Pseudo-Classes, Grid and Animations

By: Gino Fernando

CSS Pseudo-Classes

A pseudo-class lets you apply styles based on an element’s state or position in the document.

Use cases include:

- Hover effects
- Link states (visited, active)
- Focusing input fields
- Styling the first or last element in a group

Common Pseudo-Classes

Pseudo-Class	Example	Description
:hover	a:hover	When the user hovers over a link
:focus	input:focus	When an element gains focus
:active	a:active	While a link is being clicked
:checked	input:checked	Checkbox or radio is selected
:disabled	input:disabled	Input is disabled
:enabled	input:enabled	Input is enabled and usable

Pseudo-Class	**Example**	**Description**
:first-child	p:first-child	First child element of its parent
:last-child	li:last-child	Last child of a parent
:nth-child(n)	li:nth-child(3)	The nth child of a parent
:not(selector)	div:not(.active)	Excludes elements matching selector
:in-range	input:in-range	Valid numeric input in specified range
:out-of-range	input:out-of-range	Value outside a given numeric range

Structural and Utility Pseudo-Classes

Pseudo-Class	**Example**	**Description**
:only-child	p:only-child	Element is the only child of its parent
:only-of-type	li:only-of-type	Only instance of its type among siblings
:nth-of-type(n)	p:nth-of-type(2)	Selects nth <p> among siblings
:root	:root	Selects the root <html> element
:target	#section:target	Active element based on anchor target

What is CSS Grid?

- CSS Grid is a 2-dimensional layout system for the web.
- Allows developers to create complex layouts easily.
- Works with both rows and columns.
- Parent = Grid Container; Children = Grid Items.

CSS Grid vs Flexbox

- Flexbox is 1-dimensional (row OR column).
- Grid is 2-dimensional (row AND column).
- Use Flexbox for linear layouts.
- Use Grid for full-page or section layouts.

Grid Container Anatomy

- display: grid;
- grid-template-columns
- grid-template-rows
- gap (or row-gap / column-gap)
- justify-items, align-items

Grid Template Areas

```
.container {
  display: grid;
  grid-template-areas:
    "header  header"
    "sidebar main"
    "footer  footer";
  grid-template-columns: 1fr 3fr;
  grid-template-rows: auto 1fr auto;
  gap: 20px;
}
```

Define Grid Areas

```
.header {
  grid-area: header;
}
.sidebar {
  grid-area: sidebar;
}
.main {
  grid-area: main;
}
.footer {
  grid-area: footer;
}
```

Content

```
<div class="container">
  <div class="header">Header</div>
  <div class="sidebar">Sidebar</div>
  <div class="main">Main Content</div>
  <div class="footer">Footer</div>
</div>
```

What is fr??

fr stands for “fractional unit” in CSS Grid. It represents a fraction of the available space in the grid container.

grid-template-columns: 1fr 2fr;

This means:

The first column takes 1 part

The second column takes 2 parts

So if the total available space is, say, 900px:

First column = 300px

Second column = 600px

What Are Media Queries?

Media Queries are rules in CSS that let your website change how it looks on different screen sizes or devices (like phones, tablets, or desktops).

They help make websites responsive, meaning they look good on any screen size.

Why use Media Queries?

- Mobile and desktop screens are different
- Media queries help your site look good on any device

```
@media (max-width: 600px) {
  .menu {
    display: none;
  }
}
```

```
@media (min-width: 601px)
and (max-width: 1024px) {
  .sidebar {
    width: 100%;
    float: none;
  }
}
```

Why use Media Queries?

- Mobile and desktop screens are different
- Media queries help your site look good on any device

```
@media (max-width: 600px) {
  .menu {
    display: none;
  }
}
```

```
@media (min-width: 601px)
and (max-width: 1024px) {
  .sidebar {
    width: 100%;
    float: none;
  }
}
```

What is a CSS Transition?

A transition is when a style changes slowly instead of changing all at once.

Example:
A button turns blue when you hover.
With a transition, it changes color smoothly, not instantly.

```
.button {
  transition: background-color 0.5s;
}
```

Shorthand Transition

```
.button {
  transition: background-color 0.5s ease;
}
```

Full Transition Properties

```
.button {
  transition-property: background-color;
  transition-duration: 0.5s;
  transition-timing-function: ease;
  transition-delay: 0s;
}
```

What Each Property Means

Property	Description
transition-property	Which CSS property will animate (e.g. color, width, background-color)
transition-duration	How long the animation lasts (e.g. 0.5s, 1s)
transition-timing-function	How the speed changes (e.g. ease, linear, ease-in, ease-out)
transition-delay	How long to wait before starting the animation

Common Timing Functions

Value	**Meaning**
ease	Starts slow, then fast, then slow (default)
linear	Same speed all the way
ease-in	Starts slow, ends fast
ease-out	Starts fast, ends slow
ease-in-out	Slow–fast–slow

Transition More Than One Property

transition: background-color 0.5s, color 0.5s;

This will:

- Change the background color
- Change the text color
- Both will change together, slowly

Note: Use commas to combine transitions in one line.

What Is a CSS Animation?

A CSS animation makes things move or change by themselves — no need to click or hover.

You can animate:
Colors, Sizes, Positions, Opacity, etc.

Unlike transitions, animations can repeat, run by themselves, or move in steps.

Color & Background

Property	**What It Does**
color	Text color
background-color	Background color
border-color	Color of borders
outline-color	Outline color (like focus rings)
box-shadow	Shadow around the element
text-shadow	Shadow behind the text
opacity	Visibility level (0 = invisible, 1 = visible)

Size & Dimensions

Property	**What It Does**
width	Width of the element
height	Height of the element
max-width	Max allowed width
max-height	Max allowed height
min-width	Min allowed width
min-height	Min allowed height
padding	Space inside the box
margin	Space outside the box
font-size	Size of text
letter-spacing	Space between letters
line-height	Height between lines of text

Position & Movement

Property	**What It Does**
top, left, right, bottom	Moves element using positioning
transform	Rotate, scale, move, or skew elements
translateX/Y	Move element along X or Y axis (via transform)
z-index	Layer position (front/back)

Visual Effects

Property	**What It Does**
visibility	Hide or show without removing space
border-width	Thickness of border
border-radius	Round corners
filter	Blur, brightness, grayscale, etc.
clip-path	Cut the shape of the element

Other (Advanced or Visual)

Property	**What It Does**
background-position	Change position of background images
background-size	Zoom or shrink background
flex	Adjust size inside a flex container
align-items, justify-content	Animate layout alignment (in Flexbox)
grid-template-columns/rows	Change grid sizes dynamically

Multiple Animations on One Element

```
@keyframes move {
  to { transform: translateX(100px); }
}

@keyframes fade {
  to { opacity: 0; }
}

.box {
  animation: move 2s linear, fade 2s ease-in;
}
```

What happens:

The box moves to the right

At the same time, it fades out