**Pseudo-Classes (:)**

**Definition**: Pseudo-classes target elements based on their state or user interaction.

**Syntax**:

**selector:pseudo-class {**

**Property: value;**

**}**

**Examples**:

* :hover — Styles an element when a user hovers over it.
* :active — Styles an element when it's activated (e.g., clicked).
* :focus — Styles an element when it gains focus (e.g., via keyboard navigation).
* :nth-child(n) — Styles the nth child of a parent element.

**Use Case**: Applying styles based on user interactions or element states without altering the HTML structure.

**🔵 Pseudo-Elements (::)**

**Definition**: Pseudo-elements allow you to style specific parts of an element's content.

**Syntax**:

selector::pseudo-element {

property:value;

}

**Examples**:

* ::before — Inserts content before the element's actual content.
* ::after — Inserts content after the element's actual content.
* ::first-letter — Styles the first letter of the element's text.
* ::first-line — Styles the first line of the element's text.

**Use Case**: Styling specific parts of an element or inserting decorative content without modifying the HTML.

| **Feature** | **Pseudo-Class** | **Pseudo-Element** |
| --- | --- | --- |
| Targets | Element states or user interactions | Specific parts of an element's content |
| Syntax | Single colon (:) | Double colon (::) |
| Examples | :hover, :focus, :nth-child(n) | ::before, ::after, ::first-letter |
| HTML Modification | Not required | Not required |
| CSS Version | Introduced in CSS2 | Introduced in CSS2; double colon in CSS3 |

1. **display: flex**

flex property in CSS is the layout model that allows you to arrange the items inside the container flexibly and responsively.

**Usage**:

.container {

display: flex;

}

**2. flex-wrap**

* sets whether flex items are forced onto one line or can wrap onto multiple lines
* **Values**:
  + nowrap (default): All items are on a single line.
  + wrap: Items wrap onto multiple lines from top to bottom.
  + wrap-reverse: Items wrap onto multiple lines from bottom to top.
* **Usage**:

.container {

flex-wrap: wrap;

}

**3. gap**

Sets the spacing between flex items.

* **Usage**:

.container {

gap: 20px; /\* Applies both row and column gaps \*/

}

**4. justify-content**

Aligns items along the main axis (horizontal by default).

* **Common Values**:
  + flex-start (default): Items align to the start of the container.
  + flex-end: Items align to the end of the container.
  + center: Items are centered.
  + space-between: Equal space between items.
  + space-around: Equal space around items.
  + space-evenly: Equal space between and around items.
* **Usage**:

.container {

justify-content: center;

}

**5. align-items**

Aligns items along the cross axis (vertical by default).

* **Common Values**:
  + stretch (default): Items stretch to fill the container.
  + flex-start: Items align to the start of the cross axis.
  + flex-end: Items align to the end of the cross axis.
  + center: Items are centered.
  + baseline: Items align along their baselines.
* **Usage**:

.container {

align-items: center;

}

**6. align-content**

Aligns multiple lines of items along the cross axis.

* **Note**: Only effective when flex-wrap is set to wrap or wrap-reverse and there are multiple lines of items.
* **Common Values**:
  + flex-start: Lines align to the start of the container.
  + flex-end: Lines align to the end of the container.
  + center: Lines are centered.
  + space-between: Equal space between lines.
  + space-around: Equal space around lines.
  + space-evenly: Equal space between and around lines.
  + stretch (default): Lines stretch to fill the container.
* **Usage**:

.container {

align-content: space-between;

}

.html

<div class="container">

<div class="item">Item 1</div>

<div class="item">Item 2</div>

<div class="item">Item 3</div>

<!-- More items -->

</div>

.css

.container {

display: flex;

flex-wrap: wrap;

gap: 16px;

justify-content: space-between;

align-items: center;

align-content: space-around;

}

In this setup:

* display: flex initiates Flexbox layout.
* flex-wrap: wrap allows items to move to the next line if necessary.
* gap: 16px sets consistent spacing between items.
* justify-content: space-between distributes items evenly along the main axis.
* align-items: center vertically centers items within each line.
* align-content: space-around distributes lines evenly along the cross axis.