

WHAT IS CSS?

language for specifying the
presentations of Web documents

IF THERE WAS NO CSS...

This is a historical document, one of the oldest on the web. Teletronikk 4-93 was made available on www.nta.no/teletronikk/4.93 on Dec 17, 1993. It pioneered the concept of thumbnail images in HTML documents. Also, during the Quark-to-web conversion process, it became clear to the editor that a proper style sheet language for the web was needed.



text in images

Welcome to the electronic Teletronikk. The electronic version was available through the web one week before paper, and has enjoyed several corrections after the ink had dried. It has also recieved honorable mention at the [best of Web'94](#). Due to high demand, we are no longer able to send out complimentary paper copies, but help yourself to the electronic version - of which there are unlimited copies.

- [Guest editorial](#) by Håkon W Lie
- [Windows into Cyberspace](#) by Håkon W Lie
- [Altruism and benefit in Cyberspace](#) by Børre Ludvigsen
- [Listen to Internet](#) by Per E Dybvik
- [Telecommunications and CD-ROM - friends or foes?](#) by Erling Maartmann-Moe
- [The digital video album: On the merging of media types in multimedia](#) by Gunnar Liestøl
- [Hypertext reading as practical action: notes on technology, objectivation and knowledge](#) by Terje Rasmussen
- [Information exchange in MultiTorg](#) by Dag Solvoll, Geir Ivarsøy, Håkon W Lie, and Per E Dybvik
- [Media streams: an iconic visual language for video annotation](#) by Marc Davis
- [SCREAM: Screen-based navigation in voice messages](#) by Håkon W Lie, Per E Dybvik, and Jan Rygh
- [Telecommunications and social interaction - Social constructions in virtual space](#) by Ola Ødegård
- [Distributed Virtual Reality: applications for education, entertainment and industry](#) by Carl E Loeffler
- [Coordination: challenge of the nineties: Multimedia as a coordination technology](#) by Per M Schiefloe and Tor G Syvertsen
- [An informal requirements analysis of Norwegian public administration relative to CSCW](#) by Pål Sørgaard
- [International Information Infrastructure: social and policy considerations](#) by David Hakken

[howcome](#)

Timeline



Separation of CONTENT *from* PRESENTATION

CSS RULES

```
img {  
  border:1px solid black;  
}  
  
.photo {  
  width:300px;  
}  
  
.photo h3 {  
  font-weight:bold;  
}  
  
...
```

describe how markup
should be rendered

visual properties

positioning in page's layout

CSS RULES

Selector

.photo {

`width: 300px;`

}

Declaration

CSS SELECTORS

```
<!DOCTYPE html>
```

```
<html>
```

```
...
```

```
<body>
```

```
  <div class="photo">
```

```
    <h3>My first photo</h3>
```

```
    
```

```
  </div>
```

```
...
```

```
</body>
```

```
</html>
```

```
.photo {  
  width:300px;  
}
```

```
.photo h3 {  
  font-weight:bold;  
}
```

```
img {  
  border:1px solid black;  
}
```

```
...
```

map HTML elements to CSS rules

ELEMENT SELECTORS

html: ``

css: `img {
 border: 1px solid black;
}`

selects all elements matching the tag name

class SELECTORS

html:

```
<div class="photo">...
```

css:

```
.photo {  
  width: 300px;  
}
```

id SELECTORS

html:

```
<div id="llama-photo">...
```

css:

```
#llama-photo {  
    width: 300px;  
}
```

HIERARCHICAL SELECTORS

html:

```
<div class="photo">  
  <h3>My first photo</h3>...
```

css:

```
.photo h3 {  
  font-weight:bold;  
}
```

Which selectors promote
the most *reuse*?

WHY CASCADING?

more than one rule can apply to an HTML element

priority rules for resolving conflicts

more *specific* = higher priority (class trumps element)

some properties (**font-size**) are inherited, while others aren't (**border, background**)

LINKING TO HTML

(1) `<link rel="stylesheet" href="gallery.css" type="text/css"/>`

(2) `<html>
 <head>
 <style>
 h1 {color:red;}
 p {color:blue;}
 </style>`

(3) `<div style="color:blue;text-align:center">`



higher priority

CSS PROPERTIES

background

background-image

color

font-family

font-size

font-weight

font-style

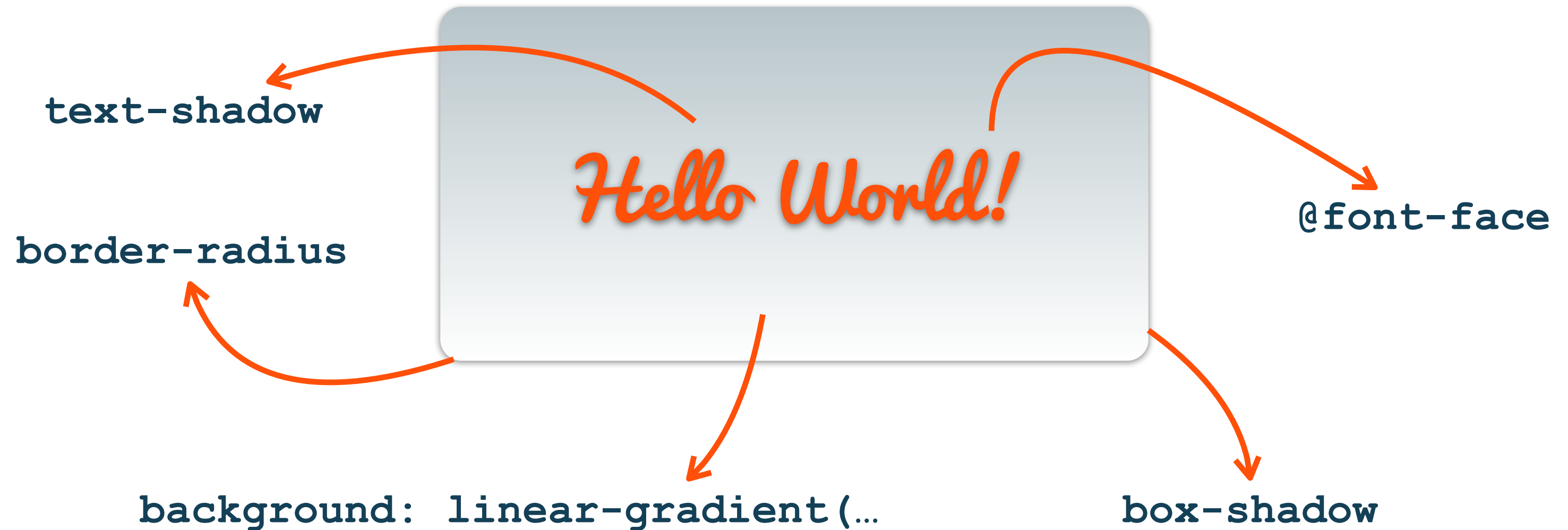
text-align

text-decoration

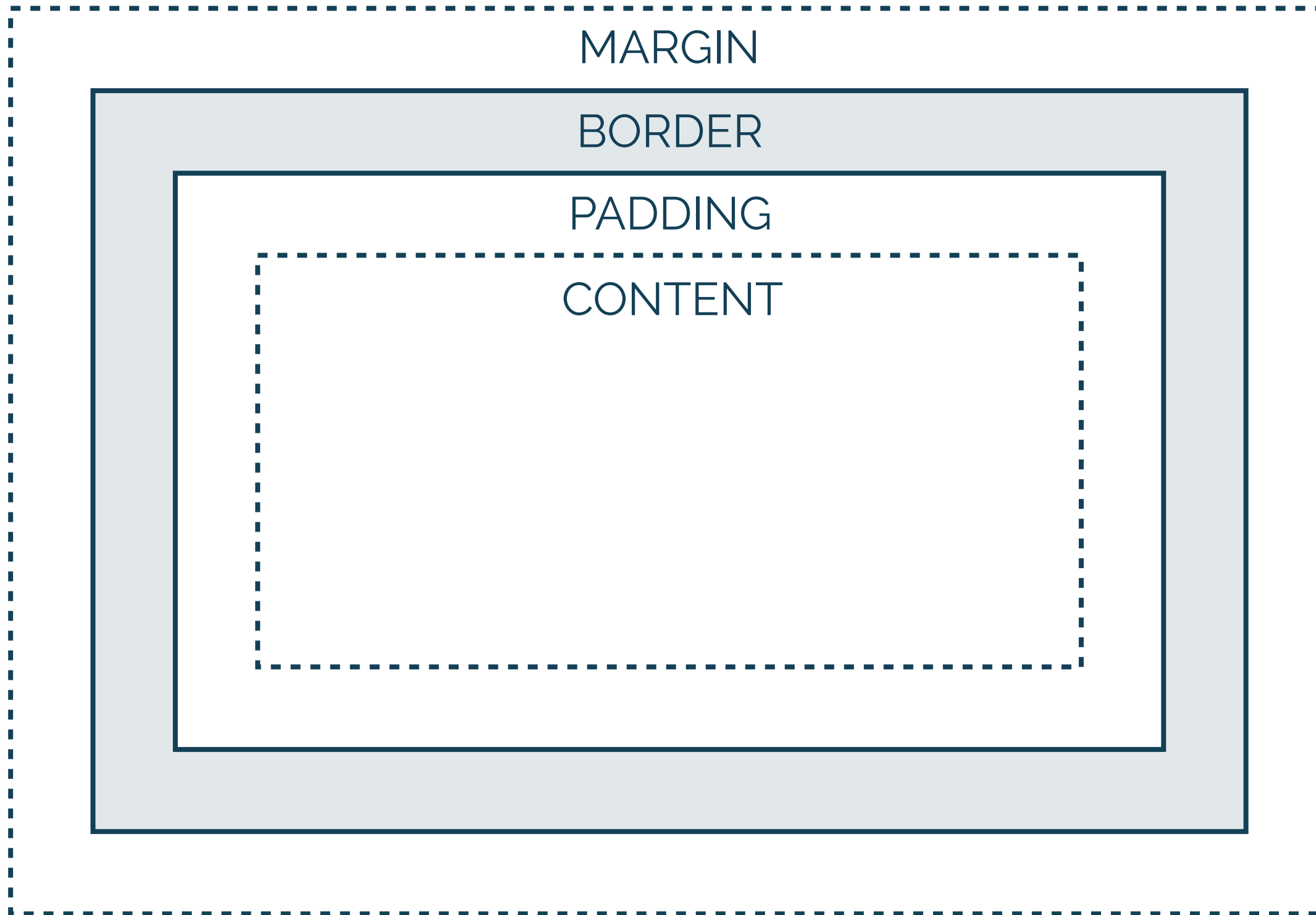


Hello World!

CSS3 PROPERTIES

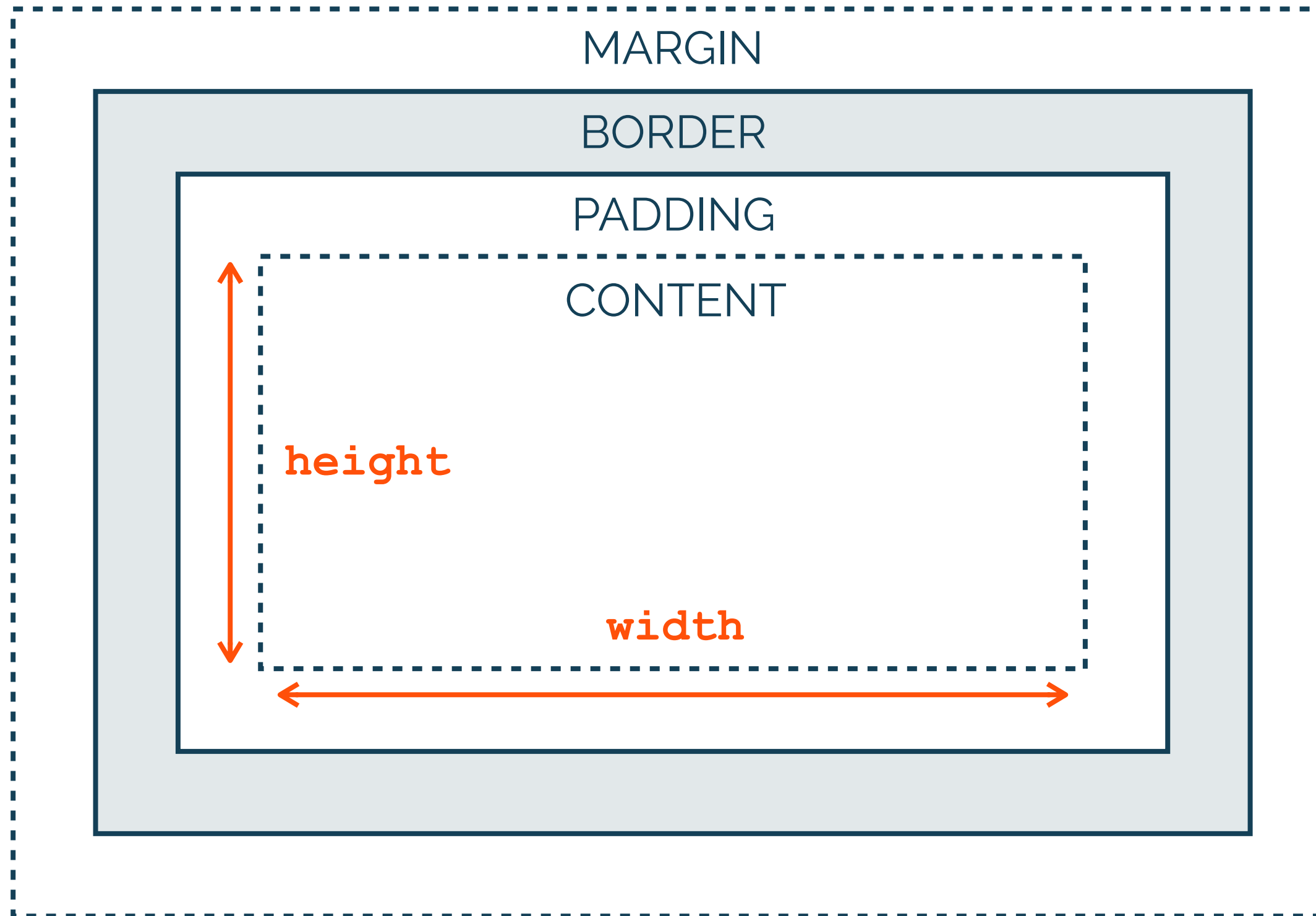


Box Model



control over
white space

Box Model



width and **height**
properties refer to content
area

to calculate full-size of the
element add padding,
border, and margins

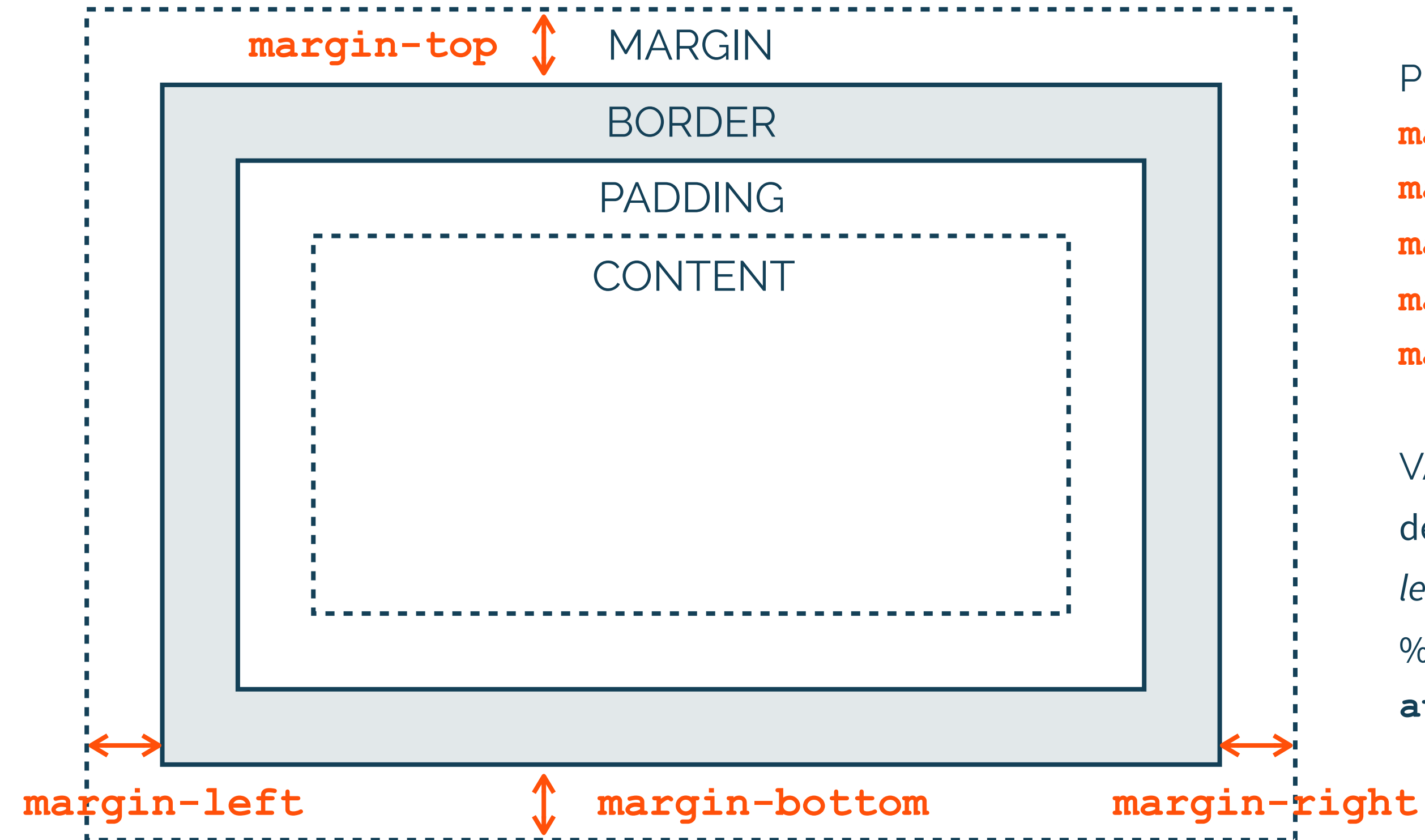
VALUES

default value is **auto**

length +/- (px, em, in, cm, pt)

% of parent's width

Box Model: Margin



PROPERTIES

`margin` (shorthand)

`margin-top`

`margin-bottom`

`margin-left`

`margin-right`

VALUES

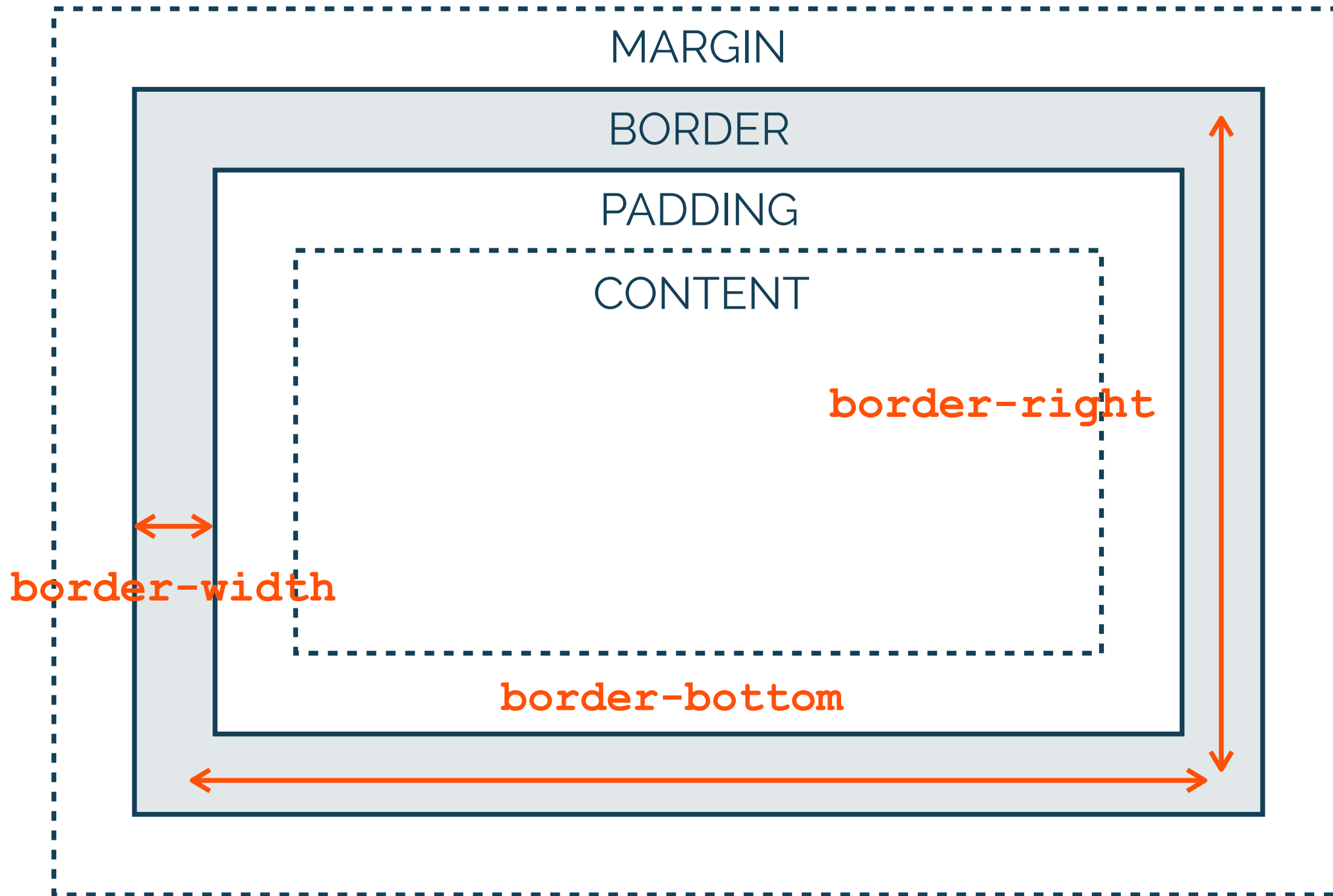
default value is 0

length +/- (px, em, in, cm, pt)

% of parent's width

auto

Box Model: Border



PROPERTIES

border (shorthand)

border-top

border-bottom

border-left

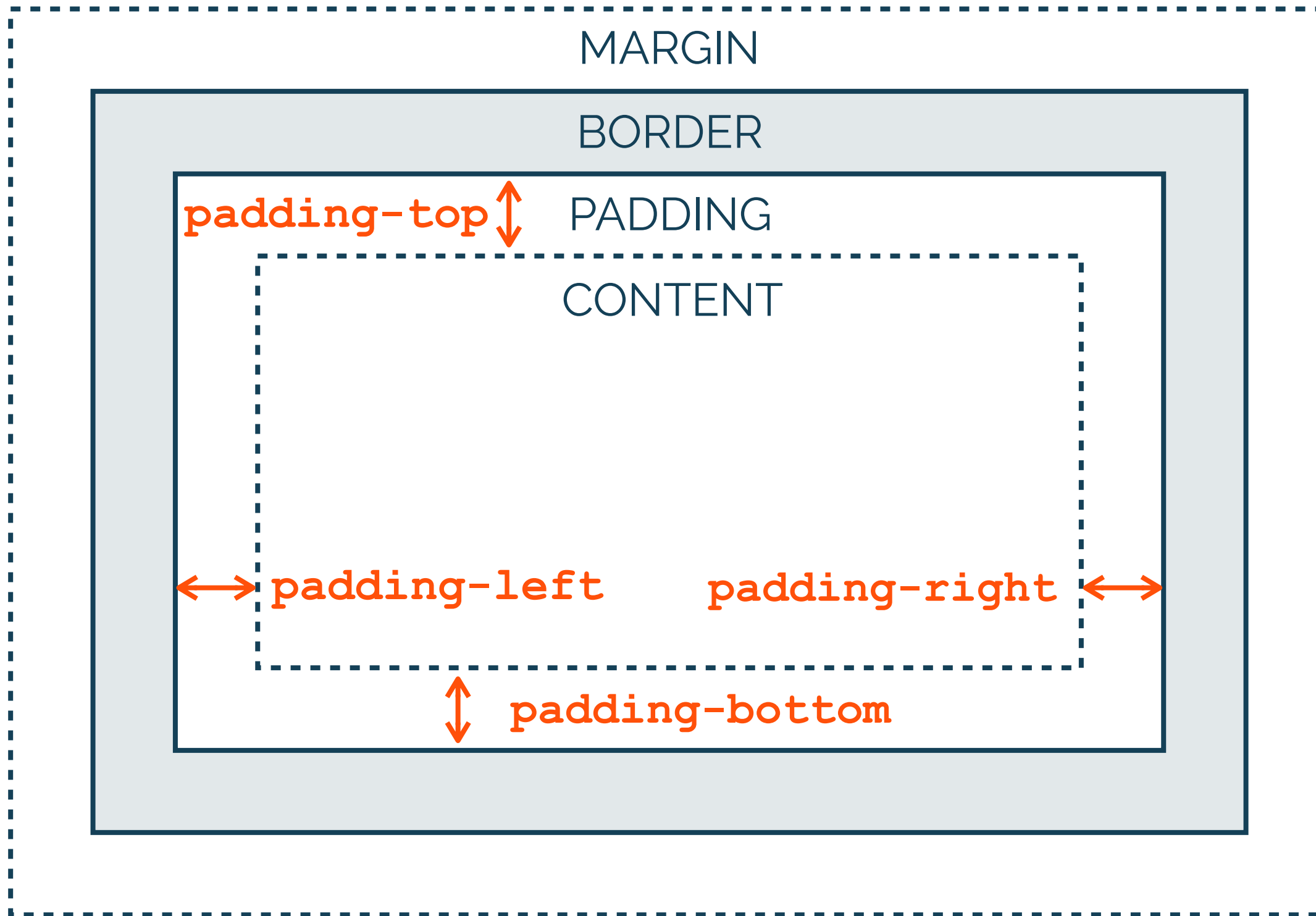
border-right

border-width

border-style

border-color

Box Model: Padding



PROPERTIES

padding (shorthand)

padding-top

padding-bottom

padding-left

padding-right

VALUES

default value is 0

length (px, em, in, cm, pt)

% of the element's width

LAYOUT

Block

rendered with preceding and following line breaks (stacked)

line breaks within nested elements collapsed if no other content

width of **auto** (default) will expand to fill entire width

Inline

rendered on a common baseline or wrap onto a new baseline below

margin, **width**, **height** properties don't affect these elements

can only contain text or other inline elements

UNITS

absolute (**px**, **in**, **cm**, **pt**) vs relative (**em**, %)

em relative to the font-size of the element

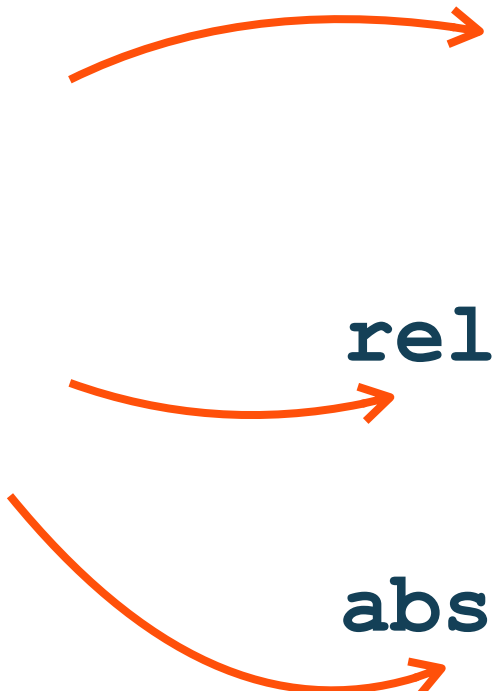
(or its parent when used to set **font-size**)

be careful when mixing different units

position


VALUE	DESCRIPTION
static	default. positioned by the flow model; unaffected by top, bottom, left, right
fixed	positioned relative to browser window; will not move when window is scrolled
relative	positioned relative to its normal position
absolute	positioned relative to the first ancestor where position!=static

use with
top
bottom
left
right



```
graph LR; A[use with<br/>top<br/>bottom<br/>left<br/>right] --> B[fixed]; A --> C[relative]; A --> D[absolute];
```

display

VALUE	DESCRIPTION
inline	default if the element is an inline element (e.g., span) displays element as inline element
block	default if the element is a block-element (e.g., div) displays element as block element
table	element behaves like table element
 none	element not displayed (doesn't appear in DOM)

not the same as
visibility: hidden;

www.w3schools.com/cssref/pr_class_display.asp

float

breaks with the flow model

pushes element to **left** or **right**, allowing other elements to wrap around it

use **clear** (**left**, **right**, **both**) to force other elements below floated ones

often used to flow text around images

Design Challenge:
horizontally center a **<div>**

CODEPEN

SOLUTION

```
<div class="outer">  
  <div class="inner">  
    </div>  
</div>
```

```
.outer {  
  height: 300px;  
  background-color: #144057;  
}  
  
.inner {  
  width: 100px;  
  height: 100px;  
  background-color: #B6C4C9;  
  
  margin: 0 auto;  
}
```

Design Challenge:
vertically center a `<div>`

CODEPEN

SOLUTION

```
<div class="outer">  
  <div class="inner">  
  </div>  
</div>
```

```
.outer {  
  height: 300px;  
  background-color: #144057;  
  
  position: relative;  
}
```

```
.inner {  
  width: 100px;  
  height: 100px;  
  background-color: #B6C4C9;  
  
  position: absolute;  
  top: 50%;  
  margin-top: -50px;  
}
```

known height!



Design Challenge:
vertically center a `<div>`
of unknown height

CODEPEN

SOLUTION

```
<div class="table-outer">
  <div class="outer">
    <div class="inner">
    </div>
  </div>
</div>
```

```
.table-outer {
  width: 100%;
  display: table;
}
```

```
.outer {
  height: 200px;
  background-color: #144057;

  display: table-cell;
  vertical-align: middle;
}
```

```
.inner {
  width: 100px;
  height: 50%;
  background-color: #B6C4C9;
}
```

css tables!



Separation of CONTENT from PRESENTATION?

purely presentational html!

```
<div class="table-outer">  
  <div class="outer">  
    <div class="inner"></div>  
  </div>  
</div>
```

a lot of HTML suffers from presentational `div` bloat

Separation of CONTENT *from* PRESENTATION?

good in theory, doesn't always work in practice

DOMs are often cluttered with presentational HTML

Add higher-level design attributes to CSS
(*i.e.*, CSS3 implemented rounded corners)

Research: Cascading Tree Sheets (CTS) [Benson et al.]

CSS PREPROCESSORS

languages that extend CSS in meaningful ways

features: variables, nesting, mixins, inheritance

shrinks developer's codebase and compiles into CSS

popular CSS preprocessors: LESS and SASS

VARIABLES

```
$heading_font: 'Source Sans Pro', sans-serif;  
$body_font: 'Raleway', sans-serif;  
$nav_font: 'Maven Pro', sans-serif;
```

```
$text_color: #181818;  
$attention_color: #ff500a;
```

```
body {  
    font-family: $body_font;  
    font-size: 14px;  
    color: $text_color;  
}
```

...

All examples are written in SASS

NESTING

```
.class {  
  div {  
    font-family: $nav_font;  
  }  
  a {  
    color: $attention_color;  
    text-decoration: none;  
  }  
  li {  
    margin-bottom: 10px;  
  }  
}
```

compiles into

```
.class div {  
  font-family: $nav_font;  
}  
.class a {  
  color: $attention_color;  
  text-decoration: none;  
}  
.class li {  
  margin-bottom: 10px;  
}
```

All examples are written in SASS

MIXINS

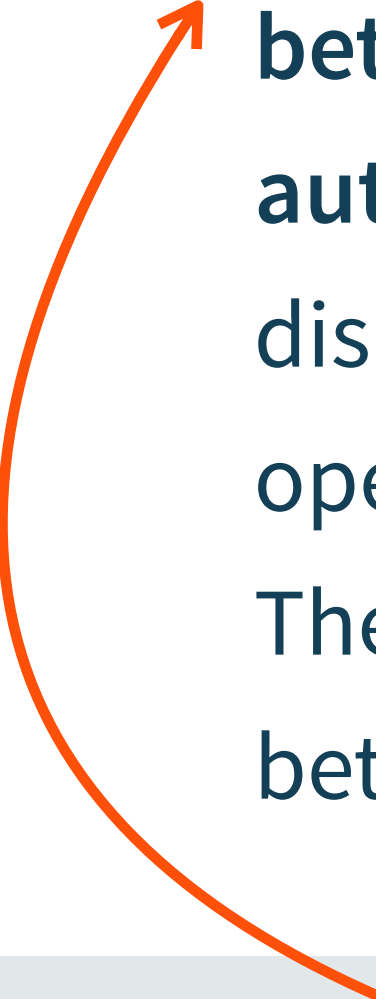
```
@mixin border-radius($radius) {  
  -webkit-border-radius: $radius;  
  -moz-border-radius: $radius;  
  -ms-border-radius: $radius;  
  border-radius: $radius;  
}  
  
.small-box { @include border-radius(5px); }  
.big-box { @include border-radius(10px); }
```

compiles into

```
.small-box {  
  -webkit-border-radius: 5px;  
  -moz-border-radius: 5px;  
  -ms-border-radius: 5px;  
  border-radius: 5px;  
}  
  
.big-box {  
  -webkit-border-radius: 10px;  
  -moz-border-radius: 10px;  
  -ms-border-radius: 10px;  
  border-radius: 10px;  
}
```

All examples are written in SASS

CSS LIMITATION?



“Having a constraint solver allows you to express relationships between arbitrary elements and have conflicts resolved **automagically**. However, things can get complex when elements disappear and new ones arrive, like they do through DOM operations. Circular dependencies must also be handled gracefully. Therefore, the idea of allowing CSS to express layout constraints between any elements were dropped at an early stage.”

can be implemented in Javascript

Håkon Wium Lie Interview

NEXT CLASS: JAVASCRIPT

courses.engr.illinois.edu/cs498rk1/