Moath Albrashi

IS AM

CSS SELECTORS

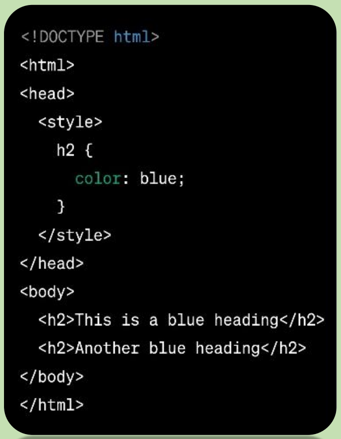
1. ***Simple Selectors and Combinators:***

- Simple selectors are methods used to target elements in an HTML or XML document. For example, `p` selects all `<p>` elements.

- Simple selectors are the most fundamental type of selectors.

- They target HTML elements based on their name or tag. For example, `h1` selects all `<h1>` elements.

- Simple selectors are essential for applying styles to specific types of elements across your web page.



```
<!DOCTYPE html>
<html>
<head>
  <style>
    h2 {
      color: blue;
    }
  </style>
</head>
<body>
  <h2>This is a blue heading</h2>
  <h2>Another blue heading</h2>
</body>
</html>
```

2. ***Class and ID Selectors:***

- Class selectors allow you to target a group of elements based on the class applied to them. For example, `<p class="highlight">`.

- ID selectors allow you to uniquely target a single element, such as `<div id="header">`.

*- **Class Selectors**:*

- Class selectors are denoted by a dot (`.`) followed by the class name (e.g., `.highlight`).

- They allow you to target multiple elements with the same class attribute.

- Useful for applying consistent styles to related elements.

- Example: `<p class="highlight">` can be styled with `.highlight`.

*- **ID Selectors**:*

- ID selectors are denoted by a hash (`#`) followed by the ID name (e.g., `#header`).

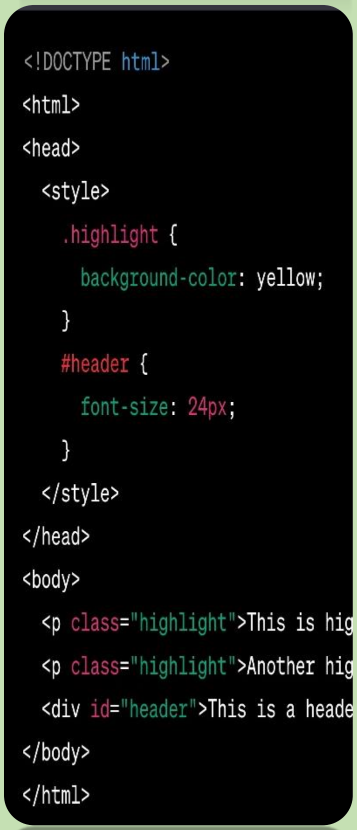
- They target a single, unique element with a specific ID attribute, and IDs should be unique on a page.

- Useful for applying unique styles to specific elements.

- Example: `<div id="header">` can be styled with `#header`.

<p class="highlight">This is highlighted text.</p>

<p class="highlight">Another highlighted text.</p>



```
<!DOCTYPE html>
<html>
<head>
  <style>
    .highlight {
      background-color: yellow;
    }
    #header {
      font-size: 24px;
    }
  </style>
</head>
<body>
  <p class="highlight">This is hig
  <p class="highlight">Another hig
  <div id="header">This is a heade
</body>
</html>
```

CSS SELECTORS

3. **Attribute Selectors:*

Common pseudo-elements include `::before` and `::after`, used for adding content before or after an element's content.

- Useful for adding decorative content or special styling to specific elements.

- Example: `p::before` adds content before all `<p>` elements.

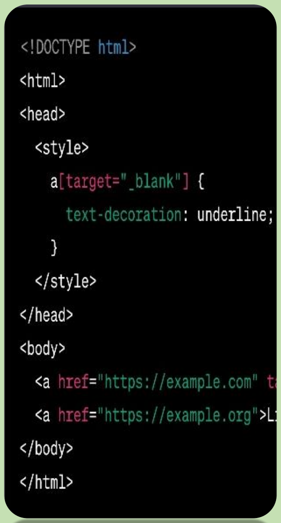
- They are used to select elements based on the values of their attributes. For example, `input[type="text"]` selects all `<input>` elements with a "text" type.

- Attribute selectors target elements based on their attributes and attribute values.

- They are enclosed in square brackets (`[]`) and can match specific attributes and values.

- Useful for styling elements with specific attributes.

- Example: `[target="_blank"]` selects elements with a "target" attribute set to "_blank".



```
<!DOCTYPE html>
<html>
<head>
  <style>
    a[target="_blank"] {
      text-decoration: underline;
    }
  </style>
</head>
<body>
  <a href="https://example.com" t
  <a href="https://example.org">L
</body>
</html>
```

4. **Pseudo-Class Selectors:**

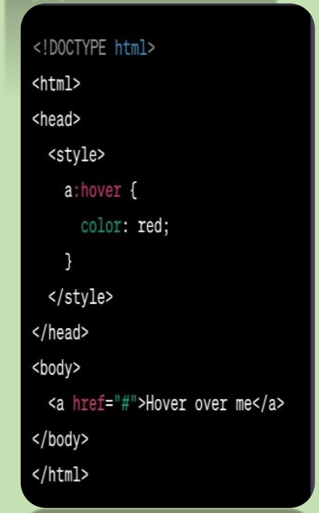
- They are used to target specific states or interactions of elements based on user actions. For instance, `a:hover` changes the formatting of links when the user hovers over them.

- Pseudo-class selectors target elements based on their state or position within the document.

- Common pseudo-classes include `:hover`, `:active`, `:focus`, `:first-child`, and `:nth-child()`.

- Useful for creating interactive and state-dependent styles.

- Example: `a:hover` selects links when the user hovers over them.



```
<!DOCTYPE html>
<html>
<head>
  <style>
    a:hover {
      color: red;
    }
  </style>
</head>
<body>
  <a href="#">Hover over me</a>
</body>
</html>
```

5. **Pseudo-Element Selectors:**

- They are used to target parts of an element itself, rather than selecting the whole element. For example, `p::before` allows you to add content before a paragraph element.

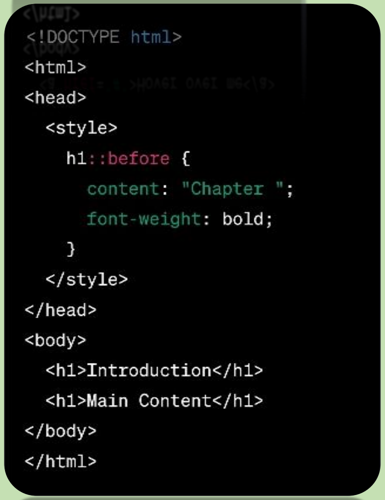
- Pseudo-element selectors target specific parts of an element's content, allowing you to style those parts separately.

- They are preceded by a double colon (`::`).Common pseudo-elements include `::before` and `::after`, used for adding content before or after an element's content.

- Useful for adding decorative content or special styling to specific elements.

- Example: `p::before` adds content before all `<p>` elements.

-



```
<!DOCTYPE html>
<html>
<head>
  <style>
    h1::before {
      content: "Chapter ";
      font-weight: bold;
    }
  </style>
</head>
<body>
  <h1>Introduction</h1>
  <h1>Main Content</h1>
</body>
</html>
```

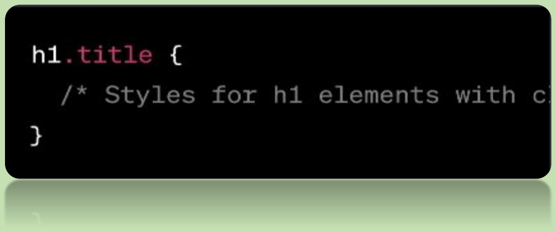
CSS SELECTORS

Combinations

CSS selectors can be combined to create more specific and targeted styling rules. Here are some common examples of combining CSS selectors:

1. **Combining Simple Selectors**:

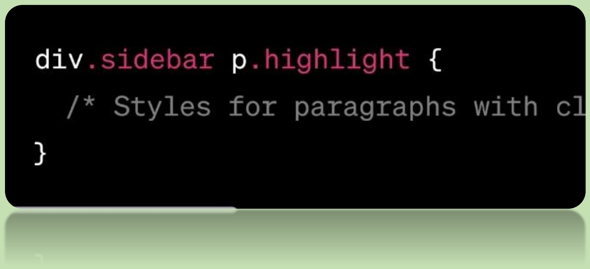
- You can combine multiple simple selectors to target specific elements. For example:



```
h1.title {
  /* Styles for h1 elements with c
}
```

2. **Combining Class and Element Selectors**:

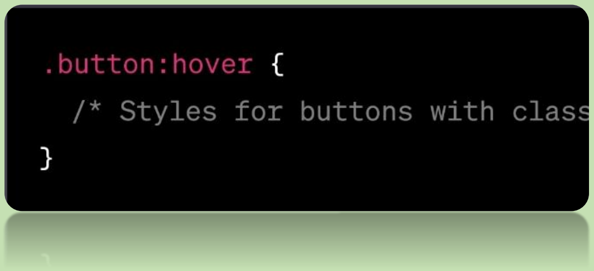
- Combining class selectors with element selectors to target specific elements with a particular class:



```
div.sidebar p.highlight {
  /* Styles for paragraphs with cl
}
```

3. **Combining Class and Pseudo-Class Selectors**:

- Combining class selectors with pseudo-class selectors for more precise styling:

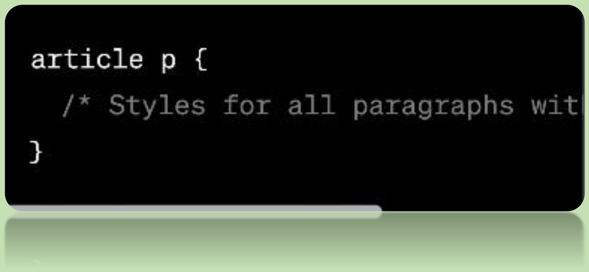


```
.button:hover {
  /* Styles for buttons with class
}
```

CSS SELECTORS

4. **Combining Descendant Selectors**:

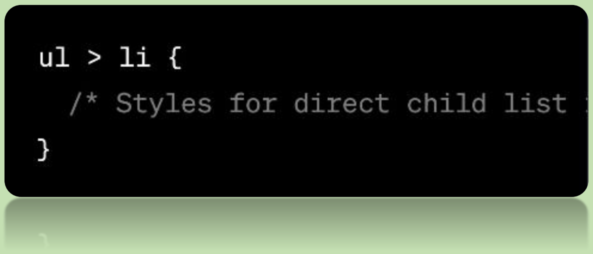
- Using a space between selectors to target elements that are descendants of another element. For example:



```
article p {
  /* Styles for all paragraphs wit
}
```

5. **Combining Child Selectors**:

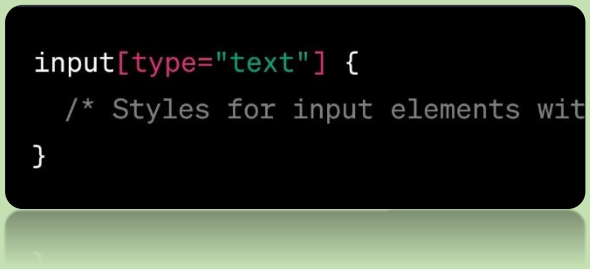
- Using the `>` symbol to target direct child elements. This is useful for styling specific child elements without affecting nested elements:



```
ul > li {
  /* Styles for direct child list
}
```

6. **Combining Attribute and Element Selectors**:

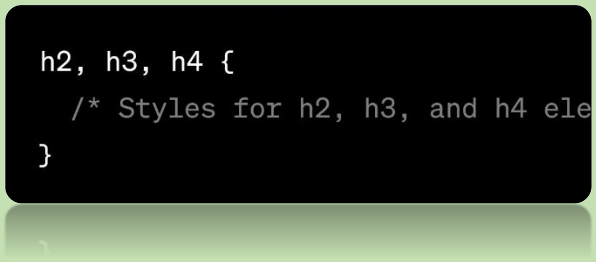
- Combining attribute selectors with element selectors to target elements with specific attributes:



```
input[type="text"] {
  /* Styles for input elements wit
}
```

7. **Combining Multiple Selectors**:

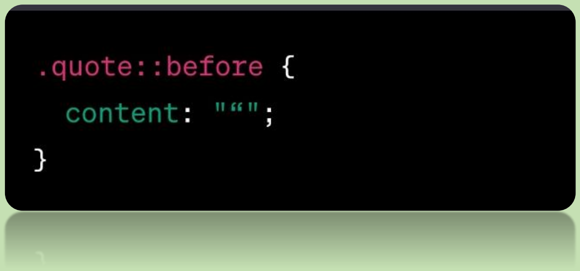
- Combining multiple selectors by separating them with commas to apply the same styles to different elements.



```
h2, h3, h4 {
  /* Styles for h2, h3, and h4 ele
}
```

8. **Combining Class and Pseudo-Element Selectors**:

- Combining class selectors with pseudo-element selectors to add content before specific elements with a class:```



```
.quote::before {
  content: "“";
}
```