Úvod do JS, DOM

- volitelné nabízení pokročilých technologií
  - **progressive enhancement** = nabízet základní funkcionalitu, volitelně přidávat
  - **graceful degradation** = nabízet pokročilou funkcionalitu, v případě nouze couvnout

- **polyfill** = JS kód, schopný doplnit chybějící funkcionalitu při zachování kompatibilního API

```
if (!("onhashchange" in window)) {
    var oldHash = window.hash;
    setInterval(function() {
        if (window.hash == oldHash) return;
        oldHash = window.hash;
        if (window.onhashchange) window.onhashchange();
    }, 100);
}
```

- loguje se většinou přes konzoli (typicky F12)
  - funkce console.{log/warn/error/time/timeEnd};

- základní strukturou bývají objekty - definovány v {}

- **iterace polí** by se neměla dělata přes for-in
  - kvůli sparse arrays – new Array(10)…
  - … a obohaceným polím – Array.prototype.X = …
  - jde o výčet klíčů, těch může být jiný počet než hodnot
  - lépe přes funkcionální iteraci
    - aplikace uživatelem zadané funkce na položky pole
    - bývá kratší/expresivnější
    - může být anonymní či pojmenovaná
    - funkce se dají aplikovat třeba i na map/filter/reduce

```
var data = [15, "babicka", true];

/* anonymni funkce */
data.forEach(function(item, index) {
    console.log(item); // 15, "babicka", true
});

/* pojmenovana funkce */
function log(item, index) {
    console.log(index);
}
data.forEach(log); // 0, 1, 2
```

```
var data = [1, 2, 3];

function square(x) { return x*x; }
var data2 = data.map(square); // 1, 4, 9

function odd(x) { return x % 2; }
var data3 = data.filter(odd); // 1, 3
```

```
var data = [1, 2, 3];

function odd(x) { return x % 2; }
data.every(odd);  // false
data.some(odd);   // true

function add(x, y) { return x+y; }
data.reduce(add); // 6
```

- **IIFE (immediately-invoked function expressions)**
  - problém se strukturováním (modularizací) JS kódu
  - soubory bývají příliš velké a musí se rozdělit do vícera souborů
  - pak mohou nastat problémy se sdílením proměnných / jmenných prostor
  - funkce ve scriptech sdílí přístup k proměnným apod. → not good

```
(function(){
    var document = "test"; // lokalni promenna
    alert(document);       // "test"
})();

alert(document);           // [object HTMLDocument]
```

DOM (Document Object Model)

- funkce, konstanty, proměnné apod. pro manipulaci se stránkou
- bývá dostupné pomocí globální proměnné **document**

Úpravy podstromu

```
var p = document.querySelector("p");

/* HTML parser, pozor na XSS! */
p.innerHTML = "<strong>toto je test</strong>";

/* jen text */
p.textContent = "<strong>toto je test</strong>";
```

Tvorba nových prvků

```
var p = document.querySelector("p");

var strong = document.createElement("strong");
p.appendChild(strong);

var text = document.createTextNode("toto je test");
p.appendChild(text);

var input = document.createElement("input");
input.type = "number";
input.id = "foo";
```

Práce s atributem class

```
var p = document.querySelector("p");

p.className = "class1";

p.classList.add("class2");
p.classList.remove("class3");
p.classList.contains("class2"); // true

p.classList.toggle("class4");
p.classList.toggle("class4", x > 15);
```