Uvod u pseudo klase

- **Pseudo klase (*pseudo-class*)** u CSS-u se dodaju na selektore kako bi mogli da stilizujemo određena, specijalna stanja u kojima se mogu naći HTML elementi.
- Ta stanja često ne zavise od stabla HTML dokumenta (elemenata), već od situacije u kome se element nalazi i ponašanja korisnika.

Pseudo klase

Pseudo klase prepoznajemo po dodatku znaka dvotačke (*colon*). Najčešće se koriste kod linkova i tada možemo imati ovakav primer u CSS-u:

```
a:link { color:#669900; text-decoration:none;}
a:visited { color:#669900; text-decoration:none;}
a:hover { color:#336600; text-decoration:underline;}
a:active { color:#336600;}
```

Pseudo klase

- Ovi navedeni opisi u primeru na prethodnom slajdu stilizuju razna stanja linkova na stranici. Posebno smo stilizovali obično stanje linkova (:*link*), kao i kada je link već posećen (:*visited*), kada je kursor miša iznad linka (:*hover*) i, naposletku, stanje u trenutku klika, tj. aktivacije samog linka (:*active*).
- Primećujemo i na ovom primeru da upotrebom pseudo klasa **nismo stilizovali posebne elemente** u HTML dokumentu, **već samo četiri stanja istog elementa**. Otud i naziv pseudo.

- Postoje i mnoge druge pseudo klase sa kojima ćemo se upoznati i mogu biti veoma korisne u velikom broju raznih situacija. Neke su uvedene u CSS specifikaciju još u verziji 2, dok su se neke pojavile kasnije u verziji 3.
- Osim Internet Explorera, drugi browseri ih uglavnom normalno prepoznaju i primenjuju. Što se tiče IE, verzija 8 je donela delimičnu podršku, dok IE9 se ponašaju mnogo bolje po tom pitanju.

Pseudo klase kod formi

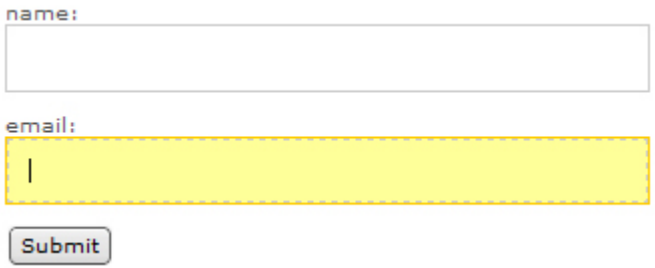
- Kod unosa formi, često se koriste pseudo klase radi lakše i brže stilizacije stanja.

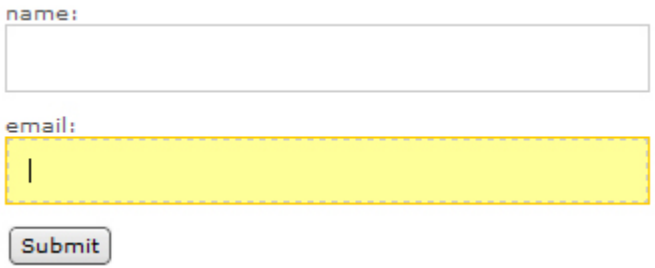
:focus

- :**focus** pseudo klasa se koristi za označavanje i stilizaciju fokusiranog elementa.
- Element je u fokusu kada ga aktiviramo, tj. pomoću tastature (taster tab) „dođemo do njega“. Osim toga, koristi se kod formi kada unosimo, popunjavamo polja. Dokle god je kursor u polju forme, i dok posetilac sajta unosi tekst, element forme je u stanju focus.
- Iako se uglavnom koristi za inpute u formama, može se primeniti za bilo koji element koji podržava :*hover* stanje

:focus

- Na slici iznad vidimo jednu jednostavnu formu koja sadrži dva tekstualna polja (input – text) i dugme submit (input – submit).
- Stilizovali smo tekstualna polja koristeći selektor atributa koji smo ranije pomenuli, a dodat je i opis za tekstualno polje u stanju focus...





Obratite pažnju da dugme submit nije dobilo nikakvu stilizaciju jer smo se ograničili na tekstualni tip inputa. Pri tom, ova dva navedena opisa u kodu iznad se dopunjuju. Prvi se uvek primenjuje nad tekstualnim input poljima, dok se drugi dodaje samo kada smo u focus stanju.

```
input[type="text"] {
 background:#FFF;
 border:1px solid #ccc;
 margin:0;
 padding:10px;
}

 input[type="text"]:focus {
 background:#FF9;
 border:1px solid #FC0;
 outline:1px dashed #ccc;
}
```

Pseudo klase kod formi

Pored focusa, postoje i sledeće pseudo klase za forme:

:enabled – Stilizuje sve dostupne i aktivne elemente.

:disabled – Kao što i ime kaže, stilizuje elemente u formi koji su disabled, odnosno one koje smo isključili. Može biti korisno oko stilizacije takvih elemenata. Po podrazumevanim podešavanjima takvi elementi su bleđi od ostalih, ali ih možemo promeniti.

:checked – Koristi se kod checkbox, radio i option kontrole. Stilizuje ih kada su štiklirane/označene.

Pseudo klasa :target

Ovu pseudo klasu koristimo u kombinaciji sa ID vrednošću određenog elementa. Upotreba je prilično jednostavna.

Ukoliko na stranici postoji, na primer, element sa ID vrednošću ***about*** i ukoliko se u trenutnoj URL adresi u browseru nalazi www.mojsajt.com/stranica.html#about onda će selektor **#about:target** funkcionisati i biti aktivan.

Pseudo klasa :target

Verovatno već pretpostavljate, :target se uglavnom koristi uz imenovana sidra (*named anchors*) kada linkujemo ka određenom delu iste ili druge stranice. **Sada uz dodatak ove pseudo klase možemo stilizovati odeljak kada se do njega dođe, kada on postane meta (target).**

Ipak, :target nije idealan i retko kad se koristi u praksi.

Relacione pseudo klase :not i :empty

Veoma zanimljiva pseudo klasa je **:not** koja proverava negaciju, koristeći za argument neki drugi jednostavan selektor. Argument pišemo u običnim zagradama.

Na primer, ukoliko postavimo:

p:not(.extended)

Ovaj selektor će pogoditi sve paragrafe koji **nemaju** klasu *extended*.

Relacione pseudo klase :not i :empty

:not ne prihvata za argument pseudo elemente niti neku drugu negaciju, ali se mogu koristiti selektori atributa.

Relacione pseudo klase :not i :empty

Može se koristiti i bez klasičnog selektora ispred. Na primer:

:not(#xyz)

Ali u tom slučaju bi ovaj opis gađao **sve** elemente **koji nemaju id vrednost *xyz***, pa čak i html, body i sve slične elemente.

Relacione pseudo klase :not i :empty

Sličan ovoj klasi je i pseudo klasa **:empty** kojom možemo selektovati npr. prazne paragrafe:

p:empty

što bi se odnosilo samo na:

** **

Pseudo klase u odnosu na poziciju elementa

- Možemo korisiti razne pseudo klase kojima na osnovu pozicije u dokumentu možemo izdvojiti određeni element i stilizovati ga bez upotrebe CLASS ili ID vrednosti.

:first-child

Selektuje i stilizuje prvi element unutar roditeljskog elementa. Možemo definisati tip roditeljskog i/ili tip samog elementa.

Na primer, možemo postaviti:

p:first-child

p:first-child

Ovako stilizujemo **svaki prvi paragraf u bilo kom roditeljskom elementu.**

Ukoliko imamo samo nekoliko paragrafa unutar body elementa, prvi od njih biće stilizovan. Ukoliko u body elementu imamo dva div elementa, i u njima po nekoliko paragrafa, i u jednom i drugom divu prvi paragraf će biti stilizovan ovom pseudo klasom, odnosno opisom koji sadrži ovaj selektor sa pseudo klasom.

Sa druge strane, ako postavimo:

#sidebar :first-child

Stilizujemo prvi element u *#sidebar* elementu, ma kog tipa taj prvi element bio. Dalje, ukoliko napišemo:

#sidebar p:first-child

Stilizujemo prvi pragraf unutar *#sidebar* elementa.

I ostali selektori u ovoj grupi se ponašaju slično po pitanju roditeljskih elemenata i tipova elementa

:last-child – Ovaj selektor je veoma sličan prethodnom, osim što pogađa poslednji element.

:nth-child(N)

- Selektuje element na osnovu rednog broja, tj. mesta pojavljivanja uz upotrebu argumenta iz zagrade.
- Argument ***N*** može biti vrednost ili jednačina. Najjednostavniji primer je samo broj u zagradi.

:nth-child(N)

Na primer:

ul li:nth-child(3) { color: red; }

Ovako bi u našoj listi **samo treći element** dobio crvenu boju teksta, samo treći *li* bi bio selektovan.

:nth-child(N)

Umesto broja, možemo pisati i ključne reči **even** ili **odd**, što prevedeno znači paran, odnosno neparan. U tom kontekstu, ukoliko napišemo:

```
ul li:nth-child(even)  { color: red; }
```

Svi neparni elementi liste će postati crveni.

Najkomplikovanija primena ove pseudo klase je upotreba nekog algebarskog izraza (jednačine) u zagradi. Na primer:

```
ul li:nth-child(2n+3)  { color: red; }
```

Upotrebom jednačine možemo dobiti veoma zanimljive rezultate. U primeru iznad postavili smo: 2n+3. Time bi svaki treći, peti, sedmi (i td.) element bio stilizovan. U tom kontekstu, varijabla ***n*** je bilo koji i svaki ceo broj. Kada kažemo “bilo koji i svaki” to znači da ***n*** istovremeno ima i vrednost 0, 1, 2, 3, 4… do beskonačnosti.

Pogledajmo tabelu sa nekim primerima:

n	2n	2n + 3	4n + 1	n + 4
0	(2 x 0) = 0	(2 x 0) + 3 = 3	(4 x 0) + 1 = 1	0 + 4 = 4
1	(2 x 1) = 2	(2 x 1) + 3 = 5	(4 x 1) + 1 = 5	1 + 4 = 5
2	(2 x 2) = 4	(2 x 2) + 3 = 7	(4 x 2) + 1 = 9	2 + 4 = 6
3	(2 x 3) = 6	(2 x 3) + 3 = 9	(4 x 3) + 1 = 13	3 + 4 = 7
4	(2 x 4) = 8	(2 x 4) + 3 = 11	(4x 4) + 1 = 17	4 + 4 = 8
…	…	…	…	…

Podrška i upotreba

- Iako je ova pseudo klasa veoma moćna sa raznolikom primenom, uostalom kao i druge u ovoj grupi, treba ih koristiti pažljivo, jer ih Internet Explorer tek od verzije 9 podržava i prepoznaje.
- Ukoliko želimo da napravimo tkzv *zebru*, koristeći ove klase, to neće biti problem čak i da se ne prikaže u starijoj verziji Explorera. Ipak, ukoliko postavimo pseudo klase nad osnovnim elementima layouta možda će se raspasti ceo sajta.

:nth-of-type(N)

Ova pseudo klasa je veoma slična prethodnoj i često se pogrešno poistovećuje sa njom. Pogledajmo na primeru:

```
<div id="main">
  <p>paragraph</p>
  <p>paragraph</p>
  <p>paragraph</p>
</div>
```

Ukoliko u CSS-u postavimo:

#main p:nth-child(2) { color: red;}

Ili ipak:

#main p:nth-of-type(2) { color: red;}

Dobijamo potpuno istu stvar. **Drugi paragraf će biti crven.**

Ali ukoliko CSS ostane isti, a promenimo HTML i postavimo ovako nešto:

```
<div id="main">
  <h2>heading</h2>
  <p>paragraph</p>
  <p>paragraph</p>
  <p>paragraph</p>
</div>
```

Sada će u varijanti sa ***#main p:nth-child(2)*** prvi paragraf, odmah ispod naslova biti crven

dok će u varijanti sa ***#main p:nth-of-type(2)*** drugi paragraf biti crven.

Otkud ta razlika, da li je to greška? Ne, upravo je tako zamišljeno da funkcionišu ove pseudo klase.

Svaka od njih proverava određene uslove, ali na nešto drugačiji način:

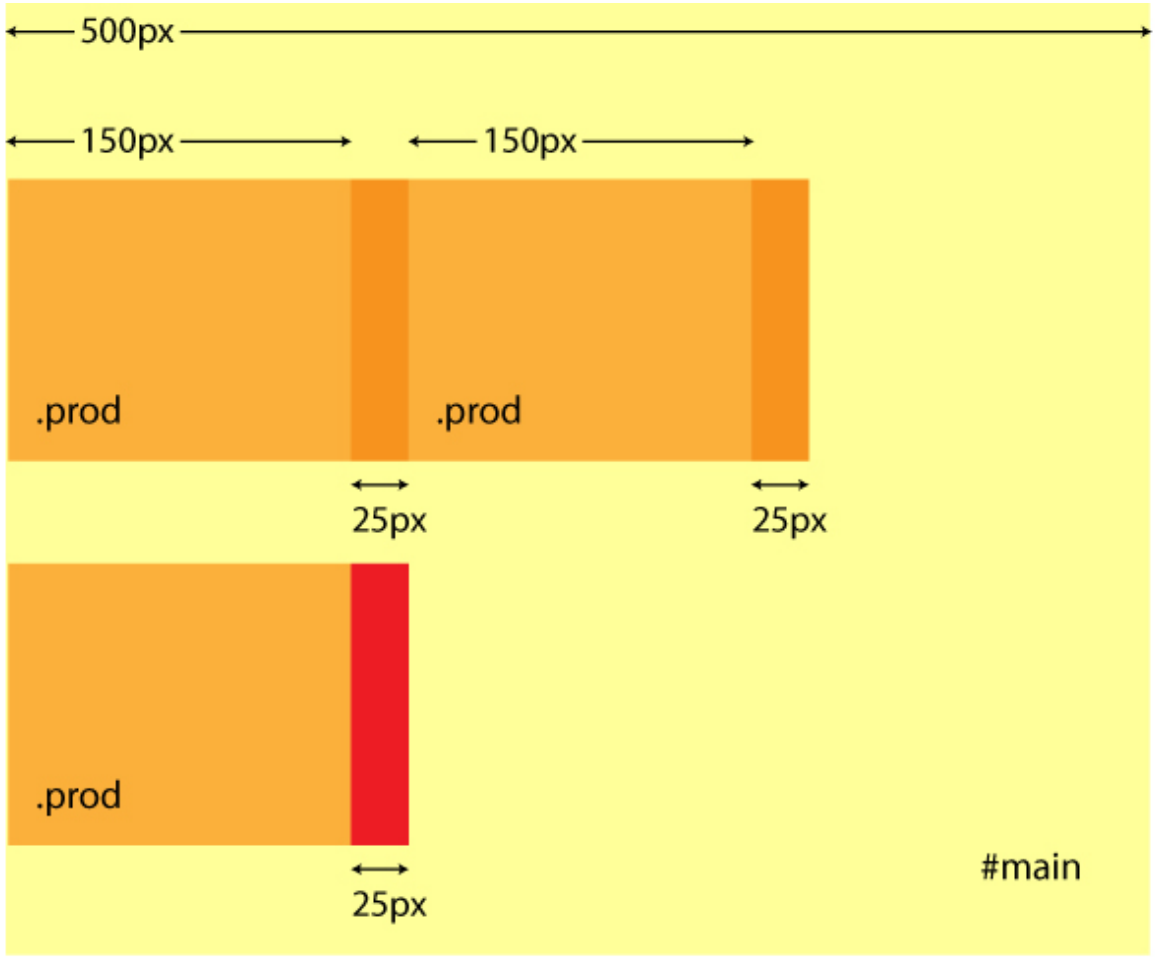
p:nth-child(2)	Da li je element tipa p?
	Da li je on drugi po redosledu u roditeljskom elementu?
p:nth-of-type(2)	Da li je drugi element tipa p u roditeljskom elementu?

U našim primerima:

- **Nth-child** traži element koji je drugi po redosledu u okviru #main i pritom proverava da li je taj element paragraf. Ta dva uslova (redni broj i tip elementa) **nisu u međusobnoj direktnoj** vezi. Moraju biti ispunjena, ali primeru iznad nije bitno to što prvi element nije paragraf, on traži drugi koji je pri tom paragraf.
- Sa druge strane, **nth-of-type** traži baš drugi paragraf u okviru #main. Proveravaće i tip svih prethodnih elemenata.
- Kao što je već pomenuto, ove klase su tako i koncipirane i imajte njihove razlike na umu prilikom primene. U principu, ako su isti tipovi elemenata u pitanju, onda je svejedno, ali čim ima varijacija tipa, ne dobijamo iste rezultate.

Primer upotrebe

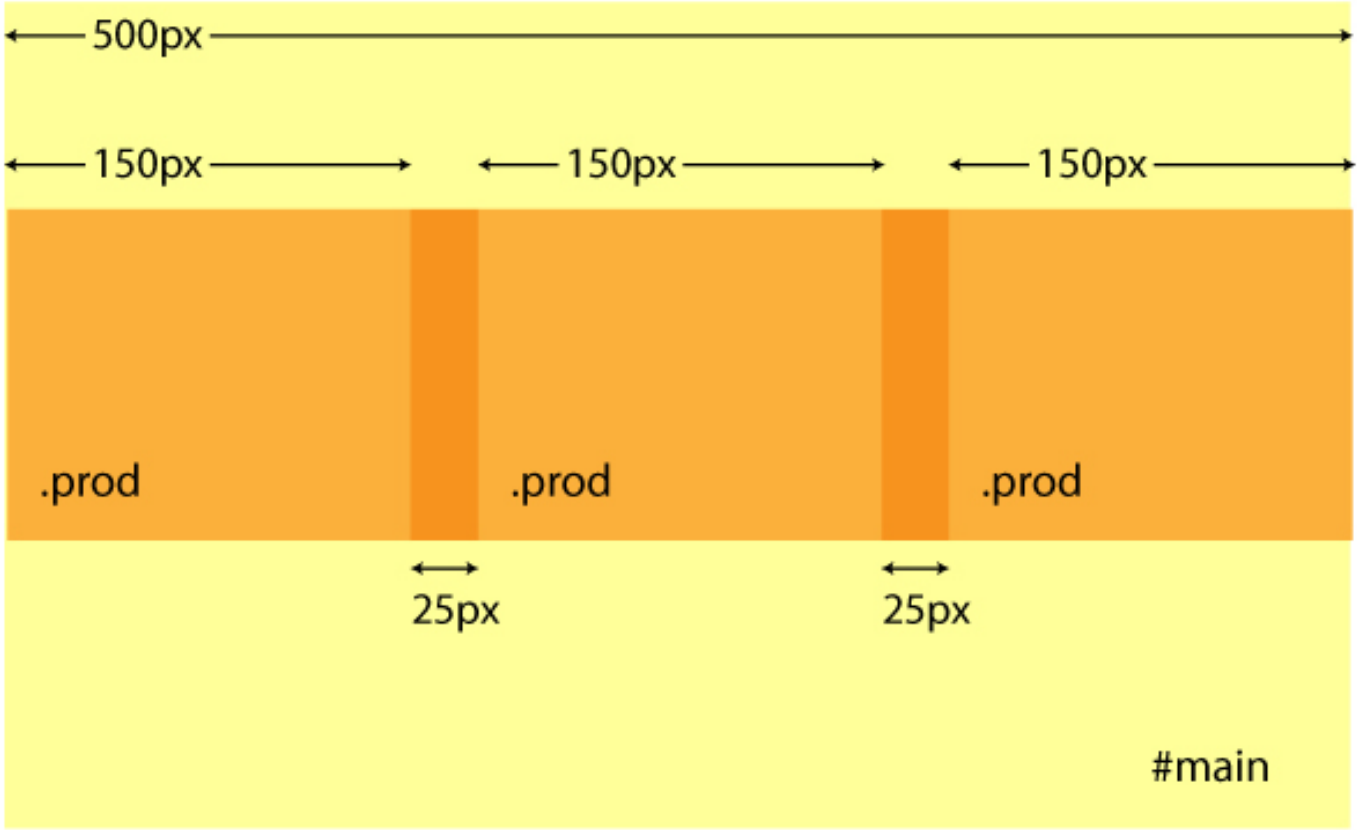
- Jedan od ova dva selektora možemo koristiti i u sledećoj situaciji.
  Ukoliko želimo da prikažemo neke div elemente po tri u jednom redu i ukoliko želimo prostor između njih, biće nam potrebna desna margina nad prvim i drugim elementom, dok nad trećim ne.
- Na primer, prikazujemo tri proizvoda koja su na sniženju u našoj web prodavnici. Postavićemo div elemente sa klasom *.prod* i preko nje dodeliti im širinu, float i sve što je potrebno.
- Ukoliko je dostupan prostor 500px širok, a širina svakog elementa je po 150px, onda nam ostaje dva puta po 25px prostora za između njih (150+25+150+25+150). Najlakše je da dodelimo desnu marginu od 25px, ali onda nastaje problem jer i poslednji, treći proizvod dobija tu marginu, koja je višak.
- Pogledajte na slici na sledećem slajdu. Margina koja je višak je obojena u crvenu boju. Ona pomera treći element u novi red jer nema dovoljno mesta za njega sa uključenom marginom.


500px
150px
150px
.prod
.prod
25px
25px
.prod
25px
#main

- Jedno rešenje je da smanjimo širinu svim elementima. To je ok, ali će ostati prazan prostor desno (ili levo), što nije dobro. Drugo rešenje je da dodamo posebnu, novu klasu trećem elementu i da njoj postavimo marginu na 0. Time se dobija ono što smo želeli.
- Ali tako smo postavili svim elementima po jednu klasu, i još, dodatno, trećem elementu novu, drugu klasu.
- Možemo primeti nešto drugačiji pristup i izbeći dodavanje nove klase i komplikovanje HTML koda.
- Možemo pomoću nth-child (ili nth-of-type) pogoditi samo treći element.

Kod bi mogao da izgleda:

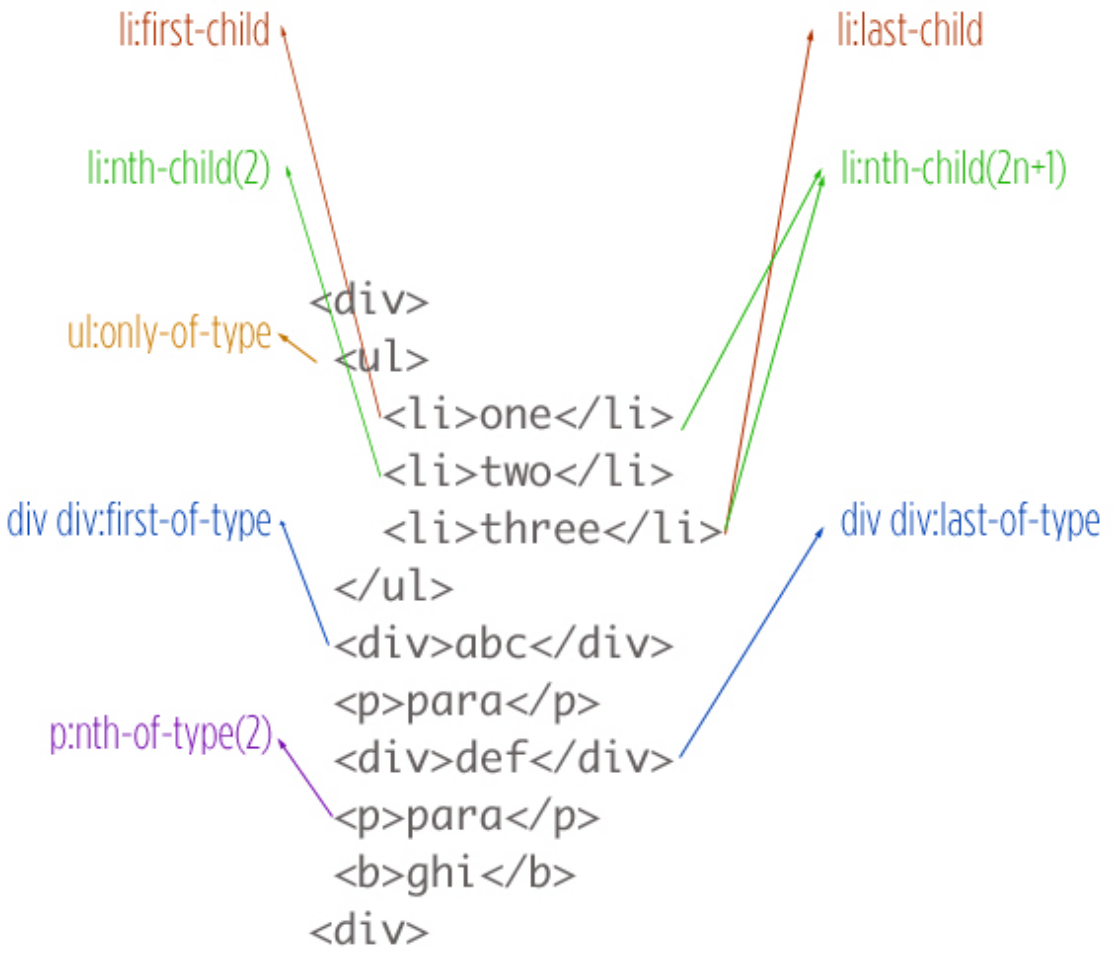
```
.prod {
  margin:0 25px 10px 0; /* gore desno dole levo */
}

.prod:nth-child(3n)  {
  margin-right:0;
}

/* Ovde u primeru su izostavljene float, width i ostale vrednosti */
```


500px
150px
150px
150px
.prod
.prod
.prod
25px
25px
#main

- Ukoliko postavimo više od tri elementa, ali po istom principu, ne moramo brinuti da li smo svakom trećem postavili dodatnu klasu, da li smo možda kasnije rotirali poziciju elemenata i slično. Ovaj primer će uvek biti ispravan, bez dodavanja novih klasa u html-u.

Još neki pseudo selektori

- **:nth-last-child(N)** - Funkcioniše isto kao :nth-child, samo što broji od kraja, od poslednjeg elementa.
- **:nth-last-of-type(N)** – Funkcioniše kao :nth-of-type, samo što broji od kraja, od poslednjeg elementa.
- **:only-of-type** – Selektuje element ukoliko je jedini tog tipa u roditeljskom elementu.
- **:first-of-type** – Slično kao :nth-of-type, ali uvek gađa samo prvi. Ne unosimo broj niti bilo šta slično.
- **:last-of-type** – Slično kao :first-of-type samo pogađa poslednji, umesto prvog.
- **:root** – Selektuje koreni (root) element stranice. U html dokumentima uvek je <html>. Preporuka je pisanje klasičnog opšteg *html* selektora umesto :*root* i izbegavanje ove klase.


li:first-child
li:last-child
li:nth-child(2)
li:nth-child(2n+1)
ul:only-of-type
div div:first-of-type
div div:last-of-type
p:nth-of-type(2)

one
two
three

abc
para
def
para
ghi

Postoji još pseudo class selektora koje nismo pomenuli, ali njihova realna upotreba je dosta ograničena.

I većina ovde navedenih se retko kad koristi.

Za više informacija možete pogledati sledeći link: https://developer.mozilla.org/en-US/docs/Web/CSS/Pseudo-classes