

Beschreibung Programmierprojekt – Lubica Mühlberger (0051755)
1578 - Anwendungspraktikum aus JAVA,
Dr. Hahsler, WS 2005/2006

CryptoManiac

Inhaltsverzeichnis

| | |
|----------------------------------|----|
| Problemstellung..... | 3 |
| Analyse und Design | 4 |
| Beschreibung Use-Case..... | 4 |
| Allgemein | 4 |
| Beschreibung des Szenarios | 6 |
| Klassendiagramm | 7 |
| Sequenz Diagramm | 8 |
| Implementierung | 9 |
| Allgemein | 9 |
| CryptoManiac..... | 9 |
| GUI..... | 9 |
| CryptAlgos | 9 |
| Caesar | 9 |
| Vigenere | 9 |
| Playfair | 9 |
| Installation und Wartung..... | 10 |
| Installation..... | 10 |
| Wartung..... | 10 |
| Abschlussbemerkungen..... | 11 |

Problemstellung

In diesem Projekt soll dem Benutzer des Tools eine Verschlüsselungshilfe für einzugebende Texte zur Verfügung stellen.

Die Verschlüsselung soll mit Hilfe von sog. klassischen Verfahren umgesetzt werden, genauer gesagt, soll es dem Benutzer möglich sein, aus drei verschiedenen Kryptografie-Verfahren jeweils ein beliebiges auszuwählen zu können, mit welchem der Text anschließend verändert werden soll.

Die in diesem Programm zur Verfügung stehenden Kryptografie-Arten sollen folgende Verfahren umfassen:

1. Cäsar Cipher: Verschiebung des Alphabets um x Zeichen
2. Vigenere¹: *„diese einfache Verschlüsselungsmethode arbeitet dem Caesar-Verfahren ähnlich mit dem Unterschied, dass das aktuelle Klartextzeichen je nach dessen Position im Klartextstrang im Alphabet verschoben wird, wobei man gegebenenfalls wieder am Anfang beginnt. So einfach, wie dieses Verfahren ist, lässt sich auch der Geheimtext schnell entschlüsseln, indem man die Zeichen je nach ihrer Position in die andere Richtung im Alphabet verschiebt.“*
Klartext : I N T E R N E T
Positionen: 1 2 3 4 5 6 7 8 (abcdefgh)
Geheimtext: J P W I W T L B
3. Playfair²: an dieser Stelle möchte ich versuchen ein etwas schwierigeres Verfahren zu implementieren: *„zunächst wird der zu verschlüsselnde Text in Großbuchstaben umgewandelt. Umlaute werden aufgelöst, Leerzeichen und Satzzeichen werden weggelassen. Der Klartext wird in Paaren aufgeschrieben. J wird zu I umgewandelt, aufeinander folgende gleiche Buchstaben werden durch X getrennt. Sollte als letztes ein einzelner Buchstabe stehen wird diesem auch noch ein X nachgestellt. Aus einem Schlüsselwort oder -satz wird ein Alphabet mit 25 Buchstaben (ohne J) gewonnen. Dieses wird in 5er Reihen aufgeschrieben: Dabei wird das Schlüsselwort zeilenweise in eine 5x5 Matrix eingetragen, wobei bereits eingetragene Buchstaben übersprungen werden. Danach werden die zum kompletten Alphabet (ohne j) fehlenden Zeichen in alphabetischer Reihenfolge ergänzt. Es werden immer Paare zu Paaren chiffriert. Stehen beide Buchstaben in der gleichen Zeile bzw. Spalte, werden jeweils die rechten bzw. unteren Nachbarn genommen. Stehen die Buchstaben am Rand wird oben bzw. links fortgesetzt. Andernfalls ersetzt man den ersten Buchstaben durch den in derselben Zeile aber in der Spalte des zweiten liegenden. Der zweite Buchstabe wird durch den in derselben Zeile aber in der Spalte des ersten liegenden Buchstaben ersetzt. Das Klartextpaar bildet also die gegenüber liegenden Ecken eines Rechtecks, das Geheimtextpaar wird aus den jeweils benachbarten Ecken gebildet.“*

Zu diesem Zweck wird eine grafische Benutzeroberfläche implementiert, auf welcher kurze Texte geschrieben werden sollen, welche dann, wie bereits beschrieben, am Interface zur

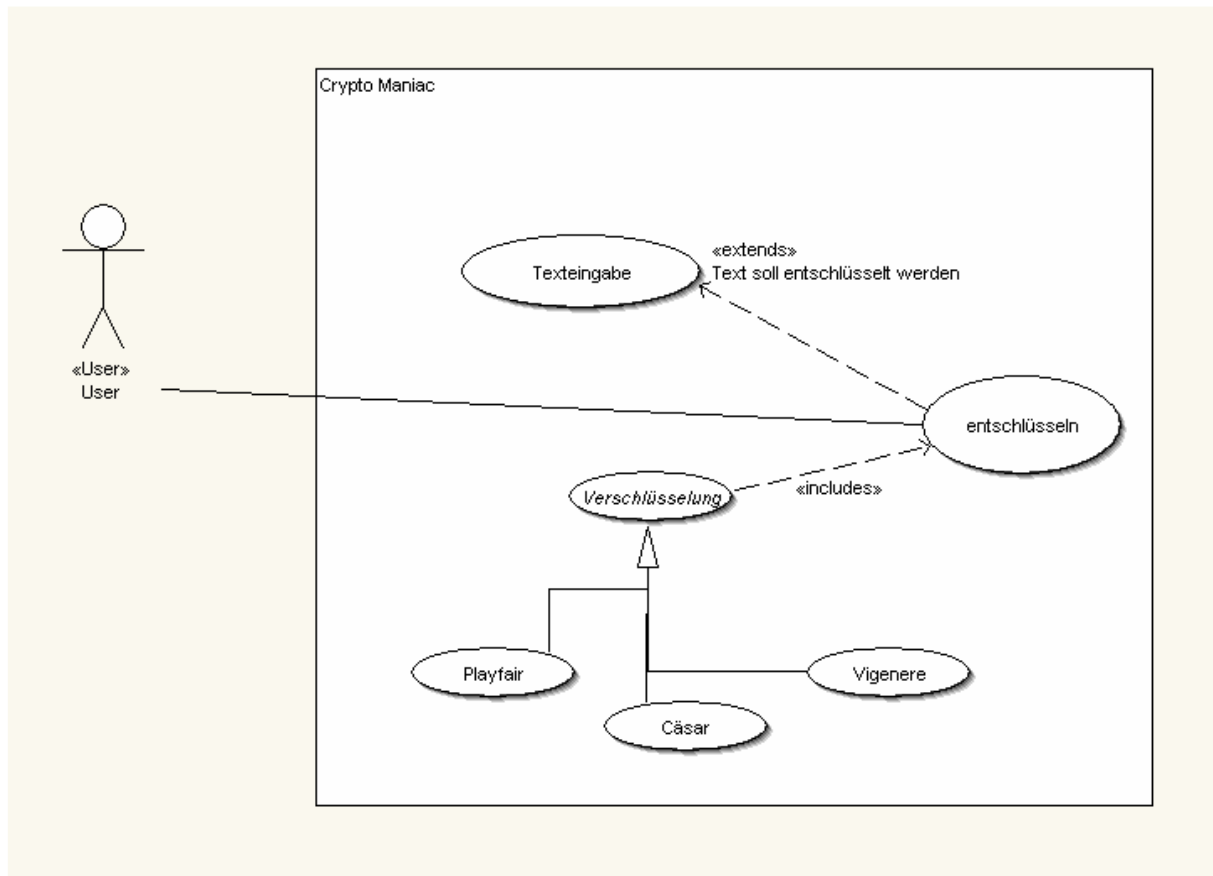
¹ Vgl. http://de.wikipedia.org/wiki/Polyalphabetische_Substitution

² Vgl. <http://de.wikipedia.org/wiki/Playfair>

Laufzeit verschlüsselt und entschlüsselt werden können. Dabei kann der gewünschte Algorithmus mittels Drop Down Liste ausgewählt werden.

Analyse und Design

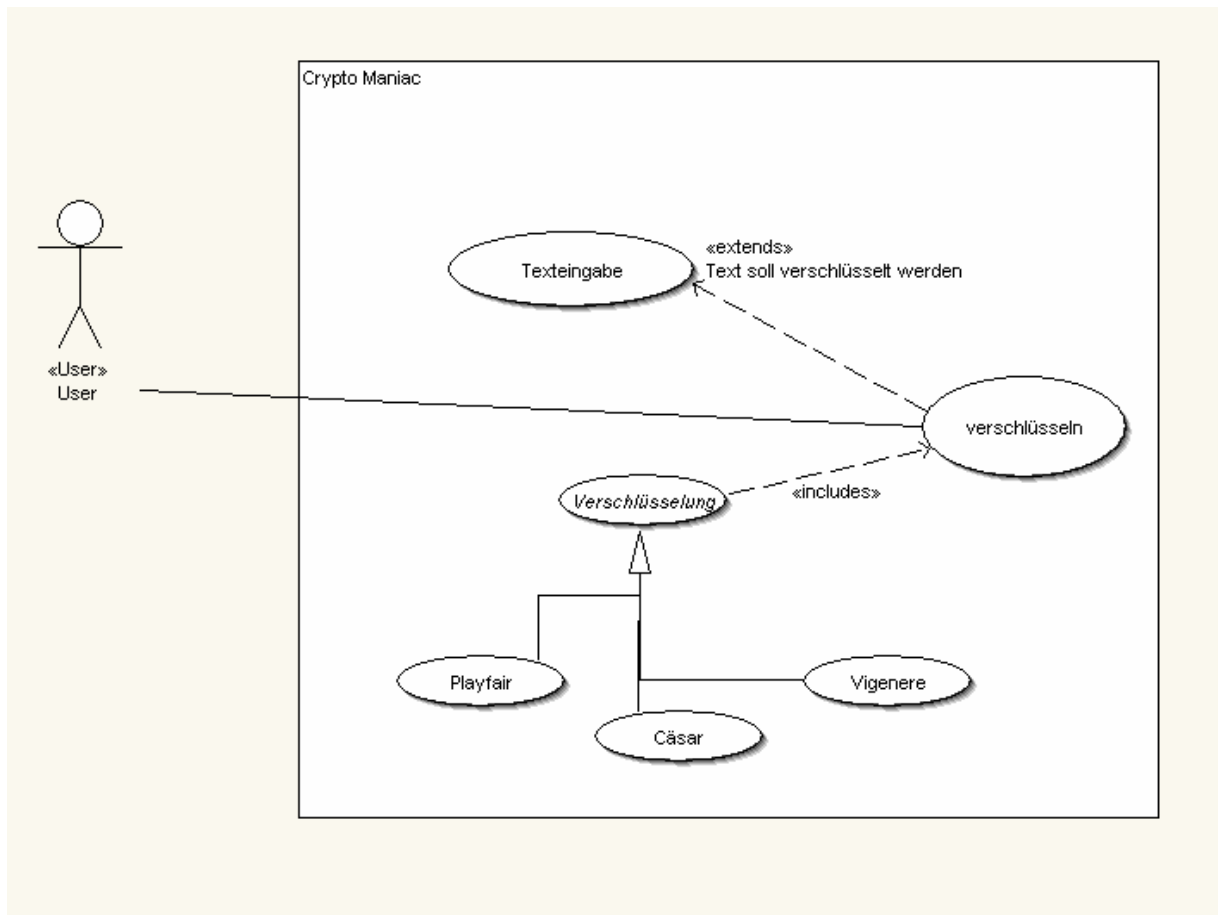
Beschreibung Use-Case



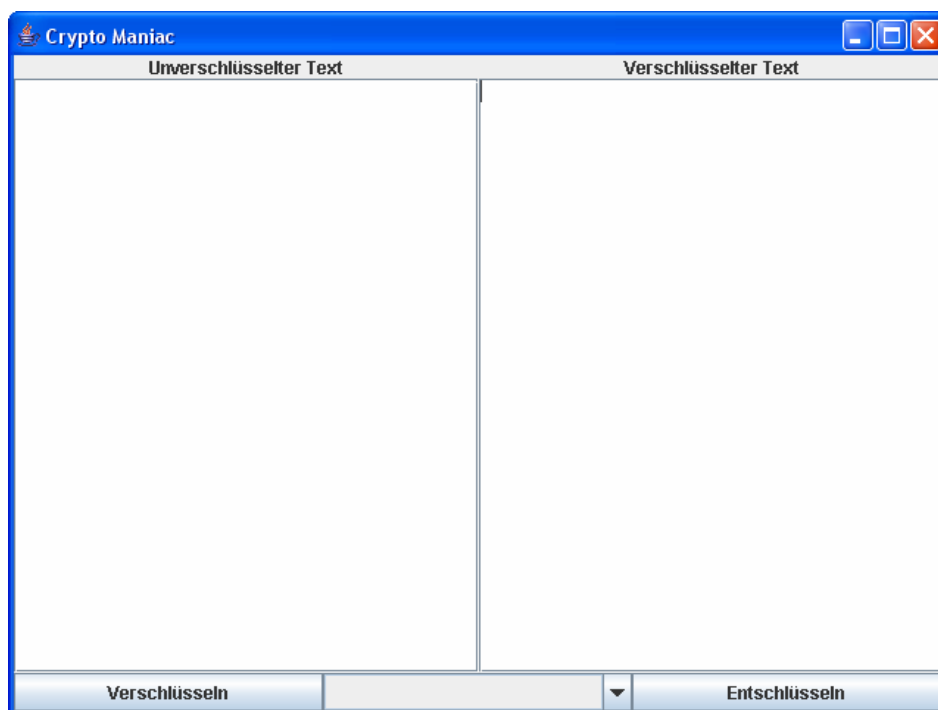
Use Case – „entschlüsseln“

Allgemein

- ✓ **Use-Case Name:** Crypto Maniac
- ✓ **Akteure:** User
- ✓ **Vorbedingungen:** Aufruf des Crypto-Tools und individuelle Texteingabe
- ✓ **Nachbedingungen:** Ver- und Entschlüsselung des eingegebenen Textes
- ✓ **Auslöser:** Befehl zur Ver- oder Entschlüsselung des Textes mittels Button



Use Case – „verschlüsseln“

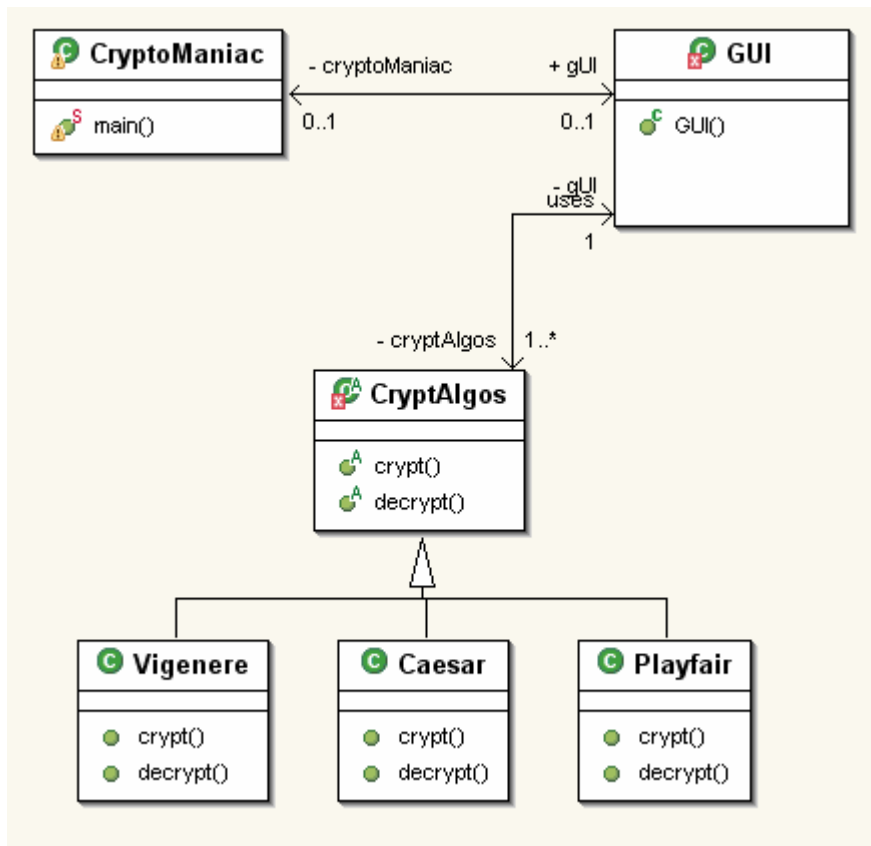


Screenshot des vorläufigen Prototyps vor Endabgabe

Beschreibung des Szenarios

- ✓ **Ablaufbeschreibung:** User tippt gewünschten Text ins linke Textfeld; er kann seinen Text nun durch Auswahl des gewünschten Verfahrens im Drop-Down Menü verschlüsseln lassen, durch Klicken auf den Button "Verschlüsseln"; der nun generierte verschlüsselte Text (=Cyphertext) wird im rechtem Feld sichtbar; ein Cyphertext kann umgekehrt durch Klicken auf den Button "Entschlüsseln" (unter erneuter Methodenwahl aus dem Drop-Down Menü) wieder in natürlichen Text umgewandelt werden (linkes Textfeld).
- ✓ **Fehlsituationen:** User gibt Cyphertext ein, welcher nach ganz anderen Verfahren verschlüsselt wurde und deshalb mit vorliegendem Tool nicht in ursprünglichen Text umgewandelt werden kann; User entschlüsselt Text in diesem Crypto-Tool, jedoch mit einem anderen Verfahren, als er zur Verschlüsselung verwendet hat
- ✓ **Variationen:** User kann zwischen drei verschiedenen Ver- und Entschlüsselungsverfahren im Drop-Down Menü wählen, Mehrfachverschlüsselung durch Wahl verschiedener Methoden daher möglich
- ✓ **Instanzen/Beispiele: Variante a)** User A möchte einen Text verschlüsseln und tippt (kopiert) den Text in die linke Spalte, wählt sodann aus dem Menü ein gewünschtes Verfahren zur Verschlüsselung aus und klickt anschließend auf den Button "Verschlüsseln". User B erhält nun den generierten Cyphertext (z.B. per E-Mail) und entschlüsselt denselben auf gleiche Weise wie User A. Anmerkung: Absprache wg. Wahl des gleichen Verfahrens zw. A und B notwendig. **Variante b)** User M verschlüsselt seinen Text durch zwei verschiedene Verfahren, sendet diesen an User P. User P entschlüsselt den Text durch Anwendung der Verfahren in umgekehrter Reihenfolge.
- ✓ **Ergebnisse:** Text Ver- oder Entschlüsselung
- ✓ **Autor:** Lubica Mühlberger

Klassendiagramm



Kurzbeschreibung der hier angeführten Klassen:

CryptoManiac: User startet Programm

GUI: Darstellung des Interfaces (Fenster)

CrypAlgos: Abstrakte Klasse eines Verschlüsselungsverfahrens

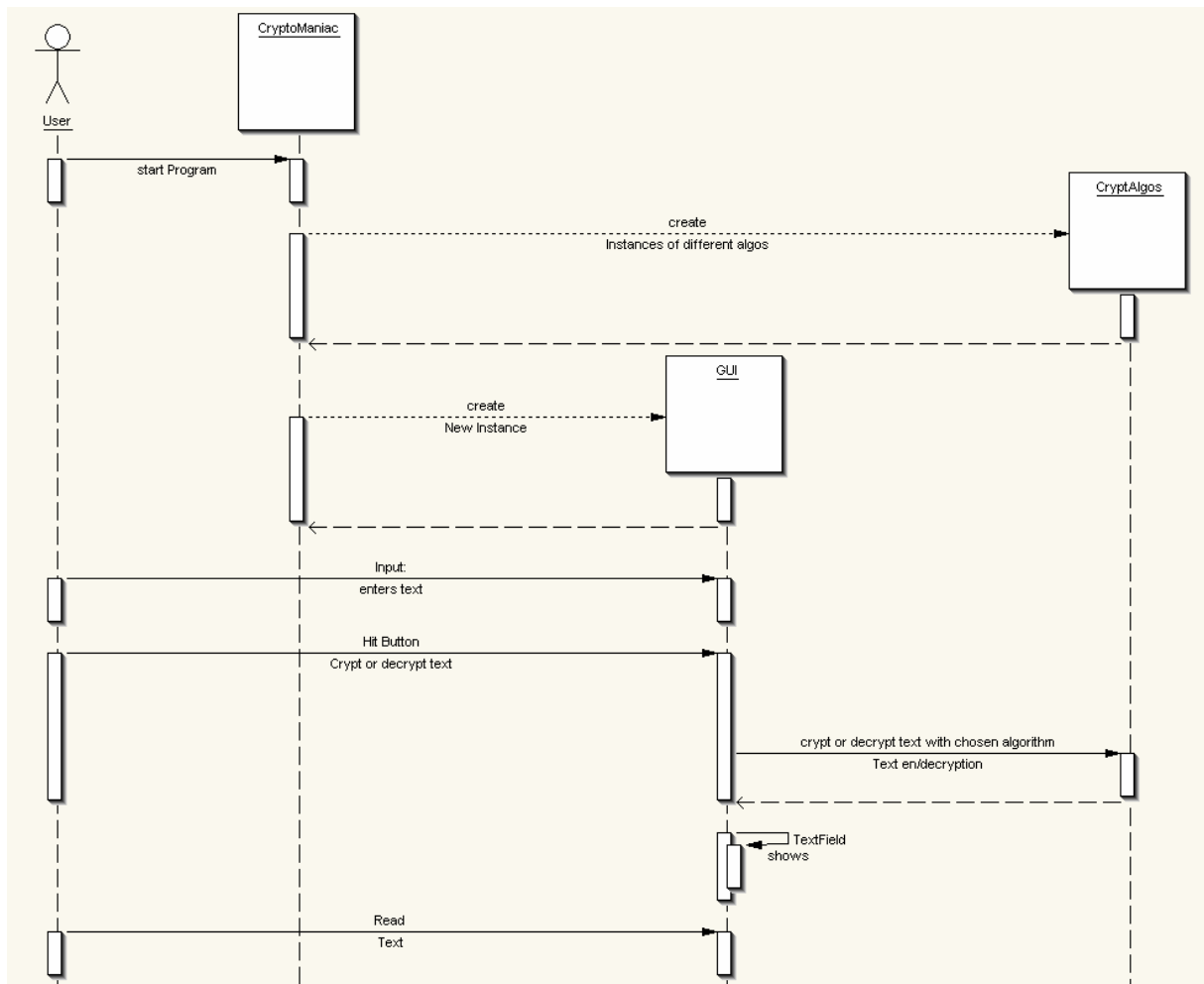
Vigenere³: Implementierung der Vigenere Verschlüsselung

Caesar: Implementierung der Cäsar Verschlüsselung

Playfair: Implementierung der Playfair Verschlüsselung

³ Genauere Beschreibung der einzelnen Verschlüsselungsverfahren Caesar, Vigenere und Playfair siehe Problembeschreibung oben

Sequenz Diagramm



Sequenzaufufe werden analog zur Beschreibung im Use Case ausgeföhrt.

Implementierung

Allgemein

Dieses Kapitel dient dem kurzen Überblick über die bereitgestellte Funktionalität der einzelnen Klassen. Für eine detaillierte Beschreibung der einzelnen Parameter und Methoden habe ich Anmerkungen und Kommentare direkt im Source-Code der jeweiligen Klasse gemacht.

CryptoManiac

Diese Klasse ist die Start- und Testklasse meines Projektes und enthält die Main Methode. Sie initialisiert eine neue Instanz von GUI und startet somit das Programm. Außerdem bietet sie noch Set- und Get-Methoden für eine GUI.

GUI

Die GUI (Graphical User Interface) übernimmt die gesamte Darstellung und Ablaufsteuerung des Programms. Sie ist mit Hilfe der Java Grafikbibliothek Swing implementiert worden. Die verschiedenen Interaktionsmöglichkeiten mit dem Programm werden über den Java Action-Listener abgewickelt, welche auf Benutzereingaben bei den Buttons „Verschlüsseln“ und „Entschlüsseln“, sowie der Drop-Down Liste mit den verschiedenen Verschlüsselungsmethoden, warten. Die GUI wurde großteils mit Hilfe des Visual Editors aus dem Eclipse Project erstellt, welcher eine grafische Oberfläche bietet. Der Code wurde dabei automatisch generiert. Bei der Erstellung der GUI wurden mehrere Swing Elemente mit Hilfe von „geschachtelten Layout-Managern“ gruppiert. Es wurde dabei ein Mix aus „Border Layout“ und „Grid Layout“ verwendet. Dadurch kann man die Größe des Fensters verändern während das Design als solches erhalten bleibt. Screenshot der GUI zum Zeitpunkt der Endabgabe siehe unten.

CryptAlgos

Dieses Interface definiert alle Methoden, welche bei jeder Verschlüsselungsart implementiert werden müssen. Konkret sind dies die beiden Methoden Ver- und Entschlüsseln, welche als Argument einen Text und ein Passwort entgegennehmen und ihrerseits wieder Text zurückgeben.

Caesar

Diese Verschlüsselungsklasse implementiert die originale Cäsar Verschlüsselung. Sie nimmt zwar ein Passwort entgegen, jedoch wird immer der fixe Wert 3 verwendet! Die Klasse ist so programmiert, dass jedoch andere Werte auch verwendet werden könnten.

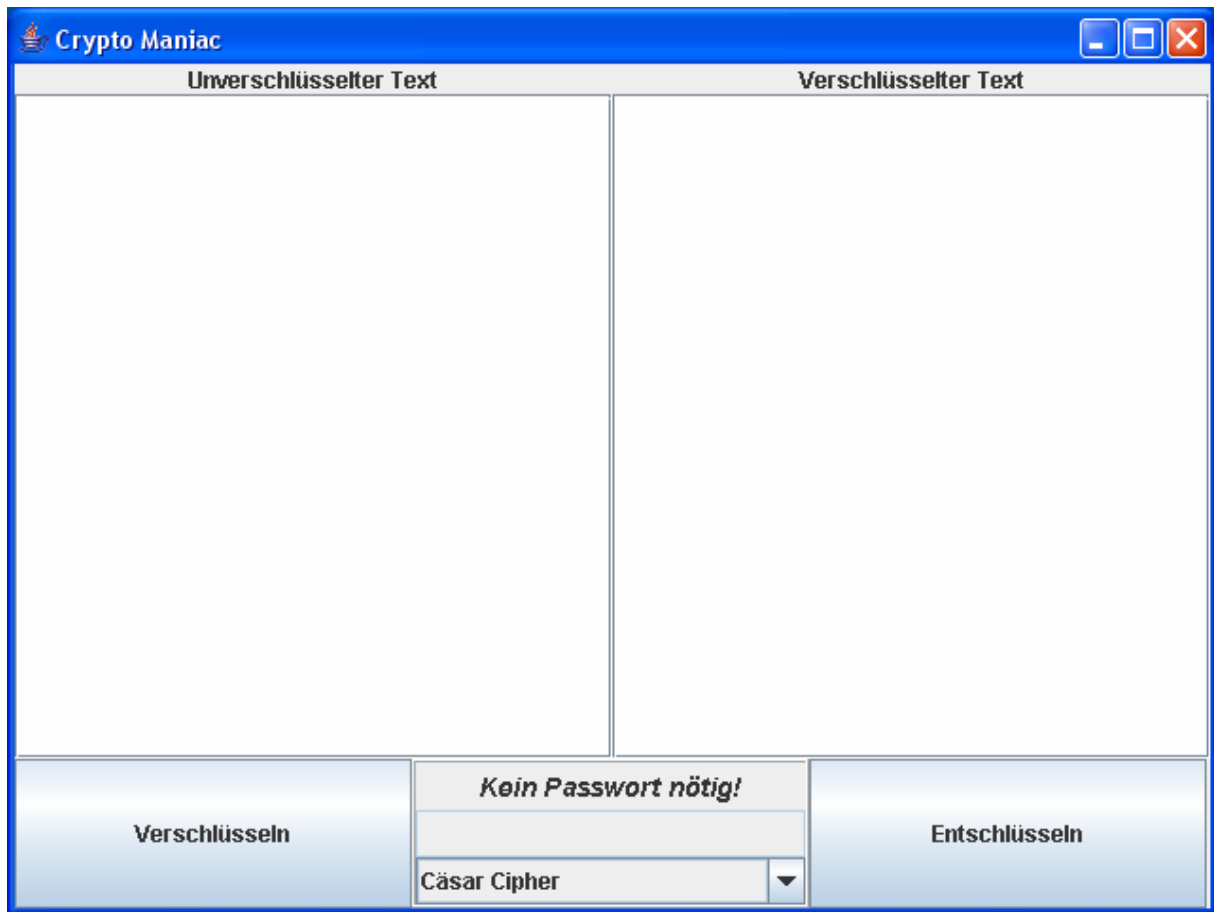
Vigenere

Diese Verschlüsselungsklasse implementiert die Vigenere Verschlüsselung (siehe Problemstellung) anhand der Interface Vorgaben in CryptAlgos.

Playfair

Diese Verschlüsselungsklasse implementiert die Playfair Verschlüsselung (Siehe Problemstellung) anhand der Interface Vorgaben in CryptAlgos.

Installation und Wartung



Screenshot der GUI bei Endabgabe

Installation

Zur leichteren Installation wurden alle Class- und Java Dateien in ein „Jar-Archiv“ verpackt. Dies hat zudem noch den Vorteil, dass sich das Programm bei installiertem Java leicht durch einen Doppelklick auf das Jar-Archiv starten lässt. Nach dem Start wird der Benutzer mit oben abgebildetem Bildschirm empfangen. Das zentrale Textfeld (hier mit „Kein Passwort nötig“ abgebildet) dient dabei als eine Art Statusleiste und ändert je nach Verschlüsselung seinen Text.

Wartung

Das Programm ist relativ leicht wart- und erweiterbar. Die wichtigsten Einstiegspunkte in den allgemeinen Code sind in der Klasse GUI zu finden. Entweder kann die visuelle Erscheinung des Programms mit dem Visual Editor verändert werden, oder die Programmlogik wird in den sog. „Action-Listnern“ manuell angepasst. Die Verschlüsselungsalgorithmen an sich sind in den jeweiligen Klassen gekapselt und können so leicht gewartet, oder ausgetauscht werden.

Will man zum Beispiel eine neue Verschlüsselungsmethode implementieren, so muss man zuerst eine Klasse kreieren, welche das Interface CryptAlgos implementiert. Danach werden die Ver- und Entschlüsselungsklassen implementiert und getestet. Alle benötigten Wartungspunkte im GUI habe ich der besseren Sichtbarkeit halber direkt im Source-Code mit W1, W2, W3 gekennzeichnet, um sie so leichter auffindbar zu machen. Um also den funktionierenden Algorithmus benutzen zu können muss zuerst eine neue Instanz in GUI kreiert werden (W1). Danach wird eine Zeile zum Drop Down Menü hinzugefügt (W2) und der Code beim Action-Listener der Combo Box (also dem Drop-Down Menü) angepasst (W3). Nun kann man ihn bereits benutzen.

Abschlussbemerkungen

Die Idee für das Projekt „CryptoManiac“ kam mir nach dem Lesen eines Spionage-Romans, welcher sich indirekt auch mit Verschlüsselungstechniken auseinander gesetzt hatte.

Danach hab ich mir überlegt ein Java-Programm zu schreiben, das etwas „Praktisches“, oder „Nützliches“ bietet und an dem man, darüber hinaus, auch Spaß beim Ausprobieren & Anwenden haben kann. Die Implementierung brachte mir neue Einsichten und Möglichkeiten mich in Java zu vertiefen, fiel mir jedoch schwerer als anfänglich geglaubt. Dies galt besonders für den damit verbundenen Zeitaufwand, welcher mit der Erlernung neuer Klassen, wie zum Beispiel Swing verbunden war. Schlussendlich hat mir das Projekt viel Spaß bereitet, vor Allem da ich jetzt etwas Lauffähiges herzeigen kann.