16.09.2020

Shenia Scherer, Benjamin Yildirim, Noah Grand I3A

Vorprojekt Abschreibungen

IDPA Abschreibungsprogramm

Inhaltsverzeichnis

[1. Teil 1 2](#_Toc51169492)

[1.1 Team und Projektdetails 2](#_Toc51169493)

[1.2 Projektarbeit 2](#_Toc51169494)

[1.2.1 Ausgangslage 2](#_Toc51169495)

[1.2.2 Aufgabenstellung 2](#_Toc51169496)

[1.2.3 Projektorganisation 3](#_Toc51169497)

[1.2.4 Projektmanagementmethode 3](#_Toc51169498)

[1.3 Deklaration der Vorkenntnisse 3](#_Toc51169499)

[1.4 Deklaration der Vorarbeiten 4](#_Toc51169500)

[1.5 Deklaration der benutzten Standards 4](#_Toc51169501)

[1.6 Zeitplan 5](#_Toc51169502)

[1.7 Arbeitsjournal 6](#_Toc51169503)

[1.8 Organisation der Arbeitsergebnisse 12](#_Toc51169504)

[2. Teil 2 13](#_Toc51169505)

[2.1 Kurzfassung des IDPA Vorprojekts 13](#_Toc51169506)

[2.2 Abschreibungsprogramm nach IPERKA 14](#_Toc51169507)

[2.2.1 Informieren 14](#_Toc51169508)

[2.2.2 Planen 17](#_Toc51169509)

[2.2.3 Entscheiden 25](#_Toc51169510)

[2.2.4 Realisieren 26](#_Toc51169511)

[2.2.5 Kontrollieren 31](#_Toc51169512)

[2.2.6 Auswerten 33](#_Toc51169513)

# 1. Teil 1

## 1.1 Team und Projektdetails

**Projektname:** Abschreibungsrechner

**Startdatum:** 19.08.2020

**Enddatum:** 16.09.2020

**Projektmitarbeiter:**

* Shenia Scherrer
* Noah Grand
* Benjamin Yildirim

**Projektbetreuer:**

* Michael Schneider
* Corrado Olivito

## 1.2 Projektarbeit

### 1.2.1 Ausgangslage

Sie arbeiten als Praktikantin/Praktikant in der Buchhaltungsabteilung einer kleineren AG. Sie sehen, dass beim Jahresabschluss die Abschreibungen immer noch manuell berechnet werden. Sie schlagen der Inhaberin vor, dass Sie für die korrekte Berechnung der Abschreibungen eine Applikation programmieren (evtl. gemäss Merkblatt der ESTV für "Abschreibungen auf dem Anlagevermögen geschäftlicher Betriebe").

### 1.2.2 Aufgabenstellung

Folgende Anforderungen wurden vorgängig erarbeitet:

1. Die Applikation muss folgende Eingaben entgegennehmen: Anschaffungswert, Nutzungsdauer in Jahre und Restwert (am Ende der geplanten Nutzungsdauer) oder Abschreibung in Prozenten; lineare oder degressive Aschreibung; direkte oder indirekte Abschreibung.
2. Die Applikation soll folgende Daten ausgeben: Abschreibungsbetrag pro Jahr, Buchwert (Restwert) auf Anlagekonto bzw. Buchwert auf Anlagekonto und Betrag auf Wertberichtigungskonto, eventueller Restwert nach Nutzungsdauer.
3. Jede Eingabe muss auf Plausibilität geprüft werden.
4. Bei Eingabefehlern soll der Benutzer sinnvolle Hinweise zur Korrektur erhalten.
5. Die Ausgaben sollen dem Benutzer erklärt werden.
6. Folgende Dinge müssen ***nicht*** berücksichtigt werden:
   * Buchungen beim Verkauf von Anlagen
   * Veräusserungsgewinne und Veräusserungsverluste

### 1.2.3 Projektorganisation

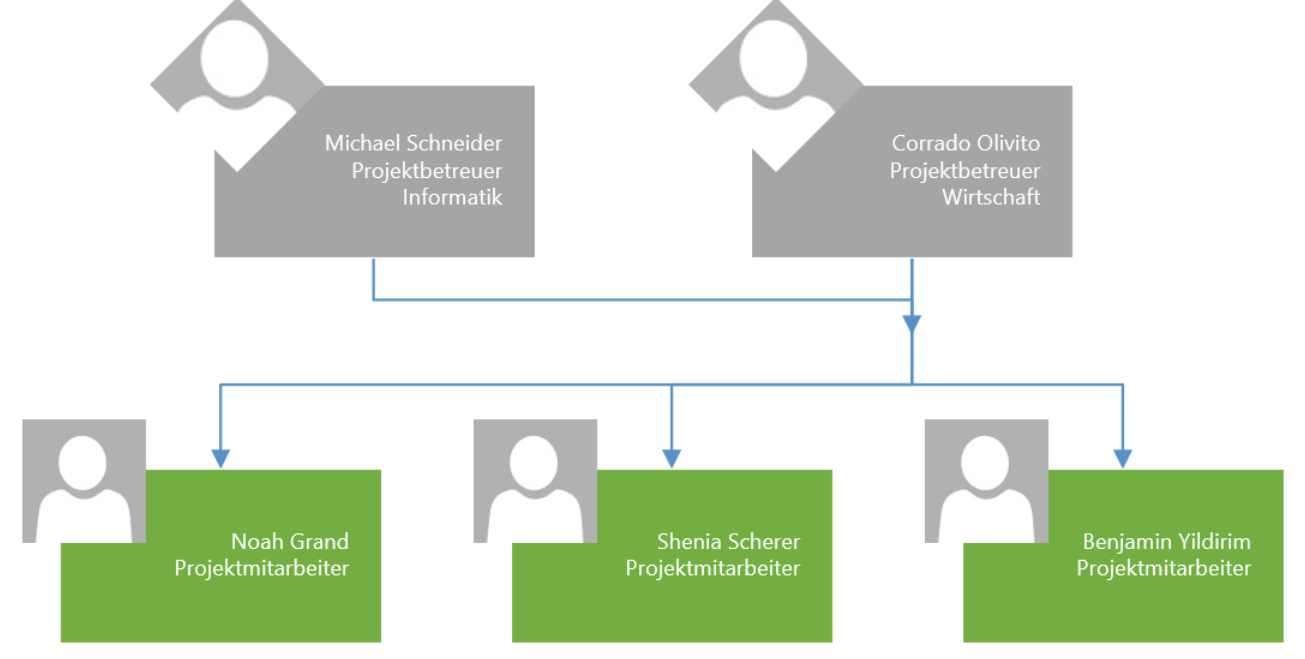


Abbildung 1 Projektorganisation

Für die Darstellung unseres Teams wurde ein Organigramm erstellt. Die Projektbetreuer sind Michael Schneider, zuständig für die Informatik und Corrado Olivito, zuständig für die Wirtschaft. Die Projektmitarbeiter sind alle auf einer Ebene, weil wir gemeinsam an einem Projekt arbeiten, in welchem es keine tieferen Strukturen braucht.

### 1.2.4 Projektmanagementmethode

Als Projektmanagementmethode haben wir IPERKA gewählt. Ihr Aufbau ist folgend aufgelistet:

1. **Informieren**
2. **Planen**
3. **Entscheiden**
4. **Realisieren**
5. **Kontrollieren**
6. **Auswerten**

Diese Methode hat viele Vorteile, insbesondere die klare Trennung zwischen Informieren und Planen. So hat jedes Gruppenmitglied die Möglichkeit sich individuell über Themen, wie die Abschreibung zu informieren und Unklarheiten klären. Bei IPERKA ist es auch zwingend nötig sich über die in der Planung gesammelten Ideen und Konzepte zu entscheiden. Dies kann Konflikte verhindern und eine effiziente Arbeit unterstützen. Der für die Teamarbeit wichtigste Punkt ist der Letzte, nämlich das Auswerten. Dort können sich alle Teammitglieder nochmals Gedanken über die Arbeit machen und diese dokumentieren. Es ist auch der Zeitpunkt, wo man sich eventuelle Verbesserungsmöglichkeiten und Negatives notiert, um dies bei zukünftigen Projektarbeiten zu verhindern.

## 1.3 Deklaration der Vorkenntnisse

|  |  |
| --- | --- |
| Teammitglied | Vorkenntnisse |
| Shenia Scherrer | Kapitel Abschreibungen, Programmierkenntnisse in Java, Maven, Git, JavaFX |
| Noah Grand | Kapitel Abschreibungen, Programmierkenntnisse in Java, Maven, Git, JavaFX |
| Benjamin Yildirim | Kapitel Abschreibungen, Programmierkenntnisse in Java, Maven, Git, JavaFX |

## 1.4 Deklaration der Vorarbeiten

Für das Programm an sich hatten wir keine Vorarbeiten, die wir benutzen konnten. Es wurde von Grund auf programmiert. Jedoch bei der Dokumentation konnten wir eine Dokumentation von den IMS-Programmierwochen benutzen, weil diese einen ähnlichen Inhalt, wie die IDPA Dokumentation hat. Der Inhalt der Dokumentation haben wir aber ebenfalls von Grund auf neu geschrieben. Als Vorarbeit könnten wir aber das Programmieren mit Maven nennen, weil wir dies auch in den Programmierwochen neu gelernt haben und es jetzt verstehen.

## 1.5 Deklaration der benutzten Standards

Als Entwicklungsumgebung haben wir uns für NetBeans IDE 8.2 entschieden. Als Buildtool für unser Programm verwendeten wir Maven. Der Programmcode wurde auf GitHub versioniert und hat uns bei der gemeinsamen Arbeit stark unterstützt. Unsere Projektmanagementmethode ist IPERKA und demnach haben wir auch unsere Dokumentation so gegliedert. Die Dokumentation des Codes wurde mit Javadoc gemacht und das Design des GUI’s wurde nach der ISO Norm 9241-110 kreiert.

Die Programmierung unseres Abschreibungsrechners haben wir MVC ähnlich umgesetzt, indem wir unser Model «Calculations.java» mit dem Singleton Pattern realisiert haben, damit von allen fxml-Controllern darauf zugegriffen werden kann.

## 1.6 Zeitplan



## 1.7 Arbeitsjournal

|  |  |  |
| --- | --- | --- |
| Datum: | 15.09.2020 | |
| **Ausgeführte Tätigkeiten:** | **Tätigkeit:** | **Person:** |
| Arbeit an der Dokumentation: Ich habe weiter offene Punkte in der Dokumentation befüllt. Im Teil Auswerten schrieb ich mein persönliches Fazit und reflektiere darin die Arbeit. | Benjamin Yildirim |
| Ich habe beim Scene Builder für zwei Lektionen gearbeitet, weil ich nicht herausfinden konnte, was das Problem war. Ich habe aber zu Hause weiter dokumentiert. | Shenia Scherer |
| Ich habe die Unittests geschrieben für die Klasse Calculations und habe das Javadoc gemacht, sowie die Testfälle ausgeführt. | Noah Grand |
| **Erreichte Ziele:** | **Ziel:** | **Person:** |
| Persönliches Fazit der Arbeit, zum Abschluss der Dokumentation gelangen. | Benjamin Yildirim |
| Das GUI-Design wurde fertiggestellt. Es gibt keinen Fehler mehr, es funktioniert wie geplant. | Shenia Scherer |
| Dokumentation des Codes, Testen des Projektes | Noah Grand |
| **Aufgetretene Probleme:** | **Problem:** | **Person:** |
| Konflikte in Git, zum Glück ohne grossen Verlust. (pull vergessen…) | Benjamin Yildirim |
| Ich hatte einige Probleme mit dem Scene-builder, das Bild passt nicht in die Anchorpan-Grösse. | Shenia Scherer |
| Singleton Pattern machte Probleme mit den Unittests | Noah Grand |
| **Hilfestellungen (wer, was):** | Herr Schneider an Benjamin | Frage zum Punkt Datenmodell in der Doku. (Eingabe und Ausgabedaten modellieren) |
| Noah und Herr Schneider an Shenia. | Frage zum Scene Builder. Bei den ersten Fenstern ist alles gleich und es funktioniert, aber bei den zweiten nicht. |
| Herr Schneider an Noah | Hilfe, beim Code analysieren, bei Singleton Unittest Problem |
|  |  |
| **Überzeiten:** | **Was:** | **Person:** |
| **Keine** | Benjamin |
| Ich konnte nicht herausfinden, wo das Problem lag, zum Glück haben Noah und ich es in letzter Minute herausgefunden. Ich habe Zeit verloren, weil ich auch an der Dokumentation arbeiten musste. | Shenia Scherer |
| Keine | Noah Grand |
| **Ungeplante Arbeiten:** | **Was:** | **Person:** |
| Neuer Klon vom Git Projekt machen. | Benjamin Yildirim |
| Überarbeitungen am GUI-Design. | Shenia Scherer |
| **Keine** | Noah Grand |
| **Pendenzenliste:** | **Pendenz:** | **Person:** |
| Fertigstellen der Doku, Testen abschliessen, Doku auf Grammatikfehler durchlesen. Präsentation vorbereiten. | Shenia Scherer |
| Fertigstellen der Doku, Testen abschliessen, Doku auf Grammatikfehler durchlesen. Präsentation vorbereiten. | Benjamin Yildirim |
| Dokumentation auf Grammatik testen, Präsentation vorbereiten | Noah Grand |
| **Erfolge:** | **Was:** | **Person:** |
| Sehen, wie die Dokumentation ein Ende nimmt. | Benjamin Yildirim |
| Das GUI-Design so benutzerfreundlich wie möglich gestalten. | Shenia Scherer |
| Völlig ausführliches Javadoc erstellt | Noah Grand |
| **Reflexion:** | **Reflexion:** | **Person:** |
| Diese Woche war sehr für mich sehr stressig, weil ich auch noch ein Bewerbungsgespräch hatte und mich darauf vorbereiten musste. Deshalb habe ich diese Woche weniger gearbeitet. Danke Noah und Shenia! | Benjamin Yildirim |
| Ich habe so viel Zeit verschwendet, nur um herauszufinden, worin das Problem lag. Ich dachte, es würde schnell gehen, so dass ich auch dokumentieren kann, aber es schien nicht so. Am Ende haben meine Kollegen und ich es herausgefunden, es war etwas Kleines. Ich habe auch zu Hause ein bisschen dokumentiert. | Shenia Scherer |
| Diese Woche war nicht sehr anstrengend. Es traten nicht wirklich grosse Probleme auf und so konnte ich das Javadoc und die Tests machen. | Noah Grand |
| **Vergleich mit dem Zeitplan:** | **Soll:** | **Ist:** |
| R-Phase abgeschlossen | R-Phase abgeschlossen |
| K-Phase ende | K-Phase ende |
| Beginn A-Phase | Beginn A-Phase |

|  |  |  |
| --- | --- | --- |
| Datum: | 08.09.2020 | |
|  | **Tätigkeit:** | **Person:** |
| **Ausgeführte Tätigkeiten:** | Ich habe für die Eingaben das GUI erstellt, welches automatisch Prozentzahl, oder Restbetrag einblenden, je nach Abschreibungsmethode. Die Eingaben werden auch nach Plausibilität geprüft und Fehler, wie Buchstaben werden abgefangen. Ebenfalls Weiterarbeit an der Dokumentation. | Benjamin Yildirim |
|  | Ich habe viel heute viel dokumentiert. Ich habe auch unsere GUI und unseren Anwendungsfall erklärt. Und habe das Design für die GUI gemacht | Shenia Scherer |
|  | Ich habe den Grundcode des Programmes beendet, die Eingaben werden nun verarbeitet, es werden Berechnungen gemacht und die Konten für jedes Jahr angegeben | Noah Grand |
| **Erreichte Ziele:** | **Ziel:** | **Person:** |
| Eingabeformular mit Validierung und Fehlerbehandlung der Eingaben. Diese Daten können jetzt für die Berechnungen verwendet werden. | Benjamin Yildirim |
| Die GUI hat ein ergonomisches Design und der Anwendungsfall ist gut erklärt, ebenso die GUI in der Dokumentation. | Shenia Scherer |
| Berechnung der Eingaben, Ausgaben der Abschreibungskonten | Noah Grand |
| **Aufgetretene Probleme:** | **Problem:** | **Person:** |
| Keine | Benjamin Yildirim |
| Keine | Shenia Scherer |
| Keine | Noah Grand |
| **Hilfestellungen (wer, was):** | **Hilfestellung:** | **Person:** |
| Keine | Benjamin Yildirim |
| Keine | Shenia Scherer |
| Keine | Noah Grand |
| **Überzeiten:** | **Was:** | **Person:** |
| Keine | Shenia Scherer |
| Keine | Benjamin Yildirim |
| Keine | Noah Grand |
| **Ungeplante Arbeiten:** | **Was:** | **Person:** |
| Keine | Shenia Scherer |
| Keine | Benjamin Yildirim |
| Keine | Noah Grand |
| **Pendenzenliste:** | **Pendenz:** | **Person:** |
| Styling mit CSS, um das GUI für den Benutzer etwas ansprechender zu machen. | Shenia Scherer |
| Fertigstellung der Dokumentation, testen | Benjamin Yildirim |
| Dokumentation in Code einbauen | Noah Grand |
| **Erfolge:** | **Was:** | **Person:** |
| Gutes, dynamisches Eingabeformular, mit Fehlerbehandlung. Gute Fortschritte in der Dokumentation | Benjamin Yildirim |
| GUI-Entwurf fertig. Erklärung für den Anwendungsfall und das GUI ist fertig | Shenia Scherer |
| Verlässliche Berechnung von Abschreibungen | Noah Grand |
| **Reflexion:** | **Reflexion:** | **Person:** |
| In dieser Woche haben wir zusammen intensiv gearbeitet und sehr gute Fortschritte erreicht. Das Programm ist fast fertig, es muss nur noch die Ausgabe dem Benutzer erklärt werden. Immer noch guter Kurs in der Doku. Wir sollten aber bald Testen. | Benjamin Yildirim |
| In dieser Woche haben wir viel erreicht. Wir konnten den R Teil abschliessen und sind nun bereit für das Testen und Auswerten, jedoch sollten wir den Blick auf die Dokumentation nicht vergessen. | Noah Grand |
| In dieser Woche haben wir gut im Team gearbeitet und deswegen konnten wir gut Ziele abarbeiten, jedoch sollten wir auch mehr Wert auf die Dokumentation legen | Shenia Scherer |
| **Vergleich mit dem Zeitplan:** | **Soll:** | **Ist:** |
| R-Phase | R-Phase |
| R-Phase |  |
| R-Phase | Ende R-Phase |

|  |  |  |
| --- | --- | --- |
| Datum: | 01.09.2020 | |
| **Ausgeführte Tätigkeiten:** | **Tätigkeit:** | **Person:** |
| Ich habe die GUI Design mit Balsamiq und Use Case Diagramm gemacht. | Shenia Scherer |
| Ich habe das Zustandsdiagramm erstellt und dieses erklärt. Ebenfalls Weiterarbeit am Programm. Weiterarbeit an der Dokumentation, mit Fertigstellung des Testkonzeptes mit Testfällen. | Benjamin Yildirim |
| Gute Fortschritte im Programm erreicht. Abschreibungsalgorithmen wurden geplant und bereits im Projekt implementiert, aber noch nicht mit dem GUI verbunden. | Noah Grand |
| **Erreichte Ziele:** | **Ziel:** | **Person:** |
| Ich habe GUI Design fertig erstellt. Ich bin aber mit Use Case noch nicht so weit. | Shenia Scherer |
| Zustandsdiagramm fertig, Testkonzept fertig, wir können zur Programmierung übergehen. | Benjamin Yildirim |
| Algorithmen zur Abschreibung implementiert und auf Funktion getestet, sodass diese mit dem GUI verbunden werden können. | Noah Grand |
| **Aufgetretene Probleme:** | **Problem:** | **Person:** |
| Ich hatte keine Probleme gehabt. | Shenia Scherer |
| Ich habe keinen grossen Problemen ausser einem kleinen Verständnisproblem zur Abschreibung gehabt. | Benjamin Yildirim |
| Keine Probleme. | Noah Grand |
| **Hilfestellungen (wer, was):** |  |  |
| Noah Grand an Benjamin Yildirim | Frage zur degressiven Abschreibung geklärt. |
|  |  |
|  |  |
| **Überzeiten:** | **Was:** | **Person:** |
| Keine | Shenia Scherer |
| Keine | Benjamin Yildirim |
| Keine | Noah Grand |
| **Ungeplante Arbeiten:** | **Was:** | **Person:** |
| Keine | Benjamin Yildirim, |
| Keine | Noah Grand |
|  | Keine | Shenia Scherer |
| **Pendenzenliste:** | **Pendenz:** | **Person:** |
| GUI design dokumentieren | Shenia Scherer |
| Klassendiagramm, Schnittstellen dokumentieren | Benjamin Yildirim |
| GUI mit den Algorithmen verbinden | Noah Grand |
| **Erfolge:** | **Was:** | **Person:** |
| GUI Design erfolgreich erstellt. | Shenia Scherer |
| Testkonzept, welches die ganze Applikation testet und schon bereitsteht, Zustandsdiagramm für ein besseres Verständnis der Situation. | Benjamin Yildirim |
| Algorithmus erfolgreich in Java implementiert | Noah Grand |
| **Reflexion:** | **Reflexion:** | **Person:** |
| Heute ist es gut gelaufen, ich bin stolz darauf, dass ich im Zeitplan immer noch sehr gut bin. | Shenia Scherer |
| Wir sind gut im Zeitplan und sind jetzt voll in der Programmierung unseres Abschreibungsrechners. Auch der Fortschritt in der Dokumentation ist gut und wird von der Gruppe auf dem Laufenden gehalten. | Benjamin Yildirim |
| Das Projekt macht im Zeitplan mit, der schwierigste Teil der Logik ist implementiert. Das Nächste Mal müssen wir unbedingt mit der Implementierung des GUIs beginnen und uns mit der Ausgabe befassen. | Noah Grand |
| **Vergleich mit dem Zeitplan:** | **Soll:** | **Ist:** |
| I-Phase abgeschlossen | I-Phase abgeschlossen |
| Ende P-Phase | P-Phase abgeschlossen |
| E-Phase abgeschlossen | E-Phase abgeschlossen |

|  |  |  |
| --- | --- | --- |
| Datum: | 25.08.2020 | |
| **Ausgeführte Tätigkeiten:** | **Tätigkeit:** | **Person:** |
| Vorlage Dokumentation,  Vorlage Arbeitsjournal,  GitHub Konto erstellen | Benjamin Yildirim |
| Vorlage Zeitplan, Teilaufgaben eintragen  GitHub Konto erstellen | Shenia Scherer |
| GitHub Projekt Erstellen  Python Entwurf des Programmes | Noah Grand |
| **Erreichte Ziele:** | **Ziel:** | **Person:** |
| Vorlagen für unserer Dokumente | Benjamin Yildirim |
| Vorlage für unserer Zeitplan | Shenia Scherer |
| GitHub einrichten | Noah Grand |
| **Aufgetretene Probleme:** | **Problem:** | **Person:** |
| Wir haben keine Probleme gehabt. | Shenia Scherer |
|  |  |
| Keine Probleme | Noah Grand |
| **Hilfestellungen (wer, was):** | Keine | Benjamin Yildirim |
| Keine | Shenia Scherer |
| Keine | Noah Grand |
|  |  |
| **Pendenzenliste:** | **Pendenz:** | **Person:** |
| GUI Design machen, dokumentieren | Shenia Scherer |
| Klassendiagramm, Schnittstellen, Datenmodell erstellen | Benjamin Yildirim |
| Algorithmen für die Berechnung ausprobieren in separatem Projekt. | Noah Grand |
| **Erfolge:** | Use-Case Diagramm erstellt | Shenia Scherer |
|  | Vorlage für die Dokumentation erstellt | Benjamin Yildirim |
|  | GitHub Repository erstellt und Projekt hochgeladen. | Noah Grand |
| **Reflexion:** | **Reflexion:** | **Person:** |
| Die Vorlagen wurde so erstellt, dass wir alle sehen, wo was noch ausgefüllt werden muss (TODO). Konzentrierte Arbeit. | Benjamin Yildirim |
| Heute ging alles gut, wir sind immer noch im Zeitplan drin. | Shenia Scherer |
| Alles nach Zeitplan | Noah Grand |
| **Vergleich mit dem Zeitplan:** | **Soll:** | **Ist:** |
| In der I-Phase | In der I-Phase |
|  |  |
|  |  |

## 1.8 Organisation der Arbeitsergebnisse

Als primäres Mittel für den Dateiaustausch verwendeten wir Microsoft Teams in einem eigenen Chat. Über dieses konnten wir auch unsere Dokumentation gemeinsam bearbeiten, in dem wir es alle mit SharePoint geöffnet hatten. Der Programmcode haben wir auf GitHub gepushed. Dieses Tool garantierte uns, dass jeder von uns mit der aktuellen Version gearbeitet hat. Wir alle hatten dabei unseren eigenen Branch, in dem wir unsere persönlichen Änderungen machten und diese mit dem Master-Branch gemerged wurde. Diese Dateien waren für uns immer zugreifbar und auch durch Versionierung dementsprechend gesichert.

1.9 Anleitung Installation und Bedienung

**Installation**

Um das Programm ausführen zu können, muss Java bereits auf ihrem PC installiert sein.

Link: <https://www.java.com/de/download/win10.jsp> (Aktuell per 16.09.2020)

Nun müssen sie das heruntergeladene Projekt entpacken, mit der Software Ihrer Wahl (z.B. WinRar).

Das letzte was sie nun tun müssen ist, das CMD zu öffnen (Windows Taste – CMD eingeben – Enter), und dann folgenden Befehl eingeben:

*Java -jar «pfad zu dem heruntergeladenen Projekt»/target/ IDPA\_Vorprojekt-1-jar-with-dependencies.jar*

Zum Beispiel:



Abbildung 2 Beispiel Ausführen des Programmes

Sollte dies nicht funktionieren, kann es sein, dass sie die Umgebungsvariable noch nicht gesetzt haben:

<https://www.java.com/de/download/help/path.xml> (Aktuell per 16.09.2020)

**Bedienung**

Die Bedienung des Programmes ist eher selbsterklärend.

1. Wählen sie die Abschreibungsart aus. (Bei direkt ist auf dem Anlagevermögen Konto der aktuelle Wert der Anlage ersichtlich, bei indirekt ist der Anschaffungswert ersichtlich.)

2. Wählen sie die Abschreibungsmethode aus. (Bei linear wird jedes Jahr der gleiche Betrag abgeschrieben, bei degressiv wird jedes Jahr ein prozentualer Teil, des aktuellen Wertes der Anlage abgeschrieben.)

Geben sie den Anschaffungswert ein, in CHF.

Geben sie die Nutzungsdauer an, bzw. über wie viele Jahre abgeschrieben werden soll.

Geben Sie nun noch ein, wie gross der Restwert der Anlage am Ende der Nutzungsdauer sein soll oder wieviel Prozent jedes Jahr abgeschrieben werden sollen. (Eingabe unterschiedlich je nach linear oder degressiv.)

Drücken sie nun unten rechts auf Berechnen.

Nun können sie durch das Klicken auf die Pfeiltasten, durch die Auszüge der verschiedenen Jahre navigieren.

1.10 Projektverträge, Disposition

**Vertrag zum Vorprojekt**

|  |  |
| --- | --- |
| **Projektname:** | Abschreibungsrechner |
| **Gruppenmitglieder mit Klasse und E-Mail:** | Noah Grand I3a, [noah.grand@students.ksba.ch](mailto:noah.grand@students.ksba.ch)  Benjamin Yildirim I3a, [benjamin.yildirim@students.ksba.ch](mailto:benjamin.yildirim@students.ksba.ch)  Shenia Scherer I3a, [shenia.scherer@students.ksba.ch](mailto:shenia.scherer@students.ksba.ch) |
| **Betreuende Lehrpersonen und E-Mail:** | Herr Schneider, [michael.schneider@kanti-baden.ch](mailto:michael.schneider@kanti-baden.ch)  Herr Olivito, [corrado.olivito@kanti-baden.ch](mailto:corrado.olivito@kanti-baden.ch) |
| **Titel:** | Abschreibungsrechner |
| **Beschreibung:** | Der Abschreibungsrechner kann Anlangen auf verschiedene Methoden abschreiben (linear, degressiv, direkt, indirekt). |
| **Startdatum des Vorprojekts:** | 19.08.2020 |
| **Abgabe mit Zeitpunkt:** | 16.09.2020 um 22:00 Uhr |
| **Anforderungen:** | Es kann eingegeben werden, ob linear, oder degressiv, indirekt, oder direkt abgeschrieben werden kann. Der Benutzer kann Anschaffungswert, Nutzungsdauer, Restwert und Abschreibung in % eingeben. Die Ausgabe wird dem Benutzer erklärt. |
| **Datum und Unterschrift der Mitglieder:** | 26.08.2020, Benjamin Yildirim  26.08.2020, Noah Grand  26.08.2020, Shenia Scherer |
| **Datum und Unterschrift der Lehrpersonen:** |  |

# 2. Teil 2

## 2.1 Kurzfassung des IDPA Vorprojekts

**Ausgangssituation**

In der Interdisziplinären Projektarbeit werden zwei Projekte erarbeitet. Vor dem Hauptprojekt wird ein in einem Team aus drei Personen ein Vorprojekt gemacht. Dieses soll uns auf die Hauptarbeit vorbereiten und uns etwas Routine bringen. Ziel dieses Projekts ist, dass wir ein Programm realisieren, welches Abschreibungen entweder direkt oder indirekt in den zwei Abschreibungsarten linear und degressiv tätigen kann. Der Benutzer kann einen Anschaffungswert eingeben, sowie Nutzungsdauer, Restbetrag und Abschreibungsbetrag in Prozent. Zum Programm muss auch eine Dokumentation geschrieben werden, die zusammen mit dem Programm abgegeben wird.

**Umsetzung**

Das Programm haben wir gemeinsam in der IDE NetBeans 8.2 programmiert. Dazu verwendeten wir das Framework JavaFX, welches eine grosse Palette an GUI-Elementen liefert. Unser erster Schritt war aber noch nicht die Programmierung, sondern die Informationsphase und die Planung. Zuerst einmal schaute jedes Teammitglied, dass es das Thema Abschreibungen aus der Finanz und Rechnungswesen versteht. Nur so konnten wir das Projekt richtig planen. Nach der Entscheidungsphase haben wir noch vor der Graphischen Benutzerschnittstelle die Algorithmen zur Berechnung der Abschreibungen der einzelnen Perioden programmiert und in der Konsole auf die Funktion getestet. Gegen Ende Implementierten wir noch das GUI und validierten die Eingaben auf Fehler und Plausibilität und sorgten dafür, dass allfällige Fehler nicht zu einem Absturz führen. Fortlaufend schrieben wir an der Dokumentation, damit keine Arbeit aufgeschoben wird.

**Ergebnis**

Unser Ergebnis ist der Abschreibungsrechner, der mit den Eingegebenen Daten die Anlage abschreibt. Damit er visuell auch noch etwas hermacht, verschönerten wir unser GUI mit CSS. Der Anwender kann die Art der Abschreibung und die Abschreibungsmethode über Radiobuttons definieren. Die Berechnung wird gestartet, sobald der Benutzer auf den Button berechnen klickt. Das Ergebnis wird auf einem zweiten Fenster in einer Pagination dargestellt. Dort kann der Benutzer ohne viel Suchaufwand sich die einzelnen Jahre anzeigen lassen, was zur verbesserten Übersicht dient. Die Dokumentation zu diesem Projekt ist nach den Phasen von IPERKA gegliedert.

## 2.2 Abschreibungsprogramm nach IPERKA

### 2.2.1 Informieren

#### 2.2.1.1 Personas

Unsere Personas richten sich an die gesamte Anwendung.

|  |  |
| --- | --- |
| Name: | Reto Steiner |
| Alter: | 49 |
| Ziele: | Will für sein Taxi die richtigen Abschreibungsbeträge in den jeweiligen Perioden schnell und einfach ausrechnen können. |
| Aufgaben: | Abschreibungsrechner bis zum 19.9.20 bereit haben, um einfachere Abschreibungen zu tätigen. |
| Kenntnisse: | Ist kein begeisterter Buchhalter und hat keine grossen Talente darin. Er hat aber gutes Gefühl, um den Wert eines Fahrzeugs bestimmen zu können. |
| Einstellung: | Er möchte nur einfach zu verstehende Programme bedienen und will direkt, ohne grosses Einarbeiten zum Ergebnis kommen. |

|  |  |
| --- | --- |
| Name: | Vreni Moser |
| Alter: | 57 |
| Ziele: | Möchte für ihre Schneiderei ein Programm haben, damit sie den Wert ihrer Anlagen besser im Auge haben kann. |
| Aufgaben: | Abschreibungsrechner bis zum 19.9.20 bereit haben, um einfachere Abschreibungen zu tätigen. |
| Kenntnisse: | Hat keine Kenntnisse in der Buchhaltung und ist deshalb auf Hilfe angewiesen. |
| Einstellung: | Sie will die Buchhaltung bald selber übernehmen, weil ein Buchhalter viel kostet. |

#### 2.2.1.2 Anforderungsanalyse

|  |  |
| --- | --- |
| Nummer | Anforderung |
| 1 | Der Benutzer kann einen Anschaffungswert, Nutzungsdauer in Jahre im Abschreibungsrechner eingeben. |
| 2 | Der Benutzer kann eine der Abschreibungsarten auswählen: linear oder degressiv |
| 3 | Der Benutzer kann entscheiden, ob seine Anlage direkt oder indirekt abgeschrieben wird. |
| 4 | Das Programm Abschreibungsrechner gibt nach der Berechnung den korrekten Abschreibungsbetrag für die lineare Abschreibung aus. |
| 5 | Das Programm Abschreibungsrechner gibt nach der Berechnung den korrekten Abschreibungsbetrag für die degressive Abschreibung aus. |
| 6 | Das Programm verbucht den richtigen Betrag im Wertberichtigungskonto bei indirekter Abschreibung. |
| 7 | Das Programm verbucht den richtigen Betrag im Konto Abschreibungen. |
| 8 | Der Abschreibungsrechner erklärt die Berechnungen dem Benutzer bei der Ausgabe nachvollziehbar. |
| 9 | Der Abschreibungsrechner fängt Eingabefehler des Benutzers ab und macht sie mit sinnvollen Hinweisen bemerkbar. |
| 10 | Das Programm Abschreibungsrechner nimmt nur plausible Eingaben entgegen. |
| 11 | Der Abschreibungsrechner wechselt automatisch zwischen Restwert und Abschreibungsbetrag in Prozent, sobald der Benutzer die Abschreibungsmethode ändert. |

#### 2.2.1.3 Use Case Diagramm

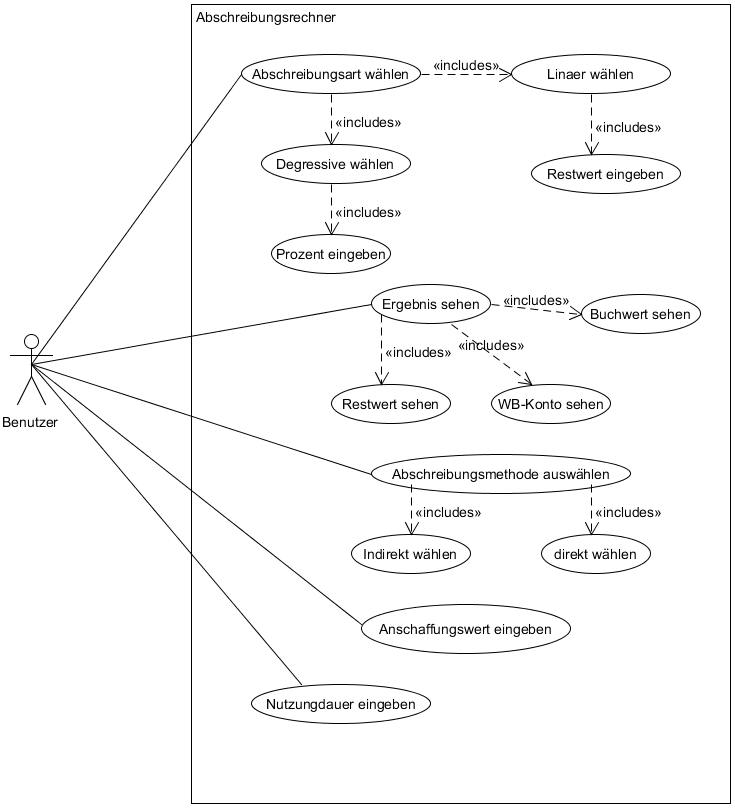


Abbildung 3 Use case diagramm

Das Programm soll die Abschreibungen berechnen. Als erstes muss der Benutzer wählen, welche Abschreibungsart er wünscht, entweder direkt oder indirekt. Später kann der Benutzer zwischen linear und degressiv wählen. Wenn der Benutzer Linear wählt, wird der Restwert berechnet, aber wenn der Benutzer degressiv wählt, wird der Prozentsatz berechnet.

Danach muss der Benutzer nur noch angeben, wieviel Anschaffungswert und Nutzungsdauer er abschreiben möchte. Nachdem der Benutzer alle Felder ausgefüllt hat, muss er auf eine Schaltfläche klicken, die die Abschreibungen zählt. Dies bringt ihn zu einer anderen Ansicht, die dem Benutzer die Abschreibungen pro Jahr, Buchwert auf Anlagekonto, Betrag auf Wertberichtigungskonto und den Restwert nach Nutzungsdauer anzeigt.

#### 2.2.1.4 Zustandsdiagramm

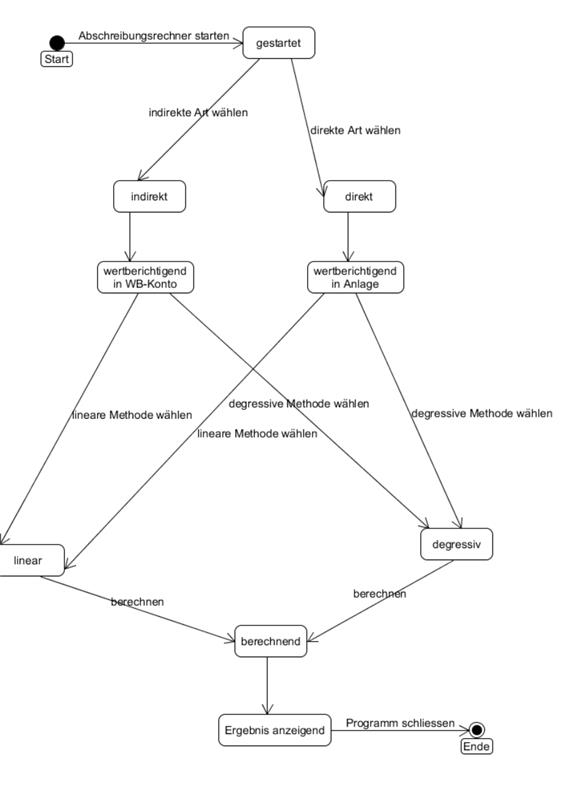


Abbildung 4 Zustandsdiagramm

Sobald der Abschreibungsrechner gestartet ist, kann der Benutzer die Art der Abschreibung auswählen. Das ist entweder indirekt oder direkt. Darauf folgt die Bestimmung der Wertberichtigung, entweder in einem separaten Wertberichtigungskonto oder gleich auf die Anlage selbst. Der Anwender kann auch bestimmen, ob linear oder degressiv abgeschrieben werden soll. Sobald der Anwender die Abschreibungsbeträge berechnen lässt, wechselt das Programm in den Zustand berechnend und anschliessend in den ausgeben Zustand. Dort kann der Benutzer auch eine Erklärung ansehen.

### 2.2.2 Planen

#### 2.2.2.1 Konzept

Der Abschreibungsrechner soll seinen Zweck erfüllen. Das GUI muss dabei den ISO 9241-110 Normen der Dialoggestaltung entsprechen. Der Benutzer wird in jedem Fenster darüber informiert, welche Eingaben zwingend getätigt werden müssen, damit das Ergebnis berechnet werden kann. Sollte er etwas vergessen, wird ihm automatisch eine Meldung am unteren Fensterrand angezeigt. Dies geschieht auch bei Fehleingaben. Das Programm kann auch Fehleingaben abfangen, wie Zeichenketten, anstatt geforderten positiven Zahlen und macht das dem Anwender bekannt. Alle Ergebnisse werden auf einem anderen Fenster ausgegeben. Die Ausgabe ist so gestaltet, dass die einzelnen Jahre, in denen abgeschrieben wurde einzeln betrachtet werden können. Dazu kann der Benutzer über eine Pagination am unteren Rand das Jahr anklicken und anzeigen lassen.

#### 2.2.2.2 GUI Design

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

Abbildung 5 GUI Design

Für das GUI-Design haben wir uns an die Richtlinien der ISO 9241-110 orientiert. Diese geben Designvorgaben an. Wir haben dies so gemacht, weil wir dieses In der Schule kennengelernt haben.

**Aufgabenangemessenheit**

Das GUI-Design für das Projekt hat alles, was Sie brauchen, um die Abschreibungen zu berechnen.

**Selbstbeschreibungsfähigkeit**

Alle Buttons sind beschriftet und es gibt Beschriftungen, um dem Benutzer mitzuteilen, was er tun sollte.

**Erwartungskonformität**

Wir haben dieses Design so gestaltet, dass es auf den ersten Blick einfach und verständlich ist. Die GUI-Elemente sind so positioniert, wie es der Benutzer auch von anderen Programmen kennt.

**Lernförderlichkeit**

Das Design selbst wird sich nicht bewegen oder verändern, so dass der Benutzer weiss, wo er sich befindet und was er tun sollte, um nicht stecken zu bleiben. Die Anzeige unten links zeigt auch an, was der Benutzer tun muss, wenn etwas einen Fehler erzeugt, oder fehlt.

**Steuerbarkeit**

Am Anfang kann der Benutzer entscheiden, ob er sich für Liner oder degressiv entscheiden möchte.

**Fehlertoleranz**

Wenn der Benutzer keine Zahl eingibt, oder Felder leer lässt und auf die Taste „Berechnen“ klickt, erscheint eine Fehlermeldung unten links.

**Individualisierbarkeit**

Das Design kann nicht weiter angepasst werden. Der Anwender kann aber das Fenster frei auf dem Bildschirm umpositionieren.

#### 2.2.2.3 Klassendiagramm

Abbildung 6 Klassendiagramm

Die Klasse Calculations ist unser Model, welches alle Funktionen zur Berechnung der Abschreibungen beinhaltet. Die Klasse Konto dient lediglich zur Verbuchung der Beträge und wird auch bei der Ausgabe gebraucht. Die beiden fxml-Controller können die Konten ebenfalls über das Model Calculations abrufen, um die Ausgabe darstellen zu können. Die Klassen liegen in Packages, welche nach dem Schema i3a.schbennoa.idpa\_vorprojekt benannt sind.

#### 2.2.2.4 Schnittstellen / Projektumfeld

Systemdarstellung:

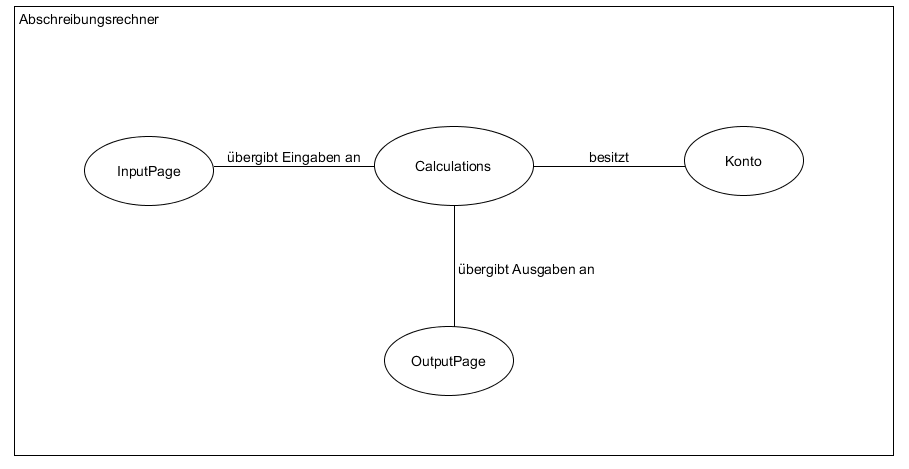


Abbildung 7 Systemdarstellung

In dieser Systemdarstellung zeigen wir, wie unser Programm sich von der Aussenwelt abgrenzt. Wir müssen keine Schnittstellen aufzeigen, weil wir für diese Anwendung keine benötigen. Zum Beispiel wird keine Datenbank gebraucht, welche über eine Schnittstelle mit dem Programm verbunden werden muss. Diese Darstellung zeigt auch die Beziehungen zwischen unseren Systemelementen auf. Das wichtigste Element, welches die Businesslogik enthält, ist Calculations. Es ist auch Bindeglied zwischen den Eingaben des Benutzers, der Ausgabe und zu den verschiedenen Konten im Zusammenhang mit Abschreibungen.

#### 2.2.2.5 Datenmodell

Datenmodell des Abschreibungsrechners:

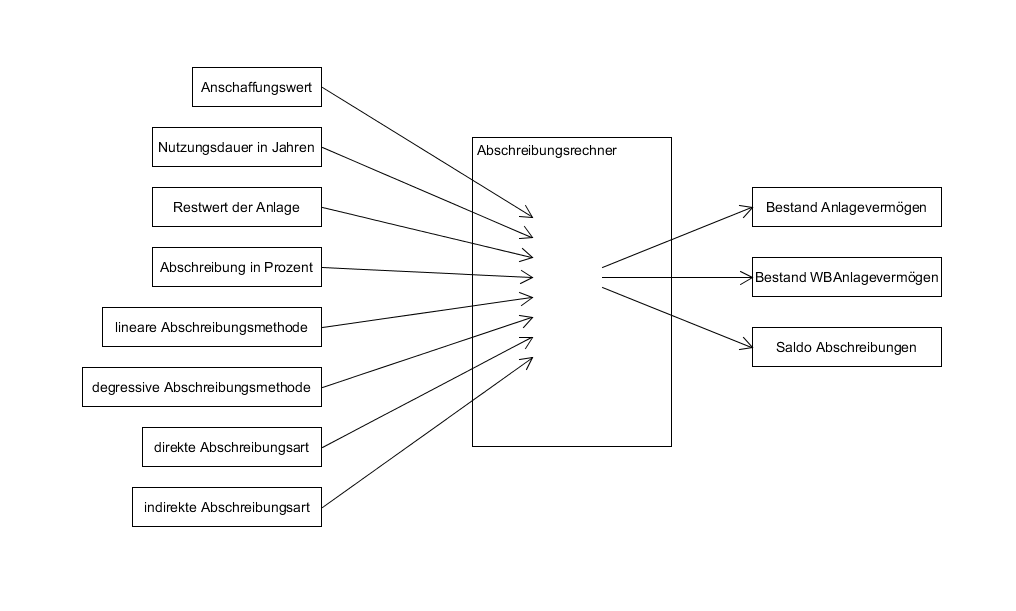


Abbildung 8 Datenmodell

Erklärung:

Die Box in der Mitte symbolisiert unser Abschreibungsprogramm. Links davon stehen alle benötigten Eingaben, die vom Benutzer kommen, um eine Berechnung durchführen zu können, rechts davon alle Ausgaben des Programms. Dazu gehört vor allem der Anschaffungswert, Nutzungsdauer, Restwert der Anlage und Abschreibung in Prozent. Die Angaben degressiv, linear, direkt und indirekt sind vom Programm bereits standardmässig gesetzt, können aber vom Anwender geändert werden. Hat der Benutzer die Berechnung gestartet, gibt der Abschreibungsrechner den Bestand des Anlagevermögens, den Bestand WBAnlagevermögen und den Saldo Abschreibungen aus.

#### 2.2.2.6 Fehlerbehandlung

Um die Möglichkeit von Fehleingaben zu minimieren, haben wir so wenige Textfelder wie möglich gemacht. Sollte der Benutzer eine fehlhafte Eingabe machen, wird er darauf hingewiesen. Das Programm führt die Berechnungen erst durch, wenn der Benutzer alle Eingaben in einem validen Format gemacht hat.

#### 2.2.2.7 Coding-Style

Unser Java Code wird nach der Standardformatierung von NetBeans formatiert. Diese wird mit Rechtsklick in den Code -> Format erreicht. Alle Variablen und Attribute in unserem Projekt sind nach Camel-Case benannt, alle mit einem Kleinbuchstaben beginnend, wie in der Java Namenskonvention.

Die Codedokumentation setzten wir mithilfe des Javadocs um, welches in einer lokalen Webseite präsentiert wird.

#### 2.2.2.8 Testkonzept

Name des Programms: Abschreibungsrechner

Testinfrastruktur

* **Testsystem:** Acer Aspire A715 71G
* **Testhilfsmittel:** Windows 10 Home, Java Version 8, Netbeans IDE 8.2

Anforderungen

|  |  |
| --- | --- |
| Nummer | Anforderung |
| 1 | Der Benutzer kann einen Anschaffungswert, Nutzungsdauer in Jahre im Abschreibungsrechner eingeben. |
| 2 | Der Benutzer kann eine der Abschreibungsarten auswählen: linear oder degressiv |
| 3 | Der Benutzer kann entscheiden, ob seine Anlage direkt oder indirekt abgeschrieben wird. |
| 4 | Das Programm Abschreibungsrechner gibt nach der Berechnung den korrekten Abschreibungsbetrag für die lineare Abschreibung aus. |
| 5 | Das Programm Abschreibungsrechner gibt nach der Berechnung den korrekten Abschreibungsbetrag für die degressive Abschreibung aus. |
| 6 | Das Programm verbucht den richtigen Betrag im Wertberichtigungskonto bei indirekter Abschreibung. |
| 7 | Das Programm verbucht den richtigen Betrag im Konto Abschreibungen. |
| 8 | Der Abschreibungsrechner erklärt dem Benutzer die Daten, mit eindeutigen, verständlichen Kontenbeschriftungen. |
| 9 | Der Abschreibungsrechner fängt Eingabefehler des Benutzers ab und macht sie mit sinnvollen Hinweisen bemerkbar. |
| 10 | Das Programm Abschreibungsrechner nimmt nur plausible Eingaben entgegen. |

Testfallspezifikationen

|  |  |
| --- | --- |
| Testfallnummer: | 1 |
| Anforderung: | 1 |
| Voraussetzungen: | Das Programm «Abschreibungsrechner» ist gestartet und zeigt Das Fenster “Abschreibungsrechner” an. |
| Eingabe: | 1. Bei Anschaffungswert «20000» eingeben 2. Bei Nutzungsdauer «5» eingeben |
| Ausgabe: | 1. Das Programm nimmt die Zahlen an und gibt keine Hinweise unten an. |

|  |  |
| --- | --- |
| Testfallnummer: | 2 |
| Anforderung: | 2 |
| Voraussetzungen: | Das Programm «Abschreibungsrechner» ist gestartet und zeigt Das Fenster « Abschreibungsrechner » an. |
| Eingabe: |  |
| Ausgabe: | 1. Unter Abschreibungsart wird «linear» und «degressiv» aufgelistet. |

|  |  |
| --- | --- |
| Testfallnummer: | 3 |
| Anforderung: | 3 |
| Voraussetzungen: | Das Programm «Abschreibungsrechner» ist gestartet und zeigt Das Fenster « Abschreibungsrechner » an. |
| Eingabe: |  |
| Ausgabe: | 1. Unter Abschreibungsmethoden wird «direkt» und «indirekt» aufgelistet. |

|  |  |
| --- | --- |
| Testfallnummer: | 4 |
| Anforderung: | 4 |
| Voraussetzungen: | Das Programm «Abschreibungsrechner» ist gestartet und zeigt Das Fenster « Abschreibungsrechner » an. |
| Eingabe: | 1. Bei Anschaffungswert «20000» eingeben 2. Bei Nutzungsdauer «5» eingeben 3. Bei Abschreibungsart «linear» auswählen 4. Bei Abschreibungsmethode «direkt» auswählen 5. Bei Restwert «0» eingeben 6. Auf den Button «berechnen» klicken |
| Ausgabe: | 1. Bei Ausgabe wird “Jahr1:Abschreibungen:4000” angezeigt |

|  |  |
| --- | --- |
| Testfallnummer: | 5 |
| Anforderung: | 5 |
| Voraussetzungen: | Das Programm «Abschreibungsrechner» ist gestartet und zeigt Das Fenster « Abschreibungsrechner » an. |
| Eingabe: | 1. Bei Anschaffungswert «20000» eingeben 2. Bei Nutzungsdauer «2» eingeben 3. Bei Abschreibungsart «degressiv» auswählen 4. Bei Abschreibungsmethode «direkt» auswählen 5. Bei Abschreibung in Prozent «20» eingeben 6. Auf den Button «berechnen» klicken |
| Ausgabe: | 1. Bei Ausgabe wird «Jahr1: Anlagevermögen 16000» angezeigt |

|  |  |
| --- | --- |
| Testfallnummer: | 6 |
| Anforderung: | 6 |
| Voraussetzungen: | Das Programm «Abschreibungsrechner» ist gestartet und zeigt Das Fenster « Abschreibungsrechner » an. |
| Eingabe: | 1. Bei Anschaffungswert «20000» eingeben 2. Bei Nutzungsdauer «5» eingeben 3. Bei Abschreibungsart «linear» auswählen 4. Bei Abschreibungsmethode «indirekt» auswählen 5. Bei Restwert «0» eingeben 6. Auf den Button «berechnen» klicken 7. Auf “5” Klicken |
| Ausgabe: | 1. Bei WB Anlagevermögen ist «20000» eingetragen. |

|  |  |
| --- | --- |
| Testfallnummer: | 7 |
| Anforderung: | 7 |
| Voraussetzungen: | Das Programm «Abschreibungsrechner» ist gestartet und zeigt Das Fenster « Abschreibungsrechner » an. |
| Eingabe: | 1. Bei Anschaffungswert «20000» eingeben 2. Bei Nutzungsdauer «5» eingeben 3. Bei Abschreibungsart «linear» auswählen 4. Bei Abschreibungsmethode «direkt» auswählen 5. Bei Restwert «0» eingeben 6. Auf den Button «berechnen» klicken |
| Ausgabe: | 1. Bei Abschreibungen steht «4000» |

|  |  |
| --- | --- |
| Testfallnummer: | 8 |
| Anforderung: | 8 |
| Voraussetzungen: | Das Programm «Abschreibungsrechner» ist gestartet und zeigt Das Fenster « Abschreibungsrechner » an. |
| Eingabe: | 1. Bei Anschaffungswert «20000» eingeben 2. Bei Nutzungsdauer «5» eingeben 3. Bei Abschreibungsart «linear» auswählen 4. Bei Abschreibungsmethode «direkt» auswählen 5. Bei Restwert «0» eingeben 6. Auf den Button «berechnen» klicken |
| Ausgabe: | 1. Es werden die Konten angezeigt, mit den Beschriftungen: “Anlagevermögen”, “WB Anlagevermögen”, “Abschreibungen” |

|  |  |
| --- | --- |
| Testfallnummer: | 9 |
| Anforderung: | 9 |
| Voraussetzungen: | Das Programm «Abschreibungsrechner» ist gestartet und zeigt Das Fenster « Abschreibungsrechner » an. |
| Eingabe: | 1. Bei Anschaffungswert «ABC» eingeben 2. Bei Nutzungsdauer «5» eingeben 3. Bei Abschreibungsart «linear» auswählen 4. Bei Abschreibungsmethode «direkt» auswählen 5. Bei Restwert «0» eingeben 6. Auf den Button «berechnen» klicken |
| Ausgabe: | 1. Im gleichen Fenster unten steht die Meldung «Es wurden nicht alle \*Felder ausgefüllt oder Buchstaben anstatt Zahlen>0 eingegeben». |

|  |  |
| --- | --- |
| Testfallnummer: | 10 |
| Anforderung: | 10 |
| Voraussetzungen: | Das Programm «Abschreibungsrechner» ist gestartet und zeigt Das Fenster « Abschreibungsrechner » an. |
| Eingabe: | 1. Bei Anschaffungswert «-20000» eingeben 2. Bei Nutzungsdauer «5» eingeben 3. Bei Abschreibungsart «linear» auswählen 4. Bei Abschreibungsmethode «direkt» auswählen 5. Bei Restwert «0» eingeben 6. Auf den Button «berechnen» klicken |
| Ausgabe: | 1. Im gleichen Fenster unten steht die Meldung «Es wurden nicht alle \*Felder ausgefüllt oder Buchstaben anstatt Zahlen>0 eingegeben |

|  |  |
| --- | --- |
| Testfallnummer: | 11 |
| Anforderung: | 11 |
| Voraussetzungen: | Das Programm «Abschreibungsrechner» ist gestartet und zeigt Das Fenster « Abschreibungsrechner » an. |
| Eingabe: | 1. Unter «Abschreibungsmethode» den Radiobutton «degressive» anklicken |
| Ausgabe: | 1. Die Beschriftung des letzten Eingabefelds wechselt von «Restwert der Anlage \*» auf «Abschreibung in Prozent \*». |

### 2.2.3 Entscheiden

#### 2.2.3.1 Entscheidungsmatrix

Unsere Entscheidungen haben wir mit der Hilfe von der Entscheidungsmatrix getroffen. Sie hilft Objektive Entschlüsse zu treffen und zeigt auch auf, nach welchen Kriterien gewertet wurde. Damit können auch Dritte diese Entscheidungen analysieren und nachvollzeihen.

MVVM vs MVC mit Singleton als Designpattern

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **MVVM-Pattern** | Gewichtung | Punkte | | | Total |
| Kriterium |  | 3 | 2 | 1 |  |
| Verständlichkeit | 2 |  | x |  | 4 |
| Professionalität | 2 | x |  |  | 6 |
| Aufwand  (weniger = besser) | -1 | x |  |  | -3 |
| Komplexität  (weniger = besser) | -2 |  | x |  | -4 |
|  |  |  |  |  | **3** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **MVC mit Singleton-Pattern** | Gewichtung | Punkte | | | Total |
| Kriterium |  | 3 | 2 | 1 |  |
| Verständlichkeit | 2 | x |  |  | 6 |
| Professionalität | 2 |  | x |  | 4 |
| Aufwand  (weniger = besser) | -1 |  |  | x | -1 |
| Komplexität  (weniger = besser) | -2 |  |  | x | -2 |
|  |  |  |  |  | **7** |

Erläuterung

Dies war unsere schwerste Entscheidung, denn wir mussten abschätzen, wie viel Aufwand es uns kostet und auch, wie verständlich es ist. Das MVVM-Pattern ist auf jeden Fall professionell und wird in vielen JavaFX Anwendungen integriert. Der Nachteil ist, dass es auf den ersten Blick sehr kompliziert aussieht und nicht sofort ersichtlich ist war genau passiert (z.B. Propertychange). Das Hauptkriterium war der Aufwand, weil wir für die Durchführung des Projekts nicht viel Zeit hatten. Aus diesen Gründen wählten wir das MVC-Pattern, in dem wir das Model mit Singleton modifizierten, um die Programmierung zu erleichtern und trotzdem garantieren können, dass es nur ein Model gibt.

ListView vs Pagination bei der Ausgabe

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ListView** | Gewichtung | Punkte | | | Total |
| Kriterium |  | 3 | 2 | 1 |  |
| Übersichtlichkeit | 3 |  |  | x | 3 |
| Professionalität | 2 |  |  | x | 2 |
| Aufwand  (weniger = besser) | -1 |  |  | x | -1 |
| Komplexität  (weniger = besser) | -2 |  |  | x | -2 |
|  |  |  |  |  | **2** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Pagination** | Gewichtung | Punkte | | | Total |
| Kriterium |  | 3 | 2 | 1 |  |
| Übersichtlichkeit | 3 |  | x |  | 6 |
| Professionalität | 2 | x |  |  | 6 |
| Aufwand  (weniger = besser) | -1 |  | x |  | -2 |
| Komplexität  (weniger = besser) | -2 |  | x |  | -4 |
|  |  |  |  |  | **6** |

Erläuterung

Wir haben uns für die Pagination entschieden, weil sich der Benutzer dann die Einzelnen Jahre genauer analysieren kann durch eine bessere Übersicht, bei vielen Einträgen. Die ListView hingegen ist zwar sehr einfach in der Umsetzung und kann sofort mittel ObservableList aktualisiert werden. Leider hat sie den Nachteil, dass sie bei vielen Einträgen schnell sehr unübersichtlich wird. Ebenfalls ein Vorteil der Pagination ist, dass der Benutzer über die Buttons auf der unteren Seite sich das gewünschte Jahr einfach und schnell anzeigen lassen kann. Das müsste er bei einer ListView zuerst selbst suchen.

### 2.2.4 Realisieren

**Woche 1**

In dieser Woche habe wir mit der Informationsphase begonnen. Hauptsächlich haben wir die Vorlagen für das Arbeitsjournal und die Dokumentation erstellt, damit wir diese nur noch ausfüllen müssen und uns keine Gedanken mehr über Aufbau machen müssen. Ebenso wurde ein Repository auf GitHub gemacht, um gemeinsam an einem Projekt arbeiten zu können. Zudem hat jeder von uns für sich nach Bedarf, das Kapitel Abschreibungen repetiert.

**Woche 2**

Die zweite Woche haben wir uns für die Planung reserviert. Wir haben das Klassendiagramm für unser Programm erstellt und haben uns Gedanken über das Design-Pattern gemacht, welches wir in unserem Programm verwenden werden. Dazu wurde auch das Testkonzept erstellt, wie auch das GUI-Design. Letztendlich machten wir uns auch schon Gedanken, wie die Algorithmen für die Berechnungen sein könnten und auch wie die Ausgabe dem Benutzer präsentiert werden könnten. Dazu haben wir uns die Pagination von JavaFX näher angeschaut.

**Woche 3**

In dieser Woche waren wir mit der Programmierung beschäftigt. Wir haben zusammen das GUI implementiert. Wir haben zwei separate Fenster gemacht: eines für die Eingaben des Benutzers und ein anderes für die Ausgaben der Resultate. Für jedes einzelne Jahr, in dem die Anlage angeschrieben wurde, haben wir eine separate Seite in einer Pagination gemacht. Diese passt sich dynamisch der Anzahl Jahre an. Der Benutzer kann dadurch genau sehen, in welchem Jahr die Anlage noch welchen Wert hat und wieviel abgeschrieben wurde. Auch an der Dokumentation haben wir viel weitergearbeitet und gute Fortschritte erzielt.

**Woche 4**

In der Woche vier mussten wir noch einige Anpassungen am Code machen, weil es noch einen Fehler beim Abfangen der Fehler gab. Diesen konnten wir in wenigen Minuten beheben. Wir haben uns auch noch ein wenig mit dem styling von unserem GUI beschäftigt und den Hintergrund etwas ansprechender gestaltet. Es soll ja auch nicht abschreckend erscheinen. Durch die Änderungen am Code mussten wir auch das Klassendiagramm dementsprechend aktualisieren. Zum Test der Berechnungen implementierten wir Unittests, welche die vier verschiedenen Möglichkeiten der Abschreibung abdecken. Wie in den Vorherigen Wochen gewann die Dokumentation weiter an Inhalt und neigte sich dem Ende zu. Zum Schluss mussten wir noch einige Sachen an unserer Dokumentation ändern.

#### 2.2.4.1 Codeausschnitte

**Funktionalität**

Es gibt 2 *Backend* Klassen, die beide ihren eigenen Zweck haben. Wir haben uns grundsätzlich für Double und nicht für BigDecimal , da zum Beispiel eine Anlage von 9.99999999999999999999 CHF nicht wirklich wahrscheinlich ist.

**Konto Klasse**

Die Konto Klasse, dient des Speicherns und Abrufen von Kontoständen.

Die Kernfunktion dieser Klasse heisst add():

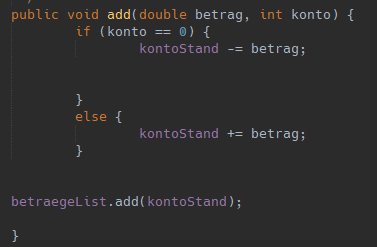


Abbildung 9 "add" Methode

Hierbei ist vor allem der Parameter “konto” zu beachten. Das Problem bei den Konten ist, dass zum Beispiel auf das WB Anlagevermögen immer Beträge hinzugefügt werden aber das Konto Anlagevermögen sowohl Addition aber auch Subtraktion machen kann.

**Die Klasse Calculations:**

Hier geschieht die ganze Magie. Alle Abschreibungen werden hier berechnet und auf die Konten gebucht.

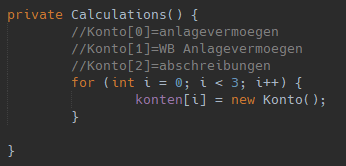


Abbildung 10 Calculations Konstruktor

Am Anfang werden alle 3 Konten erstellt.

Die Eingabe, ob direkt oder indirekt gebucht werden soll, wird mit einer Variablen an diese Klasse übergeben. 0 steht für direkt, 1 für indirekt.

**Singleton**

Damit das Eingabefenster die Daten in eine Klasse geben kann und das Ausgabefenster, Daten aus derselben Klasse verwenden kann, wurde der *Singleton Pattern* gebraucht und auf die “Calculations” Klasse angewendet:

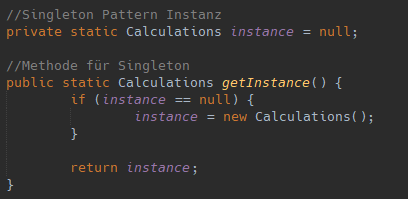


Abbildung 11 Singleton Pattern

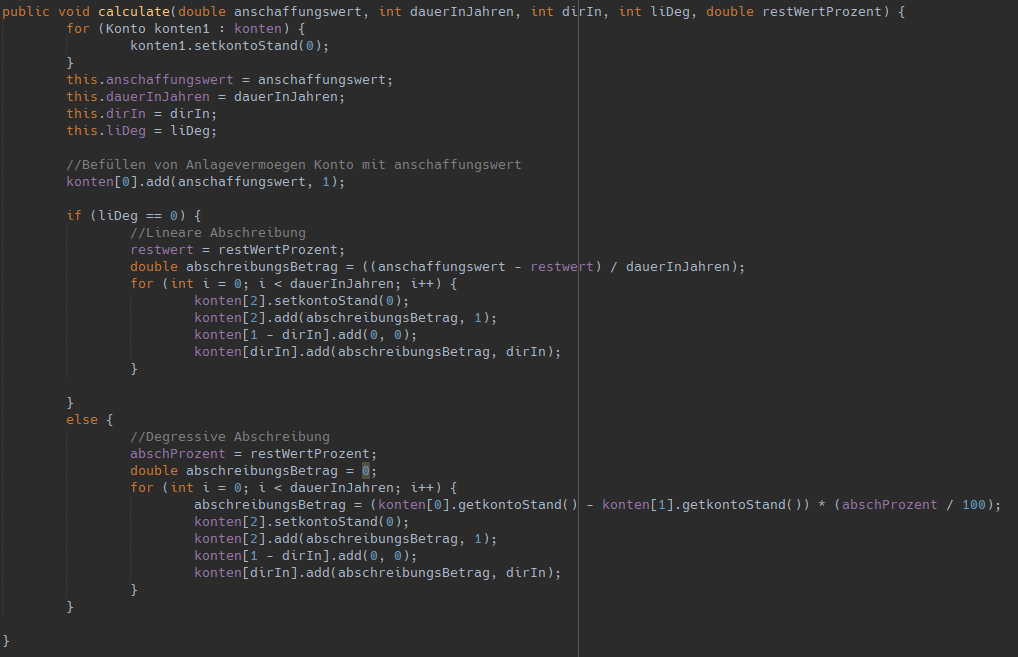


Abbildung 12 Calculate Methode

**Nun eigentliche Methode.**

Zuerst wird der Kontostand von jedem Konto auf 0 gesetzt, da sonst die Beträge von vorherigen Berechnungen in den Weg kommen könnten.

Nun wird der Anschaffungswert in das Konto Anlagevermögen gebucht. (Durch “1” als Parameter wird es addiert).

**Linear**

Sollte der Benutzer nun linear buchen wollen (Auswahl linear/degressiv über Variable liDeg 0=linear 1=degressiv), wird erst einmal der Betrag berechnet, welcher jedes Jahr abgeschrieben wird. Nun iteriert eine Schleife, so viele Male, wie der Benutzer möchte (Über wie lange abgeschrieben werden soll).

Jedes Jahr wird der Kontostand des Abschreibungskontos auf 0 gesetzt, da dieses ein Erfolgskonto ist und Erfolgskonten keinen Anfangsbestand besitzen. Nun wird der Abschreibungsbetrag auf das Konto Abschreibungen gebucht.

Nun hat man noch 2 Konten. Bei der direkten Abschreibung wird nur auf das Anlagevermögen Konto gebucht, bei der indirekten nur auf das WB Anlagevermögen Konto. Das jeweils nicht betroffene Konto, bekommt deswegen in diesem Jahr einen Bestand von 0.(Das Konto kann mit der Formel 1-dirIn angesprochen werden, da 1-0=1 und 1-1=0, so wird immer das andere Konto angesprochen.)

Auf das andere Konto, wird nun der Abschreibungsbetrag entweder addiert, oder subtrahiert.

**Degressiv**

Die degressive Abschreibung unterscheidet sich in nur einem Punkt, von der linearen Abschreibung. Deswegen wird hier auch nur darauf eingegangen.

Der degressive Abschreibungsbetrag berechnet sich folgendermassen:

(Prozentsatz\*Momentaner Wert des Anlagevermögens) /100

Heisst, wenn ich 100chf zu 10% degressiv abschreibe, ist der Betrag:

Jahr 1: (10\*100 )/100=10

Jahr 2:(10\*90 )/100=9

Da das Objekt, nach einem Jahr schon an Wert verloren hat.

In Code sieht das folgendermassen aus:

**abschreibungsBetrag = (konten[0].getkontoStand() - konten[1].getkontoStand()) \* (abschProzent / 100);**

Hier wird der aktuelle Wert des Objekts (Anlagevermögen-WB Anlagevermögen) berechnet und dann passiert das gleiche wie oben erwähnt.

So ergibt sich am Ende in jedem Konto eine Liste, mit den Beträgen des Kontos aus jedem Jahr.

Diese können nun ausgegeben werden.

**Befüllen der Pagination**

Abbildung 13 Befüllen der Pagination

Die Daten werden nun mit einer sogenannten Factory in die Pagination geladen.

Hierbei werden 4 Labels für jedes Jahr erstellt:

Label1: Hier wird das aktuelle Jahr angezeigt

Label2: Hier wird der Betrag des Anlagevermögens gezeigt

Label3: Hier wird der Betrag des WB Anlagevermögen Kontos gezeigt

Label4: Hier wird der Betrag des Abschreibungskontos gezeigt.

### 2.2.5 Kontrollieren

#### 2.2.5.1 Testprotokoll

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Testfallnummer | Name des Testers | Datum des Testes | Resultat | Bemerkungen |
| 1 | Noah Grand | 14.09.2020 | OK | KEINE |
| 2 | Noah Grand | 14.09.2020 | OK | KEINE |
| 3 | Noah Grand | 14.09.2020 | OK | KEINE |
| 4 | Noah Grand | 14.09.2020 | OK | KEINE |
| 5 | Noah Grand | 14.09.2020 | OK | KEINE |
| 6 | Noah Grand | 14.09.2020 | OK | KEINE |
| 7 | Noah Grand | 14.09.2020 | OK | KEINE |
| 8 | Noah Grand | 14.09.2020 | OK | KEINE |
| 9 | Noah Grand | 14.09.2020 | OK | KEINE |
| 10 | Noah Grand | 14.09.2020 | OK | KEINE |
| 11 | Benjamin Yildirim | 16.09.20 | OK | KEINE |

**Unittests**

Um die Funktionalitäten, des Programmes nicht nur von Menschen testen zu lassen, sondern auch jedes Mal automatisch, wenn das Programm aufstartet, wurden *Unittests* geschrieben, für die Klasse Calculations.

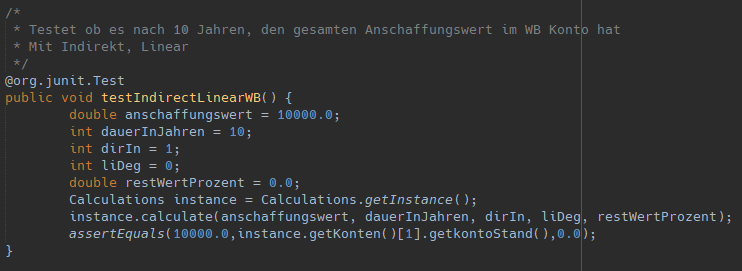


Abbildung 14 Unittest

Die Funktionalität, jedes Tests ist gleich und gleicht auch dem auf der Abbildung.

Es wird eine Berechnung durchgeführt, und am Ende wird er Betrag in einem Konto geprüft, ob er wie erwartet ist.

Insgesamt wurden 5 Unittests erstellt, für folgende Funktionalitäten:

1. Testet ob es nach 10 Jahren, den gesamten Anschaffungswert im WB Konto hat

\* Mit Indirekt, Linear

2. Testet ob das Konto Anlagevermögen nach 10 Jahren immer noch voll ist

\*Mit Indirekt, Linear

3. Testet ob das Konto Anlagevermögen nach 10 Jahren leer ist

\*Mit direkt, linear

4. Testet ob das Konto Abschreibungen nach dem 1. Jahr den korrekten Betrag hat

\*Mit direkt, degressiv

5. Testet ob das Konto Abschreibungen nach dem 1. Jahr den korrekten Abschreibungsbetrag hat

\*Mit direkt, linear

Alle Tests, laufen ohne Probleme:

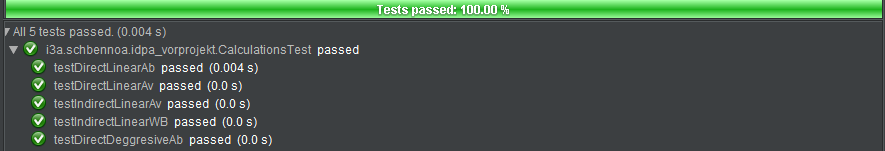


Abbildung 15 Unittest Resultate

## 2.2.6 Auswerten

#### 2.2.6.1 Testbericht

Die Tests verliefen reibungslos, es sind keine Probleme aufgetreten. Das Programm wird ohne Fehler ausgeführt und kann nun freigegeben werden.

#### 2.2.6.2 Probleme

Beim Testen sind keine Probleme aufgetreten.

#### 2.2.6.3 Persönliches Fazit

**Noah**

Zum ersten Mal konnte ich erleben, wie es ist mit einem Team zu arbeiten, wo jeder gleichviel macht. Wenn jemand programmiert, waren die anderen immer an der Dokumentation. Dadurch, dass wir uns am Anfang gut organisiert haben, die Aufgaben verteilt haben und jeder wusste, was er wann zu tun hatte, verlief dieses Projekt fast reibungslos. Wenn jemand nicht weiterwusste, konnte er immer Antworten im Team finden. In den letzten paaren Wochen wurde die Arbeit auch immer routinierter und ich denke, dass wir eine gute Basis gelegt haben, für das Hauptprojekt.

**Shenia**

Ich habe unsere Arbeit in diesem Team wirklich genossen, weil wir die Arbeit wirklich gut aufteilen konnten, so dass jeder etwas zu tun hat. Was mir auch gefällt, ist, dass immer dann, wenn einer von uns an etwas festhängt oder Fragen hat, er/sie die anderen um Hilfe bitten kann. Wir würden uns immer gegenseitig helfen und niemanden zurücklassen. Wir haben alles so gut geplant, so dass wir am Ende nicht so sehr in Panik verfallen müssen. Ich freue mich darauf, für das Hauptprojekt wieder mit dieser Gruppe zusammenzuarbeiten.

**Benjamin**

Ich habe die Arbeit in unserem Team sehr genossen, weil wir einen guten Umgang miteinander hatten. Die Durchführung des Projekts ging reibungslos vonstatten, weil wir uns bei unserer Planung genügend, aber auch nicht zu viel Zeit genommen hatten. Genau richtig. Dies stützte unsere Arbeit, weil wir so schon voraussehbare Stolpersteine aus dem Weg räumen konnten. Mir war wohl mit der Dokumentation. Dies aus dem Grund, dass wir laufend dokumentierten und nötigenfalls anpassten. Das möchte ich auf jeden Fall weiterhin beibehalten und keine Arbeit aufschieben. Ich freue mich schon mit dieser Gruppe das Hauptprojekt durchzuführen.

#### 2.3 Abbildungsverzeichnis

[Abbildung 1 Projektorganisation 3](#_Toc51157071)

[Abbildung 2 Beispiel Ausführen des Programmes 5](#_Toc51157072)

[Abbildung 3 Use case diagramm 9](https://kantonsschulebaden-my.sharepoint.com/personal/benjamin_yildirim_students_ksba_ch/Documents/Microsoft%20Teams-Chatdateien/DOKUMENTATION.docx#_Toc51157073)

[Abbildung 4 Zustandsdiagramm 10](https://kantonsschulebaden-my.sharepoint.com/personal/benjamin_yildirim_students_ksba_ch/Documents/Microsoft%20Teams-Chatdateien/DOKUMENTATION.docx#_Toc51157074)

[Abbildung 5 GUI Design 11](#_Toc51157075)

[Abbildung 6 Klassendiagramm 12](https://kantonsschulebaden-my.sharepoint.com/personal/benjamin_yildirim_students_ksba_ch/Documents/Microsoft%20Teams-Chatdateien/DOKUMENTATION.docx#_Toc51157076)

[Abbildung 7 Systemdarstellung 13](#_Toc51157077)

[Abbildung 8 Datenmodell 13](#_Toc51157078)

[Abbildung 9 "add" Methode 20](#_Toc51157079)

[Abbildung 10 Calculations Konstruktor 21](#_Toc51157080)

[Abbildung 11 Singleton Pattern 21](#_Toc51157081)

[Abbildung 12 Calculate Methode 22](#_Toc51157082)

[Abbildung 13 Befüllen der Pagination 23](#_Toc51157083)

[Abbildung 14 Unittest 24](#_Toc51157084)

[Abbildung 15 Unittest Resultate 25](#_Toc51157085)

#### 2.4 Quellenverzeichnis

**Shenia (Quellen aktuell per 16.09.2020)**

Scene Builder:

<https://docs.oracle.com/javafx/scenebuilder/1/user_guide/inspector-panel.htm>

Querformat:

https://support.microsoft.com/de-de/office/video-verwenden-von-querformat-und-hochformat-im-selben-dokument-ddd80cb6-c9ae-4493-ba75-c663074031a0

**Benjamin**

Keine Quellen

**Noah (Quellen aktuell per 16.09.2020)**

Java Input lesen:

<https://data-flair.training/blogs/read-java-console-input/>

JavaFX Pagination :

<http://tutorials.jenkov.com/javafx/pagination.html>

Java Maven Projekt Java Version Ändern

<https://stackoverflow.com/questions/32923586/maven-lambda-expressions-are-not-supported-in-source-1-5>

JavaFX neues Fenster öffnen

<https://stackoverflow.com/questions/15041760/javafx-open-new-window>

JavaFX Fenster nicht verschiebbar Bug

<https://stackoverflow.com/questions/20732100/javafx-why-does-stage-setresizablefalse-cause-additional-margins>

Javadoc Tags

<https://idratherbewriting.com/java-javadoc-tags/>

#### 2.5 Glossar

|  |  |
| --- | --- |
| Begriff | *Erklärung* |
| A |  |
|  |  |
| B  Backend |  |
| Der Teil eines Programmes, der für die Berechnungen zuständig ist und für den Benutzer nicht sichtbar ist. |
|  |  |
| C |  |
| CSS | Cascading Style Sheets für das Styling des GUIs |
|  |  |
| D |  |
|  |  |
| E |  |
|  |  |
| F |  |
|  |  |
| G |  |
| GUI | Graphical User Interface (Benutzerschnittstellt) |
|  |  |
| H |  |
|  |  |
| I |  |
|  |  |
| J |  |
| Javadoc | Tool für Java Codedokumentation |
|  |  |
| K  Konstruktor |  |
| Die Methode einer Klasse, die beim Erstellen eines Objektes aufgerufen wird. |
|  |  |
| L |  |
| ListView | JavaFX GUI-Element |
|  |  |
| M |  |
| MVVM | Softwaredesignpattern |
| MVC | Softwaredesignpattern |
| Model | Bestandteil von MVC zur Datenhaltung |
| Maven | Buildtool für Java Programme |
|  |  |
| N |  |
|  |  |
| O |  |
| ObservableList | Spezielle FXML Collection, welche zum speichern von Werten in einer ListView genutzt werden kann. |
|  |  |
| P |  |
| Pagination | JavaFX GUI-Element |
|  |  |
| Q |  |
|  |  |
| R |  |
|  |  |
| S |  |
| Singleton | Softwaredesignpattern |
|  |  |
| T |  |
|  |  |
| U |  |
| Unittest | Programmierte, automatische Tests für implementierte Lösungen |
|  |  |
| V |  |
|  |  |
| W |  |
|  |  |
| X |  |
|  |  |
| Y |  |
|  |  |
| Z |  |