Doku: Sven (später Michael)

Klassenkarten:

1. ATM

1.1 Variablen:

1.1.1 da: Bild (public): Bild des Geldautomaten

1.1.2 T1, T2, T3, T4, T5, T6, T7, T8, T9, T0, T00, Tback: Taste (public): Tasten des Tastenfelds

1.1.3 Tkarterein, Tkarteraus: Taste (public): Tasten mit den man die Kreditkarte „steuern kann“

1.1.4 \_display: Display (private): Display des Geldautomaten

1.1.5 \_inputController: CInputController (private): Objekt das die Tastendruck-Events der Tasten der Klass ATM abfängt und verwaltet

1.1.6 \_animationMgr: CAnimationMgr (private): Objekt das die Animationen des Geldautomaten verwaltet

1.1.7 \_kontoAngemeldet, \_kreditkarteEingeschoben: boolean (private static): Vars zum Testen ob die Kreditkarte eingeschoben ist und ob diese rausgezogen werden darf

1.2 Funktionen:

1.2.1 Konstruktor: Für alle Objekte wird speicher reserviert, die Attribute der Knöpfe wird gesetzt

1.2.2 Update() (public void): Führt den Core-Loop des Programms immer wieder in einer while(true)-Schleife aus (für Funktionen siehe: )

1.2.3 KontoauszugDrucken() (public void): Startet die Animation des Kontoauszuges (Siehe)

1.2.4 geld\_auszahlen(int betrag) (public static void): Startet die Animation des Geldes (siehe:)

>betrag: Betrag der Animiert wird

1.2.5 geld\_einzahlen(int betrag) (public static void): Startet die Animation des Geldes (siehe: )

>betrag: Betrag der Animiert wird

1.2.6 setKontoAngemeldet(boolean value) (public static void): Setzt den Wert von \_kontoAngemeldet

1.2.7 handleKeyEvents() (public void): Die Tastdruck-Events werden abgeprüft und verarbeitet (siehe )

2. CInputController:

2.1 \_ButtonInfo (Unterklasse):

2.1.1 Variablen:

2.1.1.1 ID (int): ID der Taste wird hier gespeichert

2.1.1.2 Status(boolean): Variable die speichert ob die Taste gedrückt wurde

2.2 Variable:

2.2.1 \_buttonStatusList (private TreeMap<String, \_ButtonInfo>): Map in der in einem Datensatz die Speicheraddresse und die \_ButtonInfo eines Knopfes Gespeichert werden

2.2.2 \_tempButtonInfo (private \_ButtonInfo): temporäres \_ButtonInfo Objekt

2.3 Funktionen:

2.3.1 Konstruktor: Speicher für \_buttonStatusListe wird reserviert

2.3.2 knopfHinzufuegen(Taste key, int ID): Funktion um eine Taste der \_buttonStatusList hinzuzufügen

>key: Taste die hinzugefügt werden soll

>ID: ID die die Taste haben soll

2.3.3 tuWas(int ID): Funktion die ausgeführt wird wenn ein Knopf gedrückt wird

>ID: ID des Knopfes der gedrückt wurde

Ab hier Michael

3. Display:

3.1 Variablen:

3.1.1: Zeichnung: Zeichnung(public): Blauer Hintergrund mit Farbverlauf

3.1.2:TasteKontoauszug,Taste1,Taste11,Taste2,Taste21,Taste3,Taste31,Taste4,Taste5,Taste6,

Taste7,Taste8,TasteAusBes,TasteEinBes,TasteAbmelden (public): Tasten die im

Display zu sehen sind

3.1.3: Anzeige: Segment8x (public): Anzeigefeld für Kontostand im Display

3.1.4:Kontotext,Gesperrt,pinTipp,Kontonummertext,pinAendern,anderesGesperrt,geldAus,

Abheben,Einzahlen,Stern1,Stern2,Stern3,Stern4,Kontoauszug, geld\_einheit,

Kontonummer, Pin, GeldBetragAnzeige: Ausgabe(public): Textfelder im Display

3.1.5: GeldAbheben,DevisenAbheben,DeGe,x,y,pos\_x,pos\_y,counter,counter1,counter2,

counter3,counter4,counter5,counterS,a,b,K,sternCounter,A,B,C,D,Kontrolle,

Kontoangemeldet: int (public): Integer für Positionierung, Counter, etc.

3.1.6: eingabeAbheben,pin1,pin2,pin3,pin4,Pin1,Pin2,Pin3,Pin4: Eingabefeld(public):

Eingabefelder auf dem Display des Automaten

3.1.7: pinHintergrund,kontoHintergrund: Rechteck(public): Hintergründe für

Kontostandsanzeige

3.1.8: MKonto1,MKonto2,MKonto3,MKonto4,MKonto5: Konto(public): Kontoobjekt

3.1.9: update\_konto, update\_kontonummer, update\_pin, kreditkarteGesperrt:

boolean(public): Variablen zum verwalten des Kreditkartenstatuses bzw. welche fügeXZifferHinzu(int ziffer) funktion gerade ausgeführt wird

3.1.10: KonZ: D\_Kontonummerzeugs (public): Klasse Kontonummerzeugs

3.1.11: S: D\_Standartfkt (public): Klasse D\_Standartfkt

3.1.12: P: D\_Pinzeugs (public): Klasse D\_Pinzeugs

3.1.13: Kon: D\_Kontozeugs (public): Klasse D\_Kontozeugs

3.1.14: \_inputController: CInputController: Klasse CInputController

3.2 Funktionen:

3.2.1: Konstruktor: Erstellen des Displays und aller sichtbaren Sachen (Tasten,Ausgaben,etc.)

3.2.2: update() (public void): Führt den Core-Loop des Programms immer wieder in einer

while(true)-Schleife aus

3.2.3: sendKeyEvent(int ID) (public void): Übergibt ID der gedrückten Taste an

CInputController

3.2.4: loginKontonummer(int konto) (public void): Übergibt Wert einer Variablen an eine

Andere und zeigt die Pinabfrage an

3.2.5: KontoAbmelden() (public void): Zeigt Kontonummereingabe an

3.2.6: handle\_tastenfeldInput(int ziffer) (public static void): Funktion prüft ob eine Taste auf von dem Tastenfeld gedrück wurde und reagiert dementsprechend

4.D\_Kontozeugs:

4.1 Variablen:

4.1.1: S: D\_Standartfkt (public): Klasse D\_Standartfkt

4.1.2: x,y,pos\_x,pos\_y, aktuelleposition: int (public): Variable für Positionierung der

sichtbaren Elemente des Displays und counter

4.1.3: geldBetrag[]: int[] (public): Array speichert dem vom User eingegebenen Betrag

4.2 Funktionen:

4.2.1: Konstruktor: Speicher für Variablen reservieren und Zuweisung von Werten um spätere

Fehler zu vermeiden

4.2.2: xy() (public void): Weist x,y,pos\_x,pos\_y die im Display angegebenen Werte zu

4.2.3: fuegeBetragZifferHinzu(int ziffer)(public void): wird ausgeführt wenn eine Taste auf dem Tastenfeld gedrückt wird -> fügt dem aktuellen Betrag eine Ziffer hinzu bzw zieht eine ab

4.2.4: geldAbheben() (public void): Blendet nur die Elemente auf dem Display ein, die für den

Aktuellen Bildschirm benötigt werden

4.2.5: abhebenBest() (public void): prüft welches Konto angemeldet ist und zieht den vorher

eingegebenen Betrag vom Angemeldeten ab

4.2.6: DevisenAbheben() (public void): siehe 4.2.4

4.2.7: DevisenAbhebenBest() (public void): siehe 4.2.5

4.2.8: geldEinzahlen() (public void): siehe 4.2.4

4.2.9: einzahlenBest() (public void): prüft welches Konto angemeldet ist und fügt den vorher

eingegebenen Betrag zum Angemeldeten hinzu

4.2.10: Deviseneinzahlen() (public void): siehe 4.2.4

4.2.11: DevisenEinzahlenBest() (public void): siehe 4.2.9

4.2.12: kontostandAnzeigen() (public void): Blendet nur ein Siebensegment mit dem

aktuellen Kontostand und ein Ausgabefeld ein

4.2.13: DevisenKontostandAnzeigen() (public void): Blendet nur ein Siebensegment mit dem

aktuellen Devisenkontostand und ein Ausgabefeld ein

4.2.14: Kontozahlausgabe() (public int): prüft welches Konto angemeldet ist und gibt den

Entsprechenden Kontostand zurück

4.2.15: Devisenzahlausgabe() (public int): prüft welches Konto angemeldet ist und gibt den

Entsprechenden Devisenkontostand zurück

4.2.16: Kontoauszug() (public void): Zeigt ein Fenster mit einem Ausgabefeld und einer Taste

An

5.D\_Standertfkt

5.1: Variablen

5.1.1: x,y,pos\_x,pos\_y: int (public): Variablen für Positionierung der Elemente des Displays

**5.1.2: KL: boolean (public):**

5.2:Funktionen

5.2.1: Konstruktor: gleicht x,y,pos\_x,pos\_y den Variablen x,y,pos\_x,pos\_y in der Klasse

Display an

5.2.2: allesAusblenden() (public void): Blendet alle Elemente des Displays aus indem sie die

Größen auf 0 setzt, steht am Anfang fast jeder Funktion einer D\_Klasse die das Layout

Des Bildschirms verändert

5.2.3 hauptmenueAzeigen() (public void): Zeigt die Elemente an, die für das

Hauptmenüfenster des Displays benötigt werden

6.D\_Kontonummerzeugs:

6.1: Variablen:

6.1.1: x,y,pos\_x,pos\_y, aktuellePos: int (public): Variablen für Positionierung der im

Display sichtbaren Elemente, und ein Counter

6.1.2: Kontonummer[]: int[] (public): Array das die vom User eingegebene Kontonummer speichert

6.1.3: S: D\_Standartfkt (public): Klasse D\_Standartfkt (siehe 5.)

6.1.4: P: D\_Pinzeugs (public): Klasse D\_Pinzeugs (siehe

6.1.5: Kon: D\_Kontozeugs (public): Klasse D\_Kontozeugs (siehe 4.)

6.2: Funktionen:

6.2.1: Konstruktor: Weist dem Array Kontonummer Werte zu, die es auf anderem Wege nicht

Erreichen kann, weist der Variable aktuellePos den Wert 0 zu

6.2.2: xy() (public void): Weist x,y,pos\_x,pos\_y die im Display angegebenen Werte zu

6.2.3: fuegeKontonummerZifferHinzu (int ziffer) (public void): wird immer dann ausgeführt wenn eine Taste auf dem Tastenfeld gedrückt wird -> fügt der eingegebenen Kontonummer eine Ziffer hinzu bzw zieht ein ab

6.2.4: andereKontonummereingabe() (public void): Zeigt nur die Elemente auf dem Display

An, die für die Anmeldung benötigt werden

6.2.5: andereKontonummereingabebest() (public void): prüft welche Kontonummer man

Eingegeben hat und meldet das entsprechende Konto an

6.2.6: resetKontonummer() (public void): Löscht alle Zahlen aus dem Eingabefeld für die

Kontonummer und setzt aktuellePos auf 0

7.D\_Pinzeugs:

7.1: Variablen

7.1.1: S: D\_Standartfkt (public): Klasse D\_Standartfkt

7.1.2: x,y,pos\_x,pos\_y, aktuellePos: int (public): Variablen für Positionierung der einzelnen

Elemente des Displays und Variable aktuellePos

7.1.3: pin[]: int[] (public): Array für Speichern des Pins

7.2: Funktionen:

7.2.1: Konstruktor: Setzt pin[] auf Werte die es nicht durch Eingabe erreichen kann und setzt

aktuellePos auf 0

7.2.2: fuegePinZifferHinzu(int ziffer) (public void): wird ausgeführt wenn eine Taste am Tastenfeld gedrückt wird- > fügt dem Pin eine Ziffer hinzu bzw. zieht eine Ziffer ab

7.2.3: xy() (public void): Weist x,y,pos\_x,pos\_y die im Display angegebenen Werte zu

7.2.4: anderePinAbfrageAnzeigen() (public void): Blendet nur die Elemente des Displays ein,

die für die Pinabfrage nötig sind

7.2.5: \_pinAendernErfolgreich() (public void): setzt aktuellePos auf 0

7.2.6: \_anmeldungErfolgreich() (public void): Zeigt Elemente des Displays an, die das

Hauptmenue bilden und führt 7.2.5 aus

7.2.7: Anmeldebestaetigung() (public void): Prüft welches Konto angemeldet ist und prüft

Den dementsprechenden Pin, wenn er stimmt führt es 7.2.6 aus

7.2.8: pinAendern() (public void): Zeigt nur die Elemente des Displays an, die für das aktuelle

Fenster nötig sind

7.2.9: pinAendernBest() (public void): prüft welches Konto angemeldet ist und setzt in diesem

Den in 7.2.8 eingegebenen Pin

8.