成 绩：

# 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：刘嘉舒

学　　号：20213585

指导教师：李健宏

2024 年 6 月 13 日

目录

# 基于Web客户端技术的个性化UI设计和实现

**摘要**：Web 平台以其跨操作系统平台的优势成为了广泛流行的软件开发技术，本项目以 Web 客户端技术为研究学习对象，探索了 HTML 内容建模、CSS 样式设计和 JavaScript 功能编程的基本技术和技巧。实现了一个个性化的用户界面（UI：user interface），该用户界面可以动态适用于 PC 端和移动端设备，以响应式技术为支撑做到了最佳适配用户屏幕，以 DOM 技术和事件驱动模式的程序为支撑实现了对鼠标、触屏、键盘的底层事件响应和流畅支持，为鼠标和触屏设计了一个对象模型，用代码实现了对这类指向性设备的模拟，这是本项目模型研究法的一次创新实践，也是本项目的亮点。为了处理好设计和开发的关系，用工程思想管理项目，我使用了软件工程的增量式增强的开发模式，每次迭代都包含了软件开发的四个经典开发过程 ADIT（A：Analysis 分析， D：Design 设计，I：Implementation 实现，T：Testing 测试）开发，逐步实现了整个 UI 应用的编写。为了与互联网上的同行合作，本项目还使用了 git 工具进行代码和开发过程日志记录，进行提交代码的操作，详细记录和展现了开发思路和代码优化的路径。最后通过 gitbash 软件把项目上传到 GitHub 服务器上，建立了自己的代码仓库，并将该代码仓库设置成为了 HTTP 服务器，实现了本 UI 应用的全球便捷访问。

**关键词**：响应式设计；鼠标；触屏；键盘控制；GitBash 技术

## 1. 前言

毕业设计和毕业论文是大学生在对大学阶段性学习的总结，更是自己把在这一段大学学习到的所有知识的运用。毕业设计的目的要求：

根据个人对毕业设计项目和毕业论文的理解，分别阐述：学习计划、项目的技术路线、参考资料和研究方法。

建议参考： https://masterlijh.github.io/testHttp.html 的第一点和第二点。

## 1.1 毕设任务分析

毕设任务分为两部分：毕业设计和毕业论文两部分。

毕业设计是为了检验学生综合运用所学理论、知识和技能解决实际问题的能力，让学生针对某一课题，作出解决实际问题的设计，是学生毕业前夕总结性的独立作业，是实践性教学最后一个环节。在教师指导下，学生就选定的课题进行工程设计和研究，要求学生运用本科阶段所学习到的计算机科学与技术知识，尤其是程序设计和软件工程领域学习的方法、训练的代码能力、架构自己感兴趣的技术路线，再结合自己要解决的问题形成软件需求，最后通过在对需求分析的基础上，进行建立模型、软件设计、系统实施、测试调试一系列的软件开发流程，

最终完成毕业设计的开发过程。

毕业论文是学生集中进行科学研究训练而要求学生在毕业前撰写的论文。毕业论文一般安排在修业的最后一学年（学期）进行，是学校对学生知识相与能力进行全面考核的环节，也是学生毕业与学位资格认证的重要依据。毕业论文的撰写应贯彻理论联系实际的原则，反映学生运用所学的专业基础理论、基本知识和基本技能分析和解决实际问题的能力。毕业论文应做到观点明确、材料翔实、论据充分、结构严谨、逻辑严密、语言流畅、格式规范。

在高校的毕业论文管理手册中，看似是以毕业论文为主，对毕业设计没有硬性要求。但是在我看来，毕业设计与毕业论文是一个等价的关系，亦或者毕业设计更加重要一点，因为只有你拥有一个好的毕业设计才可以写出一篇好的毕业论文。毕业设计是毕业论文的前提亦或者是毕业论文的开始。

## 1.2 研学计划

研学计划旨在通过实践活动和实地考察，将理论知识与实际相结合，加深学生对知识的理解，并培养学生的实践能力和创新精神。本计划结合学生的年龄、学科特点、兴趣爱好等因素，设计了一系列丰富多彩的研学活动，旨在全面提升学生的综合素质。

| 时间 | 操作 |
| --- | --- |
| 第一周 | 运用“三段式”内容概要 |
| 第二周 | 移动互联网时代窄屏的响应式设计 |
| 第三周 | 移动互联网时代窄屏与宽屏的响应式设计 |
| 第四周 | UI 设计之鼠标模型的分析与控制 |
| 第五周 | 通过 UI 设计，用一套代码同时为触屏和鼠标建模，实现鼠标、触屏事件 |
| 第六周 | UI 个性化键盘控制——巧妙运用 Keydown 和 keyup 键盘底层事件 |

## 1.3 研究方法

为了保证论文的科学性和有效性，本论文首先采用了文献综述法，通过各种文献数据库、搜索引擎和图书馆等途径进行文献检索，根据研究问题和综述范围的要求，筛选出与研究主题相关的文献。再采用增量式开发模式进行系统的开发设计，从分析、设计、实现和测试的四个阶段进行开发，一步一步的实现软件的开发过程，做到软件的实时更新、不断改进，从而达到项目的灵活开发，并且使用 Gitbash 软件对代码进行提交，在每一次提交过程中对代码的重要部分进行详细记录，保证的项目的可持续发展。最后将代码上传到 GitHub 上，进行开源化共享，吸引其他开发者的关注和使用，也可以吸引贡献者，共同改进和维护代码，通过 GitHub 的社区功能，还可以与其他开发者交流、分享经验和知识。

# 2. Web 概述

## 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[1]。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，是我的毕设项目应用的技术路线。

Web 平台：Web 主要包括服务器端、后端架构、数据库、中间件、API 五个方面，其中服务器端是 Web 服务器负责处理 HTTP 请求，提供动态内容，如 Apache、Nginx、IIS 等；后端架构是用于构建服务器端应用程序，如 Node.js（Express, Koa）、Python（Django, Flask）、Java（Spring Boot）、Ruby（Rails）等；数据库是存储和管理数据，如 MySQL、PostgreSQL、MongoDB、Redis 等；中间件是：提供额外的功能，如认证、路由、日志、缓存等；API 是：应用程序编程接口，允许不同的系统之间交换数据，如 RESTful API、GraphQL 等。

客户端技术：客户端技术分为三大类：Web 客户端技术、移动客户端技术、桌面客户端技术。其中 Web 客户端技术主要包括 HTML（超文本标记语言，用于创建网页的结构）、CSS（层叠样式表，控制网页的布局和样式）、javaScript（客户端脚本语言，实现网页的交互性和动态功能），这三种技术也是我毕业设计中所主要运用到的技术。移动客户端技术主要包括：原生应用（使用平台特定的 SDK 和编程语言）、跨平台框架（使用单一代码库开发多平台应用）、混合应用（结合 HTML、CSS 和 JavaScript，通过 WebView 运行）。我的这个毕业设计中就运用到了跨平台框架，从而使我的项目满足在 PC 端和移动端双向使用。桌面客户端技术主要包括：桌面应用框架（用于构建跨平台的桌面应用程序）、操作系统 API（直接与操作系统交互）。

### 2.2.1 History

1989 年，蒂姆·伯纳斯-李爵士发明了万维网（见最初的提案）。他在 1990 年 10 月创造了“万维网”一词，并写下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）“世界万维网”。 他编写了“超文本标记语言”（HTML）的第一个版本，这是一种具有超文本链接功能的文档格式化语言，后来成为了 Web 的主要发布格式。他对 uri、HTTP 和 HTML 的最初规范随着网络技术的传播，在更大的圈子中得到了改进和讨论。

### 2.2.2 A Consortium for the World Wide Web

1994 年，在许多公司的敦促下，决定成立万维网联盟。蒂姆·伯纳斯-李爵士开始领导网络联盟团队的基本工作，以培养一个一致的架构，以适应网络标准的快速发展，以构建网站、浏览器和设备，以体验网络所提供的一切。

在创立万维网联盟的过程中，蒂姆·伯纳斯-李爵士创建了一个同行社区。Web 技术已经如此之快，以至于组装一个组织来协调 Web 标准至关重要。蒂姆接受了麻省理工学院举办 W3C 课程的邀请。他从一开始就要求 W3C 拥有全球的足迹。

### 2.2.3 Web platform and Web Programming

web 平台。

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[1]。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。

网络是万维网的缩写。大多数人说“是网络”而不是“万维网”，我们会遵循这个惯例。Web 是一个文档的集合，被称为网页，它们由世界各地的计算机用户（在大部分时间内）共享。不同类型的网页可以做不同的事情，但至少，它们都能在电脑屏幕上显示内容。我们所说的“内容”是指文本、图片和用户输入机制，如文本框和按钮[2]。

web 编程。

Web 编程是一个很大的领域，通过不同的工具实现不同类型的 Web 编程。所有的工具都使用核心语言 HTML 来工作，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，所有的深入。这三种技术被认为是客户端网络编程的支柱。使用客户端 web 编程，所有网页计算都在终端用户的计算机（客户端计算机）[3]上执行。

## 2.2 项目的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。对于开发过程，有几个模型。我们在这里讨论最常见的两种：瀑布模型和增量模型。

### 2.2.1 瀑布模型

软件开发过程中一个非常流行的模型被称为瀑布模型，如图 1。在这个模型

中，开发过程只流向一个方向。这意味着在上一个阶段完成之前才能启动一个阶段。

例如，整个项目的分析阶段应在其设计阶段开始之前完成。整个设计阶段应在开始实施阶段之前完成。

瀑布模型既有优点也有缺点。其中一个优点是，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，因为他们拥有分析阶段的完整结果。测试阶段可以测试整个系统，因为正在开发的整个系统已经准备好了。然而，瀑布模型的一个缺点是难以定位问题：如果在部分过程中存在问题，则必须检查整个过程的[4]。

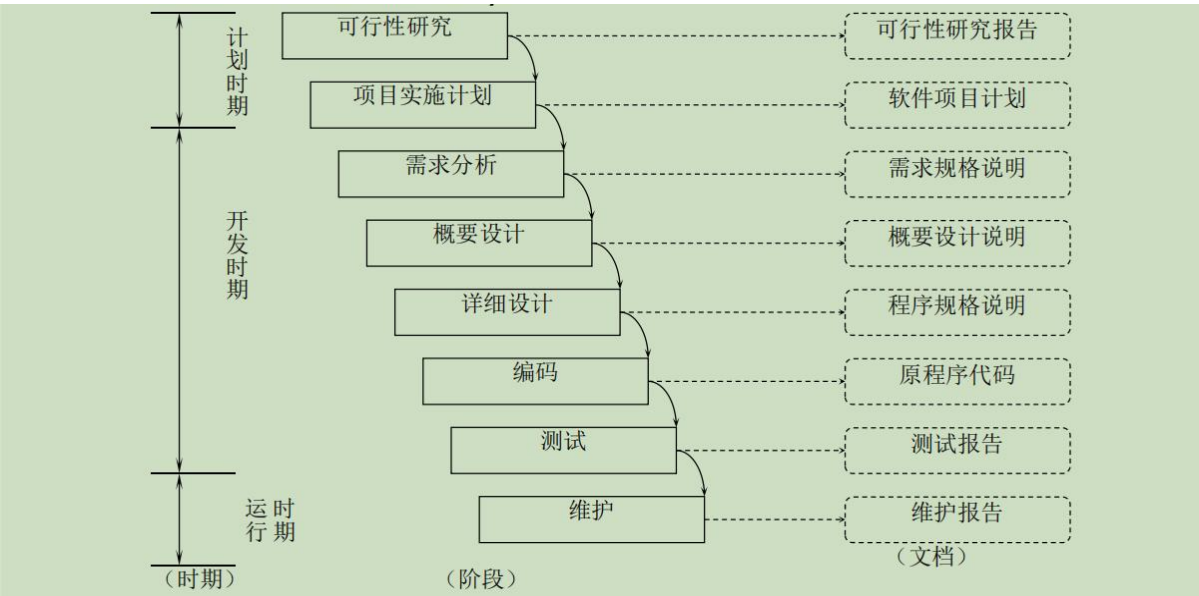



图 1

## 2.2.2 增量模型

在增量模型中，软件分一系列步骤进行开发，如图 2。开发人员首先完成了整个系统的一个简化版本。这个版本表示整个系统，但不包括详细信息。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，而一些没有完成，系统再次测试。如果有问题，开发人员就知道问题在于新功能。在现有的系统正常工作之前，它们不会添加更多的功能。这个过程继续进行，直到需要的所有功能都添加到[4]。

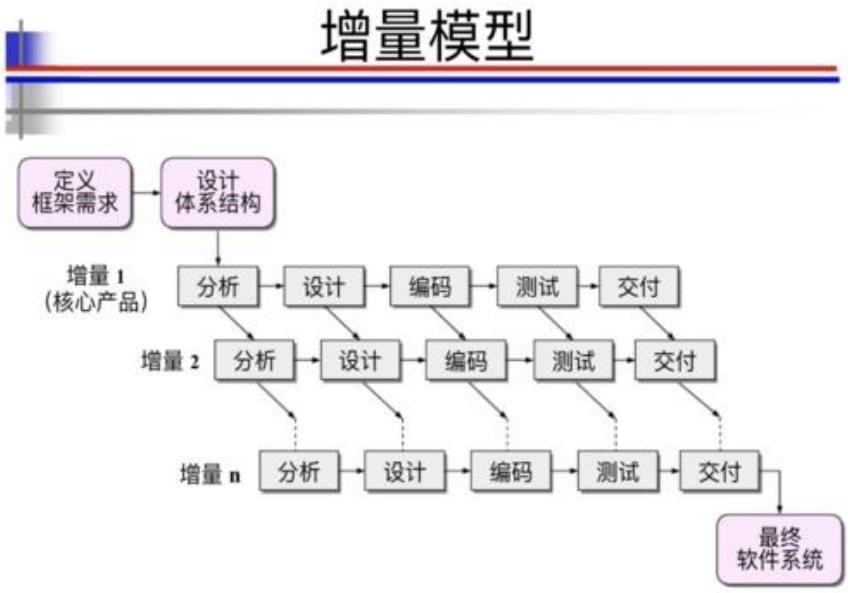



图 2

# 3. 内容设计概要

## 3.1 分析和设计

本项目最开始使用人们常用的“三段论”的形式，简洁明了的开展内容设计，首先在项目设计最上端设定一个区域为标题区域，用来展示 logo 或文章标题，吸引用户的注意力，直观的表达主题；然后是内容展示区域，我们本着“内容为王”的理念，将其设定为整个 UI 的重点；最后则是附加信息区域，用于显示一些个人信息。如图 3-1 用例图所示：

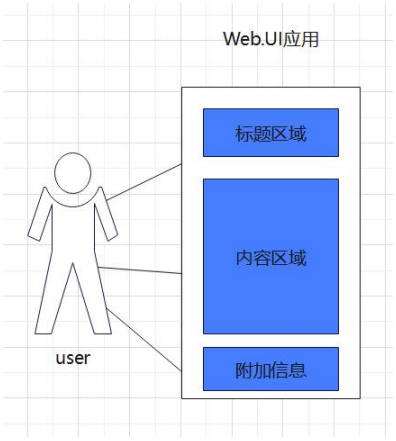



图 3-1 用例图

## 3.2 项目的实现和编程

一、HTML 代码编写如下：

```
<title> "读好书、练思维、爱编程" </title>
</head>
<body>
 <header>
   <p id="book">
     浪漫人间总是蓄满鲜花
   </p>
 </header>
 <nav>
   <button>导航一</button>
   <button>导航二</button>
   <button>导航三</button>
 </nav>
 <main id = 'main'>
   软件内容区域
 </main>

 <footer>
  <p id="statusInfo">
   刘嘉舒 江西科技师范大学 2022--2025
  </p>
 </footer>
<script src="myJs.js" >
</script>
</body>
</html>
```

二、CSS 代码编写如下：

```
 {
 margin: 10px;
 text-align: center;
}
 header{
   border: 2px solid green;
   height: 200px;
   font-size: 20px;
   font-Weight: bold;
  }
```

```
    main{
      border: 2px solid green;
      height: 400px;
      font-size: 6px;
    }
    footer{
      border: 2px solid green;
      height: 100px;
    }
   a{
    display: inline-block ;
  padding:10px ;
  color:*{
     margin: 10px;
    text-align: center;
    }
     header{
       border: 2px solid #fab1c5;
       height: 80px;
     }
     main{
       border: 2px solid #fab1c5;
       height: 300px;
     }
     nav{
       border: 2px solid #f5fc89;
       height: 50px;
     }
     footer{
       border: 2px solid #f5fc89;
       height: 50px;
     }
 white;
  background-color: green;
  text-decoration: none ;
   }
```

## 3.3 项目的运行和测试

项目的运行与测试，我使用了联想浏览器进行运行并进行测试。如图 3-2 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

3-3 的二维码，运行测试本项目的第一次开发的阶段性效果。

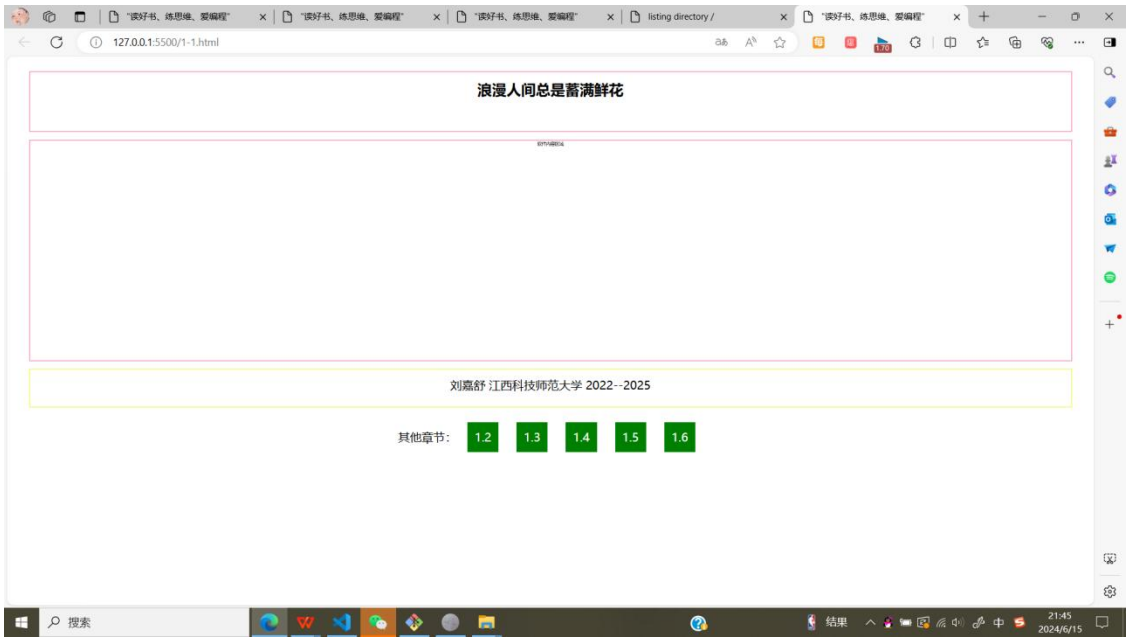



图 3-2


图 3-3

## 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：如开发者的名字和电子邮件。

进入 gitBash 命令行后，按次序输入以下命令：

```
Lenovo@DESKTOP-6CQ14HI MINGW64 /
$ cd /
Lenovo@DESKTOP-6CQ14HI MINGW64 /
$ mkdir WebUI
Lenovo@DESKTOP-6CQ14HI MINGW64 /
$ cd WebUI/
Lenovo@DESKTOP-6CQ14HI MINGW64 /WebUI
$ git init
Initialized empty Git repository in D:/javaweb/Git/WebUI/.git/
Lenovo@DESKTOP-6CQ14HI MINGW64 /WebUI (master)
$ git config user.email LJS@abcLJS
Lenovo@DESKTOP-6CQ14HI MINGW64 /WebUI (master)
$ touch 1-1.html
Lenovo@DESKTOP-6CQ14HI MINGW64 /WebUI (master)
```

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：

```
$ git add 1-1.html
Lenovo@DESKTOP-6CQ14HI MINGW64 /WebUI (master)
$ git commit -m 项目第一版：读好书、练思维、爱编码
```

成功提交代码后，`gitbash` 的反馈如下所示：

```
hp@LAPTOP-Q3M6BAIF MINGW64 /
$ cd

hp@LAPTOP-Q3M6BAIF MINGW64 ~
$ cd /

hp@LAPTOP-Q3M6BAIF MINGW64 /
$ cd WebUI/

hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ touch 1-1.html

hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ git add 1-1.html
```

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所示：

```
hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ git commit -m 项目第一版:读好书、练思维、爱编码
[master (root-commit) 6e087d3] 项目第一版:读好书、练思维、爱编码
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 1-1.html

hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ git log
commit 6e087d3dbc07b8d76138d7602e209cc338cef03e (HEAD -> master)
Author: liujiashu666 <LJS@abcljs>
Date:   Sat Jun 15 21:50:31 2024 +0800

    项目第一版:读好书、练思维、爱编码
```

# 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 分析与设计

在之前的基础上，这一次的设计我增加了导航栏区域，而且通过更改 body 对象的字体大小使之可以遗传其“子子孙孙”，并且将 body 对象的宽度和高度设置为设备/屏幕的宽度和高度，进而实现全屏，最后通过 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，达到窄屏终端的响应式设计。如图 4-1 用例图所示。

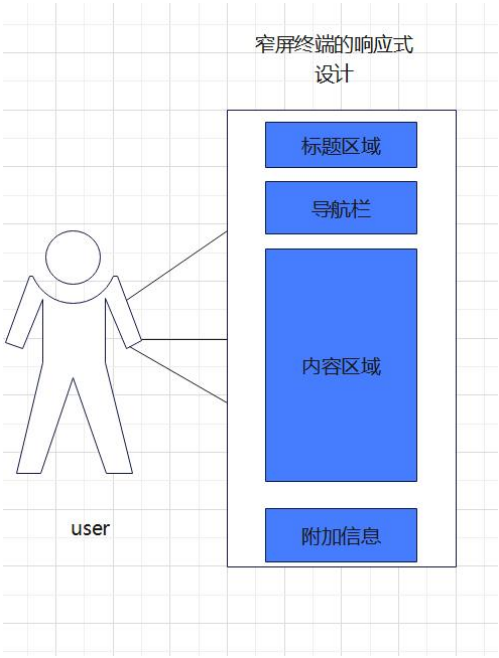



图 4-1 用例图

## 4.2 项目的实现和编程

一、HTML 代码编写如下：

```
<title> "读好书、练思维、爱编程" </title>
 </head>
 <body>
  <header>
    <p id="book">
        浪漫人间总是蓄满鲜花
    </p>
  </header>
  <nav>
    <button>导航一</button>
    <button>导航二</button>
    <button>导航三</button>
  </nav>
  <main id = 'main'>
    软件内容区域
  </main>

  <footer>
   <p id="statusInfo">
    刘嘉舒 江西科技师范大学 2025
   </p>
  </footer>
```

二、CSS 代码编写如下：

```
*{
    margin: 10px;
    text-align: center;
   }
    header{
      border: 2px solid #fab1c5;
      height: 15%;
      font-size: 20px;
      font-Weight: bold;

    }
    main{
      border: 2px solid #fab1c5;
      height: 70%;
      font-size: 6px;
      background-image: url(abc.jpg);

    }


    nav{
      border: 2px solid #f5fc89;
      height: 10%;
    }
    footer{
      border: 2px solid #f5fc89;
      height: 5%;
    }
```

三、JavaScript 代码编写如下：

设置了一个全局变量 UI，将宽度与高度这些参数进行存储，调用 innerWidth 函数和 innerHeight 函数获取屏幕的宽度与高度；然后通过把 body 对象的宽度和高度/屏幕的宽度和高度实现全屏效果，再通过 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。

```
<script>
    var UI = {};
    UI.appWidth = window.innerWidth;
    UI.appHeight = window.innerHeight;
    let baseFont = UI.appWidth / 20;
    //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 对象的高度设置为设备/屏幕的高度，实现纵向全屏。
    //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.height = UI.appHeight - 70 + "px";
</script>
```

## 4.3 项目的运行和测试

项目的运行与测试，我进行了 PC 端与移动端分别进行运行测试。如图 4-2 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的二维码，如图 4-3 所示，移动端的效果图如图 4-4 所示，运行测试本项目的第二次开发的阶段性效果。

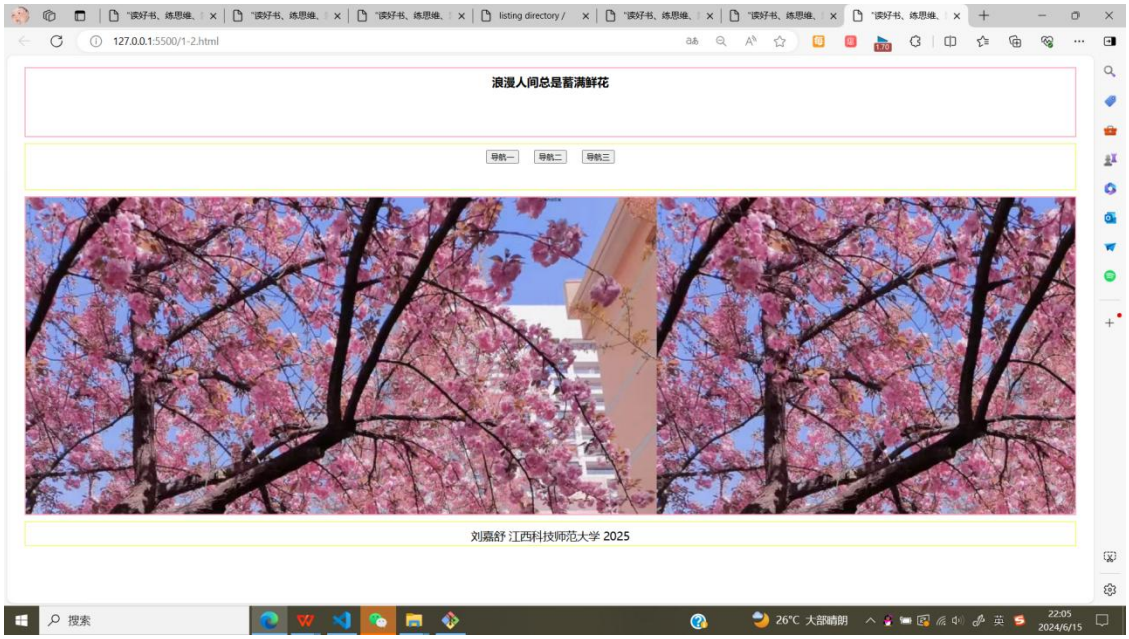



图 4-2


扫描 QR 码
https://liujiashu666.github.io/1-2.html
复制
下载

图 4-3


14:28
"读好书、练思维、爱编程"
浪漫人间总是蓄满鲜花
导航一
导航二
导航三
刘嘉舒 江西科技师范大学 2025

图 4-4

## 4.4 项目的代码提交和版本管理

编写好 1.2html 的代码，测试运行成功后，执行下面命令提交代码：

```
hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ touch 1-2.html

hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ git add 1-2.html

hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ git commit -m 项目第二版：增加了导航栏模板，通过更改body对象的字体，使之可以遗传给他的后代，并且通过将body对象的宽度和高度设置为设备或屏幕的宽度和高度，进而实现全屏。最后通过CSS对子对象百分比（纵向）配合，从而实现响应式设计的目标。
、[master f3da218] 项目第二版：增加了导航栏模板，通过更改body对象的字体，使之可以遗传给他的后代，并且通过将body对象的宽度和高度设置为设备或屏幕的宽度和高度，进而实现全屏。最后通过CSS对子对象百分比（纵向）配合，从而实现响应式设计的目标。
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 1-2.html
```

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所示：

```
hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ git log
commit f3da218a5f0cd507ac7c87dcf0ecd22bab8ab042 (HEAD -> master)
Author: liujiashu666 <LJS@abcljs>
Date:   Sat Jun 15 22:16:52 2024 +0800

    项目第二版：增加了导航栏模板，通过更改body对象的字体，使之可以遗传给他的后代，并且通过将body对象的宽度和高度设置为设备或屏幕的宽度和高度，进而实现全屏。最后通过CSS对子对象百分比（纵向）配合，从而实现响应式设计的目标。

commit 6e087d3dbc07b8d76138d7602e209cc338cef03e
Author: liujiashu666 <LJS@abcljs>
Date:   Sat Jun 15 21:50:31 2024 +0800

    项目第一版:读好书、练思维、爱编码
```

# 5. 应用响应式设计技术开发可适配窄屏和宽屏的 UI

## 5.1 分析与设计

在之前设计的基础上，完成了对宽屏和窄屏的响应式设计，增加了一个用户键盘响应区域，并且抓取了鼠标这个对象，获取到了鼠标的坐标信息，尝试对鼠标设计 UI 控制。如图 5-1 用例图所示。

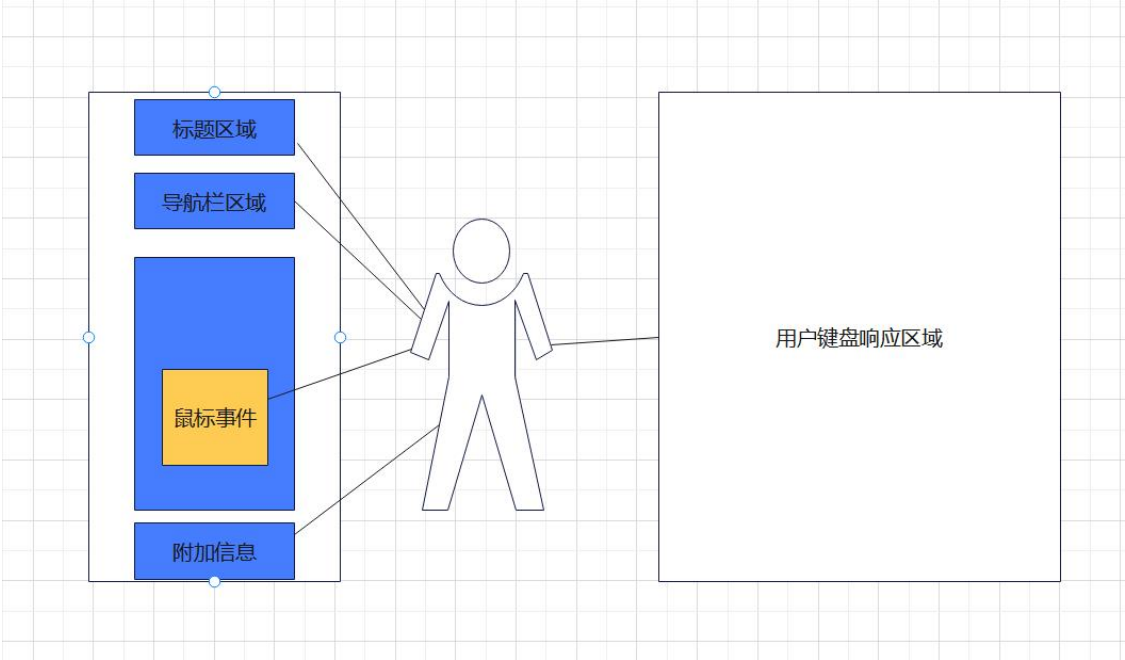



图 5-1 用例图 5.2 项目的实现和编程

## 5.2 项目的实现和编程

一、HTML 代码编写如下：

新增了一个用户键盘响应区域和书的封面区域，让项目更丰富。

```
</head>
<body >
  <header>
    <p id="book">
      浪漫人间总是蓄满鲜花
    </p>
   </header>
  <nav>
   <button>导航 1</button>
   <button>导航 2</button>
   <button>导航 3</button>
  </nav>


   <main id="main">
  <div id="bookface",>
    这是花的种类<br>
    在此对象范围拖动鼠标(本例触屏无效)
    </div>
   </main>
```

```
    <footer>

      CopyRight 刘嘉舒 江西科技师范大学 2024--2025
    </footer>
    <div id="aid">
      <p>用户键盘响应区</p>
      <p id="keyboard"></p>
    </div>
```

## 二、CSS 代码编写如下

```
*{
    text-align: center;
  box-sizing: border-box ;
  }
    header,main,div#bookface,nav,footer{
   margin:1em;
  }
    header{
      border: 2px solid #fab1c5;
      height: 15%;
      font-size: 20px;
      font-Weight: bold;

    }
    main{
      border: 2px solid #fab1c5;
      height: 70%;
      font-size: 1.2em;

    }
    nav{
      border: 2px solid #f5fc89;
      height: 10%;
        }
    nav button{
   font-size: 1.1em;
  }
    footer{
      border: 2px solid #f5fc89;
      height: 5%;
      font-size: 20px;
      font-weight: bold;
    }
```

```
    body{
   position:relative ;
  }
  #aid{
    position: absolute;
    border: 3px solid rgb(160, 192, 251);
    top: 0.5em;
    left: 600px;
  }
  #bookface{
    width: 80%;
    height: 80%;
    border:1px solid rgb(162, 255, 198);
    background-color: #f5f9b4;
    margin:auto;
  }
</style>
```

三、JavaScript 代码编写如下

通过 JavaScript 尝试对鼠标设计 UI 控制，通过抓取鼠标坐标获取鼠标的位置。

```
<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.appWidth2baseFont + "px" ;
    document.body.style.height = UI.appHeight8baseFont + "px";

if(window.innerWidth < 900){
    $("aid").style.display='none';
  }
  $("aid").style.width=window.innerWidthUI.appWidth2baseFont + 'px';
  $("aid").style.height= document.body.clientHeight + '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").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";

});
$("body").addEventListener("keypress",function(ev){
    let k = ev.key;
    let c = ev.keyCode;
    $("keyboard").textContent = "您的按键 ： " + k + " ， "+ "字符编码 ： " + c;
});
  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 $
 </script>
```

## 5.3 项目的运行和测试

项目的运行与测试，我进行了 PC 端与移动端分别进行运行测试。如图 5-2 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的二维码，如图 5-3 所示，移动端的效果图如图 5-4 所示，运行测试本项目的第二次开发的阶段性效果。


图 5-2


图 5-3

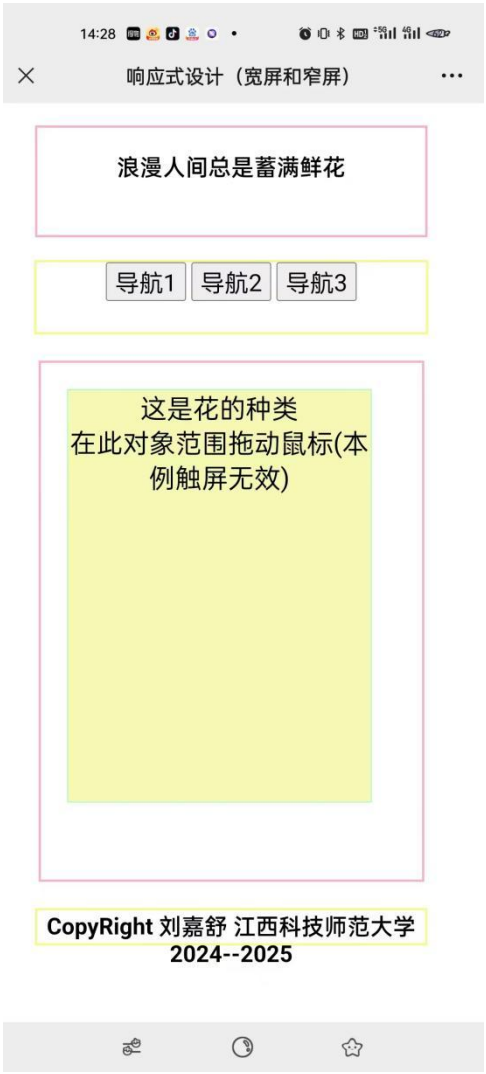



图 5-4

## 5.4 项目的代码提交和版本管理

编写好 1.3html 的代码，测试运行成功后，执行下面命令提交代码：

```
hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ touch 1-3.html

hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ git add 1-3.html

hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ git commit -m 第三版：在之前的基础上，完成了对宽屏和窄屏的响应式设计，增加了一个用户键盘响应区域，并且抓取到了鼠标这个对象，获取到了鼠标的坐标信息，尝试对鼠标进行UI设计控制。
[master df1f680] 第三版：在之前的基础上，完成了对宽屏和窄屏的响应式设计，增加了一个用户键盘响应区域，并且抓取到了鼠标这个对象，获取到了鼠标的坐标信息，尝试对鼠标进行UI设计控制。
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 1-3.html
```

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所示：

```
hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ git log
commit df1f680397d550e373646c372773cb9ba44c180e (HEAD -> master)
Author: liujiashu666 <LJS@abcljs>
Date:   Sun Jun 16 13:12:59 2024 +0800

    第三版：在之前的基础上，完成了对宽屏和窄屏的响应式设计，增加了一个用户键盘响应区域，并且抓取到了鼠标这个对象，获取到了鼠标的坐标信息，尝试对鼠标进行UI设计控制。

commit f3da218a5f0cd507ac7c87dcf0ecd22bab8ab042
Author: liujiashu666 <LJS@abcljs>
Date:   Sat Jun 15 22:16:52 2024 +0800

    项目第二版：增加了导航栏模板，通过更改body对象的字体，使之可以遗传给他的后代，并且通过将body对象的宽度和高度设置为设备或屏幕的宽度和高度，进而实现全屏。最后通过CSS对子对象百分比（纵向）配合，从而实现响应式设计的目标。

commit 6e087d3dbc07b8d76138d7602e209cc338cef03e
Author: liujiashu666 <LJS@abcljs>
Date:   Sat Jun 15 21:50:31 2024 +0800

    项目第一版:读好书、练思维、爱编码
```

# 6. 个性化 UI 设计中对鼠标交互的设计开发

## 6.1 分析与设计

在之前设计的基础上，进行了鼠标 UI 设计，达到了对鼠标模型的分析与设计。通过 mouseDown、mourseup、mourseout 事件，从而达到对鼠标的控制，实现对图书封面拖动的效果。如图 6-1 用例图所示。

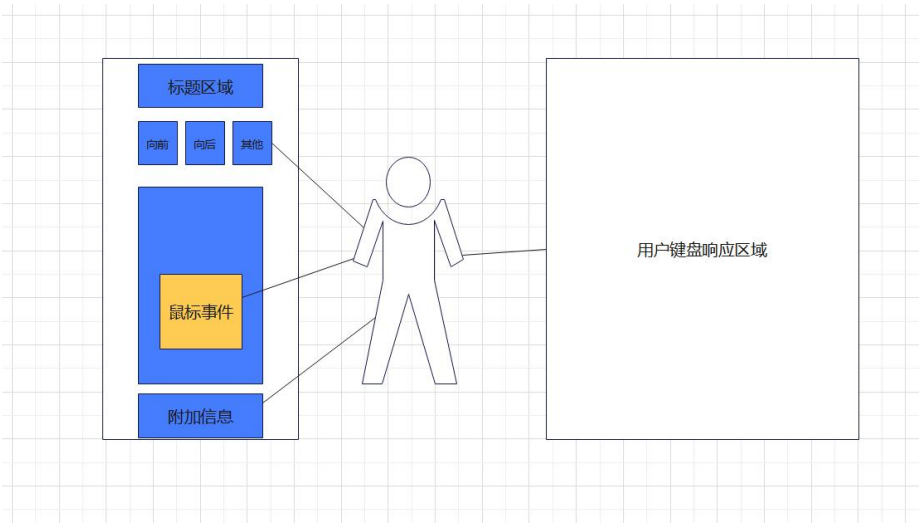



图 6-1 用例图

## 6.2 项目的实现和编程

一、HTML 代码编写如下：

```
<body >
  <header>
    <p id="book">
        浪漫人间总是蓄满鲜花
    </p>
   </header>
  <nav>
   <button>向前</button>
   <button>向后</button>
   <button>暂停</button>
  </nav>


   <main id="main">
  <div id="bookface">
        花的种类
    </div>
   </main>
  <footer>

  CopyRight from 刘嘉舒 江西科技师范大学 2022--2025
  </footer>
  <div id="aid">
    <p>用户键盘响应区</p>
  </div>
```

二、CSS 代码编写如下：

```
  *{
    margin: 10px;
    text-align: center;
  }
  header{
    border: 3px solid #fab1c5;
    height: 10%;
    font-size: 1em;

  }
  nav{
    border: 3px solid #f5fc89;
```

```
    height: 10%;
  }
  main{
    border: 3px solid #fab1c5;
    height: 70%;
    font-size: 0.8em;
    position: relative;
  }
```

```
  #box{
    position: absolute;
    right: 0;
    width: 100px;
  }
```

```
  footer{
    border: 3px solid #f5fc89;
    height:10%;
    font-size: 0.7em;
  }
  body{
    position: relative;
  }
  #aid{
    position: absolute;
    border: 3px solid rgb(160, 192, 251);
    top:0px;
    left:600px;
  }
  #bookface{
    position: absolute;
    width: 80%;
    height: 80%;
    border:1px solid rgb(162, 255, 198);
    background-color:#f5f9b4;
    left:7% ;
    top: 7% ;
  }
</style>
```

三、JavaScript 代码编写如下：

通过 mouseDown、mourseup、mourseout 这三个事件，达到对图书封面的拖动、以及完成对鼠标事件的反馈。

```
 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.appWidthbaseFont + "px";
  document.body.style.height = UI.appHeightbaseFont4 + "px";
if(window.innerWidth<1000){
    $("aid").style.display='none';
}
  $("aid").style.width=window.innerWidth-UI.appWidthbaseFont3 +'px';
  $("aid").style.height= UI.appHeightbaseFont3 +'px';
//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%' ;
```

```
  }
});
$("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.pageXmouse.x );
     $("bookface").textContent= "正在拖动鼠标，距离：" + mouse.deltaX +"px 。";
     $('bookface').style.left = mouse.deltaX + 'px' ;
  }
})
  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 $
  </script>
```

## 6.3 项目的运行和测试

项目的运行与测试，我进行了 PC 端与移动端分别进行运行测试。如图 6-2 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的二维码，如图 6-3 所示，移动端的效果图如图 6-4 所示，运行测试本项目的第二次开发的阶段性效果。

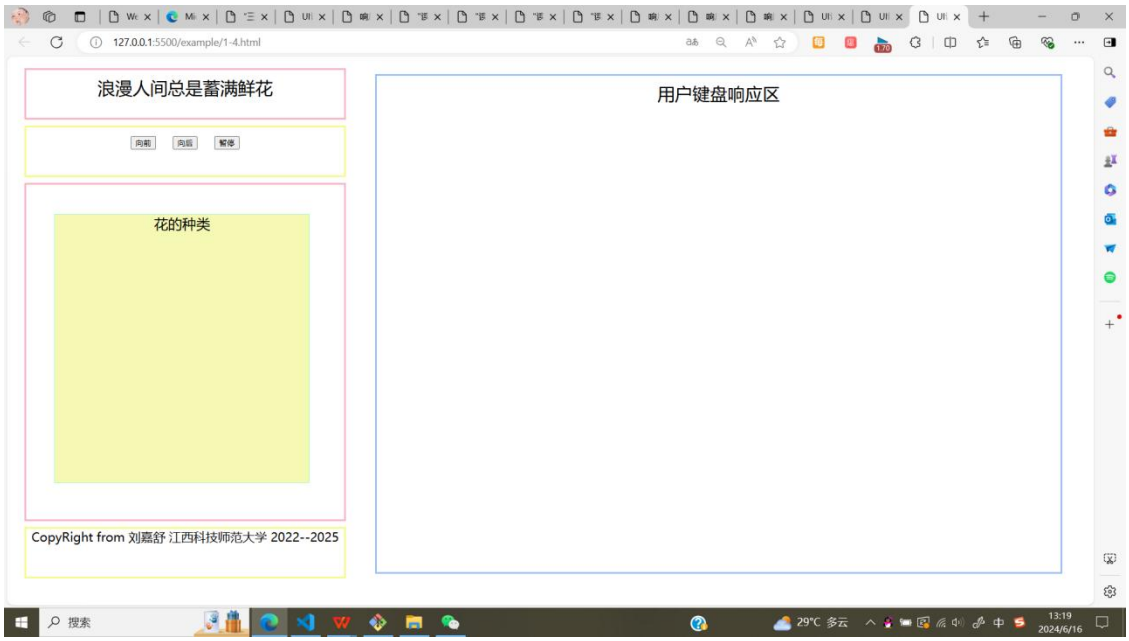



图 6-2


图 6-3

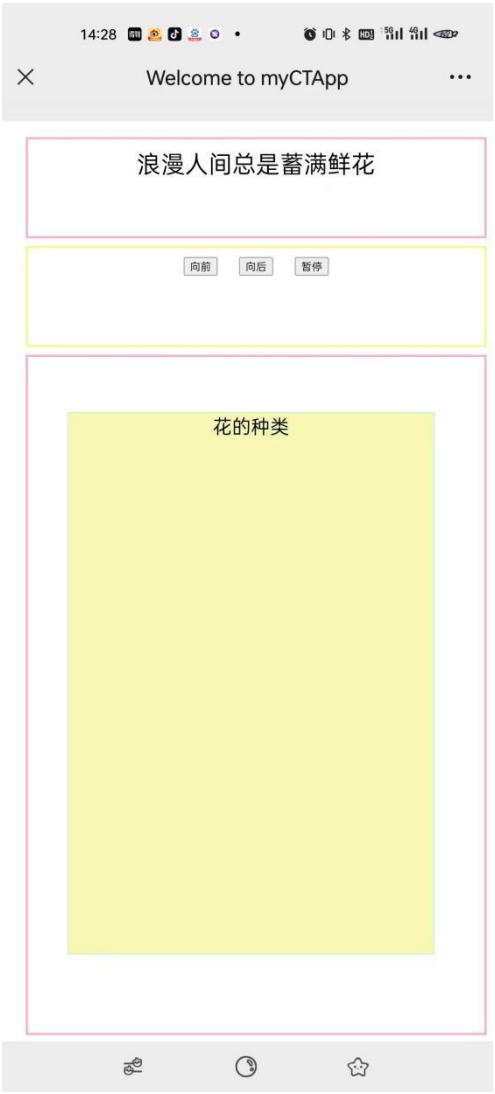



图 6-4

## 6.4 项目的代码提交和版本管理

编写好 1.4html 的代码，测试运行成功后，执行下面命令提交代码：

```
hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ touch 1-4.html

hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ git add 1-4.html

hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ git commit -m 第四版：在之前的基础上增加了鼠标的UI设计，达到了对鼠标模型的分析与设计，通过对mouseDown、mouseup、mourseout三个事件控制，实现对照片的拖动，并且对鼠标事件进行反馈。
[master 4d9097e] 第四版：在之前的基础上增加了鼠标的UI设计，达到了对鼠标模型的分析与设计，通过对mouseDown、mouseup、mourseout三个事件控制，实现对照片的拖动，并且对鼠标事件进行反馈
。
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 1-4.html
```

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所示：

```
hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ git log
commit 4d9097e74904dd77b6022ed3c4c7f2ca7ef450e9 (HEAD -> master)
Author: liujiashu666 <LJS@abcljs>
Date:   Sun Jun 16 13:27:13 2024 +0800

    第四版：在之前的基础上增加了鼠标的UI设计，达到了对鼠标模型的分析与设计，通过对mouseDow
n、mouseup、mourseout三个事件控制，实现对照片的拖动，并且对鼠标事件进行反馈。

commit df1f680397d550e373646c372773cb9ba44c180e
Author: liujiashu666 <LJS@abcljs>
Date:   Sun Jun 16 13:12:59 2024 +0800

    第三版：在之前的基础上，完成了对宽屏和窄屏的响应式设计，增加了一个用户键盘响应区域，并
且抓取到了鼠标这个对象，获取到了鼠标的坐标信息，尝试对鼠标进行UI设计控制。
```

# 7. 对触屏和鼠标的通用交互操作的设计开发

## 7.1 分析与设计

在之前代码的基础上，使用通用的 UI 设计，用一套代码同时为触屏和鼠标建模。在这个代码中，我规定了滑动/拖动超过 100px 才被视作为 UI 互动，并且实现了在用户键盘响应区域的字符响应。如图 6-1 用例图所示。

## 7.2 项目的实现和编程

一、HTML 代码编写如下：

```
<body >
 <header>
   <p id="book">
       浪漫人间总是蓄满鲜花
   </p>
 </header>
<nav>
 <button>向前</button>
 <button>向后</button>
 <button>其他</button>
</nav>



 <main id="main">
<div id="bookface">
      这是花的种类<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 刘嘉舒 江西科技师范大学 2024--2025
</footer>
<div id="aid">
  <p>用户键盘响应区</p>
</div>
```

## 二、CSS 代码编写如下：

```
*{
  margin: 10px;
  text-align: center;
}
header{
  border: 3px solid #fab1c5;
  height: 10%;
  font-size: 1em;

}
nav{
  border: 3px solid #f5fc89;
  height: 10%;
}
main{
  border: 3px solid #fab1c5;
  height: 70%;
  font-size: 0.8em;
  position: relative;
}
#box{
  position: absolute;
  right: 0;
  width: 100px;
}
footer{
  border: 3px solid #f5fc89;
  height:10%;
  font-size: 0.7em;
}
```

```
    body{
      position: relative;
    }
    button{
     font-size:1em;
    }
    #aid{
      position: absolute;
      border: 3px solid rgb(160, 192, 251);
      top:0px;
      left:600px;
    }
    #bookface{
      position: absolute;
      width: 80%;
      height: 80%;
      border:1px solid rgb(162, 255, 198);
      background-color: #f5f9b4;
      background-image: url(../example/abc.jpg);
        left:7% ;
      top: 7% ;
    }
  </style>
  </head>
```

三、JavaScript 代码编写如下：

设置了两个全局变量：UI 和 Pointer,UI 获取 innerWidth 和 innerHeight 这两个事件，进而是实现纵向全屏和字体自适应；Pointer 获取 isDown 和 touch，从而达到对触屏滑动事件的响应。并且使用`addEventListener`来监听`keydown`, `keyup`, 或 `keypress`事件，达到对用户键盘的响应。

```
  <script>
    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.appWidthbaseFont + "px";
  document.body.style.height = UI.appHeightbaseFont4 + "px";
if(window.innerWidth<1000){
    $("aid").style.display='none';
}
  $("aid").style.width=window.innerWidth-UI.appWidthbaseFont3 +'px';
  $("aid").style.height= UI.appHeightbaseFont3 +'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].pageXPointer.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.pageXPointer.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
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 $
    </script>
```

## 7.3 项目的运行和测试

项目的运行与测试，我进行了 PC 端与移动端分别进行运行测试。如图 7-1 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的二维码，如图 7-2 所示，移动端的效果图如图 7-3 所示，运行测试本项目的第二次开发的阶段性效果。

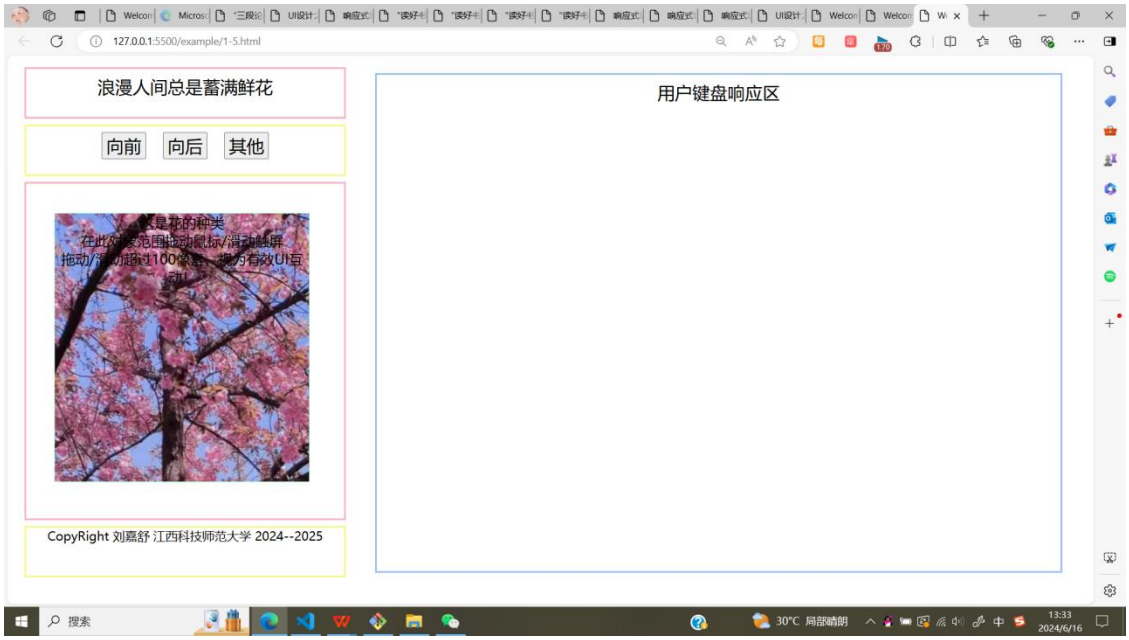



图 7-1


扫描 QR 码
https://liujiashu666.github.io/1-5.html
复制
下载

图 7-2

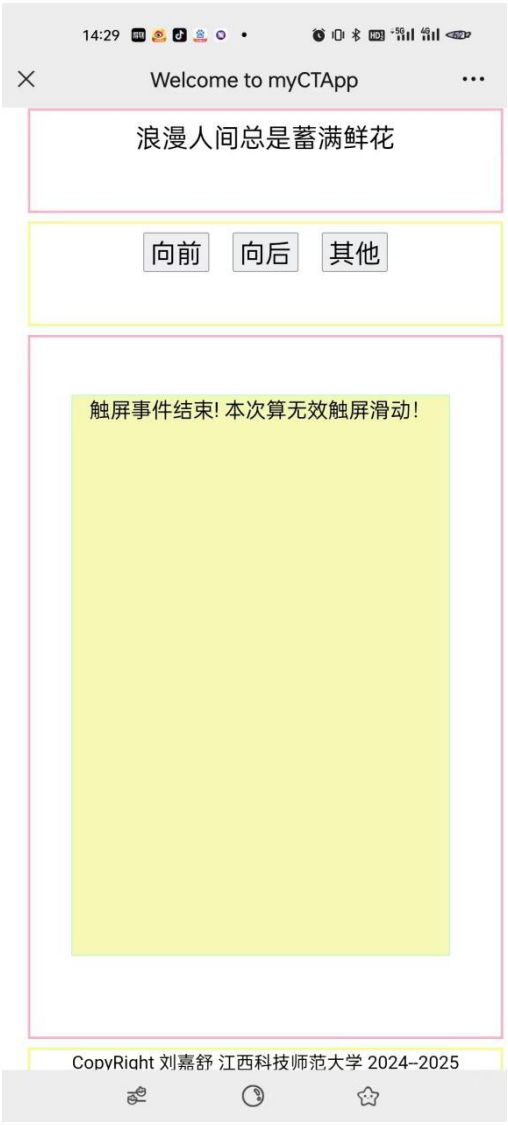

14:29
Welcome to myCTApp
浪漫人间总是蓄满鲜花
向前
向后
其他
触屏事件结束! 本次算无效触屏滑动!
CopyRight 刘嘉舒 江西科技师范大学 2024--2025

图 7-3

## 7.4 项目的代码提交和版本管理

编写好 1.5html 的代码，测试运行成功后，执行下面命令提交代码：

```
hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ touch 1-5.html

hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ git add 1-5.html

hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ git commit -m  第五版：在之前代码的基础上，增加了风景照是否有效拖动的响应，并且抓取了键盘的事件的产生，使其在用户键盘的相应区域的反馈，并且同时满足PC端和移动端运行。
[master 7d8a33c] 第五版：在之前代码的基础上，增加了风景照是否有效拖动的响应，并且抓取了键盘的事件的产生，使其在用户键盘的相应区域的反馈，并且同时满足PC端和移动端运行。
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 1-5.html
```

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所示：

```
hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ git log
commit 7d8a33c0dddd4852f808a2f6ed4c365c724a144c (HEAD -> master)
Author: liujiashu666 <LJS@abcljs>
Date:   Sun Jun 16 13:38:22 2024 +0800

    第五版：在之前代码的基础上，增加了风景照是否有效拖动的响应，并且抓取了键盘的事件的产生，使其在用户键盘的相应区域的反馈，并且同时满足PC端和移动端运行。

commit 4d9097e74904dd77b6022ed3c4c7f2ca7ef450e9
Author: liujiashu666 <LJS@abcljs>
Date:   Sun Jun 16 13:27:13 2024 +0800

    第四版：在之前的基础上增加了鼠标的UI设计，达到了对鼠标模型的分析与设计，通过对mouseDown、mouseup、mouseout三个事件控制，实现对照片的拖动，并且对鼠标事件进行反馈。
```

# 8. UI 的个性化键盘交互控制的设计开发

## 8.1 分析与设计

在之前代码的基础上，实现了利用 keydown 和 keyup 两个底层事件，实现同时输出按钮状态和文本内容。并且处理了连续空格和制表键 tap。使用户键盘响应式设计更加完善。如图 7-1 用例图所示。

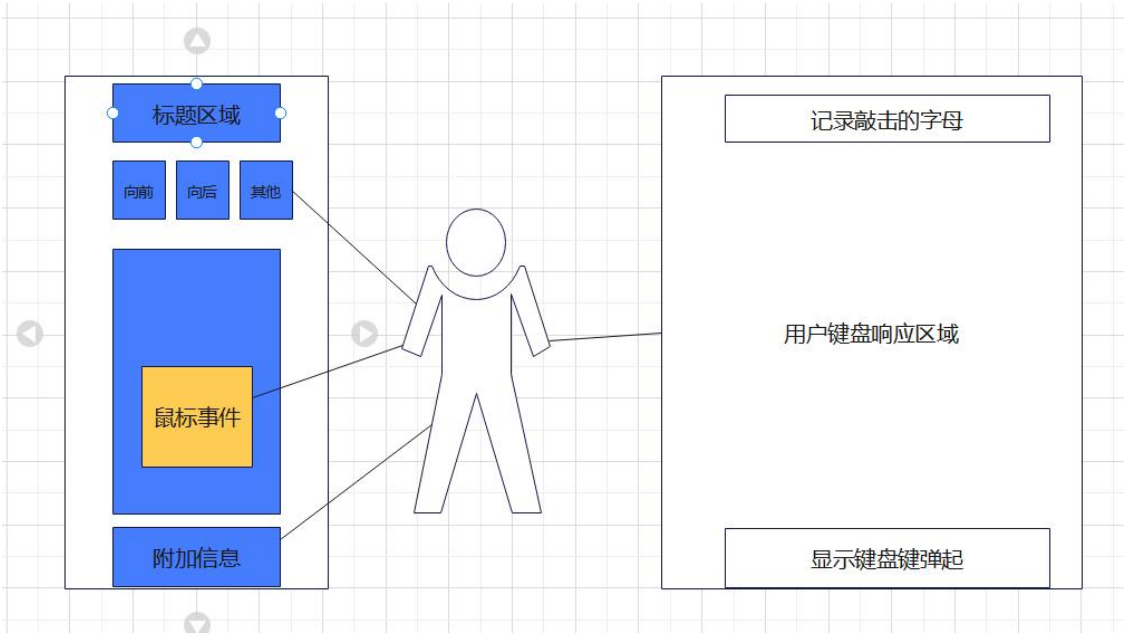



图 7-1 用例图

## 8.2 项目的实现和编程

一、HTML 代码编写如下：

```
 <title>Welcome to myCTApp</title>
</head>

<body>
    <header>
        浪漫人间总是蓄满鲜花
    </header>
    <nav>
        <button>导航一</button>
        <button>导航二</button>
        <button>导航三</button>
    </nav>
    <main id="main">
        <div id="bookface">
            这是花的种类
        </div>
    </main>
    <footer>
        刘嘉舒 江西科技师范大学 2022--2025
    </footer>
    <div id="aid">
        用户键盘响应区
```

```
        <div id="outputText"></div>
        <div id="keyStatus"></div>
    </div>
```

二、CSS 代码编写如下：

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

        body {
            position: relative;
        }
        header {
            height: 15%;
            border: 2px solid #fab1c5;
            font-size: 1.6em;
        }
        main {
            height: 70%;
            border: 2px solid #fab1c5;
            font-size: 1.2em;
            background-size: cover;
            position: relative;
        }
        #bookface {
            width: 80%;
            height: 80%;
            border: 1px solid rgb(162, 255, 198);
            background-color: #f5f9b4;
            background-image: url(../example/abc.jpg);
            position: absolute;
            left: 8%;
            top: 8%;
        }
        nav {
            border: 2px solid #f5fc89;
            height: 10%;
            font-size: 1.1em;
        }
        footer {
            min-height: 5%;
```

```
            border: 2px solid #f5fc89;
        }
        #aid {
            position: absolute;
            left: 600px;
            top: 0;
            border: 3px soli rgb(0, 246, 193);
;
        }
        #outputText {
            color: rgb(0, 163, 245);
            word-break: break-all;
            border: 1px solid rgb(0, 246, 193);
            height: 10%;
            width: 95%;
        }
        #keyStatus {
            position: absolute;
            bottom: 0;
            border: 1px solid rgb(0, 246, 193);
            width: 90%;
            height: 10%;
        }
    </style>
```

三、JavaScript 代码编写如下：

通过研究 keydown 和 keyup 两个事件，实现了同时输出按钮和文本内容，//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，如“F5 刷新页面/Ctrl+R ”、“F12 打开开发者面板”等也不再被响应。处理了连续空格和制表键 tab 的问题。

```
//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，如“F5 刷新页面/Ctrl+R ”、“F12 打开开发者面板”等也不再被响应
    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 ;
        }
      }
      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 $
```

```
</script>
```

## 8.3 项目的运行和测试

项目的运行与测试，我进行了 PC 端与移动端分别进行运行测试。如图 7-2 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的二维码，如图 7-3 所示，移动端的效果图如图 7-4 所示，运行测试本项目的第二次开发的阶段性效果。

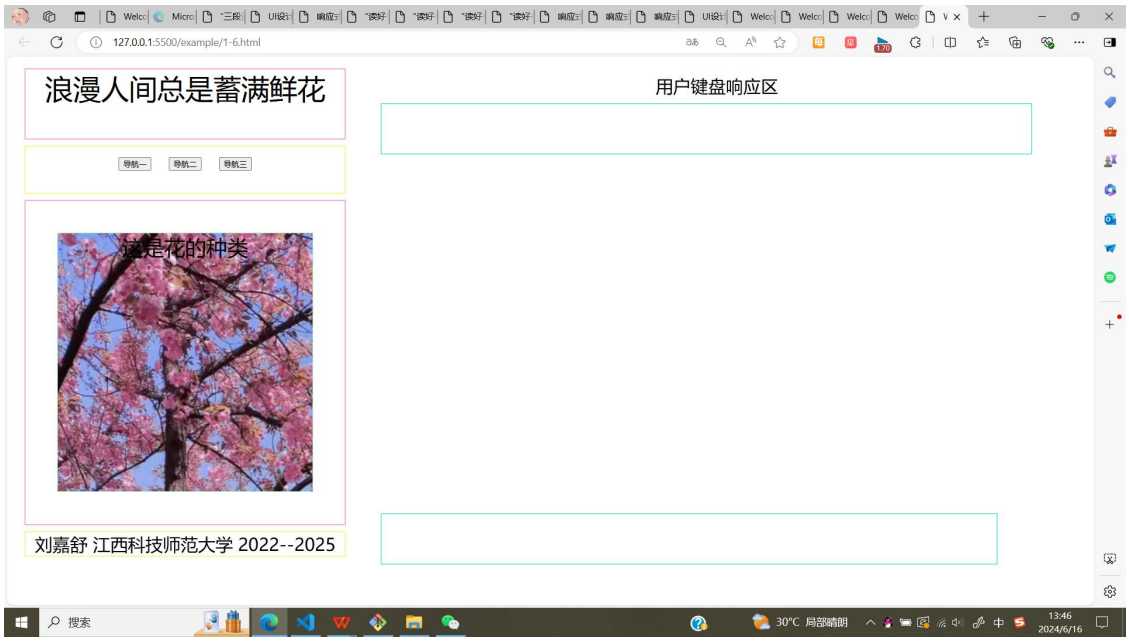



图 7-2


图 7-3


图 7-4

## 8.4 项目的代码提交和版本管理

编写好 1.6html 的代码，测试运行成功后，执行下面命令提交代码：

```
hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ touch 1-6.html

hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ git add 1-6.html

hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ git commit -m 第六版：在之前代码的基础上，利用keydown和keyup两个底层事件实现了同时输出按钮状态和文本内
容，并且处理了连续空格和制表键tap，完善了用户键盘响应区域。
[master fcde107] 第六版：在之前代码的基础上，利用keydown和keyup两个底层事件实现了同时输出按钮状态和文本内
容，并且处理了连续空格和制表键tap，完善了用户键盘响应区域。
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 1-6.html
```

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所示：

```
hp@LAPTOP-Q3M6BAIF MINGW64 /WebUI (master)
$ git log
commit fcde107841fb01c96a46480ea3a8c04a28244372 (HEAD -> master)
Author: liujiashu666 <LJS@abcljs>
Date:   Sun Jun 16 13:52:10 2024 +0800

    第六版：在之前代码的基础上，利用keydown和keyup两个底层事件实现了同时输出按钮状态和文本内容，并且处理了连续空格和制表键tap，完善了用户键盘响应区域。

commit 7d8a33c0ddd4852f808a2f6ed4c365c724a144c
Author: liujiashu666 <LJS@abcljs>
Date:   Sun Jun 16 13:38:22 2024 +0800

    第五版：在之前代码的基础上，增加了风景照是否有效拖动的响应，并且抓取了键盘的事件的产生，使其在用户键盘的相应区域的反馈，并且同时满足PC端和移动端运行。
```

# 9. 谈谈本项目中的高质量代码

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.appWidthbaseFont + "px";
document.body.style.height = UI.appHeightbaseFont5 + "px";
```

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){
```

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，如“F5 刷新页面/Ctrl+R ”、“F12 打开开发者面板”等也不再被响应

```
ev.preventDefault() ;
  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 ;
        }
      }
      return false ;
      } //function printLetter(k)
    });
```

创建一个 Pointer 对象，践行 MVC 设计模式，设计一套代码同时对鼠标和触屏实现控制。面向对象思想，封装，抽象，局部变量，函数式编程，逻辑。

（围绕着抽象定义函数、代码块、模型设计以及降低全局变量的使用来写）

# 10. 用 gitBash 工具管理项目的代码仓库和 http 服务器

## 10.1 经典 Bash 工具介绍

Bash（Bourne-Again SHell）是一个广泛使用的 Unix shell，它提供了许多内置命令和功能，使得用户可以通过命令行接口与操作系统交互。

它的由来可以追溯到早期的 Unix 系统。最初的 Unix 系统中，有一款名为 Bourne Shell(sh)的命令行解释器，由 Steve Bourne 在 1978 年开发。Bourne Shell 因其简洁和效率而受到欢迎，成为 Unix 系统上的标准 shell 之一。 随着时间的推移，用户开始希望增加一些额外的功能，比如命令历史、命令别名、文件名扩

展和作业控制等。于是，其他开发者推出了 C Shell（csh）和 Korn Shell（ksh），这两者都引入了 Bourne Shell 所缺乏的一些特性。 在 1980 年代末，为了融合 Bourne Shell 的简洁性和 C Shell、Korn Shell 的先进特性，Brian Fox 在 GNU 计划中开发了 Bash。Bash 的名字是 Bourne Again SHell 的缩写，意指它是 Bourne Shell 的一个更新版，包含了 C Shell 和 Korn Shell 的一些功能。 Bash 在 1989 年首次发布，成为了 GNU/Linux 系统上的默认 shell。它不仅继承了 Bourne Shell 的语法，还引入了命令历史记录、命令补全、别名、函数、数组等特性。随着时间的推移，Bash 逐渐成为最广泛使用的 Unix 和 Linux shell，尤其在开源社区中非常流行。至今，Bash 仍然是很多 Linux 发行版的默认 shell。

它被广泛用于以下场景：

1. 命令行交互：用户通过命令行界面（CLI）与操作系统进行交互，执行各种系统命令、管理文件和目录、运行程序等。

2. 脚本编写：Bash 脚本是一种编写自动化任务的强大工具，可以用于执行一系列命令、文件操作、数据处理、系统维护和配置管理。

3. 系统管理：系统管理员使用 Bash 脚本来自动化日常任务，如备份、日志分析、用户管理、服务监控和配置更新。

4. 软件部署：在持续集成和持续部署（CI/CD）流程中，Bash 脚本用于构建、测试和部署应用程序。

5. 自动化测试：测试工程师使用 Bash 编写测试脚本，尤其是针对命令行工具和系统的自动化测试。

6. 数据处理：数据科学家和工程师利用 Bash 进行数据预处理，如文件转换、过滤、排序和统计分析。

7. 网络管理：网络管理员通过 Bash 脚本进行网络设备配置、网络服务监控和故障排除。

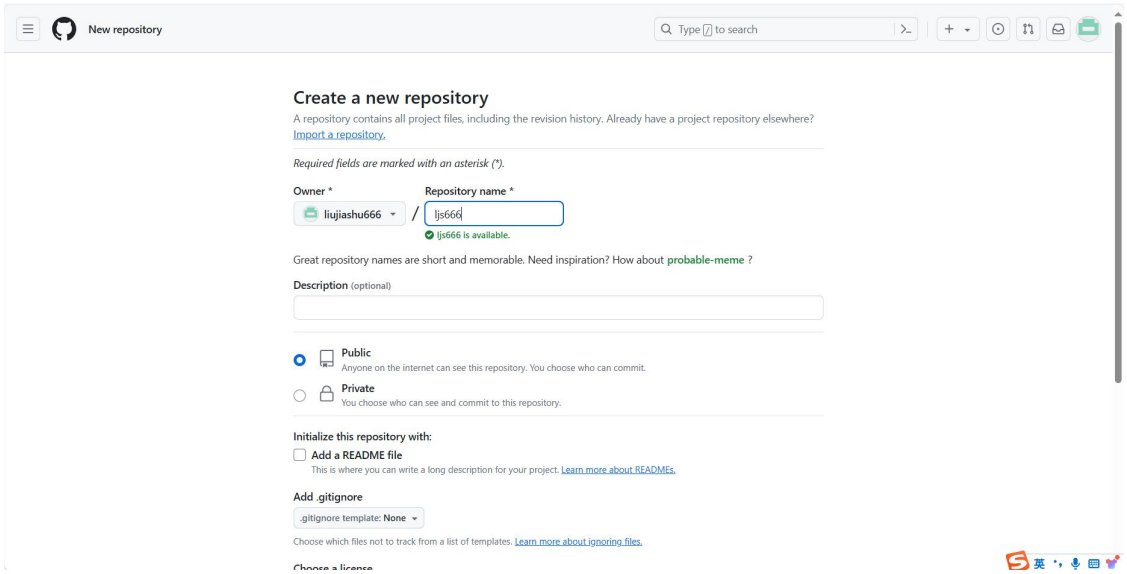
8. 教学和学习：由于其普及性和实用性，Bash 被广泛用于教授命令行操作和脚本编写基础。

9. 系统启动和初始化：在系统启动过程中，Bash 脚本用于执行系统级别的初始化任务，如启动服务、设置环境变量等。

10. 集成开发环境（IDE）和终端模拟器：许多 IDE 和终端模拟器支持 Bash 作

为内部命令行环境，方便开发者在图形界面下使用 Bash 功能。 Bash 的灵活性和广泛支持使得它成为了 Linux 生态系统中不可或缺的一部分，无论是在个人电脑还是服务器上，都能看到它的身影。

## 10.2 创建一个空的远程代码仓库





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。

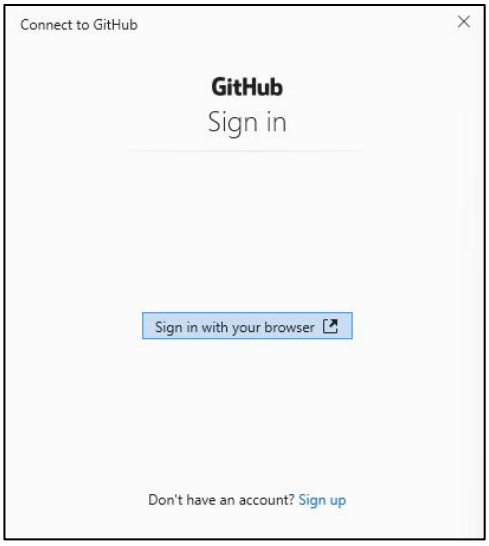
## 10.3 设置本地仓库和远程代码仓库的链接

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

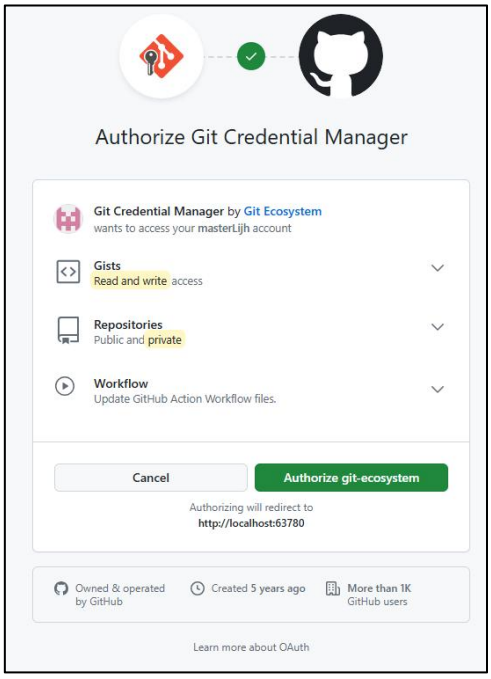
```
$ echo "WebUI 应用的远程 http 服务器设置" >> README.md
$ git init
$ git add README.md
```

```
$ git commit -m "这是我第一次把代码仓库上传至 gitHub 平台"
$ git branch -M main
$ git remote add origin
       https://liujiashu666.github.io/1-1html
$ git push -u origin main
```

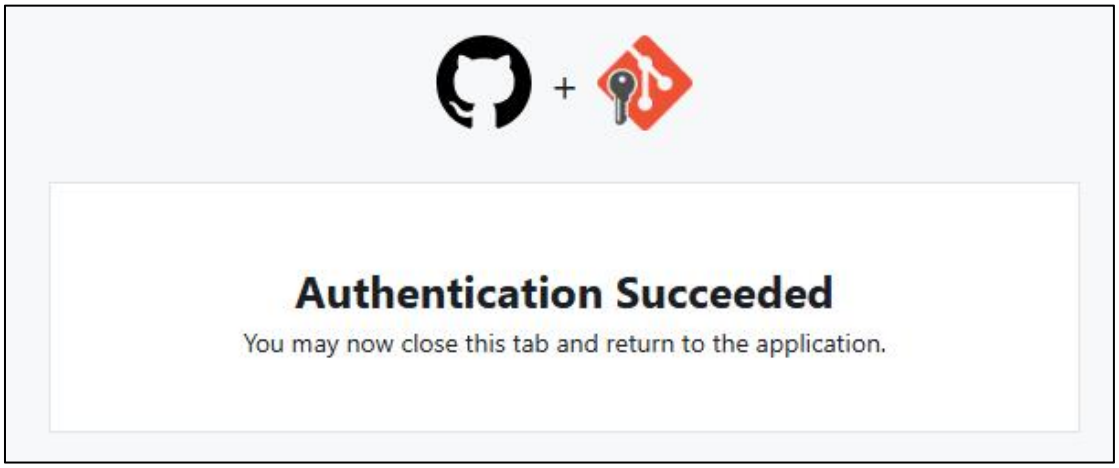
本项目使用 window 平台，gitbash 通过默认浏览器实现密钥生成和记录，第一次链接会要求开发者授权，如下图所示：




再次确认授权 gitBash 拥有访问改动远程代码的权限，如下图所示：

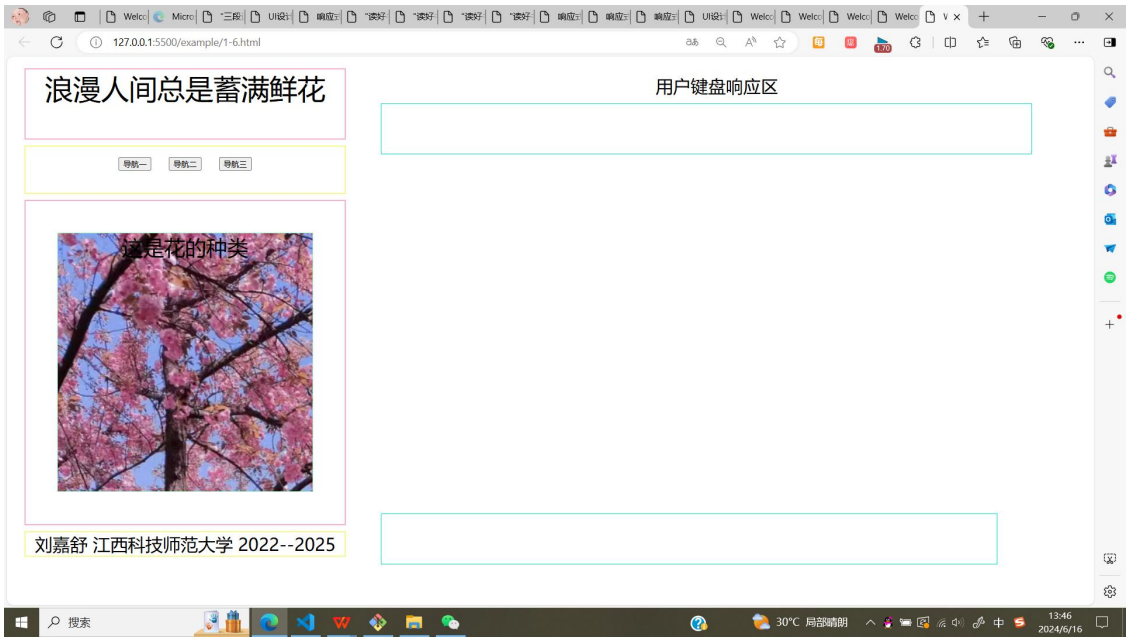


最后，GitHub 平台反馈：gitBash 和 gitHub 平台成功实现远程链接。




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 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：




全文完成，谢谢！

# 参考文献

[1] W3C. W3C's history. W3C Community. [EB/OL]. https://www.w3.org/about/. https://www.w3.org/about/history/. 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