



# Web Development I

## Les 6: CSS Lay-out – Box Model

**HO  
GENT**

# Inhoud

---

- ▶ Weergave elementen
  - block <-> inline elementen
  - display en visibility property
- ▶ Boxmodel
  - ruimte ingenomen door een element
  - padding
  - margin
  - border
  - extra
- ▶ Extra opmaak
  - afgeronde hoeken – shadow (box-tekst)
  - cursor style

**Weergave elementen**  
**block <-> inline**

# Block <-> inline elements

- ▶ Wanneer we spreken over layout in CSS dan is het interessant om een onderscheid te maken tussen:
  - Block elementen (Meer op: [https://developer.mozilla.org/en/docs/Web/HTML/Block-level\\_elements](https://developer.mozilla.org/en/docs/Web/HTML/Block-level_elements))
    - Nemen in de breedte de maximale ruimte in die ze krijgen van hun parent element.
    - De standaard hoogte wordt bepaald door de inhoud.
    - Worden voorafgegaan en gevolgd door een overgang naar een nieuwe regel.
    - Voorbeelden: header, footer, nav, article, h1, p, div,...
  - Inline elementen
    - Zijn elementen die dezelfde regel kunnen delen met andere elementen en in de breedte enkel de ruimte innemen die ze nodig hebben om hun inhoud weer te geven.
    - Ze worden niet vooraf gegaan en gevolgd door de overgang naar een nieuwe regel.
    - Voorbeelden: a, img, em, strong, span, input, label,...
    - Meer op: [https://developer.mozilla.org/en-US/docs/Web/HTML/Inline\\_elements](https://developer.mozilla.org/en-US/docs/Web/HTML/Inline_elements)

# Block <-> inline elements

- ▶ Open **01\_weergaveElementen/01\_block-vs-inline.html**
- ▶ Standaard worden block-level elementen (bijv. h1, p, header, ...) verticaal onder elkaar geplaatst en inline-level elementen (bijv. strong, a, img, ...) horizontaal naast

**London**

**London** is the capital city of England. It is the most populous city in the United Kingdom, with a metropolitan area of over 13 million inhabitants.

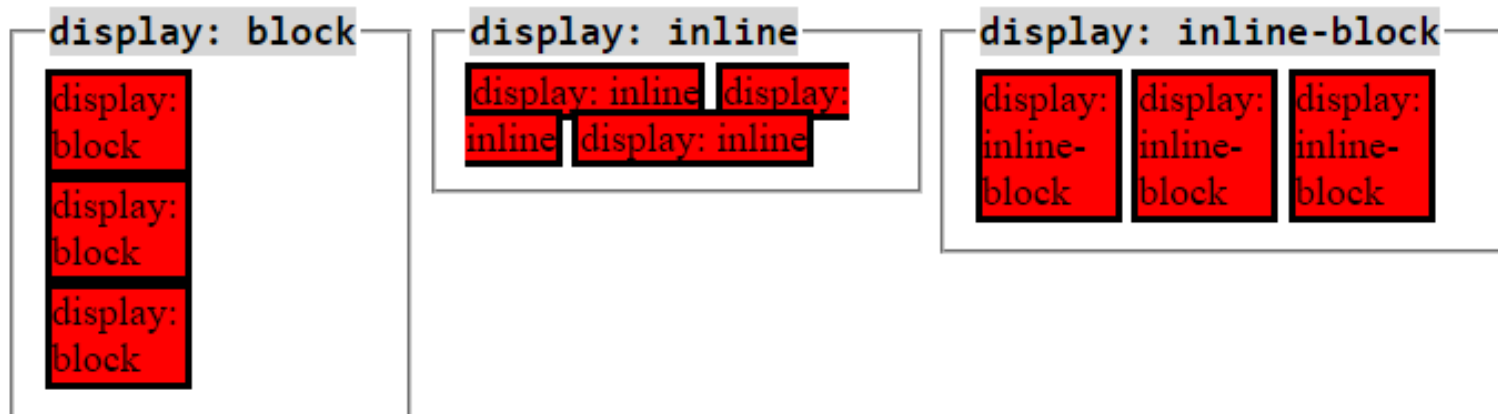
Standing on the River Thames, **London** has been a major settlement for two millennia, its history going back to its founding by the Romans, who named it Londinium.

```
<h1>London</h1>
<p><strong>London</strong> is the capital
city of England. It is the most populous
city in the United Kingdom, with a
metropolitan area of over 13 million
inhabitants.</p>
<p>Standing on the River Thames,
<strong>London</strong> has been a major
settlement for two millennia, its history
going back to its founding by the Romans,
who named it Londinium.</p>
```

```
h1, p {
  border: 2px solid green;
}
strong {
  background-color: yellow;
}
```

# CSS property: display

- ▶ De basismethode om de layout van een pagina te wijzigen is het wijzigen van de [display](#) property. Elk element heeft een default waarde voor de display property, maar die kan overschreven worden
- ▶ Enkele mogelijke waarden voor **display** zijn:
  - **inline**: element krijgt de eigenschappen van een inline element
  - **block**: element krijgt de eigenschappen van een block element
  - **inline-block**: behoudt de eigenschappen van een block element, maar wordt in de pagina (in de flow) weergegeven als een inline element.
  - **none**: element wordt niet weergegeven (neemt ook geen ruimte in)



# CSS property: display

▶ Open **01\_weergaveElementen/02\_display.html**

▶ Standaard geeft een **li** element

**display: list-item;**

dit zorgt voor een block weergave met een bijkomende marker box. Vroeger overschreef men de waarde **list-item** met **inline** of **inline-block** om de unordered list weer te geven als een horizontale navigatiebalk (zie voorbeeld hiernaast). Tegenwoordig wordt dit gedaan met Flexbox (zie volgende les).

▶ **display: none; /\* verbergt een item \*/**

```
li {  
    display: inline;  
    margin-right: 10px;  
}  
  
li.coming-soon {  
    display: none; /* verbergt een item */  
}
```

```
<ul>  
  <li>Home</li>  
  <li>Products</li>  
  <li class="coming-soon">Services</li>  
  <li>About</li>  
  <li>Contact</li>  
</ul>
```

Home Products About Contact



# CSS property: display

---

## ► Opmerkingen

- Het is niet de gewoonte om inline elementen om te switchen naar block elementen.
- inline-block elementen worden niet altijd rechtstreeks tegen elkaar weergegeven. Gewoonlijk bestaat er een kleine ruimte (een spatie) tussen twee inline-block elementen. Dit is het geval als er zich één of meerdere spatie(s) of andere witruimte tussen de twee inline-block elementen bevindt in de HTML code. Bijgevolg is het meestal meer aangewezen om de nieuwere layout-methoden **Flexbox** en **CSS grid** te gebruiken waar vroeger inline-block elementen gebruikt werden.



# CSS property: visibility

- ▶ Naast met `display: none;` kan je ook met de eigenschap [visibility](#) een element verbergen. De mogelijke waarden voor deze eigenschap zijn:
  - `visible` (default)
  - `hidden`: verbergt het element, maar behoudt de ruimte die het element inneemt in

```
<style>
  li {
    display: inline;
    margin-right: 10px;
  }
  li.coming-soon {
    visibility: hidden;
  }
</style>
```

Home Products

About Contact

# **Weergave elementen: CSS boxmodel**

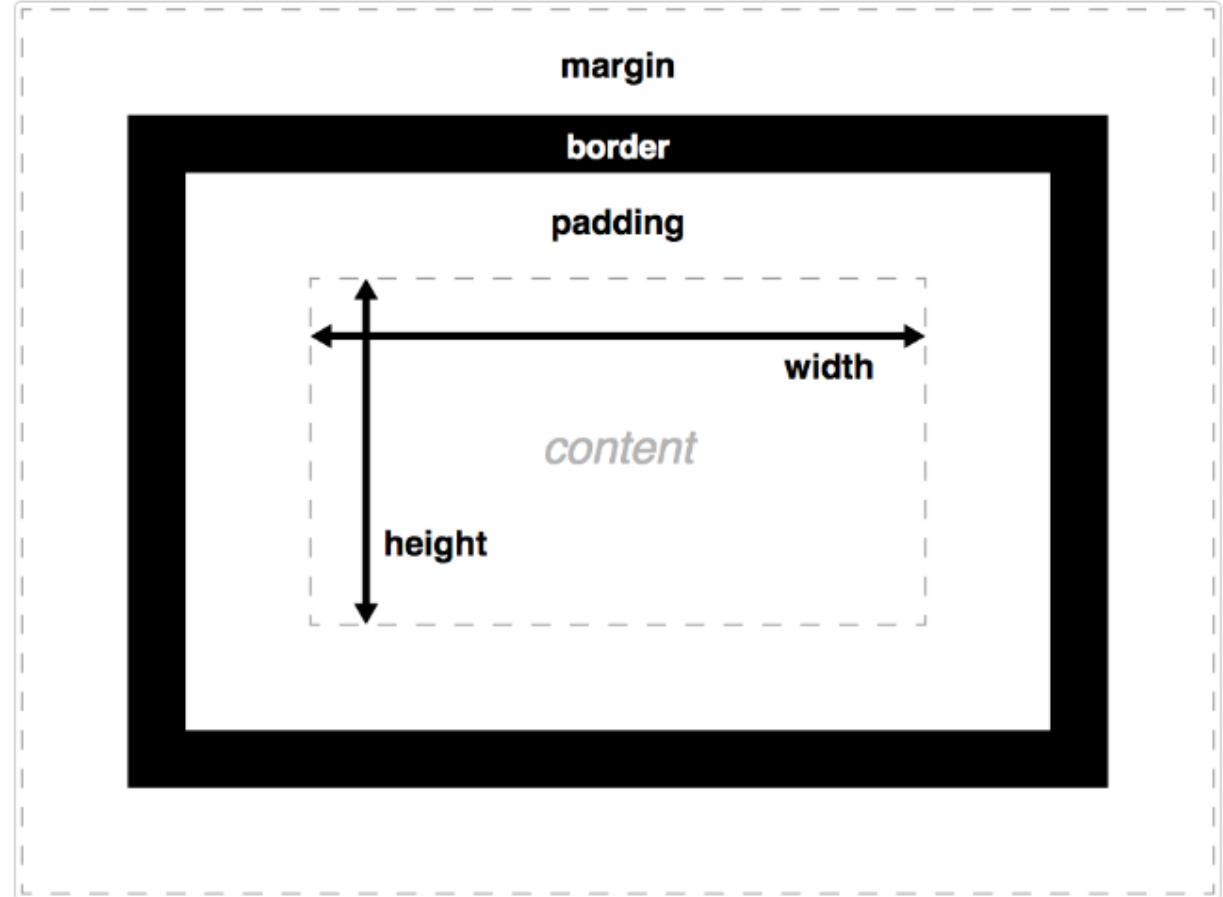
# CSS box

- ▶ Voor elk element zal de browser geen, één of meerdere ‘boxes’ genereren. Afhankelijk van de waarde van de `display` property.
- ▶ De opbouw van een box wordt beschreven in het [CSS boxmodel](#).



# CSS Box Model

- ▶ Elk element in een webpagina wordt gerepresenteerd door een box met de volgende onderdelen:
  - **Content**: bevat bijv. tekst en afbeeldingen.
  - **Padding**: ruimte rond de content, zorgt ervoor dat rand (border) niet direct aan de content vasthangt.
  - **Border**: een rand rond de padding
  - **Margin**: ruimte rond de border. Deze wordt niet beïnvloed door een achtergrondkleur van het element, is volledig transparant.



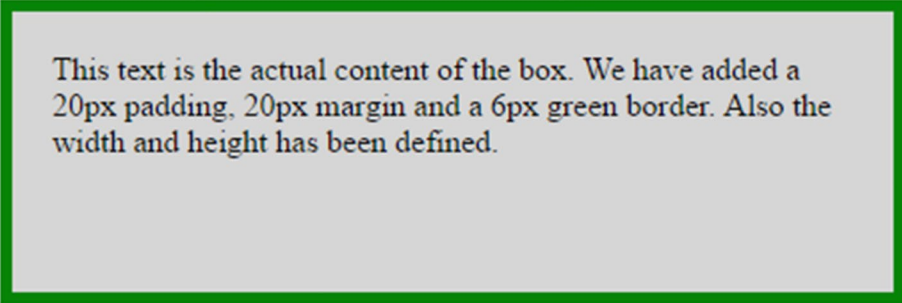
# CSS Box Model

## ▶ Open 01\_weergaveElementen/03\_box.html

```
<style>
  div {
    background-color: lightgray;
    border: 6px solid green;
    height: 100px;
    margin: 20px;
    padding: 20px;
    width: 400px;
  }
</style>
</head>
<body>
  <h2>Demonstrating the Box Model</h2>
  <p>The CSS box model is essentially a box that wraps around every
  HTML element. It consists of: borders, padding, margins, and the
  actual content.</p>
  <div>This text is the actual content of the box. We have added a 20px
  padding, 20px margin and a 6px green border. Also the width and
  height has been defined.</div>
</body>
</html>
```

## Demonstrating the Box Model

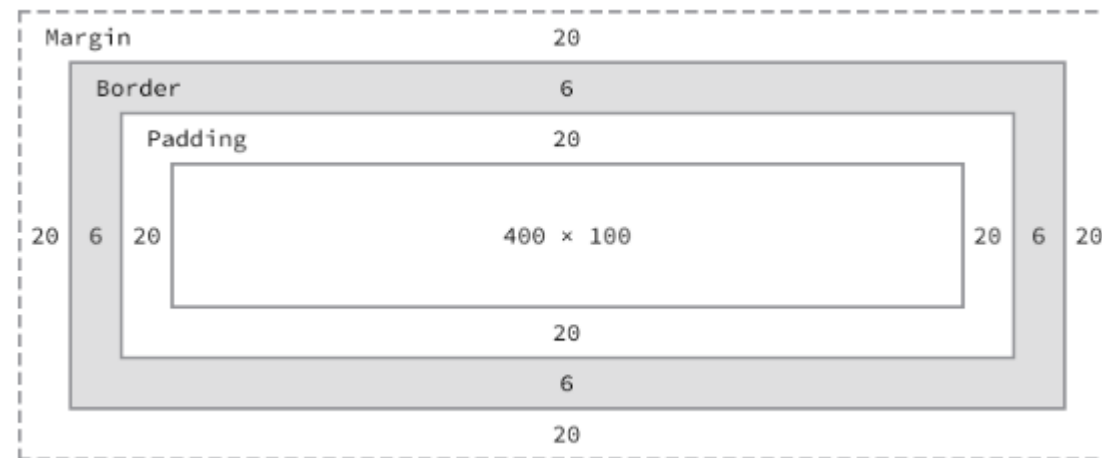
The CSS box model is essentially a box that wraps around every HTML element. It consists of: borders, padding, margins, and the actual content.



This text is the actual content of the box. We have added a 20px padding, 20px margin and a 6px green border. Also the width and height has been defined.

# CSS Box Model

- ▶ De totale ruimte die een blok inneemt in de layout
  - Total width = margin-right + border-right + padding-right + content width + padding-left + border-left + margin-left
    - Width:  $492\text{px} = 20\text{px} + 6\text{px} + 20\text{px} + 400\text{px} + 20\text{px} + 6\text{px} + 20\text{px}$
  - Total height = margin-top + border-top + padding-top + content height + padding-bottom + border-bottom + margin-bottom
    - Height:  $192\text{px} = 20\text{px} + 6\text{px} + 20\text{px} + 100\text{px} + 20\text{px} + 6\text{px} + 20\text{px}$



# CSS Box Model

- ▶ Als je twijfelt over de afmetingen van een element dan is het een zeer goed idee om via **Developer Tools > Elements** de afmetingen af te lezen

```
div { 03_box.html:7  
  box-sizing: content-box;  
  background-color: lightgray;  
  border: 6px solid green;  
  height: 100px;  
  margin: 20px;  
  padding: 20px;  
  width: 400px;  
}
```

---

```
div { user agent stylesheet  
  display: block;  
}
```

The diagram illustrates the CSS Box Model with the following dimensions:

- margin:** 20px (outermost, dashed orange border)
- border:** 6px (yellow border)
- padding:** 20px (green border)
- content:** 400x100px (blue box)

The total width of the element is 400px, and the total height is 100px. The content area is 400px wide and 100px high. The padding is 20px, the border is 6px, and the margin is 20px.



# Box: width en height

- ▶ De default breedte van een element wordt bepaald door de **display** property
  - **block** : de box neemt de volledige horizontale ruimte in voorzien door de parent.
  - **inline** en **inline-block** : neemt de breedte van de inhoud in.
- ▶ De default hoogte van een element wordt zowel voor block als inline elementen bepaald door de inhoud.
- ▶ Via de properties width en height kan je expliciet de breedte en hoogte instellen van een element. Standaard stellen deze properties de width en height in van de content area.
- ▶ **Je kan enkel een width en height instellen voor block en inline-block elementen! Je kan geen width/height instellen voor inline elementen. Deze nemen steeds de breedte en hoogte in van de inhoud!!**

# Box: width en height

---

- ▶ De value van de **width** en de **height** property kan onder andere uitgedrukt worden in:
  - CSS pixels (absolute lengte eenheid): **px**
  - percent - relatieve waarde tov van de parent container: **%**
  - **em** – relatieve waarde tov de grootte van het gebruikte lettertype. (1em = breedte van de M in de gekozen font).
  - **rem** - relatieve waarde tov de grootte van het gebruikte lettertype van het root element.

# Box: width en height

- ▶ Stel de hoogte bij voorkeur niet in!
  - Als de inhoud verandert, moet de hoogte van het element eventueel mee aangepast worden.
  - De gebruiker kan de tekstgrootte veranderen, waardoor er problemen ontstaan.
  - Bij afbeeldingen kan je wel de hoogte instellen.



# Box: margin en padding

- ▶ Door een waarde aan margin en padding te geven zal de pagina er luchtiger uitzien.



# Box: margin

---

- ▶ Ruimte aan de buitenkant van een element. Gebruiken voor creatie **witruimte** tussen elementen.
- ▶ Een marge is altijd **transparant**. We zien de achtergrondkleur van het parent element.

# Box: padding

---

- ▶ Witruimte tussen de inhoud en de rand.
- ▶ We zien de achtergrondkleur van het element waarop de padding wordt toegepast. Indien geen achtergrondkleur wordt toegepast zien we de achtergrondkleur van het parent element.

# Box: margin & padding declaraties

- ▶ CSS heeft doorgaans hetzelfde effect voor inline en block elementen. Dit geldt voor de properties font, color, background, border, ...
- ▶ Voor margin en padding geldt dit echter niet!
  - Je kan ruimte links en rechts toevoegen aan een inline element via margin-left/right, padding-left/right.
  - Maar margin-top/bottom werkt niet bij een inline element. Padding-top/bottom heeft wel een effect, maar niet op de 'normal flow'.

"Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor **in reprehenderit in voluptate velit** esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum."

"Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor **in reprehenderit in voluptate velit** esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum."

```
strong{  
    margin: 20px;  
}
```

```
strong{  
    margin: 20px;  
    padding: 20px;  
}
```



# Box: margin & padding declaraties

## ► padding en margin instellen

- Elke waarde afzonderlijk

```
p{  
    padding-top: 10px;  
    padding-right: 25px;  
    padding-bottom: 10px;  
    padding-left: 5px;  
}
```

- Verkorte notatie (shorthand notation)  
**(Top Right Bottom Left)**

```
p{  
    padding: 10px 25px 10px 5px;  
}
```

- Verkorte notatie wanneer top en bottom enerzijds, en right en left anderzijds dezelfde waarden moeten krijgen

```
p{  
    /* top en bottom krijgen waarde 0, left en right krijgen waarde 2em */  
    padding: 0 2em;  
}
```

# Horizontaal centreren

- ▶ **BELANGRIJK:** een block element kan men horizontaal centreren door de linker- en rechtermarge in te stellen op `auto`. In de volgende lessen zullen we zien dat dit ook kan met grid en flexbox.

```
div {  
  width: 80%;  
  /* top en bottom krijgen waarde 0, left en right krijgen de waarde auto.  
  Hierdoor wordt het div-element gecentreerd binnen het ouderelement */  
  margin: 0 auto;  
}
```

Vergeet niet om ook een width in te stellen!

- ▶ **Verticaal centreren**
  - Is niet mogelijk met `margin-top: auto` en `margin-bottom: auto`
  - Oplossingen: zie later bij Grid en Flexbox.

# Box: padding – margin declaraties

- ▶ Wanneer je gebruikmaakt van percentages, wordt de padding en margin berekend op basis van de **breedte** van het omvattende (parent) element.

- ▶ Bijvoorbeeld

- Het scherm is 760px breed => margin-top is 76px (top en bottom percentages worden berekend op de width, niet op de height!) padding-left is 152px

```
.html
<body>
  <p>...</p>
</body>
```

```
.css
p{
  margin-top: 10%;
  padding-left: 20%;
}
```

- margin-top is 85px padding-left is 170px

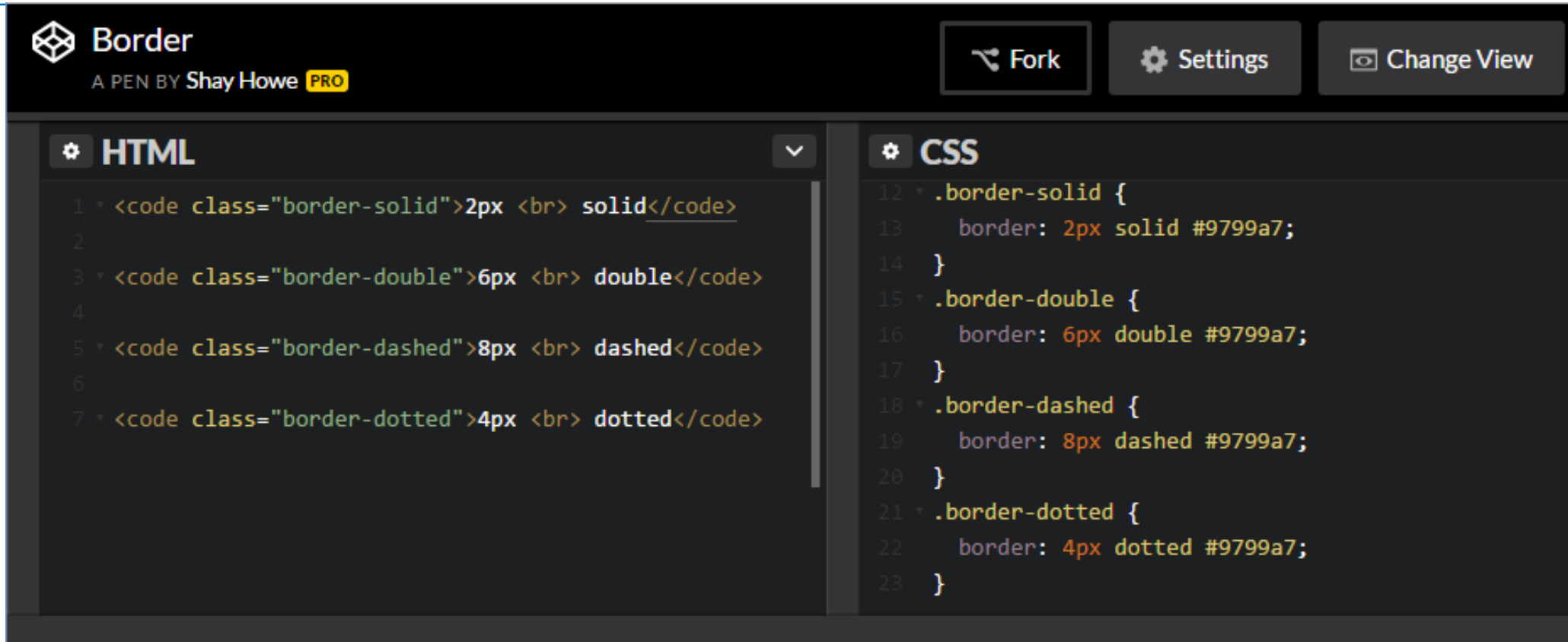
```
.html
<body>
  <div id="wrapper">
    <p>...</p>
  </div>
</body>
```

```
.CSS
#wrapper{
  width: 850px;
}
p{
  margin-top: 10%;
  padding-left: 20%;
}
```

# Box: border

- ▶ Van een border kunnen we 3 waarden instellen : de breedte, kleur en stijl
  - `border-width` (px, %, em, rem, thin, medium, thick)
  - `border-style` (none/hidden/dotted/dashed/solid/double/...)
  - `border-color` (een kleurwaarde of transparent)
- ▶ Men kan ook de drie eigenschappen combineren in een shorthand:
  - `border: 1px solid #000;` (volgorde van de waarden is niet belangrijk): dit is een zwarte solid border van 1px.
- ▶ Bij een transparante rand zie je de achtergrondkleur van het element

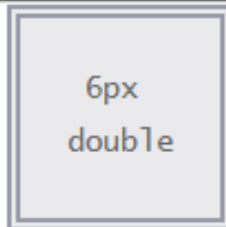
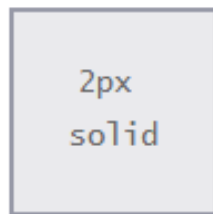
# Box: border - voorbeelden



The screenshot shows a code editor interface for a project named "Border" by Shay Howe. It features two panels: HTML and CSS. The HTML panel contains four lines of code defining different border styles. The CSS panel contains the corresponding CSS rules for each style, all using a #9799a7 color.

```
HTML
1 <code class="border-solid">2px <br> solid</code>
2
3 <code class="border-double">6px <br> double</code>
4
5 <code class="border-dashed">8px <br> dashed</code>
6
7 <code class="border-dotted">4px <br> dotted</code>

CSS
12 .border-solid {
13   border: 2px solid #9799a7;
14 }
15 .border-double {
16   border: 6px double #9799a7;
17 }
18 .border-dashed {
19   border: 8px dashed #9799a7;
20 }
21 .border-dotted {
22   border: 4px dotted #9799a7;
23 }
```



# Box: border declaraties

## ▶ border

- Je kan ook elke rand afzonderlijk instellen(border-top, border-bottom, border-right, border-left)

```
<html>
  <head>...</head>
  <body>
    <p>
      Lorem ipsum dolor
      sit amet, ...
    </p>
  </body>
</html>
```

```
p{
  border-top: thick dashed black;
}
```

- Ook de randdikte, randkleur, randstijl kan afzonderlijk worden ingesteld

```
p{
  border-top-width: 10px;
  border-top-color: red;
  border-top-style: dashed;
}
```

# Box: collapsing margins

---

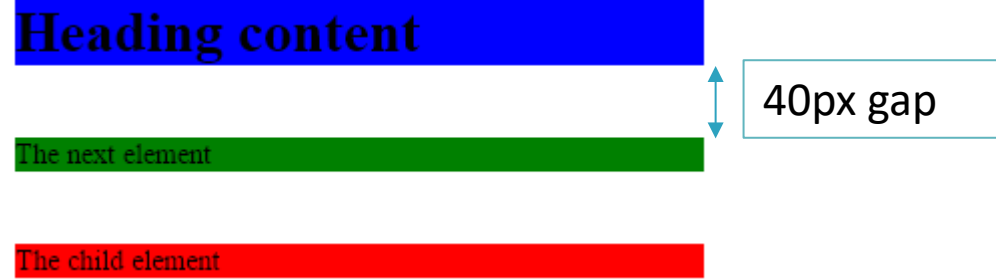
- ▶ Open **01\_weergaveElementen/04\_collapsing\_margins.html**
- ▶ De browser geeft 'boxes' niet altijd weer zoals verwacht.  
Wanneer de bottom margin van 1 element de top margin van een ander element raakt, is het resultaat NIET de som, maar de grootste margin.
- ▶ Meer informatie over dit probleem: <https://css-tricks.com/what-you-should-know-about-collapsing-margins/>



# Box: collapsing margins

- ▶ Collapsing Margins tussen opeenvolgende elementen (verticaal)
  - voor aangrenzende verticale blokelementen in de normale flow, wordt de grootste marge behouden. De kleinste marge zal dichtklappen (wordt 0)

```
h1 {  
  margin-bottom: 25px;  
  background-color: blue;  
}  
  
div {  
  margin-top: 40px;  
  margin-bottom: 25px;  
  background: green;  
  /*border: 1px solid black;*/  
}  
  
p {  
  margin-top: 20px;  
  background-color: red;  
}
```



```
<h1>Heading content</h1>  
<div>The next element</div>  
<div>  
  <p>The child element</p>  
</div>
```

# Box: collapsing margins

- ▶ Collapsing Margins tussen parent en first/last child
  - Enkel indien geen padding, borders tussen parent en child, dus enkel aaneengrenzende marges => marges worden samengevoegd en grootste wordt toegepast

```
h1 {  
  margin-bottom: 25px;  
  background-color: blue;  
}  
  
div {  
  margin-top: 40px;  
  margin-bottom: 25px;  
  background: green;  
  /*border: 1px solid black;*/  
}  
  
p {  
  margin-top: 20px;  
  background-color: red;  
}
```

Heading content

The next element

The child element

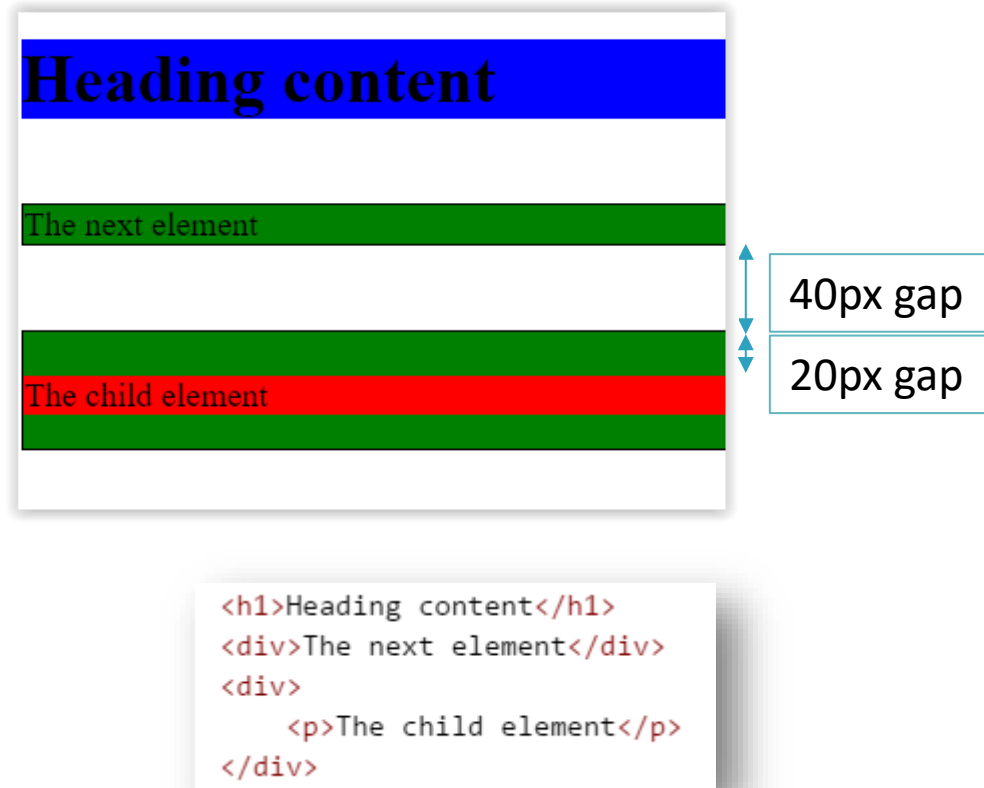
40px gap

```
<h1>Heading content</h1>  
<div>The next element</div>  
<div>  
  <p>The child element</p>  
</div>
```

# Box: collapsing margins

- ▶ Oplossing: iets toevoegen aan de box (padding of border)
  - Ofwel voeg je padding toe zo is er geen overlapping meer.
  - Ofwel voeg je een border toe.

```
h1 {  
  margin-bottom: 25px;  
  background-color: blue;  
}  
  
div {  
  margin-top: 40px;  
  margin-bottom: 25px;  
  background: green;  
  /*border: 1px solid black;*/  
}  
  
p {  
  margin-top: 20px;  
  background-color: red;  
}
```



# Box: negative margins

- ▶ Je kan ook een negatieve waarde als margin instellen.
- ▶ Een negatieve waarde vermindert de afstand. Deze eigenschap is bruikbaar om overlappende blokken te maken.
- ▶ Bijvoorbeeld  
De zin 'RAISE TUNA ...' heeft een margin-top van -10px






# Box: begrensd width en height

- ▶ Open **01\_weergaveElementen/05\_min-width-max-width.html**
- ▶ Het is ook mogelijk om een minimale en maximale breedte in te stellen. Dit zorgt ervoor dat, bij wijziging van de breedte van het browservenster, er steeds een minimale en/of maximale breedte gegarandeerd is. Door het instellen van een maximale breedte kan je ervoor zorgen dat de tekstregels niet te lang worden. Dit verbetert de leesbaarheid.
- ▶ Hiervoor worden de CSS properties [min-width](#) – [max-width](#) – [min-height](#) – [max-height](#) gebruikt.

# min-width en max-width

← → ↻ 127.0.0.1:62120/Voorbeelden%20Les/chapter-13/min-width-max-width.html

Photo	Description	Price
	The Rhodes piano is an electro-mechanical piano, invented by Harold Rhodes during the fifties and later manufactured in a number of models, first in collaboration with Fender and after 1965 by CBS. It employs a piano-like keyboard with hammers that hit small metal tines, amplified by electromagnetic pickups.	\$1400
	The Wurlitzer electric piano is an electro-mechanical piano, created by the Rudolph Wurlitzer Company of Mississippi. The Wurlitzer company itself never called the instrument an "electric piano", instead inventing the phrase "Electronic Piano" and using this as a trademark throughout the production of the instrument. It employs a piano-like keyboard with hammers that hit small metal tines, amplified by electromagnetic pickups.	\$1600
	A Clavinet is an electronically amplified clavichord manufactured by the Hohner company. Each key uses a rubber tip to perform a hammer on a string. Its distinctive bright staccato sound is often compared to that of an electric guitar. Various models were produced over the years, including the models I, II, L, C, D6, and E7.	\$1200

```
td.description {  
  min-width: 450px;  
  max-width: 650px;  
  text-align: left;  
  padding: 5px;  
  margin: 0px;}
```

# min-height en max-height / overflow

- ▶ Open **01\_weergaveElementen/06\_min-height-max-height.html**
- ▶ Op dezelfde manier als de breedte, kan men ook een minimale en maximale hoogte instellen.
- ▶ Soms is er het probleem dat de content niet meer past binnen de box.
- ▶ Dit kan men oplossen met de eigenschap [overflow](#). Deze kan meerdere waarden hebben:
  - **visible** (default value): zal de tekst buiten de box weergeven, niet vaak gewenst.
  - **hidden**: zal de tekst, die niet meer binnen de box past verbergen.
  - **scroll**: zal altijd schuifbalken weergeven in de box.
  - **auto**: zal schuifbalken weergeven als het nodig is.
- ▶ Indien er enkel een horizontale of verticale schuifbalk gewenst is kan men ook gebruik maken van de eigenschappen: `overflow-x` (horizontaal) en `overflow-y`(verticaal).



# min-height en max-height / overflow

## Fender Stratocaster

The Fender Stratocaster or "Strat" is one of the most popular electric guitars of all time, and its design has been copied by many guitar makers. It was designed by Leo Fender, George Fullerton and Freddie Tavares in 1954.

## Gibson Les Paul

The Gibson Les Paul is a solid body electric guitar that was first sold in 1952. The Les Paul was designed by Ted McCarty in collaboration with popular guitarist Les Paul, whom Gibson enlisted to endorse the new model. It is one of the most well-known electric guitar types in the world.

```
p {  
  min-height: 30px;  
  max-height: 85px;  
}
```

## Fender Stratocaster

The Fender Stratocaster or "Strat" is one of the most popular electric guitars of all time, and its design has been copied by many guitar makers. It was designed by Leo Fender, George Fullerton and Freddie Tavares in 1954.

## Gibson Les Paul

The Gibson Les Paul is a solid body electric guitar that was first sold in 1952. The Les Paul was designed by Ted McCarty in collaboration with popular guitarist Les Paul, whom Gibson enlisted to endorse the new model. It is one of the most well-known electric guitar types in the world.

```
p.one {  
  overflow: hidden;  
}  
  
p.two {  
  overflow: auto;  
}
```

## Fender Stratocaster

The Fender Stratocaster or "Strat" is one of the most popular electric guitars of all time, and its design has been copied by many guitar

## Gibson Les Paul

The Gibson Les Paul is a solid body electric guitar that was first sold in 1952. The Les Paul was designed by Ted McCarty



# Rekenen met de calc() CSS function

- ▶ Met de [calc\(\)](#)-functie kan je berekeningen uitvoeren in een declaratie:

```
width: calc(100% - 30px);
```

- ▶ Je kan de standaardoperatoren +, -, \* en / gebruiken, alsook ronde haakjes en je kan verschillende eenheden combineren.

- ▶ **Let op** de + en de - operator moeten voorafgegaan en gevolgd worden door [‘whitespace’](#).

- ▶ Je kan de [calc\(\)](#)-functie op verschillende plaatsen gebruiken, bijvoorbeeld ook bij het creëren van gradients:

```
background-image: linear-gradient(to right, silver,  
white 50px,  
white calc(100% - 50px),  
silver);
```

zie [01\\_weergaveElementen/07\\_calc-box-sizing.html](#)

# CSS property: box-sizing

- ▶ Je kan met behulp van de property [box-sizing](#) aanpassen hoe browsers omgaan met de properties width en height.
  - **box-sizing** (default waarde): `content-box`;  
De ingestelde width en height hebben enkel betrekking op de content.
  - **box-sizing**: `border-box`;  
De ingestelde width en height hebben betrekking op de content + de padding + de border.

Meer informatie: <https://developer.mozilla.org/en-US/docs/Web/CSS/box-sizing>

# Box-sizing: border-box

- ▶ Het is een best practice om steeds voor alle elementen **box-sizing** in te stellen op **border-box**. Dit kan bijvoorbeeld als volgt:

```
html {  
  box-sizing: border-box;  
}  
  
*, ::before, ::after {  
  box-sizing: inherit;  
}
```

# Box-sizing example

- ▶ Open **07\_calc-box-sizing.html** en creëer onderaan de pagina een paragraaf met **Lorem ipsum** tekst. Geef de paragraaf een donkergrijze rand van 7 pixels breed en 10 pixels padding. Zorg ervoor dat de image en de paragraaf even breed zijn.

## Box model examples

### Calc() function

The gradient used on this page uses the `calc()` function.

### Box-sizing

Lorem ipsum dolor sit amet consectetur, adipisicing elit. Nihil at vel pariatur a ipsa placeat quidem tempora numquam magni minus voluptatem, voluptates quis libero laborum aliquid eaque iusto officia enim?

Below you see a Lorem ipsum image



Create a Lorem Ipsum paragraph below with a 7px border that matches the width of the image:

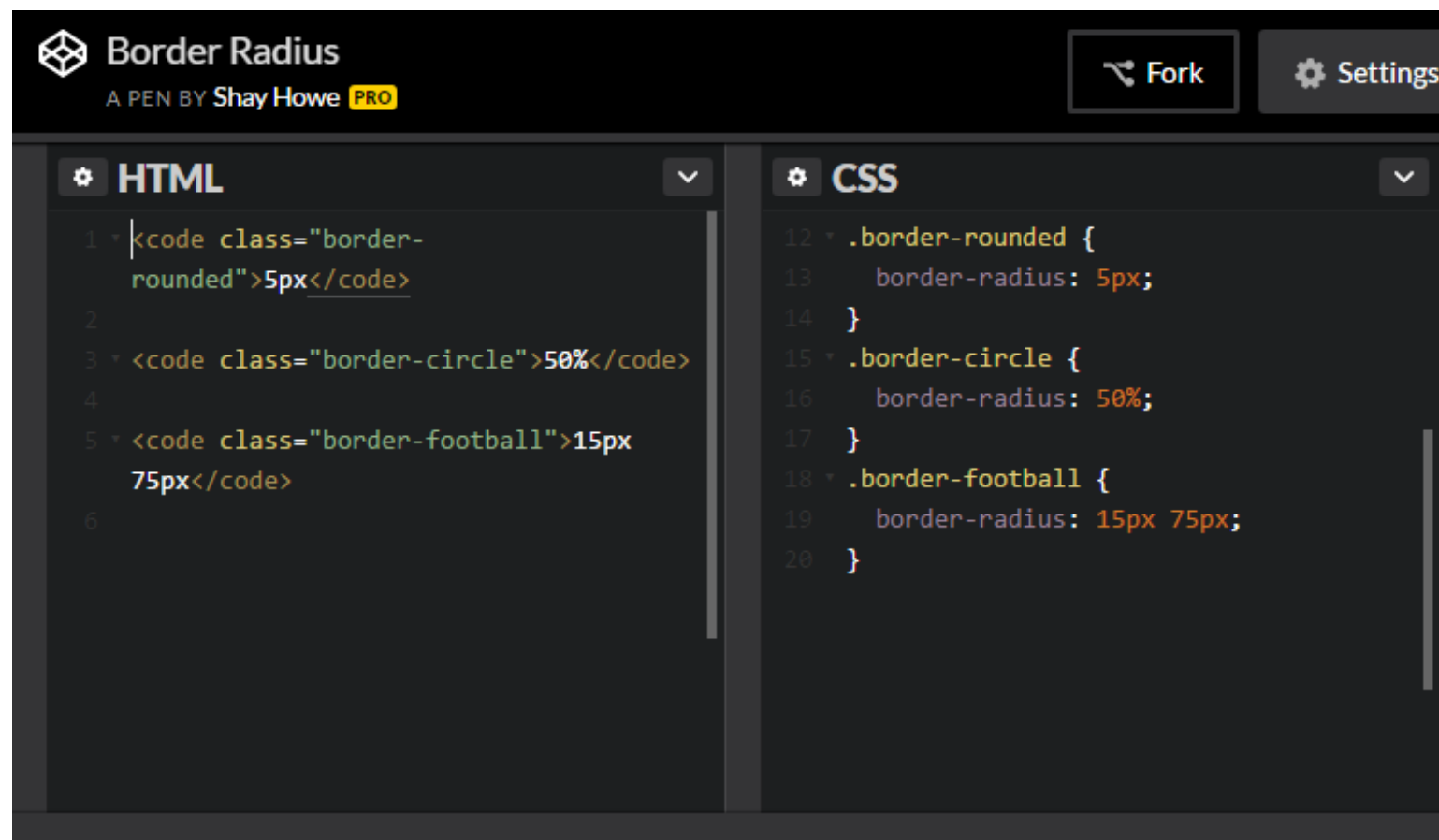
Lorem ipsum dolor sit amet consectetur adipisicing elit. Similique tempora recusandae cum esse, aperiam natus reprehenderit cumque omnis doloribus dolore quod in reiciendis quo, obcaecati facilis. Dolorum aspernatur inventore eligendi?



**Afgeronde hoeken**  
**Shadow: box & text**

# CSS property: border-radius

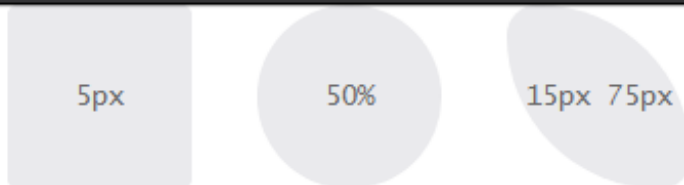
- ▶ [border-radius](#): hiermee kan je de randen van een element afronden



The screenshot shows a code editor titled "Border Radius" by Shay Howe. It is divided into two panels: HTML and CSS. The HTML panel contains three code snippets: a rounded square with a 5px border radius, a circle with a 50% border radius, and a rounded rectangle with 15px and 75px border radii. The CSS panel contains three corresponding CSS rules: a class for rounded corners (5px), a class for a circle (50%), and a class for rounded corners (15px 75px).

```
HTML
1 <code class="border-
  rounded">5px</code>
2
3 <code class="border-circle">50%</code>
4
5 <code class="border-football">15px
  75px</code>
6

CSS
12 .border-rounded {
13   border-radius: 5px;
14 }
15 .border-circle {
16   border-radius: 50%;
17 }
18 .border-football {
19   border-radius: 15px 75px;
20 }
```



# CSS property: border-radius

## ▶ Open 02\_extraOpmaak/border.html

```
p {  
  border: 5px solid #ee3e80;  
  padding: 20px;  
  width: 275px;  
  border-radius: 10px;  
}
```

Pet Sounds featured a number of unconventional instruments such as bicycle bells, buzzing organs, harpsichords, flutes, Electro-Theremin, dog whistles, trains, Hawaiian-sounding string instruments, Coca-Cola cans and barking dogs.

## ▶ Longhand properties

- `border-top-left-radius`
- `border-top-right-radius`
- `border-bottom-right-radius`
- `border-bottom-left-radius`

## ▶ Shorthand property

- `border-radius`  
(voor meer info zie [border-radius](#))

# CSS property: box-shadow

- ▶ Open **02\_extraOpmaak/02\_boxshadow.html**
- ▶ [box-shadow](#): hiermee kan je een schaduweffect toevoegen aan een box
- ▶ Minstens horizontal en vertical offset opgeven
  - Positieve waarde: rechts of onder de box
  - Negatieve waarde: links of boven de box
- ▶ Blur distance
  - vervaging
- ▶ Spread of distance
  - Positieve waarden zorgen ervoor dat de schaduw uitbreidt, groter wordt.
  - Negatieve waarden zorgen ervoor dat de schaduw krimpt, kleiner wordt.
- ▶ Kleur van de schaduw



# CSS property: box-shadow

## ▶ box-shadow

- Horizontale offset
- Verticale offset
- Blur afstand
- Spread van schaduw
- Kleur

```
p.one {  
    box-shadow: -5px -5px #777777;  
}  
  
p.two {  
    box-shadow: 5px 5px 5px #777777;  
}  
  
p.three {  
    box-shadow: 5px 5px 5px 5px #777777;  
}  
  
p.four {  
    box-shadow: 0 0 10px #777777;  
}  
  
p.five {  
    box-shadow: inset 0 0 10px #777777;  
}
```

# CSS property: box-shadow



# CSS property: text-shadow

- ▶ text-shadow
  - Horizontale offset
  - Verticale offset
  - Blur afstand (optioneel)
  - Kleur

The briard is known as a heart wrapped in fur.

The briard is known as a heart wrapped in fur.

The briard is known as a heart wrapped in fur.

The briard is known as a heart wrapped in fur.

The briard is known as a heart wrapped in fur.

```
p {
  font-size: 200%;
  padding: 20px;
  text-align: center;}
p.one {
  background-color: #eeeeee;
  color: #666666;
  text-shadow: 1px 1px 0px #000000;}
p.two {
  background-color: #dddddd;
  color: #666666;
  text-shadow: 1px 1px 3px #666666;}
p.three {
  background-color: #cccccc;
  color: #ffffff;
  text-shadow: 2px 2px 7px #111111;}
p.four {
  background-color: #bbbbbb;
  color: #cccccc;
  text-shadow: -1px -2px #666666;}
p.five {
  background-color: #aaaaaa;
  color: #ffffff;
  text-shadow: -1px -1px #666666;}
```



**Writing mode**  
**Logical properties**

# Writing modes

- ▶ In het begin ondersteunde CSS vnl. talen die horizontaal geschreven worden en van links-naar-rechts, zoals de Engelse taal. Maar nu ondersteunt CSS ook talen die verticaal geschreven worden (bijv. Japans) of van rechts-naar-links (bijv. Arabisch).
- ▶ De schrijfmodus wordt o.a. bepaald door de `writing-mode` en de `direction` CSS properties.
- ▶ Wanneer je de schrijfmodus instelt voor een volledig document, dan moet je deze instellen op het `html` element.

# CSS property: writing-mode

- ▶ Open **03\_schrijf-modi/01\_writing-modes.html**
- ▶ Deze property specificeert hoe block-level container gestapeld worden.

```
writing-mode: horizontal-tb;
```

(horizontal top-to-bottom)

```
writing-mode: vertical-lr;
```

(vertical left-to-right)

```
writing-mode: vertical-rl;
```

(vertical right-to-left)

Lorem ipsum dolor sit amet consectetur, adipisicing elit. Dolorem dignissimos, excepturi eius eos.

Exercitationem doloremque quas accusamus ipsam assumenda animi fugit sapiente nam praesentium repellendus rem ut cum et dicta

>Lorem ipsum dolor  
sit amet  
consectetur,  
adipisicing elit.  
Dolorem  
dignissimos,  
excepturi eius eos.  
Dolorem  
exercitationem  
doloremque quas  
accusamus ipsam  
assumenda animi  
fugit sapiente nam  
praesentium  
repellendus rem ut  
cum et dicta

>Lorem ipsum dolor  
sit amet  
consectetur,  
adipisicing elit.  
Dolorem  
dignissimos,  
excepturi eius eos.  
Exercitationem  
doloremque quas  
accusamus ipsam  
assumenda animi  
fugit sapiente nam  
praesentium  
repellendus rem ut  
cum et dicta

# Graphical design

- ▶ Open `03_schrijf-modi/02_graphic-design.html`
- ▶ De property `writing-mode` is niet alleen interessant als je werkt met een taal die verticaal geschreven wordt, maar verticale text wordt ook gebruikt in grafisch ontwerp.

```
h1 {  
  writing-mode: vertical-rl;  
  background-color: black;  
  color: white;  
  padding: 10px;  
  font-family: sans-serif;  
}
```

```
<h1>Play with writing modes</h1>
```

Play with writing modes

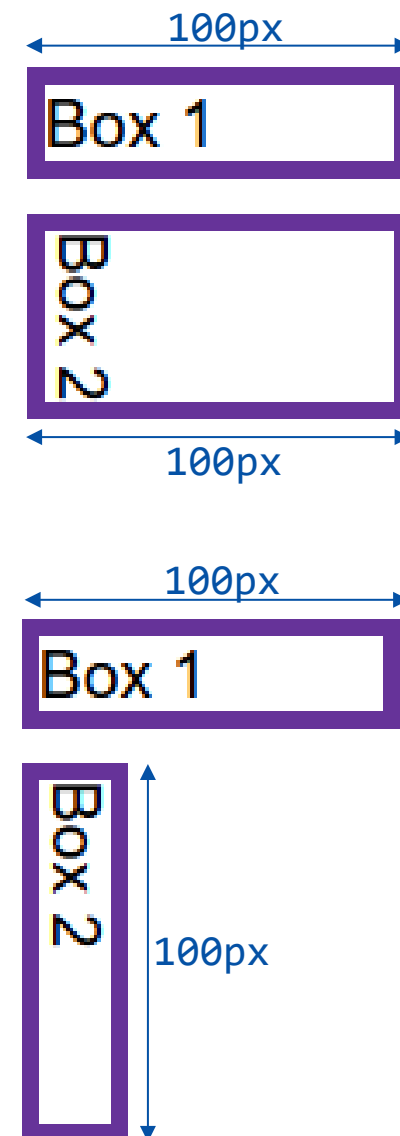
# CSS property: direction

- ▶ Naast de CSS property `writing-mode` is er ook een CSS property om de tekstrichting te wijzigen, namelijk `direction`. Bovendien kan de tekstrichting ook gewijzigd worden met het HTML `dir`-attribuut, wat trouwens de voorkeur geniet (BRON: <https://developer.mozilla.org/en-US/docs/Web/CSS/direction>).
- ▶ Het wijzigen van de tekstrichting is nodig bij talen die van rechts-naar-links geschreven worden zoals het Arabisch. Het is weinig waarschijnlijk dat je dit zal gebruiken op een creatieve manier in een grafisch ontwerp. Maar het is belangrijk om te begrijpen dat het web er niet alleen is voor talen die van links naar rechts worden weergegeven. Maar in dit opleidingsonderdeel zullen we niet verder ingaan op het werken met talen zoals bijv. het Arabisch of Japans.



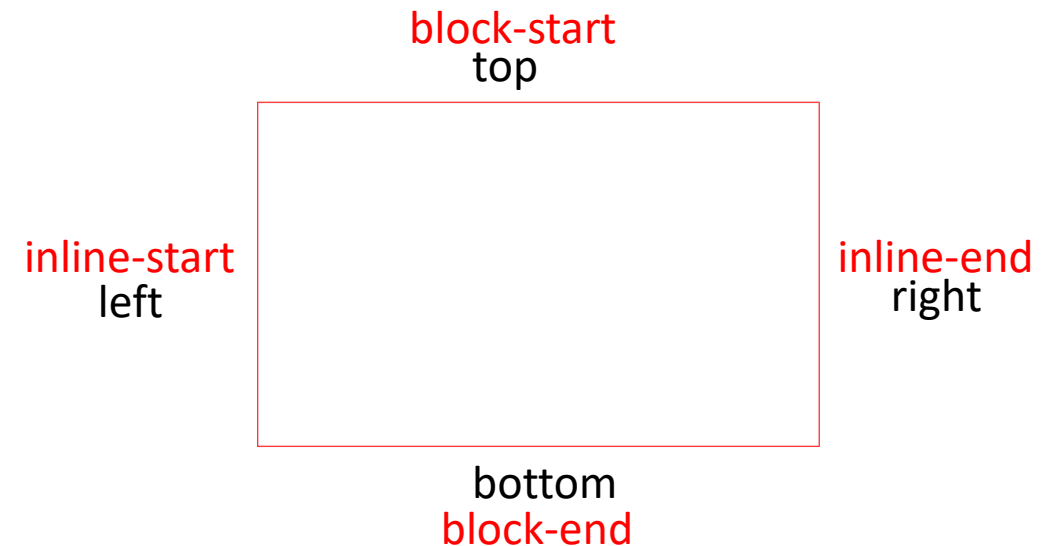
# Logical properties and values

- ▶ Open **03\_schrijf-modi/03-logical-properties.html**
- ▶ Van zodra we in een andere schrijfmodus dan `horizontal-tb` werken voelt het werken met 'Physical properties', zoals `width`, `height`, `left`, `right`, ... raar aan. Als je een box (zie **Box 1**) een `width` van `100px` geeft in `horizontal-tb` dan bepaalt dit de grootte in de inline richting, maar in `vertical-lr` bepaalt dit de grootte in de blokrichting (zie **Box 2**).
- ▶ Daarom dat er in CSS nieuwe 'Logical properties' gedefinieerd zijn voor `width` en `height` namelijk `block-size` en `inline-size`. Als je een box een `inline-size` geeft van `100px` dan speelt het geen rol of de schrijfrichting horizontaal of verticaal is. `inline-size` zal altijd de grootte in de inline richting aangeven.



# Logical properties and values

- ▶ In het begin van deze les hebben we geleerd over het CSS box model en CSS border. Bij het instellen van de margin, border en padding hebben we veel ‘physical properties’ gebruikt. Zoals `margin-top`, `padding-left` en `border-bottom`. Ook voor deze properties zijn er in CSS logical properties gedefinieerd. Zo is `padding-inline-start` de logical property voor `padding-left`. `padding-inline-start` zal altijd de padding zijn vooraan in de zin.



# Opmerking: enkele physical/logical properties

Physical property	Logical property
width	inline-size
height	block-size
margin-top	margin-block-start
margin-bottom	margin-block-end
margin-top margin-bottom	margin-block
margin-left	margin-inline-start
margin-right	margin-inline-end
margin-left margin-right	margin-inline

Physical property	Logical property
text-align: left	text-align: start
text-align: right	text-align: end
border-top	border-block-start
border-bottom	border-block-end
border-top border-bottom	border-block
border-left	border-inline-start
border-right	border-inline-end
border-left border-right	border-inline

[https://developer.mozilla.org/en-US/docs/Web/CSS/CSS logical properties and values/Margins borders padding](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_logical_properties_and_values/Margins_borders_padding)

# Opmerking: shorthand properties

- ▶ Open **03\_schrijf-modi/04-margin-shorthand.html**
- ▶ Momenteel zijn er geen shorthand properties die werken met de logical properties. Bijvoorbeeld de shorthand `margin: 10px 20px 30px 40px;` is steeds de afkorting voor

```
margin-top: 10px;  
margin-right: 20;  
margin-bottom: 30px;  
margin-left: 40px;
```

# Opmerking: physical properties/logical properties

- ▶ Moet ik nu de 'Physical properties' of de 'Logical properties' gebruiken?
  - Als je niet werkt met verschillende schrijfmodi dan kan je eventueel de voorkeur geven aan de 'Physical properties'.
  - Probeer echter binnen een website consequent te zijn. Schrijf niet de ene keer **width** en de andere keer **inline-size**.
  - Indien je moet werken aan een bestaande website waar overal width en height gebruikt wordt, dan is het wellicht aangewezen om verder width en height te blijven gebruiken in die website.

# **Cursor styles**

# CSS property: cursor

## ► cursor

- auto
- crosshair
- default
- pointer
- move
- text
- wait
- help
- url("cursor.gif")

I auto	↕ move	👉🚫 no-drop	↕ col-resize
👉🔄 all-scroll	👉 pointer	🚫 not-allowed	↕ row-resize
+ crosshair	👉🕒 progress	↔ e-resize	↗ ne-resize
👉 default	I text	↑ n-resize	↖ nw-resize
👉? help	↕ vertical-text	↓ s-resize	↘ se-resize
I inherit	🕒 wait	↔ w-resize	↙ sw-resize