



Projektteilnehmer

Mathias Grab

Gruppen-Nr. 2312

Projektbetreuer

Hr. Florian Kast

Projektzeitraum:

11.01.2022 - 11.01.2022





Inhaltsverzeichnis

Inhaltsverzeichnis.....	03	Jäger.....	30
Was ist RPG IT?.....	06	Aufbau GUI.....	30
Planung.....	06	Erstellung Mock Up.....	30
Ist Zustandsbeschreibung.....	06	Einarbeitung in Frameworks.....	32
Vorhergehende Erfahrung und Kenntnisstand.....	06	Einarbeitung in JavaFX.....	32
Arbeits- und Projektumgebung.....	06	Erstellung Dummy JavaFX.....	36
Stundenverteilung.....	06	Einarbeitung in Java Datenbank Framework h2.....	36
Core Features.....	08	Auswahlkriterien von h2.....	38
Grundsätzliches Levelingsystem.....	08	Erstellung Dummy Datenbank.....	38
Aufbau der Quests.....	14	Implementierung von Datenbank in JavaFX Dummy.....	38
Gehilfensystem.....	14	Prävention von SQL Injection.....	38
Antagonisten System.....	16	Beginn der Programmierung der Applikation.....	40
Item System.....	18	Gestaltung Fertiges GUI in JavaFX.....	40
Roadmap.....	20	Namenskonvention für JavaFX ID.....	40
Soll und Ist Vergleich.....	20	Probleme beim Erstellen der GUI.....	40
Identity Design.....	22	Erstellen der Datenbank und einpflegen der Daten.....	40
Art Style.....	22	Implementierung des JavaFX Frameworks.....	42
Farbwahl.....	22	Problematik mit Modulen.....	42
Typografie.....	24	Neuausrichtung des Projekts.....	42
Datenbank.....	26	Einarbeitung SWING.....	42
Entscheidungsfindung zur Speichermethoden.....	26	Programmierung.....	44
Datenbankgestaltung.....	28	Methode Erstellung GUI.....	44
		Methode Erstellung GridBagLayout.....	44
		Erstellung GUI mit IntelliJ Swing UI Designer.....	44
Grafiken.....	30	Implementierung Datenbank.....	46
Erstellung Charaktere.....	30	Grundlegender Aufbau.....	48
Paladin.....	30	Grundlegender Methoden Aufbau bei Queries.....	48
Druide.....	30	Fetchen der Datenbankinformationen bei Applikationsstart.....	48
Barde.....	30		
Zauberer.....	30		

Funktionsweise Inventar.....	50
Funktionsweise Charakter Attribute.....	50
Funktionsweise Progress Bar.....	50
Vorzeitiges Beenden des Projektes aufgrund von Zeitmangel.....	52
Gestrichene und nicht hinzugefügte Applikationsfeatures.....	52
Projektanalyse.....	53
Vergleich Stundenverteilung.....	53
Problemfelder.....	55
Gelernte Inhalte.....	57



Was ist RPG IT?

RPG IT ist eine To Do Liste mit der es möglich ist, Erfahrungspunkte durch das Lösen von selbst gestellten Aufgaben zu sammeln, aufzuleveln, bessere Ausrüstung zu bekommen und neue Gefährten freizuschalten.

Die Kombination aus To Do Liste und RPG erhöht die Motivation Aufgaben auszuführen und belohnt den User mit deren Fertigstellung.

Planung

Ist Zustandsbeschreibung

Vorhergehende Erfahrungen und Kenntnisstand

Es liegen rudimentäre Grundlagen in Java wie Datentypen, Objekte und Klassen, Methoden und Parameter, Scopes und Ausgaben innerhalb des Terminals vor, welche als Grundlagen im Unterrichtsfach Java während der Umschulung an der SRH gelehrt wurden.

Aus privatem Sektor geht eine rudimentäre Einarbeitung in Programmierlogik, HTML, CSS, Java Script, Python, SQL und PHP, Java Swing, sowie diverse Frameworks hervor.

Des weiteren geht ebenfalls Gestaltungsgrundlagen aus vorheriger Ausbildung zum Mediengestalter für Digital- und Printmedien hervor, sowie der gestalterischen digitalen Arbeit in UI und UX Design, künstlerisch kreative Arbeiten in digitalem Zeichnen und Malen und Videobearbeitung hervor.

Arbeits- und Projektumgebung

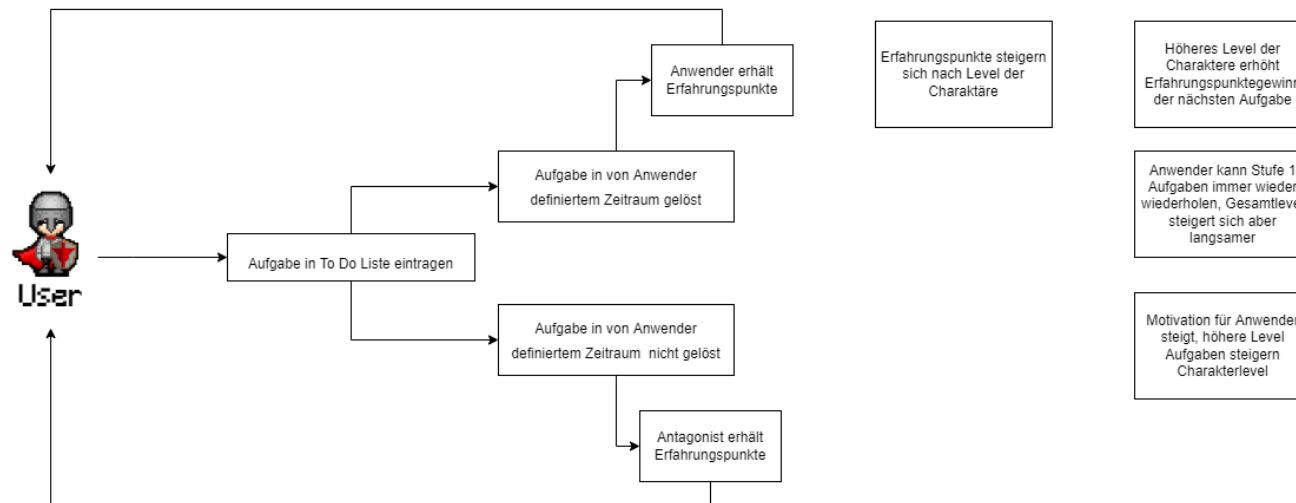
Windows 10 64bit
Breitband Internetanschluss
IntelliJ Community 2021.3.5
Microsoft Office
Java SDK/JRE 17.0.2.
Draw.io 16.1.2
SceneBuilder 17.0.0
Pencil 3.1.0
Aseprite 1.2.30
Visual Studio Code 1.64.2

Stundenverteilung

Projektplanung (inkl. Datenbankgestaltung)	15 h
UX / UI Konzeptionierung und Gestaltung	40 h
Recherche zur Konfliktlösung	15 h
Programmierung	90 h
Testung / Bug Fixing	10 h
Dokumentation	50 h
Präsentation	5h

Core Features

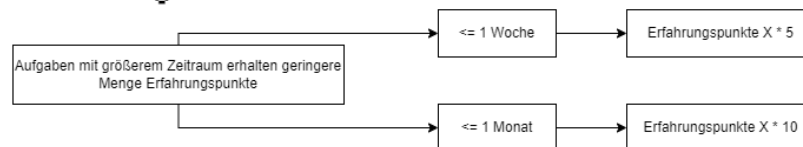
Grundlegendes System



Kleine Aufgaben



Grosse Aufgaben



Core Features

Grundlegendes Levelsystem

Wie bereits erwähnt liegt die Hauptmotivation von RPG IT in der Erfüllung von täglichen Aufgaben in einer To Do Liste und einem resultierenden Erfahrungspunktegewinn.

Nach Eingabe einer Aufgabe in einem zuvor festgelegten Zeitraum können zwei Szenarien eintreffen:

Aufgabe wird gelöst:

User erhält Erfahrungspunkte, steigt im Level auf, erhält in bestimmten Abschnitten neue Ausrüstung und bleibt damit motiviert die Aufgaben weiterzuführen.

Ein höheres Level der Charaktere erhöht dementsprechend den Erfahrungspunktegewinn der Aufgabe im höher gelegenen Level.

Es handelt sich hierbei um ein simples, aus dem Gaming Bereich sehr häufig anzutreffendes System, das System des Grinding Zyklus.

Unter dem Terminus „Grind“ versteht man in der Spieleindustrie das repetitive Lösen von Aufgaben, um damit einem Ziel näher zu kommen (Vernichte 100 Gegner um ein besonderes Item zu erhalten).

Um diesem simplen, repetitiven, meist eher eintönigen Grind einen psychologischen Vorteil zu verschaffen, wird durch den Gewinn von Erfahrungspunkten oder das Erhalten von einzigartigen Items das Belohnungszentrum angesprochen, was den User dazu animieren soll, weiterzuspielen.

Aufgabe wird nicht gelöst:

Sofern eine Aufgabe durch den User nicht gelöst wird, erhält der Antagonist Erfahrungspunkte.

Dieser wird stärker, die Motivation diesen zu besiegen steigt, ebenso die Motivation Aufgaben zu erfüllen.

Problematiken im Level System

RPG IT setzt ein sogenanntes Trust System des Users voraus. Unter Trust System bezeichnet man ein von der Applikation vorausgesetztes Vertrauen an den User voraus, die Aufgaben zu sorgfältig zu lösen.

Dennoch besitzt RPG IT Mechaniken, die es dem User vereinfachen seine Aufgaben zu lösen.

Natürlich kann als Beispiel immer wieder eine Level 1 Aufgabe gelöst werden, in Bezug auf die Erfahrungskurve (dazu später mehr) ist dies dennoch kontraproduktiv, da das Gesamtlevel so langsamer steigt, die Progression sich verzögert und langsamer Motivationspunkte für das Weiterspielen gesetzt werden.

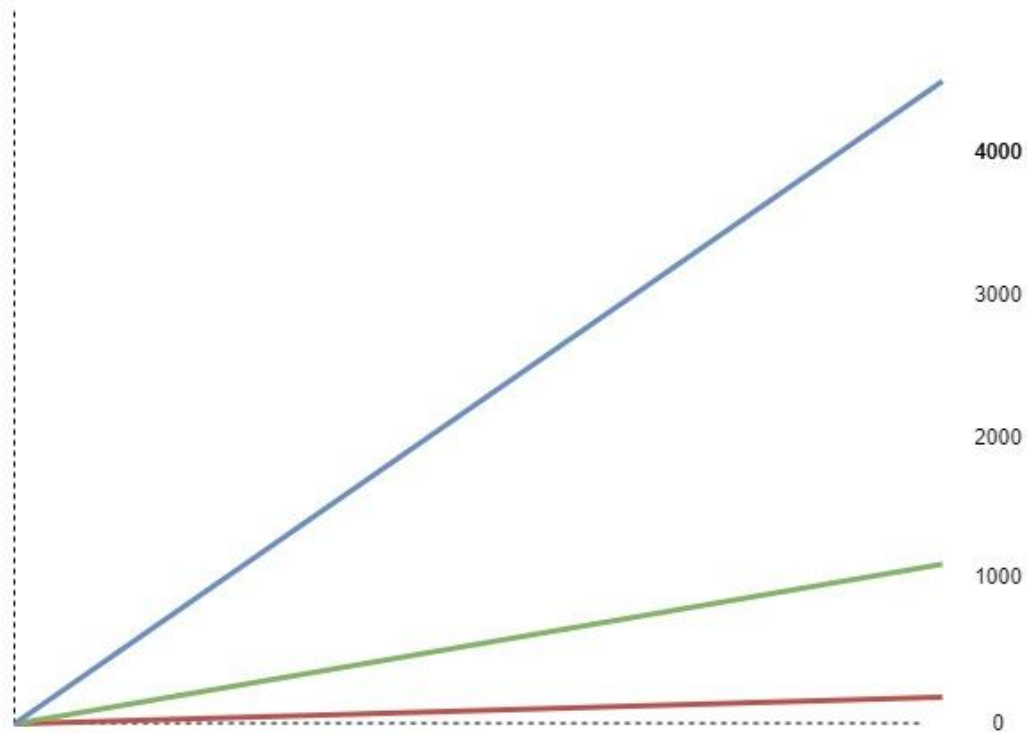
Erfahrungspunktegewinn und Leveling Kurve

Wie schon beschrieben ist ein essenzieller Teil der Aufgabenlösung das Erhalten von Erfahrungspunkten.

Kleinere Aufgaben erhalten einen Wert an Erfahrungspunkten, der diese als Ausgangspunkt für die Berechnung der Erfahrungspunkte der Wochen- und Monatsaufgaben berechnet werden.

Größere Aufgaben sollen mit mehr Erfahrungspunkten belohnt werden.

Lineare Auflistung der Erfahrungspunkte
nach Erfahrungspunkte

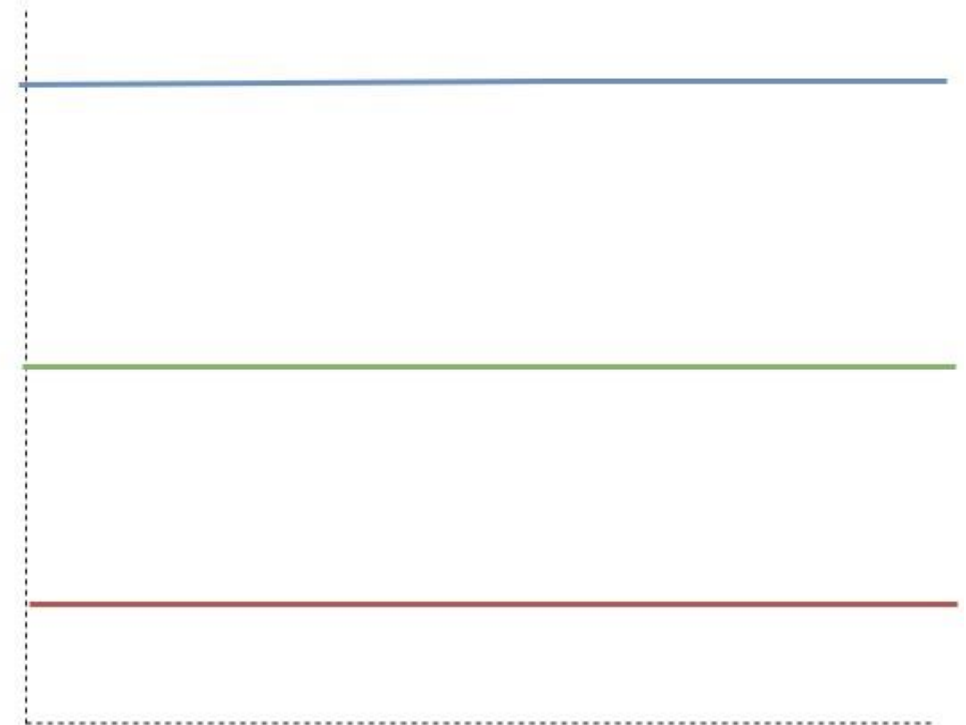


tägliche Quest

wöchentliche Quest

monatliche Quest

Lineare Auflistung der Erfahrungspunkte
nach nach Zeit



tägliche Quest

wöchentliche Quest

monatliche Quest

Grundlegend werden Aufgaben in 3 unterschiedliche Kategorien eingruppiert.

Formel zur Berechnung der Erfahrungspunkte nach Einheit :

Aufgabe gilt innerhalb **eines Tages** erledigt

$$XP_B = 150$$

XP = Erfahrungspunkte (engl. Experiencepoints) _B = Basic

Aufgabe gilt innerhalb **einer Woche** erledigt

$$XP_W = XP_B \times 7$$

XP = Erfahrungspunkte (engl. Experiencepoints) _W = weekly (wöchentlich)

Aufgabe gilt innerhalb **eins Monats** erledigt

$$XP_M = XP_B \times 30$$

XP = Erfahrungspunkte (engl. Experiencepoints) _M = monthly (monatlich)

Leveling Kurve

Das Erreichen von neuen Leveln ist ein Core Feature von RPG IT, wie schon beschrieben wird damit ein essenzieller Grundstein für die intrinsische Motivation des Users gelegt. Daher ist eine gut ausbalancierte Leveling- und Erfahrungskurve wichtig.

Unterschiede lineare und nonlineare Levelingkurve

Lineare Levelingkurve

Unter einer linearen Levelingkurve versteht man in der Gamingindustrie einen gleichbleibenden prozentualen Anstieg des Gesamtlevels bei Erhalt von Erfahrungspunkten. Es wird immer eine vorher definierte Anzahl an Erfahrungspunkten benötigt, um auf das nächste Level aufzusteigen.

Die Vorteile darin bestehen aus einer gewissen kalkulierbaren Kontinuität, das bedeutet, dass der User vorausplanen kann, wann er welches Level erreicht.

Die Nachteile allerdings lassen sich in einer monotonen Levelingphase und einer daraus resultierenden verringerten Motivation das nächste Level zu erreichen eingliedern, da dies keine Herausforderung für den User darstellt.

Nonlineare Levelingkurve

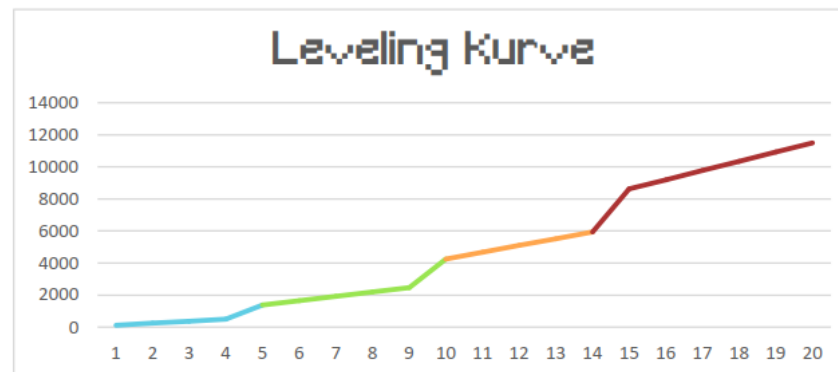
Unter einer nonlinearen Levelingkurve versteht man, dass ab einem gewissen Punkt im Voranschreiten des Gesamtlevelingerlebnisses einen Anstieg an Erfahrungspunkten benötigt wird, um in das nächste Level zu gelangen.

Levelkurve

Erfahrungspunkte bis nächstes Level
Gesamte Erfahrungspunkte

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
125	250	375	500	1375	1650	1925	2200	2475	4250	4675	5100	5525	5950	8625	9200	9775	10350	10925	11500
0	125	500	1000	2375	4025	5950	8150	10625	14875	19550	24650	30175	36125	44750	53950	63725	74075	85000	96500
Level * 100 * 1,25				Level * 100 * 1,25					Level * 100 * 1,25						Level * 100 * 1,25				

Leveling Formel
 $\text{Level} * 100 * 1,25$
 Jede 5 Level + $*100 * + 1,5$



Die Vor- und Nachteile dieser Kurve betrachten sich als die umgedrehten Vor- und Nachteile der linearen Levelingkurve.

Die Motivation für den User steigt, da es eine neue Herausforderung ist die nächste Stufe zu erreichen und es damit für den Gesamtfortschritt schwieriger zu kalkulieren ist.

Die Frage nach der Art der Levelingkurve, ob linear oder nonlinear ist an sich recht schnell beantwortet. Die Entscheidung für eine nonlineare Levelingkurve beantwortet sich in Bezug auf die Motivation und die Langzeitbenutzung der Applikation.

So ist unter einer linearen Levelingkurve das Höchstlevel schnell erreicht, was die Bindung an die Applikation deutlich schmälert.

Für das Erreichen des maximalen Levels (Level 20) gibt es hier 3 Breakpoints, an dem sich die benötigten Erfahrungspunkte vergrößern, um auf das nächste Level zu gelangen.

Diese Breakpoints befinden sich ab Level 5, Level 10 und Level 15.

$$L_{LR} = \text{Level} * 100 * (BP_x * 1,5)$$

L = Level LR = Levelrange (Level innerhalb des Breakpoints)

BP = Breakpoint x = Anzahl des Breakpoints

$$L_{1-5} = \text{Level} * 100 * (1 * 1,5)$$

$$L_{5-10} = \text{Level} * 100 * (2 * 1,5)$$

$$L_{10-15} = \text{Level} * 100 * (3 * 1,5)$$

$$L_{15-20} = \text{Level} * 100 * (4 * 1,5)$$

Heruntergerechnet auf eine einzige Tagesaufgabe der Vergleich innerhalb des jeweils ersten Levels in einem Breakpoint werden folgende Anzahl an Aufgaben benötigt, um auf das nächste Level zu gelangen.

Level 1 zu Level 2:	1 Tagesaufgabe (Quests)
Level 5 zu Level 6:	10 Tagesaufgaben (Quests)
Level 10 zu Level 11:	30 Tagesaufgaben (Quests)
Level 15 zu Level 16:	64 Tagesaufgaben (Quests)

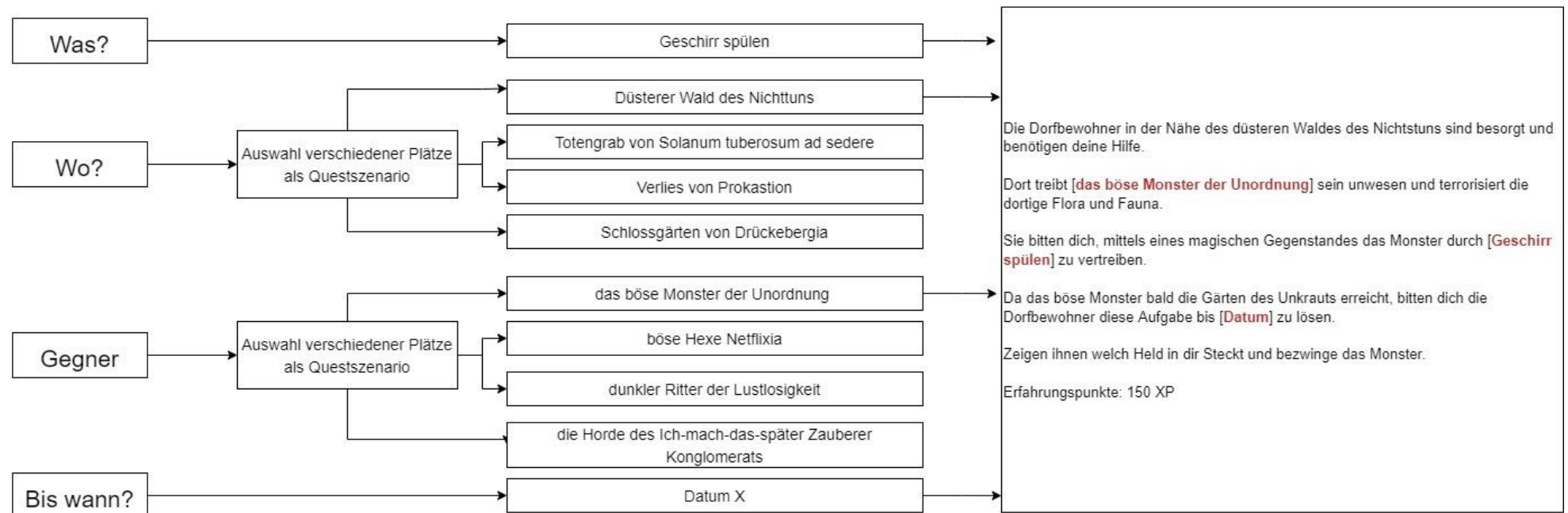
Durch das Hinzufügen von zusätzlichen Gefährten können mehrere Quests (Tagesaufgaben) pro Tag erledigt werden.

Wie an dem Rechenbeispiel zu sehen ist, lassen sich Aufgaben in den ersten Level noch schnell und einfach lösen, im späteren Bereich sind deutlich mehr Tagesaufgaben notwendig, was die Motivation steigern soll.

Bei ansteigendem Level erhält der User und der jeweilige Gehilfe + 1 Punkt auf sein Hauptattribut (mehr dazu unter Gehilfen).

Quest System

individuelle Questtexte nach Auswahl des Gebiets und des Enddatums



Aufbau des Quest Systems

Der Aufbau der Quest und die dazugehörige Questbeschreibung wird individuell mit vorgefertigten Presets an die Aufgabe des Users angepasst.

Dabei stehen fünf Eingabe- bzw. Auswahlmöglichkeiten dem User zur Verfügung

Titel der Aufgabe

Enthält die Aufgabe an sich

Umgebungsszenario

Der User kann zwischen mehreren Umgebungsszenarien auswählen, in der er/sie selbst die Rahmenbedingungen der Quest festlegen kann.

Je nach benutzen Umgebungsszenario ändert sich die Questbeschreibung.

Auswahlmöglichkeiten:

Düsterer Wald des Nichtstuns

Totengrab des Gottes der Faulheit

Verlies von Prokastion

Schlossgärten von Drückebergia

Art des Gegners

das böse Monster der Unordnung

die böse Hexe Netflixia

der dunkle Ritter der Lustlosigkeit

die Hore des Ich-mach-das-später Zauberer Konglomerats

Datum der Fertigstellung

Hier entscheidet sich die Komplexität der Quest und in welche Erfahrungsstufen Kategorie sie eingeordnet wird und wie hoch der Erfahrungspunkte Output ist.

In Bezug auf modulare Programmierung können die oben genannten Kategorien (außer Datum der Fertigstellung), sofern eine langfristige Laufzeit der Applikation geplant ist, durch weitere Patches hinzugefügt und erweitert werden.

Gehilfensystem

Im Laufe der Benutzung der Applikation und durch Anstieg der Erfahrungspunkte, wird es dem User ermöglicht bis zu insgesamt vier weitere Gefährten dazuzugewinnen.

Diese bieten die Möglichkeit mehrere Quests gleichzeitig auszuführen, was die Motivation des Users steigt mehrere Aufgaben gleichzeitig zu bearbeiten.

Die Gefährten Leveln innerhalb ihrer Attribute oder Kategorien eigenständig und erhalten die selben Erfahrungspunkte.

So lässt sich jeder Gehilfe einzeln leveln, was die Motivation Aufgaben für die bestimmten Kategorien zu erfüllen weiter steigt.

Psychologisches Ziel dabei ist es, den User langsam an mehrere Aufgaben heranzuführen und schließlich zusammen mit seinen Gefährten an seinen Aufgaben zu wachsen.

Gehilfen



Paladin



Druide



Barde



Zauberer



Jäger

Abhängig nach erreichter Erfahrungsstufe erhält der User ab einem bestimmten Level einen neuen Gefährten.

Auch kann der User einen neuen Gefährten in eine bestimmte Kategorie einordnen, welche als Übersicht über ein bestimmtes Thema der einzelnen Aufgaben dient (Haushalt, Sport, Sozial, Lernen, etc.)

Diese Kategorie wird gleichzeitig das Hauptattribut des Charakters (dazu unter Itemsystem mehr).

Druide (Level 2)

Barde (Level 5)

Zauberer (Level 8)

Jäger (Level 12)

Antagonistensystem

Um auch weiter die Motivation des Users zu steigern, befindet sich innerhalb der Applikation ein Antagonist, den es im Verlauf des Levels immer wieder zu besiegen gilt.

Grundsätzlich liegt die Erfahrungsstufe des Antagonisten zu Beginn 10 Stufen über dem Hauptcharakter.

Sollte eine Aufgabe nicht gelöst werden, so erhält der Antagonist die Erfahrungspunkte, was diesen mächtiger und schwerer zu besiegen macht.

Ab zwei bestimmten Punkten innerhalb der Applikation erhält der User die Möglichkeit gegen den Antagonisten zu kämpfen und erhält bei Sieg über diesen seltene Ausrüstung, die die Statuswerte der jeweiligen Gefährten enorm steigern.

Das Erscheinen des Antagonisten erfolgt bei Erreichen von Level 10 und Level 20.

Ob der Antagonist besiegt wird oder nicht, richtet sich nach dessen Erfahrungsstufe. Sollte der Antagonist eine Stufe über den Gefährten sein (im ersten Beispiel Level 11) ist dieser nicht zu besiegen, was den User dazu motivieren soll, die durch ihn/sie

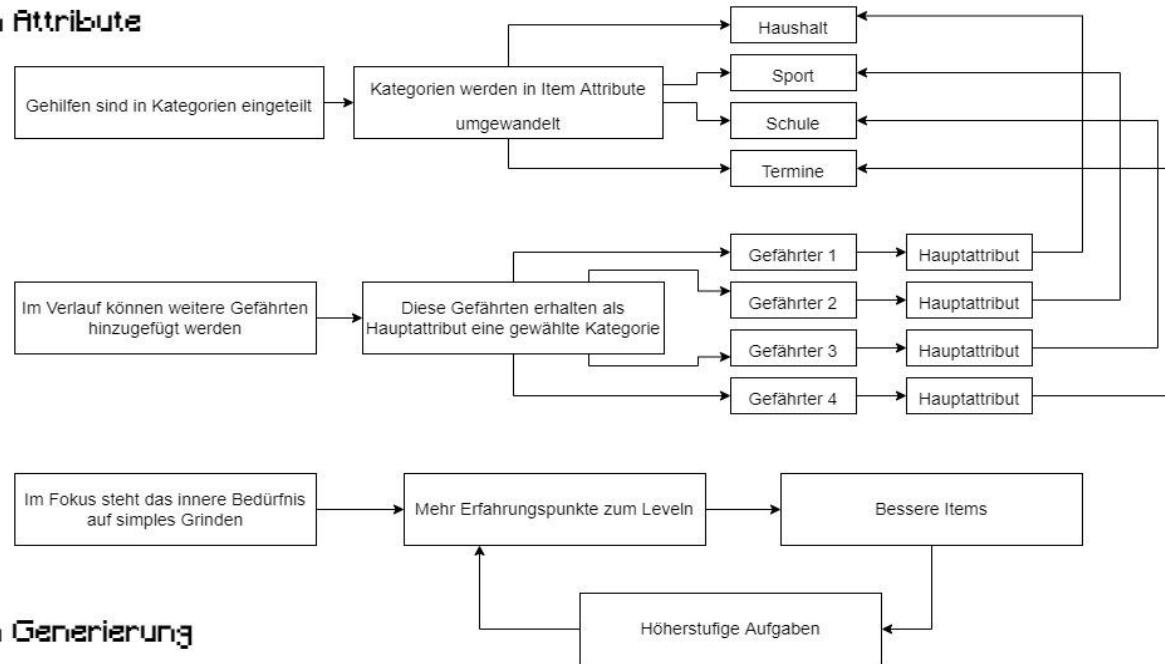
gestellten Aufgaben sorgfältiger zu lösen, da zu viele Aufgaben durch den User nicht erfüllt wurden.

Sollte ein Gefährte das Level für eine seltene Ausrüstung noch nicht erreicht haben, wird diese im Hintergrund gespeichert und bei Erreichen des jeweiligen Level automatisch ausgerüstet.

Core Features

Item System

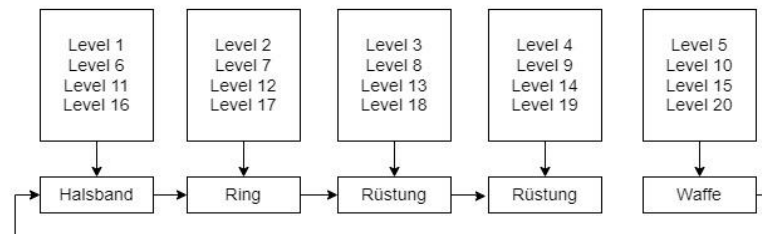
Item Attribute



Item Generierung

Items werden bei Erreichen einer neuen Erfahrungsstufe auf alle Gefährten verteilt

Item Slots



Itemsystem

Als ebenfalls zentrales Belohnungssystem nach erfolgreichem Absolvieren eines Levelaufstiegs, zählt neben dem bereits beschriebenen Level auch ein neu erhaltende Items.

Die Gruppierung der Items lässt sich in Ausrüstung Kategorisieren, dazu gehört Waffe, Helm, Rüstung, Halsband und Ring.

Bei jedem neu erreichten Level erhält die bisher bestehende Gruppe ein neues Item, das nach Priorität und Wichtigkeit des Items unterteilt ist.

Nach insgesamt fünf Leveln sind die Gehilfen also komplett mit neuer Ausrüstung ausgestattet.

Level 1, 6, 11 und 16: Halsband

Level 2, 7, 12 und 17: Ring

Level 3, 8, 13 und 18: Rüstung

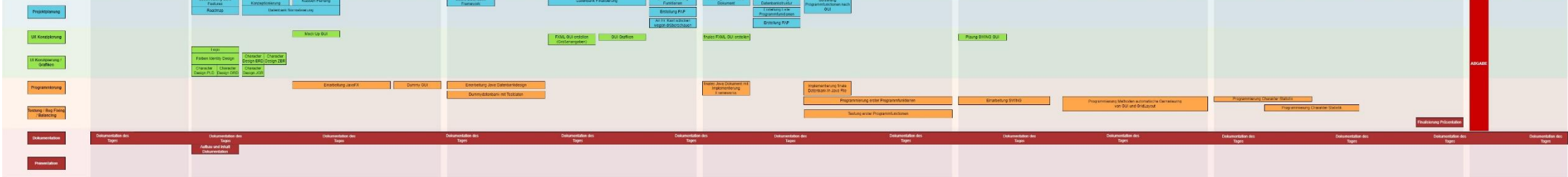
Level 4, 9, 14 und 19: Helm

Level 5, 10, 15 und 20: Waffe

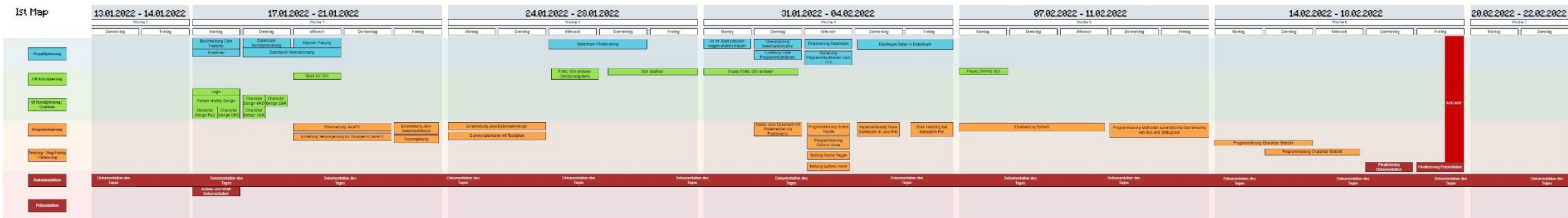
Attribute der Items

Da mit jedem neu erhaltenden Gefährten eine Kategorie der Aufgaben hinzukommt (beispielsweise Haushalt, Sport, Lernen, etc.), wird der Gehilfe nicht nur mit seinem Level stärker, sondern auch mit seinen Attributen durch erhaltene Items.

Roadmap Soll Map



Ist Map



Roadmap

Soll und Ist Vergleich

Die Unterschiede zwischen Soll und Ist Zustand der Roadmap sind im Allgemeinen recht marginal und beschränken sich grundsätzlich auf kleinere Ausdehnungen, die vermehrt Zeiten beanspruchten.

Woche 1:

Zu Soll Roadmap kam lediglich die Erstellung einer Testumgebung für JavaFX hinzu, da paralleles Arbeiten, sowohl in Tutorial, als auch in einem Dummy Sinn macht, um das Gelernte gleich umzusetzen.

Woche 2:

Die Unterschiede in dieser Woche beschränkten sich lediglich darauf, dass die finalisierte Datenbank an den Fachdozenten am kommenden Montag stattfand, da die Finalisierung Freitag erst recht spät vollendet war und die erst am kommenden Montag gelesen wurde.

Woche 3:

Die wohl größte Änderung innerhalb der Roadmap war diese Woche, weil es das erste Mal Probleme mit der Datenbankdarstellung gab, an der zu viel auf unnötige Kleinigkeiten eingegangen wurde. Die generelle Benutzung dieser Datenbankvisualisierungssoftware konnte im Verlauf angezweifelt werden, da die Zeit eher auf das direkte Arbeiten mit Datenbanken und SQL hätte gelenkt werden können. Ein weiterer kleiner Unterschied in der Soll Roadmap ist lediglich auf die genaue Aufteilung der Aufgaben zurückzuführen, da ich Anfang dieser Woche noch

keine Übersicht darüber hatte, wie ich welche Programmfunktionen strukturieren sollte.

Woche 4 und 5:

Keine Unterschiede, Soll und Ist Roadmap zeigen keine Unterschiede.

Fazit:

Die Erstellung einer Roadmap, auch im Vergleich zwischen Ist und Soll, war in diesem Projekt eine sehr große Hilfe.. Einmal aus der simplen, optisch einfachen Übersicht welche Aufgaben schon erledigt waren, also auch das simple Verschieben von Aufgaben, deren terminlicher Hintergrund sich geändert hatte.

Da die Planung wochenweise stattfand, konnte schneller auf Änderungen im Plan reagiert werden und diese in das Projekt implementieren werden.

Identity Design



#99E550



#FFA64D

Hellerer
Farbton

#5FCDE4



Hauptfarbe

#238CAD

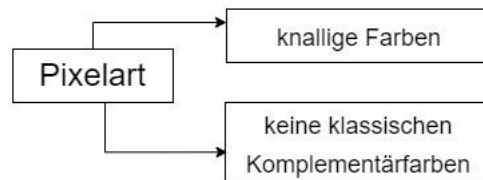


Dunklerer
Farbton

#135073



#AC3232



Identity Design

Art Style

Als Auswahl des Art Styles wurde ein Stil der 16 Bit Gaming Ära Ende der 1980er / 1990er Jahre entschieden.

Es zeigt die typisch alte Dungeons & Dragons Sparte als RPG in den 1980er Jahren. Rein aus pragmatischer Sicht sind die Grafiken in diesem Art Style aufgrund der zeitlichen Begrenzung dieses Projekts schneller und effizienter realisiert.

Der 16 Bit Pixel Art Style zeichnet sich durch die Verwendung von wenig Pixeln auf wenig Platz aus, auch wenn die Grafik am Ende hochskaliert wird.

Dieser Art Style zeichnet sich ebenso durch ein retrospektives Verhalten aus, das den User an die „guten alten Zeiten“ der Gamingindustrie erinnern soll, in denen die 16 Bit Technologie als Standard etabliert war.

Farbwahl

Die Auswahl der Farben fällt auf für 16 Bit Pixel Art typischen eher knalligen Farben, die aufgrund von Hardwarelimitationen zu dieser Zeit ihren Platz gefunden hatten.

Blau

Sanftmut, Klarheit, Kälte

Wirkung: Vertrauen, Sicherheit

Rot

Gefahr, Aggressivität, Kampf, loderndes Feuer.

Wirkung: repräsentativ, kompromisslos und energiegeladen

Kontrast zwischen Blau und Rot

Das gezielte Einsetzen von Blau und Rot in dieser Kombination dient als Kampf zwischen dem Guten (User) und dem Bösen (Antagonist)

Grün

Klarheit, Frische, Offenheit, Bewusstsein

Wirkung: ruhig und natürlich

Logo und Typografie



ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
1234567890!@#\$%^&*()=?

Headline 1 Size 30 pt

Subheadline 2 Size 24 pt

Subheadline 3 Size 21 pt

Text Size 18 pt

Organe

Erdig, Bodenständigkeit, Festigkeit
Wirkung: extrovertiert und ungehemmt

Farbübergang von Blau zu Grün

Der Übergang von Blau zu Grün symbolisiert eine Erweiterung des Guten in eine bekannte, vertraute Farbe.

Mit dem Farbverlauf innerhalb des Logos dient dies weiter dazu, dem Bild eine Tiefe zu geben.

Farbübergang von Rot zu Orange

Der Übergang von Blau zu Grün symbolisiert eine Erweiterung des Bösen (Antagonist) in eine grelle, Signalfarbe.

Mit dem Farbverlauf innerhalb des Logos dient dies weiter dazu, dem Bild eine Tiefe zu geben.

Typografie

Auch hier wurde die Typografie nach 16 Bit Pixel Look entschieden.

Die Schriftart nennt sich hier zwar „Minecraft“ dennoch handelt es sich nicht um originalen Minecraft Typografie, sondern eine einfache, pixelbasierte Schrift, welche keine Copyright Restriktionen unterliegen.

(<https://www.dafont.com/de/font-comment.php?file=minecraft>)

Da sich die Applikation im forcierten Fullscreen befinden wird, kann diese Schrift problemlos gut lesbar höher dargestellt werden.

Datenbankgestaltung

Entscheidungsfindung zur Datenbank

Bereits vor der Planungsphase musste klar sein, dass ich eine Technologie zur Speicherung des Fortschritts des Users implementieren musste, da sonst jegliche Art der Progression verloren ginge.

Nach mehreren Recherchen erreichte stellte sich heraus, dass eine Speicherung der Userdaten könne entweder per JSON File oder einer offline Datenbank im Backend realisiert werden.

Vor- und Nachteile JSON File

Vorteile

Einfachere Implementierung in Code
mehrere JSON Files können für spezifische Speicherungen wie Progression, etc. angefertigt werden

Nachteile

JSON Files müssen lokal gespeichert werden

sind die JSON Files korrumpiert oder zerstört, so ist auch die gesamte Progression des Users verloren

JSON Files sind komprimierbar, das bedeutet der User kann sich etwaige Progression erschummeln

Wie komplex ist die Implementierung von neuen JSON File Strukturen bei nachträglicher oder langfristiger Erweiterung der Applikation?

Vor- und Nachteile offline Datenbank

Vorteile

Datenbank kann in Bezug auf Erweiterbarkeit einfacher angepasst werden

Eigener Wissenstand wird bei Implementierung einer offline Datenbank erweitert (Aufbau, Normalisierung, Implementierung, SQL, etc.)

Schwer bis unmöglich wird die durch den User generierte Kompromittierung durch eine Datenbank

keine Abhängigkeit der Progression auf eine einzelne Datei, welche „offen“ auf dem System liegt

Nachteile

deutlich höherer Implementierungsaufwand

deutlich höherer Planungs- und Testungszeitraum

Im Vergleich der Vor- und Nachteile zeigte sich deutlich die Zustimmung für eine offline Datenbank.

Datenbank Finalisierung

Character

CharacterID (PK)	Class	ClassID	Level	MainAttr	HP	Dexterity	Magic	Weapon	Helmet	Chest (FK)	Earring	Ring
1	Paladin	PLD	1	12	10	5	5	1	2	3	4	5
2	Druide	DRD	1	12	10	5	15	1	1	1	1	1
3	Zauberer	ZAU	1	12	10	5	15	1	1	1	1	1
4	Jäger	JGR	1	12	10	15	5	1	1	1	1	1

Items

ItemID (PK)	Slot	Class	LevelReq	MainAttr	HP	Dexterity	Magic	PictureSM	PictureBIG	Name	Description	InInventory
1	Weapon	Paladin	5	15	12	0	0			Schwert der Bequemlichkeit		false
2	Helmet	Paladin	4	15	12	0	0			Helm der Bequemlichkeit		false
3	Chest	Paladin	3	15	12	0	0			Rüstung der Bequemlichkeit		false
4	Earring	Paladin	2	15	12	0	0			Ohrhring der Bequemlichkeit		false
5	Ring	Paladin	1	15	12	0	0			Ring der Bequemlichkeit		false
6	Weapon	Paladin	10	20	14	0	0			Schwert der Abwägung		false
7	Helmet	Paladin	9	20	14	0	0			Helm der Abwägung		false
8	Chest	Paladin	8	20	14	0	0			Rüstung der Abwägung		false
9	Earring	Paladin	7	20	14	0	0			Ohrhring der Abwägung		false
10	Ring	Paladin	6	20	14	0	0			Ring der Abwägung		false
11	Weapon	Paladin	15	25	16	0	0			Schwert der Motivation		false
12	Helmet	Paladin	14	25	16	0	0			Helm der Motivation		false
13	Chest	Paladin	13	25	16	0	0			Rüstung der Motivation		false
14	Earring	Paladin	12	25	16	0	0			Ohrhring der Motivation		false
15	Ring	Paladin	11	25	16	0	0			Ring der Motivation		false
16	Weapon	Paladin	20	30	18	0	0			Zerschneider der Prokstitution		false
17	Helmet	Paladin	19	30	18	0	0			Schützender Helm des Machens		false
18	Chest	Paladin	18	30	18	0	0			Schützende Rüstung des Super Seins		false
19	Earring	Paladin	17	30	18	0	0			Schützender Ohrhring der Anerkennung		false
20	Ring	Paladin	16	30	18	0	0			Schützender Ring der Selbstbestimmung		false
21	Weapon	Druide	5	15	12	0	17			Stab der Bequemlichkeit		false
22	Helmet	Druide	4	15	12	0	17			Hut der Bequemlichkeit		false
23	Chest	Druide	3	15	12	0	17			Robe der Bequemlichkeit		false
24	Earring	Druide	2	15	12	0	17			Ohrhring der Bequemlichkeit		false
25	Weapon	Druide	10	20	14	0	19			Stab der Abwägung		false
26	Helmet	Druide	9	20	14	0	19			Hut der Abwägung		false
27	Chest	Druide	8	20	14	0	19			Robe der Abwägung		false
28	Earring	Druide	7	20	14	0	19			Ohrhring der Abwägung		false
29	Ring	Druide	6	20	14	0	19			Ring der Abwägung		false
30	Weapon	Druide	15	25	16	0	21			Stab der Motivation		false
31	Helmet	Druide	14	25	16	0	21			Hut der Motivation		false
32	Chest	Druide	13	25	16	0	21			Robe der Motivation		false
33	Earring	Druide	12	25	16	0	21			Ohrhring der Motivation		false
34	Ring	Druide	11	25	16	0	21			Ring der Motivation		false
35	Weapon	Druide	20	30	18	0	23			Heilung der Prokstitution		false
36	Helmet	Druide	19	30	18	0	23			Heiliger Hut des Machens		false
37	Chest	Druide	18	30	18	0	23			Heilige Robe des Super Seins		false
38	Earring	Druide	17	30	18	0	23			Heiliger Ohrhring der Anerkennung		false
39	Ring	Druide	16	30	18	0	23			Heiliger Ring der Selbstbestimmung		false
40	Chest	Zauberer	8	20	14	0	19			Robe der Abwägung		false
41	Earring	Zauberer	7	20	14	0	19			Ohrhring der Abwägung		false
42	Ring	Zauberer	6	20	14	0	19			Ring der Abwägung		false
43	Weapon	Zauberer	15	25	16	0	21			Feuerhand der Motivation		false
44	Helmet	Zauberer	14	25	16	0	21			Hut der Motivation		false
45	Chest	Zauberer	13	25	16	0	21			Robe der Motivation		false
46	Earring	Zauberer	12	25	16	0	21			Ohrhring der Motivation		false
47	Ring	Zauberer	11	25	16	0	21			Ring der Motivation		false
48	Weapon	Zauberer	20	30	18	0	23			Verbrennung der Prokstitution		false
49	Helmet	Zauberer	19	30	18	0	23			Zerstörerische Hut des Machens		false
50	Chest	Zauberer	18	30	18	0	23			Zerstörerische Robe des Super Seins		false
51	Earring	Zauberer	17	30	18	0	23			Zerstörerischer Ohrhring der Anerkennung		false
52	Ring	Zauberer	16	30	18	0	23			Zerstörerischer Ring der Selbstbestimmung		false
53	Earring	Jäger	12	25	16	21	0			Ohrhring der Motivation		false
54	Ring	Jäger	11	25	16	21	0			Ring der Motivation		false
55	Weapon	Jäger	20	30	18	23	0			Durchbohrer der Prokstitution		false
56	Helmet	Jäger	19	30	18	23	0			Willhelmteilscher Helm des Machens		false
57	Chest	Jäger	18	30	18	23	0			Willhelmteilsche Rüstung des Super Seins		false
58	Earring	Jäger	17	30	18	23	0			Willhelmteilscher Ohrhring der Anerkennung		false
59	Ring	Jäger	16	30	18	23	0			Willhelmteilscher Ring der Selbstbestimmung		false

Quest

QuestID (PK)	CharacterID	completedDate	completed	QuestType (FK)	Task	Enemy (FK)	Place (FK)
1	1	2022-02-02	true	1	Geschirr spülen	1	1

QuestType

QuestTypeID (PK)	QuestLength	XP
1	1 Day	150
2	1 Week	1050
3	1 Month	4500

Enemy

EnemyID (PK)	EnemyName
1	das böse Monster der Unordnung
2	die böse Hexe Nerflia
3	der dunkle Ritter der Lustlosigkeit
4	die Horde des Ich-mach-das-später-Konglomerats

Place

PlaceID (PK)	PlaceDescription
1	düsterer Wald des Nichtstuns
2	Verlies der Prokstitution
3	Schlossgärten von Drückebergia
4	Totengrab des Gottes der Faulheit

Level

LevelID (PK)	XPNextLevel	XPTotal
1	150	0
2	300	150
3	450	600
4	600	1200
5	1500	2700
6	1800	4500
7	2100	6600
8	2400	9000
9	2700	11700
10	4500	16200
11	4950	21150
12	5400	26550
13	5850	32400
14	6300	38700
15	6750	45450
16	9600	55050
17	10200	65250
18	10800	76050
19	11400	87450
20	12000	99450

Datenbankgestaltung

Erste Version

Die Gestaltung der Datenbank entsprach einer der schwierigeren Aufgaben dieses Projekts, da bisweilen noch nicht genug Erfahrung zur Konzeptionierung, Gestaltung und Verwaltung einer Datenbank vorhanden war.

Die grobe Konzeptionierung sah vor die Datenbank in zwei große Bereiche einzuteilen:

User related Database:

Hier zu sind alle Informationen gespeichert, die durch Aktionen des Users hergeleitet werden, wie Gehilfenlevel, Erfahrungspunkte, aber ebenso ausgerüstete Items, sowie die Speicherung der Quests.

Game related Database:

In diesen Tabellen werden alle Informationen gespeichert, die ausschließlich in den Verwaltungsbereich der Applikation geraten wie eine Tabelle mit allen Items, sowie das Antagonistensystem.

Zweite Version

Es zeigte sich die erste Version als wenig suffizient, da das User und Game related Prinzip der ersten Version als zu ineinander vermischt gelten, als dass es für zwei große Blöcke Sinn machen würde.

Die zweite Version der Datenbank wurde in fünf Blöcke unterteilt:

Vier Mal eine jeweilige Struktur für jeden Charakter bestehend aus aktueller Quest, abgeschlossener Quest, Charakter und die in einzelne Tabellen zerlegten Items.

Der Hintergrund dieses Gedankengangs war, da jeder Gefährte auf seine eigene Weise interagiert, Quests löst und Items erhält würde es Sinn machen, diese Blöcke zu outsourcen. Damit könnte für jeden einzelnen Gefährten eine Datenbank erstellt werden, es können unabhängig voneinander verschiedene Connections zu den Datenbanken geöffnet werden und so können mehrere Interaktionen mit den Gefährten parallel laufen.

Als letzten Block entstand der Antagonist, der abseits dieses Systems steht.

Dritte Version

Es stellte sich bei intensiverer Auseinandersetzung mit der zweiten Version heraus, dass mit mehreren unterschiedlichen, unabhängig von einander agierenden Datenbanken ebenfalls die Masse des Codes, sowie unnötige Datenredundanzen anfielen. Aus diesem Grund wurde diese Version verworfen und es wurde die dritte Form als finale Datenbank realisiert.

Mock Up Dashboard

Window Title			Speichern									
Dashboard	Helden											
Bild Held	<p>Laufende Quest</p> <p>Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo d</p> <p>Abgeschlossen am: xx.xx.xxxx Erfahrungspunkte: xx XP</p>	<p>Abgeschlossene Quests</p> <table border="1"> <tr><td>✓text</td><td>✓text</td></tr> <tr><td>✓text</td><td>✓text</td></tr> <tr><td>✓text</td><td>✓text</td></tr> <tr><td>✓text</td><td>✓text</td></tr> </table>			✓text	✓text	✓text	✓text	✓text	✓text	✓text	✓text
✓text	✓text											
✓text	✓text											
✓text	✓text											
✓text	✓text											
Bild Held	<p>Laufende Quest</p> <p>Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo d</p> <p>Abgeschlossen am: xx.xx.xxxx Erfahrungspunkte: xx XP</p>	<p>Abgeschlossene Quests</p> <table border="1"> <tr><td>✓text</td><td>✓text</td></tr> <tr><td>✓text</td><td>✓text</td></tr> <tr><td>✓text</td><td>✓text</td></tr> <tr><td>✓text</td><td>✓text</td></tr> </table>			✓text	✓text	✓text	✓text	✓text	✓text	✓text	✓text
✓text	✓text											
✓text	✓text											
✓text	✓text											
✓text	✓text											
Bild Held	<p>Laufende Quest</p> <p>Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo d</p> <p>Abgeschlossen am: xx.xx.xxxx Erfahrungspunkte: xx XP</p>	<p>Abgeschlossene Quests</p> <table border="1"> <tr><td>✓text</td><td>✓text</td></tr> <tr><td>✓text</td><td>✓text</td></tr> <tr><td>✓text</td><td>✓text</td></tr> <tr><td>✓text</td><td>✓text</td></tr> </table>			✓text	✓text	✓text	✓text	✓text	✓text	✓text	✓text
✓text	✓text											
✓text	✓text											
✓text	✓text											
✓text	✓text											
Bild Held	<p>Laufende Quest</p> <p>Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo d</p> <p>Abgeschlossen am: xx.xx.xxxx Erfahrungspunkte: xx XP</p>	<p>Abgeschlossene Quests</p> <table border="1"> <tr><td>✓text</td><td>✓text</td></tr> <tr><td>✓text</td><td>✓text</td></tr> <tr><td>✓text</td><td>✓text</td></tr> <tr><td>✓text</td><td>✓text</td></tr> </table>			✓text	✓text	✓text	✓text	✓text	✓text	✓text	✓text
✓text	✓text											
✓text	✓text											
✓text	✓text											
✓text	✓text											
Bild Held	<p>Laufende Quest</p> <p>Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo d</p> <p>Abgeschlossen am: xx.xx.xxxx Erfahrungspunkte: xx XP</p>	<p>Abgeschlossene Quests</p> <table border="1"> <tr><td>✓text</td><td>✓text</td></tr> <tr><td>✓text</td><td>✓text</td></tr> <tr><td>✓text</td><td>✓text</td></tr> <tr><td>✓text</td><td>✓text</td></tr> </table>			✓text	✓text	✓text	✓text	✓text	✓text	✓text	✓text
✓text	✓text											
✓text	✓text											
✓text	✓text											
✓text	✓text											

GUI und Grafiken

Erstellung Charaktere

Paladin

Waffen: Schwert und Schild
 Rüstung: Schwere Rüstung
 Charakter: mürrisch, skeptisch

Druide

Waffen: Stab
 Rüstung: Robe
 Charakter: hektisch, aufgedreht

Barde

Waffen: Bogen
 Rüstung: Leichte Rüstung
 Charakter: ängstlich, hektisch

Zauberer

Waffen: Feuerhand
 Rüstung: Robe
 Charakter: ruhig, etwas besorgt

Jäger

Waffen: Dolche
 Rüstung: Leichte Rüstung
 Charakter: müde, gelangweilt

Aufbau GUI

Da dies hauptsächlich eine To Do Liste ist, liegt der UX Fokus auf der Übersicht über vergangene Aufgaben, schnelles Erkennen von aktuellen Aufgaben, sowie wenig Text und klar abgegrenzte Bereiche.

Die Entscheidung für ein Kachelsystem wurde getroffen, in dem jeder Held seine eigene Übersicht über aktuelle Quest, Inventar, Statusberichte und ein Panel für neue Quests besitzt.











Da dies in einer Kachel zu wenig Platz bedeuten würde, wird die Applikation in zwei Tabs unterteilt werden

Der erste Tab (Dashboard) enthält wichtige Informationen über die aktuell laufende Quest, sowie abgeschlossene Quests. Dies wird die erste Seite der Applikation darstellen und hat somit einen eigenen Bereich der klar als To Do Liste klassifizierbar ist.

Der zweite Tab (Helden) zeigt alle Informationen über die Helden, wie Statuswerte, Inventar und das Erstellen neuer Quests.

Damit sind alle rudimentären Funktionen der Applikation abgedeckt.

Mock Up Dashboard

Window Title				
Dashboard		Helden		Speichern
Bild Held	Charakter Statistiken Level xx 	Hauptattribut: 10 Vitalität: 15 Stärke: 13 Geschicklichkeit: 2 Rhythmusgefühl: 4 Heilungsmagie: 1 Intelligenz: 3 xxxxx / xxxxx XP	Inventar 	Quest Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo d Abgeschlossen am: xx.xx.xxxx Erfahrungspunkte: xx XP
Bild Held	Charakter Statistiken Level xx 	Hauptattribut: 10 Vitalität: 15 Stärke: 13 Geschicklichkeit: 2 Rhythmusgefühl: 4 Heilungsmagie: 1 Intelligenz: 3 xxxxx / xxxxx XP	Inventar 	Quest Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo d <input type="button" value="Neue Quest"/>
Bild Held	Charakter Statistiken Level xx 	Hauptattribut: 10 Vitalität: 15 Stärke: 13 Geschicklichkeit: 2 Rhythmusgefühl: 4 Heilungsmagie: 1 Intelligenz: 3 xxxxx / xxxxx XP	Inventar 	Quest <input type="text" value="Welche Aufgabe"/> <input type="text" value="Enddatum"/> <input type="button" value="▼"/> <input type="button" value="Szenario auswählen"/> <input type="button" value="▼"/> <input type="button" value="Gegner auswählen"/> <input type="button" value="▼"/> <input type="button" value="auf ins Abenteuer"/>
Bild Held	Charakter Statistiken Level xx 	Hauptattribut: 10 Vitalität: 15 Stärke: 13 Geschicklichkeit: 2 Rhythmusgefühl: 4 Heilungsmagie: 1 Intelligenz: 3 xxxxx / xxxxx XP	Inventar 	Quest <input type="text" value="Welche Aufgabe"/> <input type="text" value="Enddatum"/> <input type="button" value="▼"/> <input type="button" value="Szenario auswählen"/> <input type="button" value="▼"/> <input type="button" value="Gegner auswählen"/> <input type="button" value="▼"/> <input type="button" value="auf ins Abenteuer"/>
Bild Held	Charakter Statistiken Level xx 	Hauptattribut: 10 Vitalität: 15 Stärke: 13 Geschicklichkeit: 2 Rhythmusgefühl: 4 Heilungsmagie: 1 Intelligenz: 3 xxxxx / xxxxx XP	Inventar 	Quest <input type="text" value="Welche Aufgabe"/> <input type="text" value="Enddatum"/> <input type="button" value="▼"/> <input type="button" value="Szenario auswählen"/> <input type="button" value="▼"/> <input type="button" value="Gegner auswählen"/> <input type="button" value="▼"/> <input type="button" value="auf ins Abenteuer"/>

Erstellung Wire Frame

Die Erstellung des Wire Frames erfolgte mit dem Programm Pencil, eine Einarbeitung in das Programm war nicht notwendig, die Funktionen darin sind selbsterklärend und intuitiv.

Die Ausarbeitung des Wire Frames wurde erfolgreich abgeschlossen und durch einen erfahrenen UX Designer aus dem privaten Sektor, konnte dies abgenommen werden.

Erstellung Mock Up

Für das Mock Up nutzte ich wieder Scene Builder und unterteilte die einzelnen Abschnitte durch farbliche Trennung und passte die Größen und Ausrichtungen der einzelnen Bestandteile an.

Erstellung der GUI Grafiken

Die Erstellung der GUI Grafiken erfolgte mit dem Programm Aseprite.

Nachdem das Grundlayout der einzelnen Bestandteile recht schnell errichtet war, wurden verschiedene Farbkombinationen gegenübergestellt.

Die Bearbeitung der restlichen GUI Elemente war meistens Copy & Paste der vorher erstellten Grafiken, sowie einzelnes händisches Nachzeichnen, sowie Anpassen an die spezifische Größe des Elements.

Farbauswahl

Die Farbauswahl des GUI entfiel auf einen lila/dunkelblauen Farbton und dessen Verläufe. Der Effekt erinnert an Dungeons, ist eher dunkel gehalten und zeigt ähnelt retrospektiv an die Farbe des Game Boy Colors und Game Boy Advance, welche ebenfalls einen Bezug zur 8 / 16 Bit Ära herstellt.

Einarbeitung in JavaFX

Java GUI Grundlagen

Swing

Eine bereits in Java enthaltende Möglichkeit GUI zu entwickeln, liegt in Swing.

Swing liefert rudimentäre Möglichkeiten zur Gestaltung eines GUI.

JavaFX

JavaFX ist ein auf XML Integration basiertes GUI Framework mit der Möglichkeit zur Verwendung von CSS Code zur nutzerspezifischen Änderung von optischen Elementen.

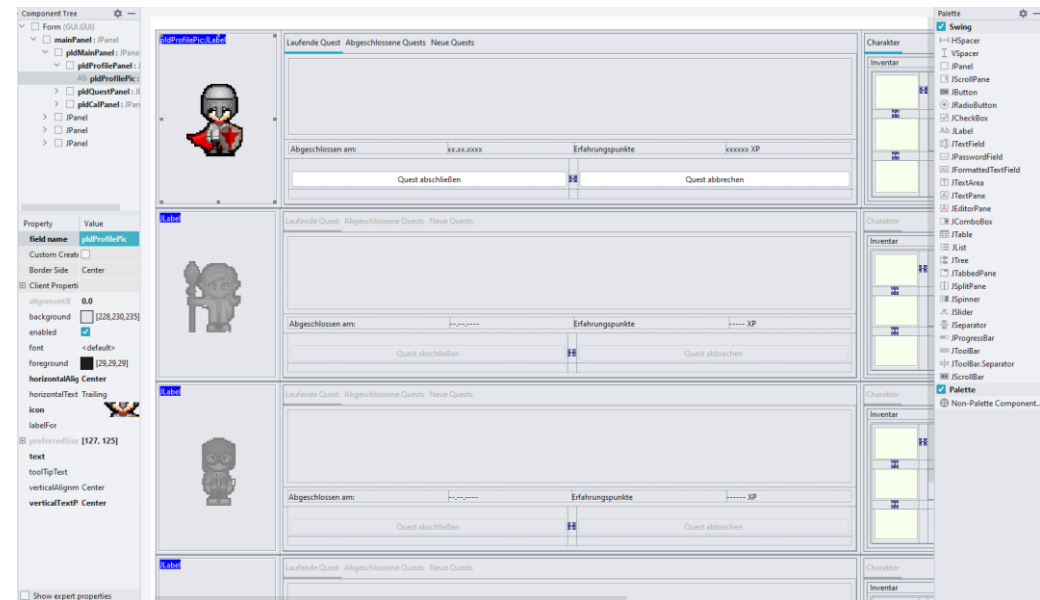
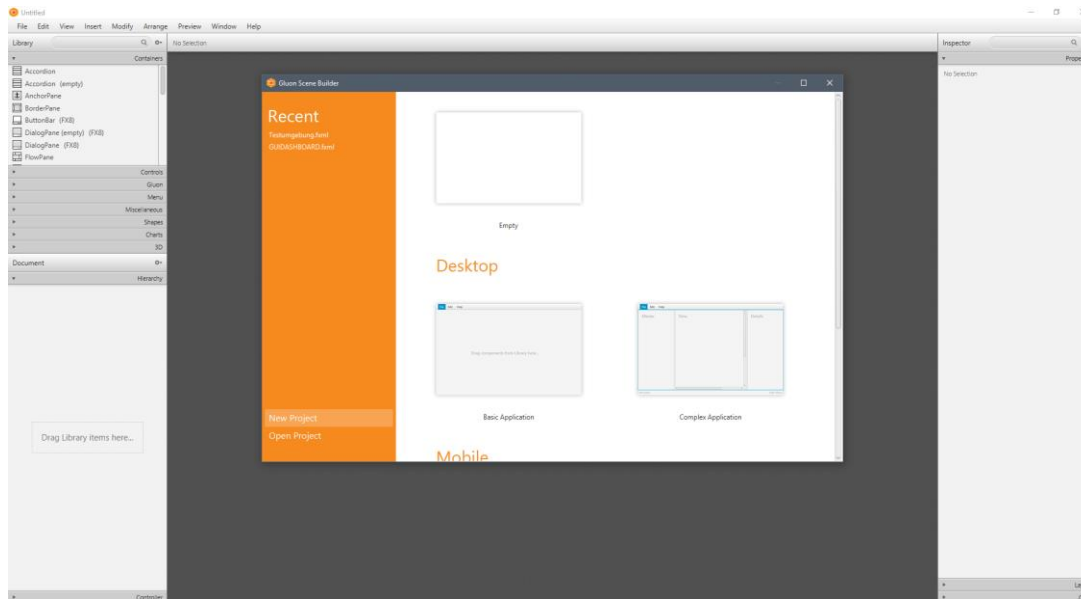
Entscheidung für JavaFX

Theoretisch liefert Swing für das Projekt alle notwendigen Elemente wie Buttons, Textboxen, Drop Down Menüs, etc. eines GUI, dennoch überwiegen folgende Gründe die Entscheidung für JavaFX.

Verwendung von CSS

Da in diesem Projekt komplexere Grafiken und eine an das 16 Bit Pixel Art Thema angepasste GUI enthalten werden, habe ich mich für dieses Framework entschieden.

JavaFX SceneBuilder vs. IntelliJ Swing UI Builder



IntelliJ interner Swing Builder

IntelliJ besitzt einen bereits vorinstallierten Swing GUI Builder, dieser ist für das Arbeiten aber eher nicht intuitiv. Die nicht gegebene Möglichkeit per Drag and Drop Elemente pixelperfect zu setzen, sondern das Layout in einem platzierbaren Grid System zu implementieren bekräftigte weiter die Zustimmung für JavaFX.

Zwar gibt es ein Plugin, das einen Scene Builder direkt in IntelliJ implementieren kann, da dieser aber kostenpflichtig ist und es auch andere Möglichkeiten zur Bearbeitung gibt, habe ich mich dagegen entschieden.

Als letzte Möglichkeit läge die IDE Eclipse nahe, welche einen Swing Builder bietet, welcher pixelperfect die Verteilung von Elementen ermöglicht. Aus Gründen der Gewohnheit an IntelliJ wurde diese Idee verworfen.

SceneBuilder

Das externe Programm SceneBuilder, welches direkt von den Programmierern des JavaFX Frameworks erstellt wurde, bietet eine große Menge an layoutorientierten Maßnahmen.

Grundlagen JavaFX

Im Vergleich zu Swing, in welchem man neue GUI Elemente direkt als Objekt in einer Java Klasse erstellen kann, wird die Programmierung von Elementen in zwei große Bereiche eingeteilt:

Java und XML.

Dabei fällt die funktionale Struktur auf Java, die optische Struktur auf XML, genauer gesagt FXML.

Funktionale Struktur

Für die Funktionale Struktur in JavaFX wird zwischen der Applikations-Klasse und der Controller-Klasse unterschieden.

In der Applikations-Klasse befindet sich die letztendlich ausführbare main Methode, alles innerhalb der Controller-Klasse beschreibt das Verhalten der einzelnen GUI Elemente, schafft in Anbetracht auf Accessibility eine Schnittstelle zwischen FXML und Java und implementiert Methoden.

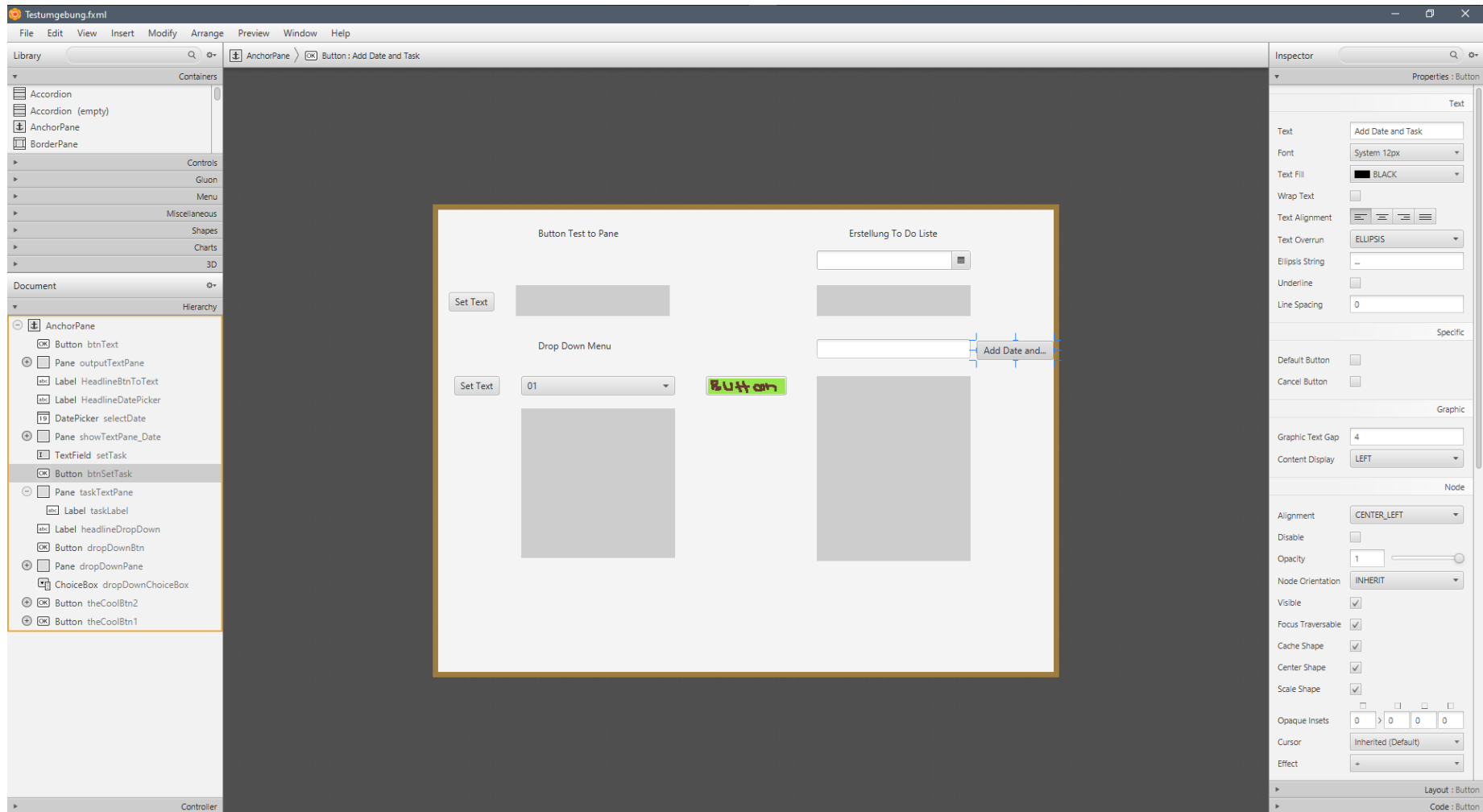
Optische Struktur

Unter FXML versteht man eine Weiterentwicklung von XML, eine mit HTML verwandten Markup Language, die durch ihre leichte Erweiterbarkeit das Hinzufügen von `<tags>` gestattet.

Da die Grundprinzipien von XML mit einem guten Verständnis von HTML bereits erreicht sind, wird hier keine große Einarbeitung benötigt.

Die Recherche zu XML erfolgte innerhalb einer halben Stunde und war / ist einfach verständlich.

Dummy und Testumgebung in SceneBuilder



Erstellung Dummy JavaFX

Die Erstellung einer Testumgebung, bzw. eines Dummys in JavaFX wurde durch das Programm Scene Builder realisiert.

In diesem Dummy waren alle wichtigen Grundfunktionen und Interaktionsmöglichkeiten meiner späteren Applikation verbaut.

Einen Button der nach einem Click einen Text in einem bestimmten Feld erscheinen lässt

Einen Date Picker, mit Ausgabe eines Textes des ausgewählten Datums

Ein Textfield, das dem User die Möglichkeit bietet, eigene Eingaben umzusetzen, welche ebenfalls in einem separaten Feld ausgegeben wird

Eine ChoiceBox, oder auch Drop Down Menü genannt, ebenso mit Ausgabe in einem eigenen Feld

Ein Mouseover Event eines Buttons, der eine Grafik wechselt.

Als Informationsquellen hierzu verwendete ich die JavaFX und Scene Builder eigene Dokumentation, Tutorials auf YouTube, Internetrecherche via Stackoverflow und diversen anderen Seiten, sowie einfaches Ausprobieren.

Der Verlauf der Erstellung verlief recht gut, es stellten sich mir jedoch einige Probleme in den Weg:

Einbettung in Java

Die Einbettung in Java gestaltete sich als problematisch, da ich zunächst das Zusammenspiel zwischen Erweiterungen, externen Libraries und Frameworks lernen musste.

Nach dem aber ein grundsätzliches, sehr rudimentäres Verständnis dafür bestand, war das Implementieren sehr einfach

Zusammenspiel zwischen JXML File, Controller und Main Methode

Das Verständnis zu erlangen, welche Aktion in welchem Teil der Applikation ausgeführt werden soll, war auch zunächst in der Dokumentation, sowie YouTube Tutorials nicht recht ersichtlich. Dank Stackoverflow und eigenem Ausprobieren konnte dieses Problem dennoch gelöst werden.

Händisches Einfügen des Codes bei Drop Down Menüs

In sämtlichen interaktiven Bedienelementen bringt Scene Builder die Möglichkeit mit, schnell und unkompliziert eine Aktion zu erstellen.

Aus irgendwelchen Gründen auch immer, ist dies bei der Ausführung, dem Einsetzen von Optionen und der Integration einer Java Methode zur Ausführung der Funktion nicht möglich.

Aus diesem Grund war eine ausgereifte Recherche notwendig, um eine Möglichkeit zu finden, diese Probleme zu kompensieren, dennoch wurde ein guter Einblick in die syntaktische Struktur von Methoden zur Ausführbarkeit von Bedienelementen in JavaFX erhalten.

H2 Datenbank Framework Konsole

English ▼ Optionen Tools Hilfe

Login

Gespeicherte Einstellung: Generic H2 (Server) ▼

Einstellungs-Name: Generic H2 (Server) Speichern Entfernen

Datenbank-Treiber Klasse: org.h2.Driver

JDBC URL: jdbc:h2:C:\Users\mathi\OneDrive\FIA\DO IT\rpg-it-swing'

Benutzername: sa

Passwort:

Verbinden Verbindung testen

Auto-Commit ☑ Maximale Anzahl Zeilen: 1000 ▼ Auto-Complete Aus ▼ Automatische Auswahl An ▼ ?

jdbc:h2:C:\Users\mathi\OneDrive\FIA\DO IT\rpg-it-swing' Ausführen Ausgewähltes Ausführen Auto-Complete Leeren SQL Befehl:

SELECT * FROM PLAYABLE_CHARACTER |

SELECT * FROM PLAYABLE_CHARACTER;

CHARACTER_ID	CLASS	CLASS_ID	LEVEL	CURRENT_XP	MAIN_ATTR	HP	DEXTERITY	MAGIC	WEAPON	HELMET	CHEST	EARRING	RING
1	Paladin	PLD	8	1800	90	56	0	0	1	1	2	2	2
2	Druide	DRD	8	9000	12	10	0	15	1	1	1	1	1
3	Zauberer	ZAU	8	9000	12	10	0	15	1	1	1	1	1
4	Jaeger	JGR	1	0	12	10	15	0	1	1	1	1	1

(4 Datensätze, 3 ms)

Bearbeiten

☰
☰ CHEST
 ☰ EARRING
 ☰ ENEMY
 ☰ HELMET
 ☰ LEVEL
 ☰ LOCATION
 ☰ PLAYABLE_CHARACTER
 ☰ QUEST
 ☰ QUEST_TYPE
 ☰ RING
 ☰ WEAPON
 ☰ INFORMATION_SCHEMA
 ☰ Benutzer
 ? H2 2.1.210 (2022-01-17)

Einarbeitung in Java Datenbank Framework h2

Auswahlkriterien von h2

Da sich die bisherige Arbeit mit Datenbanken nur auf XAMPP beschränkt hatte, waren besondere Anforderungen an das Datenbank Framework gegeben:

- Große Verbreitung, ergo große Erreichbarkeit von Hilfestellung
- Eingebettete Datenbank
- Syntaktisch einfach
- gute Dokumentation
- einfache Integration in Java
- Einsatz von IntelliJ Plug ins

Erstellung Dummy Datenbank

Die Erstellung der Dummy Datenbank erfolgte recht zügig und funktionierte ohne Probleme. Dabei ging es lediglich darum, eine kleine Dummy Datenbank aufzubauen, das gesondert zum Dummy GUI steht, um die Funktionen zu testen.

Implementierung von Dummy Datenbank in JavaFX Dummy

Die Implementierung in den JavaFX GUI Dummy erfolgte ohne Probleme und war einfach umsetzbar.

Dabei wurde der Date Picker und das Textfield verbunden, um gemeinsame Daten an die Datenbank zu schicken.

Es wurde ein eigen erstellter Mechanismus verwendet, der jeweils prüfte, ob beide Felder nicht leer sind, damit keine leeren Daten an die Datenbank geschickt werden.

Frag nach Prävention von SQL Injection

SQL Injection kann dramatische Folgen für die Datenbank im Backend haben. Dabei werden SQL Befehle in ein Textfeld eingetragen, um damit die Datenbank direkt zu manipulieren.

Dennoch besteht hier die grundsätzliche Frage in wie weit SQL Injection in einer offline Desktop Anwendung Sinn macht.

Natürlich könnte das verheerende Folgen für die Datenbank haben, da dieser allerdings offline platziert ist, würde der User sich lediglich die Applikation selbst zerstören und sonst keine großen Eingriffe auf System möglich machen.


Wäre die Applikation mit einer online Datenbank verbunden, so müsste man sich deutlich mehr Gedanken dazu machen, da eine deutlich größere Zahl an Usern betroffen wäre.

Das h2 Datenbankframework hat Maßnahmen zur Prävention von SQL Injection, dennoch kann keine valide Quelle zur Erklärung des Codes gefunden werden und so kam der Entschluss durch eine simple If Klausel, die die SQL Übertragung bei Verwendung eines „=“ Zeichens zu beendet.

Finales JavaFX GUI - Dashboard

DASHBOARD

HELDEN



Laufende Quests

Et vero eos et accusam et justo duo dolores et ea rebum. Stet olita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea, no sea takimata sanctus est


Erfahrungspunkte: xxxxxxxxxxxx
Abgeschlossen am: xxx . xxx . xxxxx

✓

✗

Abgeschlossene Quests

✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests
✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests
✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests



Laufende Quests

Et vero eos et accusam et justo duo dolores et ea rebum. Stet olita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea, no sea takimata sanctus est


Erfahrungspunkte: xxxxxxxxxxxx
Abgeschlossen am: xxx . xxx . xxxxx

✓

✗

Abgeschlossene Quests

✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests
✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests
✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests



Laufende Quests

Et vero eos et accusam et justo duo dolores et ea rebum. Stet olita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea, no sea takimata sanctus est


Erfahrungspunkte: xxxxxxxxxxxx
Abgeschlossen am: xxx . xxx . xxxxx

✓

✗

Abgeschlossene Quests

✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests
✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests
✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests



Laufende Quests

Et vero eos et accusam et justo duo dolores et ea rebum. Stet olita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea, no sea takimata sanctus est

Erfahrungspunkte: xxxxxxxxxxxx
Abgeschlossen am: xxx . xxx . xxxxx

✓

✗

Abgeschlossene Quests

✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests
✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests
✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests	✓ Laufende Quests



Beginn der Programmierung

Gestaltung finalen GUI in JavaFX

Die Gestaltung des finalen GUI gestaltete sich als recht einfach, es mussten hier und da ein paar Grafiken angepasst werden, der generelle Aufwand betrug nicht mehr als einen Tag und geschah ohne Probleme.

Die zuvor angelegte Datei mit exakter Größenangabe der einzelnen Elemente zahlte sich vollkommen aus, da die Abmessungen für die GUI Elemente bereits getroffen waren und nur noch platziert werden mussten.

Namenskonvention für JavaFX ID

Da zwischen der Kommunikation der FXML und des Java Files eindeutige IDs vergeben werden mussten und die Menge der IDs durch die Vielzahl an GUI Elementen nicht gerade wenig war, kam die Implementierung einer klaren Namenskonvention der FX:IDs in Frage.

Tab	Held	Pane	Art des GUI Elements	Verhalten
Dashboard	Paladin	Profil	Button	Hover
Held	Druide		cQuest (current Quest)	Background
	Zauberer		fQuest (finished Quest)	Pane
	Passive			
	Jäger		Inventar	etc.
			etc.	

Beispiel:

Der Name des Go Buttons im Helden Tab zum Beginn einer neuen Quest im Charakter Paladin

HeldPLDQuestButtonPassive

Durch die Verwendung des Camel Case lässt sich diese Namenskonvention gut lesen und bereits bei grobem Anschauen der FX:ID lässt sich erkennen in welchen Bereich diese fällt.

Probleme beim Erstellen der GUI

Probleme beim Erstellen des finalen GUI gab es nicht. Dies war eine simple und repetitive Aufgabe. Lediglich das Vertauschen von wenigen FX:IDs, vermutlich verursacht durch das stumpfe und repetitive Implementieren der Namen waren vorhanden.

Erstellen der Datenbank und Einpflegen der Daten

Nachdem die finale Datenbankstruktur ausgearbeitet war, kam die Idee nach Plug Ins und Third Party Tools zur Datenbankvisualisierung um, um mit diesen Plug Ins und Tools erstens eine bessere Übersicht über die Vorgänge der Datenbank zu haben, sowie die Erleichterung der Kommunikation zwischen Datenbank und IDE.

Als IntelliJ Plug In kam DB Navigator zur Verwendung, dieses unterstützte zwar das h2 Datenbank Framework, zeigte aber (ein bekanntes Problem, welches auch in der Dokumentation des Plug Ins beschrieben war) eine insuffiziente Konnektivität mit der Datenbank, sowie gravierenden Funktionseinschränkungen, welche mit das Arbeiten mit diesem Plug In deutlich erschweren würde.

Als Third Party Tool kam der Datenbank Viewer DBeaver zum Tragen, ein doch sehr mächtiges Tool um sehr intensiv mit der Datenbank zu arbeiten.

Hier zeigte sich auch das Problem, dass Änderungen an der Datenbank innerhalb des Programms zwar angezeigt wurden, diese aber nicht auf die finale Datenbank übertragen wurde.

Es kam die Idee, auch mit dem Bewusstsein dass dies mehr Zeit beanspruchen würde, die Datenbank händisch zu füllen, was sich aber als bessere Methode für das tiefere Verständnis des SQL Syntax und dessen Funktionsweise herausstellte.

Finales JavaFX GUI – Helden

DASHBOARD

HELDEN



Charakter

Hauptattribut

Leben

Geschoiktllichkeit

Magie

Level

XXX

XXXXXX / XXXXXX XP

Inventar



Lorem magna aliquam erat, sed diam voluptus. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor le mag et

Neue Quest

Lorem magna aliquam erat, sed diam voluptus. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor le mag et

GO



Charakter

Hauptattribut

Leben

Geschoiktllichkeit

Magie

Level

XXX

XXXXXX / XXXXXX XP

Inventar



Lorem magna aliquam erat, sed diam voluptus. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor le mag et

Neue Quest

Lorem magna aliquam erat, sed diam voluptus. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor le mag et

GO



Charakter

Hauptattribut

Leben

Geschoiktllichkeit

Magie

Level

XXX

XXXXXX / XXXXXX XP

Inventar



Lorem magna aliquam erat, sed diam voluptus. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor le mag et

Neue Quest

Lorem magna aliquam erat, sed diam voluptus. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor le mag et

GO



Charakter

Hauptattribut

Leben

Geschoiktllichkeit

Magie

Level

XXX

XXXXXX / XXXXXX XP

Inventar



Lorem magna aliquam erat, sed diam voluptus. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor le mag et

Neue Quest

Lorem magna aliquam erat, sed diam voluptus. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor le mag et

GO

Implementierung JavaFX Framework

Die Implementierung der GUI und des JavaFX Frameworks erfolgte ebenfalls ohne große Probleme, da als Vorlage bei weiteren Fragen die Dummy GUI heranziehen werden konnte.

Das erste Implementieren des JavaFX Frameworks zeigte sich zunächst komplikationslos, alle Grafiken wurden übernommen, die FX:ID des FXML Files waren ebenfalls noch vorhanden und alle Positionsangaben und Größeneinstellungen der Grafiken und Bedienelemente waren alle vorhanden.

Dennoch etablierte sich bei der Erstellung der ersten Funktionen das Problem, dass sich das GUI bei Veränderungen nicht anpasste, das XML File plangemäß auf die Funktionen reagierte und der Rein Code im Terminal keine Fehler aufzeigte.

Es stellte nach langer und intensiver Recherche heraus, dass es einen Fehler in der Abfrage der Modul Library gab. Die Prüfung verschiedenster Einstellungen (Einbettung der Library, Konnektivität der Module, Überprüfung der AusführungsVM und konnte diesen Fehler nicht beseitigen. Ressourcen der Recherche waren Stackoverflow, die Oracle Java Dokumentation, die JavaFX Dokumentation, sowie weitere Quellen aus dem Internet.

Dieser Fehler zeigte sich erneut bei der Erstellung eines neuen Dummies, bei dem es auch hier einen schwerwiegenden Fehler im Modul des JavaFX Frameworks gab.

Als Ausweichmöglichkeiten kam die Entscheidung, in Zusammenarbeit mit Fachinformatikern aus dem privaten Sektor, zunächst eine Erstellung mit Maven und Gradle.

Maven und Gradle sind Tools, welche die Implementierung von Libraries und wichtigen Funktionsweisen innerhalb der Projekteinstellung übernehmen.

Beide dieser Methoden fanden nach intensiver Recherche keine Möglichkeit die Funktionalität wieder herzustellen.

Neuausrichtung des Projekts

Durch die vorhandenen, schwerwiegenden Modulfehler war die Realisierung des Projekts mit JavaFX nicht möglich, auch aus zeitlichen Aspekten. In Einbezug der verbleibenden Zeit bis Projektende, folgte die teilweise Neuausrichtung des Projekts.

Dieses beinhaltete:





- Umstieg GUI auf Java Swing
 - o Keine Hintergrundgrafiken für Panels, Panes und Bedienelemente
- Kürzung der Funktionalität von vier Helden auf einen
 - o Sobald die komplette Funktionalität eines Helden funktioniert, ist die Implementierung von weiteren Helden durch den grundsätzlich modularen Aufbau ein geringerer Aufwand
- Entfernen des Antagonisten Systems

Einarbeitung Swing

Die Einarbeitung mit Java Swing war gut verständlich, aber dennoch zeitaufwendiger als ursprünglich geplant. Als Hintergrund für die Einarbeitung nutzte kam die Ressource der Webseite <https://www.java-tutorial.org/swing.html> zum Einsatz, ein umfangreiches und sehr gut beschriebenes Tutorial von Björn und Britta Petri, welche die essenziellen Bestandteile von Java Swing thematisierten. Darunter zählten Fenster und Dialoge, Menüs, Container, Bedienelemente, Event-Handling und Layouts.

Durch die Einarbeitung in Swing wurde ein guter und fundierter Einblick in dessen Funktionsweise, ebenso in das Benutzen und Verständnis der offiziellen Java Oracle Dokumentation gewährleistet.

Finales Swing GUI

	<div> <div>Laufende Quest Abgeschlossene Quests Neue Quests</div> <div></div> <div> <div>Abgeschlossen am: xx.xx.xxxx</div> <div>Erfahrungspunkte xxxxxx XP</div> </div> <div> <div>Quest abschließen</div> <div>Quest abbrechen</div> </div> </div>	<div> <div>Charakter</div> <div> <div>Inventar</div> <div></div> </div> <div> <div>Charakter Eigenschaften</div> <div> <div>Hauptattribut --</div> <div>Leben --</div> <div>Magie --</div> <div>Geschicklichkeit --</div> <div>Level --</div> <div>Erfahrungspunkte: -- / --</div> </div> </div> </div>
	<div> <div>Laufende Quest Abgeschlossene Quests Neue Quests</div> <div></div> <div> <div>Abgeschlossen am: ----</div> <div>Erfahrungspunkte ---- XP</div> </div> <div> <div>Quest abschließen</div> <div>Quest abbrechen</div> </div> </div>	<div> <div>Charakter</div> <div> <div>Inventar</div> <div></div> </div> <div> <div>Charakter Eigenschaften</div> <div> <div>Hauptattribut --</div> <div>Leben --</div> <div>Magie --</div> <div>Geschicklichkeit --</div> <div>Level --</div> <div>Erfahrungspunkte: -- / --</div> </div> </div> </div>
	<div> <div>Laufende Quest Abgeschlossene Quests Neue Quests</div> <div></div> <div> <div>Abgeschlossen am: ----</div> <div>Erfahrungspunkte ---- XP</div> </div> <div> <div>Quest abschließen</div> <div>Quest abbrechen</div> </div> </div>	<div> <div>Charakter</div> <div> <div>Inventar</div> <div></div> </div> <div> <div>Charakter Eigenschaften</div> <div> <div>Hauptattribut --</div> <div>Leben --</div> <div>Magie --</div> <div>Geschicklichkeit --</div> <div>Level --</div> <div>Erfahrungspunkte: -- / --</div> </div> </div> </div>
	<div> <div>Laufende Quest Abgeschlossene Quests Neue Quests</div> <div></div> <div> <div>Abgeschlossen am: ----</div> <div>Erfahrungspunkte ---- XP</div> </div> <div> <div>Quest abschließen</div> <div>Quest abbrechen</div> </div> </div>	<div> <div>Charakter</div> <div> <div>Inventar</div> <div></div> </div> <div> <div>Charakter Eigenschaften</div> <div> <div>Hauptattribut --</div> <div>Leben --</div> <div>Magie --</div> <div>Geschicklichkeit --</div> <div>Level --</div> <div>Erfahrungspunkte: -- / --</div> </div> </div> </div>

Programmierung

Methode Erstellung GUI

Um die Programmierung in Bezug auf objektorientierte Programmierung zu gestalten, erfolgte die Erstellung des GUI mithilfe von Methoden, die kaskadierend das GUI aufbauen.

Neben der Erstellung des mainFrames (den root Container der Applikation) wurde an einer Methode gearbeitet, deren Parameter die Eigenschaften des GridBagLayouts implementierten, sowie einen child Container erstellen, das Layout darin platzierte und den Container des vorhergehenden Layouts als parent Container definierten.

```
setGridBagLayout( ParentContainer,
    Anzahl der Grid X Punkte,
    Anzahl der Grid Y Punkte,
    Platzierung in X Grid des parent Containers,
    Platzierung in Y Grid des parent Containers,
    Ausbreitung X Achse,
    Ausbreitung Y Achse,
    ChildContainer) {

};
```

An sich hatte diese Methode ohne Probleme funktioniert, als Problem dabei ergab sich, dass kein Zugriff auf den parent Container gegeben war, da dieser bei einer noch nicht ausgeführten Methode eben noch nicht existierte. Ebenfalls der versuch diese Methode in eine Klasse zu implementieren und daraus n Objektinstanzen zu bilden, scheiterten am Fehlen der Accessibility des parent Containers. Änderungen und eine Auffrischung des grundsätzlichen Scopes in Java brachten keine Lösung des Problems.

Da die Lösung des Problems mehrere Stunden beanspruchte, fiel die Entscheidung die Erstellung des GUI via IntelliJ Swing UI Designer.

Erstellung GUI mit IntelliJ Swing UI Designer

Die Erstellung des GUI mittels IntelliJ Swing UI Designers ging ohne Probleme, dabei wurden die wichtigsten Funktionen und deren Interaktion miteinander suffizient verinnerlicht und die Implementierung des GUI wurde zügig umgesetzt, da das Panel für den Helden nur einmalig erstellt werden musste.

Für die Implementierung der weiteren Helden konnte auf Copy and Paste zurückgegriffen werden.

Auch galten die von JavaFX gesetzten Namenskonventionen. Da ab diesem Zeitpunkt klar wurde, dass das Fertigstellen der anderen Helden aus zeitlichen Gründen nicht stattfinden konnte, wurde auf die Verwendung der Namenskonvention anderer Helden verzichtet.

Einpflegen der Datenbank

```

1 DROP TABLE HELMET;
2
3 SELECT * FROM HELMET;
4
5 CREATE TABLE HELMET (
6     HELMET_ID int,
7     CLASS int,
8     LEVEL_REQ int,
9     MAIN_ATTR int,
10    HP int,
11    DEXTERITY int,
12    MAGIC int,
13    PICTURE_SM varchar(255),
14    PICTURE_BG varchar(255),
15    Name varchar(255),
16    DESCRIPTION varchar(400),
17    IN_INVENTORY boolean,
18    PRIMARY KEY (HELMET_ID)
19 );
20
21 INSERT INTO HELMET (HELMET_ID, CLASS, LEVEL_REQ, MAIN_ATTR, HP, DEXTERITY, MAGIC, PICTURE_SM, PICTURE_BG, NAME, DESCRIPTION)
22 VALUES
23 (1, 1, 4, 15, 12, 0, 0, 'Path to file', 'Path to file', 'Helm der Bequemlichkeit', 'Nur weil der Helm bequem ist, heißt das nicht, dass dich das aufhalten würde.',
24 (2, 1, 9, 20, 14, 0, 0, 'Path to file', 'Path to file', 'Helm der Abwägung', 'Dieser Helm sitzt in perfekter Balance. Und er kann dir sogar helfen den Weg zu finden.',
25 (3, 1, 14, 25, 16, 0, 0, 'Path to file', 'Path to file', 'Helm der Motivation', 'Motivation beginnt im Kopf!',
26 (4, 1, 19, 20, 18, 0, 0, 'Path to file', 'Path to file', 'Schützender Helm des Mächens', 'Dieser epische Helm motiviert dich nicht nur, nein, auch deine Frisur bleibt gleich!',
27 (5, 2, 4, 15, 12, 0, 17, 'Path to file', 'Path to file', 'Hut der Bequemlichkeit', 'Aha. Nicht machen not found!',
28 (6, 2, 9, 20, 14, 0, 19, 'Path to file', 'Path to file', 'Hut der Abwägung', 'In Futter dieses Huts ist ein kleiner Hamster eingeht, der dir jedes mal in den Kopf piekst wenn du Aufgaben nicht machen willst.',
29 (7, 2, 14, 25, 16, 0, 21, 'Path to file', 'Path to file', 'Hut der Motivation', 'Man munkelt, dass dieser Hut einmal die Katze des Dr. Suess war.',
30 (8, 2, 19, 30, 18, 0, 23, 'Path to file', 'Path to file', 'Heiliger Hut des Mächens', 'Ein Hut so legendär, selbst Mewtwo wäre stolz!',
31 (9, 3, 9, 20, 14, 0, 19, 'Path to file', 'Path to file', 'Kappe der Abwägung', 'Bei dieser Kappe brauchst du nicht die Farbe aussuchen, sie passt dir auch so.',
32 (10, 3, 14, 25, 16, 0, 21, 'Path to file', 'Path to file', 'Kappe der Motivation', 'Einst begleitete diese Kappe einen italienischen Klemmer.',
33 (11, 3, 19, 30, 18, 0, 23, 'Path to file', 'Path to file', 'Zerstörerischer Hut des Mächens', 'Gedanken, die dich daran hindern etwas zu tun, werden automatisch aus deinem Gehirn gelöscht!',
34 (12, 4, 14, 25, 16, 21, 0, 'Path to file', 'Path to file', 'Lederhelm der Motivation', 'Garantiert aus weichen Leder.',
35 (13, 4, 19, 30, 18, 21, 0, 'Path to file', 'Path to file', 'Willhelmselischer Helm des Mächens', 'Damit trifft du immer ins Rote.',

```

```

1 DROP TABLE LEVEL;
2
3 CREATE TABLE LEVEL (
4     LEVEL_ID int,
5     XP_NEXT_LEVEL int,
6     XP_TOTAL int,
7     PRIMARY KEY (LEVEL_ID)
8 );
9
10
11 INSERT INTO LEVEL (LEVEL_ID, XP_NEXT_LEVEL, XP_TOTAL) VALUES
12 (1, 150, 0),
13 (2, 300, 150),
14 (3, 450, 600),
15 (4, 600, 1200),
16 (5, 1500, 2700),
17 (6, 1800, 4500),
18 (7, 2100, 6600),
19 (8, 2400, 9000),
20 (9, 2700, 11700),
21 (10, 4500, 16200),
22 (11, 4950, 21150),
23 (12, 5400, 26550),
24 (13, 5850, 32400),
25 (14, 6300, 38700),
26 (15, 6750, 45450),
27 (16, 9600, 55050),
28 (17, 10200, 65250),
29 (18, 10800, 76050),
30 (19, 11400, 87450),
31 (20, 12000, 99450)

```

```

1 DROP TABLE QUEST;
2
3 SELECT * FROM QUEST;
4
5 CREATE TABLE QUEST (
6     QUEST_ID int,
7     QUEST_TYPE int,
8     CHARACTER_ID int,
9     COMPLETED_DATE date,
10    COMPLETED boolean,
11    TASK varchar(255),
12    ENEMY int,
13    PLACE int,
14    PRIMARY KEY (QUEST_ID)
15 );
16
17
18 INSERT INTO QUEST (QUEST_ID, QUEST_TYPE, CHARACTER_ID, COMPLETED_DATE, COMPLETED, TASK, ENEMY, PLACE) VALUES
19 (1, 1, 1, 2022-02-04, true, 'Geschirr spülen', 1, 1),
20 (2, 1, 1, 2022-12-04, true, 'Geschirr spülen', 1, 1)

```

```

1 DROP TABLE RING;
2
3 SELECT * FROM RING;
4
5 CREATE TABLE RING (
6     RING_ID int,
7     CLASS int,
8     LEVEL_REQ int,
9     MAIN_ATTR int,
10    HP int,
11    DEXTERITY int,
12    MAGIC int,
13    PICTURE_SM varchar(255),
14    PICTURE_BG varchar(255),
15    Name varchar(255),
16    DESCRIPTION varchar(400),
17    IN_INVENTORY boolean,
18    PRIMARY KEY (RING_ID)
19 );
20
21 INSERT INTO RING (RING_ID, CLASS, LEVEL_REQ, MAIN_ATTR, HP, DEXTERITY, MAGIC, PICTURE_SM, PICTURE_BG, NAME, DESCRIPTION)
22 VALUES
23 (1, 1, 4, 15, 12, 0, 0, 'Path to file', 'Path to file', 'Ring der Bequemlichkeit', 'Nur ein Ring mit einem lustigen druck!',
24 (2, 1, 9, 20, 14, 0, 0, 'Path to file', 'Path to file', 'Ring der Abwägung', 'Die Legende vermutet, man kann diesen Ring auch als Nasenring benutzen. Ob das wohl die kleinen Stacheln an dem Ring, verschreckten Hamster schon vom Weiten.',
25 (3, 1, 14, 25, 16, 0, 0, 'Path to file', 'Path to file', 'Ring der Motivation', 'Dieser Ring schützt dich vor roten Augen.',
26 (4, 1, 19, 30, 18, 0, 0, 'Path to file', 'Path to file', 'Schützender Ring der Selbstbestimmung', 'Dieser Ring ist einer der Punkte eines bekannten Auftragsverfalls.',
27 (5, 2, 4, 15, 12, 0, 17, 'Path to file', 'Path to file', 'Ring der Bequemlichkeit', 'Das Wort "Ring" wurde aus der altägyptischen kurel "helo" gebildet, was nachgemutet.',
28 (6, 2, 9, 20, 14, 0, 19, 'Path to file', 'Path to file', 'Ring der Motivation', 'Mit diesem Ring wirst du dich zu helfen Kitty's Seelengang',
29 (7, 2, 14, 25, 16, 0, 21, 'Path to file', 'Path to file', 'Heiliger Ring der Selbstbestimmung', 'Im Gegensatz zu einer Dieb, bist du mit diesem Ring wirklich selbstbestimmt.',
30 (8, 2, 19, 30, 18, 0, 23, 'Path to file', 'Path to file', 'Ring der Abwägung', 'Dieser Ring hilft CI den göttlichen Zug zu verfolgen.',
31 (9, 3, 9, 20, 14, 0, 19, 'Path to file', 'Path to file', 'Ring der Motivation', 'Dieser Ring bewahrt die Leidenschaft, wie ein gewisser kleiner Mensch an seinem Geburt.',
32 (10, 3, 14, 25, 16, 0, 21, 'Path to file', 'Path to file', 'Zerstörerischer Ring der Selbstbestimmung', 'Mit diesem Ring werden deine Gegner erschwert.',
33 (11, 3, 19, 30, 18, 0, 23, 'Path to file', 'Path to file', 'Ring der Motivation', 'Ein Ring der durch seine Motivation selbst zu berichten ist.',
34 (12, 4, 14, 25, 16, 21, 0, 'Path to file', 'Path to file', 'Willhelmselischer Ring der Selbstbestimmung', 'Der kleine Laserpointer am Ring macht dein Ziel noch präziser.',

```

Implementierung Datenbank

Die Implementierung der Datenbank ging schnell und ohne Probleme, Testfetches funktionierten reibungslos.

Für das Einpflegen der Daten in die Datenbank wurde auf die händische Methode umgestiegen. Zwar konnte mit Hilfe von exportierten Scripts aus der in DBeaver integrierten Datenbank jene Informationen direkt über das Datenbankpanel der H2 Datenbank eingefügt werden, die Entscheidung zur händischen Methode lag aber darin begründet, dass es für einen besseren Lerneffekt sorgen würde.

Die vorhanden Snippets aus den exportierten SQL Files des Datenbankviewers DBeaver konnten auf ihren Aufbau untersucht werden und es fand eine intensive Auseinandersetzung mit der Zusammensetzung von längeren SQL Codes statt.

Durch das „Zerlegen“ der Snippets wurde ein tieferer Einblick und Verständnis für SQL INSERT und CREATE Befehlen geliefert.

Methoden Kaskade

```
GUI main GUI = new GUI();  
mainGUI.setCharacterStats(1);
```

```
public void setCharacterStats(int CHARACTER_ID) {  
    SWING_OBJEKT_IN_GUI.setText(CONTROLLER_KLASSE.setCharacterHP(CHARACTER_ID);  
    ...  
    ...  
}
```

```
public static setCharacterHP(int CHARACTER_ID) {  
    CHARACTER_ID = CHARACTER_ID;  
    String characterHP = SQL_Controller.fetchInt("HP", "PLAYABLE_CHARACTER", CHARACTER_ID;  
    return characterHP;  
}
```

```
public static fetchInt(String COLUMN, String TABLE, int CHARACTER_ID) {  
    fetching snippet  
}
```

mehrfache Verwednung
des einzelnen
Codesnippets möglich

Grundlegender Aufbau

Der Grundlegende Aufbau der Applikation mit einen Klassen wurde in mehrere Packages, je nach Funktionsweise unterteilt:

- Charakter Controller
 - o Charakter_Stats_Controller: Funktionen, die für die Charakter Attribute zuständig sind, wie Leben, Hauptattribut, Level, aktuelle Erfahrungspunkte, etc.
 - o Inventory_Controller: Funktionen, die für das aktuelle Inventar zuständig sind
 - o Level-Up-Controller: Funktionen, die bei Level Up ausgelöst werden
- Quest Controller
 - o CQuest_Controller: Funktionen, die für die aktuell laufende Quest zuständig sind
 - o FQuest_Controller: Funktionen, die für die bereits abgeschlossenen Quests zuständig sind
 - o NQuest_Controller: Funktionen, die für die Erstellung einer neuen Quest zuständig sind
- General Controller:
 - o SQL-Controller: Methoden, die vorgefertigte SQL Queries besitzen
- GUI
 - o Alle Klassen bezüglich des GUI

Grundlegender Methoden Aufbau bei Queries

Es wurde eine Klasse erstellt, in der eine Methode als Basis Query Methode anzusehen ist und eine Subroutine für eine zweite Methode bildet, die die explizite Suche nach einem Datenbankeintrag an die Methode zurück gibt. Dies hat den Grund, dass die Grundlegende Query Methode sich über ca. 40 Zeilen erstreckt. So konnten Coderedundanzen vermieden werden und aus Performance Gründen konnte hier Zeit und Ressourcen eingespart werden.

Um dies zu verdeutlichen ein Beispiel:

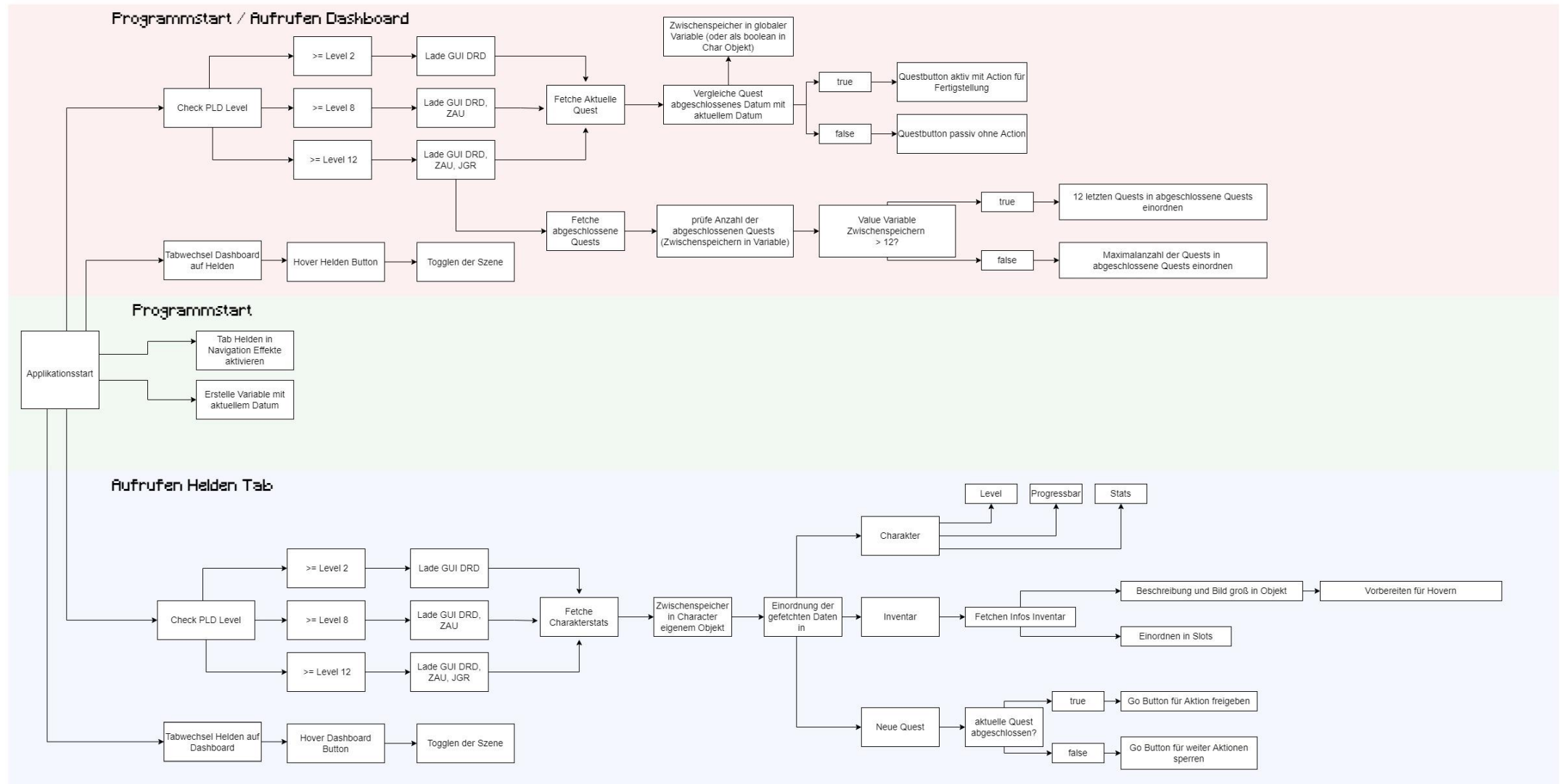
Das Setzen des Charakter Levels im Charakter Attribut Panel:

Setze Charakter Attribute in Panel, Parameter hier für ist die Charakter ID

Innerhalb dieser ersten Methode befindet sich die Methode, die als return value das gewünschte Attribut besitzt und in welcher die Query Methode sitzt.

Im letzten Bereich steht eine vorgefertigte Query Methode, deren Parameter eine genauere Suche nach spezifischen Datenbankeinträgen möglich macht.

Hintergrundfunktionen bei Start der Applikation



Dazu ebenfalls ein kleiner Vergleich:

Um ein Attribut aller vier Charaktere in die dafür vorgesehenen Panels einzuordnen, würden insgesamt vier Query Methoden und vier Return Methoden benötigt werden.

Durch das verwenden von Subroutinen reduziert sich damit die benötigte Anzahl von Methoden auf zwei. Daraus ergibt sich eine Code Ersparnis von ca. 135 Zeilen (3 x 40 Zeilen Query, 3 x 5 Zeilen Return) und trifft daher genau das Prinzip von effizienter und moderner Programmierung.

Fetchen und Einsetzen der Datenbankinformationen bei Applikationsstart

Folgende Methoden werden bei Programmstart benötigt:

- Quest
 - o Laufende Quest
 - Gibt es eine aktuelle Quest? Wenn ja, lade Informationen dazu und setze sie in den jeweiligen Bereich des GUI, wenn nein, graue Bereich aus
 - o Abgeschlossene Quest
 - Gibt es abgeschlossene Quests? Wenn ja, lade die Informationen dazu in die Liste, wenn nein, lade nichts
 - o Inventar
 - Welche Ausrüstungsgegenstände sind aktuell ausgerüstet? Lade Informationen wie Name, Beschreibung und Attribute und bereite für Hover Funktion vor
 - o Charakter Attribute
 - Lade aktuelle Charakterattribute, das Level, die aktuelle Anzahl der vorhandenen Erfahrungspunkte, die Anzahl der maximalen Erfahrungspunkte und setze diese in den jeweiligen Bereich des GUI, bzw. lade die Progress Bar

(jegliche Quest relevanten Funktionen konnten dennoch aus zeitlichen Gründen nicht implementiert werden)

Funktionsweise Inventar

Die einzelnen Items sind im Panel nach Waffe, Helm, Rüstung, Ohrring und Ring aufgeteilt und sobald die Maus über das jeweilige Item hovers, werden via `MouseEvent mouseEntered` die jeweiligen Informationen aus der Datenbank gefetcht und die leeren Labels via `Label.setText()` eingefügt.

Verlässt die Maus das betreffende Item, werden die Strings innerhalb der Labels via `MouseEvent mouseExited` „geleert“, in dem der String auf `null` gesetzt wird.

Funktionsweise Charakter Attribute

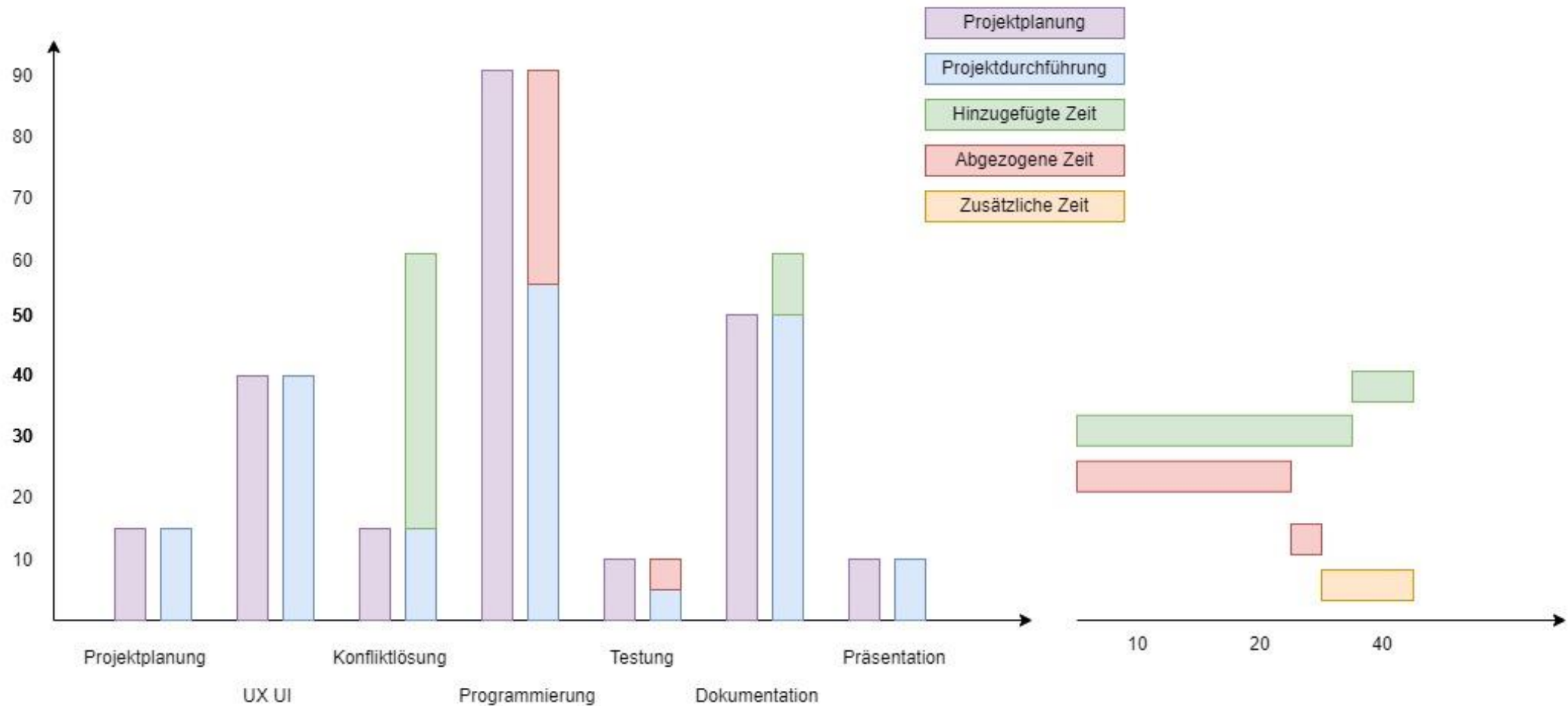
Charakter Attribute, Level Anzeige, aktuelle und maximale Erfahrungspunkte

Bei Programmstart werden die jeweiligen Informationen aus der Datenbank gefetcht und in die Labels eingefügt.

Progress Bar

Anhängig vom jeweiligen Level des Charakters werden aus der Datenbank die maximalen Erfahrungspunkte gespeichert und mittels `setMaxValue()` als Ende der Progress Bar gesetzt, sowie die aktuellen Erfahrungspunkte als `setValue()` definiert.

Zeitliche Verteilung Soll und Ist Vergleich



Vorzeitiges Beenden des Projektes aufgrund von Zeitmangel

Aufgrund der in dieser Dokumentation beschriebenen Problematiken musste das Projekt vorzeitig beendet werden und einige Features und Programmfunktionen konnten nicht implementiert werden.

Gestrichene und nicht hinzugefügte Applikationsfeatures

Reduktion von fünf Helden auf einen Held

Antagonisten System

JavaFX GUI

Generieren und Setzen des GridBagLayouts via kaskadierender Methoden

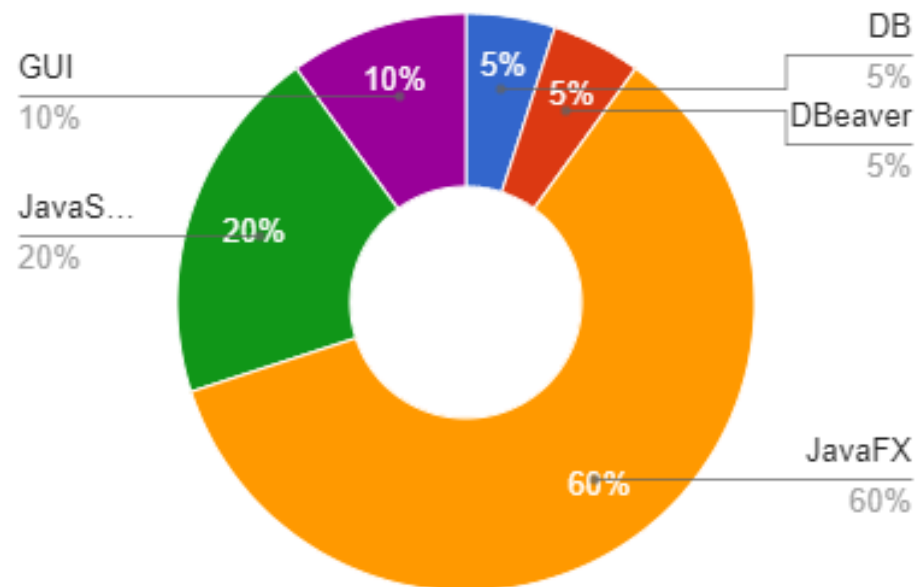
komplette Funktion des Questsystems

Projektanalyse

Vergleich Stundenverteilung

Projektplanung (inkl. Datenbankgestaltung)	15 h	Projektplanung (inkl. Datenbankgestaltung)	15 h	
UX / UI Konzeptionierung und Gestaltung	40 h	UX / UI Konzeptionierung und Gestaltung	40 h	
Recherche zur Konfliktlösung	15 h	Recherche zur Konfliktlösung	60 h	+ 45 h
Programmierung	90 h	Programmierung	55 h	- 45 h
Testung / Bug Fixing	10 h	Testung / Bug Fixing	5 h	+ 5 h
Dokumentation	50 h	Dokumentation	60 h	+ 5 h
Präsentation	5h	Präsentation	5h	
	224 h		234 h	+ 5 h

Problemstellung und Impact auf Projektablauf



Problemfelder

Folgende Problemfelder taten sich während der Ausarbeitung des Projektes auf:

Nicht ausgereiftes Datenbanklayout

Aufgrund mangelnder Praxis in der Erstellung von Datenbanken gestaltete sich die praktische Umsetzung des Datenbanklayouts als schwieriger als erwartet.

Das Bewusstsein für die logische Verknüpfung von Tabellen und deren Zugriff aufeinander mittels Primary Key und Foreign Key, sowie die Vermeidung von Datenredundanzen konnte nur mühsam eingearbeitet werden.

Durch das feinere Aufteilen der Datenbank in seine einzelnen Bestandteile konnte eine bessere Übersicht über die Zusammenhänge innerhalb der Datenbank geschaffen werden.

Durch die Erstellung mehrerer, immer feinerer Versionen des Datenbanklayouts wurde die letztendliche Finalisierung vereinfacht. Somit kann diese Methode der Datenbankgestaltung für spätere Anwendungen Anklang finden.

Dennoch lässt sich sagen, dass der allgemeine Impact auf das Zeitmanagement eher gering ausfiel und das Kosten / Nutzen Verhältnis dennoch sehr hoch war.

Datenbankviewer DBeaver

Die Idee zur vereinfachten Darstellung und vereinfachten Arbeit an Datenbanken mittels eines Datenbankvisualisierungstools stellte sich zu Beginn als keine schlechte Idee heraus.

Das Programm DBeaver bot ebenso die Möglichkeit eine fundierte Einsicht über Verhalten der Datenbank, optische Visualisierung der zugehörigen relationalen Entitäten, sowie die übersichtliche Darstellung von SQL Code und so war die Grundidee mit einem Gewinn an Erfahrung für tiefere Einblicke in Datenbankgestaltung, -verwaltung und SQL Coding zu erhalten. Dennoch ergab sich das Problem, dass die Daten innerhalb der Datenbank und DBeaver veränderbar waren, jedoch nicht auf die tatsächliche Datenbank übertragen wurden.

Das Fixen des Problems stellte sich als längere Recherche heraus, deren Ziel frustan war.

Es kam daher zu einem Breakpoint im projektorientierten Denken, komplett auf dieses Programm zu verzichten, die Datenbankeinträge händisch zu implementieren und damit einen besseren Überblick, auch über SQL Grundlagen zu erhalten.

Für das Einpflegen der Daten in die Datenbank wurde auf die händische Methode umgestiegen. Zwar konnte mit Hilfe von exportierten Scripts aus der in DBeaver integrierten Datenbank jene Informationen direkt über das Datenbankpanel der H2 Datenbank eingefügt werden, die Entscheidung zur händischen Methode lag aber darin begründet, dass es für einen besseren Lerneffekt sorgen würde.

Die vorhandenen Snippets aus den exportierten SQL Files des Datenbankviewers DBeaver konnten auf ihren Aufbau untersucht werden und es fand eine intensive Auseinandersetzung mit der Zusammensetzung von längeren SQL Codes statt.

Durch das „Zerlegen“ der Snippets wurde ein tieferer Einblick und Verständnis für SQL INSERT und CREATE Befehlen geliefert.

JavaFX GUI Implementierung

Das erste Implementieren des JavaFX Frameworks zeigte sich zunächst komplikationslos, alle Grafiken wurden übernommen, die FX:ID des FXML Files waren ebenfalls noch vorhanden und alle Positionsangaben und Größeneinstellungen der Grafiken und Bedienelemente waren alle vorhanden.

Dennoch etablierte sich bei der Erstellung der ersten Funktionen das Problem, dass sich das GUI bei Veränderungen nicht anpasste, das XML File plangemäß auf die

Funktionen reagierte und der Rein Code im Terminal keine Fehler anzeigte.

Es stellte nach langer und intensiver Recherche heraus, dass es einen Fehler in der Abfrage der Modul Library gab. Die Prüfung verschiedenster Einstellungen (Einbettung der Library, Konnektivität der Module, Überprüfung der AusführungsVM und konnte diesen Fehler nicht beseitigen. Ressourcen der Recherche waren Stackoverflow, die Oracle Java Dokumentation, die JavaFX Dokumentation, sowie weitere Quellen aus dem Internet.

Dieser Fehler zeigte sich erneut bei der Erstellung eines neuen Dummies, bei dem es auch hier einen schwerwiegenden Fehler im Modul des JavaFX Frameworks gab.

Als Ausweichmöglichkeiten kam die Entscheidung, in Zusammenarbeit mit Fachinformatikern aus dem privaten Sektor, zunächst eine Erstellung mit Maven und Gradle.

Maven und Gradle sind Tools, welche die Implementierung von Libraries und wichtigen Funktionsweisen innerhalb der Projekteinstellung übernehmen.

Beide dieser Methoden fanden nach intensiver Recherche keine Möglichkeit die Funktionalität wieder herzustellen.

Umstieg auf Java Swing

Als klar wurde, dass JavaFX eine Sackgasse war, wurde dennoch recht schnell deutlich, dass ein Umstieg auf Java Swing stattfinden musste, da bereits zu diesem GUI Framework eine rudimentäre Erfahrung vorausging.

Die Abarbeitung der Tutorialreihe zu Java Swing war sehr deutlich und klar beschrieben.

Im Allgemeinen war der Erfahrungsgewinn mit der Einarbeitung auf Java Swing enorm, auch in Sachen der Verständlichkeit von Methoden, dem Lesen und interpretieren der offiziellen Java Oracle Dokumentation.

Damit war Kosten- Nutzenvergleich durchaus lohnenswert und ergab einen enormen Erfahrungsgewinn.

Methode zur automatisierten Erstellung des GUI

Um die Programmierung in Bezug auf objektorientierte Programmierung zu gestalten, erfolgte die Erstellung des GUI mithilfe von Methoden, die kaskadierend das GUI aufbauen.

Neben der Erstellung des mainFrames (den root Container der Applikation) wurde an einer Methode gearbeitet, deren Parameter die Eigenschaften des GridBagLayouts implementierten, sowie einen child Container erstellen, das Layout darin platzierte und den Container des vorhergehenden Layouts als parent Container definierten.

An sich hatte diese Methode ohne Probleme funktioniert, als Problem dabei ergab sich, dass kein Zugriff auf den parent Container gegeben war, da dieser bei einer noch nicht ausgeführten Methode eben noch nicht existierte. Ebenfalls der Versuch diese Methode in eine Klasse zu implementieren und daraus n Objektinstanzen zu bilden, scheiterten am Fehlen der Accessibility des parent Containers. Änderungen und eine Auffrischung des grundsätzlichen Scopes in Java brachten keine Lösung des Problems.

Da die Lösung des Problems mehrere Stunden beanspruchte, fiel die Entscheidung die Erstellung des GUI via IntelliJ Swing UI Designer.

Das Kosten- Nutzenverhältnis aus dieser Problemstellung war jedoch rein zeitlich geprägter Natur. Wäre dem Projekt mehr Zeit zur Verfügung gestanden, hätte dieses Problem sicherlich gelöst werden können.

Gelernte Inhalte

Organisierte Planung

Verwendung einer Roadmap als detaillierte Übersicht über Ist und Soll Zustand

Verwenden und Erstellen von detaillierten Diagrammen zu Visualisierungszwecken

Wochenweise Planung als flexible Methode

Einarbeitung in Frameworks

Erkennen von projektspezifischen Kriterien

Recherche nach passenden Frameworks

Lesen und Verstehen der Dokumentation

Einfügen in Programmstruktur

Verständnis über eigene Framework Syntax

Error Handling

tieferes Verständnis für XML Syntax und Semantik

Rudimentäre Verwendung von Maven und Gradle

Erstellung von Dummy Umgebungen zur Übung

Entscheidungsfindung

Abwägung zwischen Vor- und Nachteilen

Abwägung im Bezug auf Produktivität und Effizienz

Datenbankgestaltung

Konzeptionierung von Datenbanken

Vertiefung Normalisierung und ERM

Datenbankvisualisierung

Vertiefung SQL Syntax

Programmierung

- Vertiefung von Methoden und Parameter
- Verschachtelung von Methoden
- Vertiefung objektorientierte Programmierung
- Einbettung von Plug Ins und Frameworks
- Vertiefung von Programmstrukturen
- Programmorganisation

Error Handling

- Erkennen von Fehlern
- Recherche von Fehlern
- Umgang mit Fehlern
- Differenzierung von Recherchequellen

Projektmanagement

- Umgang mit Fehlern
- Umgang mit Rückschlägen
- Umgang mit Neuausrichtung des Projekts
- Umgang mit freiem Zeitmanagement