JavaScript DOM

Lili Nemec Zlatolas

Anotácia

Tento kurz predstavuje objektový model dokumentu v jazyku JavaScript.

Ciele

Kurz poskytuje rýchly prehľad o potrebných znalostiach základov HTML a JavaScriptu. Študenti sa zoznámia s objektovým modelom dokumentu v jazyku JavaScript.

Kľúčové slová

JavaScript, DOM, HTML

Dátum vytvorenia

15.4.2022

Časová dotácia

12 hodín

Jazyková verzia

slovensky

Licencia

ISBN

Literatúra

[1] Matt Frisbie. Professional JavaScript for Web Developers. Publishing place: John Wiley & Sons, 2019. 978-1-119-36644-7.

[2] R. Ferguson. Beginning JavaScript: The Ultimate Guide to Modern JavaScript Development. Apress, Ocean, 2019. 3rd edition.

[3] M. Haverbeke. Eloquent JavaScript - A Modern Introduction to Programming. Third Edition. No Starch Press, San Francisco, 2018.

[4] W3schools. Javascript HTML DOM. https://www.w3schools.com/js/js_htmldom.asp.

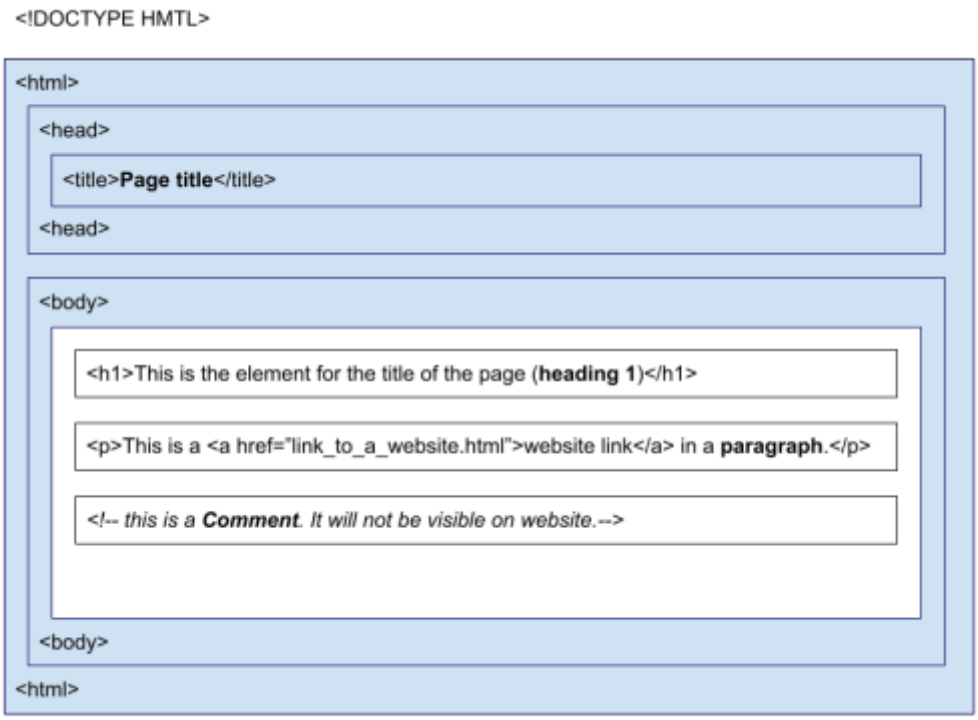
KAPITOLA 1

Základy jazyka HTML

K manipulácii s webovými stránkami pomocou objektového modelu dokumentu (Document Object Model – DOM) sú potrebné predošlé znalosti jazyka HTML a jazyka JavaScript. HTML je značkovací jazyk, ktorý popisuje štruktúru webových stránok. Jazyk HTML sa skladá z prvkov, ktoré sú označené značkami (angl. tag) a posielajú prehliadaču informácie o tom ako má zobraziť obsah dokumentu.

Posledným prijatým štandardom je HTML5. HTML sa skladá z elementov, ktoré sú definované otváracou značkou, obsahom prvku a uzatváracou značkou: <tag> Obsah </tag>

Tu je príklad HTML dokumentu:



Obr. 1. Štruktúra stránky HTML a niektoré prvky

K úprave dokumentu môžete použiť Poznámkový blok alebo podobné programy (napr. Notepad++, Visual Studio Code). Dokument HTML musí mať príponu .html.

HTML nerozlišuje veľké a malé písmena, ale odporúča sa používať malé písmena v značkách HTML.

Tabuľka 1. Niektoré základné prvky jazyka HTML

Značka	Popis
`<! DOCTYPE html>`	Toto je deklarácia HTML5 dokumentu, ktorá má byť na začiatku dokumentu HTML.
`<h1>`	V HTML sú nadpisy od <h1> do <h6>, pričom prvý je ten najväčší.
`<p>`	Tato značka označuje odstavec.
` `	Tato značka slúži k prechodu na ďalší riadok (odriadkovanie).
`<a href="https://www.websitelink.com">`	Táto značka je určená pre odkaz na webovou stránku. Používa atribúty.
`<img src="image_location.jpg" alt="image_description">`	Táto značka slúži k vloženiu obrázka. Používa tiež rôzne atribúty.
`<b>`	Tato značka slúži k zobrazeniu tučného písma v textu.

Niektoré prvky sa nazývajú nepárové prvky. Príkladom nepárového prvku je
, ktoré nemá obsah ani uzatváraciu značku. Pokiaľ však použijeme pravidlá XHTML, môže obsahovať uzatváraciu značku v rámci otváracej značky:

Značky HTML môžu mať atribúty. Niektoré prvky nemajú bez atribútov žiadne funkcie. Atribúty sa uvádzajú v otváracích značkách. Jedným z často používaných atribútov je style, ktorý je možné použiť napríklad v odstavci:

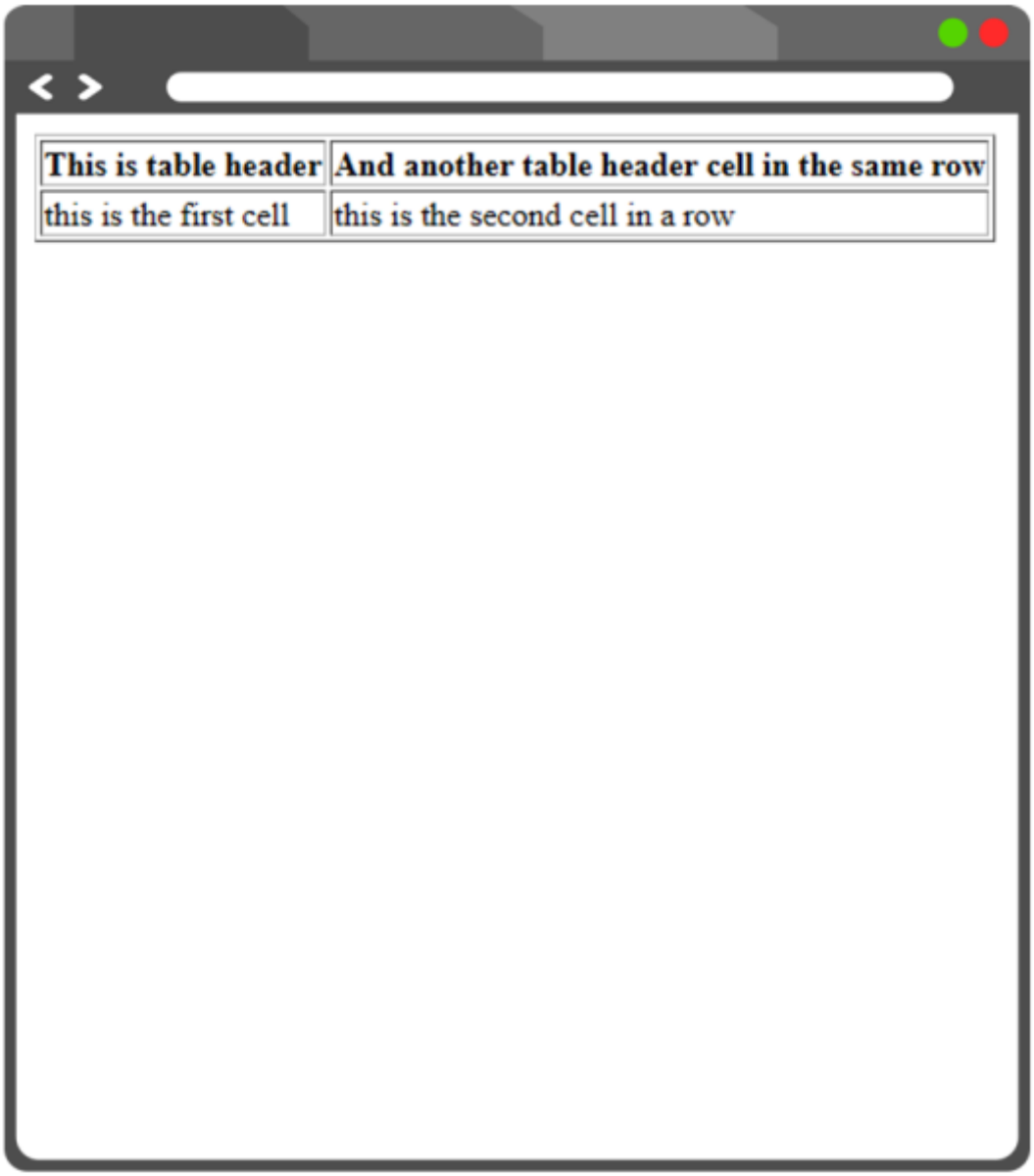
```
<p style="color:blue;"> Tento text bude modrý. </p>
```

Interaktivní prvek

V HTML sa často používajú tabuľky, ktoré sa skladajú z buniek v rámci stĺpcov a riadkov tabuľky. Jednoduchá HTML tabuľka vypadá nasledovne:

PRÍKLAD

```
<table border="1">
  <tr> <!--Toto je riadok tabulky ktory nema ziadny obsah -->
    <th> Toto ja hlavicka tabulky </th>
    <th> Toto je dalsia hlavicka bunky v tom istom riadku </th>
  </tr>
  <tr>
    <td>toto je prva bunka</td>
    <td>toto je druha bunka v rovnakom riadku</td>
  </tr>
</table>
```



Obr. 2. Ako vypadá tabuľka v prehliadači.

Interaktivní prvek

Pomocou značiek možno v HTML vytvárať usporiadané alebo neusporiadané zoznamy. Usporiadané zoznamy sú obvykle číslované a neusporiadané zoznamy majú obvykle bodové odrážky.

Tabuľka 2. Príklad zoznamu v HTML

Príklad zoznamu v jazyku HTML	Zobrazenie v prehliadači

`<ul>`
`<li>Pes</li>`
`<li>Mačka</li>`
`<li>Ryba</li>`
`</ul>`
	• Pes
	• Mačka
	• Ryba

  - Pes - Mačka - Ryba 	1. Pes 2. Mačka 3. Ryba

Interaktivní prvek

Toto bol krátky úvod do jazyka HTML. V nasledujúcej kapitole sa budeme venovať základom syntaxe jazyka JavaScript.

KAPITOLA 2

Základy jazyka JavaScript

JavaScript je programovací jazyk, ktorý umožňuje zvýšiť interaktivitu a dynamickosť webových stránok. Používa sa v jazyku HTML medzi značkami <script> a </script> buď v časti <body>, alebo <head>, prípadne v obidvoch častiach.

Kód JavaScriptu možno tiež umiestniť do súboru .html ako externý súbor s príponou .js. Napríklad:

```
<script src="JavascriptFile.js"></script>
```

Existuje niekoľko možností ako zobraziť dáta v jazyku JavaScript. Môžeme použiť:

innerHTML (napr.. document.getElementById(id))

- Týmto metódam sa budeme venovať v ďalších kapitolách, pretože sa jedná o DOM.

document.write()

- Tento postup sa obvykle používa pri testovaní, pretože z dokumentu odstráni všetky existujúce HTML a zobrazí iba obsah skriptu.

window.alert()

- Vytvorí okno upozornenia pre zobrazenie dát. Výraz window možno vynechať.

console.log()

- Obvykle sa používa pre účely ladenia a umožňuje zobraziť dáta v prehliadači.

Jednotlivé príkazy sa Javascripte oddeľujú bodkočiarkou (;).

Bloky kódu sú ohraničené (uzavreté) prostredníctvom zložených zátvoriek {}.

Interaktivní prvek

Na označenie jednoriadkového komentára sa používa //, viacriadkový komentár sa označuje znakmi /* na začiatku a */ na konci.

Niektoré z najdôležitejších kľúčových slov v JavaScripte sú uvedené v tabuľke. Uvedené vyhradené slová nemôžu byť použité ako názvy premenných.

Tabuľka 3. Vyhradené slova v jazyku JavaScript

Vyhradené slovo	Popis
`var`	Deklaruje premenú

`let`	Deklaruje blokovú premennú, ktorú možno meniť
`const`	Deklaruje premennú, ktorú nie je možné meniť
`return`	Ukončí funkciu
`if`	Začiatok kódu podmienky
`for`	Začiatok kódu slučky
`function`	Deklaruje funkciu

Interaktivní prvek

Syntax sú pravidlá JavaScriptu, podľa ktorých sú programy konštruované. Najskôr sa deklarovaním musia premenné vytvoriť a až následne sa môžu v programe použiť.

Desatinné **čísla** sa oddeľujú **čiarkou**. **Texty** sa zapisujú do **jednoduchých** alebo **dvojitých úvodzoviek**. **Znamienka rovnosti** priraďujú premenným hodnoty.

Názvy premenných musia začínať písmenom abecedy (A-Z), znakom dolár alebo podčiarkovníkom. **Rozlišujú sa veľké a malé písmena.**

Ukážka definície rôznych premenných:

```
let prveCislo;
var prvyText;
prveCislo = 13;
var druheCislo = 17;
prvyText = "Toto je číslo 13."
```

Tabuľka 4. Znamienka rovnosti v JavaScripte

Rovnítko	Popis
=	Ide o operátor priradenia (napr. **var a = a * 2;**) a nemá rovnaký význam ako algebraický znak.
==	Ide o operátor rovnosti, ktorý sa používa v algebre, napr. v prípade 7+2=9, a v JavaScripte sa používa napr. ako **if (x == 2) {};**
===	Jedná sa o rovnakú hodnotu a tiež rovnaký dátový typ.

Pomocou = a + môžete spojiť alebo vypočítať hodnotu premennej.

```
let celeMeno = "Meno" + " " + "Priezvisko";
```

Interaktivní prvek

Jazyk JavaScript má 3 logické operátory:

- && logické a
- || logické alebo

- ! negácia

Funkcie sú bloky kódu, ktoré je potrebné zavolať, aby sa vykonali. Môžu mať viac parametrov a ich kód sa nachádza v zložených zátvorkách. Sú užitočné, ak danú funkciu použijeme viackrát s rôznymi argumentmi. Príklad funkcie:

PRÍKLAD

```
function circle_area(a, b)
{
return a * a * b;  // Funkcia vracia sucin A, A a B. Ak chceme vypocitat
plochu kruhu, namiesto B musime zadat PI.
}
let area = circle_area (15, Math.PI);  // Funkcia je volaná s dlzkou
15 cm a cislom PI.
document.write("Plocha kruhu je" + area + " m2.");
```

Pri spustení vyššie uvedenej funkcie sa v prehliadači zobrazí: "**Plocha kruhu je 706.8583470577034 m2.**"

V JavaScriptu sa toho môžeme naučiť oveľa viac, ale toto je niekoľko základných a dôležitých vecí, ktoré potrebujeme pre prácu s DOM v JavaScripte.

KAPITOLA 3

Úvod do objektového modelu dokumentu (DOM)

Prehliadač pri načítaní webovej stránky vytvorí objektový model dokumentu (DOM). DOM je štandard konzorcia W3C (World Wide Web Consortium) pre prístup k dokumentom.

Model HTML DOM je vytvorený ako strom objektov.

V animácií je uvedený príklad nasledujúceho kódu.

```
<html>
  <head>
    <title> This is page title ( Toto je názov stránky). </title>
  </head>
<body>
  <h1 align="left"> This is the heading ( Toto je nadpis). </h1>
  <p style="background-color: blue">This is a paragraph ( Toto je odsek). </p>
  <img src="image. jpg" />
</body>
</html>
```

Interaktivní prvek

Animácia 1. Strom objektov v modeli HTML DOM

Strom má **uzly** pre elementy, ktoré predstavujú značky HTML a určujú štruktúru dokumentu. Štandard je navrhnutý tak, že najprv vytvoríme uzol, potom k nemu pridáme potomka a k nemu atribúty. Preto môže byť kód pomerne dlhý.

V animácií je <html> *koreňový uzol* bez nadradeného uzla, ale je *nadradený* prvému *podradenému uzlu* <head> a *poslednému podradenému uzlu* <body>.

JavaScript môže pristupovať k modelu DOM a meniť prvky a atribúty HTML stránky.

V modeli DOM sú všetky prvky HTML definované ako **objekty**.

DOM je štandard, ktorý obsahuje informácie o tom, ako sa dostávame k prvkom HTML, ako ich meníme, pridávame alebo odstraňujeme v dokumentoch HTML. DOM používa **metódy** pre prístup k prvkom HTML a vykonávaniu akcií s nimi.

Interaktivní prvek

3.1 Navigácia v DOMe – prechádzanie stromovej štruktúry

Uzly v strome uzlov majú hierarchický vzťah, čo znamená, že každý uzol má presne jedného predka, okrem prvého uzla, ktorý nemá žiadneho predka. Uzol môže mať viac potomkov. Súrodenecké uzly sú uzly s totožným predkom.

Pro navigácií medzi uzlami pomocou JavaScriptu môžeme použiť následujúce vlastnosti uzlu:

- parentNode – nadradený uzol (predok)
- childNodes[nodenumber] – výber uzla zo zoznamu podradených uzlov
- firstChild – prvý podradený uzol (potomok)
- lastChild – posledný podradený uzol (potomok)
- nextSibling – nasledujúci uzol na rovnakej úrovni
- previousSibling – predošlý uzol na rovnakej úrovni

Animácia 2. Príklad uzla v dokumente

Vlastnosť uzla *childNodes[0]* je rovnaká ako *firstChild*. Obsahuje objekt podobný poľu s vlastnosťou *length* (dĺžka/počet prvkov v poli), pomocou ktorej pristupuje k podradeným uzlom.

Interaktivní prvek

Uzlami a spôsobmi získavania obsahu z prvkov sa budeme podrobnejšie zaoberať v kapitole Uzly DOM.

KAPITOLA 4

Metódy modelu DOM

Pomocou **metód** modelu DOM môžeme vykonávať činnosti s prvkami HTML. **Vlastnosti** modelu DOM sú hodnoty prvkov HTML, ktoré môžeme nastavovať alebo meniť.

PRÍKLAD

```
<p id="example">Toto sa na stránke nezobrazí, pretože Javascript prepíše
text vo vnútri uzla odseku. </p>
<script>
document.getElementById("example").innerHTML = "Toto je text, ktorý sa
zobrazí na webovej stránke. ";
</script>
V tomto príklade sa v prehliadači pri spustení kódu zobrazí „Toto je
text, ktorý sa zobrazí na webovej stránke. “.
```

V tomto príklade je getElementById **metóda** a innerHTML je **vlastnosť**. Veľmi často sa pre vyhľadávanie prvku v dokumente používa id prvku (v príklade id="example"). V iných prípadoch môžeme pre prístup k určitým prvkom použiť nadradené a podradené uzly modelu DOM.

Vlastnosť innerHTML slúži k nastaveniu alebo vráteniu obsahu HTML elementu, vo vyššie uvedenom príklade sa jedná o zmenu textu v rámci značky <p> s parametrom id="example".

Tabuľka 5. Metódy vyhľadávania, zmeny, pridávania a odstraňovania prvkov HTML

Metoda	Popis
document.getElementById(id)	Vyhľadanie prvku podľa id prvku
document.getElementsByTagName(name)	Vyhľadanie prvku podľa názvu značky
document.getElementsByClassName(name)	Vyhľadanie prvku podľa názvu triedy
element.setAttribute(attribute, value)	Zmena hodnoty atribútu prvku HTML
document.createElement(element)	Vytvorenie prvku HTML
document.removeChild(element)	Odstránenie podradeného prvku HTML
document.appendChild(element)	Pridanie podradeného prvku HTML
document.replaceChild(new, old)	Nahradenie podradeného prvku HTML

Interaktivní prvek

Video 1. Príklady získania prvku podľa X

Interaktivní prvek

KAPITOLA 5

Uzly DOM

Všetky **prvky HTML, ich atribúty a texty sú uzly**. Niektoré prvky obsahujú aj iné uzly.

Navigácia medzi uzlami už bola opísaná v kapitole Navigácia v DOMe.

Tu je príklad, ako môžeme získať prístup k hodnotám uzlov.

```
<p id='paragaph'>Toto je prvý odstavec</p>
```

Prvok <p> obsahuje **textový uzol** s hodnotou "Toto je prvý odstavec". K textovej hodnote je možné pristupovať pomocou vlastnosti uzla **innerHTML**:

```
textFromParagraph = document.getElementById('p').innerHTML;
```

To isté možno dosiahnuť prístupom k hodnote **nodeValue**:

```
textFromParagraph
= document.getElementById('p').childNodes[0].innerHTML.nodeValue;
```

Video 2. Podradené uzly a hodnoty uzlov

Koreňové uzly majú prístup k celému dokumentu:

- document.body – Obsah dokumentu
- document.documentElement – Celý dokument

Vlastnosť **nodeValue** určuje hodnotu uzlu. Pre uzly prvkov je síce nulová, ale je užitočná pre textové uzly, kde predstavuje samotný text. Pre uzly atribútov vlastnosť nodeValue vracia hodnotu atribútu.

Vlastnosť **nodeName** určuje názov uzla a je určená iba na čítanie. nodeName uzla prvku vracia názov tagu a vracia názov atribútu uzla. nodeName textového uzla je *#text* a uzla dokumentu je *#document*.

V ďalších podkapitolách budeme používať niektoré z nasledujúcich metód pre vytváranie, odstraňovanie a nahrádzanie prvkov DOM HTML (uzlov). V tejto tabuľke sa zoznámite s metódami na vytváranie prvkov a textových uzlov.

Tabuľka 6. Metódy vytvárania nových prvkov DOM HTML

Metoda	Popis
createElement(type)	Vytvorí uzol prvku s konkrétnym typom (napr. typ "p").
createTextNode()	Vytvorí textový uzol.

V nasledujúcej tabuľke sú uvedené metódy vytvárania, odstraňovania a nahrádzania uzlov. Obvykle musíme vytvoriť prvok, potom v ňom vytvoriť textový uzol a ten pridať do existujúcej štruktúry. Príklady použitia sú uvedené v ďalších podkapitolách.

Tabuľka 7. Metódy vytvárania, odstraňovania a nahrádzania uzlov

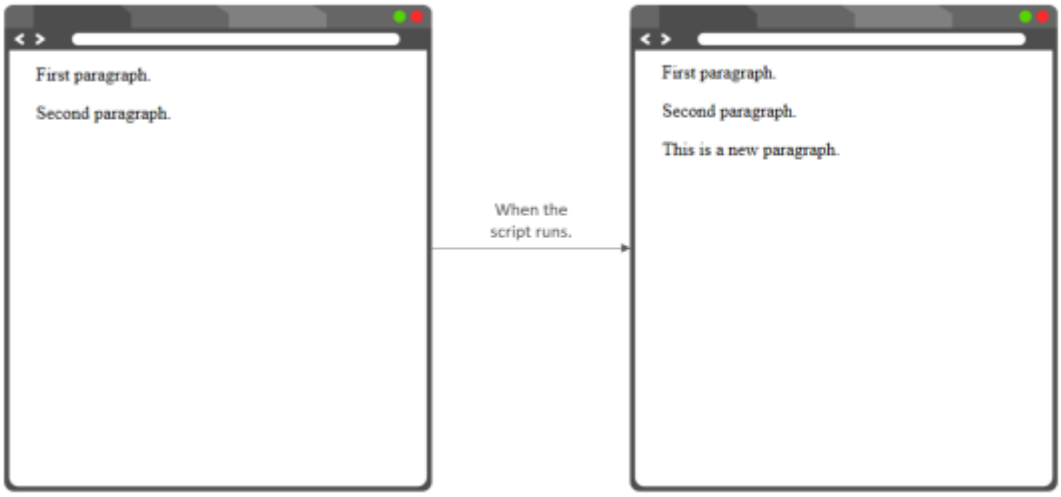
Metoda	**Popis**
appendChild(node)	Pridá k prvku nový podradený prvok (uzol).
insertBefore(new, existing)	Vloží podradený uzol pred existujúci podradený
replaceChild(new, old)	Nahradí podradený uzol novým uzlom.
remove()	Odstráni prvok z dokumentu. Nemá žiadne parametre.
removeChild(node)	Odstráni podradené prvky.

5.1 Vytváranie prvkov DOM HTML (uzlov)

V tejto kapitole sa zameriame na to ako môžeme pridávať, odstraňovať alebo nahrádzať prvky HTML DOM.

Ak chceme pridať nový prvok, musíme najprv vytvoriť uzol prvku a potom ho pridať k existujúcemu prvku. Tu je príklad kódu, v ktorom sme pridali nový odsek s podradenými uzlami.

```
<div id="div01">
 <p id="p01">Prvý odstavec.</p>
 <p id="p02">Druhý odstavec.</p>
</div>
<script>
var newParagraph = document.createElement("p");
var textForParagraph = document.createTextNode("Toto je nový odstavec.");
newParagraph.appendChild(textForParagraph);
var element = document.getElementById("div01");
element.appendChild(newParagraph);
</script>
```



Obr. 3. Ako vyzerá kód v prehliadači bez skriptu a so skriptom.

Metóda **createElement("p")** vytvorí nový prvok <p>. Ak chceme pridať text k tomuto prvku, musíme vytvoriť textový uzol pomocou metódy **createTextNode**. Potom musíme textový uzol pridať do prvku <p> pomocou metódy **appendChild**. Nakoniec musíme nový prvok pridať k existujúcemu prvku div.

Teraz použijeme rovnaký príklad, ale odsek dáme na prvú pozíciu pomocou metódy **insertBefore** namiesto poslednej pozície, ako sme to urobili pri metóde **apendChild**.

Video 3. Príklad použitia metódy insertBefore

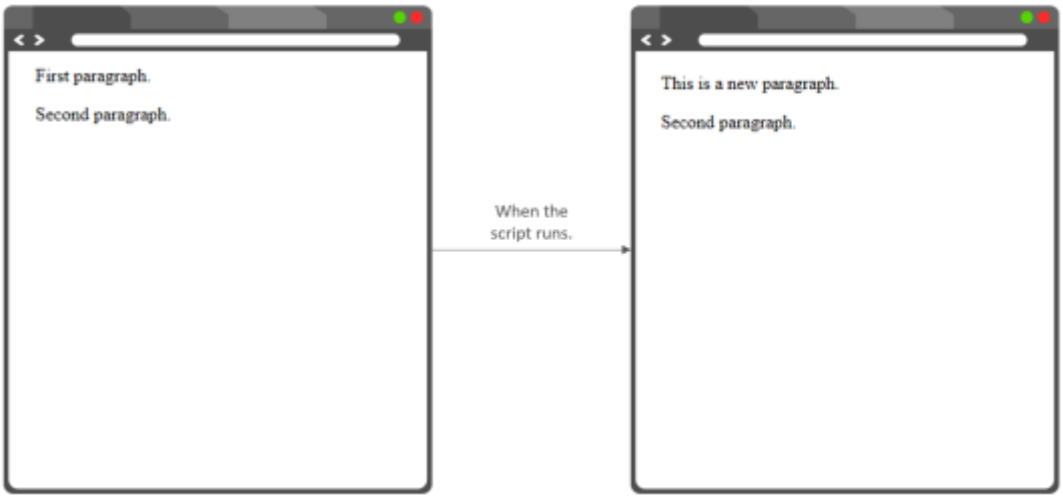
Interaktivní prvek

5.2 Nahradenie prvkov DOM HTML (uzlov)

Ak chceme nahradiť prvok, musíme najprv vytvoriť uzol prvku a potom ho nahradiť už existujúcim prvkom. Tu je príklad kódu, v ktorom sme nahradili existujúci odstavec novým odstavcom.

PRÍKLAD

```
<div id="div01">
 <p id="p01">Prvý odstavec.</p>
 <p id="p02">Druhý odstavec.</p>
</div>
<script>
var newParagraph = document.createElement("p");
var textForParagraph = document.createTextNode("Toto je nový odstavec.");
newParagraph.appendChild(textForParagraph);
var parent = document.getElementById("div01");
var child = document.getElementById("p01");
parent.replaceChild(newParagraph,child);
</script>
```



Obr. 4. Ako vyzerá kód v prehliadači bez skriptu a so skriptom.

Interaktivní prvek

5.3 Odstránenie prvkov DOM HTML (uzlov)

V tejto kapitole si ukážeme animáciu, v ktorej odstránime jeden prvok pomocou metódy remove() a jeden prvok odstránime pomocou prístupu k rodičovi pomocou metódy removeChild(). Metóda remove() nefunguje v starších prehliadačoch, preto niekedy musíme použiť metódu removeChild().

V príkaze removeChild musíme hľadať rodiča, aby sme mohli odstrániť potomka.

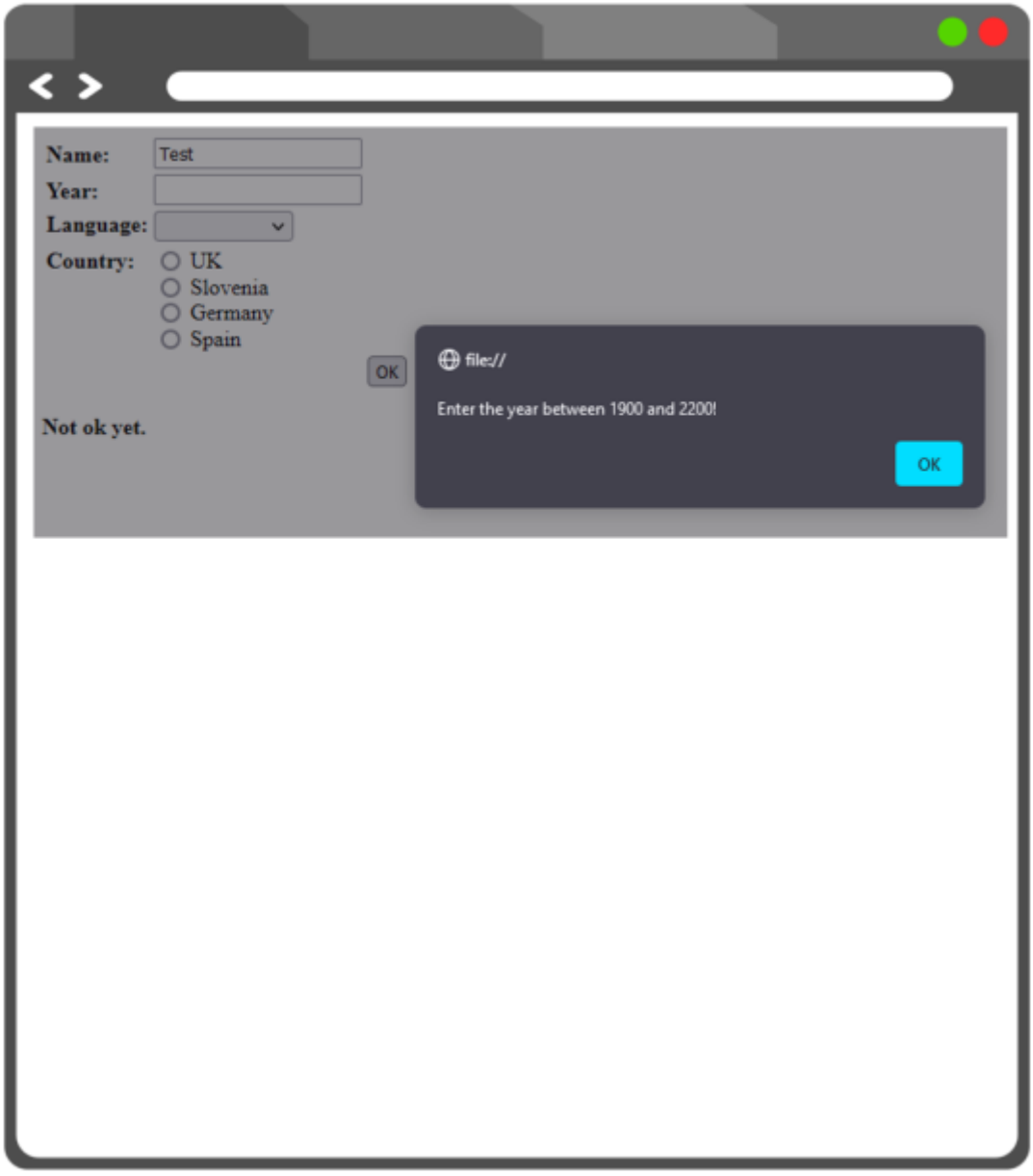
Animácia 3. Dva príklady odstránenia prvkov HTML DOM

Interaktivní prvek

KAPITOLA 6

Overovanie formulárov s využitím DOM

Overenie HTML formulára môžeme robiť pomocou JavaScriptu s využitím DOM. Zvyčajne chceme skontrolovať, či používateľ vyplnil správne údaje v správnom formáte. V tomto príklade vytvoríme jednoduchý formulár a pomocou funkcie DOM skontrolujeme, či používateľ zadal správne údaje.



Obr. 5. Príklad upozornenia a formulára, ktorý bude uvedený v nasledujúcom kóde.

Vytvoríme formulár so 4 rôznymi možnosťami zadávania údajov. Pre lepšie zobrazenie sme použili tabuľku HTML.

Vo formulári sme použili textové pole, výber z možností, výber pomocou tlačidla typu radio button a tlačidlo odoslať.

```
<form onsubmit="return validation( this)" action="#">
<table>
<tr>
<td><b>Name: </b></td>
<td><input type="text" name="name"></td>
</tr>
<tr>
<td><b>Year: </b></td>
<td><input type="text" name="year"></td>
</tr>
<tr>
<td valign="top"><b>Language: </b></td>
  <td>
  <select name="language">
<option> </option>
<option value="slo">english</option>
<option value="ang">slovenian</option>
<option value="nem">german</option>
<option value="fra">french</option>
</select>
</td>
</tr>
<tr>
<td valign="top"><b>Country: </b></td>
<td>
<input type="radio" name="country" value="1"> UK<br/>
<input type="radio" name="country" value="2"> Slovenia<br/>
<input type="radio" name="country" value="3"> Germany<br/>
<input type="radio" name="country" value="4"> Spain
  </td>
</tr>
<tr>
<td></td>
<td></td>
  <td><input type="submit" value="OK"></td>
</tr>
</table>
</form>
<script type="text/javascript">
```

Tu je vytvorená overovacia funkcia. Ak používateľ niečo nevyplní správne, zobrazí sa okno s upozornením.

```
function validation( form)
{
```

Pomocou DOM sme získali prístup k formuláru, položke meno a k jej hodnote. Pokiaľ je tento údaj prázdny, zobrazí sa upozornenie.

```
if ( form. name. value == "")
{
alert( "Enter the name! ")
return false
}
```

Pokiaľ je zadaný rok mimo rozsah 1900 a 2200, zobrazí sa upozornenie. Pomocou funkcie isNaN tiež skontrolujeme, či používateľ vôbec zadal nejaké číslo.

```
if ( isNaN( form. year. value) ||  form. year. value < 1900 ||  form. year. value >
2200)
{
alert( "Enter the year between 1900 and 2200! ")
return false
}
```

Nasledujúci kód zobrazí upozornenie, ak používateľ nevybral žiadny z jazykov v rozbaľovacom zozname jazykov.

```
if ( form. language. selectedIndex <= 0)
  {
alert( "Select the language! ")
return false
  }
```

A nakoniec, ak používateľ pomocou prepínača nevybral žiadnu z uvedených krajín, zobrazí sa mu výstražné okno.

```
var i = 0
while ( i < form. country. length && ! form. country[ i]. checked)
++i

if ( i == form. country. length)
{
alert( "Choose a country! ")
return false
}
return true;
}
</script>
```

Interaktivní prvek

KAPITOLA 7

Zmeny CSS pomocou DOM

DOM sa často používa na zmenu štýlu HTML prvkov. Syntax je nasledovná:

document.getElementById(id).style.property = new style;

Návštevník webových stránok obvykle klikne na nejaký prvok, napríklad na tlačidlo a v dokumente môže dôjsť k zmene štýlu CSS. Týmto udalostiam sa hovorí aj DOM events. Syntax je nasledujúca:

onclick=JavaScript

Interaktivní prvek

V nasledujúcom príklade sa pozrieme na niektoré vlastnosti a udalosti, ktoré môžeme použiť.

Animácia 4. Príklad zmeny CSS pomocou DOM v CSS

PRÍKLAD

```
<form name="newForm">
<p id='p01'>Lorem ipsum dolor sit amet, consectetur adipiscing elit. </p>
```

Nižšie uvedené tlačidlo zmení farbu vyššie uvedeného textu na zelenú.

```
<input type='button' onclick='document.getElementById("p01").style.color
= "green";' value='Change text color'/><br/><br/>
```

Toto tlačidlo zmení veľkosť písma vyššie uvedeného textu na 27px.

```
<input type='button'
onclick='document.getElementById("p01").style.fontSize = "27px";'
value='Change font size'/><br/><br/>
<p id='p02'>Current time will be displayed here. </p>
```

Toto tlačidlo zavolá funkciu date, ktorá sa nachádza v časti <script> a zobrazí aktuálny čas vo vyššie uvedenom odstavci.

```
<input type='button' onclick='date()' value='Display current
time'/><br/><br/>
```

Prvé uvedené zadávacie pole predstavuje pole, do ktorého používateľ napíše farbu. Nasleduje začiarkávacie políčko a posledné je tlačidlo, ktoré obsahuje funkciu, ktorá zmení pozadie

dokumentu na danú farbu pri začiarknutom začiarkávacom políčku.

```
<input type='text' id='entry' value='Enter color for background'
size='40' /> <br/><br/>
<input type="checkbox" id="box" value="box"> Change background <input
type='button' onclick='changeBackground()' value='Change background'/>
</form>
<script>
```

Nasledujúca funkcia zobrazí aktuálny dátum a čas.

```
function date() {
document.getElementById("p02").innerHTML = Date();
}
```

Táto funkcia zmení farbu pozadia na farbu, ktorú používateľ zadal do políčka a to iba v prípade, že je začiarkávacie políčko začiarknuté.

```
function changeBackground() {
var checkbox = document.forms[0].box;
if (checkbox.checked) {
var body = document.getElementsByTagName("body")[0];
body.style.background = document.getElementById("entry").value;
}
}
</script>
```

Interaktivní prvek

KAPITOLA 8

Práca s tabuľkami pomocou DOM

V tomto príklade vytvoríme zoznam úloh v tabuľke HTML. S tabuľkami môžeme manipulovať pridaním alebo odstránením riadkov, hlavičiek, buniek atď.

Tabuľka 8. Metódy pridávania a odstraňovania rôznych prvkov v tabuľkách

Metóda	Popis
deleteRow()	Odstráni <tr> z tabuľky.
insertRow()	Vytvorí prázdny prvok <tr> v tabuľke.
deleteCell()	Odstráni bunku z aktuálneho riadku tabuľky.
insertCell()	Vytvorí bunku do aktuálneho riadku tabuľky.

V nižšie uvedenom príklade vytvoríme zoznam úloh, v ktorom môžeme pridávať a odstraňovať tabuľky kliknutím na začiarkávacie políčko.

PRÍKLAD

Najprv vytvoríme dve tlačidlá na pridanie a odstránenie úlohy a textové pole, do ktorého môže používateľ napísať popis úlohy. Funkcie sa volajú pomocou funkcie onclick a nachádzajú sa v sekcií <script>.

```
<button onclick="addTask()">Add new task</button>
<button onclick="deleteTask()">Task done</button>
<input type="text" id="textbox" placeholder="Enter task"><br><br>
```

Je to jednoduchá tabuľka len s názvom, bez akýchkoľvek úloh. Tie sa budú pridávať pomocou funkcií.

```
<table id="table" style="width: 50%">
<tr>
  <th>Checkbox</th>
  <th>Row number</th>
  <th>Task</th>
</tr>
</table>

<script>
```

Najprv vytvoríme funkciu na pridanie nových úloh do tabuliek. Musíme vyhľadať tabuľku a v tomto príklade použijeme getElementById.

```
var count_lines=0;
function addTask() {
 var table = document.getElementById("table");
```

Ďalej je potrebné vložiť riadky zdola nahor pomocou metódy insertRow a parametra -1 (aby úloha prešla na spodnú časť). Potom sa do tabuľky vložia tri bunky zodpovedajúce trom stĺpcom.

```
var row = table.insertRow(-1);
 var cell1 = row.insertCell(0);
 var cell2 = row.insertCell(1);
 var cell3 = row.insertCell(2);
```

Ďalej pridáme hodnotu 1 k premennej count_lines pre druhý stĺpec, ktorý počíta aktuálny počet pridaných úloh.

```
count_lines++;
```

Do prvej bunky sa vloží začiarkavacie políčko. Do druhej bunky sa vloží počítadlo riadkov a do tretieho poľa sa vloží hodnota textu z textového poľa.

```
cell1.innerHTML = '<input type="checkbox" name="box" value="0">';
 cell2.innerHTML = count_lines;
 cell3.innerHTML = document.getElementById("textbox").value;
}
```

Ďalej vytvoríme funkciu na odstránenie dokončených úloh v tabuľke. Opäť musíme vyhľadať tabuľku a v tomto príklade použijeme getElementById.

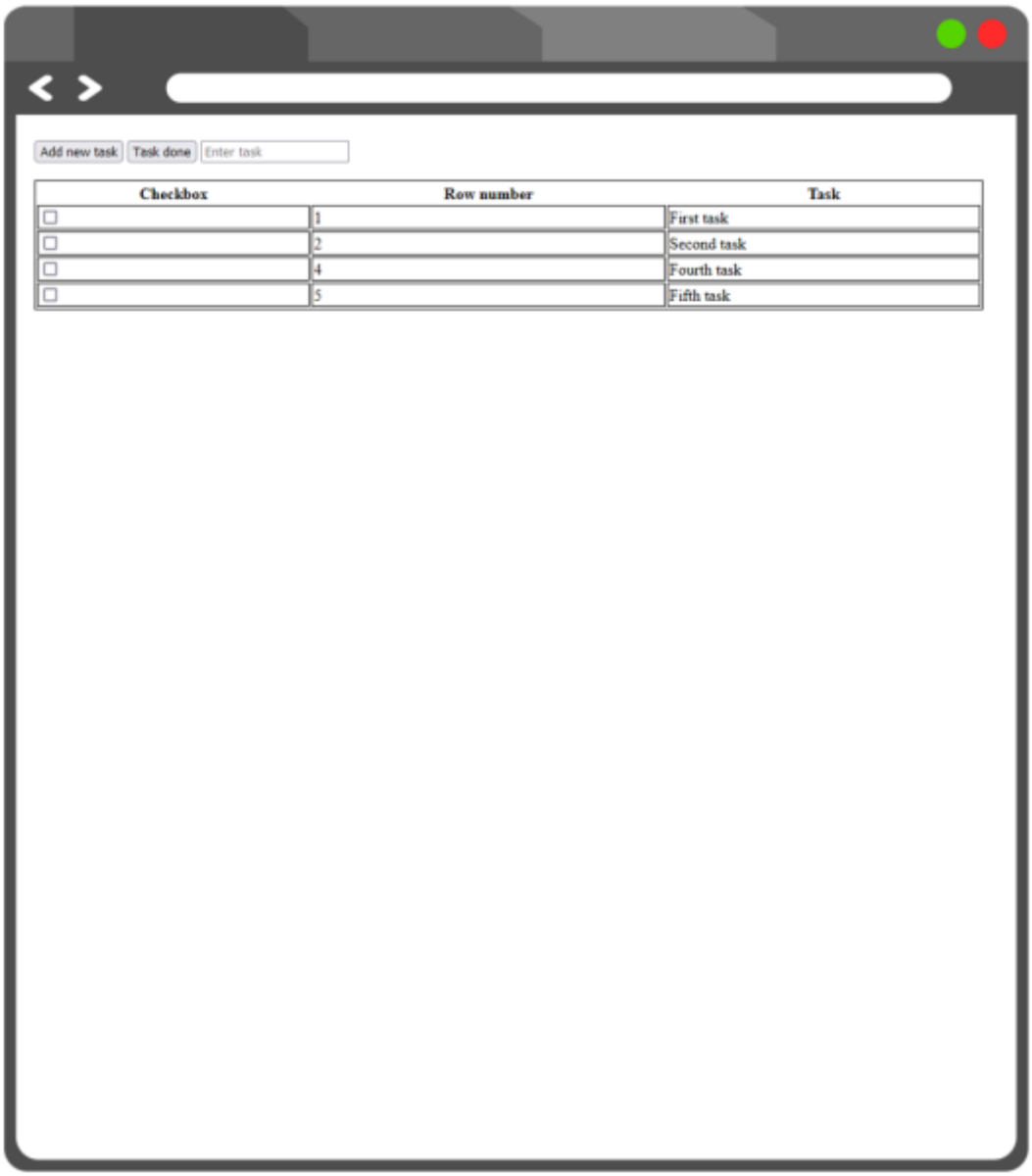
```
function deleteTask() {
 var table = document.getElementById("table");
```

Ďalej sme vytvorili premennú i aby sme našli riadok, v ktorom je začiarknuté políčko. P-otom môžeme odstrániť riadok, ktorý používateľ vybral. Použili sme childNodes a skontrolovali sme, či je začiarkavacie políčko začiarknuté.

```
let i;
 for (i = table.rows.length-1; i >= 1; i--) {
if(table.rows[i].cells[0].childNodes[0].checked==true){
table.deleteRow(i);
}
  }
}
</script>
```

Interaktivní prvek

Interaktivní prvek



Obr. 6. Ako vyzerá zoznam úloh v prehliadači

Interaktivní prvek

Toto bol úvodný kurz do jazyka JavaScript DOM.

KAPITOLA 9

Testt

Kde sa zobrazuje značka <title>?

- ◯ pod značkou <body>
- ◯ pod značkou <head>
- ◯ pod značkou <p>
- ◯ pod značkou <img>

Ktoré z uvedených výrazov označujú prvky v HTML?

- ☐ username
- ☐ začiatok značky
- ☐ obsah
- ☐ koniec značky

Ktorá značka je určená na to, aby bol text zvýraznený tučným písmom?

- ◯ <u>
- ◯ <i>
- ◯ <b>
- ◯ <q>

Co je tzv. prázdny prvok?

- ☐ prvok <>
- ☐ prvok, ktorý nemá koncovú značku

- ☐ prvok, ktorý neobsahuje obsah
- ☐ prvok bez atribútu

Kde môže byť značka <script> umiestnená?

- ☐ v značke <head>
- ☐ v značke <table>
- ☐ v značke <body>
- ☐ pred značkou <!DOCTYPE html>

Ako vytvoríte komentár v HTML?

- ◯ <!-- comment -->
- ◯ /* comment */
- ◯ // comment

Ktorá značka vytvorí zoznam s odrážkou bodka na začiatku?

- ◯
- ◯ <dl>
- ◯
- ◯

Text sa v JavaScripte označuje pomocou:

- ☐ spätného lomítka
- ☐ úvodzoviek
- ☐ na označenie textu nie sú potrebné žiadne znaky
- ☐ dvojitých úvodzoviek

Ktorá značka musí byť v HTML na spustenie kódu v Javascripte?

- ○ <js>
- ○ <src>
- ○ <script>
- ○ <body>

Ktoré výrazy deklarujú premennú v JavaScripte?

- ☐ var
- ☐ const
- ☐ for
- ☐ let

Môže mať funkcia v JavaScripte viac parametrov?

- ○ áno
- ○ nie

Ktoré uzly môžeme použiť na navigáciu medzi <html> a <head>?

- ☐ sibling
- ☐ firstChild
- ☐ sisterNode
- ☐ parentNode

Aký symbol musíme uviesť v nasledovnom príklade, ak chceme zistiť, či hodnota x je 5 a typ premennej nie je dôležitý?
if (x ??? 5) {};

- ○ ==

- ◯ =
- ◯ ===
- ◯ ====

Aké vzťahy môžu mať <body> a <head>?

- ☐ previousSibling
- ☐ nextSibling
- ☐ firstChild
- ☐ parentNode

Ktorý objekt je ako prvý v HTML DOM?

- ◯ <p>
- ◯ <html>
- ◯ document
- ◯ <head>

Ktoré z nasledovných výrazov predstavujú uzly?

- ☐ HTML prvok
- ☐ text v HTML prvkoch
- ☐ atribút HTML prvku
- ☐ celý HTML dokument

Chceme zmeniť <p> značku. Akú metódu môžeme použiť, ak značka <p> neobsahuje žiadny identifikátor?

- ◯ setAttribute
- ◯ getElementsByClassName

- ◯ getElementById
- ◯ getElementsByTagName

Ktoré metódy môžeme použiť, ak chceme vytvoriť nový odsek s uzlami?

- ☐ value
- ☐ createTextNode
- ☐ getElementById
- ☐ createElement

Ktoré prvky môžeme použiť na vytvorenie nových HTML prvkov?

- ◯ setAttribute
- ◯ createElement
- ◯ getElementById
- ◯ appendChild

Ktoré metódy môžeme použiť, ak chceme odstrániť prvok?

- ☐ removeParent()
- ☐ removeSibling()
- ☐ remove()
- ☐ removeChild()

Aké metódy potrebujeme, ak chceme vytvoriť nový <td> v novom <tr>?

- ☐ insertRow()
- ☐ checked()
- ☐ insertCell()
- ☐ deleteCell()

Na čo slúži innerHTML?

- ◯ na prístup k obsahu HTML dokumentu
- ◯ na prístup k obsahu <html> značky
- ◯ na prístup k podradenému uzlu z aktuálneho uzla

Čo robí metóda appendChild()?

- ◯ priradí k potomkovi nového potomka
- ◯ pripojí nový prvok k nadradenému prvku
- ◯ pripojí podradený prvok k súrodencovi

Čo robí childNodes[1] ?

- ◯ zoberie druhý prvok typy, ktorá hľadáme
- ◯ zoberie prvý prvok typy, ktorá hľadáme
- ◯ je to rovnaké ako v prípade lastChild

Akú metódu môžeme použiť, ak chceme pridať nový prvok k existujúcemu prvku nachádzjúcemu sa na poslednom mieste?

- ◯ appendChild
- ◯ insertBefore
- ◯ replaceChild

Ktorú možnosť môžeme použiť vo formulári na volanie funkcie?

- ◯ action
- ◯ onsubmit
- ◯ href

Akú vlastnosť môžeme použiť na prístup k obsahu poľa vo formulári HTML?

- ◯ isNaN
- ◯ year
- ◯ value

S čím získame prístup k číslu vybratej možnosti v rozbaľovacom zozname?

- ◯ checked
- ◯ selectedIndex
- ◯ value

Čo môžeme použiť na zistenie, či pole vo formulári bolo vybraté?

- ◯ checked
- ◯ selectedIndex
- ◯ value

Môžeme zmeniť CSS s DOM bez použitia udalostí?

- ◯ áno
- ◯ nie

Ktorou vlastnosťou môžeme zmeniť farbu textu?

- ◯ style.font-color
- ◯ style.color
- ◯ style.background-color

S akou vlastnosťou môžeme zmeniť farbu pozadia?

- ◯ style.background
- ◯ style.color
- ◯ style.background-color

Ako môžeme vytvoriť nové <tr> v tabuľke?

- ◯ insertCell()
- ◯ insertRow
- ◯ insertTableRow