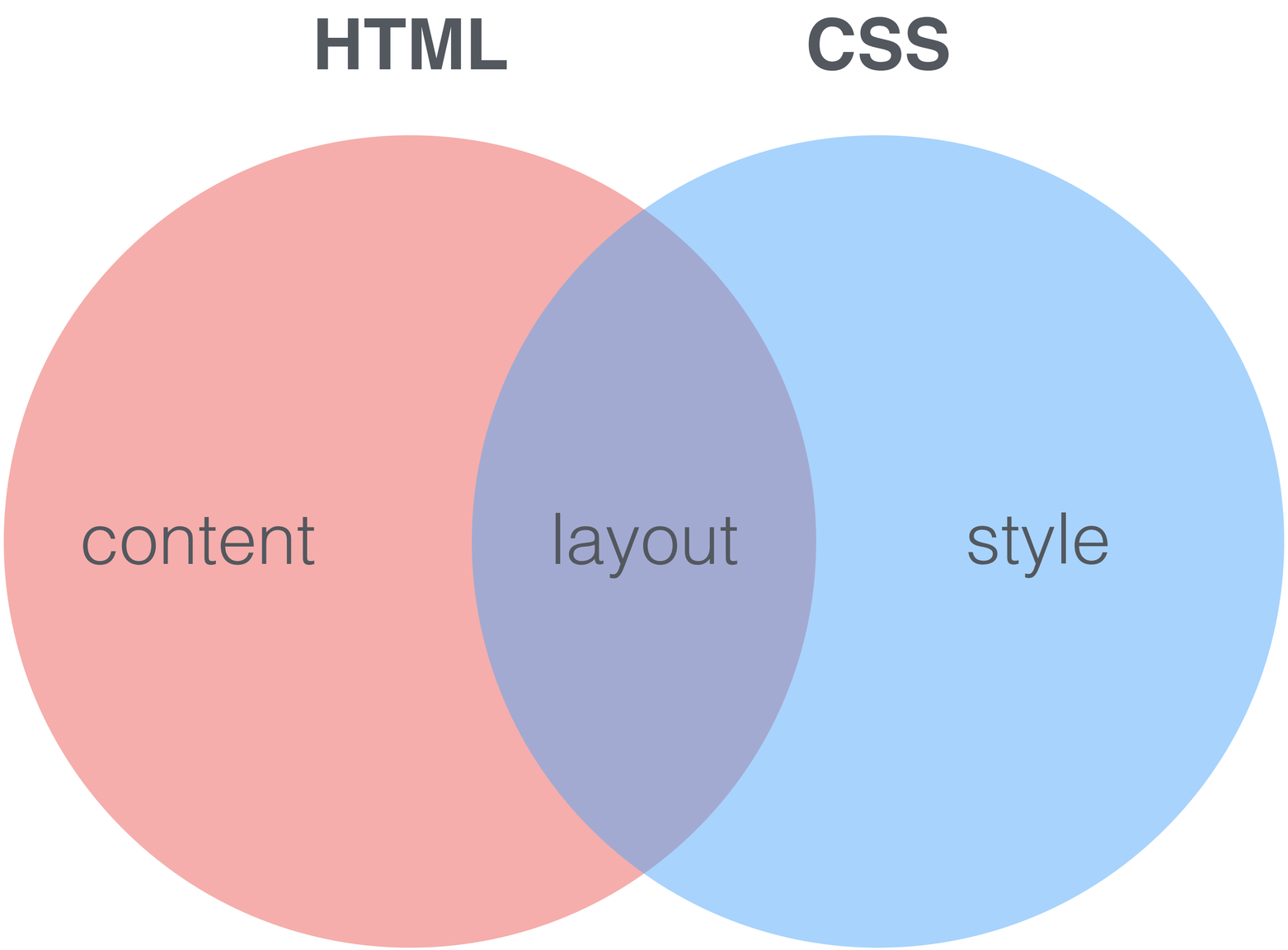
HTML & CSS

Layout laid out


HTML
CSS
content
layout
style

WITH CSS

CSS

Languages Edit

CSS (Cascading Style Sheets) is a declarative language that controls how webpages look in the browser. The browser applies CSS style declarations to selected elements to display them properly. A style declaration contains the properties and their values, which determine how a webpage looks.

CSS is one of the three core Web technologies, along with HTML and JavaScript. CSS usually styles HTML elements, but can be also used with other markup languages like SVG or XML.

A CSS rule is a set of properties associated with a selector. Here is an example that makes every HTML paragraph yellow against a black background:

```
/* The selector "p" indicate that all paragraphs in the document will be affected by that rule */
p {
  /* The "color" property defines the text color, in this case yellow. */
  color: yellow;

  /* The "background-color" property defines the background color, in this case black. */
  background-color: black
}
```

"Cascading" refers to the rules that govern how selectors are prioritized to change a page's appearance. This is a very important feature, since a complex website can have thousands of CSS rules.

Learn more

General knowledge

WITHOUT CSS

CSS

Jump to:

1. Learn more

CSS (Cascading Style Sheets) is a declarative language that controls how webpages look in the browser. The browser applies CSS style declarations to selected elements to display them properly. A style declaration contains the properties and their values, which determine how a webpage looks.

CSS is one of the three core Web technologies, along with HTML and JavaScript. CSS usually styles HTML elements, but can be also used with other markup languages like SVG or XML.

A CSS rule is a set of properties associated with a selector. Here is an example that makes every HTML paragraph yellow against a black background:

```
/* The selector "p" indicate that all paragraphs in the document will be affected by that rule */
p {
  /* The "color" property defines the text color, in this case yellow. */
  color: yellow;

  /* The "background-color" property defines the background color, in this case black. */
  background-color: black
}
```

"Cascading" refers to the rules that govern how selectors are prioritized to change a page's appearance. This is a very important feature, since a complex website can have thousands of CSS rules.

Learn more

General knowledge

- Learn CSS
- CSS on Wikipedia

Technical reference

- The CSS documentation on MDN
- The CSS Working Group current work

Learn about CSS

- The web course on codecademy.com

Document Tags and Contributors

Tags:

- CodingScripting
- CSS
- Glossary
- l10npriority
- Web

TERMS

```
article li > a:hover {
    border: 1px solid red;
    font-style: italic;
}
```

TERMS

```
article li > a:hover {
    border: 1px solid red;
    font-style: italic;
}
```

TERMS

rule

```
article li > a:hover {
    border: 1px solid red;
    font-style: italic;
}
```

TERMS

```
article li > a:hover {
    [ border: 1px solid red;
      font-style: italic;
}
```

TERMS

```
        article li > a:hover {
declaration [ border: 1px solid red;
              font-style: italic;
        }
```

TERMS

```
article li > a:hover {
    border: 1px solid red;
    font-style: italic;
}
```

TERMS

selector

```
article li > a:hover {
    border: 1px solid red;
    font-style: italic;
}
```

TERMS

```
article li > a:hover {
    border: 1px solid red;
    font-style: italic;
}
```

TERMS

property

```
article li > a:hover {
    border: 1px solid red;
    font-style: italic;
}
```

TERMS

```
article li > a:hover {
    border: 1px solid red;
    font-style: italic;
}
```

TERMS

value

```
article li > a:hover {
    border: 1px solid red;
    font-style: italic;
}
```

RULE EXAMPLE

```
article li > a:hover {
    border: 1px solid red;
    font-style: italic;
}
```

RULE EXAMPLE

apply **these** styles →

```
article li > a:hover {
    border: 1px solid red;
    font-style: italic;
}
```

RULE EXAMPLE

```
article li > a:hover {
    border: 1px solid red;
    font-style: italic;
}
```

apply **these** styles

to any elements matching **this** selector

RULE EXAMPLE

```
article li > a:hover {
    border: 1px solid red;
    font-style: italic;
}
```

apply **these** styles

to any elements matching **this** selector

even for any future changes

RULE EXAMPLE

```
article li > a:hover {
    border: 1px solid red;
    font-style: italic;
}
```

apply **these** styles

to any elements matching **this** selector

even for any future changes ***declarative!***

SELECTORS

SELECTORS

tag

SELECTORS

tag

class

SELECTORS

tag

class

id

SELECTORS

tag

class

id

attribute

SELECTORS

tag

class

id

attribute

pseudo-element

SELECTORS

tag

class

id

attribute

pseudo-element

pseudo-class

SELECTORS

tag

class

id

attribute

pseudo-element

pseudo-class

*

SELECTORS

tag	`input`
class	`.btn`
id	`#upload`
attribute	`[type="file"]`
pseudo-element	`::after`
pseudo-class	`:hover`
*	*

COMBINATORS

```
tag.class

tag .class

tag,.class

tag>.class
```

COMBINATORS

`tag.class` element with BOTH `tag` AND `.class`

`tag .class`

`tag,.class`

`tag>.class`

COMBINATORS

`tag.class` element with BOTH `tag` AND `.class`

`tag .class` element with `.class` whose ANCESTOR matches `tag`

`tag,.class`

`tag>.class`

COMBINATORS

`tag.class`	element with BOTH `tag` AND `.class`
`tag .class`	element with `.class` whose ANCESTOR matches `tag`
`tag,.class`	element with EITHER `tag` OR `.class`
`tag>.class`	

COMBINATORS

`tag.class`	element with BOTH `tag` AND `.class`
`tag .class`	element with `.class` whose ANCESTOR matches `tag`
`tag,.class`	element with EITHER `tag` OR `.class`
`tag>.class`	element with `.class` whose PARENT matches `tag`

CASCADING STYLE SHEETS

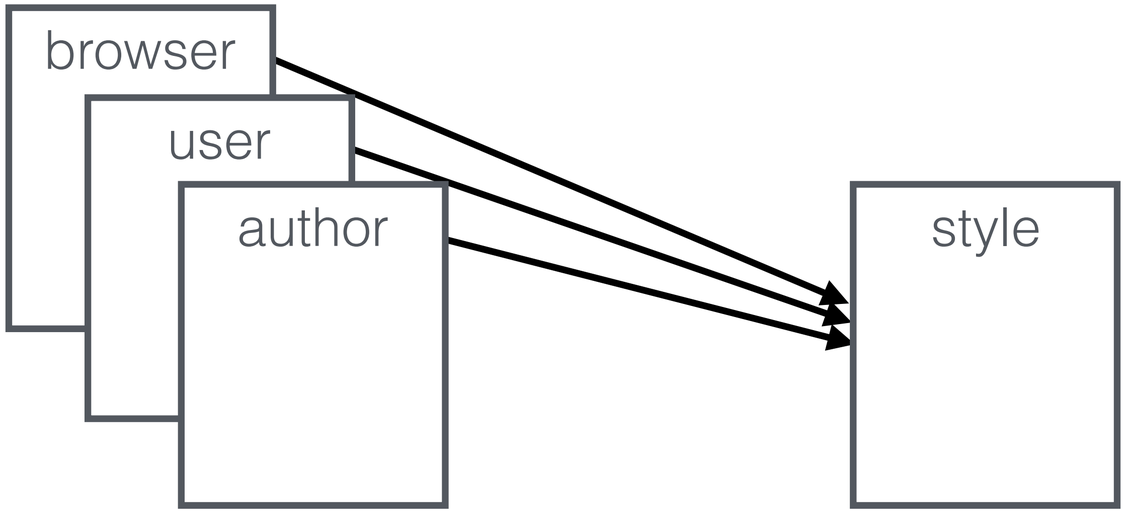
CASCADING STYLE SHEETS

CASCADING

In ~1994… *CSS had one feature that distinguished it from all the [competing style languages]: it took into account that on the Web the style of a document couldn't be designed by either the author or the reader on their own, but that their wishes had to be combined, or "cascaded," in some way.*

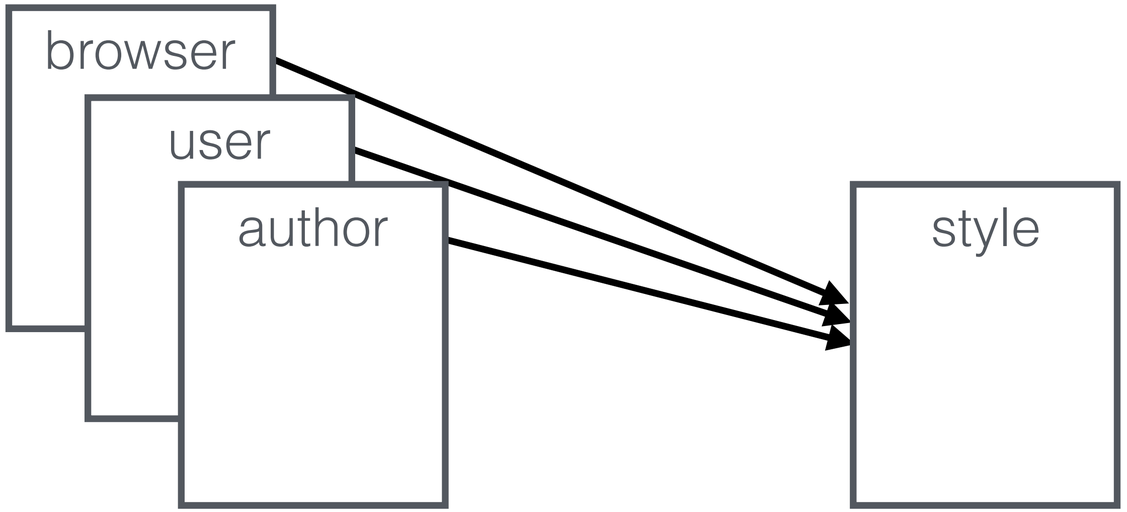
CASCADING STYLE SHEETS, DESIGNING FOR THE WEB, BY HÅKON WIUM LIE AND BERT BOS (1999) - CHAPTER 20

CASCADING



CASCADING

An element's style is a merge of every rule whose selector matches



index.html

```html
<head>
  <link rel="stylesheet" href="styles-B.css" />
  <link rel="stylesheet" href="styles-A.css" />
</head>
<body>
  <ul>
    <li style="background-color:blue;">A</li>
  </ul>
</body>
```

styles-A.css

```css
li {
  color: red;
}
```

styles-B.css

```css
li {
  font-size: 40px;
}
```

style

?

index.html

```html
<head>
  <link rel="stylesheet" href="styles-B.css" />
  <link rel="stylesheet" href="styles-A.css" />
</head>
<body>
  <ul>
    <li style="background-color:blue;">A</li>
  </ul>
</body>
```

styles-A.css

```css
li {
  color: red;
}
```

styles-B.css

```css
li {
  font-size: 40px;
}
```

style

```
element.style {
   background-color: blue;
}
li {                                   styles-A.css:1
   color: red;
}
li {                                   styles-B.css:1
   font-size: 40px;
}
li {                              user agent stylesheet
   display: list-item;
   text-align: -webkit-match-parent;
}
```

index.html

```html
<head>
  <link rel="stylesheet" href="styles-B.css" />
  <link rel="stylesheet" href="styles-A.css" />
</head>
<body>
  <ul>
    <li style="background-color:blue;">A</li>
  </ul>
</body>
```

styles-A.css

```css
li {
  color: red;
}
```

styles-B.css

```css
li {
  font-size: 40px;
}
```

style

```
element.style {
   background-color: blue;
}
li {                                   styles-A.css:1
   color: red;
}
li {                                   styles-B.css:1
   font-size: 40px;
}
li {                                   user agent stylesheet
   display: list-item;
   text-align: -webkit-match-parent;
}
```

index.html

```html
<head>
  <link rel="stylesheet" href="styles-B.css" />
  <link rel="stylesheet" href="styles-A.css" />
</head>
<body>
  <ul>
    <li style="background-color:blue;">A</li>
  </ul>
</body>
```

styles-A.css

```css
li {
  color: red;
}
```

styles-B.css

```css
li {
  font-size: 40px;
}
```

style

```
element.style {
  background-color: blue;
}
li {                                  styles-A.css:1
  color: red;
}
li {                                  styles-B.css:1
  font-size: 40px;
}
li {                                  user agent stylesheet
  display: list-item;
  text-align: -webkit-match-parent;
}
```

index.html

```html
<head>
  <link rel="stylesheet" href="styles-B.css" />
  <link rel="stylesheet" href="styles-A.css" />
</head>
<body>
  <ul>
    <li style="background-color:blue;">A</li>
  </ul>
</body>
```

styles-A.css

```css
li {
  color: red;
}
```

styles-B.css

```css
li {
  font-size: 40px;
}
```

style

```
element.style {
  background-color: blue;
}
li {                                  styles-A.css:1
  color: red;
}
li {                                  styles-B.css:1
  font-size: 40px;
}
li {                                  user agent stylesheet
  display: list-item;
  text-align: -webkit-match-parent;
}
```

index.html

```html
<head>
  <link rel="stylesheet" href="styles-B.css" />
  <link rel="stylesheet" href="styles-A.css" />
</head>
<body>
  <ul>
    <li style="background-color:blue;">A</li>
  </ul>
</body>
```

styles-A.css

```css
li {
  color: red;
}
```

styles-B.css

```css
li {
  font-size: 40px;
}
```

view

style

```
element.style {
  background-color: blue;
}
li {                                  styles-A.css:1
  color: red;
}
li {                                  styles-B.css:1
  font-size: 40px;
}
li {                                  user agent stylesheet
  display: list-item;
  text-align: -webkit-match-parent;
}
```

What happens when declarations conflict?

```
<div id="thing"></div>
```

```
div {
  background: red;
}
```


```
#thing {
  background: blue;
}
```

```
<div id="thing"></div>
```

```
div {
  background: red;
}
```


```
#thing {
  background: blue;
}
```

```
<div class="foo"></div>
```

```
div {
  background: red;
}
```


```
.foo {
  background: green;
}
```

```
<div class="foo"></div>
```

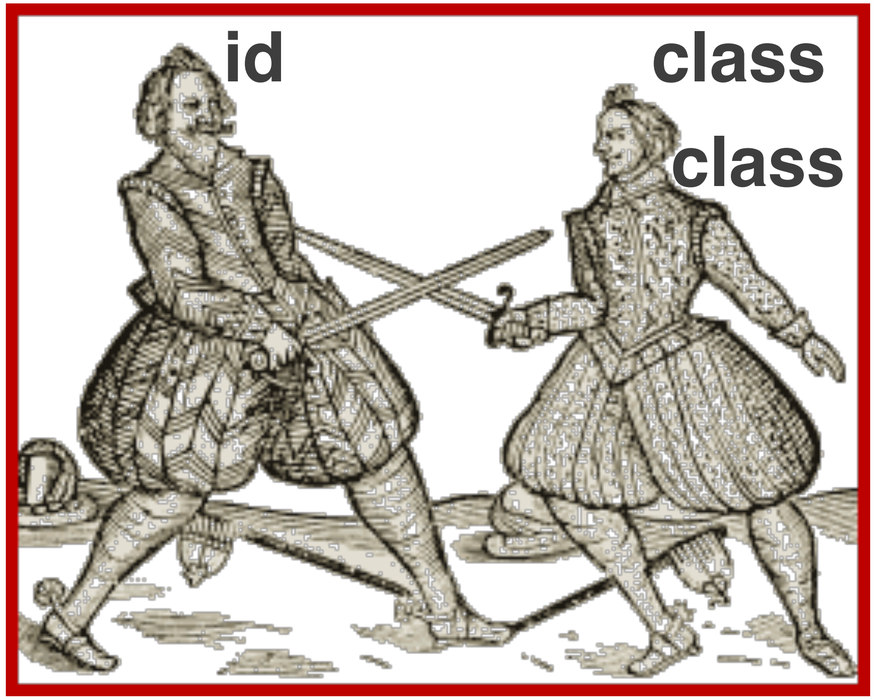
```
div {
  background: red;
}
```



```
.foo {
  background: green;
}
```

```
<div id="thing" class="foo bar"></div>
```

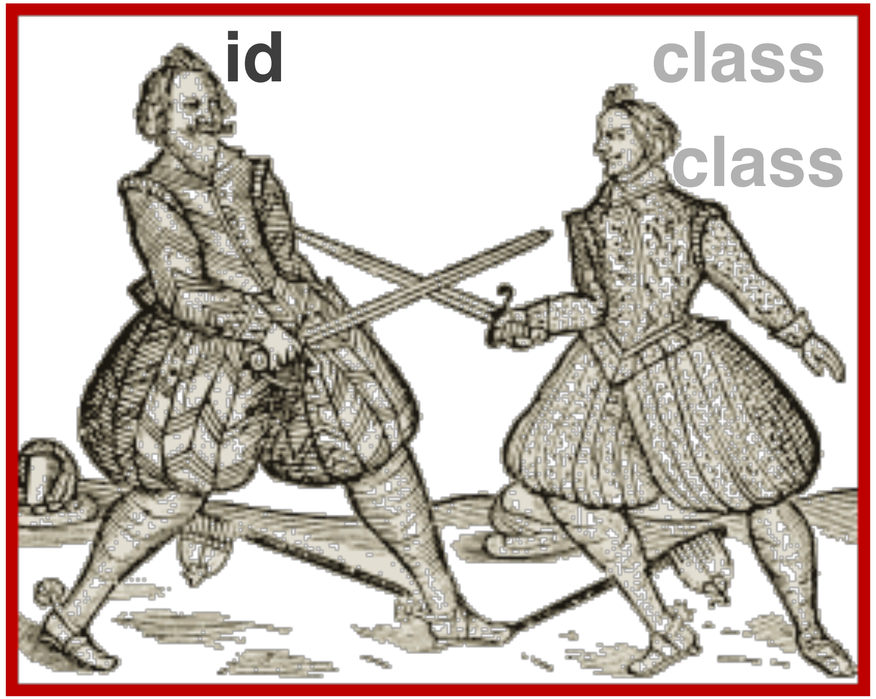
```
#thing {
  background: blue;
}
```



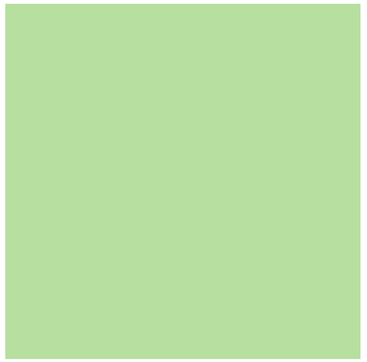

```
.foo.bar {
  background: green;
}
```

```
<div id="thing" class="foo bar"></div>
```

```
#thing {
  background: blue;
}
```

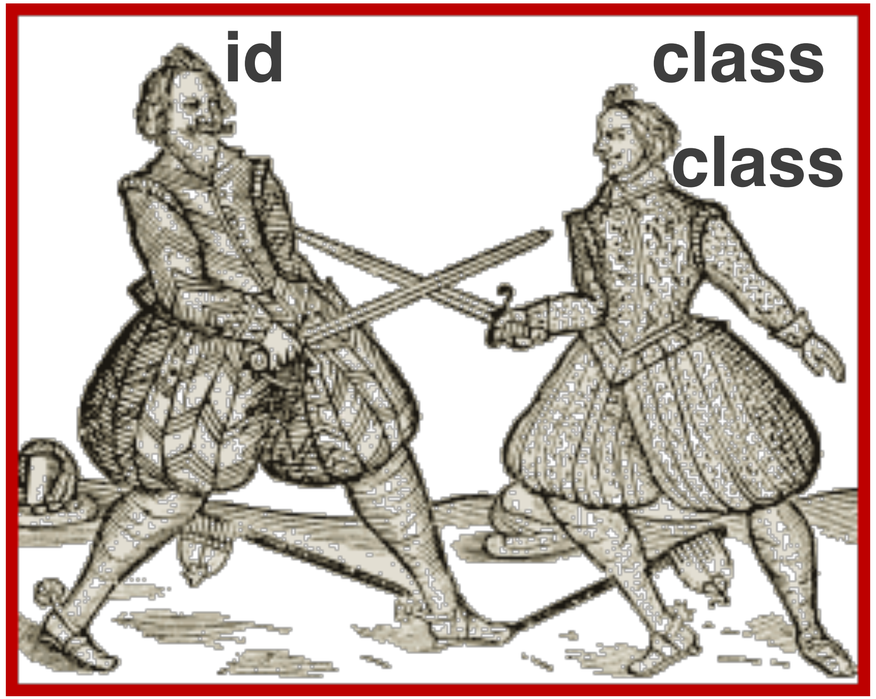



```
.foo.bar {
  background: green;
}
```



```
<div class="outer">
  <div id="thing" class="foo" style="background:orange;"></div>
</div>
```

```
#thing {
  background: blue;
}
```




```
.outer .foo {
  background: green;
}
```

```
<div class="outer">
  <div id="thing" class="foo" style="background:orange;"></div>
</div>
```

```
#thing {
  background: blue;
}
```


```
.outer .foo {
  background: green;
}
```


CSS
Global
HD


CSS
Global
HD