**Accessibility in the Design Process**

**Inclusive Design Thinking**

**Definition:** Inclusive design is the practice of creating products that are usable by people with a wide range of abilities, disabilities, and other characteristics, including age, language, culture, gender, and more.

**Key Principles:**

* **Recognize exclusion:** Identify where people may be left out due to visual, motor, cognitive, or auditory impairments.
* **Solve for one, extend to many:** Address edge cases such as screen reader users or people with color blindness; the solution often improves usability for all.
* **Design for adaptability:** Allow content to be consumed in multiple ways (text, audio, visual).

**Example Use Case:**

* A travel booking site incorporates high-contrast text and keyboard navigation to help both visually impaired users and power users who prefer not to use a mouse.
* **The interface avoids overwhelming dropdowns and uses progressive disclosure for ease of use with screen readers.**

**What It Means**

1. **Avoiding Overwhelming Dropdowns:**
   * Large dropdowns with too many options (e.g., 100+ items) can confuse users—especially those using screen readers or those with cognitive or motor disabilities.
   * These dropdowns are difficult to navigate, especially when options are unlabeled or not grouped logically.
2. **Using Progressive Disclosure:**
   * Instead of showing all the information/options at once, **progressive disclosure** means showing **only essential information** first and revealing more **only when needed**.
   * This reduces cognitive load and simplifies navigation for everyone—especially screen reader users, who navigate linearly.

**Real-World Example**

**Context:** A job application form on a recruitment site.

**Bad Practice:**

<select name="job-position">

<option>Software Engineer</option>

<option>Senior Software Engineer</option>

<option>Backend Developer</option>

...

<option>Project Manager - Construction</option>

<option>Project Manager - Healthcare</option>

<option>Project Manager - IT</option>

...

<!-- 100+ options mixed together -->

</select>

* **Issue:** Long, flat list overwhelms screen reader users. There's no logical grouping. Users must tab through every item.

**Improved with Progressive Disclosure**

**Step 1:** Ask the user for a category first:

<label for="job-category">Choose a job category:</label>

<select id="job-category">

<option value="engineering">Engineering</option>

<option value="design">Design</option>

<option value="management">Management</option>

</select>

**Step 2:** Based on that selection, show a second dropdown:

<label for="job-position">Choose a job title:</label>

<select id="job-position">

<!-- Populated dynamically based on category -->

</select>

**Why this helps:**

* Reduces options shown at once
* Makes keyboard and screen reader navigation manageable
* Follows WCAG 2.1 **Guideline 3.2 (Predictable)** and **Guideline 1.3.1 (Info and Relationships)**

**How It Helps Screen Reader Users**

* Screen readers announce only a short list at a time.
* Users are given **context** ("you are selecting from Engineering roles").
* Less chance of getting lost in a long dropdown with irrelevant choices.

**Common Barriers in UX/UI**

| **Barrier** | **Description** | **Example Issue** | **Solution** |
| --- | --- | --- | --- |
| **Low color contrast** | Hard to read text against background for users with vision impairments | Gray text on a white background | Use at least 4.5:1 contrast for normal text (WCAG 1.4.3) |
| **Missing alt text** | Screen readers can’t describe images | Informational images lack description | Add alt attribute with descriptive content |
| **Inaccessible forms** | Users can’t interact or understand purpose of fields | Form fields without labels | Use <label for=""> tags or aria-label |
| **Keyboard traps** | Navigation with keyboard stops working in modals or menus | Modal does not close on Esc key or traps focus inside | Implement focus trapping and keyboard event handling |
| **Auto-playing media** | Distracts users with cognitive or sensory disabilities | Background video starts without warning | Include a pause button and avoid autoplay |
| **No captions/transcripts** | Hearing-impaired users miss audio content | Video tutorials without captions | Add captions or transcript options |

**Color Contrast**

**Definition:** Adequate contrast between text and background helps users with low vision or color blindness.

**WCAG 2.1 Requirements:**

* **Normal text**: Contrast ratio of **4.5:1**
* **Large text** (≥18pt or bold ≥14pt): **3:1**

**Example:**

/\* Bad \*/

color: #999; /\* Light gray text \*/

background-color: #fff; /\* White background \*/

/\* Good \*/

color: #222;

background-color: #fff;

**Use Case:**

* A banking app redesigns its color scheme using the **WebAIM Contrast Checker** to ensure all UI text meets contrast standards. As a result, older users report less eye strain and better readability.

**Alt Text for Images**

**Definition:** Alternative text (alt) provides textual descriptions of images for screen readers and users with slow connections.

**Best Practices:**

* Be concise but descriptive.
* Avoid "image of" or "picture of."
* Leave alt="" for purely decorative images.

**Example:**

<img src="bar-chart.png" alt="Sales bar chart showing 20% increase in Q2" />

**Use Case:**

* An e-commerce site adds alt text to product images. Screen reader users can now identify items without needing sight, improving shopping accessibility and increasing conversion rates.

**Captions and Transcripts**

**Definition:** Provide textual representation of spoken content in videos or audio files.

**Types:**

* **Captions**: Sync with audio for video content.
* **Transcripts**: Provide full script for audio or video, often below the media.

**Example:**

<video controls>

<source src="demo.mp4" type="video/mp4">

<track kind="captions" src="captions\_en.vtt" srclang="en" label="English">

</video>

**Use Case:**

* A university publishes online lectures with captions and transcripts. Students with hearing impairments can follow along easily, and others benefit from searchable text and language translation.

**Semantic HTML**

**Definition:** Use HTML elements according to their meaning and role to support accessibility.

**Good Examples:**

* Use <button> for actions, not styled <div>s.
* Use <nav>, <header>, <main>, <footer> for layout structure.
* Use <label> for form inputs and associate them with id.

**Example:**

<nav>

<ul>

<li><a href="/about">About Us</a></li>

<li><a href="/contact">Contact</a></li>

</ul>

</nav>

**Use Case:**

* A job board site replaces <div>-based menus with semantic HTML. As a result, screen reader users can skip to navigation and explore job listings more efficiently.