JavaScript DOM

Lili Nemec Zlatolas

Annotation

V tem tečaju je predstavljen objektni model dokumenta v jeziku JavaScript.

Objectives

Tečaj omogoča hiter pregled potrebnega predhodnega znanja, kot so osnove jezikov HTML in JavaScript. Učenec bo spoznal objektni model dokumenta v javascriptu.

Keywords

JavaScript, DOM, HTML

Date of Creation

15.4.2022

Duration

12 ur

Language

English

License

ISBN

Literature

[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.

CHAPTER 1

Osnove HTML

Za upravljanje spletnih mest prek objektnega modela dokumenta (DOM) je potrebno predhodno znanje jezikov HTML in JavaScript. HTML je označevalni jezik, ki opisuje strukturo spletnega mesta. HTML je sestavljen iz elementov, ki so označeni in brskalniku pošiljajo informacije o tem, kako naj prikaže vsebino dokumenta.

Zadnji sprejeti standard je HTML5. HTML je sestavljen iz elementov, ki jih določajo začetna oznaka, vsebina elementa in končna oznaka: <tag> Vsebina </tag>

Tukaj je primer dokumenta HTML:

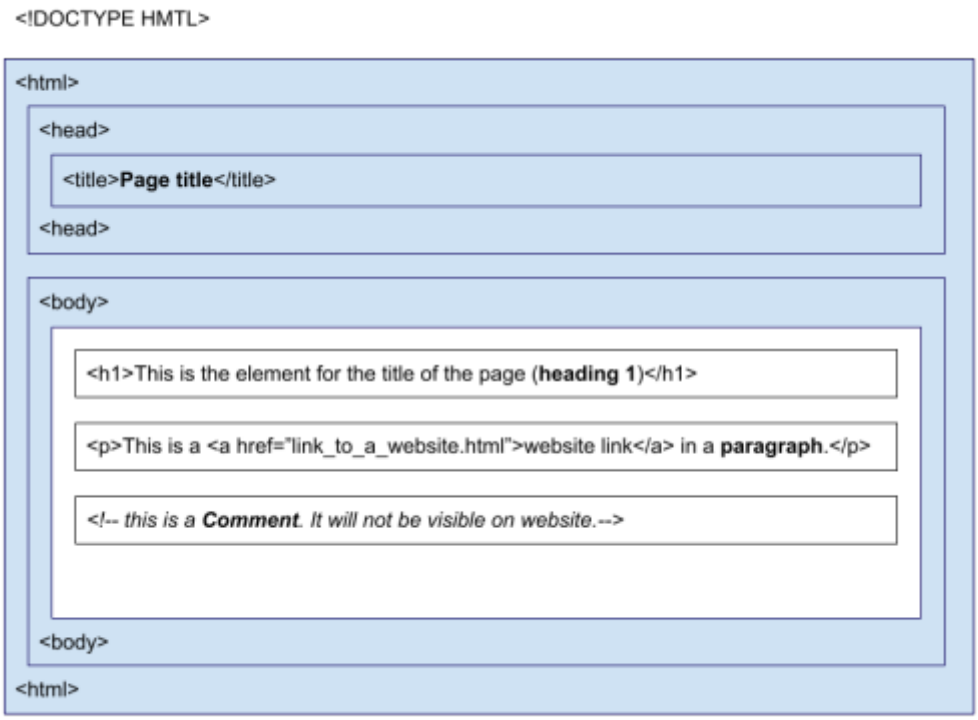

Fig. 1. Struktura strani HTML in nekateri elementi

Za urejanje dokumentov lahko uporabite program Notepad ali podobne programe (npr. Notepad++, Visual Studio Code). Dokument HTML mora imeti končnico .html.

HTML ne razlikuje velikih in malih črk, vendar je v oznakah HTML priporočljivo uporabljati male črke.

Table 1. Nekateri osnovni elementi HTML

Oznaka	Informacije
`<! DOCTYPE html>`	To je izjava za HTML5. Prikazana mora biti na vrhu dokumenta HTML.
`<h1>`	Obstajajo naslovi HTML od <h1> do <h6>, pri čemer je prvi največji naslov.
`<p>`	To je oznaka za odstavek.
` `	To je oznaka za prehod v naslednjo vrstico.
`<a href="https://www.websitelink.com">`	To je oznaka za povezavo do spletnega mesta. Uporablja atribute.
`<img src="image_location.jpg" alt="image_description">`	To je oznaka za vstavljanje slike. Uporablja tudi različne atribute.
`<b>`	To je oznaka za odebelitev besedila.

Nekateri elementi se imenujejo prazni elementi. Primer praznega elementa je
, ki nima vsebine ali končne oznake. Lahko pa vsebuje končno oznako znotraj začetne oznake, če uporabljamo smernice XHTML:

Oznake HTML imajo lahko atribute. Nekateri elementi brez atributov nimajo nobenih funkcij. Atributi se pojavijo v začetnih oznakah. Eden od pogosto uporabljenih atributov je style, ki se lahko uporablja na primer v odstavku:

```
<p style="color:blue;"> Besedilo bo modro. </p>
```

Interaktivní prvek

Pogosto se uporabljajo tabele HTML, ki so sestavljene iz celic znotraj stolpcev in vrstic tabele. Preprosta tabela HTML je:

EXAMPLE

```
<table border="1">
  <tr> <!--to je vrstica tabele, ki nima vsebine -->
    <th> To je glavo tabele </th>
    <th> In še ena celica glave tabele v isti vrstici </th>
  </tr>
  <tr>
    <td>to je prva celica</td>
    <td>to je druga celica v vrsti</td>
  </tr>
</table>
```

Interaktivní prvek

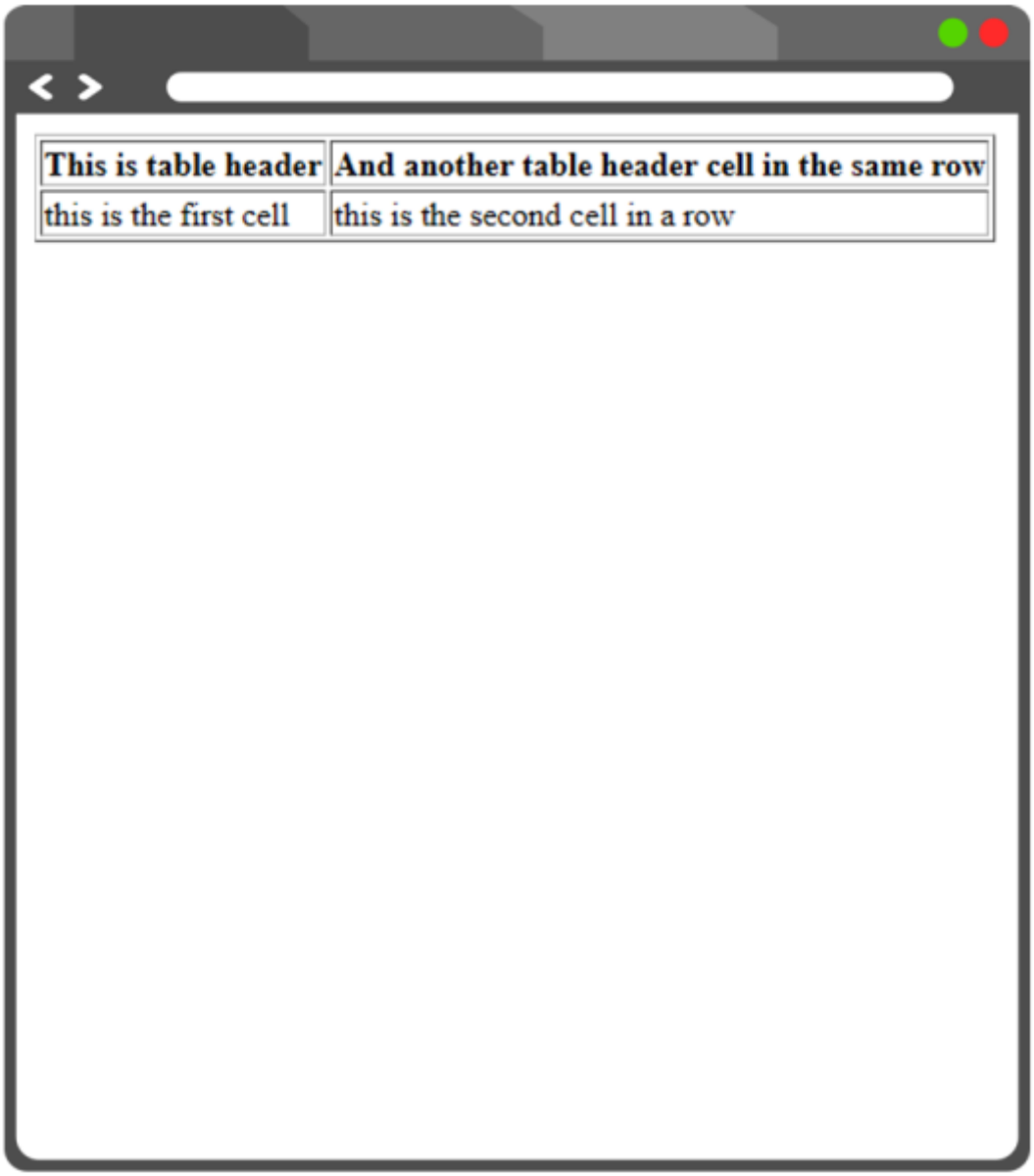

Fig. 2. Videz tabele v brskalniku.

Seznami HTML so namenjeni ustvarjanju urejenih ali neurejenih seznamov HTML. Urejeni seznami so oštevilčeni, neurejeni seznami pa imajo običajno pike.

Table 2. Primer seznamov HTML

Primer seznama	Pogled brskalnika

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

`<ol>`
`  <li>Pes</li>`
`  <li>Kot</li>`
`  <li>Ribe</li>`
`</ol>`
	1. Pes
	2. Mačka
	3. Ribe

[Interaktivní prvek]

To je bil kratek uvod v HTML. V naslednjem poglavju si bomo ogledali nekaj osnov sintakse JavaScript.

CHAPTER 2

Osnove JavaScripta

JavaScript je programski jezik, s katerim lahko spletna mesta postanejo bolj interaktivna in dinamična. Uporablja se v jeziku HTML med oznakama <script> in </script> v razdelku <body> ali <head> strani HTML ali v obeh.

JavaScript lahko v datoteko html vstavite tudi kot zunanjo datoteko s končnico .js. Primer:

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

Obstaja več možnosti za prikaz podatkov JavaScript. Uporabimo lahko:

innerHTML (npr. document.getElementById(id))

1. Te metode si bomo ogledali v naslednjih poglavjih, saj je to DOM

document.write()

2. To se običajno uporablja za testiranje, saj iz dokumenta izbriše ves obstoječi HTML in prikaže samo vsebino skripte.

okno.alert()

3. Ustvari opozorilno polje za prikaz podatkov. Okno lahko izpustite.

console.log()

4. To se običajno uporablja za odpravljanje napak in prikazuje podatke v brskalniku.

Koda JavaScript je na koncu vsake izvedljive kode ločena s podpičji (**;**).

Bloki kode so združeni v **{vijugastih oklepajih}**.

Interaktivní prvek

Pri enovrstičnih komentarjih se za začetek uporablja **//,** pri večvrstičnih pa **/***, za konec pa ***/**.

Nekatere najpomembnejše ključne besede v jeziku JavaScript so predstavljene v preglednici. To so rezervirane besede, ki jih ni mogoče uporabiti za imena spremenljivk.

Table 3. Rezervirane besede v jeziku JavaScript

Rezervirane besede / ključne besede	**Opis**

`var`	Deklarira spremenljivko, ki jo je mogoče ponovno deklarirati in posodobiti.
`let`	Deklarira blokovno spremenljivko, ki jo je mogoče posodobiti, vendar ne ponovno deklarirati.
`const`	Deklarira spremenljivko, ki je ni mogoče posodobiti ali ponovno deklarirati.
`return`	Zaključi funkcijo
`if`	Začetek kode pogoja
`for`	Začetek kode zanke
`function`	Deklarira funkcijo

Interaktivní prvek

Sintaksa JavaScripta so pravila za sestavo programov. Najprej se ustvarijo spremenljivke, ki se razglasijo, nato pa se uporabijo v programu.

Decimalna **števila** so ločena s **piko**. **Besedila** so zapisana v **enojnih ali dvojnih narekovajih**. **Znaki enakosti** pripisujejo vrednosti spremenljivkam.

Imena spremenljivk se morajo začeti s črkami abecede (A-Z), znakom dolarja ali podčrtalcem. **Pri tem se razlikujejo velike in male črke**.

Primer ustvarjanja različnih spremenljivk:

```
let firstNumber;
var firstText;
firstNumber = 13;
var secondNumber = 17;
firstText = "To je številka 13."
```

Table 4. Enaki znaki v jeziku JavaScript

Znak enakosti	**Opis**
=	To je operator pripisovanja (npr. **var a = a * 2;)** in nima enakega pomena kot algebrski znak.
==	To je operator enakosti, ki se uporablja v algebri, na primer v primeru 7+2=9, v javascriptu pa se uporablja kot npr. **if (x == 2) {};**
===	To je enaka vrednost in tudi enaka vrsta.

Za združevanje ali izračun vrednosti spremenljivke lahko uporabite = in +.

naj celotno ime = "Maya" + " " + "Priimek";

Interaktivní prvek

JavaScript ima 3 logične operatorje:

- && logični in
- || logični ali
- ! logični ne

Funkcije so bloki kode, ki jih je treba poklicati, da se izvedejo. Imajo lahko več parametrov, koda pa je v oglatih oklepajih. Uporabne so, kadar funkcije uporabimo večkrat z različnimi argumenti. Primer funkcije:

EXAMPLE

```
funkcija circle_area( a, b)
{
return a * a * b; // Funkcija vrne produkt a, a in b. Če želimo
izračunati površino kroga, moramo namesto števila B vnesti število PI.
}
let area = circle_area (15, Math. PI); // Funkcija se pokliče z dolžino
15 cm in številko PI.
document. write( "Površina kroga je " + površina + " m2 ." );
```

Ko zaženete zgornjo funkcijo, se v brskalniku prikaže: "**Površina kroga je 706,8583470577034 m2.**"

V jeziku JavaScript se lahko naučite še veliko več, vendar je to nekaj osnovnih in pomembnih stvari, ki jih potrebujemo za začetek uporabe JavaScript DOM.

CHAPTER 3

Uvod v objektni model dokumentov

Brskalnik ob nalaganju spletnega mesta ustvari objektni model dokumenta (DOM). DOM je standard konzorcija W3C (World Wide Web Consortium) za dostop do dokumentov.

HTML DOM je zgrajen kot drevo objektov.

V animaciji je predstavljen primer naslednje kode.

```
<html>
  <head>
    <title>To je naslov strani.</title>
  </head>
<body>
  <h1 align="levo">To je naslov.</h1>
  <p style="background-color: blue">To je odstavek.</p>
  <img src="image.jpg" />
</body>
</html>
```

Interaktivní prvek

Animation 1. Drevo objektov v modelu HTML DOM

Drevo ima **vozlišča za** elemente, ki predstavljajo oznake HTML in določajo strukturo dokumenta. Standard je zasnovan tako, da najprej ustvarimo vozlišče, nato mu dodamo otroka in atribute otroka. Zato je lahko koda precej dolga.

V animaciji je <html> *korensko vozlišče* brez staršev, vendar je *starš prvega otroka* <head> in *zadnjega otroka* <body>.

JavaScript lahko dostopa do DOM ter spreminja elemente in atribute na strani HTML.

V DOM so vsi elementi HTML opredeljeni kot **objekti**.

DOM je standard, ki vsebuje informacije o tem, kako pridemo do elementov HTML, kako jih spreminjamo, dodajamo ali brišemo v dokumentih HTML. DOM uporablja **metode za** dostop do elementov HTML in izvajanje dejanj na njih.

Interaktivní prvek

3.1 Navigacija po DOM

Vozlišča v drevesu vozlišč so hierarhično povezana, kar pomeni, da ima vsako vozlišče natanko enega starša, razen prvega vozlišča, ki nima starša. Vozlišče ima lahko več otrok. Sorodna vozlišča pa so vozlišča z istim staršem.

Za premikanje med vozlišči z JavaScriptom lahko uporabimo naslednje lastnosti:

- parentNode
- childNodes[nodenumber]
- firstChild
- lastChild
- nextSibling
- previousSibling

Animation 2. Primer vozlišč v dokumentu

Lastnost childNodes[0] je enaka lastnosti firstChild. Vsebuje objekt, podoben polju, z lastnostjo length, s katero dostopa do podrejenih vozlišč.

Interaktivní prvek

Vozlišča in način pridobivanja vsebine iz elementov bomo podrobneje obravnavali v poglavju Vozlišča DOM.

CHAPTER 4

Metode DOM

Z **metodami** DOM lahko izvajamo dejanja na elementih HTML. **Lastnosti** DOM so vrednosti elementov HTML, ki jih lahko nastavljamo ali spreminjamo.

EXAMPLE

```
<p id="primer"> To ne bo prikazano na strani, ker bo Javascript prepisal
besedilo znotraj vozlišča odstavka. </p>
<script>
document.getElementById("example").innerHTML = "To je besedilo, ki bo
prikazano na spletnem mestu. ";
</script>
```

V tem primeru "**To je besedilo, ki bo prikazano na spletnem mestu.** " se bo prikazalo v brskalniku, ko se bo koda izvedla.

V tem primeru je getElementById **metoda,** innerHTML pa **lastnost**. Za iskanje elementa v dokumentu se pogosto uporablja id (v primeru id="example"). Sicer pa lahko za dostop do določenih elementov uporabimo nadrejena in podrejena vozlišča.

Lastnost innerHTML se uporablja za nastavitev ali vrnitev vsebine HTML elementa, v zgornjem primeru spremeni besedilo znotraj oznake <p> z elementom h id.

Table 5. Metode za iskanje, spreminjanje, dodajanje in brisanje elementov Elementov HTML

Metoda	Opis
document.getElementById(id)	Iskanje elementa po id elementa
document.getElementsByTagName(ime)	Iskanje elementa po imenu oznake
document.getElementsByClassName(ime)	Iskanje elementa po imenu razreda
element.setAttribute(atribut, vrednost)	Spreminjanje vrednosti atributa elementa HTML
document.createElement(element)	Ustvarjanje elementa HTML
document.removeChild(element)	Odstranjevanje podrejenega elementa HTML
document.appendChild(element)	Dodajanje podrejenega elementa HTML
document.replaceChild(new, old)	Zamenjava podrejenega elementa HTML

Interaktivní prvek

Video 1. Primeri pridobivanja elementa po X

Interaktivní prvek

CHAPTER 5

Vozlišča DOM

Vsi **elementi HTML, njihovi atributi in besedila so vozlišča**. Nekateri elementi vsebujejo druga vozlišča.

Navigacija med vozlišči je bila že opisana v poglavju Navigacija DOM.

Tukaj je primer, kako lahko dostopamo do vrednosti vozlišč.

```
<p id='paragaph'>To je prvi odstavek.</p>
```

Element <p> vsebuje **besedilno vozlišče z** vrednostjo "To je prvi odstavek." Do vrednosti besedila lahko dostopate z lastnostjo innerHTML vozlišča:

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

Enako lahko storite z dostopom do **nodeValue**:

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

Video 2. Podrejena vozlišča in vrednosti vozlišč

Korenska vozlišča lahko dostopajo do celotnega dokumenta:

- document.body - Vsebina dokumenta
- document.documentElement - Celoten dokument

Lastnost **nodeValue** določa vrednost vozlišča. Pri vozliščih elementov je nična, vendar je uporabna pri vozliščih besedila, kjer predstavlja samo besedilo. Lastnost za vozlišča atributov vrne vrednost atributa.

Lastnost **nodeName** določa ime vozlišča in je namenjena samo branju. nodeName vozlišča elementa vrne ime oznake, vrne pa ime atributa vozlišča atributa. nodeName vozlišča besedila je *#text,* vozlišča dokumentov pa *#document*.

V podpoglavjih bomo za ustvarjanje, odstranjevanje in zamenjavo elementov DOM HTML (vozlišč) uporabili nekatere od naslednjih metod. V tej tabeli si bomo ogledali metode za ustvarjanje elementov in besedilnih vozlišč.

Table 6. Metode za ustvarjanje novih elementov DOM HTML

Metoda	Opis
createElement(tip)	Ustvari vozlišče elementa z določenim tipom (npr. tip "p").

createTextNode()	Ustvari vozlišče besedila.

V naslednji tabeli so predstavljene metode za ustvarjanje, odstranjevanje in zamenjavo vozlišč. Običajno moramo ustvariti element, nato v njem ustvariti besedilno vozlišče in ga dodati obstoječi strukturi. Primere uporabe si bomo ogledali v naslednjih podpoglavjih.

Table 7. Metode za ustvarjanje, odstranjevanje in zamenjavo vozlišč

Metoda	**Opis**
appendChild(vozlišče)	Elementu doda nov podrejeni element (vozlišče).
insertBefore(novo, obstoječe)	Vstavi podrejeno vozlišče pred obstoječe podrejeno vozlišče.
replaceChild(novo, staro)	Zamenja podrejeno vozlišče z novim vozliščem.
remove()	Odstrani element iz dokumenta. Nima parametrov.
removeChild(vozlišče)	Odstrani otroka elementa.

5.1 Ustvarjanje elementov DOM HTML (vozlišča)

V tem poglavju si bomo ogledali, kako lahko dodamo, odstranimo ali zamenjamo elemente HTML DOM.

Če želimo dodati nov element, moramo najprej ustvariti vozlišče elementa in ga nato dodati obstoječemu elementu. Tukaj je primer kode, v kateri smo dodali nov odstavek z vozlišči.

EXAMPLE

```
<div id="div01">
 <p id="p01">Prvi odstavek.</p>
 <p id="p02">Drugi odstavek.</p>
</div>
<script>
var newParagraph = document.createElement("p");
var textForParagraph = document.createTextNode("To je nov odstavek.");
newParagraph.appendChild(textForParagraph);
var element = document.getElementById("div01");
element.appendChild(newParagraph);
</script>
```

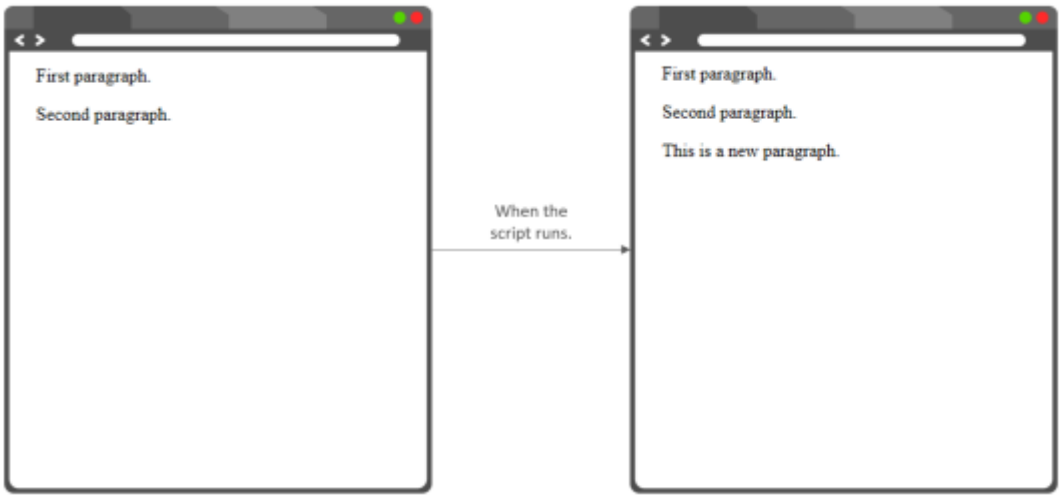

Fig. 3. Videz kode v brskalniku brez skripte in z njo.

createElement("p") ustvari nov element <p>. Če želimo temu elementu dodati besedilo, moramo ustvariti vozlišče besedila z **createTextNode**. Nato moramo vozlišče besedila dodati elementu <p> z **appendChild**. Nazadnje moramo nov element dodati obstoječemu elementu div.

Zdaj bomo uporabili isti primer, vendar bomo odstavek vstavili na prvo mesto s funkcijo insertBefore in ne na zadnje mesto, kot smo to storili s funkcijo appendChild.

Video 3. Primer metode insertBefore

Interaktivní prvek

5.2 Zamenjava elementov DOM HTML (vozlišča)

Če želimo zamenjati element, moramo najprej ustvariti vozlišče elementa in ga nato zamenjati z obstoječim elementom. Tukaj je primer kode, v kateri smo nov odstavek nadomestili s starim odstavkom.

EXAMPLE

```
<div id="div01">
 <p id="p01">Prvi odstavek.</p>
 <p id="p02">Drugi odstavek.</p>
</div>
<script>
var newParagraph = document.createElement("p");
var textForParagraph = document.createTextNode("To je nov odstavek.");
newParagraph.appendChild(textForParagraph);
var parent = document.getElementById("div01");
var child = document.getElementById("p01");
parent.replaceChild(newParagraph,child);
</script>
```

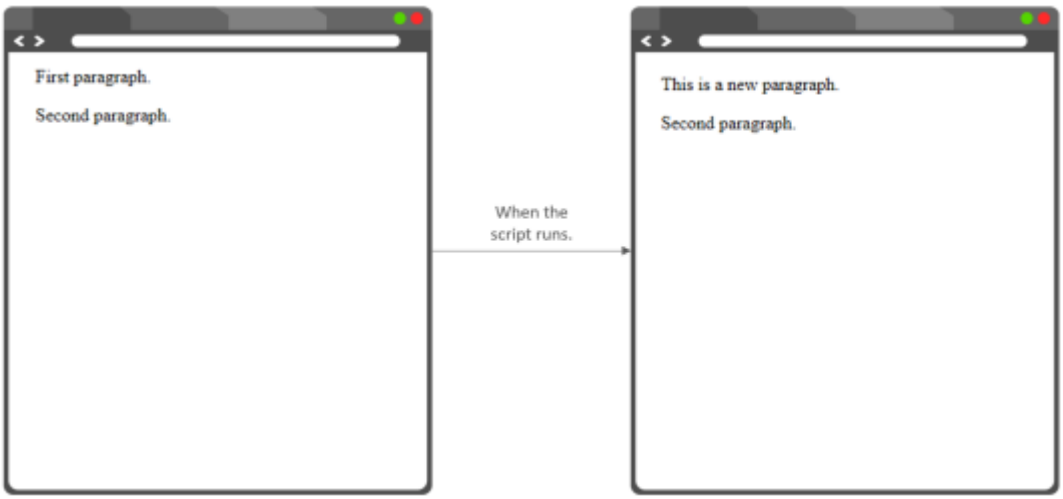

Fig. 4. Kako je koda videti v brskalniku brez skripte in z njo.

Interaktivní prvek

5.3 Odstranjevanje elementov DOM HTML (vozlišč)

V tem poglavju si bomo ogledali animacijo, v kateri bomo odstranili en element z metodo remove() in odstranili en element z dostopom do starša z metodo removeChild(). Metoda remove() ne deluje v starejših brskalnikih, zato moramo včasih uporabiti metodo removeChild().

V ukazu removeChild moramo poiskati starša, da odstranimo otroka.

Animation 3. Odstranjevanje elementov DOM HTML z dvema primeroma

Interaktivní prvek

CHAPTER 6

Preverjanje veljavnosti obrazca DOM

Validacijo obrazca HTML lahko izvedemo z JavaScriptom prek DOM. Običajno želimo preveriti, ali je uporabnik vnesel pravilne podatke v pravilni obliki. Zagotoviti želimo, da je uporabnikov vnos pravilen. V tem primeru bomo ustvarili preprost obrazec in preverili, ali je uporabnik vnesel pravilne podatke, s funkcijo s pomočjo DOM.

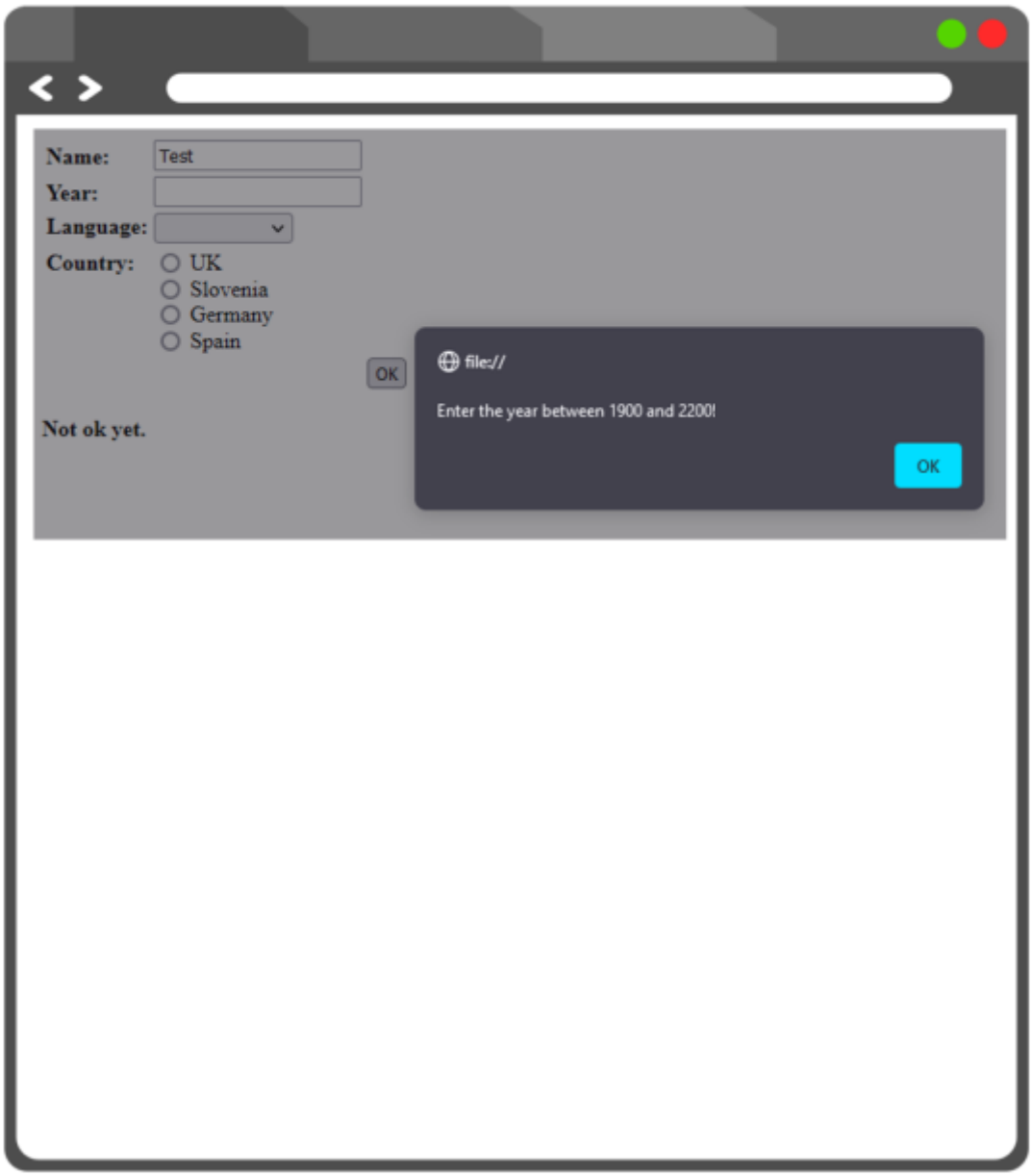

Fig. 5. Primer opozorila in obrazca, ki bosta predstavljena v spodnji kodi.

EXAMPLE

Ustvarjamo obrazec s 4 različnimi možnostmi. Za boljši prikaz smo uporabili tabelo HTML.

V obrazcu so bili uporabljeni besedilo, možnost in vnos z vrsto radijskega gumba ter gumb za pošiljanje.

```
<form onsubmit="return validation( this)" action="#">
<table>
<tr>
<td><b>Naziv: </b></td>
<td><input type="text" name="ime"></td>
</tr>
<tr>
<td><b>Leto: </b></td>
<td><input type="text" name="leto"></td>
</tr>
<tr>
<td valign="top"><b>Jazik: </b></td>
  <td>
  <select name="jezik">
<option> </option>
<option value="slo">angleščina</option>
<option value="ang">slovenski</option>
<option value="nem">nemščina</option>
<option value="fra">french</option>
</select>
</td>
</tr>
<tr>
<td valign="top"><b>Krajina: </b></td>
<td>
<input type="radio" name="country" value="1"> UK<br/>
<input type="radio" name="country" value="2"> Slovenija<br/>
<input type="radio" name="country" value="3"> Nemčija<br/>
<input type="radio" name="država" value="4"> Španija
  </td>
</tr>
<tr>
<td></td>
<td></td>
  <td><input type="submit" value="OK"></td>
</tr>
</table>
</form>
<script type="text/javascript">
```

Tu je ustvarjena funkcija potrjevanja. Ko uporabnik nečesa ne bo pravilno izpolnil, se bo pojavilo opozorilno okno.

```
funkcija validacija (obrazec)
{
```

S pomočjo DOM smo dostopali do obrazca, vnosa imena in njegove vrednosti. Če je ta prazen, se prikaže opozorilo.

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

Če je vnesena letnica zunaj razpona 1900 in 2200, se prikaže opozorilo. Z isNaN preverimo tudi, ali je uporabnik sploh vnesel številko.

```
if (isNaN(form.year.value) || form.year.value < 1900 || form.year.value >
2200)
{
alert("Vnesite leto med 1900 in 2200!")
return false
}
```

Če uporabnik za spustni seznam jezikov ni izbral nobenega indeksa, se prikaže opozorilo.

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

Če uporabnik ni izbral nobene od držav z radijskim gumbom, se prikaže opozorilo.

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

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

Interaktivní prvek

CHAPTER 7

DOM spreminja CSS

Pogosto se DOM uporablja za spreminjanje sloga elementov HTML. Sintaksa za to je:

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

Običajno mora obiskovalec spletnega mesta klikniti na nek element, na primer na gumb, in v dokumentu se lahko spremeni slog CSS. To so tudi dogodki DOM. Sintaksa za to je:

onclick=JavaScript

Interaktivní prvek

V spodnjem primeru si bomo ogledali nekaj lastnosti in dogodkov, ki jih lahko uporabimo.

Animation 4. Primer spreminjanja DOM v CSS

EXAMPLE

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

Ta gumb spodaj bo spremenil barvo besedila zgoraj v zeleno barvo.

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

Ta gumb spodaj bo spremenil velikost pisave zgornjega besedila v 27px.

```
<input type='button'
onclick='document.getElementById("p01").style.fontSize = "27px";'
value='Spremeni velikost pisave'/><br/><br/>
<p id='p02'>Tudi bo prikazan trenutni čas. </p>
```

Ta gumb bo poklical funkcijo z imenom date, ki je v delu <script>, in prikazal trenutni čas v zgornjem odstavku.

```
<input type='button' onclick='date()' value='Prikaži trenutni
čas'/><br/><br/>
```

Spodnji vnos prikazuje polje, v katero uporabnik vnese barvo. Naslednji vnos je potrditveno polje, v zadnjem pa je funkcija, ki spremeni ozadje besedila, napisanega v polju.

```
<input type='text' id='entry' value='Vnesi barvo za ozadje' size='40' />
<br/><br/>
<input type="checkbox" id="polje" value="polje"> Spremeni ozadje <input
type="button" onclick="changeBackground()" value="Spremeni ozadje"/>
</form>
<script>
```

Ta funkcija prikaže trenutni datum in čas.

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

Ta funkcija spremeni barvo ozadja na barvo, ki jo je uporabnik vnesel v polje, in to le, če je potrditveno polje označeno.

```
funkcija 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

CHAPTER 8

DOM – spreminjanje tabel

V tem primeru bomo naredili seznam opravil s tabelo HTML. Tabele lahko upravljamo tako, da dodajamo ali brišemo vrstice, glave, celice itd.

Table 8. Metode za dodajanje in brisanje različnih elementov v tabelah

Metoda	Opis
deleteRow()	Odstrani <tr> iz tabele.
insertRow()	Ustvari prazen element <tr> v tabeli.
deleteCell()	Izbriše celico iz trenutne vrstice tabele.
insertCell()	Ustvari celico v trenutni vrstici tabele.

V spodnjem primeru bomo ustvarili seznam opravil, na katerem lahko dodajamo in odstranjujemo tabele iz tabele s klikom na potrditveno polje.

EXAMPLE

Najprej bomo ustvarili dva gumba in besedilno polje, kjer lahko uporabnik doda novo nalogo, izbriše novo nalogo ali napiše, kaj je naloga. Funkcije se kličejo prek funkcije onclick in so v razdelku <script>.

```
<button onclick="addTask()">Dodaj novo opravilo</button>
<button onclick="deleteTask()">Naloga je opravljena</button>
<input type="text" id="textbox" placeholder="Vnesi nalogo"><br><br>
```

To je preprosta miza, ki ima samo glavo mize brez nalog. Te bodo dodane s funkcijami.

```
<table id="tabela" style="width: 50%">
<tr>
  <th>Pogojček</th>
  <th>Številka vrstice</th>
  <th>Naloga</th>
</tr>
</table>

<script>
```

Najprej bomo ustvarili funkcijo za dodajanje novih opravil v tabele. Poiskati moramo tabelo in v tem primeru bomo uporabili getElementById.

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

Nato je treba vstaviti vrstice od spodaj navzgor z metodo insertRow in -1 (tako da bo naloga šla na dno). Nato se v tabelo s tremi stolpci vstavijo tri celice.

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

Nato dodamo 1 v spremenljivke count_lines za drugi stolpec, ki šteje trenutno število dodanih opravil.

```
count_lines++;
```

V prvo celico je vstavljeno potrditveno polje. V drugo celico se vstavi števec vrstic, v tretje polje pa se vstavi vrednost besedila iz besedilnega polja.

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

Nato bomo ustvarili funkcijo za brisanje dokončanih opravil v tabelah. Ponovno moramo poiskati tabelo in v tem primeru bomo uporabili getElementById.

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

Nato smo ustvarili spremenljivko i, s katero poiščemo vrstico, v kateri je potrditveno polje označeno, in nato lahko izbrišemo vrstico, ki jo je uporabnik izbral kot končano. Uporabili smo chilNodes in preverili, ali je potrditveno polje označeno.

```
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

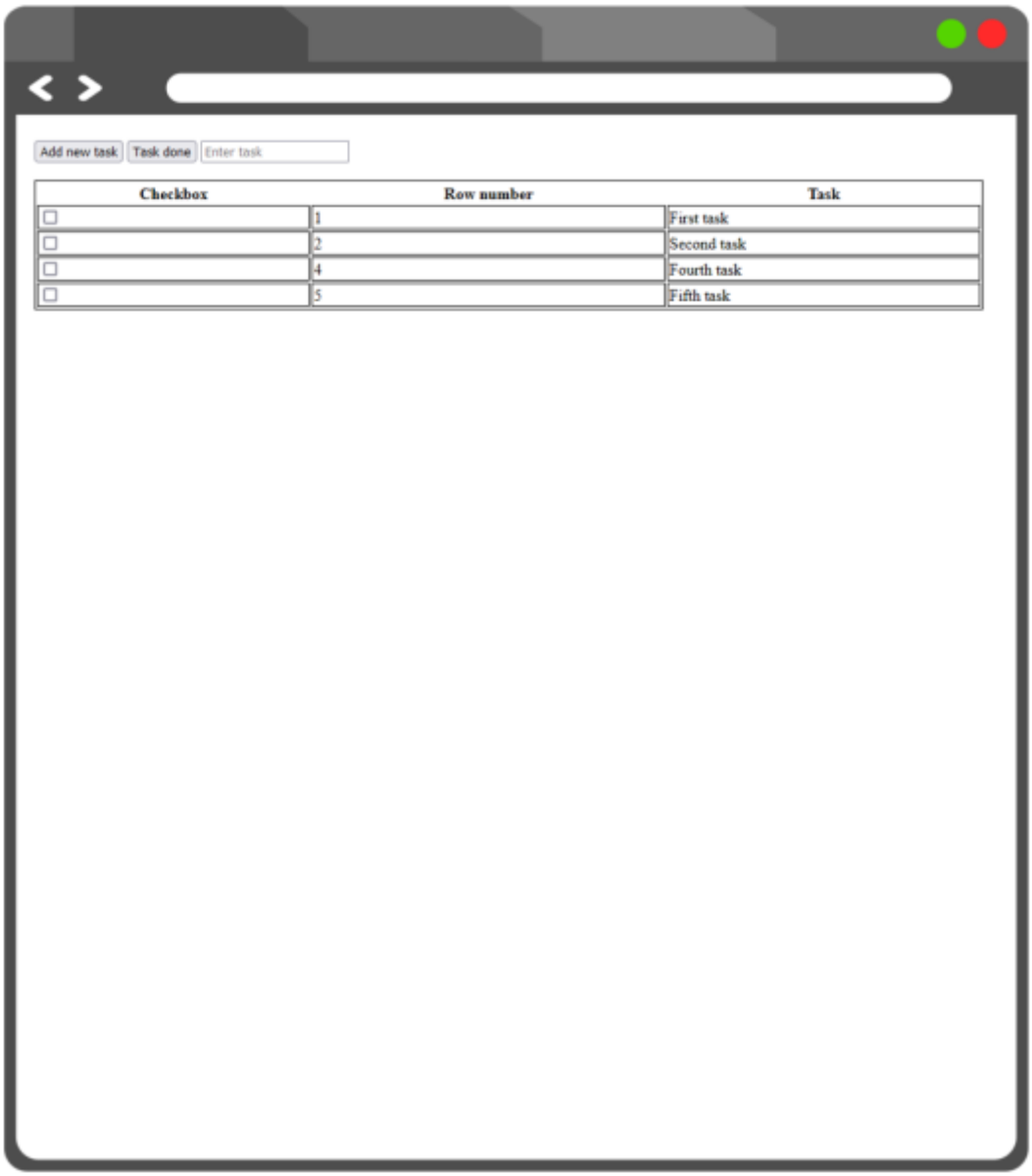

Fig. 6. Kako je seznam opravil videti v brskalniku.

Interaktivní prvek

To je bil uvodni tečaj v JavaScript DOM.

CHAPTER 9

Test

Katera oznaka je namenjena krepki pisavi besedila?

- ◯ <u>
- ◯ <i>
- ◯ <b>
- ◯ <q>

Kako ustvarite komentar v jeziku HTML?

- ◯ <!-- komentar -->
- ◯ /* komentar */
- ◯ // komentar

Katera oznaka ustvari seznam s pikami spredaj?

- ◯
- ◯ <dl>
- ◯
- ◯

Katera oznaka mora biti v jeziku HTML, da lahko zaženete kodo Javascript?

- ◯ <js>
- ◯ <src>
- ◯ <script>

◯ <body>

Ali ima lahko funkcija v javascriptu več parametrov?

◯ da

◯ ne

Kateri znak moramo vstaviti v tem primeru, če želimo ugotoviti, ali je vrednost x 5, pri čemer vrsta spremenljivke ni pomembna? if (x ??? 5) {};

◯ ==

◯ =

◯ ===

◯ ====

Kateri je prvi objekt v HTML DOM?

◯ <p>

◯ <html>

◯ document

◯ <head>

Želimo spremeniti oznako <p>. Katero metodo lahko uporabimo, če v oznaki <p> ni id?

◯ setAttribute

◯ getElementsByClassName

◯ getElementById

◯ getElementsByTagName

Katere elemente lahko uporabimo za ustvarjanje novih elementov HTML?

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

Zakaj se uporablja innerHTML?

- ◯ za dostop do vsebine dokumenta HTML
- ◯ za dostop do vsebine oznake <html>
- ◯ za dostop do podrejenega vozlišča iz trenutnega vozlišča

Kaj naredi metoda appendChild()?

- ◯ doda novega otroka k otroku.
- ◯ doda nov element nadrejenemu.
- ◯ doda podrejeni element sorodniku.

Kaj pomeni childNodes[1]?

- ◯ vzel bo drugi element vrste, ki jo iščemo.
- ◯ vzel bo prvi element vrste, ki jo iščemo
- ◯ je enak kot lastChild

Katero metodo lahko uporabimo, če želimo obstoječemu elementu na zadnjem mestu dodati nov element?

- ◯ appendChild
- ◯ insertBefore
- ◯ replaceChild

Katero možnost lahko uporabimo v obrazcu za klic funkcije?

- ◯ action
- ◯ onsubmit
- ◯ href

Katero lastnost lahko uporabimo za dostop do vsebine polja v obrazcu HTML?

- ◯ isNaN
- ◯ year
- ◯ value

S čim lahko dostopamo do številke izbrane možnosti v spustnem seznamu?

- ◯ checked
- ◯ selectedIndex
- ◯ value

S čim lahko preverimo, ali je bilo polje v obrazcu izbrano?

- ◯ checked
- ◯ selectedIndex
- ◯ value

Ali lahko spremenimo CSS s pomočjo DOM brez uporabe dogodkov?

- ◯ da
- ◯ ne

S katero lastnostjo lahko spremenimo barvo besedila?

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

S katero lastnostjo lahko spremenimo barvo ozadja?

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

Kako lahko ustvarimo nov <tr> v tabeli?

- ◯ insertCell()
- ◯ insertRow
- ◯ insertTableRow