

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：熊金菁

学    号：20213669

指导教师：李建宏

年    月    日

**摘要：**实质上是写三段论，第一段是写事件、研究的背景或目标，第二段是写你所开展的工作和工作量，第三段是描述你完成工作所使用的方法与你工作的意义。

Web 近十年来，以 html5 为核心的 web 标准软件开发技术以跨平台、开源的优势广泛的运用于应用开发的各个领域。本项目以 web 客户端技术为技术路线展开对程序设计和软件开发的为究与实践，通过广泛地查阅了相关的专业技术书籍、开发者论坛文献，在此基础上开发了一个个性化的用户界面（UI）的应用程序。

（2）在开发中，本项目直接使用了 web 客户端最底层的 api，综合运用了 html 进行内容建模，css 语言展开 UI 的样式外观设计，JavaScript 语言编程实现 UI 的用户交互功能。本项目采用了响应式编程，可以智能的适应用户屏幕多样化的需求；良好的贯彻了面向对象编程的程序设计思想，其亮点在运用自己代码建立了通用的 pointer 模型，该模型用一套代码就实现了对鼠标和触屏的控制，实现了适应多样化设备的高质量代码。从工程管理的角度本项目采用了增量式开发模式，反复进行了 6 遍的 ADIT（解释 adit），较为愉快且逐步求精的实现了项目的设计与完善，并通过了测试。从代码的开源和分享的角度看，本项目采用了 git 工具采用了项目的版本管理，本项目在在开发过程中重构代码 6 次，并正式提交了 6 次代码，在测试中提交了 2 次代码，最后利用 gitbash 把代码仓库上传到著名的 github 上，利用 github 提供的 http 服务器实现了本 UI 运用程序的高效部署，通过二维码实现了全球的便捷访问。

**关键字：**gitbash；html；web；adit

## 1.前言

毕业设计是大学生在大学期间最后一个最为重要的一次综合性实践环节，其宗旨在于培养与检测大学生实际解决问题的能力。毕业设计是学生独立完成一个其专业的一个具有价值的项目，从选题到设计到实施到最后的总结与展示皆为独立完成，在这一过程中学生可以检测出自己的不足从而达到完善自身能力的目的。通过一名学生的毕业设计，我们可以以点见面的了解到该学生的对于专业领域的独特见解与其想要表达的内涵这正是我认为毕业设计之所以独一无二的原因。对于计算机科学与技术专业的学生来说，我们通常通过写出一个程序系统，在构建系统的过程中运用到专业的技术与知识，从而展示出自己对于计算机科学的理解与认识，在这过程中我们需要考虑程序的实用性与创新性。对于计算机专业的学生来说，制定一个有效的学习计划和学习方法至关重要。以下是一些建议，旨在帮助计算机专业的学生更好地规划学习，提升学习效果。一、学习计划明确学习目标：首先，学生需要明确自己的学习目标，例如掌握某种编程语言、深入理解数据结构与算法、了解网络原理等。有了明确的目标，学生才能更有针对性地制定学习计划。划分学习阶段：将学习目标划分为若干个阶段，每个阶段设定具体的学习内容和时间节点。例如，第一阶段学习编程语言基础，第二阶段学习数据结构等。

制定时间表：根据学习阶段，制定详细的时间表，包括每天的学习时间、学习内容和休息时间。确保时间表既充实又合理，避免过度劳累。设定优先级：在学习过程中，学生可能会遇到多个任务或项目。为了保持高效，学生需要设定优先级，优先完成重要且紧急的任务。评估与调整：定期评估学习计划的执行情况，根据实际情况调整学习计划。例如，如果发现某个阶段的学习内容过于复杂，可以适当延长学习时间或调整学习方法。理论与实践相结合：计算机专业的学习需要注重理论与实践的结合。学生不仅要掌握理论知识，还要通过实践来巩固所学

知识。例如，学习编程语言时，可以通过编写实际项目来锻炼编程能力。主动学习：学生应该积极参与课堂讨论，主动向老师或同学请教问题。此外，还可以利用网络资源进行自主学习，如观看教学视频、阅读技术博客等。动手实践：计算机专业的学习需要大量的实践。学生应该多动手实践，通过编写代码、搭建网站等方式来锻炼自己的技能。实践不仅可以加深理解，还可以培养解决实际问题的能力。

归纳总结：在学习过程中，学生应该善于归纳总结所学知识。可以通过制作思维导图、撰写学习笔记等方式来巩固所学知识，并发现其中的规律和联系。寻求反馈：在学习过程中，学生应该积极寻求他人的反馈。可以向老师、同学或在线社区请教问题，了解自己的学习情况和不足之处。通过反馈，学生可以及时调整学习策略，提高学习效果。拓展视野：计算机专业的发展日新月异，学生需要不断拓展自己的视野。可以关注行业动态、参加技术交流活动、阅读技术书籍等方式来了解最新的技术趋势和发展方向。这有助于培养学生的创新思维和适应能力。

锻炼团队合作能力：在计算机专业中，团队合作能力至关重要。学生可以通过参与团队项目、与同学合作完成课程作业等方式来锻炼自己的团队合作能力。在合作过程中，学会倾听、沟通和协作，提高自己的综合素质。总之，计算机专业的学生需要制定一个明确的学习计划，并采用合适的学习方法。通过不断学习和实践，提高自己的专业技能和综合素质，为未来的职业发展打下坚实的基础。

## 1.1 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[0]。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。

让我们从对网络的简要描述开始，它是万维网的缩写。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。Web 编程是一个很大的领域，通过不同的工具实现了不同类型的 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）上执行。

HTML，CSS，JavaScript 也是体现人类社会化大生产的分工的智慧，MVC 设计模式，Model：用 HTML 标记语言建模，View：单独用 CSS 语言来实现外观，Controller：用 JavaScript 结合前二者，实现功能层面和微观代码的所有控制。程序是思想的建筑。它造价低廉，没有重量，而且很容易在我们打字的手上生长编写高质量代码要使用模块化设计，采用局部变量防止全局变量过多引起的引用

混乱。

## 2.软件开发的工程管理——增量式开发模式

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。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（图 10.2）。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

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

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。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（图 10.2）。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

本系统开发过程采用的增量开发模式（Incremental Development），其核心思想是将待开发软件系统划分成若干个独立部分，然后按照优先级分批次进行 ADIT（analysis 分析，design 设计，implementation 实施，testing 测试）如下图 2.1 所示。增量开发模式具有减低风险、灵活性高等优点，在本系统的开发中一共进行了六轮的 ADIT 从而较为完善的完成了该系统的部署。

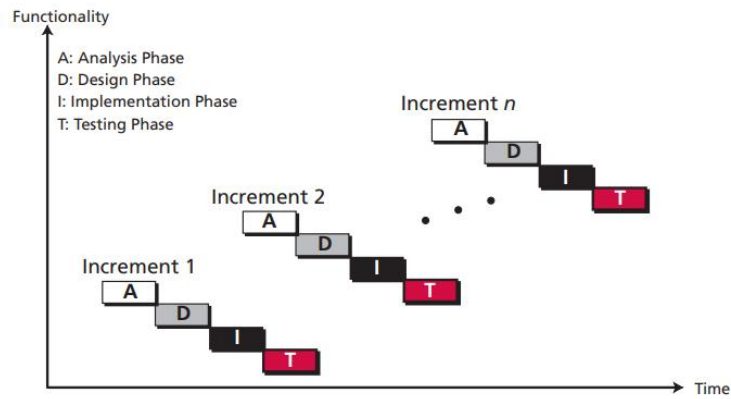


图 2.1

## 3. 内容设计概要

### 3.1 分析与设计

对于项目内容设级，采用了经典的三段论式框架设计。初步划分出项目的三大主题部分，构建出项目基本框架。项目的第一部分为项目标题，放在顶部突出强调本系统的主题、第二部分为系统内容，占用 UI 界面大部分空间，将更多的细节设计具体的呈现给用户，第三部分为作者名称与声明，此部分为第二部分内容的补充，将除了内容外部分细节说明在此展示。项目的三段论式分析过程用例图展示，如图 3.1。项目的三段论式设计过程用 DOM 图展示，如图 3.2。

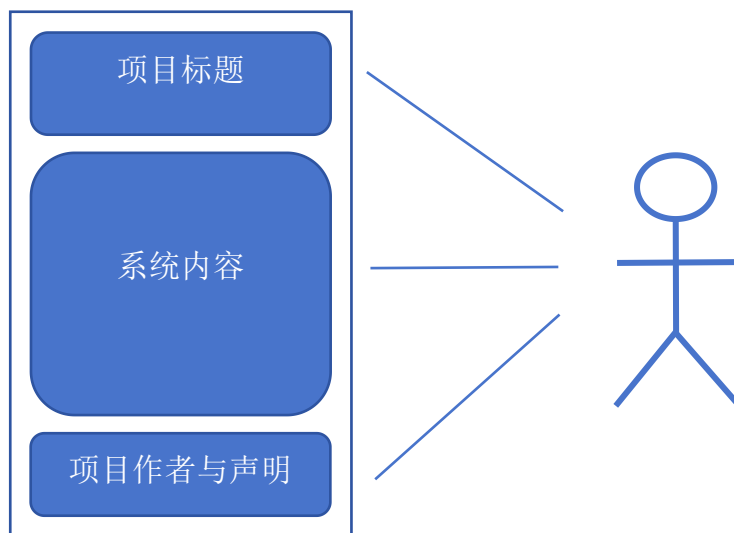


图 3.1

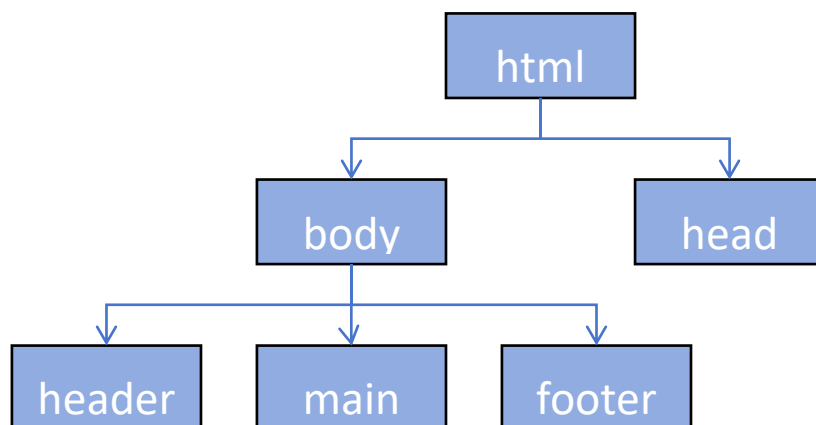


图 3.2

## 3.2 编程与实现

### 3.2.1html 代码编写如下：

```
<title>Welcome to myCTApp</title>
</head>
<body>
  <header>
    <p id="book">
      《自然野生动物鉴赏录》
    </p>
    <nav>
    </nav>
  </header>
  <main>
    <p id="statusInfo">
      ‘自然界的夜，美妙、恬静，却又有无数生灵在窃窃私语’ @daozizi 自然动物鉴赏录
    </p>
  </main>
  <footer>
    CopyRight from 熊金菁 江西科技师范大学 2022--2025
  </footer>
</body>
```

### 3.2.1css 代码编写如下：

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

```
main {  
  border: 2px solid blue;  
  height: 400px;  
}  
footer {  
  border: 2px solid blue;  
  height: 200px;  
}
```

### 3.3 测试与运行

本次测试主要是检测页面的内容、结构交互等方面的正确性。Pc 端采用的运行方式是使用浏览器直接打开 html 文件的方式，移动端则采用使用浏览器的模拟功能进行移动端的模拟运行。Pc 端效果图如图 3.4，移动端效果图如图 3.5。



图 3.4

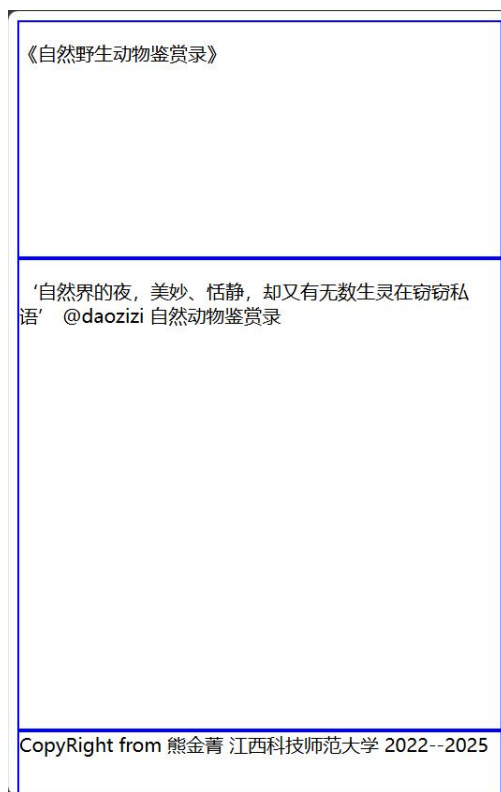


图 3.5

### 3.4 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：开发者姓名以及邮箱。

进入 gitBash 命令行后，按次序输入以下命令：

```
$ cd /  
$ mkdir xiongjj  
$ cd xiongjj  
$ git init  
$ git config user.name 熊金菁  
$ git config user.email 2276781768@qq.com  
$ touch index.html myCss.css
```

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

```
$ git add 1.1.html  
$ git commit -m 项目第一版：“三段论”式的内容设计概要开发
```

成功提交代码后，gitbash 的反馈如下所示：

```
$ git commit -m 项目第一版：“三段论”式的内容设计概要开发  
[master (root-commit) 32de024] 项目第一版：“三段论”式的内容设计概要开发  
2 files changed, 46 insertions(+)  
create mode 100644 index.html  
create mode 100644 myCss.css
```



项目代码仓库自此也开启了历史记录，我们可以输入日志命令查看，

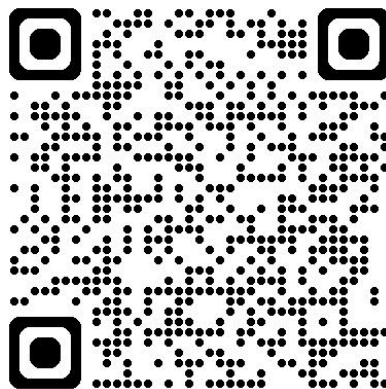
```
$ git log
```

gitbash 反馈代码的仓库日志如下所示：

```
commit ddf49079cc29a75795cc755d4d2933b82c47c9a7 (HEAD -> main)
Author: JJ <xiong>
Date:   Sun Jun 16 15:59:59 2024 +0800
```

项目第一版：三段论式的内容设计概要开发。

二维码如下：

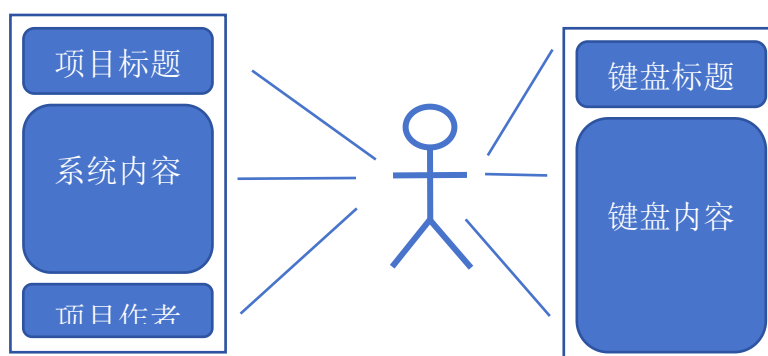


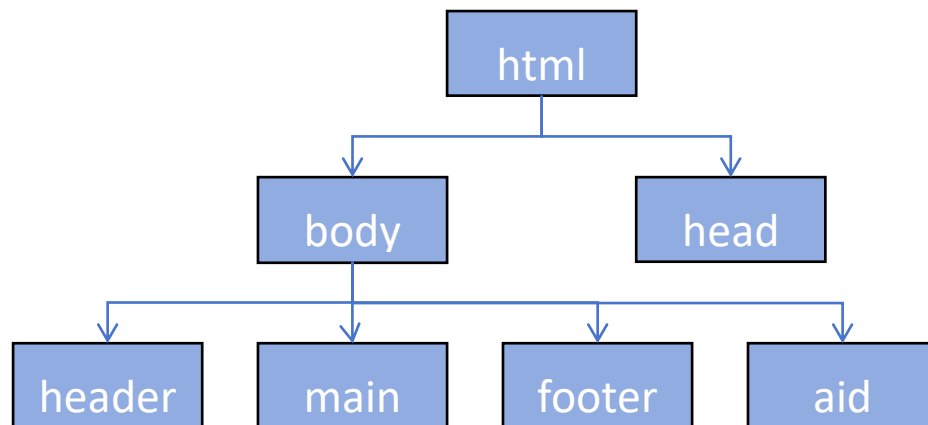
[Daozizi.github.io/1.1.html](https://Daozizi.github.io/1.1.html) at main • [Daozizi/Daozizi.github.io](https://Daozizi.github.io)

## 4. 移动互联时代的响应化设计

### 4.1 分析与设计

第二次设计依旧采用三段论框架，将文章内容填充到了第二部分的主题之中，并且优化的用户交互界面，使得 UI 更加美观。移动互联时代的响应化设计是一种重要的网页设计方法，旨在确保网页能够在各种用户设备上呈现出最佳的视觉效果和交互体验。响应式设计是一种网页设计方法，它基于设备的屏幕尺寸、分辨率和平台特性来自动调整布局和显示内容，从而提供最佳的用户体验。这种设计方法核心理念是“一次设计，到处可用”，意味着开发者只需进行一次设计，就能确保网页在各种设备上都能良好地显示和交互。





## 4.2 编程与实现

### 4.2.1html 代码编写如下：

```
<body>
  <header>
    <p id="book">
      《自然野生动物鉴赏录》
    </p>
  </header>
  <p id="statusInfo">
    <nav>
      <button>下一页 </button><button>1/10</button><button>上一页</button>
    </nav>
  </p>
  <main>
  </main>
  <footer>
    Copyright from 熊金菁 江西科技师范大学 2022--2025
  </footer>
```

### 4.2.2css 代码编写如下：

```
body {
  font-size: 1em;
  text-align: center;
}
header {
  border: 1px solid rgb(236, 253, 139);
  height: 8%;
  background-color: rgb(121, 251, 136);
```

```

    color: black;
}
nav {
    border: 1px solid rgb(236, 253, 139);
    height: 4%;
}
button {
    font-size: 1.3em;
}
p#book {
    font-size: 1.5em;
}
main {
    border: 1px solid rgb(236, 253, 139);
    height: 50%;
    background: url(../images/animal1.jpg);
    background-repeat: no-repeat;
    background-position: center;
    background-size: cover;
}
footer {
    border: 1px solid rgb(236, 253, 139);
    height: 10%;
}

```

## 4.2.2javascript 代码编写如下:

```

<script>
    var UI = {
    }
    UI.deviceWidth = window.innerWidth;
    UI.deviceHeight = window.innerHeight;
    document.body.style.height = UI.deviceHeight + "px";
    document.body.style.fontSize = UI.deviceWidth / 25 + "px";
</script>

```

## 4.3 测试与运行

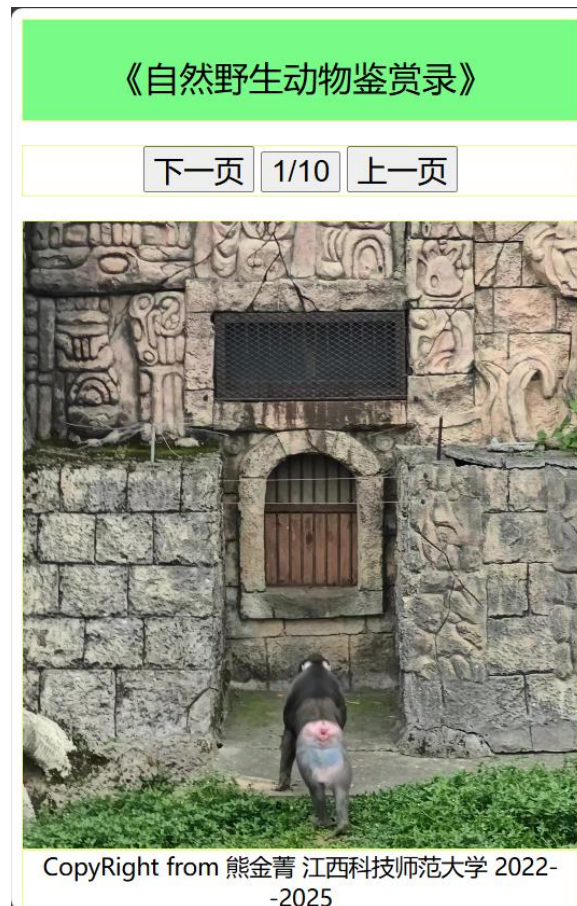


图 4.2

## 4.4 代码提交和版本控制

本项目的文件通过 `gitBash` 工具管理，作为项目的第二次迭代，在代码提交和版本管理环节，我们的目标将第二次迭代项目信息上传，并且将项目的备注注释上。

进入 `gitBash` 命令行后，按次序输入以下命令：

```
$ touch index.html myCss.css
```

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

```
$ git add 1.2.html
```

```
$ git commit -m 项目第二版：移动互联时代的响应式设计
```

成功提交代码后，`gitbash` 的反馈如下所示：

```
岛子哥@DaoziG MINGW64 /d/xiongjj/abc/exp (main)
$ git commit -m"移动互联时代的UI开发初步--窄屏终端的响应式设计"
[main 4f81f9e] 移动互联时代的UI开发初步--窄屏终端的响应式设计
1 file changed, 1 insertion(+), 9 deletions(-)
```

项目代码仓库自此也开启了历史记录，我们可以输入日志命令查看，

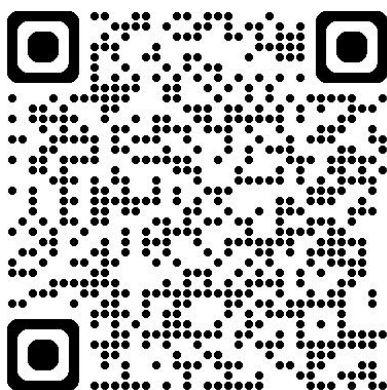
```
$ git log
```

`gitbash` 反馈代码的仓库日志如下所示：

```
岛子哥@DaoziG MINGW64 /d/xiongjj/abc/exp (main)
$ git log
commit 4f81f9efe6b9deedcbb764492c5cb5e5217d01e3 (HEAD -> main)
Author: JJ <xiong>
Date: Sun Jun 16 18:01:15 2024 +0800
```

移动互联时代的UI开发初步——窄屏终端的响应式设计

二维码如下：

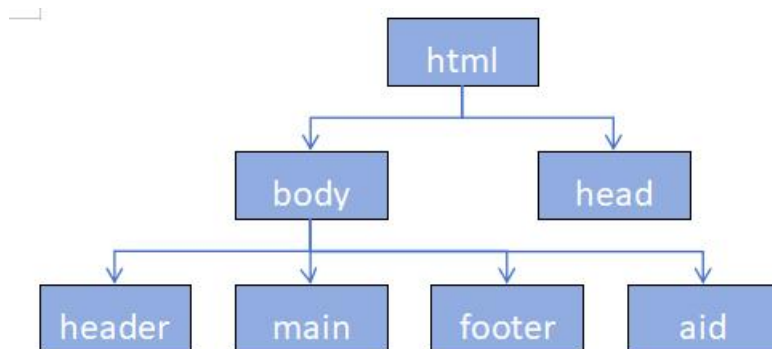
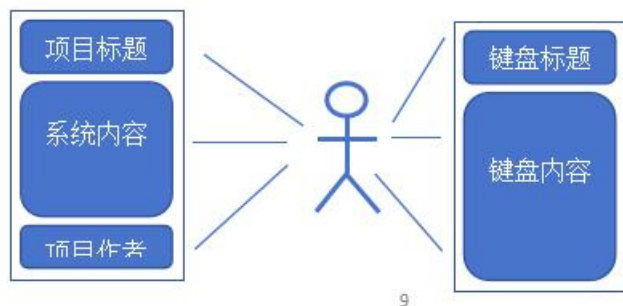


[Daozizi.github.io/1.2.html](https://Daozizi.github.io/1.2.html) at main • [Daozizi/Daozizi.github.io](https://Daozizi.github.io)

## 5. 个性化交互 UI 初步设计——鼠标模型

### 5.1 分析与设计

第三次初步尝试适配平板触屏端，将鼠标模拟为手的滑动，记录下来其位置的变化，为改变主体图像的操作过程添加了另外一种模式。此外还为 pc 端添加了用户键盘响应区，为人机交互提供了多种除了鼠标外的另外一种选择。Pc 端如图 4 所示、移动端如图 5 所示。在 HTML 个性化交互 UI 的初步设计中，鼠标模型是一个重要的考虑因素，它涉及到用户如何通过鼠标与网页进行交互。个性化交互 UI 的初步设计——鼠标模型，旨在通过符合人体工学的设计、技术创新和个性化设置，提高用户的使用体验和效率。通过不断优化和改进设计，我们相信可以为用户带来更加舒适、便捷和个性化的鼠标使用体验



## 5.2 编程与实现

### 5.2.1html 代码编写如下：

```

<body>
  <header>
    <p id="book">
      《自然野生动物鉴赏录》
    </p>
  </header>
  <main id="main">
    <div id="bookface">
    </div>
  </main>
  <footer>
    Copyright from 熊金菁 江西科技师范大学 2022--2025
  </footer>
  <div id="aid">
    <p>用户键盘响应区</p>
    <p id="keyboard"></p>
  </div>

```

## 5.2.2 javascript 代码编写如下:

```
<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.appWidth + "px";
  document.body.style.height = UI.appHeight - 62 + "px";
  if (window.innerWidth < 1000) {
    $("aid").style.display = 'none';
  }
  $("aid").style.width = window.innerWidth - UI.appWidth - 30 + 'px';
  $("aid").style.height = UI.appHeight - 62 + 'px';
  //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;
    let s1 = "按键是: ";
    let s2 = "编码是: "
    $("keyboard").textContent = s1 + k + ";" + s2 + 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>
</body>
</html>

```

### 5.2.3css 代码编写如下图 5.5 所示:

```

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

nav {
    border: 3px solid rgb(171, 220, 171);
    height: 10%;
}

main {
    border: 3px solid rgb(171, 220, 171);
    height: 100%;
    font-size: 0.8em;
}

```



```

    position: relative;
}
#box {
    position: absolute;
    right: 0;
    width: 100px;
}
footer {
    border: 3px solid rgb(171, 220, 171);
    height: 10%;
    font-size: 0.7em;
}
body {
    position: relative;
}
#aid {
    position: absolute;
    border: 3px solid rgb(171, 220, 171);
    top: 0px;
    left: 600px;
}
body {
    font-size: 1em;
    text-align: center;
}
header {
    border: 0.3em solid rgb(171, 220, 171);
    height: 13%;
    background-color: rgb(121, 251, 136);
    color: black;
}
nav {
    border: 1px solid rgb(236, 253, 139);
    height: 4%;
}
button {
    font-size: 1.3em;
}
p#book {
    font-size: 1.5em;
}
main {
    border: 0.3em solid rgb(171, 220, 171);
    height: 82%;
}

```

```

background: url(../../images/animal1.jpg);
background-repeat: no-repeat;
background-position: center;
background-size: cover;
}
footer {
border: 0.3em solid rgb(171, 220, 171);
height: 5%;
}

```

## 5.3 测试与运行

本次测试主要是检测 javascript 代码中逻辑的正确性，观察是否能鼠标移动，模拟平板使用功能，同时还需测试该程序 UI 是否能够响应式的适应多种设备，观察其不同设备中显示效果。Pc 端采用的运行方式是使用浏览器直接打开 html 文件的方式，移动端则采用使用浏览器的模拟功能进行移动端的模拟运行。Pc 端效果图如图 5.6 所示，移动端则选择使用不同设备展示该项目优越的响应式设计效果，效果图如图 5.7 所示。



图 5.6



图 5.7

## 5.4 代码提交和版本控制

本项目的文件通过 gitBash 工具管理，作为项目的第二次迭代，在代码提交和版本管理环节，我们的目标将第二次迭代项目信息上传，并且将项目的备注注释上。

进入 gitBash 命令行后，按次序输入以下命令：

```
$ touch index.html myCss.css
```

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

```
$ git add 1.3.html
```

```
$ git commit -m 项目第三版：“三段论”式的内容设计概要开发
```

成功提交代码后，gitbash 的反馈如下所示：

```
岛子哥@DaoziG MINGW64 /d/xiongjj/abc/
exp (main)
$ git commit -m"个性化交互UI初步设计--
鼠标模型"
[main 68ffa37] 个性化交互UI初步设计--
鼠标模型
1 file changed, 198 insertions(+)
create mode 100644 exp/1.3.html
```

项目代码仓库自此也开启了历史记录，我们可以输入日志命令查看，

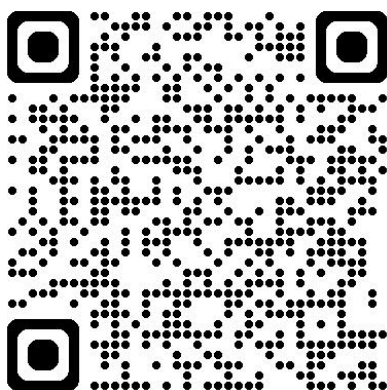
```
$ git log
```

gitbash 反馈代码的仓库日志如下所示：

```
岛子哥@DaoziG MINGW64 /d/xiongjj/abc/exp (main)
$ git log
commit 68ffa37e058e143c81a93912df266392696334af (HEAD -> main)
Author: JJ <xiong>
Date: Mon Jun 17 15:02:26 2024 +0800
```

个性化交互UI初步设计--鼠标模型

二维码如下：

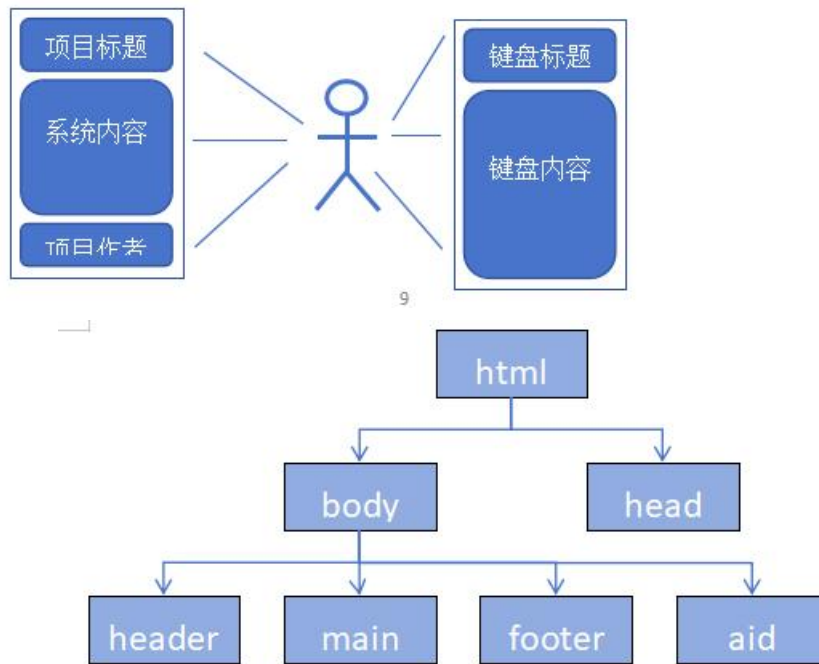


[Daozizi.github.io/1.3.html](https://Daozizi.github.io/1.3.html) at main • [Daozizi/Daozizi.github.io](https://Daozizi.github.io)

## 6.探索 UI 对鼠标拖动（模拟触屏）操作

### 6.1 分析与设计

在本次设计中，我们需要模拟触屏的拖动操作（通常也适用于鼠标拖动）通常涉及到 JavaScript 的事件监听和事件处理。首先，我们需要一个 HTML 元素，用户可以在其上进行拖动操作。例如，一个<div>元素 id 为 bookface，我们将 UI 的内容模块的部分设置为我们想要展示的图片，则上面的内容可以通过我们的鼠标拖动进行更换图片的功能，然后，使用 JavaScript 为这名为 bookface 的内容 div 添加事件监听器。对于拖动操作，通常会监听 mousedown、mousemove 和 mouseup（或 touchstart、touchmove 和 touchend 用于触屏设备）事件。在 mousedown/touchstart 事件中，记录拖动操作的开始状态，例如鼠标/触摸点的位置。同时，设置一个标志来指示拖动操作已经开始。在 mouseup/touchend 事件中，你需要重置拖动操作的开始状态标志，并可能执行一些清理操作。



## 6.2 编程与实现

### 6.2.1html 代码编写如下：

```
<body>
  <header>
    <p id="book">
      《自然野生动物鉴赏录》
    </p>
  </header>
  <main id="main">
    <div id="bookface">
    </div>
  </main>
  <footer>
    Copyright from 熊金菁 江西科技师范大学 2022--2025
  </footer>
  <div id="aid">
    <p>用户键盘响应区</p>
    <p id="keyboard"></p>
  </div>
```

## 6.2.2css 代码编写如下:

```
* {  
    margin: 10px;  
    text-align: center;  
}  
header {  
    border: 0.2em solid rgb(180, 188, 180);  
    height: 13%;  
    background-color: rgb(121, 251, 136);  
    color: black;  
}  
main {  
    border: 0.2em solid rgb(180, 188, 180);  
    height: 82%;  
    background-repeat: no-repeat;  
    background-position: center;  
    background-size: cover;  
}  
#box {  
    position: absolute;  
    right: 0;  
    width: 100px;  
}  
footer {  
    border: 3px solid rgb(180, 188, 180);  
    height: 10%;  
    font-size: 0.7em;  
}  
body {  
    position: relative;  
}  
button {  
    font-size: 1em;  
}  
#aid {  
    position: absolute;  
    border: 0.2em solid rgb(180, 188, 180);  
    top: 0px;  
    left: 600px;  
}  
#bookface {  
    position: absolute;  
    width: 80%;
```

```

    height: 80%;
    background-color: blanchedalmond;
    background: url(../images/animal2.jpg);
    background-size: cover;
}

```

### 6.2.3 javascript 代码编写如下:

```

<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.appWidth - baseFont + "px";
    document.body.style.height = UI.appHeight - baseFont * 4 + "px";
    if (window.innerWidth < 1000) {
        $(".aid").style.display = 'none';
    }
    $(".aid").style.width = window.innerWidth - UI.appWidth - baseFont * 3 + 'px';
    $(".aid").style.height = UI.appHeight - baseFont * 3 + 'px';
    //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.pageX - mouse.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;
        }
    }
}

```



```

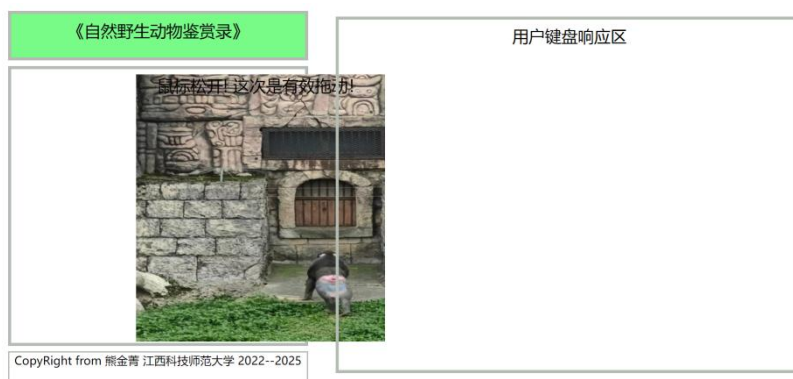
    }
  } //end of $
</script>
</body>

</html>

```

## 6.3 测试与运行

本次测试主要是检测 javascript 代码中逻辑的正确性，观察是否能够完成记录鼠标移动坐标，接受键盘输入字符，并在屏幕上展示出该字符及其编码功能，同时还需测试该程序 UI 是否能够响应式的适应多种设备，观察其在不同设备中显示效果。Pc 端采用的运行方式是使用浏览器直接打开 html 文件的方式，移动端则采用使用浏览器的模拟功能进行移动端的模拟运行。Pc 端效果图如图 6.6 所示，移动端则选择使用不同设备展示该项目优越的响应式设计效果，效果图如图 6.7 所示。



6.6



图 6.7

## 6.4 代码提交和版本控制

本项目的文件通过 gitBash 工具管理，作为项目的第二次迭代，在代码提交和版本管理环节，我们的目标将第二次迭代项目信息上传，并且将项目的备注注释上。

进入 gitBash 命令行后，按次序输入以下命令：

```
$ touch index.html myCss.css
```

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

```
$ git add 1.4.html
```

```
$ git commit -m 项目第四版：探索 UI 对鼠标拖动（模拟触屏）操作
```

成功提交代码后，gitbash 的反馈如下所示：

```
岛子哥@DaoziG MINGW64 /d/xiongjj/abc/exp (main)
$ git commit -m"探索UI对鼠标拖动（模拟触屏）操作"
[main 5626ad2] 探索UI对鼠标拖动（模拟触屏）操作
1 file changed, 186 insertions(+)
create mode 100644 exp/1.4.html
```

项目代码仓库自此也开启了历史记录，我们可以输入日志命令查看，

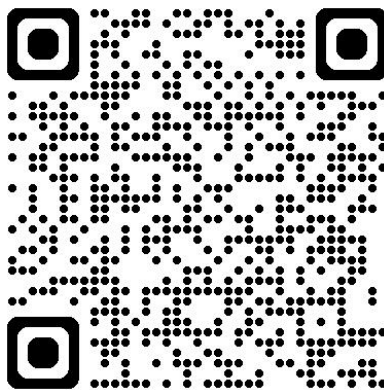
```
$ git log
```

gitbash 反馈代码的仓库日志如下所示：

```
commit 5626ad2fe4067da35912626e9f85b8a81faba79e (HEAD -> main)
Author: JJ <xiong>
Date: Mon Jun 17 15:07:45 2024 +0800

    探索UI对鼠标拖动（模拟触屏）操作
```

二维码如下：

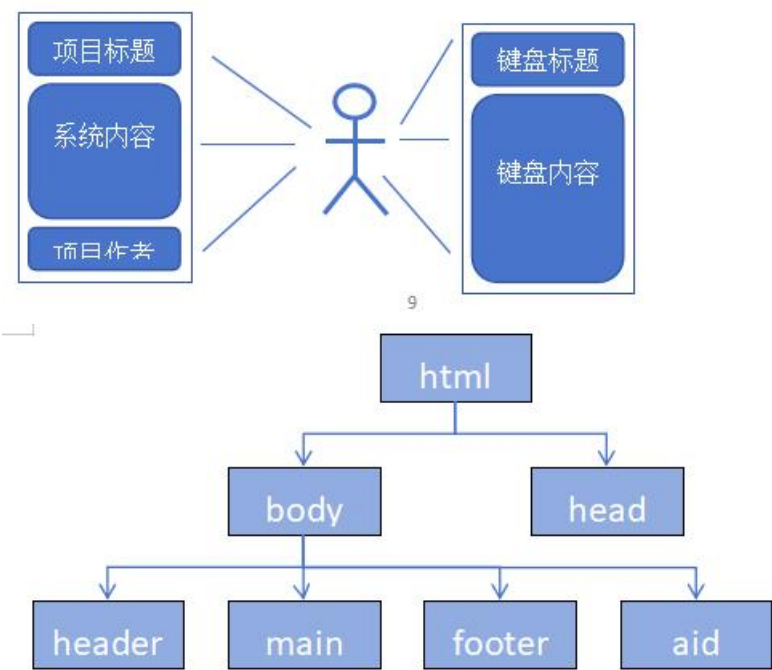


[Daozizi.github.io/1.4.html](https://Daozizi.github.io/1.4.html) at main • [Daozizi/Daozizi.github.io](https://Daozizi.github.io)

# 7.个性化 UI 对键盘的控制

## 7.1 分析与设计

个性化 UI 对键盘的控制主要体现在用户可以根据自身需求和使用习惯，对键盘的多个方面进行定制和调整，以提升输入效率和用户体验。以下是一些关于个性化 UI 对键盘控制的主要方面：用户可以根据个人喜好，选择或创建不同的键盘主题，如深色模式、浅色模式、彩色主题等，以匹配设备的整体外观或个人审美。通过个性化 UI，用户可以更改键盘的背景图像、按键图标、字体样式和颜色等，使键盘看起来更加独特和个性化。一些先进的个性化 UI 支持键盘的触觉反馈功能，如按键振动或压力感应。这些功能可以提供更直观的输入体验，帮助用户更准确地感知按键的按下和释放。综上所述，个性化 UI 对键盘的控制体现在多个方面，包括键盘布局和尺寸的定制、键盘主题和外观的自定义、按键功能和宏设置的调整、触觉反馈和声音提示的提供以及高级设置和定制选项的添加等。这些功能使得用户可以根据自己的需求和使用习惯来定制和调整键盘输入方式，从而提高输入效率和用户体验。



## 7.2 编程与实现

### 7.2.1html 代码编写如下：

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

```

        《熊金菁的毕设题目》
    </p>
</header>
<nav>
</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 熊金菁 江西科技师范大学 2024--2025
</footer>
<div id="aid">
    <p>用户键盘响应区</p>
</div>

```

## 7.2.2css 代码编写如下:

```

<style>
    * {
        margin: 10px;
        text-align: center;
    }
    header {
        border: 3px solid rgb(196, 37, 193);
        height: 10%;
        font-size: 1em;
    }
    main {
        border: 3px solid rgb(196, 37, 193);
        height: 70%;
        font-size: 0.8em;
        position: relative;
    }
    #box {
        position: absolute;
        right: 0;
        width: 100px;
    }
    footer {
        border: 3px solid rgb(196, 37, 193);
    }

```

```

        height: 10%;
        font-size: 0.7em;
    }
    body {
        position: relative;
    }
    button {
        font-size: 1em;
    }
    #aid {
        position: absolute;
        border: 3px solid blue;
        top: 0px;
        left: 600px;
    }
    #bookface {
        position: absolute;
        width: 80%;
        height: 80%;
        border: 1px solid red;
        background-color: rgb(255, 248, 199);
        left: 7%;
        top: 7%;
    }
}
</style>

```

## 7.2.3 javascript 代码编写如下:

```

<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.appWidth - baseFont + "px";
    document.body.style.height = UI.appHeight - baseFont * 4 + "px";
    if (window.innerWidth < 1000) {

```

```

        $("aid").style.display = 'none';
    }
    $("aid").style.width = window.innerWidth - UI.appWidth - baseFont * 3 + 'px';
    $("aid").style.height = UI.appHeight - baseFont * 3 + 'px';
    //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';
        }
    }
};

$("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>
</body>
</html>

```

## 7.3 测试与运行

本次测试主要是检测 javascript 代码中逻辑的正确性，观察是否能够完成记录键盘的操作，接受键盘输入字符，并在屏幕上展示出该字符及其编码功能，同时还需测试该程序 UI 是否能够响应式的适应多种设备，观察其不同设备中显示效果。Pc 端采用的运行方式是使用浏览器直接打开 html 文件的方式，移动端则采用使用浏览器的模拟功能进行移动端的模拟运行。Pc 端效果图如图 6.6 所示，移动端则选择使用不同设备展示该项目优越的响应式设计效果，效果图如图 6.7 所示。



6.6



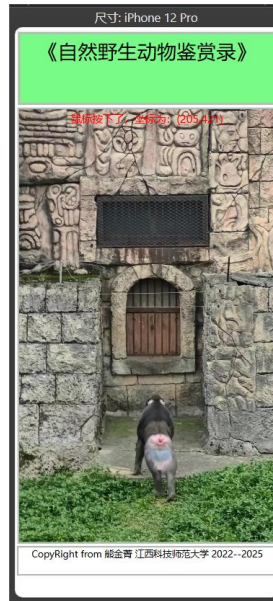


图 6.7

## 7.4 代码提交和版本控制

本项目的文件通过 gitBash 工具管理，作为项目的第二次迭代，在代码提交和版本管理环节，我们的目标将第二次迭代项目信息上传，并且将项目的备注注释上。进入 gitBash 命令行后，按次序输入以下命令：

```
$ touch index.html myCss.css
```

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

```
$ git add 1.5.html
```

```
$ git commit -m 项目第五版：个性化 UI 对键盘的控制
```

成功提交代码后，gitbash 的反馈如下所示：

```
岛子哥@DaoziG MINGW64 /d/xiongjj/abc/exp (main)
$ git commit -m"探索UI对鼠标拖动（模拟触屏）操作"
[main 5626ad2] 探索UI对鼠标拖动（模拟触屏）操作
1 file changed, 186 insertions(+)
create mode 100644 exp/1.4.html
```

项目代码仓库自此也开启了历史记录，我们可以输入日志命令查看，

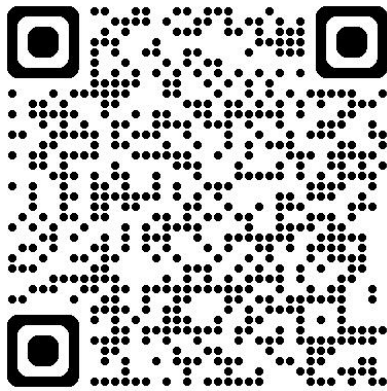
```
$ git log
```

gitbash 反馈代码的仓库日志如下所示：

```
commit 5626ad2fe4067da35912626e9f85b8a81faba79e (HEAD -> main)
Author: JJ <xiong>
Date: Mon Jun 17 15:07:45 2024 +0800

    探索UI对鼠标拖动（模拟触屏）操作
```

二维码如下：

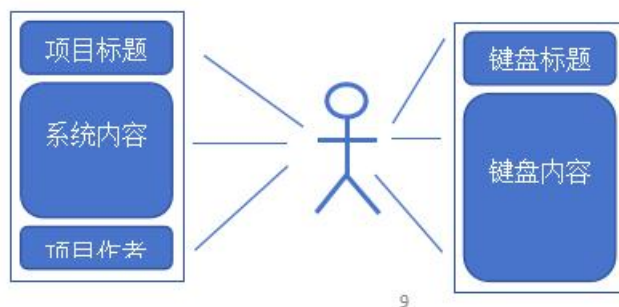


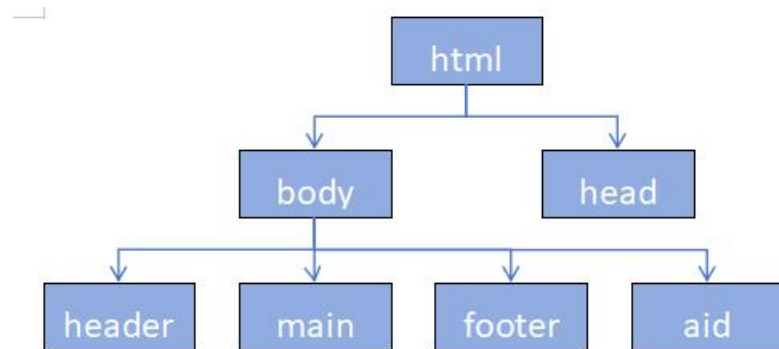
[Daozizi.github.io/1.5.html](https://Daozizi.github.io/1.5.html) at main • [Daozizi/Daozizi.github.io](https://Daozizi.github.io)

## 8.UI 的个性化键盘交互控制的设计开发

### 8.1 分析与设计

个性化 UI 对键盘的控制主要体现在用户可以根据自身需求和使用习惯，对键盘的多个方面进行定制和调整，以提升输入效率和用户体验。以下是一些关于个性化 UI 对键盘控制的主要方面：用户可以根据个人喜好，选择或创建不同的键盘主题，如深色模式、浅色模式、彩色主题等，以匹配设备的整体外观或个人审美。通过个性化 UI，用户可以更改键盘的背景图像、按键图标、字体样式和颜色等，使键盘看起来更加独特和个性化。一些先进的个性化 UI 支持键盘的触觉反馈功能，如按键振动或压力感应。这些功能可以提供更直观的输入体验，帮助用户更准确地感知按键的按下和释放。综上所述，个性化 UI 对键盘的控制体现在多个方面，包括键盘布局和尺寸的定制、键盘主题和外观的自定义、按键功能和宏设置的调整、触觉反馈和声音提示的提供以及高级设置和定制选项的添加等。这些功能使得用户可以根据自己的需求和使用习惯来定制和调整键盘输入方式，从而提高输入效率和用户体验。





## 8.2 编程与实现

### 8.2.1html 代码编写如下：

```
<body>
  <header>
    <p id="book">
      《熊金菁的毕设题目》
    </p>
  </header>
  <nav>
  </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 熊金菁 江西科技师范大学 2024--2025
  </footer>
  <div id="aid">
    <p>用户键盘响应区</p>
  </div>
```

### 8.2.2css 代码编写如下：

```
<style>
  * {
    margin: 10px;
```

```

        text-align: center;
    }
    header {
        border: 3px solid rgb(196, 37, 193);
        height: 10%;
        font-size: 1em;
    }
    main {
        border: 3px solid rgb(196, 37, 193);
        height: 70%;
        font-size: 0.8em;
        position: relative;
    }
    #box {
        position: absolute;
        right: 0;
        width: 100px;
    }
    footer {
        border: 3px solid rgb(196, 37, 193);
        height: 10%;
        font-size: 0.7em;
    }
    body {
        position: relative;
    }
    button {
        font-size: 1em;
    }
    #aid {
        position: absolute;
        border: 3px solid blue;
        top: 0px;
        left: 600px;
    }
    #bookface {
        position: absolute;
        width: 80%;
        height: 80%;
        border: 1px solid red;
        background-color: rgb(255, 248, 199);
        left: 7%;
        top: 7%;
    }
}

```

```
</style>
```

### 8.2.3 javascript 代码编写如下:

```
<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.appWidth - baseFont + "px";
    document.body.style.height = UI.appHeight - baseFont * 4 + "px";
    if (window.innerWidth < 1000) {
        $("aid").style.display = 'none';
    }
    $("aid").style.width = window.innerWidth - UI.appWidth - baseFont * 3 + 'px';
    $("aid").style.height = UI.appHeight - baseFont * 3 + 'px';
    //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';
        }
    }
};

```

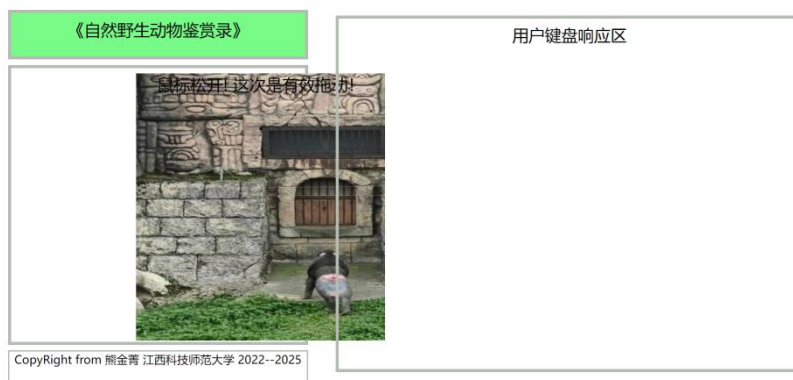
```

    }
  };
  $("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>
</body>
</html>

```

## 8.3 测试与运行

本次测试主要是检测 javascript 代码中逻辑的正确性，观察是否能够完成记录键盘的操作，接受键盘输入字符，并在屏幕上展示出该字符及其编码功能，同时还需测试该程序 UI 是否能够响应式的适应多种设备，观察其在不同设备中显示效果。Pc 端采用的运行方式是使用浏览器直接打开 html 文件的方式，移动端则采用使用浏览器的模拟功能进行移动端的模拟运行。Pc 端效果图如图 6.6 所示，移动端则选择使用不同设备展示该项目优越的响应式设计效果，效果图如图 6.7 所示。



6. 6

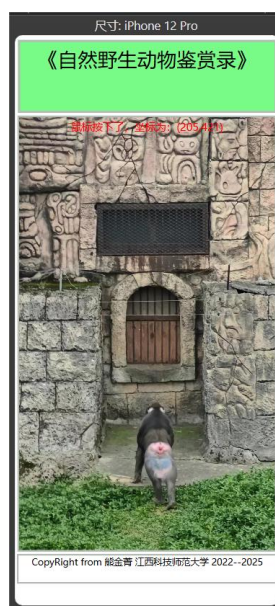


图 6. 7

## 8.4 代码提交和版本控制

本项目的文件通过 gitBash 工具管理，作为项目的第六次迭代，在代码提交和版本管理环节，我们的目标将第二次迭代项目信息上传，并且将项目的备注注释上。进入 gitBash 命令行后，按次序输入以下命令：

```
$ touch index.html myCss.css
```

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

```
$ git add 1.6.html
```

```
$ git commit -m 项目第六版: UI 的个性化键盘交互控制的设计开发  
成功提交代码后，gitbash 的反馈如下所示：
```



```
岛子哥@DaoziG MINGW64 /d/xiongjj/abc/exp (main)
$ git commit -m"UI的个性化键盘交互控制的设计开发"
[main 657e189] UI的个性化键盘交互控制的设计开发
1 file changed, 221 insertions(+)
create mode 100644 exp/1.6.html
```

项目代码仓库自此也开启了历史记录，我们可以输入日志命令查看，

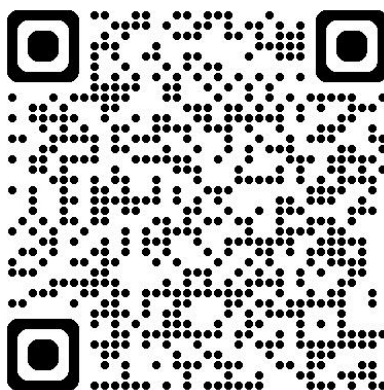
```
$ git log
```

gitbash 反馈代码的仓库日志如下所示：

```
岛子哥@DaoziG MINGW64 /d/xiongjj/abc/exp (main)
$ git log
commit 657e1897403982c5047c9ebcd049c81a980025cd (HEAD -> main)
Author: JJ <xiong>
Date: Mon Jun 17 19:07:19 2024 +0800

    UI的个性化键盘交互控制的设计开发
```

二维码如下：



[Daozizi.github.io/1.6.html](https://Daozizi.github.io/1.6.html) at main • [Daozizi/Daozizi.github.io](https://Daozizi/Daozizi.github.io)

## 9. 用 gitBash 工具管理本项目的 http 服务器

### 9.1 经典 gitbash 工具介绍

经典 Git Bash 工具是一款在 Windows 环境下为 Git 用户提供的类似 Unix 终端的命令行工具。以下是关于 Git Bash 工具的详细介绍：Git Bash 为用户提供了一个完整的、强大的 Linux 命令行接口（CLI），使得 Windows 用户能够在本地环境中轻松执行各种 Git 操作和其他类 Unix 命令。这些操作包括但不限于配置 Git 仓库、管理文件和文件夹、运行应用程序等。分布式版本控制：Git Bash 支

持 Git 的分布式版本控制功能，允许多个开发者在同一个项目上协同工作，并记录每个开发者对项目文件的修改、添加和删除等操作。分支管理：Git Bash 鼓励开发者使用分支进行并行开发和尝试新功能，然后再将分支合并到主分支中。这种分支管理策略有助于提高开发效率和代码质量。

快速和高效：Git Bash 的设计目标之一是快速执行操作，包括提交、分支切换、合并等，即使在大型项目中也能保持高效。强大的历史记录：Git Bash 记录了每个提交的详细信息，包括作者、时间戳和具体的修改内容，便于查看项目的演变历史。支持多种协作方式：Git Bash 支持通过远程仓库进行协作开发，开发者可以将自己的修改推送到远程仓库，并从其他开发者那里获取最新的修改。

Git Bash 支持常用的 Linux 命令，如 `ls`、`cd`、`mkdir`、`rm` 等，用于浏览和管理文件系统。同时，它还提供了一系列 Git 命令，如 `git init`（初始化 Git 仓库）、`git add`（添加文件到暂存区）、`git commit`（提交暂存区的文件到本地仓库）、`git push`（将本地仓库的修改推送到远程仓库）等。使用 Git Bash 的方法非常简单，用户只需安装 Git Bash 后，在 Windows 文件系统中右键单击要操作的文件夹，选择“Git Bash Here”即可打开命令行窗口。然后，用户可以在该窗口中输入 Linux 命令或 Git 命令来执行各种操作。Git Bash 是一款功能强大的命令行工具，它为用户提供了一个完整的、强大的 Linux CLI 环境，使得 Windows 用户能够在本地环境中轻松执行各种 Git 操作和其他类 Unix 命令。通过 Git Bash，用户可以更加高效地进行版本控制、代码管理和协作开发。

这是一本关于指导计算机的书。如今，电脑和螺丝刀一样常见，但它们相当复杂，让它们做你想让它们做的事情并不总是那么容易。如果你的计算机任务是一个常见的，很容易理解的任务，比如显示你的电子邮件或像一个计算器，你可以打开适当的应用程序，并开始工作。但是对于唯一的或开放式的任务，可能没有应用程序。这就是编程可能会出现的地方。编程是构建一个程序的行为——一组精确的指令，告诉计算机要做什么。因为计算机是愚蠢的，迂腐的野兽，编程从根本上来说是乏味和令人沮丧的。幸运的是，如果你能克服这个事实，甚至可以享受到愚蠢的机器能够处理的严格思考，那么编程可能是值得的。它允许你在几秒钟内完成一些永远需要手工完成的事情。这是一种让你的电脑工具做一些它以前不能做的事情的方法。它提供了一个很好的抽象思维的练习。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。


## 10.2 创建一个空的远程代码仓库

## Create a new repository

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

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

Owner \*

 masterLijh

 /

Repository name \*

✓ userName.github.io is available.

Great repository names are short and memorable. Need inspiration? How about [expert-rotary-phone](#) ?

Create repository

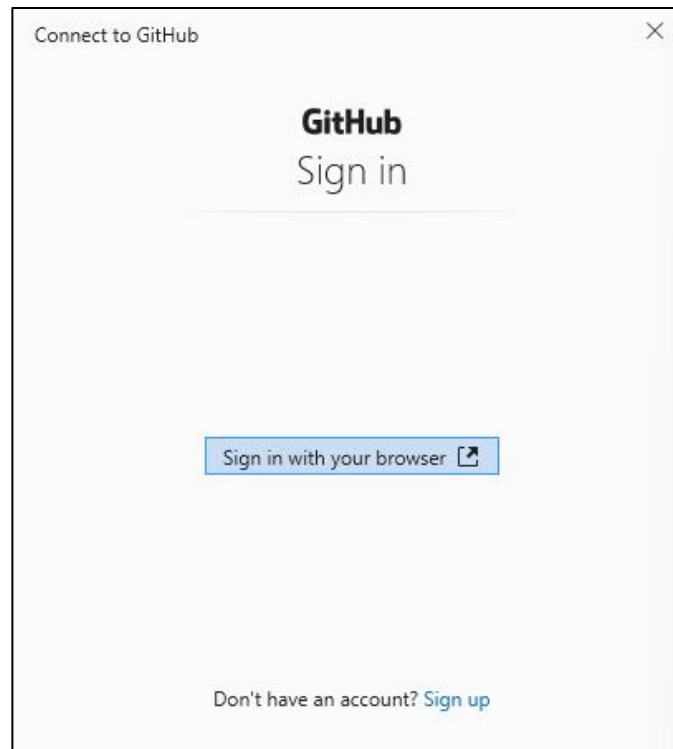
点击窗口右下角的绿色“Create repository”，则可创建一个空的远程代码仓库。

## 10.4 设置本地仓库和远程代码仓库的链接

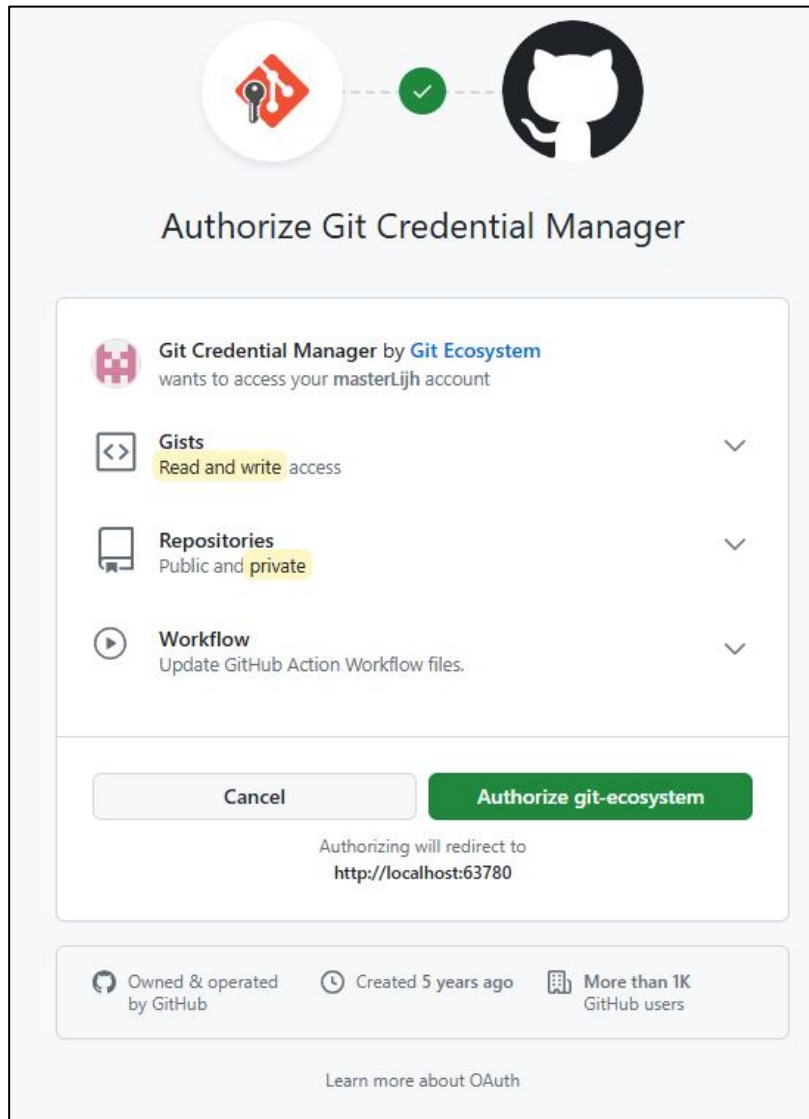
进入本地 webUI 项目的文件夹后，通过下面的命令把本地代码仓库与远程建立密钥链接

```
$ 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://github.com/daozizi.github.io
$ git push -u origin main
```

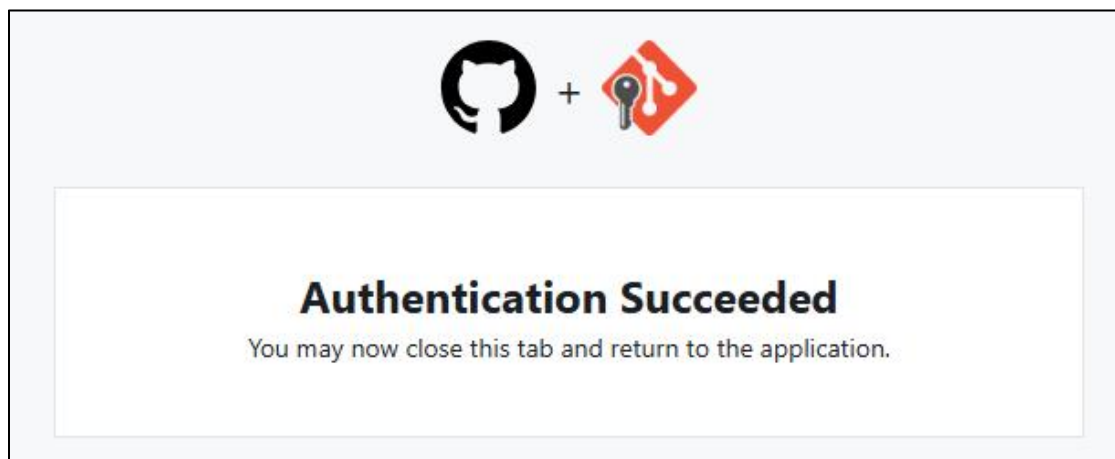
本项目使用 window 平台，gitbash 通过默认浏览器实现密钥生成和记录，第一次链接会要求开发者授权，如下图所示：



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：

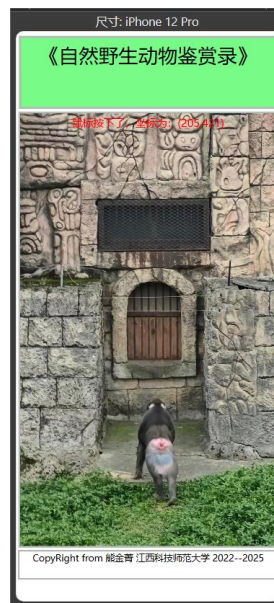


最后，GitHub 平台反馈：gitBash 和 gitHub 平台成功实现远程链接。



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：



## 参考文献：

- [0] W3C. W3C's history. W3C Community. [EB/OL]. <https://www.w3.org/about/>. 2023.12.20
- [1] Douglas E. Comer. The Internet Book [M] (Fifth Edition). CRC Press Taylor & Francis Group, 2019: 217-218
- [2] John Dean,PhD. Web programming with HTML5,CSS,and JavaScript[M]. Jones & Bartlett Learning,LLC. 2019: 2
- [3] John Dean,PhD. Web programming with HTML5,CSS,and JavaScript[M]. Jones & Bartlett Learning,LLC. 2019: xi
- [4] Behrouz Forouzan. Foundations of Computer Science[M](4th Edition). Cengage Learning EMEA,2018: 274--275
- [5] Marijn Haverbeke. Eloquent JavaScript 3rd edition. No Starch Press,Inc, 2019.

[6] William Shotts. The Linux Command Line, 2nd Edition [ M ]. No Starch Press, Inc, 245 8th Street, San Francisco, CA 94103, 2019 : 3-7

[7] Martina Seidl, Marion Scholz, et al. UML @ Classroom An Introduction to Object-Oriented Modeling [M]. Springer International Publishing Switzerland, 2015

[8] Matti Tedre, Peter J.Denning. Computational Thinking, A Professional and Historical Perspective. Computational Thinking in Education A Pedagogical Perspective[C]. Routledge Taylor & Francis Group, 2022:1-17