

JavaScript Lekce #1

Organizace JS části kurzu



Visual Studio Code



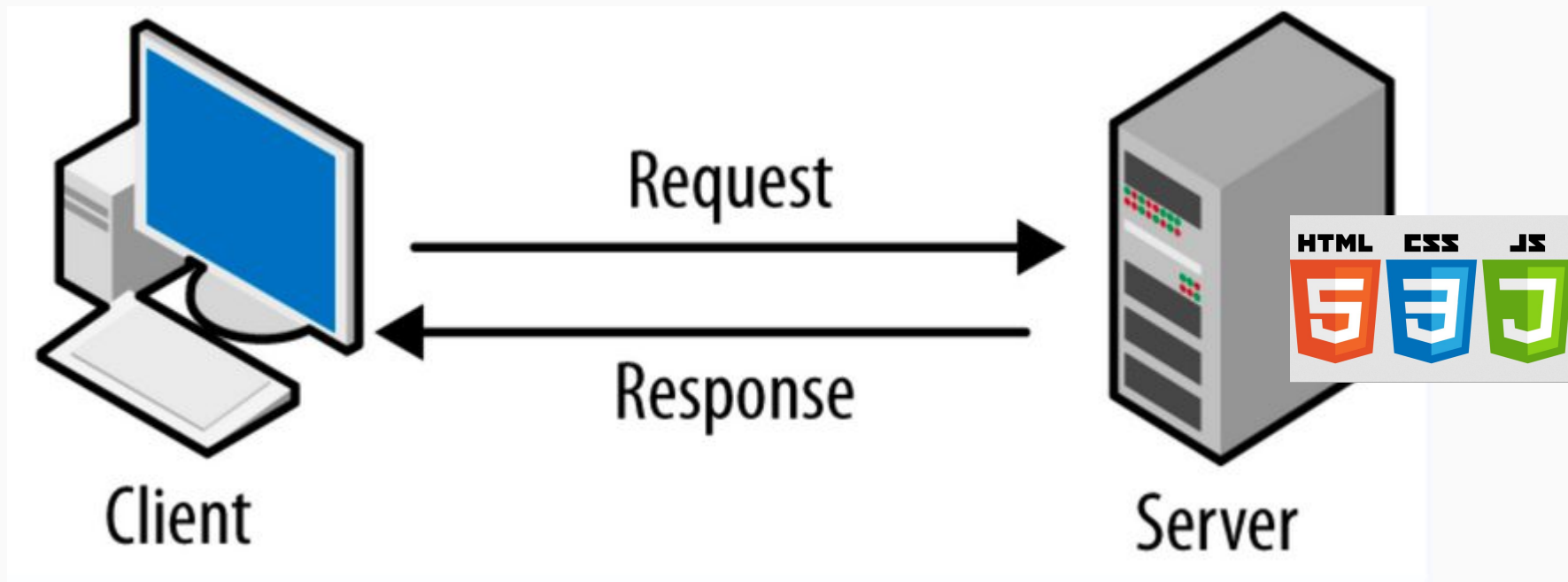
Úvod do JavaScriptu

- Co vlastně je JavaScript?
- Definice

Javascript is a **high-level**, **object-oriented**, **multi-paradigm** programming language.

- Role JavaScriptu ve web developmentu
 - umožňuje vytvořit webové stránky interaktivní




Jak to celé funguje?



Využití JavaScriptu

- Dynamické efekty, webové aplikace v prohlížeči
- Webové aplikace na web serverech
- Mobilní aplikace
- Desktopové aplikace

HTML vs. CSS vs. JS

			
Language	HTML	CSS	Javascript
Purpose	Structure, Objects, Things	Looks, Style	Actions
Syntax	<code><p> <h1>
</code>	<code>P {color: red;}</code>	<code>var x = 5;</code>
Grammar	nouns	adjectives	verbs
Building	Walls, structure	Paint, curtains	Electrical, Plumbing, AC

Historie JavaScriptu

- Začátky JavaScriptu
- ECMAScript
- JavaScript dnes

První psaní JavaScript kódu

- Vytvoření základního balíčku index.html, style.css, script.js
- Script tag src
- JS v HTML

První proměnná

- Jak se tvoří proměnná
- Jak proměnné fungují a proč je používáme
- Jaké informace mohou proměnné uchovávat
- ES6 vytvoření proměnné
- První práce s konzolí
- camelCase

Prakticke cvicenie

- Vytvořte html file
- Vytvořte js file
- Vytvořte proměnnou myName, která bude obsahovat Vaše jméno
- Proměnnou vytiskni do konzole

Vývojářské nástroje - základy

- Co to je
- K čemu slouží
- Kde je najdu
- Jak mi pomůžou

Vytváření proměnných - good practice

- Vždy používáme ES6 deklaraci proměnných `let`, `const`
- Vždy vytváříme proměnnou na svém samostatném řádku
- Seskupování stejných proměnných
- Tvořit proměnné nejblíže místu kde je potřebujeme využít
- Nepoužívat řetězení pro deklaraci proměnných
- Používat závorky pokud je proměnná dlouhá a chceme ji na dalším řádku

Break #1 (18:50 - 19:00)

Deklarace proměnných - let, const

- **ES6 (2015)**
- **let**
 - hodnotu proměnné **je možné** v programu měnit
 - **je možné** deklarovat proměnnou bez hodnoty
- **const**
 - hodnotu proměnné **není možné** v programu měnit
 - **není možné** deklarovat proměnnou bez hodnoty

Datové typy

- Každá proměnná má určitý datový typ
- JS je **dynamicky typovaný** jazyk - nedefinujeme datové typy manuálně
- operátor **typeof**
- **Objekt** vs. **Primitivní** datový typ

Primitivní datové typy - rozdělení

1. **String** - Text
2. **Number** - Čísla
3. **Boolean** - Pravdivostní hodnota
4. **Undefined** - Proměnná neobsahující žádnou hodnotu
5. **Null** - Proměnná obsahuje hodnotu, ta je však definovaná jako prázdná
6. **Symbol** - (ES 2015) - Unikátní hodnota
7. **BigInt** - (ES 2020) - Čísla větší než je bezpečné rozmezí pro typ Number

String

- "Engeto", '51'
- Řetězec znaků
- Uvnitř **uvozovek** - různé typy (" , ' `)
- Uvozovky uvnitř stringu?
 - použití různých typů, escapování
- Spojování stringů
 - operátor **+**, uvnitř `console.log()`
- Časté metody
 - `.includes()`, `.toUpperCase()`, `.toLowerCase()`, `.trim()`, `.replace()`, `.repeat()`

Number

- **8, 0, 3.14**
- Čísla - celá, desetinná
- Matematické operace
- Převádění Number na String a naopak
- Časté metody
 - `.toString()`, `.toFixed()`

Boolean

- **true, false**
- pravdivostní hodnota
- George Boole
- Využití hlavně v **podmínkách**
- Převádění jiných datových typů na Boolean
 - false: 0, -0, null, false, NaN, undefined, ""
 - true: cokoliv jiného

null, undefined

- Oba datové typy vyjadřují prázdnou proměnnou
- **undefined**
 - proměnná **byla** deklarována, ale **nebyla** jí přiřazena hodnota
- **null**
 - proměnná **byla** deklarována, a **byla** jí přiřazena hodnota null

prompt , alert, confirm

- built-in funkce sloužící pro interakci s uživatelem
- **alert()**
 - upozorní uživatele, po odkliknutí program pokračuje, vrací **undefined**
- **prompt()**
 - vyžaduje textový vstup uživatele, vrací **String**
- **confirm()**
 - vyžaduje potvrzení uživatele, vrací **Boolean**

Úloha

- Napíšeme program, který
 - Zeptá se uživatele (prompt) na jméno, uloží ho do proměnné **name**
 - Zeptá se uživatele (prompt) na věk, uloží ho do proměnné **age**
 - Upozorní uživatele (alert), že až bude o rok starší ($\text{age} + 1$), bude umět JavaScript (slovo JavaScript bude v uvozovkách)
 - BONUS - ošetřete, aby žádné vstupy uživatele neobsahovaly mezery před, ani za textem

Úloha - reseni

```
const name = prompt('Jak se jmenujes?');
```

```
const age = prompt('Kolik ti je let?');
```

```
alert("Tady sedi " + name + ". Az mu bude " + (Number(age) + 1) + ", bude umet  
JavaScript.");
```

Break #2 (19:50 - 20:00)

Aritmetické operátory

Aritmetické operátory, porovnávací operátory:

- operátory + - * / ** %
- operátory >, >=, <, <=
- operátory ==, !=, ===, !==

Externí linky:

- https://www.w3schools.com/js/js_arithmetic.asp
- https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/Math

Operator precedence

- `console.log(3 + 10 * 2);`
- `console.log(3 + (10 * 2));`
- `console.log((3 + 10) * 2);`

Externí link:

- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Operator_Precedence

JavaScript komentáře

Hlavním účelem psaní kódu je, aby jej počítač mohl interpretovat jako příkazy. Je však také důležité, aby kód, který píšeme, byl snadno interpretovatelný i pro ostatní vývojáře.

Komentáře jsou:

- poznámky zapsané v kódu, které jsou ignorovány enginem JavaScriptu
- s účelem je popsat ostatním vývojářům i vám, jak a proč kód funguje.

Jednořádkové komentáře

Jednořádkové komentáře se obvykle používají k okomentování části řádku nebo celého řádku kódu. Jednořádkové komentáře v jazyce JavaScript začínají znakem `//`. Interpret bude ignorovat vše napravo od této řídicí sekvence až do konce řádku.

Příklad:

```
// Print "Hello World" in the console  
console.log("Hello World");|
```

Jednořádkové komentáře

Pokud se jednořádkový komentář objeví na konci řádku kódu, nazývá se inline komentář.

Ty se obvykle používají k rychlému přidání poznámek:

```
let test = "Hello world"; // Assign value to variable text
```

Víceřádkové komentáře

Pokud chceme přidat poznámku, která je rozprostřena na více řádcích, můžeme zvolit víceřádkové komentáře nebo komentáře na úrovni bloku.

Víceřádkové komentáře začínají znakmi `/*` a končí `*/`:

```
/* This is a multiline JavaScript comment  
 * You can use this format with star at the beginning to align the code  
 * Your code editor should add it automatically  
 */
```

DocStrings

Následující komentář se označují jako **DocStrings**, protože jsou to v podstatě řetězce (komentáře), které tvoří dokumentaci vašeho kódu.

```
/**
 * Function that greets a user
 * @author Filip
 * @param {String} name Name of the user
 * @return {String} Greeting message
 */
function greetUser(name) {
    return `Greetings, ${name}!`;
}
```

DocStrings

Tento typ komentářů jsou opravdu užitečné pro ostatní vývojáře ve vašem týmu, protože můžete objasnit, jaký je očekávaný vstup, jaký je výstup, a také koho v případě potřeby kontaktovat.

Další výhodou je, že na základě těchto řetězců DocStrings můžete generovat dokumentaci.

Praktická část

Doplňte následující typy komentářů do vašeho kódu:

- komentář na jednom řádku
- inline komentář
- víceřádkový komentář

Komentáře - best practices

Především komentáře nejsou omluvou pro psaní nečitelného kódu, který se pak záplatuje pěti odstavci vysvětlujících komentářů. Nejprve se musíme zaměřit na psaní čistého, srozumitelného kódu, který můžeme později vylepšit konstruktivními komentáři.

V komentářích vysvětlujte, **proč** jste něco udělali, ne **jak** jste to udělali. Pokud se přistihnete, že vysvětlujete, jak jste něco udělali, pak je na čase udělat krok zpět a přeformulovat kód na něco, co se dá vysvětlit samo.

Komentáře - best practices

Další radou by bylo nepsat komentáře, které jsou zřejmé a ze své podstaty zbytečné. Například následující komentář je zcela zbytečný:

```
// Prints out the result  
console.log(result)
```

Práce s Developer Tools

Developer Tools je komplexní sada nástrojů pro vývojáře integrovaná přímo do prohlížečů. Tyto nástroje vám umožní upravovat webové stránky v reálném čase, rychleji diagnostikovat problémy a rychleji vytvářet lepší webové stránky.

Pro použití v rámci akademie budeme používat prohlížeče Chrome / Brave.

Práce s Developer Tools

The image shows a web browser window with the ENGETO logo in the top left corner. The main content area has a blue background with the text "STUDUJ IT ONLINE ZMĚŇ KARIÉRU" and a subtext "Nezáleží na tom, co umíš, kolik toho víš nebo kde pracuješ. Studuj online a rozjeď to v IT!". Below this is a green button labeled "ZAČÍT STUDOVAT ZADARMO". At the bottom, it says "ZÍSKEJ PRÁCI SNŮ U FIREM JAKO:" followed by logos for KIWI.COM, Red Hat, AT&T, IBM, and CGI.

Overlaid on the main content are two smaller screenshots. The top one is titled "PYTHON #2: FUNKCE A SMYČKY" and shows a code editor with Python code for a BMI calculator. The bottom one is titled "6. BMI kalkulačka" and shows a form with input fields and a "TEST" button. The code in the top screenshot is as follows:

```
# Kod (resetujte uživatele)
# Vstupní hodnoty
vaha = 80
vyska = 2
# Vypočet
bmi = vaha / vyska ** 2
# Vypis výsledek
print("Tvé BMI je", str(bmi))
```

The terminal output at the bottom of the code editor shows:

```
$ python3 main.py
Tvé BMI je 20.0
```

On the right side of the browser window, the developer tools are open to the "Console" tab. It shows a message "1 message" and a list of settings for the console, including "No user message", "1 error", "No warnings", "No info", "No verbose", "Hide network", "Preserve log", "Selected context only", "Group similar messages in console", "Show CORS errors in console", "Log XMLHttpRequests", "Eager evaluation", "Autocomplete from history", and "Evaluate triggers user activation".

Práce s Developer Tools

Části DevTools:

- **Console** - informace o chybách, debugovací hlášky, ...
- **Sources** - zdrojové kódy webové stránky (HTML, CSS, JS, obrázky, ...)
- **Elements** - úprava CSS atributů v reálném čase
- **Network** - seznam dotazů na server při načtení stránky (co všechno se stahuje, případně odesílá ze strany klienta)

Práce s Developer Tools

Části DevTools:

- Performance - manuálně testování stránky z pohledu rychlosti
- Memory - využití CPU a paměti RAM v rámci webové stránky
- **Application** - lokální storage a cache
- Security - přehled z pohledu bezpečnosti
- **Lighthouse** - automatizovaný audit stránky

Objekt console

- `console.log(console)`
- `console.warn()`
- `console.error()`
- `console.count()`
- a další

Jak neuspět při programování

Na začátku své cesty nemáš jasný cíl



Jak neuspět při programování

Na začátku své cesty nemáš jasný cíl

Řešení:

- Víš, proč se učíš programovat? Hledáš lepší práci? Novou kariéru?
- Představ si velký projekt / aplikaci, kterou chceš naprogramovat.
- V průběhu učení se dívej na technologie, které ti přijdou zajímavé.
- Stanov si konkrétní, měřitelný cíl, realistický a časově omezený cíl

Jak neuspět při programování

**Začneš sledováním kurzů a čtením výukových materiálů,
ale pak už jen kopíruješ kód bez zajmu o to, jak funguje.**



Jak neuspět při programování

Začneš sledováním kurzů a čtením výukových materiálů, ale pak už jen kopíruješ kód bez zajmu o to, jak funguje.

Řešení:

- Důležité je pochopit kód, který studuješ a píšeš.
- Vždy přepiš kód do svého počítače, nepoužívej copy & paste.

Jak neuspět při programování

Neupevňuješ si učivo drobnými úkoly nebo poznámkami



Jak neuspět při programování

Neupevňuješ si učivo drobnými úkoly nebo poznámkami

Řešení:

- Po osvojení nové funkce nebo konceptu ji ihned použij v kódu
- Dělejte si poznámky - formou komentářů nebo na papír
- Procvič se pomocí malých kódovacích cvičení a výzev (v průběhu kurzu od lektorů)
- Věnujte doma čas učení a zkoušení

Jak neuspět při programování

Nekódiš a nepřicházíš s vlastními nápady na projekty.



Jak neuspět při programování

Nekódiš a nepřicházíš s vlastními nápady na projekty.

Řešení:

- Programování ve volném čase je nejdůležitější věc pro posun
- Toto NENÍ volitelné! Bez praxe mimo kurzy se nikam neposuneš
- Vymýšlej vlastní nápady na projekty nebo kopíruj oblíbené stránky či aplikace, případně jen jejich části na začátku
- Nezůstávejte trčet v "výukovém pekle" (tutorial hell)

Jak neuspět při programování

**Rychle si frustrovaný/á, když tvůj kód není dokonale čistý
nebo efektivní.**



Jak neuspět při programování

**Rychle si frustrovaný/á, když tvůj kód není dokonale čistý
nebo efektivní.**

Řešení:

- Nezasekávej se při psaní dokonalého kódu.
- Prostě piš tuny kódu bez ohledu na jeho kvalitu.
- Čistý a efektivní kód přijde časem.
- Kód můžeš vždy později refaktorovat (vylepšovat).

Jak neuspět při programování

Ztrácíš motivaci, protože si myslíš, že nikdy nemůžeš vědět všechno.



Jak neuspět při programování

Ztrácíš motivaci, protože si myslíš, že nikdy nemůžeš vědět všechno.

Řešení:

- Přijměte skutečnost, že budete nikdy nebudeš vědět všechno
 - Většina programátorů to může potvrdit
- Soustřed' se jen na to, co potřebuješ abys dosáhl svého cíle!

Jak neuspět při programování

Učíš se izolovaně bez konzultace s lektorem / kolegy a ostatními studenty.



Jak neuspět při programování

Učíš se izolovaně bez konzultace s lektorem / kolegy a ostatními studenty.

Řešení:

- Komunita studentů a lektorů na Discordu - konzultace, Q&A, ...
- Vysvětlujte nové pojmy ostatním lidem. Když to dokážete vysvětlit, tak tomu opravdu rozumíte!
- Podělte se o své cíle, sdílejte svůj kód na Githubu

Jak neuspět při programování

Po absolvování online kurzů si myslíš, že si webovým vývojářem bez zkušenosti s vlastním projektem.



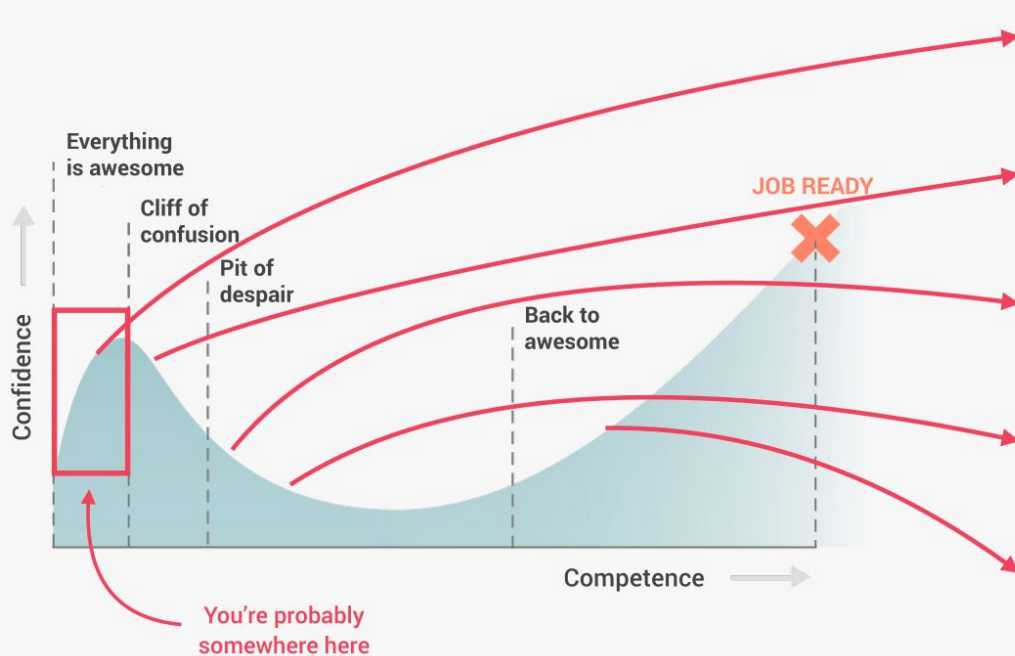
Jak neuspět při programování

Po absolvování online kurzů si myslíš, že si webovým vývojářem bez zkušenosti s vlastním projektem.

Řešení:

- Největší omyl, který lidé mají.
- Kurzy jsou úžasným startovním bodem, ale jsou jen začátkem cesty.

Jak neuspět při programování



Study courses: understand code, take challenges and notes

Stay motivated! Keep writing lots of code on your own, no matter how bad

Learn with other people, devs and beginners, and share progress

Keep challenging yourself, run into lots of problems, and fix them

Round up your skillset with best practices and tools (git, testing, ...)

JOB READY

(But the learning never stops 😊)

Jak neuspět při programování

Popis slide:

- Ucenie s lektorom, feedback, projekty
- Chceme, aby studenti boli medzi back to awesome a job ready
 - Ak sa ucite samostatne, tak nedosiahnete tuto oblast

Úkol na doma

- **Repl.it classroom úkol**
 - **1 vacsi ukol x 3 ukoly (1 pre kazdu cast)**