

CSS

Notes for Professionals

Chapter 4: Selectors

CSS selectors identify specific HTML elements as targets for CSS styles. This topic covers how CSS selectors identify specific HTML elements as targets for CSS styles. This topic covers how CSS selectors identify specific HTML elements as targets for CSS styles. This topic covers how CSS selectors identify specific HTML elements as targets for CSS styles.

Section 4.1: Basic selectors

Selector	Description
*	Universal selector (all elements)
div	Tag selector (all elements with class <code>div</code>)
.blue	Class selector (all elements with class <code>blue</code>)
.blue.red	Class selector (all elements with class <code>blue</code> and <code>red</code>)
#headline	ID selector (the element with "id" attribute set to <code>headline</code>)
:pseudo-class	All elements with pseudo-class
:pseudo-element	Element that matches pseudo-element
[lang=en]	Element that matches lang declaration, for example <code>espan</code> [lang="es"]
*div > p	Child selector

Note: The value of an ID must be unique in a web page. It is a violation of the HTML standard if the value of an ID more than once in the same document tree.

Section 4.2: Attribute Selectors

Attribute selectors can be used with various types of operators that change the way an element is selected using the presence of a given attribute or attribute value.

Selector(s)	Matched element	Selects elements...
[attr]	<div attr="val">	With attribute <code>attr</code>
[attr~"val"]	<div attr="val">	Where attribute <code>attr</code> has <code>val</code>
[attr ="val"]	<div attr="val">	Where <code>val</code> appears in the whitespace-separated list of <code>attr</code>
[attr^="val"]	<div attr="val">	Where <code>attr</code> 's value begins with <code>val</code>
[attr\$="val"]	<div attr="val">	Where <code>attr</code> 's value ends with <code>val</code>
[attr*="val"]	<div attr="val">	Where <code>attr</code> 's value contains <code>val</code>
[attr="val"]	<div attr="val">	Where <code>attr</code> 's value is <code>val</code>
[attr="val" i]	<div attr="val">	Where <code>attr</code> 's value is <code>val</code> or starts with <code>val</code> and <code>attr</code> is followed by <code>i</code> (case-insensitive)
[attr="val" i]	<div attr="val">	Where <code>attr</code> has <code>val</code> value ignoring <code>val</code> 's letter case

Notes:
1. The attribute value can be surrounded by either single-quotes or double-quotes. No quotes at all may also work, but it's not valid according to the CSS standard, and is discouraged.

Chapter 24: Grid

Grid layout is a new and powerful CSS layout system that allows to divide a web page content into rows and columns in an easy way.

Section 24.1: Basic Example

Property Possible Values
`display: grid / inline-grid`

The CSS Grid is defined as a display property. It applies to a parent element and its immediate children only.

Consider the following markup:

```
<section class="container">
  <div class="item1">Item1</div>
  <div class="item2">Item2</div>
  <div class="item3">Item3</div>
  <div class="item4">Item4</div>
</section>
```

The easiest way to define the markup structure above as a grid is to simply set its display property to grid:

```
.container {
  display: grid;
}
```

However, doing this will invariably cause all the child elements to collapse on top of one another. This is because the children do not currently know how to position themselves within the grid. But we can explicitly tell them.

First we need to tell the grid element, `.container`, how many rows and columns will make up its structure and we can do this using the `grid-columns` and `grid-rows` properties (note the pluralization):

```
.container {
  display: grid;
  grid-columns: 1fr 1fr 1fr;
  grid-rows: 50px 50px;
}
```

However, that still doesn't help us much because we need to give an order to each child element. We can do this by specifying the `grid-column` and `grid-row` values which will tell it where it sits in the grid:

```
.container .item1 {
  grid-column: 1;
  grid-row: 1;
}
.container .item2 {
  grid-column: 2;
  grid-row: 1;
}
.container .item3 {
  grid-column: 1;
  grid-row: 2;
}
.container .item4 {
  grid-column: 2;
  grid-row: 2;
}
```

CSS Notes for Professionals

Chapter 27: Animations

Parameter	Details
property	Either the CSS property to transition on, or <code>all</code> , which specifies all transition-able properties.
duration	Transition time, either in seconds or milliseconds.
timing-function	Specifies a function to define how intermediate values for properties are computed. Common values are <code>ease</code> , <code>linear</code> , and <code>step-end</code> . Check out the EASING FUNCTION CHANGE issues for more.
delay	Amount of time, in seconds or milliseconds, to wait before playing the animation.

@keyframes
[from | to] <percentage> {
 <back>
}

You can either specify a set time with a percentage value, or two percentage values, like `10%`, `90%`, for a period of time where the keyframe's six attributes are set. Any amount of CSS attributes for the keyframe.

Section 27.1: Animations with keyframes

For multi-stage CSS animations, you can create CSS **keyframes**. Keyframes allow you to define multiple animation points, called a keyframe, to define more complex animations.

Basic Example

In this example, we'll make a basic background animation that cycles between all colors.

```
@keyframes rainbow-background {
  0% { background-color: #FF0000; }
  16.66% { background-color: #FF0000; }
  33.33% { background-color: #FF0000; }
  50.00% { background-color: #00FF00; }
  66.67% { background-color: #00FF00; }
  83.33% { background-color: #00FF00; }
  100.00% { background-color: #0000FF; }
}
```

Look About

There's a few different things to note here. First, the actual `@keyframes` syntax. This sets the name of the animation to `rainbow-background`.

CSS Notes for Professionals

200+ pages of professional hints and tricks

Contents

About	1
Chapter 1: Getting started with CSS	2
Section 1.1: External Stylesheet	2
Section 1.2: Internal Styles	3
Section 1.3: CSS @import rule (one of CSS at-rule)	4
Section 1.4: Inline Styles	4
Section 1.5: Changing CSS with JavaScript	4
Section 1.6: Styling Lists with CSS	5
Chapter 2: Structure and Formatting of a CSS Rule	7
Section 2.1: Property Lists	7
Section 2.2: Multiple Selectors	7
Section 2.3: Rules, Selectors, and Declaration Blocks	7
Chapter 3: Comments	8
Section 3.1: Single Line	8
Section 3.2: Multiple Line	8
Chapter 4: Selectors	9
Section 4.1: Basic selectors	9
Section 4.2: Attribute Selectors	9
Section 4.3: Combinators	12
Section 4.4: Pseudo-classes	13
Section 4.5: Child Pseudo Class	15
Section 4.6: Class Name Selectors	16
Section 4.7: Select element using its ID without the high specificity of the ID selector	17
Section 4.8: The :last-of-type selector	17
Section 4.9: CSS3 :in-range selector example	17
Section 4.10: A. The :not pseudo-class example & B. :focus-within CSS pseudo-class	18
Section 4.11: Global boolean with checkbox:checked and ~ (general sibling combinator)	19
Section 4.12: ID selectors	20
Section 4.13: How to style a Range input	21
Section 4.14: The :only-child pseudo-class selector example	21
Chapter 5: Backgrounds	22
Section 5.1: Background Color	22
Section 5.2: Background Gradients	24
Section 5.3: Background Image	25
Section 5.4: Background Shorthand	26
Section 5.5: Background Size	27
Section 5.6: Background Position	31
Section 5.7: The background-origin property	32
Section 5.8: Multiple Background Image	34
Section 5.9: Background Attachment	35
Section 5.10: Background Clip	36
Section 5.11: Background Repeat	37
Section 5.12: background-blend-mode Property	37
Section 5.13: Background Color with Opacity	38
Chapter 6: Centering	39
Section 6.1: Using Flexbox	39
Section 6.2: Using CSS transform	40

Section 6.3: Using margin: 0 auto;	41
Section 6.4: Using text-align	42
Section 6.5: Using position: absolute	42
Section 6.6: Using calc()	43
Section 6.7: Using line-height	43
Section 6.8: Vertical align anything with 3 lines of code	44
Section 6.9: Centering in relation to another item	44
Section 6.10: Ghost element technique (Michał Czernow's hack)	45
Section 6.11: Centering vertically and horizontally without worrying about height or width	46
Section 6.12: Vertically align an image inside div	47
Section 6.13: Centering with fixed size	47
Section 6.14: Vertically align dynamic height elements	49
Section 6.15: Horizontal and Vertical centering using table layout	49
Chapter 7: The Box Model	51
Section 7.1: What is the Box Model?	51
Section 7.2: box-sizing	52
Chapter 8: Margins	55
Section 8.1: Margin Collapsing	55
Section 8.2: Apply Margin on a Given Side	57
Section 8.3: Margin property simplification	58
Section 8.4: Horizontally center elements on a page using margin	58
Section 8.5: Example 1:	59
Section 8.6: Negative margins	59
Chapter 9: Padding	61
Section 9.1: Padding Shorthand	61
Section 9.2: Padding on a given side	62
Chapter 10: Border	63
Section 10.1: border-radius	63
Section 10.2: border-style	64
Section 10.3: Multiple Borders	65
Section 10.4: border (shorthands)	66
Section 10.5: border-collapse	66
Section 10.6: border-image	67
Section 10.7: Creating a multi-colored border using border-image	67
Section 10.8: border-[left right top bottom]	68
Chapter 11: Outlines	69
Section 11.1: Overview	69
Section 11.2: outline-style	69
Chapter 12: Overflow	71
Section 12.1: overflow-wrap	71
Section 12.2: overflow-x and overflow-y	72
Section 12.3: overflow: scroll	73
Section 12.4: overflow: visible	73
Section 12.5: Block Formatting Context Created with Overflow	74
Chapter 13: Media Queries	76
Section 13.1: Terminology and Structure	76
Section 13.2: Basic Example	77
Section 13.3: mediatype	77
Section 13.4: Media Queries for Retina and Non Retina Screens	78

End of ebook preview

Download the full PDF tutorial from the link below :

[Click Here](#)