

江西科技师范大学

课程设计（论文）

题目（中文）： 基于 Web 客户端技术个性化 UI 设计和实现

（外文）： Web client based customized UI design and Programming

院（系）： 元宇宙产业学院

专 业： 计算机科学与技术

学生姓名： 李新雨

学 号： 20213655

指导教师： 李健宏

2024 年 6 月 18 日

目录

1. 前言	2
1.1 毕设任务分析	2
1.2 研学计划	2
1.3 研究方法	3
2. 技术总结和文献综述	4
2.1 Web 平台和客户端技术概述	4
2.2.1 历史	4
2.2.2 万维网联盟	4
2.2.3 Web 平台与 Web 编程	5
2.2 项目的增量式迭代开发模式	5
3. 内容设计概要	7
3.1 分析与设计	7
3.2 项目的实现和编程	8
3.3 项目的运行和调试	8
3.4 项目的代码提交和版本管理	9
4. 移动互联时代的 UI 开发初步——窄屏终端的响应式设计	11
4.1 分析和设计	11
4.2 项目的编程和实现	12
4.3 项目的测试和运行	14
4.4 项目的代码提交和版本管理	15
5. 应用响应式设计技术开发可适配窄屏和宽屏的 UI	16
5.1 分析和设计	16
5.2 项目的编程和实现	17
5.3 项目的测试和运行	20
5.4 项目的代码提交和版本管理	22
6. 个性化 UI 设计中对鼠标交互的设计开发	23
6.1 分析和设计	23
6.2 项目的编程和实现	24
6.3 项目的测试和运行	26
6.4 项目的代码提交和版本管理	28
7. 通用的 UI 设计，用一套代码同时为触屏和鼠标建模	28
7.1 分析和设计	28
7.2 项目的编程和实现	29
7.3 项目的测试和运行	32
7.4 项目的代码提交和版本管理	33
8. UI 的个性化键盘交互控制的设计开发	34
8.1 分析和设计	34
8.2 项目的编程和实现	35
8.3 项目测试与运行	39
8.4 项目的代码提交和版本管理	40
9. 用 git Bash 工具管理项目的代码仓库和 http 服务器	41

9.1 经典 Bash 工具介绍	41
9.2 创建一个空的远程代码仓库	41
9.3 设置本地仓库和远程代码仓库的链接	42
参考文献	46

基于 Web 客户端技术个性化 UI 设计 和实现

摘要: 随着互联网的快速发展, Web 客户端技术已成为用户与在线服务交互的主要途径。个性化 UI 设计作为提升用户体验的关键要素, 其设计与实现日益受到关注。用户体验是 UI 设计的核心, 它关注的是用户在使用产品或服务过程中的整体感受。优秀的交互设计能够提高用户的满意度和忠诚度。在 Web 客户端的 UI 设计中, 应注重交互的流畅性、响应的及时性和反馈的明确性, 以提供优质的用户体验。本文深入探讨了基于 Web 客户端技术的个性化 UI 设计与实现过程, 为行业提供有益的参考。在开发中综合应用了 html 语言进行内容建模、css 语言展开 UI 的外观设计、JavaScript 语言编程实现 UI 的交互功能。为了处理好设计和开发的关系, 项目用了工程思想管理, 使用了软件工程的增量式开发模式, 共做了 6 次项目迭代开发, 以逐步求精的方式编写了本 UI 的应用程序。为用户提供更加流畅和个性化的体验。

关键字: html 语言、UI 设计、增量模型

Abstract: With the rapid development of Internet, Web client technology has become the main way for users to interact with online services. As a key factor to improve user experience, the design and implementation of personalized UI design have attracted increasing attention. User experience is the core of UI design, it is concerned with the user's overall feeling in the process of using a product or service. Good interaction design can increase user satisfaction and loyalty. In the UI design of Web client, we should pay attention to the fluency of interaction, the timeliness of response and the clarity of feedback, so as to provide high-quality user experience. This paper deeply discusses the personalized UI design and implementation process based on Web client technology, and provides useful reference for the industry. In the development of the comprehensive application of html language for content modeling, css language to expand the appearance of UI design, JavaScript language programming to achieve the interactive function of UI. In order to deal with the relationship between design and development, the project uses engineering thought management and the incremental development mode of software engineering, and has done 6 project iterations to write the application program of this UI in a way of gradual refinement. Provide users with a smoother and more personalized experience

.**Keywords:** html language, UI design, incremental model

1. 前言

1.1 毕设任务分析

毕业设计作为大学教育中的重要环节，不仅是对学生专业知识掌握程度的检验，更是对个人综合素质和能力的全面锻炼。通过毕业设计学生能够将所学理论知识与实际应用相结合，培养解决实际问题的能力，为未来的职业生涯奠定坚实基础^[1]。

毕业设计在大学教育中占据着举足轻重的地位。它既是学生学术生涯的终点，也是未来创业道路的起点。在我的毕业设计中，涉及的主要理论课程有：面向对象的程序设计，Web 前端开发，软件工程，算法设计与分析等。通过毕业设计，我们不仅可以巩固和拓展专业知识，还能够提升实践能力、创新能力、团队协作能力。

总之，毕业设计是大学教育中的重要组成部分，也是我们综合素质和能力的一次全面检验。通过认真对待毕业设计，我们不仅能够取得良好的学术成果，还能够为自己的未来发展奠定坚实的基础。

1.2 研学计划

本次毕业设计研学计划从 2024 年 5 月开始，至 205 年 4 月结束，共计 1 个月。具体进度安排如下：

时间	任务
2024 年 5 月 5 日-5 月 20 日	定选题
2024 年 5 月 21 日-7 月 10 日	完成项目作品

2024 年 7 月 11 日-8 月 20 日	文献综述
2024 年 8 月 21 日-9 月 30 日	撰写论文框架
2024 年 10 月 1 日-11 月 30 日	撰写论文内容
2024 年 12 月 1 日-12 月 31 日	规范论文格式
2025 年 2 月 20 日-3 月 31 日	知网查重并修改

1.3 研究方法

毕设（毕业设计）的研究方法多种多样，具体选择哪种方法取决于研究的具体领域、目标和可用资源。毕设研究方法中的文献法，是一种通过搜集、鉴别、整理和分析相关文献材料，从而全面、正确地研究某一问题的方法。在毕设中运用文献法，有助于学生深入了解研究领域的背景、现状和发展趋势，为设计提供理论支撑和实践指导。

首先，需要明确毕设的研究目的和具体问题，以便有针对性地搜集和分析文献。根据研究目的和问题，编写文献综述的大纲，明确需要搜集和整理的文献范围和内容。通过多种渠道搜集相关文献，包括图书馆、档案馆、互联网等。在搜集过程中，要注意文献的来源和质量，确保文献的可靠性和有效性。对搜集到的文献进行鉴别和筛选，剔除与研究目的和问题无关的文献，保留有价值的文献。对筛选出的文献进行整理和分析，包括阅读、摘录、分类、比较和归

纳等。通过分析文献，可以了解研究领域的发展历程、现状和未来趋势，发现研究问题和不足，提出新的观点和思路。根据整理和分析的结果，撰写文献综述。文献综述应包括对研究领域的概述、文献的主要观点、研究方法和结论等方面介绍和评价。

总之，文献法是毕设中常用的一种研究方法，通过系统地搜集、鉴别、整理和分析相关文献，可以为研究提供有力的理论支撑和实践指导。

2.技术总结和文献综述

2.1 Web 平台和客户端技术概述

Web 之父 Tim Berners Lee 在发明 Web 的基本技术架构以后，就成立了 W3C 组织，该组织在 2010 年后推出的 HTML5 国际标准，结合欧洲 ECMA 组织维护的 ECMAScript 国际标准，几乎完美缔造了全球开发者实现开发平台统一的理想，直到今天，科学家与 Web 行业也还一直在致力于完善这个伟大而光荣的理想^[2]。学习 Web 标准和 Web 技术，学习编写 Web 程序和应用有关工具，最终架构一套高质量代码的跨平台运行的应用，这也是我的毕设项目应用的技术路线。

2.2.1 历史

1989 年，蒂姆·伯纳斯-李爵士发明了万维网(见最初的提案)。他创造了“万维网”这个词，并在 1990 年 10 月编写了第一个万维网服务器“httpd”和第一个客户端程序(一个浏览器和编辑器)“World Wide Web”。他编写了“超文本标记语言”(HTML)的第一个版本，这种文档格式语言具有超文本链接的功能，成为 Web 的主要发布格式。随着 Web 技术的传播，他对 url、HTTP 和 HTML 的最初规范进行了改进和讨论

2.2.2 万维网联盟

1994 年，在许多公司将越来越多的资源投入网络的敦促下，万维网联盟决定成立。蒂姆·伯纳斯-李爵士开始领导网络联盟团队的基本工作，以促进一个一致的架构，以适应快速发展的网络标准，用于构建网站、浏览器、设备，以体验网络所提供的一切。在创建万维网联盟时，蒂姆·伯纳斯-李爵士创建了一

个同行社区。Web 技术已经发展得如此之快，以至于组建一个单一的组织是至关重要的

2.2.3 Web 平台与 Web 编程

让我们先简单介绍一下 Web，也就是万维网的缩写。大多数人说“Web”而不是“World Wide Web”，我们将遵循这个惯例。网络是文档的集合，称为网页，由世界各地的计算机用户共享(大部分)。不同类型的网页做不同的事情，但至少，它们都在电脑屏幕上显示内容。所谓“内容”，我们指的是文本、图片和用户输入机制，如文本框和按钮^[3]。

Web 编程是一个很大的领域，不同类型的 Web 编程由不同的工具实现。所有的工具都使用核心语言 HTML，所以几乎所有的 web 编程书籍都在某种程度上描述了 HTML。这本教科书涵盖了 HTML5, CSS 和 JavaScript，所有的深度。这三种技术被认为是客户端 web 编程的支柱。使用客户端 web 编程，所有网页计算都在最终用户的计算机(客户端计算机)上执行^[4]。

Web 应用的程序设计体系由三大语言有机组成：HTML, CSS, JavaScript。这三大语言的组合也体现了人类社会化大生产分工的智慧，可以看作用三套相对独立体系实现了对一个信息系统的描述和控制，可以总结为：HTML 用来描述结构（Structure）、CSS 用来描述外表（presentation）、JavaScript 用来描述行为（Behavior）^[5]；这也可以用经典的 MVC 设计模式来理解 Web 平台架构的三大基石，Model 可以理解为 HTML 标记语言建模，View 可以理解为用 CSS 语言来实现外观，Controller 则可理解为用 JavaScript 结合前面二个层次，实现了在微观和功能层面的代码控制。

2.2 项目的增量式迭代开发模式

本项目作为一个本科专业学生毕业设计的软件作品，与单一用途的程序相比较为复杂，本项目所涉及的手写代码量远超过简单一二个数量级以上，从分析问题的到初步尝试写代码也不是能在几天内能落实的，可以说本项目是一个系统工程，因此需要从软件工程的管理视角来看待和规范项目的编写过程。

而本项目考虑选择的软件工程开发过程管理模式有两种经典模型：瀑布模型（The waterfall model）和增量式迭代模型(The incremental model)。而任何开

发模式则都必须同样经历四个阶段：分析（Analysis）、设计（Design）、实施（Implementation）、测试（test）。

瀑布模型需要专业团队完美的配合，从分析、设计到实施，最后到测试，任何阶段的开始必须基于上一阶段的完美结束。而这对于大多数普通开发者是不太现实的，作为小微开发者由于身兼数职，其实无法1次就能完美完成任何阶段的工作，比如在实施过程中，开发者会发现前面的设计存在问题，则必须在下一次迭代项目时改良设计。在当今开源的软件开发环境中，开发者在软件的开发中总是在不断地优化设计、重构代码，持续改进程序的功能和代码质量。因此在本项目的开发中，也采用了增量模型的开发模式^[6]。本项目中我一共做了六次项目的开发迭代，如下图2-1所示：

The incremental model

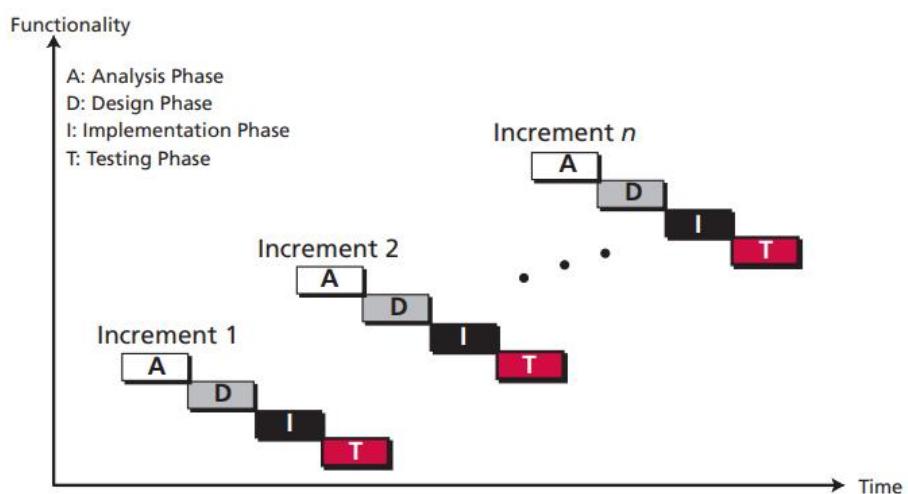


图 2-1

增量模型

在增量模型中，软件是按一系列步骤开发的。开发人员首先完成整个系统的简化版本。这个版本代表整个系统，但不包括细节。图中显示了增量模型概念。在第二个版本中，添加了更多的细节，而一些未完成，并再次测试系统。如果有问题，开发人员就会知道问题出在新功能上。在现有系统正常工作之前，它们不会添加更多的功能。这个过程一直持续到所有需要的功能都具备为止。

3. 内容设计概要

3.1 分析与设计

这一步是项目的初次开发，本项目最初使用人们习惯的“三段论”式简洁方式开展内容设计，首先用一个标题性信息展示 logo 或文字标题，吸引用户的注意力，迅速表达主题；然后展现主要区域，也就是内容区，“内容为王”是项目必须坚守的理念，也是整个 UI 应用的重点；最后则是足部的附加信息，用来显示一些用户可能关心的细节变化。如图 3-1 用例图所示：

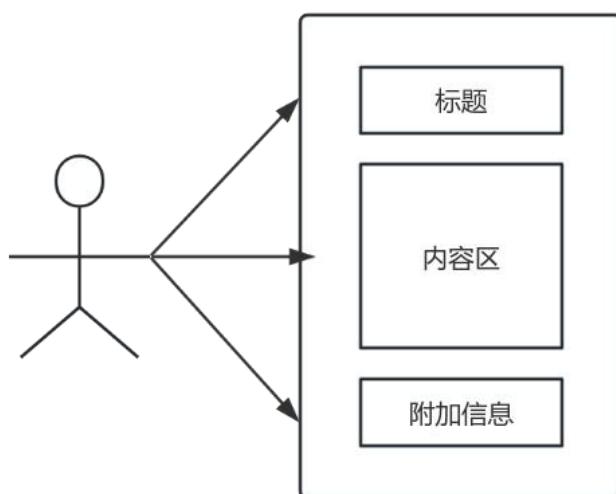
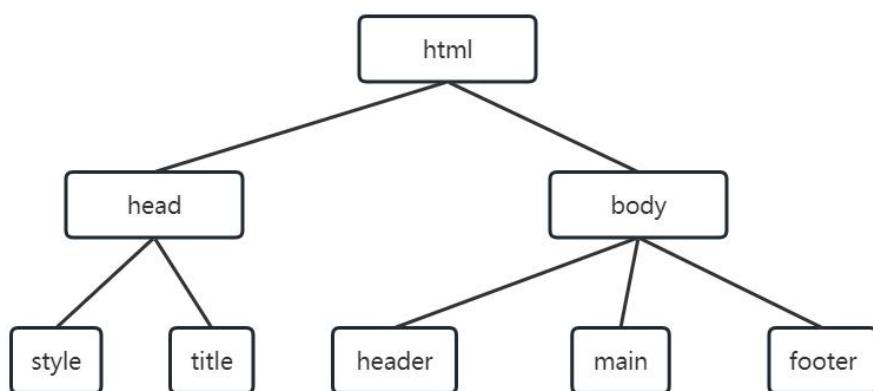


图 3-1 用例图



3-2 DOM 树状图

3.2 项目的实现和编程

(1) ① webUI1.html 的 HTML 代码编写如下图 3-2 所示

```
<body>
  <header>
    |   | 《我的毕设题目》
  </header>
  <main>
    | ‘读好书、练思维、勤编程’ @masterLixy 计算思维系列课程
  </main>
  <footer>
    CopyRight from 李新雨 江西科技师范大学 2021--2025
  </footer>
</body>
```

图 3-3 webUI1.html HTML

② webUI1.html CSS 代码编写如下图 3-3:

```
*{
  margin: 10px;
  text-align: center;
font-size:30px ;
}

header{
  border: 2px solid blue;
  height: 200px;
}

main{
  border: 2px solid blue;
  height: 400px;
}

footer{
  border: 2px solid blue;
  height: 100px;
}

a{
  display: inline-block ;
padding:10px ;
color: white;
background-color: blue;
text-decoration: none ;
}
```

图 3-4 webUI1.html CSS

3.3 项目的运行和调试

项目的运行和测试至少要通过二类终端，本文此处仅给出 PC 端用浏览器打开该项目的结果，如下图所示。由于本项目的阶段性文件已经上传 github 网站，移动端用户可以通过扫描下图的二维码，运行测试本项目的第一次开发的阶段性效果。



图 3-5PC 端运行效果图
端二维码

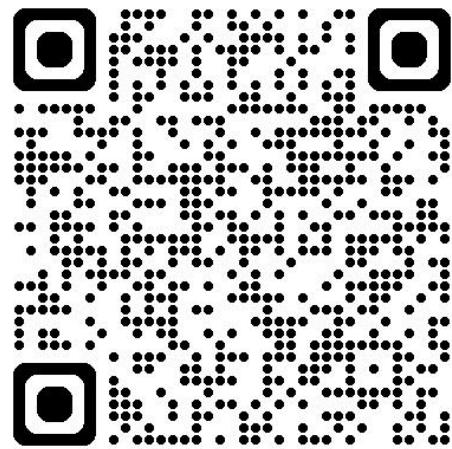


图 3-6 移动

3.4 项目的代码提交和版本管理

本项目的文件通过 git Bash 工具管理，作为项目的第一次迭代，在代码提交和版本管理环节，我们的目标是建立项目的基本文件结构，还有设置好代码仓库的基本信息：如开发者的名字和电子邮件。进入 git Bash 命令行后，按次序输入以下命令：

```
给你个机会@LAPTOP-6A0M1O15 MINGW64 ~
$ cd /d
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d
$ mkdir webUI
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d
$ cd webUI
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI
$ git init
Initialized empty Git repository in D:/webUI/.git/
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (master)
$ git config user.name 江科师大李新雨
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (master)
$ git config user.email 2079078551@qq.com
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (master)
$ touch index.html myCss.css
```

编写好 index.html 和 myCss.css 的代码，测试运行成功后，执行下面命令提交代码：

```
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (master)
$ git add index.html myCss.css
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (master)
$ git commit -m 项目第一版：“三段论”试的内容设计概要开发
```

成功提交代码后，git bash 的反馈如下所示：

```
[master (root-commit) cf89d29] 项目第一版：“三段论”试的内容设计概要开发
2 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 index.html
create mode 100644 myCss.css
```

项目代码仓库自此也开启了严肃的历史记录，我们可以输入日志命令查看，

```
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (master)
$ git log
```

Git bash 反馈代码的仓库日志如下所示：

```
commit cf89d2998cda1dee7f52af5b5192f5d5fa407043 (HEAD -> master)
Author: 江科师大李新雨 <2079078551@qq.com>
Date:   Sat Jun 8 15:04:37 2024 +0800
```

项目第一版：“三段论”试的内容设计概要开发

4. 移动互联时代的 UI 开发初步——窄屏终端的响应式设计

4.1 分析和设计

响应式设计——适应显示硬件

计算机所使用的显示硬件千差万别，显示器的大小和分辨率取决于成本。设计师们选择让网页给出一般的布局准则，并允许浏览器选择如何在给定的计算机上显示页面，而不是为每种类型的显示提供每个网页的版本。因此，一个网页不能提供很多细节。例如，网页的作者可以指定一组句子组成一个段落，但作者不能指定诸如一行的确切长度或是否缩进段落开头等细节^[7]。

允许浏览器选择显示细节会产生一个有趣的结果：当通过两个浏览器或在硬件不同的两台计算机上浏览时，网页可能会显示不同的内容。如果一个屏幕比另一个屏幕宽，则可以显示的文本行的长度或图像的大小不同。重点是：网页给出了关于期望呈现的一般指导方针；浏览器在显示页面时选择详细信息。因此，同一网页在两台不同的计算机或不同的浏览器上显示时可能会略有不同。

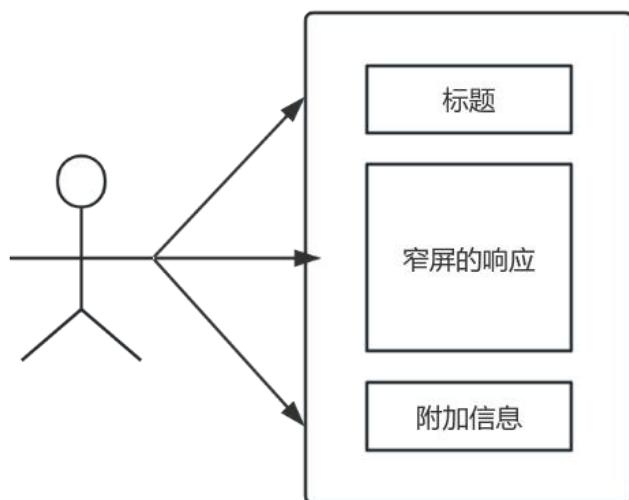


图 4-1 用例图

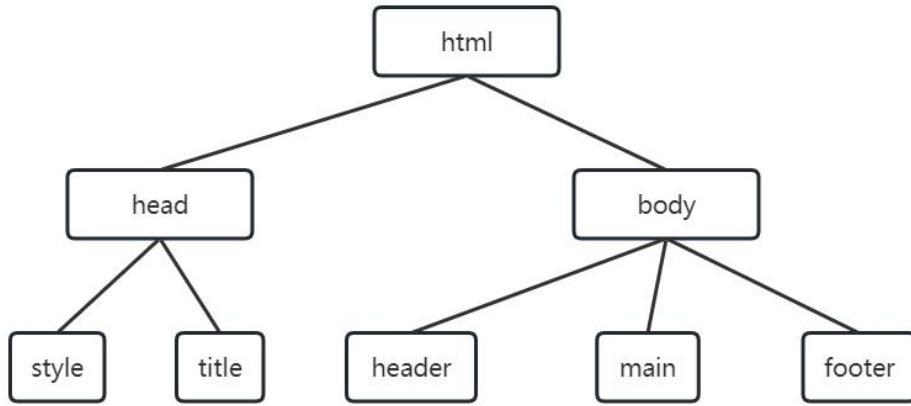


图 4-2 DOM 树状图

4.2 项目的编程和实现

用 JavaScript 开动态读取显示设备的信息，然后按设计，使用 js + css 来部署适配当前设备的显示的代码。

用汉语言来描述我们是如何实现的，与上一阶段比较，本阶段初次引入了 em 和 %，这是 CSS 语言中比较高阶的语法，可以有效地实现我们的响应式设计。如图 4-3 所示：

```
*{
    margin: 10px;
    text-align: center;
}

header{
    border: 2px solid blue;
    height: 15%;
    font-size: 1.66em;
}

main{
    border: 2px solid blue;
    height: 70%;
    font-size: 1.2em;
}

nav{
    border: 2px solid blue;
    height: 10%;
}

nav button{
    font-size: 1.1em;
}

footer{
    border: 2px solid blue;
    height: 5%;
}
```

图 4-3

用汉语言来描述我们是如何实现的：与上一阶段比较，本阶段首次使用了 JavaScript，首先创建了一个 UI 对象，然后把系统的宽度和高度记录在 UI 对象中，又计算了默认字体的大小，最后再利用动态 CSS，实现了软件界面的全屏设置。如图 4-4 所示：

```
var UI = {};
UI.appWidth = window.innerWidth > 600 ? 600 : window.innerWidth ;
UI.appHeight = window.innerHeight;
const LETTERS = 22 ;
const baseFont = UI.appWidth / LETTERS;

//通过更改body对象的字体大小，这个属性能够遗传其子子孙孙
document.body.style.fontSize = baseFont + "px";
//通过把body对象的宽度和高度设置为设备/屏幕的宽度和高度，实现全屏。
//通过CSS对子对象百分比（纵向）的配合，从而实现响应式设计的目标。
document.body.style.width = UI.appWidth - 2*baseFont + "px" ;
document.body.style.height = UI.appHeight - 4*baseFont + "px";
```

图 4-4

4.3 项目的测试和运行

编写好代码后，我们需要进行运行和测试工作，其中在 PC 端用浏览器打开项目的结果，如下图 4-4 所示。由于本项目的阶段性文件已经上传 github 网站，移动端用户可以通过扫描图 4-7 的二维码并且移动端的效果图如下图 4-6 所示，运行测试本项目的第二次开发的阶段性效果。

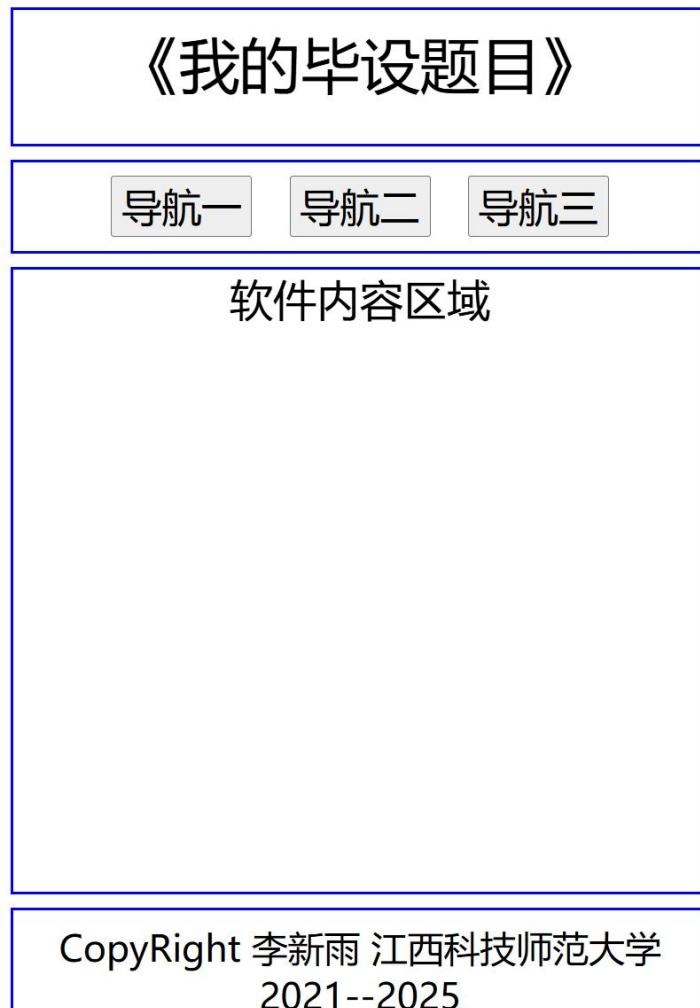


图 4-5 PC 端效果图

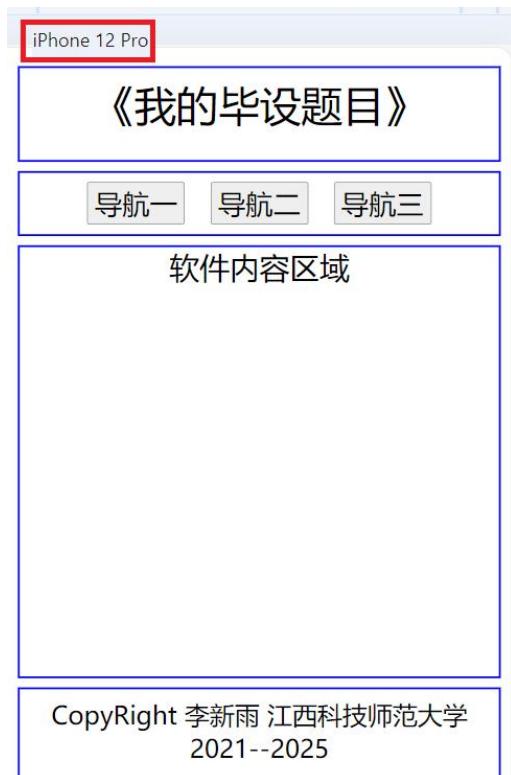


图 4-6 iPhone 12 Pro 效果图
二维码

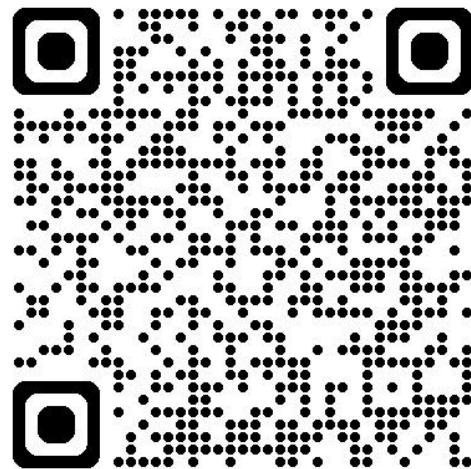


图 4-7 移动端

4.4 项目的代码提交和版本管理

编写好 webUI2.html 的代码，并且测试运行成功后，执行下图 4-8 所示命令提交代码：

```
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (main)
$ git add webUI2.html

给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (main)
$ git commit -m 项目第二版“窄屏终端的响应式设计”
```

图 4-8

代码提交以后 git bash 给出的反馈如下图 4-9 所示：

```
[main f282822] 项目第二版“窄屏终端的响应式设计”
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 webUI2.html
```

图 4-9

项目代码仓库自此也开启了严肃的历史记录，我们可以输入如下图 4-10 所示的日志命令查看

```
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (main)
$ git log
```

图 4-10

Git bash 反馈代码的仓库日志如下图 4-11 所示：

```
commit f28282za32d122156042346986cc11816c894ed0 (HEAD -> main)
Author: 江科师大李新雨 <2079078551@qq.com>
Date:   Thu Jun 13 09:51:33 2024 +0800

    项目第二版“窄屏终端的响应式设计”
```

图 4-11

5. 应用响应式设计技术开发可适配窄屏和宽屏的 UI

5.1 分析和设计

适配宽屏和窄屏的 UI 设计是一项重要的任务，它涉及到确保用户界面在不同屏幕尺寸下都能提供一致且优质的用户体验。宽屏和窄屏在尺寸和分辨率上存在差异，这直接影响了 UI 元素的布局和显示方式。响应式设计技术是一种非常有效的解决方案，能够适配不同屏幕尺寸的 UI，包括宽屏和窄屏。不同设备的屏幕尺寸和分辨率差异显著，这使得传统的固定布局 UI 设计难以满足所有用户的需求。响应式设计技术通过自动调整布局和元素尺寸，能够很好地解决这一问题。无论用户使用的是宽屏设备还是窄屏设备，他们都能获得清晰、易用且符合预期的界面。

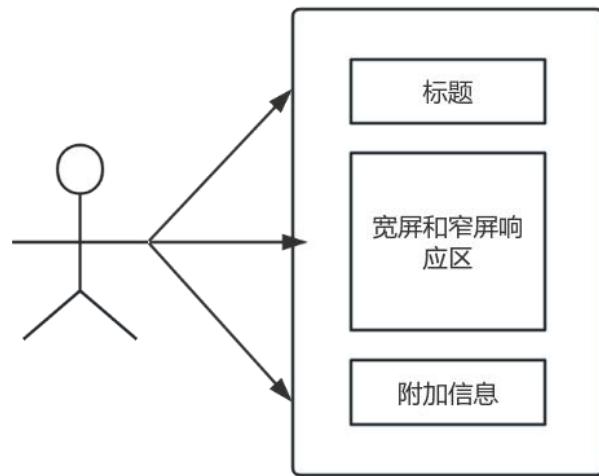


图 5-1 用例图

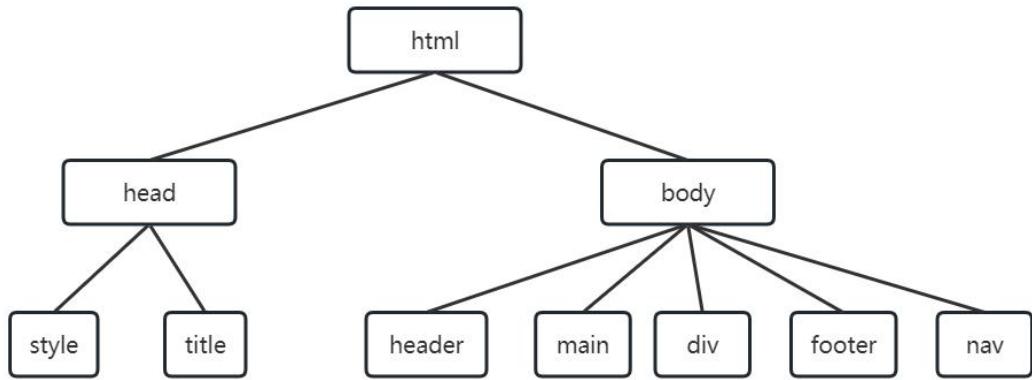


图 5-2 DOM 树状图

5.2 项目的编程和实现

为了实现宽屏和窄屏通用的响应式设计，需要根据屏幕尺寸动态调整布局、字体大小、间距等。下面是一个简单的 HTML 如下图所示

① HTML 代码编写

```

<header>
  <p id="book">
    | 《现代电影赏析》
  </p>
</header>
<nav>
  <button>向前</button>
  <button>向后</button>
  <button>暂停</button>
</nav>
<main id="main">
<div id="bookface">
  | | 书的封面图
</div>
</main>
<footer>
CopyRight from 李新雨 江西科技师范大学 2021--2025
</footer>
<div id="aid">
  <p>用户键盘响应区</p>
  <p id="keyboard"></p>
</div>

```

图 5-3 webUI3.html HTML

```

var UI = {};
if(window.innerWidth>600){
  UI.appWidth=600;
  | }else{
  UI.appWidth = window.innerWidth;
}
UI.appHeight = window.innerHeight;
let baseFont = UI.appWidth /20;
//通过改变body对象的字体大小，这个属性可以影响其后代
document.body.style.fontSize = baseFont +"px";
//通过把body的高度设置为设备屏幕的高度，从而实现纵向全屏
//通过CSS对子对象百分比（纵向）的配合，从而达到我们响应式设计的目标
document.body.style.width = UI.appWidth + "px";
document.body.style.height = UI.appHeight - 62 + "px";
if(window.innerWidth<1000){
  | $("aid").style.display='none';

  $("aid").style.width=window.innerWidth-UI.appWidth-30+'px';
  $("aid").style.height= UI.appHeight-62+'px';
}

```

图 5-4 webUI3.html HTML

```
//尝试对鼠标设计UI控制
var mouse={};
mouse.isDown= false;
mouse.x= 0;
mouse.deltaX=0;
$("bookface").addEventListener("mousedown",function(ev){
    let x= ev.pageX;
    let y= ev.pageY;
    console.log("鼠标按下了，坐标为: +"( "+x+", "+y+ ")");
    $("bookface").textContent= "鼠标按下了，坐标为: +"( "+x+", "+y+ ")";
});
$("bookface").addEventListener("mousemove",function(ev){
    let x= ev.pageX;
    let y= ev.pageY;
    console.log("鼠标正在移动，坐标为: +"( "+x+", "+y+ ")");
    $("bookface").textContent= "鼠标正在移动，坐标为: +"( "+x+", "+y+ ")";
});
$("bookface").addEventListener("mouseout",function(ev){
    //console.log(ev);
    $("bookface").textContent="鼠标已经离开";
});
```

图 5-5 webUI3.html

```
$("body").addEventListener("keypress",function(ev){
    let k=ev.key;
    let c=ev.keyCode;
    let s1="按键是: ";
    let s2="编码是: ";
    $("keyboard").textContent=s1+k+";"+s2+c;
});
```

图 5-6 webUI3.html HTML

```
function $(ele){  
    if (typeof ele !== 'string') {  
        throw("自定义的$函数参数的数据类型错误，实参必须是字符串！");  
        return  
    }  
    let dom = document.getElementById(ele) ;  
    if(dom){  
        return dom ;  
    }else{  
        dom = document.querySelector(ele) ;  
        if (dom) {  
            return dom ;  
        }else{  
            throw("执行$函数未能在页面上获取任何元素，请自查问题！");  
            return ;  
        }  
    }  
} //end of $
```

图 5-7 webUI3.html HTML

5.3 项目的测试和运行

在本次代码中，通过 CSS 对子对象纵向百分比的配合，实现了响应式设计的目标，通过不同的设备查询，可以更改不同屏幕尺寸下的布局的样式，如下图 5-10 所示，为 PC 端宽屏样式。



图 5-10 PC 端浏览图

如下图 5-11 所示为移动端样式，可以扫描图 5-12 的二维码在移动端查看本项目第三次开发效果。

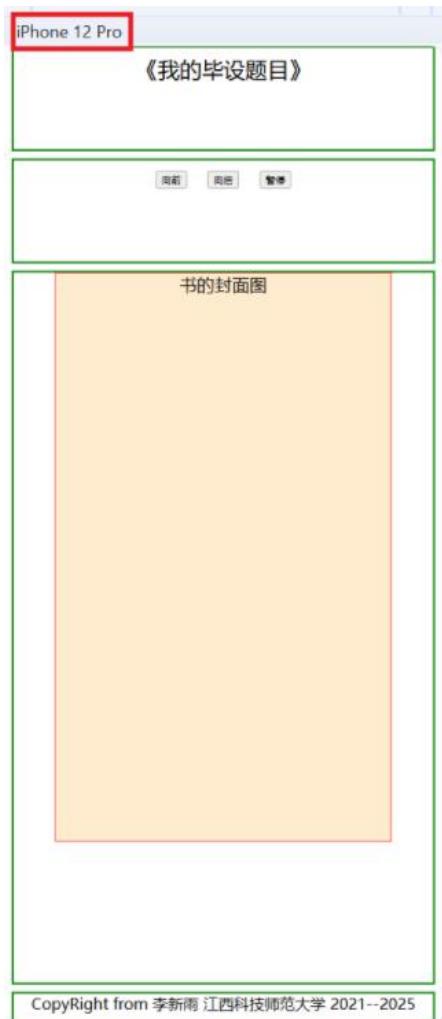


图 5-11 iPhone 12 pro 浏览图
二维码

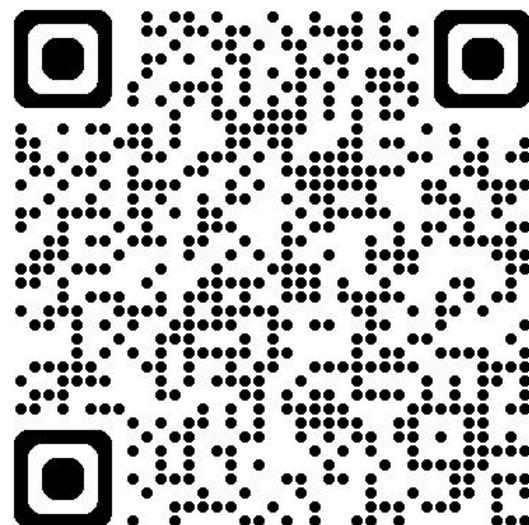


图 5-12 移动端

5.4 项目的代码提交和版本管理

编写好 webUI3.html 代码，并且运行和测试成功后，执行如下图 5-13 所示命令提交代码

```
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (main)
$ git add webUI3.html
```

```
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (main)
$ git commit -m 项目第三版“可适配宽屏和窄屏的UI”
```

图 5-13

提交代码后 git bash 给出的反馈如下图 5-14 所示

```
[main 98533ee] 项目第三版“可适配宽屏和窄屏的UI”
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 webUI3.html
```

图 5-14

项目代码仓库自此也开启了严肃的历史记录，我们可以输入如下图 5-15 所示的日志命令查看

```
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (main)
$ git log
```

图 5-15

Git bash 反馈代码的仓库日志如下图 5-16 所示：

```
commit 98533eed974f2f2ddb19a799e6e47ea8877e1610 (HEAD -> main)
Author: 江科师大李新雨 <2079078551@qq.com>
Date:   Thu Jun 13 10:07:20 2024 +0800
```

```
项目第三版“可适配宽屏和窄屏的UI”
```

图 5-16

6. 个性化 UI 设计中对鼠标交互的设计开发

6.1 分析和设计

在个性化 UI 设计中，鼠标模型的设计是一个既重要又独特的环节。鼠标作为用户与计算机系统进行交互的关键工具，其设计应充分考虑到用户的操作习惯、手感舒适度以及功能需求，同时还应与整体 UI 设计风格相协调，形成统一而个性化的用户体验。

个性化 UI 设计中的鼠标模型还应注重简洁实用性的设计理念。过于繁琐的设计不仅可能增加制造的复杂度和成本，还可能使用户感鼠标的 design 应尽可能到困惑和不适应。因此，简约，去除多余的功能和按键，保持清晰简单的操作界面。同时，根据用户的实际需求，提供一些实用的功能和操作方式，以增强用户的体验和操作效率。

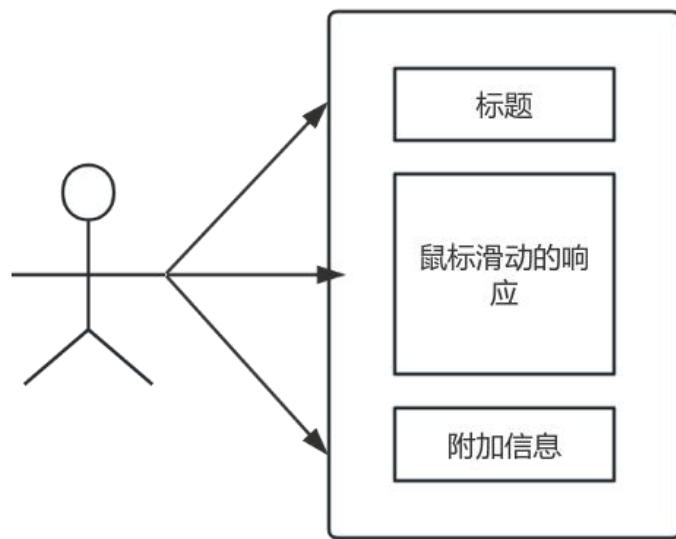


图 6-1 用例图

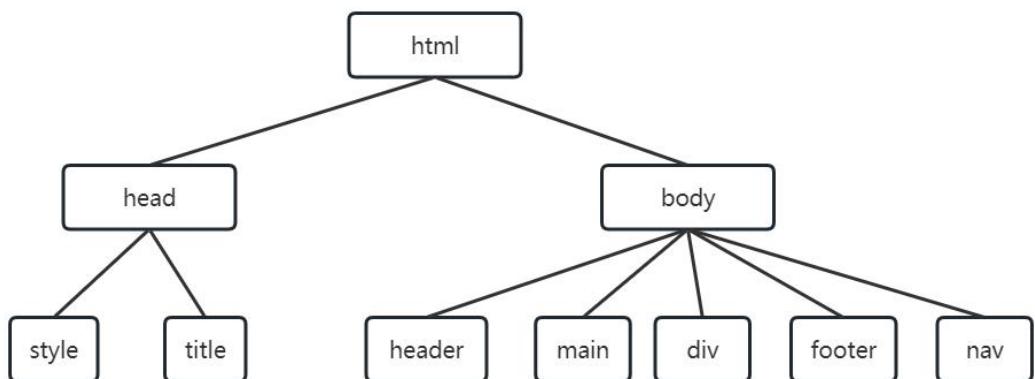


图 6-2 DOM 树状图

6.2 项目的编程和实现

① 鼠标模型的 HTML 代码编写如下图所示

```

var UI = {};
if(window.innerWidth>600){
    UI.appWidth=600;
} else{
    UI.appWidth = window.innerWidth;
}
UI.appHeight = window.innerHeight;
let baseFont = UI.appWidth /20;
//通过改变body对象的字体大小，这个属性可以影响其后代
document.body.style.fontSize = baseFont +"px";
//通过把body的高度设置为设备屏幕的高度，从而实现纵向全屏
//通过CSS对子对象百分比（纵向）的配合，从而达到我们响应式设计的目标
document.body.style.width = UI.appWidth - baseFont + "px";
document.body.style.height = UI.appHeight - baseFont*4 + "px";
if(window.innerWidth<1000){
    $("aid").style.display='none';
}
$("aid").style.width=window.innerWidth-UI.appWidth - baseFont*3 +'px';
$("aid").style.height= UI.appHeight - baseFont*3 +'px';

```

图 6-3 webUI4.html HTML

```

//尝试对鼠标设计UI控制
var mouse={};
mouse.isDown= false;
mouse.x= 0;
mouse.y= 0;
mouse.deltaX=0;
$("#bookface").addEventListener("mousedown",function(ev){
    mouse.isDown=true;
    mouse.x= ev.pageX;
    mouse.y= ev.pageY;
    console.log("mouseDown at x: "+("+"+mouse.x +"," +mouse.y +")" ) ;
    $("#bookface").textContent= "鼠标按下，坐标: "+("+"+mouse.x+"," +mouse.y+")";
});
$("#bookface").addEventListener("mouseup",function(ev){
    mouse.isDown=false;

    $("#bookface").textContent= "鼠标松开!";
    if(Math.abs(mouse.deltaX) > 100){
        $("#bookface").textContent += "，这是有效拖动！" ;
    }else{
        $("#bookface").textContent += " 本次算无效拖动！" ;
        $("#bookface").style.left = '7%' ;
    }
});


```

图 6-4 webUI4.html HTML

```

$( "bookface" ).addEventListener( "mouseout", function(ev){
    ev.preventDefault();
    mouse.isDown=false;
    $( "bookface" ).textContent= "鼠标松开!";
    if(Math.abs(mouse.deltaX) > 100){
        $( "bookface" ).textContent += " 这次是有效拖动! " ;
    }else{
        $( "bookface" ).textContent += " 本次算无效拖动! " ;
        $( "bookface" ).style.left = '7%' ;
    }
});
$( "bookface" ).addEventListener( "mousemove", function(ev){
    ev.preventDefault();
    if (mouse.isDown){
        console.log("mouse isDown and moving");
        mouse.deltaX = parseInt( ev.pageX - mouse.x );
        $( "bookface" ).textContent= "正在拖动鼠标, 距离: " + mouse.deltaX +"px 。";
        $( 'bookface' ).style.left = mouse.deltaX + 'px' ;
    }
});

```

图 6-5 webUI4.html HTML

```

function $(ele){
    if (typeof ele !== 'string'){
        throw("自定义的$函数参数的数据类型错误, 实参必须是字符串! ");
        return
    }
    let dom = document.getElementById(ele) ;
    if(dom){
        return dom ;
    }else{
        dom = document.querySelector(ele) ;
        if (dom) {
            return dom ;
        }else{
            throw("执行$函数未能在页面上获取任何元素, 请自查问题! ");
            return ;
        }
    }
} //end of $

```

图 6-6 webUI4.html HTML

6.3 项目的测试和运行

本次代码实现了鼠标交互设计，拖动鼠标，浏览器会给出响应，项目运行后可用户在 PC 端浏览器中查看效果如下图 6-7 所示



图 6-7 PC 端效果图

由于本次代码只实现了鼠标交互的设计，移动端触屏用户滑动页面时如下图 6-11 所示项目无法做出响应。

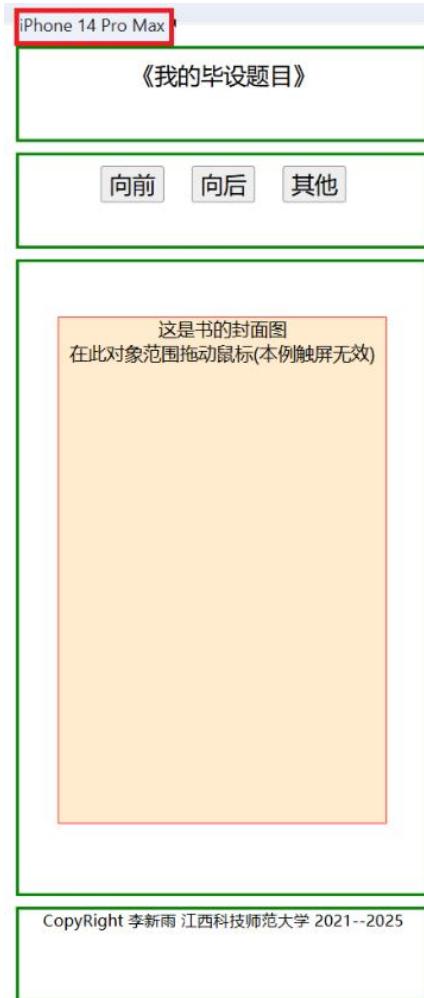


图 6-8 iPhone 14 Pro Max 效果图

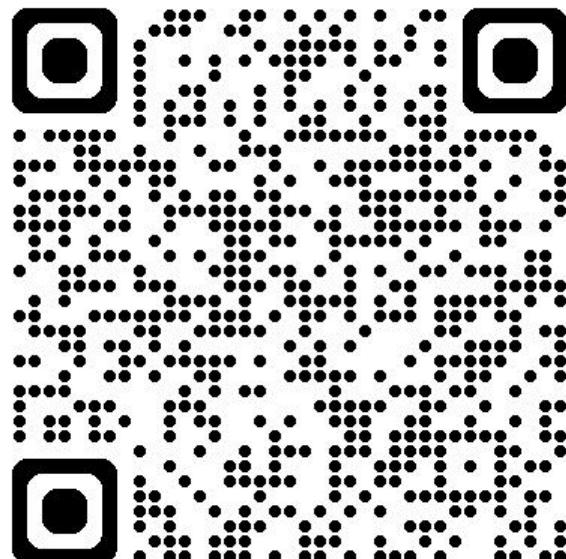


图 6-8 移动端二维码

6.4 项目的代码提交和版本管理

编写好 webUI4.html 代码，并且运行和测试成功后，执行如下图 6-9 所示命令提交代码

```
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (main)
$ git add webUI4.html

给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (main)
$ git commit -m 项目第四版“个性化UI设计中对鼠标交互的设计开发”
```

图 6-9

提交代码后 git bash 给出的反馈如下图 6-10 所示

```
[main 25ce03a] 项目第四版“个性化UI设计中对鼠标交互的设计开发”
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 webUI4.html
```

图 6-10

项目代码仓库自此也开启了严肃的历史记录，我们可以输入如下图 6-11 所示的日志命令查看

```
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (main)
$ git log
```

图 6-11

Git bash 反馈代码的仓库日志如下图 6-12 所示：

```
commit 25ce03a1ee191cd5caed5ac546acc49f381ead9e (HEAD -> main)
Author: 江科师大李新雨 <2079078551@qq.com>
Date:   Thu Jun 13 10:22:03 2024 +0800

 项目第四版“个性化UI设计中对鼠标交互的设计开发”
```

图 6-12

7. 通用的 UI 设计，用一套代码同时为触屏和鼠标建模

7.1 分析和设计

在现代用户界面设计中，实现触屏和鼠标两种输入方式的兼容是一个重要的挑战。为了实现这一目标，设计师和开发者需要构建一种通用的 UI 设计，使其既能在触屏设备上提供直观和自然的交互体验，又能在鼠标操作的桌面环境中实现精确和高效的交互。使用响应式布局和流式网格，使界面元素能够自适应不同屏幕尺寸和分辨率。这样，无论是触屏设备还是桌面显示器，都能呈现

出合适的界面布局。事件监听与处理：在代码中同时监听触屏和鼠标事件，如点击、滑动、拖拽等。根据事件类型和设备类型，执行相应的交互逻辑

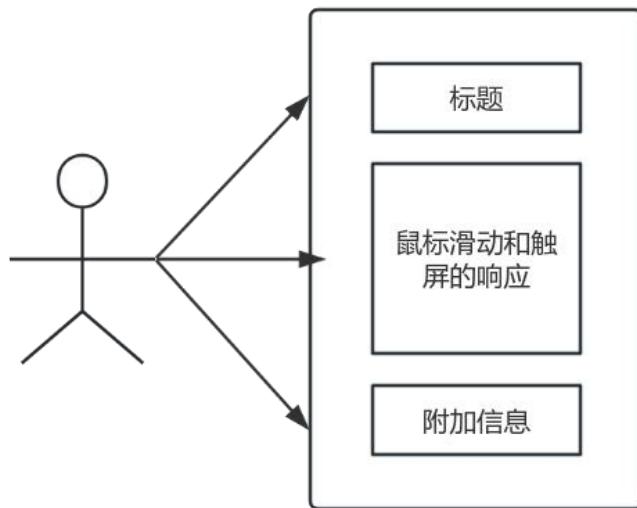


图 7-1 用例图

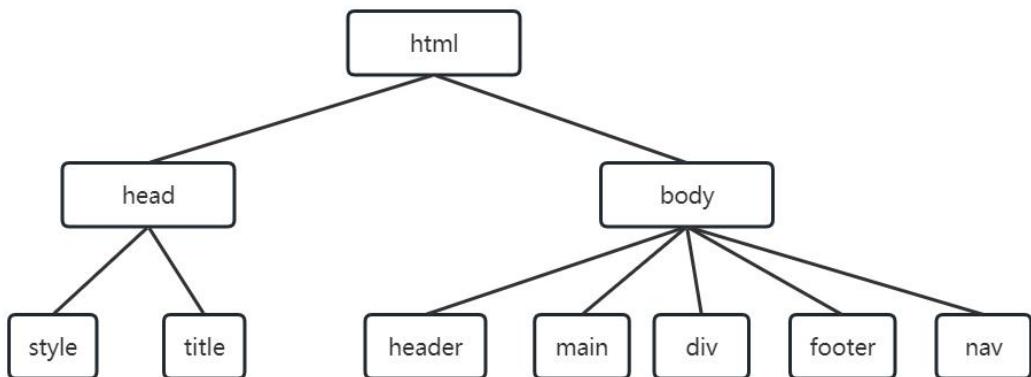


图 7-2 DOM 树状图

7.2 项目的编程和实现

要为触屏和鼠标同时建模，我们需要编写代码来检测和处理这两种输入设备的事件。下面是一个使用 CSS 和 HTML 的基本示例，它演示了如何为 UI 元素添加触屏和鼠标事件监听器，并处理这些事件。

① webUI5.html 的 HTML 代码编写如下图

```

var UI = {};
if(window.innerWidth>600){
    UI.appWidth=600;
} else{
    UI.appWidth = window.innerWidth;
}
UI.appHeight = window.innerHeight;
let baseFont = UI.appWidth /20;
//通过改变body对象的字体大小，这个属性可以影响其后代
document.body.style.fontSize = baseFont +"px";
//通过把body的高度设置为设备屏幕的高度，从而实现纵向全屏
//通过CSS对子对象百分比（纵向）的配合，从而达到我们响应式设计的目标
document.body.style.width = UI.appWidth - baseFont + "px";
document.body.style.height = UI.appHeight - baseFont*4 + "px";
if(window.innerWidth<1000){
    $("aid").style.display='none';
}
$("aid").style.width=window.innerWidth-UI.appWidth - baseFont*3 +'px';
$("aid").style.height= UI.appHeight - baseFont*3 +'px';

```

图 7-3 webUI5.html HTML

```

//尝试对鼠标和触屏设计一套代码实现UI控制
var Pointer = {};
Pointer.isDown= false;
Pointer.x = 0;
Pointer.deltaX =0;
{ //Code Block begin
let handleBegin = function(ev){
    Pointer.isDown=true;
    if(ev.touches){console.log("touches1"+ev.touches);
        Pointer.x = ev.touches[0].pageX ;
        Pointer.y = ev.touches[0].pageY ;
        console.log("Touch begin : "+"("+Pointer.x +"," +Pointer.y +")" );
        $("bookface").textContent= "触屏事件开始，坐标: "+"("+Pointer.x+","+Pointer.y+")";
    }else{
        Pointer.x= ev.pageX;
        Pointer.y= ev.pageY;
        console.log("PointerDown at x: "+"("+Pointer.x +"," +Pointer.y +")" );
        $("bookface").textContent= "鼠标按下，坐标: "+"("+Pointer.x+","+Pointer.y+")";
    }
};
let handleEnd = function(ev){
    Pointer.isDown=false;
    ev.preventDefault()
}

```

图 7-4 webUI5.html HTML

```

//console.log(ev.touches)
if(ev.touches){
    $("bookface").textContent= "触屏事件结束!";
    if(Math.abs(Pointer.deltaX) > 100){
        $("bookface").textContent += "，这是有效触屏滑动！" ;
    }else{
        $("bookface").textContent += " 本次算无效触屏滑动！" ;
        $("bookface").style.left = '7%' ;
    }
}else{
    $("bookface").textContent= "鼠标松开!";
    if(Math.abs(Pointer.deltaX) > 100){
        $("bookface").textContent += "，这是有效拖动！" ;
    }else{
        $("bookface").textContent += " 本次算无效拖动！" ;
        $("bookface").style.left = '7%' ;
    }
}
};


```

图 7-5 webUI5.html HTML

```

let handleMoving = function(ev){
    ev.preventDefault();
    if (ev.touches){
        if (Pointer.isDown){
            console.log("Touch is moving");
            Pointer.deltaX = parseInt( ev.touches[0].pageX - Pointer.x );
            $("bookface").textContent= "正在滑动触屏，滑动距离: " + Pointer.deltaX +"px 。";
            $('bookface').style.left =  Pointer.deltaX + 'px' ;
        }
    }else{
        if (Pointer.isDown){
            console.log("Pointer isDown and moving");
            Pointer.deltaX = parseInt( ev.pageX - Pointer.x );
            $("bookface").textContent= "正在拖动鼠标，距离: " + Pointer.deltaX +"px 。";
            $('bookface').style.left =  Pointer.deltaX + 'px' ;
        }
    }
};

$("bookface").addEventListener("mousedown",handleBegin );
$("bookface").addEventListener("touchstart",handleBegin );
$("bookface").addEventListener("mouseup", handleEnd );
$("bookface").addEventListener("touchend",handleEnd );
$("bookface").addEventListener("mouseout", handleEnd );
$("bookface").addEventListener("mousemove", handleMoving);
$("bookface").addEventListener("touchmove", handleMoving);
$("body").addEventListener("keypress", function(ev){
    $("aid").textContent += ev.key ;
});
} //Code Block end

```

图 7-6 webUI5.html HTML

```

function $(ele){
    if (typeof ele !== 'string'){
        throw("自定义的$函数参数的数据类型错误，实参必须是字符串！");
        return
    }
    let dom = document.getElementById(ele) ;
    if(dom){
        return dom ;
    }else{
        dom = document.querySelector(ele) ;
        if (dom) {
            return dom ;
        }else{
            throw("执行$函数未能在页面上获取任何元素，请自查问题！");
            return ;
        }
    }
} //end of $

```

图 7-7 webUI5.html HTML

7.3 项目的测试和运行

本次代码在前一次的基础上，再次实现了浏览器中鼠标拖动效果，与此同时也实现了移动端触屏效果，其中 PC 端浏览器的效果图如下图 7-8 所示



图 7-8 PC 端效果图

移动端的触屏效果图如下图 7-9 所示



图 7-9 iPhone 14 Pro Max 效果图
移动端用户可以扫描下图 7-10 的二维码查看效果：

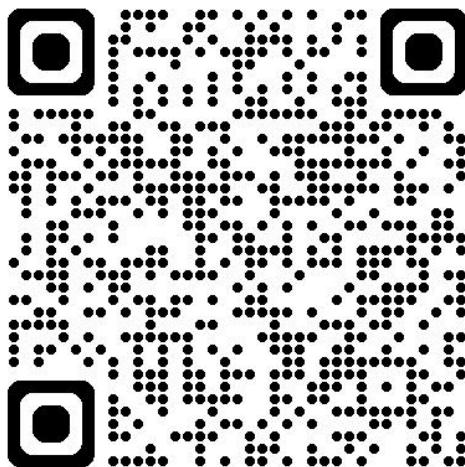


图 7-10 移动端二维码

7.4 项目的代码提交和版本管理

编写好 webUI5.html 代码，并且运行和测试成功后，执行如下图 7-11 所示

命令提交代码：

```
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/abc/webUI (master)
$ git add webUI5.html
```

```
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/abc/webUI (master)
$ git commit -m "用一套代码同时为鼠标和触屏建模"
```

图 7-11

提交代码后 git bash 给出的反馈如下图 7-12 所示：

```
[master 4f4e119] 用一套代码同时为鼠标和触屏建模"
2 files changed, 197 insertions(+)
create mode 100644 webUI/webUI5
create mode 100644 webUI/webUI5.html
```

图 7-12

项目代码仓库自此也开启了严肃的历史记录，我们可以输入如下图 7-13 所示的日志命令查看：

```
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/abc/webUI (master)
$ git log
```

图 7-13

Git bash 反馈代码的仓库日志如下图 7-14 所示：

```
commit 4f4e119578bc69c921ab1f956cbf708aa3bc5e8b (HEAD -> master)
Author: lixinyu945 <102445129+lixinyu945@users.noreply.github.com>
Date:   Fri Jun 14 10:41:18 2024 +0800
```

用一套代码同时为鼠标和触屏建模“

图 7-14

8. UI 的个性化键盘交互控制的设计开发

8.1 分析和设计

键盘布局和功能设计是个性化键盘设计的核心。设计师需要根据用户需求和使用场景，合理规划键盘的按键布局和功能设置。个性化键盘的交互逻辑和用户体验同样重要。设计师需要确保键盘的交互逻辑清晰、简洁，易于用户理解和操作。同时，还需要关注用户在输入过程中的反馈和提示。在开发个性化键盘时，需要选择合适的技术和工具。根据项目的需求，可以选用原生开发、跨平台框架或 Web 技术等方式进行实现。在开发过程中，需要充分考虑性能、兼容性和可维护性等因素。

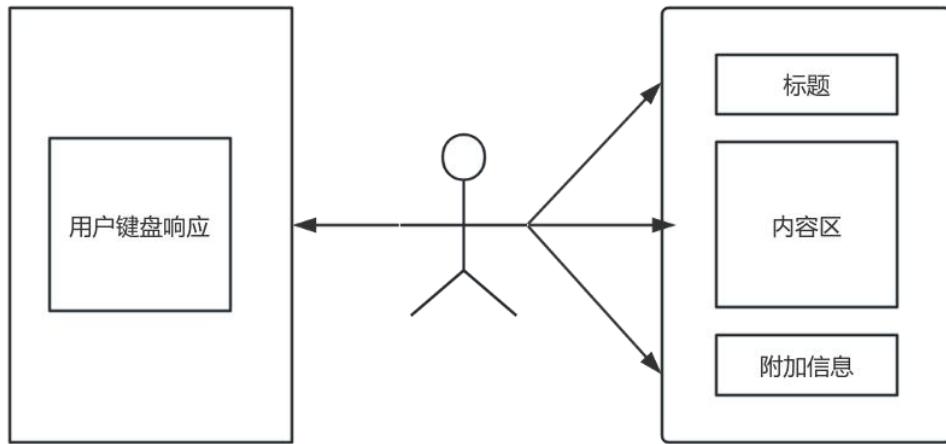


图 8-1 用例图

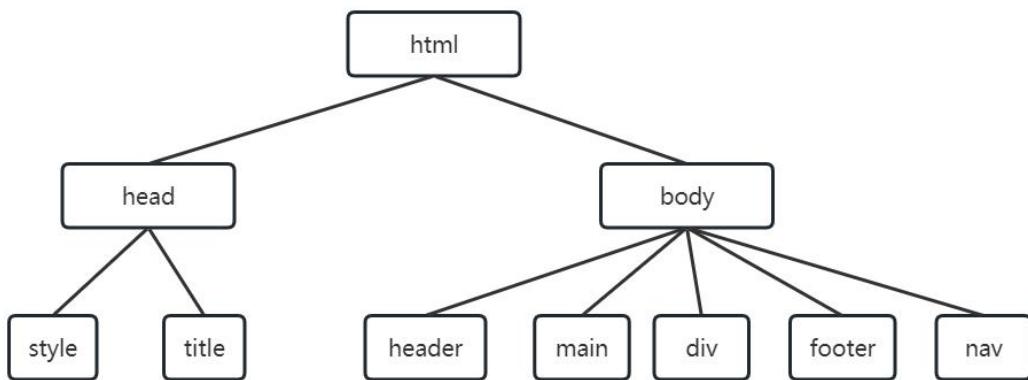


图 8-2 DOM 树状图

8.2 项目的编程和实现

阐述探索和利用、键盘底层事件，为未来 UI 的键盘功能提供底层强大的潜力。

因为系统中只有一个键盘，所以我们在部署代码时，把键盘事件的监听设置在 DOM 文档最大的可视对象——body 上，通过测试，不宜把键盘事件注册在 body 内部的子对象中

① webUI6.html 的 HTML 代码编写如下图所示

```

var UI = {};
if(window.innerWidth>600){
    UI.appWidth=600;
} else{
    UI.appWidth = window.innerWidth;
}
UI.appHeight = window.innerHeight;
let baseFont = UI.appWidth /20;
//通过改变body对象的字体大小，这个属性可以影响其后代
document.body.style.fontSize = baseFont +"px";
//通过把body的高度设置为设备屏幕的高度，从而实现纵向全屏
//通过CSS对子对象百分比（纵向）的配合，从而达到我们响应式设计的目标
document.body.style.width = UI.appWidth - baseFont + "px";
document.body.style.height = UI.appHeight - baseFont*5 + "px";
if(window.innerWidth<1000){
    $("aid").style.display='none';
}
$("aid").style.width=window.innerWidth-UI.appWidth - baseFont*3 +'px';
$("aid").style.height= UI.appHeight - baseFont*3 +'px';

```

图 8-3 webUI6.html HTML

```

//尝试对鼠标和触屏设计一套代码实现UI控制
var Pointer = {};
Pointer.isDown= false;
Pointer.x = 0;
Pointer.deltaX =0;
{ //Code Block Begin
let handleBegin = function(ev){
    Pointer.isDown=true;
    if(ev.touches){console.log("touches1"+ev.touches);
        Pointer.x = ev.touches[0].pageX ;
        Pointer.y = ev.touches[0].pageY ;
        console.log("Touch begin : "+("+"+Pointer.x +"," +Pointer.y +")" );
        $(<u>bookface</u>).textContent= "触屏事件开始，坐标: "+("+"+Pointer.x+","+Pointer.y+"));
    }else{
        Pointer.x= ev.pageX;
        Pointer.y= ev.pageY;
        console.log("PointerDown at x: "+("+"+Pointer.x +"," +Pointer.y +")" );
        $(<u>bookface</u>).textContent= "鼠标按下，坐标: "+("+"+Pointer.x+","+Pointer.y+"));
    }
};

let handleEnd = function(ev){
    Pointer.isDown=false;
    ev.preventDefault()
}

```

图 8-4 webUI6.html HTML

```
if(ev.touches){
    $("bookface").textContent= "触屏事件结束!";
    if(Math.abs(Pointer.deltaX) > 100){
        $("bookface").textContent += ", 这是有效触屏滑动! ";
    }else{
        $("bookface").textContent += " 本次算无效触屏滑动! ";
        $("bookface").style.left = '7%' ;
    }
}else{
    $("bookface").textContent= "鼠标松开!";
    if(Math.abs(Pointer.deltaX) > 100){
        $("bookface").textContent += ", 这是有效拖动! ";
    }else{
        $("bookface").textContent += " 本次算无效拖动! ";
        $("bookface").style.left = '7%' ;
    }
}
};
```

图 8-5 webUI6.html HTML

```
let handleMoving = function(ev){
    ev.preventDefault();
    if (ev.touches){
        if (Pointer.isDown){
            console.log("Touch is moving");
            Pointer.deltaX = parseInt( ev.touches[0].pageX - Pointer.x );
            $("bookface").textContent= "正在滑动触屏, 滑动距离: " + Pointer.deltaX +"px 。";
            $('bookface').style.left =  Pointer.deltaX + 'px' ;
        }
    }else{
        if (Pointer.isDown){
            console.log("Pointer isDown and moving");
            Pointer.deltaX = parseInt( ev.pageX - Pointer.x );
            $("bookface").textContent= "正在拖动鼠标, 距离: " + Pointer.deltaX +"px 。";
            $('bookface').style.left =  Pointer.deltaX + 'px' ;
        }
    }
};

$("bookface").addEventListener("mousedown",handleBegin );
$("bookface").addEventListener("touchstart",handleBegin );
$("bookface").addEventListener("mouseup", handleEnd );
$("bookface").addEventListener("touchend",handleEnd );
$("bookface").addEventListener("mouseout", handleEnd );
$("bookface").addEventListener("mousemove", handleMoving);
$("bookface").addEventListener("touchmove", handleMoving);
```

图 8-6 webUI6.html HTML

```

//提出问题：研究利用"keydown"和"keyup"2个底层事件，实现同时输出按键状态和文本内容
$("body").addEventListener("keydown",function(ev){
  ev.preventDefault() ; //增加“阻止事件对象的默认事件后”，不仅 keypress 事件将不再响应，而且系统的热键,
  let k = ev.key;
  let c = ev.keyCode;
  $("keyStatus").textContent = "按下键 [" + k + "] " + "编码 [" + c + "]";
});

$("body").addEventListener("keyup",function(ev){
  ev.preventDefault() ;
  let key = ev.key;
  $("keyStatus").textContent =  key + " 键已弹起" ;
  if (printLetter(key)){
    $("typeText").textContent += key ;
  }
  function printLetter(k){
    if (k.length > 1){ //学生须研究这个逻辑的作用
      return false ;
    }
    let puncs = ['~','`','!','@','#','$','%','^','&','*','(',')','_','+','=','`','.',',';','`'];
    if ((k >= 'a' && k <= 'z')|| (k >= 'A' && k <= 'Z')|| (k >= '0' && k <= '9')) {
      console.log("letters") ;
      return true ;
    }
    for (let p of puncs ){
      if (p === k) {
        console.log("puncs") ;
        return true ;
      }
    }
  }
}

```

图 8-7 webUI6.html HTML

```

  return false ;
  //提出更高阶的问题，如何处理连续空格和制表键tab?
} //function printLetter(k)
});
} //Code Block End
function $(ele){
  if (typeof ele !== 'string'){
    throw("自定义的$函数参数的数据类型错误，实参必须是字符串！");
    return
  }
  let dom = document.getElementById(ele) ;
  if(dom){
    return dom ;
  }else{
    dom = document.querySelector(ele) ;
    if (dom) {
      return dom ;
    }else{
      throw("执行$函数未能在页面上获取任何元素，请自查问题！");
      return ;
    }
  }
} //end of $

```

图 8-8 webUI6.html HTML

② webUI6.html 的 CSS 代码编写如下图所示

```
#bookface{  
    position: absolute;  
    width: 80%;  
    height: 80%;  
    border: 1px solid red;  
    background-color: blanchedalmond;  
    left: 7%;  
    top: 7%;  
}  
#aid{  
    position: absolute;  
    border: 3px solid blue;  
    top: 0px;  
    left: 600px;  
}  
#typeText{  
    border: 1px solid blue;  
    padding: 0.2em;  
    color: gray;  
}  
#keyStatus{  
    position: absolute;  
    border: 1px solid blue;  
    width: 90%;  
    right: 0;  
    bottom: 0;  
    font-size: 0.6em;  
    padding: 0.5em;  
}
```

图 8-9 webUI6.html HTML

8.3 项目测试与运行



图 8-10 PC 端运行效果图

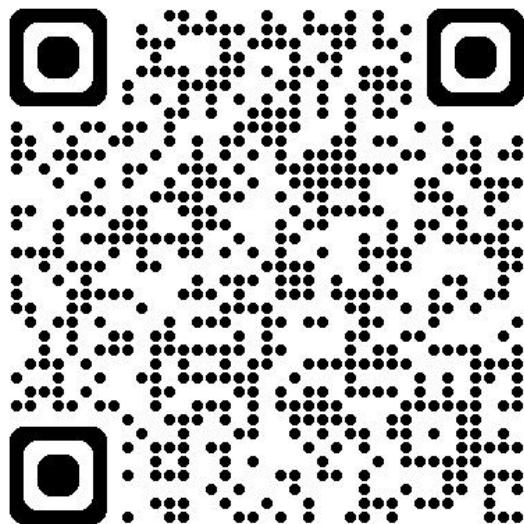


图 8-11 移动端二维码

8.4 项目的代码提交和版本管理

编写好 webUI6.html 代码，并且运行和测试成功后，执行如下图 8-12 所示命令提交代码

```
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/abc/webUI (master)
$ git add webUI6.html

给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/abc/webUI (master)
$ git commit -m "UI的个性化键盘交互控制的设计开发"
```

图 8-12

提交代码后 git bash 给出的反馈如下图 8-13 所示

```
[master 6523f89] UI的个性化键盘交互控制的设计开发
1 file changed, 238 insertions(+)
create mode 100644 webUI/webUI6.html
```

图 8-13

项目代码仓库自此也开启了严肃的历史记录，我们可以输入如下图 8-14 所示的日志命令查看

```
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/abc/webUI (master)
$ git log
```

图 8-14

Git bash 反馈代码的仓库日志如下图 8-15 所示：

```
commit 6523f89b5da00cda1cbcbf2d9d9f0c8800168da7 (HEAD -> master)
Author: lixinyu945 <102445129+lixinyu945@users.noreply.github.com>
Date:   Fri Jun 14 10:48:59 2024 +0800
```

UI的个性化键盘交互控制的设计开发

图 8-15

9. 用 git Bash 工具管理项目的代码仓库和 http 服务器

9.1 经典 Bash 工具介绍

当我们谈到命令行时，我们实际上指的是 shell。shell 是一个接受键盘命令并将其传递给操作系统执行的程序。几乎所有的 Linux 发行版都提供了一个来自 GNU 项目的 shell 程序，名为 bash。这个名字是 Bourne -again shell 的首字母缩略词，指的是 bash 是 sh 的增强替代品，sh 是 Steve Bourne 编写的原始 Unix shell 程序。

像 Windows 一样，像 Linux 这样的类 unix 操作系统用所谓的分层目录结构来组织文件。这意味着它们被组织成树状的目录模式(在其他系统中有时称为文件夹)，其中可能包含文件和其他目录。文件系统中的第一个目录称为根目录。根目录包含文件和子目录，子目录包含更多的文件和子目录，以此类推。

9.2 创建一个空的远程代码仓库

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (*).

Owner *	Repository name *
 lixinyu5	/ lixinyu.github.io
<small>lixinyu.github.io is available.</small>	

Create repository

图 9-1

点击窗口右下角的绿色“Create repository”，则可创建一个空的远程代码仓

库。

9.3 设置本地仓库和远程代码仓库的链接

进入本地 webUI 项目的文件夹后，通过下面的命令把本地代码仓库与远程建立密钥链接：

```
给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (master)
$ echo "WebUI应用的远程http服务器设置" >> README.md

给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (master)
$ git init
Reinitialized existing Git repository in D:/webUI/.git/

给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (master)
$ git add README.md
warning: LF will be replaced by CRLF in README.md.
The file will have its original line endings in your working directory

给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (master)
$ git commit -m "这是我第一次把代码仓库上传至github平台"
[master b430bfc] 这是我第一次把代码仓库上传至github平台
 1 file changed, 1 insertion(+)
 create mode 100644 README.md

给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (master)
$ git branch -M main

给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (main)
$ git remote add origin https://github.com/lixinyu5/lixinyu.github.io.git

给你个机会@LAPTOP-6A0M1O15 MINGW64 /d/webUI (main)
$ git push -u origin main
```

图 9-2

本项目使用 window 平台，git bash 通过默认浏览器实现密钥生成和记录，第一次链接会要求开发者授权，如下图所示：

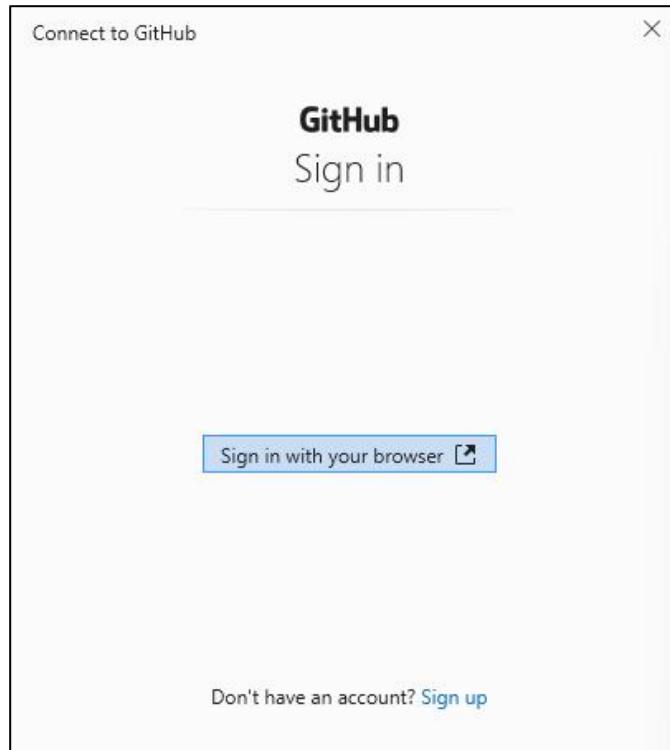


图 9-3

再次确认授权 git Bash 拥有访问改动远程代码的权限，如下图所示：

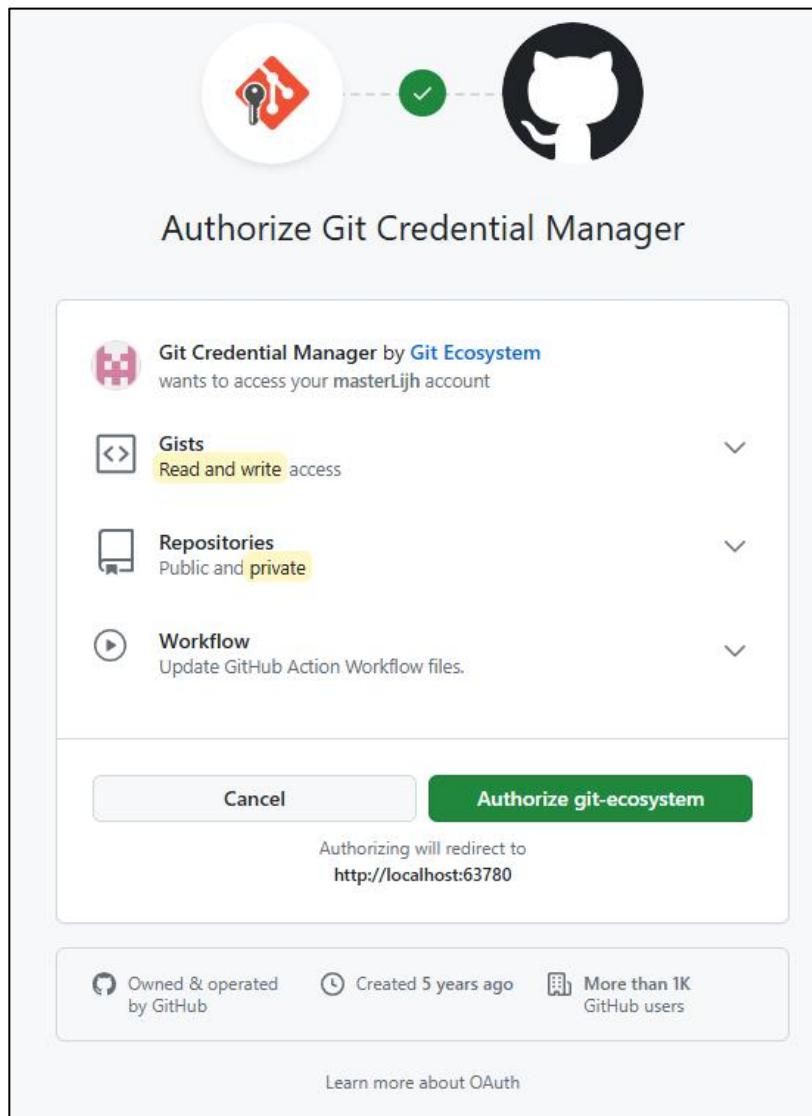


图 9-4

最后，GitHub 平台反馈：git Bash 和 github 平台成功实现远程链接。

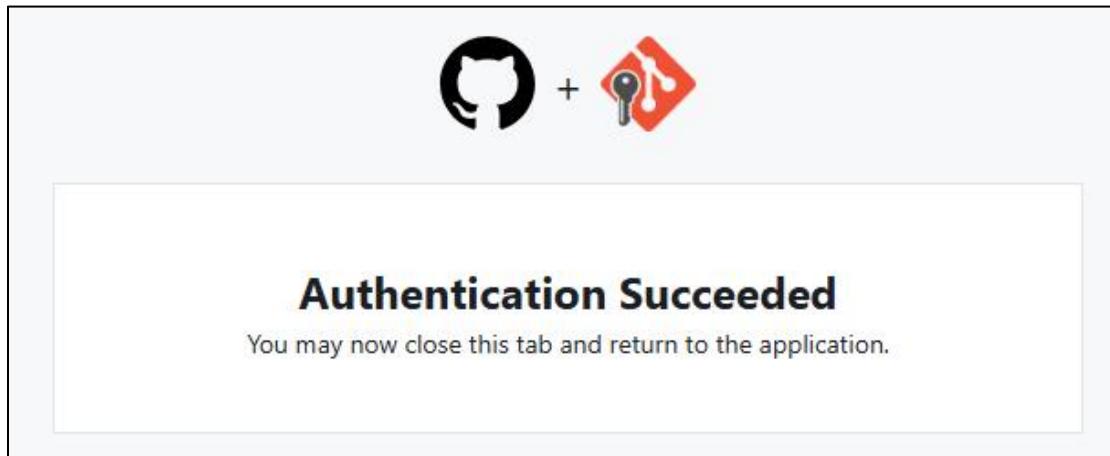
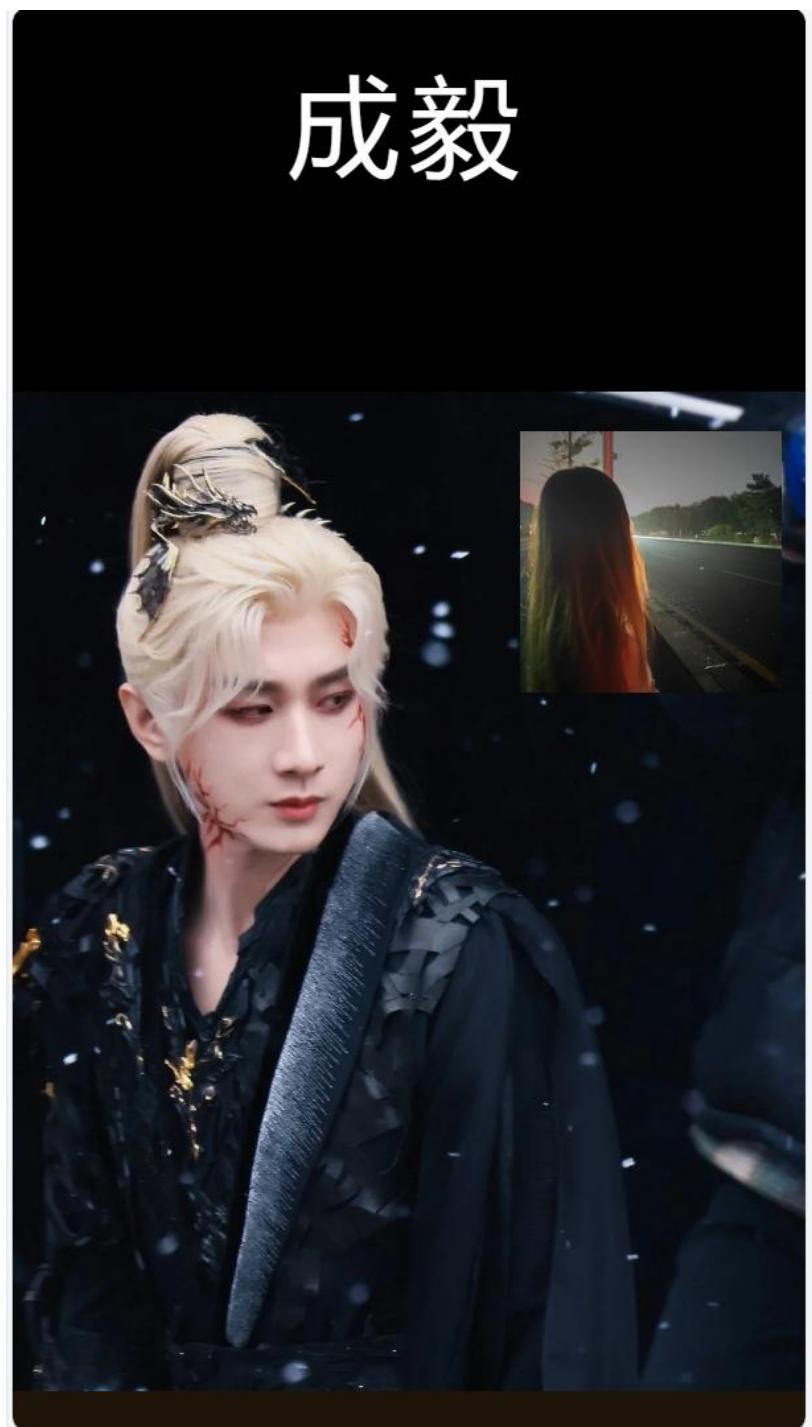


图 9-5

从此，我们无论在本地做了任何多次代码修改，也无论提交了多少次，上

传远程时都会把这些代码和修改的历史记录全部上传 github 平台，而远程上传命令则可简化为一条：git push ， 极大地方便了本 Web 应用的互联网发布。

远程代码上传后，项目可以说免费便捷地实现了在互联网的部署，用户可以通过域名或二维码打开，本次使用 PC 的微软 Edge 浏览器打开，本文截取操作中间的效果图，如下所示：



参考文献

- [1] 冯传春. 新媒体视域下广播电视台新闻宣传的优化措施 [J]. 卫星电视与宽带多媒体, 2024, 21 (10): 70-72.
- [2] 闫梦婷. 基于信创人才培养的“Java Web 综合应用”课程教学改革的探讨 [J]. 科技风, 2024, (14): 142-144. DOI:10.19392/j.cnki.1671-734webUI202414048.
- [3] 余晓平,李志刚,高攀,等. 以效果为导向的 Java Web 课程思政探索 [J]. 计算机教育, 2024, (05): 123-127. DOI:10.16512/j.cnki.jsjjy.2024.05.010.
- [4] 郑思伟,王家乐,周诗源,等. 基于复杂网络理论和软件度量的 Web 服务演化研究 [J/OL].
计 算 机 应 用 与 软 件 , 1-11[2024-06-09].
<http://kns.cnki.net/kcms/detail/3webUI1260.tp.20240427.1918.002.html>.
- [5] 何家乐,杜学文,洪思云,等. 智能制造数据可视化大屏的 UI 设计探究 [J]. 现代传输, 2024, (02): 50-53.
- [6] 宗林,胡姝帆. 课程思政背景下“UI 界面设计”课程探索 [J]. 三角洲, 2024, (08): 245-247.
- [7] 黄然歆. UI 界面中扁平化设计的美学研究及应用 [J]. 大观, 2024, (03): 10-12.
- [8] 周沫玲. 多屏时代下 WebUI 设计研究 [J]. 计算机光盘软件与应用, 2013, 16 (16): 247-248.