

În această lecție, ne vom ocupa de stilizarea CSS a unor elemente specifice, precum listele, tabelele și formularele. La final, vom vedea și ce sunt pseudoclasele.

## Stilizarea listelor

În ceea ce privește stilizarea listelor, putem diferenția stilul în care lista își păstrează propriile funcții de listare, dar putem să o stilizăm și își pierde caracteristicile originale. Acest lucru se întâmplă atunci când, de exemplu, folosim o listă ca bază pentru crearea unui meniu. Pentru moment, vom rămâne la domeniul listelor care listează.

### List-style-type

Utilizăm această proprietate asupra elementelor `ul` și `ol`, pentru a stiliza punctele (*bullets*) găsite în fața elementului. Acestea sunt diferite pentru listele ordonate și pentru cele neordonate.

Pentru listele **neordonate**, avem opțiunile:

- disc,
- circle,
- square.

În timp ce pentru listele **ordonate**, avem următoarele:

- decimal – 1 2 3,
- decimal-leading-zero – 01 02 03,
- lower-alpha – a b c,
- upper-alpha – A B C,
- lower-roman – i. ii. lii,
- upper-roman – I II III.

Codul CSS este destul de simplu:

```
ol {  
  list-style-type: decimal;  
}
```

Totuși, o astfel de descriere ar influența toate listele neordonate de pe site. Este mai bine să definim o dată elementele *ul* și *ol* ale listei, iar mai târziu să le schimbăm după necesitate, folosind clasa sau valorile ID:

```
Ol.listX {  
  list-style-type: decimal;  
}
```

## List-style-image

În locul elementelor standard, putem folosi imagini ca bullets. Deci, setăm proprietatea *list-style-image*, a cărei valoare este calea până la imagine, la fel ca și fundalul elementului.

```
ul {  
  list-style-image: url("images/star.png");  
}
```

În ceea ce privește listele, este important de știut că acestea au margini și padding în mod implicit și că toate setările pozițiilor, în special în momentul aplicării acestei proprietăți, se pot face cu margini și/sau cu valorile padding.

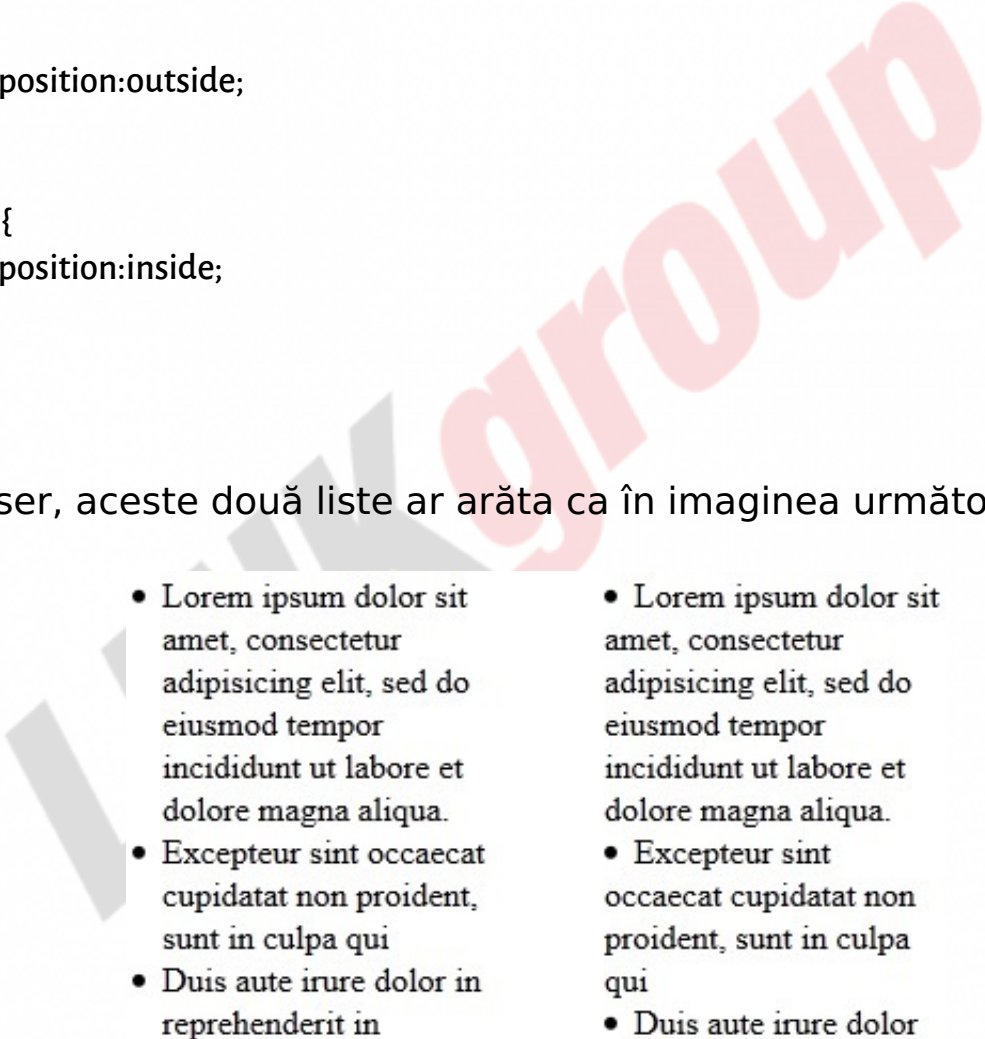
## List-style-position

Această proprietate stabilește dacă bullets se află în afara elementelor (în mod implicit) sau în cadrul lor. Prin urmare, valorile disponibile sunt *outside* și *inside*.

```
ul.first {  
  list-style-position: outside;  
}
```

```
ul.second {  
  list-style-position: inside;  
}
```

În browser, aceste două liste ar arăta ca în imaginea următoare:

- 
- |  |  |
|--|--|
| <ul style="list-style-type: none"><li>• Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.</li><li>• Excepteur sint occaecat cupidatat non proident, sunt in culpa qui</li><li>• Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.</li></ul> | <ul style="list-style-type: none"><li>• Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.</li><li>• Excepteur sint occaecat cupidatat non proident, sunt in culpa qui</li><li>• Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.</li></ul> |
|--|--|

*Imaginea 17.1 - Lista neordonată cu valoarea list-style-position la outside (stânga) și inside (dreapta)*

## List-style

Ca și în cazul altor proprietăți, și aici există o versiune prescurtată. Putem scrie tipul și poziția listei împreună cu ajutorul proprietății *list-style*.

```
ul {  
  list-style: inside upper-roman;  
}
```

## Stilizarea tabelelor

În ceea ce privește tabelele și stilizarea, în general se stilizează tag-urile td, respectiv celulele. Tag-urilor td le putem adăuga valorile CLASS sau le putem ținti cu regulile CSS prin intermediul tipului lor (td), dar acest lucru nu este recomandat.

Pe părțile tabelului, se pot aplica diferite proprietăți, precum proprietățile din modelul CSS box (width, height, padding, margin, border), dar și tipografice, precum font-family, letter-spacing și altele.

Iată un exemplu cu mai multe detalii. În secțiunea HTML, avem:

```
<tr>  
  <th>name</th>  
  <th>surname</th>  
  <th class="record">ID</th>  
</tr>  
<tr class="even">  
  <td>Tricia</td>  
  <td>McMillan</td>  
  <td class="record">332</td>  
</tr>  
<tr>  
  <td>Arthur</td>  
  <td>Dent</td>  
  <td class="record">1234</td>  
</tr>
```

```
<tr class="even">
  <td>Ford</td>
  <td>Prefect</td>
  <td class="record">9876</td>
</tr>
```

În CSS, avem:

```
body {
  font-family: Georgia, "Times New Roman", Times, serif;
}
```

```
th, td {
  padding: 7px 10px 10px 10px;
}
```

```
th {
  text-transform: uppercase;
  letter-spacing: 0.2em;
  font-size: 90%;
  border-bottom: 2px solid #111111;
  border-top: 1px solid #999;
  text-align: left;
}
```

```
tr.even {
  background-color: #6FC;
}
```

```
tr:hover {
  background-color: #fff;
}
```

```
.record {
  text-align: right;
}
```

Iar în browser se afișează următoarele:

NAME	SURNAME	ID
Tricia	McMillan	332
Arthur	Dent	1234
Ford	Prefect	9876

*Imaginea 17.2 - Un tabel stilizat*

După cum putem vedea din acest exemplu, am combinat diferite reguli CSS și am adăugat anumite atribute CLASS unor elemente diferite, pentru a le ținti cu regulile CSS.

Dacă o luăm de la început, vedem că alegerea fontului este definită pe întreaga pagină (body). Apoi, am setat spații (padding) pe celule. Am definit aspectul titlurilor și, la final, am adăugat o zebră (rândurile even și odd) și am aliniat ultima coloană.

## Stilizarea formularelor

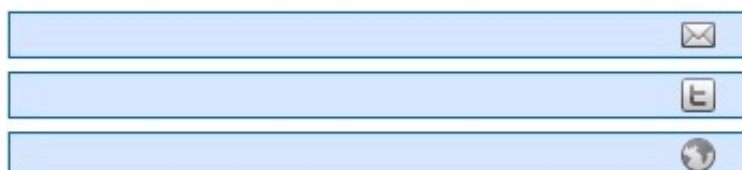
Stilizarea formularelor este foarte asemănătoare cu stilizarea tabelelor, dar poate fi mai complicată. În primul rând, din cauza faptului că majoritatea controalelor folosesc obiecte *input*, așadar nu putem folosi selectori generali. Să luăm această parte a formularului:

```
<input type="text" id="email">  
<input type="text" id="twitter">  
<input type="text" id="web">
```

Dacă o stilizăm cu aceste reguli CSS:

```
input {  
  width:300px;  
  color: #006;  
  background-color: #D6E6FE;  
  border: 1px solid #06C;  
  padding: 2px 45px 2px 2px;  
  background-repeat: no-repeat;  
  background-position: 95% center;  
  display: block;  
  margin-bottom: 6px;  
}  
  
input#email {  
  background-image: url("images/email.png");  
}  
  
input#twitter {  
  background-image: url("images/twitter.png");  
}  
  
input#web {  
  background-image: url("images/web.png");  
}
```

vom obține rezultatul următor:



The image shows three horizontal input fields stacked vertically. Each field is light blue with a thin dark blue border. The top field has an envelope icon on the right, the middle one has a Twitter bird icon, and the bottom one has a globe icon. The fields are separated by small gaps.



### *Imaginea 17.3 - Formular stilizat (câmpuri input)*

Este important de remarcat că am folosit o singură descriere pentru toate input-urile, dar am adăugat o descriere separată pentru detaliile care se deosebesc. Problema acestui exemplu este că prima descriere a influențat toate tag-urile input, inclusiv butonul submit, radio și opțiunile checkbox etc. O soluție poate fi adăugarea aceleiași valori CLASS tuturor câmpurilor textuale și corectarea primei descrieri. Există și o a doua variantă - selectorului îi putem adăuga un tip de câmp. De exemplu:

```
input[type=text] {...}
```

Type poate fi și altceva. Trebuie să corespundă atributului type al tag-ului input. Trebuie doar să fim mai atenți cu browser-ele mai vechi, deoarece nu toate suportă acest mod de selectare a elementelor.

## **Pseudoclasele**

Pe lângă modul tradițional de selectare a elementelor, în anumite situații putem selecta, ținti, stările lor cu ajutorul pseudoclaselor. Pe acestea le setăm scriind semnul două puncte (:) după selector sau după o anumită parte a selectorului, iar apoi adăugând cuvântul-cheie.

De exemplu, hyperlinkurile pot avea starea link, hover, visited sau active la un anumit moment. Link este starea normală a linkului, *hover* este starea care se activează când ținem mouse-ul deasupra linkului, *visited* este starea linkurilor vizitate anterior, în timp ce starea *active* se instaurează când dăm clic pe link. De exemplu:

```
a,  
a:link {  
  color: #cc0000;  
  text-decoration:none;
```



```
}  
  
a:visited {  
  color: #990000;  
  text-decoration: none;  
}  
  
a:hover {  
  color: #3333cc;  
  text-decoration: underline;  
}  
  
a:focus {  
  outline: thin dotted;  
}
```

În acest exemplu, vedem că prima regulă CSS a fost setată pentru toate linkurile (*a selector*) și pentru linkurile în stare normală (*a:link*). Am definit culoarea textului și am înlăturat underline (linia de sub link). Poate vă întrebați de ce am scris doi selectori. Nu este suficient să scriem doar *a:link*? În principiu este suficient, dar pot apărea probleme dacă cineva nu adaugă atributul href corect în tag. În acest caz, starea *:link* nu va funcționa.

În continuare, vedem selectorul *a:visited* în a doua descriere, care definește linkurile vizitate, marcându-le printr-o culoare puțin mai închisă. Starea *a:hover* schimbă culoarea linkurilor în albastru, pentru ca utilizatorul să primească feedback că se află deasupra unui link. Putem adăuga și starea *a:active*, pe care nu am folosit-o aici.

Folosind stările speciale, practic prin CSS putem să tratăm un element în diferite moduri și să distanțăm descrierile.

În majoritatea cazurilor, practic întotdeauna, pseudoclaselor se folosesc

pentru linkuri.

## Border-radius

Cea mai des menționată proprietate CSS3 este *border-radius*. O utilizăm pentru a defini curbura unghiului pentru marginea setată. Să privim codul:

```
.box {  
  border:1px solid #333;  
  border-radius:10px;  
}
```

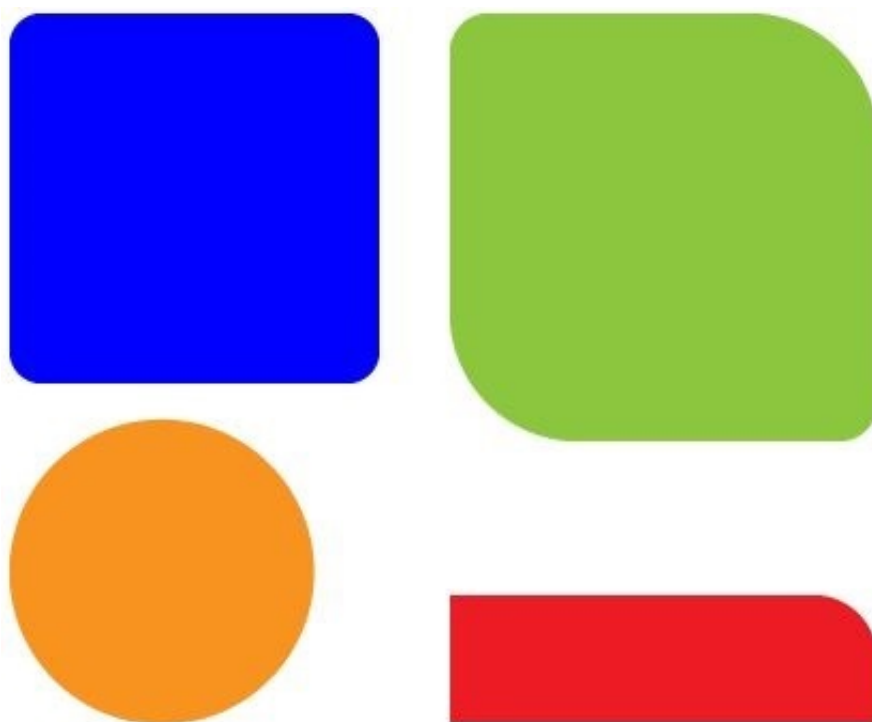
În acest fel, pe un element cu clasa *box* am inserat o margine gri simplă (*border*) cu grosimea de 1 pixel, cu colțurile rotunjite la un unghi de 10 pixeli.

În acest exemplu, se poate vedea foarte bine cum vechea specificație CSS *border* colaborează cu cea nouă, *border-radius*.

Pe lângă curbura egală, putem defini și valori speciale pentru toate cele patru unghiuri. Introducem patru valori pentru unghiurile din stânga-sus, dreapta-sus, dreapta-jos, stânga-jos:

```
border-radius:10px 0 20px 0;
```

Un cod introdus astfel va rotunji unghiul din stânga-sus și pe cel din dreapta-jos.



*Imaginea 17.4 - Exemple de aplicații border-radius afișate în browser*

## Box-shadow

Această proprietate definește umbra pe care un anumit element o aruncă sub el. Solicită mai mulți parametri. În cod putem scrie, de exemplu:

```
.box {  
  box-shadow: 10px 20px 5px 0px #000000;  
}
```

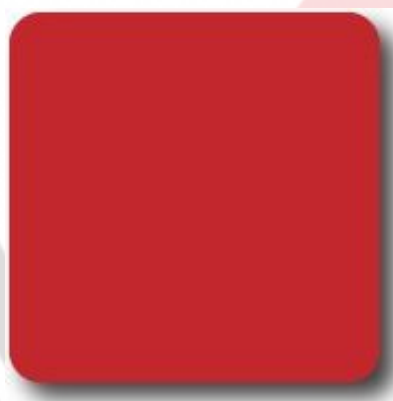
Acest exemplu pune o umbră pe element - o umbră mutată cu 10 px în partea dreaptă față de element, cu 20 px în partea de jos a elementului, cu valoarea *blur* (estompare) de 5 px, valoarea *spread* (care extinde umbra) de 0 px și, la final, culoarea umbrei în sine. În

acest exemplu, am folosit *hex*, dar putem utiliza și *RGBA*, de exemplu.

Deci, proprietatea *box-shadow* necesită următoarele valori:

`box-shadow: hor-position vert-position blur spread color`

Dacă vrem ca o umbră să cadă în interiorul elementului, putem introduce și cuvântul-cheie *inset*.



*Imaginea 17.5 - Exemplu de drop shadow (umbră lăsată) afișată în browser*