VŠB TECHNICKÁ UNIVERZITA OSTRAVA | FAKULTA ELEKTROTECHNIKY A INFORMATIKY | KATEDRA INFORMATIKY

Základy informačních technologií

Ing. Michal Radecký, Ph.D. MBA

JavaScript

Co je to JavaScript

Skriptovací programovací jazyk (interpretovaný, zpracovává se přímo zdrojový kód) určený pro řešení dynamiky WWW stránek na straně klienta.

Nejen pro webový frontend (např. node.js)

- Součást zdrojového kódu HTML (DOM)
- Multiplatformní
- Závislý na interpretačním prostředí (prohlížeči)
- V principu objektově orientovaný, ale beztřídní (prototypy), dnes i nativní podpora OOP
- Case-senzitivní
- Syntaxí podobný jazykům typu C/C++/Java (někdy volnější, Python)
- Beztypový

Historie JavaScriptu

- První představení (LiveScript) v roce 1995 jako součást Netscape Navigatoru a jako reakce na potřebu zajištění interaktivnosti webových stránek jinými prostředky než Java Applety.
- Rychlé rozšíření s ohledem na téměř žádné vstupní požadavky (syntaxe C/C++/Java, žádný kompilátor, atd.)
- Reakce Microsoftu formu VBScriptu, který byl podporován pouze na platformě Windows. V roce 1996 byla v rámci IE 3.0 uvedena portace JScript.
- V roce 1997 byla standardizována verze **ECMAScript**, která poskytovala jádro společné a podporované všemi prohlížeči.
- Prohlížeče od HTML 4.0 podporovali **DOM (Document Object Model, Level 1)**, nicméně opět v různé implementaci (document.layers vs. document.all)
- Po roce 2000 W3C DOM až Level 3, který už prohlížeče interpretovaly univerzálně.
- Dnes součást HTML Living Standardu (WHATWG, bez verzí)

Co umí JavaScript

Současný **JavaScript** se vyvinul z jednoduchého skriptovacího jazyka
v **univerzální a komplexní nástroj** nejen pro web. Inspiroval se v jazycích **Perl, C/C++, Java, Python** i **TCL**.

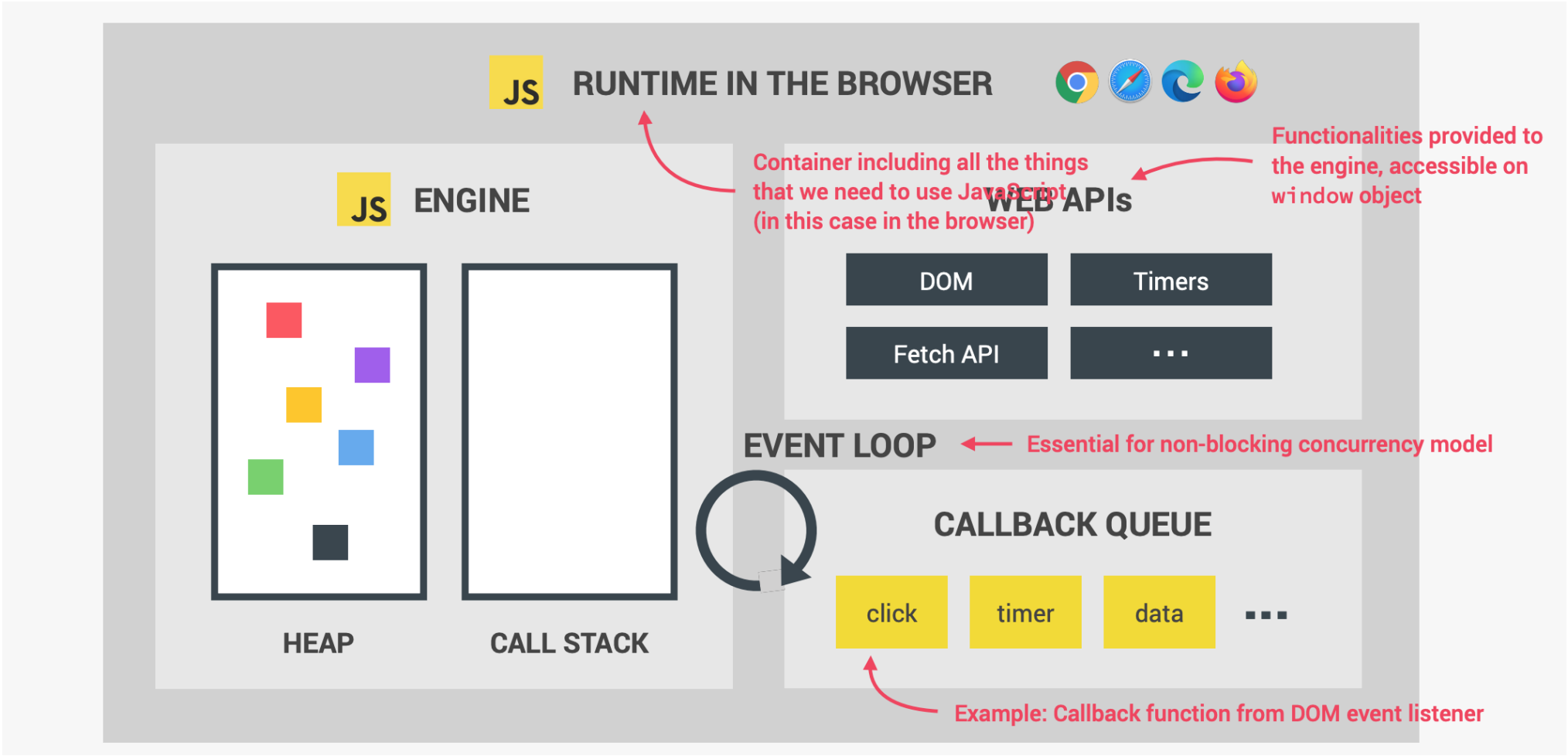
- Ovlivňovat vzhled a obsah HTML dokumentu (DOM)
- Manipulovat s obrázky a prvky stránky
- Provádět algoritmy a matematické výpočty
- Ovládat a zpracovávat formuláře a jejich data
- Reagovat na události uživatele (kliknutí, klávesy, pohyb myši)
- Pracovat s webovými API
- Pracovat s úložišti v prohlížeči (LocalStorage, SessionStorage) a datovými formáty (JSON, XML, Data URL, atd.)
- Číst a ukládat cookies
- Spolupracovat s externími rozhraními a pluginy

V prostředí webu je **pracovní prostor JavaScriptu omezen prohlížečem** – nemá plný přístup k systému (z důvodu bezpečnosti).

Co neumí JavaScript

- Přímo přistupovat k síti – může jen využívat rozhraní prohlížeče (např. HTTP fetch, WebSocket – ale ne TCP/UDP spojení)
- Číst nebo zapisovat soubory na lokálním disku (kromě File API, uživatelské dialogy a interakce)
- Provádět autentizaci a autorizaci sám o sobě (využívá protokoly a služby serveru – např. OAuth, JWT)
- Spouštět aplikace nebo příkazy operačního systému
- Přímo kreslit vektorovou grafiku (přes SVG)
- Fungovat proti vůli uživatele

https://code-kage.hashnode.dev/what-is-javascript-and-how-it-works-behind-the-scene



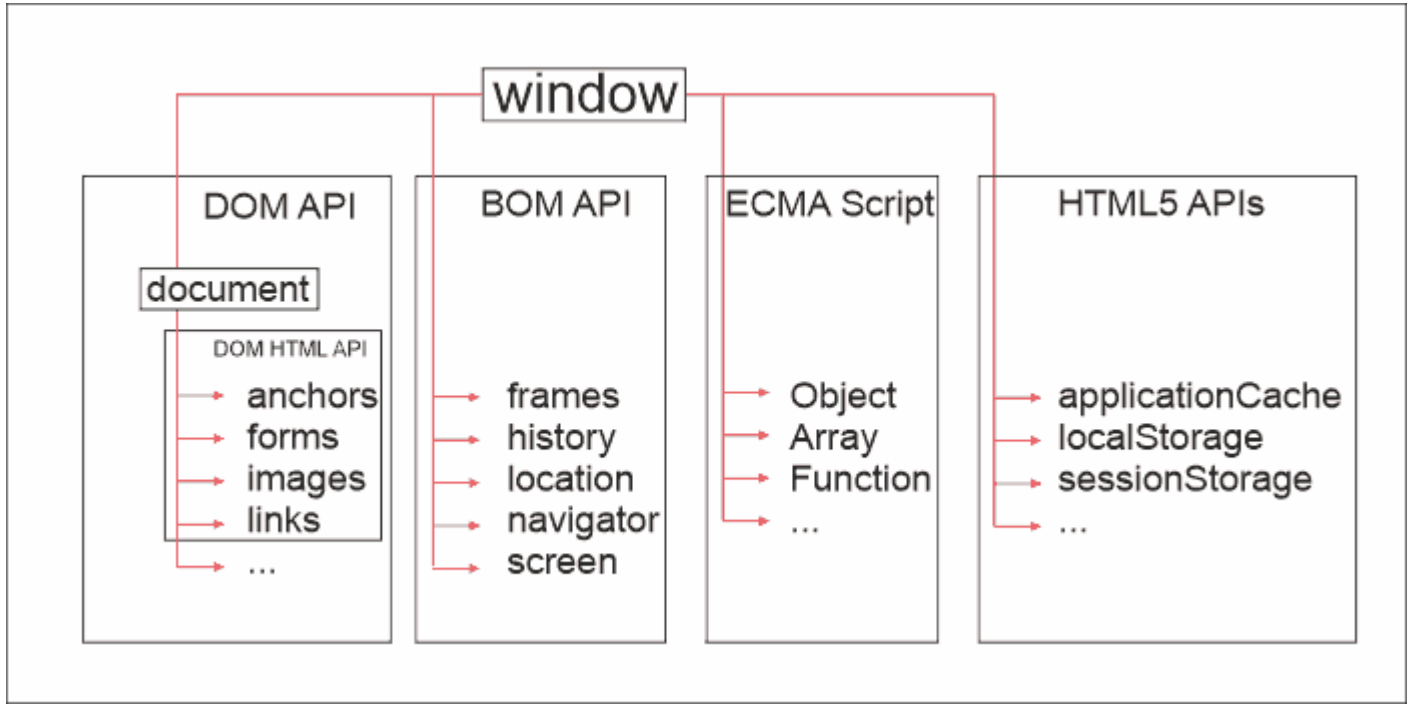
- Engine (V8 v Chrome): parser, interpreter JIT, heap, call stack
- V principu běží v jednom vlákně, asynchronní chování díky prohlížeči (event loop, callback queue)
- https://www.youtube.com/watch?v=eiC58R16hb8

DOM (Document Object Model)

- DOM je objektová reprezentace HTML dokumentu v paměti prohlížeče jako „podklad pro vizualizaci“.
- Každý prvek (tag, atribut, text) je uzlem (node) ve stromové struktuře.
- Umožňuje programový přístup k jednotlivým prvkům stránky – jejich vlastnostem, atributům a metodám, vč. napojení událostí.
- Změna DOMu znamená přepočítání a překreslení stránky prohlížečem.
- Virtuální DOM
  řešení v moderních frameworcích (React, Vue, …)
  vrstva mezi DOMem a aplikační logikou
  řeší optimalizaci změn v DOMu v čase – provádí jen ty změny, které jsou skutečně nutné
- https://www.w3schools.com/js/js_htmldom.asp

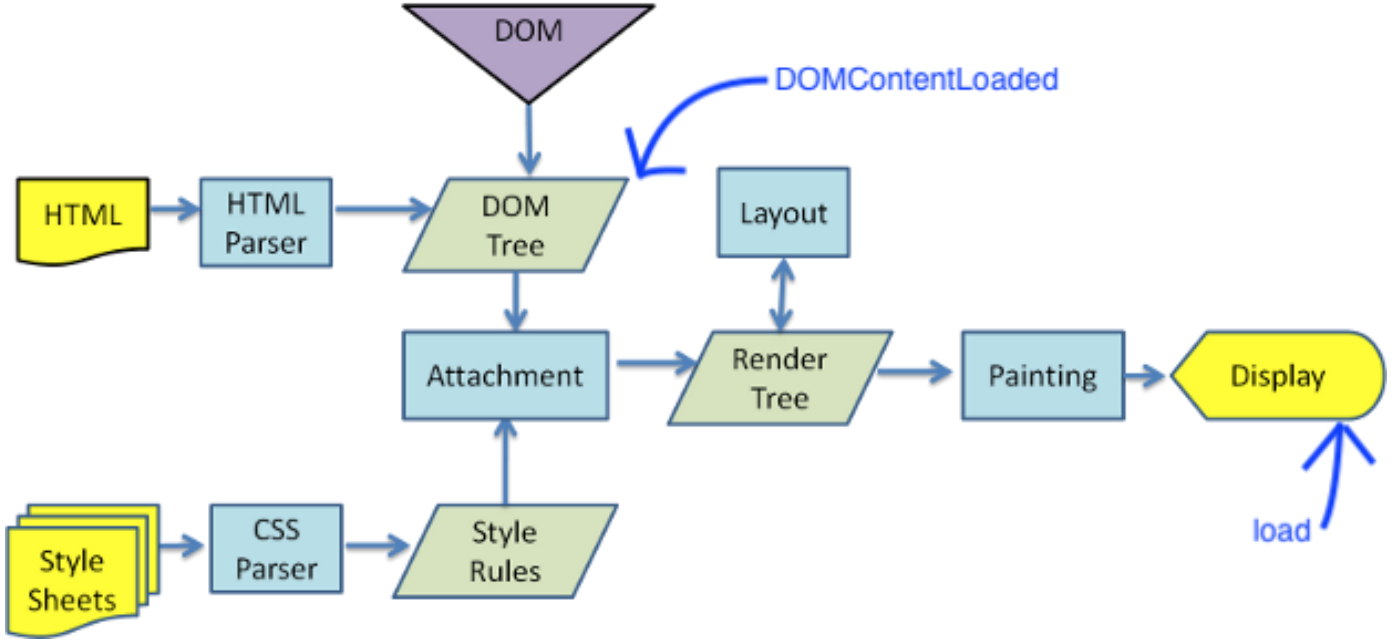
https://www.on-design.de/tutor/javascript

Základní objekty



- Objekt `window` je základní objekt prostředí prohlížeče – „globální kontejner“ pro všechno, co se na stránce děje. V jiných prostředích plní jeho úlohu jiné objekty např. `global`.
  - globální proměnné a funkce
  - přístup k prohlížečovým (HTML5) API
  - umožňuje komunikaci mezi okny nebo iframy
  - poskytuje informace o velikosti, poloze a stavu okna
  - umožňuje pracovat s událostmi na úrovni okna

Proces načtení DOMu




```
window.addEventListener("load", (event) => {
  console.log("page is fully loaded");
});

document.addEventListener("readystatechange", (event) => {
  console.log("page DOM is ready in different states");
});

document.addEventListener("DOMContentLoaded", (event) => {
  console.log("page DOM is loaded with defered scripts etc.");
});
```

Objekt document

Události jednotlivých elementů

- každá aktivita vyvolává událost a ta se šíří stromem (DOM) a může být zachycena libovolným prvkem (addEventListener, onX, v HTML)
- způsob šíření – capturing, bubbling
- přerušení – stopPropagation, preventDefault

Dotazování v rámci DOMu (rekurzivně)

- getElementById
- getElementsByTagName
- getElementsByClassName
- querySelector, querySelectorAll

Tvorba a modifikace DOMu

- innerText, innerHTML
- createElement, createTextNode
- appendChild

Asynchronní programování

Přepínání operací a Event Loop

- nedochází k blokování vlákna v případě operací vyžadujících čas
- fetch, setTimeout, addEventListener
- Event Loop zajišťuje zpracování fronty (message queue) – úlohy jsou vykonávány postupně, jakmile je hlavní vlákno volné
- pořadí zpracování nelze přesně řídit, závisí na čase dokončení a prioritě úloh.

Callback funkce

- předávání funkce/metody jako argument jiné funkce a její následné volání
- řetězení callback funkcí (callback hell)
- okamžité vykonávaní předávaných metod, pouze jedno vyvolání

Promise objekty, async/await

- funkcionální přístup - budoucí příslib návratu/hodnoty
- možnost řetězení a snadnější zachytávání chyb (z celého řetězce), násobné vykonávání
- async/await – automatizace Promise konstrukcí, píše se synchronně, ale funguje assynchronně

Web Worker

- umožňuje spouštět algoritmus v dalším vlákně

ECMAScript

- ECMAScript (ES) je standardizovaná specifikace jazyka JavaScript.
- Určuje, jak má jazyk fungovat – jeho syntaxi, datové typy, operátory, objekty, funkce atd.
- JavaScript = implementace tohoto standardu v prohlížeči (např. V8, SpiderMonkey, JavaScriptCore).

- Zlomový rok 2009 – ECMAScript 5 – základ pro HTML 5 API
- Moderní JavaScript – ES6+ (ES2015) – (let, const, class), moduly, arrow funkce, Promises … async/await

https://tc39.github.io/ecma262/

HTML 5 API (Web API)

LocalStorage, SessionStorage

- ukládání dat lokálně do prohlížeče v rámci aplikace
- dvojice klíč/hodnota

IndexedDB

- databázové úložiště na principu indexovaných objektů – větší množství strukturovaných dat
- dříve i WebSQL

Service Worker

- základní objekt pro off-line řešení webové aplikace
- algoritmus běžící mimo hlavní vlákno a zajišťující cachování síťové komunikace a další funkcionality (PWA)

Web Worker

- možnost spouštět algoritmy v samostatném vlákně
- má celou řadu omezení a je izolovaný – předání dat, spuštění, předání výsledku zpět

Web Socket

- vyspělé rozhraní pro obousměrnou asynchronní komunikaci (klient-server) pomocí otevřeného komunikačního kanálu (RFC 6455)
- nutná podpora na obou stranách komunikace (HTTP, TCP)

HTML 5 API (Web API)

Drag and drop

- je možné přesouvat jakýkoliv prvek v DOMu – atribut draggable
- implementace událostí ondragstart, ondrop, ondragover

File API

- přesunutí objektu z lokálního počítače do prostoru webové stránky
- vychází z principu Drag and Drop, kdy se zachytí událost „ondrop“ na odpovídajícím elementu
- přístup k přenášeným souborům pomocí DataTransfer.files
- File API poskytuje objekty File, FileList, Blob, FileReader, URL Base64

Fetch API

- moderní síťová komunikace na principu Promises
- oproti XHR (AJAX) přístupu, který používal callbacky
- lepší podpora pro JSON, CORS, atd.

Geolokalizace

- možnost získání GPS (latitude, longitude, altitude, accurancy, speed, timestamp) souřadnic polohy uživatele
- podmíněno povolením ze strany uživatele a zabezpečeným připojením
- závislé na technických možnostech zařízení, příp. lokalizace pomocí internetového připojení
- API přístupné v objektu navigator.geolocation

https://web.dev/articles/devices-introduction

JavaScript frameworky

Jedná se o JavaScriptové knihovny, které zjednodušují a rozšiřují možnosti standardního JavaScriptu. Díky JS frameworku se může vývojář více soustředit na samotné řešení, nikoliv na optimalizaci a ladění pro různé prohlížeče, koncepty, apod.

- Jedná se v podstatě o skripty (napsané v JS), které rozšiřují stávající objekty, metody, apod.
- Obvykle mají možnost využití a výběru z velkého množství pluginů či principů, které již řeší obvyklé aktivity a funkce (animace, AJAX, DOM, atd.)

Knihovna (library)

- poskytuje sadu funkcí, které volá vývojář (např. jQuery, D3.js)

Framework

- definuje strukturu aplikace a volá kód vývojáře (např. React, Angular, Vue)
- komplexní vývoj moderních webových aplikací
- Výhody: rychlejší vývoj díky opakovanému použití komponent, menší starost o kompatibilitu prohlížečů, udržovatelný modulární kód, integrace pokročilých nástrojů (routing, state management, build), aktivní komunita a ekosystém pluginů