

成 绩:

# 江西科技师范大学

## 毕业设计（论文）

题目（中文）：基于 Web 客户端技术的个性化 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：张梓鑫

学号：20213627

指导教师：李健宏

2024 年 6 月 17 日

# 目录

1 前言 .....	4
1.1 毕业设计任务分析 .....	4
1.1.1 明确毕业论文和毕业设计的关系 .....	4
1.1.2 对毕业设计/论文内涵和意义理解 .....	5
1.1.3 毕业设计/论文和个人专业成长关系 .....	5
1.2 研学计划 .....	6
1.2.1 实现目标的路径规划 .....	6
1.2.2 毕设的技术路线规划 .....	7
1.3 研究方法 .....	7
1.3.1 文献法 .....	8
1.3.2 模型研究法 .....	8
1.3.3 其他会误解的研究方法 .....	9
2 技术总结和文献综述 .....	9
2.1 Web 平台和客户端技术概述 .....	9
2.1.1 历史 .....	10
2.1.2 万维网联盟 .....	10
2.1.3 Web 平台和 Web 编程 .....	10
2.2 项目的瀑布模型与增量式迭代开发模式 .....	11
2.2.1 瀑布模型 .....	11
2.2.2 迭代增量式模型 .....	11
3 用简洁的“三段论”方式开展内容设计 .....	12
3.1 分析和设计 .....	12
3.1.1 基本分析 .....	12
3.1.2 相关图设计 .....	13
3.2 项目的实现和编程 .....	14
3.2.1 HTML 代码编写如下： .....	14
3.2.2 CSS 代码编写如下： .....	15
3.3 项目的运行和测试 .....	16
3.4 项目的代码提交和版本管理 .....	17
4 移动互联时代的 UI 开发初步——窄屏终端的响应式 .....	18
4.1 分析和设计 .....	18
4.1.1 基本分析 .....	18
4.1.2 相关图设计 .....	18
4.2 项目的实现和编程 .....	19
4.2.1 HTML 代码编写如下： .....	19
4.2.2 CSS 代码编写如下： .....	20
4.2.3 JavaScript 代码编写如下： .....	22
4.3 项目的运行和测试 .....	22
4.4 项目的代码提交和版本管理 .....	24
5 应用响应式设计技术开发可适配窄屏和宽屏的 UI .....	25
5.1 分析和设计 .....	25
5.1.1 基本分析 .....	25

5.1.2 相关图设计 .....	25
5.2 项目的实现和编程 .....	26
5.2.1 HTML 代码编写如下: .....	26
5.2.2 CSS 代码编写如下: .....	27
5.2.3 JavaScript 代码编写如下: .....	30
5.3 项目的运行和测试 .....	30
5.4 项目的代码提交和版本管理 .....	33
6 个性化 UI 设计中鼠标移动点击和封面轮播模型 .....	34
6.1 分析和设计 .....	34
6.1.1 基本分析 .....	34
6.1.2 相关图设计 .....	34
6.2 项目的实现和编程 .....	35
6.2.1 HTML 代码编写如下: .....	35
6.2.2 CSS 代码编写如下: .....	36
6.2.3 JavaScript 代码编写如下: .....	39
6.3 项目的运行和测试 .....	41
6.4 项目的代码提交和版本管理 .....	43
7 通用的 UI 设计, 同时为触屏和鼠标拖拽移动行为建模 .....	44
7.1 分析和设计 .....	44
7.1.1 基本分析 .....	44
7.1.2 相关图设计 .....	44
7.2 项目的实现和编程 .....	45
7.2.1 HTML 代码编写如下: .....	45
7.2.2 CSS 代码编写如下: .....	46
7.2.3 JavaScript 代码编写如下: .....	49
7.3 项目的运行和测试 .....	53
7.4 项目的代码提交和版本管理 .....	54
8 UI 的个性化键盘交互控制的设计开发 .....	55
8.1 分析和设计 .....	55
8.1.1 基本分析 .....	55
8.1.2 相关图设计 .....	55
8.2 项目的实现和编程 .....	56
8.2.1 HTML 代码编写如下: .....	56
8.2.2 CSS 代码编写如下: .....	57
8.2.3 JavaScript 代码编写如下: .....	60
8.3 项目的运行和测试 .....	65
8.4 项目的代码提交和版本管理 .....	66
9 用 Git Bash 工具管理项目的代码仓库和 Http 服务器 .....	67
9.1 跨世纪的经典 Bash 工具 .....	67
9.2 通过 Git Hub 平台实现本项目的全球域名 .....	68
9.3 创建一个空的远程代码仓库 .....	68
9.4 设置本地仓库和远程代码仓库的链接 .....	68
参考文献 .....	72

# 基于 Web 客户端技术个性化 UI 设计与实现

**摘要：**近十年来，基于 HTML5 核心的 Web 标准软件开发技术因其跨平台和开源的优势，已广泛应用于各类领域的应用软件开发中。通过分析本次毕设任务，本项目选择 html5 的 web 客户端技术为技术路线，展开对程序设计和软件开发的研究和实践。通过广泛查阅相关技术书籍、开发者论坛和文献，设计开发了一个个性化的用户界面（UI）的应用程序。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，除直接使用了 Web 客户端底层的 API 外，本项目的每条代码都是手工逐条编写，没有导入他人的任何的代码（框架和库）。项目中融入了响应式设计，确保了应用程序能够适应不同尺寸的屏幕，满足移动互联网用户的需求。同时，项目中还大量采用了面向对象的编程技术，特别是开发了一个统一的 pointer 模型，用以同时支持鼠标和触屏操作，提升了代码的通用性和质量，这也是项目的一大特色。从工程管理的角度看，本项目采用的增量式开发模式，以逐步求精的方式展开了六次代码的增量式重构（A:分析 D:设计 I:实现 T:测试），比较愉快地实现项目的设计开发和测试。在代码管理方面，项目使用了 Git 进行版本控制，确保了开发过程的有序性。在开发和测试过程中，代码经过了六次迭代和提交。最终，通过 Git-Bash 将代码仓库推送至 GitHub，并利用 GitHub 的 HTTP 服务功能，实现了应用程序的全球部署，用户可以通过网址或二维码轻松访问，实现了跨平台的高效互动体验。

**关键字：**Web 客户端技术；自适应设计；增量式开发；Git 版本控制。

## 1 前言

### 1.1 毕业设计任务分析

学生需将本科期间所掌握的计算机科学技能，包括程序设计和软件工程的理论知识与编码实践，应用于构建个性化的技术解决方案。基于个人的研究兴趣，明确软件需求，进而系统性地执行从问题解析、模型构建到软件架构设计、编码实现以及系统测试的整个软件生命周期。通过毕业设计项目，学生将整合开发经验并撰写相关文档，实现理论与实践的融合，全面提升计算思维和工程分析能力。

#### 1.1.1 明确毕业论文和毕业设计的关系

学校的毕业论文（设计）管理手册主要强调了论文部分，对设计作品的要求则没有明确规定。我认为这样的表述对于软件开发等实践性较强的毕设项目来说，可能不够准确，因为通常我们需要先完成软件的开发，然后才能基于此撰写相应的论文。

为了更清晰地界定本专业毕设与论文的关系，我建议将手册中的表述从“毕业论文（设计）”调整为“毕业设计/论文”。这样的定义更明确地指出，论文是基

于设计成果（即学生完成的软件作品）的总结和理论提升。在后续的文档中，我们将统一使用“毕业设计/论文”这一表述方式。

此外，在计算机专业的软件开发毕设论文中，我们可以通过插入软件界面的截图来辅助说明，展示软件的实际运行效果，从而增强论文的说服力。同时，论文中还可以直接提供软件的访问网址或二维码，让读者能够直接体验和使用毕设作品。这种做法不仅能够证明论文内容的真实性，也是本专业论文的一大特色。相比仅依赖文本描述的论文，这种结合实际作品的论文无疑具有更大的优势。

通过这些调整，我们可以使毕设要求更加明确，同时充分发挥计算机专业论文的特点，提高论文的质量和可信度。

### 1.1.2 对毕业设计/论文内涵和意义理解

作为计算机科学与技术专业的学生，我们在本科学习即将结束时，设计并开发一个体现本专业特色的项目是非常有意义的。这不仅是对所学知识体系的回顾和总结，更是展示我们实际能力的机会。

在我的毕业设计中，我深入探讨了面向对象编程、数据结构与算法、操作系统和软件工程等核心课程的理论。以往这些课程的理论知识显得有些抽象，主要侧重于理论学习，使得我们感觉缺乏实践应用。若能将这些核心课程的关键知识点应用到实际项目中，无疑将在理解深度和技术层面提升我的专业素养。因此，我认为毕业设计是将大学期间所学的理论知识在实践中进行综合应用和总结的过程。同时，我也会学习并应用当前流行的技术，以形成对计算机软硬件系统的全面和深入理解，进而撰写毕业论文，这正是毕业设计的核心所在。

对我们专业的开发者来说，深刻理解计算机系统至关重要。作为即将成为国家现代化建设的工程师，我们与其他专业的区别在于，我们对计算机系统的理解不应停留在表面，而应更深入地接近其本质。对技术的掌握也应力求深入到技术的基础和原理层面。

通过这样的学习和实践过程，我们不仅能够巩固和提升自己的专业知识，还能够形成对计算机系统的深刻理解，为将来的职业生涯打下坚实的基础。

### 1.1.3 毕业设计/论文和个人专业成长关系

毕业设计/论文环节在本科教育中扮演着至关重要的角色，它不仅综合性强，而且能够充分展现个性，对提高本科生的综合能力具有深远的影响。我们应该从

个人成长的视角来重视这一环节，结合国家现代化的目标，深刻理解教育部门和学校的相关规划。

正如我的导师李教授所言，毕业设计任务一开始，我们就应该从自身的需求出发，主动规划，将设计/论文作为自己主动学习和自我教育的重要阶段。在专业目标的指引下，选择适合自己的方向和技术路径，在导师的引导下，努力成为具备以下素质的专业人才：（1）拥有现代工程思维的工程师；（2）掌握现代数字工具的能力；（3）具有国际视野的自主学习者；（4）兼具人文关怀的全面发展者。其中，前两点对于毕设的开发至关重要，而后两点则对论文撰写大有裨益。

通过参与查资料、撰写论文、参加组会和答辩等活动，我更加明白了本科生经历这些过程的必要性。

面对毕业设计/论文这一重要任务，我思考了如何积极、愉快地完成这一充满挑战和不确定性的旅程。经过深思熟虑，我认为，要少走弯路，就需要亲自实践（如编写代码）、广泛阅读（如查阅文献）和乐于总结（如撰写文档）。我相信，付出与回报是成正比的，只要坚持这样的行为准则，我们就能不断成长。

我期待通过这一年学习研究，能够培养出教授经常提到的计算思维和编程能力，同时体验到个人能力提升带来的快乐。

## 1.2 研学计划

磨刀不误砍柴功，要落实好做毕业设计、写好毕业论文，做好规划和选择技术路线是这项系统工程的顶层设计。

### 1.2.1 实现目标的路径规划

我的毕业设计分为两个主要步骤。首先，我将挑选一条自己热衷的技术路径，深入学习并整合关键技术。以导师的项目为例，我将重点理解不同技术间的相互作用及其在项目中的具体角色，同时在实践中提升编写优质代码的技能。

一旦掌握了模仿导师案例所需的技术，我将进入第二阶段，着手开发自己的毕业设计软件。这一阶段将遵循软件工程的标准流程：首先，根据自己面临的问题进行定义和分析；其次，设计一套适宜的技术解决方案；然后，按照设计方案编写代码并部署技术；最后，进行调试、测试和性能优化。在第三和第四步中发现的问题可能会导致我多次回到第二步和第三步，以修正设计缺陷或代码错误。

大部分工作将集中在第三步，即构建代码体系和实现软件架构的细节。

与个人开发者类似，我的毕设在设计和实现阶段缺乏经验丰富的团队支持，因此，方案设计的细化和代码部署往往是交替进行的，这有助于我在微观层面上关注细节，同时确保宏观设计满足需求。

在开发过程中，我将生成大量的开发文档。通过对这些文档的整理，并结合专业知识理论，我将撰写自己的论文。这一过程不仅加深了我对本科学习理论的理解，而且通过实践来驱动理论学习，实现理论与实践的最佳结合。

### 1.2.2 毕设的技术路线规划

技术路线是理工科领域实现复杂工程的计划，它强调专业性和逻辑性，用以证明项目或研究的可行性。计算机科学与技术是一个包含众多专业方向的大学科。选择合适的技术路线对于保持学习热情和效率至关重要。

我将计算机科学与技术专业方向分为三类：硬件体系结构、系统软件的体系和管理、应用软件的开发。基于个人兴趣和社会需求，我选择了应用软件发展方向，它不仅需求广泛，也适合我倾向于解决具体问题的学习方式。

软件发展方向技术路线丰富且复杂，涉及多种编程语言和平台。跨平台开发是关键问题，HTML5 和 ECMAScript 标准为实现这一目标提供了可能。我的技术路线是学习 Web 标准和相关技术，开发高质量、跨平台的 Web 应用。

技术路线的实践分为三个层次：

- (1) 对计算机硬件和操作系统的理解，以及如何提升代码性能和适用性。
- (2) 掌握面向对象编程，尤其是 JavaScript 和 ECMAScript，以及 MVC 设计模式的应用。
- (3) 理解软件工程原理，掌握软件开发流程和版本控制，特别是 Git 的使用，以促进代码分享和团队合作。

通过这条技术路线，我不仅提升了代码能力，也加深了对计算机系统的理解，学会了如何更有效地分析和解决问题。这证明了在技术学习初期，应专注于完善技术学习和规划具体技术路线，而不是急于分析问题和需求。

## 1.3 研究方法

在学习的旅途中，不同领域的研究策略往往有共通之处，我们可以跨学科地

应用和借鉴这些策略。对于本科生而言，研究方法可能显得有些模糊，但事实上，无论我们探索哪个学科，其根本的研究技巧都是相似的，以下我将简述对本专业研究方法的一些基本理解。

### 1.3.1 文献法

文献法作为一种研究手段，其核心在于搜集、评估和归纳各类文献资料，并通过深入分析这些资料来构建对特定现象或领域的科学理解。这一方法历史悠久且持续发展，它要求研究者在继承前人成果的同时，进行批判性思考，旨在通过对文献的系统性搜集和分析，揭示特定社会或教育现象的特征，探讨其成因，并可能在此基础上提出新的见解或理论。

文献法的信息来源极为广泛，不仅限于传统书籍和学术文章，还包括报纸、杂志、官方统计数据、年鉴、政策文件、学术论文和调研报告等。在运用文献法时，研究者必须对所搜集的文献进行严格的真实性和适用性评估，以保证研究的严谨性和有效性。优质的文献资源应具有真实性、创新性、全面性和准确性。

文献研究的一般流程涵盖以下几个步骤：首先，根据现有理论和实际需求提出研究问题或假设；其次，进行研究设计，明确研究目标和方法；然后，广泛搜集相关文献；接着，对搜集到的文献进行整理和归纳；最后，撰写文献综述，总结现有研究的主要发现和观点。文献法的关键在于对文献进行深入分析和重新解读，以形成对研究主题的全面理解。

### 1.3.2 模型研究法

模型研究法，亦称为模拟法，是一种科学探究手段，它通过构建与研究对象相似的模型来进行间接性研究。此法特别适用于那些直接研究存在困难的系统。研究者首先搜集与研究对象相关的各类数据和资料，以便掌握其属性、构造和运作方式。随后，在对这些信息进行综合分析的基础上，运用创造性思维和逻辑推理来构建模型。接下来，通过观察、实验和分析模型，进而对模型进行深入研究。最终，研究者将从模型中得到的结论推广至原形。

此法涉及的方法多样，可以是物理模拟、数学模拟，也可是功能模拟或智能模拟，选择哪种方法取决于研究对象的特性、复杂性及其应用目的。在模型分析过程中，会使用到系统分析、综合、结构分析、要素分析、功能分析、优化分析等方法，同时还会用到计算机技术和逻辑推理。

模型研究法已成为理论研究和工程研究中的重要工具，它通过模型的简化和理想化展示，帮助我们理解原型的复杂性，是连接理论与实践的纽带。此外，模型本身也是对假设的一种体现，它需要在实际操作中不断得到验证和完善。

构建模型时，应确保模型既真实反映原型，又足够简洁以便于数学处理，同时还要保证模型的完整性和规范性，满足基本要素的需求。模型研究法的步骤包括信息搜集、模型构建、观察实验、分析研究，以及结论的外推。

### 1.3.3 其他会误解的研究方法

案例研究法专注于详尽地剖析个别案例，是管理学等学科中极为重要的研究手段。例如，产品经理可能通过对某一软件产品进行深入分析来运用此法。不过，在以技术为主导的计算机软件开发领域，这种方法可能并不十分契合。

实验法多用于医学研究，通过对照不同实验条件下的数据来探索因果关系，这与计算机科学中的程序和代码开发存在差异，后者更侧重于对现实世界现象的模拟与抽象化处理。

实践法并非一个独立的研究方法，而是一个应用理论于现实世界的过程，用以验证理论的有效性。在计算机科学中，无论是编程还是系统开发，都需要通过实践来证实其有效性和效率。即便是计算机科学的基础理论研究，如理论和数学研究，虽可能不直接涉及实际操作，但其最终目标也是为了更深入地理解现实世界，并为实践提供理论支撑和指导。

## 2 技术总结和文献综述

### 2.1 Web 平台和客户端技术概述

蒂姆·伯纳斯-李（Tim Berners-Lee）在构建了万维网（Web）的基础架构之后，发起成立了万维网联盟（W3C）。W3C 在 2010 年之后推出了 HTML5 这一全球性标准，并且与欧洲计算机制造商协会（ECMA）共同维护了 ECMAScript 标准，这两项标准极大地推动了全球开发平台的统一化。直至今日，科技专家和网络行业的专业人士仍在持续地努力，以进一步优化和提升这一宏伟目标。

掌握 Web 标准和相关技术，学习开发 Web 程序和利用相应工具，以构建一套能够在不同平台上运行的高质量应用程序，这是我毕业设计项目所遵循的技术路径。

### 2.1.1 历史

在 1989 年，蒂姆·伯纳斯-李爵士提出了万维网的概念，并撰写了最初的提案。他首次提出了“万维网”这一术语，并在 1990 年 10 月开发出了首个万维网服务器程序“httpd”以及首个客户端工具——集浏览器和编辑器功能于一体的“WorldWideWeb”。

伯纳斯-李还制定了“超文本标记语言”（HTML）的初始版本，这是一种支持超链接的文档编写语言，后来成为互联网上最主要的内容发布格式。随着万维网技术的普及，他对统一资源标识符（URI）、超文本传输协议（HTTP）和 HTML 的最初定义进行了持续的优化和深入讨论。

### 2.1.2 万维网联盟

1994 年，在众多企业纷纷增加网络资源投入的背景下，万维网联盟（W3C）宣告成立。蒂姆·伯纳斯-李爵士担任了联盟的领导角色，致力于推动统一的网络架构发展，以适应迅速演变的网络标准，这些标准是构建网站、浏览器和设备，体验网络服务的基础。

成立 W3C 时，伯纳斯-李爵士汇集了一个由同行组成的社区，以应对 Web 技术的迅猛发展，确保有一个统一的组织来协调和制定 Web 标准。他接受了麻省理工学院的邀请，凭借自己在联盟中的丰富经验，负责 W3C 的运营，并从一开始就确立了 W3C 的全球性愿景。

### 2.1.3 Web 平台和 Web 编程

首先，让我们对 Web，即万维网，进行简要说明。通常人们更倾向于使用“Web”而非完整的“World Wide Web”称呼，本文也将遵循这一习惯。网络实际上是由众多网页组成的文档系统，这些网页由全球的计算机用户所共享，它们在电脑屏幕上展示信息。这里所指的“信息”包括文本、图像以及诸如文本框和按钮等用户交互元素。

Web 编程是一个广泛的领域，涵盖了多种不同的编程工具和语言。尽管存在多样性，但所有这些工具都围绕着核心语言 HTML 展开，这也是几乎所有 Web 编程书籍都会涉及的内容。本书将深入探讨 HTML5、CSS 和 JavaScript 这三种技术，它们构成了客户端 Web 编程的基础。客户端 Web 编程的特点是，所有的网页计算任务都在用户的个人电脑（即客户端）上完成。

HTML、CSS 和 JavaScript 的结合，也展现了现代社会分工的智慧，它们在 MVC（模型-视图-控制器）设计模式中各司其职：HTML 负责数据的建模，CSS 独立处理界面的表现，而 JavaScript 则将前两者结合起来，负责实现功能逻辑和具体的代码控制。

## 2.2 项目的瀑布模型与增量式迭代开发模式

软件生命周期中的开发过程包括四个阶段：分析、设计、实现和测试。开发过程有几个模型。我们在这里讨论两个最常见的模型：瀑布模型和增量模型。

### 2.2.1 瀑布模型

瀑布模型是一种广泛采用的软件开发流程框架，如图 2.1 所示。该模型的特点是开发流程呈单向直线型进展，不允许反向进行。这表示每个开发阶段必须在进入下一阶段之前完全结束。

以此类推，项目必须在分析阶段彻底完成后，才能迈入设计阶段。相应地，设计阶段也需全部完成，开发团队才能启动实施阶段的工作。

瀑布模型有优点也有缺点。一个优点是每个阶段都在下一个阶段开始之前完成。例如，在设计阶段工作的小组确切地知道该做什么，因为他们有分析阶段的完整结果。测试阶段可以测试整个系统，因为整个开发中的系统已经准备好了。然而，瀑布模型的一个缺点是难以定位问题：如果在部分流程中存在问题，则必须检查整个流程<sup>[4]</sup>。

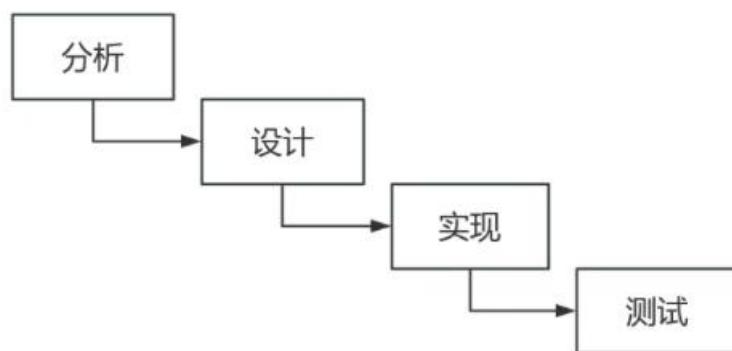


图 2.1 瀑布模型

### 2.2.2 迭代增量式模型

增量式迭代开发是一种灵活的软件开发方法，它将项目分解为多个小的、可

管理的增量阶段，每个阶段都包含需求、设计、实现、测试和部署等完整的开发周期。每个增量都是一个独立的、可运行的产品或功能，可以单独使用或与之前的增量集成。

这种方法的优势在于其适应性，允许团队根据用户反馈和市场变化快速调整开发方向。它强调持续集成和严格的质量保证，确保新功能能够无缝融入现有系统，同时保证最终产品的质量。

通过逐步交付价值，增量式迭代开发促进了团队的持续学习和改进，加速了产品上市时间，并提供了频繁的用户反馈。这种快速迭代有助于产品更好地满足用户需求，同时有效的风险管理减少了后期重构的风险，提高了客户满意度。

图 2.2 显示了增量模型的概念。

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添加了所有需要的功能<sup>[5]</sup>。

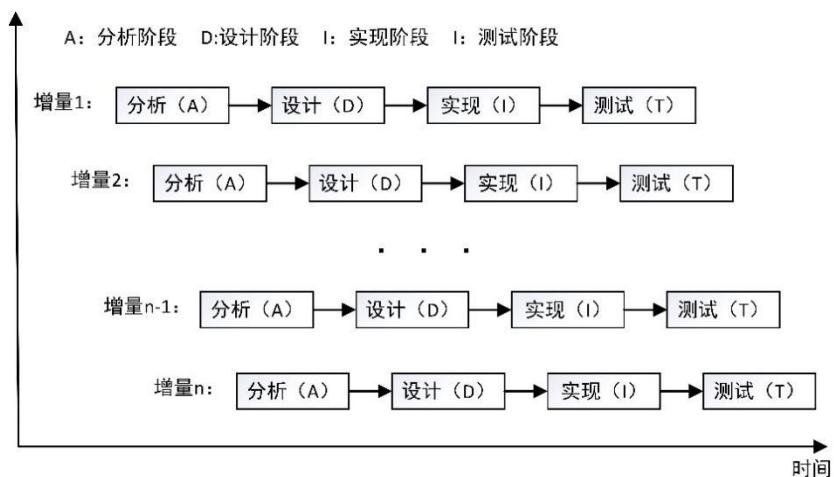


图 2.2 增量式迭代模型

### 3 用简洁的“三段论”方式开展内容设计

#### 3.1 分析和设计

##### 3.1.1 基本分析

本项目的界面设计遵循了用户体验设计的核心原则，以确保用户在使用过程中能够获得舒适、直观且高效的体验。设计上采用了“三段论”原则，即突出标题性信息、核心内容展示和底部附加信息，以引导用户的视觉焦点和满足其信息

需求。

在此设计理念下，我们的项目采用了清晰而合理的 UI 布局，以突出重点内容的同时兼顾细节展示。通过采用“三段论”的设计方法，我们不仅符合用户的视觉习惯，还能够提升用户体验。我们致力于精心设计每个部分，以打造既美观又实用的 UI 界面，让用户在使用过程中感到舒适和愉悦。同时，我们注重 UI 设计的一致性和可扩展性，保持整体风格统一的同时为未来可能的功能扩展留出了空间。通过模块化的设计方法，我们可以灵活地添加或调整功能模块，以满足不断变化的业务需求。这种设计理念将确保我们的项目在用户体验和功能扩展方面具备竞争优势，为用户提供更加愉悦和高效的体验。

### 3.1.2 相关图设计

总而言之，我们在 UI 设计上力求简约，直观和一致，同时兼顾内容的质量和细节的展示。我们坚信，通过精心的设计和持续的优化，能够为用户提供优质的使用体验，如下图 3.1 第一次增量迭代用例图所示。

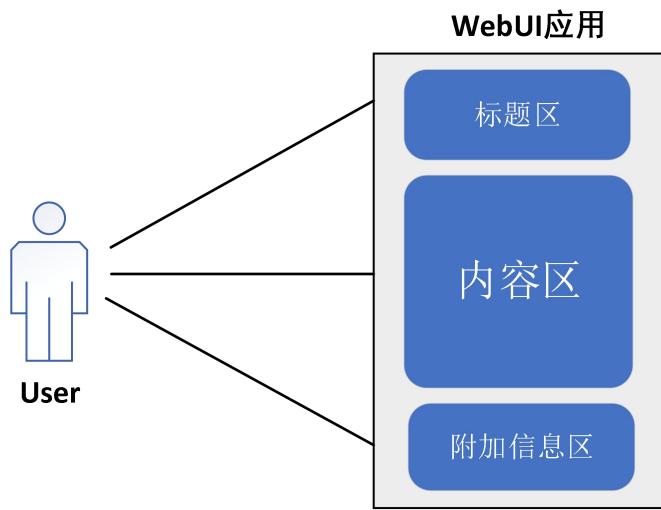


图 3.1 第一次迭代项目用例图

DOM（文档对象模型）树是指代表网页文档结构的一种树形数据结构。它由各种元素（例如标签、文本和属性）组成，这些元素按照其在文档中的嵌套关系形成了层级结构。DOM 树可以被视为网页文档的抽象表示，允许通过编程方式进行访问和操作。

DOM 树具有以下特点和作用：

(1) 层级结构：DOM 树以文档的根节点开始，然后分支到不同的子节点，形成了一个层级结构，这反映了 HTML 文档中各个元素之间的嵌套关系。

(2) 提供接口：作为网页内容的抽象表示，DOM 树为程序提供了访问和操作网页元素的接口。通过 JavaScript 等脚本语言，可以方便地使用 DOM API 对网页进行增删查改操作。

(3) 动态性：DOM 树不仅代表了页面的初始状态，还能够反映页面的动态变化。当网页发生交互或者通过脚本动态修改时，DOM 树会相应地更新。

(4) 作为基础：在 Web 开发中，DOM 树是构建网页交互和动态效果的基础。通过操作 DOM 树，开发人员可以实现对网页内容的动态改变，以及响应用户的交互行为。

因此为了方便项目的模块改动和查看各个元素之间的层级结构，我们先画出项目的相关 DOM 树。第一次迭代项目的 DOM 树如下图 3.2 所示。

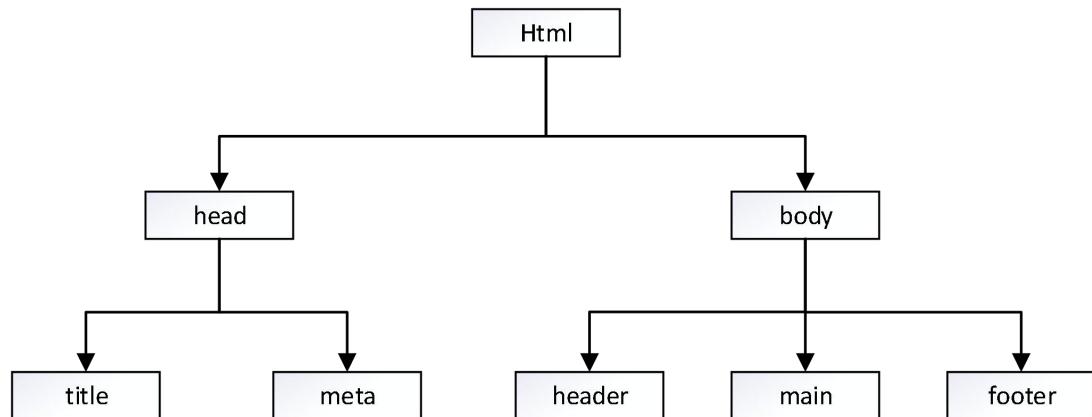


图 3.2 第一次迭代项目 DOM 树

## 3.2 项目的实现和编程

第一次提交，我们完成了三个部分。header 部分放了软件的标题，main 部分放了软件的内容，footer 部分放了软件的动态反馈。

3.2.1 HTML 代码编写如下：

```
<body>
  <header>
    《 基于 Web 客户端技术的个性化 UI 设计和实现 》
  </header>
  <main>
    我的主题内容：‘读好书、练思维、勤编程’
  </main>
  <footer>
    CopyRight 张梓鑫 江西科技师范大学 2024-2025
  </footer>
```

```
</footer>
迭代版本:
<a href="WebUI2.html" target="_blank">WebUI2</a>
<a href="WebUI3.html" target="_blank">WebUI3</a>
<a href="WebUI4.html" target="_blank">WebUI4</a>
<a href="WebUI5.html" target="_blank">WebUI5</a>
<a href="WebUI6.html" target="_blank">WebUI6</a>
</body>
```

3.2.2 CSS 代码编写如下：

```
<style>
body {
    margin:0;
    padding:0;
    font-family: Arial, sans-serif;
}
header, main, footer {
    margin: 10px;
    padding: 20px;
    text-align: center;
    border: 2px solid #3498db;
    background-color: #f0f0f0;
    position: relative; /* 添加相对定位 */
}
header {
    height: 100px;
    line-height: 100px;
    font-size: 24px;
    color: #3498db;
}
main {
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
    height: 300px;
    font-size: 20px;
    color: #555;
}
footer {
    height: 80px;
    line-height: 80px;
    font-size: 16px;
    color: #777;
}
```

```
a {  
    display: inline-block;  
    padding: 10px 20px;  
    margin: 5px;  
    color: #fff;  
    background-color: #3498db;  
    text-decoration: none;  
    border-radius: 5px;  
}  
</style>
```

### 3.3 项目的运行和测试

在浏览器中访问部署到 Git Hub 上面的网址，可以直接访问增量迭代第一版，效果图如下图 3.3 运行效果图。



图 3.3 第一次迭代界面

本章代码 URL 地址：<https://iasdkasdj.github.io/exp/WebUI1.html>

由于本项目的阶段性文件已经上传 Git Hub 网站，移动端用户可以通过扫描图 3.4 的二维码，运行测试本项目的第一次开发的阶段性效果。

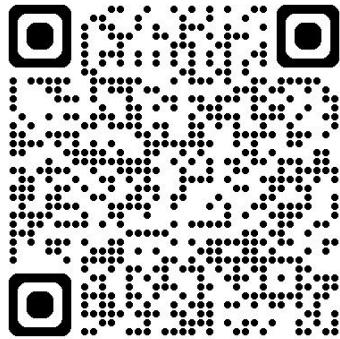


图 3.4 阶段 1 增量迭代二维码

### 3.4 项目的代码提交和版本管理

本项目的文件通过 gitBash 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：如开发者的姓名和电子邮件。

进入 gitBash 命令行后，按次序输入以下命令：

```
$ cd /  
$ mkdir webUI  
$ cd webUI  
$ git init  
$ git config user.name 江科师大张梓鑫  
$ git config user.email 2105058384@qq.com  
$ touch index.html
```

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

```
$ git add index.html  
$ git commit -m 第一次增量迭代，“三段论”式的内容设计概要开发，完成了软件开发环境的搭建，本次代码提交了 index.html 文件。把整个软件分成了上中下三个部分，header 里面放标题用于页面的头部的展示，main 里面放软件的内容，就是软件的主体内容与页面的展示，footer 里面放软件的动态反馈，也就是 javascript 的代码。
```

将代码用 Git Bash 对项目代码进行版本管理如下图 3.5 提交代码图所示。

```
[master (root-commit) c9634ec] 第一次增量迭代，“三段论”式的内容设计概要开发，完成了软件开发环境的搭建，本次代码提交了 index.html 文件。把整个软件分成了上中下三个部分，header 里面放标题用于页面的头部展示，main 里面放软件的内容，就是软件的主体内容与页面的展示，footer 里面放软件的动态反馈，也就是 javascript 的代码  
1 file changed, 71 insertions(+)  
create mode 100644 index.html
```

图 3.5 提交代码图

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。

```
$ git log
```

gitbash 反馈代码的仓库日志如下图 3.6 所示：

```
$ git log  
commit c9634ec9d0dab8ca00c282cf9980ee23b6d47315 (HEAD -> master)  
Author: 江科师大张梓鑫 <2105058384@qq.com>  
Date:   Wed Jun 12 18:27:36 2024 +0800
```

第一次增量迭代，“三段论”式的内容设计概要开发，完成了软件开发环境的搭建，本次代码提交了 `index.html` 文件。把整个软件分成了上中下三个部分，`header` 里面放标题用于页面的头部展示，`main` 里面放软件的内容，就是软件的主体内容与页面的展示，`footer` 里面放软件的动态反馈，也就是 `javascript` 的代码

图 3.6 gitbash 反馈图

## 4 移动互联时代的 UI 开发初步——窄屏终端的响应式

### 4.1 分析和设计

#### 4.1.1 基本分析

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<sup>[6]</sup>。

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<sup>[7]</sup>。

#### 4.1.2 相关图设计

本次更新迭代代码要实现在不同用户的不同设备上正确运行，主要针对移动端的用户，给用户最佳体验，因此要完成响应式设计并且适应窄屏的自动调节。如下图 4.1 第二次迭代项目用例图所示。

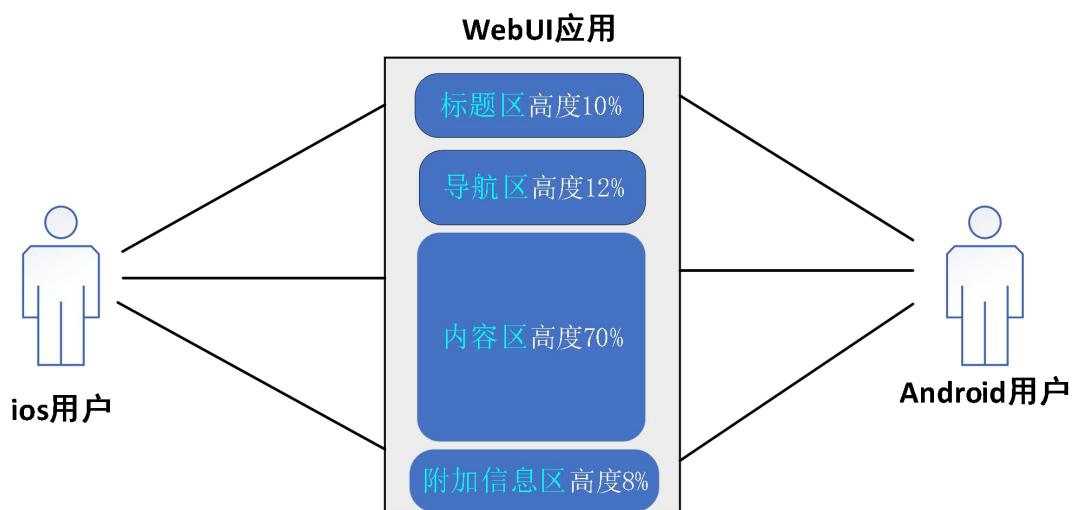


图 4.1 第二次迭代项目用例图

第二次迭代在第一次的基础上新增了一些元素，详细的元素见相关 DOM 树下图 4.2 所示。

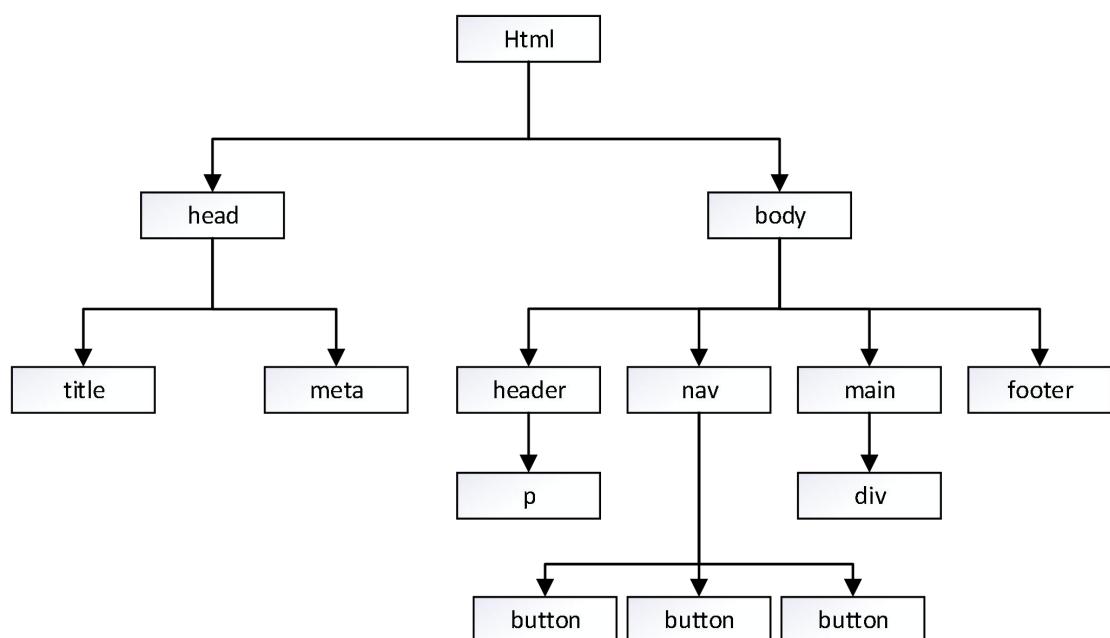


图 4.2 第二次迭代项目 DOM 树

## 4.2 项目的实现和编程

### 4.2.1 HTML 代码编写如下：

```

<body>
<header>
  <p id="book">JavaScript</p>
  
```

```
</header>
<nav>
    <button id="backwardBtn">后退</button>
    <button id="pauseBtn">暂停播放</button>
    <button id="forwardBtn">前进</button>
</nav>
<main id="main">
    <div id="bookface"></div>
</main>
<footer> CopyRight 张梓鑫 江西科技师范大学 2024--2025</footer>
</body>
```

#### 4.2.2 CSS 代码编写如下：

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。

```
<style>
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

body {
    font-family: 'Roboto', sans-serif;
    line-height: 1.6;
    background-color: #f0f0f0;
    color: #333;
}

header, nav, main, footer {
    background-color: #fff;
    margin: 4px;
    padding: 15px;
    border-radius: 8px;
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
}

header {
    height: 10%;
    text-align: center;
    background-color: #4CAF50;
    color: white;
}
```

```
header p {
    font-size: 1em;
    font-weight: bold;
}

nav {
    display: flex;
    justify-content: center;
    gap: 10px;
    height: 12%;
}

button {
    padding: 10px 20px;
    font-size: 0.8em;
    cursor: pointer;
    border: none;
    border-radius: 5px;
    background-color: #4CAF50;
    color: white;
    transition: background-color 0.3s ease, transform 0.3s ease;
}

button:hover {
    background-color: #45a049;
    transform: scale(1.05);
}

main {
    display: flex;
    justify-content: center;
    align-items: center;
    height: 70%;
}

#bookface {
    width: 80%;
    height: 100%;
    border: 1px solid #ddd;
    background-image: url('/lesson/NinjaJS.jpg');
    background-size: cover;
    background-position: center;
    border-radius: 8px;
    box-shadow: 0 2px 8px rgba(0, 0, 0, 0.1);
```

```
}

footer {
    text-align: center;
    font-size: 0.6em;
    color: #777;
    background-color: #4CAF50;
    color: white;
    padding: 15px;
    border-radius: 5px;
    height: 8%;
} </style>
```

#### 4.2.3 JavaScript 代码编写如下：

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。

```
<script>
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 + "px";
    document.body.style.height = UI.appHeight - 1 * baseFont + "px";
</script>
```

### 4.3 项目的运行和测试

移动互联时代的窄屏设计界面以 iPhone12 pro 和 Samsung Galaxy S8+为例，如下图 4.3 和 4.4 所示：

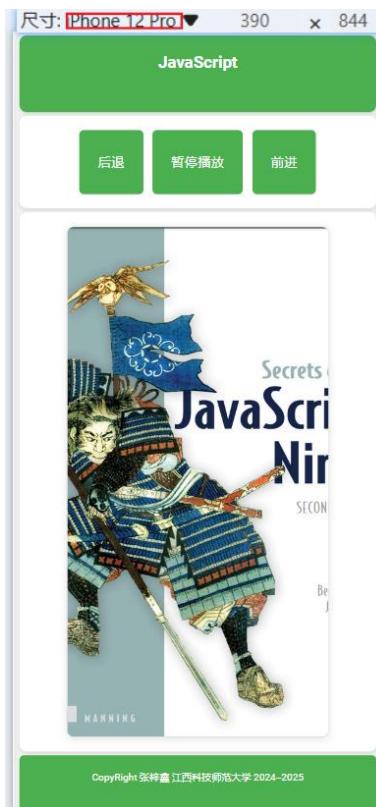


图 4.3 ios 移动终端

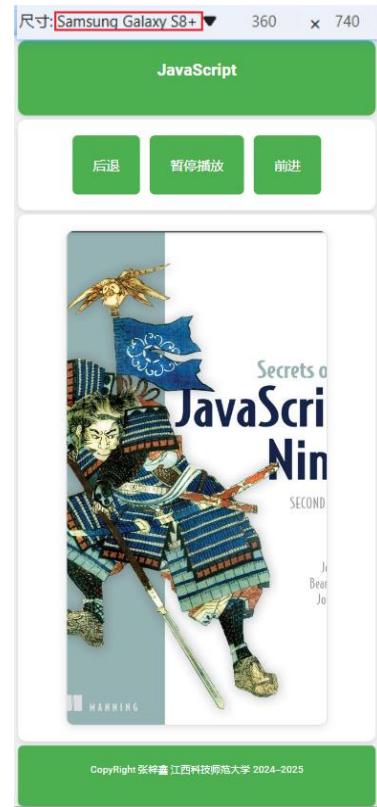


图 4.4 Android 移动终端

在实现窄屏代码的同时也保留了在现代浏览器中的代码，但是并没有实现完全的宽屏响应设计，还有一部分空间未使用，这是为了之后的功能添加，界面如下图 4.5 所示：



图 4.5 现代浏览器中的宽屏设计

本章代码 URL 地址：<https://iasdkasdj.github.io/exp/WebUI2.html>

手机端可以扫描下图 4.6 阶段 2 增量迭代二维码，访问响应式设计与适应窄

屏实现的文件。

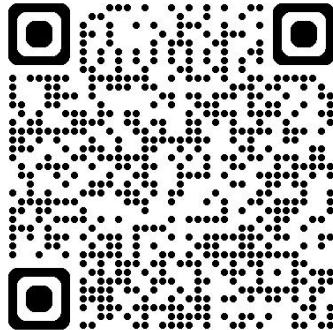


图 4.6 阶段 2 增量迭代图

#### 4.4 项目的代码提交和版本管理

作为项目的第二次迭代，在代码提交和版本管理环节，我们的目标是对 `index.html` 文件进行更新，同时将历史版本命名为 `WebUI1.html`，编写好代码，测试运行成功后，按次序输入以下命令：

```
$ git add index.html exp/WebUI1.html
$ git commit -m 第二次增量迭代——本次提交了'移动互联时代的响应式设计(窄屏)'

代码,本次代码运行了应用程序的响应式设计:1.声明了一个全局 UI 对象,用来存放需要使用的属性,2,为四个区域设置了占整个用户界面的比例(以高度为准)。3,用 Javascript 获取了应用屏幕的宽度和高度,并依据需要来进行计算基础字体的大小。4,在不同的区域根据内容的重要程度设定了该区域字体的相对大小。5.改变了颜色和对比色,对组区域的背景图做了优化外观的设计。6.总之,初步为用户的的应用程序做了响应式设计并部署了代码。
```

将代码用 Git Bash 对项目代码进行版本控制如下图 4.7 提交代码图所示：

```
[master eb5fe41] 第二次增量迭代——本次提交了'移动互联时代的响应式设计(窄屏)'

2 files changed, 142 insertions(+), 71 deletions(-)
copy index.html => exp/webui1.html (100%)
 rewrite index.html (70%)
```

图 4.7 提交代码图

gitbash 反馈代码的仓库日志如下图 4.8 所示。

```
commit eb5fe41a9a59f65e7df2faaaab9231ff0660ab64 (HEAD -> master)
Author: 江科师大张梓鑫 <2105058384@qq.com>
Date:   wed Jun 12 20:56:36 2024 +0800
```

第二次增量迭代—本次提交了'移动互联时代的响应式设计(窄屏)'代码,本次代码运行了应用程序的响应式设计:1.声明了一个全局UI对象,用来存放需要使用的属性,2,为四个区域设置了占整个用户界面的比例(以高度为准)。3,用Javascript获取了应用屏幕的宽度和高度,并依据需要来进行计算基础字体的大小。4,在不同的区域根据内容的重要程度设定了该区域字体的相对大小。5.改变了颜色和对比色,对组区域的背景图做了优化外观的设计。6.总之,初步为用户的的应用程序做了响应式设计并部署了代码。

图 4.8 gitbash 反馈图

## 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

#### 5.1.1 基本分析

响应式设计是一种网络设计和开发方法,旨在创建能够自适应不同屏幕尺寸和分辨率的网页或应用程序。这种设计方法确保用户无论使用什么设备,无论是桌面电脑、笔记本电脑、平板电脑还是智能手机都能获得良好的浏览体验。

本次代码进行了应用程序的响应式设计: 1.为四个区域设置了占整个用户界面的比例(以高度为准); 2.用 java script 获取了用户屏幕的宽度和高度, 并依据我们的需要进行了计算, 设置了基础字体的大小; 3.在不同区域根据内容的重要程度设定了该区域自己的相对大小; 4.改变了 UI 颜色和对比色, 对主区域的背景图做了优化外观的设计。

#### 5.1.2 相关图设计

总之,初步为用户的的应用程序做了响应式设计并部署了代码,初步实现了宽屏和窄屏通用的响应式设计。如下图 5.1 第三次增量迭代用例图所示。

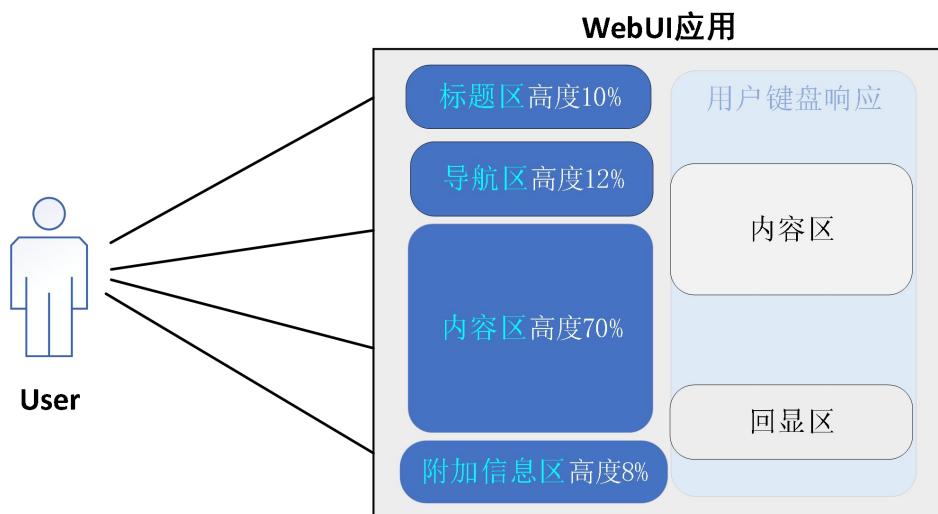


图 5.1 第三次增量迭代用例图

第三次迭代在第二次迭代的基础上又新增了一些元素，具体见下图 5.2 所示。

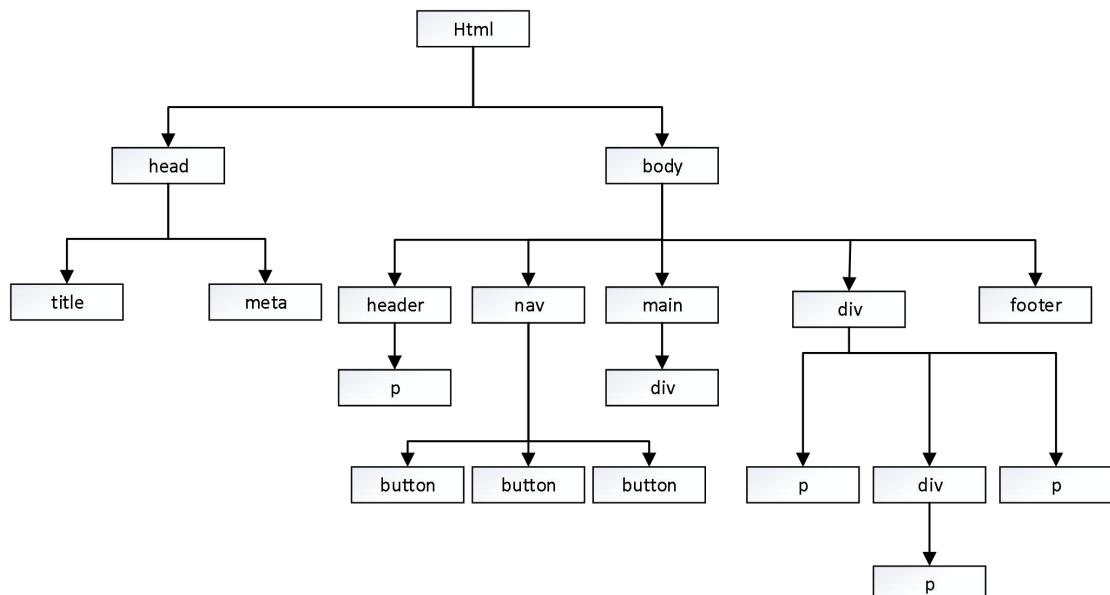


图 5.2 第三次迭代项目 DOM 树

## 5.2 项目的实现和编程

5.2.1 HTML 代码编写如下：

```

<body>
    <header>
        <p id="book">JavaScript</p>
    </header>
    <nav>

```

```
<button id="backwardBtn">后退</button>
<button id="pauseBtn">暂停播放</button>
<button id="forwardBtn">前进</button>
</nav>
<main id="main">
    <div id="bookface"></div>
</main>
<footer> CopyRight 张梓鑫 江西科技师范大学 2024--2025</footer>
<div id="aid">
    <p id="userResponseArea">用户键盘响应区</p>
    <div id="keyboard">
        <p id="displaytxt">
            &nbsp;
        </p>
    </div>
    <p id="displayCode">
        &nbsp;
    </p>
</div></body>
```

5.2.2 CSS 代码编写如下：

```
<style>
    * {
        margin: 0;
        padding: 0;
        box-sizing: border-box;
    }

    body {
        font-family: 'Roboto', sans-serif;
        line-height: 1.6;
        background-color: #f0f0f0;
        color: #333;
    }

    header, nav, main, footer {
        background-color: #fff;
        margin: 4px;
        padding: 15px;
        border-radius: 8px;
        box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
    }

    header {
        height: 10%;
```

```
    text-align: center;
    background-color: #4CAF50;
    color: white;
}

header p {
    font-size: 1em;
    font-weight: bold;
}

nav {
    display: flex;
    justify-content: center;
    gap: 10px;
    height: 12%;
}

button {
    padding: 10px 20px;
    font-size: 0.8em;
    cursor: pointer;
    border: none;
    border-radius: 5px;
    background-color: #4CAF50;
    color: white;
    transition: background-color 0.3s ease, transform 0.3s ease;
}

button:hover {
    background-color: #45a049;
    transform: scale(1.05);
}

main {
    display: flex;
    justify-content: center;
    align-items: center;
    height: 70%;
}

#bookface {
    width: 80%;
    height: 100%;
    border: 1px solid #ddd;
```

```
background-image: url('/lesson/NinjaJS.jpg');
background-size: cover;
background-position: center;
border-radius: 8px;
box-shadow: 0 2px 8px rgba(0, 0, 0, 0.1);
}

footer {
    text-align: center;
    font-size: 0.6em;
    color: #777;
    background-color: #4CAF50;
    color: white;
    padding: 15px;
    border-radius: 5px;
    height: 8%;
}
#aid {
    top: 0.3em;
    left: 650px;
    position: absolute;
    align-content: normal;
    border: 2px solid black;
    box-shadow: 2px 3px 5px 0px rgba(0, 0, 0, 0.2);
}

#userResponseArea {
    margin: 0.8em;
    font-size: 1.66em;
}

#keyboard {
    border: 3px solid black;
    margin-left: 10%;
    width: 80%;
    height: 60%;
    box-shadow: 2px 3px 5px 0px rgba(0, 0, 0, 0.4);
}

#displayCode {
    border: 3px solid black;
    margin-left: 10%;
    width: 80%;
    position: absolute;
```

```
        bottom: 0.3em;
        box-shadow: 2px 3px 5px 0px rgba(0, 0, 0, 0.4);
    }</style>
```

5.2.3 JavaScript 代码编写如下：

```
<script>
    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 + "px";
    document.body.style.height = UI.appHeight - 1 * baseFont + "px";
    UI.appHeight = window.innerHeight;
    if (window.innerWidth < 1000) {
        $("aid").style.display = 'none';
    }
    $("aid").style.width = window.innerWidth - UI.appWidth - baseFont
* 3 + 'px';
    $("aid").style.height = UI.appHeight - baseFont * 1 + 'px';
    $("body").addEventListener("keypress",function(ev){
        let key=ev.key;
        let code=ev.keyCode;
        let string="键的代码是: ";
        console.log(key);
        $("displayCode").innerText=key+string+code;
    })</script>
```

### 5.3 项目的运行和测试

移动互联时代的窄屏设计界面以 iPhone12 pro 和 Samsung Galaxy S8+为例，  
如下图 5.3 和 5.4 所示：

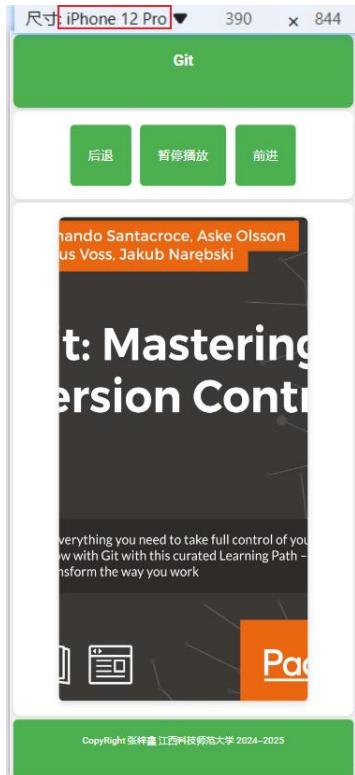


图 5.3 ios 移动终端

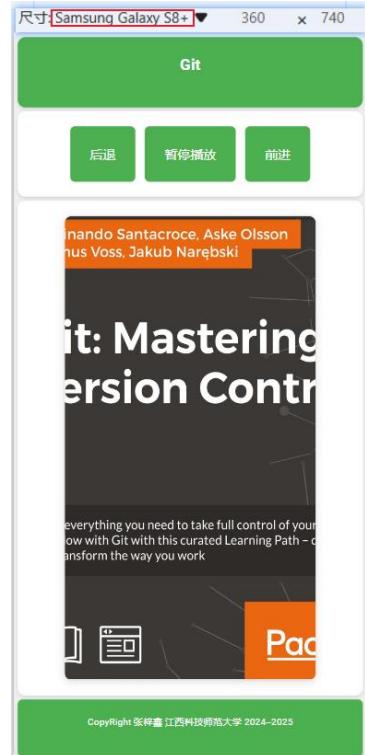


图 5.4 Android 移动终端

在浏览器中访问部署到 GitHub 上面的网址，可以直接访问增量迭代第三版，效果图如下图 5.5 正常电脑运行效果图所示。



图 5.5 PC 正常运行效果图

在图 5.5 中可知，系统已经满足了前面需求分析的要求，第三次增量迭代在第二次增量迭代的基础上，需添加一个用户的键盘响应区，这个响应区可以实现监控

电脑端的键盘输入信息，并在特定的响应区里面显示用户的键盘输入信息，但这个功能需要考虑用户的设备宽度，不然页面的展示效果不佳，所以需要添加判断条件来保证页面的装饰效果。如果满足条件就显示用户键盘响应区，不然就不显示以保证页面的整体效果。这个条件就是当用户的设备宽度超过 900 就会显示用户键盘显示区，在图 5.3 中，在 iPhone 12 Pro 设备上面的运行结果就没有显示用户键盘显示区，这是因为此设备的宽度没有超过 900，用户键盘显示区就会被 JavaScript 动态的隐藏起来，以保证用户良好的体验。如果不加这条判断就会显示异常如下图 5.6 不良效果图所示。

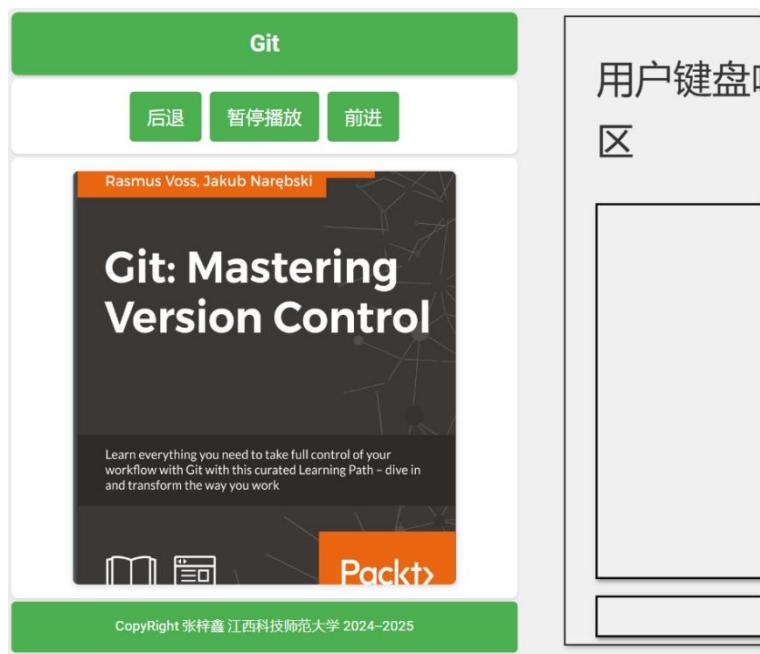


图 5.6 不良效果图

在图 5.6 不良效果图上面可以看出，将 JavaScript 里面的动态判断用户设备宽度大小的条件去掉后就会显示很拥挤，减少用户使用本项目的兴趣。在原来的逻辑判断中是用户的设备宽度超过 900 才会显示，现在没有限制了，页面还是会显示用户键盘响应区。这也从侧面表示出程序员需要考虑用户的大部分需求，以保证用户拥有较大兴趣使用本项目。

本章代码 URL 地址：<https://iasdkasdj.github.io/exp/WebUI3.html>

手机端可以扫描下图 5.7 阶段 3 增量迭代二维码，访问宽屏和窄屏通用的响应式设计文件。

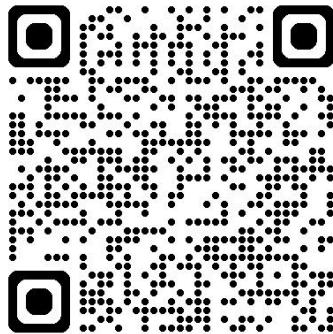


图 5.7 阶段 3 增量迭代二维码

## 5.4 项目的代码提交和版本管理

作为项目的第三次迭代，在代码提交和版本管理环节，我们的目标是对 `index.html` 文件进行更新，同时将这次历史版本命名为 `WebUI2.html`，编写好代码，测试运行成功后，按次序输入以下命令：

```
$ git add index.html exp/WebUI2.html  
$ git commit -m 第三次增量迭代——本次提交了‘响应式设计(宽屏和窄屏)’代码，本次改进了页面对宽屏与窄屏的兼容。可以让移动设备的屏幕宽度和高度，与 window 对象的 inner width 和 inner Height 精确对应用户页面实现了响应式设计，对用户设备的页面考虑了不同的大小并思考如何兼容设备页面在页面上准确显示。进行了鼠标与键盘的跟踪，并在页面实时显示出来。用户错误使用系统会在浏览器的控制台抛出具体的错误内容。
```

先将代码用 Git Bash 对项目代码进行版本控制如下图 5.8 提交代码图所示：

```
$ git commit -m "本次提交了‘响应式设计(宽屏和窄屏)’代码，本次改进了页面对宽屏与窄屏的兼容。可以让移动设备的屏幕宽度和高度，与 window 对象的 inner width 和 inner Height 精确对应用户页面实现了响应式设计，对用户设备的页面考虑了不同的大小并思考如何兼容设备页面在页面上准确显示。进行了鼠标与键盘的跟踪，并在页面实时显示出来。用户错误使用系统会在浏览器的控制台抛出具体的错误内容."  
[master 8bfcae] 本次提交了‘响应式设计(宽屏和窄屏)’代码，本次改进了页面对宽屏与窄屏的兼容。可以让移动设备的屏幕宽度和高度，与 window 对象的 inner width 和 inner Height 精确对应用户页面实现了响应式设计，对用户设备的页面考虑了不同的大小并思考如何兼容设备页面在页面上准确显示。进行了鼠标与键盘的跟踪，并在页面实时显示出来。用户错误使用系统会在浏览器的控制台抛出具体的错误内容。  
2 files changed, 198 insertions(+), 5 deletions(-)  
create mode 100644 exp/webui2.html
```

图 5.8 提交代码图

gitbash 反馈代码的仓库日志如下图 5.9 所示：

```
commit 8bfcae300831040a89b555e3631e2285bc310b (HEAD -> master)
```

```
Author: 江科师大张梓鑫 <2105058384@qq.com>
```

```
Date: Thu Jun 13 11:51:38 2024 +0800
```

本次提交了‘响应式设计(宽屏和窄屏)’代码，本次改进了页面对宽屏与窄屏的兼容。可以让移动设备的屏幕宽度和高度，与window对象的inner width和inner Height精确对应用户页面实现了响应式设计，对用户设备的页面考虑了不同的大小并思考如何兼容设备页面在页面上准确显示。进行了鼠标与键盘的跟踪，并在页面实时显示出来。用户错误使用系统会在浏览器的控制台抛出具体的错误内容。

图 5.9 gitbash 反馈图

## 6 个性化 UI 设计中鼠标移动点击和封面轮播模型

### 6.1 分析和设计

#### 6.1.1 基本分析

本次代码进行了应用程序的功能设计，希望给用户一些交互式的体验，而不是简单的看一张图片，尝试对鼠标设计 UI 控制：在用户鼠标响应区域中，当用户点击鼠标时，记录下并显示鼠标当前的坐标信息；当用户移动鼠标时，持续记录并显示鼠标在移动过程中的坐标信息；而当用户将鼠标移出响应区的时候，显示鼠标已经离开。此外对鼠标内容区进行了优化，实现了图书的封面的更换和轮播，使得用户可以方便的切换相关图书的封面。

#### 6.1.2 相关图设计

如下图 6.1 第四次增量迭代用例图所示，比第三次迭代在内容区界面进行了优化，新增了图片轮播，增加的内容主要是在用户交互上，页面布局并没有什么大的改变。

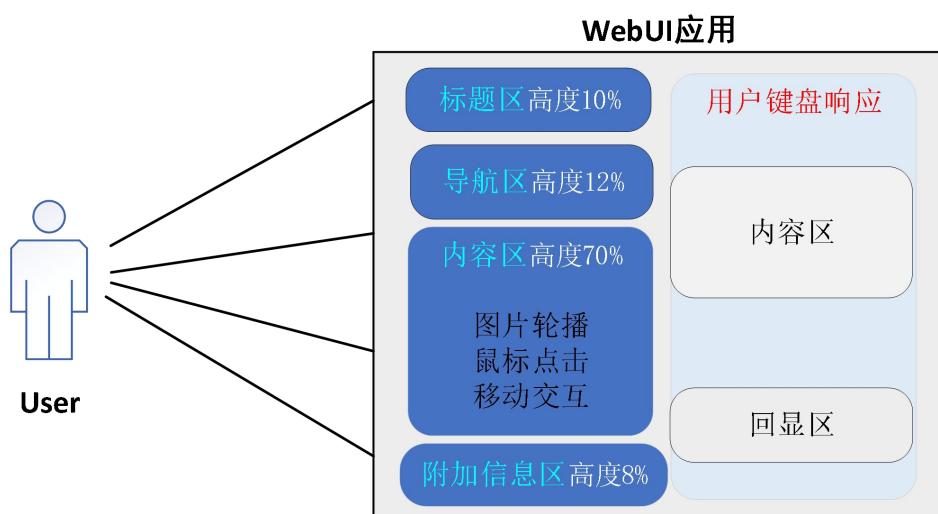


图 6.1 第四次迭代增量用例图

第四次迭代的 DOM 树如下图 6.2 所示，因为这次主要是针对用户交互的迭代，因此在页面布局上并没有什么变化，元素与第三次迭代保持一致。

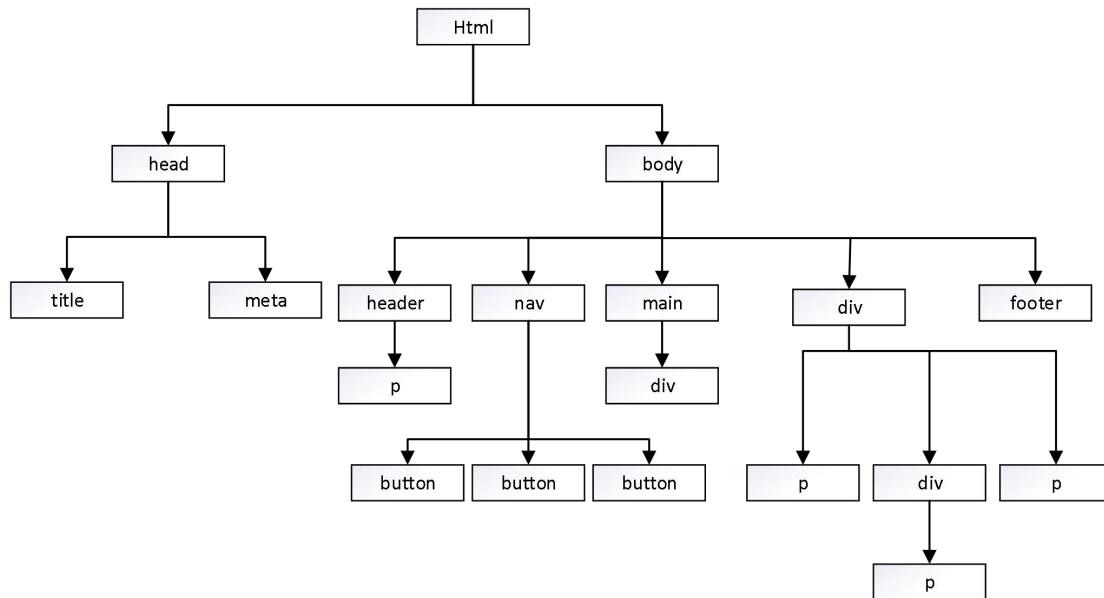


图 6.2 第四次迭代项目 DOM 树

## 6.2 项目的实现和编程

6.2.1 HTML 代码编写如下：

```
<body>
    <header>
        <p id="book">
            计算机相关书籍展示
        </p>
    </header>
    <nav>
        <button id="backwardBtn">后退</button>
        <button id="pauseBtn">暂停播放</button>
        <button id="forwardBtn">前进</button>
    </nav>

    <main id="main">
        <div id="bookface">
            书的封面图
        </div>
    </main>

    <footer>
```

CopyRight from 张梓鑫 江西科技师范大学 2024--2025

```
</footer>
<div id="aid">
    <p id="userResponseArea">用户键盘响应区</p>
    <div id="keyboard">
        <p id="displaytxt">
            &nbsp;
        </p>
    </div>
    <p id="displayCode">
        &nbsp;
    </p>
</div></body>
```

6.2.2 CSS 代码编写如下：

```
<style>
    * {
        margin: 0;
        padding: 0;
        box-sizing: border-box;
    }

    body {
        font-family: 'Roboto', sans-serif;
        line-height: 1.6;
        background-color: #f0f0f0;
        color: #333;
    }

    header, nav, main, footer {
        background-color: #fff;
        margin: 4px;
        padding: 15px;
        border-radius: 8px;
        box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
    }

    header {
        height: 10%;
        text-align: center;
        background-color: #4CAF50;
        color: white;
    }
</style>
```

```
header p {
    font-size: 1em;
    font-weight: bold;
}

nav {
    display: flex;
    justify-content: center;
    gap: 10px;
    height: 12%;
}

button {
    padding: 10px 20px;
    font-size: 0.8em;
    cursor: pointer;
    border: none;
    border-radius: 5px;
    background-color: #4CAF50;
    color: white;
    transition: background-color 0.3s ease, transform 0.3s ease;
}

button:hover {
    background-color: #45a049;
    transform: scale(1.05);
}

main {
    display: flex;
    justify-content: center;
    align-items: center;
    height: 70%;
}

#bookface {
    position: relative;
    width: 80%;
    height: 100%;
    border: 1px solid #ddd;
    background-image: url();
    background-size: cover;
    background-position: center;
```

```
border-radius: 8px;
box-shadow: 0 2px 8px rgba(0, 0, 0, 0.1);
}

footer {
    text-align: center;
    font-size: 0.6em;
    color: #777;
    background-color: #4CAF50;
    color: white;
    padding: 15px;
    border-radius: 5px;
    height: 8%;
}
#aid {
    top: 0.3em;
    left: 650px;
    position: absolute;
    align-content: normal;
    border: 2px solid black;
    box-shadow: 2px 3px 5px 0px rgba(0, 0, 0, 0.2);
}

#userResponseArea {
    margin: 0.8em;
    font-size: 1.66em;
}

#keyboard {
    border: 3px solid black;
    margin-left: 10%;
    width: 80%;
    height: 60%;
    box-shadow: 2px 3px 5px 0px rgba(0, 0, 0, 0.4);
}

#displayCode {
    border: 3px solid black;
    margin-left: 10%;
    width: 80%;
    position: absolute;
    bottom: 0.3em;
    box-shadow: 2px 3px 5px 0px rgba(0, 0, 0, 0.4);
}    </style>
```

### 6.2.3 JavaScript 代码编写如下：

```
<script>
    const images = ['../lesson/bitCoin.jpg', '../lesson/canvas.jpg',
'../lesson/NinjaJS.jpg','../lesson/Git.jpg","../lesson/CS.jpg","../less
on/CSS.jpg"];
let currentIndex = 0;
let intervalId = null;
let isPlaying = true; // 标记自动播放状态， 默认为 true 表示正在播放
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 + "px";
    document.body.style.height = UI.appHeight - 1 * baseFont + "px";
    UI.appHeight = window.innerHeight;
    if (window.innerWidth < 1000) {
        $("aid").style.display = 'none';
    }
    $("aid").style.width = window.innerWidth - UI.appWidth - baseFont
* 3 + 'px';
    $("aid").style.height = UI.appHeight - baseFont * 1 + 'px';

//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").innerText= "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").innerText= "鼠标正在移动，坐标为: +"(+x+","+y+")";
})

$("bookface").addEventListener("mouseout",function(ev){
    // console.log(ev);
    $("bookface").innerText="鼠标已经离开";
})

$("body").addEventListener("keypress",function(ev){
    let key=ev.key;
    let code=ev.keyCode;
    let string="键的代码是: ";
    console.log(key);
    $("displayCode").innerText=key+string+code;
})

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 $
// 更新显示的图片
function updateImage() {
    document.getElementById('bookface').style.backgroundImage =
`url('${images[currentIndex]}`);
}

// 点击前进按钮
document.getElementById('forwardBtn').addEventListener('click',
function() {
    currentIndex = (currentIndex + 1) % images.length;
    updateImage();
}

```

```
});

// 点击后退按钮
document.getElementById('backwardBtn').addEventListener('click',
function() {
    currentIndex = (currentIndex - 1 + images.length) % images.length;
    updateImage();
});

// 下一张图片
function nextImage() {
    currentIndex = (currentIndex + 1) % images.length;
    updateImage();
}

// 初始化显示第一张图片
updateImage();

// 自动播放
intervalId = setInterval(nextImage, 3000);

// 停止自动播放
document.getElementById('pauseBtn').addEventListener('click', function()
{
    clearInterval(intervalId);
});

// 切换自动播放状态
document.getElementById('pauseBtn').addEventListener('click', function()
{
    const pauseBtn = document.getElementById('pauseBtn');
    if (isPlaying) {
        clearInterval(intervalId); // 停止自动播放
        pauseBtn.textContent = '开始播放'; // 更改按钮文本
    } else {
        intervalId = setInterval(nextImage, 3000); // 开始自动播放
        pauseBtn.textContent = '暂停播放'; // 更改按钮文本
    }
    isPlaying = !isPlaying; // 切换播放状态
}); </script>
```

### 6.3 项目的运行和测试

移动窄屏界面以 iPhone12 pro 和 Samsung Galaxy S8+为例，见下图 6.3 和 6.4。

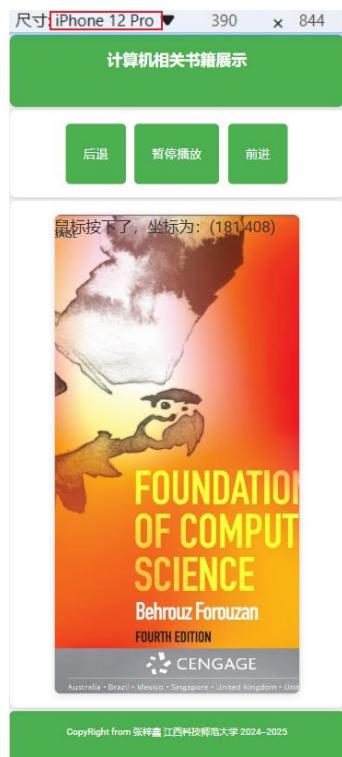


图 6.3 iOS 移动终端

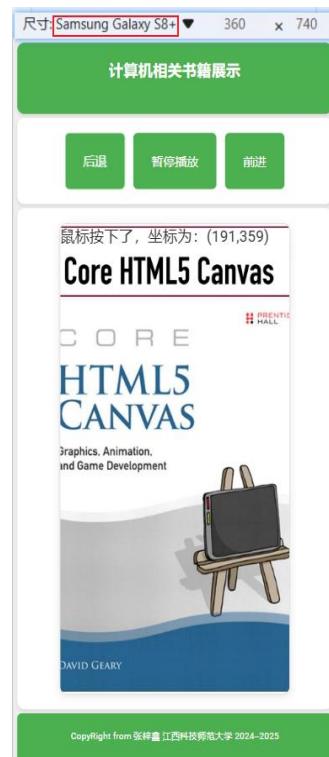


图 6.4 Android 移动终端

首先我们可以清楚地看到鼠标在响应区的操作提示，会显示当前操作和坐标，另外我们会发现内容区书的封面在变化，这就是图片轮播，此外用户还可以点击后退前进按钮实现图片的更换。

与移动终端响应类似，PC 终端的用户交互包括鼠标移动和点击等，具体见下图 6.5 所示。



图 6.5 PC 界面

本章代码 URL 地址: <https://iasdkasdj.github.io/exp/WebUI4.html>

手机端可以扫描下图 6.6 阶段 4 增量迭代二维码，访问个性化 UI 设计的文件。

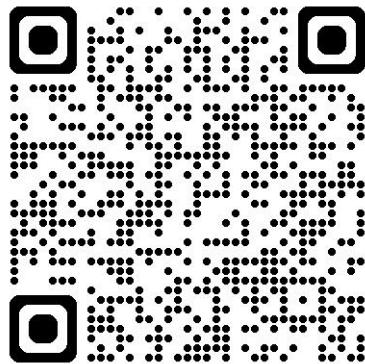


图 6.6 阶段 4 迭代增量二维码

## 6.4 项目的代码提交和版本管理

作为项目的第四次迭代，在代码提交和版本管理环节，我们的目标是对 index.html 文件进行更新，同时将这次历史版本命名为 WebUI3.html，编写好代码，测试运行成功后，按次序输入以下命令：

```
$ git add index.html exp/WebUI3.html  
$ git commit -m 第四次增量迭代——本次提交了‘UI设计之鼠标模型的移动、点击和封面轮播’代码，本次添加了内容区的移动和点击响应，可以在计算机上面利用鼠标进行点击和移动，此外还在内容区增加了书封面的轮播和封面切换功能，方便用户进行交互。
```

先将代码用 Git Bash 对项目代码进行版本控制如下图 6.7 提交代码图所示：

```
$ git commit -m "第四次增量迭代——本次提交了‘UI设计之鼠标模型的移动、点击和封面轮播’代码，本次添加了内容区的移动和点击响应，可以在计算机上面利用鼠标进行点击和移动，此外还在内容区增加了书封面的轮播和封面切换功能，方便用户进行交互。"  
[master 3050837] 第四次增量迭代——本次提交了‘UI设计之鼠标模型的移动、点击和封面轮播’代码，本次添加了内容区的移动和点击响应，可以在计算机上面利用鼠标进行点击和移动，此外还在内容区增加了书封面的轮播和封面切换功能，方便用户进行交互。  
2 files changed, 320 insertions(+), 35 deletions(-)  
create mode 100644 exp/WebUI3.html
```

图 6.7 提交代码图

gitbash 反馈代码的仓库日志如下图 6.8 所示：

```
commit 305083769b1b65fb3d56ec6b05ceef7968df5565 (HEAD -> master)
Author: 江科师大张梓鑫 <2105058384@qq.com>
Date: Thu Jun 13 16:04:27 2024 +0800
```

第四次增量迭代—本次提交了‘UI设计之鼠标模型的移动、点击和封面轮播’代码，本次添加了内容区的移动和点击响应，可以在计算机上面利用鼠标进行点击和移动，此外还在内容区增加了书封面的轮播和封面切换功能，方便用户进行交互。

图 6.8 gitbash 反馈图

## 7 通用的 UI 设计，同时为触屏和鼠标拖拽移动行为建模

### 7.1 分析和设计

#### 7.1.1 基本分析

本次代码在第四次迭代的基础上进行优化，第四次迭代是完成了对鼠标的移动和点击交互的反馈，发现该设计并不满足对软件设计的需求，于是做出了更加合适的改进，在原有代码的基础上实现了更加完善的对用户鼠标点击事件的响应，主要是将系统监听鼠标移动事件改为了系统监听用户长按鼠标拖动事件，当用户点击鼠标并未拖动或并未产生有效拖动距离时，显示本次算是无效拖动；而当用户产生有效拖动距离时，以 px 为单位计算出该距离并显示这次是有效拖动以及拖动的有效距离。

同时仍然保持 PC 和移动端的双适应，原理上和鼠标拖动事件类似，只是需要增加一个对用户所使用设备的判断，以此来确定时进行鼠标事件监听还是触屏事件监听。

#### 7.1.2 相关图设计

第五次迭代主要是鼠标和触屏的行为建模，因此页面并未有什么大变化。如下图 7.1 第五次增量迭代用例图所示。

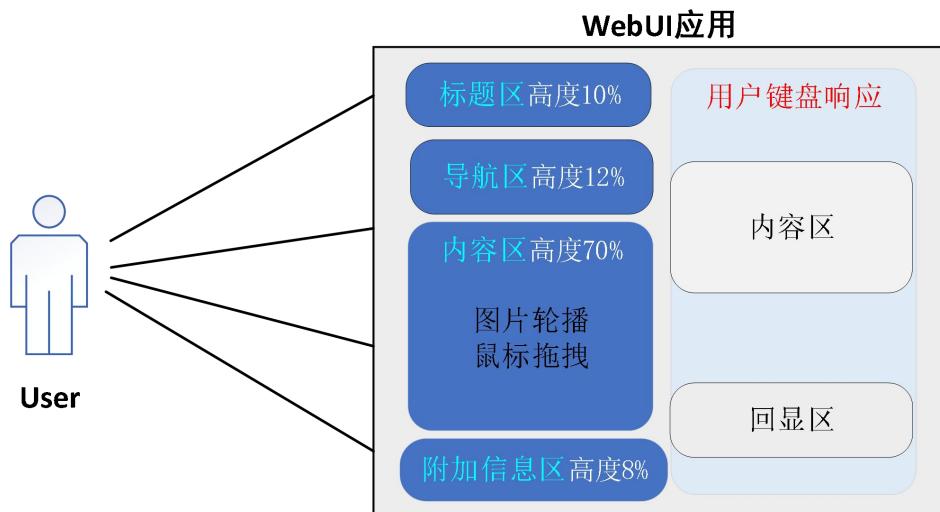


图 7.1 第五次增量迭代用例图

第五次迭代的 DOM 树如下图 7.2 所示，因为这次主要是针对用户交互的迭代，因此在页面布局上并没有什么变化，元素与第四次迭代保持一致。

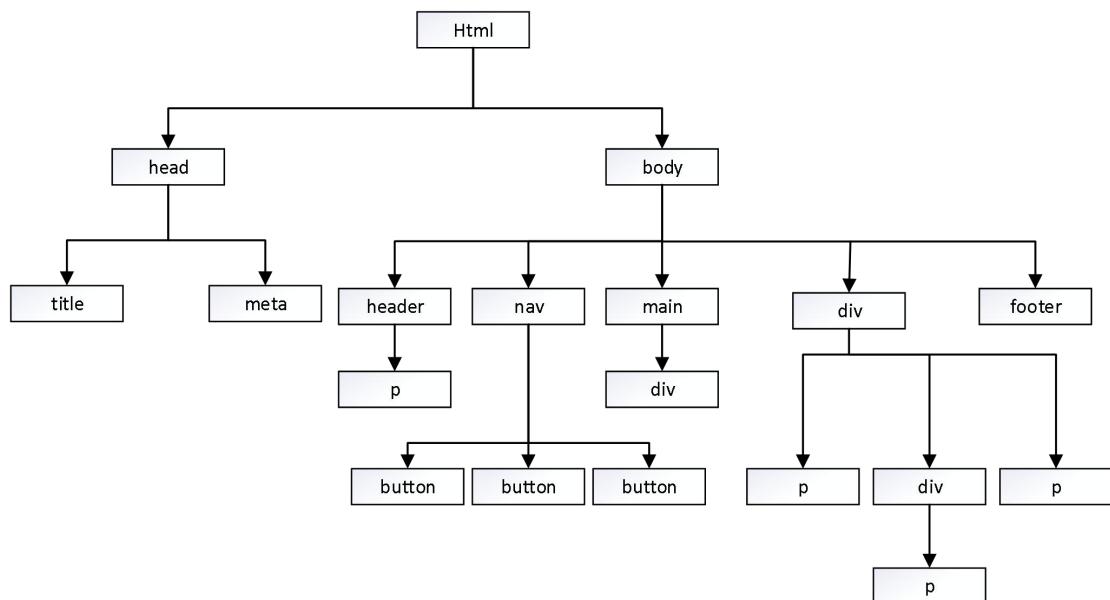


图 7.2 第五次迭代项目 DOM 树

## 7.2 项目的实现和编程

### 7.2.1 HTML 代码编写如下：

```
<body>
  <header>
    <p id="book">
      计算机相关书籍展示
    </p>
  </header>
</body>
```

```

</p>
</header>
<nav>
    <button id="backwardBtn">后退</button>
    <button id="pauseBtn">暂停播放</button>
    <button id="forwardBtn">前进</button>
</nav>
<main id="main">
    <div id="bookface">
        这是书的封面图<br>
        在此对象范围拖动鼠标
    </div>
</main>
<footer>
    CopyRight from 张梓鑫 江西科技师范大学 2024--2025
</footer>
<div id="aid">
    <p id="userResponseArea">用户键盘响应区</p>
    <div id="keyboard">
        <p id="displaytxt">
            &ampnbsp
        </p>
    </div>
    <p id="displayCode">
        &ampnbsp
    </p>
</div></body>

```

7.2.2 CSS 代码编写如下：

```

<style>
    * {
        margin: 0;
        padding: 0;
        box-sizing: border-box;
    }
    body {
        font-family: 'Roboto', sans-serif;
        line-height: 1.6;
        background-color: #f0f0f0;
        color: #333;
    }
    header, nav, main, footer {
        background-color: #fff;
        margin: 4px;
        padding: 15px;
    }

```

```
border-radius: 8px;
box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
}
header {
    height: 10%;
    text-align: center;
    background-color: #4CAF50;
    color: white;
}
header p {
    font-size: 1em;
    font-weight: bold;
}
nav {
    text-align: center;
    height: 12%;
}
button {
    padding: 10px 20px;
    font-size: 0.8em;
    cursor: pointer;
    border: none;
    border-radius: 5px;
    background-color: #4CAF50;
    color: white;
    transition: background-color 0.3s ease, transform 0.3s ease;
}
button:hover {
    background-color: #45a049;
    transform: scale(1.05);
}
main {
    text-align: center;
    height: 70%;
}
#bookface {
    text-align: center;
    left: 41px;
    position: relative;
    width: 80%;
    height: 100%;
    border: 1px solid #ddd;
    background-image: url();
    background-size: cover;
```

```
background-position: center;
border-radius: 8px;
box-shadow: 0 2px 8px rgba(0, 0, 0, 0.1);
user-select: none;
-moz-user-select: none;
-webkit-user-select: none;
-ms-user-select: none;
}
footer {
    text-align: center;
    font-size: 0.6em;
    color: #777;
    background-color: #4CAF50;
    color: white;
    padding: 15px;
    border-radius: 5px;
    height: 8%;
}
#aid {
    top: 0.3em;
    left: 650px;
    position: absolute;
    align-content: normal;
    border: 2px solid black;
    box-shadow: 2px 3px 5px 0px rgba(0, 0, 0, 0.2);
}
#userResponseArea {
    margin: 0.8em;
    font-size: 1.66em;
}
#keyboard {
    border: 3px solid black;
    margin-left: 10%;
    width: 80%;
    height: 60%;
    box-shadow: 2px 3px 5px 0px rgba(0, 0, 0, 0.4);
}
#displayCode {
    border: 3px solid black;
    margin-left: 10%;
    width: 80%;
```

```
        position: absolute;
        bottom: 0.3em;
        box-shadow: 2px 3px 5px 0px rgba(0, 0, 0, 0.4);
    }</style>
```

7.2.3 JavaScript 代码编写如下：

```
<script>
    const images = ['./lesson/cssAnimation.jpg',
'./lesson/linuxCMD.jpg', './lesson/UML.jpg'];
let currentIndex = 0;
let intervalId = null;
let isPlaying = true; // 标记自动播放状态，默认为 true 表示正在播放
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 + "px";
    document.body.style.height = UI.appHeight - 1 * baseFont + "px";
    UI.appHeight = window.innerHeight;
    if (window.innerWidth < 1000) {
        $("aid").style.display = 'none';
    }
    $("aid").style.width = window.innerWidth - UI.appWidth - baseFont
* 3 + 'px';
    $("aid").style.height = UI.appHeight - baseFont * 1 + 'px';

//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()
    //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%' ;
        }
    }
};

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' ;
    }
}
};

//下面是用户交互的代码，交互的设备包括鼠标，键盘，触屏。
$("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);}

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 $

// 更新显示的图片
function updateImage() {
    document.getElementById('bookface').style.backgroundImage =
`url('${images[currentIndex]}`);
}

```

```
// 点击前进按钮
document.getElementById('forwardBtn').addEventListener('click',
function() {
    currentIndex = (currentIndex + 1) % images.length;
    updateImage();
});

// 点击后退按钮
document.getElementById('backwardBtn').addEventListener('click',
function() {
    currentIndex = (currentIndex - 1 + images.length) % images.length;
    updateImage();
});

// 下一张图片
function nextImage() {
    currentIndex = (currentIndex + 1) % images.length;
    updateImage();
}

// 初始化显示第一张图片
updateImage();

// 自动播放
intervalId = setInterval(nextImage, 3000);

// 停止自动播放
document.getElementById('pauseBtn').addEventListener('click', function()
{
    clearInterval(intervalId);
});

// 切换自动播放状态
document.getElementById('pauseBtn').addEventListener('click', function()
{
    const pauseBtn = document.getElementById('pauseBtn');
    if (isPlaying) {
        clearInterval(intervalId); // 停止自动播放
        pauseBtn.textContent = '开始播放'; // 更改按钮文本
    } else {
        intervalId = setInterval(nextImage, 3000); // 开始自动播放
        pauseBtn.textContent = '暂停播放'; // 更改按钮文本
    }
    isPlaying = !isPlaying; // 切换播放状态
});    </script>
```

## 7.3 项目的运行和测试

移动窄屏界面以 iPhone12 pro 和 Samsung Galaxy S8+为例,见下图 7.3 和 7.4。

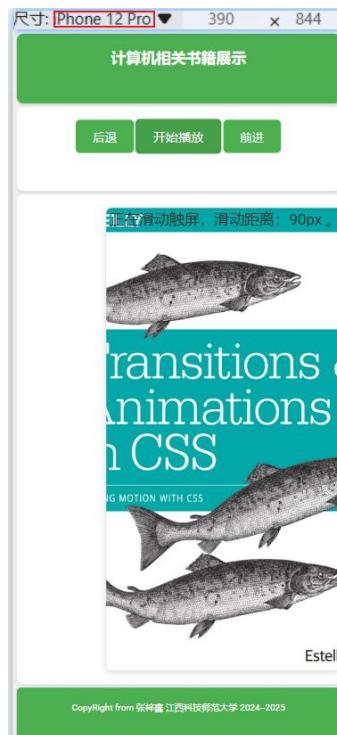


图 7.3 ios 移动终端

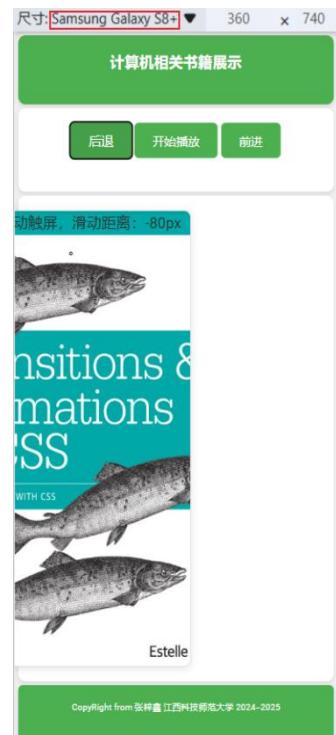


图 7.4 Android 移动终端

我们可以清楚的看到触屏拖拽时候的内容区响应内容，会提示滑动距离。

与移动终端响应类似，PC 终端的用户具体见下图 7.5 所示。



图 7.5 PC 界面

本章代码 URL 地址: <https://iasdkasdj.github.io/exp/WebUI5.html>

手机端可以扫描下图 7.6 阶段 5 增量迭代二维码，访问通用的 UI 设计的文件。

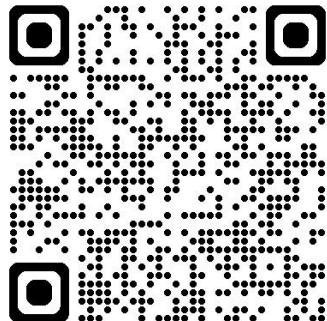


图 7.6 阶段 5 增量迭代二维码

## 7.4 项目的代码提交和版本管理

作为项目的第五次迭代，在代码提交和版本管理环节，我们的目标是对 `index.html` 文件进行更新，同时将这次历史版本命名为 `WebUI4.html`，编写好代码，测试运行成功后，按次序输入以下命令：

```
$ git add index.html exp/WebUI4.html
$ git commit -m 第五次增量迭代——本次提交了‘UI设计之鼠标和触屏拖拽’代码，本次添加了内容区的拖拽响应，可以在计算机上面利用鼠标进行内容区的拖拽，本次将代JS代码整合在一个全局对象 Pointer 中，这个对象中存储了本程序的所有异步代码，这个对象在本程序的JS里面可以任意调用，简化了代码的复杂度，优化了代码的重复使用性。这样可以减小浏览器的响应时间，提供用户更好的使用体验。
```

将代码用 Git Bash 对项目代码进行版本控制如下图 7.7 提交代码图所示：

```
$ git commit -m "第五次增量迭代——本次提交了‘UI设计之鼠标和触屏拖拽’代码，本次添加了内容区的拖拽响应，可以在计算机上面利用鼠标进行内容区的拖拽，本次将代JS代码整合在一个全局对象 Pointer 中，这个对象中存储了本程序的所有异步代码，这个对象在本程序的JS里面可以任意调用，简化了代码的复杂度，优化了代码的重复使用性。这样可以减小浏览器的响应时间，提供用户更好的使用体验。"
[master 898dc8b] 第五次增量迭代——本次提交了‘UI设计之鼠标和触屏拖拽’代码，本次添加了内容区的拖拽响应，可以在计算机上面利用鼠标进行内容区的拖拽，本次将代JS代码整合在一个全局对象 Pointer 中，这个对象中存储了本程序的所有异步代码，这个对象在本程序的JS里面可以任意调用，简化了代码的复杂度，优化了代码的重复使用性。这样可以减小浏览器的响应时间，提供用户更好的使用体验。
 2 files changed, 380 insertions(+), 49 deletions(-)
 create mode 100644 exp/webui4.html
```

图 7.7 提交代码图

gitbash 反馈代码的仓库日志如下图 7.8 所示：

```
$ git log  
commit 898dc8b7156807d1c3bdac638b690bb3727bb3b1 (HEAD -> master)  
Author: 江科师大张梓鑫 <2105058384@qq.com>  
Date: Thu Jun 13 17:44:57 2024 +0800
```

第五次增量迭代—本次提交了‘UI设计之鼠标和触屏拖拽’代码，本次添加了内容区的拖拽响应，可以在计算机上面利用鼠标进行内容区的拖拽，本次将代JS代码整合在一个全局对象Pointer中，这个对象中存储了本程序的所有异步代码，这个对象在本程序的JS里面可以任意调用，简化了代码的复杂度，优化了代码的重复使用性。这样可以减小浏览器的响应时间，提供用户更好的使用体验。

图 7.8 gitbash 反馈图

## 8 UI 的个性化键盘交互控制的设计开发

### 8.1 分析和设计

#### 8.1.1 基本分析

本次代码研究了利用 key down 和 key up 两个底层事件，实现同时输出按键状态和文本内容，在研究中发现增加“阻止事件对象的默认事件后”，不仅 key press 事件将不再响应，而且系统的热键，如“F5 刷新页面/Ctrl+R”、“F12 打开开发者面板”等也不再被响应，随后提出更高阶的问题，如何处理连续空格和制表键 tab？这些都是我们要解决的问题，为了解决这些问题，我们提出了更高层次的挑战：如何处理连续的空格键输入以及制表键 Tab 的特殊情况？这些是我们需要进一步研究和解决的关键问题。

因此，在本次代码实现中，我们不仅需要确保基本的按键事件能够被正确捕捉和响应，还需要对特殊按键行为进行细致的分析和处理，以提高用户体验并确保程序的健壮性。我们还将对代码进行彻底的测试，确保在各种情况下都能保持良好的性能和稳定性。这包括单元测试、集成测试以及用户验收测试，以确保我们的解决方案能够在实际应用中发挥预期的效果。通过这些努力，我们相信能够为用户带来更加流畅和高效的输入体验。

#### 8.1.2 相关图设计

第六次迭代在第五次的基础上对键盘页面进行了优化，如下图 8.1 第六次增量迭代用例图所示。

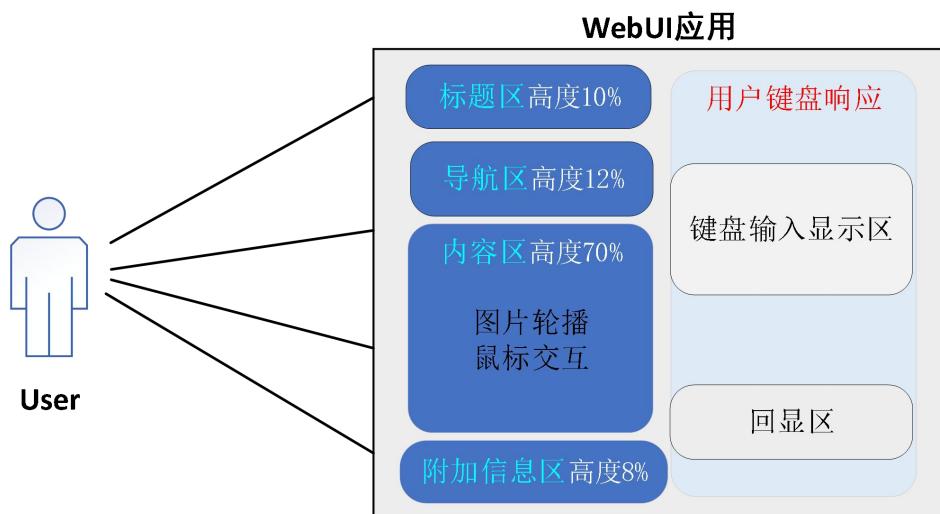


图 8.1 第六次增量迭代用例图

第六次迭代的 DOM 树如下图 8.2 所示，页面布局并没有太大变化。

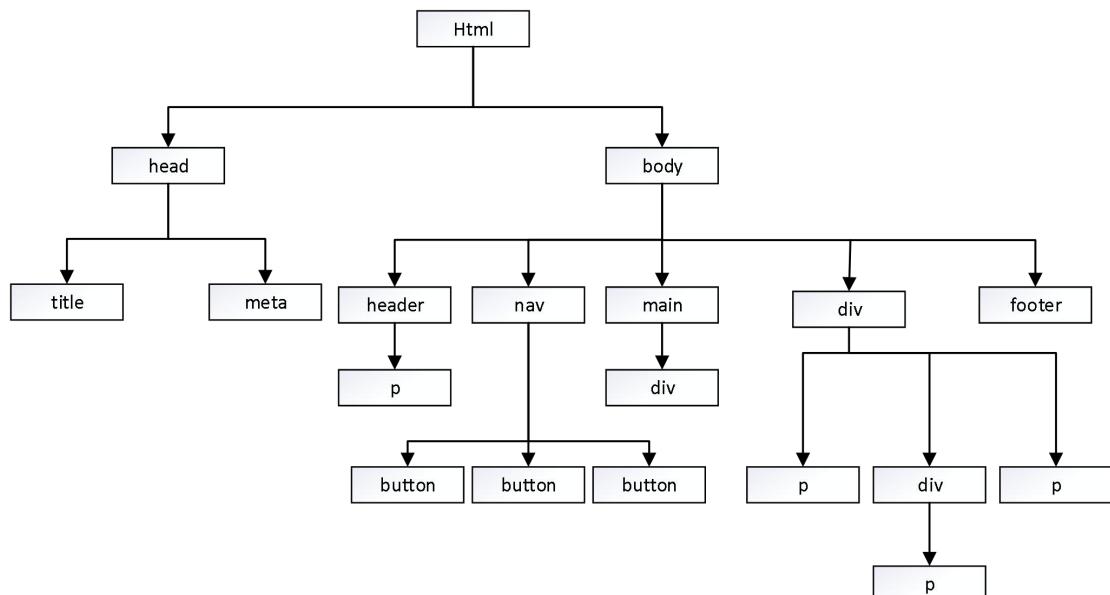


图 8.2 第六次迭代项目 DOM 树

## 8.2 项目的实现和编程

### 8.2.1 HTML 代码编写如下：

```

<body>
  <header>
    <p id="book">
      计算机相关书籍展示
    </p>
  </header>

```

```

</p>
</header>
<nav>
  <button id="backwardBtn">后退</button>
  <button id="pauseBtn">暂停播放</button>
  <button id="forwardBtn">前进</button>
</nav>
<main id="main">
  <div id="bookface">
    这是书的封面图<br>
    在此对象范围拖动鼠标/滑动触屏<br>
    拖动/滑动超过 100 像素，视为有效 UI 互动！
  </div>
</main>
<footer>
  CopyRight 张梓鑫 江西科技师范大学 2024--2025
</footer>
<div id="aid">
  <p id="userResponseArea">用户键盘响应区</p>
  <div id="keyboard">
    <p id="displaytxt">
      &nbsp;
    </p>
  </div>
  <p id="displayCode">
    &nbsp;
  </p>
</div></body>

```

8.2.2 CSS 代码编写如下：

```

<style>
  * {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
  }
  body {
    font-family: 'Roboto', sans-serif;
    line-height: 1.6;
    background-color: #f0f0f0;
    color: #333;
  }
  header, nav, main, footer {
    background-color: #fff;
    margin: 4px;
  }

```

```
padding: 15px;
border-radius: 8px;
box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
}
header {
height: 10%;
text-align: center;
background-color: #4CAF50;
color: white;
}
header p {
font-size: 1em;
font-weight: bold;
}
nav {
text-align: center;
height: 12%;
}
button {
padding: 10px 20px;
font-size: 0.8em;
cursor: pointer;
border: none;
border-radius: 5px;
background-color: #4CAF50;
color: white;
transition: background-color 0.3s ease, transform 0.3s ease;
}
button:hover {
background-color: #45a049;
transform: scale(1.05);
}
main {
text-align: center;
height: 70%;
}
#bookface {
text-align: center;
left: 41px;
position: relative;
width: 80%;
height: 100%;
border: 1px solid #ddd;
background-image: url();
```

```
background-size: cover;
background-position: center;
border-radius: 8px;
box-shadow: 0 2px 8px rgba(0, 0, 0, 0.1);
user-select: none;
-moz-user-select: none;
-webkit-user-select: none;
-ms-user-select: none;
}
footer {
    text-align: center;
    font-size: 0.6em;
    color: #777;
    background-color: #4CAF50;
    color: white;
    padding: 15px;
    border-radius: 5px;
    height: 8%;
}
#aid {
    top: 0.3em;
    left: 650px;
    position: absolute;
    align-content: normal;
    border: 2px solid black;
    box-shadow: 2px 3px 5px 0px rgba(0, 0, 0, 0.2);
}

#userResponseArea {
    margin: 0.8em;
    font-size: 1.66em;
}
#keyboard {
    text-align: center;
    background-color: #A8E6CF;
    background-image: linear-gradient(90deg, #A8E6CF 0%, #DCEDC1 50%, #FFD3B6 100%); /* Updated gradient colors to green shades */
    border: 3px solid black;
    margin-left: 10%;
    width: 80%;
    height: 60%;
    box-shadow: 2px 3px 5px 0px rgba(0, 0, 0, 0.4);
}
#displayCode {
```

```

        border: 3px solid black;
        margin-left: 10%;
        width: 80%;
        position: absolute;
        bottom: 0.3em;
        box-shadow: 2px 3px 5px 0px rgba(0, 0, 0, 0.4);
    }
</style>

```

8.2.3 JavaScript 代码编写如下：

```

<script>
    const images = ['./lesson/GRE.jpg', './lesson/internet.jpg',
'./lesson/nutrition.jpg'];
let currentIndex = 0;
let intervalId = null;
let isPlaying = true; // 标记自动播放状态，默认为 true 表示正在播放
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 + "px";
    document.body.style.height = UI.appHeight - 1 * baseFont + "px";
    UI.appHeight = window.innerHeight;
    if (window.innerWidth < 1000) {
        $("aid").style.display = 'none';
    }
    $("aid").style.width = window.innerWidth - UI.appWidth - baseFont
* 3 + 'px';
    $("aid").style.height = UI.appHeight - baseFont * 1 + 'px';
//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()
    //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%' ;
        }
    }
};

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' ;
    }
}
};

//下面是用户交互的代码，交互的设备包括鼠标，键盘，触屏。
$("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("keydown", function(ev) {
    let k = ev.key;
    let c = ev.keyCode;
    $("displayCode").textContent = "按下键 : " + k + " , " +
"编码 : " + c;
});
$("body").addEventListener("keyup", function(ev) {
    let k = ev.key;
    let c = ev.keyCode;
    $("displayCode").textContent = k + " 键已弹起" + " , " +
"编码" + c;
/*判断键盘按下的是否为 Enter 键，如果是添 p 元素实现换行*/
if (k === "Enter") {
    //有且只能在 document 中创建子节点
    let p = document.createElement("p");
    //通过创建 p 元素，添加子节点来实现换行。
    $("keyboard").appendChild(p);
} else if (k === "Backspace") {
    /*没有字符可以删除了，则将该子节点删除*/
    if ($("#keyboard").lastElementChild.textContent ===
"") {
        /*删除前保证 keyboard 中至少有一个字节点*/
        if ($("#keyboard").childElementCount > 1) {

$("keyboard").removeChild($("#keyboard").lastElementChild);

```

```

        }
    } else {
        $("keyboard").lastElementChild.textContent =
$("keyboard").lastElementChild.textContent.slice(0, -1);
    }
} else if (printLetter(k)) {
    $("keyboard").lastElementChild.textContent += k;
}
function printLetter(k) {
    if (k.length > 1) { //学生须研究这个逻辑的作用
        return false;
    }
    let puncs = ['~', '`', '!', '@', '#', '$', '%', '^',
'&', '*', '(', ')', '-', '_', '+', '=', ',', '.', ';', ',', '<', '>', '?',
 '/', ' ', '\'', '\"'];
    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;
        }
    }
    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;
    }
}

// 更新显示的图片
function updateImage() {
    document.getElementById('bookface').style.backgroundImage =
`url('${images[currentIndex]}`);
}

// 点击前进按钮
document.getElementById('forwardBtn').addEventListener('click',
function() {
    currentIndex = (currentIndex + 1) % images.length;
    updateImage();
});

// 点击后退按钮
document.getElementById('backwardBtn').addEventListener('click',
function() {
    currentIndex = (currentIndex - 1 + images.length) % images.length;
    updateImage();
});

// 下一张图片
function nextImage() {
    currentIndex = (currentIndex + 1) % images.length;
    updateImage();
}

// 初始化显示第一张图片
updateImage();

// 自动播放
intervalId = setInterval(nextImage, 3000);

// 停止自动播放
document.getElementById('pauseBtn').addEventListener('click', function()
{
    clearInterval(intervalId);
});

// 切换自动播放状态
document.getElementById('pauseBtn').addEventListener('click', function()
{
```

```

const pauseBtn = document.getElementById('pauseBtn');
if (isPlaying) {
    clearInterval(intervalId); // 停止自动播放
    pauseBtn.textContent = '开始播放'; // 更改按钮文本
} else {
    intervalId = setInterval(nextImage, 3000); // 开始自动播放
    pauseBtn.textContent = '暂停播放'; // 更改按钮文本
}
isPlaying = !isPlaying; // 切换播放状态
}); </script>

```

### 8.3 项目的运行和测试

因为本次设计是针对键盘，而移动端没有键盘响应，所以这里不进行展示。  
本次设计的界面效果如下图 8.3 所示，仅展示宽屏界面：



图 8.3 PC 界面

本章代码 URL 地址：<https://iasdkasdj.github.io/exp/WebUI6.html>

我们在 PC 端可以复制上述地址，在现代浏览器（含微信）中运行。

本章代码移动端运行二维码：

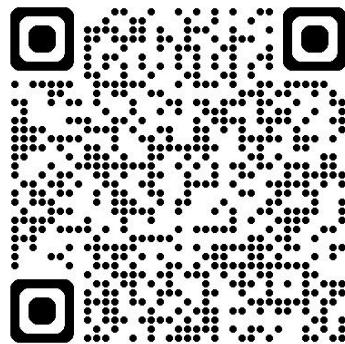


图 8.4 阶段 6 增量迭代二维码

## 8.4 项目的代码提交和版本管理

作为项目的第六次迭代，在代码提交和版本管理环节，我们的目标是对 index.html 文件进行更新，同时将这次历史版本命名为 WebUI5.html，将 index.html 文件命名为 WebUI6.html。编写好代码，测试运行成功后，按次序输入以下命令：

```
$ git add index.html exp/WebUI5.html exp/WebUI6.html
```

\$ git commit -m 第六次增量迭代——本次提交了‘UI 设计之个性化键盘’代码，旨在提供用户更好的使用体验。为了实现这一目标，我们特别对 PC 端用户的键盘输入逻辑进行了优化。通过调整和改进，我们确保了输入过程更加流畅和直观，减少了用户在进行键盘操作时可能遇到的障碍。此外，我们还对键盘的响应速度和准确性进行了提升，使得用户在输入时能够感受到更加迅速和准确的反馈。这些改进不仅提升了用户的操作效率，也增强了整体的使用满意度。我们相信，通过这些精心设计的更新，用户在使用个性化键盘时将享受到更加舒适和个性化的输入体验。

将代码用 Git Bash 对项目代码进行版本控制如下图 8.5 提交代码图所示：

```
[master 250335b] 第六次增量迭代——本次提交了‘UI 设计之个性化键盘’代码，旨在提供用户更好的使用体验。为了实现这一目标，我们特别对 PC 端用户的键盘输入逻辑进行了优化。通过调整和改进，我们确保了输入过程更加流畅和直观，减少了用户在进行键盘操作时可能遇到的障碍。此外，我们还对键盘的响应速度和准确性进行了提升，使得用户在输入时能够感受到更加迅速和准确的反馈。这些改进不仅提升了用户的操作效率，也增强了整体的使用满意度。我们相信，通过这些精心设计的更新，用户在使用个性化键盘时将享受到更加舒适和个性化的输入体验。
3 files changed, 798 insertions(+), 42 deletions(-)
 create mode 100644 exp/webUI5.html
 create mode 100644 exp/webUI6.html
```

## 8.5 提交代码图

gitbash 反馈代码的仓库日志如下图 8.6 所示：

```
$ git log  
commit 250335b6d5b2d9efeb6ab2ddb60a6c9f607f77f3 (HEAD -> master)  
Author: 江科师大张梓鑫 <2105058384@qq.com>  
Date:   Thu Jun 13 18:39:13 2024 +0800
```

第六次增量迭代—本次提交了‘UI设计之个性化键盘’代码，旨在提供用户更好的使用体验。为了实现这一目标，我们特别对PC端用户的键盘输入逻辑进行了优化。通过调整和改进，我们确保了输入过程更加流畅和直观，减少了用户在进行键盘操作时可能遇到的障碍。此外，我们还对键盘的响应速度和准确性进行了提升，使得用户在输入时能够感受到更加迅速和准确的反馈。这些改进不仅提升了用户的操作效率，也增强了整体的使用满意度。我们相信，通过这些精心设计的更新，用户在使用个性化键盘时将享受到更加舒适和个性化的输入体验。

## 8.6 gitbash 反馈图

# 9 用 Git Bash 工具管理项目的代码仓库和 Http 服务器

## 9.1 跨世纪的经典 Bash 工具

当我们提及命令行界面（CLI），实际上我们是指 Shell，这是一种命令行解释器，它允许用户通过键盘输入文本命令并将其发送至操作系统以执行相应的操作。Shell 作为用户与操作系统内核之间的中介，不仅接收用户的指令，还处理这些指令并提供反馈结果，如在显示器上显示输出或将结果写入文件中。

在 Linux 操作系统中，大多数发行版默认配备的 Shell 是由 GNU 项目开发的 Bash（Bourne Again SHell）。Bash 是 Steve Bourne 所编写的原始 Unix Shell 程序 sh 的增强版本，“Bash”这个名字正是“Bourne-again shell”的缩写。Bash 继承了 sh 的特点，并在此基础上增加了许多新的功能和命令，使其更加强大和灵活。

Bash 提供了丰富的特性，如命令补全、命令历史记录、别名、管道（pipelining）、输入输出重定向以及强大的脚本编程能力。这些功能极大地提升了用户执行命令和编写脚本的效率。此外，Bash 还支持多种 Shell 脚本编程结构，如循环、条件判断、函数等，使得用户可以编写更加复杂的脚本以实现自动化任务。

Bash 的脚本编程能力，使得它不仅仅是一个命令执行器，更是一个功能强大的自动化工具。用户可以编写 Shell 脚本来自动化日常任务，如系统管理、文件处理、网络通信等。此外，Bash 还具有良好的兼容性和可移植性，可以在多种操作系统和平台上运行。

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<sup>[8]</sup>。

## 9.2 通过 Git Hub 平台实现本项目的全球域名

### GitHub Pages

[GitHub Pages](#) is designed to host your personal, organization, or project pages from a GitHub repository.

Your site is live at <https://iasdkasdj.github.io/>  
Last deployed by  iasdka... yesterday

[Visit site](#)

...

图 9.1 本项目的全球域名

如上图 9.1 所示, 为本项目在 Git Hub 平台上的全球域名, 可以在 PC 端的微软 Edge 浏览器中输入该域名进行访问。

## 9.3 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](#).

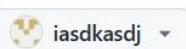
Required fields are marked with an asterisk (\*).

#### Repository template

[No template](#) ▾

Start your repository with a template repository's contents.

Owner \*



Repository name \*

/ zhangzixin.github.io

 zhangzixin.github.io is available.

图 9.2 创建空的远程代码仓库

如上图 9.2 所示, 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。

## 9.4 设置本地仓库和远程代码仓库的链接

进入本地 web UI 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

```
$ git init  
$ git commit -m "把代码仓库上传至 git hub 平台"  
$ git branch -m master main  
$ git remote add origin  
https://github.com/laishengzhen/laishengzhen.github.io.git  
$ git push -u origin main
```

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 9.3 所示：

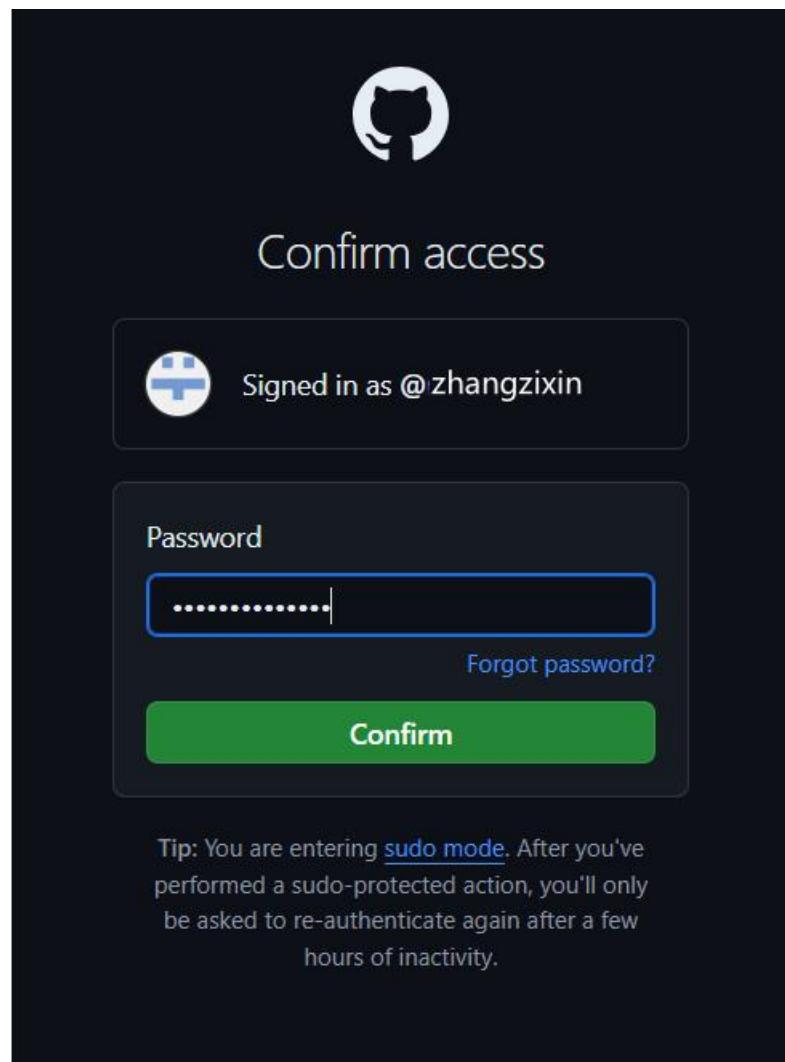


图 9.3 开发者授权

如下图 9.4 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：

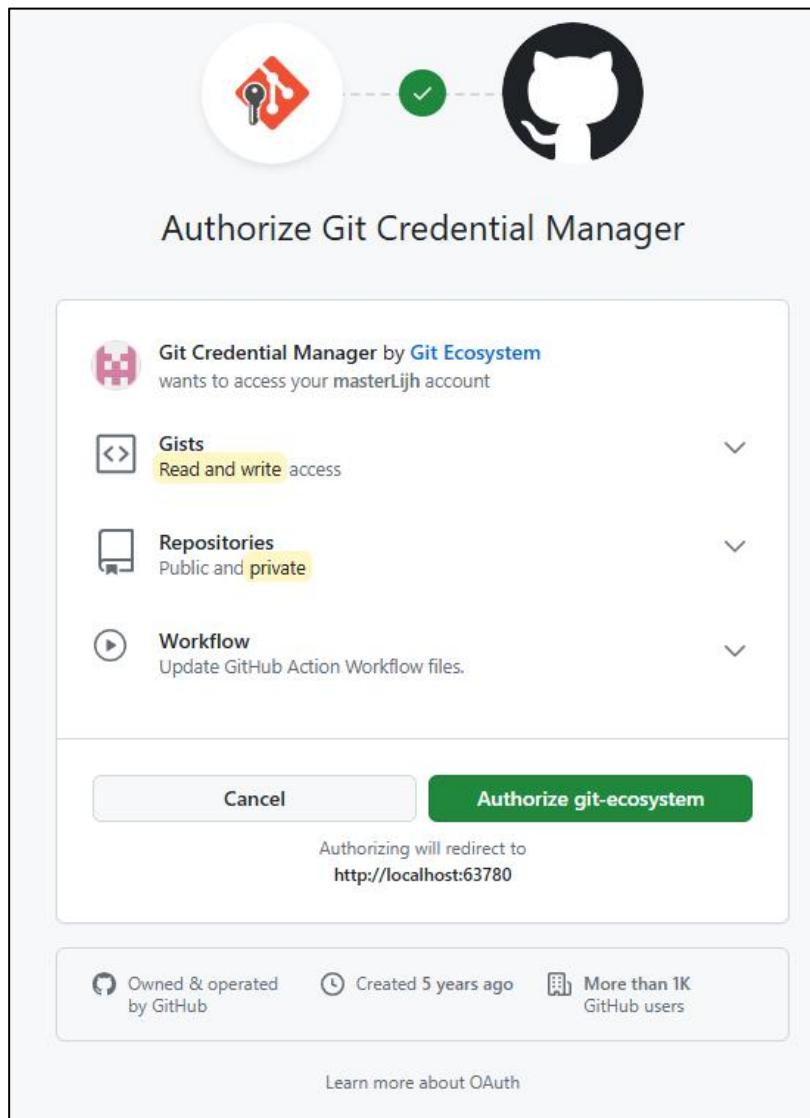


图 9.4 再次确认授权

最后，GitHub 平台反馈：git bash 和 git hub 平台成功实现远程链接如下图 9.5 所示。

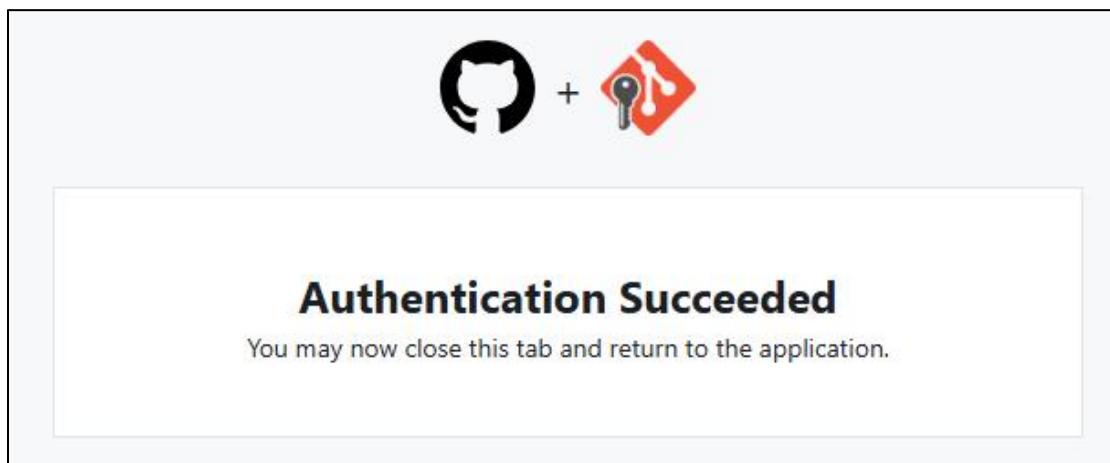


图 9.5 成功实现远程连接

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 git hub 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，如下图 9.6 所示：



图 9.6 项目在互联网的部署

## 参考文献

- [1] W3C. W3C's history. W3C Community. [EB/OL]. <https://www.w3.org/about/>. 2023.12.20
- [2] Douglas E. Comer. The Internet Book [M] (Fifth Edition). CRC Press Taylor & Francis Group, 2019: 217-218
- [3] John Dean,PhD. Web programming with HTML5,CSS,and JavaScript[M]. Jones & Bartlett Learning,LLC. 2019: 2
- [4] John Dean,PhD. Web programming with HTML5,CSS,and JavaScript[M]. Jones & Bartlett Learning,LLC. 2019: xi
- [5] Behrouz Forouzan. Foundations of Computer Science[M](4th Edition). Cengage Learning EMEA,2018: 274--275
- [6] Marijn Haverbeke. Eloquent JavaScript 3rd edition. No Starch Press,Inc, 2019.
- [7] William Shotts. The Linux Command Line, 2nd Edition [ M ]. No Starch Press, Inc, 245 8th Street, San Francisco, CA 94103, 2019: 3-7
- [8] Martina Seidl, Marion Scholz, et al. UML @ Classroom An Introduction to Object-Oriented Modeling [M]. Springer International Publishing Switzerland, 2015