
czechitas
CSS Grid
Luděk Roleček

CSS Grid

Nový CSS modul pro layout

- grid = mřížka
- podobný flexboxu, ale dvourozměrný
- nový, ale už ho lze celkem spolehlivě a bezpečně použít…

czechitas

Podpora v prohlížečích

Podpora v moderních prohlížečích je dobrá, ale je třeba dávat pozor hlavně na IE *(jak jinak.. ach jo)*.

https://caniuse.com/#feat=css-grid

IE11 Grid podporuje, ale pouze starší specifikaci. Je třeba na to myslet, pokud potřebujete IE, což doufejme nikoho z vás nepotká.

czechitas

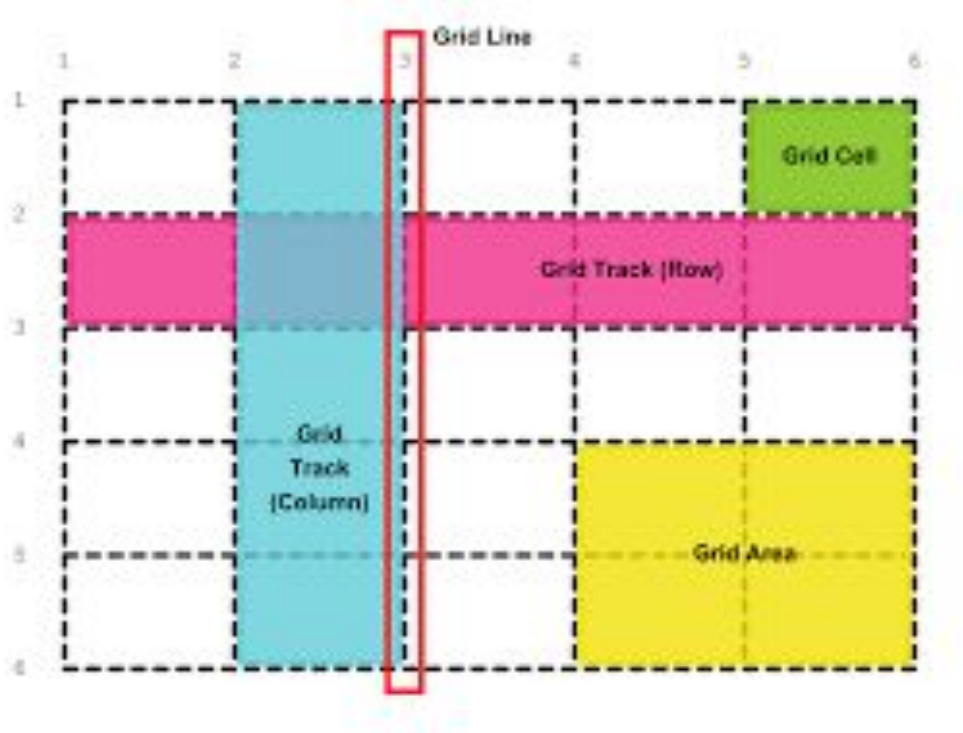
CSS grid je ta nejlepší věc ve vesmíru!

A osobně budu bojovat s každým, kdo tvrdí opak!

czechitas

Grid - jak vypadá?

Mřížka, do které můžeme umisťovat objekty.



Mohu umisťovat na libovolnou pozici mřížky.

Oblasti mohou zabírat více buněk mřížky.

Jako “tabulka”, ale OMG, o tolik lepší!

czechitas

Jak použít

- Aplikuje se stejně, jako flexbox
- Na rodičovský prvek (kontejner) dáme display: grid;
- Všichni přímí potomci se stanou položkami gridu

```
display: grid;
```

```
<div class=”mujgrid”>
   <div> Grid item </div>
   <div> Grid item </div>
   <div> Grid item </div>
</div>
```

czechitas

Grid terminologie

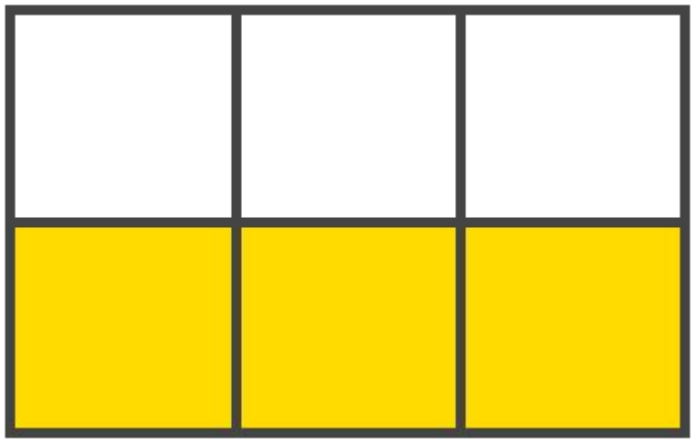
Line / linie - dělící “čára” mezi jednotlivými buňkami.

Objekty *neumisťujeme dovnitř buněk*, ale hranice objektů **pokládáme na linie gridu**.

Čísluje se od 1. Na obrázku **linie 3**. Automaticky jsou přidělena i záporná čísla v opačném směru. Tj. na linii na obrázku mohu odkázat i jako na **-2**.

Grid terminologie

Track / dráha - termín pro řadu nebo sloupec.



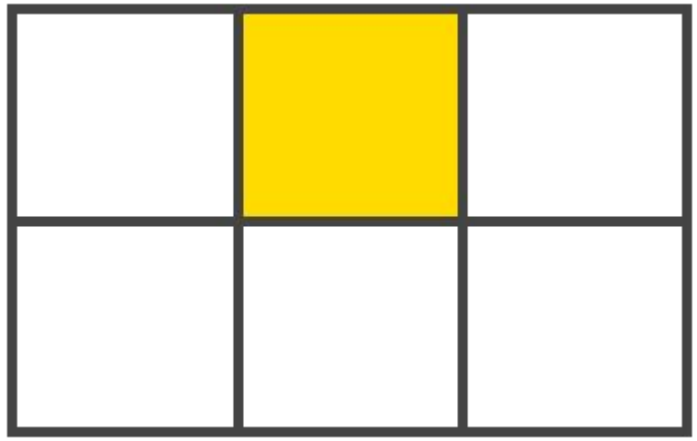
Dráha je vymezena dvěma liniemi.

Dráha tvořená řádkovými liniemi 2 a 3.

czechitas

Grid terminologie

Cell / buňka - políčko uvnitř “tabulky”.



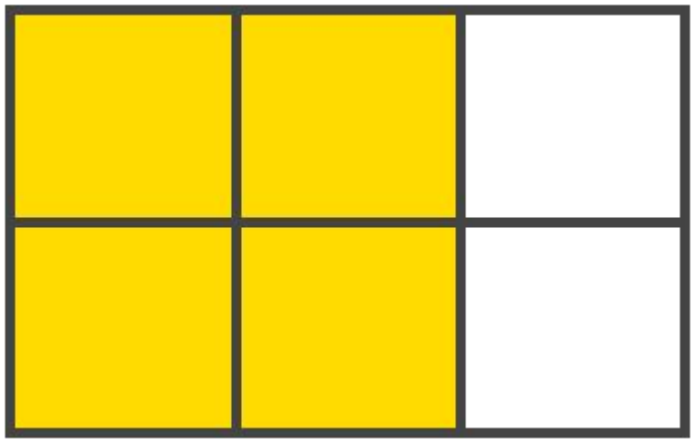
Vymezena dvěma svislými a dvěma vodorovnými liniemi.

Buňka tvořená sloupcovými liniemi 2 a 3 a řádkovými liniemi 1 a 2.

czechitas

Grid terminologie

Area / oblast - větší oblast tvořená sousedícími buňkami.



Vymezena dvěma svislými a dvěma vodorovnými liniemi.

Oblast tvořená sloupcovými liniemi 1 a 3 a řádkovými liniemi 1 a 3.

Rozložení CSS GRIDU

Jak vypadá mřížka?

Jak velké jsou sloupce a řádky? A kolik jich je?

Vlastnosti CSS Gridu

Stejně jako u flexboxu, rozdělujeme:

- vlastnosti, které se aplikují na **kontejner**
- jiné vlastnosti, které se aplikují na **položky** gridu

Nejdříve to nejdůležitější

Aplikujeme na kontejner.

```
display: grid;
```

Nadefinujeme dráhy

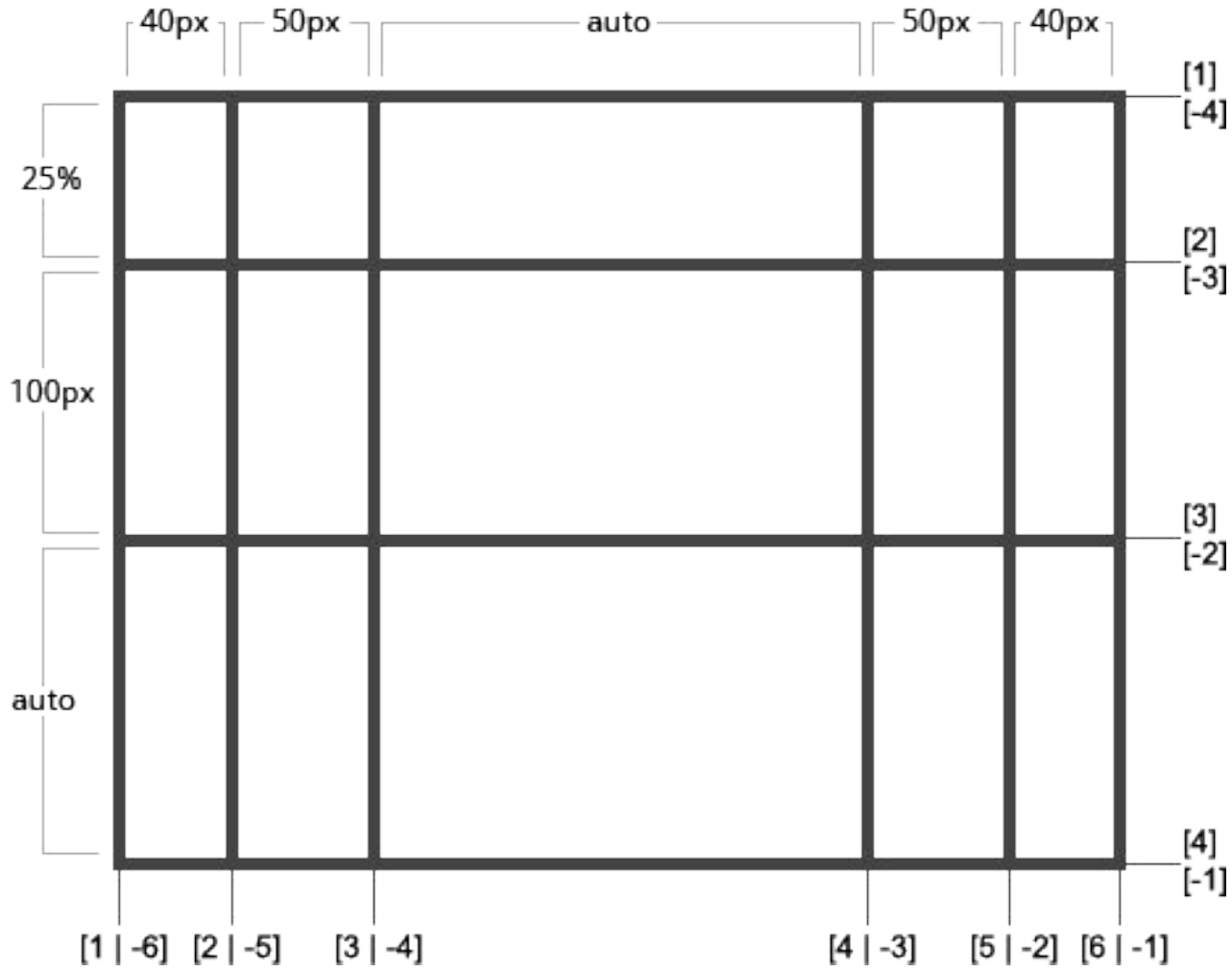
Gridu musíme nadefinovat mřížku - tj. sloupce a řádky.

Aplikujeme na kontejner.

```
grid-template-columns: 100px auto 30% ...;

grid-template-rows: 200px auto 2fr ...;
```

Rozložení drah a linií v rámci gridu



czechitas

Opakování

Pokud máme více stejných sloupců nebo řádků, můžeme použít funkci repeat pro opakování:

```
grid-template-columns: repeat(3, 250px);
```

Vytvoří **3** sloupce o šířce **250px**.

Lze samozřejmě kombinovat společně s dalšími hodnotami pro sloupce nebo řádky:

```
grid-template-rows: 100px repeat(3, 250px) 400px;
```

Vytvoří řádky: 100px, 250px, 250px, 250px, 400px

Umisťujeme položky dovnitř CSS GRIDU

Umístění položky gridu

Pro položky uvnitř gridu nastavíme, kde leží a jak jsou velké.

Aplikujeme na položku gridu.

```
grid-column-start: 2;
grid-column-end: 4;
```

Položka sahá od druhé sloupcové linie zleva až do čtvrté.

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

Položka sahá od první řádkové linie shora až do třetí linie od konce.

Umístění položky gridu

Místo samostatných vlastností pro počáteční a koncový sloupec nebo řádek (z předchozí stránky), můžeme použít **zkrácený zápis**:

```
grid-column: 2 / 4;
```

Položka sahá od druhé sloupcové linie až do čtvrté.

```
grid-row: 1 / -3;
```

Položka sahá od první řádkové linie shora až do třetí linie od konce.

Nastavíme umístění položky gridu

Mohu použít i zápis pomocí span, kterým mohu nastavit počáteční linii a pak za slovem span určit, kolik sloupců (nebo řádků) položka v mřížce zabírá:

```
grid-column: 2 / span 3;
```

Položka začíná na druhé sloupcové linii a zabírá v mřížce 3 sloupce.

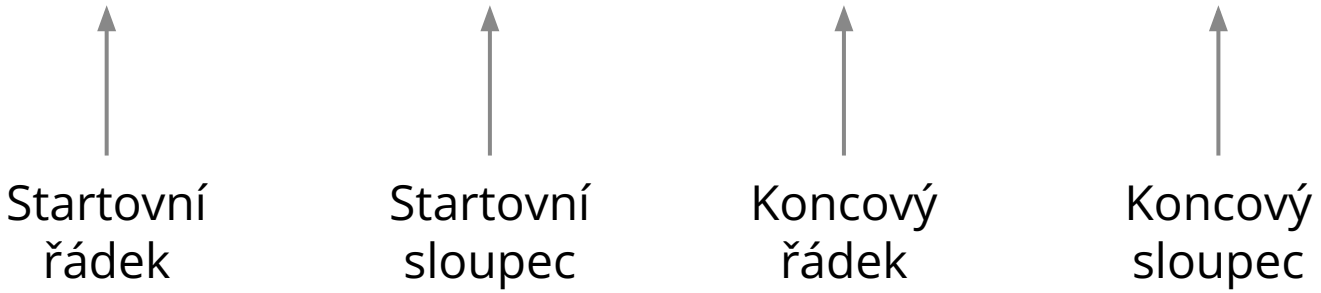
```
grid-row: 1 / span 2;
```

Položka začíná na první řádkové linii a zabírá v mřížce 2 řádky.

Nastavíme umístění položky gridu

Případně mohu použít ještě zkrácenější zápis, kdy vymezím položce celou oblast umístění jedním řádkem:

```
grid-area: row1 / col1 / row2 / col2;
```



Mezery mezi buňkami gridu

Mezera mezi buňkami mřížky

Když chci, aby buňky nebyly nacpané na sobě, ale byla mezi nimi mezera:

```
gap: 30px;
```

Mezera 30px mezi řádky a sloupci.

```
gap: 20px 40px;
```

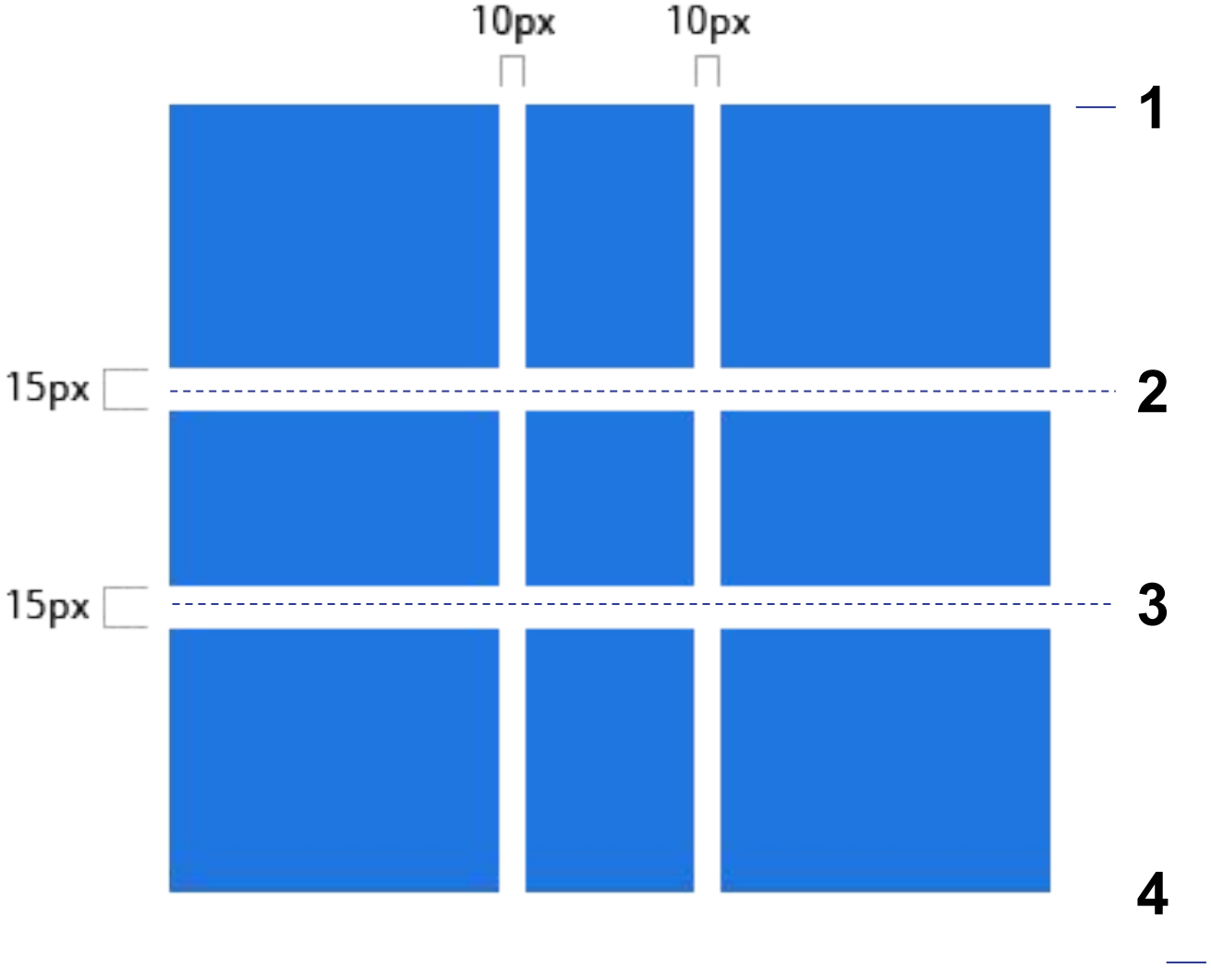
Mezera 20px mezi řádky a 40 px mezi sloupci.

Nebo lze nastavit každou zvlášť pomocí:

```
row-gap: 30px;
```

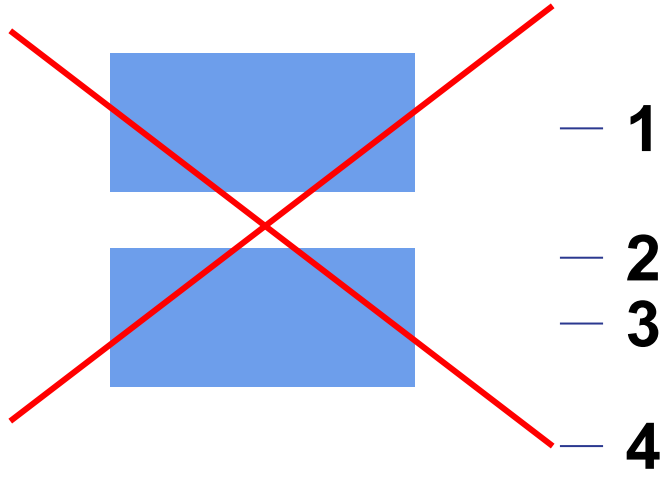
```
column-gap: 20px;
```

Mezera mezi buňkami mřížky



Mezery mezi řádky a sloupci nemění čísla linií.

Zjednodušeně: linie vede středem mezery. Mezera nevytváří dvě oddělené linie.



czechitas

Mezera mezi buňkami mřížky

V případě, že prohlížeč ještě nepodporuje nově zaváděnou vlastnost gap (a row-gap, column-gap), můžeme použít starší specifikaci.

```
grid-gap: 30px;
```

Zkratka: všude stejná mezera 30px.

```
grid-gap: 30px 20px;
```

Zkratka: 30px mezi řádky, 20px mezi sloupci.

Zarovnávání položek gridu

Zarovnávání položek gridu

Podobně jako u flexboxu, výchozí hodnota pro zarovnání položek je stretch, tj. položka se roztáhne přes celou buňku (v horizontálním i vertikálním směru).

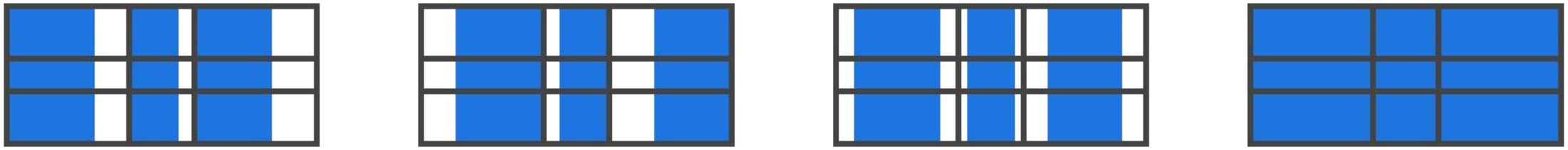
Samozřejmě ale jde nastavit zarovnání položek vlevo, vpravo nebo na střed vzhledem k buňkám gridu, do kterých je položka umístěna.

Horizontální zarovnání

Aplikuje se na kontejner.

Možné hodnoty: **start**, **end**, **center**, **stretch** (výchozí)

```
justify-items: start;
```

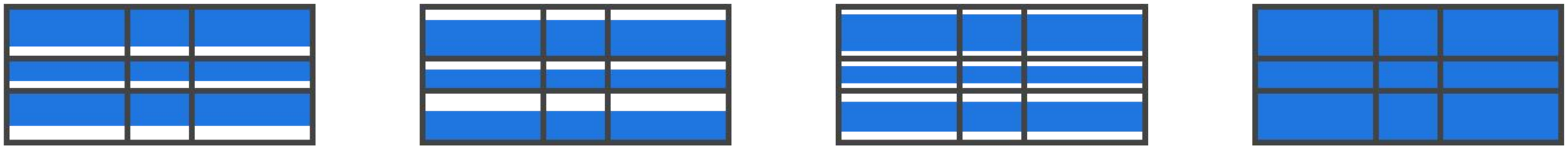


Vertikální zarovnání

Aplikuje se na kontejner.

Možné hodnoty: **start**, **end**, **center**, **stretch** (výchozí)

```
align-items: center;
```



Zarovnání jednotlivé položky

Aplikuje se na položku gridu.

Možné hodnoty: **start**, **end**, **center**, **stretch** (výchozí)

```
justify-self: end;
align-self: center;
```

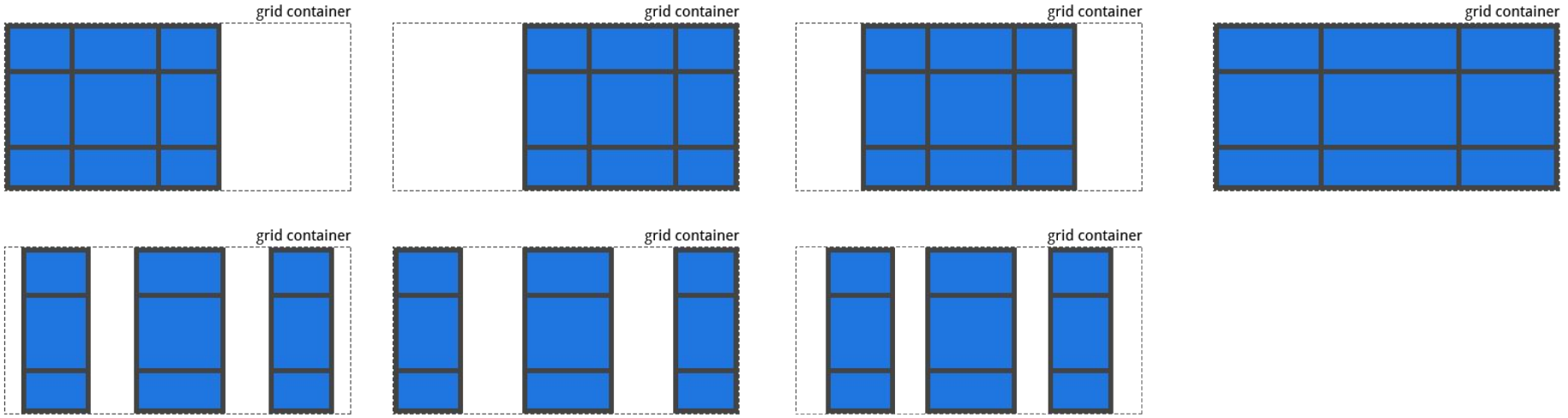
Když chci zarovnat jednu položku jinak, než jak je nastaveno na kontejneru.

Zarovnání sloupců uvnitř gridu

Aplikuje se na kontejner. Možné hodnoty:

start, end, center, stretch, space-around, space-between, spave-evenly

```
justify-content: center;
```



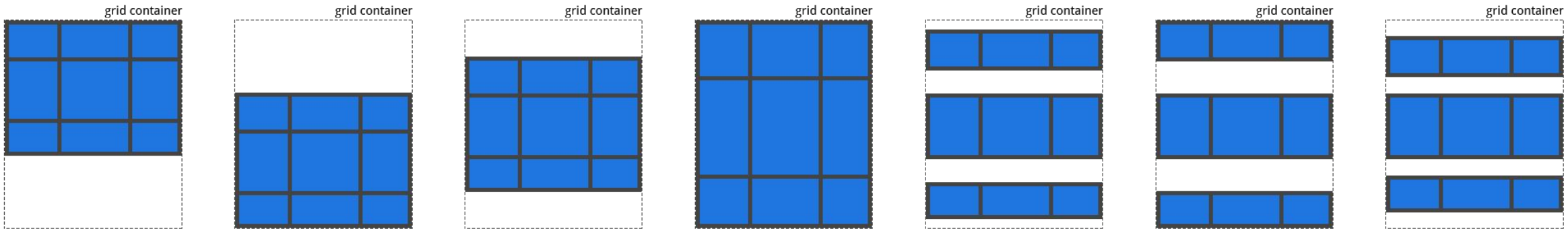
czechitas

Zarovnání řádků uvnitř gridu

Aplikuje se na kontejner. Možné hodnoty:

start, end, center, stretch, space-around, space-between, space-evenly

```
align-content: center;
```



czechitas

Explicitní vs. implicitní sloupce a řádky gridu

Když si grid přidává řádky a sloupce sám

Explicitní vs. implicitní sloupce a řádky

Sloupce a řady nadefinované pomocí grid-template-columns a grid-template-rows označujeme jako explicitní.

V rámci gridu ale můžeme položky umisťovat i mimo explicitně nadefinovanou mřížku. Prohlížeč pak vytvoří v gridu tzv. implicitní řádky a sloupce, aby mohl položku umístit.

Explicitní vs. implicitní sloupce a řádky

Někdy také dopředu nevíme, kolik sloupců a nebo řádků bude naše mřížka mít.

Implicitním řádkům a sloupcům můžeme nastavit velikost:

```
grid-auto-columns: 1fr;
grid-auto-rows: 150px;
```

minmax

Minmax

Ne vždy chceme šířku/výšku sloupce nastavovat na pevnou hodnotu.

```
grid-template-columns:
    minmax(200px, 500px) 1fr;
```

První sloupec bude široký minimálně 200px a maximálně 500px, druhý pak bude zabírat zbytek šířky gridu.

Minmax

Většinou se používá ve spojení s **fr** jednotkami.

```
grid-template-columns:
  repeat(5, minmax(200px, 1fr));
```

Vytvoří 5 sloupců, kde každý zabírá ⅕ šířky, ale má vždy minimálně alespoň 200px.

AUTO-FILL / AUTO-FIT

Automaticky generované sloupce

Automaticky generované sloupce

Mřížku můžeme přizpůsobit a automaticky podle velikosti gridu vytvořit tolik sloupců, kolik se vejde.

```
grid-template-columns:
    repeat(auto-fill, 250px);
```

```
grid-template-columns:
    repeat(auto-fit, 250px);
```

Automaticky generované sloupce

auto-fill a auto-fit jsou fantastické ve spojení s minmax.

```
grid-template-columns:
    repeat(auto-fill, minmax(250px, 1fr));
```

```
grid-template-columns:
    repeat(auto-fit, minmax(300px, 1fr));
```

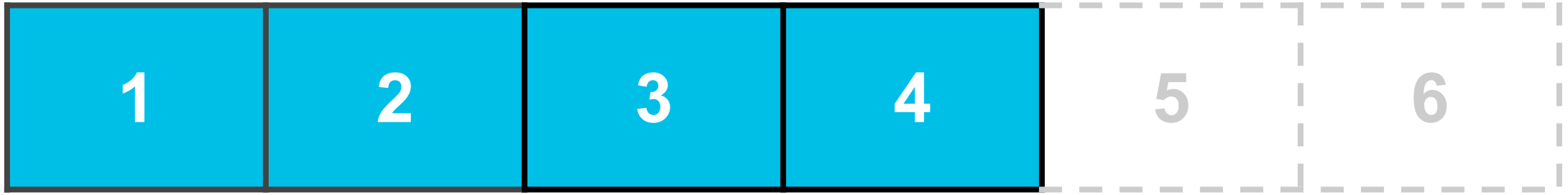
Rozdíly mezi auto-fill a auto-fit

Auto-fill se snaží do mřížky vměstnat co nejvíce implicitních sloupců může. Vytvořené sloupce mohou zůstat prázdné, ale budou zabírat vyhrazené místo.

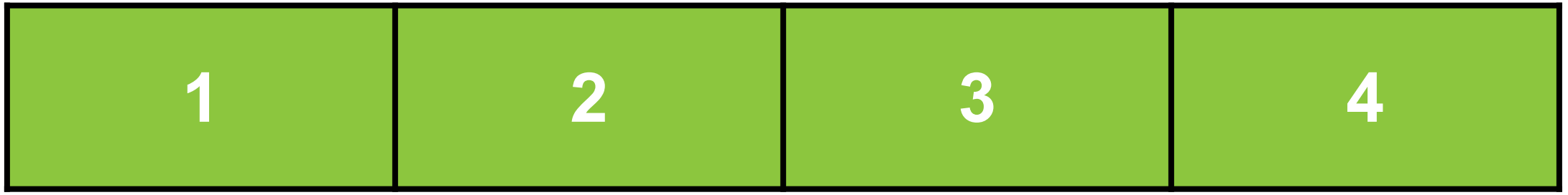
Auto-fit vezme aktuálně dostupné sloupce a natáhne je tak, aby vyplnili celou mřížku. Auto-fit přidá sloupce stejně jako auto-fill, ale prázdné sloupce smrskne na nulovou šířku.

Rozdíly mezi auto-fill a auto-fit

auto-fill



auto-fit

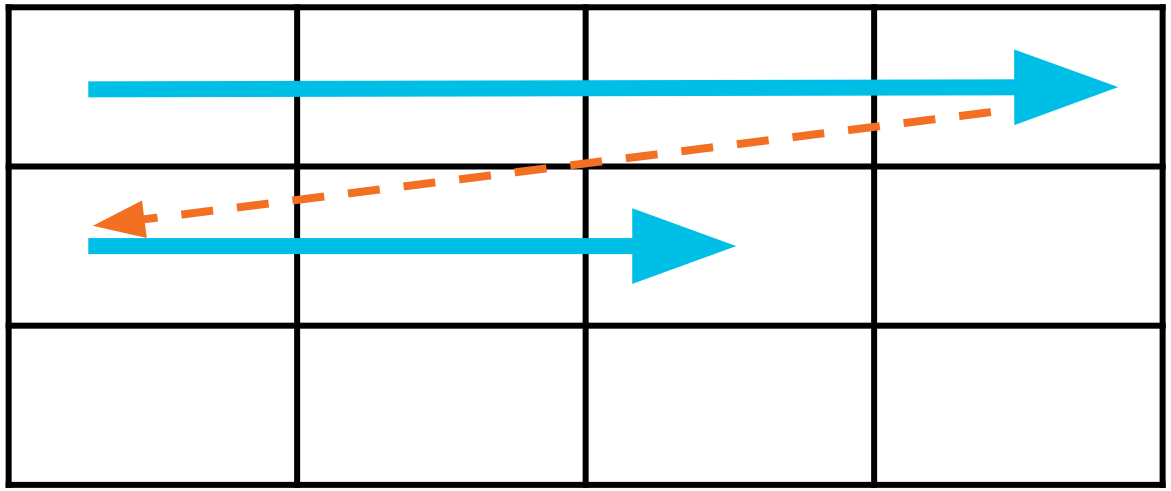


Automatické umisťování položek gridu

Automatické umisťování položek gridu

Nastavuje, jakým směrem grid vyplňuje řádky nebo sloupce, když umisťuje položky bez specifikovaného místa v gridu.

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

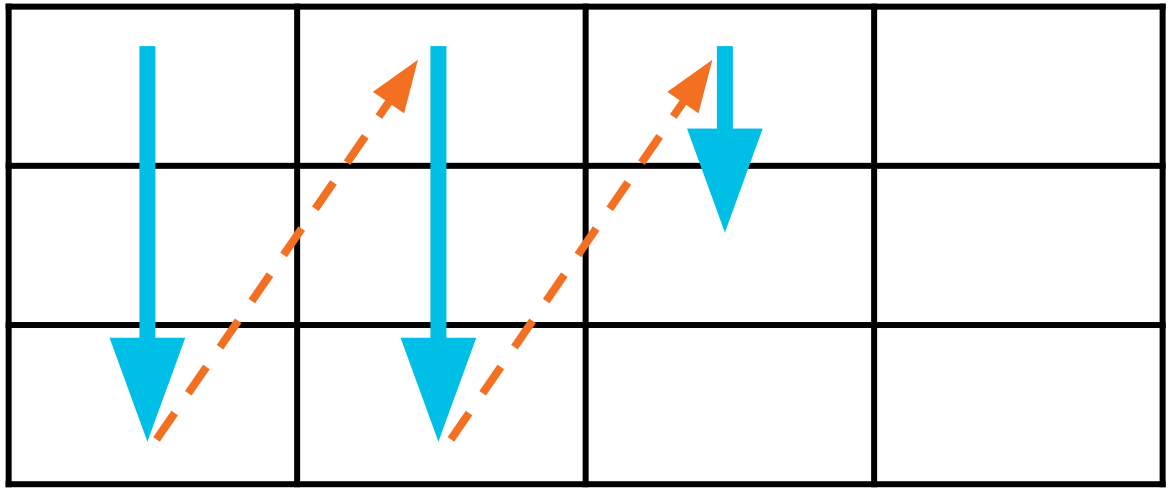


Vyplňuje postupně řádky a přidává implicitní řádky podle potřeby.

czechitas

Automatické umisťování položek gridu

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



Vyplňuje postupně sloupce a přidává implicitní sloupce podle potřeby.

Automatické umisťování položek gridu

```
grid-auto-flow: dense;
```

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

Umisťuje položky i do vzniklých mezer v gridu tak, aby byla mřížka pokud možno co nejvíce vyplněná.

I tak mohou vzniknout mezery.

Může výrazně ovlivnit pořadí prvků v gridu. Vhodné použít pro “masonry” typ layoutu, kdy chceme v mřízce různě velké objekty a nezáleží nám příliš na pořadí.

Pojmenování linií

Pojmenování linií v gridu

Abychom si nemuseli stále pamatovat čísla linií. Mezi šířky sloupců napíšeme v hranatých závorkách název linie:

```
grid-template-columns:
  [zacatek] 100px [stred] 1fr [konec];
```

Jedna linie může mít více jmen:

```
grid-template-columns: [menu-start] 100px [menu-end obsah-start] 1fr [obsah-end];
```

Umisťování na pojmenované linie

Položky umisťujeme stejně, jen místo čísla linie uvedeme její jméno (bez hranatých závorek).

```
grid-column: menu-start / menu-end;
```

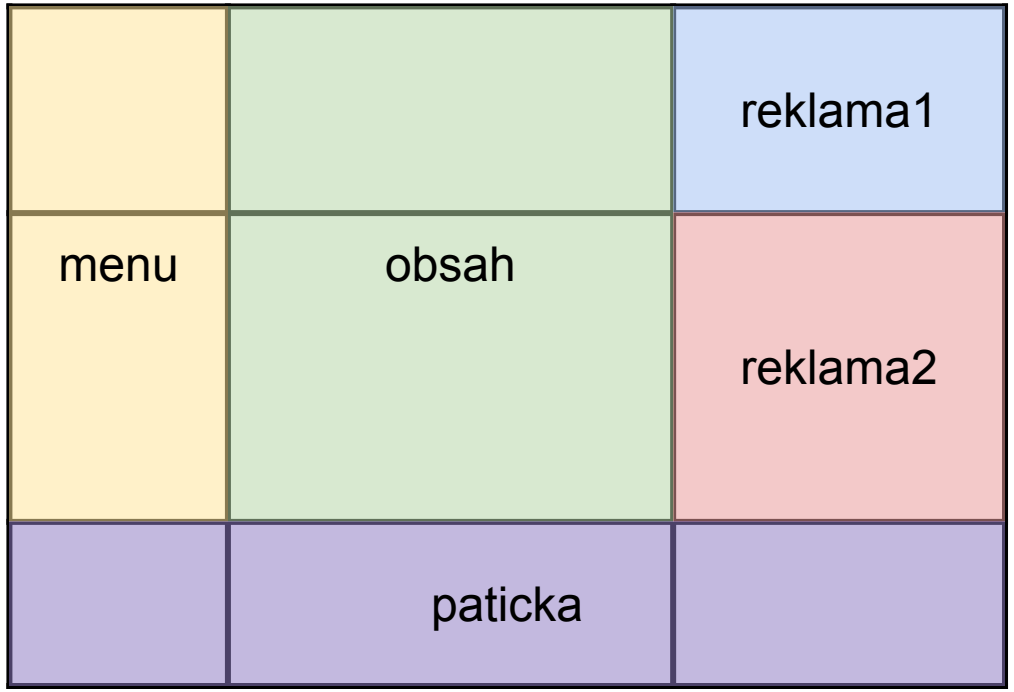
```
grid-row: paticka-start / paticka-end;
```

Pojmenované oblasti gridu

Pojmenování oblastí gridu

Grid má nadefinové řádky a sloupce. Souvislé oblasti můžeme pojmenovat. *Nastavujeme na kontejner.*

```
grid-template-areas:
   “menu obsah reklama1”
   “menu obsah reklama2”
   “paticka paticka paticka”;
```



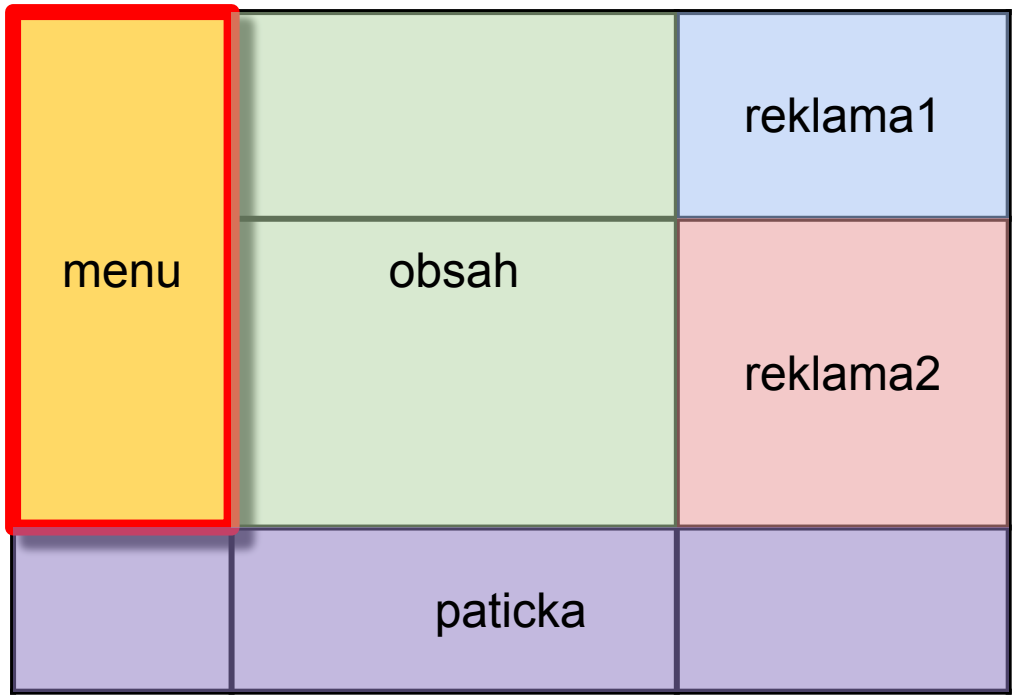
czechitas

Umisťění do pojmenované oblasti

Nastavujeme pro položku gridu.

Jen uvedeme jméno oblasti.

```
grid-area: menu;
```



Pořadí položek v gridu

Pořadí položek v gridu

Stejně jako u flexboxu. Pokud neumisťujeme položky na konkrétní pozici v mřížce, ale necháváme na gridu, kam je umístí, můžeme pomocí order měnit jejich vizuální pořadí nezávisle na pořadí ve zdrojovém kódu.

Nastavujeme pro položku gridu.

```
order: 2;
```