

24-11-2017
Utrecht
Puur Utrecht V2 / wijzijnpuurutrecht

WORKSHOP

CSS Layout

Hidde de Vries
@hdv

Voorstellen

hiddedevries.nl

moz://a



En jullie?

Hoe heet je?
Waar ben je vandaan gereisd?
Wat is je favoriete CSS property?

programma

09.30-09.45 CSS layout introductie

09.45-10.15 Flexbox 101

10.15-12.30 Oefeningen met Flexbox

– lunch –

13.30-14.15 Grid Layout 101

14.15-16.00 Oefeningen met Grid Layout

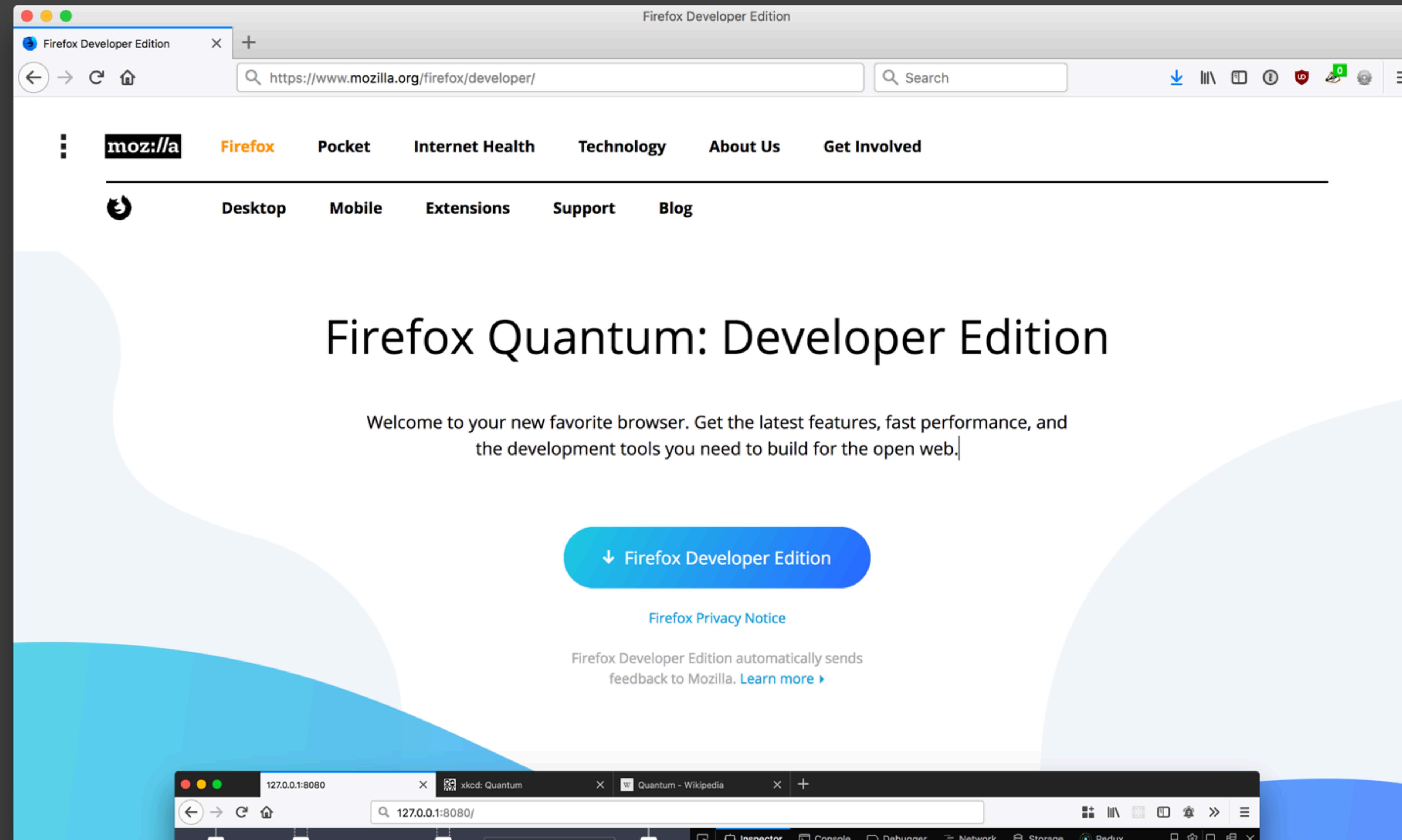
16.00-17.00 Grid in productie, fallbacks

**Wat heb je
nodig?**

Wat heb je nodig?

Je favoriete text editor; Codepen/JSBin mag ook
Een browser met goede grid-tools

<https://www.mozilla.org/firefox/developer/>



Wat heb je nodig?

Je favoriete text editor; Codepen/JSBin mag ook

Een browser met goede grid-tools

Voorbeeldcode

<https://github.com/hidde/workshop-css-layout/>

Wat heb je nodig?

Je favoriete text editor; Codepen/JSBin mag ook

Een browser met goede grid-tools

Voorbeeldcode

**Geen conventies... style gerust op ID's, gebruik gerust
verkeerde markup, en let niet op onderhoudbaarheid.**

CSS Layout



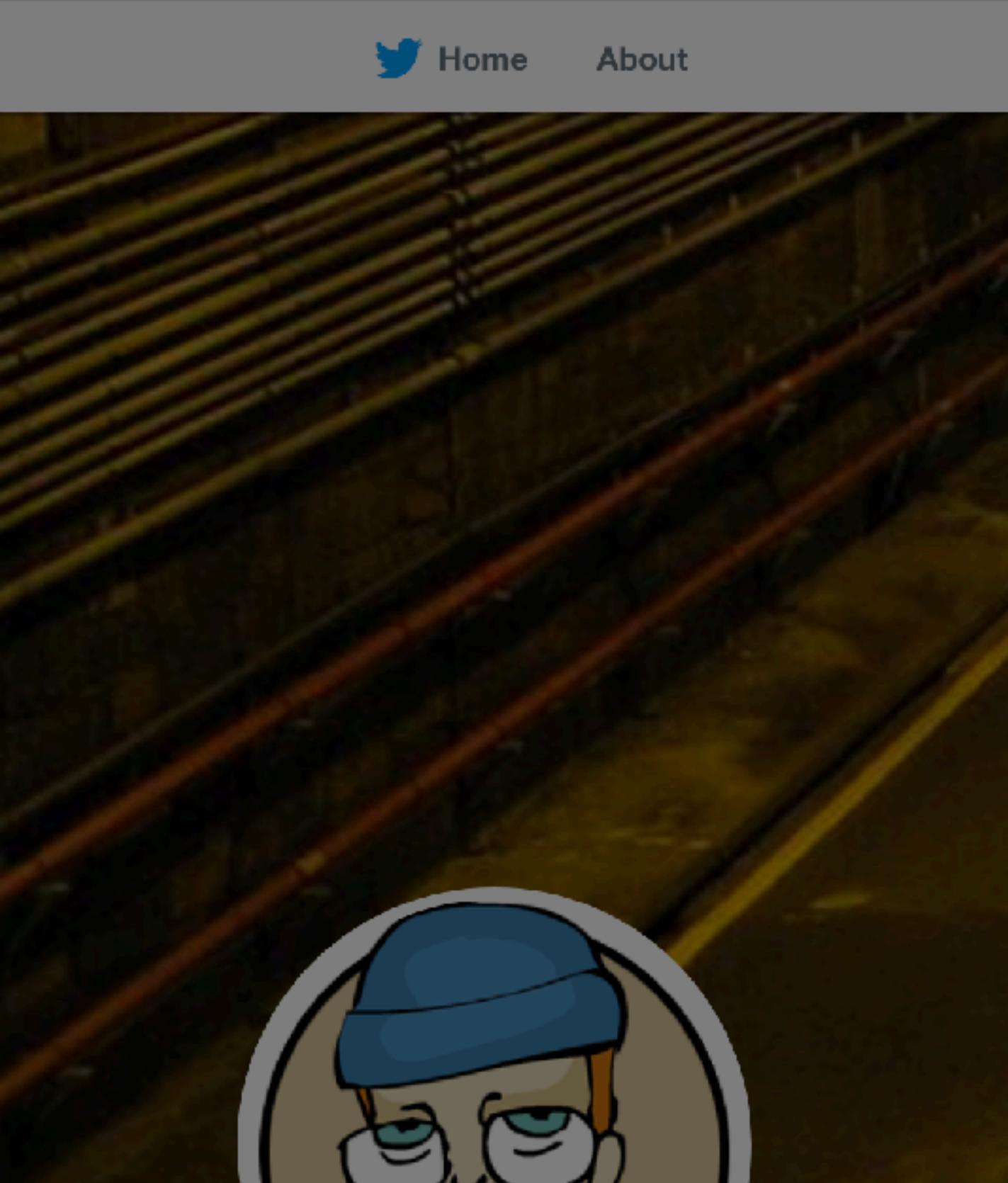
Eric Meyer

@meyerweb

Web standards, HTML/CSS, microformats, community, writing, speaking, signing guy. Husband, father, agnostic in principle, atheist in practice. #663399becca

📍 Cleveland Heights, OH

🔗 meyerweb.com



Eric Meyer
@meyerweb

Follow ▾

On the left: the 2nd and 3rd editions of “CSS: The Definitive Guide”. On the right, a single copy of the fourth edition.

?

1:25 PM - 10 Nov 2017

751 Retweets 2,039 Likes

Have an account? Log in ▾

© 2017 Twitter About Help Center Terms
Privacy policy Cookies Ads info

1 spec → vele modules

**Specs worden sneller
geïmplementeerd**

**Er is nu dedicated
CSS voor lay-out**



**Flexibiliteit zit in
het web ingebouwd.**

**Flexibiliteit zit in
CSS ingebouwd.**

Medium-agnostisch

“

CSS is a language for describing the rendering of structured documents (such as HTML and XML) on screen, on paper, in speech, etc.

– elke CSS spec

Geen programmeertaal

The screenshot shows the W3C Candidate Recommendation page for the CSS Grid Layout Module Level 1. The page title is "CSS Grid Layout Module Level 1" and it was last updated on 09 May 2017. The left sidebar contains a "TABLE OF CONTENTS" with sections such as Introduction, Overview, Grid Layout Concepts and Terminology, Reordering and Accessibility, Grid Containers, Grid Items, Defining the Grid, and an Abstract section. The main content area includes links to the latest published version, editor's draft, and previous versions. It also features sections for Test Suite, Issue Tracking, Editors, and Former Editors.

The screenshot shows the W3C Candidate Recommendation page for the CSS Flexible Box Layout Module Level 1. The page title is "CSS Flexible Box Layout Module Level 1" and it was last updated on 19 October 2017. The left sidebar contains a "TABLE OF CONTENTS" with sections such as Introduction, Flex Layout Box Model and Terminology, Flex Containers, Flex Items, Ordering and Orientation, Flex Lines, Flexibility, and Alignment. The main content area includes links to the latest published version, editor's draft, and previous versions. It also features sections for Test Suite, Issue Tracking, Editors, and Former Editors.

Bij twijfel, raadpleeg de spec

CSS Grid Layout

Related Topics
CSS

CSS Reference
CSS Grid Layout
Guides

Basics concepts of grid layout
Relationship to other layout methods
Line-based placement
Grid template areas
Layout using named grid lines
Auto-placement in grid layout
Box alignment in grid layout
Grids, logical values and writing modes
CSS Grid Layout and Accessibility
CSS Grid Layout and Progressive Enhancement
Resizing common layouts using grids

Properties

grid
grid-area
grid-auto-columns

HTML

```
1 <div class="wrapper">
2   <div class="one">One</div>
3   <div class="two">Two</div>
4   <div class="three">Three</div>
5   <div class="four">Four</div>
6   <div class="five">Five</div>
7   <div class="six">Six</div>
8 </div>
```

CSS

CSS Grid Layout excels at dividing a page into major regions, or defining the relationship in terms of size, position, and layer, between parts of a control built from HTML primitives.

Like tables, grid layout enables an author to align elements into columns and rows. However, many more layouts are either possible or easier with CSS grid than they were with tables. For example, a grid container's child elements could position themselves so they actually overlap and layer, similar to CSS positioned elements.

Basic example

The below example shows a three column track grid with new rows created at a minimum of 100 pixels and a maximum of auto. Items have been placed onto the grid using line-based placement.

A Complete Guide to Flexbox

Code Snippets » CSS »

BY CHRIS COYIER LAST UPDATED ON NOVEMBER 12, 2017

FLEXBOX, LAYOUT

▶ Background

▶ Basics & Terminology

container

items

Of MDN, CSS Tricks

layout modes

layout modes

inline

block

table

positioned

layout modes

inline

block

table

positioned

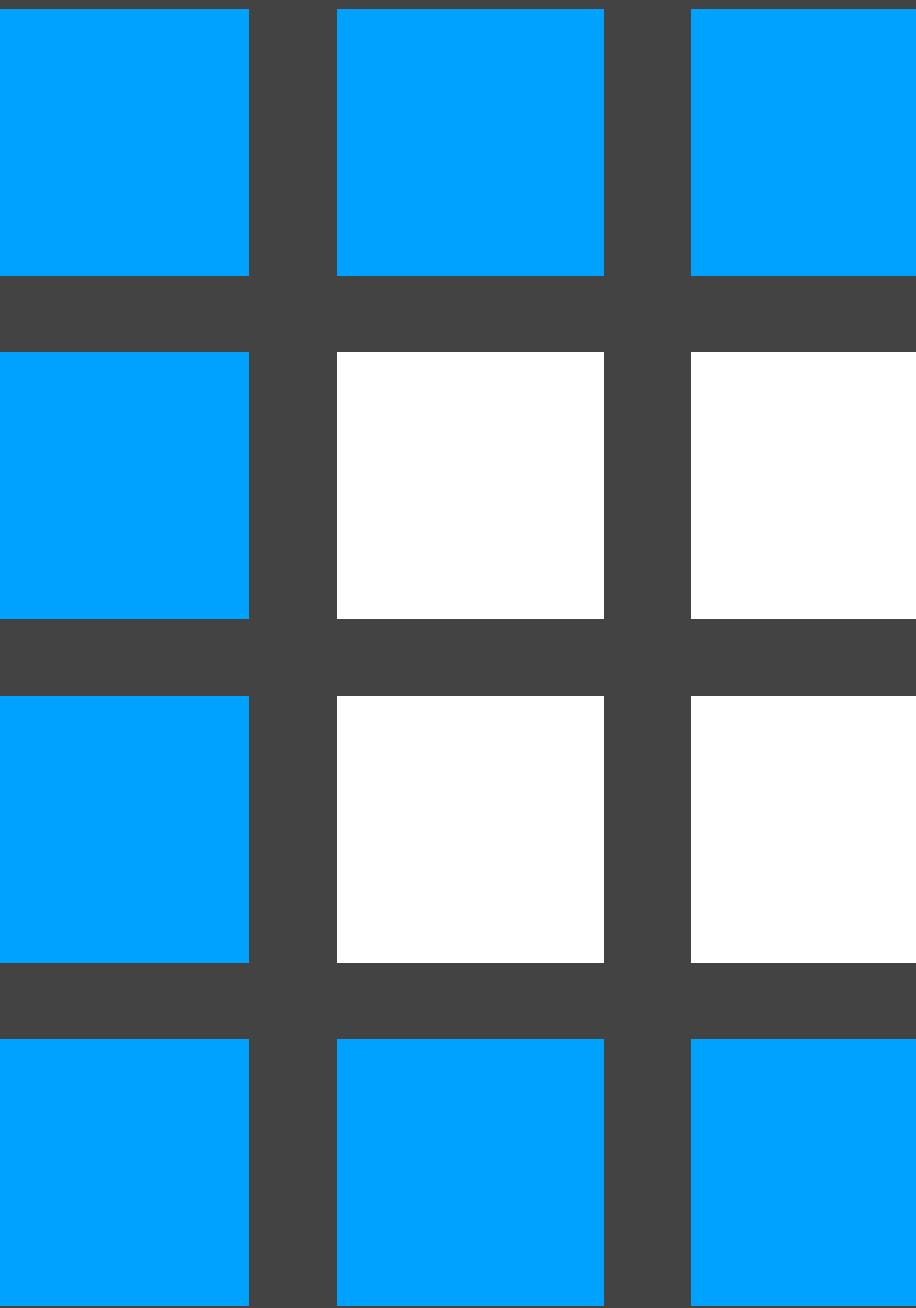
flex

grid

flex



grid



flex

vanuit de
content

groottes
gedefinieerd
op het item

grid

vanuit de
container

ruimtes
gedefinieerd
op de container

logische vs fysieke properties

`align-items: start`

`grid-column-start: start`

`margin-start / margin-end`

Logical properties and values specification

<https://www.w3.org/TR/css-logical-1/>

The screenshot shows a web browser displaying the 'CSS Logical Properties and Values Level 1' specification. The page is a W3C First Public Working Draft from May 18, 2017. It features a sidebar with a table of contents and a main content area with various sections and links.

Table of Contents (Sidebar):

- 1 Flow-Relative Values: 'block-start', 'block-end', 'inline-start', 'inline-end'
- 1.1 Logical Values for the 'caption-side' Property
- 1.2 Flow-Relative Values for the 'float' and 'clear' Properties
- 1.3 Flow-Relative Values for the 'text-align' Property
- 1.4 Flow-Relative Values for the 'resize' Property
- 2 Flow-Relative Page Classifications
- 3 Flow-Relative Box Model Properties
 - 3.1 Logical Height and Logical Width: the 'block-size' and 'inline-size' properties
 - 3.2 Flow-relative Margins: the 'margin-block-start', 'margin-block-end', 'margin-inline-start', 'margin-inline-end' properties and 'margin-block' and 'margin-inline' shorthands
 - 3.3 Flow-relative Offsets: the 'inset-block-start', 'inset-block-end', 'inset-inline-start', 'inset-inline-end' properties and 'inset-block', 'inset-inline', and 'inset' shorthands
 - 3.4 Flow-relative Padding: the 'padding-block-start', 'padding-block-end', 'padding-inline-start', 'padding-inline-end' properties and 'padding-block' and 'padding-inline' shorthands
 - 3.5 Flow-relative Borders
 - 3.5.1 Flow-relative Border Widths: the 'border-block-start-width', 'border-block-end-width'

Main Content Area:

CSS Logical Properties and Values Level 1

W3C First Public Working Draft, 18 May 2017

This version: <https://www.w3.org/TR/2017/WD-css-logical-1-20170518/>

Latest published version: <https://www.w3.org/TR/css-logical-1/>

Editor's Draft: <https://drafts.csswg.org/css-logical/>

Issue Tracking:
[Inline In Spec](#)
[GitHub Issues](#)

Editors:
[Rossen Atanassov](#) (Microsoft)
[Elika J. Etemad / fantasai](#) (Invited Expert)

Copyright © 2017 W3C® (MIT, ERCIM, Keio, Beihang). W3C liability, trademark and document use rules apply.

Abstract

This module introduces logical properties and values that provide the author with the ability to control layout through logical, rather than physical, direction and dimension mappings. The module defines logical properties and values for the features defined in [CSS21]. These properties are writing-mode relative equivalents of their corresponding physical properties.

CSS is a language for describing the rendering of structured documents (such as HTML and XML) on screen, on paper, in speech, etc.

Sommige CSS creeert een
block formatting context.

Sommige CSS creëert een
block formatting context.

Dit betekent dat vanaf dan,
het blok zelf verantwoordelijk
is voor lay-out.

Sommige CSS creëert een
block formatting context.

Het is waarbinnen dingen
uitgelijnd worden.

Block formatting contexts worden gemaakt door:

root element

float (behalve float: none)

position: absolute|fixed

display: inline-block

display: table-cell

display: table-caption

overflow (behalve overflow: visible)

**Block formatting contexts
worden gemaakt door:**

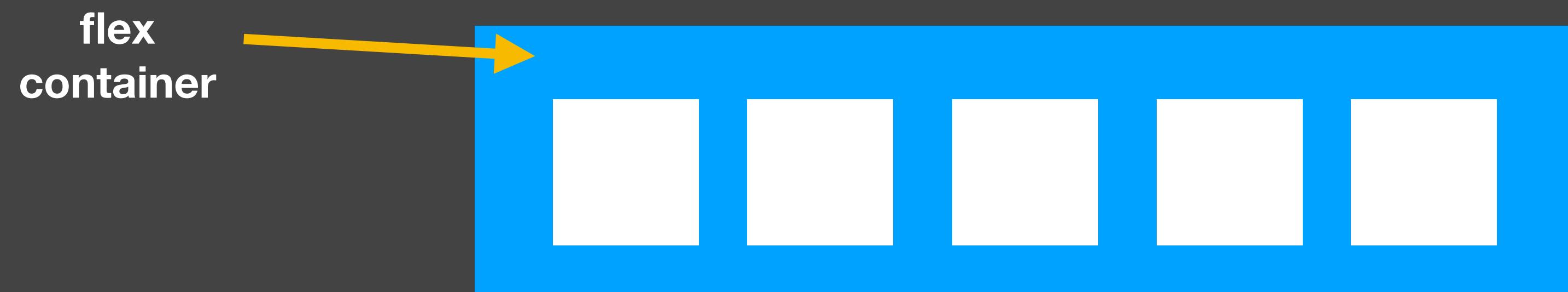
display: grid

display: flex

Flexbox

**flexbox laat je elementen
uitlijnen en overblijvende
ruimte slim gebruiken**

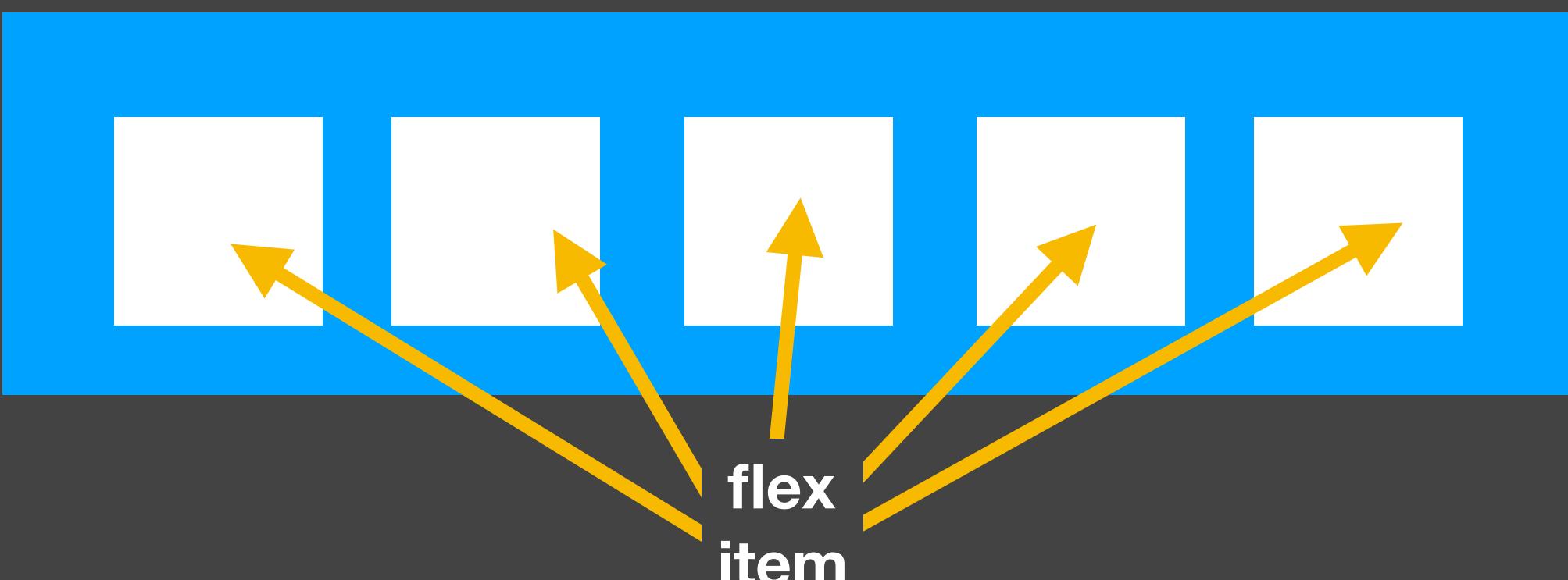
flex container



```
#container { display: flex; }
```

flex container

alle directe kinderen worden flex items



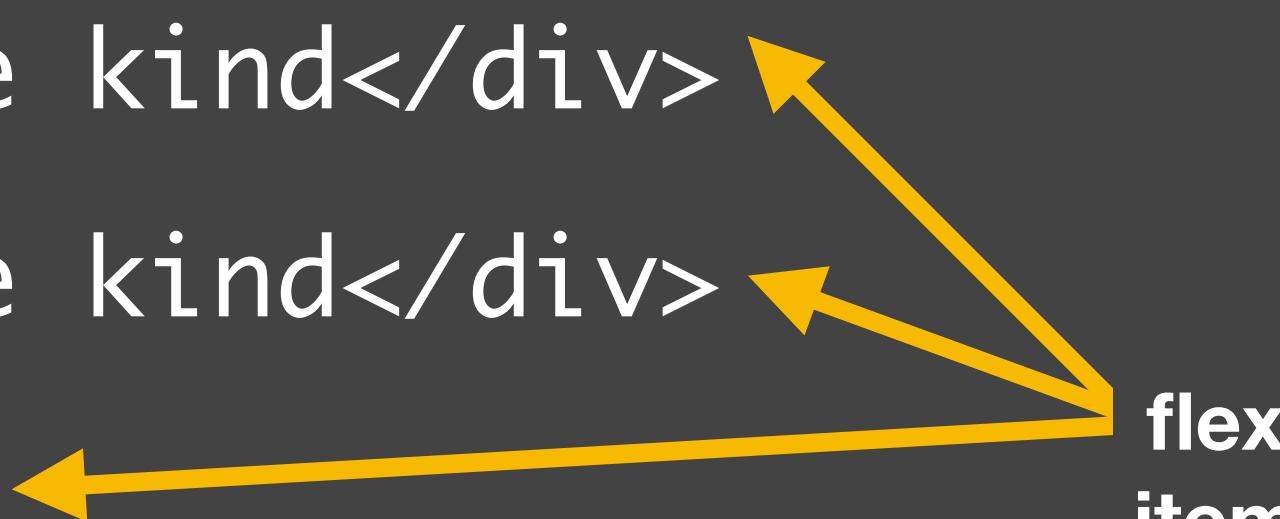
```
#container { display: flex; }
```

zolang ze
'in flow' zijn
exclusief
witruimte,
inclusief
losse tekst

flex container

alle directe kinderen worden flex items

```
<div id="container">  
    <div>Eerste kind</div>  
    <div>Tweede kind</div>  
    Derde kind  
</div>
```



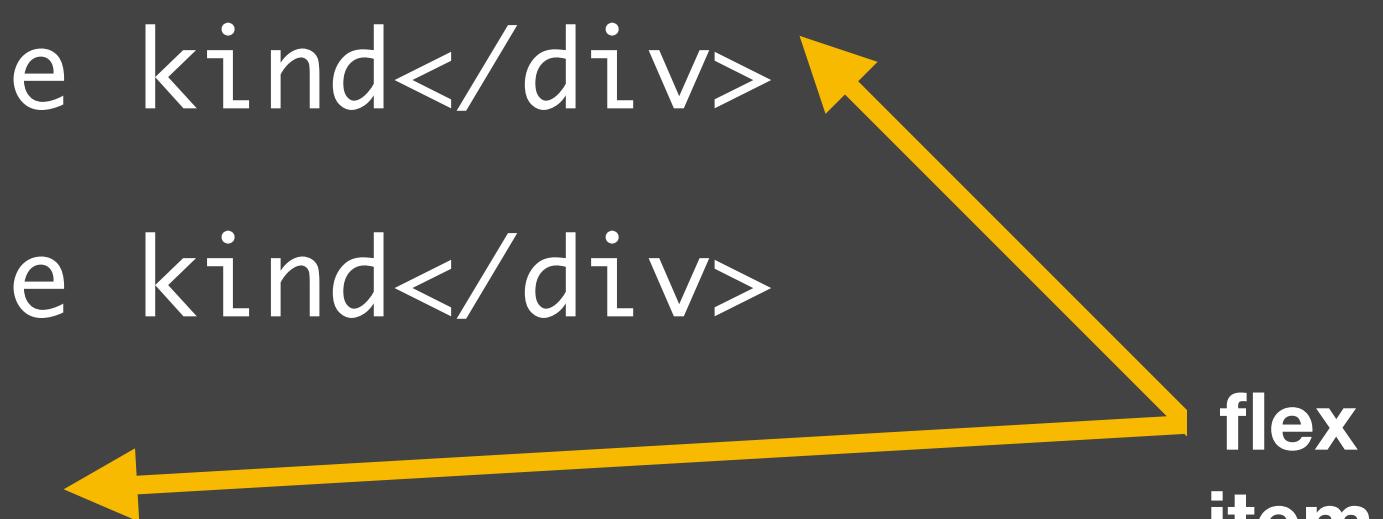
flex item

```
#container { display: flex; }
```

flex container

alle directe kinderen worden flex items

```
<div id="container">  
    <div>Eerste kind</div>  
    <div>Tweede kind</div>  
    Derde kind  
</div>
```



A yellow arrow originates from the text 'flex item' and points to the third child element 'Derde kind'.

```
#container { display: flex; }  
div:nth-child(2) { position: absolute; }
```

**als een element een flex
container wordt, verandert
het effect van sommige CSS**

flex item

werkt niet meer:

- vertical-align
- float

werkt nu:

- flex
- automagische margins
- align-self
- order

**FLEX
ITEM**

flex

```
#item { flex: 1; }
```

**FLEX
ITEM**

flex

```
#item { flex: 0 1 auto; }
```



grow

flex

```
#item { flex: 0 1 auto; }
```

↑ ↑
grow *shrink*

flex

```
#item { flex: 0 1 auto; }  
      ↑   ↑   ↑  
      grow  basis  
           shrink
```

**“basis” kan auto, content
of een lengte zijn (em/px/% etc)**

FLEX ITEM

```
#item { flex: initial; }  
// equivalent aan '0 1 auto'
```

laat item niet
groter worden,
wel kleiner

```
#item { flex: auto; }  
// equivalent aan '1 1 auto'
```

volledig flexibel –
laat item zowel
groter als
kleiner worden

```
#item { flex: none; }  
// equivalent aan '0 0 auto'
```

niet flexibel –
laat item niet
groter of
kleiner worden

OEFENING

Maak een pagina met daarop:

- een rij met 3 kolommen
- een rij met 4 kolommen
- een rij met 5 kolommen

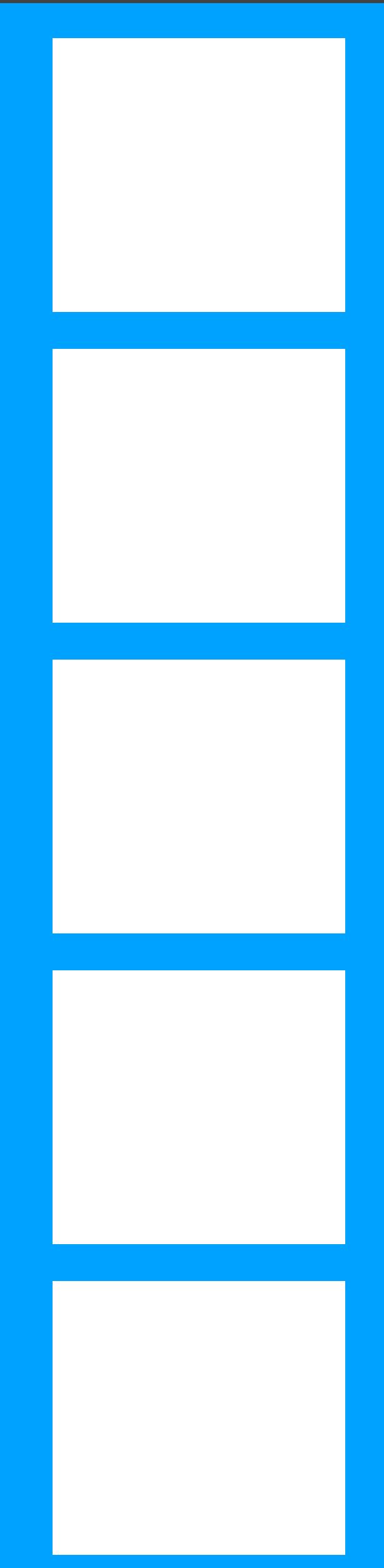
OEFENING

**Lore
m ipsum
dolor sit met**

Tijd over?
Doe dit alleen op
viewports groter
dan 50em.



cross



main axis vs cross axis

flex-direction: column;

main axis vs cross axis

flex-direction: row;



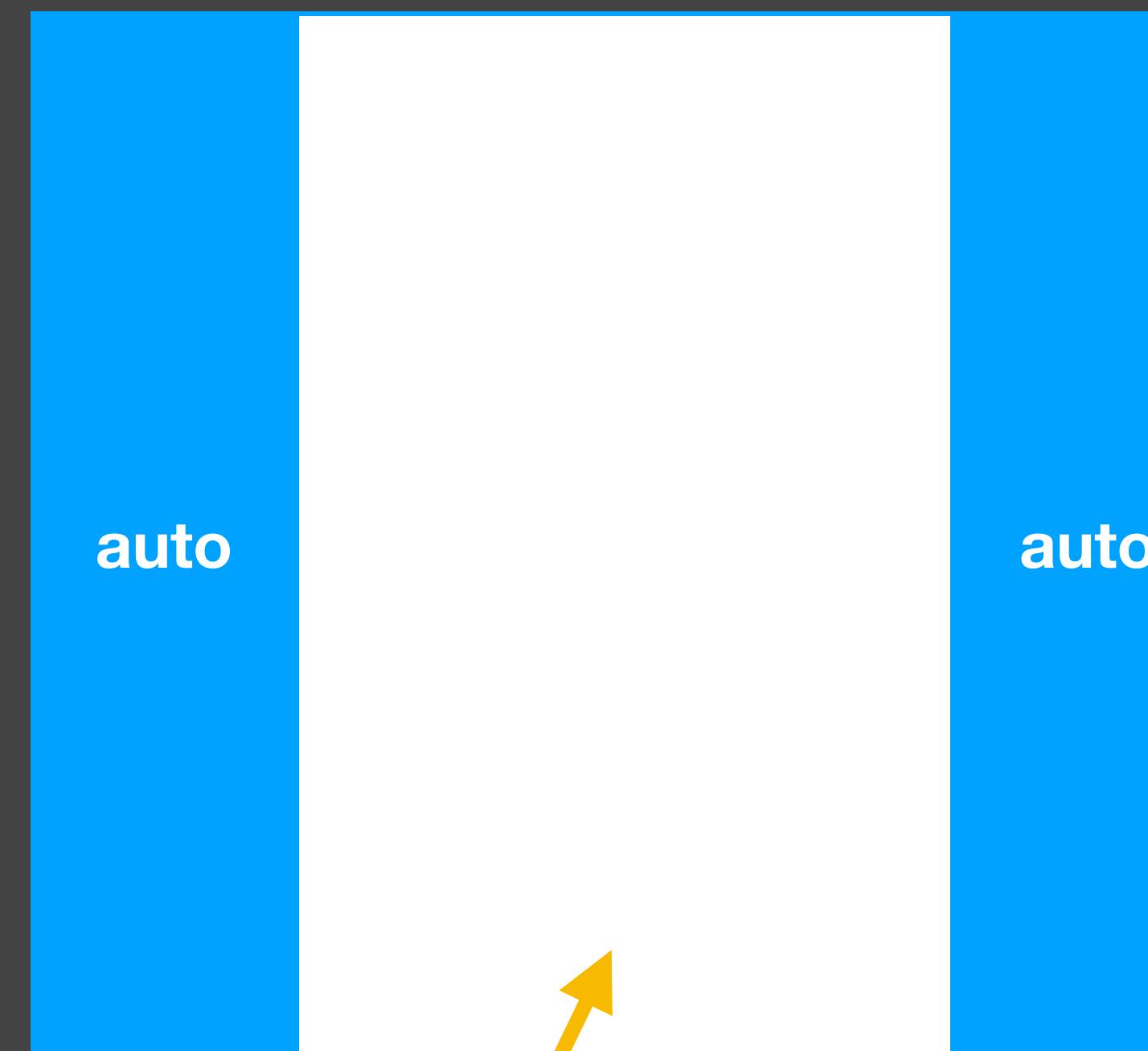
**FLEX
ITEM**

wrapping

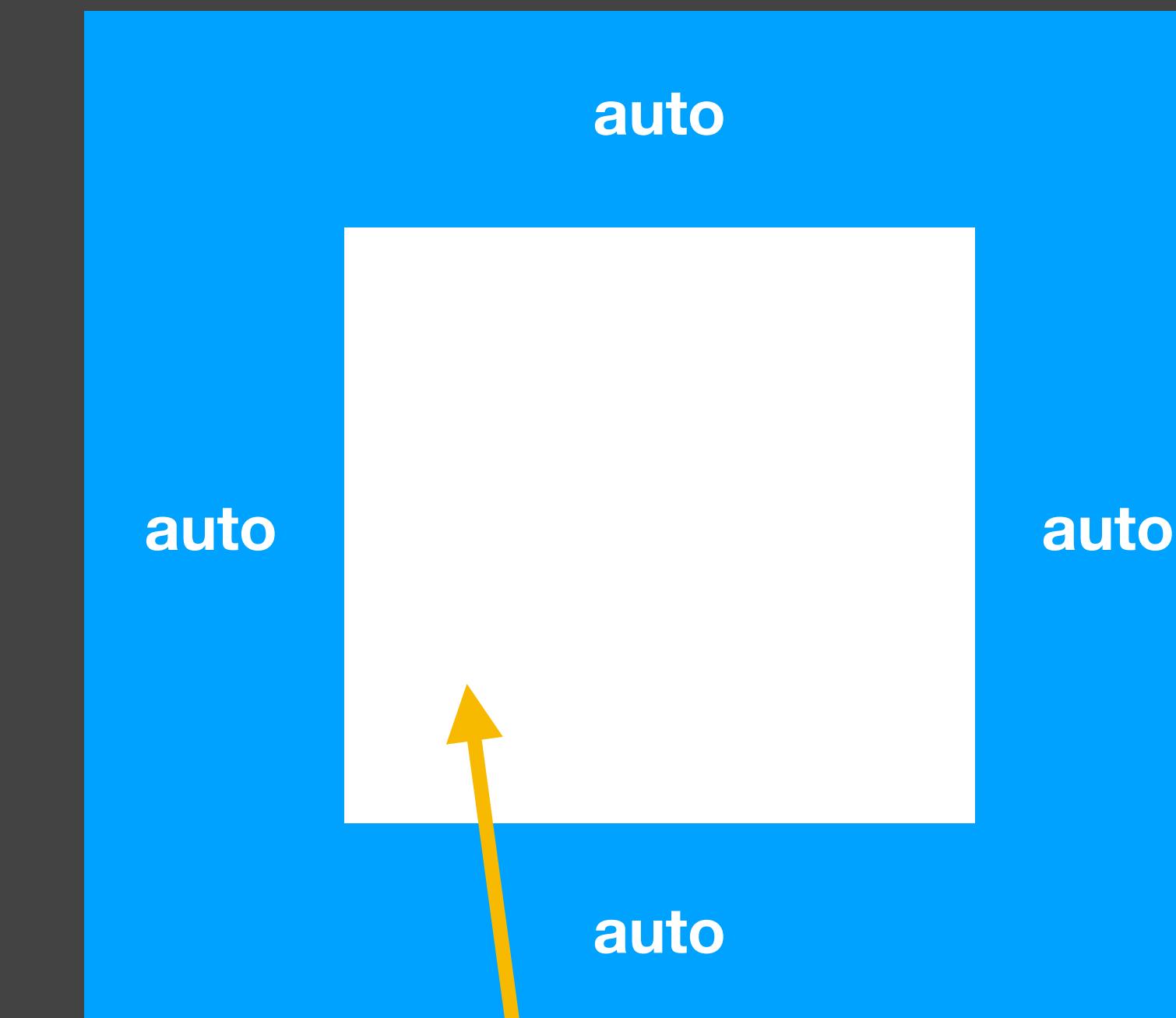
Wil je items over meerdere kolommen/rijen of juist niet?

```
#container {  
    flex-wrap: nowrap | wrap;  
}
```

automagische margins

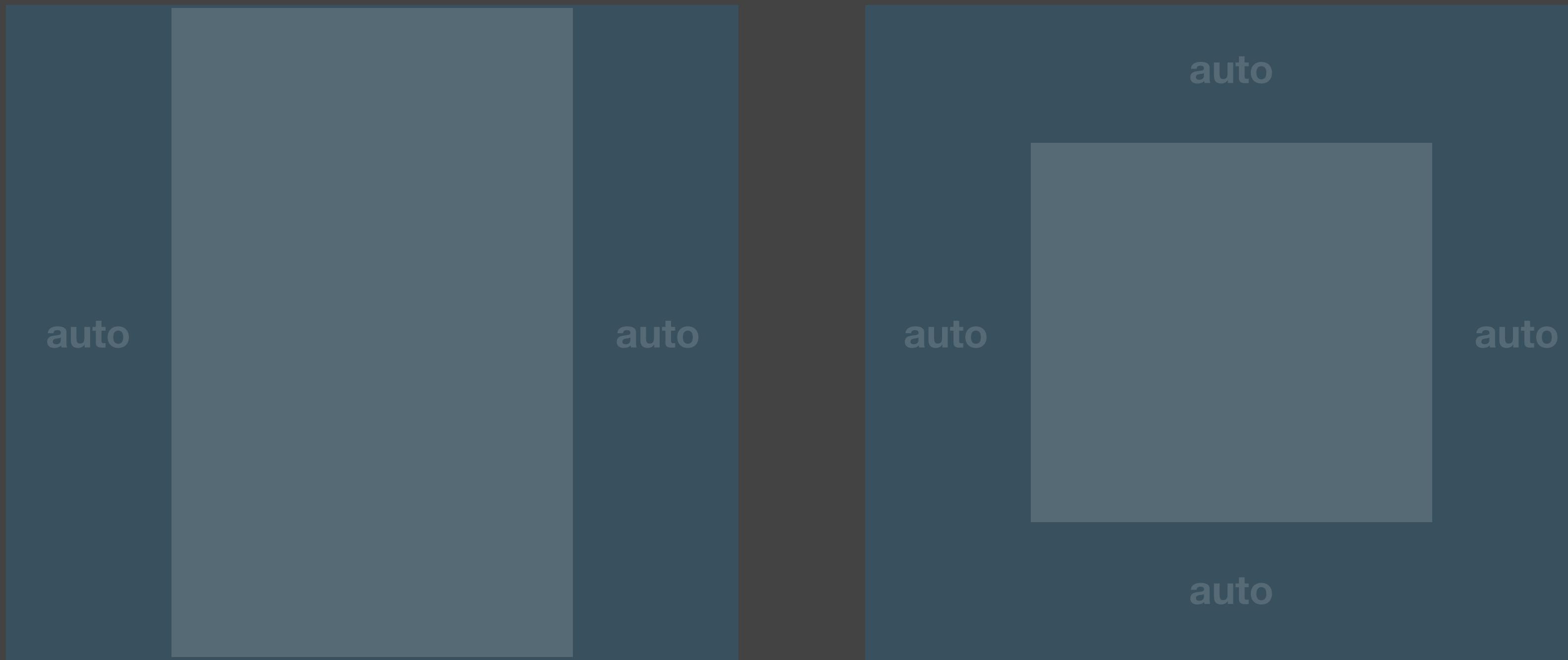


`margin: 0 auto;`



`margin: auto;`

automagische margins



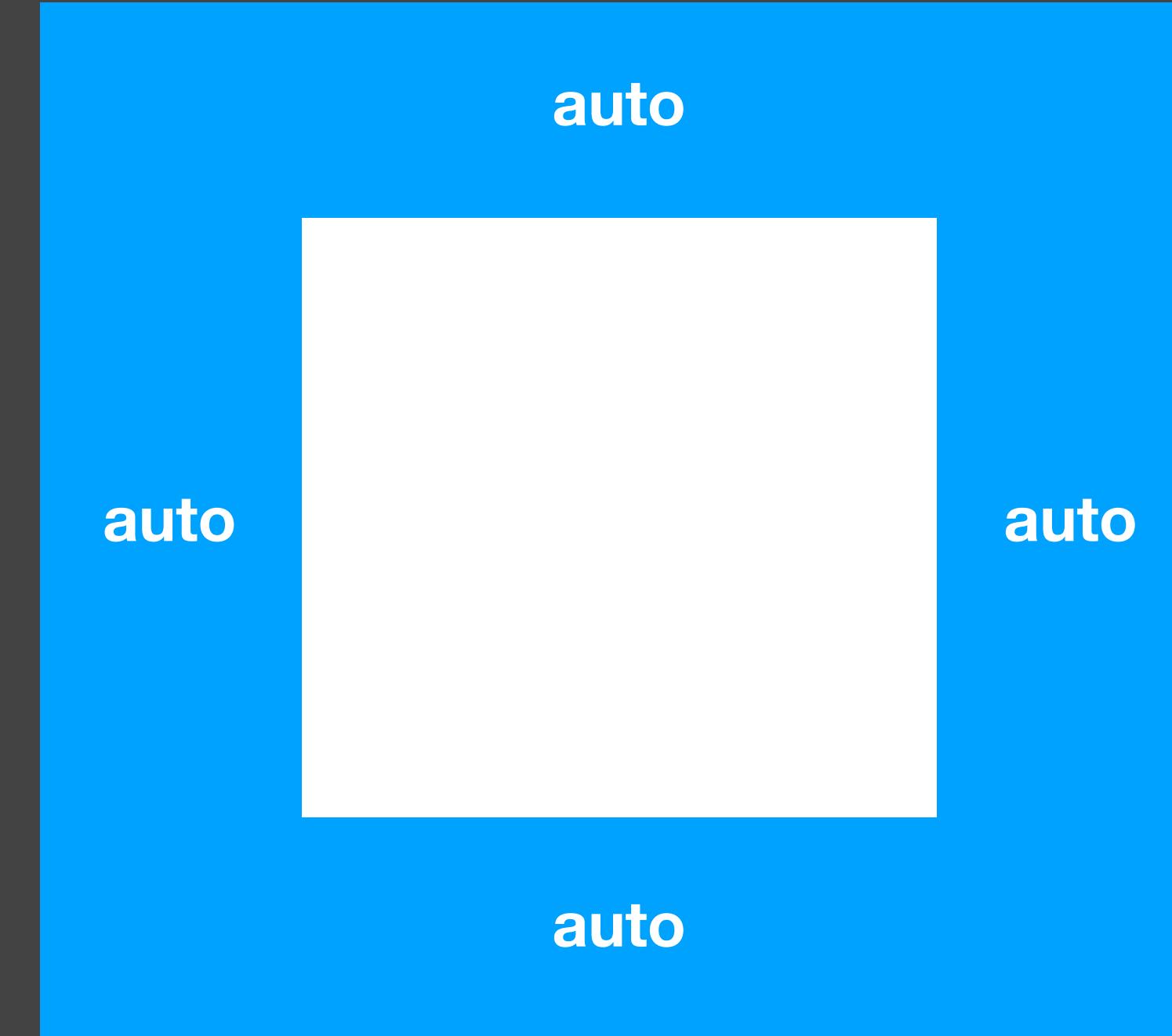
FLEX ITEM

**één item uitlijnen
op de main axis**

**Op kolommen als je kolommen maakt,
op rijen als je rijen maakt.**

**Lijn één item anders uit dan de rest, of, als
je maar één item hebt, verdeel omringende
ruimte automatisch.**

```
#item {  
  margin: auto;  
}
```

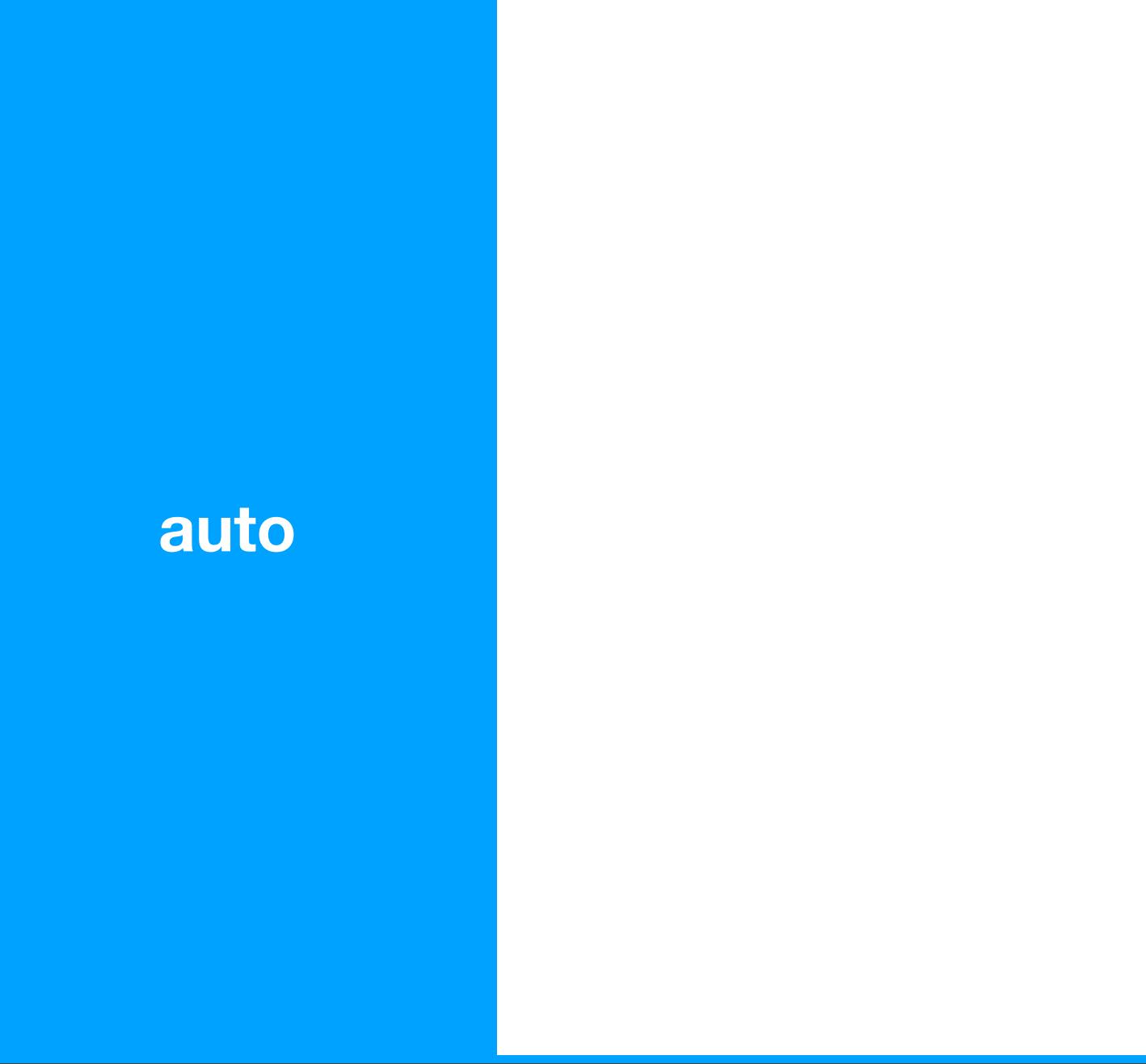


FLEX ITEM

**één item uitlijnen
op de main axis**

**Op kolommen als je kolommen maakt,
op rijen als je rijen maakt.**

**Lijn één item anders uit dan de rest, of, als
je maar één item hebt, verdeel omringende
ruimte automatisch.**



auto

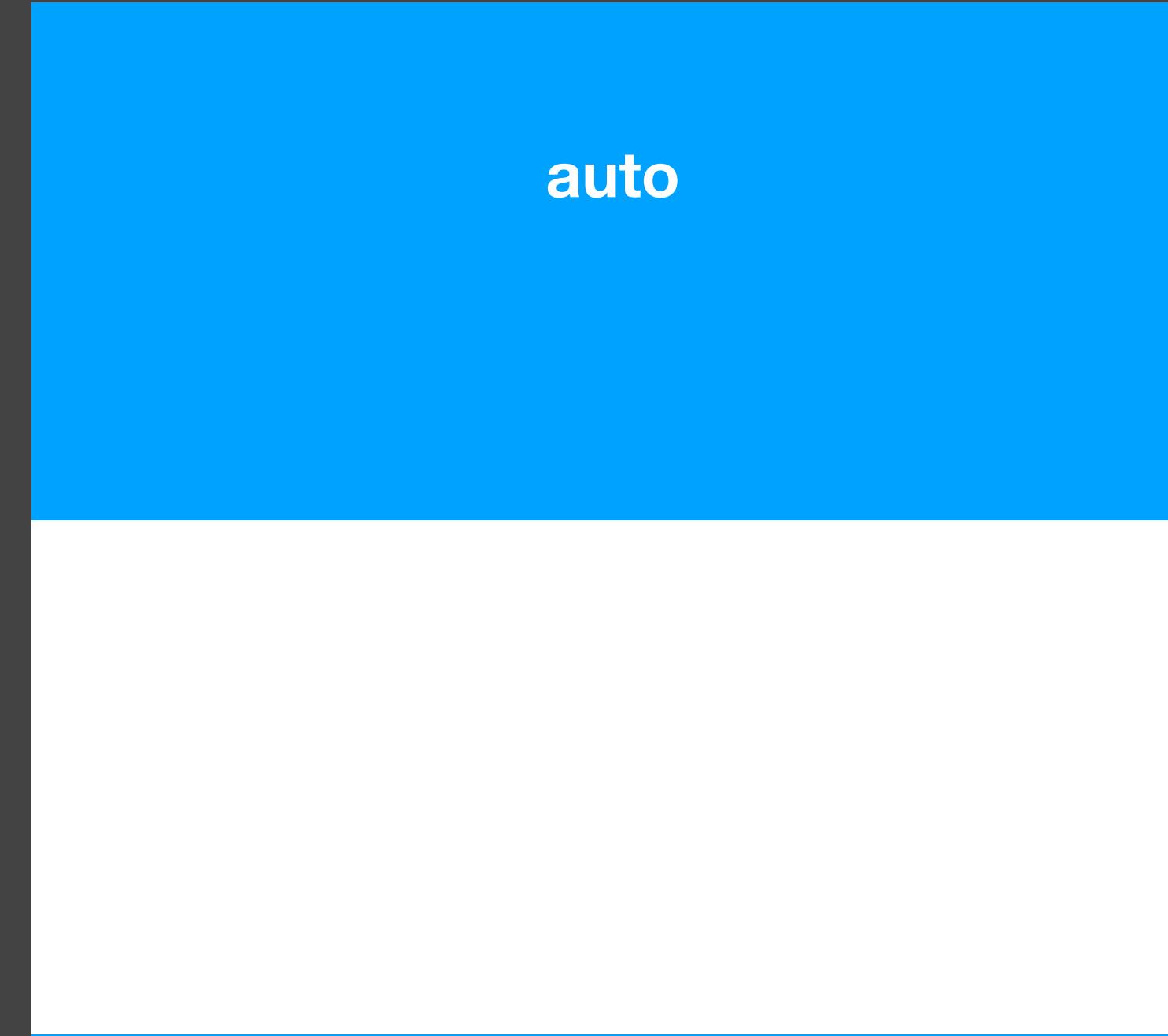
```
#item {  
  margin-left: auto;  
}
```

FLEX ITEM

**één item uitlijnen
op de main axis**

**Op kolommen als je kolommen maakt,
op rijen als je rijen maakt.**

**Lijn één item anders uit dan de rest, of, als
je maar één item hebt, verdeel omringende
ruimte automatisch.**



```
#item {  
  margin-top: auto;  
}
```

FLEX ITEM

één item uitlijnen op de cross axis

Op kolommen als je rijen maakt,
op rijen als je kolommen maakt.

Lijn één item anders uit dan de rest.

```
#item {  
  align-self: flex-start  
    | flex-end | center |  
  baseline | stretch;  
}
```

FLEX CONTAINER

FLEX CONTAINER

**items uitlijnen op
de main axis**

**Op kolommen als je kolommen maakt,
op rijen als je rijen maakt.**

**Zet de items in deze container
gezamenlijk op een plek, bv aan
het begin of het eind.**

```
#container {  
  justify-content:  
    flex-start | flex-  
    end | center |  
    space-between |  
    space-around;  
}
```

FLEX CONTAINER

items uitlijnen op de cross axis

Op kolommen als je rijen maakt,
op rijen als je kolommen maakt.

Zet de items in deze container
gezamenlijk op een plek, bv aan
het begin of het eind.

```
#container {  
  align-items: stretch  
  | flex-start | flex-end  
  | center | baseline;  
}
```

FLEX CONTAINER

**meerdere regels
uitlijnen op de
cross axis**

**Op kolommen als je rijen maakt,
op rijen als je kolommen maakt.**

```
#container {  
    align-content: stretch  
    | flex-start | flex-  
    end | center | space-  
    between | space-  
    around;  
}
```

The Daily Cascade: meta info

THE DAILY  **CASCADE**

Since 1994 · CSS news for all

24 November 2017

The Daily Cascade: highlights

New box alignment spec in the works

Subgrid and the
need for it

order

#item1

#item2

#item3

#item4

#item3 { order: 2; }

**Let op met order. De locatie
van items kan onlogisch worden
voor je keyboardgebruiker.**

Zet de tweede highlight
aan het begin

Subgrid and the
need for it

New box alignment spec in the works

Grid Layout

**Met CSS Grid Layout kun je
ruimtes definiëren en daar
content in (laten) plaatsen**



neu Japhabet

a possibility
for
the
new
development

een
mogelijkheid
voor
de
nieuwe
ontwikkeling

une
possibilité
pour
le
nouveau
développement

eine
Gelegenheit
für
die
neue
Entwicklung

In
Introduction
for
the
programmed
typography

j b c d e f g h
i t l o n o p
q r s t u v w
x y z





CSS Grid Layout Module Level 1

W3C Candidate Recommendation, 09 May 2017

TABLE OF CONTENTS

- 1 Introduction**
 - 1.1 Background and Motivation
 - 1.1.1 Adapting Layouts to Available Space
 - 1.1.2 Source-Order Independence
- 2 Overview**
 - 2.1 Declaring the Grid
 - 2.2 Placing Items
 - 2.3 Sizing the Grid
- 3 Grid Layout Concepts and Terminology**
 - 3.1 Grid Lines
 - 3.2 Grid Tracks and Cells
 - 3.3 Grid Areas
- 4 Reordering and Accessibility**
- 5 Grid Containers**
 - 5.1 Establishing Grid Containers: the 'grid' and 'inline-grid' 'display' values
 - 5.2 Sizing Grid Containers
 - 5.3 Clamping Overly Large Grids
- 6 Grid Items**
 - 6.1 Grid Item Display
 - 6.2 Grid Item Sizing
 - 6.3 Reordered Grid Items: the 'order' property
 - 6.4 Grid Item Margins and Paddings
 - 6.5 Z-axis Ordering: the 'z-index' property
 - 6.6 Implied Minimum Size of Grid Items
- 7 Defining the Grid**
 - 7.1 The Explicit Grid
 - 7.2 Explicit Track Sizing: the 'grid-template-rows' and 'grid-template-columns' properties
 - 7.2.1 Named Grid Lines: the '<custom-ident>*' syntax

This version: <https://www.w3.org/TR/2017/CR-css-grid-1-20170509/>

Latest published version: <https://www.w3.org/TR/css-grid-1/>

Editor's Draft: <https://drafts.csswg.org/css-grid/>

Previous Versions:

- <https://www.w3.org/TR/2017/CR-css-grid-1-20170209/>
- <https://www.w3.org/TR/2016/WD-css-grid-1-20160519/>
- <https://www.w3.org/TR/2015/WD-css-grid-1-20150917/>
- <https://www.w3.org/TR/2015/WD-css-grid-1-20150806/>
- <https://www.w3.org/TR/2015/WD-css-grid-1-20150317/>
- <https://www.w3.org/TR/2014/WD-css-grid-1-20140513/>
- <https://www.w3.org/TR/2014/WD-css-grid-1-20140123/>
- <https://www.w3.org/TR/2013/WD-css3-grid-layout-20130402/>
- <https://www.w3.org/TR/2012/WD-css3-grid-layout-20121106/>

Test Suite: http://test.csswg.org/suites/css-grid-1_dev/nightly-unstable/

Issue Tracking:

- [Disposition of Comments](#)
- [Inline In Spec](#)
- [GitHub Issues](#)

Editors:

- [Tab Atkins Jr. \(Google\)](#)
- [Elika J. Etemad / fantasai \(Invited Expert\)](#)
- [Rossen Atanassov \(Microsoft\)](#)

Former Editors:

- [Alex Mogilevsky \(Microsoft Corporation\)](#)
- [Phil Cupp \(Microsoft Corporation\)](#)

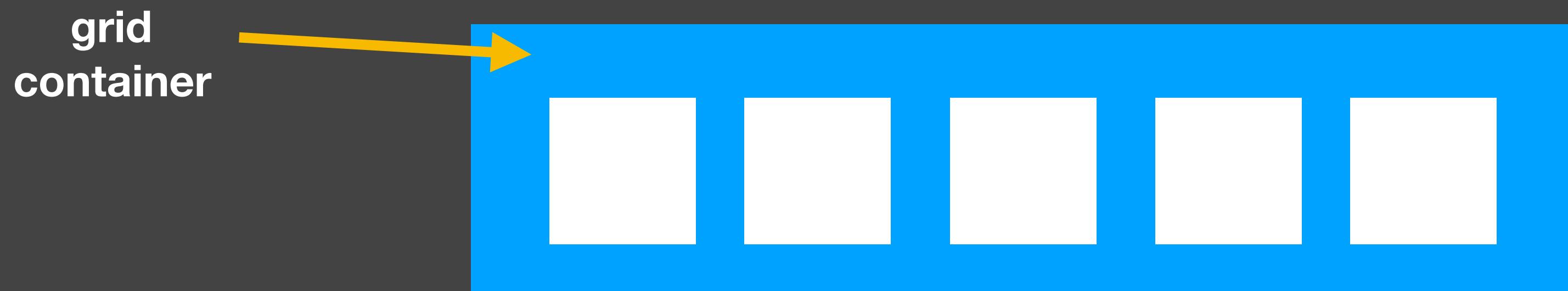
Copyright © 2017 W3C® ([MIT](#), [ERCIM](#), [Keio](#), [Beihang](#)). W3C liability, trademark and disclaimer apply.

**Het mooie van CSS Grid
is dat je grids met veranderende
content kunnen meegroeien**

een grid definiëren

```
#container { display: grid; }
```

grid container



```
#container { display: grid; }
```

grid container

werkt niet meer:

- column-*
- float
- clear
- vertical-align
- ::first-[item|letter]

werkt nu:

- grid-template-columns
- grid-template-rows
- grid-auto-flow
- justify-content
- align-content

grid items

werkt niet meer:

- vertical-align
- ::first-[item|letter]

werkt nu:

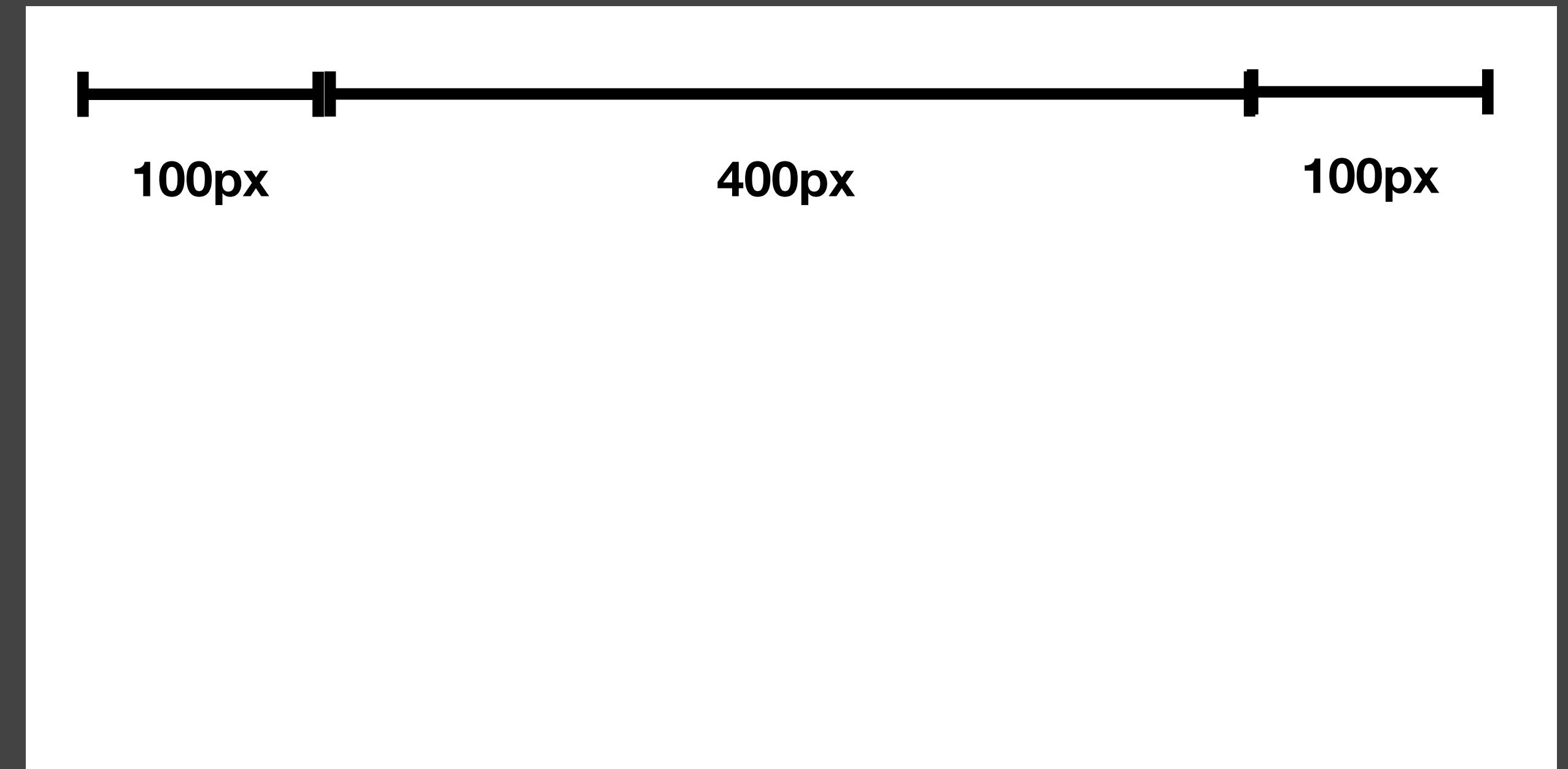
- grid-area
- grid-[column|row]
- order
- align-self
- justify-self

Je kunt in grid layout
~ dezelfde alignment properties
gebruiken als in flexbox



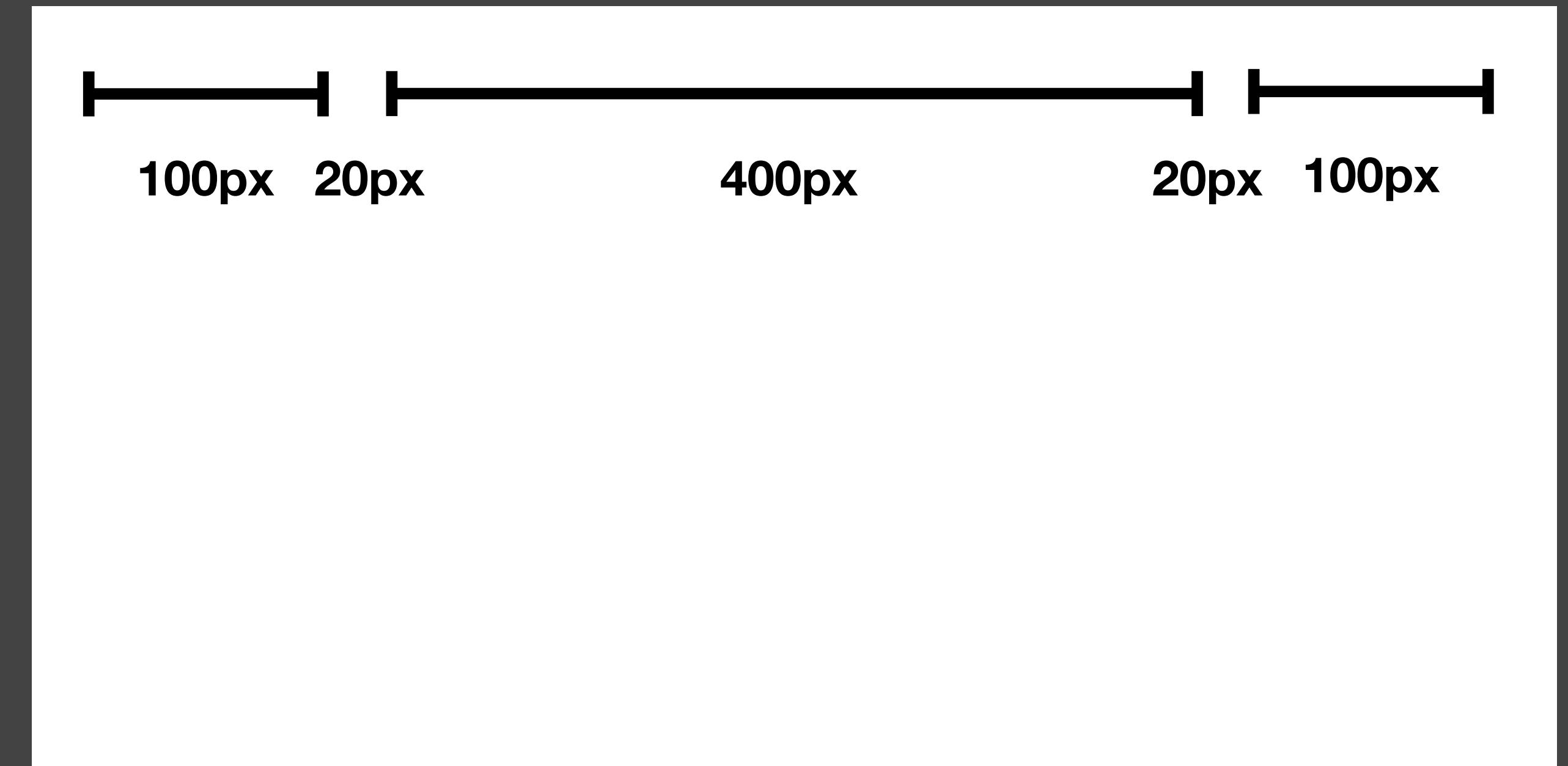
een grid definiëren

```
#container {  
    display: grid;  
    grid-template-columns:  
        100px 400px 100px;  
}
```



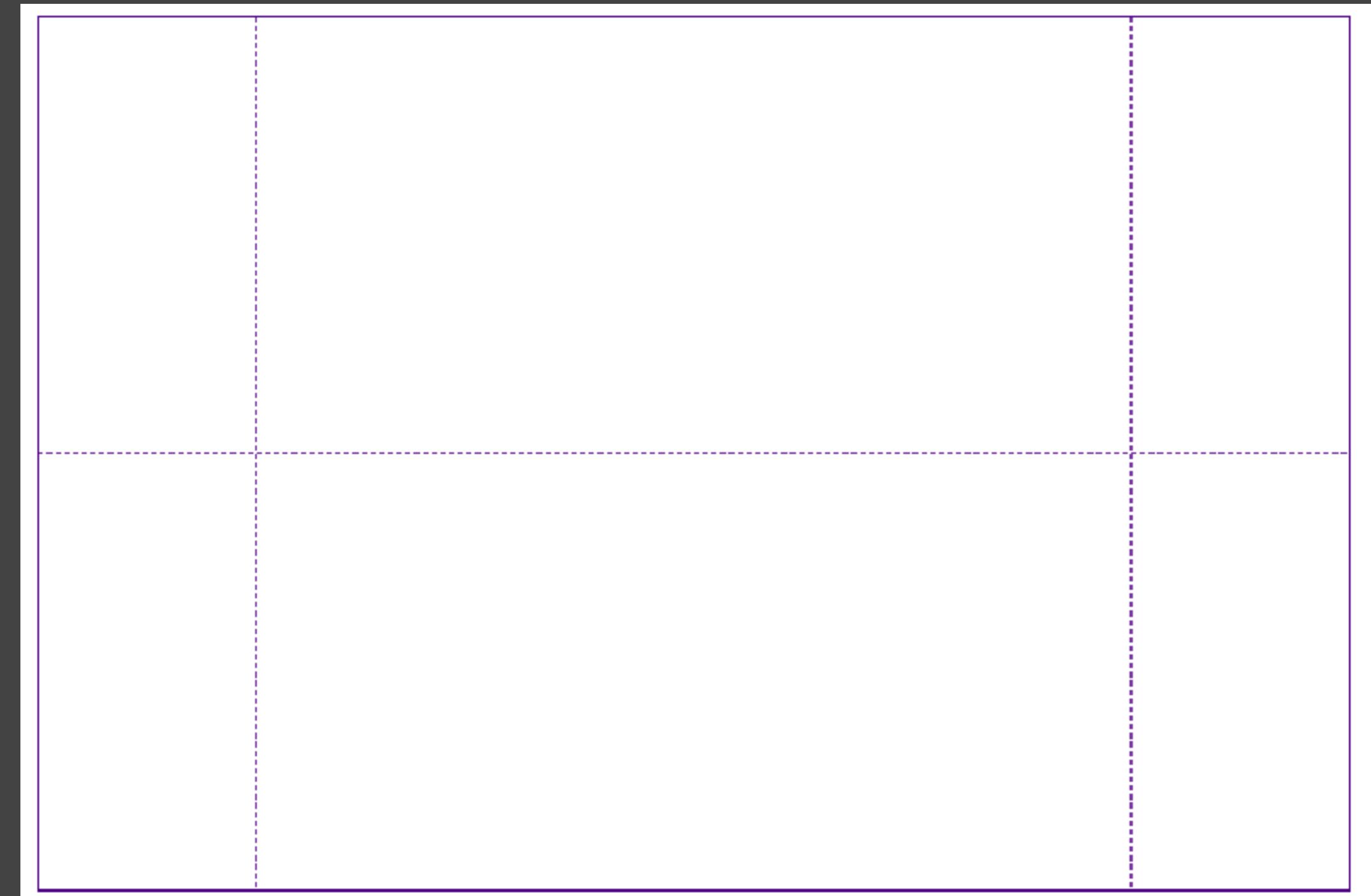
een grid definiëren

```
#container {  
    display: grid;  
    grid-template-columns:  
        100px 400px 100px;  
    grid-gap: 20px;  
}
```



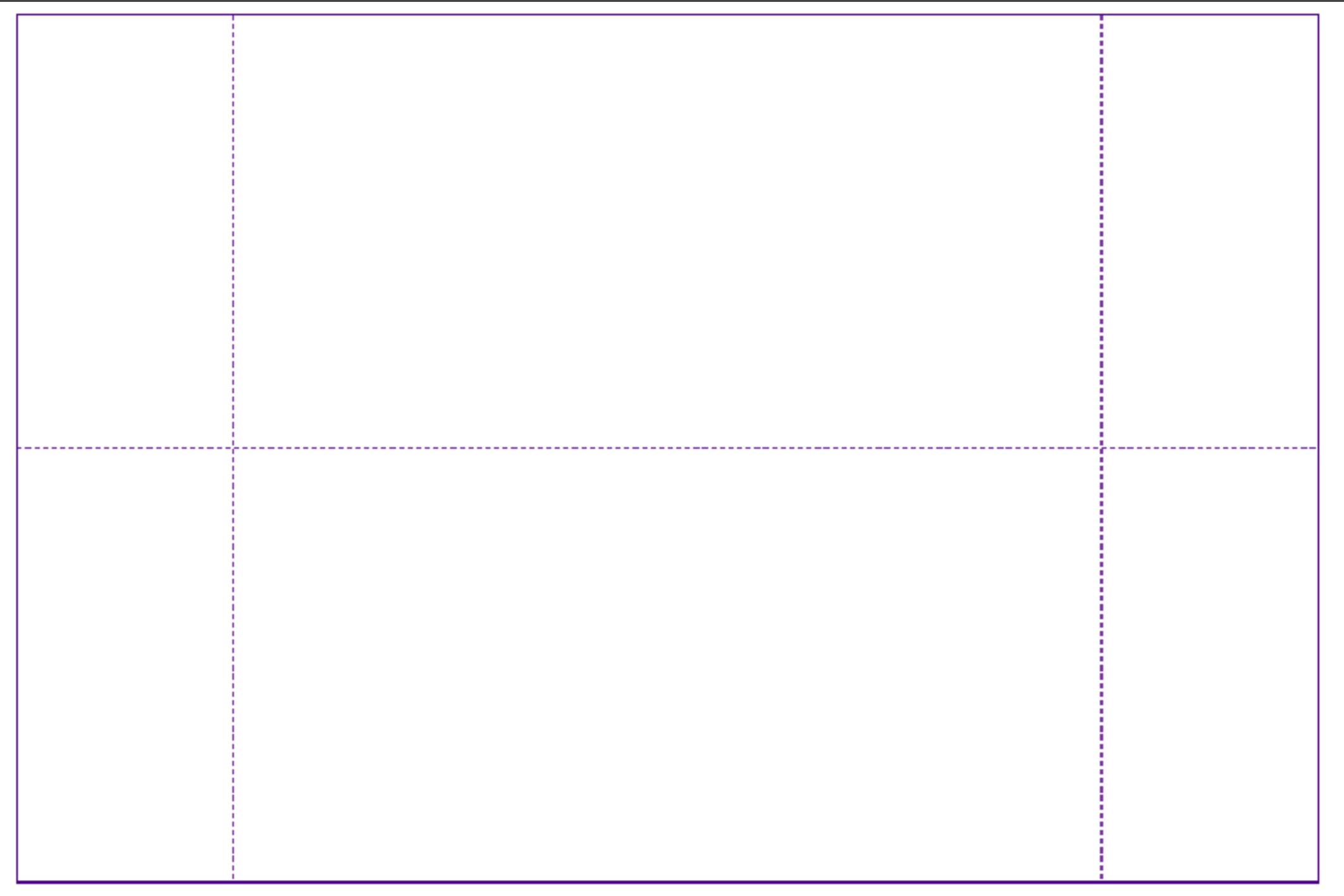
een grid definiëren

```
#container {  
    display: grid;  
    grid-template-columns:  
        100px 400px 100px;  
    grid-template-rows:  
        200px 200px;  
}
```



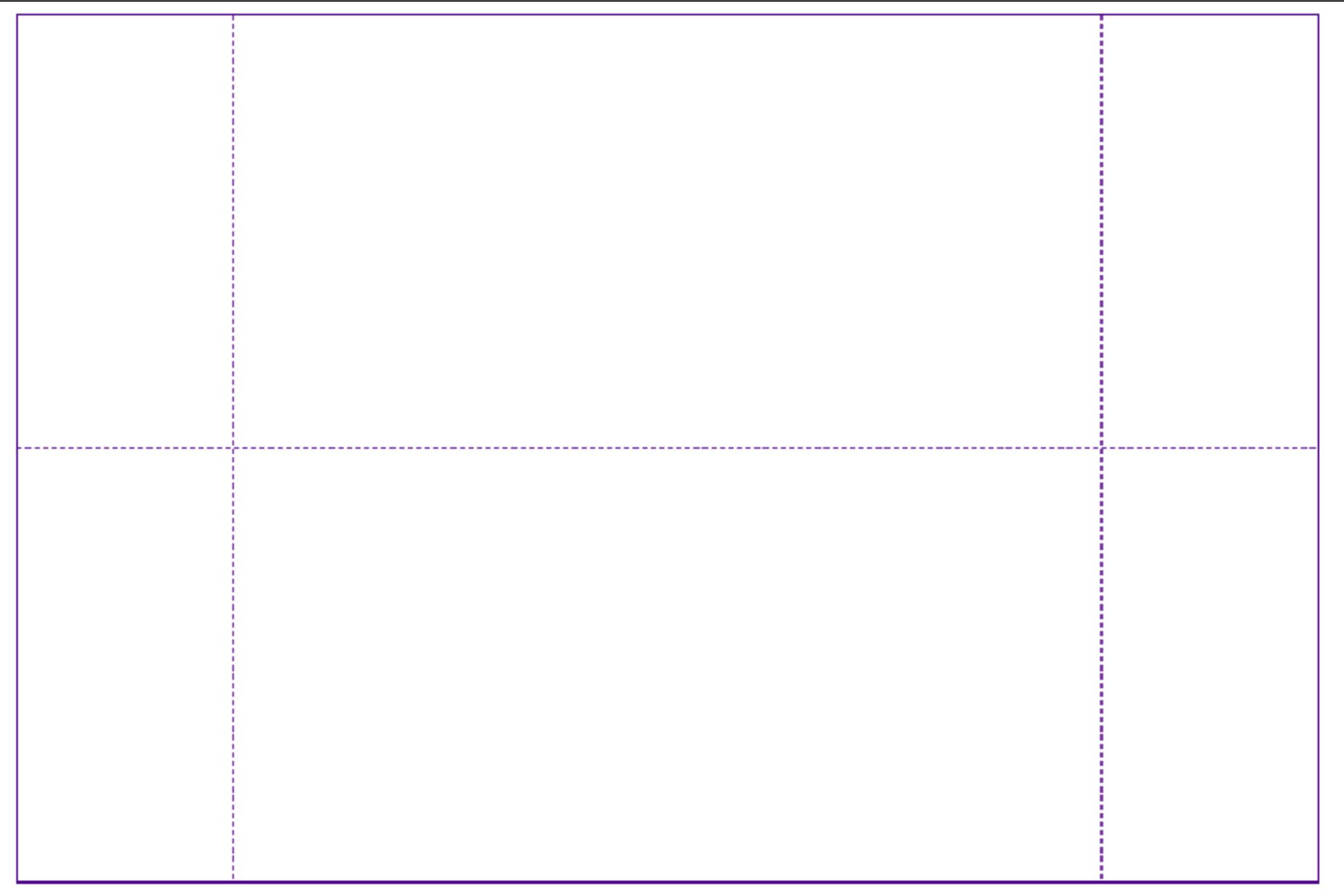
flexibele units voor grid tracks: fr

```
#container {  
    display: grid;  
    grid-template-columns:  
        1fr 3fr 1fr;  
    grid-template-rows:  
        1fr 1fr;  
}
```



flexibele units voor grid tracks: ch

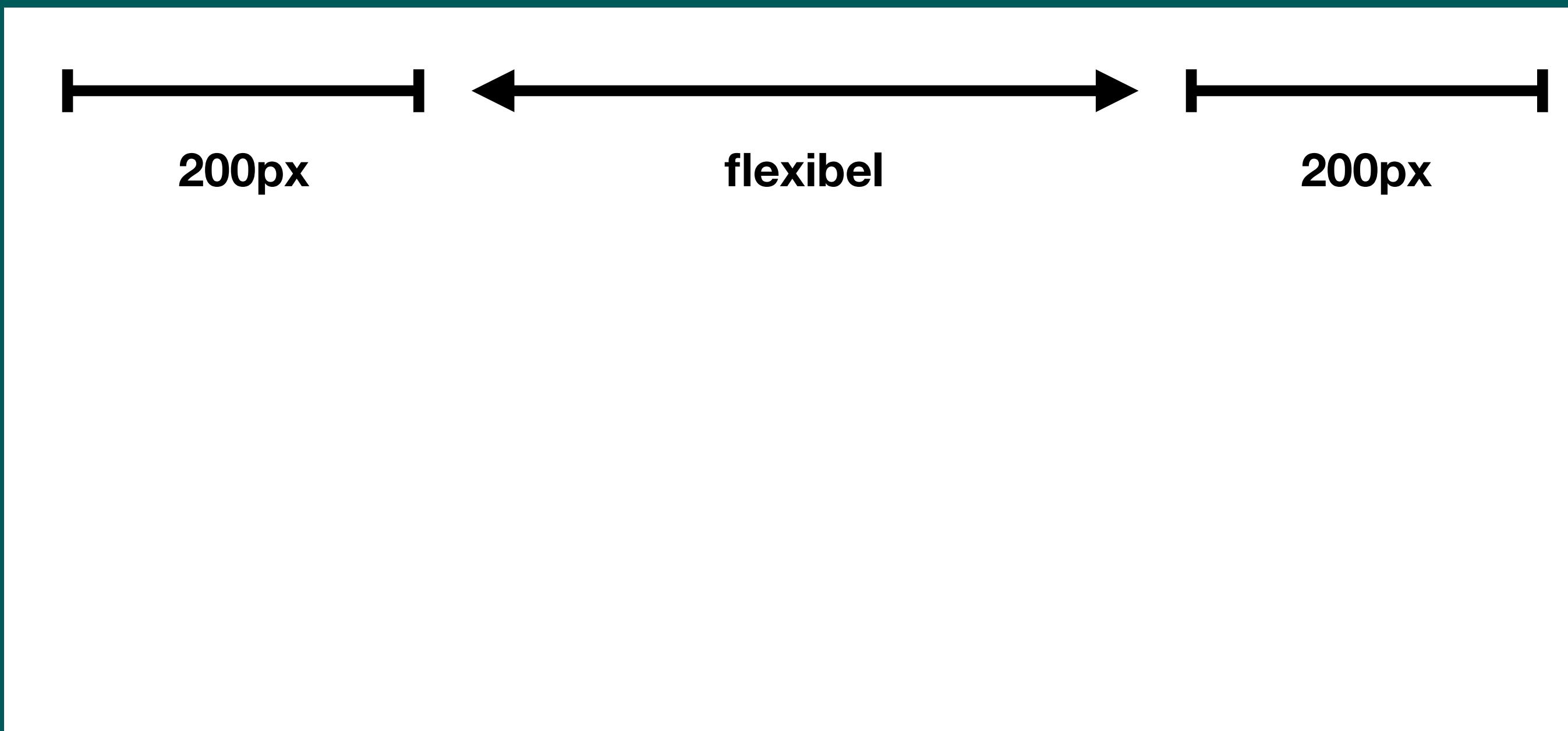
```
#container {  
    display: grid;  
    grid-template-columns:  
        1fr 60ch 1fr;  
    grid-template-rows:  
        1fr 1fr;  
}
```



flexibele units voor grid tracks: minmax()

```
#container {  
    display: grid;  
    grid-template-columns:  
        1fr minmax(40em, 50em) 1fr;  
    grid-template-rows:  
        1fr 1fr;  
}
```

The Daily Cascade: artikelen in "Holy Grail" layout



THE DAILY CASCADE

Since 1994 · CSS news for all

24 November 2017

This daily is made with
love.

The Story of CSS Grid, from Its Creators

by Aaron Gustafson · October 19, 2017

Designers have used grids for centuries. And after more than 20 years of waiting, they are finally here for the browser. This is the story of CSS Grid. It took a lot of people in the right place and at the right time to make it happen.

Please donate.

Practical CSS Grid: Adding Grid to an Existing Design

by Eric Meyer · March 23, 2017

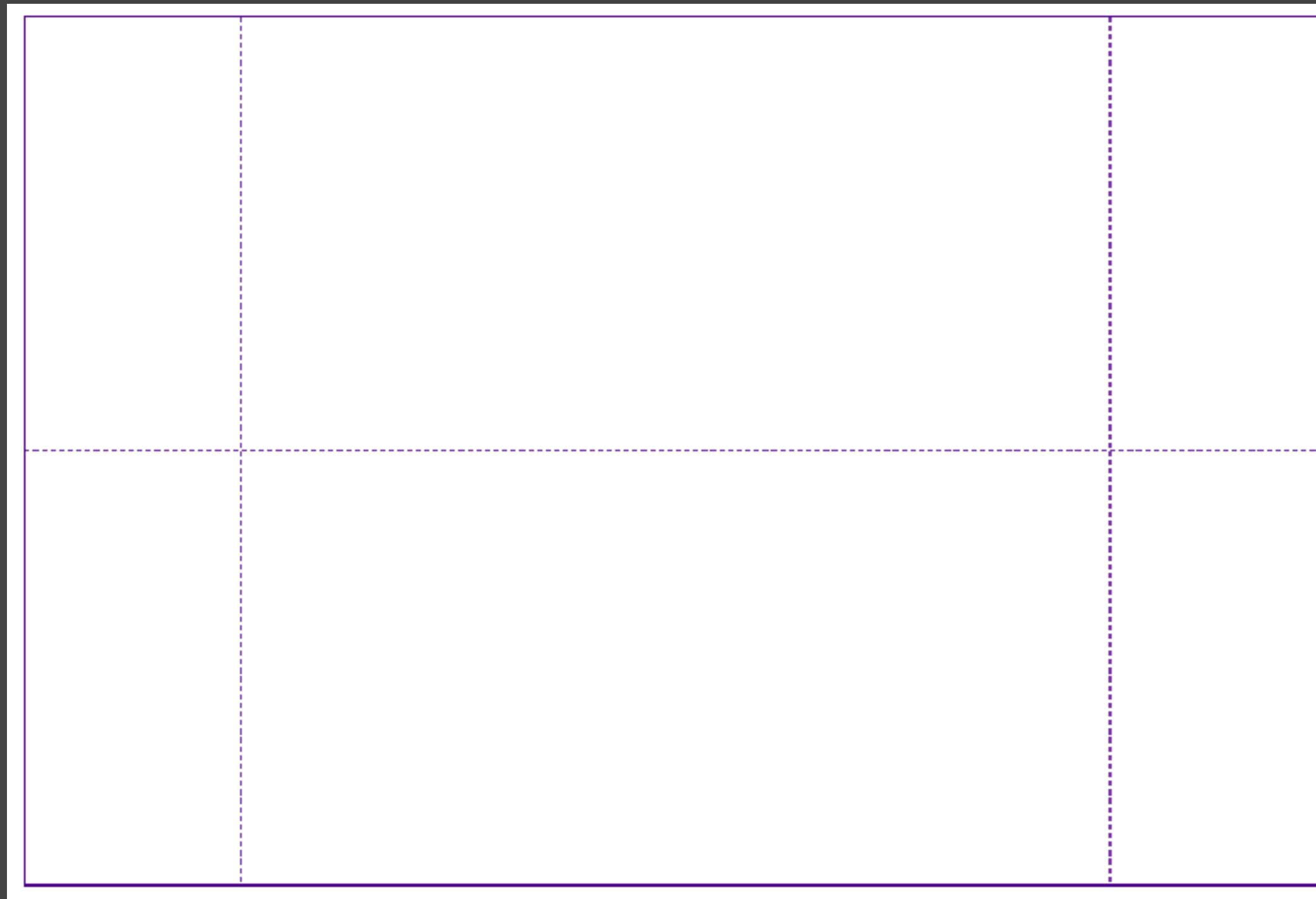
CSS Grid is here—and easier than you might expect. Eric Meyer shows us how to use Grid on an existing design without breaking things for non-grid browsers. With pictures! Also a couple of gotchas.

Learning from Lego: A Step Forward in Modular Web Design

by Samantha Zhang · December 22, 2016

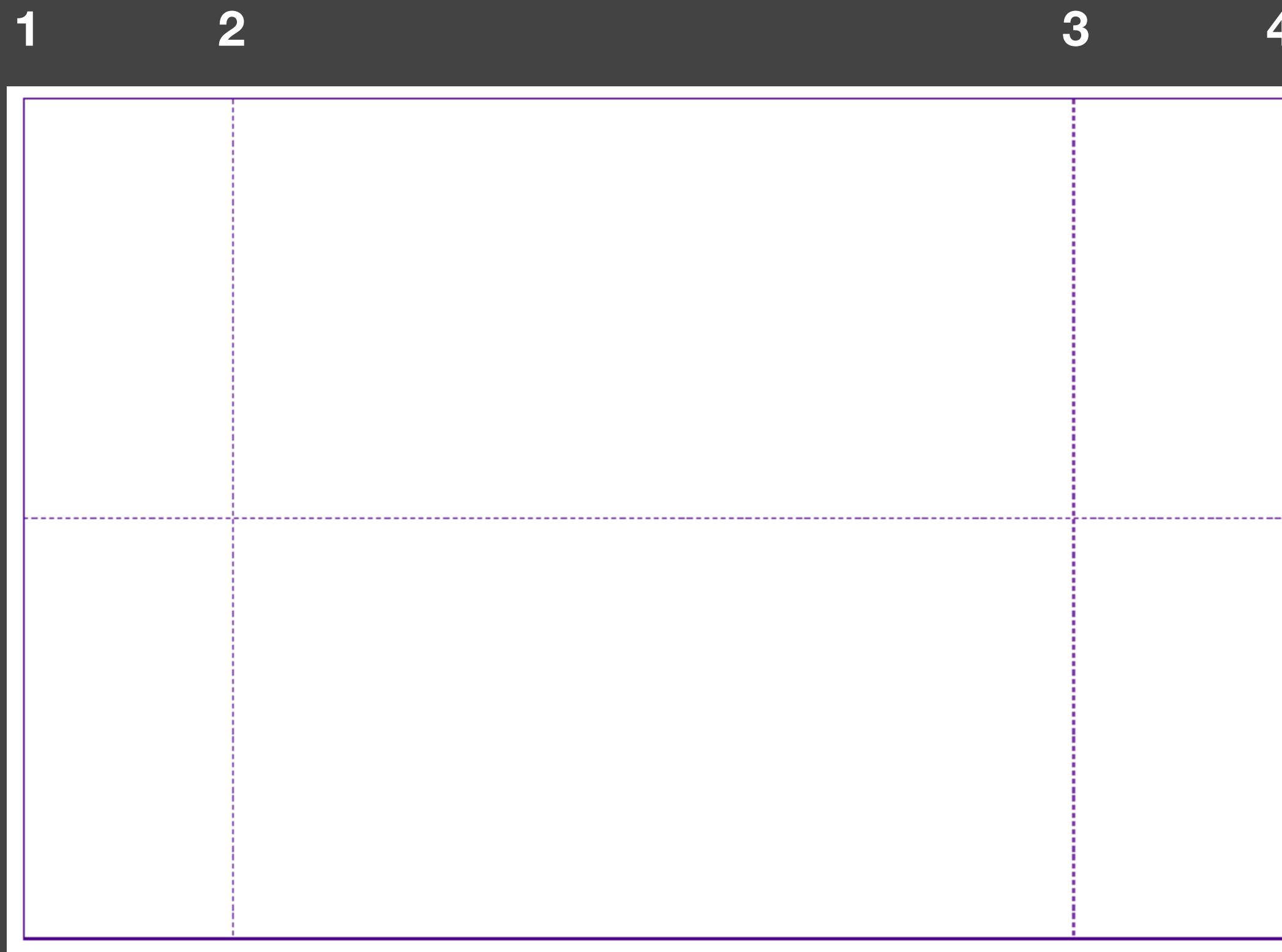
Tijd over?
Positioneer de
content aan de
onderkant

items op het grid plaatsen



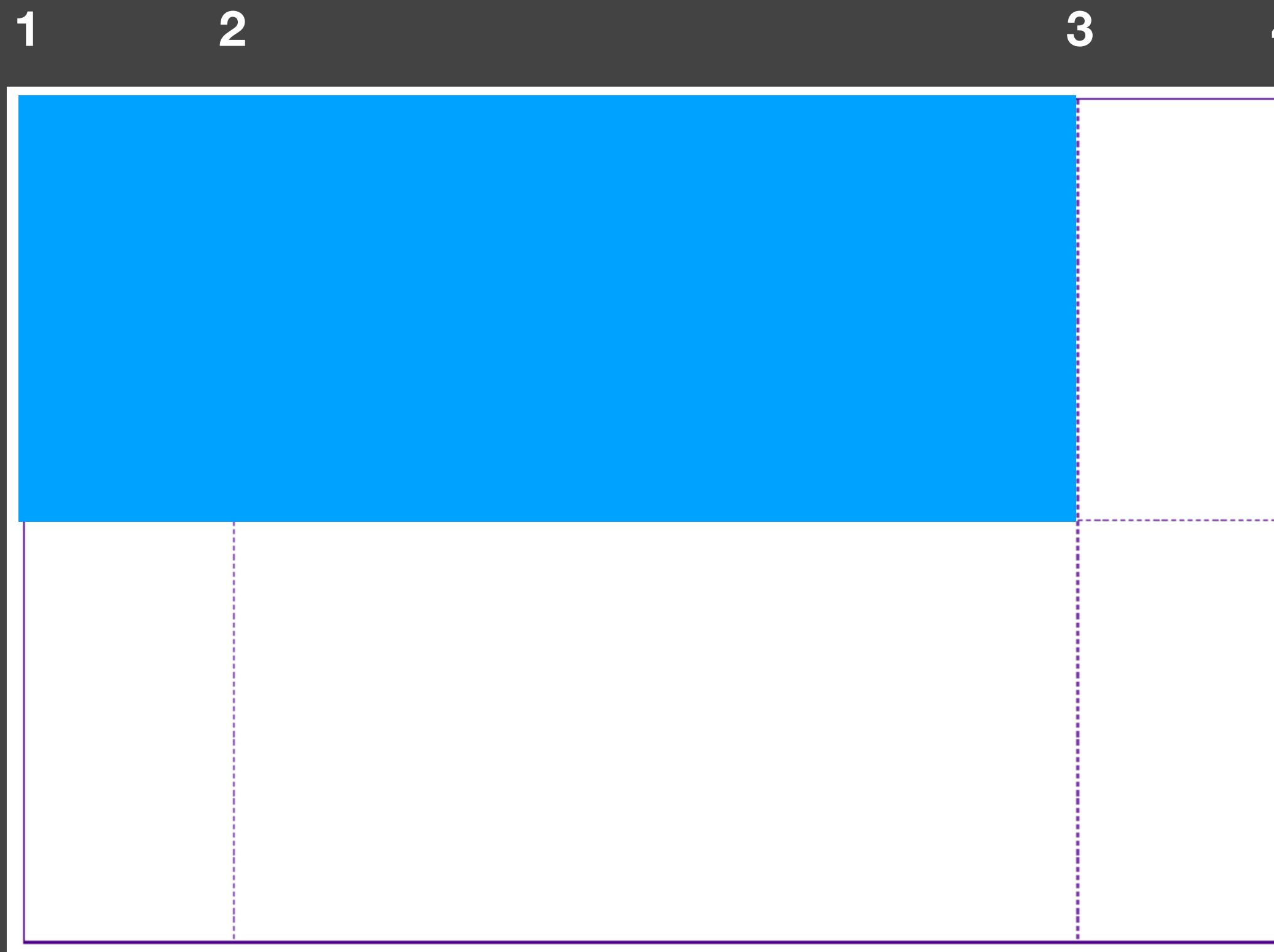
```
#item {  
    grid-column-start: 1;  
    grid-column-end: 3;  
}
```

items op het grid plaatsen



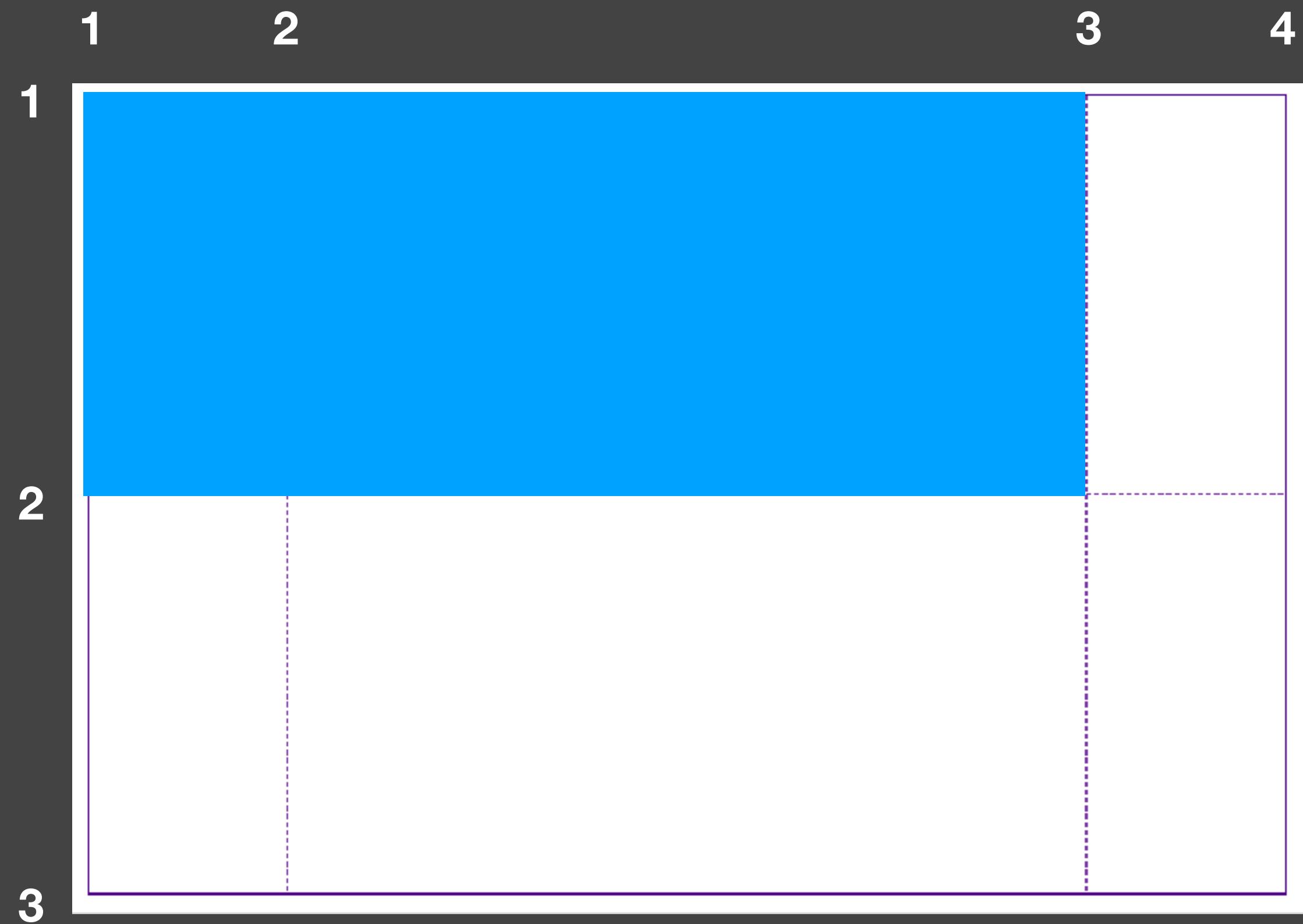
```
#item {  
    grid-column-start: 1;  
    grid-column-end: 3;  
}
```

items op het grid plaatsen



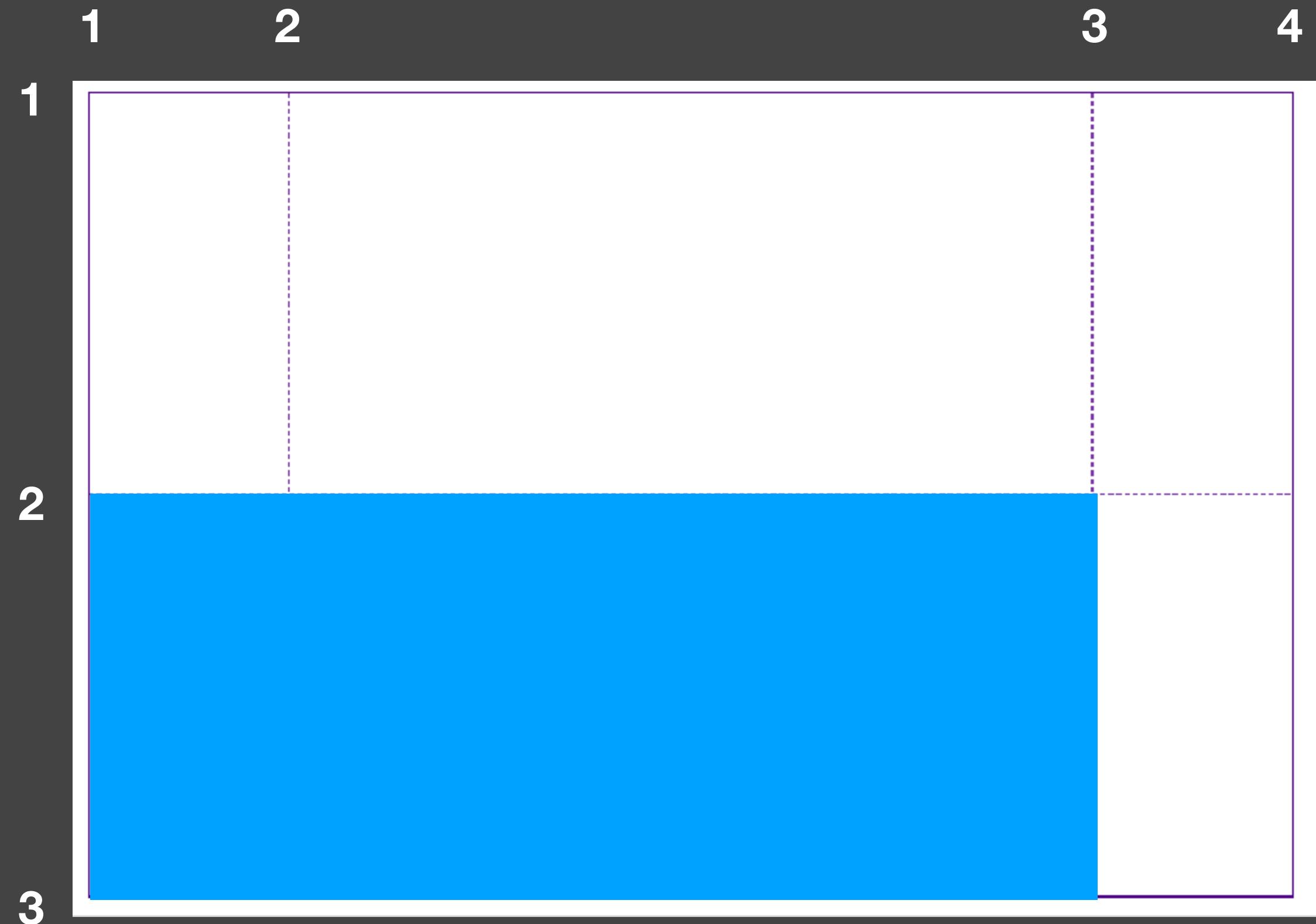
```
#item {  
    grid-column-start: 1;  
    grid-column-end: 3;  
}
```

items op het grid plaatsen



```
#item {  
    grid-column-start: 1;  
    grid-column-end: 3;  
}
```

items op het grid plaatsen



```
#item {  
    grid-column-start: 1;  
    grid-column-end: 3;  
    grid-row-start: 2;  
    grid-row-end: 3;  
}
```

je kunt grid lines ook
namen geven

```
#container {  
    grid-template-columns:  
        [sidebar-start] 1fr [sidebar-end] 4fr;  
}
```

```
#item {  
    grid-column-start: sidebar-start;  
    grid-column-end: sidebar-end;  
}
```

OEFENING

The Daily Cascade: bedenk zelf een grid en lijn de artikelen daarop uit

**Let op: in deze oefening zit er
geen wrapper om de artikelen**

OEFENING

Grids van Wim Crouwel

Georg Baselitz

stedelijk van abbemuseum eindhoven

dagelijks geopend
van 10-17 uur
zon- en feestdagen
van 13-17 uur
dinsdag- en
donderdagavond
van 20-22 uur

hipogima

panelen van iri maruki en toshiko akamatsu

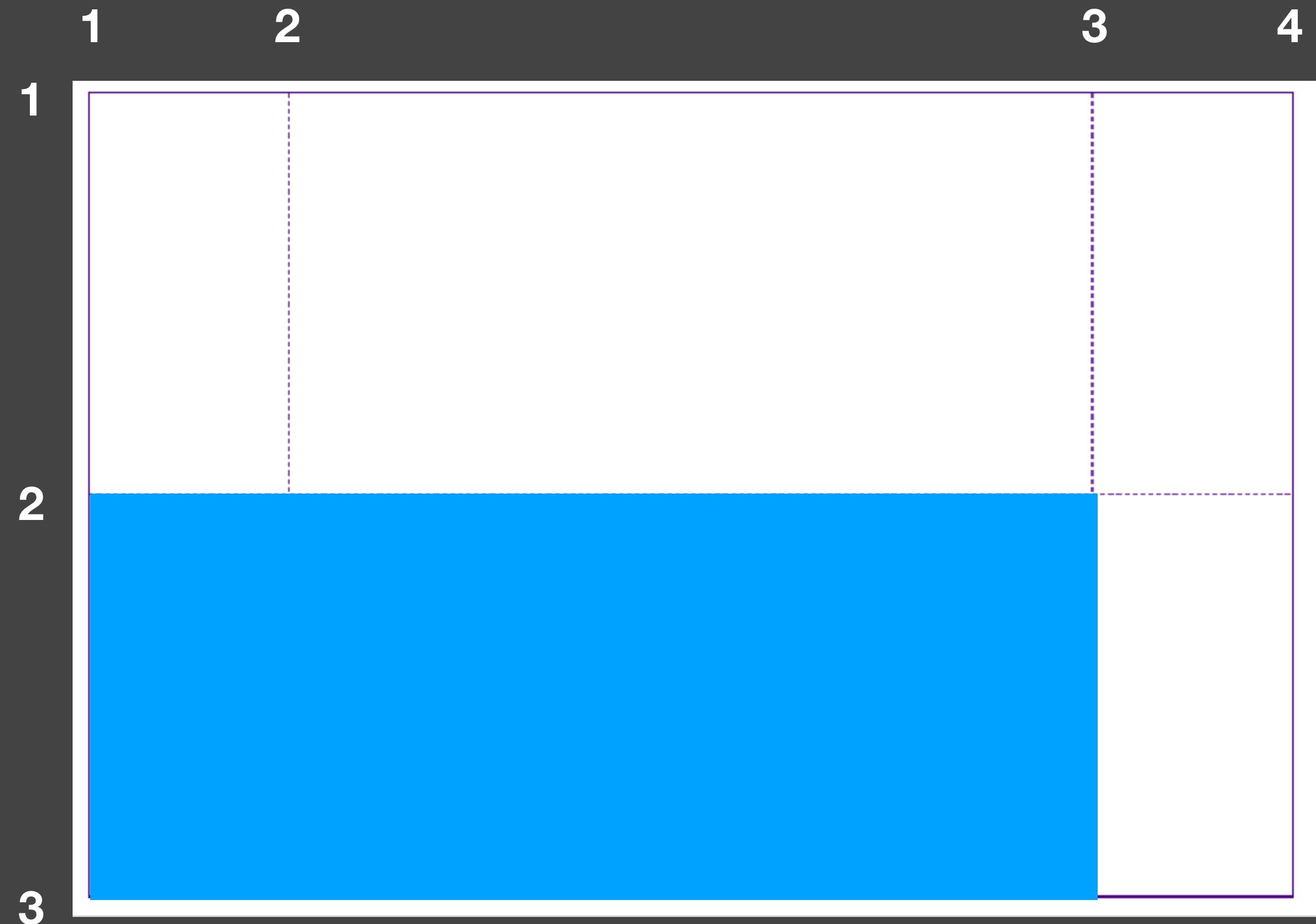
30 maart tot
15 april 1957

Baselitz

1960-83

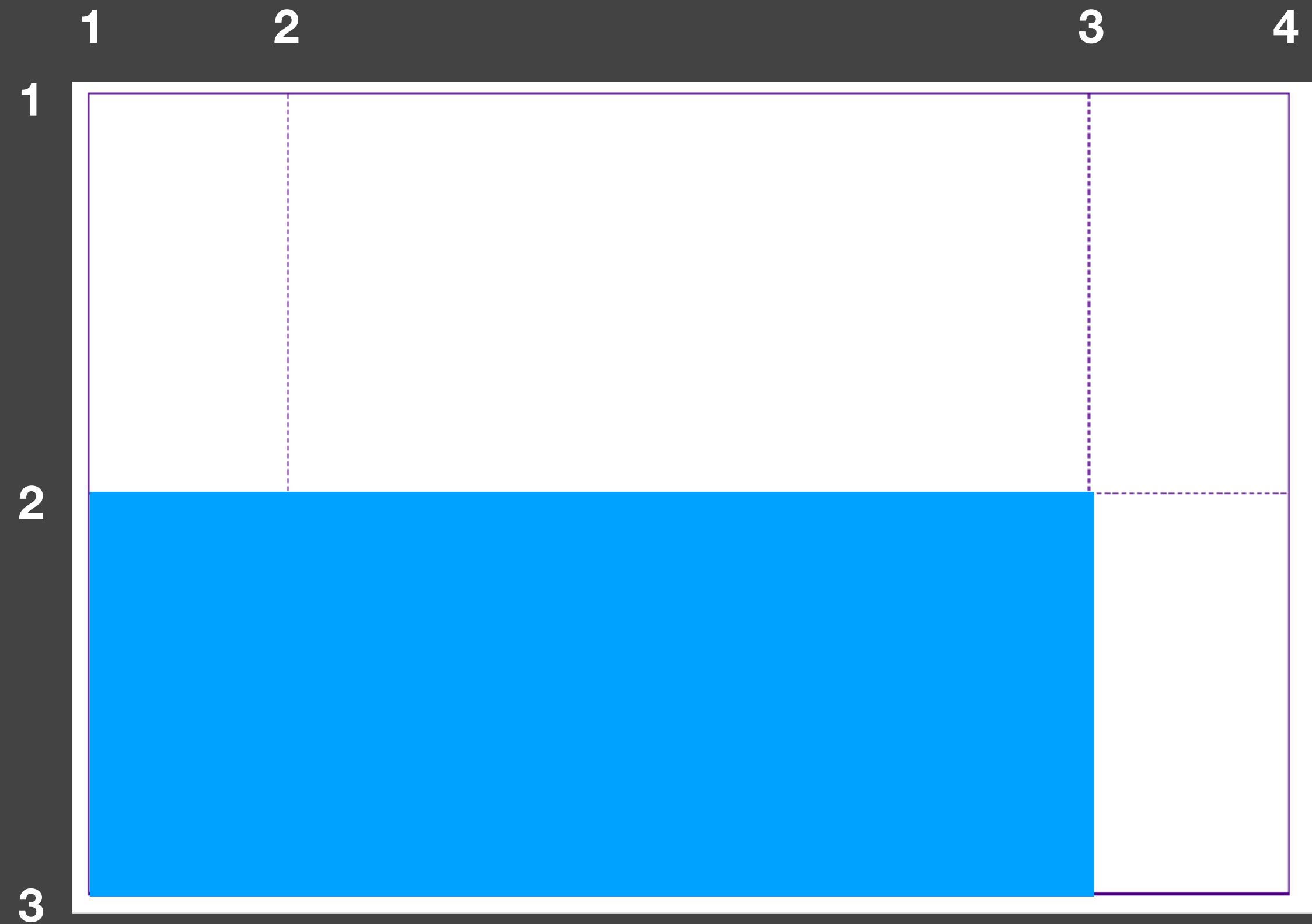
Schilderijen / Paintings

items op het grid plaatsen (2)



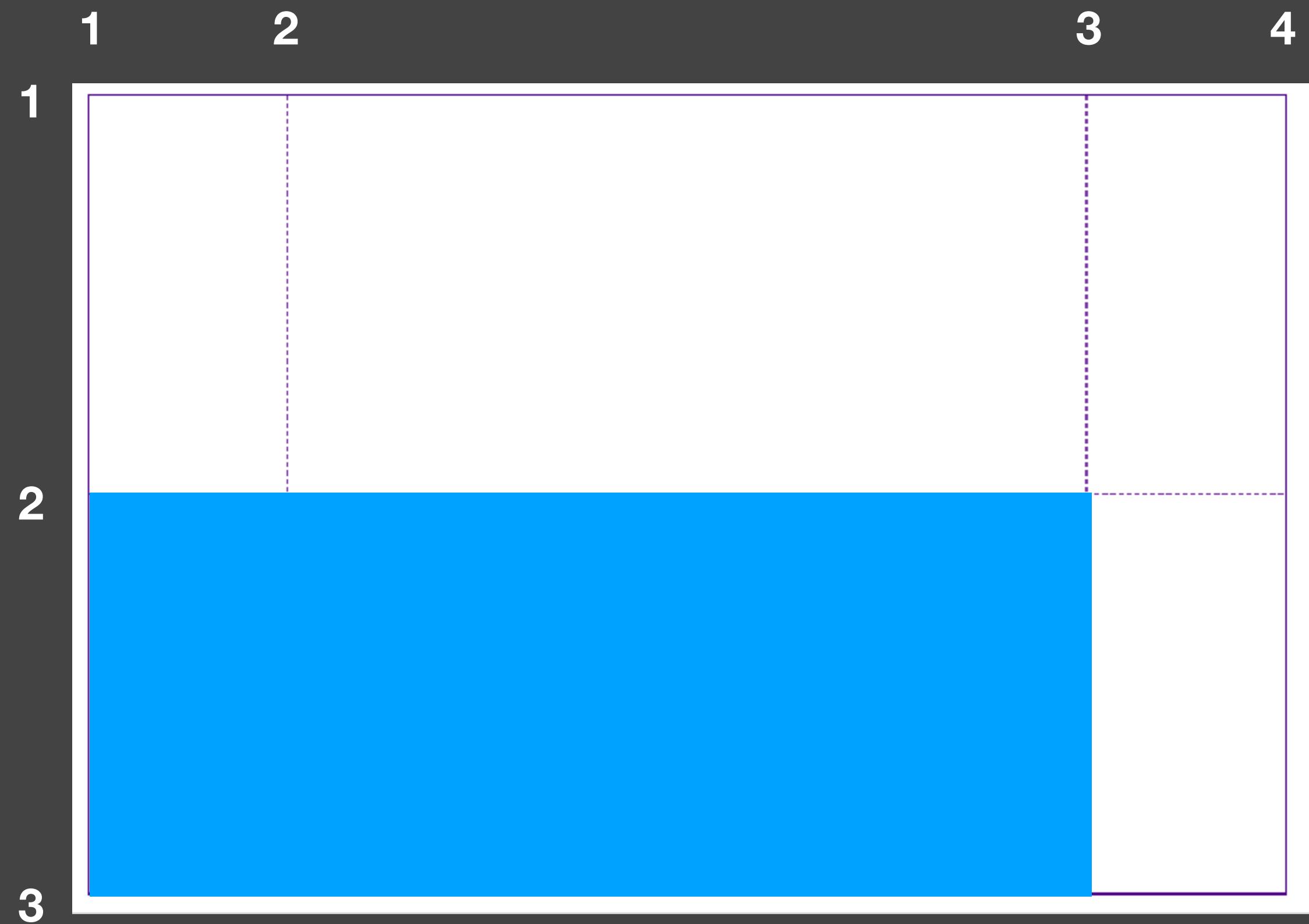
```
#item {  
    grid-column-start: 1;  
    grid-column-end: 3;  
    grid-row-start: 2;  
    grid-row-end: 3;  
}
```

items op het grid plaatsen



```
#item {  
    grid-column: 1 / 3;  
    grid-row: 2 / 3;  
}
```

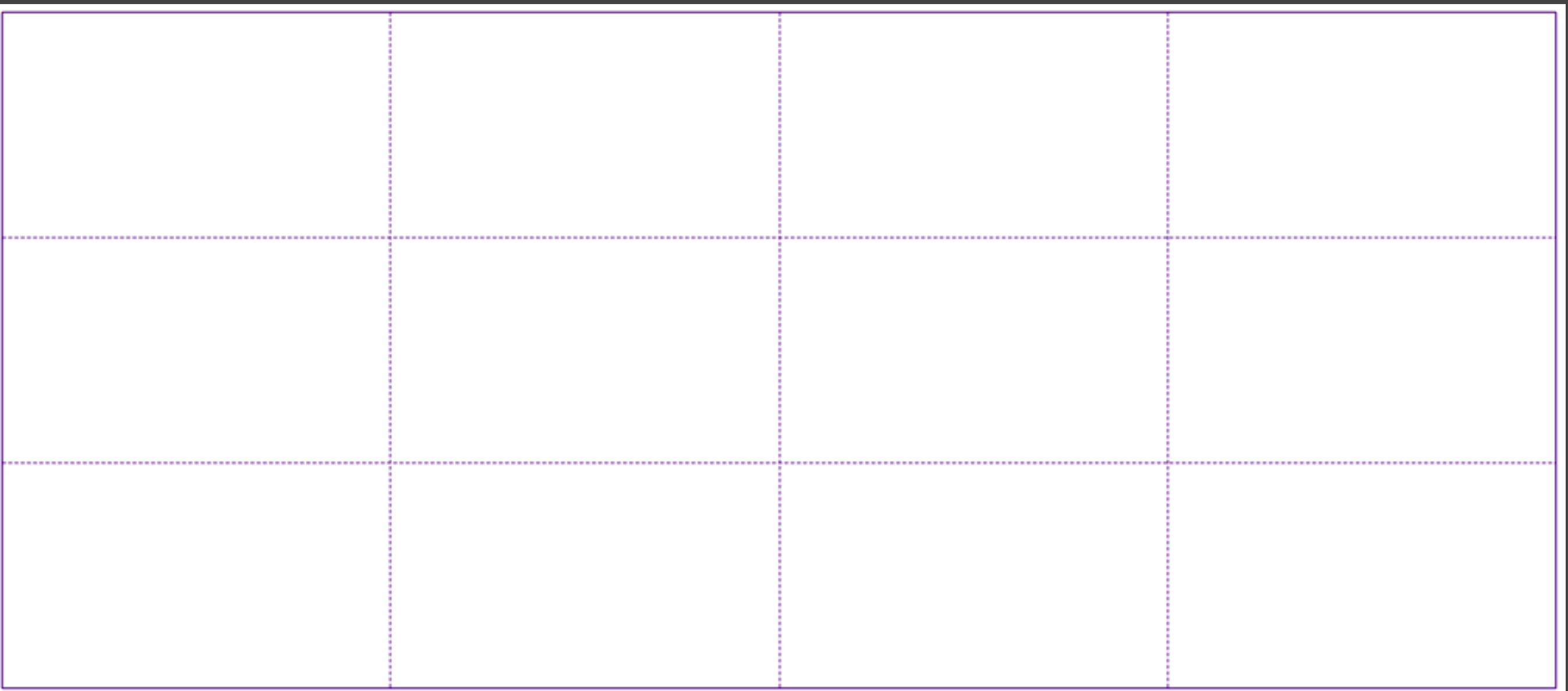
Het span keyword



```
#item {  
  grid-column: 1 / span 3;  
  grid-row: 2 / span 1;  
}
```

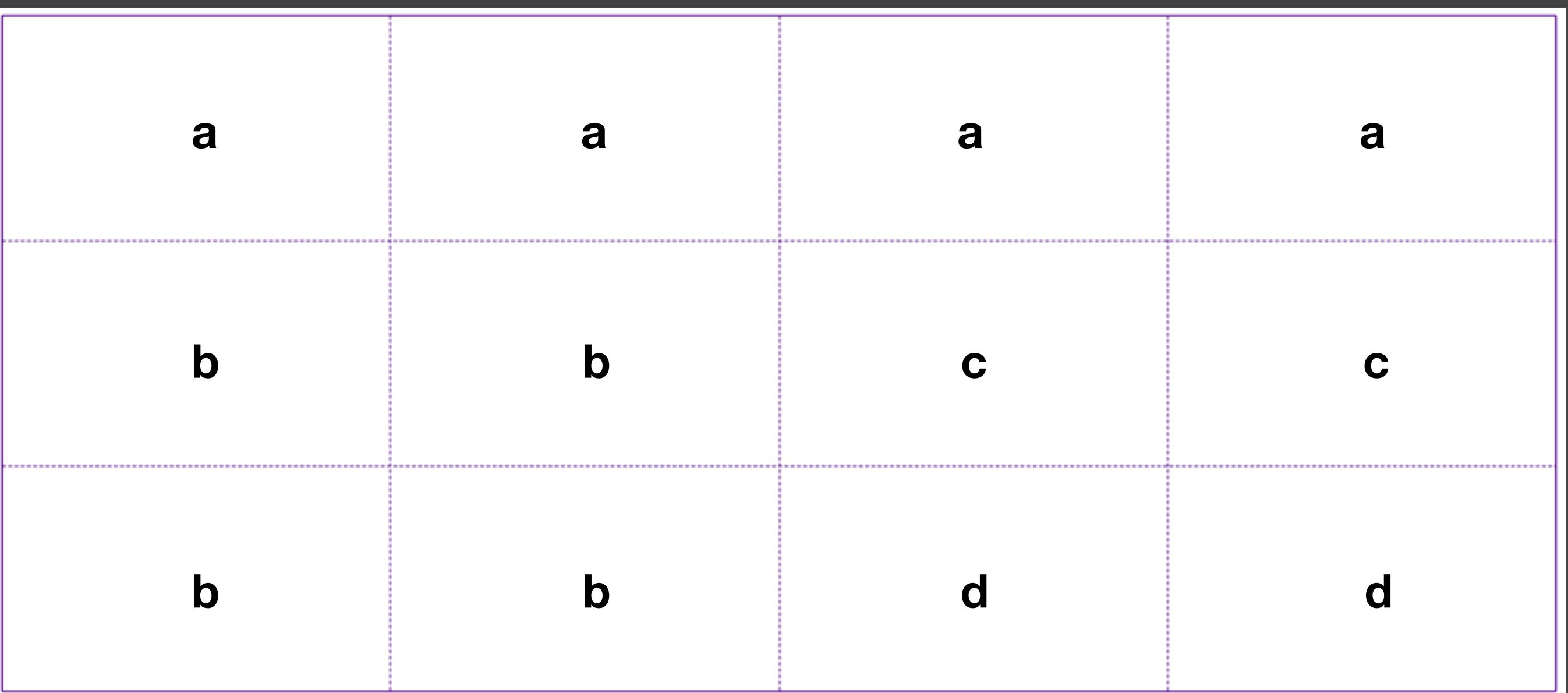
grids met named areas

```
#container {  
  
    display: grid;  
  
    grid-template-areas:  
        "a a a a"  
        "b b c c"  
        "d d d d";  
  
}
```



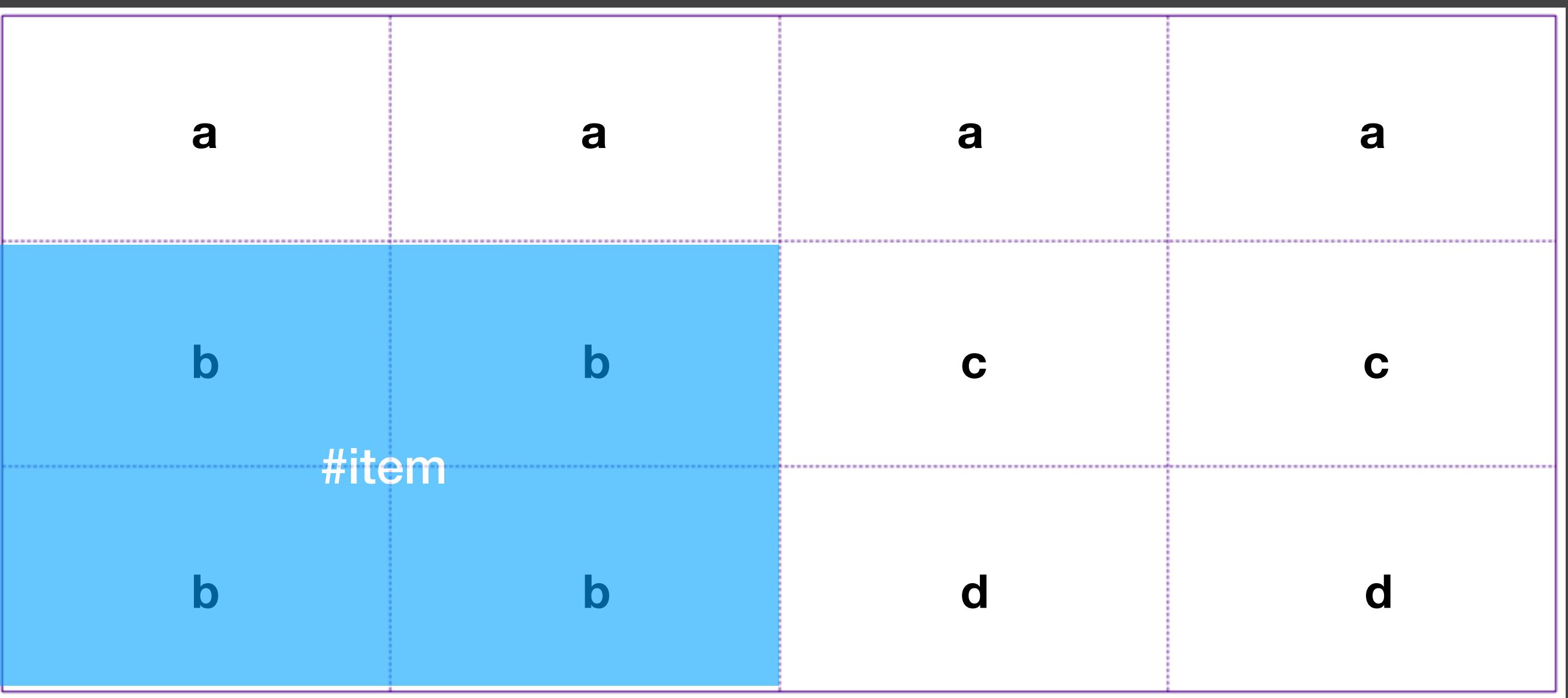
grids met named areas

```
#container {  
  
    display: grid;  
  
    grid-template-areas:  
        "a a a a"  
        "b b c c"  
        "b b d d";  
  
}
```



grids met named areas

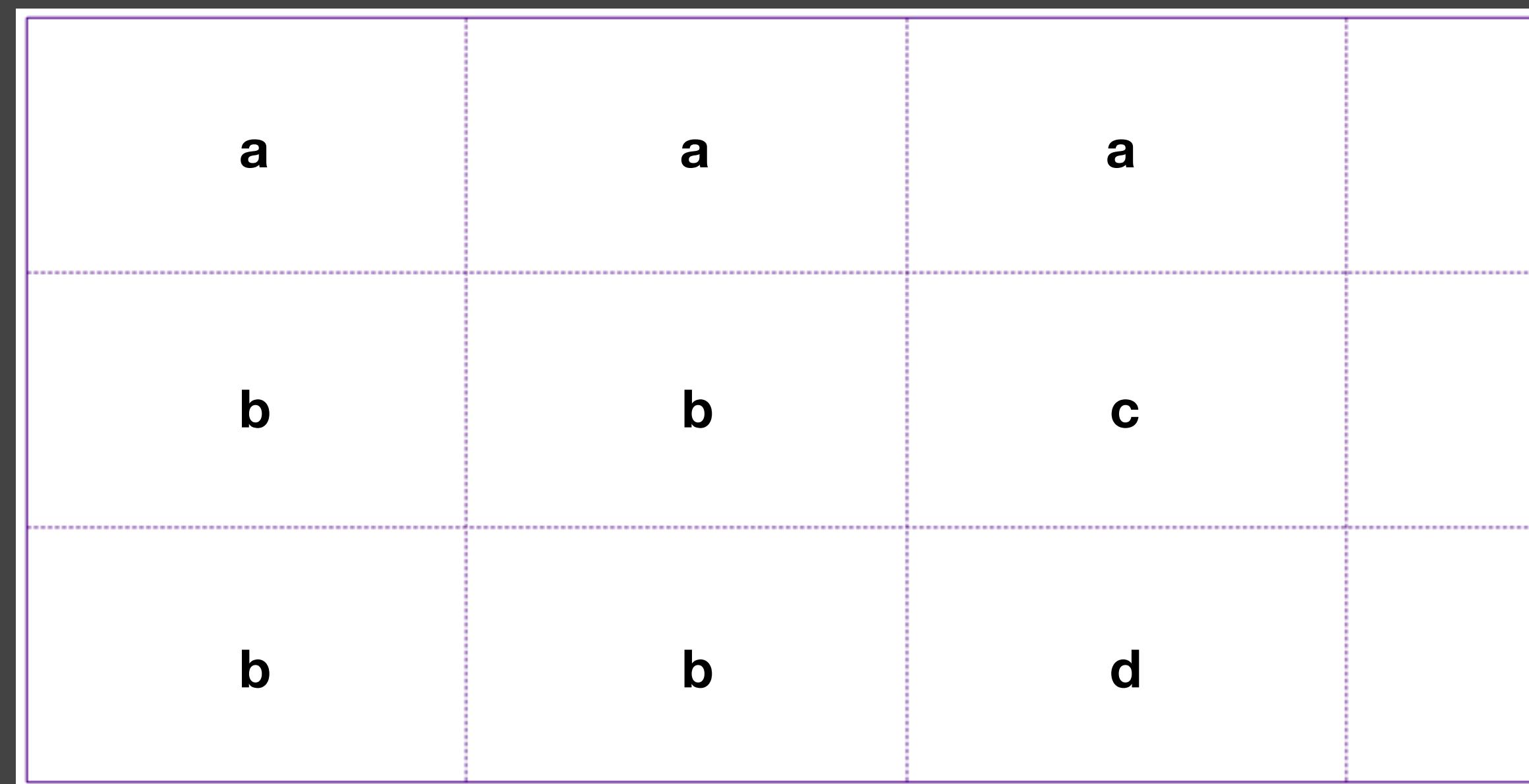
```
#item {  
  grid-area: b;  
}
```



grid lines krijgen
automatisch namen

grids met named areas

```
#item {  
  
  grid-column-start: b-start;  
  
  grid-column-end: b-end;  
}
```



```
#container {  
    grid-template-columns:  
        [sidebar-start] 1fr [sidebar-end] 4fr;  
}
```

```
#item {  
    grid-column-start: sidebar-start;  
    grid-column-end: sidebar-end;  
}
```

```
#container {  
    grid-template-columns:  
        [sidebar-start] 1fr [sidebar-end] 4fr;  
}
```

```
#item {  
    grid-area: sidebar;  
}
```

OEFENING

In The Daily Cascade, geef je lijnen namen en positioneer je items vervolgens in de nieuw ontstane areas

Tijd over?
Draai het om:
benoem areas
en positioneer
op de ontstane
lijnen

herhaling in grid definities
met repeat()

herhaling in grid definities

```
#container {  
    grid-template-columns: repeat([count], [size]);  
}
```

herhaling in grid definities

```
#container {  
    grid-template-columns: repeat(4, 1fr);  
}
```

herhaling met autofocus

```
#container {  
    grid-template-columns: repeat(autofill, 10em);  
}
```

OEFENING

Herhalende grid items

picasso

braque

chagall

leger

miro

ernst

mondriaan

lipchitz

kandinsky

kokoschka

zadkine

appel

bazaine

corneille

delaunay

dubuffet

dagelijks geopend van 10-17 uur zondag van 14-18 uur

**van
abbemuseum
eindhoven**

autoplacement

Autoplacement is wat de browser doet met grid items die niet op het grid zijn geplaatst.

**Als daar niet genoeg rijen/
kolommen voor zijn, maakt de
browser extra rijen ('implicit grid')**

maten van het implicit grid

`grid-auto-rows: [size];`

`grid-auto-columns: [size];`

richting van het implicit grid

`grid-auto-flow: row | column;`

richting van het implicit grid

`grid-auto-flow: row | column;`

lege ruimtes
automatisch opvullen

grid-auto-flow: [row | column] dense;

**Let op met auto-flow: dense.
De locatie van items kan
onlogisch worden voor je
keyboardgebruiker.**

OEFENING

Freestyle! Wees creatief
en maak een grid voor
jouw project, hobby of werk

Grid in productie

“Is er ook een polyfill?”

Simplele fallbacks met de cascade

Feature queries

Feature queries

```
@supports ( display: grid ) {  
  #item {  
    margin: 0;  
    width: auto;  
  }  
}
```

**“Is er ook een Grid
Layout framework?”**

OEFENING

Neem je grid, en voeg zoveel fallbacks toe als je nodig vindt. Gebruik feature queries voor je Grid-only CSS.

Meer bronnen

The New CSS Layout, Rachel Andrew

<https://abookapart.com>



De PhD thesis van Håkon Wium Lie

<http://www.wiumlie.no/2006/phd/css.pdf>

The screenshot shows a Mac OS X PDF viewer window with two pages of a thesis titled "CASCADING STYLE SHEETS" by Håkon Wium Lie.

Title Page (Page 1):

CASCADING STYLE SHEETS
Håkon Wium Lie
Thesis submitted for the degree of Doctor Philosophice
Faculty of Mathematics and Natural Sciences
University of Oslo
Norway
2005

Abstract Page (Page 2):

Håkon Wium Lie © 2005
Published by the Faculty of Mathematics and Natural Sciences
University of Oslo
Sofia 1, 0300 Oslo 3, Norway, as part of the degree
Doctor Philosophiae in the subject
Applied Mathematics and Natural Sciences
with speciality
Computer Science
and a minor subject from the Faculty of Mathematics and Natural Sciences
with speciality
Computer Science
Date: 2005
ISBN 82-316-1000-0

Copyright © 2005 by Håkon Wium Lie. All rights reserved.
This thesis may be reproduced and given to others, provided the
author is acknowledged and the thesis is not resold in whole or in part.
Approved for carriage in the University's library by the author's supervisor
Date: 2005

University Logo:

UNIVERSITAS OSLOENSIS
MDCCCCXI

Page Headers:

css.pdf (page 1 of 306) — Locked ✓
1
2
ABSTRACT

Jen Simmons | Labs

Jen Simmons | Labs [x](#) + [labs.jensimmons.com](#)

Intro to CSS Grid

5 Basic Examples of how CSS Grid Works

12 Variations of Card Layouts

Example of Nesting Flexbox and Grid

AEA Boston 2017 2016 demos

See how these demos work by inspecting them with the [Firefox Grid Inspector Tool](#). Learn how to use [CSS Grid](#).

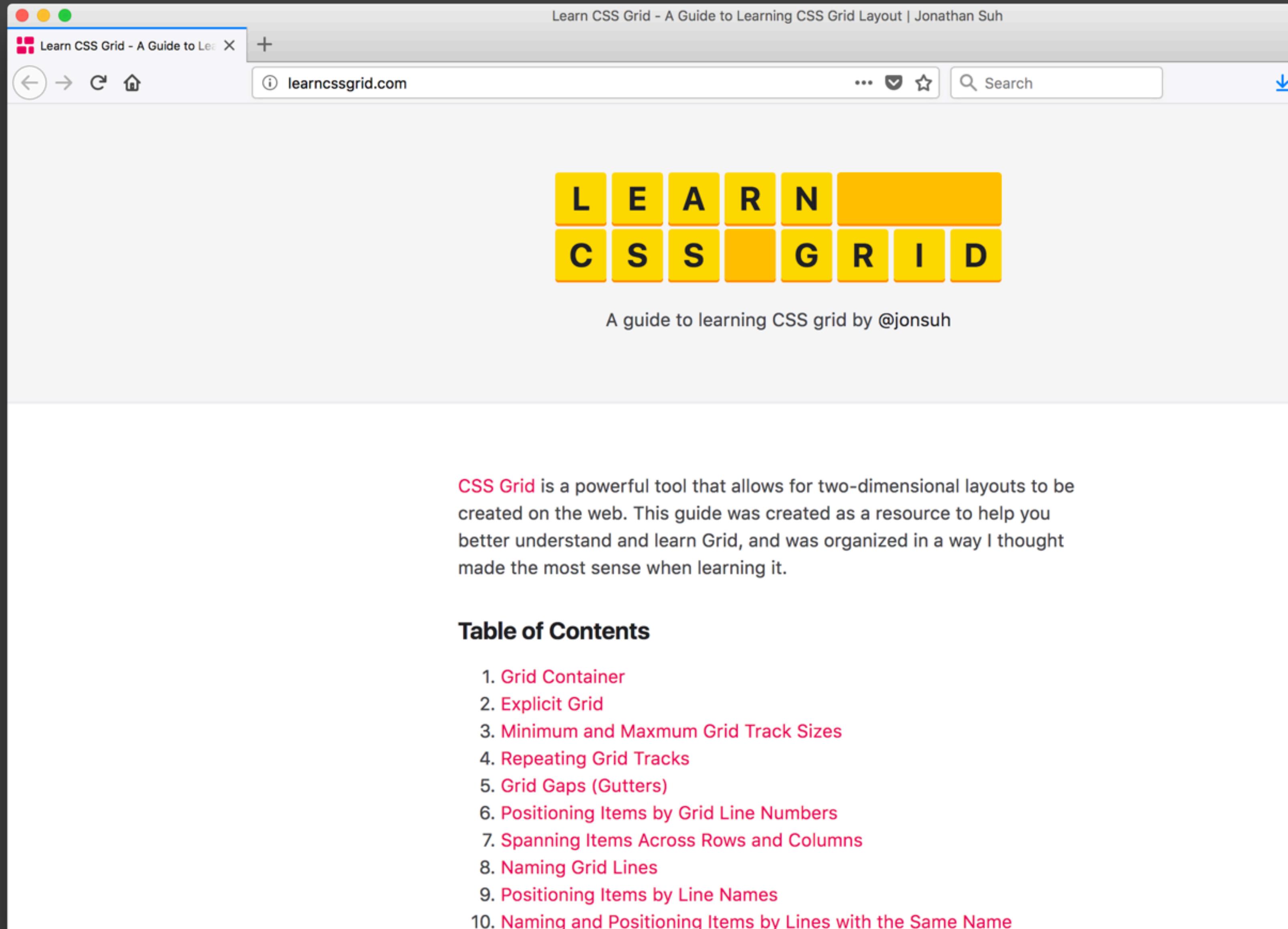
on Codepen

The Experimental Layout Lab of Jen Simmons

<http://labs.jensimmons.com/>

Learn CSS Grid

<http://learncssgrid.com>



The screenshot shows a web browser window with the title bar "Learn CSS Grid - A Guide to Learning CSS Grid Layout | Jonathan Suh". The address bar shows the URL "learncssgrid.com". The main content area features a large graphic of the words "LEARN CSS GRID" composed of yellow squares. Below the graphic is the text "A guide to learning CSS grid by @jonsuh". To the right of the graphic, there is a paragraph about CSS Grid. At the bottom, there is a "Table of Contents" section with a numbered list of 10 items.

CSS Grid is a powerful tool that allows for two-dimensional layouts to be created on the web. This guide was created as a resource to help you better understand and learn Grid, and was organized in a way I thought made the most sense when learning it.

Table of Contents

1. Grid Container
2. Explicit Grid
3. Minimum and Maximum Grid Track Sizes
4. Repeating Grid Tracks
5. Grid Gaps (Gutters)
6. Positioning Items by Grid Line Numbers
7. Spanning Items Across Rows and Columns
8. Naming Grid Lines
9. Positioning Items by Line Names
10. Naming and Positioning Items by Lines with the Same Name

CSS Day 2018, 14th and 15th of June, Amsterdam

CSS Day 2018, 14th and 15th of June, Amsterdam X +

https://cssday.nl/2018

Search

2018 2017 2016 2015 2014 2013



#ux-special + .css-day {
 June 14 & 15, 2018;
}

Compagnietheater, Amsterdam

Subscribe for updates!

On Thursday 14th and Friday 15th of June, 2018, CSS Day is returning with two conference days: a UX Special on Thursday, and a regular CSS Day on Friday.

See the [coverage of the 2017 edition](#) for an idea of what you can expect. (Or [2016](#), for that matter.)

As always the conference will take place in [Het Compagnietheater](#). Ticket sales haven't started yet, but you can [subscribe for updates](#).

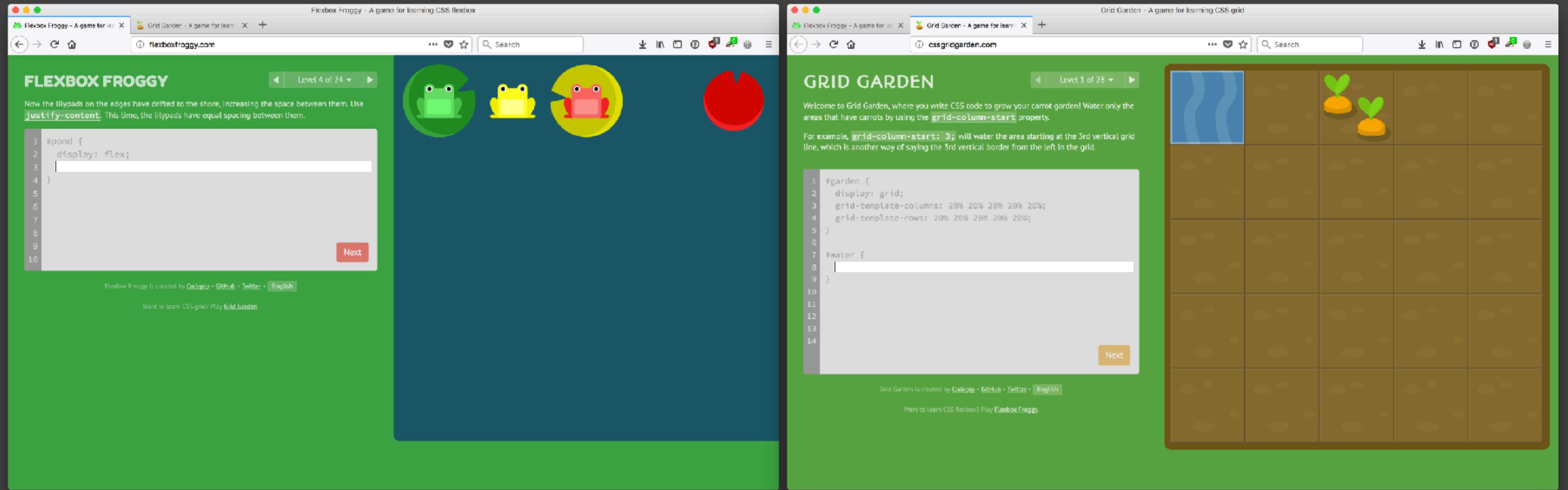
We hope to see you then!

CSS Day

<https://cssday.nl/2018>



€50 korting op
twee dagen met
HDV50



Flexbox Froggy / CSS Grid Garden

<http://flexboxfroggy.com/>

<http://cssgridgarden.com/>



Jen Simmons / @jensimmons

Rachel Andrew / @rachelandrew

Elika J. Etemad / @fantasai

Tab Atkins / @tabatkins

Hui Jing Chen / @hj_chen

Mary Lou / @crnacura

Eric Meyer / @ericmeyer

Keith Grant / @keithjgrant

CSS Working Group / @csswg

WORKSHOP

CSS Layout

Bedankt voor jullie
komst! Zijn er vragen?

Mag ook naar:

@hdv
hidde@hiddedevries.nl