



## 5 FLEXBOX

*Flexbox is een toevoeging aan moderne browsers.*

*Het helpt je om 'Flex-items' uit te lijnen in een Flex-container, ongeacht de hoogte of breedte van de items.*

*1) Wat is een 'Flex-container'?*

*2) Wat is een 'Flex-item'?*

# VOORBEELD

## *eerste stappen met flexbox: container & items*

*We starten met een simpel voorbeeld.*

*We nemen een `<ul>` met daarin drie `<li>`'s.*

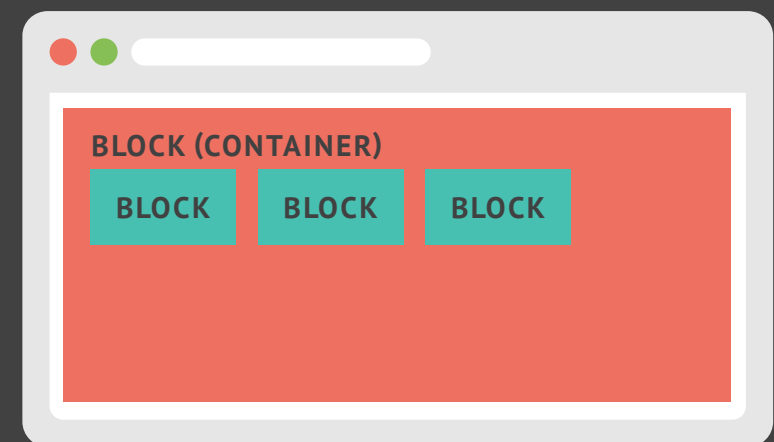
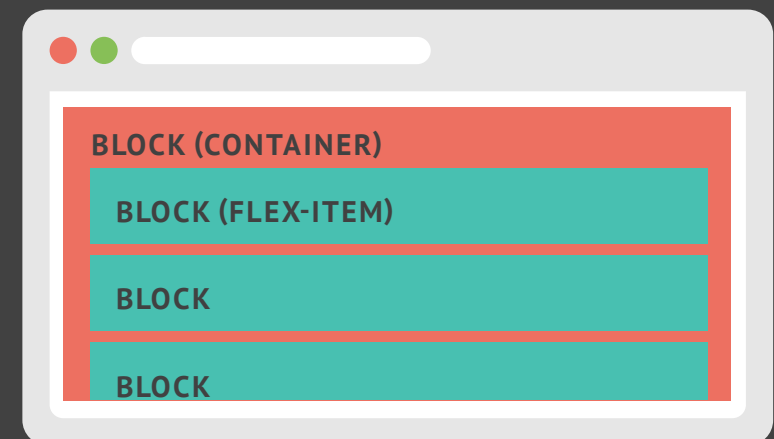
*Zowel de `<ul>` als de `<li>` zijn **block-level** elementen en stapelen dus **opelkaar**.*

*Door simpelweg de `<ul>` op: `display: flex` te zetten verandert de weergave.*

*In dit voorbeeld is de `<ul>` de **flex-container** geworden en de `<li>`'s automatisch de **flex-children**.*

*We zien nu dat de 'flex-children' automatisch in parent passen.*

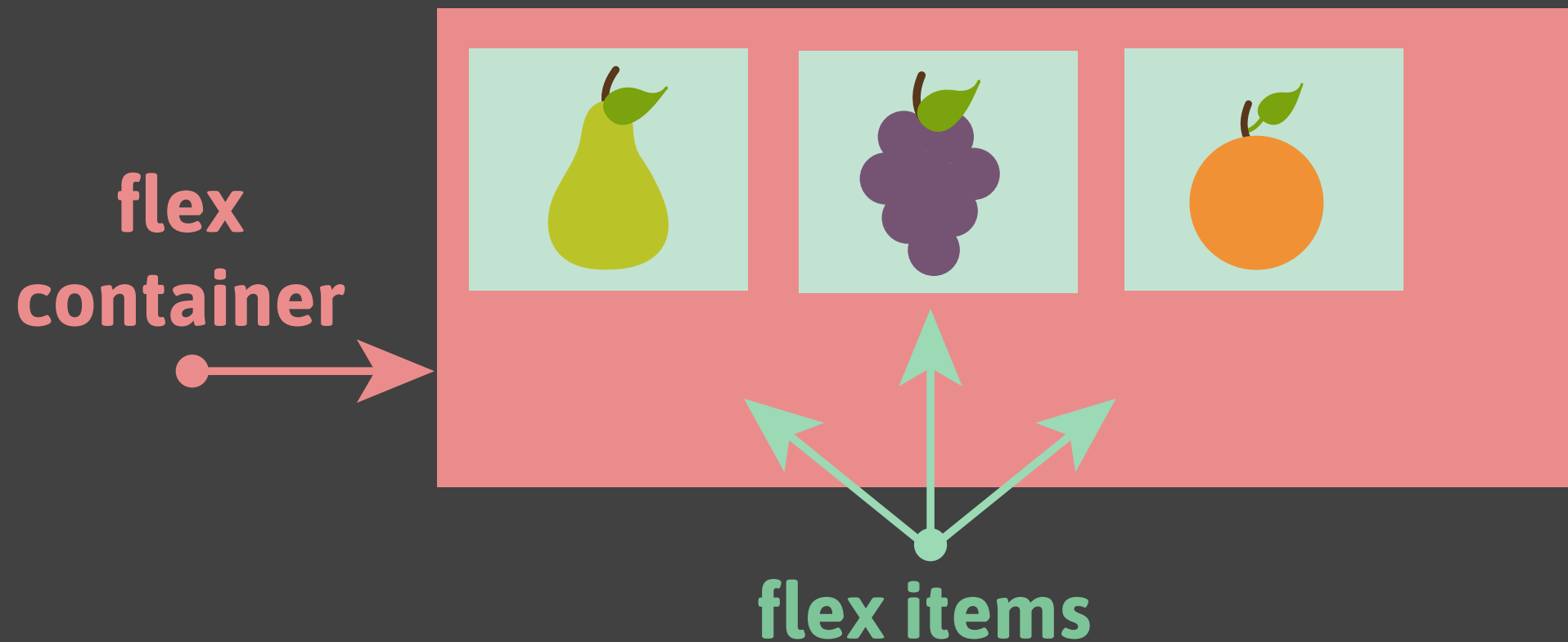
```
<ul>  
  <li>one</li>  
  <li>two</li>  
  <li>three</li>  
</ul>
```



<https://codepen.io/petervandenheuvel/pen/WaejdP?editors=1100>

# 1- FLEXCONTAINER

*en flex-items (direct children)*



*In het vorige voorbeeld maakten we gebruik van een **flex-container** en **flex-items**.  
Als we een element op 'display: flex' zetten, wordt dat automatisch een **flex-container**.*

*Alle directe kinderen worden dan '**flex-items**'.*

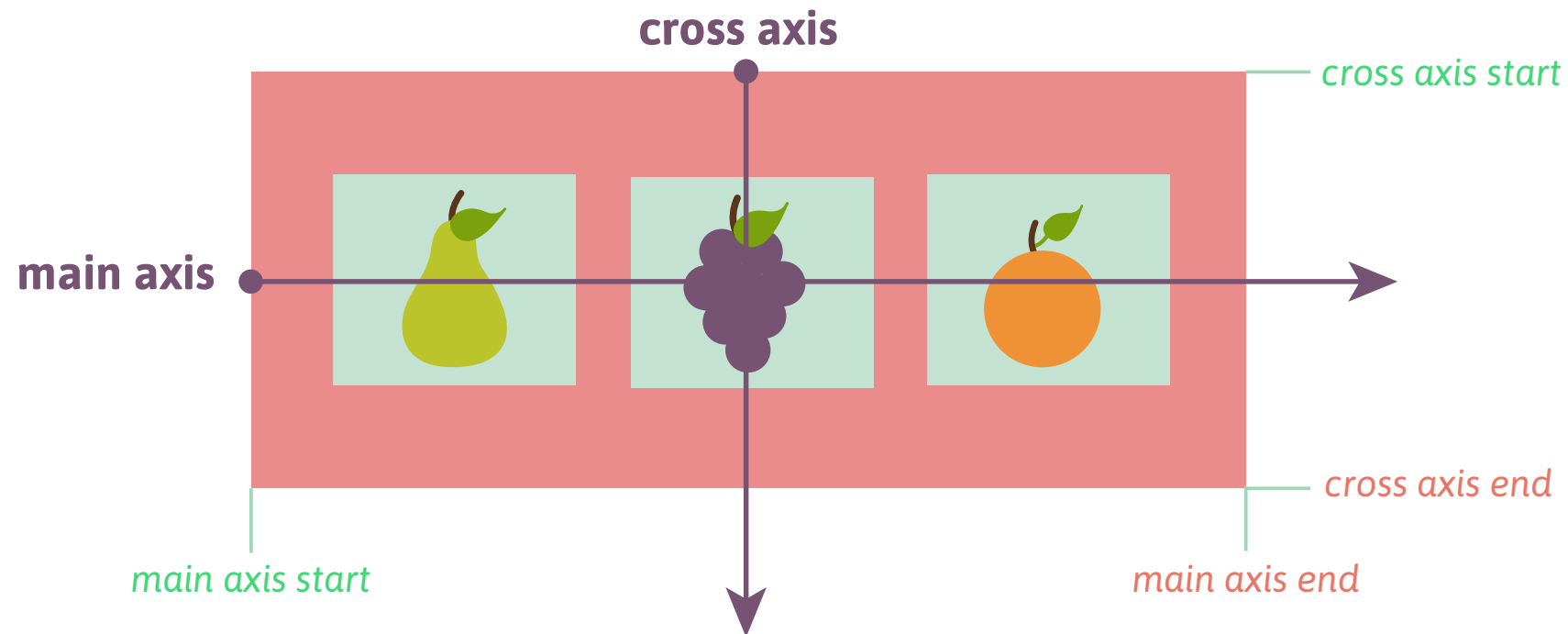
## A FLEXBOX - CONTAINER

---

*Welke **properties** kan ik gebruiken op de flex-container?*

# FLEX DIRECTION

*horizontaal of verticaal*

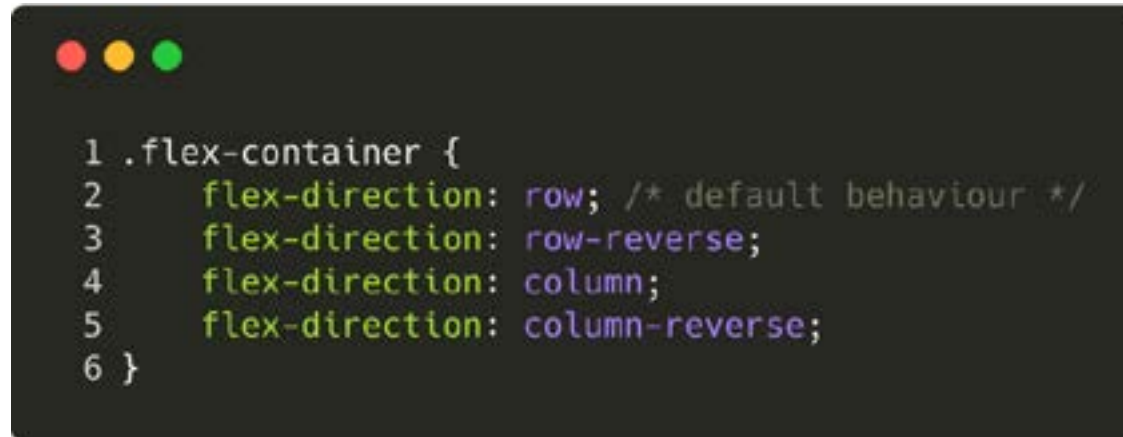


De '*flex-direction*' property geeft je de mogelijkheid om te kiezen hoe de *flex-children* geplaatst worden in de *flex-container*: Horizontaal, Verticaal of omgekeerd in beide richtingen.

- Men spreekt echter niet over horizontaal en verticaal maar over de **main-axis** en de **cross-axis**.
- Met *Flex-direction*: row of column draai je de assen om.

# FLEX DIRECTION

*Probeer deze eens uit op het vorige voorbeeld (de lijst) icm. 'align-content: flex-end;'*



```
1 .flex-container {  
2   flex-direction: row; /* default behaviour */  
3   flex-direction: row-reverse;  
4   flex-direction: column;  
5   flex-direction: column-reverse;  
6 }
```

## ***flex-direction: row***

*Alle items komen naast elkaar te staan.*

## ***flex-direction: row-reverse***

*Alle items komen naast elkaar te staan,  
in omgekeerde volgorde.*

## ***flex-direction: column***

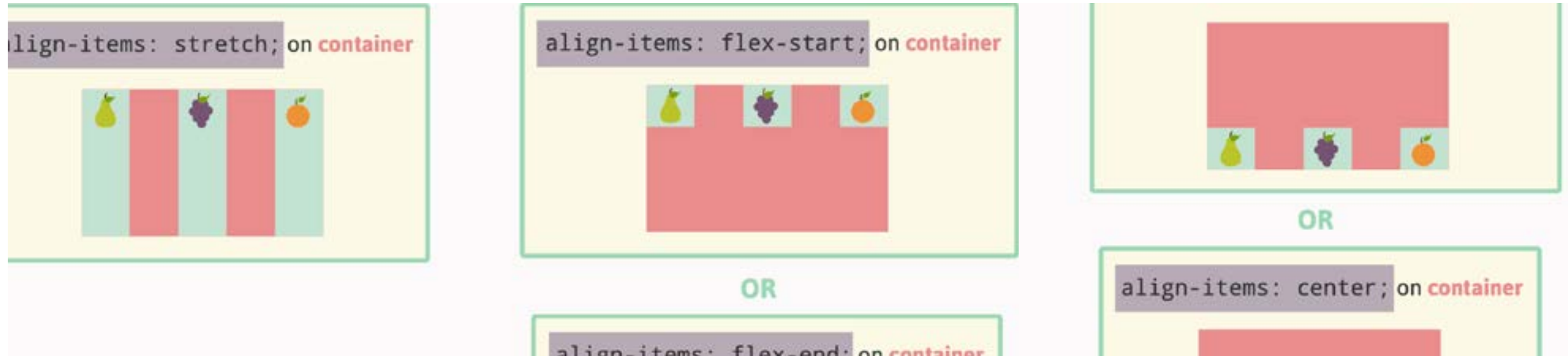
*Alle items komen opelkaar te staan.*

## ***flex-direction: column-reverse***

*Alle items komen opelkaar te staan,  
in omgekeerde volgorde*

# JUSTIFY-CONTENT

*uitlijnen van items langs de main-axis (horizontaal)*



***justify-content: flex-start (default)***

*om items aan het begin van de container.*

***justify-content: flex-end***

*om alle items aan het eind van de container te plaatsen.*

***justify-content: center***

*om alle items te centreren.*

***justify-content: space-between***

*om alle items evenveel tussenruimte te geven*

***justify-content: space-around***

*om alle items gelijk ruimte rondom te geven.*

<https://codepen.io/petervandenheuvel/pen/XxrgPV>

<https://www.freecodecamp.org/news/an-animated-guide-to-flexbox-d280cf6afc35/#property-3-justify-content>



# FLEX-WRAP

*als de flex-items niet meer op de rij passen, wat dan?*

```
1 .flex-container {  
2     flex-wrap: nowrap; /* default */  
3     flex-wrap: wrap;  
4     flex-wrap: wrap-reverse;  
5 }
```

## *flex-wrap (default)*

*Alle items komen naast elkaar te staan, en gaan door de container als er teveel zijn.*

## *flex-wrap: wrap*

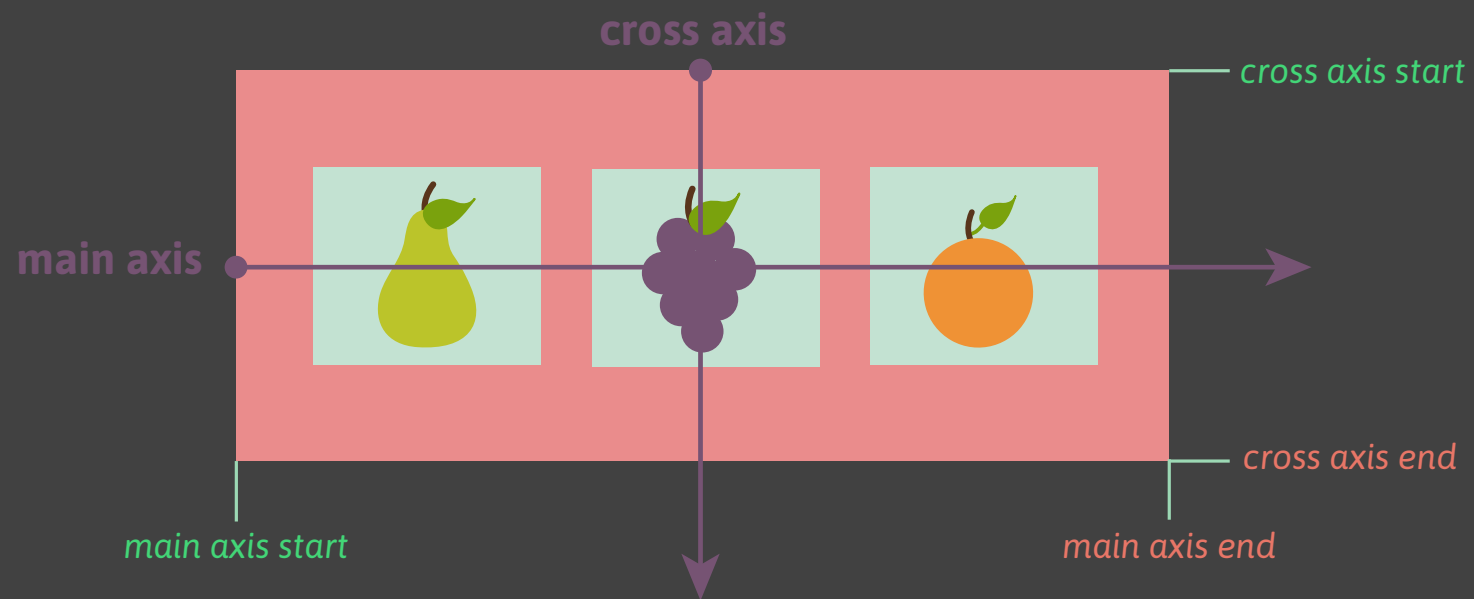
*Naar nieuwe regel als er te weinig ruimte is*

## *flex-wrap: wrap-reverse*

*Naar nieuwe regel, maar in omgekeerde volgorde.*

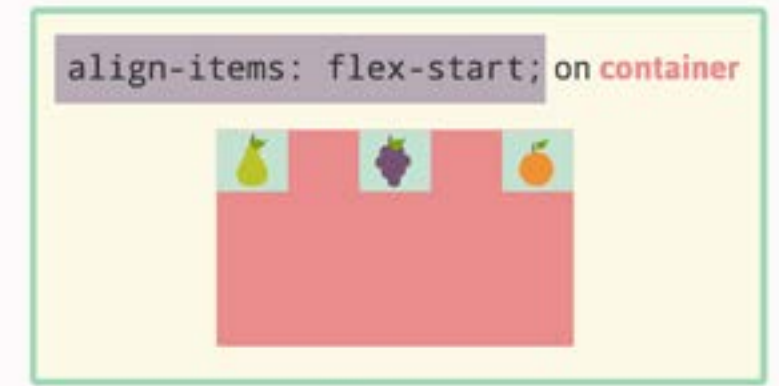
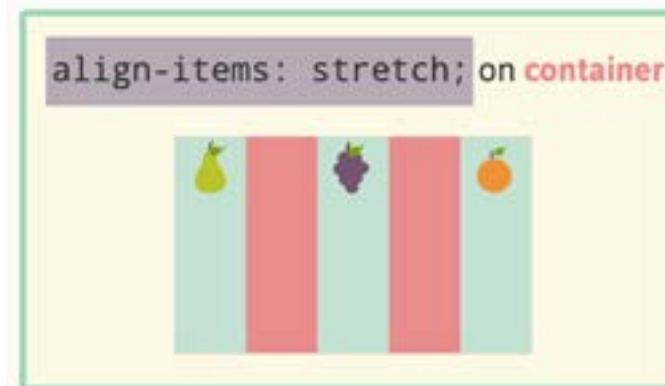
# VERTICAAL

*flex-container: cross-axis*

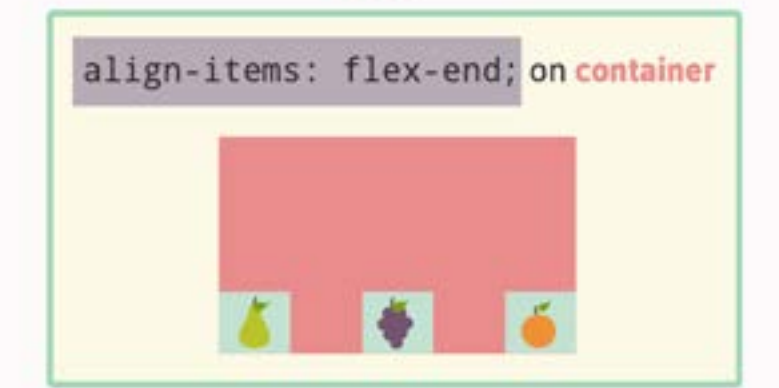
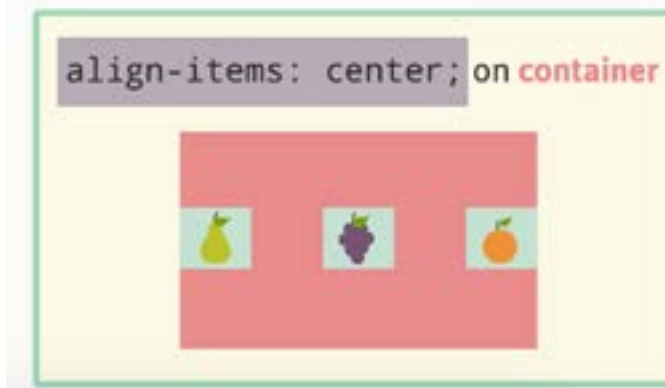


# ALIGN-ITEMS

## *Hoe verticaal uitlijnen?*



OR



***align-items: stretch (default)***

*de kinderen vullen de hoogte automatisch.*

***align-items: flex-start***

*Verticaal bovenaan*

***align-items: flex-end***

*Verticaal onderaan*

***align-items: center***

*Verticaal centeren; eindelijk eenvoudig zonder hacks..*

***align-items: baseline***

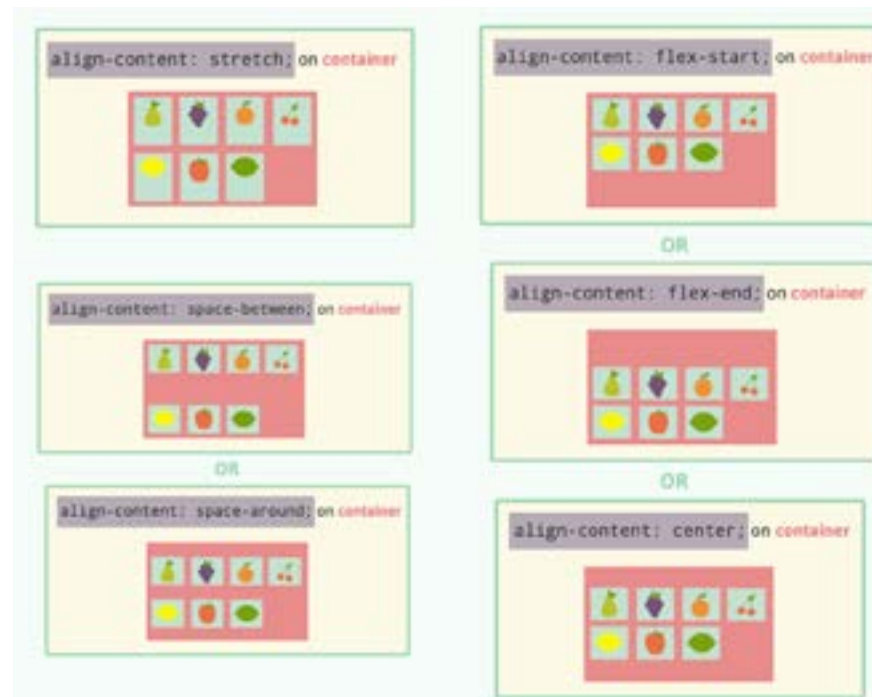
*Uitlijnen op de het font-basislijn stramien.*

# ALIGN-CONTENT

*Meerdere lijnen met inhoud verticaal verdelen*

LET OP:  
DIT WERKT ALLEEN ALS JE MEERDERE LIJNEN  
INHOUD HEBT.

ZET 'FLEX-WRAP' DUS OP: WRAP



## ***align-content: stretch (default)***

*Al de parent te groot is, en er vind wrapping plaats, kan je de items de hoogte automatisch laten opvullen.*

## ***align-content: flex-start***

*Opvullen vanaf boven*

## ***align-content: flex-end***

*Opvullen vanaf beneden*

## ***align-content: center***

*inhoud in het midden plaatsen*

## ***align-content: space-between***

*ruimte tussen de elementen*

## ***align-content: space-around***

*ruimte tussen de container en de items gelijk verdelen.*

## B FLEXBOX - FLEX-ITEMS

---

Welke *properties* kan ik gebruiken op de *flex-items*?

1 FLEX-GROW

2 FLEX-SHRINK

3 FLEX-BASIS

*shorthand: flex*

# FLEX-ITEM: GROW

*relatief vergroten items*

## 1 FLEX-GROW

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

.flex-item-1 {
  flex-grow: 0; /* default value */
}

.flex-item-2 {
  flex-grow: 1;
}

.flex-item-3 {
  flex-grow: 2;
}
```

*Geeft aan, hoeveel een item relatief mag groter worden als er meer ruimte is, ten opzichte van de andere flex-items.*

*Standaard is de waarde: '0', en dus kan je procentueel verhogen tov. andere items.*

<https://www.freecodecamp.org/news/even-more-about-how-flexbox-works-explained-in-big-colorful-animated-gifs-a5a74812b053/>

<https://codepen.io/petervandenheuvel/pen/pxzaGz?editors=1100>

# FLEX-ITEM: GROW

## voorbeeld

```
html, body {
  padding: 0; margin: 0;
}

input {
  font-size: 14px;
  font-family: Helvetica, sans-serif;
}

body {
  background-color: #BBB;
  font-family: Helvetica, sans-serif;
  padding-bottom: 100px;
}

h2, h3 {
  margin: 0 0 .75em 0;
}

/* first example */
.container {
  max-width: 750px;
  margin: 20px auto 0 auto;
  padding: 30px;
  background-color: #FFF;
}

.form-row {
  padding: 10px 0;
}

.form-row label {
  padding-right: 10px;
}

.form-row input {
}
```

```
<!-- first example -->
<div class="container">
  <form action="">
    <div class="form-row">
      <label for="name">Name:</label>
      <input type="text" id="name">
    </div>
    <div class="form-row">
      <label for="favColor">Favorite Color:</label>
      <input type="text" id="favColor">
    </div>
  </form>
</div>
```

## 1 FLEX-GROW

*Het uitvullen van label & input is lastig,  
Responsive, zeer lastig!*

Name:

Favorite Color:

# FLEX-ITEM: GROW

*oplossing*

## 1 FLEX-GROW

```
.form-row {  
  padding: 10px 0;  
  display: flex;  
}  
  
.form-row label {  
  padding-right: 10px;  
  flex-grow: 0 /* default */;  
}  
  
.form-row input {  
  flex-grow: 1;  
}
```

*In dit voorbeeld is het heel eenvoudig om de input 'de overige ruimte' te laten innemen, (zonder berekeningen).*

FLEX-GROW: 0   FLEX-GROW: 1

Name:

FLEX-GROW: 0   FLEX-GROW: 1

Favorite Color:



# NOG EEN VOORBEELD

*een kolommen-layout*



# FLEX-ITEM: GROW

## voorbeeld

### 1 FLEX-GROW

```
<!-- second example -->
<div class="column-layout">
  <div class="main-column">
    <h2>Main Column</h2>
    <p>Lorem ipsum dolor sit
    reiciendis debitis omnis fugi
    autem, voluptate modi delenit
    <p>Lorem ipsum dolor sit
    vitae numquam dolore sunt que
    Voluptatibus soluta, consequi
    <p>Lorem ipsum dolor sit
    reiciendis debitis omnis fugi
    autem, voluptate modi delenit
    <p>Lorem ipsum dolor sit
    vitae numquam dolore sunt que
  </div><!-- end of main-colu

  <div class="sidebar-one">
    <h3>Sidebar One</h3>
    <p>Lorem ipsum dolor sit
    dignissimos nulla, blanditiis
  </div><!-- end of sidebar-c

  <div class="sidebar-two">
    <h3>Sidebar Two</h3>
    <p>Lorem ipsum dolor sit amet,
    dignissimos nulla, blanditiis nesci
  </div><!-- end of sidebar-two -->
</div><!-- end of column-layout -->
```

```
/* second example */
.column-layout {
  max-width: 1300px;
  background-color: #FFF;
  margin: 40px auto 0 auto;
  line-height: 1.65;
  padding: 20px 50px;
  display: flex;
}

.main-column {
}

.sidebar-one {
}

.sidebar-two {
}
```

*Simpele kolommen layout:  
Start allemaal als block-level: op elkaar.*

#### Main Column

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ipsum voluptatibus soluta, consequuntur, reiciendis debitis omnis fugiat libero pariatur amet laudantium minima consectetur tenetur. Repudiandae, autem, voluptate modi delenit sequi voluptatibus!

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nisi earum fugit veritatem amet quae sint vitae numquam dolore sunt quod a odio officia voluptate doloribus, at. Cum quasi mollitia eaque. Voluptatibus soluta, consequuntur, reiciendis debitis omnis fugiat libero pariatur amet laudantium.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ipsum voluptatibus soluta, consequuntur, reiciendis debitis omnis fugiat libero pariatur amet laudantium minima consectetur tenetur. Repudiandae, autem, voluptate modi delenit sequi voluptatibus!

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nisi earum fugit veritatem amet quae sint vitae numquam dolore sunt quod a odio officia voluptate doloribus, at. Cum quasi mollitia eaque.

#### Sidebar One

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Voluptatum modi nisi tenetur sint dignissimos nulla, blanditiis nesciunt.

#### Sidebar Two

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Voluptatum modi nisi tenetur sint dignissimos nulla, blanditiis nesciunt.

```

/* second example */
.column-layout {
  max-width: 1300px;
  background-color: #FFF;
  margin: 40px auto 0 auto;
  line-height: 1.65;
  padding: 20px 50px;
  display: flex;
}

.main-column {
  flex: 3;
  /* order: 2; */
}

.sidebar-one {
  flex: 1;
  /* order: 1; */
}

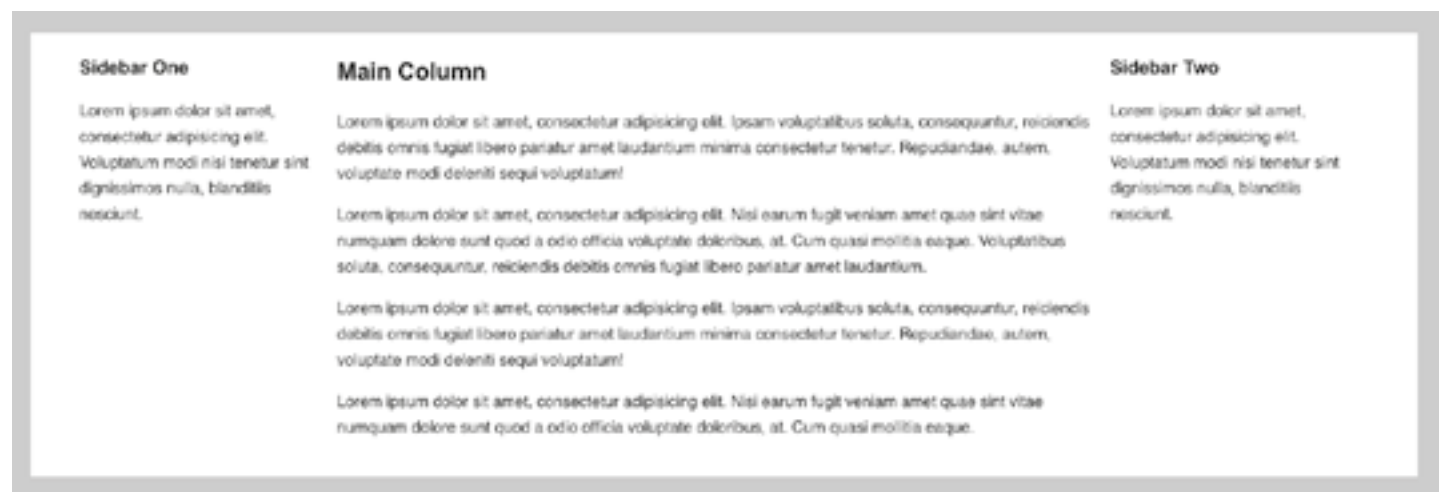
.sidebar-two {
  flex: 1;
  /* order: 3; */
}

```

*In het voorbeeld links zie je dat we shorthand 'flex' gebruiken zodat we niet 'flex-grow', 'flex-shrink' en 'flex-basis' op 3 lijnen moeten schrijven.*

*De waarden staan in verhouding tot elkaar.*

*Met 'order' kunnen we de volgorde van de kolommen sturen, zonder de HTML te moeten aanpassen.*



# FLEX-ITEM: SHRINK

*relatief verkleinen items*

*Als de Flex-container te klein wordt:  
dan is 'flex-shrink' de verhouding waarin het  
item verkleint.*

```
<div class="wrapper flex-container">
  <div class="box one">1</div>
  <div class="box two">2</div>
  <div class="box three">3</div>
</div>
```

```
.wrapper{
  width: 90vw;
  max-width: 960px;
  margin: 0 auto;
}

.flex-container{
  display: flex;
  background: #fff;
}

.box{
  height: 100px;
  width: 320px;
}

.one{
  background: red;
  flex-shrink: 1;
}

.two{
  background: blue;
  flex-shrink: 2;
}

.three{
  background: green;
  flex-shrink: 3;
}
```

## 2 FLEX-SHRINK

# FLEX-ITEM: FLEX BASIS

*de start waarde van een item*

## 3 FLEX-BASIS

*Flex-basis is eenvoudig;  
het is de beginwaarde  
van een item, vanwaar  
het schalen vertrekt.*

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

.item{
  background: lightblue;
  margin: 0.5em;
  flex-grow: 1;
  flex-shrink: 1;
}

.one{
  flex-grow: 7.5; /* te veel ruimte: pak 75% */
  flex-shrink: 5; /* te weinig ruimte: pak 50%; */

  flex-basis: 100px;
}
```

*Flex-basis is een van de eigenschappen die je op de flex-items kan toepassen. Door Flex-grow en Flex-shrink te gebruiken, kun je elk item baseren op een bepaalde breedte. Indien er ruimte over is, (of te weinig), kun je aangeven wat daarmee moet gebeuren. Flex-basis is de waarde waar hij van vertrekt. Het verschil met de gewone 'width' property: als je de assen draait kan het ook een 'height' zijn.. vandaar.*



# SHORT HAND

*in 1 lijn: flex-grow, flex-basis & flex-shrink: flex*



```
.flex-container{  
    display: flex;  
}  
  
.item{  
    background: lightblue;  
    margin: 0.5em;  
  
    flex-grow: 1;  
    flex-shrink: 2;  
    flex-basis: 100px;  
  
    flex: 1 2 100px;  
}
```

The image shows a code editor window with a dark background and three colored window control buttons (red, yellow, green) at the top left. The code is written in a light-colored font. The first block defines a flex container with `display: flex;`. The second block defines a flex item with `background: lightblue;`, `margin: 0.5em;`, and three flex properties: `flex-grow: 1;`, `flex-shrink: 2;`, and `flex-basis: 100px;`. Below these is the shorthand `flex: 1 2 100px;`. Three arrows are drawn on the left side of the code: a white arrow points from the `flex-grow` property to the first value '1' in the shorthand; a green arrow points from the `flex-shrink` property to the second value '2' in the shorthand; and an orange arrow points from the `flex-basis` property to the third value '100px' in the shorthand.

# OPDRACHT

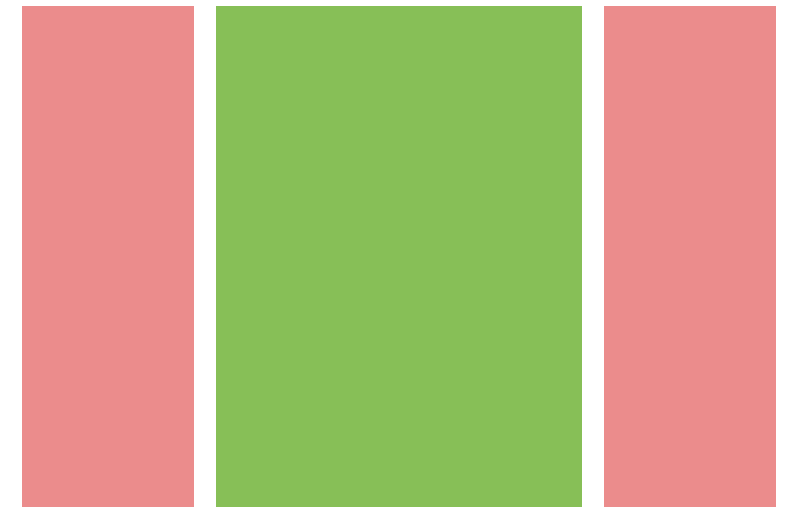
## layout

### ***Maak een layout:***

*Als de pagina groter is dan 800px;*

*- maak dan een 3 kolommen layout waarin:*

- kolom 1 gelijk is aan 25%*
- kolom 2 gelijk is aan 50%*
- kolom 3 gelijk is aan 25%*

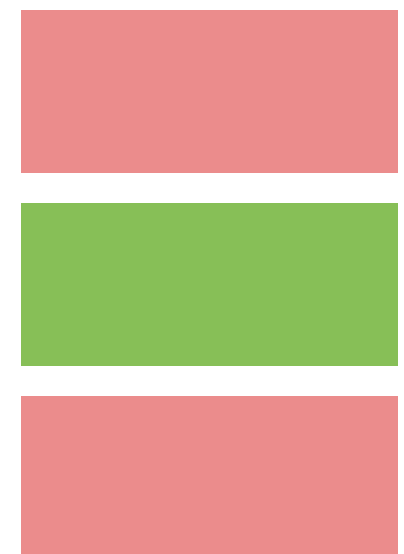


### ***Mobile versie:***

*- Als de pagina kleiner is dan 800px:*

*- stapel dan alle kolommen op elkaar.*

*Tip: - gebruik een media-query.*

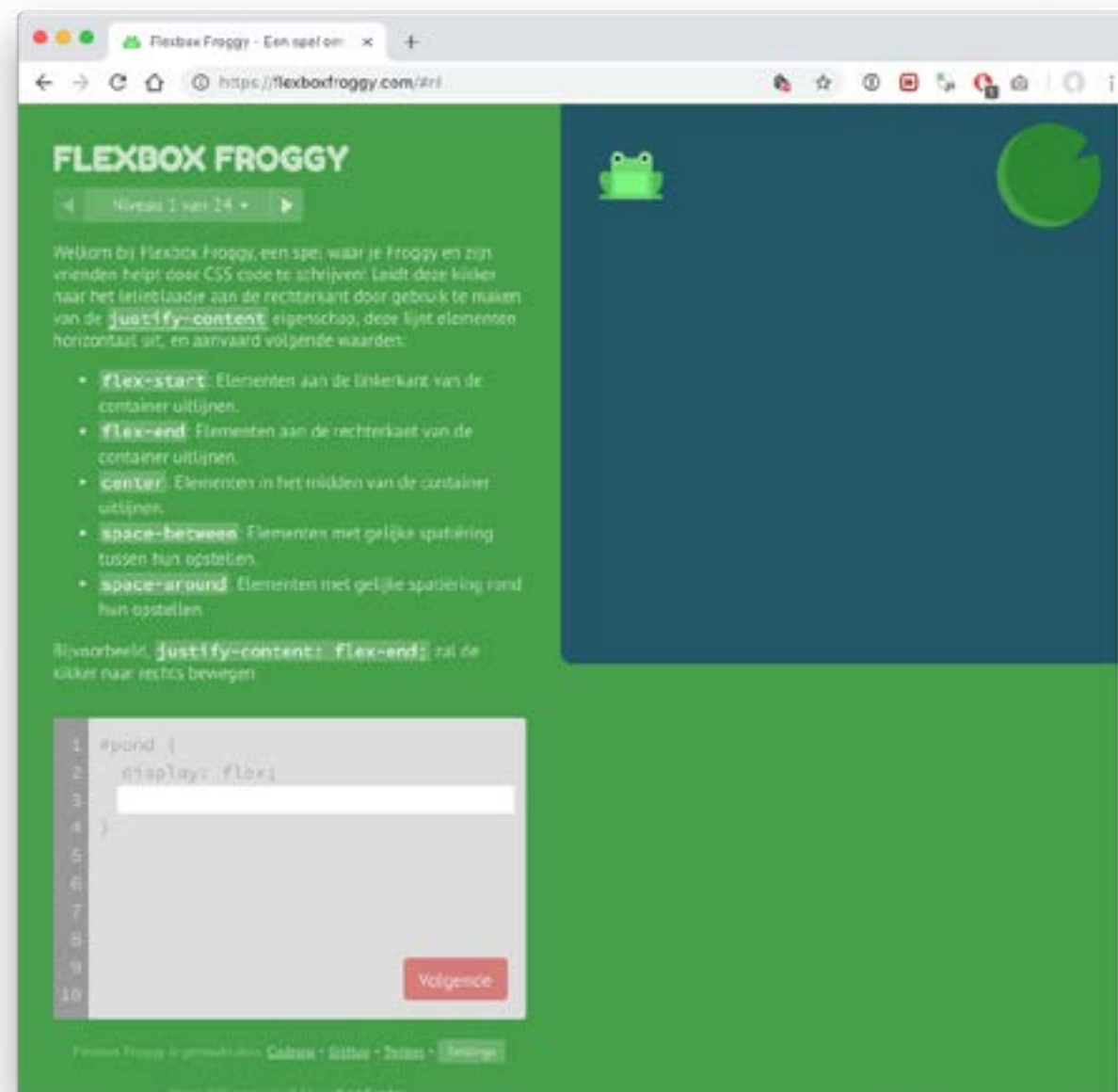


<https://codepen.io/petervandenheuvel/pen/JmPZYY?editors=1100>

# VERDER OEFENEN?

*probeer 'Flexbox-froggy'-game*

---

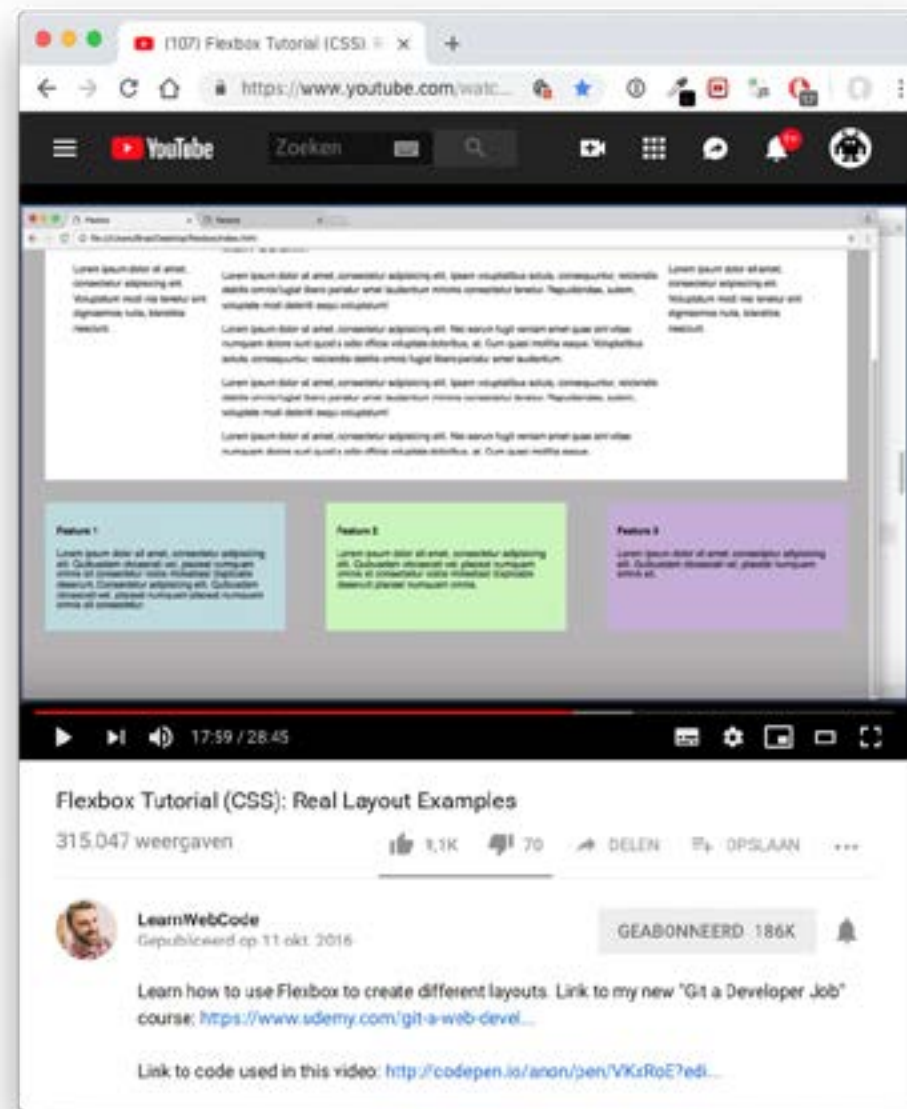




# BRONNEN:

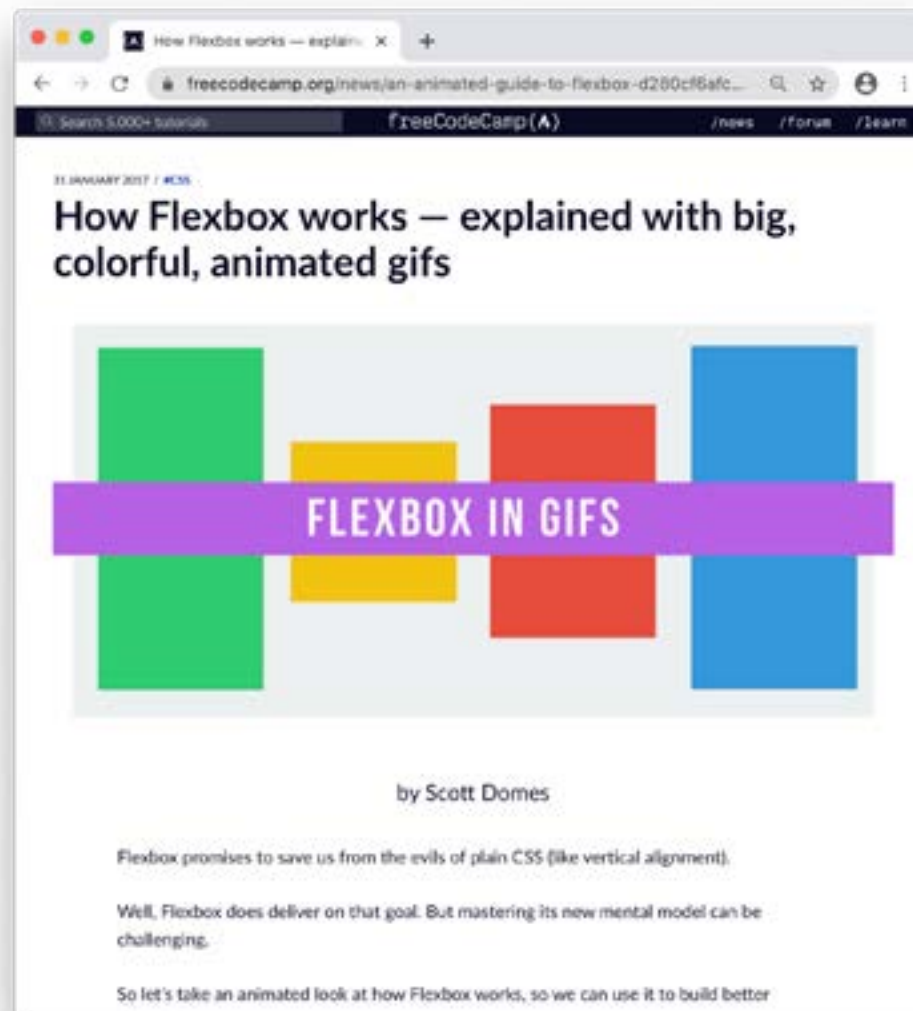
## *LearnWebCode & Kevin Powell*

---



# FLEXBOX IN GIFS:

<https://www.freecodecamp.org/news/an-animated-guide-to-flexbox-d280cf6afc35/#.xdqa0my2e>



# FLEXBOX VOOR LAYOUT:

<https://css-tricks.com/the-thought-process-behind-a-flexbox-layout/>

