1. Introduction

Various ways to provide CSS

Normalizing CSS

Selectors

Colors

Gradients

Borders

Shadows

2. Providing CSS

|  |  |  |
| --- | --- | --- |
|  |  | Precedence |
| External stylesheets | Write styles in a separate file | 3 |
| Embedded stylesheets | Include stylesheet in HTML document (in the head section) | 2 |
| Inline styles | Apply styles directly to an HTML element | 1 |

Problems with embedded stylesheets

Not scalable. It works only for one HTML page

It violates the “separation of concerns” rule

External stylesheet – more commonly used

Use <link> in HTML file to link to the CSS file

|  |  |
| --- | --- |
| <link> | <a> (anchor) |
| Empty element | Non-empty element |
| Self-closing | Non self-closing |
| Specify a relationship with another document | Specify an object to be created on the page |
| Only in the <head> section | Only in the <body> section |

The rule defined in an embedded stylesheet overrides the rule defined in an external stylesheet.

Inline styles – try to avoid it as much as you can.

3. Normalizing CSS (need improvement)

An approach to normalize the view of webpages throughout different browsers

Website: necolas.gitb.io – normalize.css

Add the CSS file to your project

In the HTML file, add a link to the CSS file in the <head> section

4. Selectors

|  |  |
| --- | --- |
|  | Example |
| By element type | p {color: red;} |
| By ID (must be unique) | #product {color: red;} |
| By class name | .highlight {color: red;} |
| By attribute (not so common) | a[target]  a[href^]=”http”  a[href\*]=”google”  a[target]=”\_blank” |

5. Relational Selectors

Suppose there is a section with an ID of “products”

|  |  |
| --- | --- |
| #products p | Affects all <p> elements in the section |
| #products>p | Affects only the immediate <p> child of the section |
| #products+p | Affects only the immediate <p> sibling of the section |
| #products~p | Affects all <p> siblings of the section |

Relational selectors make cleaner markup. However, they can be fragile. A change of relative location can mess up the display. Also, it is not as fast a basic selectors.

Use relative selectors only when the structure of a website does not change often.

6. Pseudo-class selectors

Build-in classes that browsers recognize by default.

All pseudo classes start with a colon (:)

|  |  |
| --- | --- |
|  |  |
| first-child | The style applies to the first child element. It is fragile since it does not recognize the type of the child element. |
| first-of-type | The style applies to each of the first unique type of child elements |
| Last-child and last-of-type |  |
| nth-child() | Style only certain child elements based on the rule given in () (even, odd, 3th, etc) |
| Visited | Style the visited link |
| link | Style the link |
| Hover, focus | Style the content when the mouse hovers over it ()hover) or it is selected by a keyboard (focus). Always use hover and focus together to make the effect consistent. |

7. Pseudo-element selectors

Build-in element selectors to style part of an element

Use double colon (::) to quote the selector

|  |  |
| --- | --- |
|  |  |
| ::first-letter |  |
| ::first-line |  |

8. Selector specificity (weight)

|  |  |
| --- | --- |
| **Selector** | **Weight** |
| ID selector | Highest |
| Class & attribute selector | Medium |
| Element selector | lowest |

Q: What if there are two rules with equal weight?

A: The last rule applies

Q: What if I want to override the selector specificity?

A: Two solutions:

1. Use the “important” keyword (color: dodgerblue !important;). But avoid it as much as you can

2. Write more specific rules. For example, combine class and ID selectors. But don’t over do it.

9. Inheritance

Some HTML elements inherit CSS properties from parents

Example: A <strong> element inherits the color property of its <p> parent element

\* To enforce inheritance, use "inherit".

\* To remove inheritance, use "initial".

Topographic elements use inheritance a lot.

10. Colors

|  |  |
| --- | --- |
| Named colors |  |
| RGB (read, green, and blue) | An “rgba” system allows transparency setting  “a” means alpha channel. |
| HSL (hue, saturation, and lightness) | Hue – the degree of the color on a standard color wheel  Saturation – percentage of color saturation  Lightness – brightness or darkness of the color  Also has a variation of “hsla” to allow transparency setting. |
| Hexadecimal | Can be used to represent RGB, but does not have the transparency effect.  A more common way because it is more compact |

11. Gradients

To create smooth transitions between two colors

Gradients are images. So we cannot use “background-color” for gradients. Instead, we should use either “background-image” or “background” for gradients.

|  |  |
| --- | --- |
|  |  |
| Linear-gradient | (color1, color2, …)  (to right, color1, color2)  (to bottom right, color1, color2)  (45deg, color1, color2)  (color1, color2 30%) (30% defines where color2 should start) |
| Radio-gradient |  |
|  |  |

Use “gradient generator” to generate gradients

<https://cssgradient.io/>

12. Borders

Syntax: border: weight style color (The order of properties can be changed)

|  |  |
| --- | --- |
|  |  |
| border-top/bottom/left/right |  |
| Border-width | 4 values: top, right, bottom, left  3-values: top, right and left, bottom  2-values: top and bottom, right and left  1-value : all borders |
| Border-radius | Makes rounded corner |

Go to <https://css-tricks.com/the-shapes-of-css/> to find all css tricks

13. Shadows

Box-shadow: horizontal-distance, vertical-distance, (softening effect)

Positive value means moving right or down

Negative value means moving left or up

Trick: use rgba and transparency to make the shadow blend seamlessly to the background.