Stiluri CSS3

CSS3 permite să includă și alte elemente de design într-un mod revoluționar care să aducă multe beneficii: cod organizat care să fie accesibil pentru oameni sau chiar pentru alte mașini, cod usor de întreținut, mai puține imagini și încărcare mai rapidă a paginilor web.

Deși pare că există o mare diferență între CSS2 și CSS3 toate browserele importante au deja support pentru cele mai multe caracteristici CSS3, care sunt disponibile în prezent. Ca și în HTML5, W3C consideră că CCS3 este în continuă dezvoltare și este foarte puțin probabil să obțină rapid o versiune finală.

Poate că cea mai mare diferență dintre CSS2 și CSS3 este separarea în module. În timp ce în versiunea anterioară totul a fost specificat în linii mari prin definirea unor caracteristici diferite, CSS3 este împărțit în mai multe documente care sunt numite module. Fiecare modul are noi capabilități fără să afecteze compatibilitatea cu versiunea anterioara.

Suportul pentru mass-media poate fi cel mai important plus față de CSS. Ceea ce face este simplu: permite anumite condiții astfel încât să fie aplicate diferite stiluri CSS care sa facă pagina web să se potrivească pentru toate tipurile de dimensiuni de ecran. Suportul mass-media permite dezvoltatorilor să își adapteze pagina la diferite rezoluții fără a modifica sau elimina conținut sau cod HTML.

<!— HTML5 – Link -->

<link rel="stylesheet" href="../style/stylefile.css">

Pentru XML: <?xml-stylesheet href=”exemplu.css” type=”text/css”?>

Noile caracteristici în CSS3 includ suport pentru selectori suplimentari, umbre, colturi rotunjite, fundaluri multiple, animație, transparență și multe altele. CSS3 este diferit de HTML5.

Înainte de CCS3 fiecare browser avea câte un prefix care se adăuga la unele elemente CSS pentru a ajuta dezvoltatorul să se asigure de uniformitatea stilului în pagină:

- moz-: Firefox
- -WebKit-: Safari și Chrome
- -o-: Opera
- -ms-: Internet Explorer

Cu noul CSS3 aceste prefixe o să dispară.

Elemente de chenar

Adăugarea de colțuri rotunjite în CSS2 a fost dificilă. Era nevoie de a se folosi imagini diferite pentru fiecare colț. În CSS3 crearea de colțuri rotunjite se face foarte ușor pentru că a fost introdusă proprietatea **"border-radius":**

```
div
{
border:2px solid;
border-radius:25px;
}
```

Pentru a încadra un element cu o imagine a fost introdusa proprietatea **"border-image":**

```
div
{
/* Safari 5 */
-webkit-border-image:url(border.png) 30 30 round;
-o-border-image:url(border.png) 30 30 round;
/* Opera 10.5-12.1 */
border-image:url(border.png) 30 30 round;
}
```

Pentru crearea efectului de umbră a fost introdusă proprietatea **"box-shadow":**

```
div
{box-shadow: 10px 10px 5px #ccccccc;
}
```

Elemente de culoare, fundal și gradient

CSS3 conține mai multe proprietăți de fundal noi care permit un control mai mare al elementului de fundal. Proprietatea "background-size" specifică dimensiunea imaginii de fundal. Înainte de CSS3 dimensiunea imaginii de fundal era determinată de dimensiunea reală a imaginii. În CSS3 este posibil să se specifice dimensiunea imaginii de fundal ceea ce ne permite să se reutilizăm imaginea de fundal și în alte contexte. Se poate specifica dimensiunea în pixeli sau în procente. **Dacă se specifică dimensiunea în procente dimensiunea este relativă la lățimea și înălțimea elementului părinte.**

```
div
{
background:url(img.jpg);
background-size:100% 100%;
background-repeat:no-repeat;
}
```

Proprietatea "**background-origin**" specifică zona de poziționare a imaginilor de fundal. Imaginea de fundal poate fi plasată în proprietățile **"content-box", "padding-box", sau "border-box".**

```
div
{
background:url(img.jpg);
background-repeat:no-repeat;
background-size:100% 100%;
background-origin:content-box;
}
```

Mai mult decât atât, se pot poziționa mai multe imagini de fundal la un element.

```
body
{
background:url(img1.jpg),url(img2.jpg);
background-size:100% 100%;
background-repeat:no-repeat;
}
```

Cu noul CSS3 se pot aplica gradiente ca fundal în loc de imagini. Gradientele sunt tranziții line între două sau mai multe culori. Anterior trebuia să utilizam imagini de fundal pentru aceste efecte. Folosind gradienți în CSS3 putem reduce timpul de descărcare și utilizarea lățimii de bandă. În plus, elementele cu gradienți arată mai bine atunci când sunt mărite deoarece gradientul este generat de către browser. În cazul gradientului linear trebuiesc definite cel puțin două culori unde ultima culoare reprezintă culoarea către care vrem să facem tranziția. De altfel, trebuie să stabilim un punct de pornire și o direcție sau un unghi.

```
#grad
{
background: -webkit-linear-gradient(180deg, red, blue); /* For Safari 5.1 to 6.0 */
background: -o-linear-gradient(180deg, red, blue); /* For Opera 11.1 to 12.0 */
background: -moz-linear-gradient(180deg, red, blue); /* For Firefox 3.6 to 15 */
background: linear-gradient(180deg, red, blue); /* Standard syntax */
}
```

Se pot defini desigur și gradienți de mai multe culori:

```
#grad
{
/* For Safari 5.1 to 6.0 */
background: -webkit-linear-
gradient(left,red,orange,yellow,green,blue,indigo,violet);


/* For Opera 11.1 to 12.0 */
background:
 -o-linear-gradient(left,red,orange,yellow,green,blue,indigo,violet);
/* For Fx 3.6 to 15 */
background: -moz-linear-
gradient(left,red,orange,yellow,green,blue,indigo,violet);
/* Standard syntax */
background: linear-gradient(to right,
red,orange,yellow,green,blue,indigo,violet);
}
```

Gradientele CCS3 suportă și transparența pentru un efect special. Pentru setarea gradului de transparență se adaugă un număr între 0 și 1 la parametrul rgb(), unde 1 reprezintă lipsa transparenței.

```
#grad
{
background: -webkit-linear-gradient(left,rgba(255,0,0,0),rgba(255,0,0,1));
/*Safari 5.1-6*/
background: -o-linear-gradient(right,rgba(255,0,0,0),rgba(255,0,0,1)); /*Opera
11.1-12*/
background: -moz-linear-gradient(right,rgba(255,0,0,0),rgba(255,0,0,1)); /*Fx
3.6-15*/
```

```
background: linear-gradient(to right, rgba(255,0,0,0), rgba(255,0,0,1)); /*Standard*/
}
```

Gradientul dorit se poate repeta folosind proprietatea “repeating-linear-gradient”.

```
#grad
{
/* Safari 5.1 to 6.0 */
background: -webkit-repeating-linear-gradient(red, yellow 10%, green 20%);
/* Opera 11.1 to 12.0 */
background: -o-repeating-linear-gradient(red, yellow 10%, green 20%);
/* Firefox 3.6 to 15 */
background: -moz-repeating-linear-gradient(red, yellow 10%, green 20%);
/* Standard syntax */
background: repeating-linear-gradient(red, yellow 10%, green 20%);
}
```

Pe lângă gradientul linear se poate desena și un gradient circular (radial) folosind radial-gradient().

```
#grad
{
background: -webkit-radial-gradient(red, green, blue);
/* Safari 5.1 to 6.0 */
background: -o-radial-gradient(red, green, blue); /* For Opera 11.6 to 12.0 */
```

```
background: -moz-radial-gradient(red, green, blue); /* For Firefox 3.6 to 15 */
background: radial-gradient(red, green, blue); /* Standard syntax */
}
```

Efecte text

CSS3 elimină în cele din urmă necesitatea de prelucrare în Photoshop a unui text atunci când doriți să îi faceți o simplă umbră. Proprietatea "text-shadow" este exemplificată după cum urmează:

```
text-shadow: 2px 2px 2px #000000;
```

Acest lucru produce un efect de umbră în dreapta și sub text cu dimensiunea de 2px și adaugă un efect de blur (estompare) tot de 2 px.

Există însă o proprietate în CSS3 care o sa fie utilizat în fiecare design. Este unul foarte util atunci când avem nevoie de el pentru că este mai mult decât practic. Proprietatea despre care vorbesc este "word-wrap" și chiar dacă pare imposibil, funcționează în fiecare browser inclusiv în toate versiunile de IE. De fapt, a fost suportat chiar încă de la IE5.

Cu toate că s-ar putea asocia în mod normal CSS3 cu browser-e moderne cum ar fi Safari și Chrome, trebuie remarcat faptul că specificațiile CSS3 există din 2001. Așa că, sunt câteva proprietăți (cum ar fi și "word-wrap") care au avut suport de foarte mult timp.

```
#selector {
   word-wrap: break-word;
```

```
}
```

Practic, această proprietate setează conținutul textului vizat în elementul “selector“ astfel încât să nu se depășească lățimea setată pentru “selector“. De exemplu, dacă avem o lățime de 200px pentru un anumit element se poate să avem în interiorul acelui element un cuvânt foarte lung care este mai mare decât lățimea elementului (200px). Cu această setare se poate forța textul să se rupă astfel încât să nu mai iasa înafara elementului (chiar dacă se rupe cuvântul la mijloc).

Valorile luate de această proprietate mai pot fi:

- Normal - valoare implícită, spargerea textului făcându-se după regulile uzuale
- break-all - liniile se pot sparge între oricare două litere
- keep-all – cuvintele de două litere nu se pot sparge
- initial – setează valoarea default
- inherit – preia setarea de la elementul părinte

Animație 2D

Cu CSS3 putem crea animații în pagina web fără a avea nevoie de animații Flash, imagini sau JavaScript. **O animație CSS3 permite unui element să își schimbe treptat stilul de la o stare la alta.** Există posibilitatea de a modifica oricâte proprietăți dorim și ori de cate ori avem nevoie. Schimbarea se poate specifica în procente (**de la 0 la 100, unde 0 reprezintă începutul animației și 100 sfârșitul ei**) sau folosind cuvinte cheie precum **“from” sau “to”.**

Pentru a realiza o animație avem nevoie de selectorul "**@keyframes**" care indică crearea animației. La acest selector trebuie precizat un element pentru care se vrea animația. Cel mai simplu exemplu este dat mai jos, în care pentru o etichetă avem atașată proprietatea "animation". Animația este creată prin cuvintele "from" și "to", și modifică treptat culoarea de fundal a etichetei din rosu în galben (exemplu 1).

Exemplu 1

```
<!DOCTYPE html>
<html>
<head>
<style>
div
{
width:100px;
height:100px;
background:red;
-webkit-animation:anim 5s; /* Chrome, Safari, Opera */
animation:anim 5s;
}
/* Chrome, Safari, Opera */
@-webkit-keyframes anim
{
from {background:red;}
to {background:yellow;}
}
/* Standard syntax */
div
@keyframes anim
{
from {background:red;}
to {background:yellow;}
}
</style>
</head>
<body>
```

```
<div>
</div>
</body>
</html>
```

Dacă se dorește o animație cu mai multe etape de modificare se folosește tehnica cu procente, astfel încât la orice pas (procent) dorit se face trecerea la o alta stare (exemplu 2).

Exemplu 2

```
<!DOCTYPE html>
<html>
<head>
<style>
div
{
width:100px;
height:100px;
background:red;
-webkit-animation:anim 5s; /* Chrome, Safari, Opera */
animation:anim 5s;
}
/* Chrome, Safari, Opera */
@-webkit-keyframes anim
{
0%   {background:red;}
25%  {background:yellow;}
50%  {background:blue;}
100% {background:green;}
}
/* Standard syntax */
@keyframes anim
{
0%   {background:red;}
25%  {background:yellow;}
50%  {background:blue;}
100% {background:green;}
}
</style>
</head>
<body>
```

```
<div>text
</div>
</body>
</html>
```

Animație 3D

Cu adevărat animația 3D nu există în plan bidimensional, însă cu așa numitele transformări din CSS3 se poate rezolva schimbarea perspectivei asupra unui obiect 2D. CSS3 introduce 2 tipuri de transformări: 2D și 3D. Transformările 2D sunt deja suportate de aproape toate browser-ele, însă cele 3D doar într-un număr limiat.

Cu o singura linie de cod CSS se poate transforma un element după următoarele reguli:

- skew – transformare oblică
- scale – mărire sau micșorare
- rotate – rotire
- translate – decalare

```
#skew {
  transform:skew(35deg);
}
#scale {
  transform:scale(1,0.5);
}
#rotate {
  transform:rotate(45deg);
}
#translate {
  transform:translate(10px, 20px);
}
#rotate-skew-scale-translate {
  transform:skew(30deg) scale(1.1,1.1) rotate(40deg)
translate(10px, 20px);
}
```

Transformările 3D sunt similare celor 2D, principalele proprietăți fiind “translate3d”, “scale3d”, “rotateX/Y/Z”, Un cod simplificat care să exemplifice proprietățile de mai sus:

```
#trans div {
  transition:all 2s ease-in-out;
  perspective: 800px;
  perspective-origin: 50% 100px;
}
#trans:hover #rotateX
{
  transform:rotateX(180deg);
}
#trans:hover #rotateY {
  transform:rotateY(180deg);
}
#trans:hover #rotateZ {
  transform:rotateZ(180deg);
}
```

Coloane CSS3

Cu CSS se pot crea diferite stiluri de prezentări web asemănătoare ziarelor tipărite, așa numite coloane multiple. Cu ajutorul “column-count” se poate preciza în câte coloane trebuie împărțit un anumit element. Dacă dorim să avem 3 coloane într-o etichetă numită “news” nu trebuie decât să adăugăm această proprietate la codul CSS al respectivului . În acest fel textul conținut în “news” o să fie împărțit în coloane.

```
.news
{
-webkit-column-count:3; /* Chrome, Safari, Opera */
```

```
-moz-column-count:3; /* Firefox */
column-count:3;
}
```

Pentru a specifica distanța dintre coloane s-a introdus proprietatea "column-gap" care preia o mărime declarată în pixeli. Pentru a continua cu stilizarea coloanelor se poate seta proprietatea "column-rule" cu ajutorul căreia putem seta dimensiunea, stilul și culoarea pentru fiecare coloana.

```
.news
{
-webkit-column-count:3; /* Chrome, Safari, Opera */
-moz-column-count:3; /* Firefox */
column-count:3;
-webkit-column-gap:40px; /* Chrome, Safari, Opera */
-moz-column-gap:40px; /* Firefox */
column-gap:40px;
-webkit-column-rule:4px dotted #aa0000; /* Chrome, Safari, Opera */
-moz-column-rule:4px dotted #aa0000; /* Firefox */
column-rule:4px dotted #aa0000;
}
```

Site responsive

- **utilizare viewport (port de vizualizare)**

Un port de vizualizare este un meta tag de bază în crearea unui site responsive, fără de care întregul proces nu ar fi posibil.

Acesta indică browser-ului că pagina trebuie să fie scalată pentru a se potrivi pe dimensiunea ecranului. Sunt mai multe moduri în care poți aplica acest viewport, însă iat-o pe cea mai uzuală:

HTML5 a introdus o metodă pentru a permite proiectanților web să preia controlul asupra viewport-ului, prin eticheta <meta>.
Ar trebui să includeți următorul element <meta> viewport în toate paginile dvs. web:

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

Un element de vizualizare <meta> oferă instrucțiunile browserului despre cum să controlați dimensiunile și scalarea paginii.
Lățimea = partea lățimii dispozitivului stabilește lățimea paginii pentru a urma lățimea ecranului dispozitivului (care va varia în funcție de dispozitiv).
Partea inițială = 1,0 setează nivelul de zoom inițial atunci când pagina este încărcată prima dată de browser.

Acesta se plasează de obicei în secțiunea <head> a paginilor HTML ale site-ului tău.

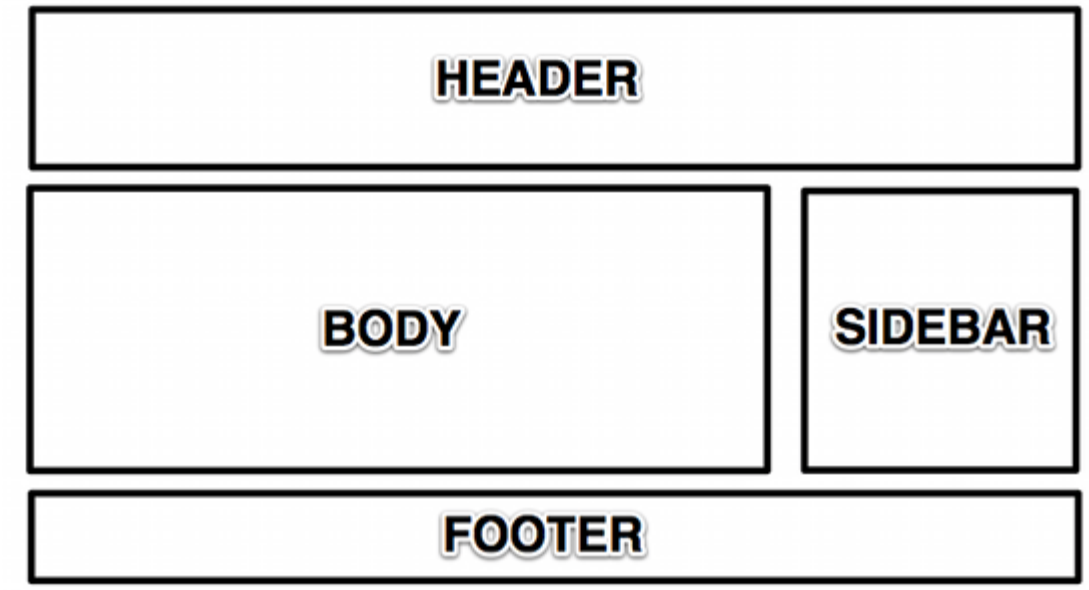
Note: The meta tag should be added in the head tag in HTML document.

A Responsive tags has the following attributed:

- **width**: Width of the virtual viewport of the device.
- **height**: Height of the virtual viewport of the device.
- **initial-scale**: Zoom level when the page is first visited.
- **minimum-scale**: Minimum zoom level to which a user can zoom the page.
- **maximum-scale**: Maximum zoom level to which a user can zoom the page.
- **user-scalable**: Flag which allows the device to zoom in or out. (value= yes/no).

Setează fiecare breakpoint

Este necesar să setezi o serie de breakpoint-uri în locurile în care vrei ca elementele din pagina ta să se deplaseze astfel încât să se afișeze bine pe ecran.



Scalarea paginii pentru dispozitivele mobile se poate face printr-o serie de modificări la CSS.

Interogarea media este o tehnică CSS introdusă în CSS3.

Folosește regula @media pentru a include un bloc de proprietăți CSS numai dacă o anumită condiție este adevărată.

Exemplu

Dacă fereastra browserului este de 600px sau mai mică, culoarea de fundal va fi deschisă la culoare:

```
@media numai ecran și (lățime maximă: 600px) {
  corp {
    fundal-culoare: lightblue;
  }
}
```

Pentru a finaliza setarea breakpoint-ului, trebuie să constrângem și lățimea maximă.

Constrângerea lățimii minime ne ajută să facem un site responsive pentru dispozitivele cu ecran mic, iar cea a lățimii maxime este destinată adaptării site-ului pentru ecranele mari.

Pe ecranele foarte largi, există riscul ca textul și paragrafele, de exemplu, să se afișeze sub formă de rânduri foarte lungi. De aceea, trebuie să alegem să constrângi rândurile într-o anumită dimensiune.

Exemplu ca lățimea maximă a punctului de întrerupere să fie 1200 px:

1. Includem pagina intr-un container

   …

Un container este un element cu class = "container". Containerul va afecta toate elementele din containerul . Containerele au în mod implicit un capitonaj de 15 pixeli, care îl amortizează de la sfârșitul paginii și de la alte containere. Rândurile și coloanele vor fi adăugate în interiorul containerelor.

2. **Definim secțiuni pentru fiecare container**

<h2>Textul nostru</h2>…

Exemplu

Prefixes define what device the column is meant for. For example, the *sm* is designed for small screens, like phones.

Size	**Prefix**	**Column**	**For**
Extra Small	None (default)	.col-	<576
Small (phones)	sm	.col-sm-	≥576px
Medium (tablets)	md	.col-md-	≥768px
Large (laptops)	lg	.col-lg-	≥992px
Extra Large (desktops)	xl	.col-xl-	≥1200px

The *ColumnNumber* at the end of grid class, such as `class="col-md-4"`, establishes how many columns that it should span. In the example, `class="col-md-4"`, the content within the column's `<div>` would span four columns on a medium sized device, like a laptop. When a column is set with a prefix for a smaller device, such as a *sm*, it will display similarly on larger devices as well. In other words, a column defined as *sm* will work for devices of tablet size or larger.

To make columns align side-by-side, include them in the same row.

```
<div class="row">
  <div class="col-md-4"> Content </div>
  <div class="col-md-8"> Content </div>
</div>
```

In the end, you want your columns in a row to add up to twelve.

Adjusting Columns for Different Devices

You can also adjust your columns so that they will have different column widths per device size. To do this, you simply include another `col-prefix-columnNumber` to your class. It will look something like this:

```
<div class=”container”>
  <div class=”row”>
    <div class=”col-xs-6 col-sm-4”> Content </div>
  </div>
</div>
```

3 Definim marginea maximă a containerului în CSS:

.container {

margin : auto ;

max-width : 1200px ;

}

- **Seteam marginile și reducem dimensiunea textului**

Pe un port de vizualizare îngust, site-ul se afișează pe ecranele mici, spațiul de afișare a textului nu este suficient de mare încât setările de formatare (heading-uri, bold-uri, liste, tabele etc) să se afișeze corect.

Pentru un port de vizualizare mare, deși chiar daca am setat ca textul să se centreze și să nu se afișeze pe rânduri foarte lungi, trebuie stiut ca, vizitatorul se află la distanță de ecran si, exista riscul ca textul cu un font 14 sau 16 să nu se poată citi.

Trebuie realizata o adaptare dimensiunii textului pentru ecranul pe care este afișat. Se recomanda marirea textului din anumite zone, pentru ca acestea sa fie evidentiate.

Exemplu marirea dimensiunii antetelor pentru fiecare secțiune de conținut si stabilirea dimensiunii acestora ca fiind maxim 10% din lățime:

#headline {

padding : 20px 10% ;

}

- **Adaptarea elementelor**

✓ Adaptarea elementelor vizuale;

- ✓ Repoziționare videoclipurilor și imaginilor;
- ✓ Adaptarea formularele pentru ecranele cu dimensiuni mici;
- ✓ Extindere tabele pentru ecranele de dimensiuni mari.

Efectuarea acestor setări depinde de complexitatea fiecărui site.

Aplicatii

What is Responsive Web Design?

Responsive Web Design is about using HTML and CSS to automatically resize, hide, shrink, or enlarge, a website, to make it look good on all devices (desktops, tablets, and phones):

Example

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

This will set the viewport of your page, which will give the browser instructions on how to control the page's dimensions and scaling.

Responsive Images

```
<img src="img_girl.jpg" style="width:100%;">
```

Notice that in the example above, the image can be scaled up to be larger than its original size. A better solution, in many cases, will be to use the `max-width` property instead.

Using the max-width Property

If the `max-width` property is set to 100%, the image will scale down if it has to, but never scale up to be larger than its original size:

Example

```
<img src="img_girl.jpg" style="max-width:100%;height:auto;">
```

Show Different Images Depending on Browser Width

The HTML `<picture>` element allows you to define different images for different browser window sizes.

Resize the browser window to see how the image below change depending on the width:

Example

```
<picture>
  <source srcset="img_smallflower.jpg" media="(max-width: 600px)">
  <source srcset="img_flowers.jpg" media="(max-width: 1500px)">
  <source srcset="flowers.jpg">
  <img src="img_smallflower.jpg" alt="Flowers">
</picture>
```

Responsive Text Size

The text size can be set with a "vw" unit, which means the "viewport width".

That way the text size will follow the size of the browser window:

Hello World

Resize the browser window to see how the text size scales.

Example

```
<h1 style="font-size:10vw">Hello World</h1>
```

Viewport is the browser window size. 1vw = 1% of viewport width. If the viewport is 50cm wide, 1vw is 0.5cm.

Media Queries

Example

```
<style>
.left, .right {
  float: left;
  width: 20%; /* The width is 20%, by default */
}

.main {
  float: left;
  width: 60%; /* The width is 60%, by default */
}

/* Use a media query to add a breakpoint at 800px: */
```

```
@media screen and (max-width: 800px) {
  .left, .main, .right {
    width: 100%; /* The width is 100%, when the viewport is 800px or smaller */
  }
}
</style>
```