

Popište základní značky HTML

HyperText Markup Language (HTML) je jazyk pro vytváření struktury a obsahu webových stránek.

Základem jsou značky (tags), které mohou být párové a nepárové.

```
<h1>Nadpis1</h1> <!--Párová značka, h1-h6 jsou různé velikosti nadpisů, musí být i uzavřena -->
```

```
<br> <!-- Nepárová značka, zalomení řádku, neuzavíráme -->
```

Každá značka může mít atributy, které píšeme do závorky otevíracího prvku:

```
<div class="mojeTrida">
    Lorem ipsum dolor sit amer
</div>
```

Nejčastěji používáme `class` a `id`, které můžeme používat u všech prvků. Class používáme převážně pokud máme více prvků, které mají mít stejný styl a id pokud chceme definovat styl pro jeden určitý prvek. U určitých prvků musíme použít určitý atribut. (například pro `<a>` (hypertextový odkaz) musíme použít `href="/odkaz/na/stranku"`) Každá značka může mít libovolný počet atributů.

Příklad použití HTML značek a atribut

```
/* CSS soubor definující styly */
.mojeTrida { /* Nastavuje styl pro značky s class="mojeTrida" */
    color: blue; /* Modrá barva písma */
}

#mujPrvek { /* Nastavujeme styl pro značky s id="mujPrvek" */
    background-color: #FF00FF; /* Magenta pozadí písma */
}
```

```
<!-- HTML kód -->

<!-- Tento kód vypíše -->
<div class="mojeTrida">
    Tento text bude červený
</div>
```

```
<a href="https://www.youtube.com/watch?v=dQw4w9WgXcQ">
    Úžasná písnička
</a>

<h1 id="mujPrvek">Tento nadpis bude mít Magenta pozadí</h1>
```

Výsledek tohoto kódu bude vypadat takto

Tento text bude červený
Úžasná písnička

Tento nadpis bude mít
Magenta pozadí

Příklady často používaných značek v HTML

```
<h1> Hlavní nadpis </h1>
<h6> Nadpisy pokračují až do 6. řádu </h6>

<p>
    P jako paragraph se používá pro bloky textu
</p>

<!-- Seznamy -->

<!-- <ol> je řazený seznam (ordered list), na začátku bude mít čísla dle
pořadí -->
Nejlepší školy na světě:
<ol>
    <li>SPŠ Jedovnice</li> <!-- <li> je položka seznamu -->
    <li>SPŠ Jedovnice</li>
    <li>SPŠ Jedovnice</li>
    <li>SPŠ Jedovnice</li>
    <li>Gymnázium Blansko</li>
</ol>

<!-- <ul> je neřazený seznam (unordered list), za začátku bude mít • -->
Žáci 4.B:
<ul>
```

```

    <li>Daniel Rajtšlégr</li>
    <li>Lukáš Kučera</li>
    <li>Maxim Vašíček</li>
</ul>

<!-- Kontejnery -->
<div>
    Div je blokový kontejner, sám o sobě nic nedělá. Zabírá celou šířku
    stránky.
    Je to jeden z nejpoužívanějších prvků
</div>

<span>Stejné jako div,</span> <span>akorát nezabírá celou šířku (neblokový)
</span>

<br> <!-- Zalomení řádku -->

<!-- Vložení obrázku -->


<!-- Tabulka -->
<table> <!-- Definujeme tabulku -->
    <tr> <!-- Řádek tabulky -->
        <th> Nadpis1 </th> <!--<th> (table head) je nadpis, můžeme
        použít i <td>-->
        <th> Nadpis2 </th>
        <th colspan="2"> Wide Nadpis </th> <!-- colspan = pole přes
        více sloupců -->
    </tr>
    <tr>
        <td> Obsah1 </td> <!-- <td> (table data) použijeme na obsah
        tabulky -->
        <td> Obsah2 </td>
        <td> Obsah3 </td>
        <td rowspan="2">Vysoký jalovec</td> <!-- rowspan = pole přes
        více řádků -->
    </tr>
    <tr>
        <td> Obsah5 </td>
        <td> Obsah6 </td>
        <td> Obsah7 </td>
    </tr>
</table>

```

<table> v základu nemá viditelné čáry mezi buňkami, takže potřebujeme následující CSS:

```
table, th, td {
  border: 1px solid black; /* Plná 1px široká černá čára*/
}
```

Výsledek tohoto kódu bude následující:

Hlavní nadpis

Nadpisy pokračují až do 6. řádu

P jako paragraph se používá pro bloky textu

Nejlepší školy na světě:

1. SPŠ Jedovnice
2. SPŠ Jedovnice
3. SPŠ Jedovnice
4. SPŠ Jedovnice
5. Gymnázium Blansko

Žáci 4.B:

- Daniel Rajtšlégr
- Lukáš Kučera
- Maxim Vašíček

Div je blokový kontejner, sám o sobě nic nedělá. Zabírá celou šířku stránky. Je to jeden z nejpoužívanějších prvků. Stejně jako div, akorát nezabírá celou šířku (neblokovaný)



Nadpis1	Nadpis2	Wide Nadpis	
Obsah1	Obsah2	Obsah3	Vysoký jalovec
Obsah5	Obsah6	Obsah7	

Uved'te postup tvorby statické HTML stránky a možnosti jejího publikování / Vysvětlete postup vytváření obsahu stránek

Příprava

Nejdříve je dobré si promyslet, co tvoříme a jak to chceme udělat. Nejlépe si načrtnout na papír nebo v nějakém designovém programu (např. Figma nebo Adobe XD) jak bude stránka vypadat.

Realizace stránky

Jako první je dobré udělat si strukturu souborů. Nejdůležitější soubor v každém adresáři (složce) je `index.html`, což je výchozí soubor, který se zobrazí pro daný adresář. Ve složce můžeme mít i css soubor (standardně `styles.css`) a JavaScript soubor (standardně `main.js`)

Příklad struktury

```
mujWeb
|   index.html    // https://mole.lol/
|   styles.css
|   main.js
|   kocicky.html  // https://mole.lol/kocicky.html
|
|   └─ foto
|       |   index.html    // https://mole.lol/foto/
|       |   amerika.html  // https://mole.lol/foto/amerika.html
|       |
|       └─ dovolena
|           |   index.html    // https://mole.lol/foto/dovolena/
|           |   rybicky.html  // https://mole.lol/foto/dovolena/rybicky.html
```

Dále napíšeme samotný kód stránek do HTML souborů, jejich styly do CSS souborů a funkcionálnítu do JS souborů

Nyní můžeme HTML soubor otevřít v prohlížeči.

Výběr hostingu

Hosting je služba, na které můžeme publikovat naši stránku. Pokud máme pouze frontend (tj. stránka není dynamicky generovaná, máme jen HTML, CSS a JS v prohlížeči) můžeme použít nějaký z hostingů zdarma (např. [Github Pages](#)). Pokud chceme mít i backend (dynamickou generaci stránek, databázi, perzistentní data atd.) musíme si zaplatit (většinou desítky až stovky Kč měsíčně podle parametrů).

Registrace domény

Doména je slovní název naší stránky (např. `spsjedovnice.cz` nebo `mole.lol`). U poskytovatele domén si můžeme zaregistrovat jakoukoliv dostupnou doménu a pak ji přesměrovat na naši stránku.

Vysvětlete problematiku formátování stránek včetně CSS

Cascading Style Sheets (kaskádové styly) je jazyk pro formátování webových stránek.

```
selector {  
    styl1: hodnota1;  
    styl2: hodnota2;  
}
```

Selektory

Selektory umožňují vybrat pro které prvky chceme styly aplikovat. Můžeme použít následující selektory:

```
body { /* Jméno prvku (např. body, h1, div) - aplikuje se na všechny prvky  
druhu */  
    styl1: hodnota1;  
}  
  
.jmenoTridy { /* Jméno třídy - aplikuje se na všechny prvky s  
class="jmenoTridy"*/  
    styl1: hodnota1;  
}  
  
#idPrvku { /* ID prvku - aplikuje na prvek s daným ID */  
    styl1: hodnota1;  
}  
  
body h1 { /* Selektory může vrstvit - např. všechny h1 v body */  
    styl1: hodnota1;  
}
```

Pseudo třídy (body:hover)

K selectoru můžeme přidávat i pseudo třídy, které definují nějakou podmínku.

[Seznam všech pseudo tříd](#)

```
button:hover { /* Najetí kurzoru na objekt */  
    background-color: blue;  
}  
  
.obrazek:first { /* První výskyt na stránce */  
    background-color: blue;  
}
```

Deklarace

Deklarace umožňují aplikovat určitý styl na vybraný prvek

```
body {  
    color: blue; /* Barva písma */  
    background-color: #00ffff; /* Barva pozadí */  
    font-family: Georgia, serif; /* Písmo */  
    margin: 5px; /* Margin viz box model */  
    padding: 10%; /* Padding viz box model */  
}
```

Připojení CSS do HTML dokumentu

V HTML <head>

CSS můžeme napsat přímo do <head> dokumentu. Stačí do <head> vložit <style> a do něj dát přímo CSS kód:

```
<head>  
    <style>  
  
        .modra {  
            background-color: blue;  
        }  
  
    </style>  
</head>
```

Jako atribut

CSS můžeme také vložit přímo do značky následovně:

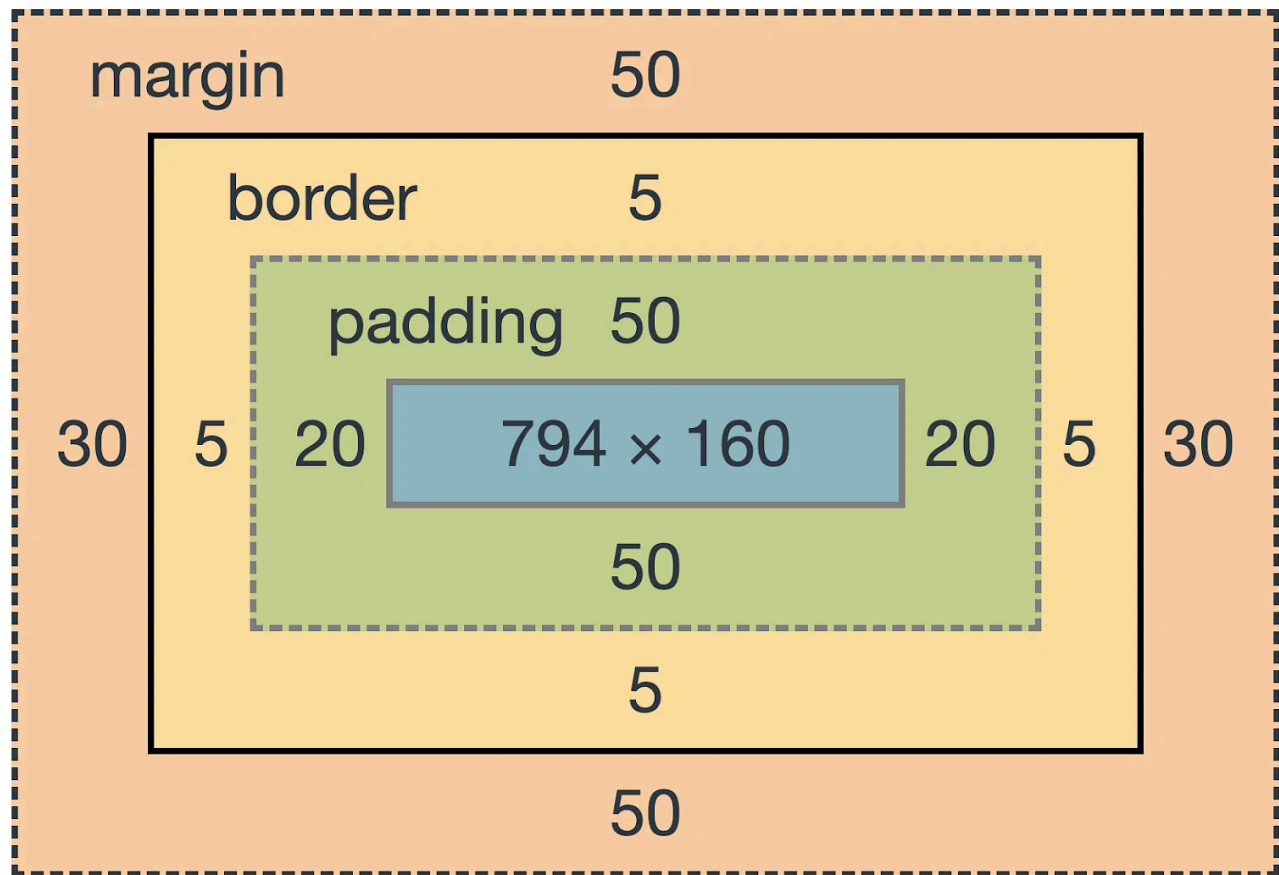
```
<div style="background-color: blue;">  
    Modrá je dobrá  
</div>
```

V externím souboru

CSS můžeme importovat v <head> z externího CSS souboru

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

Box model



CSS Používá box model pro Padding (odsazení uvnitř prvku), Border (obrys prvku) a Margin (Odsazení vně prvku). Můžeme je definovat i pro každou stranu zvlášť (např. `padding-left`)

Media query (responsivita)

Responzivní design stránky umožňuje podporu různých zařízení (můžeme např. stránku změnit při úzké obrazovce mobilu, aby se lépe používala). Na to používáme Media query, které nám umožňují kondicionálně (při splnění nějaké podmínky) přidávat styly.

```
/* Text v body zčervená, pokud má zařízení méně než 600px na šířku*/  
@media only screen and (max-width: 600px) {  
  body {  
    color: red;  
  }  
}
```

Animace a přechody

Pomocí CSS můžeme vytvářet přechody a animace. Přechody se automaticky spouští při změně vlastností (např. přes pseudo třídu nebo JavaScript)

```
/* Příklad přechodu */
div {
    transition: width 2s;
}
```

Animace používají několik přechodů jako snímky:

```
/* Kód animace */
@keyframes mojeAnimace {
    from {
        background-color: red;
    }
    to {
        background-color: yellow;
    }
}

/* Aplikace animace */
div {
    background-color: red;
    animation-name: mojeAnimace;
    animation-duration: 4s;
}
```

Grid a flex

Alternativní způsoby rozložení. Grid je moderní náhrada za tabulku. Stačí tuto vlastnost nastavit kontejneru.

```
.container {
    display: grid; /* Definujeme, že je to grid */
    column-gap: 50px; /* Mezera mezi sloupci */
    row-gap: 50px; /* Mezera mezi řádky */
}
```

Flex umožňuje dynamicky řadit prvky (například různý počet prvků zarovnávat na střed a v případě potřeby zalamovat)

```
.flex-container {
    display: flex; /* Definujeme, že je to flex */
}
```

```
flex-wrap: wrap; /* Zapneme zalamování řádků */
flex-direction: row; /* Prvky se řadí do řádků */
}
```

!important

Za deklaraci můžeme přidat `!important`, což přepíše všechny ostatní deklarace stejné vlastnosti (např. ID má přednost přes classou, `important` přebije vše).

```
#podleId {
    background-color: blue;
}













.podleTridy {
    background-color: gray;
}

p {
    background-color: red !important;
}
```

Výsledek bude následující:

```
<p class="podleTridy" id="podleId"> Stejně budu červený kvůli important </p>
```



 <p>a</p> <p>1 x element selector</p> <p>Sith: 0, 0, 1</p>	 <p>p a</p> <p>2 x element selectors</p> <p>Sith: 0, 0, 2</p>	 <p>.whatever</p> <p>1 x class selector</p> <p>Sith: 0, 1, 0</p>	 <p>a.whatever</p> <p>1 x element selector 1 x class selector</p> <p>Sith: 0, 1, 1</p>
 <p>p a.whatever</p> <p>2 x element selectors 1 x class selector</p> <p>Sith: 0, 1, 2</p>	 <p>.whatever .whatever</p> <p>2 x class selectors</p> <p>Sith: 0, 2, 0</p>	 <p>p.whatever a.whatever</p> <p>2 x element selectors 2 x class selectors</p> <p>Sith: 0, 2, 2</p>	 <p>#whatever</p> <p>1 x id selector</p> <p>Sith: 1, 0, 0</p>
 <p>a#whatever</p> <p>1 x element selector 1 x id selector</p> <p>Sith: 1, 0, 1</p>	 <p>.whatever a#whatever</p> <p>1 x element selectors 1 x class selector 1 x id selector</p> <p>Sith: 1, 1, 1</p>	 <p>.whatever .whatever #whatever</p> <p>2 x class selectors 1 x id selector</p> <p>Sith: 1, 2, 0</p>	 <p>#whatever #whatever</p> <p>2 x id selectors</p> <p>Sith: 2, 0, 0</p>

CSS Frameworky

Aktuálně se už čisté CSS moc nepíše. Převážně ho nahrazují frameworky, které se píší přímo do class. Tím pádem není potřeba mít zvláštní CSS soubor. Nejznámější jsou Tailwind a Bootstrap.

```
<!-- Tailwind -->
<div class="flex items-baseline space-x-1">
  <span class="text-lg font-semibold">20.1.2025</span>
  <span class="text-lg font-semibold">18:00</span>
</div>
```

Popište možnosti použití skriptovacího jazyka JavaScript (JS)

JS je vysokoúrovňový interpretovaný jazyk. (vysokoúrovňový = velká abstrakce od kódu procesoru, interpretovaný = není předem kompilovaný do kódu procesoru, kompiluje se řádek po řádku za běhu). Jazyk je objektově orientovaný, ale lze ho psát funkcionálně (což je častější). V základu nemá statické tipování proměnných (pokud chceme staticky tipovat, musíme použít TypeScript, což je v praxi velice časté). Nejběžnější použití je tvorba interaktivních webových stránek, ale lze ho použít i pro desktopové ([Electron](#)) a mobilní ([React Native](#)) aplikace. Pomocí [NodeJS](#) nebo [Bun](#) můžeme spouštět JS i na serveru. Je podporován všemi moderními prohlížeči. Některé prohlížeče mají větší a menší podporu pro různé JavaScript API (application interface) ([seznam podpory](#)). Moderní použití JS silně závisí na importu knihoven kódu, ze kterých můžeme importovat funkce ([package manager NPM](#)). Umožňuje manipulovat s obsahem v reálném čase přímo v prohlížeči uživatele (kód se spouští přímo v prohlížeči uživatele v sandboxu - odděleně od zbytku jeho počítače). Můžeme reagovat na události vyvolané uživatelem (např. klikání myší, mačkání kláves). Můžeme také dynamicky upravovat stránku na základě získávání dat ze serveru (používáme funkce `fetch`, dříve `XMLHttpRequest`). Tímto můžeme implementovat AJAX (Asynchronous JavaScript and XML).