**UX Instructions Exercise 1:**

Create a **responsive landing page** that:

* Uses semantic HTML
* Implements flexible images
* Uses a fluid grid layout
* Includes a "skip link" for accessibility
* Demonstrates good structure and responsive behavior

**Step-by-Step: index.html**

<!DOCTYPE html>

<html lang=“en">

* <!DOCTYPE html>: Declares this is an HTML5 document.
* lang="en": Improves accessibility and SEO by specifying the language.

<head>

<meta charset="UTF-8">

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

<title>Responsive Design Example</title>

<link rel="stylesheet" href="styles.css">

</head>

* <meta charset="UTF-8">: Character encoding for special characters.
* <meta name="viewport"...>: Crucial for mobile responsiveness. Ensures the layout scales properly on small devices.
* <title>: Page title shown in browser tabs.
* <link>: Links to your external CSS file.

<body>

<a href="#main" class="skip-link">Skip to main content</a>

* Adds a keyboard-accessible link for screen readers and keyboard users to jump past repetitive navigation.

<header class="site-header">

<h1>Responsive Web Lab</h1>

</header>

* Semantic <header> and <h1> for page heading.

<main id="main" class="container">

<section class="intro">

<h2>Welcome</h2>

<p>This page demonstrates responsive techniques including flexible images, fluid grids, and media queries.</p>

</section>

* <main> contains the primary content.
* The intro section sets up the purpose of the lab.

<section class="grid">

<article class="card">

<img src="images/sample.jpg" alt="Sample responsive" />

<p>This card scales on any screen.</p>

</article>

<article class="card">

<img src="images/sample.jpg" alt="Another example" />

<p>Try resizing the browser window.</p>

</article>

<article class="card">

<img src="images/sample.jpg" alt="Responsive image" />

<p>Images and layout adjust accordingly.</p>

</article>

</section>

</main>

* .grid is a responsive layout container using CSS Grid.
* Each .card has an image and a paragraph. Images will resize fluidly.
* alt attributes are essential for accessibility.

<footer class="site-footer">

<p>&copy; 2025 Responsive Design Lab</p>

</footer>

</body>

</html>

* <footer> concludes the layout with basic copyright.
* No <script> tags included for this version.

**styles.css Instructions & Explanation**

:root {

--primary-color: #0076ce;

--padding: 1rem;

--max-width: 1200px;

}

* Defines **CSS variables** that make styles reusable and easy to update.
* You can change your brand color or layout spacing globally by editing these values.

\* {

box-sizing: border-box;

}

* Applies border-box sizing to all elements for consistent layout calculations (padding and border are included in width).

body {

font-family: sans-serif;

margin: 0;

line-height: 1.6;

font-size: clamp(1rem, 1vw + 0.5rem, 1.2rem);

padding: var(--padding);

color: #111;

background-color: #f9f9f9;

}

* clamp() scales font size responsively between 1rem and 1.2rem based on viewport width.
* Clean font, dark text, and a light background ensure good readability.

header, footer {

background-color: var(--primary-color);

color: white;

text-align: center;

padding: 1rem 0;

}

* Uses the theme color from :root.
* Centers and spaces content within header/footer

.container {

max-width: var(--max-width);

margin: 0 auto;

}

* Centers the main content and keeps it from stretching too wide.

.grid {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(280px, 1fr));

gap: 1.5rem;

margin-top: 2rem;

}

* Creates a **fluid grid layout**.
* auto-fit automatically fills the row with as many columns as will fit.
* minmax(280px, 1fr) ensures a minimum size while allowing flexibility.

.card {

background-color: white;

border-radius: 8px;

overflow: hidden;

box-shadow: 0 4px 8px rgba(0,0,0,0.1);

padding: 1rem;

text-align: center;

}

* Creates polished-looking cards with padding, shadow, and rounded corners.

.card img {

max-width: 100%;

height: auto;

}

* Ensures images never overflow their containers.
* Keeps image proportions intact.

.skip-link {

position: absolute;

top: -40px;

left: 0;

background: #000;

color: #fff;

padding: 8px;

z-index: 100;

transition: top 0.3s;

}

.skip-link:focus {

top: 0;

}

* Makes the skip link invisible until it receives focus via keyboard, improving accessibility without clutter.

@media (min-width: 1000px) {

.card {

padding: 2rem;

font-size: 1.2rem;

}

}

* Basic media query to improve layout on wider screens.
* Increases padding and font size when the screen is at least 1000px wide.

**Summary**

This layout teaches:

* How to create flexible, fluid layouts using Grid
* How to use CSS variables for scalable design
* How to structure semantic, accessible HTML
* How to handle images responsively
* Basic media queries to adapt design to larger screens

**Non-Responsive Version – index.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

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

<title>Non-Responsive Web Example</title>

<link rel="stylesheet" href="styles.css" />

</head>

<body>

<a href="#main" class="skip-link">Skip to main content</a>

<header class="site-header">

<h1>Fixed-Width Web Lab</h1>

</header>

<main id="main" class="container">

<section class="intro">

<h2>Welcome</h2>

<p>This example uses fixed widths and does not adapt to screen size.</p>

</section>

<section class="grid">

<article class="card">

<img src="images/sample.jpg" alt="Sample fixed" />

<p>This card stays fixed in width.</p>

</article>

<article class="card">

<img src="images/sample.jpg" alt="Another example" />

<p>Try resizing the window. It won't adapt.</p>

</article>

<article class="card">

<img src="images/sample.jpg" alt="Fixed image" />

<p>Notice the layout overflow on small screens.</p>

</article>

</section>

</main>

<footer class="site-footer">

<p>&copy; 2025 Non-Responsive Design Lab</p>

</footer>

</body>

</html>

**styles.css (Non-Responsive Version)**

:root {

--primary-color: #0076ce;

--padding: 20px;

--fixed-width: 960px;

}

\* {

box-sizing: border-box;

}

body {

font-family: sans-serif;

margin: 0;

padding: var(--padding);

font-size: 16px;

color: #111;

background-color: #f9f9f9;

width: 100%;

overflow-x: auto; /\* Allows horizontal scrolling if too narrow \*/

}

header, footer {

background-color: var(--primary-color);

color: white;

text-align: center;

padding: 1rem 0;

}

.container {

width: var(--fixed-width);

margin: 0 auto;

}

.grid {

display: flex;

justify-content: space-between;

gap: 20px;

margin-top: 2rem;

}

.card {

width: 300px;

background-color: white;

border-radius: 8px;

overflow: hidden;

box-shadow: 0 4px 8px rgba(0,0,0,0.1);

padding: 1rem;

text-align: center;

}

.card img {

width: 100%;

height: auto;

}

.skip-link {

position: absolute;

top: -40px;

left: 0;

background: #000;

color: #fff;

padding: 8px;

z-index: 100;

transition: top 0.3s;

}

.skip-link:focus {

top: 0;

}

This version demonstrates:

* Layout breaks on mobile (try it on a phone!)
* Need for horizontal scrolling on narrow screens
* Fonts, grids, and cards don’t scale with device size
* Accessibility still works (e.g., skip link), but UX suffers

**Skip Link:**

The **skip link** is a small but powerful accessibility feature that greatly improves the experience for users who navigate with a keyboard or screen reader. A **skip link** allows users to bypass repetitive navigation and jump **directly to the main content** of a page.

This is especially useful for:

* People using **screen readers**
* Users **navigating via keyboard (Tab key)** rather than a mouse
* Users with **mobility impairments**

**Example in Your HTML**

<a href="#main" class="skip-link">Skip to main content</a>

href="#main" targets the <main> tag with id="main":  
<main id="main"> ... </main>

* This means if a user presses Tab and activates the skip link, the focus jumps straight to your page’s content (bypassing header/nav).

**Styling (in styles.css)**

.skip-link {

position: absolute;

top: -40px; /\* Hides it off-screen initially \*/

left: 0;

background: #000;

color: #fff;

padding: 8px;

z-index: 100;

transition: top 0.3s;

}

.skip-link:focus {

top: 0; /\* Brings it into view when focused \*/

}

**Try It Yourself:**

1. Load your page in a browser.
2. Press Tab on your keyboard.
3. The “**Skip to main content”** link appears.
4. Hit Enter – it jumps past the header straight to your content.

**Why It Matters**

* **Required for WCAG 2.1 AA** accessibility standards
* Helps users **avoid frustration** from repetitive tabbing
* Gives your page a **professional, inclusive UX**

**Note that users have to know to press the Tab key** to access them.

**Why This Is Still Standard Practice**

Even though it seems subtle or hidden, skip links are:

* **Expected** by screen reader and keyboard-only users
* Part of the **Web Content Accessibility Guidelines (WCAG)** because they solve a real problem: skipping repetitive content
* Used by **government, educational, and large org sites**

**Tips to Make It More Discoverable**

**Add a visible keyboard outline**  
Many users discover skip links because of focus outlines:

body.keyboard-navigation :focus {

outline: 2px dashed var(--primary-color);

outline-offset: 4px;

}

1. **Mention it in accessibility documentation or onboarding**  
   If you're designing for a known audience (e.g., students), mention it:  
   "Use the Tab key to jump to the 'Skip to content' link and bypass navigation."

**Make it always visible (optional)**  
For demo or educational purposes, you can temporarily show it at all times:  
  
.skip-link {

position: relative;

top: 0;

}

**Skip Link:**

|  |  |
| --- | --- |
| **Pros** | **Limitations** |
| Keyboard-friendly | Users must know to press Tab |
| Screen reader compatible | Invisible by default |
| Accessibility best practice (WCAG AA) | Not mouse-friendly |