同济大学

TONGJI UNIVERSITY

《WEB 技术》

实验报告（大作业）

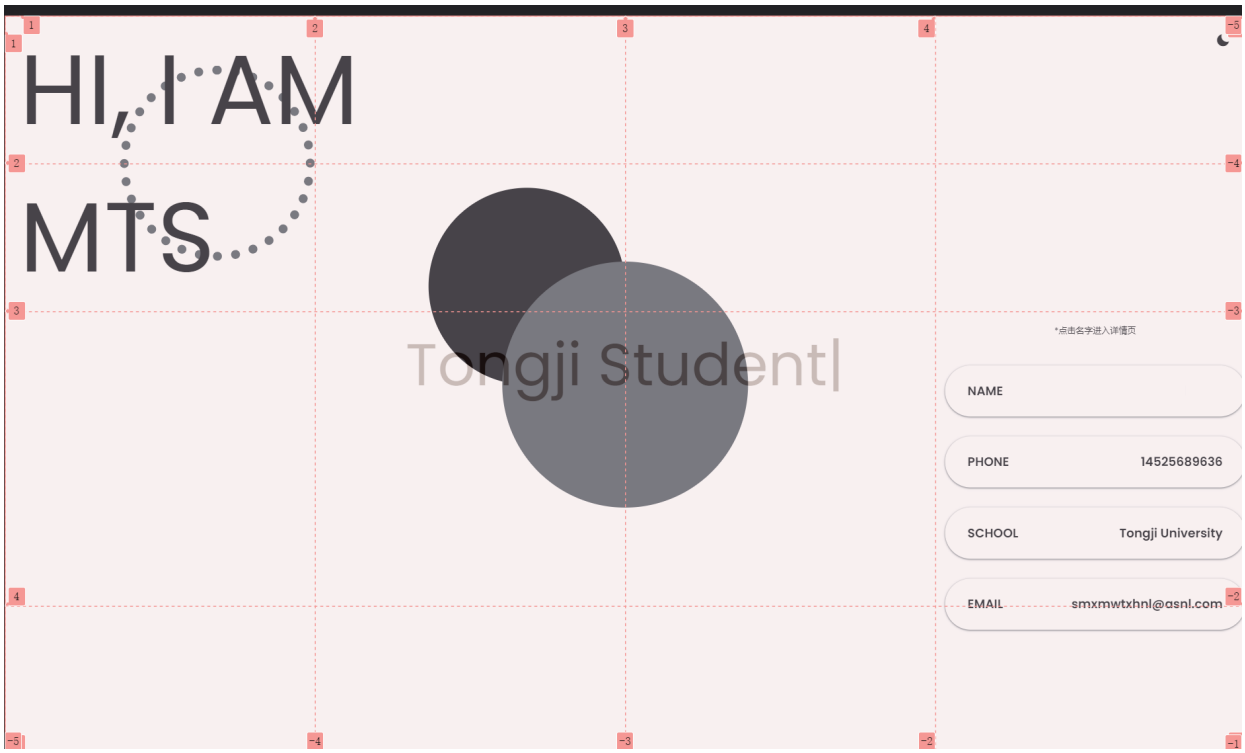
实验名称	静态网页设计
小组成员	
学院（系）	电子与信息工程学院
专　　业	计算机科学与技术专业
任课教师	郭玉臣
日　　期	2023 年　3 月 28 日

Content

整体构思

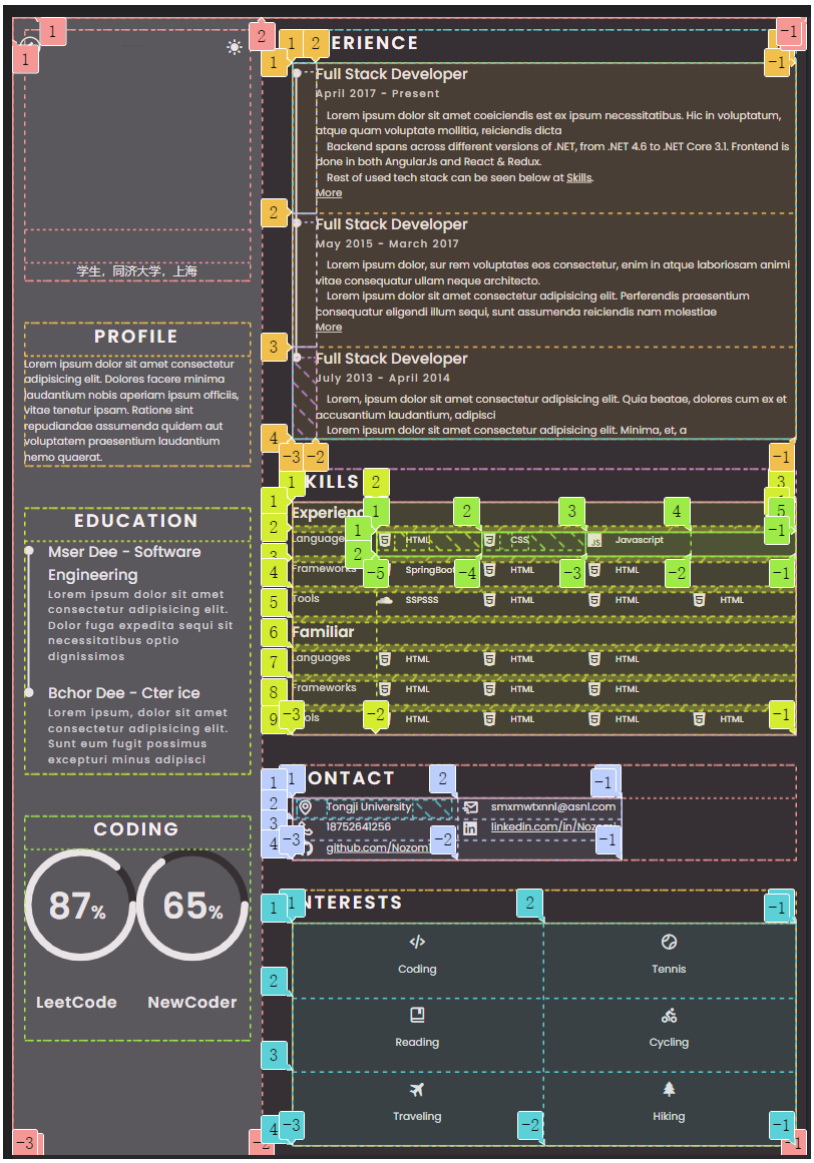
布局排版

- 首页排版：



首页整体采用 `grid` 布局，主要分为三部分内容，分别为左上角主题字，中间变化主题字，右下角基本信息

- 主页排版



主体采用 `grid` 排版，其中关于每一分区部分同样采用 `grid` 排版，对于每一个小项采用 `flex` 排版

保证项目之间的间距在盒子内得到控制，尽量减少`margin` 属性的使用

css全局基本设置

全局基础标签属性设置

- 盒子模型

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

> 在两个不同区块之间设置 `margin` 时可能会发生 `css` 边距重叠缩小，相邻的两个元素的边距会合并成一个单独的边距，而不是像我们期望的那样直接相加。这种现象主要发生在上下相邻的两个元素之间的垂直边距（即 margin-top 和 margin-bottom）

> 在实际书写文本时，可能会出现一些标签内部默认 `padding`，对于控制边距十分不便

在初始时重新设置边距可以清除所有盒子模型的 `margin` 和 `padding` 边距以及一些默认边距，后续可以从无边距开始设计。

同时，设置 `box-sizing` 使得整个盒子模型更加可控。盒模型包含内容区域、内边距、边框和外边距等部分。默认的盒模型是"content-box"，即元素的宽度和高度只包含内容区域，不包括边框和内边距。这使得在布局时需要考虑到元素的宽度和高度、边框和内边距的计算，并且会导致在设置元素宽度和高度时需要频繁计算。而将全局选择器设置为 `box-sizing: border-box`，则使元素的宽度和高度包括边框和内边距，这样在布局时更加直观和方便，并且可以减少计算量。这样，在设计响应式布局时，也可以更加方便地处理元素宽度和高度的变化。

- 基本标签设置

```
l {
  list-style: none;
  padding: 0;
}

a {
  text-decoration: none;
  color: var(--text-color);
}

html {
  scroll-behavior: smooth;
}

body {
  margin: 0;
}

a,
i,
div {
  -webkit-tap-highlight-color: transparent;
  -webkit-touch-callout: none;
}
```

设置有序以及无序列表的前端标识为空，可以更好控制列表样式

设置链接无 `text-decoration` 可以更好控制连接样式

`html` 标签设置滚动 `smooth` 使得页面内跳转更加平滑

- 全局变量设置

在`css`中可以在`:root`设置全局变量来更好管理页面的样式，其中包括了元素的字体样式、间距系统、颜色系统、阴影大小、盒子圆角大小、视觉层次等

字体设置

```
  --body-font: "Poppins", sans-serif;
```

颜色:

简约庄重一般作为简历的风格，所以颜色要偏于同一色调，通过更改颜色的敏感来体现处颜色的层次，主要分为主色调、灰色以及强调色

此处选择黑色`#212529ff`作为黑色基准色，由于此颜色本身包含了灰色，则辅助色灰色可以利用主色调调浅来实现，由于并不需要对比强调色所以此处对强调色略去

将选择好的色调与相应的文本颜色、标题颜色、主体颜色进行相应匹配

```
:root {
    ...
      --white-1: #f8f9faff;
      --white-2: #e9ecefff;
      --white-3: #dee2e6ff;
      --grey-1: #ced4daff;
      --grey-2: #adb5bdff;
      --grey-3: #6c757dff;
      --black-1: #495057ff;
      --black-2: #343a40ff;
      --black-3: #212529ff;

    --title-color: var(--black-3);
  --text-color: var(--black-2);
  --text-color-alt: var(--grey-3);
  --body-color: var(--white-1);
  --body-color-alt: var(--grey-1);

  --title-color-inv: var(--white-1);
  --text-color-inv: var(--white-2);
  --body-color-inv: var(--black-3);
    ...
}
```

字体大小、粗细设置

```
--h1-font-size: 1.5rem;
--h2-font-size: 1.25rem;
--h3-font-size: 1rem;
--normal-font-size: 0.875rem;
--small-font-size: 0.688rem;

--normal-font-weight: 400;
--medium-font-weight: 500;
--bold-font-weight: 600;
```

对主页面的小节标题样式、小节内容进行规范，其中包括间距、填充、字体大小、粗细、间距等

```
section {
  display: flex;
  flex-direction: column;
  align-items: center;
  padding: var(--m-2);
  margin-bottom: var(--m-3);
}

.section-title {
  font-size: var(--h1-font-size);
  font-weight: var(--bold-font-weight);
  color: var(--title-color);
  padding-bottom: var(--m-1);
  letter-spacing: 3px;
  text-transform: uppercase;
}

.section-content {
  display: grid;
  /* flex-direction: column; */
  grid-template-columns: repeat(4, 1fr);
}
```

边距设置

```
--m-1: 0.5rem;
--m-2: 1rem;
--m-3: 1.5rem;

--header-height: 4rem;
```

间距的设置保证了在进行设置的过程中不会出现空白比例失调的情况，保证整个页面的同一性，对于调节页面来说及其重要

全局变量可以让页面颜色调用更加清晰，同时也使得在`body`下更新变化全局颜色成为可能

- 夜间模式切换

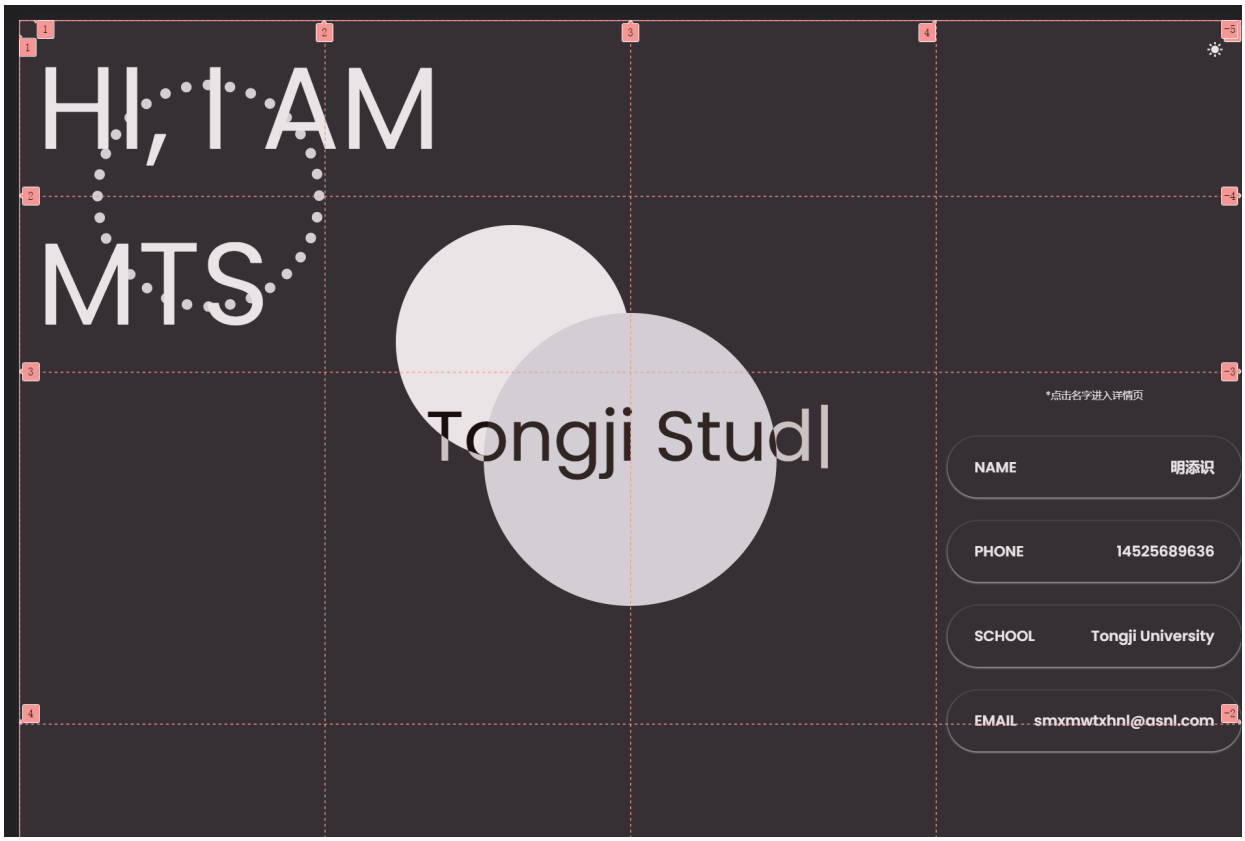
全局颜色变量的设置让更改主体颜色更加简单易行，我们可以通过`js`在对`body`添加`dark`属性，在`body.dark`具有共同标签的选择下修改全局变量的值，实现文字、图像的颜色变换

```
body.dark {
  --title-color: var(--white-1);
  --text-color: var(--white-2);
  --text-color-alt: var(--grey-1);
  --body-color: var(--black-3);
  --body-color-alt: var(--black-1);

  --title-color-inv: var(--black-3);
  --text-color-inv: var(--black-2);
  --body-color-inv: var(--white-1);

  --scrollbar-track: var(--black-1);
  --scrollbar-thumb: var(--grey-2);
  --scrollbar-thumb-hover: var(--white-1);
}

body.dark .home__theme--light {
  display: none;
}

body.dark .home__theme--dark {
  display: block;
}
```

首页设计

首页排版设计

首页主要分为三部分，由`grid`进行划分，左上角区域加入总标题，中间实现主视觉图像设计，右下角展示基本信息



整体盒子构架

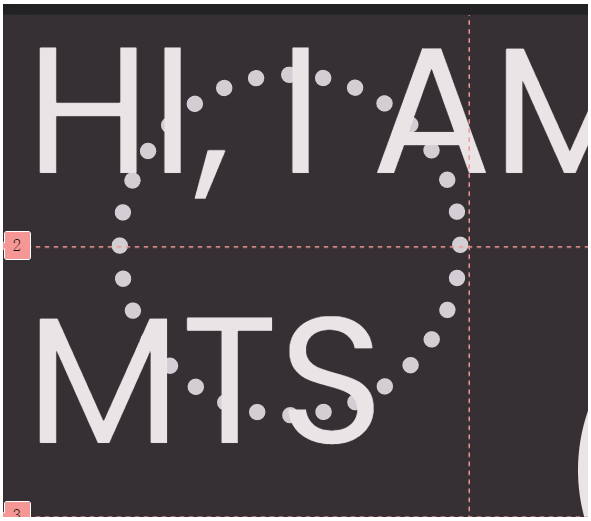
```
/*主盒子模型*/
.main-title {
  display: grid;
  grid-template-columns: repeat(4, 1fr);
  grid-template-rows: 1fr 1fr 2fr 1fr;
  justify-content: center;
  align-items: center;
  height: 100%;
}
```

其中左上角文字部分直接在`grid`中指定起止位置，使得排版更加方便，同时响应式设计更加灵活

```css
/*左上角文字*/
.resumeFont {
  display: flex;
  justify-content: flex-start;
  grid-row-start: span 2;  /*指定起止位置*/
  font-size: var(--main-cover-font-size-max);
  margin-left: var(--m-3);
}
```

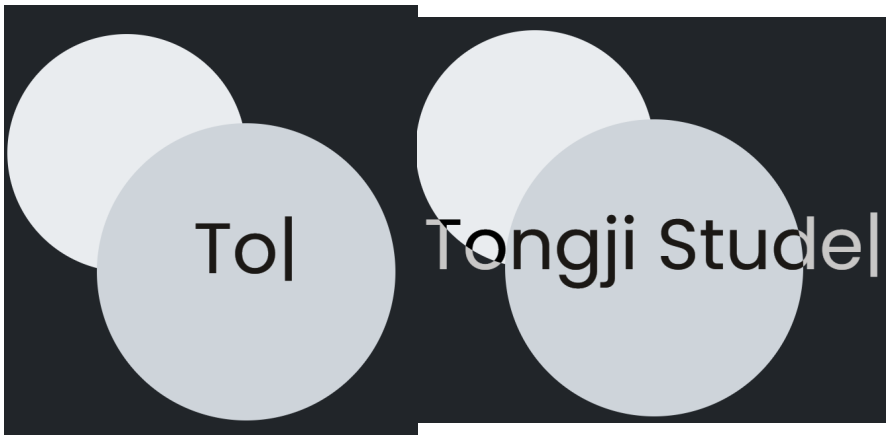
首页装饰设计

- 文字背景设计



该环形圆圈与是右下角主视图图片的一个伪元素，移动到右下角盒子的左上角位置后用`transform`的方法进行根据自身原点位置的移动，将`border`设置`dot`样式，并进行加粗，得到如图所示的效果

- 主视觉动画设计



背景部分：在中间部分设置两个绝对定位的元素，分别指定`z-index`为`-1`和`-2`保证其层叠关系

文字部分：此处使用`js`进行内容的填充和读取，通过设置一定的时间间隔，读取文字达到在屏幕上打字的效果

其中文字根据背景颜色变换通过`mix-blend-mode: difference;`实现

```
<div class="brief-intro">
    <p class="brief-intro-content" id="brief-intro-content-j"></p>
    <div class="deco deco-1"></div>
    <div class="deco deco-2"></div>
</div>
```

元素位置以及颜色等

```
.brief-intro-content {
  mix-blend-mode: difference;  /*设置文字颜色与背景颜色相反*/
  font-size: var(--main-cover-font-size-min);
}

/*伪元素设计*/
.deco {
  position: absolute;
  top: 50%;
  left: 50%;
  border-radius: 50%;
}

.deco-1 {
  height: 25rem;
  width: 25rem;
  background-color: var(--text-color-alt);
  transform: translate(-50%, -50%);
  z-index: -1;
}

.deco-2 {
  height: 20rem;
  width: 20rem;
  background-color: var(--text-color);
  transform: translate(-100%, -100%);
  z-index: -2;
}
```

文字变化

```
let th = 0;
function func() {
```

```
  let divtextList = [
    "Tongji Student",
    "Creative",
    "Regoroius",
    "Industrious",
    "OK!!!",
  ];

  let divText = divtextList[th];
  let len = divText.length;
  let divNode = document.getElementById("brief-intro-content-j");
  let i = 0;
  let f = 0;
  let coding = setInterval(function () {
    if (f == 0) {
      if (i < len) {
        divNode.innerHTML = divText.slice(0, i) + "|";
      } else {
        if (i == len) {
          divNode.innerHTML = divText.slice(0, i);
          f = 1;
        }
      }
      i++;
    } else if (f == 1) {
      if (i <= len && i >= 2) {
        divNode.innerHTML = divText.slice(0, i) + "|";
      } else {
        if (i < 2) {
          divNode.innerHTML = divText.slice(0, i);
          f = 0;
        }
      }
      i--;
    }
  }, 180);
}

func();
```

- 简要信息设置

右下角简要介绍是由无序列表组织，每一个`item`都是一个`flexbox`

结构为无序列表 `flexbox`(纵向)+内容块 `flexbox`

```
<ul class="info-list">
  <p class="tip">*点击名字进入详情页</p>
  <li class="info-item">
    <p class="item-type name">name</p>
    <a href="index.html" class="item-content name">     </a>
  </li>
  <li class="info-item">
    <p class="item-type name">phone</p>
    <a href="#" class="item-content phone">14525689636</a>
  </li>
  <li class="info-item">
    <p class="item-type name">school</p>
    <a href="#" class="item-content school">Tongji University</a>
  </li>
  <li class="info-item">
    <p class="item-type name">email</p>
    <a href="#" class="item-content email">smxmwtxhnl@asnl.com</a>
  </li>
</ul>
```

`css` 给出两层 `flexbox` 结构，并用 `space-between` 方式将容器空间充分利用

```
.info ul {                      /*纵向 flexbox*/
  display: flex;
  align-items: center;
  flex-direction: column;
  gap: 2rem;
  padding-left: var(--m-2);
```

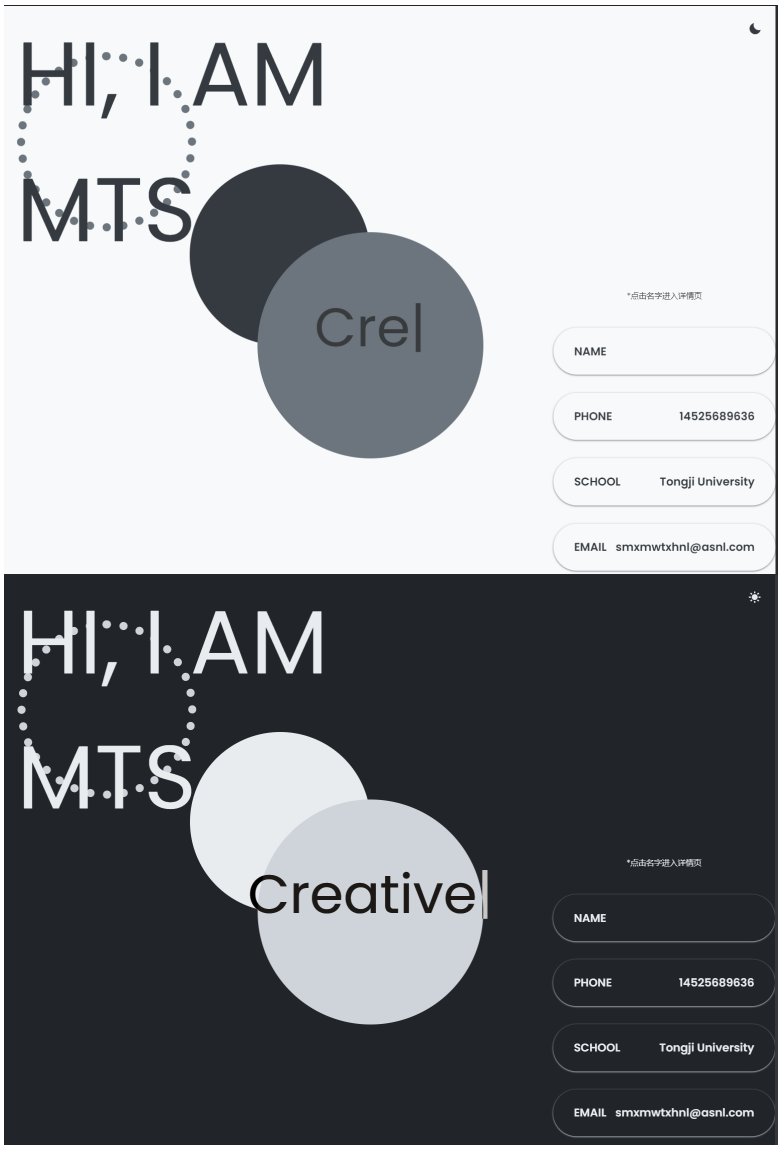
```
}

.info-item {
  display: flex;                /*横向flexbox*/
  width: 100%;
  gap: var(--m-2);
  align-items: center;
  justify-content: space-between;
  font-size: var(--h2-font-size);
  font-weight: var(--bold-font-weight);
  box-shadow: 0 1px 3px 0 var(--text-color-alt);  /*右下角阴影*/
  padding: 0.4rem 0.8rem;
  border-radius: 100px;
}

.item-content {
  margin-right: var(--m-3);
}

.item-type {
  text-transform: uppercase;   /*题头大写*/
  padding-left: var(--m-3);
}

.item-content {
  font-weight: var(--bold-font-weight);
  font-size: var(--h2-font-size);
}
```

夜间模式

点击右上角图标，由`js`捕捉点击事件，为`body`添加`dark`标签，，`css`改变全局颜色变量，实现反转颜色的效果

在`html`中设置月亮和太阳的图片，正常状态下现实月亮图标，在点击后由`js`改变其是否可见的属性，改为太阳图标

其中颜色全部变为原颜色的互补色，完成夜间模式的转换



两图标`html`，通过改变可见程度来进行切换

```
<div class="home__theme home__theme--light">
    <i class="bx bxs-moon"></i>
</div>
<div class="home__theme home__theme--dark">
    <i class="bx bxs-sun"></i>
</div>
```

相关`css`

```
body.dark .home__theme--light {
  display: none;
}

body.dark .home__theme--dark {
  display: block;
```

```
}

.home__theme {
  position: absolute;
  right: var(--m-3);
  top: var(--m-3);
  font-size: var(--h1-font-size);
  cursor: pointer;
  transition: all var(--transition-time);
}

.home__theme--dark {
  display: none;
}

.home__theme--light {
  display: block;
}
```

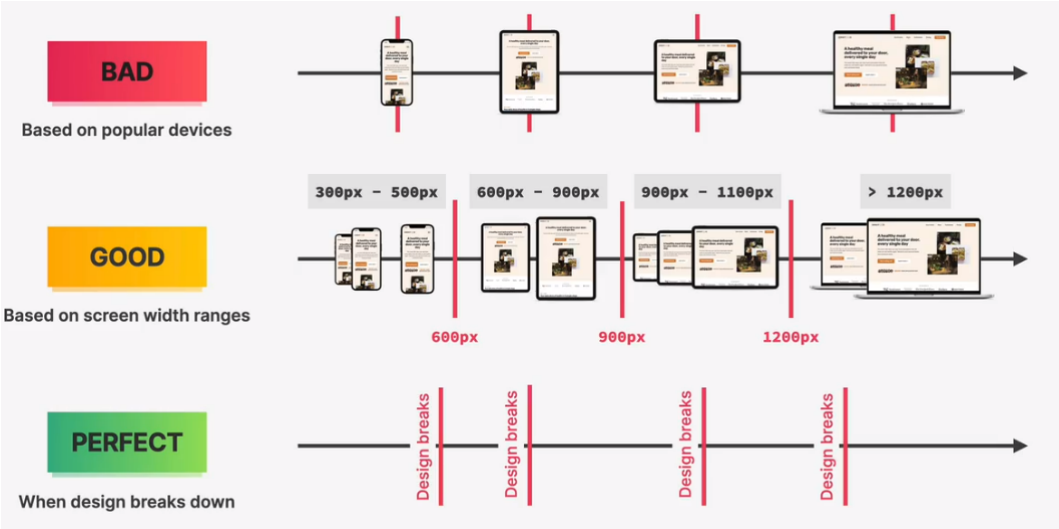
`js` 捕捉点击事件并响应

```
const cachedTheme = localStorage.getItem("__theme");

if (cachedTheme === undefined) {
  cachedTheme = "light";
  localStorage.setItem("__theme", "light");
}

if (cachedTheme === "dark") {
  document.body.classList.add("dark");
}

const theme = document.querySelectorAll(".home__theme");
theme.forEach((t) =>
  t.addEventListener("click", () => {
    if (document.body.classList.contains("dark")) {
      document.body.classList.remove("dark");
      localStorage.setItem("__theme", "light");
    } else {
      document.body.classList.add("dark");
      localStorage.setItem("__theme", "dark");
    }
  })
);
```

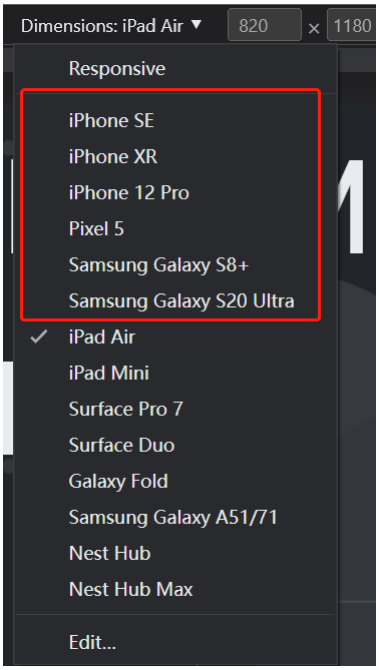
在主页中同样进行夜间模式的设计

首页响应式设计

对于不同设备屏幕该网页可以进行一定程度适应，选择位置通常位于当前设计变形位置左右，其中参考：



此处设置媒体查询的宽度(`max_-width`)的宽度分别为`1235px`，`985px`，`720px`,结果发现对于更窄屏幕的`iphone SE`，`iphone XR`等网页观看效果差别并不明显，最后停留在`720px`处

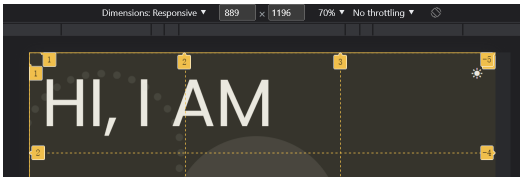


- 在`1235px`处改变左上角文字跨度范围，改变主视图背景圆圈的透明度，并将其中的动画取消

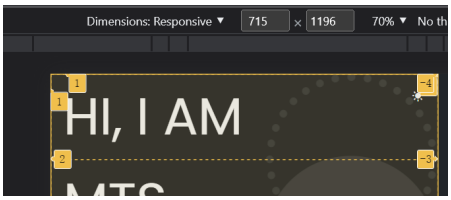
```
@media screen and (max-width: 1235px) {
  .brief-intro {
    opacity: 10%;
  }
  .brief-intro-content {
    display: none;
  }
```

```
}

@media screen and (max-width: 1235px) {
  .phrase1 {
    grid-column: 1 / -1;
    grid-row: 1 / 2;
  }
}
```

在`985px`处将列数改为三列，同样改变标题大小，以及右下角简要信息介绍的跨度



在`720px`处将整个页面变成上下布局



主页设计

内容设置

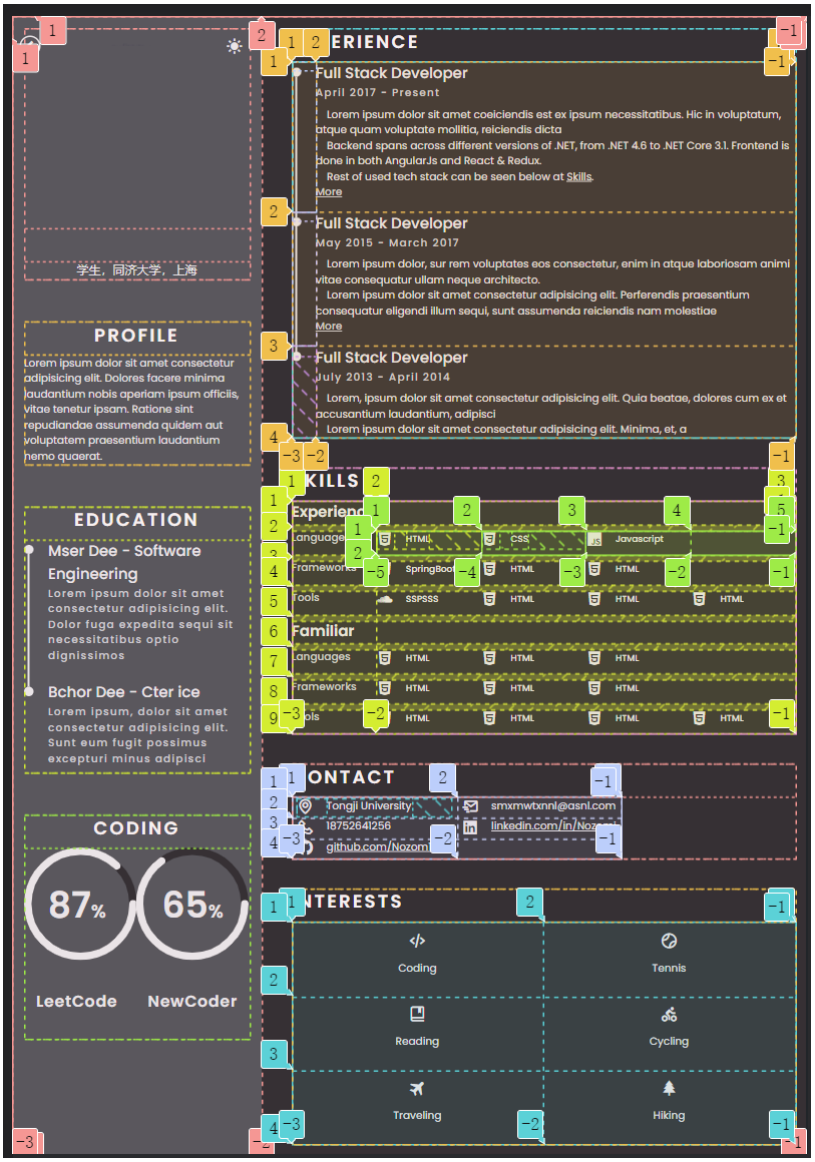
内容主要分为7个部分，左栏分为简介以及教育经历、编程经验等，右栏分为经历、技能、联系方式和兴趣类。

当屏幕宽度小于`768px`时整体布局会成为一栏

主要布局格式

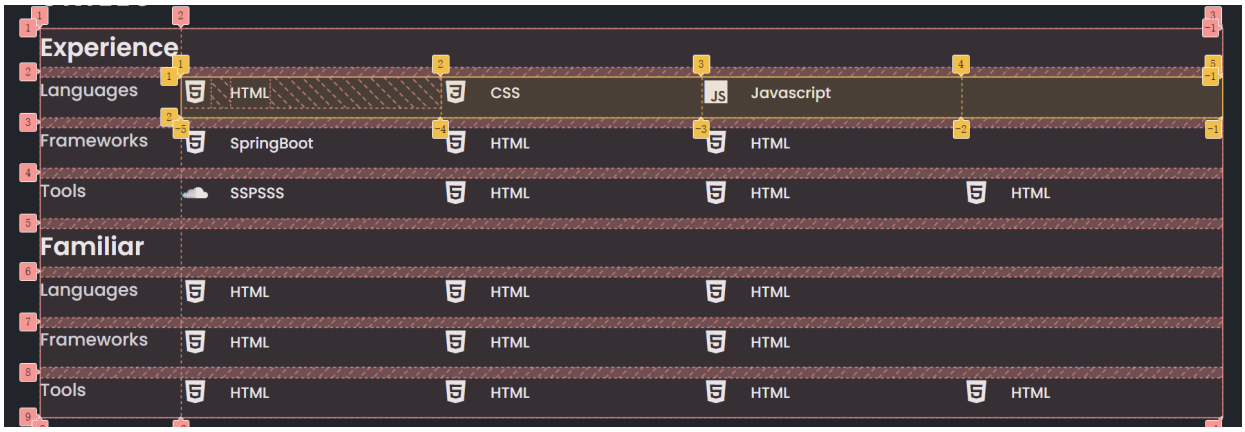
整体采用`grid`进行布局，因对于简历分块区间很明显，用`grid`布局灵活

对于水平一维布局，使用`flexbox`可以有效对各个水平元素之间的空隙进行控制，同时`flexbox`可以用来进行竖直栏的排版，控制竖直间距十分有效



> 本简历所有文字均为`vscode`自动随机生成

对于密集部分采用`grid`布局更加灵活，尤其对于`2D`部分。而对于水平一维部分应嵌套`flexbox`



流程节点装饰设计

流程节点条效果

本盒子中存在两个元素，使用`flexbox`进行竖直排版，调节间距。用`css`控制形状和展示为竖直连接线的形式。

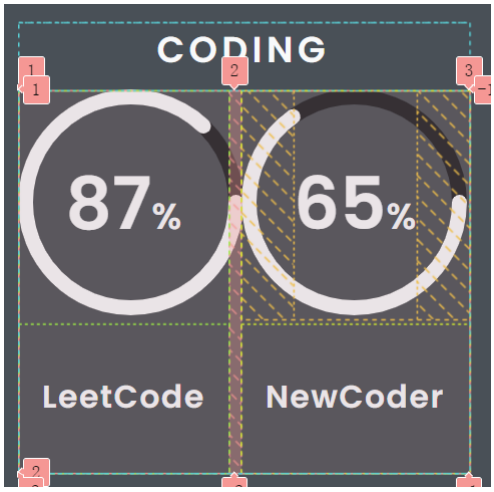
```
<div class="timeline">
  <div class="timeline__rounded"></div>
  <div class="timeline__line"></div>
</div>
```

```
.timeline {
  display: flex;
  flex-direction: column;
}

.timeline__rounded {
  position: relative;
  top: 0.5rem;
  width: 0.8rem;
  height: 0.8rem;
```

```
  background-color: var(--text-color);
  border-radius: 50%;
}

.timeline__line {
  position: relative;
  top: 0.5rem;
  left: 0.3rem;
  height: 100%;  /*竖直线充满高度*/
  border-left: 0.22em solid var(--text-color);
}
```

环形进度条

环形进度条效果




```
<div class="process-bar process-bar-1">
  <div class="percent">
    <svg>
      <circle cx="70" cy="70" r="70"></circle>
      <circle cx="70" cy="70" r="70"></circle>
    </svg>
    <div class="number">
      <h2>87<span>%</span></h2>
    </div>
  </div>
  <h2 class="text">LeetCode</h2>
</div>
```

其中两进度在`grid`中进行排版,每一小部分用`flexbox`进行排版

环形进度条的设计主要是利用`svg`进行圆形绘图，环形的`stroke-width`进行加粗, 利用画笔偏移`stroke-dashoffset`进行进度的绘制

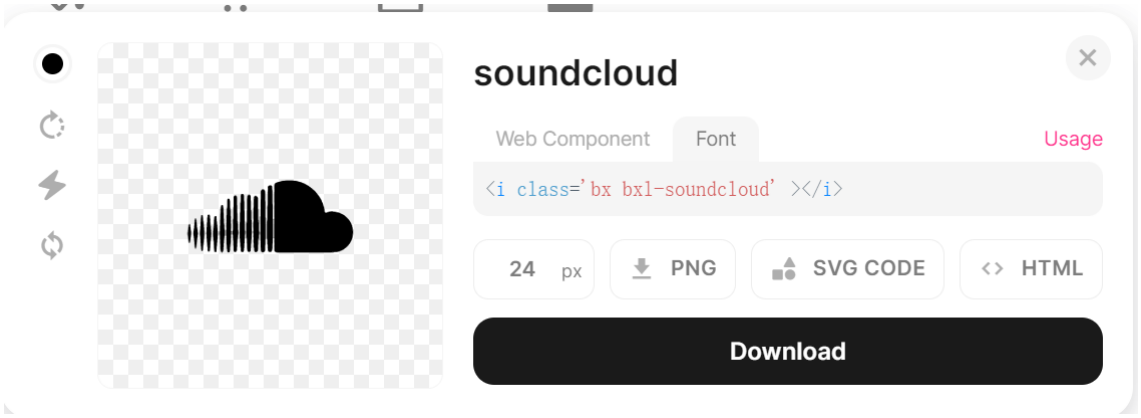
- 绘图部分

```
.process-bar .percent svg circle {
  width: 150px;
  height: 150px;
  fill: none;
  stroke-width: 10;
  stroke: var(--text-color);
  transform: translate(5px, 5px);
  stroke-dasharray: 440;
  stroke-dashoffset: 440;  /*此处的长度由 2 * pi * r*计算得出*/
  stroke-linecap: round;
}

.process-bar .percent svg circle:nth-child(1) {
  stroke-dashoffset: 0;
  stroke: var(--body-color);
}

.process-bar .percent svg circle:nth-child(2) {
  stroke: var(--text-color);
}

.process-bar-1 .percent svg circle:nth-child(2) {
  stroke-dashoffset: calc(440 - (440 * 87) / 100);  /*通过百分比计算偏移度*/
}

.process-bar-2 .percent svg circle:nth-child(2) {
  stroke-dashoffset: calc(440 - (440 * 65) / 100); /*通过百分比计算偏移度*/
}
```

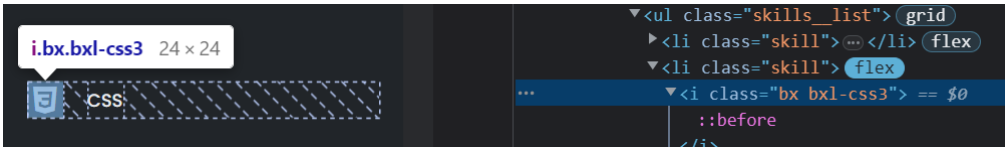
图标使用

本简历在返回封面页、夜间模式转换、技能区域、联系方式和兴趣爱好区域进行大量使用，图标可以有效引导用户进行响应操作，对于页面引导和可读性有重要作用

本简历采用在`html`中导入连接的方式使用远程图标库`boxicon`




```
<link
      href="https://unpkg.com/boxicons@2.0.9/css/boxicons.min.css"
      rel="stylesheet"
/>
```



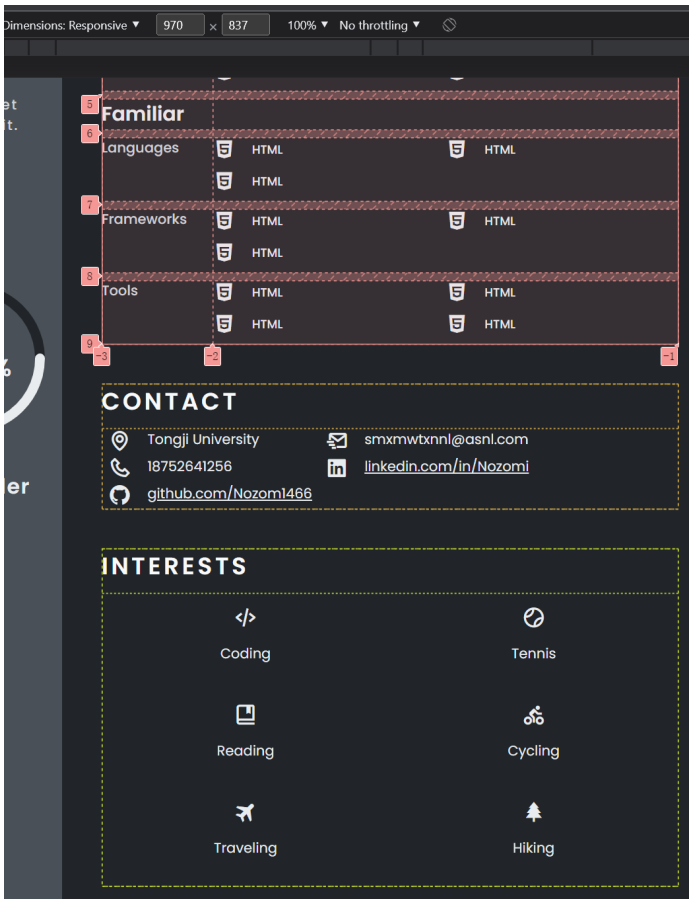
或者可以在页面尾部插入`js`远程脚本的方式对`html`文本中的图标进行注入。

值得注意的是，在调节图标的颜色时，不同远程链接具有不同方式，如果识别为文本则可以用`color`，识别为矢量图则可以使用`stroke`等

一般图标的大小可以在网站进行设置，默认为`24px`，则在实际使用时尽量按照渲染大小使用，此处调节为`24px *24px`

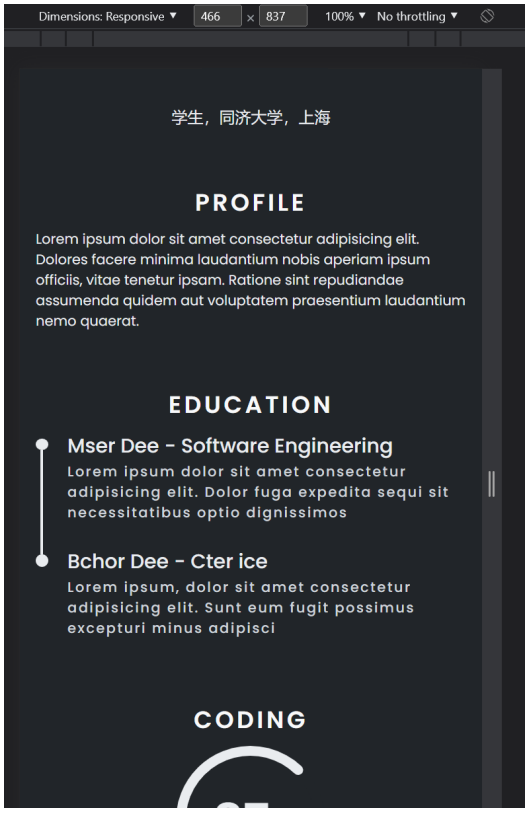
响应式设计

主页面的响应式设计与封面响应式大致相同，在`1025px`与`769px`之间主要是右侧的`grid`竖直分栏数发生变化



此处从4列变化为2列

当页面宽度小于`769px`时，页面盒子的`grid`元素分栏变为竖直一栏，同时将左栏的背景颜色切换至主背景颜色，得到颜色一致的背景



```
@media screen and (max-width: 769px) {
  .resume {
    display: grid;
    min-height: 100%;
    grid-template-columns: 1fr 3fr;
    margin-bottom: 0;
    overflow: hidden;
  }

  .resume__left {
    background-color: var(--body-color-alt);  /*切换背景颜色*/
  }

  .resume__right {
    padding-left: var(--m-3);
  }
  .resume__right section {
    align-items: flex-start;
    padding-bottom: 0;
  }
}
```

同时右侧页面顺序显示，`grid`方式进行响应式设计十分便捷，对用手机端显示页面较为美观

总结

通过设计一个个人简历的页面，我熟悉了`css`的使用，熟悉了`grid`布局与`flexbox`布局。同时进行了响应式设计，完成静态页面在不同设备屏幕下的适配。同时通过对`css`全局变量的学习我了解了在构建过程中实现页面颜色、页面间距、页面字号等的统一的重要性。同时了解了夜间模式的工作原理并进行了相应实现。同时在页面设计的过程中进行了`js`的尝试，设计了动态的页面效果。