MODERNE LAYOUT

MET FLEXBOX

FIRST SCREEN: HERO SECTION

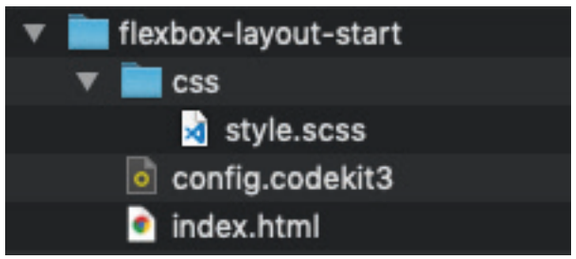


SECOND SCREEN:
CALL TO ACTION (CTA)

THIRD SCREEN:
SALESPOINTS

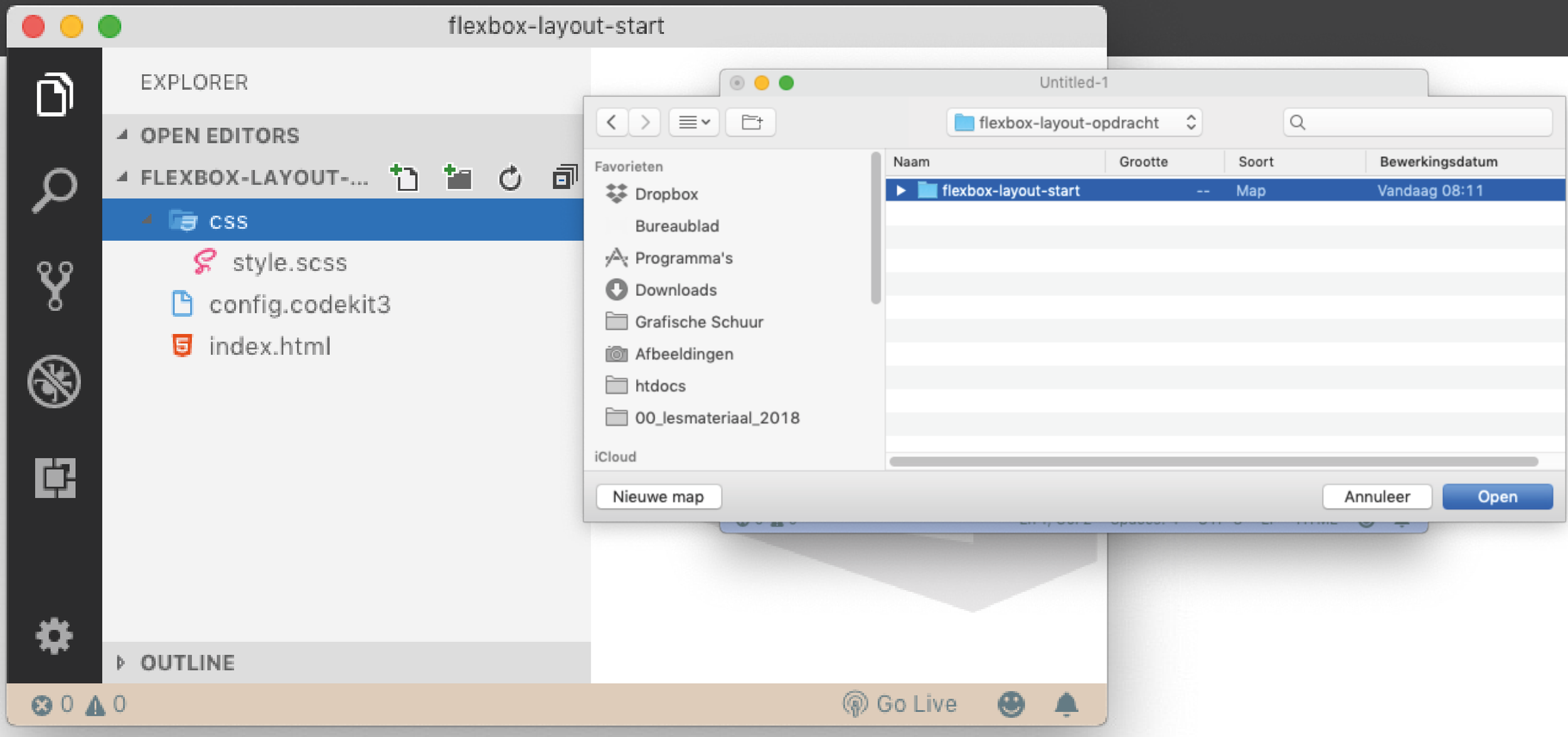
PAGINA MET GIT REPO

https://bitbucket.org/PetervandenHeuvel/flexbox-layout-opdracht/downloads



START PROJECT

1) 'flexbox-layout-start' op desktop,
2) sleep deze op icoon Visual Studio
(openen als project)



SCHRIJVEN VAN HTML

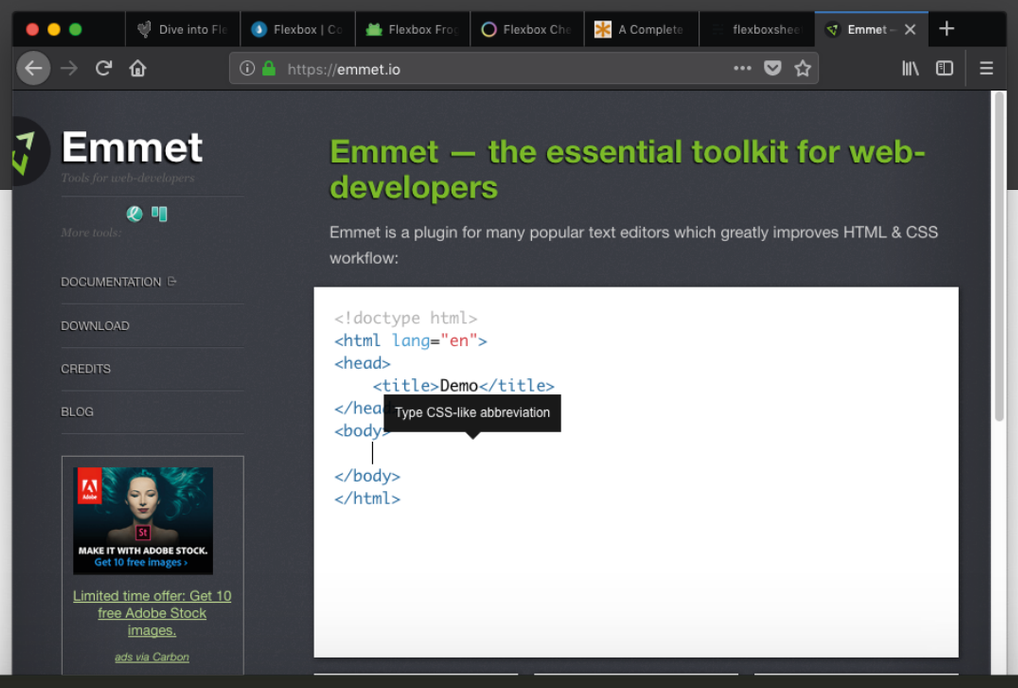
het aanmaken van het 'hero' gedeelte

1 HERO

meer info over deze syntax?
www.emmet.io

div.hero>p+h1+p

- *Verder een <h1> met een *
- *Een <p> met de -tags.*




```
<div class="hero">
  <p>Things are <span>awesome</span> here at</p>
  <h1>Awesome<span>Co</span></h1>
  <p>We <strong>Design</strong> and <strong>Develope Awesome</strong> stuff</p>
</div>
```

HTML

basic site structure

2 SALES POINTS

```
<div class="sales-points">
  <div class="sales-point">
    <h2>Sales Point 1</h2>
    <p>Maecenas id volutpat tellus. Praesent at rhoncus ante, non sollicitudin nisl.</p>
  </div>
  <div class="sales-point">
    <h2>Sales Point 2</h2>
    <p>Maecenas id volutpat tellus. Praesent at rhoncus ante, non sollicitudin nisl.</p>
  </div>
  <div class="sales-point">
    <h2>Sales Point 3</h2>
    <p>Maecenas id volutpat tellus. Praesent at rhoncus ante, non sollicitudin nisl.</p>
  </div>
</div>
```

div.sales-points>(div.sales-point>(h2>{sales-point $})+p>lorem)*3

Parent-div met de klasse 'sales-point' met daarin 3x een .sales-point.
Je kan dit in kleiner stukjes opbreken, als je dat eenvoudiger vindt. Of kijk naar het voorbeeld in de HTML.

Tip: er bestaat een 'cheat-sheet voor Emmet'. Je kan deze vinden als PDF op Google.

HTML

basic site structure

3 CALL TO ACTION

div.cta>div.cta-text

een div met de klasse .cta en daarin een child-div

h2>{call to action}

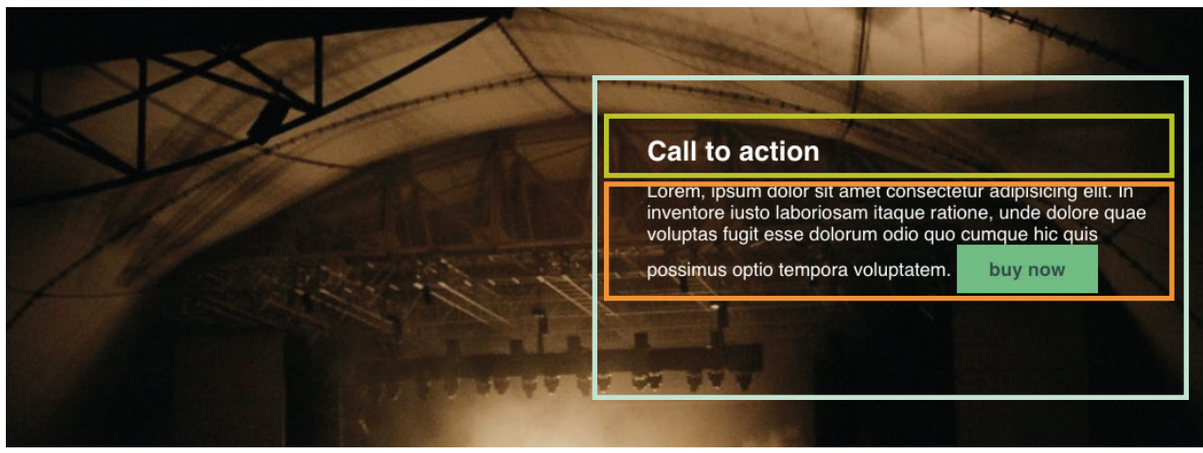
*wrap 'action in een *

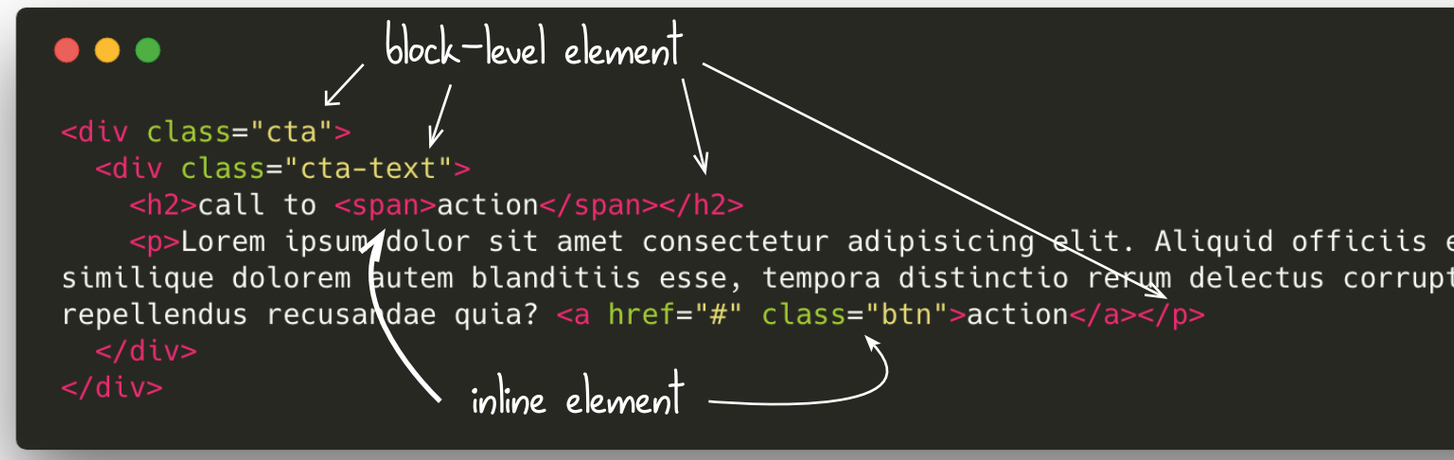
p>lorem

een paragraaf met wat lorem ipsum

a.btn>{action}

een anchor-tag met een klasse .btn





WAT BASIS 'STYLING'

CSS VOOR ONZE MARK-UP

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

body {
  font-family: 'Poppins', sans-serif;
  color: #444;
  font-size: 1.1rem; }

h1 {
  font-size: 5rem;
  color: #0F9;
  text-transform: uppercase;
  letter-spacing: 15px;
  outline: 3px solid #FFF;
  outline-offset: .25em;
  display: inline-block;
  margin: 0.25em 0 0.5em; }
  h1 span {
    font-weight: 300; }

h2 {
  font-size: 1.6rem;
  margin-bottom: 0.5em; }

p span {
  color: #0F9; }

.btn {
  display: inline-block;
  padding: .7em 1.7em;
  background: #0F9;
  color: #444;
  font-weight: 600;
  text-decoration: none; }
  .btn:hover {
    background: #00cc7a; }
```

Verander het box-model

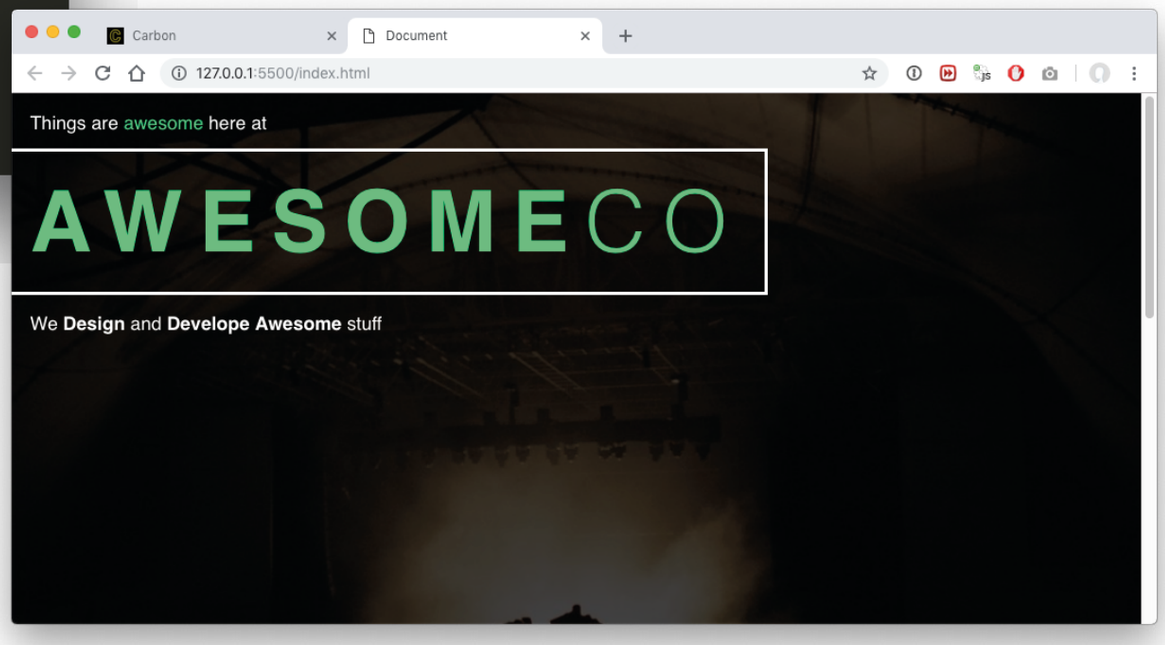
De <h1> neemt niet 100% in, maar enkel de breedte van de tekst. Doordat het inline-block is, kunnen we marge geven.

1 CSS - HERO

```
/*
  #### Hero
  */
.hero {
  background-color: #444;
  background-image: url("../img/600.jpg");
  background-blend-mode: multiply;
  background-size: cover;
  height: 100vh;
  padding: 1em;
  color: #FFF;
  /* Your Flex-code goes here: */ }
```

Achtergrondkleur van zwart (#444)
Achtergrond afbeelding, samen gemixed door blend-mode.

Altijd 100% van de hoogte van het scherm, cover om mooi te schalen, met wat padding en een text-color.

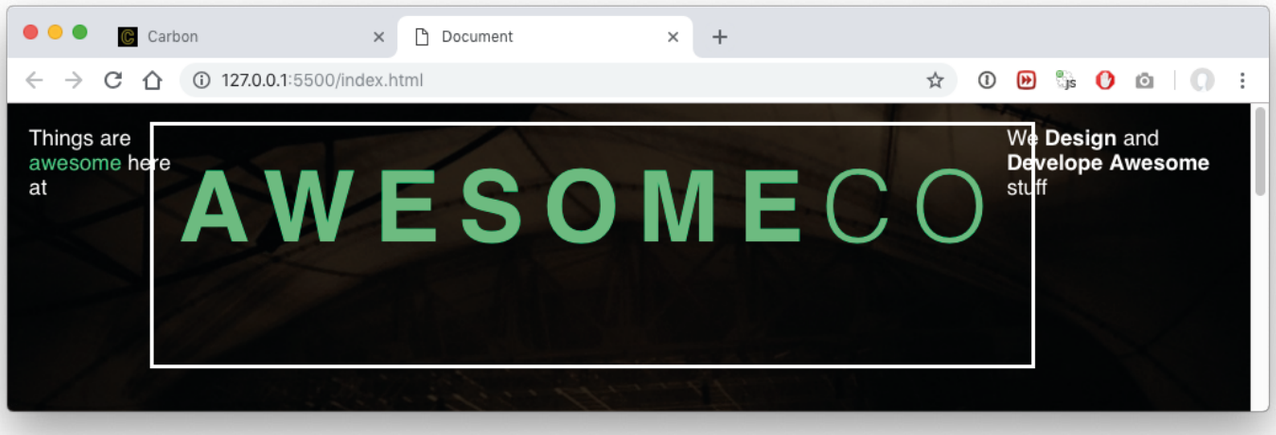


1 CSS - HERO

```css
/*
  #### Hero
  */
.hero {
  background-color: #444;
  background-image: url("../img/600.jpg");
  background-blend-mode: multiply;
  background-size: cover;
  height: 100vh;
  padding: 1em;
  color: #FFF;
  /* Your Flex-code goes here: */
1 display: flex;
2 flex-direction: column;
  align-items: center;
  justify-content: center;
}
```

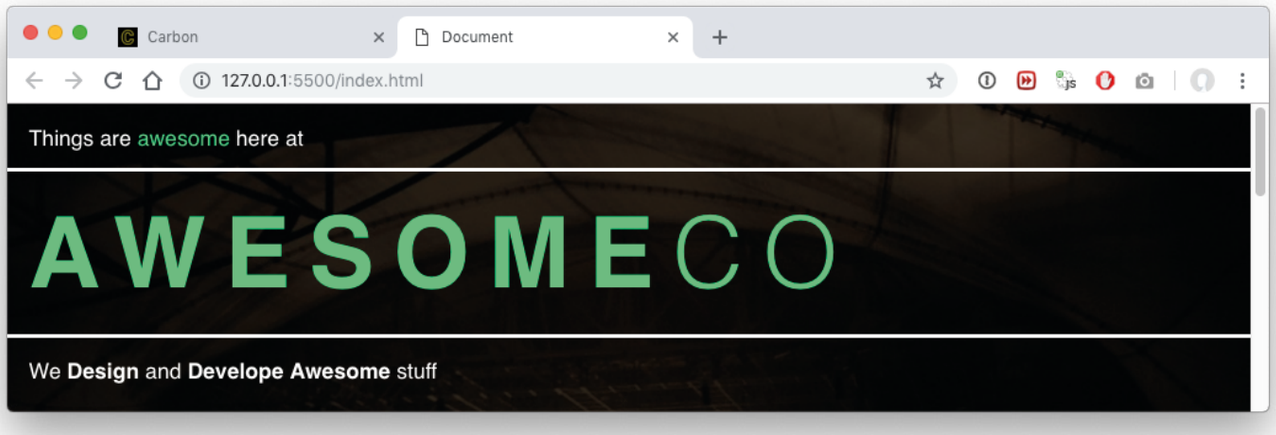
Stap 1:

Binnen de .hero, kiezen we voor display: flex.
Je zal zien dat direct alle flex-items in een rij komen staan.



Stap 2:

Als je dit wil wijzigen, kies je voor:
flex-direction: column;



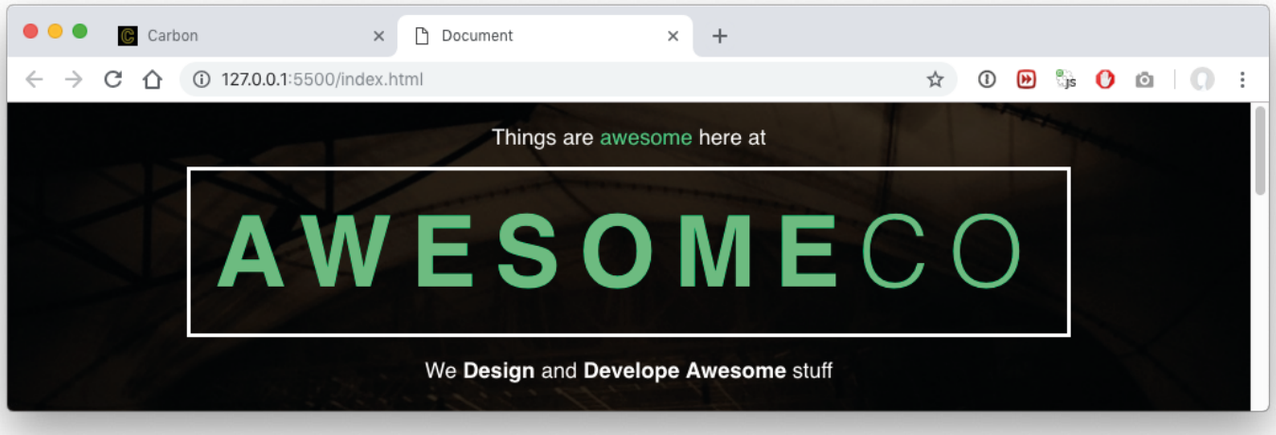
1 CSS - HERO

```
/*
  #### Hero
  */
.hero {
  background-color: #444;
  background-image: url("../img/600.jpg");
  background-blend-mode: multiply;
  background-size: cover;
  height: 100vh;
  padding: 1em;
  color: #FFF;
  /* Your Flex-code goes here: */
  display: flex;
  flex-direction: column;
3 align-items: center;
4 justify-content: center;
}
```

Stap 3:

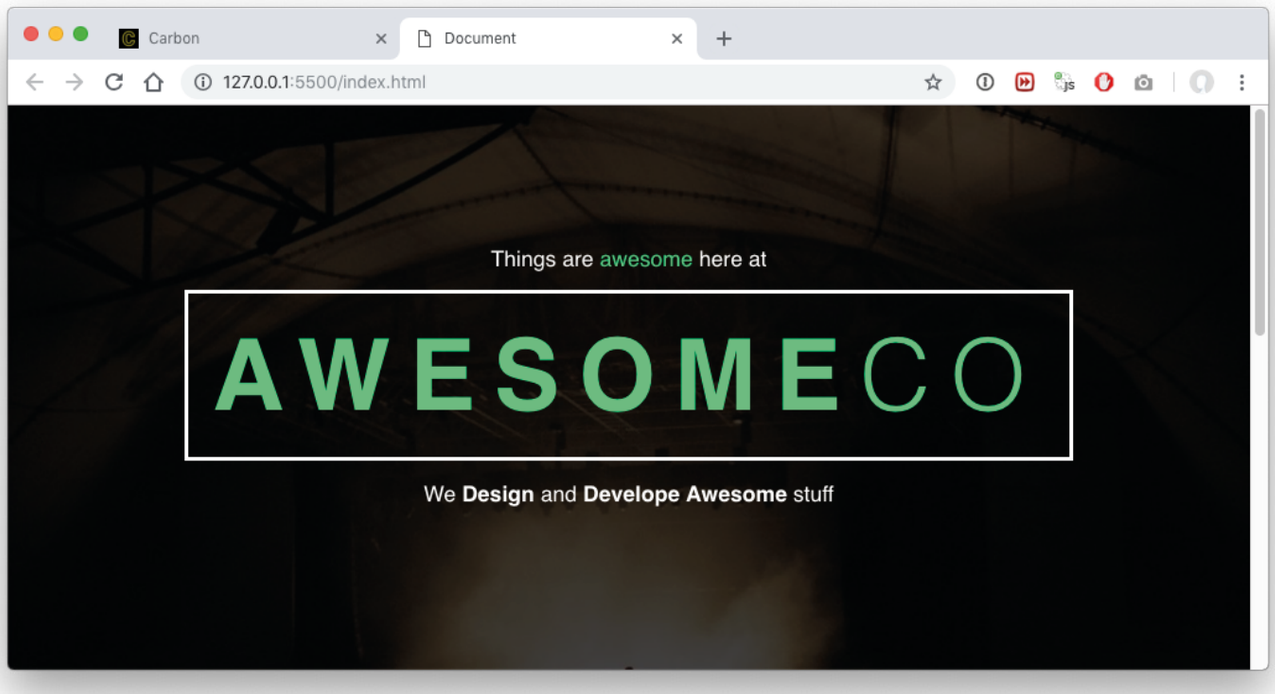
Je zal zien dat ook de 'inline-block' voor de <h1> niet meer gerespecteerd wordt; alles is nu 'stretching' om het vlak te vullen.

Als we nu alle flex-items willen centreren kiezen we voor: align-items: center.

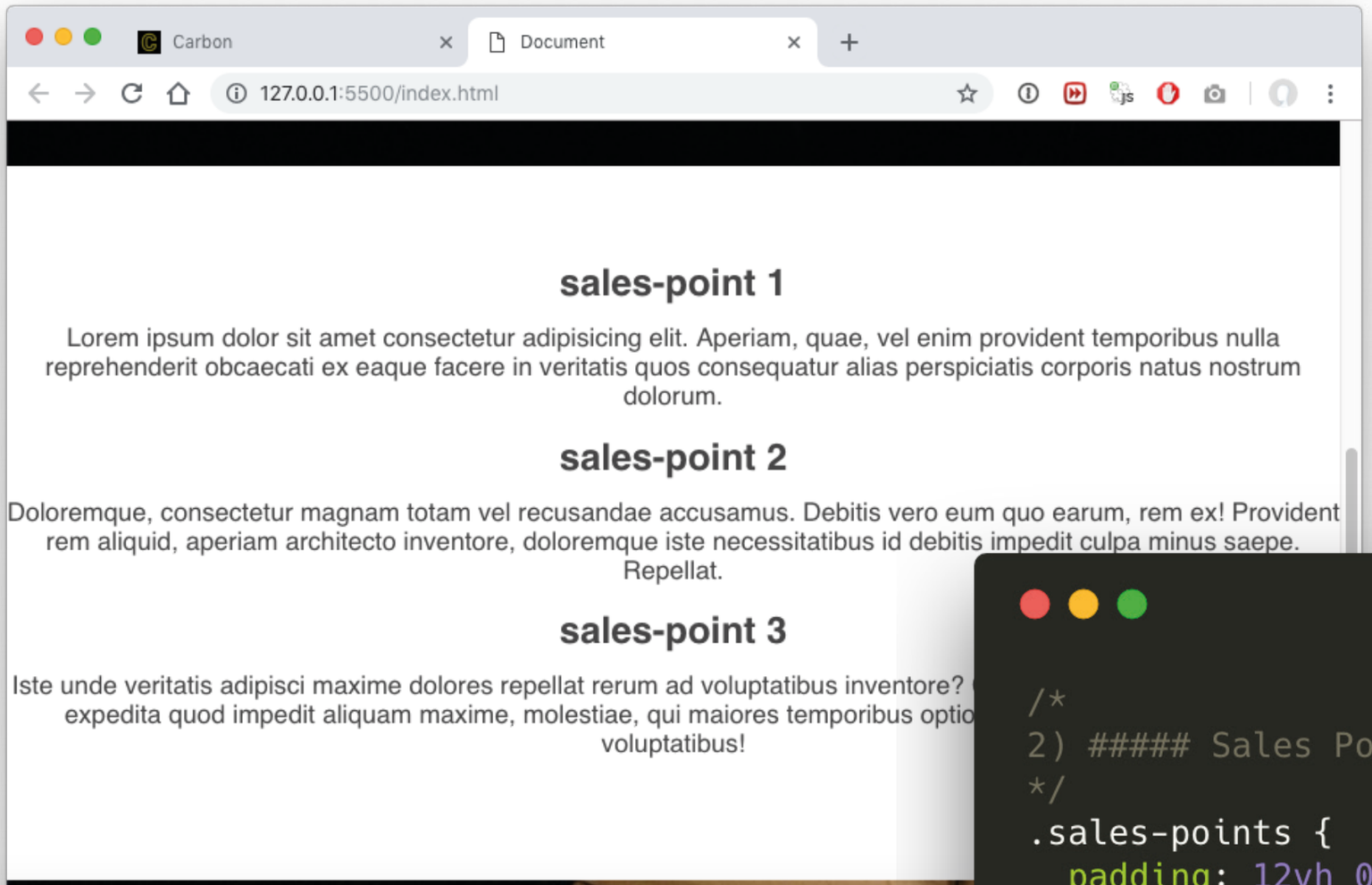


Stap 4:

Om nu alle items verticaal te centreren op de achtergrond, kiezen we voor: justify-content: center.



2 CSS - SALES POINTS




```
    <div class="sales-points">
        <div class="sales-point">
            <h2>sales-point 1</h2>
            <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Ap
temporibus nulla reprehenderit obcaecati ex eaque facere in veritatis quos
corporis natus nostrum dolorum.</p>
        </div>
        <div class="sales-point">
            <h2>sales-point 2</h2>
            <p>Doloremque, consectetur magnam totam vel recusandae accusam
rem ex! Provident rem aliquid, aperiam architecto inventore, doloremque is
impedit culpa minus saepe. Repellat.</p>
        </div>
        <div class="sales-point">
            <h2>sales-point 3</h2>
            <p>Iste unde veritatis adipisci maxime dolores repellat rerum ad voluptatibus inventore? Cumque
```

```
/*
2) ##### Sales Points
*/
.sales-points {
  padding: 12vh 0;
  text-align: center;
  /* Your Flex-code goes here: */ }

.sales-point {
  background-color: red; }

@media (min-width: 40rem) {
  .sales-points {
    background-color: yellow;
    /* Your Flex-code goes here: */ } }
```

2 CSS - SALES POINTS

```
/*
2) ##### Sales Points
*/
  .sales-points {
    padding: 12vh 0;
    text-align: center;
    /* Your Flex-code goes here: */
    display: flex;
    justify-content: space-around;
    flex-direction: column;
  }
  .sales-point {
    flex-basis: 30%;
    background-color:red;
  }

  /* kleiner scherm: op elkaar. */
  @media (min-width: 40rem) {
    .sales-points {
      background-color: yellow;
      /* Your Flex-code goes here: */
      flex-direction: row;
    }
  }
```

Stappen bij 'Sales Points':

Binnen de .sales-points, kiezen we voor display: flex.

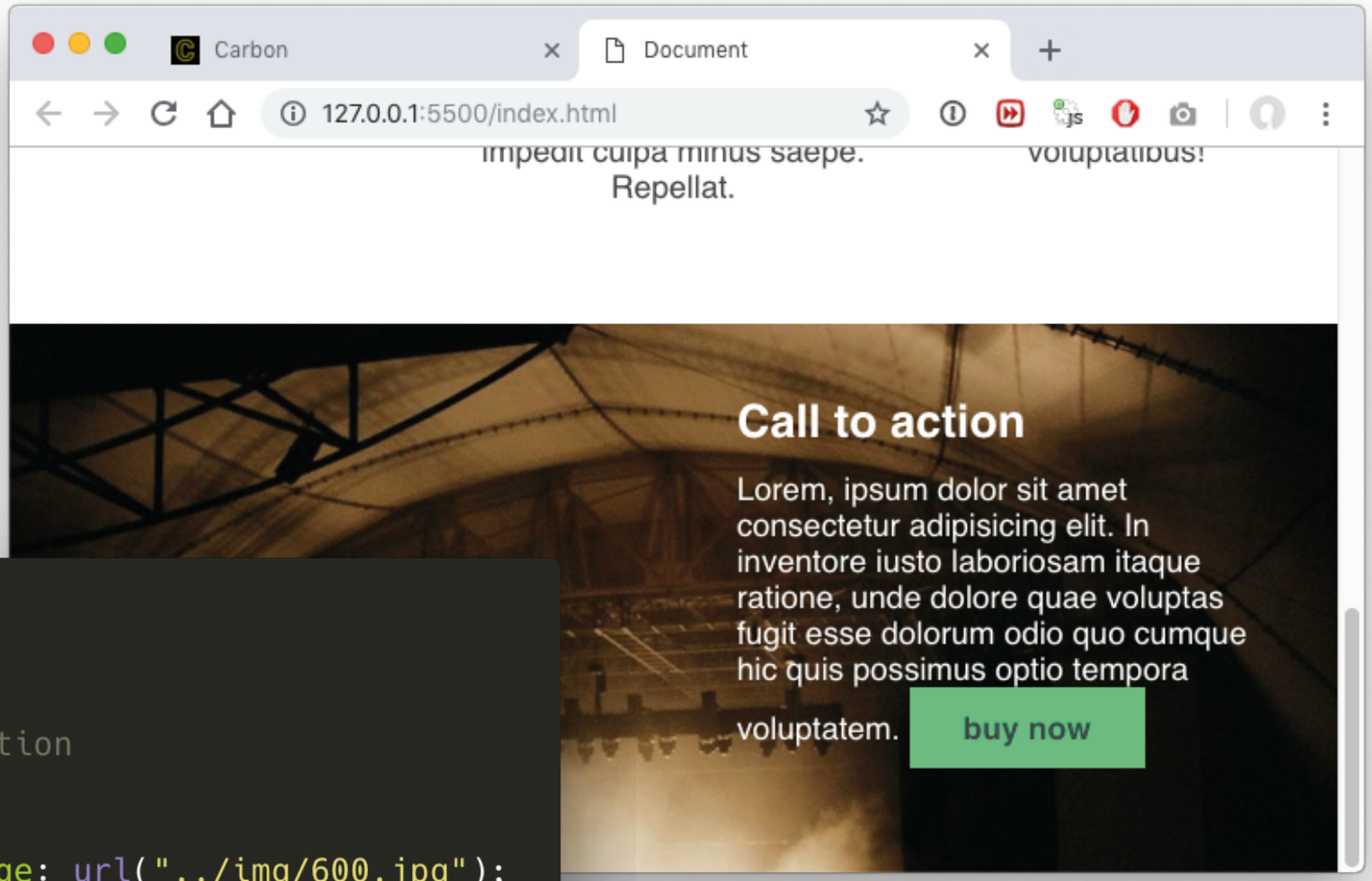
We lijnen de kolommen uit, door ze allereerst kleiner te maken: .sales-point{ flex-basis: 30%;}, daardoor blijft er ruimte over om te gebruiken als tussen ruimte.

Flex-basis dient om een gewenste minimale breedte in te stellen.

Door op op de Flex-container 'space-around' te gebruiken, bekomen we een mooie uitlijning.

Flex-direction: column gebruiken we om de blokken mooi opelkaar te stapelen bij kleine weergaven, op grote schermen, gebruiken we een media-query om alles naast elkaar te zetten.

3 CSS - CALL TO ACTION




```
/*
##### Call to Action
*/
div.cta {
  background-image: url("../img/600.jpg");
  background-size: cover;
  padding: 10vh 0;
  color: #FFF;
  /* Your Flex-code goes here: */ }

div.cta-text {
  padding: 0 2em;
  /* Your Flex-code goes here: */ }
  @media (min-width: 40rem) {
    div.cta-text {
      /* Your Flex-code goes here: */ } }
```

Stappen bij 'Call to action':

We zien een container met een achtergrond afbeelding, met daarop een kolom van 50%, uitgelijnd aan de linker-kant.

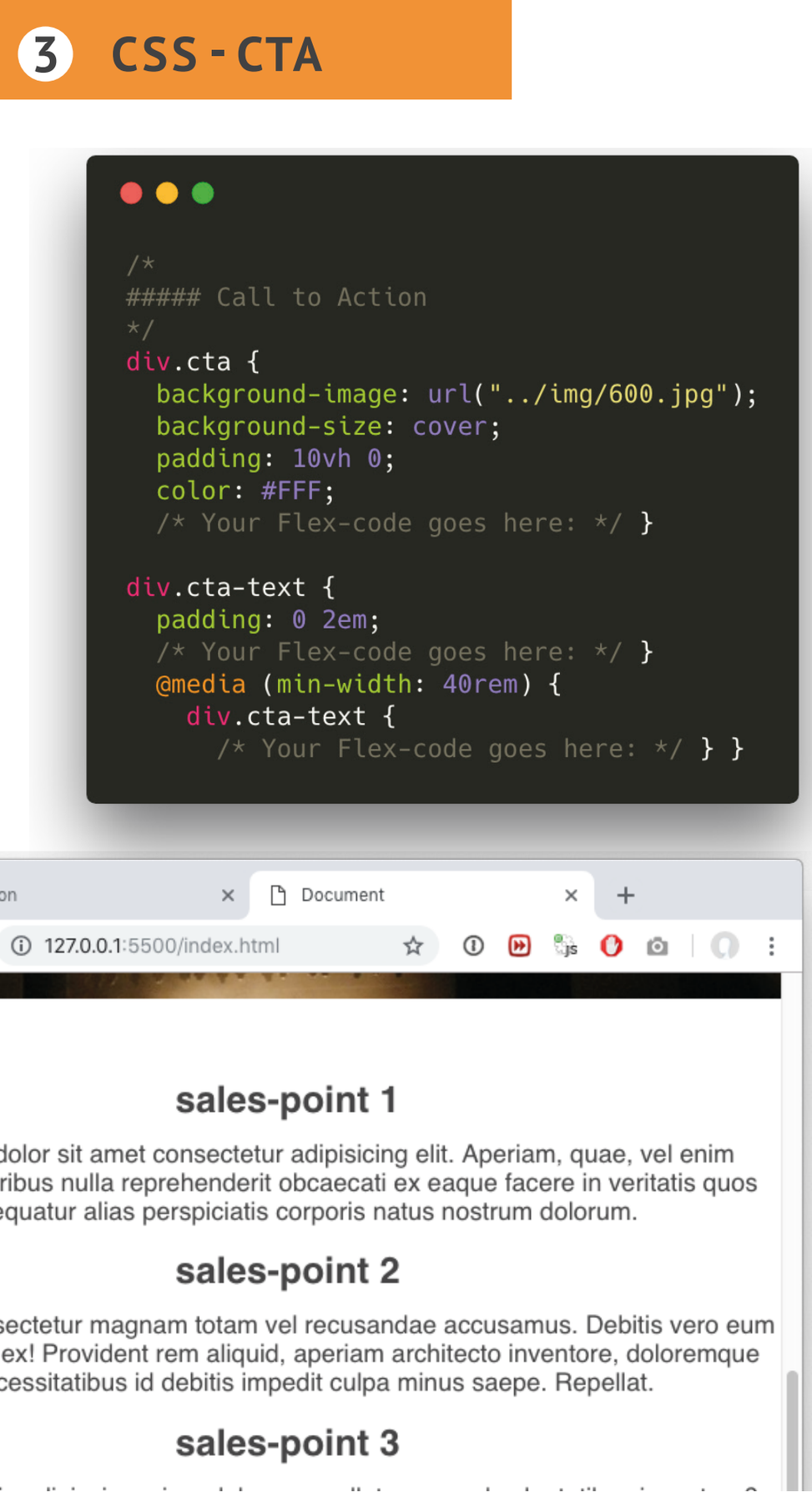
Traditioneel zou men een float:right gebruiken op de kolom en een clearfix op de container.

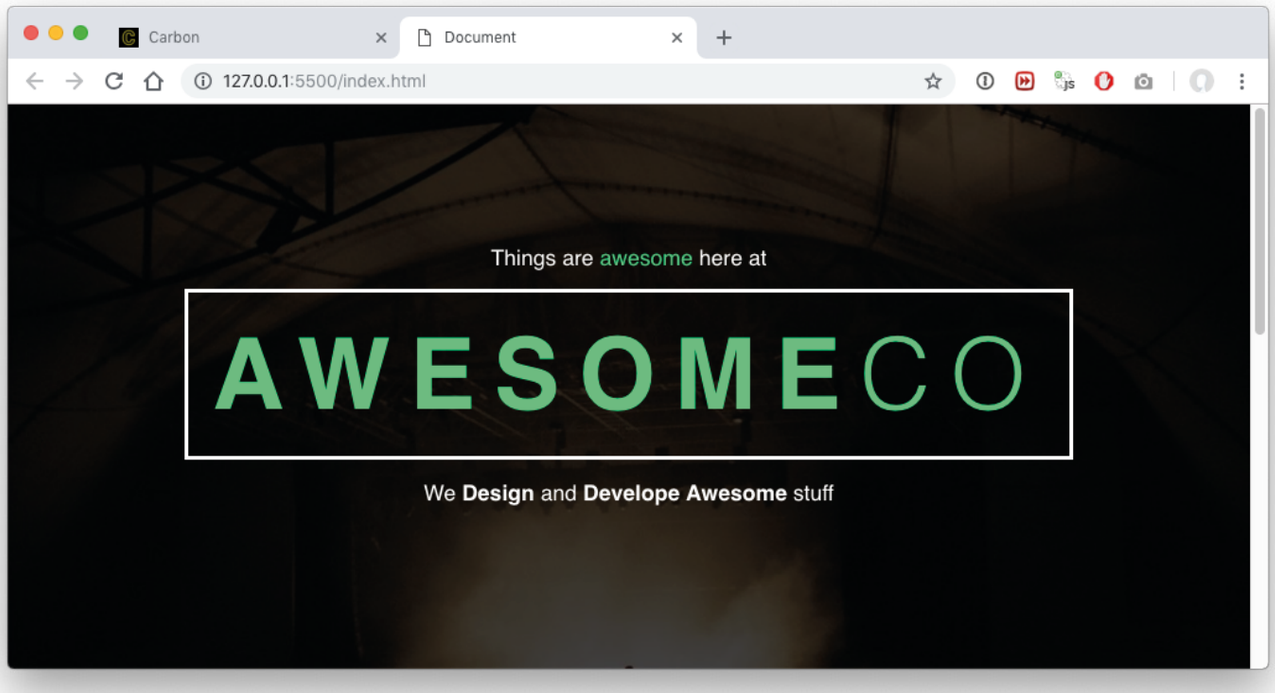
Met Flexbox is dit eenvoudiger.

3 CSS - CTA

```css
/*
##### Call to Action
*/
div.cta {
  background-image: url("../img/600.jpg");
  background-size: cover;
  padding: 10vh 0;
  color: #FFF;
  /* Your Flex-code goes here: */ }

div.cta-text {
  padding: 0 2em;
  /* Your Flex-code goes here: */ }
  @media (min-width: 40rem) {
    div.cta-text {
      /* Your Flex-code goes here: */ } }
```





3 CSS - CALL TO ACTION

```
/*
3) ##### Call to Action
*/
.cta {
  background-image: url("../img/600.jpg");
  background-size: cover;
  padding: 10vh 0;
  color: #FFF;
  /* Your Flex-code goes here: */
1 display: flex;
2 justify-content: flex-end; }

.cta-text {
  padding: 0 2em;
  /* Your Flex-code goes here: */
4 flex-basis: 100%; }
  @media (min-width: 40rem) {
    .cta-text {
      /* Your Flex-code goes here: */
3     flex-basis: 50%; } }
```

1 - We activeren de flex-container

Op de parent-container '.cta' activeren we Flexbox op de gebruikelijke manier: display: flex;

2 - We lijnen uit aan het einde van de container

Met 'justify-content: flex-end' plaatsen we de inhoud aan het einde van de flex-container.

3 - We verkleinen de basisgrootte van de kolom

In de media-query verkleinen we de flex-items van grootte als het scherm te groot wordt.

4 - We zetten de basis van de kolom op 100%

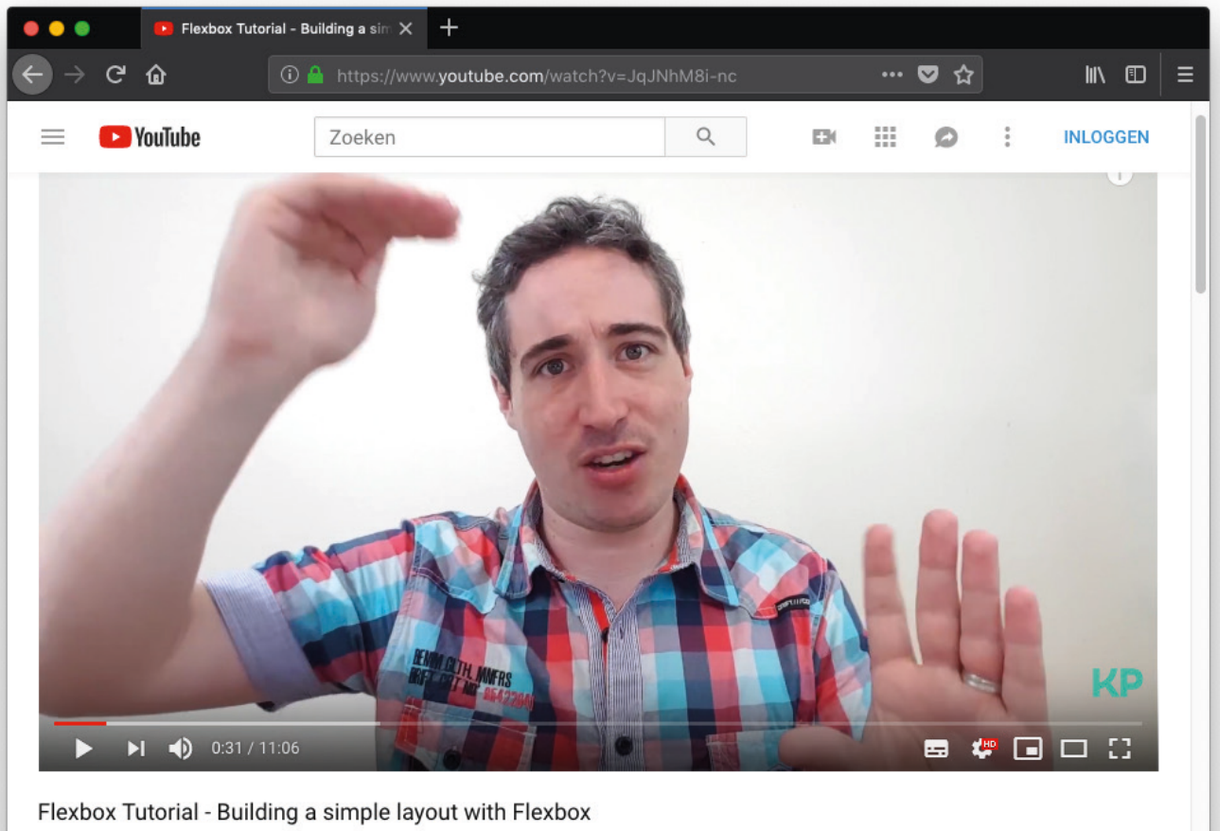
Uiteraard moet de kolom standaard op kleine schermen naar de hele breedte van het scherm gaan.

BRON

https://www.youtube.com/watch?v=JqJNhM8i-nc

KEVIN POWELL

HTTPS://WWW.YOUTUBE.COM/CHANNEL/UCJZV4D5RBIKD4QHMPKCABCW



CARABAS HANDS-ON TRAINING PRINT·WEB·OFFICE