JavaScript

Jiří Zacpal

KATEDRA INFORMATIKY
UNIVERZITA PALACKÉHO V OLOMOUCI

KMIWEBA Webové aplikace

Napojení na API

- Jazyk JavaScript (JS) je základní webová technologie, která umožňuje manipulovat s webovou stránkou na straně klienta (přesněji řečeno s DOM).
- Později ukážeme, že toho umí mnohem více, nyní se zaměříme na základy klientského JavaScriptu.

Rozšíření syntaxe

```
// proměnné
var x = true;
x = false;
console.log(x);

// konstantní proměnné
const x = true;
x = false; // způsobí chybu

// operátor let, lexikální rozsah platnosti na úrovni bloku
var x = 42;

if (x) {
  var x = 0;
  console.log(x); // vypíše 0
}
console.log(x); // vypíše 0!
```

Rozšíření syntaxe

```
// varianta s let
var x = 42;

if (x) {
  let x = 0; // omezení rozsahu platnosti na blok
  console.log(x); // vypíše 0
}

console.log(x); // vypíše 42

// analogicky pro cyklus for
```

Rozšíření syntaxe

```
// template string

// old way
console.log(fisrtName + ", " + lastName);

// new way
console.log(`${fisrtName}, ${lastName}`); // pozor na znak `
```

Rozšíření syntaxe

```
// funkce
// výchozí parametr funkce
function show(x = 42) {
  console.log(x);
}

show(); // 42

// šipkový operátor (anonymní funkce)
const log = x => console.log(x);
log(42);

// při vrácení objektů je třeba navíc ()
const user = (firstName, lastName) => ({givenname: firstName, surname: lastName});
console.log(user("Samantha", "Carter"));
```

Rozšíření syntaxe

```
// object literal enhancement (nový způsob vytváření objektů)

var givenname = "Samantha";
var surname = "Carter";

const user = {
  givenname,
  surname,
  log() {
    console.log(`${this.givenname}, ${this.surname}`);
  }
};

user.log();

// destrukturalizace objektu
const blah = {test1: 42, test2: 43};

var {test1} = blah;
console.log(test1);
```

Praktické ukázky

Změna vzhledu

```
// změna vlastnosti
const element = document.querySelector(".box"); // vrací první nalezený
element.style.color = "red";

// přidání a odebrání třídy
element.classList.add("box--marked");
element.classList.remove("box--marked");

// více elementů
var color = "blue";

const elements = document.querySelectorAll(".box");
elements.forEach(e => e.style.color = color);
```

Validace formuláře

- Základní validace je poskytována na úrovni prohlížeče. Například atribut required. Tato validace je poskytována formou API a JS se na toto API umí napojit. Uvažme následující vstupní pole.

```
<form method="post" action="">
    <label for="mail">E-mail address:</label>
    <input type="email" id="mail" name="mail">

    <button type="submit">Submit</button>
  </form>
```

Validace formuláře

```
const email = document.getElementById("mail");
const form = document.querySelector("form");

// validace při psaní
email.addEventListener("input", e => {
  if (email.validity.typeMismatch) {
    email.setCustomValidity("I am expecting an e-mail address!");
    email.reportValidity();
  } else {
    email.setCustomValidity("");
  }
});

// validace při odeslání
form.addEventListener("submit", e => {
  if (!email.value) {
    showError(email);
    event.preventDefault(); // zabránění odeslání formuláře
  }
});
```

Validace formuláře

```
// pomocná funkce
function showError(e) {
  e.style.backgroundColor = "#ffcccb"; // vestvěné hlášky sami zmizí
  e.setCustomValidity("E-mail have to be entered!");
  e.reportValidity();
}

// poznámka:  lze snadno rozšířit, třeba přidáním span s display: none za input a například
// e.nextElementSibling.style.display="inline";
```

Modální okno s potvrzením

- Následující kódy ukazují možné řešení vyvolání modálního okna, které obsahuje žádost o potvrzení akce. Uvedené řešení je jedno z mnoha možných. Zcela univerzání řešení neexistuje.

```
<!-- tlačítka -->
<button class="button button--delete" data-action="users/delete/1">Delete
forever</button>
<button class="button button--delete" data-action="users/delete/2">Delete
forever</button>
<button class="button button--delete" data-action="users/delete/3">Delete
forever</button>

<!-- modální dialogové okno -->
  <dialog id="dialog">
  <p id="dialog__text">Do you really want to delete item <span id="dialog__item-to-
delete">X</span>? This action cannot be undone.</p>
  <a href="" id="dialog__confirm-link">yes</a>
  <a href="#" onclick="closeDeleteDialog()">no</a>
</dialog>
```

Modální okno s potvrzením

```
// najdeme všechny delete tlačítka
const deleteButtons = document.querySelectorAll(".button--delete");

// navázání události
deleteButtons.forEach(b => b.addEventListener("click", e => {
  const dialog = document.getElementById("dialog");
  const action = b.dataset.action; // data z data-* atributu

  // parsování id z data atributu
  const item = document.getElementById("dialog__item-to-delete");
  item.innerHTML = action.split("/").pop();;
```

Modální okno s potvrzením

```
 // změna linku
  const link = document.getElementById("dialog__confirm-link");
  link.setAttribute("href", action);

  dialog.showModal();
}));

// zavření dialogového okna
function closeDeleteDialog() {
  const dialog = document.getElementById("dialog");
  dialog.close();
}
```

Mizející notifikace

```
// najdeme všechny notifikace
const notifications = document.querySelectorAll(".notification");

// po 3s je odstraníme
setTimeout(() => {notifications.forEach(n => n.style.display="none")}, 3000);

// nebo fadeout efekt
var opacity = 1;
var fadeOutInterval = setInterval(() => {
  opacity = opacity - 0.01;
  notifications.forEach(n => {
    n.style.opacity = opacity;
    if (opacity <= 0) {
      n.style.display = "none";
      clearInterval(fadeOutInterval);
    }
  })}, 30);
```

Mizející notifikace

```
// případně hezčí a čistější řešení pomocí rekurze
const fadeOutEffect = (opacity = 1, delay = 30, decrement = 0.01) => {
  const notifications = document.querySelectorAll(".notification");

  if (opacity > 0) {
    notifications.forEach(n => n.style.opacity = opacity);
    setTimeout(() => fadeOutEffect(opacity - decrement, delay, decrement), delay);
  }
  else {
    notifications.forEach(n => n.style.display = "none");
  }
};

fadeOutEffect();

// poznámka: stejně tak je možné funkci spustit přes IIFE (() => {})();
```

Kam dál

- Možnosti JS jsou velmi rozsáhlé.
- V případě jednoduchých webových stránek se velmi často opakují běžné uživatelské prvky. Například rozklikávací menu, modální okna, úpravy UI dle požadavků, fotogalerie, slideshow a další.
- Dodejme, že existuje řada JS knihoven, které usnadňují vytváření, nebo přímo obsahují uvedené prvky.
- Jedna z nejuniverzálnějších knihoven, je knihovna jQuery. Ta již není tak populární, jako před několika lety, ale stále se hojně používá.
- Současným trendem je využívání menších jednoúčelových knihoven a celkové odlehčení JS, pokud je to možné.

Úkoly

ÚKOL 1

- V administračním rozhraní z minulých seminářů přidejte, validaci na straně klienta, dotaz při mazaní a notifikaci o úspěšném vložení, která automaticky zmizí.