



Digitální učebnice algoritmů

Gymnázium, Praha 6, Arabská 14

obor programování

zpracoval Jakub Švéda

vedoucí Mgr. Jan Lána

29. dubna 2022

Prohlašuji, že jsem jediným autorem tohoto projektu, všechny citace jsou řádně označené a všechna použitá literatura a další zdroje jsou v práci uvedené. Tímto dle zákona 121/2000 Sb. (tzv. Autorský zákon) ve znění pozdějších předpisů uděluji bezúplatně škole Gymnázium, Praha 6, Arabská 14 oprávnění k výkonu práva na rozmnožování díla (§ 13) a práva na sdělování díla veřejnosti (§ 18) na dobu časově neomezenou a bez omezení územního rozsahu.

V dne

Podpis:

Mé poděkování patří panu profesoru Mgr. Janu Lánovi, který mi dovolil změnit téma práce v průběhu roku, neboť mi původní nevyhovovalo. Dále bych mu rád poděkoval za poskytování případné podpory po celou dobu tvoření práce. Velké díky patří také panu Doc. RNDr. Pavlu Töpferovi. Ovlivnil mě při výběru tématu práce. Dokázal ve mně vzbudit zájem o svět algoritmů a touhu po jejich poznání.

Název práce: Digitální učebnice algoritmů

Autor: Jakub Švéda

Anotace:

Cílem ročníkového projektu je vytvořit program, který poslouží jako učebnice, jež vysvětlí všechny základní algoritmy. Tedy takové algoritmy, které by měl každý programátor znát. Tento program bude vytvořen pomocí programovacího jazyka Java a jeho grafické knihovny JavaFXML.

The goal of the project is to create a program that serves as a textbook that explains all the basic algorithms. It means, algorithms that every programmer should know. This program will be created using the Java programming language and its graphical library JavaFXML.

Obsah

1. Úvod	6
2. Použité technologie	7
2.1. Java	7
2.1.1. JavaFX	8
2.1.2. Apache Maven	8
2.2. CSS	8
2.3. HTML	9
3. Učebnice	10
3.1. Určení	10
3.2. Výuka	10
3.3. Vývoj	12
4. Závěr	13
5. Bibliografie	14

1. Úvod

Tento byl vytvořen za účelem zvýšit povědomí o důležitosti algoritmů v programátorském světě. Je určen především pro začátečníky, kteří již mají základní povědomí o jazyku Java. Učebnice podrobně popisuje základní algoritmy a snaží se pochopitelně vysvětlit jejich princip. Pro vizuální zobrazení kódu jednotlivých algoritmů používá přehledné obrázky. Dále obsahuje testy, které zkouší zkušenosti nabrané v jednotlivých kapitolách

2. Použité technologie

Tato kapitola popisuje použité technologie a důvod jejich použití ve finálním programu.

2.1. Java

Java je objektově orientovaný jazyk založený na třídách (class). Snaží se tak o co největší obecnost finálního kódu. Jeho oblíbenost spočívá zejména na jeho všestrannosti. Jazyk je postaven tak, aby mohl fungovat na jakémkoli zařízení (většinou bez čipové architektury ARM) nebo operačním systému. V praxi programátor vytvoří aplikaci, která se přeloží ("zkompiluje") a je tak čitelná pro Java virtual machine (JVM). To je program, jež umí přeloženou Javu spustit. Zde je právě její hlavní výhoda. JVM lze nainstalovat prakticky na jakékoli zařízení a následně program spustit. U jiných programovacích jazyků, které tuto metodu nepoužívají, je nutné program znovu přeložit, tak aby mu operační systém rozuměl, což znamená především u rozsáhlejších programů mnoho práce počítače navíc.^[1]

Java byla vytvořena v roce 1995 firmou Sun Microsystems. Původně byla navržena pro použití v televizích. V roce 2006 se stala Java dostupnou široké veřejnosti. V roce pak 2010 převzala právně vývoj společnost Oracle. Zároveň se tím Java rozšířila i na nejpoužívanější systémy, tedy na macOS a Windows.

Hlavní výhoda Javy je již popsána výše tedy systémová všestrannost. Dalšími výhodami může být například její bezpečnost a stabilita. Na druhou stranu má i své nevýhody. Někteří lidé si často stěžují na velikost paměti, jež programy napsané v Javě zabírají. Dále na její menší rychlost jako daň za všestrannost, ale hlavně na její grafické knihovny. Na mnou použitou se podíváme v další podkapitole.^[2]

2.1.1. JavaFX

JavaFX je jedna ze tří grafických knihoven Javy. Oproti ostatním knihovnám (Swing a AWT) je jednodušší, pro člověka lépe čitelná a nabízí mnoho různých grafických komponent.^[3]

Pro snazší tvorbu samotného designu a vizuálních částí stránek jsem použil drobné rozšíření JavaFX o podporu FXML formátu. FXML je značkovací jazyk (markup language), pomocí kterého je jednoduché vytvořit předlohy budoucích oken, které se automaticky vykreslí díky JavaFX. Pro zaznamenání předlohy se používají soubory s příponou *.fxml*.^[4]

Pro pohodlné nedesignování stránek napsaných v FXML se používá mnoho grafických programů, které vytvoří její náhled v reálném čase. Já jsem při vývoji zvolil z nich asi nejlepší program SceneBuilder^[5]. SceneBuilder je open-source program plně dostupný na verzovací platformě GitHub^[6].

2.1.2. Apache Maven

Maven je nástroj, který udržuje základní informace o projektu (jeho základní popis, použité knihovny) a pomáhá při finálním vytváření cílové verze projektu, jež je čitelný pro počítače. Je velkým pomocníkem, když identifikujeme potřebné knihovny, pluginy a jiné externí zdroje projektu.^[7]

2.2. CSS

CSS neboli do češtiny přeložené Kaskádové styly je jazyk pro designování webových, ale i jiných aplikací. Pomocí CSS lze předefinovat vzhled stránky. Lze s ním například zarovnat prvky do středu, přidat některé animace nebo nastavit šířku určitých elementů před zobrazením. V mé aplikaci se vyskytuje ve dvou případech.^[8]

Prvním případem je design *.xml* souborů před tím, než se vykreslí. Pomocí něho je v programu například nastavena barva pozadí, zarovnání některých textů nebo design posuvné lišty (scroll bar) nebo tlačítek.

V druhém případě je použito při designování nadpisů a jiných elementů v souborech *.html*. O nich je řeč v následující podkapitole.

2.3. HTML

HTML je zkratka pro značovací jazyk propojený s hypertextovými odkazy. Je základním stavebním kamenem webových stránek. Dokáže zobrazit text, nadpisy a jiné elementy. V mém projektu se používá jako nástroj pro zobrazení obsahu kapitoly. Zvolil jsem ho kvůli jeho jednoduchosti a nedostatku jeho alternativ.^[9]

3. Učebnice

V této kapitole se podíváme na to, jakým způsobem program algoritmy vysvětluje. Dále vysvětlím, co mě osobně vedlo k vytvoření takové učebnice, a také komu je určena.

3.1. Určení

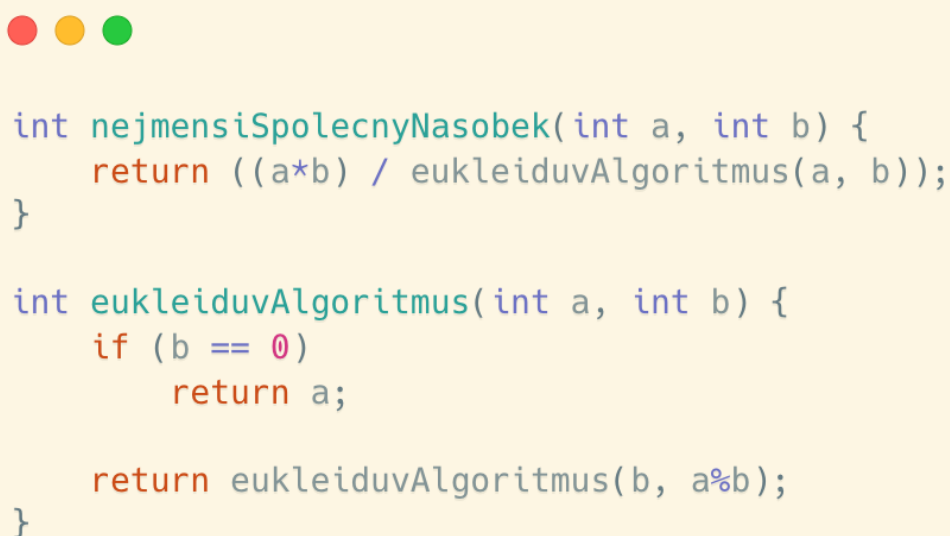
Učebnice je už od začátku stavěná tak, aby ji pochopil i nezkušený programátor. Přesněji řečeno je určená pro programátory, kteří mají již základní znalosti programovacího jazyka Java, ale nemají zkušenosti s algoritmy. Ty se jim tato učebnice snaží předat.

3.2. Výuka

Učebnice vysvětluje algoritmy od elementárních pojmů až po komplexnější algoritmy. Uvádí tak čtenáře do světa algoritmů. Při výkladu se snaží vzbudit myšlenky o těchto algoritmech přemýšlet. Zejména vzbudit myšlenky o tom, jak ušetřit práci nebo paměť jednotlivých algoritmů. V praxi učebnice představí algoritmus, který je napsaný nejjednodušším způsobem, ale je neefektivní. Poté postupně předkládá další vylepšené verze, čímž ukazuje jakou úvahou vylepšení dosáhl. Tímto způsobem by měla čtenáře naučit, jak o algoritmech přemýšlet. Principy, jak algoritmus zlepšit se totiž často opakují.

Když jsem hovořil o představení algoritmů, základním kritériem je jejich asymptotická časová složitost a samotná implementace algoritmu v Javě. Ta je zobrazena přehledným obrázkem, ve kterém je vepsaný blok kódu algoritmu. Takovéto obrázky byly vytvořeny webovou aplikací Carbon^[10]. Ta je přímo vytvořena k vytváření obrázku kódu, ve kterém barevně odděluje jednotlivé prvky (viz ukázka na obrázku 3.2.1.).

3.2.1. - Ukázka obrázku kódu



```
int nejmensiSpolecnyNasobek(int a, int b) {  
    return ((a*b) / eukleiduvAlgoritmus(a, b));  
}  
  
int eukleiduvAlgoritmus(int a, int b) {  
    if (b == 0)  
        return a;  
  
    return eukleiduvAlgoritmus(b, a%b);  
}
```

Důležitou poznámkou je, že z obrázků kód nezle kopírovat. Je tomu tak za záměrem donutit uživatele přemýšlet a naučit ho dané algoritmy. I přes to, že by kód opisoval přímo z aplikace, by si čtenář program zažil do paměti jeho přepsáním. Dále je to kvůli ztížení testů, o kterých bude řeč následovně.

Samotný program je rozdělen do deseti kapitol. Každá kapitola obsahuje algoritmy nebo pojmy týkající se jejich tématu. Některé obsahují také test. Test je zadání úlohy pro čtenáře. Úloha se týká vždy tématu, které se v kapitole vyskytlo. Má ověřit poznatky, které si uživatel ze čtení odnesl. Zadanou úlohu uživatel napíše dle instrukcí v libovolném programu. Po dokončení úlohy vloží její zdrojový kód s příponou `.java` pomocí příslušného tlačítka do programu. Po vložení se zobrazí v aplikaci název vloženého souboru. Poté se soubor zkompiluje (přeloží), program z něj načte příslušnou metodu a spustí ji. Po spuštění uloží výsledek a ten následně ověří pomocí testovací knihovny JUnit^[11] se správným výsledkem. V případě, že uživatel úkol splnil úspěšně, se zobrazí v aplikaci zelená „fajfka“. V opačném případě se zobrazí červený křížek. Tímto způsobem je uživatel schopen ověřit své znalosti.

Znalosti o všech algoritmech jsem čerpal zejména z výkladů pana Doc. RNDr. Pavla Töpfera v rámci hodin algoritmů.

3.3. Vývoj

V této podkapitole bych rád představil hlavní myšlenku, která mě k vývoji inspirovala. V dnešní době totiž neexistuje mnoho výukových materiálů formou aplikace než-li tento. Možná ani podobná učebnice vůbec neexistuje. Je jisté, že na internetu lze najít mnoho návodů ke konkrétním algoritmům, ale ty je možné najít pouze, pokud víme, co hledáme. Když se podíváme na tištěné učebnice, těch také neexistuje mnoho. Přesto jsou o mnoho přínosnější než internetové návody. Jako hlavní výhodu programu vnímám testy k jednotlivým kapitolám. Troufám si říci, že podobný veřejný výukový nástroj, kde se vyskytuje kombinace teorie a praxe, neexistuje.

4. Závěr

Cíl své práce jsem splnil, přesto je stále prostor pro zlepšení. V první řadě bych rozšířil obsah učebnice například o kapitolu o práci s dlouhými čísly, binárními stromy, základními datovými strukturami, jako je halda, zásobník nebo fronta, a nebo o hešování. Možná by také stálo za to rozšířit počet testů. Při práci na podobných projektech bych vybral lepší vývojové nástroje. Konkrétně mám na mysli malou responzivnost grafické knihovny JavaFX. Ač je nejlepší grafickou knihovnou v Javě, v dnešní době komplexních designových prvků (animace, změna velikosti dynamicky, ...), je zkrátka nedostačující. Existuje spousta různých jiných nástrojů, ve kterých je vývoj těchto prvků o poznání jednodušší a rychlejší. Stálo by také zvážit, zda by nebylo lepší aplikaci udělat webovou. Pak by problémy s designem zmizely a aplikace by byla pro čtenáře dostupnější. Mohli by ji navštívit i uživatelé na mobilních telefonech a byla by plně přístupná na internetu. Otevřít odkaz v prohlížeči je totiž dnes o dost jednodušší a rychlejší, než-li instalovat aplikaci do počítače. Zároveň by byl zachován programovací jazyk, ve kterém jsem projekt tvořil, jímž byla Java. Dle mého názoru se na tyto a podobné úkoly hodí. Už jenom z toho důvodu, že dokáže pracovat jako backend webových aplikací. Na závěr bych chtěl poznamenat, že místo HTML jazyka pro vytváření obsahu bych v případě jednodušší implementace bez váhání použil například soubory s příponou *.md*, tedy „Markdown“ soubory. Jsou totiž přehlednější a do programátorského světa se více hodí, zejména díky jejich zobrazení částí kódu.

5. Bibliografie

- [1] Java (programming language) - Wikipedia. [online] [cit. 25.04.2022].
Dostupné z: [https://en.wikipedia.org/wiki/Java_\(programming_language\)](https://en.wikipedia.org/wiki/Java_(programming_language))
- [2] Java Tutorial: Learn Java Basics For Free | Codecademy. *Learn to Code - for Free | Codecademy* [online]. Copyright © [cit. 25.04.2022].
Dostupné z: <https://www.codecademy.com/learn/learn-java>
- [3] JavaFX. *JavaFX* [online] [cit. 25.04.2022].
Dostupné z: <https://openjfx.io/>
- [4] JavaFX FXML. Jenkov.com Tech & Media Labs - Resources for Developers, IT Architects and Technopreneurs [online] [cit. 25.04.2022].
Dostupné z: <https://jenkov.com/tutorials/javafx/fxml.html>
- [5] Scene Builder - Gluon. *Rapid Enterprise Mobile Apps: Build, Connect, Manage with Gluon* [online]. Copyright © 2022 [cit. 25.04.2022].
Dostupné z: <https://gluonhq.com/products/scene-builder/>
- [6] GitHub: Where the world builds software · GitHub. *GitHub: Where the world builds software · GitHub* [online]. Copyright © 2022 GitHub, Inc. [cit. 25.04.2022].
Dostupné z: <https://github.com/>
- [7] Maven - Welcome to Apache Maven. *Maven - Welcome to Apache Maven* [online] [cit. 25.04.2022].
Dostupné z: <https://maven.apache.org/>
- [8] CSS Tutorial: Learn CSS For Free | Codecademy. *Learn to Code - for Free | Codecademy* [online]. Copyright © [cit. 25.04.2022].
Dostupné z: <https://www.codecademy.com/learn/learn-css>
- [9] HTML Tutorial: Learn HTML For Free | Codecademy. *Learn to Code - for Free | Codecademy* [online]. Copyright © [cit. 25.04.2022].
Dostupné z: <https://www.codecademy.com/learn/learn-html>
- [10] Carbon | Create and share beautiful images of your source code. *Carbon | Create and share beautiful images of your source code* [online] [cit. 25.04.2022].
Dostupné z: <https://carbon.now.sh/i0Y3hZmWl177kxlanpOo>
- [11] JUnit 5. [online]. Copyright © 2022 The JUnit Team [cit. 25.04.2022].
Dostupné z: <https://junit.org/junit5/>
- [12] Merge Sort (With Code in Python/C++/Java/C). Programiz: Learn to Code for Free [online]. Copyright © Parewa Labs Pvt. Ltd. All rights reserved. [cit. 25.04.2022].
Dostupné z: <https://www.programiz.com/dsa/merge-sort>
- [13] Download dependencies for java class javax.tools. Download JAR files with all dependencies [online]. Copyright © 2015 [cit. 25.04.2022].
Dostupné z: https://jar-download.com/maven-repository-class-search.php?search_box=javax.tools