Responsive Web Design Lab: Victoria's Journal

Overview

Transform a personal journal web page into a fully responsive experience that looks great on any device. You'll use modern CSS techniques including flexbox, CSS Grid, and media queries to create a mobile-first responsive design.

Learning Objectives

- Implement mobile-first responsive design
- Use CSS Grid and Flexbox for layout
- Apply media queries with breakpoints
- Create fluid, accessible interfaces
- Follow modern HTML/CSS best practices

Setup

Files Provided:

- `index.html` - Victoria's journal page (needs refactoring)
- `basic.css` - Basic typography styles (already linked)
- `background.gif` - Background texture image

File to Create:

- `journal_responsive.css` - Your responsive stylesheet

Exercise 1: Mobile-First HTML Structure (20 minutes)

Goal

Refactor the HTML using semantic HTML5 elements and prepare it for responsive styling.

Requirements

1. Update HTML Structure

Replace generic `<div>` elements with semantic HTML5:

- Use `<header>` for the page header
- Use `<main>` for the primary content
- Use `<article>` for each journal entry
- Use `<aside>` for the friends list (Exercise 5)
- Use `<footer>` if adding footer content
- Use `<section>` to group related content

2. Add Appropriate Classes

Use BEM (Block Element Modifier) naming convention:

html

```
<!-- Example -->
<article class="journal-entry">
  <h2 class="journal-entry__title">Entry Title</h2>
  <img class="journal-entry__image" src="..." alt="...">
  <p class="journal-entry__content">...</p>
</article>
```

3. Ensure Accessibility

- Add proper `alt` attributes to all images
- Use heading hierarchy correctly (h1 → h2 → h3)
- Add `lang` attribute to `<html>` tag
- Include viewport meta tag in `<head>`:

html

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

HTML Template Structure

html

```
<!DOCTYPE html>
<!--
  Student Name: [Your Name]
  Date: [Today's Date]
  Project: Responsive Victoria's Journal
-->
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Victoria's Journal</title>
  <link rel="stylesheet" href="basic.css">
  <link rel="stylesheet" href="journal_responsive.css">
</head>
<body>
  <!-- Your semantic HTML goes here -->
</body>
</html>
```

Exercise 2: Mobile Styles - Foundation (20 minutes)

Goal

Create the base mobile styles (screens 320px and up). This is your default styling.

CSS Structure to Create

css

```css
/*
  Student Name: [Your Name]
  Date: [Today's Date]
  File: journal_responsive.css
  Description: Responsive stylesheet for Victoria's Journal
*/

/* ==========================================
   CSS VARIABLES (Custom Properties)
   ========================================== */
:root {
  /* Colors */
  --color-primary: #A8F0F0;
  --color-secondary: #E8FBFB;
  --color-accent: #C2E9E9;
  --color-border: blue;
  --color-background: white;

  /* Spacing */
  --spacing-xs: 5px;
  --spacing-sm: 10px;
  --spacing-md: 15px;
  --spacing-lg: 20px;

  /* Borders */
  --border-thin: 2px;
  --border-medium: 4px;
  --border-thick: 5px;
}

/* ==========================================
   MOBILE STYLES (Default - 320px and up)
   ========================================== */

/* Container - full width on mobile */
.page-container {
  width: 100%;
  padding: var(--spacing-sm);
```

```
  box-sizing: border-box;
}
```

Requirements for Mobile

1. Layout

- Single column layout
- Full-width elements
- Adequate padding for touch targets (minimum 44px × 44px)
- Stack all content vertically

2. Header Section

- Background color: `var(--color-primary)`
- Padding: 15px
- Text alignment: center or left
- Remove border or keep minimal bottom border

3. Journal Entries

- Background: `var(--color-secondary)`
- Padding: 10px
- Margin-bottom: 15px
- Border: 4px solid `var(--color-accent)`
- Images: full width, `max-width: 100%`, `height: auto`

4. Typography

- Ensure readable font sizes (minimum 16px for body text)
- Adjust heading sizes for mobile readability

Exercise 3: Tablet Styles - Two Column Emergence (20 minutes)

Goal

Add responsive behavior for tablet-sized screens using media queries.

Breakpoint: 768px (tablets)

css

```css
/* ==========================================
   TABLET STYLES (768px and up)
   ========================================== */
@media screen and (min-width: 768px) {

  /* Container gets margins on sides */
  .page-container {
    margin: 0 5%;
    width: 90%;
  }

  /* Journal entries can show images floating */
  .journal-entry__image {
    float: right;
    width: 40%;
    margin-left: var(--spacing-md);
    margin-bottom: var(--spacing-md);
  }

  /* Header text can align right */
  .page-header__title {
    text-align: right;
  }
}
```

Requirements

- Container width: 90% with 5% margins on each side
- Images can float beside text in journal entries
- Consider using CSS Grid or Flexbox for layout
- Ensure adequate spacing between columns

Exercise 4: Desktop Styles - Full Layout (20 minutes)

Goal

Implement the full two-column layout for larger screens.

Breakpoint: 1024px (laptops/desktops)

css

```css
/* ==========================================
   DESKTOP STYLES (1024px and up)
   ========================================== */
@media screen and (min-width: 1024px) {

  /* Use CSS Grid for two-column layout */
  .page-container {
    display: grid;
    grid-template-columns: 250px 1fr;
    grid-template-areas:
      "header header"
      "sidebar main";
    gap: var(--spacing-lg);
    max-width: 1200px;
    margin: 0 auto;
  }

  .page-header {
    grid-area: header;
  }

  .friends-sidebar {
    grid-area: sidebar;
  }

  .journal-main {
    grid-area: main;
  }
}
```

Requirements

- Maximum width: 1200px, centered
- Friends list appears as left sidebar (250px wide)
- Main journal content takes remaining space
- Use CSS Grid or Flexbox
- Background image applied to container

Exercise 5: Large Desktop Optimization (15 minutes)

Breakpoint: 1440px (large desktops)

css

```css
/* ============================================
   LARGE DESKTOP STYLES (1440px and up)
   ============================================ */
@media screen and (min-width: 1440px) {

  .page-container {
    max-width: 1400px;
    grid-template-columns: 300px 1fr;
  }

  /* Increase spacing for larger screens */
  .journal-entry {
    padding: var(--spacing-lg);
  }
}
```

Requirements

- Increase maximum width to 1400px
- Optimize spacing and typography for large displays
- Ensure content doesn't stretch too wide

Exercise 6: Interactive Elements & Finishing Touches (15 minutes)

Requirements

1. Hover Effects

css

```css
/* Elegant hover effect for all interactive elements */
.journal-entry:hover,
.friends-sidebar:hover,
.page-header:hover {
  background-color: lightyellow;
  transition: background-color 0.3s ease;
}

/* Specific element hover - CHALLENGE: Do this with ONE selector! */
* {
  transition: background-color 0.3s ease;
}

*:hover {
  background-color: lightyellow;
}
```

2. Focus States (for accessibility)

css

```css
a:focus,
button:focus {
  outline: 3px solid var(--color-border);
  outline-offset: 2px;
}
```

3. Print Styles (bonus)

css

```
  @media print {
    .friends-sidebar {
      display: none;
    }

    .journal-entry {
      page-break-inside: avoid;
    }
  }
```

Testing Checklist

Responsive Testing

- [ ] 320px - Small mobile (iPhone SE)
- [ ] 375px - Medium mobile (iPhone 12)
- [ ] 768px - Tablet (iPad)
- [ ] 1024px - Laptop
- [ ] 1440px - Desktop
- [ ] 1920px - Large desktop

Browser Testing

- [ ] Chrome
- [ ] Firefox
- [ ] Safari
- [ ] Edge

Accessibility Testing

- [ ] Keyboard navigation works
- [ ] Screen reader compatible
- [ ] Sufficient color contrast
- [ ] Images have alt text
- [ ] Zoom to 200% without breaking layout

Advanced Challenges

Challenge 1: CSS Grid Mastery

Refactor the entire layout using CSS Grid only (no flexbox). Implement named grid areas for all breakpoints.

Challenge 2: Dark Mode

Add a dark mode toggle using CSS custom properties and JavaScript:

css

```
@media (prefers-color-scheme: dark) {
  :root {
    --color-background: #1a1a1a;
    --color-text: #ffffff;
    /* ... other dark mode colors */
  }
}
```

Challenge 3: Container Queries

Use the new CSS Container Queries (instead of media queries) to make components truly reusable:

css

```
.journal-entry {
  container-type: inline-size;
}

@container (min-width: 500px) {
  .journal-entry__image {
    float: right;
    width: 40%;
  }
}
```

Challenge 4: CSS Animations

Add subtle animations:

- Fade-in effect when page loads
- Smooth transitions when resizing
- Animated underlines on hover for links

Challenge 5: Performance Optimization

- Implement lazy loading for images
- Use `srcset` for responsive images
- Optimize CSS delivery (critical CSS)
- Minify CSS for production

Challenge 6: Advanced Layout Patterns

Implement a "magazine-style" layout where:

- First journal entry spans full width
- Subsequent entries appear in a masonry grid
- Use CSS Grid or Flexbox creatively

Challenge 7: Accessibility Enhancement

- Add skip navigation links
- Implement ARIA landmarks
- Create a fully keyboard-navigable interface
- Add focus management for dynamic content

Submission Requirements

1. **HTML file** with semantic structure and comments
2. **CSS file** with organized, commented code
3. **README.md** documenting:
    - Your design decisions
    - Breakpoints chosen and why
    - Challenges faced
    - Screenshots at different breakpoints
4. **Validate your code**:
    - HTML: https://validator.w3.org/
    - CSS: https://jigsaw.w3.org/css-validator/

Resources

Documentation

- MDN: Responsive Design
- CSS Tricks: A Complete Guide to Grid
- Web.dev: Responsive Design

Tools

- Chrome DevTools Device Mode
- Firefox Responsive Design Mode
- Responsively App - Multi-viewport browser

Inspiration

- MediaQueri.es - Responsive design examples
- Responsive Design Patterns

Grading Rubric

Criteria	Points
Semantic HTML structure	15
Mobile-first CSS approach	20
Proper use of media queries	20
Layout works at all breakpoints	20
Code organization & comments	10
Accessibility features	10
Creative enhancements	5
Total	**100**

Happy Coding! 🎨📱💻

Remember: A responsive website isn't just about making things fit—it's about creating the best experience for each device your users choose.