WEB DESIGN AND DEVELOPMENT

Week 2

CSS Specificity, IDs, and Classes

1. Specificity in CSS

When two or more rules target the same element, the browser must decide which wins.

This is **specificity**.

- Tag selectors (`p {}`) → weakest
- Class selectors (`.highlight {}`) → stronger
- ID selectors (`#intro {}`) → very strong
- Inline styles (`style="..."`) → strongest
- If still tied → the rule that appears **later** in the CSS wins.

2. Tag Selectors

- Apply to all elements of a type.
  - `p { font-size: 16px; }`
    - Every `<p>` will use `16px` font.
    - Simple, but not flexible — affects all tags of that type.

3. Classes

- Reusable styles for multiple elements
  - `<p class="warning">System Error</p>`
  - `<p class="warning">Low Battery</p>`
  - `warning { color: red; font-weight: bold; }`
    - Both `<p>` appear bold red.
      - An element can hold multiple classes (`class="warning bold center"`).

4. IDs

- Unique style hook for a single element.
  - `<h1 id="top">Welcome!</h1>`
  - `<a href="#top">Back to top</a>`
  - `#top { text-align: center; }`

✓ Should only appear once per page.
✓ Commonly used for JavaScript (`document.getElementById`) or anchor links.

5. Classes vs IDs

- **Classes**
  - repeatable styling, groups of elements.

- **IDs**
  - unique styling or JavaScript hook.

- **Best practice**:
  - Use classes for most styling; keep IDs for special cases.

6. Common Gotchas

1. Duplicate IDs

- Only one element should have a given ID.
- If repeated, JS only hooks the first — bugs guaranteed.

2. Multiple Classes Allowed

- `<p class="warning bold center">` is fine.
- IDs cannot be duplicated, but classes can be stacked.

3. Specificity Overload

- ID selectors override class selectors, even if the class should logically “win.”
- Overusing IDs makes CSS hard to maintain.

4. Order Matters

- If two rules tie, the last one written in the CSS wins.
- Example:
- `p { color: red; }`
- `p { color: green; }        Text will be green.`

5. Inline Styles Trump All

- `style="color: blue;"` beats tag, class, and ID rules.
- Avoid inline styles in real projects — messy and unscalable.

6. Syntax Mix-Ups

- `.className` → dot for classes.
- `#idName` → hash for IDs.
- Common beginner error: swapping them.

7. Browser Defaults (User Agent Styles)

- Browsers apply their own styles (e.g., blue underlined `<a>`).
- Sometimes your CSS “fails” because it’s being overridden by defaults.
- Use resets if necessary.

Takeaway

- **Tag selectors**
  - all elements of that type.
- **Classes**
  - reusable, repeatable.
- **IDs**
  - unique, one-time only.
- **Specificity hierarchy**
  - Inline > ID > Class > Tag.
- **If tied, last rule wins.**