



Web Development I

Les 8: Lay-out – Grid, Float, Position

**HO
GENT**

Inhoud

- ▶ Basisconcepten – CSS Grid
- ▶ Inleidend voorbeeld CSS Grid
- ▶ CSS Grid
 1. de *grid container* (*display: grid*)
 2. definiëren van de *grid* (*grid-template-columns, ...*)
 3. positioneren van *grid items* op de *grid* (*grid-row, grid-column, ...*)
 - *Grid-template-areas* property
 - Box alignment in CSS Grid Layout (*justify-self, align-self, ...*)
- ▶ Float
- ▶ Position



Basisconcepten – CSS Grid

CSS Grid Layout

- ▶ CSS Grid Layout is een twee dimensionaal lay-out model voor CSS. Het heeft veel mogelijkheden voor de **positionering** van boxes en hun inhoud, alsook voor het controleren van de afmetingen (**sizing**) van de boxes.

Flexible Box Layout vs Grid Layout

- ▶ In tegenstelling tot 'Flexible Box layout' (kort Flexbox of flex), dat een eendimensioneel lay-outstelsel is, is 'Grid Layout' (kort Grid) een tweedimensioneel lay-outstelsel.



Figure 1 Representative Flex Layout Example



Figure 2 Representative Grid Layout Example

- ▶ Grid en flex kunnen samen gebruikt worden om complexe lay-outs te maken.

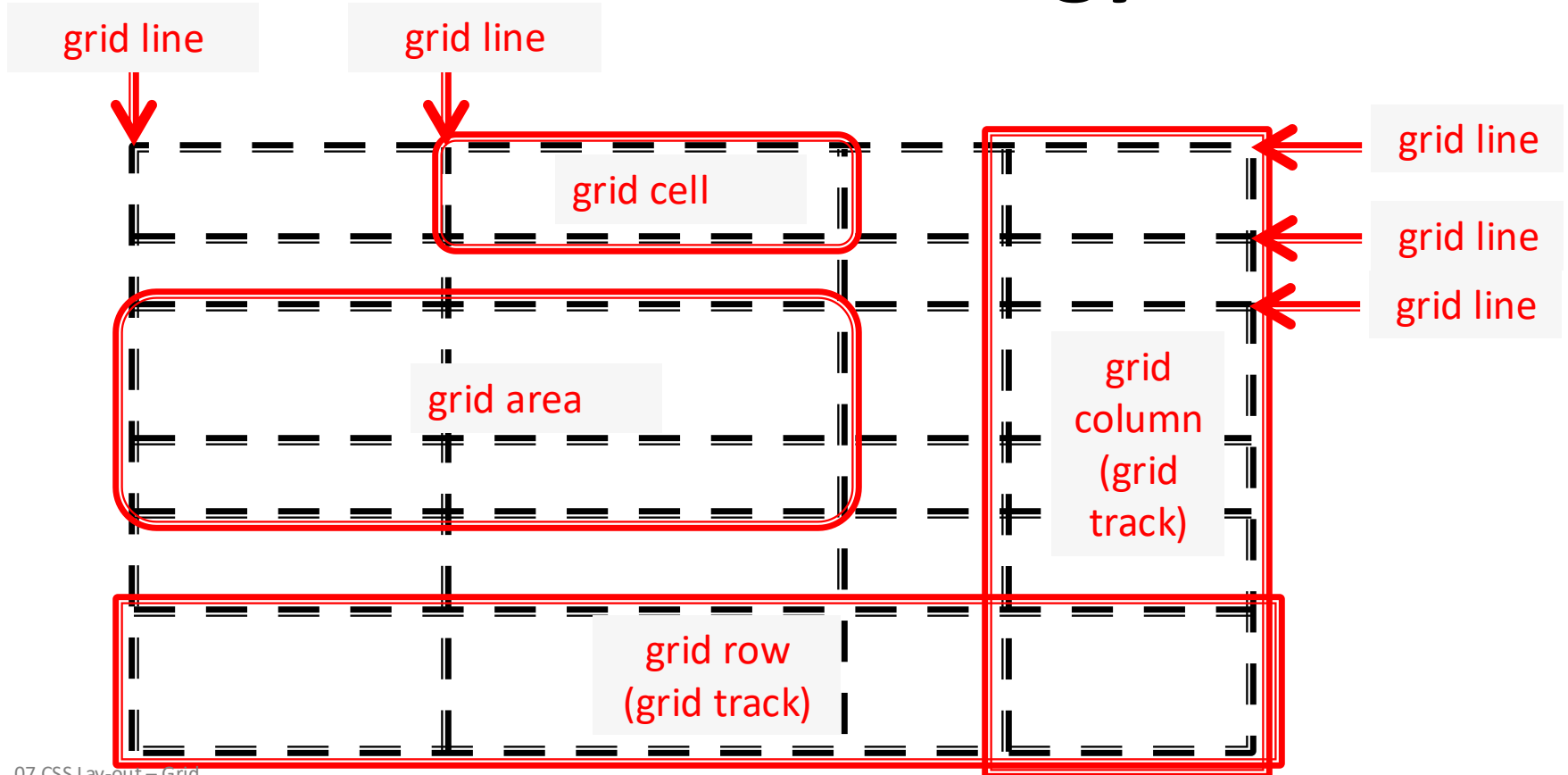
Wat is een grid?

- ▶ Een *grid* (of raster) bestaat uit (denkbeeldige) horizontale en verticale lijnen die een *grid container* opdelen in rijen, kolommen en cellen. Te vergelijken met een tabel.
- ▶ Op de volgende dia vind je een afbeelding van een grid waarop verschillende grid-termen zijn aangeduid.

Meer info vind je op

<https://www.w3.org/TR/css-grid-1/#grid-concepts>

Grid terminology



A large, stylized orange number '4' is positioned on the left side of the slide, partially overlapping the text.

Inleidend voorbeeld CSS Grid

Theoriebestanden (voorbeelden)

- ▶ Open in Visual Studio Code de map **01-css-grid-inleidend-voorbeeld**

Inleidend voorbeeld CSS Grid

- ▶ Beschouw een element met daarin een aantal child elements.

Voorbeeld:

```
<div>  
  <div>item 1</div>  
  <div>item 2</div>  
  <div>item 3</div>  
  <div>item 4</div>  
  <div>item 5</div>  
</div>
```

(Voorbeeld)

```
<body>  
  <header>Header</header>  
  <nav>Navigation</nav>  
  <main>Main area</main>  
  <footer>Footer</footer>  
</body>
```

HTML

```
<div class="grid-container">
  <div>item 1</div>
  <div>item 2</div>
  <div>item 3</div>
  <div>item 4</div>
  <div>item 5</div>
</div>
```

CSS

```
/* De boxes opmaken */
* {
  box-sizing: border-box;
}

.grid-container {
  background-color: cyan;
  border-radius: 5px;
  padding: 5px;
}

.grid-container div {
  border: 1px solid black;
  background-color: white;
  border-radius: 5px;
  padding: 1em;
}
```

Hoe nu gebruikmaken van CSS Grid Layout?

▶ Inleidend voorbeeld CSS Grid:

1. Creëer een *grid container* met

`display: block grid;` of (vroeger) `display: grid;`

De eerste waarde duidt aan dat de container een block element is.

De tweede waarde heeft tot gevolg dat alle directe kind-elementen van dit *grid container*-element, *grid items* worden.

Met `display: block grid` definieer je dus niet alleen de *grid container*, maar ook impliciet de *grid items*.

2. Definieer de *grid* (het aantal rijen en/of kolommen en hun hoogte/breedte), met

`grid-template-columns`

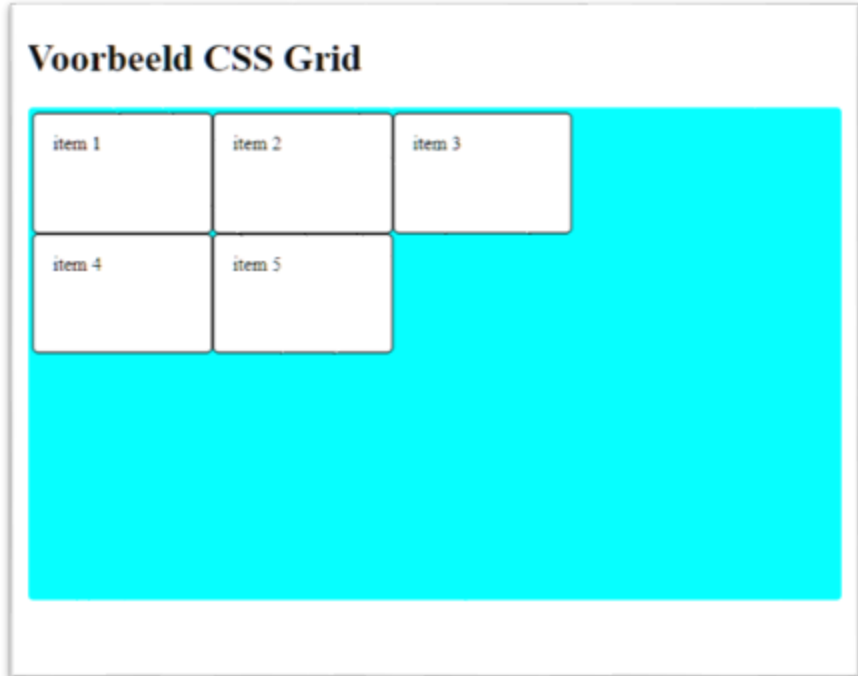
`grid-template-rows`

(kan ook nog op andere manieren, zie later)

CSS

```
/* CSS Grid Layout */  
.grid-container {  
  display: block grid;  
  grid-template-columns:  
    150px 150px 150px;  
  grid-template-rows:  
    100px 100px 100px 100px;  
}
```

Merk op dat de grid items automatisch rij per rij in de grid geplaatst worden.



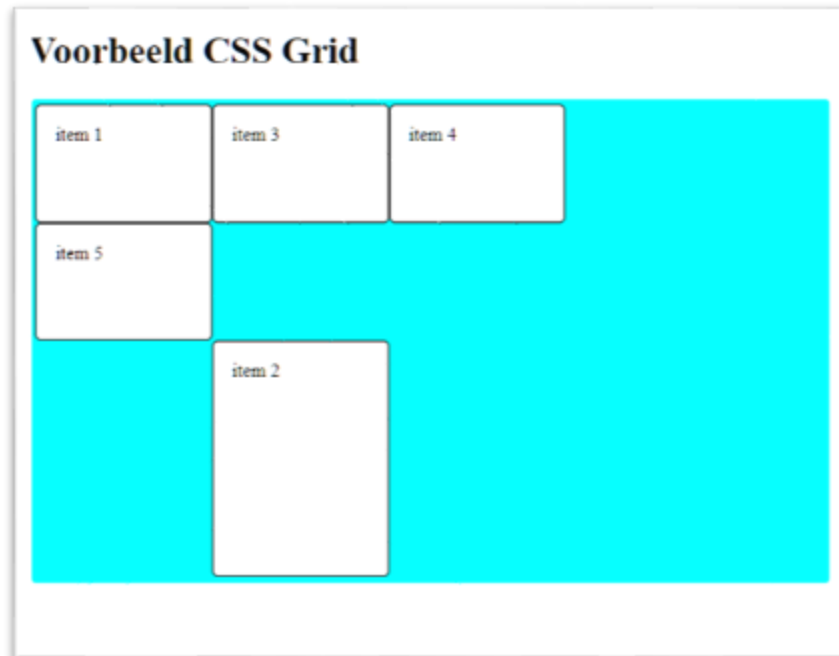
Grid items positioneren op de grid met Line Numbers

Inleidend voorbeeld CSS Grid:

We kunnen grid-items ook expliciet positioneren op de grid. In het voorbeeld wordt 'item 2' gepositioneerd op de grid.

```
/* CSS Grid Layout */  
.grid-container {  
  zie vorig dia  
}  
div:nth-of-type(2) {  
  grid-row: 3/5;  
  grid-column: 2;  
}
```

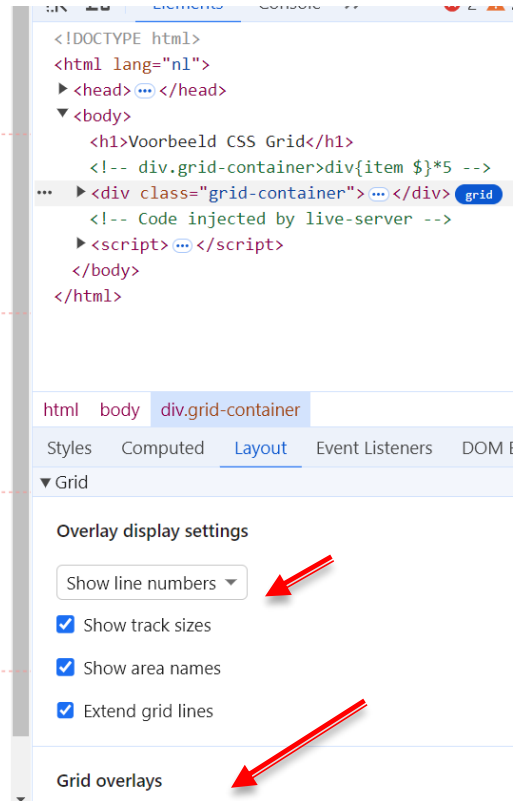
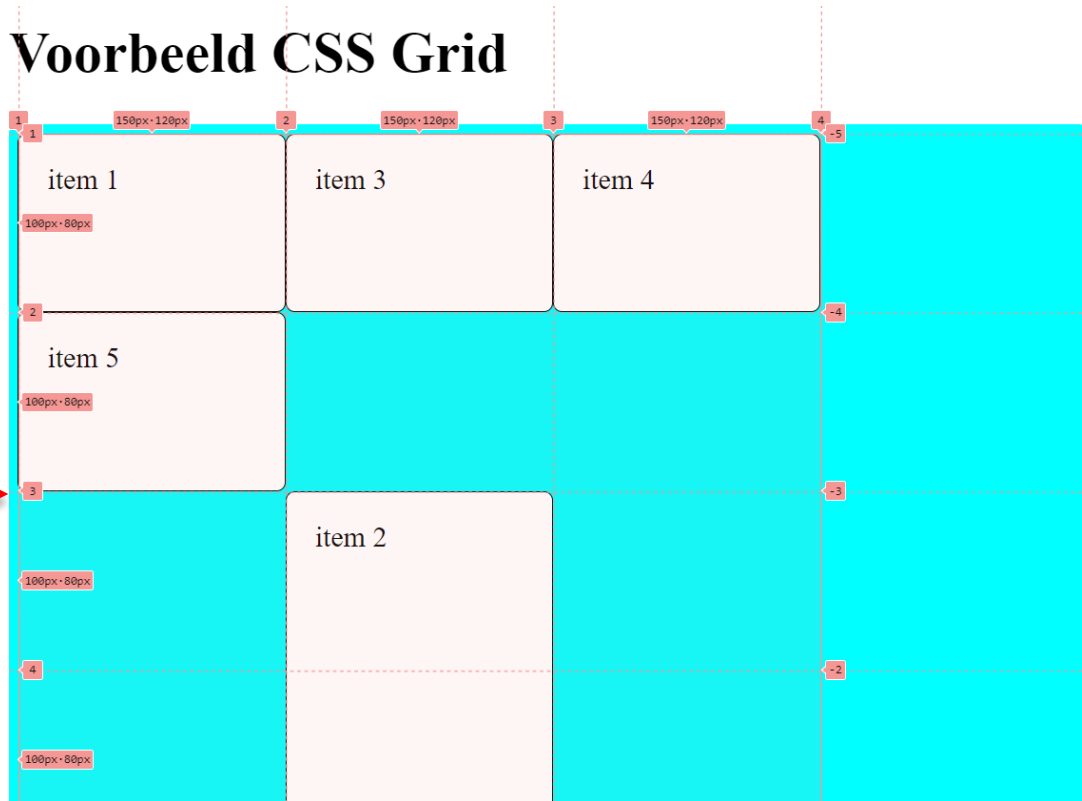
Merk op dat de getallen bij grid-row en grid-column lijnnummers zijn (zie volgende dia). 'grid-row: 3/5' (betekent dus dat de div rij 3 en rij 4 overspant



Inleidend voorbeeld CSS Grid:

Afbeelding met Grid Lines en Line Numbers zichtbaar via de Grid Inspector

Voorbeeld CSS Grid



grid-row-start

grid-row-end

A large, stylized orange number '4' is positioned on the left side of the slide. The text 'CSS Grid Layout' is centered horizontally across the middle of the '4' and extends slightly to the right.

CSS Grid Layout

Waarom CSS grid layout?

- ▶ Zorgt voor een goede scheiding tussen inhoud (html) en opmaak (css). We kunnen een html-element een andere positie op de grid geven zonder dat we daarvoor de markup (html) moeten wijzigen.

Algemene procedure gebruik Grid Layout

1. Creëer een *grid container*. Hiermee definieer je impliciet ook de *grid items*.
2. Definieer de *grid*
3. Plaats de *grid items* op de grid.

1. Creëer de *grid container*

Je creëert een *grid container*-element met

- **display**: `block grid` creëert een block-level-element.
 - Is hetzelfde als **display**: `grid`
- **display**: `inline grid` creëert een inline-level-element.
 - Is hetzelfde als **display**: `inline-grid`

Het creëren van een *grid container* heeft tevens tot gevolg dat alle directe kind-elementen van het *grid container*-element, *grid items* worden.

De *grid container* zelf is dus een *block* of *inline* element, afhankelijk van de eerste waarde

2. Definieer de *grid*

De drie properties

`grid-template-rows`,
`grid-template-columns` en
`grid-template-areas` (zie later)

definiëren de ‘explicit grid’ van een grid-container.

De uiteindelijke grid kan echter groter zijn vanwege grid-items die buiten de ‘explicit grid’ geplaatst worden.

In dit geval zullen er impliciet tracks gecreëerd worden.

De afmetingen van deze tracks worden bepaald door `grid-auto-rows` en `grid-auto-columns`.

Voorbeeld 01:

- ▶ In het volgende voorbeeld definiëren we een grid container met een grid bestaande uit drie kolommen van elk **150px** breed.
- ▶ Aangezien we niet expliciet het aantal en de hoogte van de rijen opgeven, zullen er automatisch rijen worden aangemaakt, waarbij ook de hoogte automatisch bepaald wordt.
- ▶ Er is in dit voorbeeld geen expliciete plaatsing van de grid items, waardoor de grid items in 'HTML source order' in de grid geplaatst zullen worden.

```
<body>
  <h1>Voorbeeld CSS Grid</h1>
  <!-- div.grid-container>div{item $}*5 -->
  <div class="grid-container">
    <div>item 1</div>
    <div>item 2</div>
    <div>item 3</div>
    <div>item 4</div>
    <div>item 5</div>
  </div>
</body>
```

```
/* CSS Grid Layout */
/***** */
.grid-container {
  display: block grid;
  grid-template-columns: 150px 150px 150px;
}
```

item 1

item 2

item 3

item 4

item 5

grid-template-rows en grid-template-columns

- ▶ De waarde voor [grid-template-columns](#) (en [grid-template-rows](#)) kan een **<track-list>** zijn of een **<auto-track-list>**
- ▶ Mogelijke track sizes bij een **<track-list>** zijn:
 - [<length>](#)
 - [<percentage>](#)
 - [<flex>](#)
 - [min-content](#)
 - [max-content](#)
 - [minmax\(min,max\)](#)
 - [fit-content\(value\)](#)
 - **auto**

```
/* Voorbeelden
<track-list> */
grid-template-columns: 20ch 200px 200px;
grid-template-columns: 20% 60% 40%;
grid-template-columns: 100px 1fr 2fr;
grid-template-columns: minmax(100px, 200px);
grid-template-columns: fit-content(40%);
grid-template-columns: auto 100px 1fr;
```

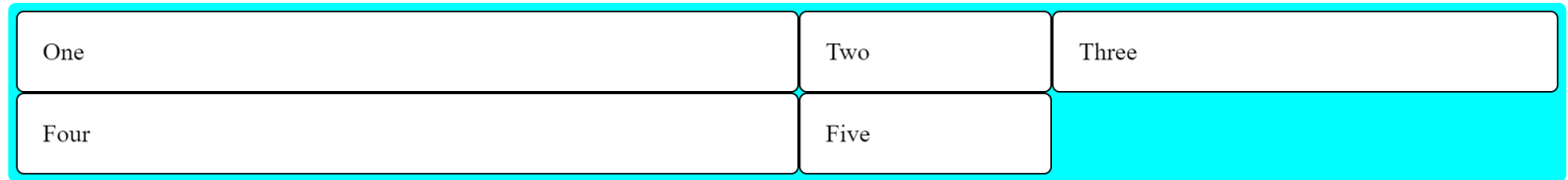
De *fr*-eenheid (<flex> CSS data type)

- ▶ Met de *fr*-eenheid kan je 'flexible grid tracks' (rijen of kolommen) maken.
- ▶ De *fr*-eenheid stelt een fractie voor van de 'leftover space'. De 'leftover space' is de ruimte die niet ingenomen wordt door 'non-flexible grid tracks'.

/* Voorbeeld met één non-flexible column track (500px) en twee flexible column tracks.

De 'leftover space' wordt verdeeld over de tweede en de derde kolom. Hiervoor wordt de 'leftover space' eerst in drie gedeeld en hiervan krijgt de tweede kolom één deel en de derde kolom twee delen.*/

```
/* CSS Grid Layout */  
/*****  
.grid-container {  
  display: block grid;  
  grid-template-columns: 500px 1fr 1fr;  
}
```



/* Voorbeeld met één kolom met breedte van 2fr en twee kolommen met een breedte van 1fr. De beschikbare ruimte wordt dus in vier gedeeld. De eerste kolom krijgt twee delen en de twee overige kolommen elk één deel.*/

```
.grid-container {  
  display: block grid;  
  grid-template-columns: 2fr 1fr 1fr;  
}
```

One	Two	Three
Four	Five	

/* Voorbeeld met drie kolommen van gelijke breedte, die automatisch groeien of krimpen afhankelijk van de beschikbare ruimte*/

```
.grid-container {  
  display: block grid;  
  grid-template-columns: 1fr 1fr 1fr;  
}
```

One	Two	Three
Four	Five	

repeat()-functie

Met de repeat()-functie kan je een (deel van een) 'track-list' herhalen.

- ▶ Zo kan je

```
grid-template-columns: 1fr 1fr 1fr;
```

korter schrijven als:

```
grid-template-columns: repeat(3, 1fr);
```

- ▶ Een meer uitgebreid voorbeeld:

```
grid-template-columns: 20px repeat(4, 1fr 2fr);
```

is hetzelfde als:

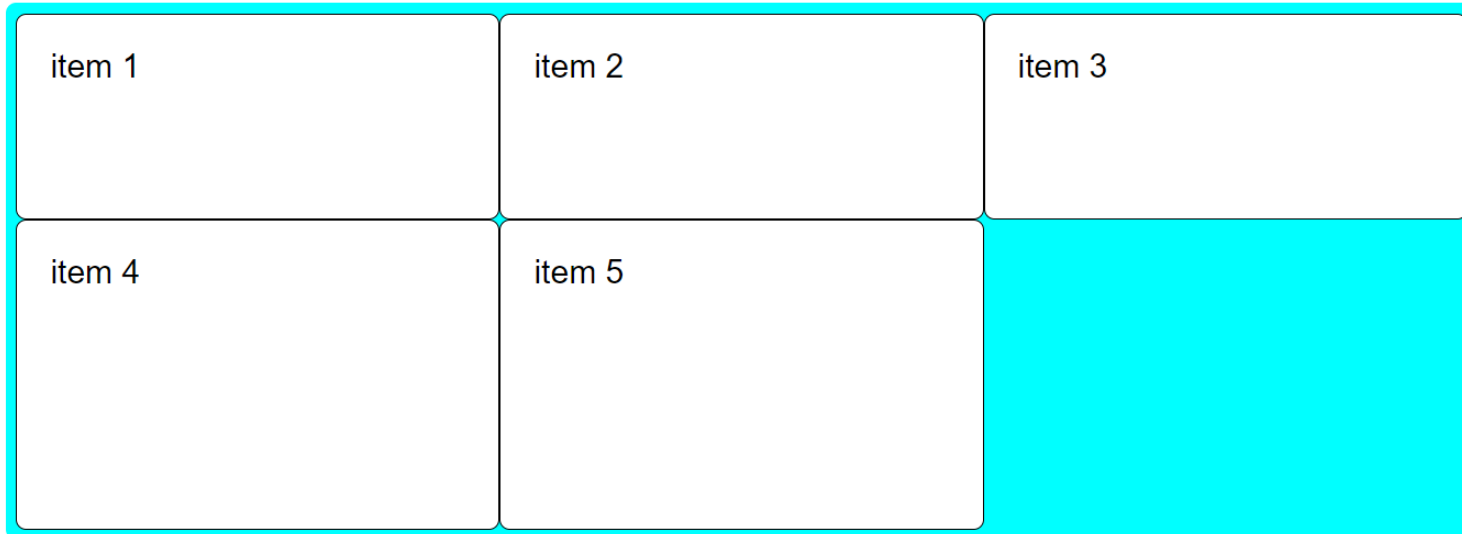
```
grid-template-columns: 20px 1fr 2fr 1fr 2fr 1fr 2fr  
1fr 2fr;
```

grid-auto-rows en grid-auto-columns

- ▶ In de vorige voorbeelden hebben we de kolommen (column tracks) expliciet gedefinieerd met de **grid-template-columns**-property en de rijen werden impliciet gecreëerd.
- ▶ Het instellen van de hoogte van deze impliciet gecreëerde rijen doe je via [grid-auto-rows](#).
(Voor de kolommen is er [grid-auto-columns](#)).
- ▶ De initial value voor deze properties is **auto**. Impliciet gecreëerde rijen (en kolommen) zijn dus standaard 'auto-sized'.

```
/*de hoogte van de eerste rij stellen we in op 110px  
bijkomende impliciet gemaakte rijen krijgen een hoogte van 150px*/
```

```
.grid-container {  
  display: block grid;  
  grid-template-columns: 1fr 1fr 1fr;  
  grid-template-rows: 100px;  
  grid-auto-rows: 150px;  
}
```



grid-auto-rows en grid-auto-columns

- ▶ Bij **grid-auto-rows/grid-auto-columns** kunnen we dezelfde ‘track sizes’ gebruiken als bij **grid-template-rows/grid-template-columns**.

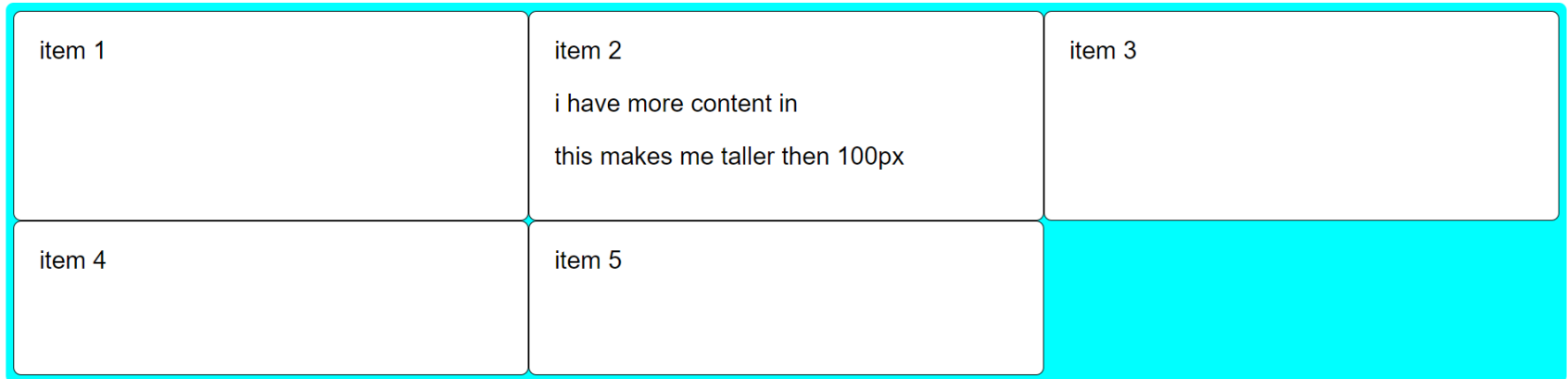
- ▶ Voorbeeld `minmax()`:

```
grid-auto-rows: minmax(100px, auto);
```

In dit voorbeeld zal de rijhoogte steeds aangepast worden aan de inhoud (`auto`), maar de rijhoogte zal nooit kleiner worden dan `100px`.



```
.grid-container {  
  display: block grid;  
  grid-template-columns: 1fr 1fr 1fr;  
  grid-auto-rows: minmax(100px, auto);  
}
```



repeat() met auto-fill en auto-fit

- ▶ De waarde voor **grid-template-columns** (en **grid-template-rows**) kan niet alleen een `<track-list>` zijn (zie vroeger), maar ook een **<auto-track-list>**.
- ▶ Bij een **<auto-track-list>** gaan we bij de [repeat\(\)](#)-functie, als eerste argument i.p.v. een getal op te geven, de waarde **auto-fill** of **auto-fit** gebruiken.

Voorbeeld: repeat(auto-fill,...)

```
grid-template-columns: repeat(auto-fill, minmax(150px, 1fr));
```

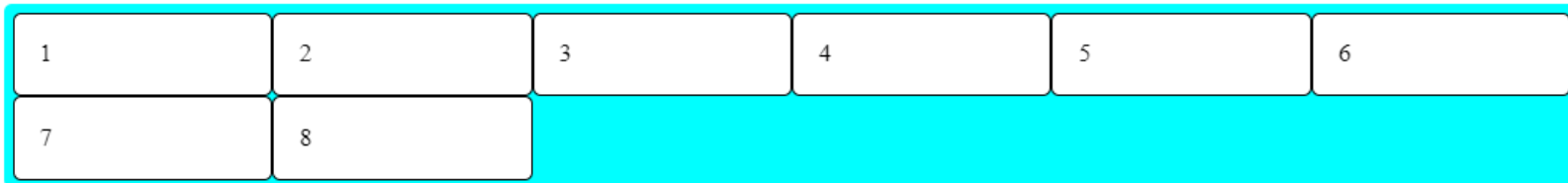
We geven zelf geen vast aantal kolommen op bij de `repeat()`-functie maar laten de browser het aantal kolommen bepalen (`auto-fill`).

Voorbeeld: bij 800px is er niet genoeg ruimte voor 6 kolommen van 150px ($6 * 150px = 900px$) en een kolom mag niet smaller zijn dan 150px, dus maakt de browser 5 kolommen.

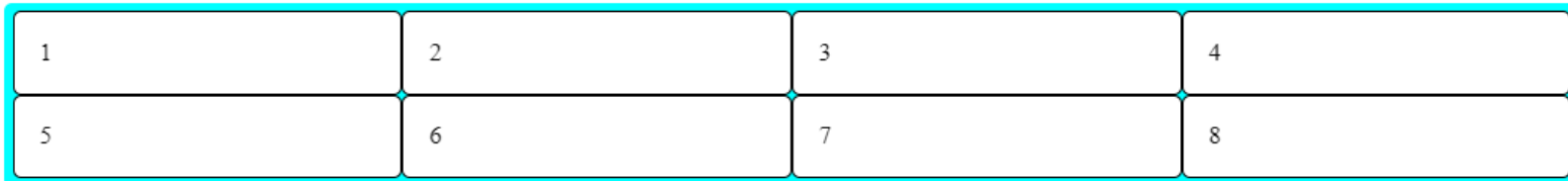
Door het toevoegen van de max-waarde **1fr** wordt bovendien de overgebleven ruimte verdeeld over de 5-kolommen.

Voorbeeld: repeat(auto-fill,...)

```
grid-template-columns: repeat(auto-fill, minmax(150px, 1fr));
```



```
grid-template-columns: repeat(4, 1fr);
```



Meer info over auto-fill en auto-fit

- ▶ **Auto-fill** en **auto-fit** werken op dezelfde manier, behalve als er lege tracks zijn.
- ▶ Een goede video over hoe **auto-fill** en **auto-fit** werken en wat het verschil is tussen de twee vind je op de website van <https://gridbyexample.com> van [Rachel Andrew](#).

Videolink:

<https://gridbyexample.com/video/series-auto-fill-auto-fit/>

3. positioneren van grid items op de grid

Tot nu toe hebben we enkel de grid, met andere woorden de grid structuur (de rijen en de kolommen) gedefinieerd.

We gaan nu kijken naar de mogelijkheden om de grid items te positioneren op de grid.

Hierbij kunnen grid items ook meerdere rijen en kolommen overspannen.

Grid Lines

- ▶ Als je een grid definieert dan krijgen de lijnen waaruit de grid bestaat automatisch nummers. Je kan deze lijnnummers gebruiken om een 'grid item' expliciet te positioneren op de grid.
- ▶ Je kan in de inspector, bij Layout, kiezen om de line numbers of line names te tonen van de selecteerde grid
- ▶ De getallen verwijzen naar de lijnen en niet naar de kolommen of de rijen. Zo bevat een 3-column-layout 4 lijnen.
- ▶ Naast de positieve lijnnummers zijn er ook negatieve lijnnummers, waarbij er achterwaarts geteld wordt.

The screenshot shows the 'Layout' tab in the browser's developer tools. Under the 'Grid' section, there are several settings:

- 'Overlay display settings' section:
 - A dropdown menu set to 'Show line numbers'.
 - Three checked checkboxes: 'Show track sizes', 'Show area names', and 'Extend grid lines'.
- 'Grid overlays' section:
 - A checked checkbox next to 'div.grid-container', which is highlighted in orange in the original image.

Afbeelding met Grid Lines en Line Numbers zichtbaar via de Chrome Inspector

The image shows a CSS Grid layout with 4 columns and 2 rows. The grid is defined by the following track sizes: 1fr, 184.92px, 1fr, 184.93px, 1fr, 184.92px. The items are arranged as follows:

- Row 1: item 1 (width: 1fr, height: 80px), item 2 (width: 1fr, height: 80px), item 3 (width: 1fr, height: 80px)
- Row 2: item 4 (width: 1fr, height: 80px), item 5 (width: 1fr, height: 80px), a cyan-colored cell (width: 1fr, height: 80px)

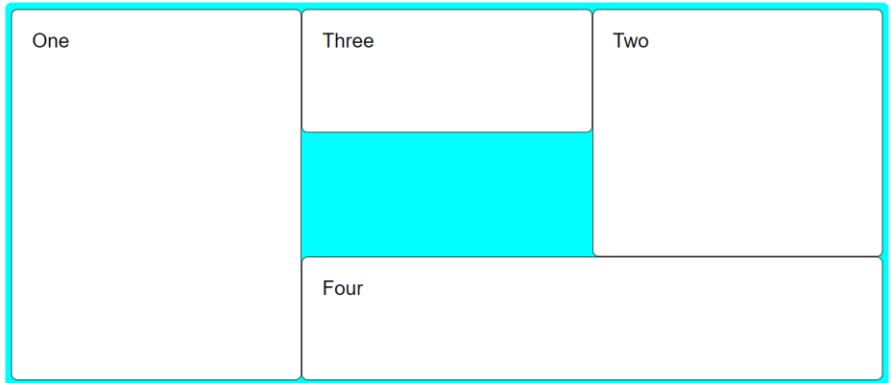
The Chrome DevTools interface on the right shows the following settings:

- Grid overlay settings:
 - Show line numbers:
 - Show track sizes:
 - Show area names:
 - Extend grid lines:
- Grid overlays:
 - div.grid-container

Grid Items positioneren op de Grid met Line Numbers

```
.box1 {grid-column-start: 1;grid-column-end: 2;grid-row-start: 1;grid-row-end: 4;}  
.box2 {grid-column-start: 3;grid-column-end: 4;grid-row-start: 1;grid-row-end: 3;}  
.box3 {grid-column-start: 2;grid-column-end: 3;grid-row-start: 1;grid-row-end: 2;}  
.box4 {grid-column-start: 2;grid-column-end: 4;grid-row-start: 3;grid-row-end: 4;}
```

```
<div class="grid-container">  
  <div class="box1">One</div>  
  <div class="box2">Two </div>  
  <div class="box3">Three</div>  
  <div class="box4">Four</div>  
</div>
```



Default Spans

- ▶ In het vorige voorbeeld werd elke 'grid-row-end' en 'grid-column-end' expliciet vermeld. Als het item echter maar één track (rij of kolom) overspant, mag je de 'grid-row-end' of 'grid-column-end' waarden weglaten. Het vorige voorbeeld kan dus ook geschreven worden als:

```
.box1 {grid-column-start: 1;grid-row-start: 1;grid-row-end: 4;}  
.box2 {grid-column-start: 3;grid-row-start: 1;grid-row-end: 3;}  
.box3 {grid-column-start: 2;grid-row-start: 1;}  
.box4 {grid-column-start: 2;grid-column-end: 4;grid-row-start: 3;}
```


Het sleutelwoord 'span'

- ▶ In plaats van het 'end' lijnnummer op te geven, kan je ook het aantal tracks opgeven dat je wilt overspannen.
- ▶ Voorbeeld:

```
grid-row-start: 1;grid-row-end: 4;}
```

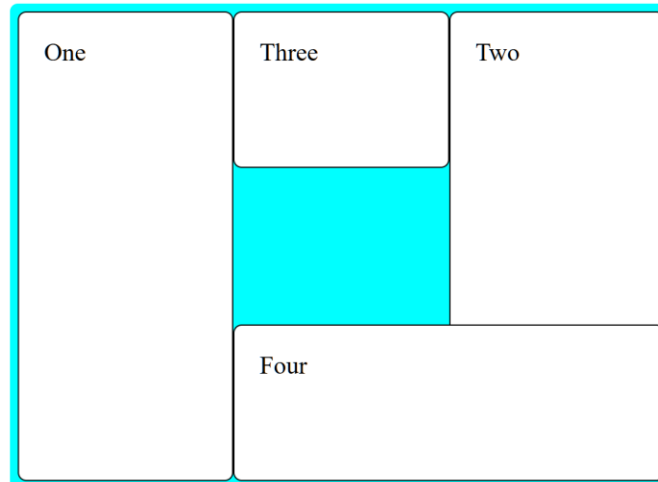
kan je dus ook schrijven als

```
grid-row-start: 1;grid-row-end: span 3;}
```

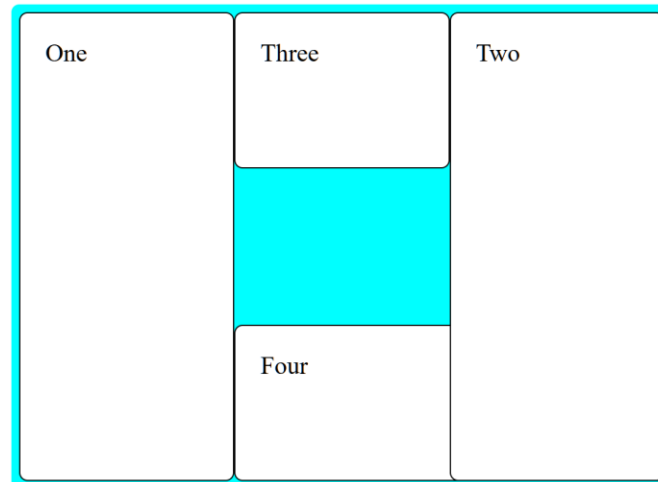
Overlappende grid items

- ▶ Grid items kunnen overlappen.
- ▶ Als we in het voorbeeld op de volgende dia `.box2` laten eindigen bij rijlijn 4 i.p.v. bij rijlijn 3, dan zit `.box2` gedeeltelijk verscholen achter `.box4`. Wensen we `.box2` naar de voorgrond te brengen dan moeten we de [z-index](#)-property aanpassen.

```
.box2 {  
  grid-column-start: 3;  
  grid-row-start: 1;  
  grid-row-end: 4;  
}
```



```
.box2 {  
  grid-column-start: 3;  
  grid-row-start: 1;  
  grid-row-end: 4;  
  z-index: 1;  
}
```



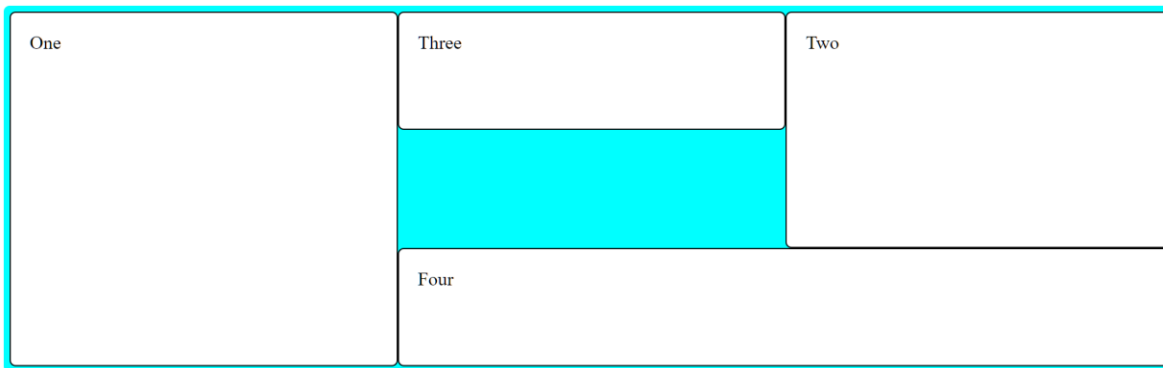
Grid items plaatsen

- ▶ Om *grid items* in de *grid* te plaatsen bestaan er naast de grid-placement properties **grid-column-start**, **grid-column-end**, **grid-row-start** en **grid-row-end** ook de shorthands **grid-column**, **grid-row** en **grid-area**.



Bron: <https://www.w3.org/TR/css-grid-1/#common-uses>

- ▶ Op de volgende dia's vind je, hoe je het voorbeeld korter kan schrijven met deze properties. Meer info over het plaatsen van grid items vind je o.a. in de specification: <https://www.w3.org/TR/css-grid-1/#placement>

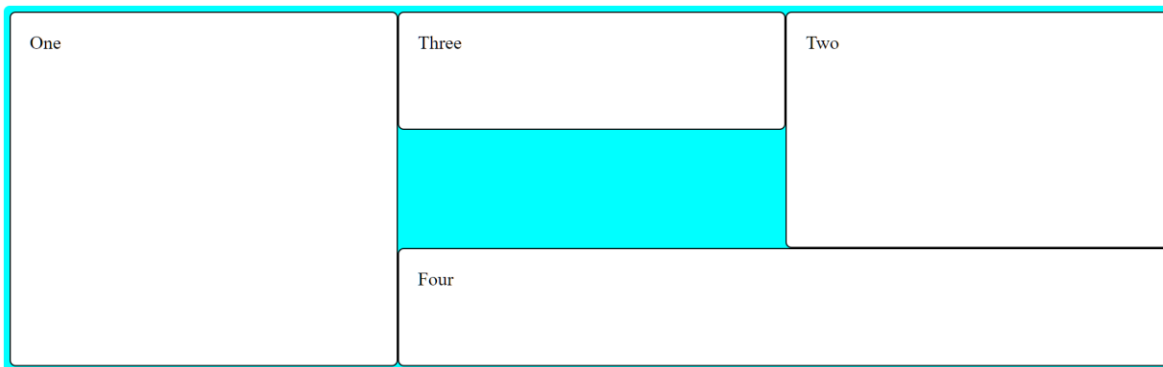


Voorbeeld ...

```
.box1 {grid-column-start: 1;grid-row-start: 1;grid-row-end: 4;}  
.box2 {grid-column-start: 3;grid-row-start: 1;grid-row-end: 3;}  
.box3 {grid-column-start: 2;grid-row-start: 1;}  
.box4 {grid-column-start: 2;grid-column-end: 4;grid-row-start: 3;}
```

...kan korter geschreven worden met de `grid-column` en `grid-row` shorthand properties

```
.box1 {grid-column: 1;grid-row: 1 / 4;}  
.box2 {grid-column: 3;grid-row: 1 / 3;}  
.box3 {grid-column: 2;grid-row: 1;}  
.box4 {grid-column: 2 / 4;grid-row: 3;}
```



Voorbeeld 06...

```
.box1 {grid-column-start: 1;grid-row-start: 1;grid-row-end: 4;}
.box2 {grid-column-start: 3;grid-row-start: 1;grid-row-end: 3;}
.box3 {grid-column-start: 2;grid-row-start: 1;}
.box4 {grid-column-start: 2;grid-column-end: 4;grid-row-start: 3;}
```

...kan ook korter geschreven worden met de `grid-area` property

```
.box1 {grid-area: 1 / 1 / 4 / 2;} /* De volgorde van de waarden
.box2 {grid-area: 1 / 3 / 3 / 4;} voor de grid-area property is:
.box3 {grid-area: 1 / 2 / 2 / 3;} grid-row-start
.box4 {grid-area: 3 / 2 / 4 / 4;} grid-column-start
grid-row-end
grid-column-end */
```

Achterwaarts tellen

- ▶ Het is ook mogelijk om achterwaarts te tellen bij het opgeven van de lijnnummers. Zo kan je in Voorbeeld 06 het lijnnummer 4 vervangen door lijnnummer -1 en kan

```
.box1 {grid-column:1; grid-row: 1 / 4;}
```

ook geschreven worden als

```
.box1 {grid-column:1; grid-row: 1 / -1;}
```

Gutters

- ▶ Witruimte tussen kolommen en rijen, genoemd ‘Gutters’, creëer je met **column-gap** en **row-gap** of de shorthand **gap**.
- ▶ **Opmerking** Sommige browsers gebruiken nog de oude notaties met de prefix **grid-** nl. **grid-column-gap**, **grid-row-gap** en **grid-gap**, maar de meeste browser-creators hebben hun browser reeds aangepast.


```
.grid-container {  
  display: block grid;  
  grid-template-columns: 1fr 1fr 1fr;  
  grid-template-rows: 50px 50px;  
  column-gap: 10px;  
  row-gap: 1rem;  
}
```

item1

item2

item3

item4

item5

item6



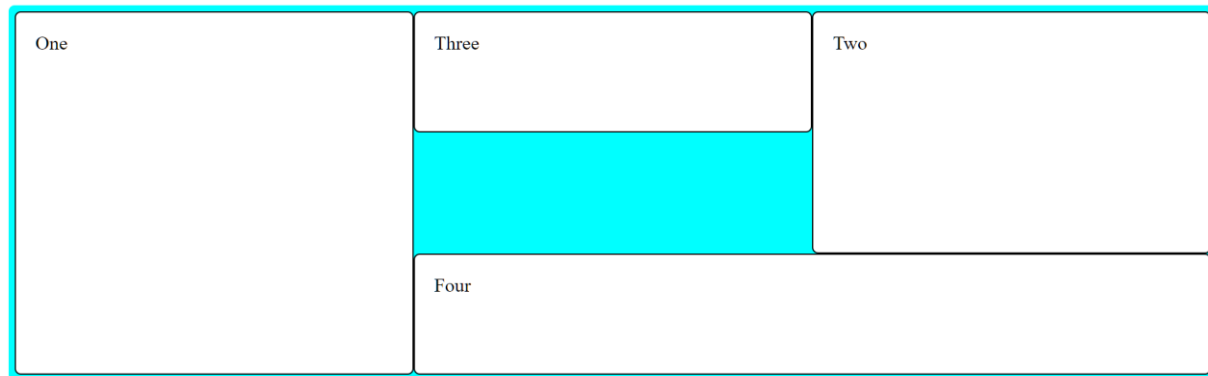
Grid-template-areas property

grid-template-areas property

- ▶ Een andere manier om grid items in de grid te plaatsen is met de [grid-template-areas](#) property. Hiermee creëer je namen voor grid areas die je dan kan gebruiken in bijv. de `grid-area` property om grid items te plaatsen in een grid area.
- ▶ Op de volgende dia vind je Voorbeeld 06 herschreven met `grid-template-areas`. Merk op dat
 - er een rij gecreëerd wordt voor elke string.
 - je meerdere rijen of kolommen kan overspannen door de naam te herhalen;
 - je voor cellen zonder naam een punt `.` gebruikt.
 - de waarde van `grid-template-areas` de structuur van de grid visualiseert.

Voorbeeld 06 herschreven met de `grid-template-areas` property

```
.grid-container {  
  display: block grid;  
  grid-template-columns: repeat(3, 1fr);  
  grid-template-rows: repeat(3, 100px);  
  grid-template-areas:  
    "one three two"  
    "one . two"  
    "one four four";  
}
```



```
.box1 {grid-area: one;}  
.box2 {grid-area: two;}  
.box3 {grid-area: three;}  
.box4 {grid-area: four;}
```

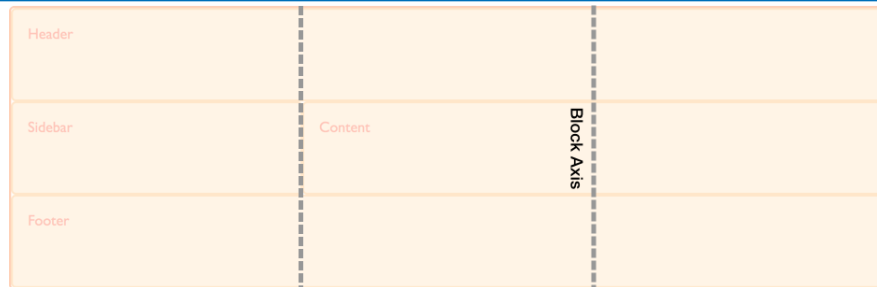
Opmerking 'Named Grid Lines'

- ▶ Je kan niet alleen namen geven aan grid areas, maar ook aan grid lines. Het werken met 'Named Grid Lines' zullen we echter niet bespreken.
- ▶ Wie hierin toch geïnteresseerd is zie: [MDN - Layout using named grid lines](#)

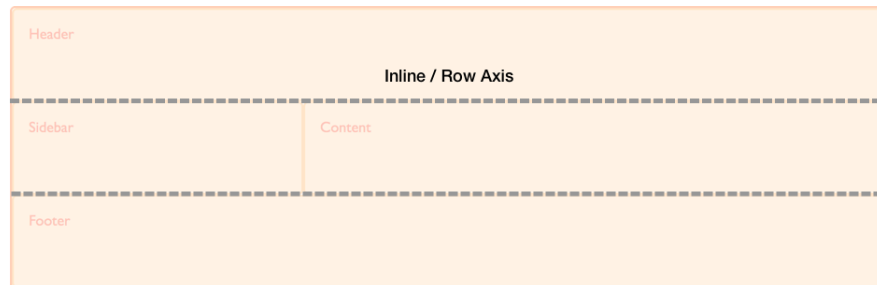
Box alignment in CSS Grid Layout

BRON: MDN Box alignment in CSS Grid Layout (https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Grid_Layout/Box_Alignment_in_CSS_Grid_Layout)

Box alignment in CSS Grid Layout



Om items uit te lijnen langs de *block axis* gebruik je de properties [align-items](#) en [align-self](#).



Items uitlijnen langs de *inline axis* doe je met [justify-items](#) and [justify-self](#).

Theoriebestanden (voorbeelden)

- ▶ Open in Visual Studio Code de map **02-css-grid-box-alignment**

We gebruiken [de voorbeelden van de MDN-website](#). Deze voorbeelden gebruiken allemaal de volgende basis HTML en CSS ...

HTML

```
<div class="wrapper">
  <div class="item1">Item 1</div>
  <div class="item2">Item 2</div>
  <div class="item3">Item 3</div>
  <div class="item4">Item 4</div>
</div>
```

CSS

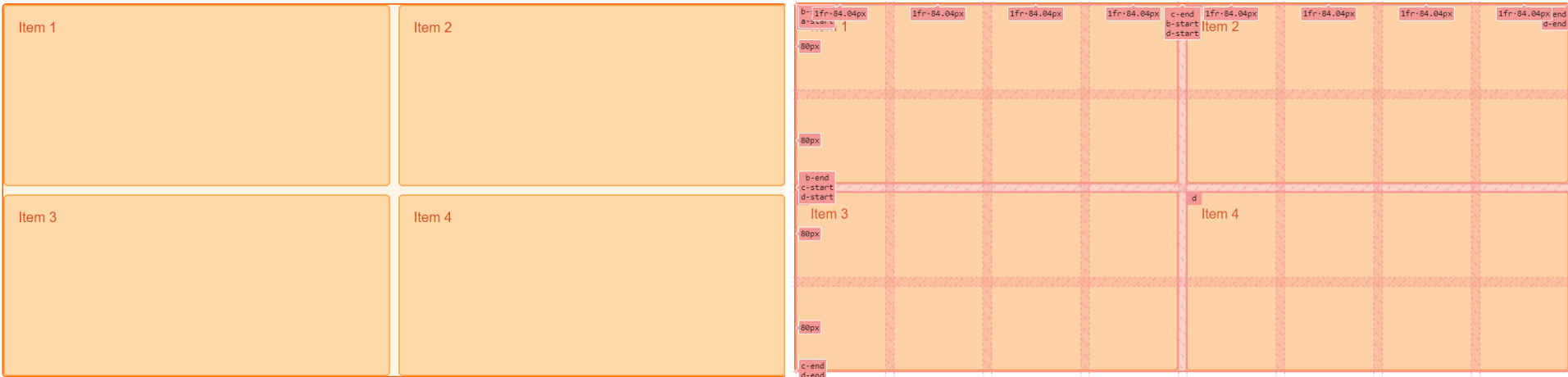
```
.wrapper {
  display: block grid;
  grid-template-columns: repeat(8, 1fr);
  grid-gap: 10px;
  grid-auto-rows: 100px;
  grid-template-areas:
    "a a a a b b b b"
    "a a a a b b b b"
    "c c c c d d d d"
    "c c c c d d d d";
}

.item1 {grid-area: a;}
.item2 {grid-area: b;}
.item3 {grid-area: c;}
.item4 {grid-area: d;}
```

... met het onderstaande als resultaat.

In de rechtse afbeelding is de grid gevisualiseerd met de inspector van Chrome.

In dit startvoorbeeld zijn er geen expliciete Box alignment properties ingesteld;

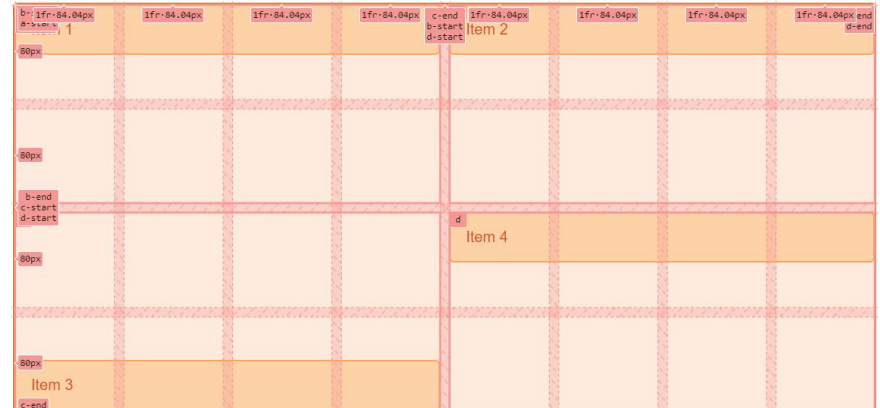


Block axis: align-self en align-items

- ▶ De **align-self** property stelt de uitlijning in van een grid item binnen zijn grid area/cell langs de block axis.
- ▶ De default voor **align-self** komt overeen met **stretch**. Bij elke andere waarde worden de afmetingen van het item automatisch berekend zodat de inhoud er net in past.
- ▶ De **align-items** property stelt de **align-self** property in voor alle grid-items. De **align-items** property stel je in op de grid container.

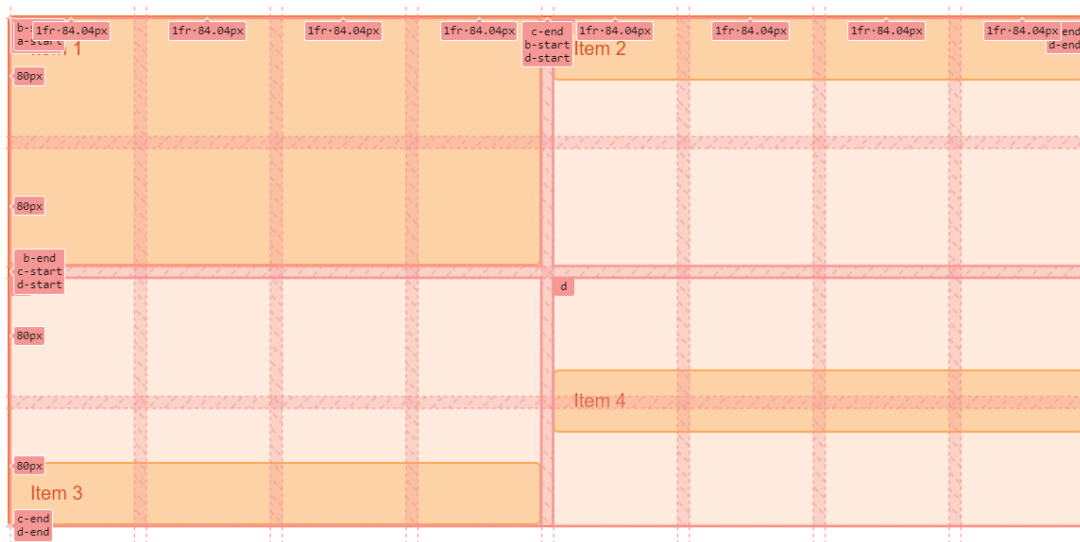
Voorbeeld 1a: items uitlijnen langs de block axis met `align-items`

```
.wrapper {  
  align-items: start;  
}
```



Voorbeeld 1b: items uitlijnen langs de block axis met `align-self`

```
.item2 {align-self: start;}  
.item3 {align-self: end;}  
.item4 {align-self: center;}
```

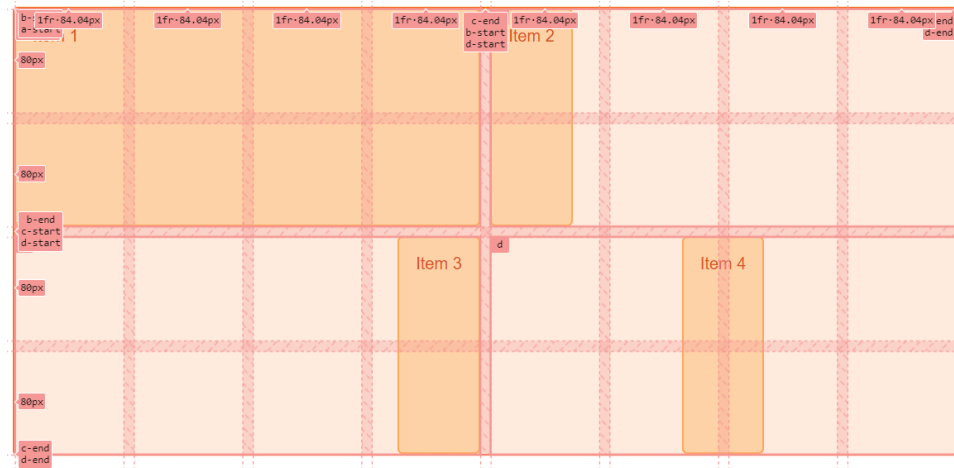


Inline axis: justify-self en justify-items

- ▶ De **justify-self** property stelt de uitlijning in van een grid item binnen zijn grid area/cell langs de inline axis.
- ▶ De default voor **justify-self** komt overeen met **stretch**. Bij elke andere waarde worden de afmetingen van het item automatisch berekend zodat de inhoud er net in past.
- ▶ De **justify-items** property stelt de **justify-self** property in voor alle grid items.
De **justify-items** property stel je in op de grid container.

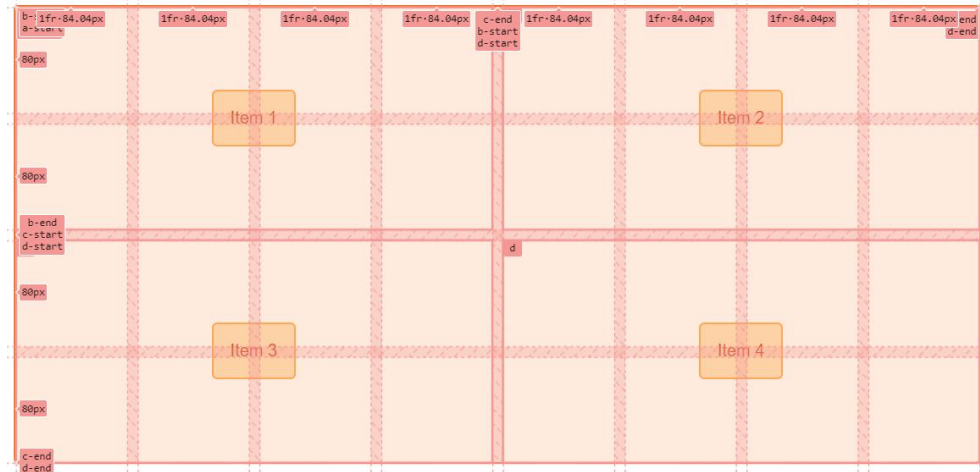
Voorbeeld 2: items uitlijnen langs de inline axis met `justify-self`

```
.item2 {justify-self: start;}  
.item3 {justify-self: end;}  
.item4 {justify-self: center;}
```



Voorbeeld 3: items centreren met `align-items` en `justify-items`

```
.wrapper {  
  align-items: center;  
  justify-items: center;  
}
```



align-content en justify-content

De [align-content](#) en de [justify-content](#) properties worden gebruikt als de grid tracks (rijen of kolommen) niet de volledige grid container innemen. Je kan in dit geval de tracks zelf uitlijnen binnen de container.

Extra info (geen examenstof)

Nog enige extra info in verband met CSS Grid:

- ▶ Ook voor de **grid-template-rows**, **grid-template-columns** en **grid-template-areas** zijn er shorthands, namelijk [grid](#) en [grid-template](#).
- ▶ Het artikel [MDN - Auto-placement in CSS Grid Layout](#) bevat onder andere info over het gebruik van [grid-auto-flow](#).
- ▶ CSS Grid level 2 (sinds sept 2023 in de recentste browsers) bevat de definitie van [subgrid](#).

Float

Floating elements (vlotten)

▶ float : left / right / none

- Elementen worden uit de normale flow gehaald. Men kan dan meegeven in welke richting (right – left) ze zullen vlotten binnen hun bevattende container (parent block). Elementen worden tegen de opgegeven rand geplaatst.
- De overige elementen binnen deze container (parent block) zullen dan de vrijgekomen plaats proberen op te vullen en zullen zich rond het element plaatsen.
- Het is duidelijk dat voor het vlottende element een breedte zal moeten worden ingesteld (een block element neemt altijd de maximale breedte in van de bevattende container)

Floating left en right

► 02-Float/01-float.html

```
<body>
<h1>The Evolution of the Bicycle</h1>
<blockquote>
  "Life is like riding a bicycle. To keep your balance you must keep
  moving." - Albert Einstein
</blockquote>
<p>
  In 1817 Baron von Drais invented a walking machine that would help him get
  around the royal gardens faster: two same-size in-line wheels, the front
  one steerable, mounted in a frame upon which you straddled. The device was
  propelled by pushing your feet against the ground, thus rolling yourself
  and the device forward in a sort of gliding walk.
</p>
<p>
  The machine became known as the Draisienne (or "hobby horse"). It was made
  entirely of wood. This enjoyed a short lived popularity as a fad, not
  being practical for transportation in any other place than a well
  maintained pathway such as in a park or garden.
</p>
<p>
  The next appearance of a two-wheeled riding machine was in 1865, when
  pedals were applied directly to the front wheel. This machine was known as
```

Floating left en right

► Normal flow

```
<style>
  body {
    width: 750px;
    font-family: Arial, Verdana, sans-serif;
    color: #665544;
  }
  h1 {
    background-color: #efefef;
    padding: 10px;
  }
  blockquote {
    width: 250px;
    font-size: 130%;
    font-style: italic;
    font-family: Georgia, Times, serif;
    margin: 0px 0px 10px 10px;
    padding: 10px;
    border-top: 1px solid #665544;
    border-bottom: 1px solid #665544;
  }
</style>
```

The Evolution of the Bicycle

"Life is like riding a bicycle. To keep your balance you must keep moving." - Albert Einstein

In 1817 Baron von Drais invented a walking machine that would help him get around the royal gardens faster: two same-size in-line wheels, the front one steerable, mounted in a frame upon which you straddled. The device was propelled by pushing your feet against the ground, thus rolling yourself and the device forward in a sort of gliding walk.

The machine became known as the Draisienne (or "hobby horse"). It was made entirely of wood. This enjoyed a short lived popularity as a fad, not being practical for transportation in any other place than a well maintained pathway such as in a park or garden.

The next appearance of a two-wheeled riding machine was in 1865, when pedals were applied directly to the front wheel. This machine was known as the velocipede (meaning "fast foot") as well as the "bone shaker," since it's wooden structure combined with the cobblestone roads of the day made for an extremely uncomfortable ride. They also became a fad and indoor riding academies, similar to roller rinks, could be found in large cities.

Floating left en right

► Blockquote floating

```
blockquote {  
  float: right;  
  width: 250px;  
  font-size: 130%;  
  font-style: italic;  
  font-family: Georgia, Times, serif;  
  margin: 0px 0px 10px 10px;  
  padding: 10px;  
  border-top: 1px solid #665544;  
  border-bottom: 1px solid #665544;  
}
```

The Evolution of the Bicycle

In 1817 Baron von Drais invented a walking machine that would help him get around the royal gardens faster: two same-size in-line wheels, the front one steerable, mounted in a frame upon which you straddled. The device was propelled by pushing your feet against the ground, thus rolling yourself and the device forward in a sort of gliding walk.

The machine became known as the Draisienne (or "hobby horse"). It was made entirely of wood. This enjoyed a short lived popularity as a fad, not being practical for transportation in any other place than a well maintained pathway such as in a park or garden.

"Life is like riding a bicycle. To keep your balance you must keep moving." - Albert Einstein

1865, when pedals were applied directly to the front wheel. This machine was known as the velociped or "boneshaker," since its wooden structure combined with the cobblestone roads of the day made for an extremely uncomfortable ride. They also became a fad and could be found in large cities.

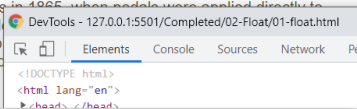
The Evolution of the Bicycle

In 1817 Baron von Drais invented a walking machine that would help him get around the royal gardens faster: two same-size in-line wheels, the front one steerable, mounted in a frame upon which you straddled. The device was propelled by pushing your feet against the ground, thus rolling yourself and the device forward in a sort of gliding walk.

"Life is like riding a bicycle. To keep your balance you must keep moving." - Albert Einstein

The machine became known as the Draisienne (or "hobby horse"). It was made entirely of wood. This enjoyed a short lived popularity as a fad, not being practical for transportation in any other place than a well maintained pathway such as in a park or garden.

The next appearance of a two-wheeled riding machine was in 1817 when pedals were applied directly to the front wheel. This machine was known as the velociped or "boneshaker," since its wooden structure combined with the cobblestone roads of the day made for an extremely uncomfortable ride. They also became a fad and could be found in large cities.



Floating: stacking order

- ▶ Stacking order : meerdere vlottende elementen
 - Floating elements worden vaak gebruikt om block elementen naast elkaar te plaatsen. Dit kan soms voor problemen zorgen.
 - Floating elements vlotten eerst tegen de bovenrand van de parent en dan tegen de volgende beschikbare rand

```
<body>
<h1>The Evolution of the Bicycle</h1>
<div>
  <p>
    In 1817 Baron von Drais invented a walking machine that would help him
    get around the royal gardens faster.
  </p>
  <p>
    The device know as the Draisienne (or "hobby horse") was made of wood,
    and propelled by pushing your feed on the ground in a gliding movement.
  </p>
  <p>
    It was not seen a suitable for any place other than a well maintained
    pathway.
  </p>
  <p>
    In 1865, the velocipede (meaning "fast foot") attached pedals to the
    front wheel, but its wooden structure made it extremely uncomfortable.
  </p>
  <p>
    In 1870 the first all-metal machine appeared. The pedals were attached
    directly to the front wheel.
  </p>
  <p>
    Solid rubber tires and the long spokes of the large front wheel provided
    a much smoother ride than its predecessor.
  </p>
</div>
```


Floating: stacking order

```
<style>
  body {
    width: 760px;
    font-family: Arial, Verdana, sans-serif;
    color: ■ #665544;
  }
  /*
  div {
    border: 1px solid #665544;
  } */
  p {
    width: 230px;
    height: 125px;
    float: left;
    margin: 5px;
    padding: 5px;
    background-color: □ #efefef;
  }
</style>
```

The Evolution of the Bicycle

In 1817 Baron von Drais invented a walking machine that would help him get around the royal gardens faster.

The device known as the Draisienne (or "hobby horse") was made of wood, and propelled by pushing your feet on the ground in a gliding movement.

It was not seen a suitable for any place other than a well maintained pathway.

In 1865, the velocipede (meaning "fast foot") attached pedals to the front wheel, but its wooden structure made it extremely uncomfortable.

In 1870 the first all-metal machine appeared. The pedals were attached directly to the front wheel.

Solid rubber tires and the long spokes of the large front wheel provided a much smoother ride than its predecessor.

Floating: stacking order

```
p {  
  width: 230px;  
  /* height: 125px; */  
  float: left;  
  margin: 5px;  
  padding: 5px;  
  background-color: #efefef;  
}  
</style>
```

The Evolution of the Bicycle

In 1817 Baron von Drais invented a walking machine that would help him get around the royal gardens faster.

The device known as the Draisienne (or "hobby horse") was made of wood, and propelled by pushing your feet on the ground in a gliding movement.

It was not seen a suitable for any place other than a well maintained pathway.

In 1865, the velocipede (meaning "fast foot") attached pedals to the front wheel, but its wooden structure made it extremely uncomfortable.

In 1870 the first all-metal machine appeared. The pedals were attached directly to the front wheel.

Solid rubber tires and the long spokes of the large front wheel provided a much smoother ride than its predecessor.

Floating: stacking order

- ▶ Een oplossing voor dit probleem kan zijn om het element dat vastzit, de float te laten clearen met `clear: left;`
- ▶ Om terug te keren naar de normale flow van de pagina moeten we de **floatende elementen clearen**. De **clear** eigenschap kan volgende waarden hebben: **left** – **right** – **both** – **none**.
- ▶ De eigenschap clear betekent: plaats het element (vlottend of niet) onder het voorafgaande floating element.

Floating: stacking order

```
p:nth-of-type(4) {  
  clear: left;  
}
```

The Evolution of the Bicycle

In 1817 Baron von Drais invented a walking machine that would help him get around the royal gardens faster.

The device know as the Draisienne (or "hobby horse") was made of wood, and propelled by pushing your feed on the ground in a gliding movement.

It was not seen a suitable for any place other than a well maintained pathway.

In 1865, the velocipede (meaning "fast foot") attached pedals to the front wheel, but its wooden structure made it extremely uncomfortable.

In 1870 the first all-metal machine appeared. The pedals were attached directly to the front wheel.

Solid rubber tires and the long spokes of the large front wheel provided a much smoother ride than its predecessor.

Clearing floats

- ▶ **Problem:** de background en border van parent loopt niet tot onder floating elementen.

The Evolution of the Bicycle

In 1817 Baron von Drais invented a walking machine that would help him get around the royal gardens faster.

The device know as the Draisienne (or "hobby horse") was made of wood, and propelled by pushing your feed on the ground in a gliding movement.

It was not seen a suitable for any place other than a well maintained pathway.

In 1865, the velocipede (meaning "fast foot") attached pedals to the front wheel, but its wooden structure made it extremely uncomfortable.

In 1870 the first all-metal machine appeared. The pedals were attached directly to the front wheel.

Solid rubber tires and the long spokes of the large front wheel provided a much smoother ride than its predecessor.

```
div {  
  border: 1px solid #665544;  
  background-color: #445566;  
}
```

- ▶ **Problem:** Of indien binnen een container alle elementen floated zijn, dan is het alsof de bevattende container geen hoogte of breedte meer heeft, dit is gekend onder de term: **container collapse.**

Clearing floats

- ▶ **Probleem:** De background en border van parent loopt niet tot onder floating elementen
 - **Oplossing 1 :** voeg extra leeg element toe

```
<p>
  Solid rubber tires and the long
  a much smoother ride than its pr
</p>
<div></div>
</div>
```

```
p:nth-of-type(4) {
  clear: left;
}
div > div {
  clear:both; /* left is ook goed*
}
</style>
```

The Evolution of the Bicycle

In 1817 Baron von Drais invented a walking machine that would help him get around the royal gardens faster.

The device know as the Draisienne (or "hobby horse") was made of wood, and propelled by pushing your feed on the ground in a gliding movement.

It was not seen a suitable for any place other than a well maintained pathway.

In 1865, the velocipede (meaning "fast foot") attached pedals to the front wheel, but its wooden structure made it extremely uncomfortable.

In 1870 the first all-metal machine appeared. The pedals were attached directly to the front wheel.

Solid rubber tires and the long spokes of the large front wheel provided a much smoother ride than its predecessor.

Clearing floats

- ▶ **Probleem:** De background en border van parent lopen niet tot onder floating elementen
 - **Oplossing 2:** display: flow-root toepassen op parent-element.

```
div.clearfix {  
  display: flow-root;  
}
```

- **Oplossing 3 (verouderd):** clearfix toe te passen op parent element
 - <https://css-tricks.com/snippets/css/clear-fix/>



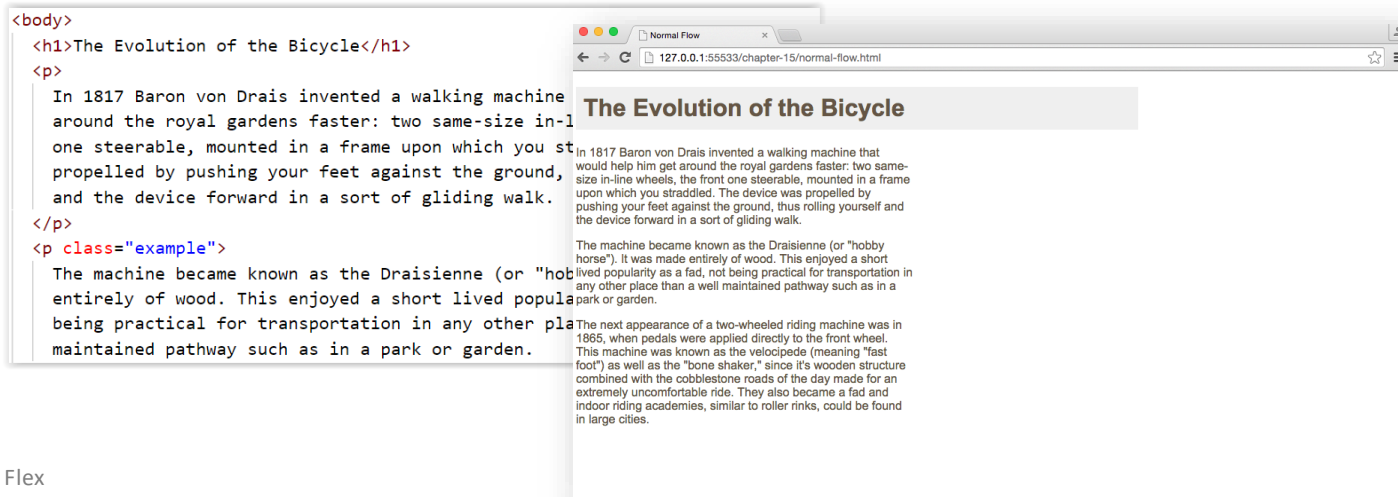
Position

Layout: met positionering

Om de normale flow te doorbreken bekijken we nu:

- relatieve positionering: position: relative
- absolute positionering: position: absolute
- fixed positionering: position: fixed

De normale flow ziet er als volgt uit (position: static)



The image shows a browser window displaying a page titled "The Evolution of the Bicycle". The page content is as follows:

```
<body>  
<h1>The Evolution of the Bicycle</h1>  
<p>  
  In 1817 Baron von Drais invented a walking machine  
  around the royal gardens faster: two same-size in-  
  line steerable, mounted in a frame upon which you st  
  propelled by pushing your feet against the ground,  
  and the device forward in a sort of gliding walk.  
</p>  
<p class="example">  
  The machine became known as the Draisienne (or "hob  
  entirely of wood. This enjoyed a short lived popula  
  being practical for transportation in any other pla  
  maintained pathway such as in a park or garden.
```

The browser window shows the rendered output of this code. The title "The Evolution of the Bicycle" is centered at the top. Below it, the first paragraph is displayed. The second paragraph, which has the class "example", is also displayed. The browser's address bar shows the URL "127.0.0.1:55533/chapter-15/normal-flow.html".

Relatieve positionering

- ▶ **position: relative**
- ▶ Relatieve positionering verplaatst het element **relatief tov zijn positie in de normale flow**. Dit heeft geen invloed op de positie van de andere elementen. Deze behouden hun normale positie.
- ▶ Offset wordt bepaald door:
 - verticale verplaatsing: **top – bottom**
 - horizontale verplaatsing: **left – right**

```
body {  
  width: 750px;  
  font-family: Arial, Verdana, sans-serif;  
  color: #665544;  
}  
  
p {  
  width: 450px;  
}  
  
p.example {  
  position: relative;  
  top: 275px;  
  left: 100px;  
}
```

The Evolution of the Bicycle

In 1817 Baron von Drais invented a walking machine that would help him get around the royal gardens faster: two same-size in-line wheels, the front one steerable, mounted in a frame upon which you straddled. The device was propelled by pushing your feet against the ground, thus rolling yourself and the device forward in a sort of gliding walk.

The next appearance of a two-wheeled riding machine was in 1865, when pedals were applied directly to the front wheel. This machine was known as the velocipede (meaning "fast foot") as well as the "bone shaker," since its wooden structure combined with the cobblestone roads of the day made for an extremely uncomfortable ride. They also became a fad and indoor riding academies, similar to roller rinks, could be found in large cities.

The machine became known as the Draisienne (or "hobby horse"). It was made entirely of wood. This enjoyed a short lived popularity as a fad, not being practical for transportation in any other place than a well maintained pathway such as in a park or garden.

Absolute positionering

- ▶ **position: absolute**
- ▶ Absolute positionering verplaatst het element **relatief tov zijn eerste niet static parent** element, of het body element indien alle parent elementen static zijn.
Voor de overige elementen is het alsof dit element nooit aanwezig is geweest in de normale flow.
Ze nemen dus posities in zonder rekening te houden met het absolute gepositioneerde element. Bij het scrollen beweegt het element mee.
- ▶ Offset (px - % - em) wordt bepaald door:
 - verticale verplaatsing: **top – bottom**
 - horizontale verplaatsing: **left – right**

Absolute positioning

```
body {  
  width: 750px;  
  font-family: Arial, Verdana, sans-serif;  
  color: #665544;}  
h1 {  
  position: absolute;  
  top: 0px;  
  left: 500px;  
  width: 250px;}  
p {  
  width: 450px;}
```

← → ↻ 127.0.0.1:51307/chapter-15/position-absolute.html

In 1817 Baron von Drais invented a walking machine that would help him get around the royal gardens faster: two same-size in-line wheels, the front one steerable, mounted in a frame upon which you straddled. The device was propelled by pushing your feet against the ground, thus rolling yourself and the device forward in a sort of gliding walk.

The Evolution of the Bicycle

The machine became known as the Draisienne (or "hobby horse"). It was made entirely of wood. This enjoyed a short lived popularity as a fad, not being practical for transportation in any other place than a well maintained pathway such as in a park or garden.

The next appearance of a two-wheeled riding machine was in 1865, when pedals were applied directly to the front wheel. This machine was known as the velocipede (meaning "fast foot") as well as the "bone shaker," since its wooden structure combined with the cobblestone roads of the day made for an extremely uncomfortable ride. They also became a fad and

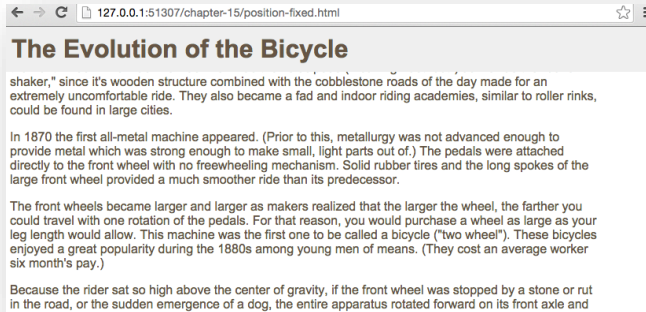
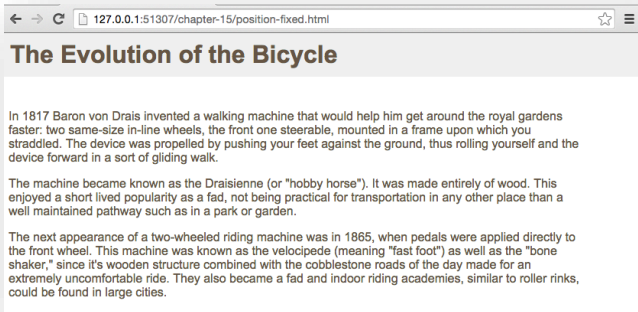
Vaste positionering

- ▶ **position: fixed**
- ▶ Vaste positionering verplaatst het element **relatief tov het browser venster**. Voor de overige elementen is het alsof dit element nooit aanwezig is geweest in de normale flow. Ze nemen dus posities in zonder rekening te houden met het vast gepositioneerde element. Bij het scrollen beweegt het element NIET mee. Wordt gedaan bij menubalken die niet mogen meescrollen.
- ▶ Offset (px - % - em) wordt bepaald door:
 - verticale verplaatsing: **top – bottom**
 - horizontale verplaatsing: **left – right**

Vaste positionering

```
body {
  width: 750px;
  font-family: Arial, Verdana, sans-serif;
  color: #665544;}
h1 {
  position: fixed;
  top: 0px;
  left: 0px;
  padding: 10px;
  margin: 0px;
  width: 100%;
  background-color: #efefef;}
p.example {
  margin-top: 100px;}
```

Header blijft vast bij het scrollen.



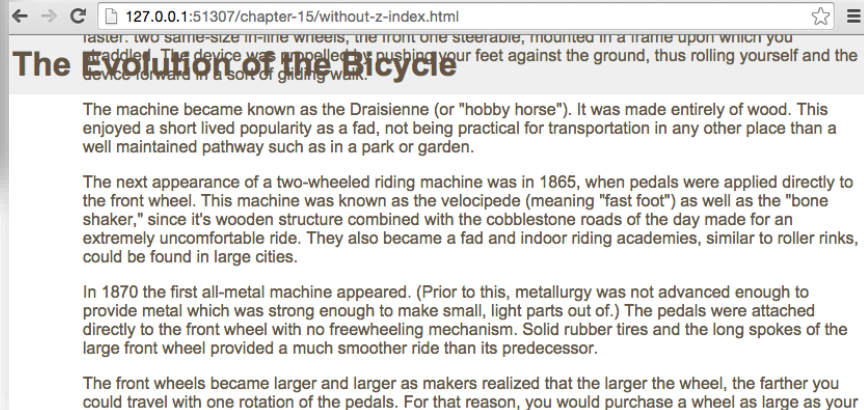
Positionering: z-index

- ▶ Indien de positionering van elementen gewijzigd worden, kan het zijn dat elementen gaan overlappen. De volgorde van de elementen in de html pagina bepaalt welke bovenaan staat: het bovenste element zit steeds onder een element daaronder (stapelen van dozen, te beginnen met het eerste element)
- ▶ Deze volgorde kan gewijzigd worden door de **z-index**. de mogelijke waarde is een geheel getal. Hoe hoger de waarde, hoe hoger op de stapel.

Positioning: z-index

▶ zonder z-index

```
h1 {  
  position: fixed;  
  top: 0px;  
  left: 0px;  
  margin: 0px;  
  padding: 10px;  
  width: 100%;  
  background-color: #efefef;}  
  
p {  
  position: relative;  
  top: 70px;  
  left: 70px;}
```



Positionering: z-index

▶ met z-index

```
color: #0000FF;
h1 {
  position: fixed;
  top: 0px;
  left: 0px;
  margin: 0px;
  padding: 10px;
  width: 100%;
  background-color: #efefef;
  z-index: 10;}
p {
  position: relative;
  top: 70px;
  left: 70px;}
```

