

## Tema: Noțiunea de "responsive design". Media-queries

După această lecție:



- vei cunoaște elemente HTML ce au apărut în versiunea 5, utilizate pentru machetarea paginilor web;
- vei afla ce presupune noțiunea de "responsive web design" și ce proprietăți de stil ar trebui să definești ca să creezi conținuturi web sensibile la ecranul de prezentare a conținuturilor web;
- vei afla ce este "viewport-ul" și ce sunt punctele de control;
- vei afla ce sunt interogările media sau "media queries" și pentru ce este necesar ca să fie definite ele.

La această lecție vom învăța cum poate fi realizat procesul de machetare și câteva proprietăți CSS, ce pot fi utilizate pentru a realiza conținuturi web, "sensibile" la ecranul de prezentare a acestor pagini. Cred ca ai văzut site-uri cu pagini, care arată bine și pe ecrane mari de televizor, și pe ecrane de laptop, dar și pe ecrane de telefoane inteligente. Dar mai există încă site-uri în mediul web, care pe ecrane de telefon smart, nu pot fi accesate comod de utilizator.

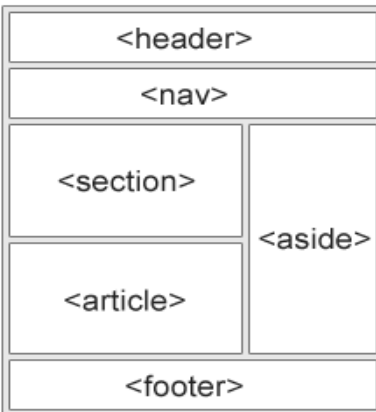
Poți citi mai multe informații, dar și vedea exemple de site-uri, ce țin cont de "experiența utilizatorului", la acest link: <https://www.invisionapp.com/inside-design/examples-responsive-web-design/>. Dar, vom începe cu definirea noțiunii de "machetare"...

### Machetarea paginilor site-ului web

**Machetarea** este procesul de transformare a desenului unei pagini web (de exemplu un fisier *.psd* (*PhotoShop Document*)) într-un document HTML, utilizând HTML și CSS. Acest document, interpretat de browser, va prezenta în fereastră un conținut asemănător desenului. Să știi, că o pagină cu același conținut poate fi machetată foarte diferit, schimbând doar stilurile.

Consortiul **w3** recomandă utilizarea **div**-urilor pentru machetare și utilizarea corespunzătoare a proprietăților CSS de poziționare. Trebuie să mai știi că în HTML5 au apărut elemente speciale, ce pot fi utilizate la machetarea site-ului, doar împreună cu CSS. Descrierea lor succintă este prezentată în următorul tabel:

Denumirea elementului	Descrierea	Imagine de ansamblu
-----------------------	------------	---------------------

<b>header</b>	Definește antetul unui document sau secțiuni	
<b>nav</b>	Definește un container pentru link-uri	
<b>section</b>	Definește o secțiune a documentului	
<b>article</b>	Definește un articol independent	
<b>aside</b>	Definește un conținut alături de conținutul de bază, precum o bară laterală	
<b>footer</b>	Definește subsolul unui document sau pentru o secțiune	

Suplimentar, vedeți și alte elemente apărute în HTML5, la acest link: [http://www.w3schools.com/html/html5\\_new\\_elements.asp](http://www.w3schools.com/html/html5_new_elements.asp).

Voi prelua exemplul de pagină web, realizată în cadrul lecției 16, și voi macheta pagina, astfel încât imaginile să fie pe partea stângă a paginii, iar textul pe partea dreaptă. Voi utiliza elemente DIV, iar apoi voi redefini pagina cu elementele apărute în HTML5 și proprietățile CSS. Pentru aranjarea în pagină a elementelor voi utiliza proprietatea "float".



Codurile HTML:



```
<body>
  <div class="menu">
    <span>Coffee time</span>
    <a href="#unu">Despre noi</a>
    <a href="#doi">Produse</a>
```

```

        <a href="#trei">Contacte</a>
    </div>
    <div class="content">
        <div class="images">
            
            ...
            
        </div>
        <div class="text">
            <h1>Cafeaua - băutură energizantă...</h1>
            <p>Deoarece cafeaua are un conținut ridicat ... etc.</p>
        </div>
    </div>
    <div class="footer"></div>
</body>

```

Stilurile scrise:



```

html{
    margin:0;
    padding:0;
}
body{
    font: 1rem/2rem Georgia, serif;
    color: white;
    background-color: rgba(77, 46, 51, 0.8);
    margin: 0;
    padding: 0;
}
*{
    box-sizing: border-box;
}
a:link {
    color: rgb(199, 240, 247);
    text-decoration: none;
    padding:10px;
    margin:5px;
    font-size: 1.3em;
}
a:visited { color: rgb(253, 228, 245);}
a:hover {
    color: whitesmoke;
    text-decoration: underline;
    padding: 10px;
    margin:5px;
}
a:active { color: #cacadf;}
.menu, .content, .footer {
    margin:0 auto;
}

```

```

padding: 10px 0;
margin: 0 auto;
text-align: center;
width: 100%;
}
.menu, .footer {
background-color: rgb(77, 46, 70);
height: 10%;
}
.menu span{
text-transform: uppercase;
color: rgba(214, 248, 21, 0.8);
font-size: 1.5em;
margin-right: 20px;
}
.content img {
max-width: 22%;
height: auto;
border: 3px dotted rgb(247, 238, 239);
padding: 3px;
margin: 5px;
}
.text span {
color: rgb(243, 250, 140);
}
.text p{
padding: 15px;
width:100%;
font-size: 1.1em;
color: whitesmoke;
background-color: rgba(77, 46, 51, 0.7);
margin:0;
}
.menu{
position:fixed;
top:0;
left:0;
margin:0;
padding:15px;
}
.content{
margin-top: 60px;
}
.content .images{
width: 70%;
float: left;
}
.content .text{
width: 28%;
margin-top: 10px;

```

```

margin-right: 0;
background-image: url("../images/background.jpg");
background-position: 50% 50%;
background-size: cover;
border: 3px dotted rgb(247, 238, 239);
float: right;
}
h1 {
color: rgb(247, 238, 239);
}
.footer{
background-image: url("../images/logo.png");
background-repeat: repeat-x;
background-size: 5%;
clear: both;
}

```

În continuare voi lăsa aceleași proprietăți de stil, dar voi schimba elementele HTML - **DIV**, cu cele noi apărute:



```

<body>
  <header class="menu">
    <span>Coffee time</span>
    <a href="#unu">Despre noi</a>
    <a href="#doi">Produse</a>
    <a href="#trei">Contacte</a>
  </header>
  <section class="content">
    <article class="images">
      
      ...
      
    </article>
    <aside class="text">
      <h1>Cafeaua - băutură energizantă...</h1>
      <p>Deoarece cafeaua are un conținut ... etc.</p>
    </aside>
  </section>
  <footer class="footer"></footer>
</body>

```

Rezultatul interpretării acestor coduri, cu aplicarea aceluiași stiluri va fi:



Ai observat vreo diferență? Nici nu este.

### Necesitatea creării conținuturilor sensibile la dispozitivele de prezentare

Presupun că ai observat deja, că majoritatea din noi accesează diferite site-uri, magazine electronice etc., cu un telefon mobil. Ponderea utilizării dispozitivelor mobile, utilizate pentru navigare în Internet este în creștere, de la an la an, dar, din păcate, o mare parte din produsele web existente, încă nu sunt adaptabile la toate ecranele, precum cele ale: telefoanelor, "phablets", tabletelor, desktop-urilor, consolelor de jocuri, televizoarelor smart etc.

Dimensiunile dispozitivelor de afișare vor fi mereu în schimbare, deci este important ca un site să se poată adapta la orice dimensiune a ecranului, dispozitivului de prezentare.

### HTML Responsive Web Design

Un website este "*responsive*" (noțiune preluată din limba engleză) sau *sensibil*, dacă are capacitatea să-și adapteze conținutul paginilor **în funcție de mediul de vizionare** și de rezoluția ecranului, fără a suferi schimbări mari, pentru a le oferi utilizatorilor o **experiență bună de navigare**.



### Ce presupune crearea unui design responsive?

Dacă la început web developerii defineau lăţimea paginilor ca o mărime fixată, fiindcă ecranele calculatoarelor erau identice la toţi utilizatorii, în ultimii ani **gridurile fluide** au luat locul celor fixe, datorită multitudinii de rezoluţii ale ecranelor, existente în prezent. Gridurile fluide se bazează pe calculul proporţiilor şi îşi pot modifica lăţimea şi înălţimea vizibilă, în funcţie de rezoluţia ecranului. Dimensiunile nu se mai măsoară în pixeli, ci în procente şi alte unităţi relative.

**Imaginile flexibile**, de asemenea îşi pot modifica dimensiunile în funcţie de rezoluţia dispozitivului şi dimensiunea gridului.

Pentru a realiza griduri fluide, în CSS pot fi utilizate mai multe proprietăţi.

### Setarea spaţiului vizibil pentru dispozitiv

Vom începe cu menţiunea că, obligatoriu trebuie configurat spaţiul de vizualizare al dispozitivului. Documentele web, numaidecât **trebuie** să conţină în containerul **head** meta-tag-ul cu numele **viewport**. "Fereastra de vizibilitate" sau **VIEWPORT**-ul, utilizatorului, se schimbă în funcţie de dispozitivul utilizat şi va fi mult mai mic pe un telefon, decât pe un ecran de calculator. Viewport-ul anunţă browserul cum trebuie acesta să controleze dimensiunile şi scara paginii web. Deci, verifică să ai în elementul HEAD al documentului web, linia:

```
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
```

unde valoarea **initial-scale=1**, este specificată pentru a asigura scara de 1:1 între pixelii CSS şi pixelii independenţi ai dispozitivului.



## Definirea gridurilor fluide

Trebuie să știi, că orice pagină web poate fi divizată, imaginar, în 12 coloane verticale, care corespund lățimii întregii pagini, adică 100%. Unei coloane, din cele 12, îi revine aproximativ 8.33% și poți verifica:  $8.33 \times 12 = 99.96$ , ceea ce este aproximativ 100%.

Voi defini și eu următoarele clase și stiluri, necesare definirii unei grile fluide pentru pagina web:



```
.col-1 {width: 8.33%;}
.col-2 {width: 16.66%;}
.col-3 {width: 25%;}
.col-4 {width: 33.33%;}
.col-5 {width: 41.66%;}
.col-6 {width: 50%;}
.col-7 {width: 58.33%;}
.col-8 {width: 66.66%;}
.col-9 {width: 75%;}
.col-10 {width: 83.33%;}
.col-11 {width: 91.66%;}
.col-12 {width: 100%;}
```

## Media query

**Interogările media** (în limba engleză **media queries**) reprezintă un procedeu eficient de a încărca diferite proprietăți CSS, în funcție de rezoluția dispozitivului. Site-ul recunoaște automat tipul dispozitivului și rezoluția acestuia, încărcând proprietățile CSS corespunzătoare, cu scopul redimensionării, ascunderii, măririi sau mutării conținutului, astfel încât acest conținut să arate bine pe orice tip de ecran.

Sintaxa recomandată la definirea *media Query* este:

```
@media (query) {
    /* proprietăți CSS */
}
```

Voi adăuga clasele, definite mai sus, corespunzătoare elementelor, care ar trebui să fie sensibile, la schimbarea dimensiunilor ferestrei de vizualizare:



```
<section class="content">
    <article class="images col-9">
```



```













</article>
<aside class="text col-3">
  <h1>Cafeaua - băutură energizantă...</h1>
  <p>Deoarece cafeaua are un conținut ridicat ... etc.</p>
  <p>Istoria cafelei datează de mai mult de 1000 ... </p>
</aside>
</section>

```

Observă că **"col-9"** va corespunde la 9 coloane din grila paginii, iar **"col-3"** – va corespunde la 3 coloane.

Astfel, voi aplica principiul *"mobile first"* și voi scrie următorul stil:



```

[class*="col-"] {
  width: 100%;
}

```

Adică, toate elementele ce conțin clase ce au în denumire **"col-"**, pe un ecran mic vor ocupa 100% din lățimea ecranului. Am utilizat mai sus un *selector-condițional*, prezentând condiția între paranteze pătrate.

Atunci când se definesc stilurile și utilizez și *media Query*, se recomandă definirea **punctelor de control**. Ele reprezintă acele puncte, de la care, atunci când se încearcă micșorarea sau mărirea ferestrei browserului, design-ul paginii devine mai prost. Eu am ales în calitate de punct de control punctul 770px. Pot fi definite stiluri și pentru mai multe puncte de control.

Astfel voi scrie următorul mediaQuery:



```
@media only screen and (min-width: 770px) {
  .col-1 {width: 8.33%;}
  .col-2 {width: 16.66%;}
  .col-3 {width: 25%;}
  .col-4 {width: 33.33%;}
  .col-5 {width: 41.66%;}
  .col-6 {width: 50%;}
  .col-7 {width: 58.33%;}
  .col-8 {width: 66.66%;}
  .col-9 {width: 75%;}
  .col-10 {width: 83.33%;}
  .col-11 {width: 91.66%;}
  .col-12 {width: 100%;}
}
```

unde, proprietatea **min-width** – va fi aplicată în browser, lăţimea ferestrei căruia este mai mare decât valoarea specificată în interogare. Aceste stiluri se vor aplica atunci când fereastra browserului atunci când lăţimea devine mai mare decât 770px.

Rezultatul aplicării acestor stiluri, va fi prezentarea diferită ale elementelor pe ecrane de diferite dimensiuni. Pe ecran mic pagina va fi prezentată astfel:



Observă cum blocul cu textul migrează sub blocul cu imagini, și ambele au lățimea de 100%, pe ecranul mic. De asemenea, vezi cum imaginile își schimbă dimensiunile, iar textul este citibil.

Pe ecran mare pagina va arăta astfel:



Observă că în acest caz, că blocurile sunt prezentate alături – în stânga cel cu imagini și care ocupă  $\frac{3}{4}$  din lățimea ecranului (deoarece are specificată clasa **col-9**), ceea ce corespunde la 75%, iar în dreapta – blocul cu textul, care ocupă doar  $\frac{1}{4}$  din lățimea ecranului (deoarece are specificată clasa **col-3**). Vezi și filmulețul **responsive.mp4**.

Să recapitulăm cele învățate în cadrul acestei lecții:



- în HTML5 au apărut câteva elemente noi, care se recomandă a fi utilizate la machetarea paginilor web, elemente precum: NAV, HEADER, SECTION, ASIDE etc.;
- designul responsive al paginilor web este utilizat pentru a prezenta într-o formă accesibilă și comodă utilizatorului, același conținuturi pe diferite ecrane ale dispozitivelor: telefoane, tablete, ecrane de PC etc.;
- pentru a realiza corect conținuturi sensibile la ecranul de prezentare, se recomandă definirea corectă a viewportului, găsirea punctelor de control și definirea corectă a interogărilor media.

Încheiem lecția aici. Data viitoare vom analiza și alte metode de definire a gridurilor fluide pentru pagini.