Flex elementi

U prethodnim lekcijama ilustrovane su osnove funkcionisanja Flexbox sistema za raspoređivanje elemenata. Demonstrirana su različita CSS svojstva koja se mogu koristiti na kontejnerskim elementima. Pored takve grupe svojstava, Flexbox poznaje i različita svojstva koja se mogu definisati na fleksibilnim elementima, tj. potomcima. Njima će biti posvećena lekcija pred vama.

Svojstva flex elemenata

Svojstva koja se mogu definisati na fleksibilnim elementima, tj. potomcima su sledeća:

- `order`
- `flex-basis`
- `flex-grow`
- `flex-shrink`
- `flex`
- `align-self`

Order

CSS svojstvom `order` utiče se na redosled elemenata unutar kontejnera. Podrazumevano, elementi se unutar kontejneraređaju redosledom kojim su navedeni u HTML dokumentu. Tako nešto je moguće promeniti definisanjem vrednosti svojstva `order` na pojedinačnim fleksibilnim elementima.

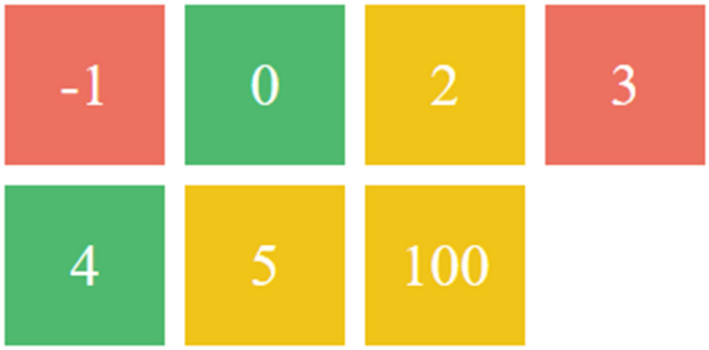
Sledeći primer ilustruje definisanje redosleda elemenata:

```
<div id="flexbox1">
      <div style="order: 5;">5</div>
      <div style="order: 4;">4</div>
      <div style="order: 3;">3</div>
      <div style="order: 2;">2</div>
      <div>0</div>
      <div style="order: -1;">-1</div>
      <div style="order: 100;">100</div>
</div>
```

Na elementima koji se nalaze unutar Flexbox kontejnera, definisana su CSS svojstva `order`. Njihove vrednosti oslikavaju redosled kojim će fleksibilni elementi biti poređani.

Takođe, bitno je primetiti da nad jednim `div` elementom nije definisano `order` svojstvo. To je učinjeno sa razlogom, kako bi se ilustrovalo da je podrazumevana vrednost `order` svojstva nula (0).

Prikazani kod proizvešće efekat kao na slici 19.1.



Slika 19.1. Efekat order svojstva

Analizom slike 19.1. lako se zaključuje da se elementi sa nižom vrednošću `order` svojstva prikazuju pre onih sa višom.

Flex basis

Inicijalna veličina koju je potrebno da fleksibilni element poseduje unutar kontejnera definiše se korišćenjem svojstva `flex-basis`. Ovo svojstvo može da prihvati vrednosti izražene različitim apsolutnim ili relativnim jedinicama. Sledeći primer ilustruje upotrebu `flex-basis` svojstva.

```
<div id="flexbox1">
      <div style="flex-basis: 5em;">5em</div>
      <div style="flex-basis: 10em;">10em</div>
      <div style="flex-basis: 5em;">5em</div>
</div>
```

U prikazanom primeru navedene su vrednosti korišćenjem relativnih `em` jedinica. Efekat će biti kao na slici 19.2.

Slika 19.2. Efekat flex-basis svojstva

Sa slike 19.2. može se videti da elementi imaju inicijalnu veličinu koja je zadata korišćenjem svojstva `flex-basis`.

Pored vrednosti iskazanih relativnim i apsolutnim jedinicama, vrednost svojstva `flex-basis` može se postaviti i na `auto`. To je inače podrazumevana vrednost, po kojoj se za utvrđivanje veličine elementa uzima u obzir veličina koja je definisana korišćenjem CSS `width` ili `height` svojstava. Ukoliko element ne poseduje eksplicitno definisanu veličinu ni korišćenjem ovih svojstava, njegova veličina će biti taman tolika da obuhvati pripadajući sadržaj.

Flex grow

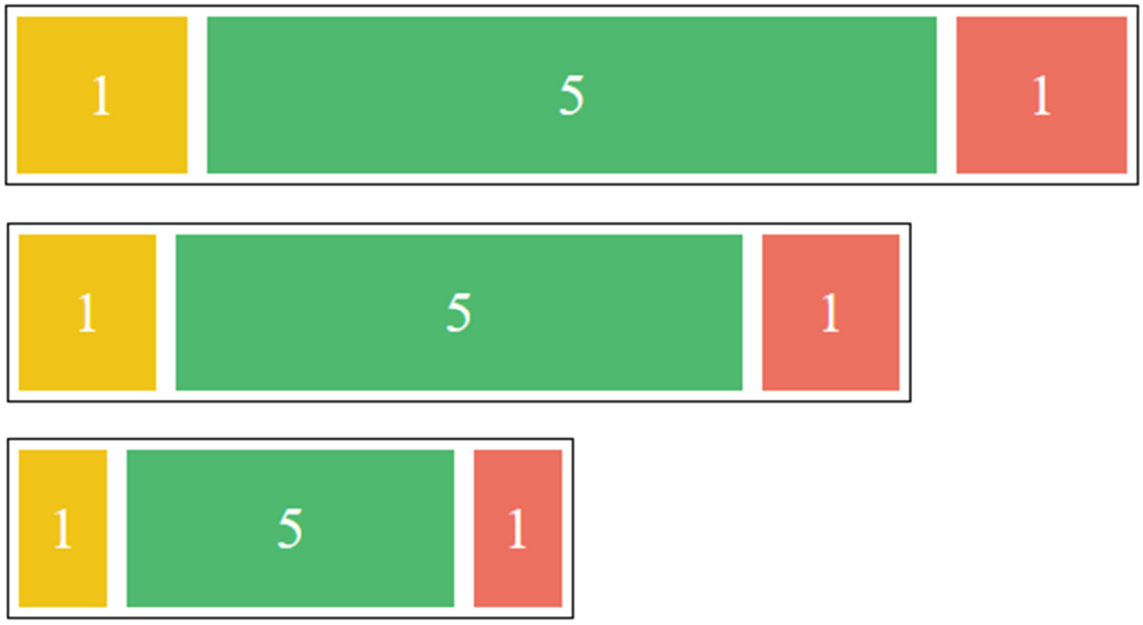
Upravo je prikazano kako se definiše početna veličina fleksibilnih elemenata duž glavne ose. Postavlja se pitanje, šta će se dogoditi sa veličinom elemenata u situacijama kada je dostupan prostor kontejnera veći ili manji od onog koji je potreban za prikaz elemenata sa inicijalnom veličinom?

Svojstvo `flex-grow` utiče na mogućnost povećavanja elementa ukoliko postoji prostor koji je potrebno popuniti. Spomenuti dostupni prostor može se videti na slici 19.2, na desnoj strani.

Svojstvo `flex-grow` prihvata celobrojnu pozitivnu vrednost, koja predstavlja proporciju koju element treba da zauzme unutar svog roditeljskog kontejnera. Podrazumevana vrednost je nula, što znači da element neće povećavati svoju veličinu. Sledeći primer ilustruje upotrebu svojstva `flex-grow`:

```
<div id="flexbox1">
      <div style="flex-grow: 1;">1</div>
      <div style="flex-grow: 5;">5</div>
      <div style="flex-grow: 1;">1</div>
</div>
```

Na elementima su definisane vrednosti `flex-grow` svojstava. Prvi i poslednji element imaju vrednost 1, dok središnji element ima vrednost 5. Na ovaj način je definisano da će središnji element uvek biti pet puta veći od prvog i poslednjeg. Takvo ponašanje ilustruje slika 19.3.



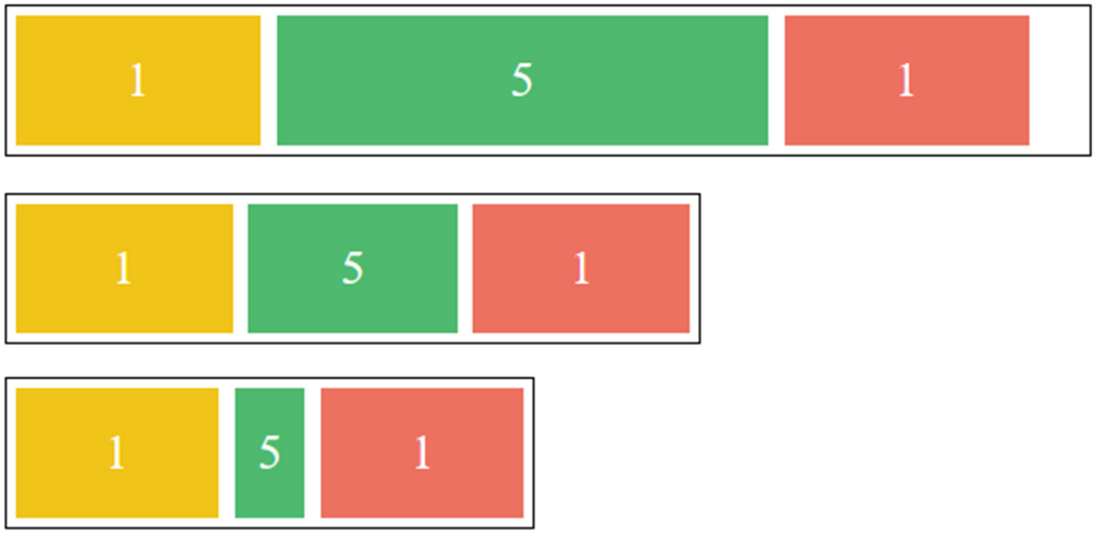
Slika 19.3. Efekat flex-grow svojstva

Flex shrink

Podrazumevano, kada se veličina kontejnera smanji, smanjuju se i veličine njegovih elemenata. Svojstvo `flex-shrink` definiše koliko će se element smanjivati u odnosi na ostale elemente Flexbox kontejnera. Sledeći primer ilustruje upotrebu ovog svojstva:

```
<div id="flexbox1">
      <div style="flex-basis: 5em; flex-shrink: 1;">1</div>
      <div style="flex-basis: 10em; flex-shrink: 5;">5</div>
      <div style="flex-basis: 5em; flex-shrink: 1;">1</div>
</div>
```

U primeru je definisano da će se središnji `div` element smanjiti (ukoliko je to moguće) pet puta više od elemenata koji kao vrednost svojstva `flex-shrink` imaju vrednost 1. Efekat je ilustrovan slikom 19.4.



Slika 19.4. Efekat flex-shrink svojstva

Flex

S obzirom na to da su prethodna tri navedena svojstva, `flex-basis`, `flex-grow` i `flex-shrink` povezana, CSS poznaje svojstvo `flex`, kojim je moguće na lak način definisati vrednosti sva tri spomenuta svojstva. Njegova sintaksa je sledeća:

```
flex: flex-grow flex-shrink flex-basis;
```

Sledeći primer ilustruje definisanje vrednosti `flex` svojstva:

```
flex: 2 2 10%;
```

Svojstvo `flex` je moguće navesti sa jednom, dve ili tri vrednosti. Različiti scenariji postavljaju vrednosti različitih svojstava, a sledeći primer ilustruje takve različite situacije:

```
/* 0 0 auto */
flex: none;

/* One value, unitless number: flex-grow */
flex: 2;

/* One value, width/height: flex-basis */
flex: 10em;
flex: 30px;
flex: auto;
flex: content;

/* Two values: flex-grow | flex-basis */
flex: 1 30px;

/* Two values: flex-grow | flex-shrink */
flex: 2 2;

/* Three values: flex-grow | flex-shrink | flex-basis */
flex: 2 2 10%;
```

Pitanje

Odabrati svojstvo čija vrednost se ne može postaviti korišćenjem svojstva `flex`:

- `flex-basis`
- `flex-grow`
- `flex-shrink`
- **`order`**

Objašnjenje:

Svojstvo `flex` može se koristi za objedinjeno definisanje vrednosti svojstava `flex-basis`, `flex-grow` i `flex-shrink`, ali ne i svojstva `order`.

Align self

U prethodnoj lekciji opisano je svojstvo `align-items`, kojim se postiže poravnavanje elemenata po osi preseka. Takvim svojstvom se poravnavaju svi elementi kontejnera, a ukoliko je potrebno definisati poravnanje za samo jedan element, može se koristiti svojstvo `align-self`. Na taj način, moguće je pregaziti način poravnanja koji je definisan za ceo kontejner i definisati specifično poravnanje na nivou jednog elementa. Svojstvo `align-self` može imati vrednosti identične onima koje može imati svojstvo `align-items` (tabela 19.1).

Vrednost	Opis
`flex-start`	element se poravnava u odnosu na početak ose preseka
`flex-end`	element se poravnava u odnosu na kraj ose preseka
`center`	element se centrira po osi preseka
`baseline`	element se poravnava po osnovnoj liniji sadržaja
`stretch`	element se razvlači kako bi popunio dostupan prostor unutar kontejnera po osi preseka; ovo je podrazumevana vrednost

Tabela 19.1. Vrednost align-self svojstva

Napomena

Vrednost `baseline` na samo jednom elementu ima efekat identičan onom koji proizvodi vrednost `flex-start`. Zato je za testiranje efekta `baseline` vrednosti neophodno postaviti takvo poravnanje na najmanje dva elementa.

Primer 1 – Kreiranje layouta korišćenjem Flexboxa

Naredni primer prikazaće kreiranje standardnog layouta web strane korišćenjem Flexbox pristupa. Struktura HTML elemenata izgledaće ovako:

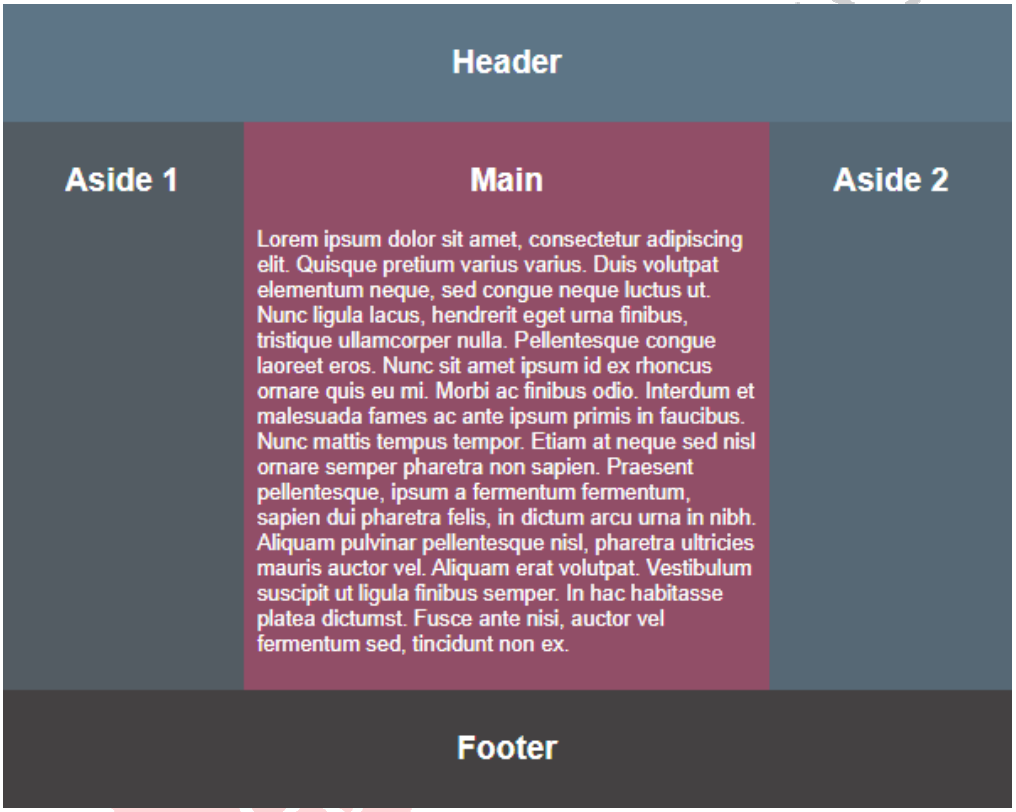
```
<div id="wrapper">
      <header id="header">
            <h2>Header</h2>
      </header>
      <aside class="aside" id="aside-1">
            <h2>Aside 1</h2>
      </aside>
      <div id="main">
            <h2>Main</h2>
            <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.
            Quisque pretium varius varius. Duis volutpat elementum
            neque, sed congue neque luctus ut. Nunc ligula lacus,
            hendrerit eget urna finibus, tristique ullamcorper nulla.
            Pellentesque congue laoreet eros. Nunc sit amet ipsum id ex
            rhoncus ornare quis eu mi. Morbi ac finibus odio. Interdum
            et malesuada fames ac ante ipsum primis in faucibus. Nunc
            mattis
            tempus tempor. Etiam at neque sed nisl ornare semper
            pharetra non sapien. Praesent pellentesque, ipsum a
            fermentum fermentum, sapien dui pharetra felis, in dictum
            arcu urna in nibh. Aliquam pulvinar pellentesque nisl,
            pharetra ultricies mauris auctor vel. Aliquam erat volutpat.
            Vestibulum suscipit ut ligula finibus semper. In hac
            habitasse platea dictumst. Fusce ante nisi, auctor vel
            fermentum sed, tincidunt non ex. </p>
      </div>
      <aside class="aside" id="aside-2">
            <h2>Aside 2</h2>
      </aside>
      <footer id="footer">
            <h2>Footer</h2>
      </footer>
</div>
```

Element sa id-jem `wrapper` poslužiće kao Flexbox kontejner (omotač), dok će elementi `header`, `article`, `footer` i dva `aside` elementa biti Flexbox elementi. Kako bi se tako nešto postiglo, koristiće se sledeći CSS kod:

```
body {
	margin: 0;
	color: white;
	font-family: sans-serif;
}

#wrapper {
	display: flex;
	flex-flow: row wrap;
	text-align: center;
	margin: 0;
}

#header,
#footer {
	flex-basis: 100%;
	/*shorthand*/
	/*flex: 1 100%;*/
}

.aside {
	flex-basis: auto;
	flex-grow: 1;
	/*shorthand*/
	/*flex: 1 auto; */
}

#main {
	flex-grow: 5;
	flex-basis: 0%;
	/*shorthand*/
	/*flex: 5;*/
}

#header,
#footer,
.aside,
#main {
	padding: 10px;
}

#header {
	background-color: #5D7586;
}

#main {
	background-color: #914E67;
	text-align: left;
}

#main>h2 {
	text-align: center;
}
```

```
#footer {
	background-color: #444142;
}

#aside-1 {
	background-color: #535c63;
}

#aside-2 {
	background-color: #566875;
}
```

Efekat koji se na ovakav način dobija prikazan je slikom 19.5.



Slika 19.5. Primer realizacije layouta korišćenjem Flexboxa

Primer 2 – Kreiranje prilagodljivog grida korišćenjem Flexboxa

U jednom od prethodnih modula ilustrovano je kreiranje prilagodljivog grida korišćenjem osnovnih CSS pristupa koji su podrazumevali upotrebu svojstava `float` i `clear` u kombinaciji sa medija upitima. U narednim redovima će biti prikazano kako identičan efekat dobiti korišćenjem Flexboxa.

HTML struktura primera će izgledati ovako:

```
<div class="container">
      <div class="row">
            <div class="column">
            <div class="box"></div>
            </div>
            <div class="column">
            <div class="box"></div>
            </div>
            <div class="column">
            <div class="box"></div>
            </div>
            <div class="column">
            <div class="box"></div>
            </div>
            <div class="column">
            <div class="box"></div>
            </div>
            <div class="column">
            <div class="box"></div>
            </div>
      </div>
</div>
```

Element sa klasom `container` je okružujući element za kompletan grid. On će biti stilizovan na sledeći način:

```
.container {
      max-width: 960px;
      width: 90%;
      margin: 0 auto;
}
```

Širina glavnog okružujućeg elementa ograničena je na 960px. Kada se unutar viewporta ne može smestiti grid ove širine, njegova širina će iznositi 90% širine vidnog polja. Svojstvom `margin` obavljeno je centriranje ovog elementa unutar prozora web pregledača.

Prikazani grid sastoji se iz jednog reda i šest kolona. Red će biti stilizovan na sledeći način:

```
.row {
      display: flex;
      flex-flow: row wrap;
      justify-content: flex-start;
}
```

Red grida je proglašen Flexbox kontejnerskim elementom, definisanjem svojstva `display` sa vrednošću `flex`. Zatim su definisane orijentacija glavne ose i mogućnost prelamanja redova, a sve to korišćenjem svojstva `flex-flow`, kojim se zapravo objedinjeno definišu vrednosti svojstava `flex-direction` i `flex-wrap`. Svojstvom `justify-content` definisano je poravnanje svih flex elemenata, koji će seređati od početka glavne ose.

U prikazanom primeru, fleksibilni elementi su zapravo kolone grid sistema. Njih ima ukupno šest, a stilizovane su na sledeći način:

```
.column {
	flex-basis: 16.666%;
	box-sizing: border-box;
	padding: 8px;
}
```

Korišćenjem svojstva `flex-basis` definisana je inicijalna veličina koju je potrebno da fleksibilni elementi, odnosno kolone, poseduju unutar redova. Vrednost ovog svojstva je postavljena na 16.666%, a to je vrednost koja se dobija kada se 100 podeli sa 6 (broj kolona). Na kolonama grida definisano je i svojstvo `box-sizing` sa vrednošću `border-box`, kako bi unutrašnji razmak i okviri ušli u veličinu elemenata.

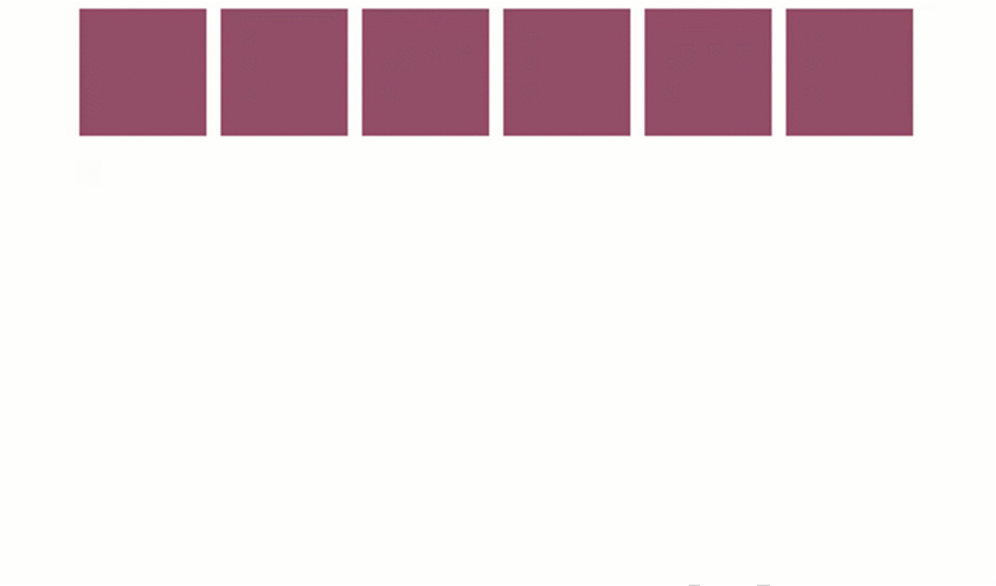
Na kraju, svaka kolona poseduje imaginarni sadržaj koji je predstavljen elementima sa klasom `box`. Oni su stilizovani na sledeći način:

```
.box {
	height: 144px;
	background-color: #914E67;
}
```

Kako bi upravo kreirani grid postao responsive, dovoljno je kontrolisati širinu kolona, jednostavnom intervencijom nad svojstvom `flex-basis`:

```
@media screen and (max-width: 1024px) {
	.column {
	flex-basis: 33.33%;
	}
}
@media screen and (max-width: 768px) {
	.column {
	flex-basis: 50%;
	}
}
@media screen and (max-width: 414px) {
	.column {
	flex-basis: 100%;
	}
}
```

Na ovaj način biće dobijen efekat ilustrovan animacijom 19.1.



Animacija 19.1. Primer realizacije responsive grida korišćenjem Flexboxa

Rezime

- Fleksibilni elementi se unutar kontejnera podrazumevano ređaju redosledom kojim su navedeni u HTML dokumentu.
- CSS svojstvom `order` utiče se na redosled elemenata unutar kontejnera.
- Inicijalna veličina koju je potrebno da fleksibilni element poseduje unutar kontejnera definiše se korišćenjem svojstva `flex-basis`.
- Svojstvo `flex-grow` utiče na mogućnost povećavanja elementa, ukoliko postoji prostor koji je potrebno popuniti.
- Svojstvo `flex-shrink` definiše koliko će se element smanjivati u odnosi na ostale elemente Flexbox kontejnera.
- Svojstvo `flex` može se koristi za objedinjeno definisanje vrednosti svojstava `flex-basis`, `flex-grow` i `flex-shrink`.
- Poravnanje pojedinačnih elemenata po osi preseka se može obaviti korišćenjem svojstva `align-self`.