STŘEDNÍ PRŮMYSLOVÁ ŠKOLA  
MLADÁ BOLESLAV

**ROČNÍKOVÁ PRÁCE**

Vojtěch Žmolík

Mladá Boleslav 2022

STŘEDNÍ PRŮMYSLOVÁ ŠKOLA  
MLADÁ BOLESLAV

**ROČNÍKOVÁ PRÁCE**

**Autor: Jan Novák  
Studijní obor: 18-20-M/01 Informační technologie  
Vedoucí práce: Jan Till**

Mladá Boleslav 2022

# Obsah

[Obsah 3](#_Toc102307376)

[1 Úvod 6](#_Toc102307377)

[2 Použité Technologie 7](#_Toc102307378)

[2.1 JavaFX 7](#_Toc102307379)

[2.2 IntelliJ IDEA 7](#_Toc102307380)

[3 Způsoby řešení a použité postupy 9](#_Toc102307381)

[3.1 Grafický vzhled 9](#_Toc102307382)

[3.1.1 Scéna výběrů formuláře 9](#_Toc102307383)

[3.1.2 Scéna vyplnění formuláře 10](#_Toc102307384)

[3.2 Vnitřní fungování aplikace 12](#_Toc102307385)

[3.2.1 Zapnutí aplikace 12](#_Toc102307386)

[3.2.2 Konfigurační soubor pro formuláře 13](#_Toc102307387)

[3.2.3 Operace konfiguračního souboru formulářů 13](#_Toc102307388)

[3.2.4 Načtení vstupů od uživatele 14](#_Toc102307389)

[3.2.5 Vypisování dat do souboru 15](#_Toc102307390)

[3.2.6 Přepínání scén 15](#_Toc102307391)

[4 Závěr 17](#_Toc102307392)

[5 Přílohy 18](#_Toc102307393)

[5.1 Seznam obrázků 18](#_Toc102307394)

[5.2 Zdroje 18](#_Toc102307395)

**Prohlášení**

Prohlašuji, že jsem svou ročníkovou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v přiloženém seznamu.

Nemám závažný důvod proti zpřístupňování této ročníkové práce v souladu se zákonem č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) v platném znění.

V Mladé Boleslavi dne podpis:

# Úvod

Smyslem této práce je prověřit autorovu znalost v oblasti programování oblasti IT. Zadání práce pro autora bylo vytvořit funkční aplikaci pro vyplňování formulářů v programovacím jazyce JavaFX. Práce byla vybrána autorem ze seznamu témat pro ročníkové práce zcela dobrovolně pro autorův zájem v tématice programování grafických aplikací a zdánlivé jednoduchosti zadání.

# Použité Technologie

## JavaFX

Jedná se o programovací jazyk vytvořen na bázi fungování a syntaxe programovacího jazyku Java a nástupce skriptovacího jazyka JavaFX script, pro který byla ukončena podpora na verzi JavaFX 2.0. Je zároveň rozšířen +o nové funkce vytvořen společností Sun Microsystems roku 2008. Byl vytvořen s podporou pro všechny formy multimediálních prvků a jakéhokoliv prostředí, kde běží JRE (Java Runtime Environment) a od verze 2.2 lze díky Native Packaging spustit aplikaci i na prostředí bez JRE[[1]](#footnote-1)



Obrázek Logo JavaFX [https://en.wikipedia.org/wiki/File:JavaFX\_Logo.png]

## IntelliJ IDEA

Jedná se o jedno z vývojových prostředí vytvořeno firmou JetBrains pro tvorbu kódu v jazycích Java, Groovy, atd. zároveň obsahující všechny nástroje, které uživatel pro tvorbu aplikací potřebuje s možností použít rozšíření vytvořené ostatními uživateli pro vylepšení vývojového prostředí. Pro vývoj grafického prostředí je IntelliJ vybaven i grafickým editorem Scene Builder Jediné doplňky, které používal autor, byly pro změnu vzhledu vývojového prostředí, pro lepší orientaci ve struktuře projektu.*[[2]](#footnote-2)*



Obrázek Logo IntelliJ IDEA [https://upload.wikimedia.org/wikipedia/commons/thumb/9/9c/IntelliJ\_IDEA\_Icon.svg/80px-IntelliJ\_IDEA\_Icon.svg.png]

# Způsoby řešení a použité postupy

Pro přehlednost bylo třeba vytvořit dvě různé scény, jednu pro výběr formuláře a druhou pro jeho následné vyplnění. Každá scéna se skládala z několika komponentů a okolo dvě stě řádků kódu.

## Grafický vzhled

Scény jsou základním stavebním blokem aplikace. Jejich vzhled byl kvůli omezenému množství komponentů dostupných pro JavaFX byl zvolen velmi minimalisticky a velmi utilitárně bez jakýchkoliv efektů či lépe vypadajících komponentů. Scene Builder zároveň měl vlastní problémy, kdy například nebylo možné provádět interakce kliknutím na komponent, ale bylo ho třeba najít v menu všech komponentů.

### Scéna výběrů formuláře

Hlavním komponentem scény je rozbalovací nabídka, použitá za účelem vybrání formuláře pro vyplnění uživatelem. Mezi další komponenty jsou zařazena dvě tlačítka, první inicializuje vyplňování zvoleného formuláře za pomocí funkce ve svém kontroléru a druhé aplikaci vypne. Zároveň lze bez jakéhokoliv negativního dopadu na aplikaci ji vypnout i jinak, než přes toto tlačítko

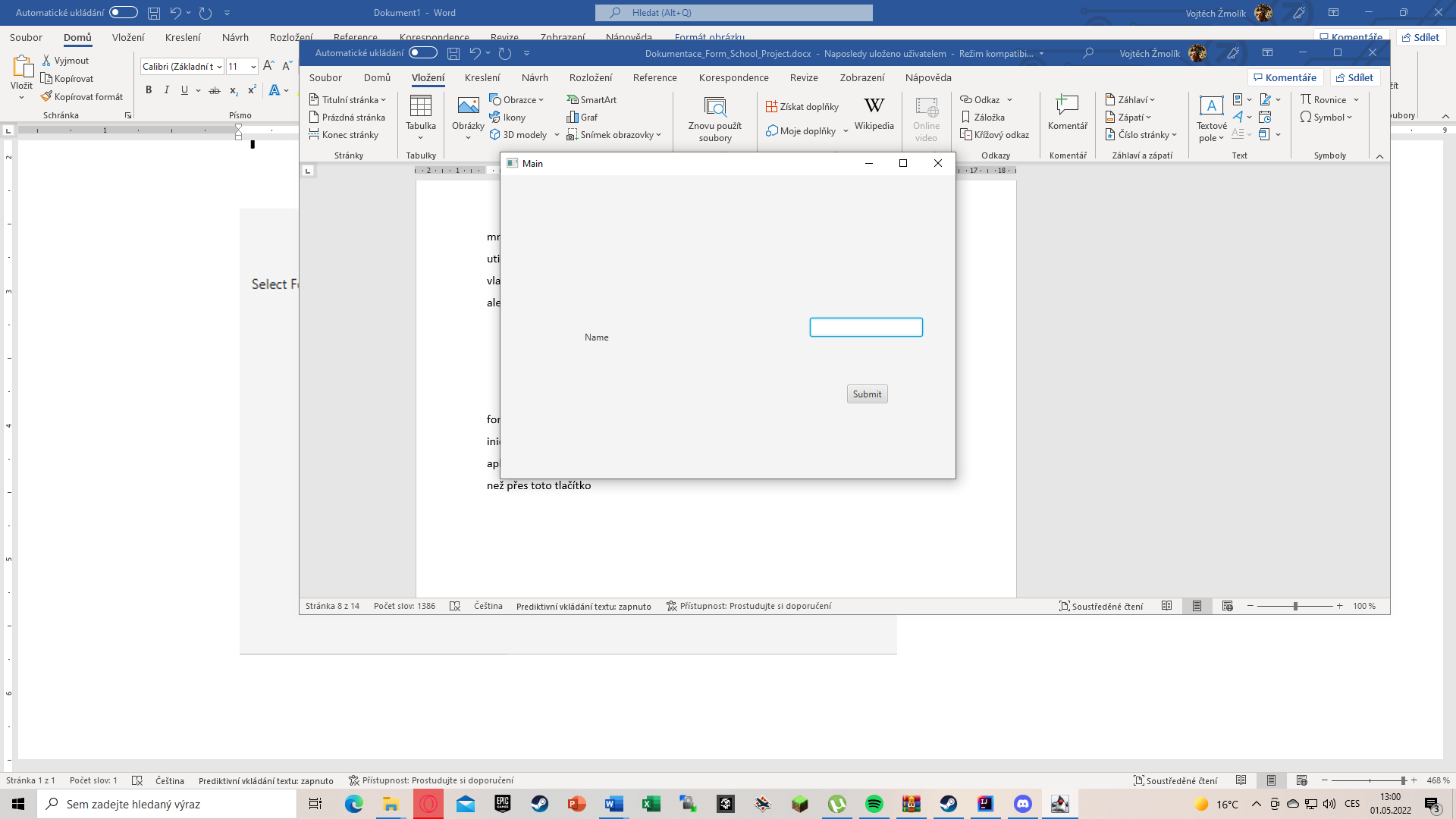
Obsah obrázku text, snímek obrazovky, monitor, interiér

Popis byl vytvořen automaticky

Obrázek Scéna vybrání formuláře

### Scéna vyplnění formuláře

Scéna pro samotné vyplňování formulářů se skládá pouze ze tří komponentů, které jsou ovšem pro aplikace velice důležité, jelikož kdyby byly odstraněny či nefungovaly, jak mají, tak by aplikace nemohla správně fungovat. Nejdůležitějším komponentem je vyplňovací pole, do které uživatel zapíše své odpovědi na dotaz zobrazený před polem ve formě neformátovaného textu. Pro potvrzení odpovědi je zde přítomno tlačítko, kterým se uživatel posune na další dotaz, či se vrátí do výběru formuláře, pokud je na všechny dotazy odpověděno. Zároveň tato část podléhala největšímu a jedinému problému, jelikož při cyklu vyplňování bylo třeba čekat na odpověď uživatele. Toto bylo nakonec díky konzultaci vyřešeno rozpůlením cyklu na dvě metody, kde se jedna přepíná dotaz a druhá posílá data do FileWriteru. Viz zdrojový kód 1.



Obrázek Scéna vyplnění dotazu formuláře

public **void** doForm(ActionEvent event) throws IOException, InterruptedException {

switchSceneFill();

formWhole = logic.getForm(formBox.getValue());

boolean run = true;

**while** (run) {

File file = new File(System.getProperty("user.home") + "/Desktop/" + formBox.getValue() + " (" + fileNum + ")" + ".txt");

**if** (!file.createNewFile()) {

fileNum++;

} **else** {

run = false;

}

}

currentRound = 1;

setFormData();

}

private **void** setFormData() {

**if** (currentRound < formWhole.length) {

question.setText(formWhole[currentRound]);

fillField.setText("");

} **else** {

currentRound = 0;

switchSceneMain();

}

}

public **void** submitPressed() throws IOException {

fileWriter = new FileWriter(System.getProperty("user.home") + "/Desktop/" + formBox.getValue() + " (" + fileNum + ")" + ".txt", true);

fileWriter.write(formWhole[currentRound] + ": " + fillField.getText() + "\n");

fileWriter.flush();

fileWriter.close();

currentRound++;

setFormData();

}

Zdrojový kód č.1

## Vnitřní fungování aplikace

Pro správné fungování aplikace je třeba k ní napsat kód, který se postará o veškeré funkce a procesy. Kód napsaný v aplikaci je vymyšlený autorem samotným s výjimkou kódu pro samotné vyplňování nastaveného formuláře, jehož správné fungování bylo docíleno konzultací. Toto znamená že žádný úryvek kód nelze najít na internetu, a pokud se kód na internetu nachází, tak se jedná o náhodu.

### Zapnutí aplikace

Díky vývojovému prostředí IntelliJ, které vygenerovalo celou třídu pro zobrazení prvotní scény, nebylo pro autora třeba vytvářet kód a vývoj byl zkrácen o nejvíce půl hodiny. Jaká scéna se zobrazí jako první je pevně dáno a program si pouze načte kontrolér scény a scénu podle instrukcí v metodě initialize zobrazí. Viz zdrojový kód 3 a 4.

@Override

public void start(Stage stage) throws IOException {

FXMLLoader fxmlLoader = new FXMLLoader(HelloApplication.class.getResource("Main.fxml"));

Scene scene = new Scene(fxmlLoader.load());

stage.setTitle("Main");

stage.setScene(scene);

stage.show();

}

public static void main(String[] args) {

launch();

}

Zdrojový kód 2

@Override

public **void** initialize(URL url, ResourceBundle resourceBundle) {

logic = new Logic();

logic.loadForms();

**for** (String s : logic.getForms()) {

formBox.getItems().add(s);

}

fillField.setVisible(false);

question.setVisible(false);

submit.setVisible(false);

}

Zdrojový kód 3

### Konfigurační soubor pro formuláře

Konfigurační soubor pro formuláře v sobě ukládá všechna data o každém z formulářů. Nachází se zde jejich jména i jednotlivé komponenty. Pro jednoduchost bylo zvoleno vytvořit textový soubor, který bude jako konfigurační soubor sloužit. Viz zdrojový kód 4.

Form Name;Question 1;Question 2;Question 3;...

Zdrojový kód 4

### Operace konfiguračního souboru formulářů

Konfigurační soubor je velmi důležitou částí aplikace, jelikož v sobě uchovává jména a jednotlivé dotazy všech formulářů v programu. Pro práci s formuláři byla vytvořena třída Logic, která má v sobě všechny potřebné metody. Viz zdrojový kód 5.

Nejdůležitější metodou je načtení formulářů z konfiguračního souboru, to je provedeno metodou používající třídu Scanner, která načte řádek z konfiguračního souboru, jenž je následně rozdělen na jednotlivé segmenty, které vloženy do pole, jež jsou následně přidány do kolekce ArrayList, která obsahuje všechny formuláře v použitelném formátu, toto je prováděno pro každý řádek v konfiguračním souboru.

Následné operace může již uživatel vidět, jelikož se jedná o operaci extrakci jmen ze všech formulářů, které následně využívá výběrová nabídka, prováděno cyklem, který zjistí jména všech formulářů a vrátí je v kolekci ArrayList metodě, která zavolala tuto metodu. Další operací, kterou uživatel může vidět je vybrání specifického formuláře, konkrétně formuláře vybraného ve výběrové nabídce. Ten je vybrán z kolekce uchovávající všechny formuláře podle jména a vrátí formulář jehož jméno se shoduje se jménem formuláře ve výběrové nabídce volající metodě.

private ArrayList<String[]> forms = new ArrayList<>();

public **void** loadForms (){

try {

File file = new File("src/main/java/com/example/form\_school\_project/config/config.txt");

Scanner sc = new Scanner(file);

**while** (sc.hasNextLine()){

String line = sc.nextLine();

String[] form = line.trim().split(";");

forms.add(form);

}

} catch (FileNotFoundException e) {

e.printStackTrace();

}

}

public ArrayList<String> getForms(){

ArrayList<String> formNames = new ArrayList<>();

**for** (String[] form : forms){

formNames.add(form[0]);

}

**return** formNames;

}

public String[] getForm(String value){

**for** (String[] form:forms) {

**if** (form[0].equals(value)){

**return** form;

}

}

**return** null;

}

Zdrojový kód 5

### Načtení vstupů od uživatele

V aplikaci jsou celkem dvě situace, které vyžadují vstup uživatele. První z nich je vybrání formuláře pro vyplnění, což je uděláno jednoduchou rozbalovací nabídkou obsahující jména všech formulářů. Po vybrání formuláře je uživateli zpřístupněno tlačítko pro započetí vyplňování vybraného formuláře. Druhou situací vyžadující vstup uživatele je samotné vyplnění formuláře. Pro vyplnění formuláře uživatel zadá informaci po něm žádanou v dotazu do vyplňovacího pole, pokud uživatel nechce informaci zadávat stačí pole nechat prázdné. Následně je třeba stisknout tlačítko Submit, které pošle odpověď dál, vymaže vyplňovací pole a zadá nový dotaz, pokud jsou všechny dotazy vyplněny, je vypnut FileWriter a přepnuto zpět na výběr formuláře. Viz Zdrojový kód 6.

### Vypisování dat do souboru

Aplikace vypisuje data získána od uživatele na plochu, pojmenovaných podle vybraného formuláře a počtu již vytvořených formulářů, do textového souboru s každým dotazem a odpovědí na dotyčný dotaz na jednom řádku. Tohoto je dosaženo pomocí třídy FileWriter, která dostane URL adresu, textového souboru vytvořeného pro vyplňovaný formulář, inicializovanou při započetí vyplňovacího procesu. Jakmile uživatel odpoví na dotaz, odpověď je následně spojena s dotazem a zapsána do souboru pokaždé, jakmile je stisknuto tlačítku submit, toto má za následek procedurální ukládání dat do souboru a při ukončení aplikace při vyplňování nedojde ke ztrátě dat, nevýhodou ovšem je že nelze do vyplnit formulář pokud je přerušeno jeho vyplňování. Viz zdrojový kód 6.

public **void** submitPressed() throws IOException {

fileWriter = new FileWriter(System.getProperty("user.home") + "/Desktop/" + formBox.getValue() + " (" + fileNum + ")" + ".txt", true);

fileWriter.write(formWhole[currentRound] + ": " + fillField.getText() + "\n");

fileWriter.flush();

fileWriter.close();

currentRound++;

setFormData();

}

Zdrojový kód 6

### Přepínání scén

V aplikaci ve skutečnosti neexistují dvě scény, přepínání vzhledu bylo původně navrženo jako právě pro dvě rozdílné scény, které by měly své vlastní komponenty, mezi kterýma bylo přepínáno. Ovšem se vyskytly problémy s posíláním dat mezi oběma scénami, konkrétně se jednalo o vracení dat do původní scény, byly všechny komponenty přesunuty na jednu scénu. Jelikož však nešlo mít všechny komponenty na jedné scéně, jelikož by scéna působila stísněně a nepřehledně. Viz Zdrojový kód 7.

Obsah obrázku text, monitor, snímek obrazovky, elektronika

Popis byl vytvořen automaticky

Obrázek Vzhled scény bez úprav atributů

private **void** switchSceneFill(){

formBox.setVisible(false);

fill.setVisible(false);

staticLabel.setVisible(false);

end.setVisible(false);

fillField.setVisible(true);

question.setVisible(true);

submit.setVisible(true);

}

private **void** switchSceneMain(){

formBox.setVisible(true);

fill.setVisible(true);

staticLabel.setVisible(true);

end.setVisible(true);

fillField.setVisible(false);

question.setVisible(false);

submit.setVisible(false);

}

Zdrojový kód 7

# Závěr

Cílem autora bylo vytvořit formulář v programovacím jazyce JavaFX. Práce byla zvolena kvůli své jednoduchosti a autorovým zkušenostem s tématem. Samotný projekt byl na kódován během jednoho týdne počítaje v to komplikace ve formě cyklu pro vyplňování formuláře a vytvoření kompatibilního grafického rozhraní, což zabralo okolo jedné hodiny a ostatních komponentů vytvořených během dvou dnů.

# Přílohy

## Seznam obrázků

[Obrázek 1 Logo JavaFX [https://en.wikipedia.org/wiki/File:JavaFX\_Logo.png] 7](#_Toc102303107)

[Obrázek 2 Logo IntelliJ IDEA [https://upload.wikimedia.org/wikipedia/commons/thumb/9/9c/IntelliJ\_IDEA\_Icon.svg/80px-IntelliJ\_IDEA\_Icon.svg.png] 8](#_Toc102303108)

[Obrázek 3 Scéna vybrání formuláře 10](#_Toc102303109)

[Obrázek 4 Scéna vyplnění dotazu formuláře 10](#_Toc102303110)

[Obrázek 5 Vzhled scény bez úprav atributů 13](#_Toc102303111)

## Zdroje

JavaFX. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-, 15. 4. 2022 [cit. 2022-04-26]. Dostupné z: <https://cs.wikipedia.org/wiki/JavaFX>

Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2022-04]. Dostupné z: https://cs.wikipedia.org/wiki/IntelliJ\_IDEA

1. JavaFX. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-, 15. 4. 2022 [cit. 2022-04-26]. Dostupné z: https://cs.wikipedia.org/wiki/JavaFX [↑](#footnote-ref-1)
2. Citováno z <https://cs.wikipedia.org/wiki/IntelliJ_IDEA> , dne 25.04.2022 20:03 [↑](#footnote-ref-2)