10 ROZMIESTŇOVANIE OBJEKTOV

Štandardne sa elementy, z ktorých sa skladá stránka, zobrazujú v prehliadači „jeden za druhým" v takom poradí, v akom sú definované v HTML kóde. Od typu elementu, t.j. či je element *blokový* alebo *riadkový*, závisí, či sa elementy zoraďujú pod seba, alebo vedľa seba.

- Blokové elementy sa radia za sebou zhora nadol. Štandardne je šírka elementu na celú šírku okna prehliadača, resp. maximálna šírka v rámci elementu, v ktorom je vložený. Z toho vyplýva, že blokové elementy nemôžu byť vedľa seba v jednom riadku (obrázok 10.1).
- Riadkové elementy sa radia vodorovne vedľa seba zľava doprava. Šírka elementu je daná jeho obsahom (obrázok 10.2).

Obrázok 10.1 Radenie blokových elementov.

Obrázok 10.2 Radenie riadkových elementov.

Relatívne umiestňovanie

Každý objekt (element) umiestnený normálnym spôsobom, môžeme relatívne posunúť vzhľadom k jeho základnej polohe (tzv. relatívne umiestňovanie). To urobíme tak, že elementu nastavíme vlastnosť `position` na hodnotu `relative` a zároveň pomocou niektorej (niektorých) z vlastností `top`, `right`, `bottom`, `left` určíme jeho posunutie.

PRÍKLAD 10.1

V editore JSFiddle vložme do HTML časti kód z obrázka 10.1 (súbor `10/relativne.txt`). Element `div` posunieme voči jeho **základnej polohe** o 100 bodov **zľava** a 10 bodov **zdola** (obrázok 10.3). V časti CSS definujeme štýl:

```
div {
  position: relative;
  left: 100px;
  bottom: 10px;
}
```

Obrázok 10.3 Relatívne posunutie elementu (sivý text naznačuje pôvodnú polohu elementu div).

Pri relatívnom umiestňovaní majú vlastnosti `top`, `right`, `bottom` a `left` takýto význam:

- `top` určuje, o koľko posúvame objekt **zhora** (teda koľko miesta nad ním sa má vynechať),
- `right` určuje, o koľko posúvame objekt **sprava** (teda koľko miesta vpravo od neho sa má vynechať),
- `bottom` určuje, o koľko posúvame objekt **zdola** (teda koľko miesta pod ním sa má vynechať),
- `left` určuje, o koľko posúvame objekt **zľava** (teda koľko miesta vľavo od neho sa má vynechať).

ÚLOHA 10.2

V kóde z *príkladu 10.1*:

- experimentujte s hodnotou vlastnosti `left`, skúšajte napr. hodnoty `10px`, `50px`, `300px`, `-20px`, `1em`, `10%`, `50%`, …
- experimentujte s hodnotou vlastnosti `bottom`, skúšajte napr. `40px`, `80px`, `-20px`, `5%`, `1em`, …
- namiesto vlastnosti `left` nastavte `right`: `10px`, potom skúšajte iné hodnoty,
- namiesto vlastnosti `bottom` nastavte `top`: `1em`, potom skúšajte iné hodnoty,
- nastavte potrebné vlastnosti a ich hodnoty tak, aby element `div` bol vedľa nadpisu `h2` (obrázok 10.4).

Obrázok 10.4 Element div vedľa elementu h2.

Pri relatívnom umiestnení objektu prehliadač vyhradí priestor, v ktorom by bol objekt normálne zobrazený, no pomocou vlastností `top`, `left`, `right` či `bottom` môžeme objekt z tohto priestoru vysunúť. Posun objektu pomocou `position: relative` nemá žiaden vplyv na umiestnenie iných objektov. Môže sa teda stať, že relatívne umiestnený objekt bude prekrývať iné objekty. Relatívne umiestňovanie sa používa skôr na „dokreslenie“ objektov, a nie definovanie nejakého základného rozloženia (štruktúry) objektov.

Absolútne umiestňovanie

PRÍKLAD 10.3

V CSS kóde z *príkladu 10.1* nahradíme hodnotu `relative` hodnotou `absolute`.

Na obrázku 10.5 vidíme výsledok - element `div` „odskočil” k dolnému okraju stránky.

```
div {
  position: absolute;
  left: 100px;
  bottom: 10px;
}
```

Obrázok 10.5 Absolútne umiestnenie elementu.

Skôr, ako si vysvetlíme, čo vlastne absolútna pozícia znamená, budeme experimentovať s predchádzajúcim kódom.

ÚLOHA 10.4

V kóde z *príkladu 10.3* riešte nasledujúce úlohy. Po každej zmene hodnoty meňte v editore JSFiddle veľkosť časti s výslednou stránkou (Result) - šírku aj výšku.

- experimentujte s hodnotami vlastnosti `left`,
- experimentujte s hodnotami vlastnosti `bottom`, vyskúšajte aj záporné hodnoty,
- namiesto vlastnosti `left` použite vlastnosť `right`, experimentujte s jej hodnotami,
- namiesto vlastnosti `bottom` použite vlastnosť `top`, experimentujte s jej hodnotami.

Na základe nášho skúmania by sa dalo predpokladať, že pri absolútnom umiestnení sa objekt umiestni na zadanú pozíciu (hodnotami `top`, `left`, `bottom`, `right`) v rámci stránky. Nasledujúci príklad nás presvedčí o tom, že to nie je úplne tak.

PRÍKLAD 10.5

V kóde z *príkladu 10.3* „zabalíme" všetky údaje o súťaži PALMA do elementu `<section>`. Elementu `section` v CSS nastavíme `position` na hodnotu `relative` a kvôli názornosti mu ešte pridáme orámovanie. Výsledná stránka je na obrázku 10.6.

```
<body>
  <h1>Informatické súťaže</h1>
  <section>
    <h2>Palma</h2>
    <div>zameranie:
programovanie</div>
    <p>Súťaž je určená študentom SŠ.
Organizuje ju Prírodovedecká fakulta
UPJŠ v Košiciach a združenie
STROM.</p>
  </section>
</body>
```

```
div {
  position: absolute;
  left: 100px;
  bottom: 10px;
}
section {
  border: 1px solid red;
  position: relative;
}
```

Obrázok 10.6 Absolútne umiestnenie elementu v rámci sekcie.

Vidíme, že element `div` už nie je umiestnený `10px` od spodného okraja stránky. Poďme znova skúmať jeho polohu.

ÚLOHA 10.6

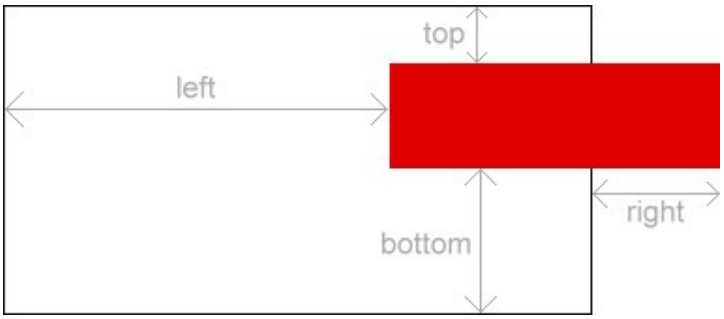
V CSS kóde z *príkladu 10.5* experimentujte so štýlom pre `div`:

- skúšajte rôzne hodnoty pre vlastnosť `bottom`,
- skúšajte rôzne hodnoty pre vlastnosť `left`,
- nahraďte vlastnosť `bottom` vlastnosťou `top` a skúšajte rôzne hodnoty,
- nahraďte vlastnosť `left` vlastnosťou `right` a skúšajte rôzne hodnoty,
- zrušte vlastnosť `border` pre `section` a nastavte potrebné vlastnosti tak, aby element `div` bol vedľa nadpisu `h2` (obrázok 10.4),
- pridajte informácie o ďalších súťažiach (môžete použiť kód zo súboru `10/sutaze.txt`).

Zdá sa, že vlastnosti `top`, `left`, `right` a `bottom` nám teraz určujú pozíciu v rámci elementu `section`.

ZAPAMÄTAJTE SI

- Umiestnený objekt je objekt, ktorý má nastavenú vlastnosť `position` na inú ako prednastavenú hodnotu (`static`).
- Absolútne umiestnený objekt (teda objekt, ktorému nastavíme `position: absolute`) sa posunie na danú pozíciu v rámci najbližšieho umiestneného rodičovského objektu. Ak absolútne umiestnený objekt nemá žiadneho umiestneného rodiča, umiestňuje sa v rámci elementu `body`.
- Pozíciu absolútne umiestneného objektu definujeme pomocou vlastností `top`, `right`, `bottom` a `left`, ktoré v tomto prípade majú takýto význam (pozri aj obrázok 10.7):
  - `top` určuje, o koľko je horný okraj elementu posunutý od horného okraja umiestneného rodiča, resp. elementu body (ďalej len rodiča),
  - `right` určuje, o koľko je pravý okraj elementu posunutý vľavo od pravého okraja rodiča,
  - `bottom` určuje, o koľko je dolný okraj elementu posunutý hore od dolného okraja rodiča,
  - `left` určuje, o koľko je ľavý okraj elementu posunutý vpravo od ľavého okraja rodiča.
- Absolútne umiestnený objekt nemá žiaden vplyv na umiestnenie ostatných objektov na jeho úrovni (vnútri umiestneného rodičovského objektu).
- Absolútne umiestnené objekty môžu prekrývať iné objekty na stránke.



Obrázok 10.7 Význam vlastností top, right, bottom a left pri absolútnom umiestňovaní.
Červený objekt je absolútne umiestnený objekt v rámci umiestneného rodiča (objekt s čiernym rámikom).
Hodnota right bude v tomto prípade záporná.

V *príklade 10.5* sme absolútne umiestnili element `div`. Jeho rodičom je element `section`, ktorý má definovanú vlastnosť `position: relative`, teda je umiestnený. Umiestnenie elementu `div` sa určuje vzhľadom na element `section`. V *príklade 10.3* element `div` žiadneho umiestneného rodiča nemal, preto sa jeho poloha určovala vzhľadom na element `body`.

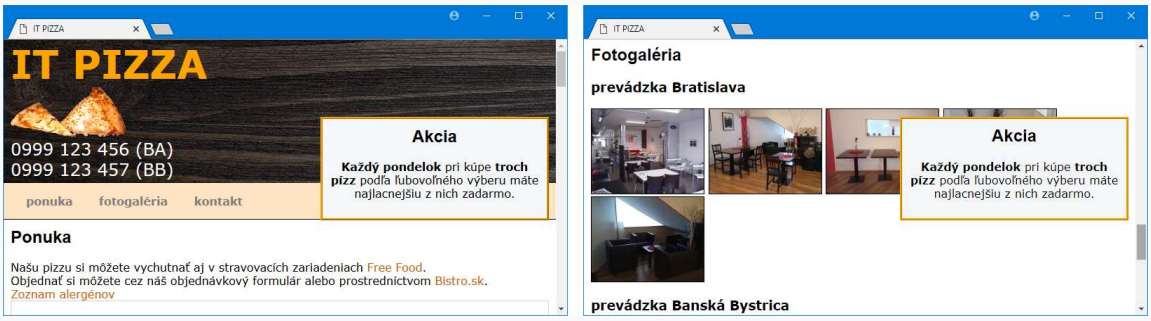
Fixné umiestňovanie

Okrem hodnôt `relative` a `absolute` môže mať vlastnosť `position` aj hodnotu `fixed`.

PRÍKLAD 10.7

Na stránke IT Pizza umiestnime časť *Akcia* tak, aby bola vždy v pravej hornej časti okna prehliadača, bez ohľadu na to, ktorú časť stránky práve vidíme (obrázok 10.8). Budeme vychádzať zo súboru `10/index.html`.

```
aside {
  ...
  position: fixed;
  right: 10px;
  top: 100px;
  width: 300px;
}
```



Obrázok 10.8 Fixovaná časť Akcie (vľavo hlavička a ponuka, vpravo fotogaléria).

Ak nejakému elementu nastavíme `position: fixed`, potom hodnotami `top`, `left`, `bottom` a `right` určujeme jeho pozíciu vzhľadom na okraje okna prehliadača. Táto pozícia zostáva fixná, aj keď stránku posúvame.

ÚLOHA 10.8

Skúmajte stránku IT Pizza z predchádzajúceho príkladu:

- posúvajte stránku hore a dolu a pozorujte umiestnenie časti *Akcia*,
- meňte veľkosť okna prehliadača a pozorujte umiestnenie časti *Akcia*,
- meňte hodnoty `top` a `right` a pozorujte umiestnenie časti *Akcia*,
- namiesto `top` a `right` vyskúšajte `bottom` a `left`.

Plávajúce umiestňovanie

Plávajúce objekty

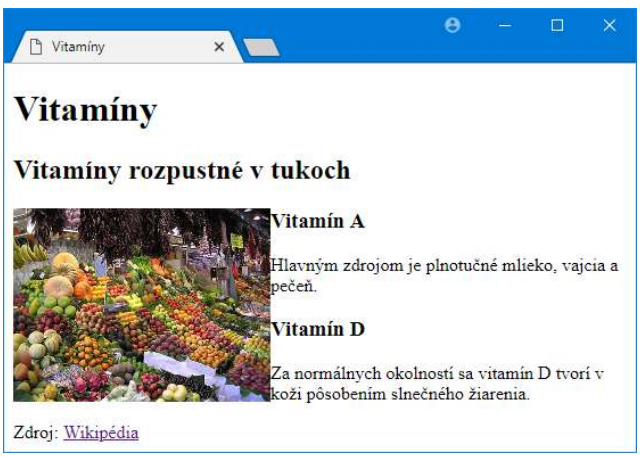
V textových dokumentoch vieme umiestniť obrázok do textu tak, aby text obtekal okolo obrázka. Efekt obtekania obrázka textom vieme čiastočne vytvoriť aj na webových stránkach.

Obrázok 10.9 Stránka s obrázkom: text je nad a pod obrázkom.

PRÍKLAD 10.9

Na *obrázku 10.9* je stránka o vitamínoch a jej zdrojový kód (súbor `10/vitaminy1.html`). Vidíme, že na stránke sa nachádza obrázok medzi dvoma nadpismi, presnejšie jeden nadpis je nad obrázkom a druhý pod obrázkom, pričom vedľa obrázka vpravo je prázdny priestor. Upravíme stránku tak, aby obrázok bol vľavo a ostatné objekty ho obtekali sprava (*obrázok 10.10*). Obtekanie obrázka vyriešime definovaním štýlu pre element `img`.

```
<style>
  img {
    float: left;
  }
</style>
```



Obrázok 10.10 Plávajúce umiestnenie elementu img (obrázok).

Pomocou vlastnosti `float` sme z obrázka spravili tzv. **plávajúci objekt**. Ako sa taký plávajúci objekt správa, resp. ako sa správajú iné objekty na stránke, ak je medzi nimi plávajúci objekt, budeme skúmať v nasledujúcej úlohe.

ÚLOHA 10.10

V kóde z *príkladu 10.9*:

- meňte šírku okna a pozorujte, ktoré objekty obtekajú obrázok,
- nastavte šírku okna na najväčšiu možnú a určte, ktoré objekty obtekajú obrázok,
- zmeňte hodnotu vlastnosti `float` na `right` a zopakujte predchádzajúce úlohy,
- v HTML kóde pridajte za element `img` ďalší `img` element (môžete použiť aj ten istý) a pozorujte obtekanie.

Plávajúce objekty sú posunuté úplne vľavo alebo vpravo v rámci bloku, v ktorom sa nachádzajú (buď k okraju bloku alebo k inému plávajúcemu objektu). Okolo plávajúcich objektov môže (ale nemusí) obtekať ďalší obsah stránky. Každý plávajúci objekt má presne stanovenú šírku, buď explicitne určenú vlastnosťou `width`, alebo prirodzenú šírku (podľa obsahu). Pripomíname, že „neplávajúci“ blokový objekt zaberá celú šírku stránky alebo nadradeného elementu, aj keby jeho obsah bol menší.

Vlastnosť `float` môže nadobúdať hodnoty:

- `left` – objekt sa posunie k ľavému okraju,
- `right` – objekt sa posunie k pravému okraju,
- `none` – zrušenie plávania objektu.

PRÍKLAD 10.11

Upravíme CSS kód z *príkladu 10.9* tak, aby päta stránky (`footer`) bola vždy pod obrázkom, t.j. aby ho nemohla obtekať, alebo ešte inak povedané, aby obrázok nemohol plávať vľavo vedľa päty, čo pri doterajšom riešení mohol a pri istej šírke okna aj pláva.

```
<style>
  img {
    float: left;
  }
  footer {
    clear: left;
  }
</style>
```

Pomocou vlastnosti `clear` definujeme, na ktorej strane objektu nemôžu byť plávajúce objekty. Vlastnosť `clear` môže nadobúdať hodnoty:

- `left` – na ľavej strane objektu nemôže byť plávajúci objekt,
- `right` – na pravej strane objektu nemôže byť plávajúci objekt
- `both` – na ľavej ani pravej strane objektu nemôže byť plávajúci objekt,
- `none` – okolo objektu môžu byť plávajúce objekty.

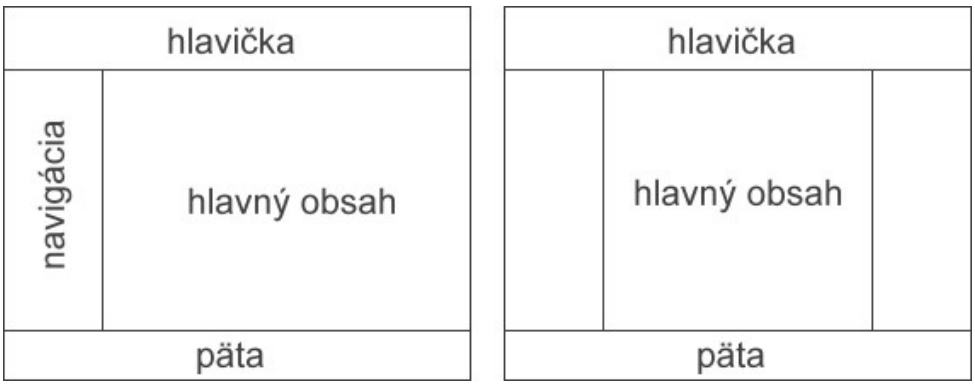
Vlastnosť `clear` zvyčajne nastavujeme objektu nasledujúcemu za plávajúcim objektom.

- Ak plávajúci objekt pláva vľavo, potom nasledujúcemu objektu nastavíme `clear: left` (ale môžeme aj `clear:both`).
- Ak plávajúci objekt pláva vpravo, použijeme `clear: right` (alebo `clear: both`).
- Ak máme viacero plávajúcich objektov s rôznymi nastaveniami pre `float`, použijeme `clear: both`.

Plávajúcim objektom môže byť ľubovoľný element, nielen obrázok.

Použitie plávajúcich objektov na vytvorenie viacstĺpcového dizajnu

Plávajúce objekty sa často využívajú na vytvorenie viacstĺpcového rozloženia objektov na stránke. Ide o také rozloženie objektov, kde celá stránka alebo jej časť je rozdelená na niekoľko stĺpcov (zvyčajne dva alebo tri). Schematické znázornenie dvoj- a trojstĺpcového rozloženia stránky si môžete pozrieť na obrázku 10.11.



Obrázok 10.11 Schéma dvojstĺcového a trojstĺcového rozloženia stránky.

ODPOVEDZTE

Akú schému majú stránky, ktoré najčastejšie navštevujete? Do koľkých stĺpcov sú rozložené a čo sa nachádza v jednotlivých stĺpcoch? Nájdite aspoň dve stránky s dvojstĺpcovým a aspoň dve stránky s trojstĺpcovým rozložením. Ako sa tieto stránky zobrazujú v mobile či tablete?

Stránka IT Pizza, tak ako sme ju navrhli v predchádzajúcich kapitolách (od prvej až po deviatu), má jednostĺpcové rozloženie objektov. Pri takomto rozložení sa stránka dobre zobrazí na obrazovkách rôznych zariadení, od mobilov až k počítačom.

V nasledujúcich príkladoch budeme postupne meniť rozloženie objektov na stránke IT Pizza tak, aby sa zobrazovali vo viacerých (v dvoch, v troch) stĺpcoch. Viacstĺpcové rozloženie objektov však už nie je vhodné pre zobrazovanie napr. v mobiloch či tabletoch. Preto ďalšie vlastnosti, ktoré jednotlivým elementom stránky IT Pizza od tohto momentu pridáme, nebudeme pridávať k už definovaným vlastnostiam, ale definujeme ich samostatne, za všetkými doteraz vytvorenými štýlmi. Ak napríklad chceme pridať spodný vnútorný okraj pre element `div` s telefónnymi číslami, tak to neurobíme takto:

```
header div {
  color: #FFF;
  font-size: 1.5em;
  padding-bottom: 1em;
}
```

ale takto:

```
<style>
  /* všetky doteraz definované štýly */
  ...
  header div {
    color: #FFF;
    font-size: 1.5em;
  }
  ...

  /* štýly súvisiace s viacstĺpcovým rozložením */
  header div {
    padding-bottom: 1em;
  }
</style>
```

PRÍKLAD 10.12

Na stránke IT Pizza rozdelíme fotky v galérii do dvoch stĺpcov: v ľavom stĺpci budú fotky z prevádzky v Bratislave, v pravom stĺpci fotky z prevádzky v Banskej Bystrici (*obrázok 10.12*). Vychádzame zo súboru `10/index-float.html`.

Obrázok 10.12 Galéria v dvoch stĺpcoch (vľavo Bratislava, vpravo Banská Bystrica).

Potrebujeme presne určiť, čo patrí do ľavého a čo do pravého stĺpca. Do ľavého stĺpca patrí nadpis `<h3>prevádzka Bratislava</h3>` a nasledujúcich päť `img` elementov. Zabalíme ich do jedného elementu `div`. Nadpis `<h3>prevádzka Banská Bystrica</h3>` a nasledujúcich päť `img` elementov vnoríme do ďalšieho elementu `div`. Sekcia s fotogalériou bude mať teda nasledujúci kód:

```
<section id="galeria">
  <h2>Fotogaléria</h2>
  <div>
    <h3>prevádzka Bratislava</h3>
    <img src="obrazky/foto-01m.jpg" alt="foto 1" ...>
    <img src="obrazky/foto-02m.jpg" alt="foto 2" ...>
    <img src="obrazky/foto-03m.jpg" alt="foto 3" ...>
    <img src="obrazky/foto-04m.jpg" alt="foto 4" ...>
    <img src="obrazky/foto-05m.jpg" alt="foto 5" ...>
  </div>
```

```
  <div>
    <h3>prevádzka Banská Bystrica</h3>
    <img src="obrazky/foto-06m.jpg" alt="foto 6" ...>
    <img src="obrazky/foto-07m.jpg" alt="foto 7" ...>
    <img src="obrazky/foto-08m.jpg" alt="foto 8" ...>
    <img src="obrazky/foto-09m.jpg" alt="foto 9" ...>
    <img src="obrazky/foto-10m.jpg" alt="foto 10" ...>
  </div>
</section>
```

Z oboch stĺpcov spravíme objekty plávajúce vľavo a nastavíme im šírku 50% (aby boli oba rovnaké a spoločne zaberali celú šírku stránky). Keďže oba elementy `div` budú mať rovnaké vlastnosti a iné elementy `div` v sekcii s identifikátorom `galeria` nemáme, môžeme definovať štýl `#galeria div`, t.j. štýl pre tie elementy `div`, ktoré sú v elemente s `id galeria`).

```
/* štýly súvisiace s viacstĺpcovým rozložením */
#galeria div {
  float: left;
  width: 50%;
}
```

Všimnite si, že žltá farba pozadia päty „pretiekla" aj do sekcie s galériou, ktorá mala pôvodne biele pozadie. Ešte by sme mali zabezpečiť, aby objekt, ktorý nasleduje za plávajúcimi `div` elementami, nemohol mať vedľa seba žiadne plávajúce objekty. Za sekciou galéria nasleduje päta stránky, preto vytvoríme štýl pre `footer` s nastavením `clear: left`.

```
/* štýly súvisiace s viacstĺpcovým rozložením */
#galeria div {
  float: left;
  width: 50%;
}
footer {
  clear: left;
}
```

ODPOVEDZTE

Čo by sa zmenilo, keby sme v *príklade 10.12* použili `float: right` a `clear: right`? Vyskúšajte.

ÚLOHA 10.13

Na stránke IT Pizza využite plávajúce objekty na vytvorenie trojstĺpcového rozloženia päty (obrázok 10.13). Pomôcka: Vytvorte div elementy pre každý zo stĺpcov (kontakt Bratislava, kontakt Banská Bystrica, kontakty na sociálne siete), nastavte im vhodné identifikátory a definujte potrebné štýly. Šírky stĺpcov rozumne prispôsobte (nemusia byť rovnako široké).

Obrázok 10.13 Rozloženie informácií v päte do troch stĺpcov.

ZAPAMÄTAJTE SI

Ak chceme plávajúce objekty použiť na vytvorenie viacstĺpcového vzhľadu stránky alebo jej časti:

- definujeme každý stĺpec ako samostatný element (ak stĺpec tvorí viacero elementov, vnoríme ich napr. do elementu `div`),
- z každého stĺpca spravíme objekt plávajúci vľavo pomocou `float: left`,
- objektu, ktorý v HTML kóde nasleduje bezprostredne po plávajúcich objektoch (stĺpcoch), zakážeme, aby vedľa neho vľavo bol nejaký plávajúci objekt pomocou `clear: left`, resp. `clear: both`,
- každému stĺpcu definujeme šírku tak, aby súčet šírok vrátane ľavých a pravých okrajov bol spolu 100%,
- ak treba, nastavíme iné potrebné vlastnosti (napr. výšku pre niektorý z plávajúcich objektov).

S akými problémami sa môžeme stretnúť pri použití plávajúcich objektov?

- Pri zmenšení okna prehliadača sa začnú jednotlivé stĺpce prekrývať a informácie v nich budú nečitateľné. Riešenie tohto problému si ukážeme v *kapitole 11*.
- Môže sa stať, že obsah objektu „vytečie" z boxu, ktorý je preň definovaný. Plávajúce objekty dobre fungujú vtedy, ak vieme garantovať, že výšky jednotlivých stĺpcov sú približne rovnaké, a to aj pri zmene veľkosti prehliadača. Inak je výhodnejšie použiť tzv. flexibilné boxy, ktoré dokážu prispôsobiť výšky jednotlivých stĺpcov podľa najvyššieho stĺpca. O flexibilných boxoch sa dozvieme neskôr v tejto kapitole.

Vlastnosť display

Vlastnosť display definuje spôsob zobrazenia elementu. Pomocou vlastnosti display môžeme riadkový element zmeniť na blokový alebo naopak blokový na riadkový, či dokonca element skryť. Môže nadobúdať množstvo hodnôt, my sa oboznámime s niektorými z nich.

ÚLOHA 10.14

V editore JSFiddle do časti HTML napíšte nasledujúci kód:

```
V tejto vete je použitý <em>riadkový</em> element em.
<p>Toto je odsek, štandardne blokový element. Mal by teda začínať
na samostatnom riadku.</p>
Toto je obyčajný text za odsekom.
```

V CSS časti postupne definujte nasledujúce štýly a pozorujte zobrazenie elementu `em` a `p`.

a)
```
em {background-color: lightgreen; }
```

b)
```
em {
  background-color: lightgreen;
  display: block;
}
```

c)
```
em {
  background-color: lightgreen;
  display: block;
  width: 150px;
}
```

d)
```
em {
  background-color: lightgreen;
  display: inline;
  width: 150px;
}
```

e)
```
em {
  background-color: lightgreen;
  display: inline-block;
  width: 150px;
}
```

f)
```
p {
  border: 1px solid red;
}
```

g)
```
p {
  border: 1px solid red;
  display: inline;
}
```

h)
```
p {
  border: 1px solid red;
  display: none;
}
```

Význam použitých hodnôt vlastnosti `display` popisujeme v tabuľke 10.1. Vlastnosť `display` môžeme využiť na rôzne zmeny vzhľadu navigácie.

Tabuľka 10.1 Popis vybraných hodnôt vlastnosti display.

hodnota	popis
`block`	element sa zobrazí ako blokový; začína na novom riadku, zaberá celú šírku nadradeného elementu; môžeme nastavovať šírku a výšku
`inline`	element sa zobrazí ako riadkový; nemôžeme nastavovať šírku ani výšku (tá závisí od obsahu elementu)
`inline-block`	element sa zobrazí ako riadkový, ale môžeme nastavovať šírku a výšku
`none`	element sa nezobrazí; stránka sa zobrazí tak, akoby element neexistoval

PRÍKLAD 10.15

V editore JSFiddle máme definovanú jednoduchú vodorovnú navigáciu (*obrázok 10.14*, súbor `navigacia.txt` – samostatne nakopírujeme HTML časť a CSS časť).

Obrázok 10.14 Vodorovná navigácia s odkazmi rôznej šírky.

Zmeníme jednotlivé odkazy tak, aby mali všetky rovnakú šírku a text v nich bol centrovaný (obrázok 10.15).

Obrázok 10.15 Vodorovná navigácia s odkazmi rovnakej šírky.

Element `a` je štandardne riadkový, teda vlastnosť `display` má nastavenú na hodnotu `inline`. Ak chceme, aby sa odkazy zobrazovali vedľa seba ako riadkové elementy, ale mohli sme im nastavovať šírku ako blokovým elementom, zmeníme hodnotu vlastnosti display na `inline-block`. Potom už zostáva len nastaviť vhodnú šírku a centrovanie textu.

```
<nav>
  <a href="#">vitamín A</a>
  <a href="#">vitamín B -
tiamín</a>
  <a href="#">vitamín C</a>
  <a href="#">vitamín D</a>
  <a href="#">vitamín E</a>
</nav>
```

```
nav a {
  background-color: #e6e6ff;
  padding: 10px;
  display: inline-block;
  width: 120px;
  text-align: center;
}

nav a:hover {
  background-color: #c2d6d6;
}
```

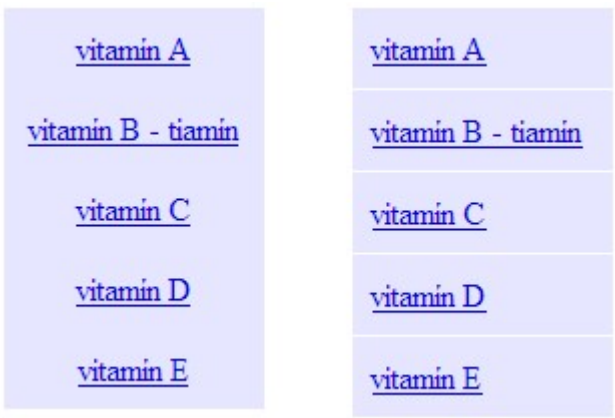
PRÍKLAD 10.16

Navigáciu z *príkladu 10.15* zmeníme na zvislú, t.j. jednotlivé odkazy sa budú zobrazovať pod sebou (*obrázok 10.16* vľavo). Umiestnenie odkazov pod sebou dosiahneme tým, že im zmeníme spôsob zobrazovania na taký, aký majú blokové elementy.

```
nav a {
  background-color: #e6e6ff;
  padding: 10px;
  display: block;
  width: 120px;
  text-align:center;
}
```

ÚLOHA 10.17

Zrušte centrovanie textu v odkazoch a vytvorte medzi nimi medzeru (*obrázok 10.16* vpravo).



Obrázok 10.16 Zvislá navigácia.

Flexibilné boxy

Aktuálne najnovšou technológiou na definovanie rozloženia objektov na webových stránkach sú tzv. **flexibilné boxy**. Ich výhodou je, že stránky vytvorené pomocou nich vieme ľahko prispôsobiť rôznym zobrazovacím zariadeniam (nielen obrazovky počítačov či notebookov, ale aj tablety, či mobily). Nevýhodou je, že nie sú podporované staršími verziami prehliadačov.

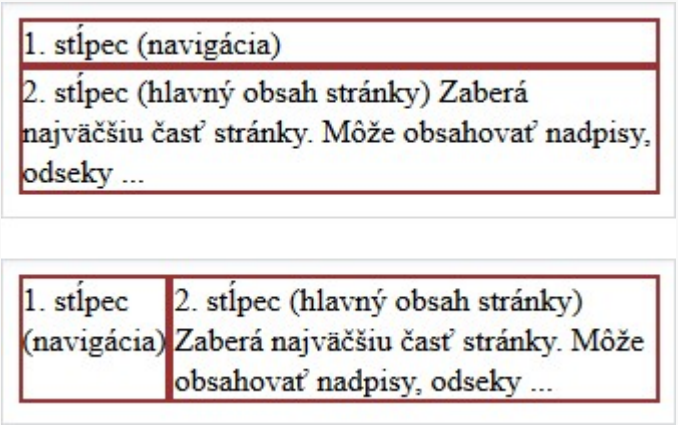
PRÍKLAD 10.18

Budeme experimentovať s flexibilnými boxami. V editore JSFiddle vytvoríme jeden `div` element s dvoma vnorenými `section` elementami (súbor `10/flex.html`). Pre sekcie definujeme v CSS jednoduché orámovanie, napr. `section {border: 2px solid #993333;}`. Meníme veľkosť okna s výslednou stránkou a pozorujeme (*obrázok 10.17* hore).

```
<div>
  <section>1. stĺpec (navigácia)</section>
  <section>2. stĺpec (hlavný obsah stránky) Zaberá najväčšiu časť
stránky. Môže obsahovať nadpisy, odseky ...
  </section>
</div>
```

V CSS nastavíme elementu `div` vlastnosť `display: flex`. Meníme veľkosť okna s výslednou stránkou a pozorujeme (obrázok 10.17 dolu).

```
section {
  border: 2px solid #993333;
}
div {
  display: flex;
}
```



Obrázok 10.17 Zobrazenie sekcií pred (hore) a po (dolu) nastavení display: flex.

Nepoužili sme žiaden druh umiestňovania (ani cez `position` ani cez `float`), ani sme sekciám nezmenili spôsob zobrazenia na `inline`, no napriek tomu sa sekcie zobrazili **vedľa** seba. Umožňuje to nastavenie `display: flex`. Toto nastavenie definujeme pre ten element, ktorý je bezprostredne nadradený elementom, ktoré chceme umiestniť vedľa seba, t.j. vytvára pre ne akýsi kontajner. Všimnite si, že nech akokoľvek zmeníme šírku okna, výška všetkých častí v rámci flexibilného kontajnera je rovnaká.

Element s nastavením `display: flex` budeme nazývať **flexibilný box.**

ÚLOHA 10.19

V HTML kóde z *príkladu 10.18* vykonajte postupne nasledujúce zmeny.

- Doplňte ďalšiu sekciu, napr. `<section>3. stĺpec (reklama, sponzori, akcie)</section>`.
- Obsah stredného stĺpca naformátujte tak, aby text *2. stĺpec (hlavný obsah stránky)* bol nadpis a zvyšný text bol odsek. Umiestnia sa nadpis a odsek vedľa seba alebo pod seba?
- Meňte šírku „prehliadača" a všímajte si nasledujúce vlastnosti.
  - Sú jednotlivé sekcie rovnako široké? Od čoho závisí ich šírka?
  - Zaberajú všetky sekcie celú šírku stránky alebo len časť? Ak len časť, sú zarovnané vľavo, vpravo, či centrované?
  - Sú jednotlivé sekcie (stĺpce) tesne vedľa seba alebo sú medzi nimi medzery?
  - Ak je stránka príliš úzka, sú jednotlivé sekcie stále vedľa seba alebo pod sebou?

ZAPAMÄTAJTE SI

Ak chceme niekoľko elementov umiestniť vedľa seba tak, aby mali garantovanú rovnakú **výšku**, vnoríme ich do flexibilného boxu, t.j. elementu (zvyčajne `div`), s vlastnosťou `display: flex`.

Vlastnosti flexibilných boxov

Flexibilným boxom môžeme nastavovať niekoľko vlastností, ktoré ovplyvňujú spôsob rozloženia prvkov v nich. Význam a použitie niektorých z nich si ukážeme na príkladoch a úlohách v tejto časti.

PRÍKLAD 10.20

Na stránke IT Pizza zmeníme hlavičku tak, aby nadpis, logo a telefónne čísla boli na jednom riadku: nadpis vľavo, logo v strede, telefónne čísla vpravo (*obrázok 10.18*).

Obrázok 10.18 Hlavička stránky IT Pizza v troch stĺpcoch.

Využijeme flexibilný box. Budeme vychádzať zo súboru `10/index-flex.html`. Opäť budeme všetky nové vlastnosti pridávať samostatne až za riadok `/* štýly súvisiace s viacstĺpcovým rozložením */`.

Z elementu `header` spravíme flexibilný box pomocou nastavenia `display: flex`. Jeho tri prvky (`h1`, `img` a `div`) chceme rozložiť do jedného riadku tak, aby dva boli na kraji a jeden v strede. Na to použijeme vlastnosť `justify-content` s hodnotou `space-between`. V HTML časti nie je potrebné nič meniť.

```
/* štýly súvisiace s viacstĺpcovým rozložením */
header {
  display: flex;
  justify-content: space-between;
}
```

ÚLOHA 10.21

V predchádzajúcom príklade:

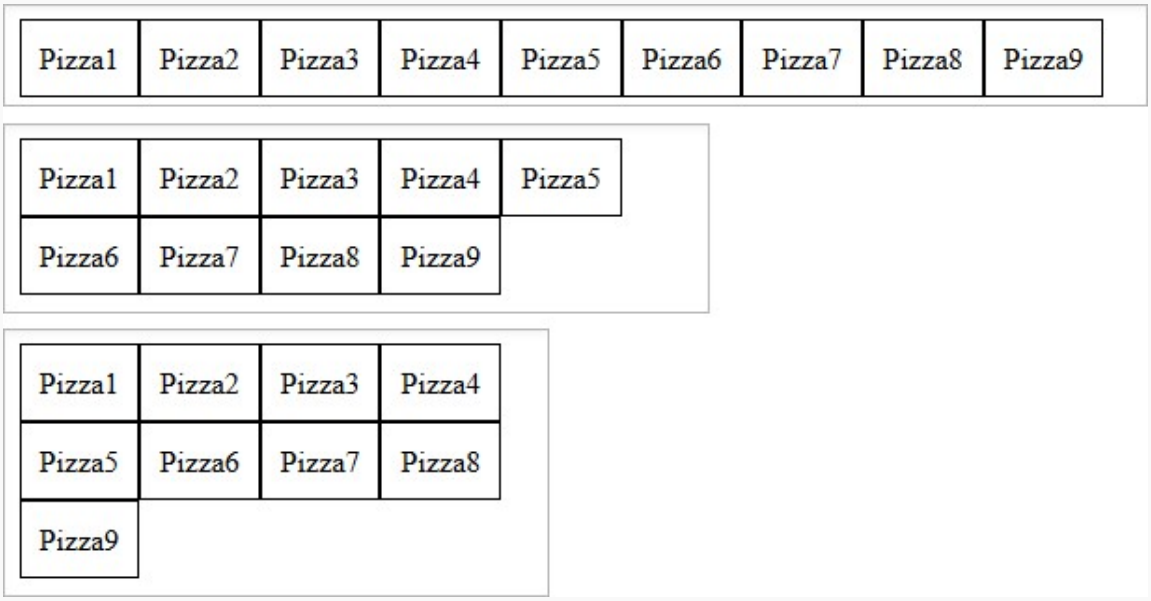
- skúšajte meniť šírku stránky a pozorujte rozloženie jednotlivých častí hlavičky,
- postupne skúšajte ďalšie hodnoty vlastnosti `justify-content`: `flex-start`, `flex-end`, `center` a `space-around`, pozorujte, ako sú rozmiestnené jednotlivé časti hlavičky pri rôznych šírkach stránky.

ÚLOHA 10.22

Zmeňte pätu na stránke IT Pizza z *príkladu 10.20* tak, aby adresy prevádzok v Bratislave, v Banskej Bystrici a kontakty na sociálne siete boli v jednom riadku, ako na *obrázku 10.13* (v *úlohe 10.13*). Využite flexibilný box a jeho vlastnosti.

PRÍKLAD 10.23

Upravíme ponuku pízz na stránke IT Pizza tak, aby sa pizze zobrazovali vedľa seba po riadkoch, pričom počet pízz v jednom riadku bude závisieť od šírky okna prehliadača (*obrázok 10.19*). Vychádzame z kódu z *príkladu 10.20*, resp. *úlohy 10.22*.

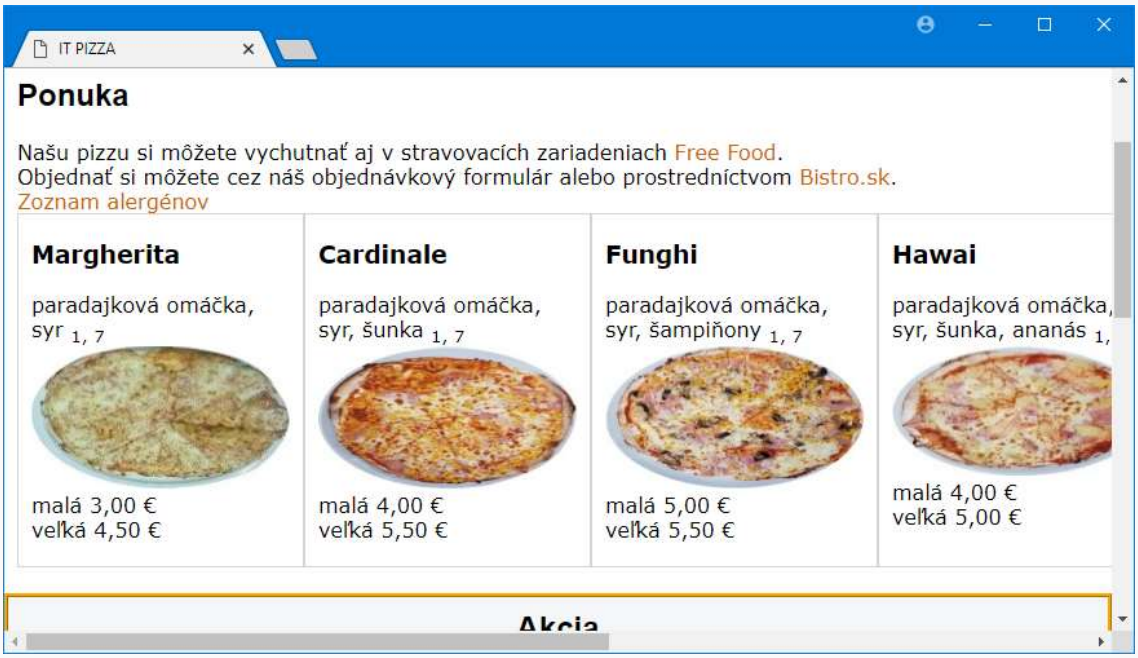


Obrázok 10.19 Schémy zobrazenia ponuky pízz pri rôznych šírkach stránky.

Najprv všetky pizze (teda všetky elementy `article` v sekcii *ponuka*) zabalíme do jedného elementu `div`, z ktorého spravíme flexibilný box. Po definovaní uvedeného štýlu vidíme (*obrázok 10.20*), že kartičky s pizzou sú rôznych šírok, zobrazujú sa všetky v jednom riadku a niektoré z nich ani nevidno.

```
/* štýly súvisiace s viacstĺpcovým rozložením */
...
#pizze {
  display: flex;
}
```

```
...
<section id="ponuka">
  <h2>Ponuka</h2>
  ...
  <div id="pizze">
    <article>
      <h3>Margherita</h3>paradajková omáčka, syr ...<br>
        <img src="obrazky/margerita.jpg" alt="Margherita"...><br>
        malá 3,00 &euro;<br>
      veľká 4,50 &euro;
      </article>
     ...
      <article>
        <h3>Tonno</h3>paradajková omáčka, mozarella, ...<br>
        <img src="obrazky/tonno.png" alt="Tonno"...><br>
        malá 4,00 &euro;<br>
        veľká 5,50 &euro;
    </article>
  </div>
</section>
```



Obrázok 10.20 Ponuka pízz: rôzne široké kartičky, všetky v jednom riadku

Potom v štýle `section article` nastavíme šírku tak, aby sa do nej zmestili všetky informácie o pizzi, a to pre ľubovoľnú pizzu (vhodnú šírku zistíme experimentovaním). Tak zabezpečíme rovnakú šírku kartičiek. Na to, aby sa kartičky zobrazovali vo viacerých riadkoch, ak sa do jedného riadku nezmestia, použijeme vlastnosť `flex-wrap` s hodnotou `wrap`.

```
/* štýly súvisiace s viacstĺpcovým rozložením */
...
#pizze {
  display: flex;
  flex-wrap: wrap;
}
section article {
  width: 380px;
}
```

ÚLOHA 10.24

So stránkou z predchádzajúceho príkladu robte postupne nasledujúce činnosti.

- Meňte šírku stránky a pozorujte, koľko pízz sa zmestí do jedného riadka.
- Vyskúšajte ďalšie hodnoty vlastnosti `flex-wrap`: `nowrap` a `wrap-reverse`. Pozorujte, ako sú rozmiestnené jednotlivé kartičky s pizzou. Skúmajte pri rôznych šírkach stránky.
- Nastavte `flex-wrap` na `wrap` a doplňte do štýlu `#pizze` vlastnosť `justify-content`. Skúšajte rôzne hodnoty `justify-content` a pozorujte rozloženie kartičiek s pizzou pri rôznych šírkach stránky.

ÚLOHA 10.25

V prehliadači zobrazte súbor `10/skumajflex.html`. Otvorte si tiež zdrojový kód súboru a prezrite si ho. V zdrojovom kóde:

- do štýlu pre `div` pridajte nastavenie `align-items`: `flex-start` a pozorujte zobrazenie jednotlivých častí flexibilného boxu, aj pri rôznych šírkach stránky,
- postupne priraďujte vlastnosti `align-items` hodnoty `flex-start`, `flex-end`, `center`, `stretch` a `baseline` a pozorujte zobrazenie častí flexibilného boxu, aj pri rôznych šírkach stránky,
- slovne popíšte, čo podľa vás vlastnosť `align-items` a jej jednotlivé hodnoty znamenajú.

ÚLOHA 10.26

V zdrojovom kóde z predchádzajúcej úlohy postupne vykonajte nasledujúce zmeny.

- Doplňte do štýlu pre flexibilný box nastavenie `flex-direction: row-reverse`. Čo sa zmenilo?
- Zmeňte hodnotu `flex-direction` na `column`.
- Zmeňte hodnotu `flex-direction` na `column-reverse`.
- Sformulujte, čo ovplyvňuje vlastnosť `flex-direction`.

Ak si chcete precvičiť vlastnosti flexibilných boxov, zahrajte sa hru na stránke flexboxfroggy.com.

Prehľad vybraných vlastností flexibilných boxov, ich hodnoty a popis uvádzame v *tabuľke 10.2*. Vždy prvá hodnota z uvedených je prednastavená.

Ukázali sme si niekoľko spôsobov ako rozložiť objekty na stránke do viacerých stĺpcov: absolútne umiestňovanie, plávajúce objekty a flexibilné boxy. Absolútne umiestňovanie je málo flexibilné a ťažšie sa dokáže prispôsobovať rôznym veľkostiam okna prehliadača, vhodnejšie je využiť plávajúce objekty či flexibilné boxy. Ďalšou možnosťou je využitie vhodného frameworku, napr.

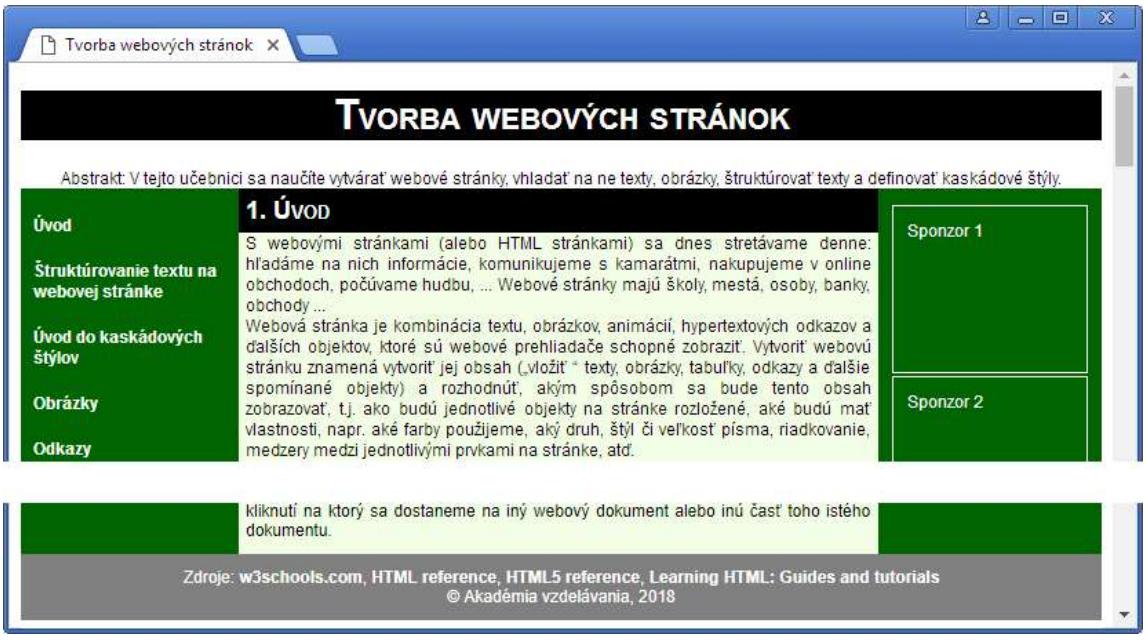
Bootstrap (https://getbootstrap.com/) alebo Foundation (https://foundation.zurb.com/). To je však už nad rámec našej učebnice.

Tabuľka 10.2 Vlastnosti flexibilných boxov.

vlastnosť	hodnoty	popis
`flex-direction`	row, row-reverse, column, column-reverse	definuje, či sa majú prvky flexibilného boxu radiť pod seba alebo vedľa seba; reverse je vždy v opačnom poradí ako poradie v HTML
`flex-wrap`	nowrap, wrap, wrap-reverse	definuje, či sa majú prvky flexibilného boxu zobrazovať v jednom riadku alebo vo viacerých riadkoch, ak sa nezmestia na šírku stránky; nemá zmysel, ak sú prvky flexibilného boxu v jednom stĺpci
`justify-content`	flex-start, flex-end, center, space-around, space-between	definuje spôsob vodorovného rozloženia prvkov flexibilného boxu
`align-items`	stretch, flex-start, flex-end, center, baseline	definuje zvislé zarovnanie prvkov flexibilného boxu

ÚLOHA 10.27

Pre stránku `10/ucebnica.html` vytvorte trojstĺpcový dizajn (*obrázok 10.21*): hore bude hlavička, pod ňou v ľavom stĺpci navigácia, v strede jednotlivé kapitoly a v pravom stĺpci logá sponzorov, dolu pod všetkým bude päta. Zvoľte si spôsob, ktorý považujete za najvhodnejší.



Obrázok 10.21 Trojstĺpcový vzhľad stránky s učebnicou.