Web Essentials

Hoofdstuk 6

Basis structuur en positionering

DE HOGESCHOOL MET HET NETWERK

Elfde-Liniestraat 24, 3500 Hasselt, www.pxl.be

Vroeger en nu

Oude manier van positioneren

- Vroeger werden webpagina's ingedeeld met behulp van tabellen.

- Nadelen:
  - Er is veel (extra) HTML-code nodig
  - Er zijn veel minder mogelijkheden omdat alles in een tabelvorm moet passen
  - ...

Nieuwe manier van positioneren

- Tegenwoordig worden layers gebruikt om te komen tot de juiste positionering van de elementen op een webpagina.

- Deze techniek positioneert elementen zonder veel interactie van of met andere elementen.

In dit hoofdstuk

- Herhalen en verdiepen we enkele belangrijke onderdelen uit de voorgaande hoofdstukken.
- Gaan we de standaardopmaak van de webbrowser leren verwijderen om vanaf nul onze opmaak volledig zelf te bepalen.
- In een later hoofdstuk leren we enkele speciale manieren en mogelijkheden van positionering.

De structuurelementen

Standaard structuurelementen

- Structuurelementen kunnen ons helpen om de indeling van een webpagina vast te leggen.

- Vanaf nu is het gebruik van structuurelementen op iedere webpagina verplicht!

Standaard structuurelementen

- Hoofding `<header>…</header>`
- Navigatieblok `<nav>…</nav>`
- Hoofdgedeelte `<main>…</main>`
- Sectie (deel van een geheel) `<section>…</section>`
- Artikel (losstaand geheel) `<article>…</article>`
- Voetnoot of voettekst `<footer>…</footer>`
- Randinformatie `<aside>…</aside>`
- Figuren `<figure>…</figure>`

Divisies

- Een -element gebruik we voor blokelementen zonder betekenis.
- Bij een -element geven wij bijna altijd een class- of id-attribuut mee.

```
<div id=”geenspecifiekebetekenis”>
    ...
</div>
```

Stijleigenschap display

Blokelementen en inline-elementen

- *Blokelementen* nemen altijd een nieuwe regel in beslag en maken gebruik van de volledige breedte van het scherm.

```
Dit is een blok element.
```

- *Inline-elementen* worden in de huidige regel getoond en zijn even breed als de inhoud van het element.

```
Dit is een inline element.
```

Stijleigenschap Display

- Via de stijleigenschap ‘display’ kunnen we het standaard weergavegedrag (blok/inline) van een element aanpassen.

Waarde	Uitleg
block	Het element wordt een blok-element in een blokcontainer en neemt een volledig nieuwe regel in beslag.
inline-block	Het element blijft een inline-element, maar kan zonder problemen met blokcontainer stijleigenschappen overweg.
inline	Het element wordt een inline-element en wordt in de huidige regel getoond.
none	Het element wordt onzichtbaar en neemt geen plaats meer in op de webpagina.

Stijleigenschap box-sizing

Box-sizing

- Bij het box-model hebben we gezien dat we een element een padding, border en margin kunnen geven. Deze waardes tellen allemaal mee voor de totale breedte of hoogte van het element.

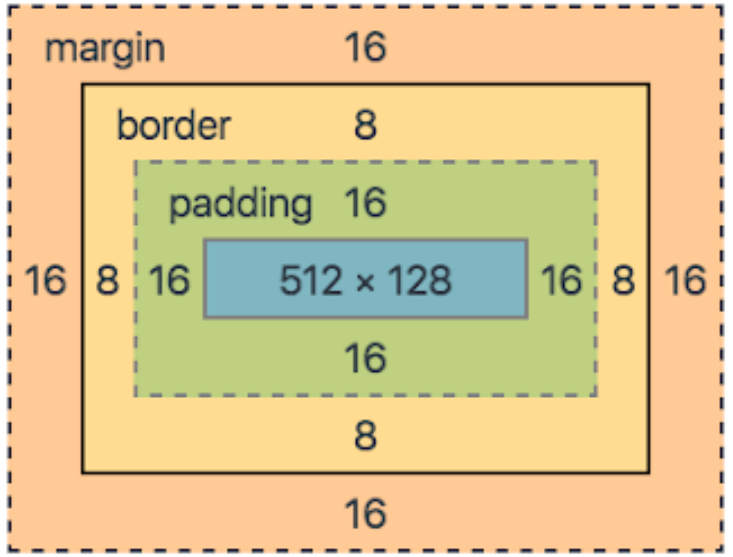
- Via stijleigenschap ‘box-sizing’ en waarde ‘border-box’ kunnen we van deze standaard optelsom afwijken door in de breedte of hoogte van het element, ook de padding en de border op te nemen.

Origineel optelsom

Eigenlijke breedte = breedte element + padding + border + margin

*592 = 512 + (16*2) + (8*2) + (16*2)*

Eigenlijke hoogte = hoogte element + padding + border + margin
*208 = 128 + (16*2) + (8*2) + (16*2)*



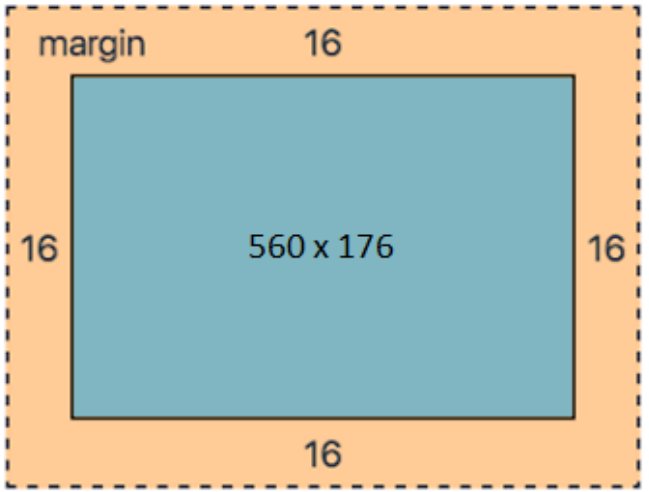
Optelsom met ‘box-sizing: border-box;’

Eigenlijke breedte = breedte element + margin

*592 = 560 + (16*2)*

Eigenlijke hoogte = hoogte element + margin

*208 = 176 + (16*2)*



Voordelen

- We maken minder fouten, want de optelsom is eenvoudiger.

- We kunnen met meerdere meeteenheden door elkaar werken.
  - Bvb. Het element is 90% breed en heeft een margin van 5%.
    - Het element heeft een volle rand van 1px aan alle zijdes
    - Het element heeft een padding van 20px aan alle zijdes

Standaardopmaak resetten

Standaardopmaak resetten

- Vaak willen we afwijken van de standaardopmaak gegeven door de webbrowser aan een webpagina.
- Door de standaardopmaak te resetten, is het veel eenvoudiger om de opmaak volledig naar onze wensen (of het opgegeven design) te maken.
- Dit gaan we vanaf nu ALTIJD doen!

```
* {
        margin: 0;
        padding: 0;
        box-sizing: border-box;
}
```

```
// We verwijderen op ALLE elementen de standaard margin, de standaard padding en we
   berekenen de hoogtes en de breedte via stijleigenschap box-sizing waarde ‘border-box’.
```

Flexbox

Inleiding

- Met CSS3 Flexbox kunnen we makkelijker een responsive design voorzien, zonder gebruik te maken van zwevende opmaak of positionering.

- Wanneer we aan de slag gaan met CSS3 Flexbox, moeten we minstens volgende onderdelen voorzien:
  - Een Flex-container
  - Flex-items in de flex-container

Flex-container en flex-item

```
<main class=”flex-container”>
      <article id=”deel1”></article>
      <article id=”deel2”></article>
      <article id=”deel3”></article>
</main>
```

Flex-container

Flex-container stijleigenschappen

- display
- flex-direction
- flex-wrap
- flex-flow
- justify-content
- align-items
- align-content

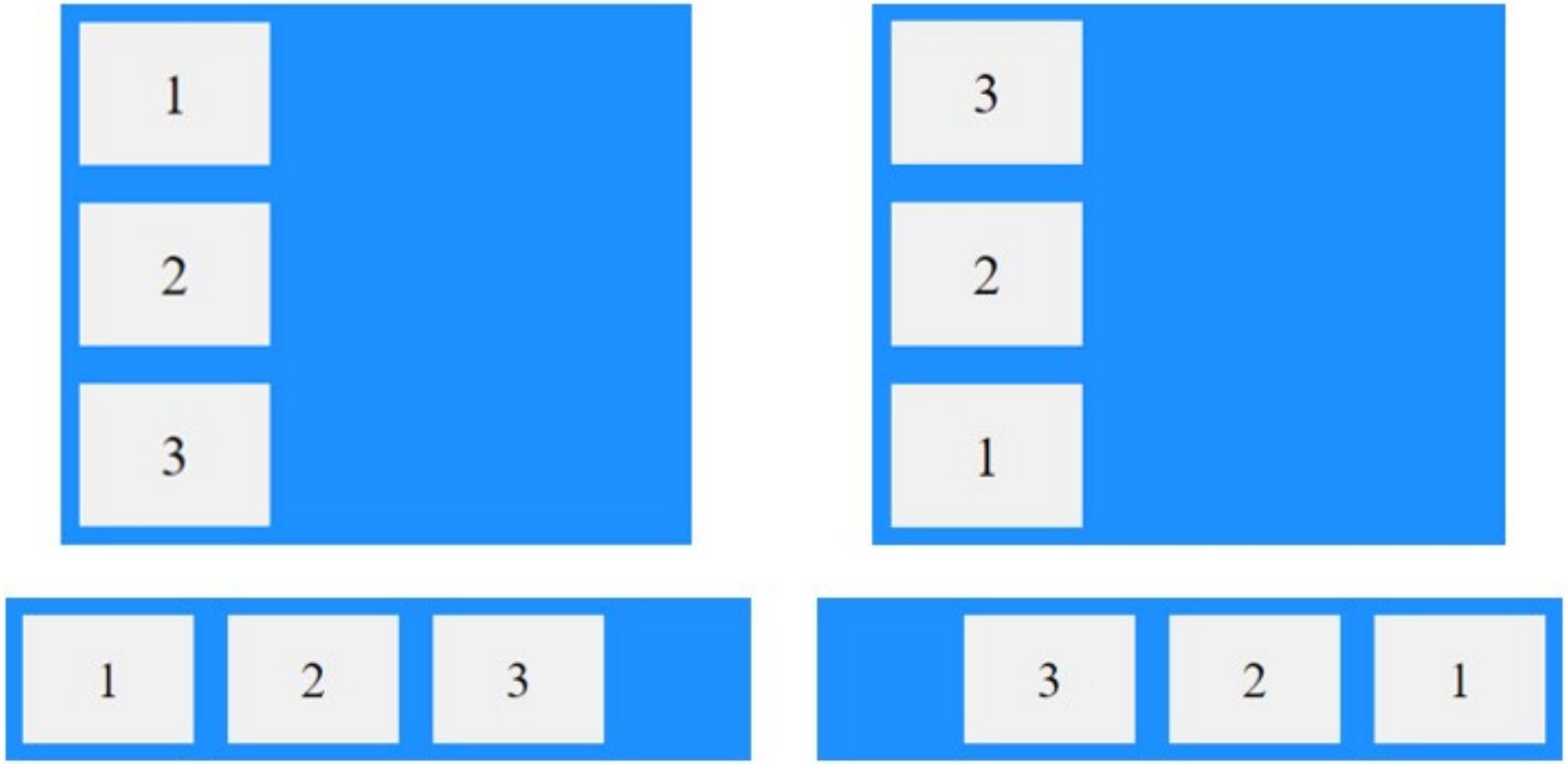
Display

- Via de display eigenschap kunnen we ervoor zorgen dat de flex-container flexibel wordt.

```
.flex-container {
    display: flex;
}
```

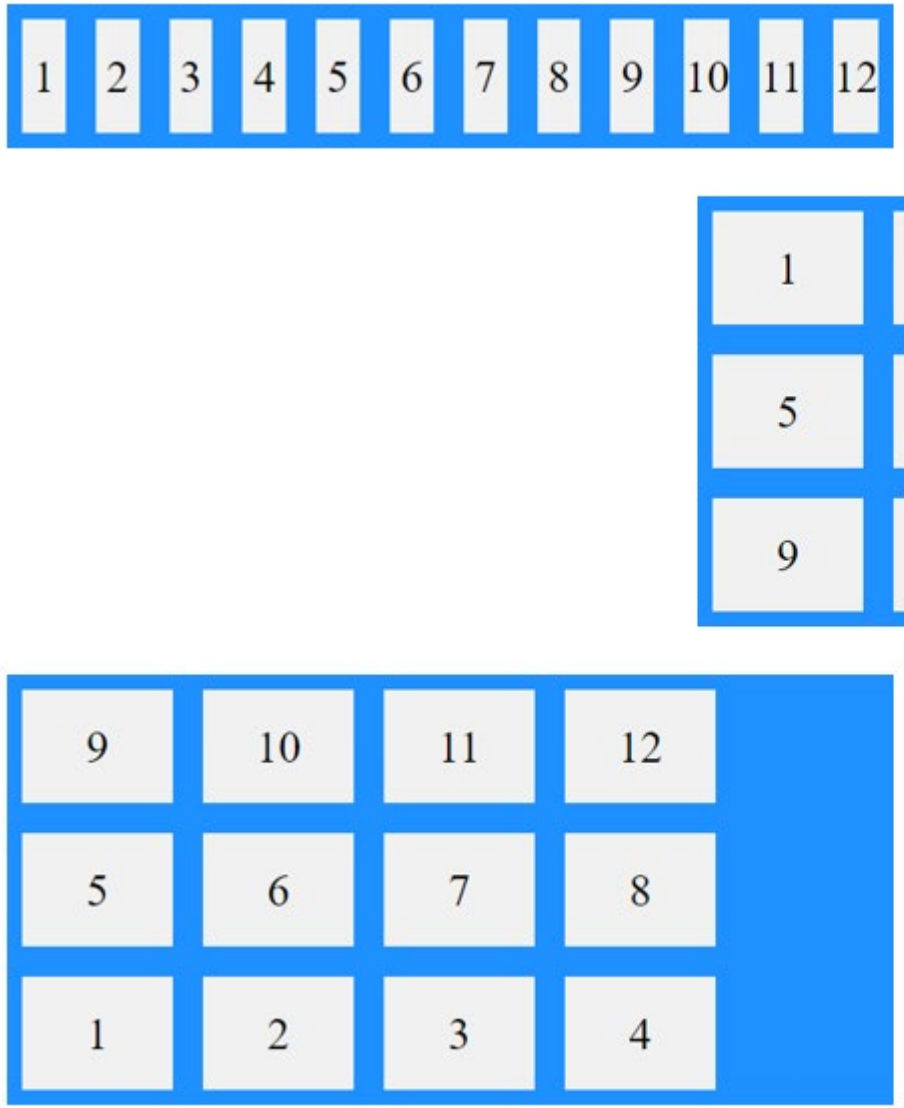
Flex-direction

- Via de flex-direction eigenschap kunnen we de richting van de flex-items instellen. Volgende waarden zijn hierbij mogelijk:
  - column
  - column-reverse
  - row
  - row-reverse


1
2
3
3
2
1
1
2
3
3
2
1

Flex-wrap

- Via de flex-wrap eigenschap kunnen we aangeven of alle flex-items op één of meerdere rijen of kolommen moeten komen te staan. Volgende waarden zijn hierbij mogelijk:
  - nowrap
  - wrap
  - wrap-reverse


1 2 3 4 5 6 7 8 9 10 11 12
1 2 3 4
5 6 7 8
9 10 11 12
9 10 11 12
5 6 7 8
1 2 3 4

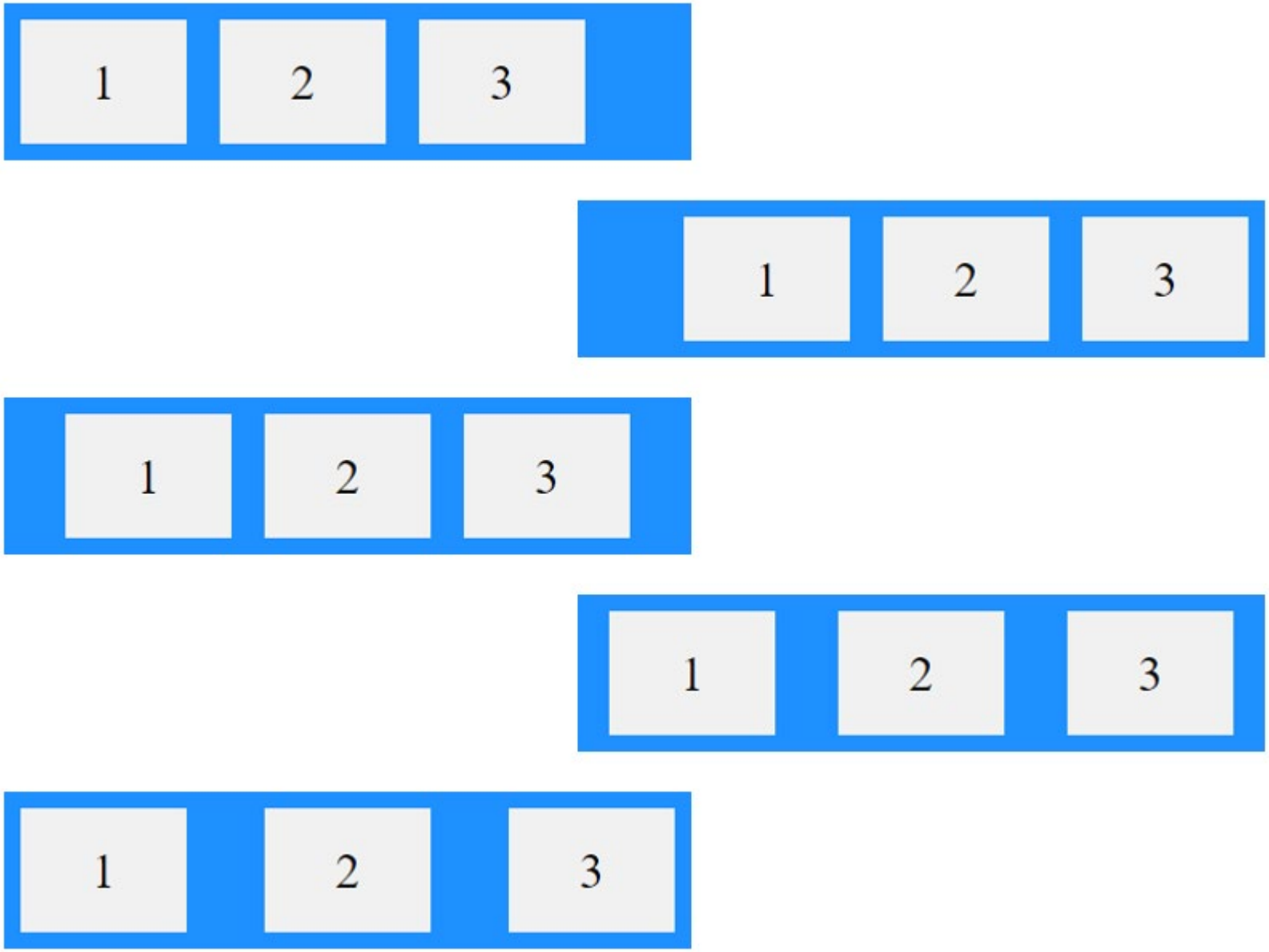
Flex-flow

- Via de flex-flow eigenschap kunnen we de flex-direction en flex-wrap op basis van één regel instellen.

```
.flex-container {
    display: flex;
    flex-flow: row wrap;
}
```

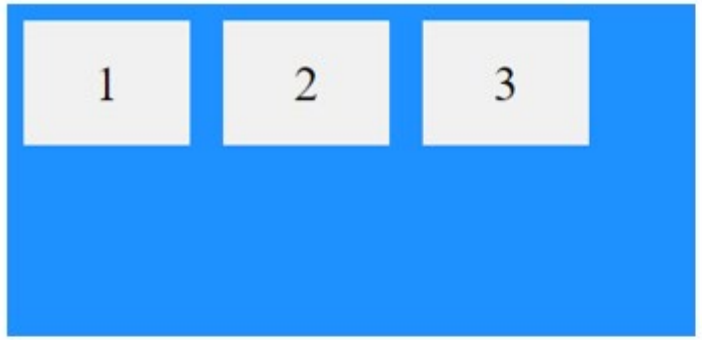
Justify-content

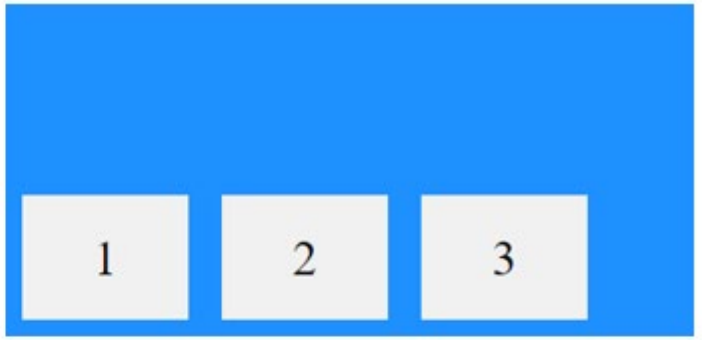
- Via de justify-content eigenschap kunnen we de lay-out van de flex-items in de container bepalen. Volgende waarden zijn hierbij mogelijk:
  - flex-start
  - flex-end
  - center
  - space-around
  - space-between
  - space-evenly

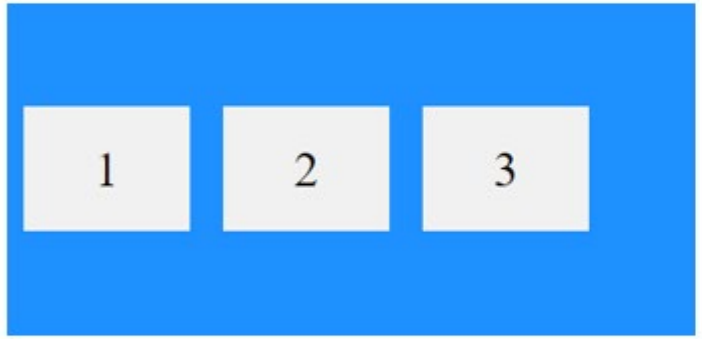

1
2
3
1
2
3
1
2
3
1
2
3
1
2
3

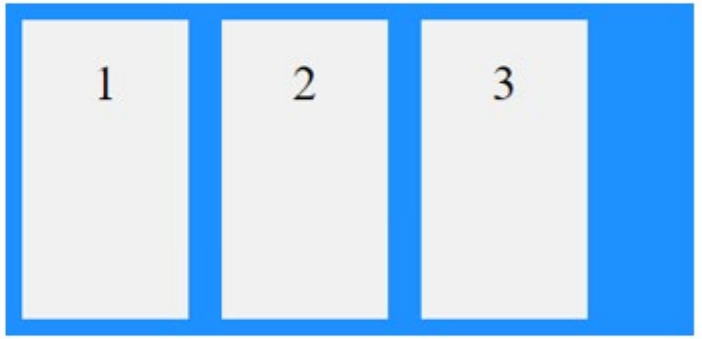
Align-items

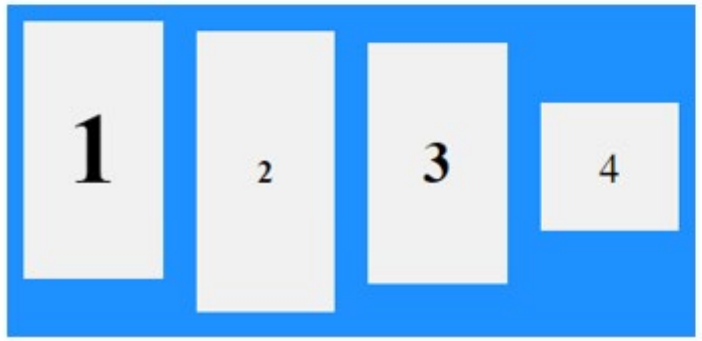
- Via de align-items eigenschap kunnen we de verticale uitlijning van de flex-items instellen. Volgende waarden zijn hierbij mogelijk:
  - flex-start
  - flex-end
  - center
  - stretch
  - baseline


1
2
3


1
2
3

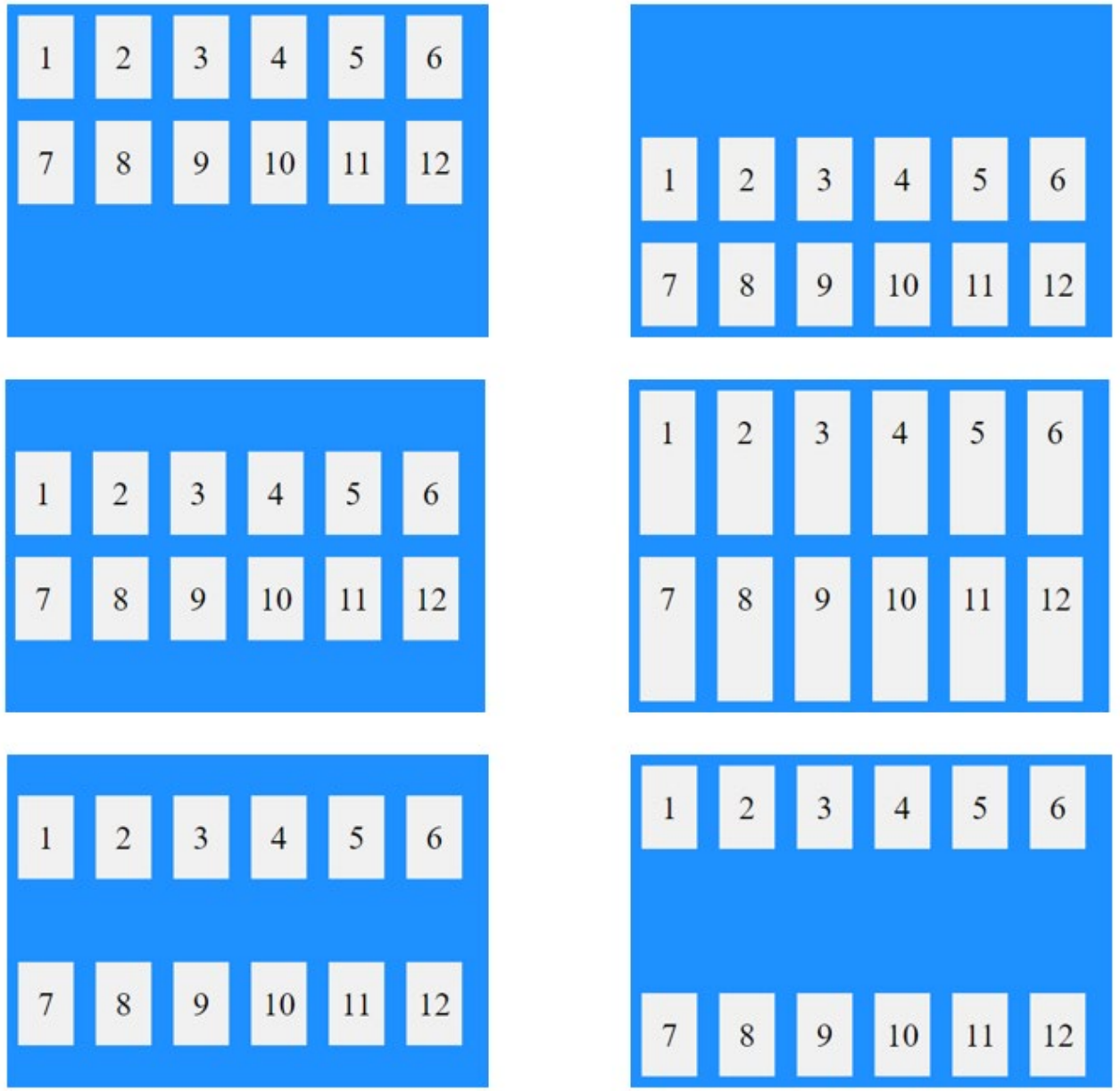

1
2
3


1
2
3


1
2
3
4

Align-content

- Via de align-content eigenschap kunnen we de regels met flex-items verticaal uitlijnen. Volgende waarden zijn hierbij mogelijk:
  - flex-start
  - flex-end
  - center
  - stretch
  - space-around
  - space-between


1 2 3 4 5 6
7 8 9 10 11 12
1 2 3 4 5 6
7 8 9 10 11 12
1 2 3 4 5 6
7 8 9 10 11 12
1 2 3 4 5 6
7 8 9 10 11 12
1 2 3 4 5 6
7 8 9 10 11 12
1 2 3 4 5 6
7 8 9 10 11 12

Flex-items

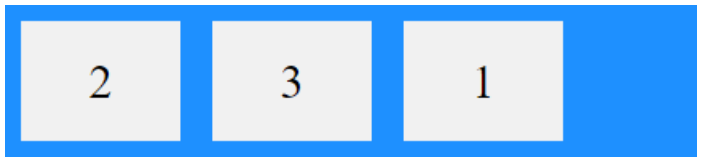
Flex-items stijleigenschappen

- order
- flex-grow
- flex-shrink
- flex-basis
- flex
- align-self

Order

- Via de order eigenschap kunnen we de volgorde van de flex-items in de flex-container aanpassen.

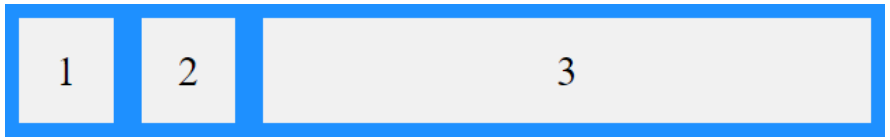
```
#deel1 {
      order: 3;
}
#deel2 {
      order: 1;
}
#deel3 {
      order: 2;
}
```



Flex-grow

- Via de flex-grow eigenschap kunnen we aangeven hoeveel een flex-item relatief moet groeien tegenover de rest van de flex-items.

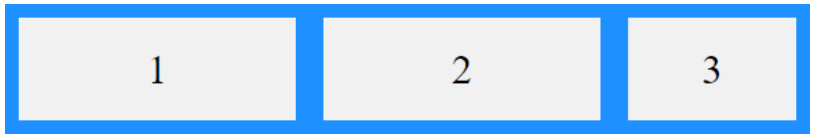
```
#deel1 {
      flex-grow: 1;
}
#deel2 {
      flex-grow: 1;
}
#deel3 {
      flex-grow: 8;
}
```



Flex-shrink

- De flex-shrink eigenschap is het tegenovergestelde van de flex-grow eigenschap. Via de flex-shrink eigenschap kunnen we aangeven hoeveel een flex-item relatief moet krimpen tegenover de rest van de flex-items.

```
#deel3 {
      flex-shrink: 2;
}
```



Flex-basis

- Via de flex-basis eigenschap kunnen we de initiële lengte van een flex-item vastleggen.

```
#deel3 {
      flex-basis: 200px;
}
```

Flex

- Via de flex eigenschap kunnen we de flex-grow, flex-shrink en flex-basis op basis van één regel instellen.

- Met deze verkorte notatie krijgen de niet-ingestelde eigenschappen automatisch de meest optimale waarden.

```
#deel3 {
     flex: 0 0 200px;
}
```

<flex-grow> <flex-shrink> <flex-basis>	kies volledig zelf wat je wenst voor het item op vlak van groeien, krimpen en initiële lengte.
initial	Verkorte instelling voor ‘0 1 auto’. Het item mag niet groeien, maar wel krimpen. De basisafmeting wordt bepaald door de ingestelde breedte/hoogte.
auto	Verkorte instelling voor ‘1 1 auto’. De beschikbare ruimte in de container wordt volledig benut. Groeien en krimpen gebeurt naar behoefte.
none	Verkorte instelling voor ‘0 0 auto’. Het item krijgt de waarde van de ingestelde breedte/hoogte en is niet flexibel. Er is geen groei en geen krimp.
<getal>	Verkorte instelling voor ‘<getal> 1 0’. Het item groeit met de vrije ruimte in de container. Het getal geeft het proportionele deel van de vrije ruimte aan. Als alle items hetzelfde getal hebben, krijgen ze een gelijk deel van de vrije ruimte.

Align-self

- Via de align-self eigenschap kan je de verticale uitlijning van één of meerdere flex-items apart aanpassen. De align-self eigenschap overschrijft de align-items eigenschap van de flex-container. Volgende waarden zijn hierbij mogelijk:
  - flex-start
  - flex-end
  - center
  - stretch
  - baseline

Align-self

```
#deel2 {
	align-self: flex-end;
}

#deel3 {
	align-self: center;
}
```