Práca s elementami v JS

Výber elementov

Na výber elementov možno využiť viacero metód, my však budem využívať hlavne:

- querySelector()
- querySelectorAll()

Ide o metódy objektu document – voláme ich teda nad týmto objektom. Obe metódy majú jeden parametre – reťazec (string) pozostávajúci z jedného alebo viacerých selektorov. Tento reťazec pritom musí byť platným reťazcom selektora CSS.

Selektor pre výber konkrétneho typu elementu napr. div

```
"div"
```

Selektor pre výber elementu konkrétnej triedy napr. s názvom box

```
".box"
```

Selektor pre výber elementu s konkrétnym id napr. submit-button

```
"#submit-button"
```

Kombinácia selektorov napr. pre výber elementov triedy box avšak len takých, ktoré sú potomkami elementu s id box-container.

```
"#box-container > .box"
```

Rozdiel medzi vyššie uvedenými metódami pritom spočíva v ich návratovej hodnote. Zatiaľ čo prvá metóda vracia objekt element predstavujúci prvý element v HTML dokumente, ktorý zodpovedá zadanej množine CSS selektorov, druhá z metód vracia list všetkých elementov zodpovedajúcich zadaným CSS selektorom. To možno vidieť aj pri nasledujúcej práci s konzolou.

```
> document.querySelector(".box");
<   <button class="box" onclick="deleteBox(event)">1</button> flex
```

```
> document.querySelectorAll(".box");
<  NodeList(6) [button.box, button.box, button.box, button.box,
   button.box, button.box]
     0: button.box
     1: button.box
     2: button.box
     3: button.box
     4: button.box
     5: button.box
     length: 6
     [[Prototype]]: NodeList
```

Dynamické vytvorenie elementu a pridanie atribútov

Na vytvorenie elementu budeme využívať metódu objektu document createElement() pričom jej budeme odovzdávať jeden argument a to reťazec, ktorý určuje typ elementu, ktorý sa má vytvoriť (názov tagu).

Takto vytvorený element možno pridať do HTML dokumentu napríklad pomocou metódy append(), pričom objekt nad ktorým voláme túto metódu bude predstavovať rodičovský element do ktorého chceme pripojiť element odovzdaný tejto metóde ako argument.

Majme nasledujúci fragment HTML dokumentu

```
<div id="box-container">
    <div class="box">1</div>
</div>
```

Potom čo sa vykoná nasledujúci fragment kódu

```
let newElement = document.createElement("span");
let boxContainer = document.querySelector("#box-container");
boxContainer.append(newElement);
```

Bude vyššie uvedená časť HTML dokumentu vyzerať nasledovne

```
<div id="box-container">
    <div class="box">1</div>
    <span></span>
</div>
```

Pridávanie resp. zmenu atribútov jednotlivých elementov možno využiť metódu setAttribute(). Táto metóda je volaná nad elementom, ktorému chceme nastaviť daný atribút a má dva parametre name a value, teda názov atribútu, ktorý chceme nastaviť a hodnotu na ktorú chceme daný atribút nastaviť. Napr. Nastavenie atribútu class novovytvorenému elementu na box volaním metódy setAttribute.

```
newElement.setAttribute("class","box");
```

Pre pristupovanie k jednotlivým atribútom nejakého elementu možno použiť aj bodkovú notáciu (všimnime si však, že názvy atribútov sa môžu pri ich reprezentácii v JS líšiť od názvu atribútov uvádzaných v rámci HTML dokumentu napr. class - className). Napr. Nastavenie atribútu class novovytvorenému elementu na box využitím bodkovej notácie.

```
newElement.className="box";
```

The best practice: Druhým spôsobom postupujeme hlavne v prípade ak chceme meniť resp. nastavovať atribút style nejakého elementu.

Pri nastavovaní atribútu style si však musíme uvedomiť, že jeho hodnota je v podstate tiež objektom a jednotlivé vlastnosti tohto objektu v podstate predstavujú vlastnosti uvádzané v rámci CSS ako napríklad: color, display, background...

Príklad:

```
newElement.style.backgroundColor = "red";
newElement.style.display = "block";
newElement.style.color = "violet";
```

Eventy

Udalosti (events) sú v podstate reprezentáciou toho, že nejaká činnosť bola vykonaná používateľom alebo prehliadačom. V prípade aj je JS kód pripojený do HTML, JS na tieto udalosti môže reagovať spustením určitej časti kódu. Tento proces reakcie na udalosti sa nazýva Event Handling. JS teda spracováva udalosti HTML prostredníctvom obslužných programov (funkcii) udalostí.

Jedna zo základných druhov udalosti je tzv. onclick udalosť – ktorá predstavuje stlačenie tlačidla. Zostáva už len uviesť ako naviazať vykonanie určitej časti JS kódu na túto udalosť. Na to nám slúži atribút onclick ako si to môžeme všimnúť na nasledujúcom príklade.

```
<div class="button" onclick="addBox()">Add Box</div>
```

Máme teda tlačidlo Add Box, v prípade ak klikneme na toto tlačidlo pôjde o onclick udalosť, ktorá vyvolá addBox funkciu deklarovanú v JS kóde pripojenom k tomuto HTML dokumentu. Funkcia addBox je teda handler tejto udalosti (kliku na tlačidlo Add Box).