取字符串并存储的接口函数。

参数描述: stringfname: 需要读取的文件名

返回值描述: 整形, 表示文件读取是否成功

重要局部变量定义: 无

重要局部变量用途描述: 无

函数算法描述: 打开相应文件, 先计算文件长度, 然后动态分配相应长度的字符串, 并将文件内容读取到字符串中。关闭文件。

函数原型: `void drawCode()`

功能描述: 根据读取的字符串绘制 html 代码的接口函数。

参数描述: 无

返回值描述: 无

重要局部变量定义: 无

重要局部变量用途描述: 无

函数算法描述: 以换行符为界读取字符串, 计算当前行的绘图坐标, 然后逐字节 (汉字为每两字节) 绘制, 根据 '<' '>' 字符的匹配个数计算当前字符是否需要绿色高亮。字符串和参数高亮额外特判。

函数原型: `void drawHtml()`

功能描述: 根据读取的字符串绘制网页的接口函数。

参数描述: 无

返回值描述: 无

重要局部变量定义: 无

重要局部变量用途描述: 无

函数算法描述: 调用 parser 函数递归进行 html 语法解析。

函数原型: `void gettitleR()`

功能描述: 读取*spt 字符指针中读取的 title 信息

参数描述：无

返回值描述：无

重要局部变量定义：*p

重要局部变量用途描述：*p 通过字符查找检索 title 信息

函数算法描述：利用 while 不断使指针移动，考虑各种标签的可能性，首先定位出<title>的位置，之后通过移动指针利用字符查找函数定位下一个 ‘<’ 的信息，并获取<title>与</title>之间的标题信息

interface.h

函数原型：void initialization(int n)

功能描述：初始化标签

参数描述：n 用于记录当前需要初始化的标签序号

返回值描述：无

重要局部变量定义：无

重要局部变量用途描述：无

interface.c

函数原型：void display()

功能描述：显示函数

参数描述：无

返回值描述：无

重要局部变量定义：UseHelpFlag（全局），ColorTheme（全局）

重要局部变量用途描述：ColorTheme 用于判断绘制 5 种主题颜色中的哪一种，UseHelpFlag 用于判断绘制什么东西（是帮助栏中的使用帮助，还是主页，还是无法搜索成功的报错，或者是绘制网页）

函数算法描述：绘制交互上的与网页内容无关的所有东西（包括，按钮，菜单栏，搜索框，状态栏），并且 display() 将在 TimerEventProcess(int timerID)，

void CharEventProcess(char ch), void KeyboardEventProcess(int key, int event), void MouseEventProcess(int x, int y, int button, int event)这 4 个回调函数中被反复调用, 实现在时间流动、字符变动、鼠标移动、键盘操作这 4 种情况下不断刷新网页。

函数原型: **void HtmlAnalyse()**

功能描述: 整合实现了 HTML 文件的读入, HTML 文件的解析和 HTML 与网页的绘制

参数描述: 无

返回值描述: 无

重要局部变量定义: DefineTranslationFlag (全局), TranslationFlag (全局)

重要局部变量用途描述: DefineTranslationFlag 判断是否处于自动解析模式, TranslationFlag 判断是否需要绘制解析部分

函数算法描述: 调用 readHtml(labels[nowlabel]->labelurl), drawCode()、DrawHtml 函数, 并利用 TranslationFlag 判断是否需要进行 html 的解析, 并且利用 labelstate 改变状态栏信息。

函数原型: **void TimerEventProcess(int timerID)**

功能描述: 计时器事件

参数描述: timerID 是计时器的类型

返回值描述: 无

重要局部变量定义: 无

重要局部变量用途描述: 无

函数算法描述: 通过计时器的时间, 反复调用 display() 实现刷新的功能。

函数原型: **CharEventProcess(char ch)**

功能描述: 字符事件

参数描述: 无

返回值描述：无

重要局部变量定义：无

重要局部变量用途描述：无

函数算法描述：通过字符的改变，反复调用 `display()` 实现刷新的功能。

函数原型： `void KeyboardEventProcess(int key, int event)`

功能描述：键盘事件

参数描述：key 是键盘的按键，event 是事件

返回值描述：无

重要局部变量定义：无

重要局部变量用途描述：无

函数算法描述：通过对键盘的操作，反复调用 `display()` 实现刷新的功能。

函数原型： `void MouseEventProcess (int x, int y, int button, int event)`

功能描述：鼠标事件

参数描述：(x, y) 是鼠标的位置，button 是按钮，event 是事件

返回值描述：无

重要局部变量定义：无

重要局部变量用途描述：无

函数算法描述：通过对鼠标的操作，反复调用 `display()` 实现刷新的功能。同时，`ClickHyper(x, y, button, event)` 用于判断鼠标是否位于超链接的范围内，`SearchWhich(x, y, button, event)` 用于判断在主页情况下（由于有两个搜索框，并且是独立的）通过鼠标的点击区域（需要点击文本框，才能修改文本框的内容），判断搜索哪个搜索框中的内容。

函数原型： `void drawMenu()`

功能描述：绘制菜单栏

参数描述：无

返回值描述：无

重要局部变量定义：y, h, w, wlist

重要局部变量用途描述：y:菜单位置,h:控件宽度, w: 控件宽度, wlist: 选项宽度。static char * menuColorTheme[]为主题内容的菜单, static char * menuTool[]为工具菜单的内容, static char * menuHelp[]为帮助菜单的内容, static char * menufavourite[]为收藏夹餐单的内容

函数算法描述：绘制含有主题、帮助、工具、收藏夹菜单栏。其中主题通过分支结构对画笔、按钮、文本框颜色做出修改, 达到修改主题色的目的。收藏夹通过更改 interfacehtml 配合 DoSearch () 函数实现页面的跳转。

函数原型：void drawButtons()

功能描述：按钮的绘制（包括，搜索按钮、前进按钮、后退按钮、“进行解析”按钮）

参数描述：无

返回值描述：无

重要局部变量定义：无

重要局部变量用途描述：无

函数算法描述：绘制搜索、前进、后退、解析按钮。其中搜索通过 DoSearch () 函数实现网页的跳转, 前进, 后退通过链表的移动实现网页的切换, “进行解析”按钮通过更改全局变量 TranslationFlag 对 HTML 进行解析。

函数原型：void DrawLabels()

功能描述：绘制标签页

参数描述：无

返回值描述：无

重要局部变量定义：labelnumber, nowlabel(全局)

重要局部变量用途描述：labelnumber 静态局部变量, 用于存储当前的 label 数量。nowlabel, 存储当前显示页面信息

函数算法描述：分为添加, 更改, 删除三种情况。添加通过 labelnumber++ 与 nowlabel 移动和链表实现

函数原型: `void drawTextIn()`

功能描述: 绘制标签页

参数描述: 无

返回值描述: 无

重要局部变量定义: `char interfacehtml[80]` (全局)

重要局部变量用途描述: 搜索框中的内容

函数算法描述: 绘制上方的搜索框, 并通过传递搜索框中的内容
`interfacehtml` 实现搜索的跳转

函数原型: `void HelpSwitchOne()`

功能描述: 绘制帮助栏的第 1 项——关于我们

参数描述: 无

返回值描述: 无

重要局部变量定义: 无

重要局部变量用途描述: 无

函数算法描述: 调用 `drawLable()` 绘制帮助栏的第 1 项——关于我们的
具体内容

函数原型: `void HelpSwitchTwo()`

功能描述: 绘制帮助栏的第 2 项——使用帮助

参数描述: 无

返回值描述: 无

重要局部变量定义: 无

重要局部变量用途描述: 无

函数算法描述: 调用 `drawLable()` 绘制帮助栏的第 2 项——使用帮助的
具体内容

函数原型: `void HelpSwitchThree()`

功能描述: 绘制帮助栏的第 3 项——想要更多

参数描述：无

返回值描述：无

重要局部变量定义：无

重要局部变量用途描述：无

函数算法描述：调用 drawLable()绘制帮助栏的第 3 项——想要更多的具体内容

函数原型：void DefineColorTheme()

功能描述：设置主题色

参数描述：无

返回值描述：无

重要局部变量定义：ColorTheme（全局）

重要局部变量用途描述：通过 ColorTheme，判断采用哪种主题

函数算法描述：通过 switch-case 的结构区调用定义在 colorpakc.h 中的色包

函数原型：void Wrong_Site()

功能描述：绘制搜索错误的情况

参数描述：无

返回值描述：无

重要局部变量定义：无

重要局部变量用途描述：无

函数算法描述：无算法

函数原型：void DrawHome()

功能描述：绘制主页

参数描述：无

返回值描述：无

重要局部变量定义：无

重要局部变量用途描述：无

函数算法描述：无算法

函数原型: `void SearchWhich(int x, int y, int button, int event)`

功能描述: 判断在主页时, 两个搜索框的情况下, 搜索哪个搜索框的内容

参数描述: x, y 是鼠标的位置, `button` 是按钮, `event` 是鼠标的事件

返回值描述: 无

重要局部变量定义: 无

重要局部变量用途描述: 无

函数算法描述: 在鼠标回调函数中得到鼠标的位置和事件, 通过判断鼠标点击的区域(编辑文本框需要点击)判断操作的是那个文本框, 则按下回车后对哪个文本框进行操作

函数原型: `void refresh(double x1, double y1, double x2, double y2)`

功能描述: 在 $(x1, y1)$ 左下, $(x2, y2)$ 右上的矩形区域内进行部分刷新

参数描述: $x1\ y1\ x2\ y2$ 用于储存刷新函数两个角点的位置信息来定位刷新区域

返回值描述: 无

重要局部变量定义: 无

重要局部变量用途描述: 无

函数算法描述: $(x1, y1)$ 左下, $(x2, y2)$ 右上内进行擦除, 由 `display()` 自动将会重新画上, `pause` 进行一定的延迟, 使得刷新的功能更加显著。

函数原型: `void ClickHyper(int x, int y, int button, int event)`

功能描述: 超链接跳转

参数描述: x, y 是鼠标的位置, `button` 是按钮, `event` 是鼠标的事件

返回值描述: 无

重要局部变量定义: 无

重要局部变量用途描述: 无

函数算法描述: 在鼠标回调函数中得到鼠标的位置和事件, 通过判断鼠标点击的区域是否在所画的超链接的相关的矩形框中, 如果是, 则实现对应的超链接的跳转。

函数原型: `void DoSearch()`

功能描述: 搜索

参数描述: 无

返回值描述: 无

重要局部变量定义: 无

重要局部变量用途描述: 无

函数算法描述: 首先判断输入框中的内容是否以.html 结尾, 不是则加上, 之后调用 `readHtml(interfacehtml)` 进行文件的读入, `GettitleR()` 得到标题, `insertLabel(nowlabel, titleR, interfacehtml)` 增加标签页。

函数原型: `void freeList(struct label *head)`

功能描述: 释放链表空间

参数描述: 无

返回值描述: 无

重要局部变量定义: 无

重要局部变量用途描述: 无

函数算法描述: 释放链表空间, 如果是中间部分, 则将该链表的后一个链表连到该链表的前一个链表上。

函数原型: `void insertLabel(int labelInd, char *labelname, char *labelurl)`

功能描述: 插入一个标签页

参数描述: 无

返回值描述: 无

重要局部变量定义: 无

重要局部变量用途描述: 无

函数算法描述: 通过链表, 插入一个标签页。

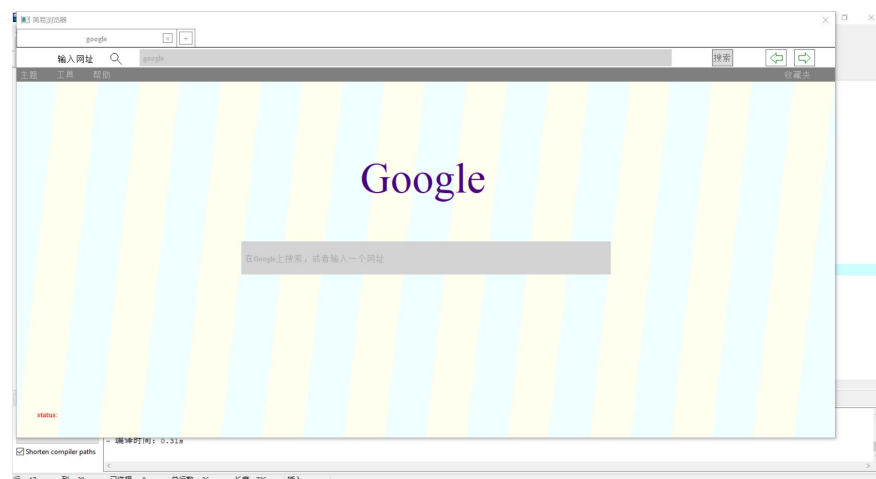
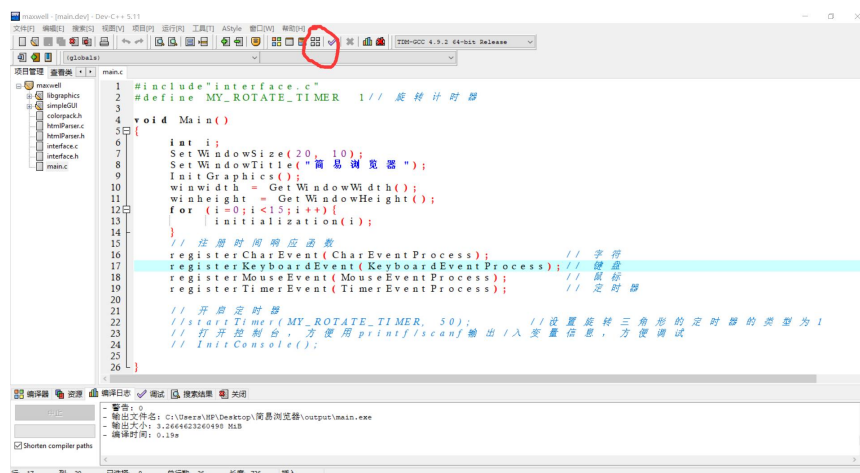
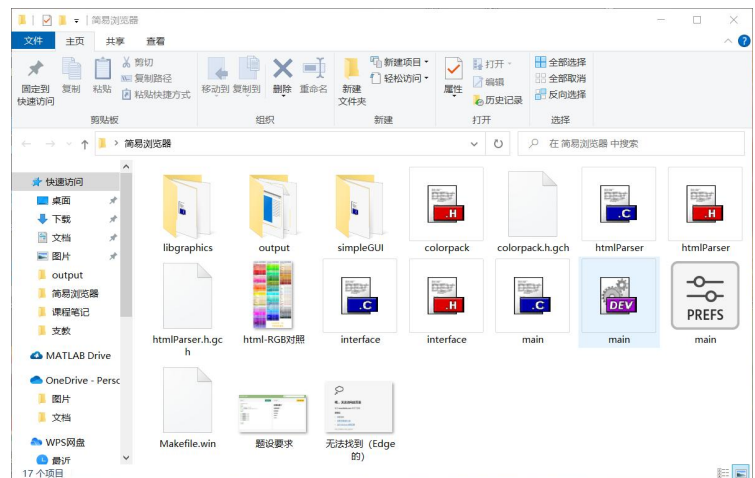
4 部署运行和使用说明

4.1 编译安装

1. 进行解压

2. 打开工程文件

3. 点击“全部编译”，再点击“运行”



4.2 运行测试

增加了读取超文本标记语言<title>的功能之后，一搜索就会出现闪退的问题，通过观察资源管理器可以发现，Dev-C++所占用的内存并没有持续上升，而是直接掉到了 0，说明应该问题出在了某个地方直接释放了内容，当查找小组同学所增加的功能的工程中，发现了 `free (fspt);` 出现了问题，首先 `readhtml` 后，在 `fspt` 不一定是有东西的，所以此时 `drawCode` 和 `drawHtml` 如果不对 `fspt` 进行判断，有可能发生错误，因此我在这部分的代码中增加了判断是否成功读取并报错的代码

```
if(fp == NULL){

    fp = fopen("wrong.html", "rb+");

    if (fp == NULL) return 0;

    UseHelpFlag=4;//跳出报错页 }
```

另一个出问题应该是因为在 `GettitleR` 中释放了 `fspt`，这里释放了 `fspt`，但你的 `HtmlAnalyze` 里直接 `drawCode` 了，所以把 `free (fspt);` 注解掉了

```
void GettitleR() {
    char *pf = fspt;
    int i = 0;
    while (*pf != '<' || *(pf + 1) != 't') pf++;
    pf = strchr(pf, '>') + 1;
    while (*pf != '<')
        titleR[i++] = *(pf++);
    titleR[i] = '\0';
    free(fspt);
}
```

以及超链接部分和 `fspt` 是分开的，但 `freeFileStr` 却是将这两者同时释放的。因此，我把释放过程拆了出来，独立成了函数

```
void freeList(struct label *head){
    struct label *p = head;

    while (p != NULL)
    {
        struct label *tmp = p;
```

```

        p = p->next;
        free(tmp);
    }
}

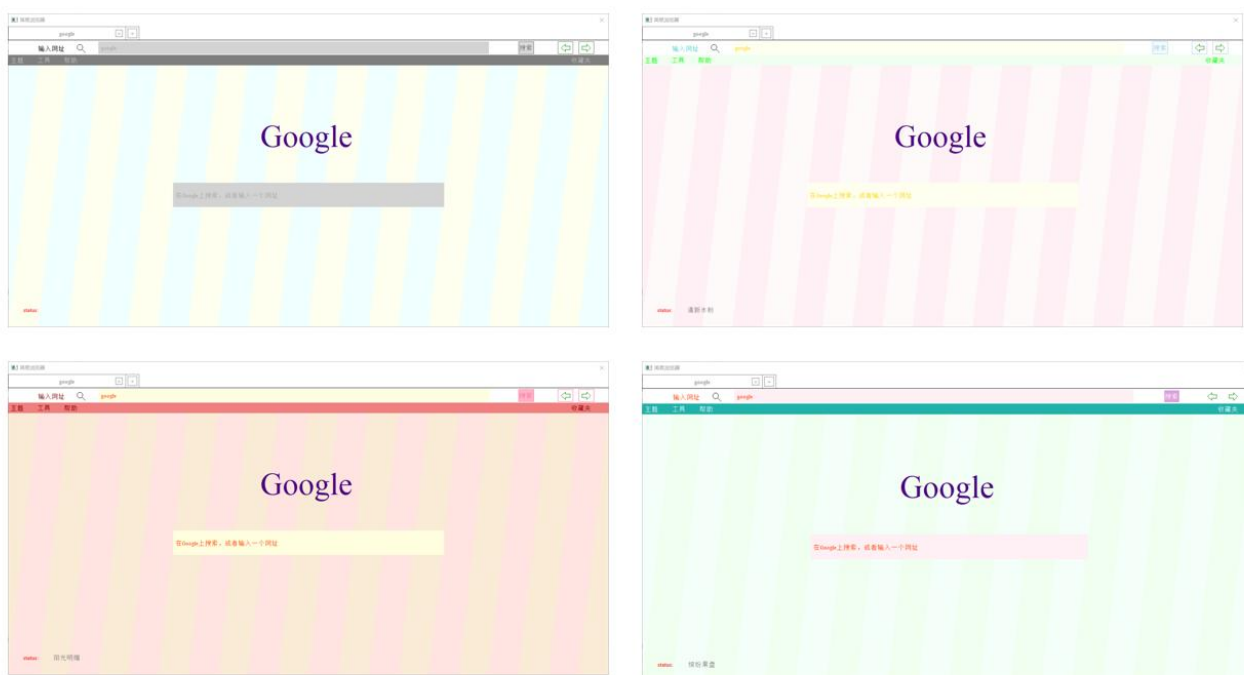
```

4.3 使用操作

菜单栏

点击收藏夹中，可以快速访问已经存储的网页

可以在主题工具栏中换取你想要的皮肤颜色，具体包括“经典原色”，“清新水粉”，“阳光明媚”，“缤纷果盘”和“摩登时代”。



网页功能

你可以在输入框中输入网页的地址，点击搜索或者按下回车搜索跳转网页。

（点击搜索框，其中的内容会被自动清空，不需要手动清除）

按下“解析”，解析 html 文件，或者在工具一栏中点击“自动解析”，之后网页的解析都将自动完成，你也可以使用 Ctrl+R 或关闭自动解析。

有关于浏览器的进一步功能，你可以通过按下“+”按钮新建



一个标签页，点击“X”按钮关闭标签页，点击标签页实现标签的跳转。

点击“<”实现当前标签的后退，点击“>”实现当前标签的前进（注意，如果可以进行前进或或后退，**按钮会被填充**）。

快捷功能

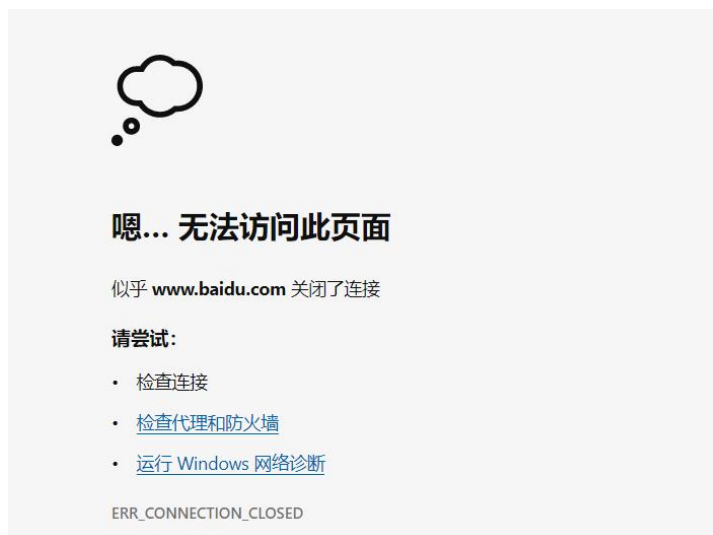
按下 F5 可以刷新网页；**回车键**实现快捷搜索。

如果想要了解我们可以访问“帮助工具栏”或者按下“Ctrl+W”；如果想要获取更多信息，请按“Ctrl+C”与我们取得联系，“Ctrl+H”获得使用帮助；你可以使用菜单栏或者按下“Ctrl+E”退出浏览器；“Ctrl+R”实现自动解析。

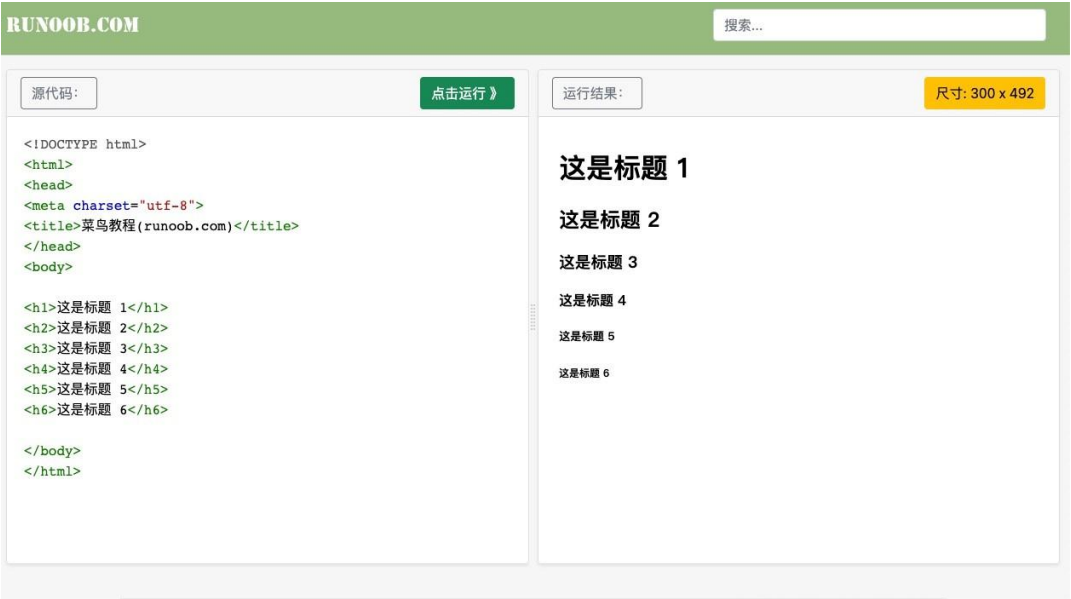
其他事项

我们实现了中英 html 文件的解析，但是汉字只支持 ANSI（GBK）的编码方式可以显示中文（暂不支持 utf-8 的中文编码），在自己创建 html 文件的时候，注意编码方式。

当你的网页无法被搜索到时，会呈现出类似如下图（Edge 浏览器中无法找到）效果：



点击“收藏夹”中的“菜鸟”，你可以看到类似下图（即题目要求）的一个效果（*题目描述：基于 libgraphics，实现一个简易的网页浏览器，可以实现网页路由（前进、后退、主页、刷新、输入地址并跳转），可以支持分两个窗口，左边显示 HTML 代码，右边显示解析结果。类似如图所示：*



5 团队合作

5.1 任务分工

A 同学:

菜单系统	网页路由的主页、刷新、输入地址+跳转 帮助的关于本软件、使用方法、更多
图标工具栏	构思和开发，菜单中常用功能/命令
快捷键	与菜单栏对应的快捷键，以及回车实现搜索
状态信息栏	初步实现在窗口底部，即时显示操作的中间状态/结果
网页路由	输入网址，并且实现网页的跳转
丰富功能	colorpack.h,实现所有 html 语言颜色的定义
	改变主题色（5 种）的功能
	主页的绘制，如果搜索失败的绘制 优化的交互：点击搜索框，搜索框内字自动消失 可以选择自动解析
其他	对队友工作的整合：

	<p>C 同学：</p> <p>修改 <code>htmlPraser.c</code>，如增加 <code>fclose</code>，<code>free</code> 和一些报错机制，使程序运行更加稳定</p> <p>超链接功能的具体实现</p> <p>B 同学：</p> <p>刷新的具体实现，以及移植的时候坐标的确定。</p> <p>重写了链表，重新实现了前进后退</p>
	制作演示视频

C 同学：

HTML 文件读取、解析和绘制。按照 HTML 语法对 `<hx>``<p>````<i>``<a>``<html>``<head>``<body>``<title>` 等标签的内容和参数进行读取，设定相应文本格式，并在指定区域进行显示。同时在另一块区域绘制代码，实现代码高亮。实现了 `style` 参数进行文本颜色的改变。实现了超链接标签，记录所有超链接的矩形区域坐标和地址，用以与其他模块对接。

B 同学：

译制了部分 `libgraphics` 的库函数；实现部分刷新的功能的原函数。设计了多标签页，实现了切换、增加、删除的标签页功能，并配合 URL 读入实现加装。实现了网页信息存储，前进，后退的功能。实现了状态栏的部分功能。实现了独立解析 HTML 的 `title` 功能。辅助 A 同学修改和整合了部分快捷键、工具栏、刷新功能。具体完善状态栏。

5.2 开发计划

按照作业要求，对具体功能的实现进行了分工，如下表所示¹

考评项		分数	说明
菜单系统		10	网页路由：前进、后退、主页、刷新、输入地址+跳转 多标签页 帮助：关于本软件、使用方法，等等
图标工具栏		5	菜单中常用功能/命令，不少于 5 个
快捷键		5	与菜单功能对应，不少于 5 个
状态信息栏		5	在窗口底部，即时显示操作的中间状态/结果
功能	网页路由	35	可以输入网址，并且实现网页的跳转 打开网页后，可以选择前进、后退、刷新
	多标签页	5	可以打开多个标签页，在多个标签页中切换
文件	文件读取	10	实现网页的解析
	多文件组织	5	系统程序必须采用多文件组织的方式
大程序报告	简介与需求分析	5	大程序简介，简要介绍程序开发的背景意义，目标要求，并进行程序功能需求分析
	程序设计	5	开发设计，包括总体架构、功能模块、数据结构、以及代码文件组织等，描述功能结构，全局、函数及重要算法说明，源程序中功能、函数、文件的组织关系
	部署和运行	5	编译安装、运行测试、用户使用手册等
	分工合作	4	成员各自承担任务、计划、编码规范、代码量、挑战点和感言；合作记录、系统开发亮点和应用知识点总结

	参考文献 资料	1	
总分	100		
额外加分项， 总分不超过 100	5	综合评价：简洁美观、易用、执行速度、代码规范等	
	5	原创或扩展功能和特色，如自动美化知识图谱排版（增加了颜色的解析，斜体，加粗的解析，主题颜色的更换，优化的交互：点击搜索框，搜索框内字自动消失，可以解析中英文 html 文件，可以解析超链接，可以选择自动解析）	
总分	10		

由 A 同学同学完成了简易浏览器框架的搭建，C 同学同学独立完成对 html 文件的解析，B 同学同学完成了部分刷新的函数，之后由 A 同学同学进行了整合到简易浏览器上，并修改了 C 同学同学的源代码，加上了报错和防崩溃的一些机制。

于此同时，C 同学同学完成了报告第一部分——“大程序简介”的撰写，B 同学同学完成了报告第二部分——“需求分析”的撰写。

之后，再完成了上述工作后，B 同学同学进行前进、后退、多标签页的开发，同时 A 同学同学进行主题色、主页、搜索错误的开发。之后进行了整合，并且修改了一些 bug。

最后，我们分别完成了各自部分程序报告的撰写，并由组长在 PTA 平台上上交了我们小组的作品。

5.3 编码规范

htmlParser.h 和 interface.c 中都有头文件保护

```

[*] interface.c  htmlParser.c  htmlParser.h
#endif
#define _HTML_PARSER_

#endif

```

采用较多的 tab 缩进来排版，使得代码的可读性更强。

```

599 void DrawHome() {
600     int ix;
601     int colorchoice = 0;
602     SetFont("Times");
603     for(ix = -1; ix <= 20; ix++) {
604         colorchoice = !colorchoice;
605         if(colorchoice) {
606             switch(ColorTheme) {
607                 case 0:
608                     SetPenColor("Azure");
609                     break;
610                 case 1:
611                     SetPenColor("LavenderBlush");
612                     break;
613                 case 2:
614                     SetPenColor("Linen");
615                     break;
616                 case 3:
617                     SetPenColor("MintCream");
618                     break;
619                 case 4:
620                     SetPenColor("WhiteSmoke");
621                     break;
622             }
623         }
624         else {
625             switch(ColorTheme) {
626                 case 0:
627                     SetPenColor("Ivory");
628                     break;
629                 case 1:
630                     SetPenColor("Snow");
631                     break;
632                 case 2:
633                     SetPenColor("MistyRose");
634                     break;
635                 case 3:
636                     SetPenColor("Honeydew");
637                     break;
638                 case 4:
639                     SetPenColor("AliceBlue");
640                     break;
641             }
642         }
643         MovePen(ix, 0);
644         StartFilledRegion(1);
645         DrawLine(1, 8.8);
646         DrawLine(1, 0);
647         DrawLine(-1, -8.8);
648         DrawLine(-1, 0);
649         EndFilledRegion();
650     }
651     SetPointSize(75);

```

采用较多的 static 类型变量（能够使用 static 尽量使用 static），使得很多变量约束在该文件内，防止出现同名变量而造成干扰

5.4 合作总结

这次程序设计专题大作业开发，在我们的代码整合过程中不断出现了闪退和奔溃的 bug，我们的解决方法是与队友交流，并自己静下心来对代码做微调，或

者单独把一段代码拎出来做调试。这一过程中交流起到了很大的作用，一些自己发现不了的 bug 通过和队友的交流分析能很快地解决，同时积极的交流也使得我们更好地理解队友写的代码，对整个大程序的实现及优化有很大益处。

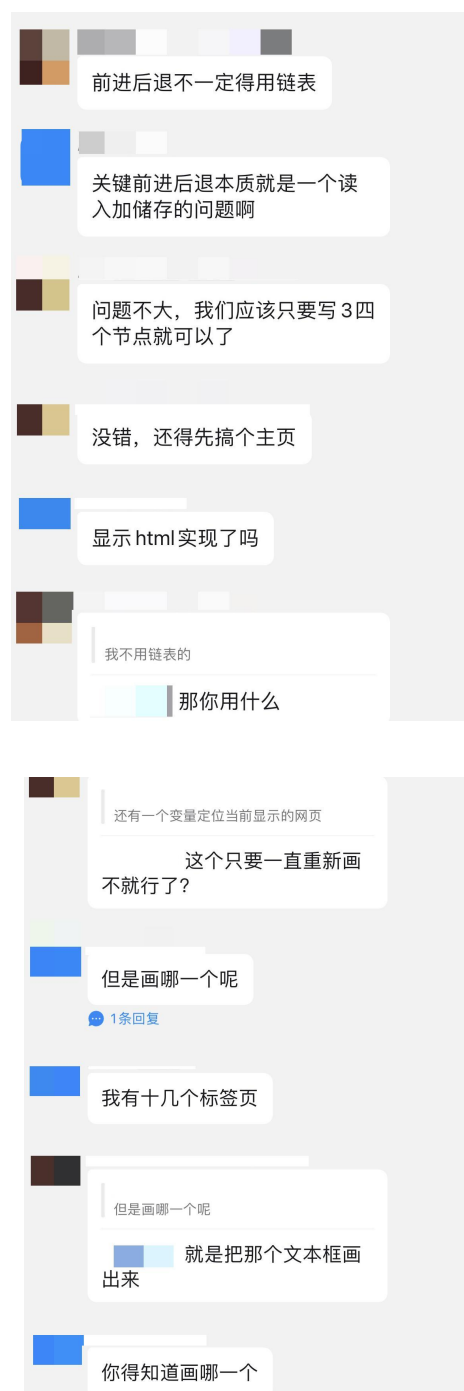
在写 html 解析这部分代码之初，一开始的困难是确定具体需要实现哪些功能，最初框架设计较为凌乱复杂，很多地方不规范，出现较多冗余和不优秀的实现方式，这给团队带来了较大的麻烦，通过与队友积极交流，解决了中文支持等相关问题，精简了实现的标签种类，解决了很多难题，同时也进一步加深了我对多文件编程相关知识的理解和图形库的使用。

同时我们也意识到，个人独立写代码时代码的清晰性与可读性不会有太大的影响，但在与他人的合作中，这一点就会变得尤为重要，应使自己的代码尽量精简且可读性强。

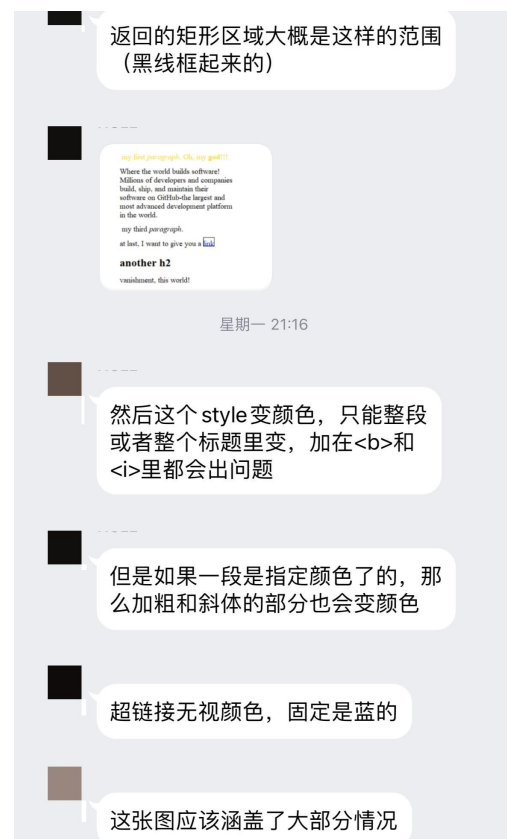
2022 年 5 月 7 日：组内成员意见非常统一，确定了本次大作业的主题为“简易浏览器”，在分工之前我们都尝试着先各自学习了 html 语言，大致了解我们这次大作业需要做些什么。之后我们在水墨云峰进行了一次线下讨论，再次明确了这次大作业我们需要且期望实现的功能，并进行了分工和初期 ddl 的确定。

2022 年 5 月 21 日：在合作开发的过程中，组员之间的沟通一直都非常及时，组长会比较及时地完成自己的那部分任务，然后与剩下两位组员讨论，也会对组员的代码提出很好的修改意见，使得我们的整个大程序更加优化。在组员写有关“前进、后退”部分的时候，一开始并没有使用链表的数据结构，最终使用了在结构数组外套链表的方式来实现“前进”“后退”。

2022 年 5 月 30 日：在写 html 解析这部分的时候，最开始读入和画是写在一个函数中的，但 display 的机制会导致如果一直在 open/close 的话会出现卡死的情况，之后我们



拆分成了两个函数，先读文件，再在 `display` 中画。在递归解析时，记录当前行距、当前坐标，并先统一调整好文本格式，再一起绘制，可以减少出错可能。除此之外，最初因为不熟悉各种中文文本编码的性质，采用 UTF-8 编码使得总是无法支持中文的读取和显示，后来与组员讨论后改用 GBK 并判断中文字符、每两个字节一起处理，解决了中文支持相关问题。



5.5 收获感言

A 同学：

这次程序设计专题大作业的开发，让我印象最深刻的是，整合过程中不断出现的闪退和奔溃的 bug——大多数是因为文件操作的不规范（比如，`fopen` 之后没有 `fclose`，分配空间之后没有及时 `free`），以及很多时候，由于封装的问题，代码移植后会发现一些变量的改变，但是很多时候却难以发现到底是哪些变量变了。

很多时候，认为技术上是可行的，实际上也是可行的，但是就是仍然会出现各种各样的错误（往往是某个小地方疏忽了 orz，代码的规范性仍然需要加强）。这让我想起了 C 小程中，也经常会遇到类似的问题，比如说，一道题目，我觉得怎么看都是对的，就是可能一个很小的地方挖了个坑，结果我就往里面跳了。

Debug 是最难受的一部分，希望以后自己的代码规范性能够加强，（并且这

次的 `interface.c` 里面函数过多，也许把他再拆成几个子功能模块会更好）从源头上减少这类问题。

C同学:

在此次程序设计专题的大作业中，我学习了多人合作编程的一些注意事项，懂得了代码规范的重要性。同时进一步加深了我对多文件编程相关知识的理解和图形库的使用。

锻炼了解决编程中遇到的各种问题的能力。在我写html解析这部分代码之初，框架设计较为凌乱复杂，很多地方不规范，出现较多冗余和不优秀的实现方式，这给我和我的队友都带来了较大的麻烦，通过与队友积极交流，解决了中文支持等相关问题，精简了实现的标签种类，解决了很多难题。

最后，在程序开发中我也体验到了需求引导的编程方式：先实现出最基本的框架，然后根据需求的丰富不断进行添加、完善和优化，在此过程中进行版本迭代。

B 同学:

此次大程序让我意识到了把代码写干净些的重要性，因为之前都是独立作业，代码的清晰性与可读性不会有太大的影响。但在与他人的合作中，这一点就会变得尤为重要，我在这方面还需要不断训练。同理，在此次程序设计中部分函数的模块化程度较弱，会对主体函数产生影响，这也是个人作业不会出现的情况。清晰化和模块化会是程序的难点与目标。

我收获了更好的读程序能力，由此了解到他人的思路与想法，而不再仅仅是书籍上的经典代码风格。我收获了更好的模块化思维，明白了程序能够完成很多事，就在于它模块化的化繁为简的方式。

5.6 根据各自工作，说明在项目开发过程中的难点是什么？ 你是如何解决的？

A 同学：

队友的代码规范方面存在一定问题，在整合时出现了较多问题。以下略举几例。

readhtml 里的获取文件长度，可以这么写

```
long begin, end;
fseek(fp, 0, SEEK_SET);
begin = ftell(fp);
fseek(fp, 0, SEEK_END);
end = ftell(fp);

len = end - begin;
```

下面的读取可以这么写

```
fseek(fp, 0, SEEK_SET);
// for(i = 0; i < len; ++i) fspt[i] = fgetc(fp);
fread(fspt, len, 1, fp);
```

这行代码，直接 $p = p + 6$

```
} else {
    for(i = 1; i <= 6; ++i) ++p;
    char *q = p;
```

少了 $col[i] = "\0"$ ；，这会导致比较时可能会出现问题。如果使用 for，逐个字节复制，那么 col 的最后一个字节不一定是 \0 当执行字符串比较的时候，有可能导致溢出

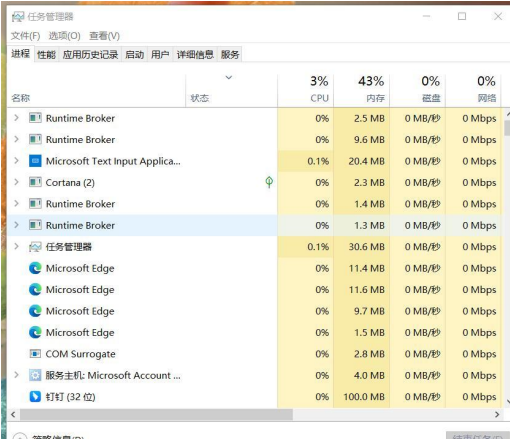
```
for(i = 1; i <= 6; ++i) ++p;
char *q = p;
while(*(p++) != ';');
col = (char*)malloc(sizeof(char)*(p-q+5));
memset(col, 0, sizeof(col));
for(i = 0; q + 1 < p; ++q, ++i) col[i] = *q;
col[i] = '\0';
}
while(*(n++) != '>');
```

在与处理标签页的队友配合时，进行标签页处理的函数，由于刚开始定义的是一个数组，后来的同学写标签页和前进后退的时候又交叉使用了数组，不仅代

码很杂很乱，而且代码规范性也有问题（释放了用于解析的指针）。因此，我不得不直接自己重新写了一个链表，来解决这些问题。

由于解析 html 和标签页的程序对文件和链表的处理存在一些问题，导致出现了很多堆区、栈区溢出的情况（经常会通过 windows 的资源管理器可以发现，Dev—C++占用的内存越来越多，并且最后被 windows 的自带机制所关闭）或者就是没有报错机制，而导致闪退（比如说你读取一个文件，不一定能够读到，但是队友的代码里却对它进行下一步的操作了（对空指针操作....）），而且查找这些错误需要一定的经验，以及需要看懂所有的代码。尤其时后期整合的过程中，随着代码量和变量的增加，很多时候动一个地方就会牵连整个程序，修改代码的过程是非常难受的。

比如说这个左图，拿到标题之后就直接把文件指针 free 了，后面对超文本标记语言处理的指针实际上是个空指针。但是由于队友电脑上有之前处理过的缓存，移植到我的电脑上，就没有缓存，就直接崩溃了，但是在队友那是显示正常的，并且他认为没有问题。因此只能由我来修改代码，之后出现的情况便是对前进后退的移植过程中，我实际上没怎么用队友写的代码（因为，太乱了，还不如自己写 orz）。



名称	状态	3% CPU	43% 内存	0% 磁盘	0% 网络
Runtime Broker		0%	2.5 MB	0 MB/秒	0 Mbps
Runtime Broker		0%	9.6 MB	0 MB/秒	0 Mbps
Microsoft Text Input Appa...		0.1%	20.4 MB	0 MB/秒	0 Mbps
Cortana (2)		0%	2.3 MB	0 MB/秒	0 Mbps
Runtime Broker		0%	1.4 MB	0 MB/秒	0 Mbps
Runtime Broker		0%	1.3 MB	0 MB/秒	0 Mbps
任务管理器		0.1%	30.6 MB	0 MB/秒	0 Mbps
Microsoft Edge		0%	11.4 MB	0 MB/秒	0 Mbps
Microsoft Edge		0%	11.6 MB	0 MB/秒	0 Mbps
Microsoft Edge		0%	9.7 MB	0 MB/秒	0 Mbps
Microsoft Edge		0%	1.5 MB	0 MB/秒	0 Mbps
COM Surrogate		0%	2.8 MB	0 MB/秒	0 Mbps
服务主机: Microsoft Account ...		0%	4.0 MB	0 MB/秒	0 Mbps
钉钉 (32 位)		0%	100.0 MB	0 MB/秒	0 Mbps

```
extern UseHelpFlag;

void GettitleR() {
    char *pf = fspt;
    int i = 0;
    while (*pf != '<' || *(pf + 1) != 't')
        pf = strchr(pf, '>') + 1;
    while (*pf != '<')
        titleR[i++] = *(pf++);
    titleR[i] = '\0';
    // free(fspt);
}

int readHtml(string fname) {
    if (!fname) return 0;
    FILE *fn = fopen(fname, "r");
```

撰写报告的过程中，有些报告内容参差不齐（代码也很多时候排版结构混乱），最后还需要我额外的时间来进行美化。

libgraphics 中也有一些问题。

比如一下这个 StringMatch(s1,s2)如果使用他注释里的写法，有可能比较错误。比如"ABC"和"ABCD"，用他的写法，比较出来就是相等的

```
static bool StringMatch(string s1, string s2)
{
    register char *cp1, *cp2;

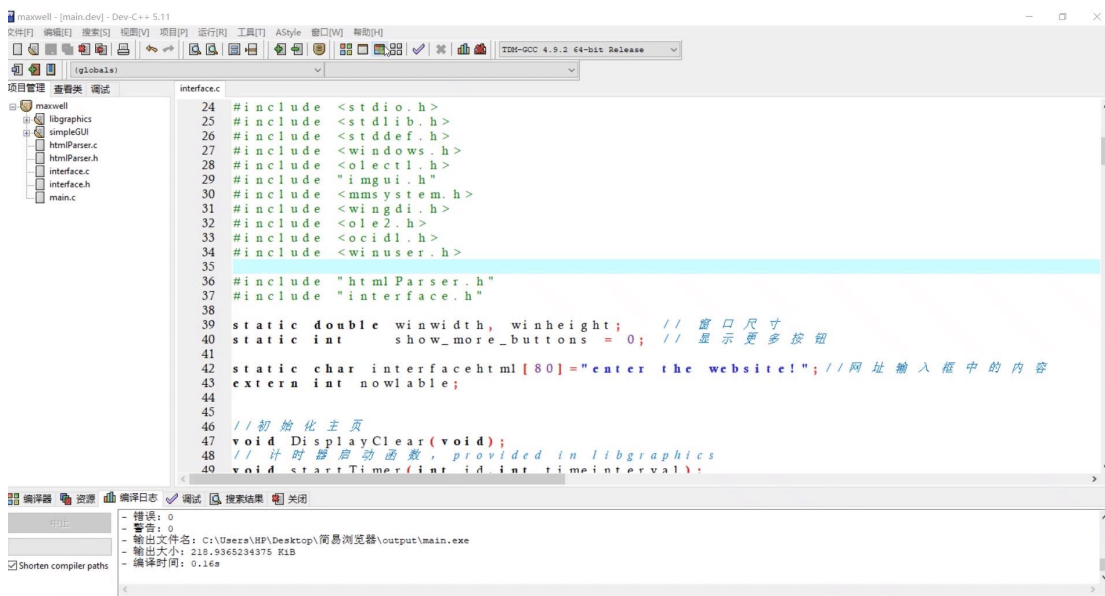
    cp1 = s1;
    cp2 = s2;

    while (*cp1 != '\0' && *cp2 != '\0')
    {
        if (tolower(*cp1) != tolower(*cp2)) return (FALSE);
        cp1++;
        cp2++;
    }
    if (*cp1 != *cp2) return (FALSE);

    return (TRUE);

    /*
    while (tolower(*cp1) == tolower(*cp2)) {
        if (*cp1 == '\0') return (TRUE);
        cp1++;
        cp2++;
    }
    */
    // return (FALSE);
}
```

出现了莫名奇妙的问题，比如下动图中的跳字现象，经检查，代码无误（把一个变量改成常量就不会跳字）。



libgraphics 功能也有待完善。比如说 libgraphics 不支持中文的输出（会显示乱码），需要自己修改源代码。我本来想要实现一个把网页加入到收藏夹，或者删除收藏夹的功能，但是二级菜单没法实现（需要较多的代码改动）。Libgraphics 和 imgui 的单位不同，并且坐标系的原点不同造成了不少的困扰，比如说下图，原本设置的左上角 222 却在右下角。

向其中添加一些较容易实现的其他功能，并逐步优化，最终成为如今版本。

B 同学：

部分刷新的实现过程中，本身想利用 timer 来控制重新时间，但实际过程中会比较复杂与困难，可移植性较差。解决方法：通读了 extgraph.c 和 libgraphic.c，一开始尝试用基于逻辑坐标的像素擦除函数直接从源头上解决问题，但是这样操纵函数的接口会较差，较为复杂，我相信会有更为简单的实现方式，终于，在第二遍阅读库的时候，我寻找到了合适的 Pause 函数，配合 seterasemode 函数实现我想要实现的功能。

多标签页的实现过程，由于和队友的 html 解析机制不匹配，我反复调整了标签页功能和 URL 的读入和 HTML 解析的顺序，但是牵一发而动全身，最终效果依然不理想。解决方法：在两个多小时的改写过程当中，我逐渐清晰了 HTML 解析的逻辑顺序，意识到不可能提前读入 title 存入我的标签页，于是自行写了一个 title 的读入。

前进后退的链表实现：由于一开始没有使用链表的数据结构，我最终采用了在结构数组外套链表的方式实现前进后退。

与队友的整合：这是最为困难的情况，程序会出现莫名其妙的崩溃甚至代码数据丢失。解决方法：与队友交流，并自己静下心来对代码做微调，或者单独把一段拎出来做调试。

6 参考文献资料

- [1] Eric S. Roberts. The Art and Science of C[M]. 北京:机械工业出版社, 1994-9-10
- [2] M. Morris Mano. Charles Kime. Tom Martin. Logic and Computer Design Fundamentals Fifth Edition Pearson ,2015

- [3] Mark Allen Weiss. 数据结构与算法分析[M]. 北京:人民邮电出版社, 2007
- [4] Stephen Prata. C++ Primer plus [M]. 北京:人民邮电出版社, 2015
- [5] Elisabeth Robson. Head First HTML and CSS [M]. 北京:中国电力出版社, 2013

7*演示视频

百度网盘

链接: <https://pan.baidu.com/s/1j8pLPIsbvoGBET8ccWfQcA>

提取码: 3790

浙大云盘

分享了一个文件(夹)给你: **【程序设计专题——简易浏览器展示.mp4】**, 点击链接查看 <https://pan.zju.edu.cn/share/81b3878e5d95502d5af01219d5> 访问密码: 3790