Návrhové vzory

Semestrálna Práca

Aplikácia na šifrovanie a dešifrovanie vstupu

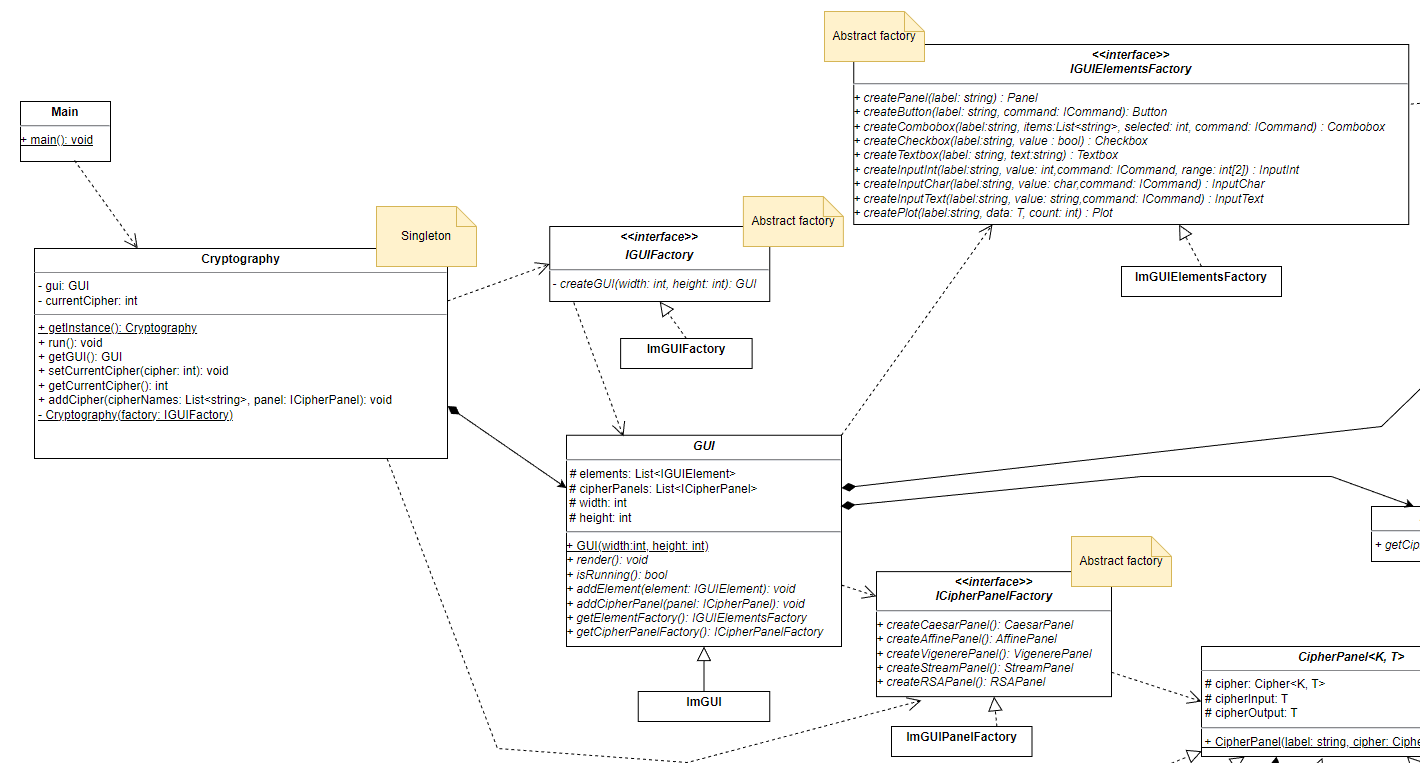
**Juraj Šibík 5ZIS21**

# Popis Problému

Prenos správ a ich uchovávanie je činnosť, s ktorou sa stretávame skoro každý deň, či už pri odoslaní správy cez sociálnu sieť alebo poštou. Môže sa však stať, že správa skončí v nesprávnych rukách a daná osoba zistí informáciu, ktorá môže byť citlivá. Taktiež môže danú správu modifikovať a poslať ju pôvodnému príjemcovi, čo je nežiadúce. Utajeniu informácie v správe sa venuje oblasť kryptografie a táto aplikácia ponúka niekoľko základných šifier, ktoré dokážu správu zašifrovať a rozšifrovať s možnosťou rozšírenia o ďalšie šifry v prípade záujmu používateľa. Taktiež je pri niektorých z ponúkaných šifier možnosť útoku na šifru s cieľom uhádnuť kľúč, ktorým boli správy zašifrované.

# Popis univerzálneho návrhu aplikácie

## Rozšíriteľnosť GUI



Grafickú časť je možné jednoducho rozšíriť o nové grafické rozhranie vďaka **abstraktným továrňam**, ktoré dovoľujú vytvoriť nové GUI spoločne s grafickými elementami špecifickými pre novovytvorené gui. Taktiež je možné tieto elementy doplniť o nové typy, avšak bude potrebné pre každý nový typ doplniť abstraktnú metódu do abstraktnej továrne IGUIElementsFactory a potom ju v potomkovi prekryť.

Obrázok, na ktorom je text, mapa, vnútri

Automaticky generovaný popis

Grafické elementy je možné rozšíriť, pokiaľ implementujú interface IGUIElement. Element Panel je implementovaný na štýl **kompozitu**, aby bolo možné v ňom uchovávať viacero IGUIElementov, vrátane ďalších Panelov. Niektoré elementy môžu obsahovať **commandy**, ktoré slúžia na spustenie nejakej funkcie ako napríklad rozšifrovanie textu pomocou konkrétnej šifry.

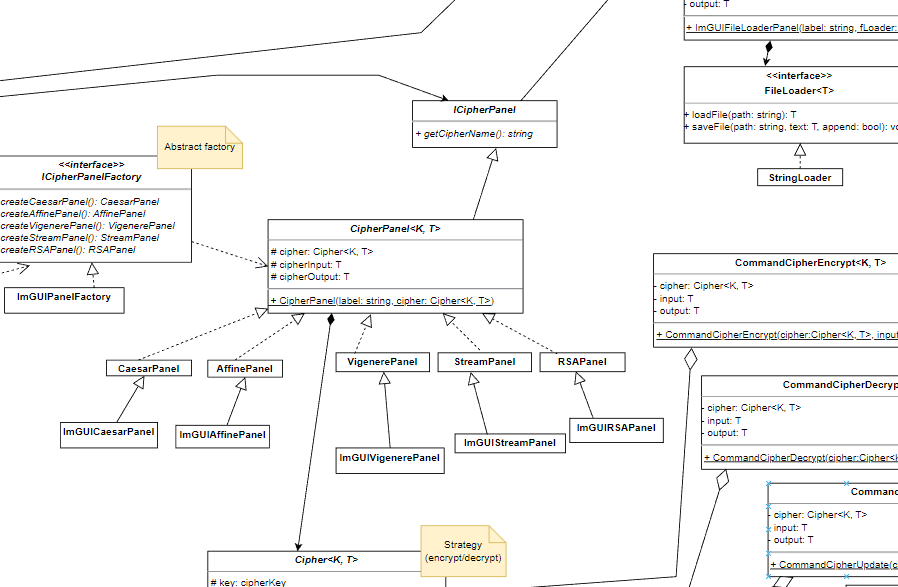
## Komunikácia medzi GUI a aplikáciou

Obrázok, na ktorom je text, vnútri, mapa, počítač

Automaticky generovaný popis

Komunikácia medzi aplikáciou a grafickým rozhraním je riešená cez **commandy.** Každý command implementuje rozhranie ICommand, takže v prípade rozšírenia o ďalšie commandy stačí implementovať toto rozhranie a prekryť metódu execute. Nachádzajú sa tu aj dva commandy špecifické pre grafickú časť ImGUI, konkrétne na otváranie panelu ImGUIFileLoaderPanel<T>, kde T je typ vstupu, ktorý aktuálne zvolená šifra vie spracovať. Nachádza sa tu aj **null object** command z toho dôvodu, že pri niektorých grafických elementoch ako napríklad InputInt nie je potrebné pri zmene hodnoty spustiť nejakú funkcionalitu, avšak niekedy sa zíde. Vďaka null commandu, ktorý je pri týchto elementoch špecifikovaný ako default parameter potom vôbec nie je potrebné špecifikovať nejaký command.

## Pridávanie nových šifier



Každá šifra má svoj špecifický panel, ktorým sa na GUI zobrazujú možnosti pre danú šifru. Tieto panely sú uložené v zozname v triede GUI ako panely ICipherPanel. Pri tvorbe novej šifry je potrebné rozšírenie **abstraktnej továrne** ICipherPanelFactory o názov novej šifry. Tú potom bude možné pridať v triede Cryptography cez metódu addCipher.

Obrázok, na ktorom je text, mapa, vnútri

Automaticky generovaný popis

Šifry je možné rozšíriť o novú šifru vytvorením potomka triedy Cipher<K, T>, kde K je kľúč novej šifry a T je typ vstupu ako napríklad string, s ktorým vie následne šifra pracovať. Pre šifrovanie a dešifrovanie sa využíva **stratégia**, vďaka ktorej dostanem po zavolaní metódy výsledok metódy zavolanej po spracovaní šifrou nad ktorou bola metóda zavolaná s rovnakým typom aký som do metódy poslal cez parameter. Metóda update slúži na dodatočnú úpravu vstupu na výstup modifikovaním parametrov kľúča. Vykonáva sa vždy nad poslednou zavolanou metódou šifrovania alebo dešifrovania. Metóda tryFindKey je metódou, v ktorej sa spustí útok na šifru s pokusom o uhádnutie kľúča, ktorým bol vstup zašifrovaný.

# Popis implementácie aplikácie

Aplikácia bola naprogramovaná v programovacom jazyku C++ so štandardom C++17 a v softvéri Microsoft Visual Studio 2022. Pre jednoduchšie nakonfigurovanie projektu spoločne s riešením importu knižníc tretích strán bol použitý premake. Pri zmene niektorého zo súborov premake5 je potrebné spustiť build.bat, nachádzajúce sa v koreni adresára projektu.

Na začiatku sa vytvára **singleton** triedy Cryptography, kde sa posiela **abstraktná továreň** ImGUIFactory implementujúca rozhranie IGUIFactory. Táto továreň vytvára triedu grafického rozhrania ImGUI, ktorá je potomkom abstraktnej triedy GUI. Táto trieda má metódy, z ktorých sa následne získavajú jednotlivé typy GUI elementov, ako aj jednotlivé typy panelov šifier. Implementované šifry spoločne s ich panelmi majú každá konkrétny typ vstupu a výstupu, ktorý vedia spracovať. Pri šifrách sa využíva **stratégia** na voľbu šifrovania či dešifrovania ako aj na zopár ďalších metód. Komunikácia medzi GUI elementami ako napríklad Button a následnej reakcie aplikácie na jeho stlačenie je riešená cez **commandy**. Do niektorých IGUIElementov je možné vložiť ICommand, ktorý sa vykoná, ak sa napríklad zmení hodnota elementu alebo sa v prípade Buttonu stlačí. Ak je však potrebné vykonať viacero ICommandov, existuje aj **kompozit** CommandContainer, ktorý v sebe obsahuje zoznam ICommandov a spustí ich v poradí, v akom do neho boli poukladané. Taktiež existuje **null object** command CommandNull, ktorý sa využíva najmä v konštruktoroch IGUIElementov, kde by potreba špecifikovania ICommandu vždy, bola na prítaž.

Implementácia aplikácie kladie dôraz na rozširovateľnosť do budúcna vďaka riešeniu orientovanému na rozhrania.

# Sekvenčný diagram na zobrazenie fungovania aplikácie pri žiadosti o dešifrovanie vstupu

Obrázok, na ktorom je text, snímka obrazovky, vnútri

Automaticky generovaný popis

# Programátorská príručka pre budúce rozšírenie

Pokiaľ by chcel používateľ rozšíriť aplikáciu o vlastné implementácie šifier, tak je potrebné:

1. Vytvoriť novú šifru, ktorá dedí po triede Cipher<K, T> a implementovať jednotlivé metódy, pričom K je kľúč šifry a T je generický typ, ktorý sa (de)šifruje. Šifry v aplikácií používajú typ string.
2. Vytvoriť nový panel, ktorý dedí po triede CipherPanel<K, T>, pričom generiká znamenajú to isté čo v predchádzajúcom kroku. Do konštruktora sa potom posiela samotná implementácia šifry.
3. Po vytvorení panela je potrebné pridať do ICipherPanelFactory tvorbu novej šifry.
4. Následne je na používateľovi, ako bude riešiť GUI a či už vo vytvorenom paneli prekryje metódu draw(), ktorá je vyžadovaná na vykreslenie panela. V aktuálnej implementácií sa používa ImGUI a pre každú šifru je ešte panel zdedený konkrétnou implementáciou, nachádzajúcou sa v ImGUICipherPanel.h.
5. Nakoniec je potrebné pridať šifru v triede Cryptography.cpp do sekcie, ktorá je zobrazená na obrázku nižšie.

Obrázok, na ktorom je text

Automaticky generovaný popis

Čo sa týka pridania vlastného GUI, tak je potrebné podediť po triede GUI, implementovať jednotlivé metódy, vytvoriť továreň na jednotlivé GUIElementy, ktoré budú implementovať IGUIElement a dediť budú buď už po konkrétnych elementoch ako Button, Panel,... alebo budú spravené nové elementy, ktoré budú vyhovovať používateľovi. Taktiež bude potrebné spraviť továreň na jednotlivé CipherPanely, ktoré budú dediť po CipherPanel<K, T>. Čo sa týka načítavania súborov, je možné si spraviť vlastný načítavač, ktorý bude implementovať FileLoader<T>.