# Konfigurisanje CSS Grida

U prethodnoj lekciji bilo je reči o osnovama CSS Grid sistema. Definisani su koncepti od kojih je Grid sačinjen i obavljeno je kreiranje prvog Grida unutar jednog HTML dokumenta. Lekcija pred vama donosi neke od najznačajnijih pristupa koje je moguće koristiti za prilagođavanje osobina Grida sopstvenim potrebama.

## Implicitni redovi i kolone i automatsko raspoređivanje

CSS Grid sačinjen je iz međusobno ispresecanih horizontalnih i vertikalnih linija. Horizontalne linije grade redove, a vertikalne kolone. Tako je jedan red unutar CSS Grida, zapravo, prostor između dve horizontalne linije. Po istom principu, Grid kolona je prostor između dve vertikalne linije.

Prvi primer kreiranja CSS Grida ilustrovao je jednu važnu osobinu. Iako mi samostalno nismo definisali nikakvo ponašanje, elementi se unutar Grida podrazumevano smeštaju unutar jedne kolone. Podsetimo se takvog primera. HTML struktura je sledeća:

```
<div id="grid1">
<div>One</div>
<div>Two</div>
<div>Three</div>
<div>Four</div>
<div>Five</div>
<div>Six</div>
</div>
```

Stilizacija izgleda ovako:

```
#grid1 {
      display: grid;
}
```

Na ovaj način se dobija Grid ilustrovan slikom 21.1.


*Slika 21.1. Podrazumevana struktura CSS Grida sa šest elemenata*

Copyright © Link group

Sa slike 21.1. može se videti da je CSS Grid elemente smestio unutar jedne kolone. Pri tome je za svaki od elemenata automatski izvršeno kreiranje po jednog novog reda. Drugim rečima, iako samostalno ništa nismo definisali, obavljeno je automatsko kreiranje kolona i redova. Takve kolone i redovi se drugačije nazivaju **implicitni**. Postupak kojim se obavlja kreiranje implicitnih redova i kolona funkcioniše na osnovu algoritma za **automatsko raspoređivanje elemenata**.

Na osobine automatskog raspoređivanja elemenata može se uticati korišćenjem CSS svojstva **grid-auto-flow**. Podrazumevana vrednost ovog svojstva je `row` i upravo zbog toga se novi elementi inicijalno smeštaju u novokreirane redove. Ukoliko se vrednost svojstva `grid-auto-flow` postavi na `column`, dobija se efekat kao na slici 21.2.


*Slika 21.2. Grid sa grid-auto-flow: column;*

Ovoga puta se unutar CSS Grida za svaki element obavlja kreiranje nove kolone. Pošto naš Grid ima 6 elemenata, obavlja se kreiranje šest novih kolona.

Na osobine implicitno kreiranih redova i kolona moguće je uticati korišćenjem dva CSS svojstva:

- `grid-auto-columns`
- `grid-auto-rows`

Korišćenjem dva upravo prikazana svojstva moguće je definisati visinu i širinu implicitno kreiranih redova i kolona:

```
#grid1 {
    display: grid;
    grid-auto-flow: column;
    grid-auto-columns: 180px;
    grid-auto-rows: 200px;
}
```

Efekat svojstava `grid-auto-columns` i `grid-auto-rows` ilustrovan je slikom 21.3.


*Slika 21.3. Grid sa definisanim osobinama implicitno kreiranih kolona i redova*

Copyright © Link group

Unutar radnog okruženja prikazan je primer sa korišćenjem *grid-auto-flow*, *grid-auto-columns* i *grid-auto-rows* svojstvima. Eksperimentisati sa različitim vrednostima ovih svojstava.

**Radno okruženje**

**HTML fajl:**

```
<div id="grid1">
<div>One</div>
<div>Two</div>
<div>Three</div>
<div>Four</div>
<div>Five</div>
<div>Six</div>
</div>
```

**CSS fajl:**

```
#grid1 {
    display: grid;
    grid-auto-flow: column;
    grid-auto-columns: 180px;
    grid-auto-rows: 200px;
}

#grid1 > div {
    background: coral;
    margin: 10px;
    padding: 10px;
    color: white;
}
```

## Eksplicitni redovi i kolone

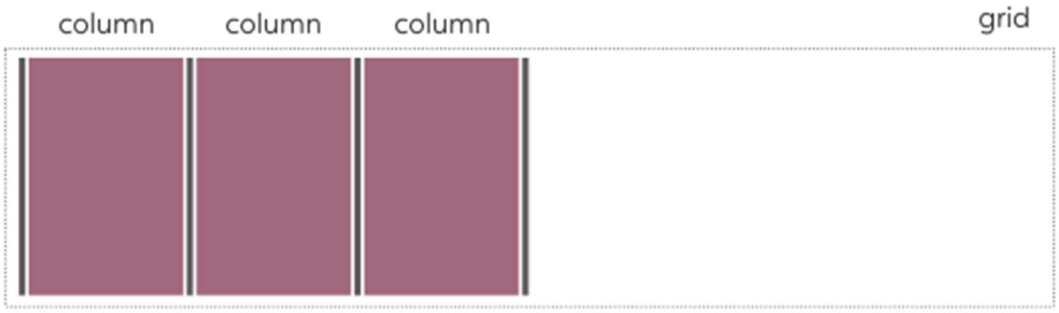
U prethodnim redovima ilustrovane su osobine automatskog raspoređivanja elemenata i implicitnih redova i kolona koji se tom prilikom kreiraju. Ipak, CSS Grid omogućava i preuzimanje pune kontrole nad kreiranjem kolona i redova. Takve kolone i redovi se drugačije nazivaju **eksplicitnim**.

Za kreiranje eksplicitnih kolona koristi se svojstvo **grid-template-columns**:

```
#grid1 {
display: grid;
grid-template-columns: 160px 160px 160px;
}
```

U primeru je upotrebljeno svojstvo `grid-template-columns`, čija se vrednost sastoji iz tri dela, odnosno iz tri pojedinačne vrednosti od `160px`. Na ovaj način je definisano da će CSS Grid imati tri kolone i da će svaka od njih biti široka po 160px (slika 21.4).

Copyright © Link group




*Slika 21.4. Grid sa tri kolone širine od po 160px*

U upravo prikazanom primeru, širine kolona unutar Grida definisane su korišćenjem apsolutnih jedinica – piksela (`px`). Naravno, prilikom definisanja kolona moguće je koristiti i druge jedinice i vrednosti. Na primer, moguće je napisati ovako nešto:

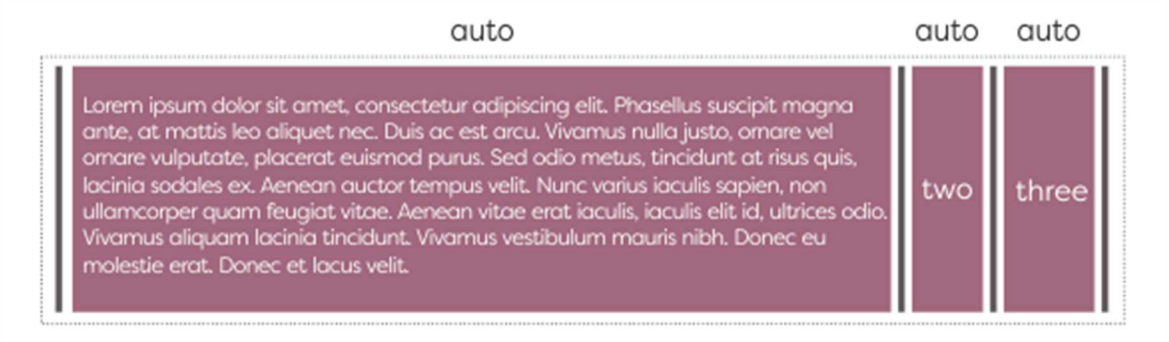
```
#grid1 {
display: grid;
grid-template-columns: auto auto auto;
}
```

Sada su pojedinačne vrednosti unutar `grid-template-columns` svojstva postavljene na `auto`. Na taj način, dostupna širina unutar Grida automatski se će se raspodeliti između svih kolona. Ukoliko ćelije unutar različitih kolona jednog reda poseduju identičnu količinu sadržaja, prostor će se jednako raspodeliti (slika 21.5).


*Slika 21.5. Grid sa tri auto kolone*

Vrednost `auto` prilikom raspodele prostora u obzir uzima količinu sadržaja koji se nalazi u pojedinačnim ćelijama. Tako će u slučaju postojanja nejednake količine sadržaja prednost biti data kolonama sa više sadržaja, dok će preostale kolone biti taman tolike da obuhvate sadržaj koji se unutar njih nalazi (slika 21.6).

Copyright © Link group




*Slika 21.6. Tri `auto` kolone sa različitom količinom sadržaja*

**fr jedinica**

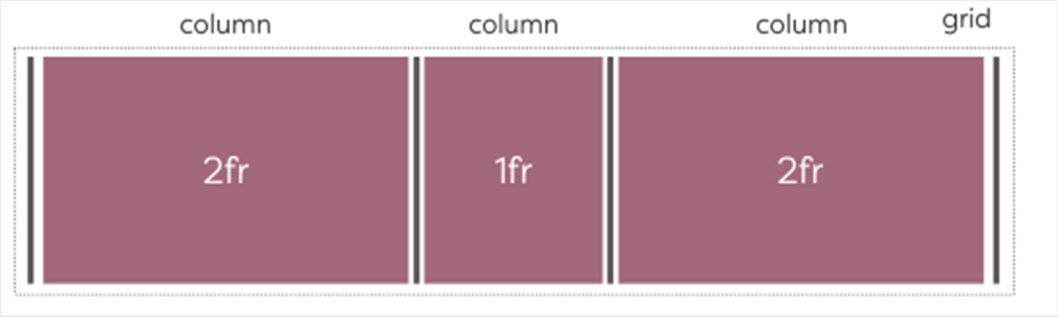
CSS poznaje i jednu posebnu jedinicu koja je specijalno kreirana za Grid sistem. Reč je o jedinici `fr`, što je zapravo skraćenica od reči *fractional*. Tako `fr` jedinica omogućava efikasnu raspodelu prostora unutar Grida:

```
#grid1 {
display: grid;
grid-template-columns: 1fr 1fr 1fr;
}
```

Na ovaj način je rečeno da će sve tri kolone Grida jednako podeliti dostupan prostor po širini. Naravno, `fr` jedinice su posebno korisne kada je potrebno izvršiti nejednaku podelu prostora između više kolona:

```
#grid1 {
display: grid;
grid-template-columns: 2fr 1fr 2fr;
}
```

Na ovaj način je definisano da će prva i treća kolona Grida imati duplo veću širinu od druge (slika 21.7).




*Slika 21.7. Efekat fr jedinica prilikom dimenzionisanja kolona*

Copyright © Link group

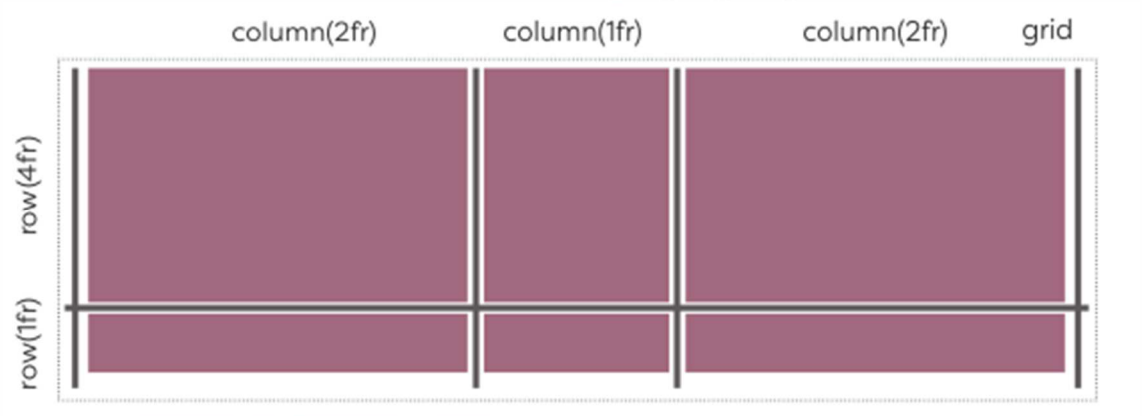
Bitno je razumeti da `fr` jedinica ne uzima u obzir sadržaj koji se nalazi unutar kolona ili redova. Tako se definisani odnos veličina uvek poštuje, za razliku od efekta koji se dobija korišćenjem `auto` vrednosti.

Upravo prikazani primeri ilustrovali su tehnike za eksplicitno definisanje kolona. U takvim situacijama, redovi se za nas automatski kreiraju u zavisnosti od broja Grid elemenata. Drugim rečima, kolone su eksplicitne, a redovi implicitni. Grid iz primera poseduje šest elemenata, pa je upravo zbog toga obavljeno automatsko kreiranje dva implicitna reda (3 kolone x 2 reda = 6 elemenata).

Redove unutar CSS Grida eksplicitno je moguće definisati korišćenjem svojstva **grid-template-rows**:

```
#grid1 {
    display: grid;
    grid-template-columns: 2fr 1fr 2fr;
    grid-template-rows: 4fr 1fr;
}
```

Na ovaj način, kolone i redovi su eksplicitno definisani i pri tome je korišćenjem `fr` jedinice definisan odnos njihovih visina i širina. Efekat je kao na slici 21.8.




*Slika 21.8. Grid sa tri kolone i dva reda, koji su definisani upotrebom fr jedinica*

Upravo je kreiran Grid sa tri eksplicitno definisane kolone i dva eksplicitno definisana reda. Čak i u ovakvoj situaciji, ukoliko broj elemenata premaši kapacitet eksplicitno definisanih kolona i redova, Grid će automatski dodati potreban broj novih redova ili kolona. Da li će automatski biti obavljeno kreiranje implicitnih redova ili kolona, pre svega zavisi od vrednosti svojstva `grid-auto-flow`.

Copyright © Link group

**Pitanje**

Kolone i redovi koje CSS Grid samostalno kreira nazivaju se:

- **implicitni**
- eksplicitni
- statički
- dinamički

**Objašnjenje:**

*Unutar Grida se mogu naći implicitne i eksplicitne kolone i redovi. Eksplicitne kolone i redove programer samostalno kreira, a implicitne kreira sam CSS Grid.*

## repeat() CSS funkcija

U jednom od prethodnih primera, tri kolone unutar Grida definisane su na sledeći način:

```
grid-template-columns: 160px 160px 160px;
```

Na ovaj način je obavljeno kreiranje tri eksplicitne kolone, pri čemu će svaka biti široka po 160px. Ali, šta ukoliko je potrebno kreirati veći broj kolona identične širine (10, 20 ili više)? U takvoj situaciji, svakako je legitimno napisati:

```
grid-template-columns: 40px 40px 40px 40px 40px 40px 40px 40px 40px 40px
40px 40px;
```

Primer ilustruje kreiranje 12 kolona od po 40px širine. Ipak, kod je vrlo nepregledan, a kako bi se učinio preglednijim, moguće je koristiti CSS funkciju **repeat()**:

```
grid-template-columns: repeat(12, 40px);
```

Prvi parametar unutar funkcije `repeat()` definiše broj kolona ili redova koje je potrebno kreirati, dok drugi definiše veličinu. Tako će prikazanim primerom biti kreirano 12 kolona, širine od po 40px.

## minmax() CSS funkcija

U određenim situacijama, veoma je korisna mogućnost definisanja minimalne i maksimalne veličine koju neka kolona ili red može imati. Tako nešto se unutar CSS Grida postiže korišćenjem CSS funkcije **minmax()**:

```
grid-auto-rows: minmax(120px, auto);
```

U primeru je CSS funkcija `minmax()` iskorišćena za definisanje vrednosti svojstva `grid-auto-rows`, odnosno za postavljanje visine implicitnih redova. Ipak, funkciju `minmax()` je moguće koristiti za definisanje veličine kako implicitnih tako i eksplicitnih redova i kolona.

Prvi parametar koji u prikazanom primeru funkcija `minmax()` prihvata (`120px`) odnosi se na minimalnu visinu koju implicitni redovi mogu imati. Drugi parametar (`auto`) definiše

Copyright © Link group

maksimalnu vrednost. Konkretnom vrednošću `auto`, definisano je da će maksimalna visina redova biti određena na osnovu količine sadržaja koji se unutar njih nalazi.

## Kombinovanje repeat() i minmax() funkcija

Veoma česta praksa prilikom definisanja kolona ili redova Grida, jeste kombinovanje funkcija `repeat()`i `minmax()`:

```
grid-template-columns: repeat(12, minmax(150px, auto));
```

Upravo prikazanom linijom, unutar Grida je definisano 12 kolona sa identičnim osobinama. Osobine takvih kolona određene su funkcijom `minmax()`. Minimalna širina kolona je postavljena na `150px`, dok maksimum nije definisan i zavisi od sadržaja unutar kolona.

Upravo prikazana linija koda predstavlja veoma dobro rešenje, sve dok se u priču ne uvrsti i responsive design. S obzirom na to da je unutar upravo prikazane linije kao prvi parametar funkcije `repeat()` postavljena vrednost 12, Grid će uvek imati 12 kolona, bez obzira na količinu dostupne širine unutar vidnog polja. Jasno je da 12 kolona neće zadovoljiti sve tipove uređaja od pametnih telefona do desktop kompjutera sa displejima velike dijagonale. Upravo zbog toga se prikazana linija može unaprediti korišćenjem dve posebne vrednosti:

- `auto-fit`
- `auto-fill`

Dve upravo prikazane vrednosti, koje se mogu definisati kao prvi parametar funkcije `repeat()`, govore web pregledaču da samostalno definiše broj kolona unutar Grida, a sve u zavisnosti od dostupnog prostora unutar vidnog polja i vrednosti koje su definisane kao minimalne i maksimalne za kolone Grida. Efekti ove dve vrednosti veoma su slični i između njih postoji samo jedna mala razlika.

Vrednost **auto-fill** učiniće da se red ispuni maksimalnim brojem kolona. Prilikom utvrđivanja maksimalnog broja kolona, u obzir se uzima definisana minimalna širina jedne kolone. Kada se unutar jednog reda utvrdi maksimalni broj kolona, u zavisnosti od broja elemenata unutar Grida, neke od takvih kolona mogu biti i prazne. Ipak, iako prazne, one svakako zauzimaju prostor unutar reda, baš kao da se unutar njih nalaze elementi.

Vrednost **auto-fit** čini da elementi unutar Grida zauzmu kompletan dostupan prostor unutar jednog reda. Eventualne prazne kolone se uklanjanju, a njihovo mesto zauzimaju kolone sa sadržajem. Opisane osobine vrednosti `auto-fill` i `auto-fit` ilustrovane su slikom 21.9.

Copyright © Link group


*Slika 21.9. Razlika između vrednosti auto-fill i auto-fit*

Upravo ilustrovane vrednosti `auto-fill` i `auto-fit` u narednoj lekciji će biti iskorišćene za kreiranje jednog primera responsive layouta korišćenjem CSS Grida.

Unutar radnog okruženja prikazano je korišćenje minmax funkcije unutar grid-auto-rows svojstva. Pokušajte da nad grid-auto-columns svojstvom upotrebite kombinaciju repeat i minmax funkcija.

**Radno okruženje**

**HTML fajl:**

```
<div id="grid1">
<div>One</div>
<div>Two</div>
<div>Three</div>
<div>Four</div>
<div>Five</div>
<div>Six</div>
</div>
```

**CSS fajl:**

```
#grid1 {
    display: grid;
    grid-auto-flow: column;
    grid-auto-columns: auto auto auto ;
    grid-auto-rows: minmax(120px, auto);
}

#grid1 > div {
    background: coral;
    margin: 10px;
    padding: 10px;
    color: white;
}
```

Copyright © Link group

## Definisanje proreda između redova i kolona

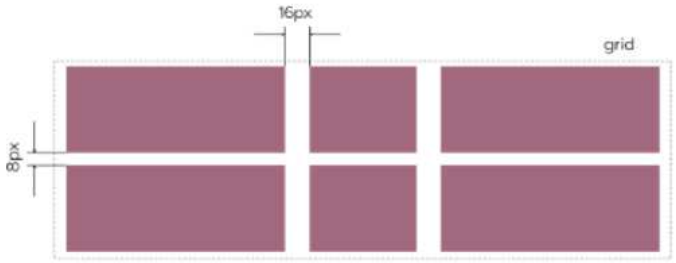
Prored je prostor unutar Grida koji razdvaja horizontalne i vertikalne trake, odnosno kolone i redove. Prored (*gap, gutter*) unutar CSS Grida može se kontrolisati korišćenjem svojstava:

- `column-gap` (`grid-column-gap`) – definiše prored između kolona;
- `row-gap` (`grid-row-gap`) – definiše prored između redova;
- `gap` (`grid-gap`) – definiše prored između redova i kolona;

Primer CSS Grida sa definisanim proredom između redova i kolona može da izgleda ovako:

```
#grid1 {
display: grid;
grid-template-columns: 2fr 1fr 2fr;
column-gap: 16px;
row-gap: 8px;
}
```

Efekat koji se na ovakav način dobija ilustrovan je slikom 21.10.



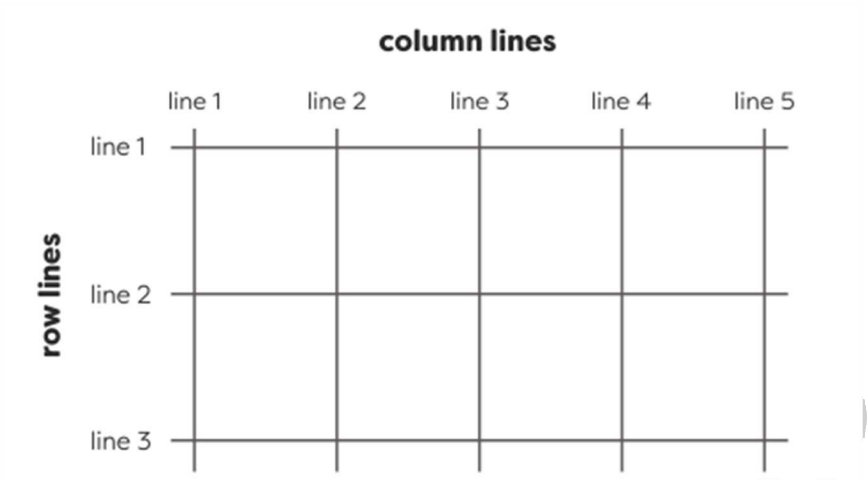

*Slika 21.10. Definisanje proreda između kolona i redova Grida*

Prilikom navođenja svojstava za kreiranje proreda unutar Grida, potrebno je zapaziti i svojstva sa prefiksom *grid-* koja su definisana u zagradama. Reč je o starijim varijantama svojstava koje su zamenjene novim, bez prefiksa.

## Raspoređivanje elemenata unutar Grida

Već više puta do sada rečeno je da osnovu Grid sistema čine horizontalne i vertikalne linije. Ipak, iako su osnova Grida, takvim linijama se ne može rukovati direktno. Jednostavno, one su imaginarni elementi koje automatski kreira sam Grid, i to u zavisnosti od broja redova i kolona. Podsetimo se kako linije Grid sistema izgledaju (slika 21.11).

Copyright © Link group




*Slika 21.11. Linije Grid sistema*

Sa slike 21.11. se može videti da se linije podrazumevano numerišu počevši od broja 1, i to sleva nadesno i odozgo nadole.

Linije su osnovna komponenta Grid sistema koja se koristi za pozicioniranje elemenata. Za obavljanje takvog posla koristi se nekoliko CSS svojstava (tabela 21.1).

| Svojstvo za pozicioniranje | Opis |
|---|---|
| `grid-column-start` | definiše startnu poziciju elementa unutar kolona Grida |
| `grid-column-end` | definiše krajnju poziciju elementa unutar kolona Grida |
| `grid-column` | svojstvo koje omogućava objedinjeno definisanje `grid-column-start` i `grid-column-end` svojstava; njihove vrednosti se odvajaju karakterom / |
| `grid-row-start` | definiše startnu poziciju elementa unutar redova Grida |
| `grid-row-end` | definiše krajnju poziciju elementa unutar redova Grida |
| `grid-row` | svojstvo koje omogućava objedinjeno definisanje `grid-row-start` i `grid-row-end` svojstava; njihove vrednosti se odvajaju karakterom / |

*Tabela 21.1. Svojstva za pozicioniranje elemenata unutar Grida*

Sledećih nekoliko primera ilustrovaće korišćenje upravo navedenih svojstava.

Prvi primer podrazumeva pozicioniranje jednog elementa unutar Grida sa tri kolone i dva reda. HTML struktura će izgledati ovako:

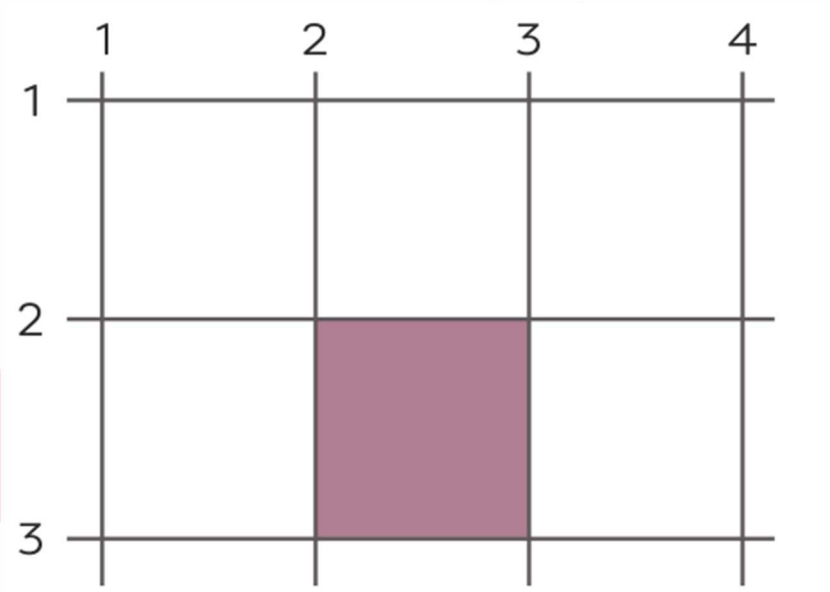
```
<div id="grid1">
<div id="box1">One</div>
</div>
```

Copyright © Link group

CSS kod je:

```
#grid1 {
display: grid;
grid-template-columns: 1fr 1fr 1fr;
grid-template-rows: 1fr 1fr;
column-gap: 16px;
row-gap: 8px;
}

#box1 {
grid-column-start: 2;
grid-column-end: 3;
grid-row-start: 2;
grid-row-end: 3;
}
```

Na ovaj način, kreira se jedan Grid sa eksplicitno definisanim kolonama i redovima. Kolona ima tri, a redova dva, što praktično znači da postoji ukupno šest ćelija unutar kojih je moguće smestiti elemente. Ipak, unutar Grida postoji samo jedan element i on je smešten u drugu kolonu drugog reda (slika 21.12).




*Slika 21.12. Element koji je pozicioniran u ćeliju preseka druge kolone i drugog reda Grid sistema*

Ovako nešto postignuto je korišćenjem svojstava `grid-column-start`, `grid-column-end`, `grid-row-start` i `grid-row-end`.

Element je pozicioniran u drugu kolonu, tako što je za vrednost svojstva `grid-column-start` postavljeno 2. To znači da će element biti pozicioniran počevši od druge vertikalne linije. Element će se prostirati sve do treće vertikalne linije, što je definisano CSS svojstvom `grid-column-end`. Na identičan način funkcionišu i svojstva `grid-row-start` i `grid-row-end`, samo što se njima konfiguriše pozicija elemenata u odnosu na horizontalne linije.

Copyright © Link group

Identičan efekat je mogao biti postignut i na nešto kompaktniji način:

```
#box1 {
grid-column: 2 / 3;
grid-row: 2 / 3;
}
```

Ovoga puta su za pozicioniranje elemenata upotrebljena svojstva `grid-column` i `grid-row`.

**grid-column i grid-row**

Svojstva `grid-column` i `grid-row` obezbeđuju jednostavniju sintaksu za raspoređivanje elemenata. Njihova struktura je sledeća:

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

Unutar radnog okruženja prikazan je gore pomenut primer. Pokušajte da isti prikaz postignete alternativnom metodom koristeći svojstva *grid-column* i *grid-row*.

Radno okruženje

**HTML fajl:**

```
<div id="grid1">
<div id="box1">1</div>
</div>
```

**CSS fajl:**

```
#grid1 {
display: grid;
grid-template-columns: 1fr 1fr 1fr;
grid-template-rows: 1fr 1fr;
column-gap: 16px;
row-gap: 8px;
}

#box1 {
height: 100px;
width: 100px;
color: white;
font-size: 5rem;
text-align: center;
background: coral;
grid-column-start: 2;
grid-column-end: 3;
grid-row-start: 2;
grid-row-end: 3;
}
```

Copyright © Link group

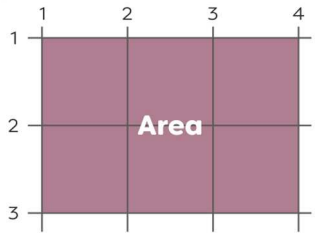
## Oblasti unutar Grida

Prethodni primer ilustrovao je pozicioniranje jednog Grid elementa unutar jedne ćelije. Unutar CSS Grida, jedan element je moguće smestiti i unutar većeg broja susednih ćelija. Na taj način se dobija oblast (*area*).

Sledeći primer ilustruje kreiranje oblasti unutar Grida:

```
#box1 {
grid-column: 1 / 4;
grid-row: 1 / 3;
}
```

Na ovaj način definisano je da će se Grid element protezati kroz tri kolone i dva reda, te će na taj način biti stvorena jedna oblast (slika 21.13).




*Slika 21.13. Jedna oblast unutar Grid sistema*

Kreiranje Grid oblasti moguće je jednostavnije obaviti korišćenjem CSS svojstva `grid-area`.

**grid-area**

Svojstvo `grid-area` može se koristiti za jednostavnije kreiranje oblasti. Tako ovo svojstvo objedinjuje svojstva `grid-column-start`, `grid-column-end`, `grid-row-start` i `grid-row-end`. Njegova sintaksa izgleda ovako:

```
grid-area: grid-column-start / grid-row-start / grid-column-end /
grid-row-end;
```

## Imenovane oblasti

Prethodni primeri podrazumevali su kreiranje oblasti na osnovu linija Grid sistema. Pored takvog pristupa, CSS poznaje i pojam imenovanih oblasti, koje se kreiraju na nešto drugačiji način. HTML struktura primera će izgledati ovako:

```
<div id="grid1">
<div id="box1">Area 1</div>
<div id="box2">Area 2</div>
</div>
```

Element `grid1` biće Grid kontejner koji će unutar sebe imati dva Grid elementa. Svi oni će biti stilizovani na sledeći način:

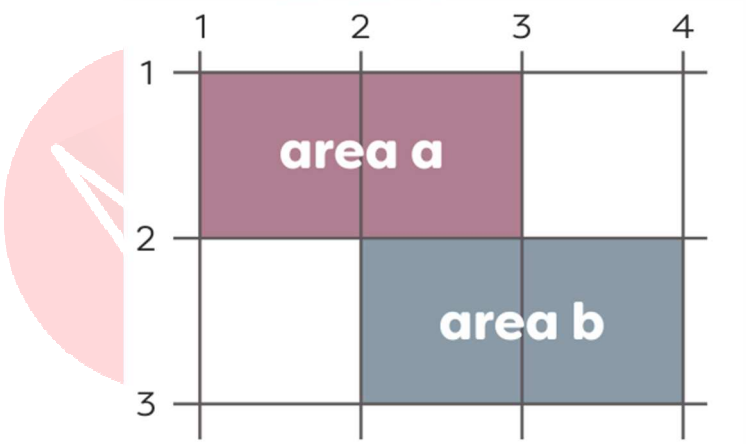
Copyright © Link group

```
#grid1 {
    display: grid;
    grid-template-columns: 160px 160px 160px;
    column-gap: 16px;
    row-gap: 8px;

    grid-template-areas:
        "a a ."
        ". b b";
}

#box1 {
    grid-area: a;
}

#box2 {
    grid-area: b;
}
```

Neophodno je primetiti upotrebu dva nova CSS svojstva:

- `grid-template-areas` – svojstvo koje se koristi za raspoređivanje imenovanih oblasti;
- `grid-area` – svojstvo za imenovanje oblasti.

Korišćenjem svojstva `grid-area` izvršeno je imenovanje dve oblasti. Njihova imena su postavljena na *a* i *b*. Korišćenjem svojstva `grid-template-areas` obavljeno je raspoređivanje oblasti unutar Grida. Na ovaj način se dobija efekat ilustrovan slikom 21.14.




*Slika 21.14. Dve oblasti (a, b) unutar Grid sistema*

Na slici 21.14. može se videti da obe kreirane oblasti obuhvataju po dve ćelije Grid sistema. Prva se proteže kroz prvu i drugu kolonu prvog reda, a druga kroz drugu i treću kolonu drugog reda. Ovakav raspored je postignut definisanjem vrednosti svojstva `grid-template-areas` u posebnom obliku. Ukoliko malo bolje pogledate vrednost ovog svojstva, moći ćete da vidite

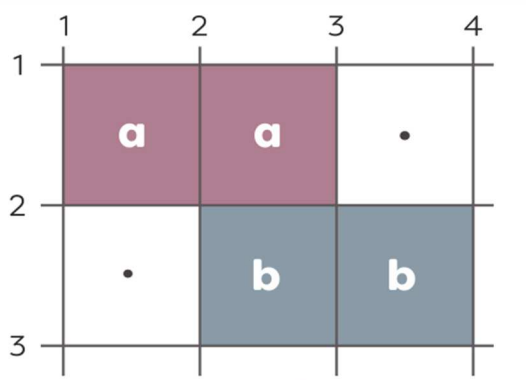
Copyright © Link group

da ona oponaša strukturu CSS Grida na koji se primenjuje. Sastoji se iz dva pojedinačna dela izdvojena navodnicima – po jedan za svaki red Grida.

"`a a .`" označava da će se oblast sa nazivom `a` prostirati preko prve i druge kolone prvog reda, dok će treća kolona prvog reda biti prazna, s obzirom na to da je na trećoj poziciji postavljen karakter tačka (.).

"`. b b`" znači da će prva kolona drugog reda biti prazna (zato što je na prvom mestu tačka), a da će druga i treća kolona drugog reda biti okupirane oblašću sa nazivom `b`.

Upravo opisana logika ilustrovana je slikom 21.15.




*Slika 21.15. Logika po kojoj funkcioniše svojstvo grid-template-areas*

## Rezime

- Elementi unutar Grida se podrazumevano smeštaju unutar jedne kolone.
- Kolone i redovi unutar Grida mogu biti implicitni i eksplicitni.
- Eksplicitne kolone i redove programer samostalno kreira.
- Implicitne kolone i redove kreira sam CSS Grid, ukoliko nema dovoljno eksplicitnih kolona i redova za prikaz elemenata.
- Postupak kojim se obavlja kreiranje implicitnih redova i kolona funkcioniše na osnovu algoritma za automatsko raspoređivanje elemenata.
- Na osobine automatskog raspoređivanja elemenata može se uticati korišćenjem CSS svojstva `grid-auto-flow`.
- Ukoliko se vrednost svojstva `grid-auto-flow` postavi na `column`, elementi Grida se raspoređuju u jednom redu, odnosno svaki element u novoj koloni.
- Na osobine implicitno kreiranih redova i kolona moguće je uticati korišćenjem dva CSS svojstva: `grid-auto-columns` i `grid-auto-rows`.
- Za kreiranje eksplicitnih kolona koristi se svojstvo `grid-template-columns`.
- Vrednost `auto` svojstva `grid-template-columns` prilikom raspodele prostora u obzir uzima količinu sadržaja koji se nalazi u pojedinačnim ćelijama.
- Jedinica `fr` omogućava efikasnu raspodelu prostora unutar Grida.
- `fr` ne uzima u obzir sadržaj koji se nalazi unutar kolona ili redova, tako da se definisani odnos veličina uvek poštuje.
- Redove unutar CSS Grida je eksplicitno moguće definisati korišćenjem svojstva `grid-template-rows`.

Copyright © Link group

- CSS funkcija `repeat()` može se koristiti za jednostavno definisanje većeg broja kolona ili redova identičnih osobina.
- CSS funkcija `minmax()` se može koristiti za definisanje maksimalne i minimalne veličine Grid kolona i redova.
- `column-gap` (`grid-column-gap`) definiše prored između kolona.
- `row-gap` (`grid-row-gap`) definiše prored između redova.
- `gap` (`grid-gap`) definiše prored između redova i kolona.
- Linije su osnovna komponenta Grid sistema koja se koristi za pozicioniranje elemenata; za pozicioniranje elemenata na osnovu linija Grid sistema mogu se koristiti svojstva `grid-column-start`, `grid-column-end`, `grid-column`, `grid-row-start`, `grid-row-end`, `grid-row`.
- Kada se jedan element smesti unutar većeg broja susednih ćelija, dobija se oblast.
- Kreiranje Grid oblasti najjednostavnije je obaviti korišćenjem CSS svojstva `grid-area`.
- Svojstvo `grid-area` koristi se za kreiranje imenovane oblasti.
- Svojstvo `grid-template-areas` koristi se za raspoređivanje imenovanih oblasti.

Copyright © Link group