A logo with a swirl design

Description automatically generated

Učebnice

Důležitá terminologie a pojmy v IT

Ondřej Herian

© 2025 HeriART s.r.o. v Praze

**Čestné prohlášení**

Prohlašuji, že svou práci "Důležitá terminologie a pojmy v IT" jsem vypracoval samostatně s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne datum odevzdání \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Poděkování**

Rád bych touto cestou poděkoval Mně, za to, že jsem to nevzdal.

Důležitá terminologie a pojmy v IT

Abstrakt

Tato práce se zaměřuje na osvojení klíčové terminologie a pojmů v oblasti IT, přičemž slouží i jako výukový materiál pro zájemce o vzdělávání. Témata vycházejí z autorových zkušeností během vysokoškolského studia a praxe Softwarového inženýra. Ke správnému pochopení některých konceptů jsou definovány i pojmy mimo IT oblast. Práce využívá osvědčené metody učení, jako je ruční psaní poznámek (Mueller a Oppenheimer, 2014) a vizuální prezentace informací (Mayer, 2002), což napomáhá k lepšímu zapamatování a pochopení. Součástí práce jsou i grafické prvky, které podporují efektivní učení.

**Klíčová slova:** Softwarové inženýrství, IT terminologie, Efektivní učení, Vizualizace informací, Didaktické materiály

**Obsah**

[1 Úvod 9](#_Toc186972635)

[2 Cíl práce a metodika 10](#_Toc186972636)

[2.1 Cíl práce 10](#_Toc186972637)

[2.2 Metodika 10](#_Toc186972638)

[3 Odborné pojmy 11](#_Toc186972639)

[3.1 React 11](#_Toc186972640)

[3.1.1 Proč? 11](#_Toc186972641)

[3.1.2 Vlastnosti 11](#_Toc186972642)

[3.1.3 Komponenta 12](#_Toc186972643)

[3.1.4 Hooks (Hooky) 12](#_Toc186972644)

[3.1.5 JSX (JavaScript XML) 13](#_Toc186972645)

[3.2 DOM (Document object model) 14](#_Toc186972646)

[3.3 Node (Node.js) 14](#_Toc186972647)

[3.3.1 Kde ale běží samotný runtime? Musí mít také runtime? 15](#_Toc186972648)

[3.4 Vite vs NPX (Create React app) 15](#_Toc186972649)

[3.4.1 Výhody Vite oproti použití npx create-react-app (CRA) 15](#_Toc186972650)

[3.4.2 HMR (Hot Module Replacement) 16](#_Toc186972651)

[4 Terminologie 17](#_Toc186972652)

[4.1 Nativní (UI) 17](#_Toc186972653)

[4.1.1 Vlastnosti 17](#_Toc186972654)

[4.2 Nativní aplikace 17](#_Toc186972655)

[4.2.1 Vlastnosti 17](#_Toc186972656)

[4.3 Nativní funkce 17](#_Toc186972657)

[5 Výsledky a diskuse 18](#_Toc186972658)

[5.1 Podkapitola úroveň 2 18](#_Toc186972659)

[5.1.1 Podkapitola úroveň 3 18](#_Toc186972660)

[5.1.2 Podkapitola úroveň 3 18](#_Toc186972661)

[Závěr 19](#_Toc186972662)

[6 Seznam použitých zdrojů 20](#_Toc186972663)

[7 Seznam obrázků, tabulek, grafů a zkratek 21](#_Toc186972664)

[7.1 Seznam obrázků 21](#_Toc186972665)

[7.2 Seznam tabulek 21](#_Toc186972666)

[7.3 Seznam grafů 21](#_Toc186972667)

[7.4 Seznam použitých zkratek 21](#_Toc186972668)

[Přílohy 22](#_Toc186972669)

# Úvod

Tato práce má za úkol, aby si autor osvojil danou terminologii a pojmy, a zároveň učebnice mohla sloužit jako výukový materiál pro další osoby se zájmem vzdělávat se. Jsou zde řešena témata, na která autor během svého vysokoškolského studia a praxe jako Softwarový inženýr narazil. Témata jsou většinou úzce spojena, ale není to pravidlem. Ke správnému pochopení některých témat je občas nutné definovat pojmy, které nejsou přímo spojené s IT, ale jsou klíčové k pochopení jiných konceptů. Proces odborného zpracovávání zdrojů a informací v této práci napomáhá k jejich lepšímu zapamatování.

Podle výzkumu ručního psaní (Mueller a Oppenheimer, 2014) studie porovnávající ruční psaní a psaní na klávesnici zjistila, že studenti, kteří si ručně dělali poznámky, si zapamatovali více informací. Ruční psaní aktivuje hlubší zpracování informací, protože vyžaduje shrnutí a reorganizaci myšlenek. Proto si autor důležité pojmy nejdříve zpracoval ručně a následně byly převedeny do této práce. Studie na efektivním učení (Mayer, 2002) nám také říká, že Dr. Richard Mayer, expert na kognitivní vědy, ukázal, že strukturované a vizuálně prezentované informace (např. diagramy, grafy) vedou k lepšímu pochopení a zapamatování než čistý text. Proto se v práci budou, pokud možno objevovat i grafické prvky.

# Cíl práce a metodika

## Cíl práce

Cílem této práce je vytvořit ucelený materiál, který autorovi umožní osvojit si klíčové pojmy a terminologii z oblasti IT, a zároveň bude sloužit jako výukový materiál pro ostatní zájemce o vzdělávání. Práce má za úkol propojit získané poznatky z autorova vysokoškolského studia a praxe Softwarového inženýra s metodami efektivního učení, aby podpořila hlubší pochopení a zapamatování informací.

## Metodika

Práce vychází z odborných zdrojů, které jsou analyzovány a zpracovány do přehledné formy. Klíčové pojmy byly nejprve ručně zpracovány, což je podloženo výzkumem (Mueller a Oppenheimer, 2014), a následně převedeny do této práce. Využívány jsou také vizuální prvky, jako diagramy a grafy, v souladu s teorií multimediálního učení (Mayer, 2002). Témata jsou systematicky strukturována a doplněna o vzájemné souvislosti, aby napomohla lepšímu porozumění.

# Odborné pojmy

## React

JavaScriptová knihovna pro tvorbu UI. Zaměřuje se na vytváření komponent, což jsou v podstatě znovupoužitelné části kódů, které se kombinují. Existuje více metodologií návrhu UI, ale jednou z těch hlavních v Reactu je metodologie „ATOMIC Design“.

React developer píše React-specific JavaScript kód

### Proč?

1. Komponentový přístup (Znuvupoužitelnost, Přehlednost = *Clean code*)
2. Jednoduchá správa stavu (Pomocí Hooků)
3. Virtuální DOM (Rychlá a efektivní aktualizace UI)
4. JSX (JavaScript XML – Dovoluje psát dynamický JavaScript do HTML)
5. Ekosystém a flexibilita (Velký ekosystém knihoven a nástrojů)

### Vlastnosti

* One-way data flow
  + Data v aplikaci tečou jedním směrem. Jednosměrný tok dat v Reactu znamená, že data v React aplikaci vždy tečou jedním směrem: od rodiče k dítěti (Parent to Child). To znamená, že rodičovská komponenta může předávat data (tzv. props) své dětské komponentě, ale naopak to neplatí – dětská komponenta nemůže přímo měnit data rodiče.
* Single web page
  + Veškeré soubory jsou po Build procesu spojeny do jednoho index.html, ve kterém je veškerá funkcionalita, jen se pomocí JS vykresluje obsah, který je zrovna potřeba.
* Virtual DOM
  + Udržuje v paměti virtuální kopii DOMu => aktualizuje pouze elementy, které se změnili, ne celý DOM. Pomocí funkce diff porovná DOM s Virtual DOMem a upraví jen to co se liší.
* React Native
  + Umožňuje psát nativní mobilní aplikace.

### Komponenta

V Reactu je komponenta základní stavební blok aplikace. Je to funkce = „Functional component“ (může být i třída = „Class component“, ale to už je zastaralý přístup), která vrací část uživatelského rozhraní (UI), obvykle ve formě HTML kódu (psaného jako JSX). Komponenta může obsahovat logiku, stylování a definovat, jak by měla vypadat a fungovat určitá část aplikace.

* Komponenta = znovupoužitelný kousek kódu, který popisuje, jak by měla vypadat a fungovat konkrétní část uživatelského rozhraní.
* Jsou v základu stateless, ale díky React Hooks mohou mít stav.
* Přijímají props (vstupy) a vrací JSX, což definuje vzhled komponenty.
* Měli by být obecné a pomocí props upravovat výstupy.

### Hooks (Hooky)

Přidávají funkcionálním komponentám stav („useState“ hook) a umožňují využívat funkce životního cyklu („useEffect“ hook). Pokud se změní stav aplikace, tak React přerendruje relevantní elementy DOMu.

Životní cyklus komponenty = různé fáze, kterými komponenta prochází od svého vytvoření až po odstranění:

* Montování (Mounting):
  + Tato fáze nastává, když je komponenta vytvořena a vložena do DOMu. Spustí se při prvním vykreslení komponenty.
* Aktualizace (Updating):
  + Tato fáze nastává, když se změní stav (state) nebo vstupy (props) komponenty. Komponenta se znovu vykreslí, aby odrážela aktuální stav nebo props.
* Odmontování (Unmounting):
  + Tato fáze nastává, když je komponenta odstraněna z DOMu.

A screenshot of a computer program

Description automatically generated

1Životní cyklus komponenty (ANON. nedatováno)

### JSX (JavaScript XML)

Syntaxové rozšíření (Syntactic sugar = zjednodušení syntaxe, ale stejná funkcionalita) JavaScriptu, která se používají v Reactu. Dovolují psát HTML podobný kód, tedy psát dynamický JavaScript přímo do HTML.

* V Reactu se to používá v komponentách. Funkcí komponenty vracejí JSX.
* V realitě je při interpretaci JSX přeložen do JS funkcí („React.CreateElement“).

A screen shot of a computer

Description automatically generated

2JSX (ANON. nedatováno)

## DOM (Document object model)

Reprezentace HTML dokumentu v prohlížeči, se kterou prohlížeč pomocí JS může manipulovat.

* Manipulace pomocí DOMu se explicitně v Reactu nepoužívá, protože to React dělá za nás. My jen využíváme stavy (Hooky) a React na základě těchto stavů aktualizuje UI (manipuluje s DOMem)



3DOM (ANON. nedatováno)

## Node (Node.js)

Jedná se o JavaScript runtime environment (běhové prostředí) => Běhové prostředí, je prostředí, kde může být daný kód spuštěn. Je to v podstatě všechno, co je potřeba k tomu, aby byl programovací jazyk (kód) spuštěn a vykonán (software, nástroje, systémové zdroje). Runtime může například obsahovat:

1. Interpret / Compiler (Transformace programovacího jazyka do strojově čitelného kódu)
2. Knihovny
3. API
4. Správa paměti
5. Propojení s operačním systémem

Node umožňuje vykonávat JS mimo webový prohlížeč (na Backend serveru). Překládá JS do strojového kódu.

### Kde ale běží samotný runtime? Musí mít také runtime?

Node.js je runtime environment, který běží na operačním systému, ale sám o sobě není schopen běžet z „ničeho“. Potřebuje:

1. Operační systém
2. Shell / bash (Pro komunikaci s uživatelem, abychom ho mohli spustit) =>   
   1) Bash převezme příkaz “node app.js” a spustí proces Node.js  
   2) Node spustí a vykoná kód z „app.js“  
   3) Bash zároveň (real time) zachytává výstup z procesu do logů

## Vite vs NPX (Create React app)

Vite (vyslovuje se jako "vit", což znamená "rychlost" ve francouzštině) je moderní nástroj pro vytváření JavaScriptových projektů, navržený tak, aby poskytoval velmi rychlé buildy a vývojové prostředí.

Npx je nástroj, který je součástí Node.js (od verze 5.2.0 a novější). Slouží ke spouštění balíčků z npm registry bez jejich globální instalace. Příklad: Když použijete npx create-react-app, npx stáhne jednorázově balíček create-react-app z npm registry a spustí ho.

### Výhody Vite oproti použití npx create-react-app (CRA)

1. Rychlost vývoje – Využívá ESM (moduly přímo načítané prohlížečem),
   1. Nezačíná bundlovat celý projekt při startu.
   2. **Aktualizace během vývoje (HMR)** jsou okamžité, protože při změně se pře načítají pouze upravené moduly, ne celý projekt.
2. Lehkost a menší zátěž – Má menší závislosti a rychlejší instalaci.
3. Jednoduché nastavení – Nabízí konfiguraci přímo přes jednoduchý (vite.config.js).
4. Rychlost buildování - Používá Rollup pro buildování produkce, což je efektivní a optimalizované pro menší výstupy.
5. Modernější ekosystém – Otevřená architektura
6. Velikost výsledného buildu – Vytváří optimalizované balíčky s menší velikostí díky Rollupu.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

### HMR (Hot Module Replacement)

HMR (Hot Module Replacement) je funkce, která umožňuje okamžitě aktualizovat části kódu v běžící aplikaci bez nutnosti znovu načítat celou stránku.

V kontextu vývoje to znamená, že když provedete změnu ve vašem kódu, Vite dokáže aplikaci dynamicky aktualizovat pouze v té části, kde se změna odehrála, aniž byste přišli o aktuální stav aplikace.

# Terminologie

## Nativní (UI)

Nativní UI označuje uživatelské rozhraní (grafiku, tlačítka, menu atd.), které je vytvořené tak, aby odpovídalo vzhledu a chování operačního systému, na kterém aplikace běží.

### Vlastnosti

* UI vypadá a funguje jako součást operačního systému.
* Např. tlačítka, přepínače a animace vypadají přesně tak, jak je uživatel zvyklý z jiných aplikací na svém zařízení.

## Nativní aplikace

Nativní aplikace je software vytvořený přímo pro konkrétní operační systém (například iOS nebo Android).

Je napsán pomocí nástrojů a programovacích jazyků určených pro daný systém:

* iOS: Swift nebo Objective-C.
* Android: Kotlin nebo Java.

### Vlastnosti

* Aplikace je optimalizovaná pro danou platformu, takže má vysoký výkon.
* Přístup k nativním funkcím zařízení, jako je fotoaparát, GPS nebo push notifikace, je bez omezení.

## Nativní funkce

Funkce, která je přirozenou součástí systému („fetch“ je funkce, která je nativní funkce Reactu => už se nemusí stahovat)

# Výsledky a diskuse

## Podkapitola úroveň 2

Text…

### Podkapitola úroveň 3

Text…

### Podkapitola úroveň 3

Text…

# Závěr

Text…

# Seznam použitých zdrojů

ANON., nedatováno. *1\*v1SniQtV3REmyBGGXEJWKg.png (571×312)* [online] [vid. 4. leden 2025 a]. Získáno z: https://miro.medium.com/v2/resize:fit:571/1\*v1SniQtV3REmyBGGXEJWKg.png

ANON., nedatováno. *pic\_htmltree.gif (486×266)* [online] [vid. 4. leden 2025 b]. Získáno z: https://www.w3schools.com/js/pic\_htmltree.gif

ANON., nedatováno. Understanding React Hooks. *SST Guide* [online] [vid. 4. leden 2025 c]. Získáno z: https://guide.sst.dev/archives/understanding-react-hooks.html

MAYER, Richard E., 2002. Multimedia learning. In: *Psychology of Learning and Motivation* [online]. B.m.: Academic Press, s. 85–139 [vid. 4. leden 2025]. Získáno z: doi:10.1016/S0079-7421(02)80005-6

MUELLER a OPPENHEIMER, 2014. *The Pen Is Mightier Than the Keyboard: Advantages of Longhand Over Laptop Note Taking - Pam A. Mueller, Daniel M. Oppenheimer, 2014* [online] [vid. 4. leden 2025]. Získáno z: https://journals.sagepub.com/doi/10.1177/0956797614524581

# Seznam obrázků, tabulek, grafů a zkratek

## Seznam obrázků

[1Životní cyklus komponenty (ANON. nedatováno) 13](#_Toc186972684)

[2JSX (ANON. nedatováno) 13](#_Toc186972685)

[3DOM (ANON. nedatováno) 14](#_Toc186972686)

## Seznam tabulek

Odkazovaný seznam tabulek

## Seznam grafů

Odkazovaný seznam grafů

## Seznam použitých zkratek

Soupis a definování zkratek (vyskytuje-li se jich v textu velké množství)

# Přílohy

Odkazovaný seznam příloh