Technische Hochschule
Brandenburg
University of
Applied Sciences

Fachbereich Wirtschaft

MASTERARBEIT

Konzeption eines JavaFX-Frameworks zur Erstellung von Gantt-Diagrammen - Anforderungen, Architektur und Implementierung eines Prototyps

Vorgelegt von:

Dallysse Laure Djouhou Tamdjo

am:

26. Januar 2023

zur

Erlangung des akademischen Grades

Master of Science

(M.sc.)

Erstbetreuerin:

Prof. Dr. Winfried Pfister

(Technische Hochschule Brandenburg)

Zweitbetreuerin:

Prof. Dr. Kai Jander

(Technische Hochschule Brandenburg)

Ich möchte mich bei meinen Betreuerinnen, Prof. Dr. Winfried Pfister und Prof. Dr. Kai Jander, die mich bei der Erstellung dieser Masterarbeit unterstützt haben. Ihre Geduld mit mir, Ihre Vorschläge und Ihre Verfügbarkeit waren mir eine große Hilfe.

Anschließend möchte ich Gott und all den liebevollen Menschen danken, die mich ermutigt haben.

.....

Inhaltsverzeichnis

1 Einführung

1.1 Motivation und Aufgabenstellung

Die vorliegende Masterarbeit befasst sich mit dem Entwurf und der Erstellung eines JavaFX-Framework-Prototyps.

Frameworks sind vorgefertigte Programmierstrukturen, die die Entwicklung der eigentlichen Anwendung beschleunigen sollen und in der Regel fertige Lösungen für typische Aufgaben anbieten, in unserem Fall für den Entwurf eines Gantt-Diagramms. Jeder Softwareentwickler hat wahrscheinlich schon einmal Frameworks verwendet, manchmal ohne es zu wissen. Diese werden häufig bei der Entwicklung von Anwendungen mit grafischen Benutzeroberflächen (sog. GUI-Frameworks: z. B. Angular[1], JavaFX) eingesetzt und bieten dem Softwareentwickler Komponenten und Prozesse zur visuellen Darstellung von Daten. Frameworks unterscheiden sich somit von Klassenbibliotheken, da sie nicht nur eine Sammlung einzelner Komponenten sind, sondern auch Strukturen und Prozesse anbieten.

Das Gantt-Diagramm ermöglicht es, die verschiedenen Aktivitäten, aus denen sich ein Projekt zusammensetzt, zeitlich darzustellen. Das Gantt-Diagramm wird in verschiedenen Bereichen eingesetzt, z. B. bei der Terminplanung, im Projektmanagement und bei der Planung von Reihenfolgenplanung. Zwei der Hauptziele des Ganttchart, ist die Verbesserung der Planung und eine Kommunikation über den Kalender. Mithilfe der Tabelle lassen sich die Termine für die Durchführung eines Projekts bestimmen und der Fortschritt oder die Verzögerung der Arbeit erkennen. Die Methode besteht darin, die auszuführenden Aktivitäten im Gantt-Diagramm in der durch die Priorität bestimmten Reihenfolge zu setzen und dabei die noch verfügbaren Ressourcen zu berücksichtigen. In einem Gantt-Diagramm werden die in Monaten, Wochen oder Tagen ausgedrückten Zeiteinheiten auf der Abszisse und die verschiedenen Aktivitäten auf der Ordinate dargestellt.

Das hier behandelte Problem entstand also aus dem Wunsch, die bei der Erstellung von Gantt-Diagrammen auftretenden Probleme schnell und effizient mit einem Open-Source-Mittel lösen zu können. Das hier vorgestellte Projekt geht über die Funktionen einer herkömmlichen Programmbibliothek hinaus und bietet darüber hinaus vordefinierte Prozesse und Problemlösungen. Das daraus resultierende Framework definiert somit einen geeigneten Tool, mit dem Gantt-Diagramme entwickelt werden können.

[1] https://angular.io/guide/what-is-angular

Ziel dieser Masterarbeit ist es, ein Konzept zu entwickeln, das definiert, welche Funktionen bei der Erstellung eines Gantt-Diagramms mit einem JavaFX-Framework vorhanden oder nicht vorhanden, aber dennoch relevant sind, was der Benutzer von JavaFX-Frameworks bei der Erstellung eines Gantt-Diagramms erwarten kann und schließlich die Implementierung eines prototypischen Open-Source-JavaFX-Frameworks für die Erstellung von Gantt-Diagrammen.

1.2 Aufbau des Dokuments

Die Arbeit ist in drei Hauptteile gegliedert. Im ersten Teil werden die Grundlagen, Modelle und Konzepte von Gantt Diagramm und der Mehrwert von Framework insbesondere des JavaFX-Frameworks erläutert. Im zweiten Teil geht es um die Implementierung des Prototyps des Frameworks zur Erstellung von Gantt-Diagrammen, deren allgemeinen bzw. speziellen Anforderungen, sowie Architekturentwurf und Realisierung mit JavaFX. Und im dritten Teil wird eine Bewertung der erstellten Prototypen vorgenommen. In diesem Teil wird gezeigt, wie der Prototyp aussieht, was funktioniert hat und was nicht.

Grundlegende Kenntnisse in Java und JavaFX sind erforderlich, um die Implementierung und die hier vorgestellte und die Codefragmente zu verstehen, zusätzlich kommt JavaFX 19 zum Einsatz. Als Programmiersprache wird Java verwendet.

2 Grundlagen und Begriffe

2.1 Java und JavaFX

JavaFX ist selbstverständlich Teil der seit vielen Jahren etablierten Java-Plattform. Um JavaFX zu verstehen oder generell mit ihm umgehen zu können, sollten diese zumindest die Grundlagen von Java selbst kennen. Lassen im Folgenden also zunächst einen kurzen Blick auf seine Struktur und die Geschichte von Java insgesamt werfen.

2.1.1 Was ist Java?

Einerseits ist Java eine Programmiersprache, andererseits bezeichnet es aber auch eine ganze Plattform zur Ausführung von stabilen, sicheren und leistungsfähigen Programmen unabhängig vom zugrunde liegenden Betriebssystem. Im Allgemeinen spricht man von der Java-Technologie. Dabei handelt es sich um eine Reihe von Spezifikationen, die einerseits eine Programmiersprache und andererseits verschiedene Ausführungsumgebungen einschließlich umfangreicher Bibliotheken für Computerprogramme definieren. Java ist eine einfache, objektorientierte, dezentralisierte, interpretierte, stabile, sichere, architekturunabhängige, tragbare und dynamische Sprache.

2.1.2 Was ist JavaFX?

JavaFX soll Swing in Java-Anwendungen als Framework für die grafische Benutzeroberfläche ersetzen. JavaFX stellt seine eigenen Komponenten bereit. JavaFX unterstützt verschiedene Betriebssysteme, darunter Windows, Linux und Mac OS. Die Skriptsprache von JavaFX eignet sich für Komplexe Grafiken und reichhaltige Benutzeroberflächen, da sie die Prism-Grafikpipeline zum Rendern verwendet. Sie enthält reichhaltige Grafikbibliotheken, mit denen sich Anwendungen mit grafischer Benutzeroberfläche (GUI) erstellen lassen.

JavaFX ist eine Java-Bibliothek, die zur Entwicklung von Desktop-Anwendungen sowie von Rich Internet Applikation (RIAs) verwendet wird. Mit JavaFX erstellte Anwendungen können auf verschiedenen Plattformen laufen, darunter das Web, Mobiltelefone und Desktops. Ein Vorteil dieses Frameworks ist die

weit verbreitete Java-Umgebung. Neben den archivierten Funktionen stellt JavaFX den Benutzern auch Schnittstellen und Vorlagen zur Verfügung, mit denen sie statische Grafiken, Animationen und komplexe Benutzeroberflächensteuerungen kombinieren können. JavaFX ist auch mit mehreren Technologien kompatibel, die auf der Java Virtual Machine (JVM) basieren, wie z. B. Java, Groovy,kotlin und JRuby. JavaFX bietet jedoch auch volle Funktionalität für Benutzer, die nur JavaFX ohne weitere Frameworks verwenden möchten.

JavaFX wurde die erstmals auf der JavaOne-Konferenz von Sun im Jahr 2007 als neue Skriptplattform für Web-, Desktop- und mobile Anwendungen angekündigt. JavaFX wurde von Chris Oliver entwickelt. Ursprünglich hieß das Projekt Form Follows Functions (F3). Es sollte reichhaltigere Funktionen für die Entwicklung von GUI-Anwendungen bieten. Später erwarb Sun Micro-systems das F3-Projekt unter dem Namen JavaFX im Juni 2005 [Jai11]. Sun Micro-systems kündigte es 2007 auf der W3-Konferenz offiziell an. Ende 2007 und Anfang 2008 erschienen die ersten offiziellen Vorabversionen und um den Jahreswechsel 2008 bis 2009 wurde die erste endgültige Version geschrieben.

In den frühen Versionen stellte eine integrierte Skriptsprache namens JavaFX Script als zentraler Kern der gesamten Technologie die Mittel zur Verfügung, um allgemeine und leistungsfähige visuelle Anwendungen auf Java-Basis zu erstellen. Obwohl sehr einfache Anwendungen mit der Konsolenausgabe erstellt werden können, war JavaFX Script von Anfang an darauf ausgelegt, den kreativen Prozess der Erstellung von Benutzerschnittstellen zu rationalisieren. Dieser Ansatz mit JavaFX Script kann jedoch als gescheitert angesehen werden, und die Skriptsprache wird ab Version 2.0 nicht mehr in JavaFX bereitgestellt[Ste14].

JavaFX ist ab Version 2.0 referenziert. Derzeit ist die letzte Version von JavaFX JavaFX 19, die am 19. September 2022 veröffentlicht wurde. Neuere Versionen von JavaFX ermöglichen die Verwendung von JavaFX neben HTML5- und JavaScript-Frameworks anstelle von Plug-ins, die von den meisten Geräten und Browsern nicht mehr unterstützt werden.

2.1.3 Schlüsselfunktionen von JavaFX

Die Tabelle 2.1 beschreibt die Hauptfunktionen, die in JavaFX 8 und späteren Versionen enthalten sind.

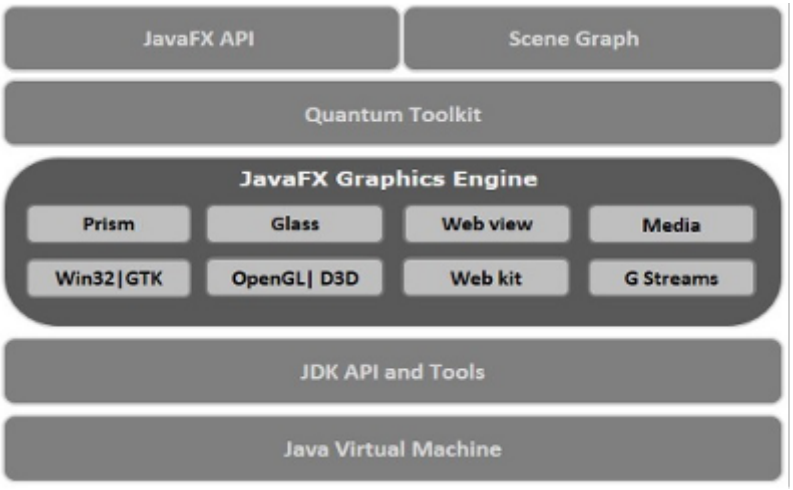
Tabelle 2.1: Schlüsselfeatures von JavaFX

Merkmale	Beschreibung
Java API	Java API ist eine Java-Bibliothek, die aus Klassen und Schnittstellen besteht, die in nativem Java-Code geschrieben sind. Die APIs sind als benutzerfreundliche Alternative zu den Sprachen der Java Virtual Maschine (Java VM), wie JRuby und Scala, gedacht.
FXML und Scene Builder	FXML und Scene Builder ist eine deklarative, XML-basierte Auszeichnungssprache zum Aufbau einer Benutzeroberfläche für JavaFX-Anwendungen. Ein Designer kann in FXML codieren oder JavaFX Scene Builder verwenden, um die grafische Benutzeroberfläche (GUI) interaktiv zu gestalten. Scene Builder erzeugt ein FXML-Markup, das auf eine IDE portiert werden kann, wo ein Entwickler die Geschäftslogik hinzufügen kann.
WebView	WebView Eine Webkomponente, die die WebKitHTML-Technologie nutzt, um die Integration von Webseiten in eine JavaFX-Anwendung zu ermöglichen. In WebView ausgeführtes JavaScript kann Java-APIs aufrufen, und Java-APIs können in WebView ausgeführtes JavaScript aufrufen.
Interoperabilität mit Swing	Bestehende Swing-Anwendungen können mit neuen JavaFX-Funktionen aktualisiert werden, z. B. mit der Wiedergabe von Rich Graphik Media und eingebetteten Webinhalten.
Integrierte UI- und CSS-Steuerelemente	JavaFX bietet alle wichtigen UI-Steuerelemente, die für die Entwicklung einer vollständigen Anwendung erforderlich sind. Die Komponenten lassen sich mit Standard-Webtechnologien wie CSS beeinflussen.
Canvas API	Canvas API ermöglicht das direkte Zeichnen in einem Bereich der JavaFX-Szene, der aus einem grafischen Element (Knoten) besteht.
Unterstützung von Multitouch	JavaFX unterstützt Multitouch-Operationen, je nach den Möglichkeiten der zugrunde liegenden Plattform.
Reichhaltiger API-Satz JavaFX	bietet einen umfangreichen API-Satz zur Entwicklung von GUI-Anwendungen
Integrierte Grafikbibliothek	Für 2D- und 3D-Grafiken steht ein integrierter Satz von Klassen zur Verfügung.
Hardwarebeschleunigte Grafikpipeline	JavaFX-Grafiken basieren auf der Grafik-Rendering-Pipeline (Prism). JavaFX bietet flüssige Grafiken, die von Prism schnell gerendert werden, wenn es mit einer unterstützten Grafikkarte oder Grafikprozessoreinheit (GPU) verwendet wird. Wenn ein System nicht über einen der empfohlenen, von JavaFX unterstützten Grafikprozessoren verfügt, verwendet Prism standardmäßig das Java 2D-Software-Stack.
Hochleistungsfähige Medien-Engine	Die Medienpipeline unterstützt die Wiedergabe von Web-Medieninhalten. Sie bietet einen stabilen Medienrahmen mit niedriger Latenz, der auf dem GStreamer-Medienrahmen basiert.
Modell für die Bereitstellung von eigenständigen Anwendungen	Eigenständige Anwendungspakete enthalten alle Ressourcen der Anwendung und eine private Kopie der Java- und JavaFX-Laufzeitmaschinen. Sie werden als native Installationspakete verteilt und bieten die gleiche Installations- und Starterfahrung wie native Anwendungen für dieses Betriebssystem.

2.1.4 JavaFX Architektur

JavaFX hat verschiedene integrierte Komponenten, die miteinander verbunden sind. Es enthält einen reichhaltigen Satz von APIs, die mehr als ausreichend sind, um reichhaltige Internetanwendungen zu entwickeln, die auf vielen Plattformen einheitlich funktionieren. Die Abbildung2.1 zeigt die Architektur der JavaFX-API, sie gibt eine deutliche Übersicht über die Komponenten, die die JavaFX-API unterstützen.

Abbildung 2.1: Übersicht der JavaFX Architektur[tut11a]



2.1.5 JavaFX API

JavaFX bietet eine umfassende API mit einem reichen Satz an Klassen und Schnittstellen, um GUI-Anwendungen mit reichhaltiger Grafik zu bauen. Wichtige Pakete dieser API sind:

Tabelle 2.2: Die Liste der Pakete der JavaFX-API

Merkmale	Beschreibung
javafx.animation	enthält Klassen, mit denen JavaFX Knoten übergangsbasierte Animationen wie Füllen, Blenden, Drehen, Skalieren und Translation hinzugefügt werden können. Die Klasse Timeline stellt eine Animationszeitleiste dar. Auf der Zeitleiste können Keyframes hinzugefügt werden, zwischen denen die Animationsfunktionen interpoliert werden.
javafx.application	enthält eine Reihe von Klassen, die für den Lebenszyklus von JavaFX-Anwendungen zuständig sind.
javafx.css	enthält Klassen, mit denen JavaFX-GUI-Anwendungen ein CSS-ähnlicher Stil hinzugefügt werden kann. Es gibt mehrere verschiedene Methoden, um einen CSS-Stil auf eine JavaFX-Komponente anzuwenden. Diese Methoden lauten: JavaFX default CSS stylesheet, Scene specific CSS stylesheet, Parent specific CSS stylesheet und Component style property
javafx.event	enthält Klassen und Schnittstellen zur Ausgabe und Verwaltung von JavaFX-Ereignissen.
javafx.geometry	enthält Klassen zum Definieren von 2D-Objekten und zum Ausführen von Operationen mit ihnen.
javafx.scene	dieses Paket enthält Klassen und Schnittstellen zur Unterstützung des Scene Graph. Darüber hinaus stellt es auch Unterpakete wie z. B. javafx.scene.canvas, javafx.scene.chart, javafx.scene.control, javafx.scene.effect, javafx.scene.image, javafx.scene.input, javafx.scene.layout, javafx.scene.media, javafx.scene.paint, javafx.scene.shape, javafx.scene.text, javafx.scene.transform, javafx.scene.web zur Verfügung. Die JavaFX-Scene enthält alle visuellen JavaFX-GUI-Komponenten in JavaFX Scene graph.

2.1.6 Scene Graph

In JavaFX wurden GUI-Anwendungen mithilfe eines Scene Graph codiert. Ein Scene Graph ist der Ausgangspunkt für den Aufbau der GUI-Anwendung. Er enthält die Primitive der Anwendung (GUI), die als Knoten bezeichnet werden.

Ein Knoten ist ein visuelles und grafisches Objekt, das Folgendes umfassen kann Ein geometrische (grafische) Objekte (2D und 3D) wie z.B. Kreis, Rechteck, Polygon, ein Steuerelemente für Benutzeroberflächen wie Schaltflächen, Kontrollkästchen, Auswahlboxen, Textfelder. Sowie Container (Layoutbereiche) wie z. B. Randbereich, Rasterbereich, Flussbereich und Medienelemente wie Audio-, Video- und Bildobjekte.

Im Allgemeinen bildet eine Sammlung von Knoten einen Scene Graph. Alle diese Knoten sind in einer hierarchischen Ordnung angeordnet.

Jeder Knoten im Scene Graph hat nur ein Elternteil, und der Knoten, der kein Elternteil hat, wird

Root Node genannt. In ähnlicher Weise hat jeder Knoten ein oder mehrere Kinder, und der Knoten ohne Kinder wird Leaf Node genannt; ein Knoten mit Kindern wird Branch Node genannt.

Eine Knoteninstanz kann nur einmal zu einem Scene Graph hinzugefügt werden. Knoten in einem Scene Graph können Effekte, Deckkraft, Transformationen, Ereignis Behandler und anwendungsspezifische Zustände haben.

2.1.7 Graphics Engine

Die JavaFX-Grafik-Engine liefert die grafische Unterstützung für die Grafikkomponente der Szene. Graphics Engine unterstützt in der Regel 2D- und 3D-Grafiken. Sie sorgt auch für das Software-Rendering, wenn die auf dem System vorhandene Grafikhardware kein hardwarebeschleunigtes Rendering unterstützen kann. Die beiden Grafikbeschleunigungs-Pipelines von JavaFX sind Prism und Quantum Tool Kit.

Die Prism ist ein hochleistungsfähiges Hardwarebeschleunigungsmedium für Grafiken, das das Rendern von 2D- und 3D-Grafiken durchführen kann. Prism implementiert je nach Plattform verschiedene Möglichkeiten, Grafiken zu rendern.

Das Quantum Toolkit wird verwendet, um das Prism Toolkit und Glass Windowing zu verbinden und sie für die oberen Schichten des Stacks verfügbar zu machen.

Das Glass Windowing Tool Kit existiert auf der untersten Ebene des JavaFX-Grafikstapels. Es kann als plattformabhängige Schicht betrachtet werden, die als Schnittstelle zwischen der JavaFX-Plattform und dem nativen Betriebssystem fungiert. Es ermöglicht den Zugriff auf native operative Funktionen wie Fensterverwaltung, Timer oder Ereignisverwaltung.

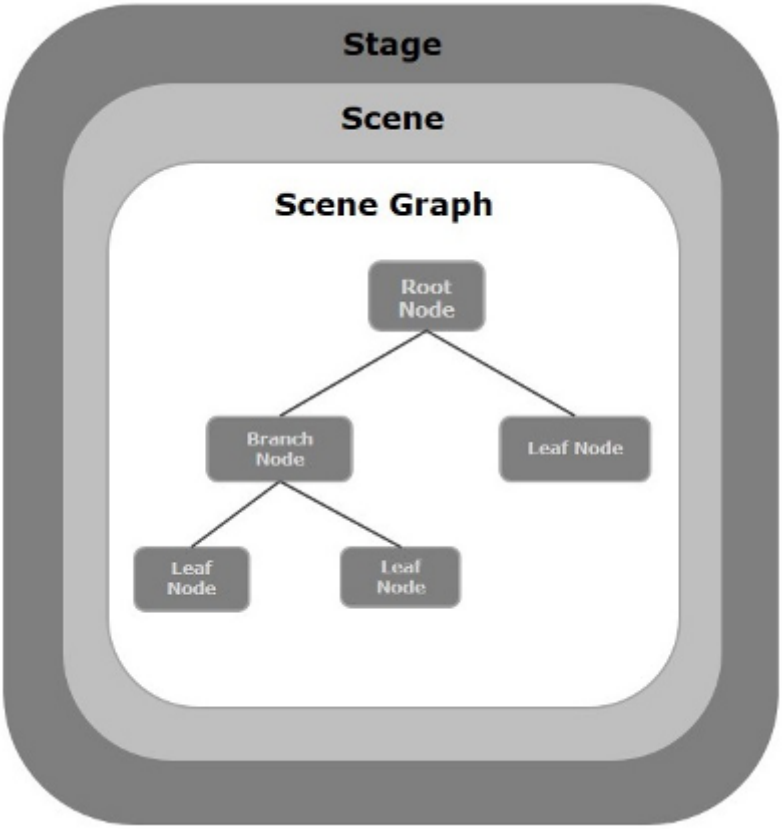
Die Komponente Web Engine oder WebView ermöglicht die Integration von Webinhalten in JavaFX-Anwendungen. Dies umfasst HTML-Rendering auf der Basis der Webkit-Engine und hardwareabhängiges Rendering über Prism. Die Web-Engine-Komponente bietet die Möglichkeit, auf das DOM (Document Object Model) zuzugreifen und es zu bearbeiten. Der WebView unterstützt auch HTML5, CSS, JavaScript und SVG.

Medien-Engine bietet Werkzeuge zur Erstellung von Multimedia-Anwendungen, die die Wiedergabe von Medien im Desktopfenster oder auf einer Webseite auf unterstützten Plattformen ermöglichen. Die Medien-Engine von JavaFX basiert auf einer Open-Source-Engine, die als G Streamer bekannt ist und die Wiedergabe von Video- und Audioinhalten unterstützt. Das Paket javafx.scene.media enthält alle Klassen und Schnittstellen, die JavaFX-Anwendungen mit Medienfunktionen ausstatten können.

2.1.8 JavaFX Anwendung Struktur

In diesem Teil werden wir die Struktur einer JavaFX-Anwendung im Detail besprechen, wie man eine JavaFX-Anwendung erstellt. Im Allgemeinen besteht eine JavaFX-Anwendung aus drei hierarchischen Hauptkomponenten, nämlich Stage, Scene und Node, wie in der Abbildung (2.2) dargestellt.

Abbildung 2.2: JavaFX Anwendung Struktur[tut11b]



Eine technische Beschreibung von Scene Graph und Nodes wurde in Kapitel 2.1.10 gegeben.

2.1.9 Stage

Eine Stage ist der Hauptcontainer und der Einstiegspunkt in die Anwendung. Sie stellt das Hauptfenster dar und das erstellte Stage-Objekt wird als Argument an die Methode start() der Klasse Applikation übergeben. Wenn eine JavaFX-Anwendung startet, erzeugt sie ein Root-Stage-Objekt, das an die Methode start(Stage primaryStage) der Root-Klasse der JavaFX-Anwendung übergeben wird. Dieses Stage-Objekt stellt das Primärfenster Ihrer JavaFX-Anwendung dar. Für ein Stage sind mehrere Operationen möglich. Eine Stage kann erstellt und angezeigt werden, man kann ihr eine Scene und einen Titel zuweisen, ihr eine Position, eine Breite und eine Höhe geben. Eine JavaFX- Stage kann

zu einer anderen Stage gehören. Stage kann zum Vollbildmodus wechseln. Es ist auch möglich, die Modalität des Fensters einer JavaFX Stage festzulegen. Die Modalität der Stage bestimmt, ob das Fenster, das die Stage darstellt, andere Fenster blockiert, die von derselben JavaFX-Anwendung geöffnet werden. Es gibt drei Modalitätsmodi. Die Modalitätsmodi Modality.APPLICATION_MODAL und Modality.WINDOW_MODAL sind nützlich für Stage-Objekte, die Fenster darstellen, die als "Assistenten" oder "Dialoge" fungieren und die Anwendung oder das Fenster blockieren sollen, bis der Benutzer den Prozess des Assistenten oder des Dialogs beendet. Die Modalität Modality.NONE ist nützlich für Stage-Objekte, die Fenster darstellen, die nebeneinander existieren können, wie verschiedene Browserfenster in einer Browseranwendung. Ein JavaFX-Stage kann verschiedene Styles haben. Die zur Verfügung stehenden Styles sind:

- Eine StageStyle.DECORATED ist ein Standardfenster mit OS-Dekorationen (Titelleiste und Schaltflächen Minimieren / Maximieren / Schließen) und einem weißen Hintergrund.
- Eine StageStyle.UNDECORATED ist ein Standardfenster ohne OS-Dekorationen, aber mit einem weißen Hintergrund.
- Eine tageStyle.TRANSPARENT ist ein nicht dekoriertes Fenster mit einem transparenten Hintergrund.
- Eine StageStyle.UTILITY mit einfarbig weißem Hintergrund und minimaler Fensterdekoration.

Es ist auch möglich, auf einer JavaFX-Stage auf Tastaturereignisse zu hören. Auf diese Weise kann man alle Tastaturereignisse auffangen, die eintreten, wenn die Stage den Fokus hat.

2.1.10 Scene

Die Scene ist ein Container für den visuellen Inhalt der Szene. Die SCene enthält UI-Elemente wie Bildansichten, Schaltflächen, Raster und TextBoxen. Die Klasse Javafx.scene.Scene des Pakets javafx.scene stellt alle Methoden zur Manipulation eines Scene-Objekts zur Verfügung. Eine Scene kann erstellt werden, indem das Objekt der Klasse Scene erzeugt und das Layout-Objekt an den Konstruktor der Klasse Scene übergeben wird.

2.1.11 Entwicklungsumgebungen und SDKs

Wie bei vielen Programmiersprachen oder Frameworks in der Softwareentwicklung ist eine Entwicklungsumgebung (IDE) nicht unbedingt notwendig, wird aber in der Regel von der Mehrheit der Softwareentwickler aus Gründen der Effizienz eingesetzt, da sie wiederkehrende Aufgaben und Funktionen übernimmt, die den Entwicklungsalltag erleichtern. Für JavaFX im Allgemeinen sind Eclipse,

IntelliJ IDEA, NetBeans und Visual Studio Code weit verbreitete IDEs, die die Produktivität von Programmierern erheblich steigert.

Um eine JavaFX-Anwendung zu implementieren, müssen je nach verwendeter IDE zusätzliche Komponenten installiert werden. Die folgenden Komponenten werden für jede IDE benötigt:

- Die Installation eines JDK: JavaFX 19 erfordert eine JDK-Version, die größer als JDK 11 ist. Für unseren Prototypen wird JDK 16 verwenden.
- Der optionale JavaFX Scene Builder Tools (JavaFX Scene Builder wird in Kapitel 3.5 genauer behandelt) für unser Prototyp wird Scene Builder version 19 verwenden.

2.1.12 JavaFX ohne Java programmieren zu müssen.

Obwohl JavaFX Script in den neuen Versionen von JavaFX nicht mehr vorhanden ist, wurde die ursprüngliche Absicht von JavaFX, nämlich dass die Erstellung von grafischen Benutzeroberflächen für Java-Anwendungen vereinfacht wird und dass Kenntnisse von Java im Besonderen keine zwingende Voraussetzung mehr sind, nicht aufgegeben. Denn JavaFX bietet mit FXML eine leicht zu erlernende deklarative Sprache anstelle von JavaFX Script, das eine alternative Definition von grafischen Oberflächen nur über XML bietet. Außerdem können dort auch Webtechnologien wie CSS (Cascading Style Sheets) oder JavaScript (aber auch andere Sprachen wie Groovy) verwendet werden, sie sind in den XML-Code eingebettet oder mit ihm verlinkt. Dieser Ansatz wird vom Szenengenerator unterstützt (ein GUI-Designer mit visuellen Steuerelementen zur Erstellung von Schnittstellen der Szenengenerator). Dies ermöglicht sogar die Erstellung einer grafischen Benutzeroberfläche per Drag-and-Drop oder rein visuell.

2.1.13 JavaFX Scene Builder und FXML

JavaFX Scene Builder ist ein Werkzeug zum Erstellen von visuellen Benutzeroberflächen, mit dem Benutzer schnell Benutzeroberflächen von JavaFX-Anwendungen entwerfen können, ohne zu codieren. Die Benutzer können Komponenten der Benutzeroberfläche per Drag-and-Drop in einen Arbeitsbereich ziehen, ihre Eigenschaften ändern, Stylesheets anwenden, und der FXML-Code des von ihnen erstellten Layouts wird automatisch im Hintergrund generiert. Das Ergebnis ist eine FXML-Datei, die dann mit einem Java-Projekt kombiniert werden kann, indem die Benutzeroberfläche mit der Logik der Anwendung verknüpft wird. JavaFX Scene Builder hat folgende Funktionen:

- UI Layout Tool Mit Scene Builder können JavaFX-Benutzeroberflächensteuerelemente, Grafiken, Formen und Container einfach layouten, um schnell Prototypen von Benutzeroberflächen erstellt werden.

- FXML Visual Editor Scene Builder erzeugt FXML, eine XML-basierte Auszeichnungssprache, mit der Benutzer die Benutzeroberfläche einer Anwendung getrennt von der Anwendungslogik definieren können. Diese Datei kann auch geöffnet und bearbeitet und anschließend mit Scene Builder revitalisiert werden.
- Integrated Developer Workflow Scene Builder kann in Kombination mit jeder Java-IDE verwendet werden, ist aber enger mit der NetBeans-IDE integriert. Im Rahmen dieser Arbeit wird Visual Studio Code verwenden, das ebenfalls eine gute Integration von Scene Builder bietet. Die Benutzeroberfläche wird daher direkt mit dem Quellcode verknüpft, der die Ereignisse und Aktionen, die auf jedes Element angewendet werden, durch einen einfachen Prozess verwaltet: die Anwendung in Eclipse ausführen, und jede Änderung, die in Eclipse an FXML vorgenommen wird, spiegelt sich auch im Scene Builder-Projekt wider.
- Preview Your Work Zu jedem Zeitpunkt während der Projekterstellung kann man sich eine Vorschau ansehen, wie die Benutzeroberfläche nach dem Einsatz tatsächlich aussieht.
- Cross Platform, Self Contained Scene Builder basiert auf JavaFX und wird von Windows, Mac OS X und Linux unterstützt. Es ist eine vollwertige JavaFX-Desktop-Anwendung. Scene Builder ist wie eine eigenständige Anwendung, was bedeutet, dass es mit einer eigenen privaten Kopie der JRE ausgeliefert wird.
- Mit der CSS-Unterstützung für Komponenten, die im Scene Builder verfügbar sind, ist es möglich, das Aussehen mithilfe von Stylesheets zu wählen.

Nach der Eröffnung eines neuen Projekts in Scene Bluider sind zu sehen (siehe Abbildung 3.5):

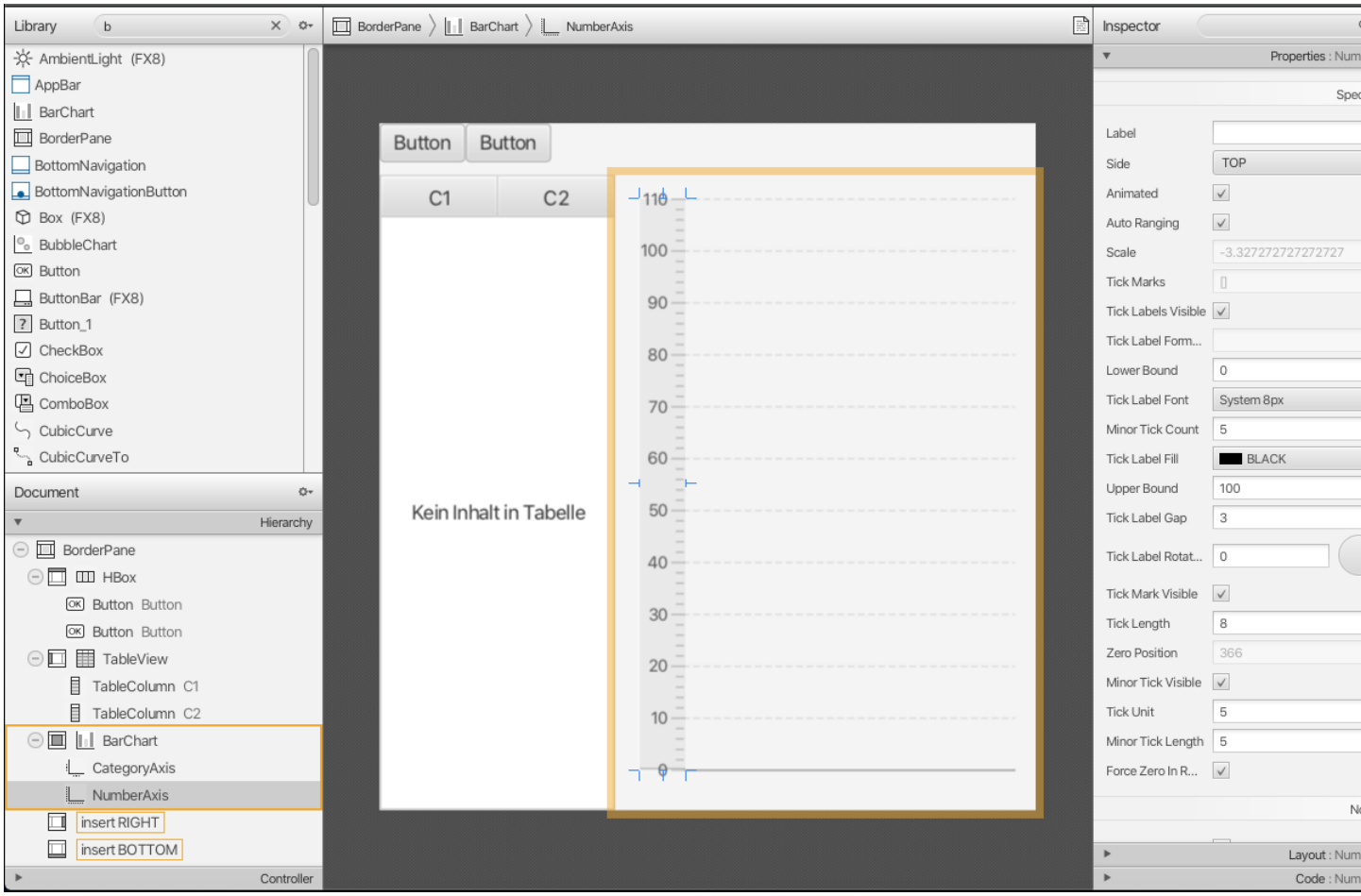
- Oben ein klassisches Menü, daneben ein Bereich für Fehlermeldungen und (interaktive) Statusanzeigen.
- Auf der linken Seite befindet sich standardmäßig ein Bibliotheksbereich, der eine Komponentenbibliothek enthält. Hier können Komponenten auswählt werden, aus denen die Benutzeroberfläche erstellt werden können. Dazu gehören z. B. Schaltflächen und allgemeine Formularfelder, Menüs, grafische Formen.
- Standardmäßig steht unter dem Abschnitt Bibliothek einen Abschnitt Hierarchie. Es befinden sich Komponenten, die in einer Art Baumstruktur verschachtelt sind. Wenn eine Schaltfläche beispielsweise in ein Fenster gezogen wird, wird die Schaltfläche in dem Bereich platziert, der dem Fenster in der Hierarchie untergeordnet ist. Allerdings können dort auch Komponenten ausgewählt oder per Drag & Drop aus dem Bibliotheksbereich in die Hierarchie einer grafischen Benutzeroberfläche verschoben werden.
- Der zentrale Bereich des Scene Builder bietet in der Regel den eigentlichen Arbeitsbereich. Hier

befindet sich die zukünftige grafische Benutzeroberfläche in einer Vorschau. Komponenten aus dem Bibliotheksbereich können per Drag & Drop in diesen Bereich gezogen werden oder hier ausgewählt und konfiguriert werden. Dieser Bereich wird auch als Inhaltsbereich bezeichnet. Standardmäßig wird eine neue FXML-Datei im JavaFX Scene Builder einen Wurzelcontainer des Typen AnchorPane bereitstellen. Die Vorlage ist entsprechend konfiguriert.

- Der rechte Bereich sollte den Standardinspektor anzeigen. Im Grunde ist dies der Bereich, in dem alle relevanten Eigenschaften und Funktionen der einzelnen Komponenten der Benutzeroberfläche angezeigt und eingestellt werden können.

Im mittleren Bereich von Scene Builder Beispiel-Prototyp (Abbildung2.3) wird als Layout ein Borderpane verwendet. Ein BorderPane hat einen Top, einen Left, einen Center, einen Right und einen Bottom-Bereich. In Top wird eine HBox gesetzt, die als Kindknoten zwei Buttons enthält, in Left befindet sich ein TableView, in Center ein BarChart und die Bereiche Right und Bottom sind leer.

Abbildung 2.3: Überblick eines Prototyps in Scene Builder



Der Beispiel 3.2 ist das gültige FXML-Dokument, der Scene Builder erstellten Prototypen generiert wurde.

Algorithmus 2.1 Erzeugter FXML-Code des Prototyps (siehe Abbildung (2.3)) in Eclipse

```
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <?import javafx.scene.chart.BarChart?>
4  <?import javafx.scene.chart.CategoryAxis?>
5  <?import javafx.scene.chart.NumberAxis?>
6  <?import javafx.scene.control.Button?>
7  <?import javafx.scene.control.TableColumn?>
8  <?import javafx.scene.control.TableView?>
9  <?import javafx.scene.layout.BorderPane?>
10 <?import javafx.scene.layout.HBox?>
11
12 <BorderPane xmlns="http://javafx.com/javafx/19" xmlns:fx="http://javafx.
       com/fxml/1" fx:controller="application.GanttChartController">
13    <center>
14       <BarChart prefHeight="513.0" prefWidth="541.0" BorderPane.alignment
             ="CENTER">
15         <xAxis>
16           <NumberAxis side="TOP" tickLabelFill="#573737" />
17         </xAxis>
18         <yAxis>
19           <CategoryAxis side="LEFT" />
20         </yAxis>
21       </BarChart>
22    </center>
23    <left>
24       <TableView prefHeight="200.0" prefWidth="200.0" BorderPane.
             alignment="CENTER">
25         <columns>
26           <TableColumn prefWidth="75.0" text="C1" />
27           <TableColumn prefWidth="75.0" text="C2" />
28         </columns>
29       </TableView>
30    </left>
31    <top>
32       <HBox prefHeight="100.0" prefWidth="200.0" BorderPane.alignment="
             CENTER">
33         <children>
34            <Button mnemonicParsing="false" text="Button">
35               <graphic>
36                  <Button mnemonicParsing="false" text="Button" />
37               </graphic>
38            </Button>
39         </children>
40       </HBox>
41    </top>
42 </BorderPane>
```

2.1.14 Zusammenfassung

JavaFX ist eine Erweiterung von Java. Sie wurde entwickelt, um eine leichtgewichtige und beschleunigte Java UI-Plattform für Unternehmensanwendungen bereitzustellen. Seine Anwendungen werden vollständig in Java entwickelt. Java FX bietet einen reichhaltigen Satz an UI-Steuerelementen, Grafiken und Medien-APIs mit hardwarebeschleunigten Hochleistungsgrafiken und Medien-Engines, um die Entwicklung visueller Anwendungen zu vereinfachen. Die Verwendung des JavaFX-Frameworks ist nicht auf die Programmierung mit Java beschränkt. Es kann auch in andere Programmiersprachen integriert werden. In dieser Arbeit wird Java verwendet, weil es populär ist und daher eine große Entwicklergemeinschaft hat, die im Falle eines Fehlers oder eines Programmabsturzes hilfreich sein kann.

2.2 Framework

2.2.1 Was sind Frameworks? Definitionsversuch und Merkmale

Eine allgemeingültige Definition eines Frameworks ist schwierig zu finden, da ein breites Spektrum der Nutzung möglich ist. Wörtlich bedeutet ein Framework "Gerüst". Der Begriff Framework wird in verschiedenen Kontexten verwendet. Es handelt sich dabei um einen Satz struktureller Softwarekomponenten, der es Entwicklern ermöglicht, Webanwendungen effizienter zu gestalten, indem er eine fertige und wiederverwendbare Architektur und Softwarekomponenten bereitstellt [Gra19].

Frameworks sind in vielfältigen Formen zu finden, die alle oder einige der folgenden Elemente enthalten können:

- Eine Reihe von Klassen, die in der Regel in Form von Bibliotheken zusammengefasst sind, um Dienste anzubieten.
- Einen auf Design Patterns basierenden Gestaltungsrahmen, um ein Anwendungsskelett ganz oder teilweise anzubieten.
- Empfehlungen für die Implementierung und Anwendungsbeispiele.
- Werkzeuge, die die Umsetzung erleichtern.

2.2.2 Zusammenfassung

Frameworks helfen Entwicklern bei der Erstellung von Sofware Produkten. Sie stellen eine Infrastruktur bereit, die die Low-Level-Details bereits gelöst hat, sodass sich der Entwickler auf die Details des spezifischen Projekts konzentrieren kann, anstatt sich mit wiederkehrenden technischen Aufgaben

wie der grundlegenden Anwendungsarchitektur, dem Datenzugriff, der Internationalisierung, der Ereignisprotokollierung (Logging), der Sicherheit (Authentifizierung und Rollenmanagement) oder sogar der Konfiguration der Anwendung zu beschäftigen. Der Entwickler muss auch weniger Code schreiben. Und weniger Code bedeutet weniger Fehleranfälligkeit und weniger Entwicklungszeit.

Frameworks helfen Entwicklern bei der Erstellung von Sofware Produkten. Sie stellen eine Infrastruktur bereit, die die Low-Level-Details bereits gelöst hat, sodass sich der Entwickler auf die Details des spezifischen Projekts konzentrieren kann, anstatt sich mit wiederkehrenden technischen Aufgaben wie der grundlegenden Anwendungsarchitektur, dem Datenzugriff, der Internationalisierung, der Ereignisprotokollierung (Logging), der Sicherheit (Authentifizierung und Rollenmanagement) oder sogar der Konfiguration der Anwendung zu beschäftigen. Der Entwickler muss auch weniger Code schreiben.

Der Entwurf eines Frameworks zur Erstellung von Gantt-Diagrammen wird hier die folgenden Schritte erfordern:

- Festlegung des Umfangs und der Ziele des Frameworks: Das Ziel ist es, die Erstellung von Gantt-Diagrammen zu erleichtern. Es müssen die Merkmale und Funktionen identifiziert werden, die das Framework enthalten soll.
- Anschließend wird nach bestehenden Lösungen gesucht: bestehende Gantt-Diagramm-Bibliotheken und -Frameworks, um ihre Stärken und Schwächen zu verstehen und sich für diese Lösung von ihnen inspirieren zu lassen.
- Identifizierung der Schlüsselkomponenten: Bestimmen von Schlüsselkomponenten, aus denen das Framework besteht, wie z. B. das Datenmodell, die Rendering-Engine und die Benutzeroberfläche. Diese Komponenten sollten modular aufgebaut und wiederverwendbar sein.
- Architektur entwerfen: Festlegen der Beziehungen zwischen den Komponenten, der Schnittstellen, die sie implementieren sollen, und der Gesamtstruktur des Frameworks. Die Architektur sollte flexibel und leicht erweiterbar sein.
- Das Framework implementieren: Verwenden das Framework nach bewährten Verfahren wie den SOLID-Prinzipien, um den Code wartbar und skalierbar zu machen. Das Framework muss das Datenmodell, das Rendering und die Benutzerinteraktion verwalten.
- Testen und validieren: Testen das Framework gründlich und lassen es im Vergleich zu den Anforderungen und Zielen, die im Anwendungsbereich festgelegt wurden, validieren.
- Funktionen hinzufügen: hinzufügen von Funktionnen, die häufig in Gantt-Diagrammen verwendet werden, wie z. B. Aufgabenabhängigkeiten, kritischer Pfad und Ressourcenzuweisung.

2.3 Rich Internet Applications

Da bei JavaFx bereits erwähnt wurde, dass es sich um ein RIA-Framework handelt. In diesem Abschnitt werden wir kurz über RIA beschreiben. (RIAs) sind Webanwendungen, die Daten verwenden, die sowohl vom Server als auch vom Client verarbeitet werden können. Außerdem findet der Datenaustausch asynchron statt, sodass der Client reaktionsfähig bleibt, während er Teile der Benutzeroberfläche ständig neu berechnet oder aktualisiert[KN]. Für die Users bieten RIAs ein ähnliches Aussehen und Gefühl wie Desktop-Anwendungen. RIAs zeichnen sich im Wesentlichen durch eine Vielzahl von interaktiven Bedienelementen, die Möglichkeit, die Anwendung online oder offline zu nutzen, aus.

Die Entwickler von Webseiten wollten die Vorteile klassischer Desktop-Anwendungen, mit denen des Internets kombinieren. RIAs von herkömmlichen Webanwendungen unterscheiden (s. Abbildung 2.4).

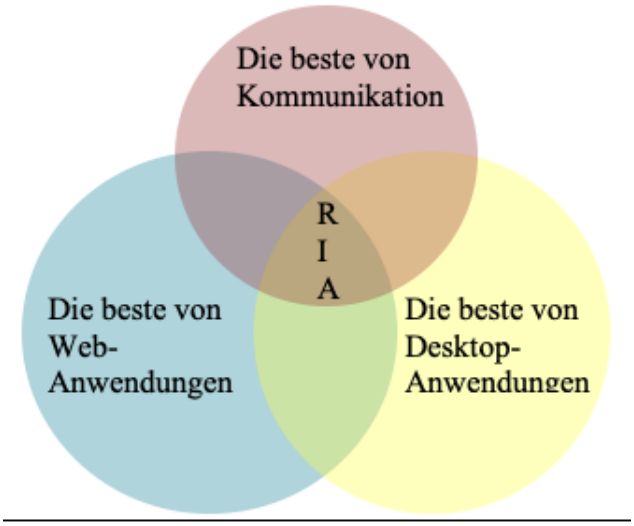


Abbildung 2.4: Einordnung der RIA-Technologie

2.3.1 RIA-Frameworks

Neben JavaFX, dem hier ein eigenes Kapitel gewidmet ist, sind folgende alternative RIA-Plattformen und Frameworks zu nennen:

HTML5 ist im Wesentlichen das Ergebnis der Verschmelzung des Besten aus HTML4, JavaScript, CSS, JavaScript-Bibliotheken und Flash in einer einzigen Spezifikation, die das API-Modell nutzt[GÃŒ18]. Es ist eine offene Technologie, was bedeutet, dass es kein einziges Führungsgremium wie Adobe für Flash oder Microsoft für Silverlight geben wird. Derzeit wird HTML5 von allen wichtigen Browsern unterstützt.

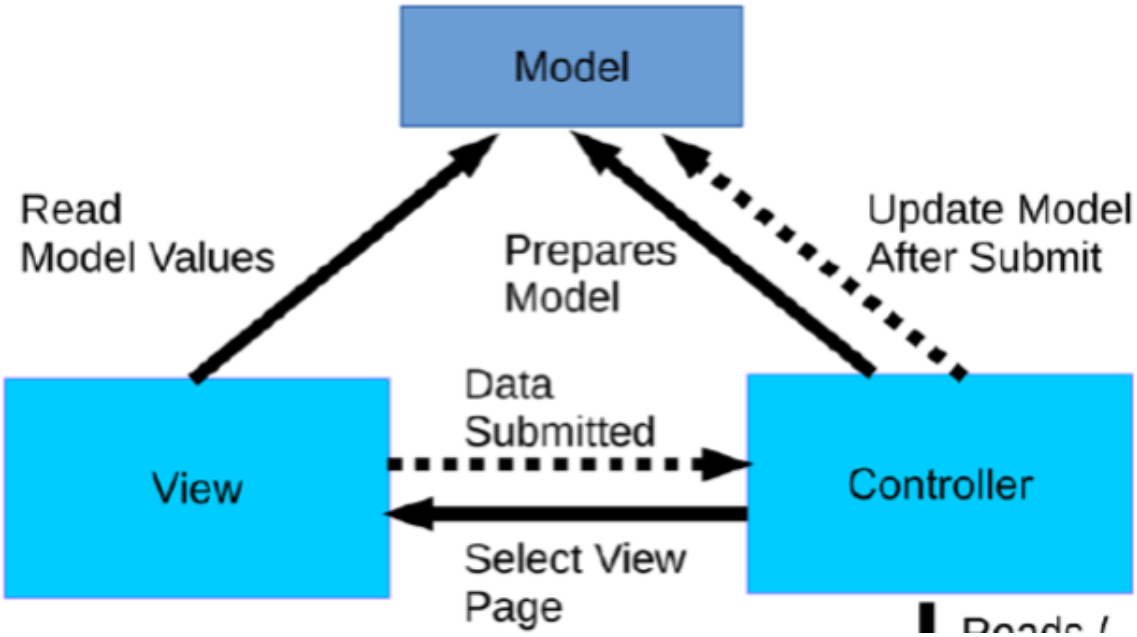
Angular[1] ist eine Entwicklungsplattform, die auf JavaScript aufbaut. Angular umfasst ein komponentenbasiertes Framework für die Erstellung skalierbarer Webanwendungen, eine reichhaltige Sammlung von Bibliotheken, die eine Vielzahl von Funktionen abdecken, wie z. B. Routing, Formularverwaltung, Client-Server-Kommunikation, und eine Reihe von Entwicklungswerkzeugen, die der Entwickler bei bei der Entwicklung, dem Aufbau, dem Testen und der Aktualisierung Ihres Codes helfen[HÃ¶22]. Die neueste Version von Angular ist Version 15. In Version 15 unterstützt angular keine TypeScript-Versionen mehr, die älter als 4.8 sind. Derzeit werden die Node.js-Versionen 14.20.x, 16.13.x und 18.10.x unterstützt.

2.3.2 Model View Controller (MVC)

JavaFX ermöglicht die Entwicklung mit Model View Controller (MVC) durch die Verwendung von FXML und Java. MVCbasiert auf drei Komponenten namens Model, View und Controller und in Abbildung 2.5 dargestellt. Die Vorlage wird hauptsächlich zur Verwaltung von Daten in einer Anwendung verwendet. Man kann sich die Sicht als die im Modell enthaltenen Daten vorstellen, die der Benutzer auf dem Bildschirm sieht. Die Ansicht stellt sicher, dass das Modell die aktuellen Informationen des Benutzers anzeigt. Wenn ein Benutzer die Daten in der Anwendung ändern möchte, informiert View eine andere Komponente namens Controller, die die Aktualisierung selbst nach Modell vornimmt. Die Controller sind das, was sie zusammenhält und die Koordination zwischen dem Modell und die View verwaltet. Durch die in der Anmeldung vorgenommene Aufteilung der Zuständigkeiten wird die Verwaltung der Anmeldung erleichtert. Wenn eine Änderung an der Schnittstelle vorgenommen werden muss, wird in der Tat nur die Ansicht geändert, wenn eine Änderung computerbezogen ist, müssen nur der Controller und das Modell geändert werden.

[1] https://angular.io/guide/what-is-angular

Abbildung 2.5: mvc[SpÃ20]



2.4 Tools zur Erstellung eines Gantt-Diagramm

Gantt-Diagramme ermöglichen eine übersichtliche Darstellung der zeitlichen Merkmale eines Prozesses. Es bestehen einige Java-basierte Tools zur Erstellung von Gantt-Diagrammen: SwiftGantt, GanttProject und ProjectLibre sind Open-Source-Tools. Darüber hinaus gibt es auch kostenpflichtige Tools wie FlexGanttFX. FlexGanttFX ist das JavaFX-basierte Gantt-Diagramm-Framework, das derzeit für Java verfügbar ist.

FlexGanttFX[2] ist ein sehr leistungsfähiges und komplexes benutzerdefiniertes Steuerelement für JavaFX-Anwendungen, ist eine Framework. FlexGanttFX ermöglicht das Bearbeiten oder Ändern von Daten, so dass die Benutzer Planungs- und Terminierungsvorgänge durchführen können. FlexGanttFX enthält keine Geschäftslogik, d. h. es liegt an der Anwendung, richtig zu reagieren, wenn der Benutzer eine Änderung vorgenommen hat. Jedes der vier folgenden Pakete, aus denen FlexGanttFX besteht, enthält eigene Klassen[Lem16].

[2]https://flexgantt.atlassian.net/wiki/spaces/FFXMAN/overview?homepageId=491528

Tabelle 2.3: List von FlexGanttFX-Paketen [Lem16]

FlexGanttFX-Pakete	Inhalt
com.flexganttfx.core	enthält verschiedene Utility-Klassen und Lizenzunterstützung.
com.flexganttfx.extra	enthält zusätzliche Klassen, wie z. B. eine GanttChartToolBar und eine GanttChartStatusBar.
com.flexganttfx.view	enthält die Visualisierungsklassen. Wie z. B. GanttChart
com.flexganttfx.model	enthält alle Klassen, die mit dem Datenmodell verbunden sind (Aktivitäten, Strecken, Depots)

GanttProjekt ist eine Open-Source-Initiative, ist eine Anwendung, die in Java und Koltin programmiert wurde. Ihr Ziel ist es, ein Programm zu erstellen, das Diagramme erstellen kann, die die kurz-, mittel- und langfristige Verteilung der Aufgaben eines Projekts darstellen. Gantt-Projekt ist besser geeignet für kleine Projekte. Es eignet sich daher auch für die Erstellung von Diagrammen für den persönlichen Gebrauch, z. B. für die Verwaltung von Gruppenprojekten an der Universität. Das Gantt-Projekt ermöglicht es, den Fortschritt von Aufgaben zu testen und eine Kopie ihres Status in Bezug auf den Zeitpunkt ihrer Ausführung zu speichern, um sie später, wenn sie abgeschlossen sind, vergleichen zu können. GanttProjekt läuft auf den folgenden Betriebssystemen: Windows, Linux und MacOS. Die aktuelle Version 3.2.3240 wurde veröffentlicht am 31. März 2022[3].

SwiftGantt ist eine OpenSource Java Swing Gantt Chart Komponente, die es Ihnen ermöglicht, den Projektplan als Gantt-Diagramm anzuzeigen, entweder als Client- oder als Server-Anwendung. SwiftGantt ist eine freie Software, die in der Liste der Programme Components & Libraries veröffentlicht wurde und zur Kategorie Entwicklung gehört. Die Tabelle 2.4 gibt einen allgemeinen Überblick über die wichtigsten Merkmale der SwiftGantt. SwiftGantt ist mit den folgenden Betriebssystemen kompatibel: Linux, Mac, Windows. Die neueste Version, die von seinem Entwickler veröffentlicht wurde, ist 0.4.0[4].

[3]https://www.ganttproject.biz/
[4]https://swiftgantt.sourceforge.net/en/index.html

Tabelle 2.4: List der SwiftGantt Funktionen[Sof11]

SwiftGantt Funktionen	Beschreibung
Zeiteinheit	Verschiedene Arten von Zeiteinheiten in SwiftGantt: Stunde, Tag, Woche, Monat, Jahr. Flexible Einstellungen für die Dauer der Arbeitstage in jeder Woche und die Dauer der Arbeitsstunden in jedem Tag.
Anpassen der Farben	man kann die Farben der Elemente im Gantt-Diagramm festlegen.
Anpassen der Größe	Die Breite der Zeiteinheit, die Höhe der Zeile, die Höhe der Aufgabenleiste und die Höhe des Fortschrittsbalkens lassen sich individuell anpassen.
Exportieren von Bildern	Das in SwiftGantt verteilte Gantt-Diagramm kann als Bilddateien in den Formaten .png und .jpg exportiert werden.
Mehrsprachigkeit	SwiftGantt unterstützt mehrere Sprachen, darunter Englisch, vereinfachtes Chinesisch, traditionelles Chinesisch, Japanisch, Französisch, Spanisch, Italienisch und Portugiesisch.

2.5 Grundlegendes Verständnis des Gantt-Diagramms

Gantt-Diagramme werden in vielen Bereichen eingesetzt, vor allem im Projektmanagement und in der Prozessplanung (Detailplanung, Sequenzplanung, Maschinenbelegungsplanung). Sie sind eine verbreitete und effektive Methode, um Aktivitäten (Aufgaben, Ressourcen und Ereignisse) in Abhängigkeit von der Zeit darzustellen.

Das Gantt-Diagramm ermöglicht es, die verschiedenen Aufgaben des Projekts, den Beginn und das Ende einer Aufgabe, die erwartete Dauer jeder Aufgabe, die Überschneidungen der Aufgaben und die Dauer dieser Überschneidungen, das Start- und Enddatum des gesamten Projekts schnell und effizient zu visualisieren.

Die Prozessplanung bildet die Grundlage für den Produktionsanlauf. Hier wird festgelegt, wann und in welcher Reihenfolge die verschiedenen Ressourcen die ihnen zugewiesenen Aufträge für einen bestimmten Zeitraum, z. B. einen Tag oder eine Woche, bearbeiten sollen.

In den 1890er Jahren erstellte der polnische Ingenieur Karol Adamiecki im Rahmen seiner Forschungen im Bereich der Management- und Planungstechniken das erste Gantt-Diagramm. Doch erst die 15 Jahre später von dem amerikanischen Ingenieur und Unternehmensberater Henry Gantt erstellte Version wurde unter dem Namen ihres Erfinders endgültig in den westlichen Ländern eingeführt.[Gan22] .

Ursprünglich wurden Gantt-Diagramme in mühevoller Handarbeit erstellt; jedes Mal, wenn sich ein Projekt änderte, musste das Diagramm geändert oder neu gezeichnet werden, was seinen Nutzen einschränkte, da ständige Veränderung ein Merkmal der meisten Projekte ist. Heute jedoch, mit dem Aufkommen von Computern und Projektmanagementsoftware können Gantt-Diagramme leicht erstellt,

aktualisiert und ausgedruckt werden.

Am häufigsten werden Gantt-Diagramme heute zur Verfolgung von Projektzeitplänen verwendet. Dazu ist es hilfreich, wenn zusätzliche Informationen zu den einzelnen Aufgaben oder Phasen des Projekts angezeigt werden können, z. B. die Beziehungen zwischen den Aufgaben, den Fortschritt jeder Aufgabe, die verwendeten Ressourcen, vorrangige Aufgaben usw.

In klassischen Gantt-Diagrammen wird ein zeitlicher Ablauf in Form von Balken dargestellt. Abbildung 2.6 zeigt ein klassisches Gantt-Diagramm. Auf der linken Seite der Abbildung sehen Sie die typische Tabelle eines Gantt-Diagramms, in der die Start- und Endzeiten sowie der Name jeder Aufgabe angegeben sind. Aktivität oder, im Gantt-Diagramm, der Name der Aufgabe. Auf der rechten Seite der Abbildung sehen Sie das Gantt-Diagramm. Dieses Diagramm verfügt über eine Zeitleiste, auf der das Jahr, der Monat und die einzelnen Tage angegeben sind. Auf dieser Zeitleiste sind Rechtecke angeordnet, die die verschiedenen Vorgänge darstellen. Die Breite eines Rechtecks steht für die Dauer der Aufgabe. Der Ablauf der einzelnen Vorgänge wird durch Pfeile definiert. Die Pfeile verbinden die einzelnen Vorgänge und stellen außerdem den zeitlichen Abstand zwischen den Vorgängen dar.

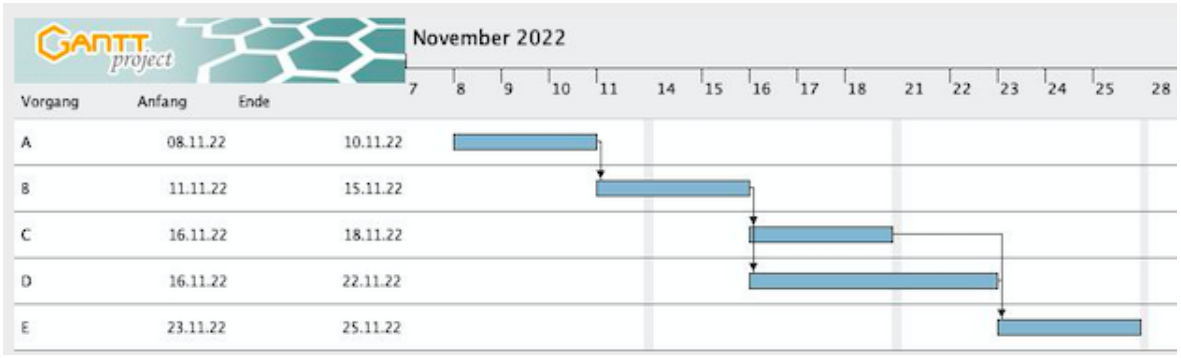


Abbildung 2.6: Ein einfaches Gantt Diagramms aus GanttProject

Die wichtigsten Elemente eines Gantt-Diagramms sind:

- Aktivitäten stellen die Aufgaben eines Projekts dar. Eine Aktivität wird mithilfe eines Zeitraums beschrieben, der mit einer Startzeit beginnt und grafisch durch einen Balken dargestellt wird. Aktivitäten können auch Meilensteine eines Projekts sein und/oder als kritisch markiert werden, wenn die Aufgabe bestimmte Kriterien erfüllt.
- Beziehungen definieren die Interdependenzen zwischen zwei Aktivitäten. Beispielsweise muss eine Aktivität abgeschlossen sein, bevor eine neue Aktivität gestartet werden kann. Beziehungen werden durch Linien und Pfeile dargestellt, die die Aktivitäten miteinander verbinden.
- Ressourcen stellen Menschen, technische Ausrüstung und andere Gegenstände dar, die für die Durchführung von Aktivitäten benötigt werden. Diese beginnen und beenden die Aktivitäten

nach einem bestimmten Zeitraum (oft abhängig von anderen Variablen) und schließen sie so ab. Im Gantt-Diagramm kann der Aktivitätenplan durch eine Übersicht über die Belastung der Ressourcen durch die Aktivitäten ersetzt werden.

3 Konzeption und Realisierung

3.1 Anforderungsanalyse

In diesem Kapitel werden die qualitativen und quantitativen Merkmale des Gantt Frameworks beschrieben. Die Grundanforderungen des Gantt-Diagramms definieren die Mindestfunktionen des Prototyps. Diese minimalen Funktionen werden das Hauptsystem unseres Prototyps bilden.

Die hier vorgeschlagene Architektur ist modular aufgebaut. Ziel ist es, die Funktionalitäten klar voneinander zu trennen, so dass Funktionen hinzugefügt oder entfernt werden können, ohne das gesamte Framework zu beeinflussen. Die Anforderungen 1 bis 3 beziehen sich auf technische Aspekte und sind daher funktionale Anforderungen, während Anforderung 5 die User Experience betrifft und daher nicht funktional ist.

- Ein Diagramm, das den Beginn und das Ende der Nutzung einer Ressource oder den Beginn und das Ende einer Aktivität mithilfe eines Balkens anzeigt. Das Diagramm soll also eine Liste von Tagen als Abszisse haben.
- Eine Tabelle, in der die verschiedenen Objekte (Aktivitäten, Ressourcen) und ihre verschiedenen Eigenschaften aufgelistet sind, z. B. der Name, die Priorität der Aktivitäten oder die Rolle der Ressourcen.
- Ein Menü, mit dem es möglich ist, im Diagramm zu navigieren, z. B. vom Ende des Diagramms zurück zum Anfang oder zum aktuellen Datum zu scrollen.
- die User Experience betriff der End User der Anwendung. Bei der Umsetzung der oben genannten Anforderungen ist zu bedenken, dass sie später bei der Entwicklung der Anwendung verwendet werden, daher müssen sie flexibel und anpassungsfähig sein.

3.2 Entwurf des Frameworks für die Erstellung des Gantt-Diagramms

Dieser Teil behandelt das Konzept eines Frameworks mit JavaFX. Er versucht, auf der Grundlage der zuvor erarbeiteten Konzepte und technischen Grundlagen einen Framework-Prototypen zu planen und zu entwickeln. Die dazu notwendigen Überlegungen werden in den folgenden Kapiteln erläutert.

Das Framework soll die Erstellung von allgemeinen Gantt-Diagrammen ermöglichen. Da JavaFX ein Framework ist, sollten dessen Funktionen, Möglichkeiten und Einschränkungen bei der Entwicklung des Frameworks für die Erstellung von Gantt-Diagrammen berücksichtigt werden. Um ein allgemeines Verständnis zu erlangen, das nicht auf eine bestimmte Implementierung beschränkt ist, haben wir Funktionen definiert, die in vielen populären Tools zur Erstellung von Gantt-Diagrammen verfügbar sind. Wir analysierten zunächst die vorhandenen Tools, überprüften die bisherige Forschung auf dem Gebiet der Gantt-Diagrammerstellung und schlossen JavaFX mit ein. Konkret konzentrierte sich unsere Analyse der Tools zur Erstellung von Gantt-Diagrammen auf GanttProject und FlexganttFX. Einige Tools werden als Anwendung zur Erstellung von Gantt-Diagrammen implementiert, während andere alle Funktionen über ein einzelnes Tool hinaus eng integrieren, sodass sich die Methode eher auf konzeptionelle Kategorien als auf implementierungsspezifische Details konzentriert. Ausgehend von diesem Materialsatz haben wir unsere grundlegenden Controls und Models definiert, die jedoch in den verschiedenen Quellen oft unterschiedlich beschrieben oder implementiert werden.

Dieses Kapitel vertieft einige Überlegungen aus der Anforderungsanalyse zur Funktionalität und versucht ein Konzept zu erarbeiten, wie diese umgesetzt werden können.

Zur Bereitstellung allgemeiner UI-Controls (eine Tabelle, ein Menu und eine Graphik) sind Methoden nötig, die bestimmte Aktionen innerhalb der Anwendung realisieren. Der Entwickler muss in der Lage sein, UI-Control entsprechend den Anforderungen seiner Anwendung intuitiv zu verwenden. Die bereits in den Anforderungen entwickelten Aspekte zur Bedienbarkeit, sollen daher hier weiter vertieft und deren Umsetzungsmöglichkeiten betrachtet werden.

3.2.1 Architektur des Prototyps

Die modulare Programmierung ist eine Technik der Softwareentwicklung, bei der ein Programm in unabhängige, wiederverwendbare Module aufgeteilt wird. Jedes dieser Module enthält bestimmte Funktionen und kann getrennt von den anderen Modulen getestet und gepflegt werden. Dies ermöglicht eine bessere Organisation des Codes, eine Verringerung der Komplexität und eine einfachere Wartung. Die modulare Programmierung wird in vielen Programmiersprachen wie Java, Python, C++ usw. verwendet. Durch die Verwendung von Modulen können Entwickler zuverlässigere und leichter zu wartende Programme erstellen, indem sie vorhandene Codeteile wiederverwenden und Änderungen in bestimmten Abschnitten des Programms isolieren.

Zur Erstellung des Prototyps werden folgenden Pakete und Dateien benötigt (siehe 3.1):

- DlGanttFx-modell enthält alle zum Datenmodell gehörigen Klassen (Aktivity, GanttTask, GanttRessourcen, Reservation, ResourceType, TaskPriority)

- DlGanttFx-control enthält die Steuerelement DatelineGraphControl, GanttTableControl, GanttBar, ObservableGanttBar und GanttChartHbox.
- DlGanttFx-util enhält die Klasse DateUtil.
- CSS enhält die verwendeten Stylesheets.

Die oben genannten Pakete und Dateien werden dann im .jar-Format exportiert. Eine JAR-Datei (Java Archive) ist eine ZIP-Datei, die zur Verteilung eines Satzes von Java-Klassen verwendet wird. Die exportierte Datei DlGanttFx.jar enthält die Definitionen der Klassen, aus denen sich unser gesamtes Programm zusammensetzt.

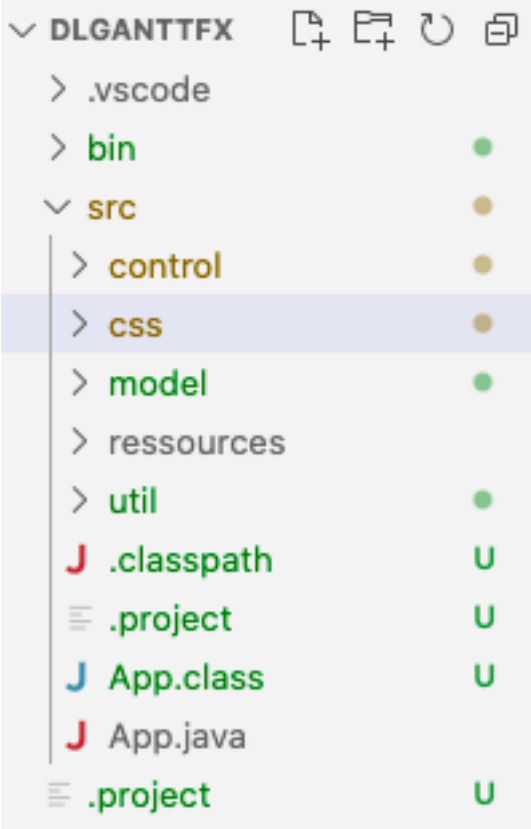


Abbildung 3.1: Projektaufbau des Prototyps

3.2.2 Datenmodell

Die gesamte Datenmodell des Prototyps ist in dem Model-Paket zu finden. Das Model-Paket kapselt das Datenmodell Activity und die zugrunde liegenden Datenmodelle(siehe 3.2) und stellt sie die Control des Control-Pakets zur Verfügung. einige dieser Klassen werden im Folgenden beschrieben.

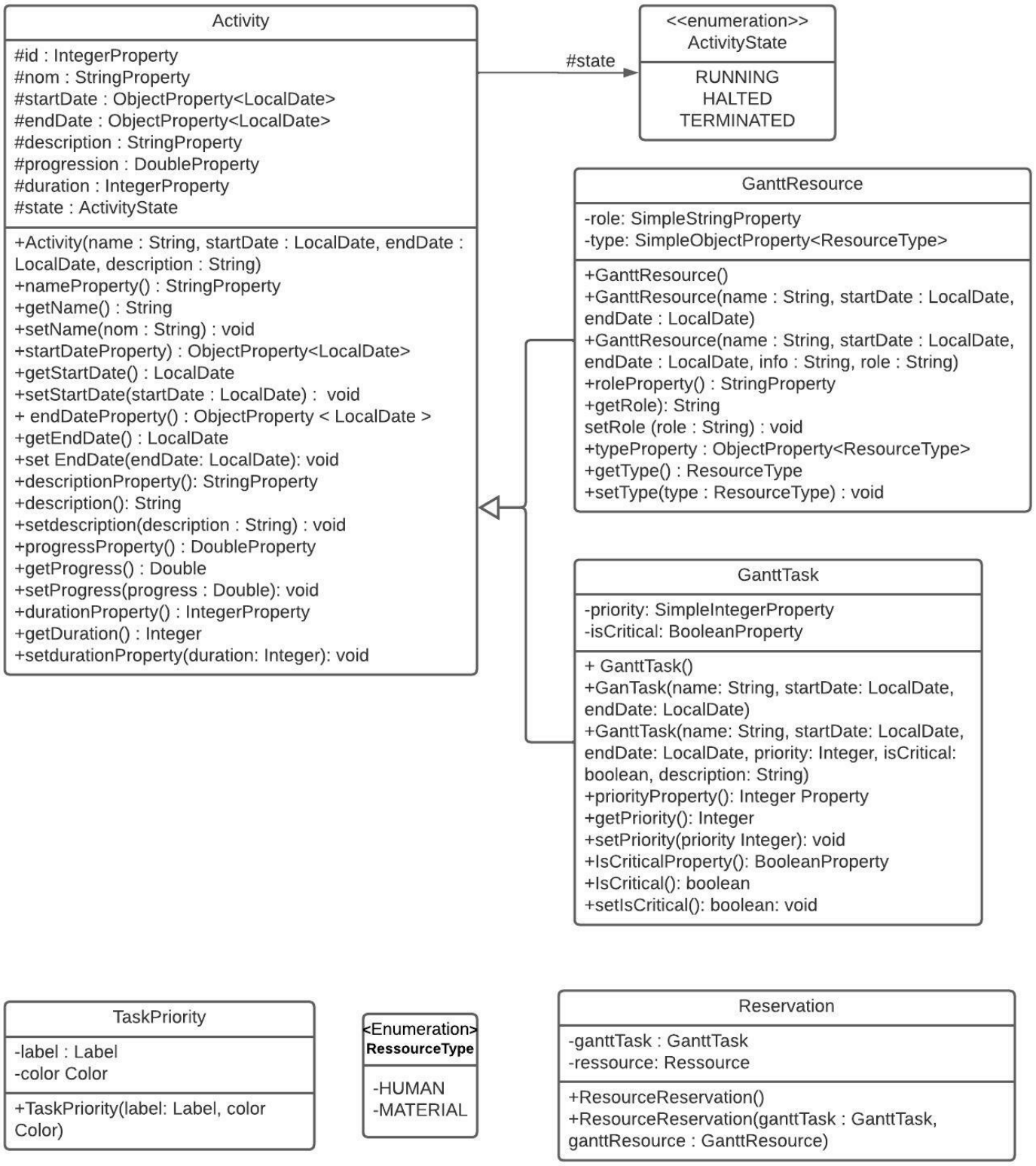


Abbildung 3.2: Übersicht der Klassen des Datenmodells

3.2.2.1 Die Klasse Activity

In Ein typisches Basisobjekt (Die Klasse Activity), das in einem Gantt-Diagramm dargestellt werden kann, wurde mit seinen grundlegenden Eigenschaften wie ID, Name, Start, Ende, und Dauer und Description definiert. Diese Klasse kann erweitert werden, je nachdem, ob der Entwickler eine Aufgabe, eine Ressource oder etwas anderes verwenden möchte. Das Model-Paket enthält zum Beispiel Geschäftsobjekte (GanttTask und GanttResource). Die Klasse Activity enthält die Eigenschaften, die notwendig sind, um ein Gantt-Diagramm zu erstellen. Darüber hinaus enthält das Activity protected-Methoden, die von den Geschäftsobjekten geerbt werden können. Das Klassendiagramm in Abbildung (3.2) listet alle Elemente der Klasse Activity auf.

3.2.2.2 Die Klasse ActivityState

Die Klasse ActivityState ist ein Enum Klasse, das je nachdem, ob die Aufgabe noch nicht begonnen hat, beendet ist oder noch läuft, HALTED, TERMINATED oder RUNNING zurückgibt.

3.2.2.3 Die Klasse GanttTask

Die Klasse GanttTask ist eine Erweiterung der Klasse Activity. Sie implementiert die Attribute und die Methode, die für eine Aufgabe spezifisch sind.

3.2.2.4 Die Klasse TaskPriority

Die Klasse TaskPriority ist eine Klasse, mit der ein Label und eine Farbe entsprechend der Priorität der Aufgabe festgelegt werden können.

3.3 Implementierung

Anhand des zuvor erarbeiteten Entwurfs wird in den folgen Kapiteln die Realisierung des Prototyp-Frameworks zur Erstellung von Gantt-Diagramme demonstriert. Da hier nicht auf alle Teile des Quellcodes eingegangen werden kann, wurde der Code mit Kommentaren versehen, sodass sich dieses Kapitel auf das Wesentliche konzentriert. Hier erfolgt eine Beschreibung von Klassen und Objekten zum Implementieren von Modellen und Steuerelemente.

3.3.1 Die Klasse Activity der Model-Paket

In Listing 5.1 zeigt, wie die Attribute und der Konstruktor der Klasse Activity aussehen. Die Eigenschaften id, name, startDate, endDate und Description werden vom Benutzer eingegeben und die Eigenschaften Duration, isWorkComplete und State werden automatisch in der Tabelle ausgefüllt. Um die Duration zu berechnen, zählen wir die Anzahl der Tage zwischen StartDate und EndDate und addieren eins dazu. Die Eigenschaft isWorkComplete wird anhand des StartDatums, des aktuellen Datums und des EndDatums berechnet. Der Status wird je nachdem, ob die Aufgabe abgeschlossen ist (isWorkComplete gleich 1, d.h. das EndDate ist abgeschlossen), State gleich TERMINATED, die Aufgabe nicht begonnen hat (isWorkComplete ist gleich 0, d.h. das StartDate ist kleiner als das aktuelle Datum), State gleich HALTED ist, oder die Aufgabe in Bearbeitung ist (isWorkComplete ist zwischen 0 und 1, d.h. das aktuelle Datum liegt zwischen dem Start- und Enddatum), State gleich RUNNING ist.

Algorithmus 3.1 Konstruktoren der GanttDataModel-Klasse

```
1  package model;
2
3  public class Activity {
4
5    public Activity(int id, String name, LocalDate startDate, LocalDate
       endDate, String description) {
6        this.id = new SimpleIntegerProperty(id);
7        this.name = new SimpleStringProperty(name);
8        this.startDate = new SimpleObjectProperty<>(startDate);
9        this.endDate = new SimpleObjectProperty<>(endDate);
10       double totalWorkingDays = Duration.between(startDate.atStartOfDay
           (), endDate.atStartOfDay()).toDays() + 1;
11       this.duration = new SimpleIntegerProperty((int) totalWorkingDays)
           ;
12       if (startDate == null || endDate == null || LocalDate.now().
           isBefore(startDate)) {
13           this.progress = new SimpleDoubleProperty(0.0);
14       } else if (LocalDate.now().isAfter(endDate)) {
15           this.progress = new SimpleDoubleProperty(1.0);
16       } else {
17           this.progress = new SimpleDoubleProperty(
18                   (double) (Duration.between(startDate.atStartOfDay(),
                       LocalDate.now().atStartOfDay()).toDays() + 1)
19                           / totalWorkingDays);
20       }
21       this.state = (progress == null || progress.get() == 0.0) ?
           ActivityState.HALTED
22               : ((progress.get() == 1.0) ? ActivityState.TERMINATED :
                     ActivityState.RUNNING);
23       this.description = new SimpleStringProperty(description);
24   }
25
26 }
```

3.3.2 Die Steuerungselement

Das Control-Paket enthält Definitionen und Angaben, die für die Erstellung der grafischen Benutzeroberfläche notwendig sind. Dabei handelt es sich hauptsächlich um die Klassen DatelineGraphControl, GanttTableControl und GanttChartHbox, Ganttbar, ObservableGanttBar.

3.3.2.1 Ganttbar

Die Klasse, die ein Balken implementiert, wird durch die Datei Ganttbar.java definiert. Die Klasse Ganttbar ist eine Erweiterung von JavaFX Scene Layout Pane. Die Ganttbar wird in vier Typen unterschieden

: PieceType.COMPLETE, PieceType.BEGINNING, PieceType.CENTER und PieceType.END. Jeder dieser Typen wird in der CSS-Datei definiert.

3.3.2.2 Die Klasse DatelineGraphControl

Die Klasse, die das Diagramm implementiert, wird durch die Datei DatelineGraphControl.java definiert. Das Diagramm wird in Form einer Tabelle implementiert, deren Spalte die Woche, deren Unterspalte die Tage und deren Zeile die Objekte ist. Das Diagramm beginnt am ersten Tag des aktuellen Monats. Der Entwickler hat die Möglichkeit, je nach Bedarf die Anzahl der Monate oder Tage einzugeben, aus denen das Diagramm bestehen soll. Er kann auch einfach ein Start- und Enddatum für das Diagramm eingeben. Die Spalten, die für Wochenenden stehen, werden farbig dargestellt, ebenso wie die Zeile der Spalte des aktuellen Tages. Die Aktivitäten werden durch einen Balken dargestellt, der den Anfang und das Ende des Objekts anzeigt. Um das Gantt-Diagramm mit Inhalt zu füllen, ist eine Funktion zum Hinzufügen von Activity (Aktivitäten, Ressourcen, Reservierungen) erforderlich.

Die DatelineGraphControl ist eine generische, abstrakte Klasse. Er erweitert der javafx.scene.control.TableView. In der Abbildung 3.3 sind alle Attribute und Methode zur Implementirung der DatelineGraphControl zu sehen.

Die Methode drawDiagram() fügt Balken in die Grafik ein. Die Zeilen entsprechen jeweils einer Aktivität. Es wird also nach der Start- und Endspalte einer Aktivität gesucht und in den Zellen der Zeile einer Aktivität wird von der Start- bis zur Endspalte jeweils PieceType.BEGINNING, PieceType.CENTER und PieceType.End hinzugefügt, und wenn bei einer Aktivität die Start- und Endspalte gleich sind, wird PieceType.COMPLETE hinzugefügt.

Die Methode ScrolltoColunmEvent() ermöglicht es, dass bei einem Klick auf eine Aktivität in der Tabelle zu dieser Aktivität in der Grafik gescrollt wird.

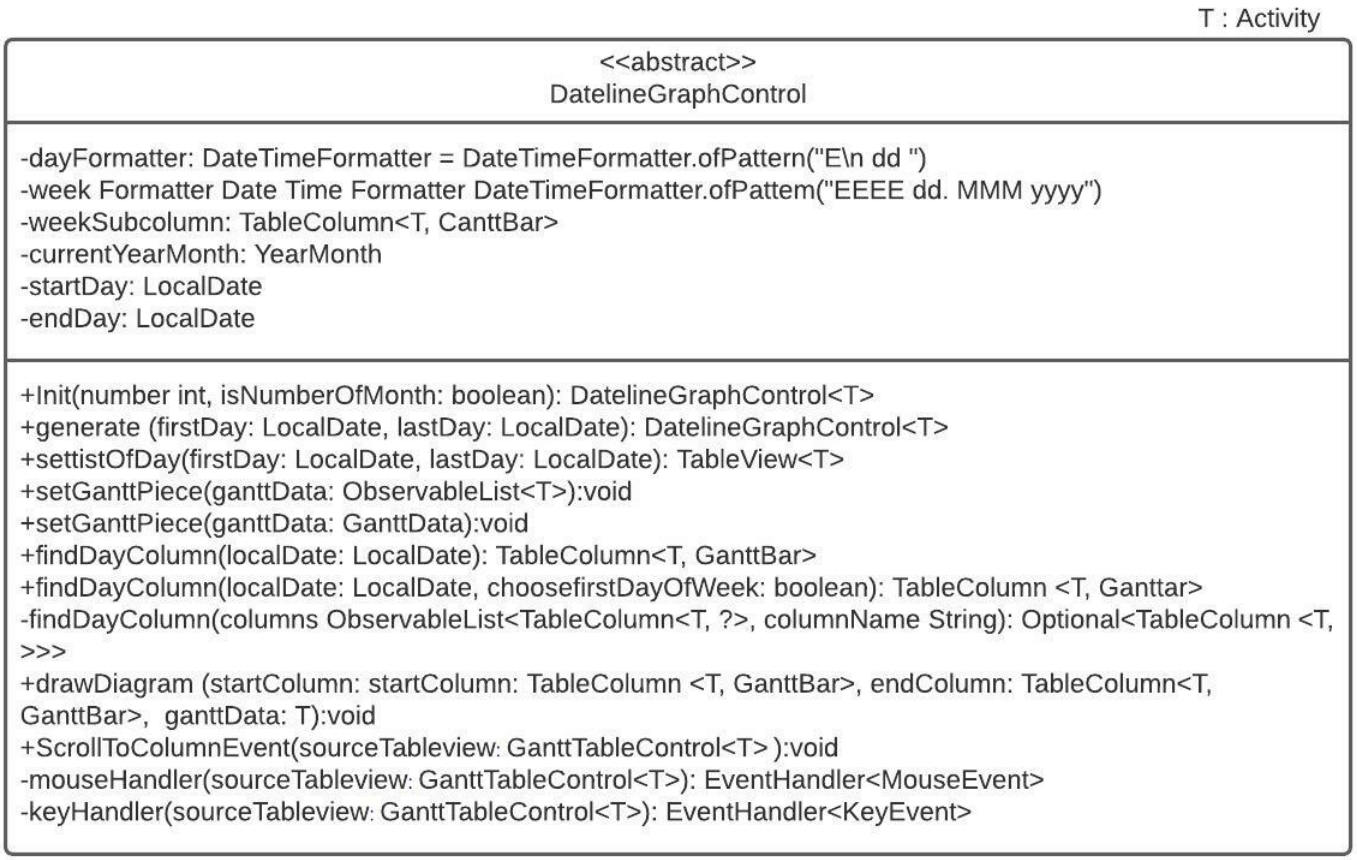


Abbildung 3.3: Das Klassediagramm der DatelineGraphControl aus dem Control-Paket des entwickelten Prototyps.

3.3.2.3 Die Klasse GanttTableControl und GanttTaskControl

Die Klasse, die die Tabelle implementiert, in der die verschiedenen Objekte und ihre Eigenschaften aufgelistet sind, wird in der Datei GanttTableControl.java definiert. GanttTableControl ist eine generische, abstrakte Klasse, die der javafx.scene.control.TableView erweitert.

Eine Tabelle, die es ermöglicht, ein Objekt anzuzeigen, das beispielsweise eine Aufgabe oder eine Ressource sein kann. In der Spalte werden der Name des Objekts, sein Anfang, sein Ende, seine Dauer, seine Informationen, sein Status, und dank eines Fortschrittsbalkens seine Entwicklung angezeigt. Sie bietet auch eine Legende für die Prioritäten. wenn eine Spalte für die Prioritäten in der Tabelle vorhanden ist.

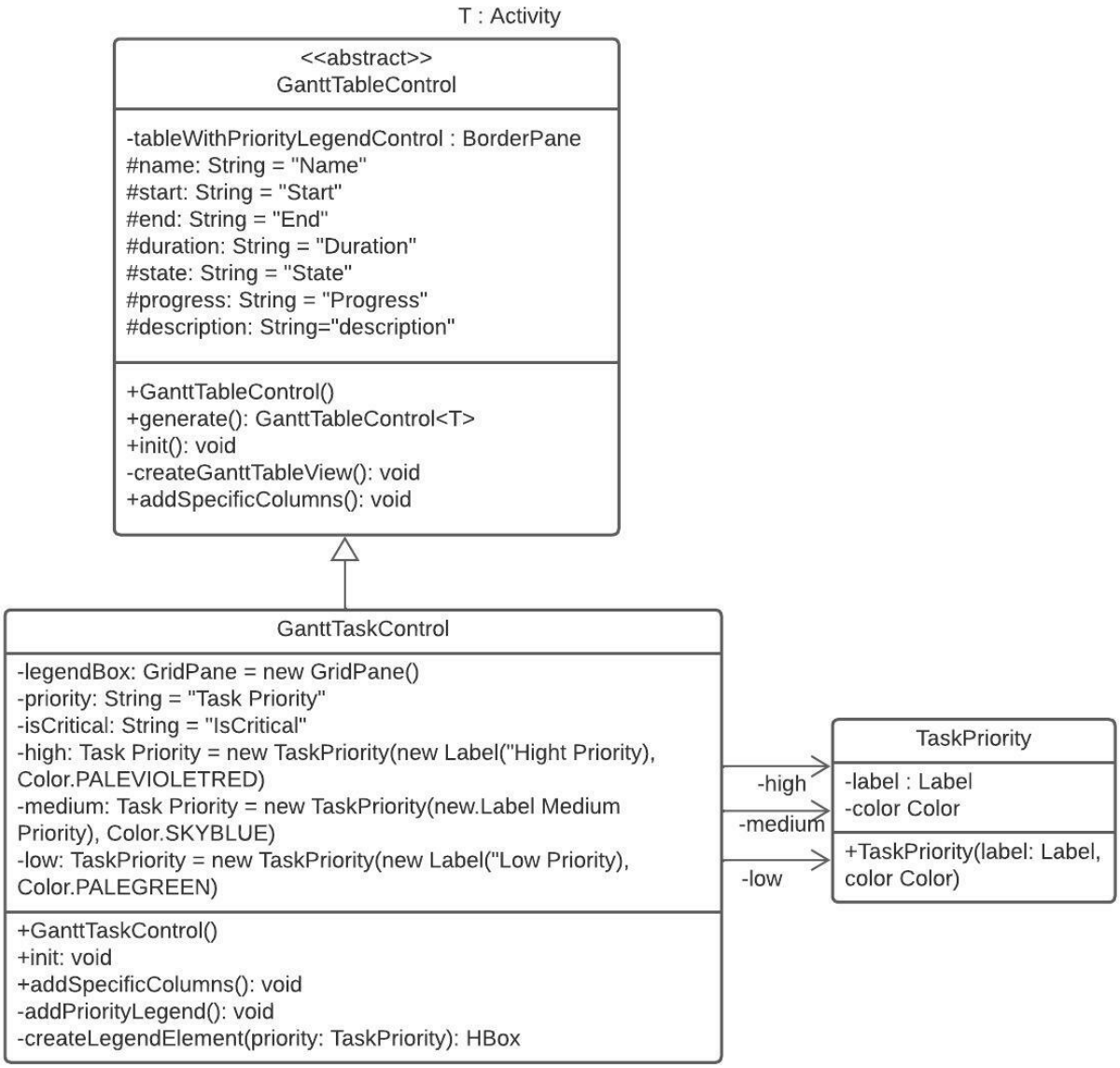


Abbildung 3.4: Übersicht der Klassediagramme des GanttTaskControls aus dem Control-Paket des entwickelten Prototyps.

In GanttTableControl werden mit der Methode createGanttTableView fünf Spalten erstellt, nämlich Name, Startdatum, Enddatum, Dauer, Status, Fortschritt und die Beschreibung. Wie bereits erwähnt, Nach dem Hinzufügen einer Aktivität in der Tabelle, werden der Dauer, der Status und der Fortschritt automatisch berechnet und direkt in die entsprechenden Spalten der Tabelle eingegeben. Anschließend wird auf die Werte vom Typ Double in der Spalte Progress zugegriffen. Diese Werte werden dann als Parameter in einer ProgressBar der Java Scene Control eingetragen, um eine grafische Darstellung des Fortschritts der Aufgabe zu erhalten (siehe Listing 3.2).

In der Klasse GanttTableControl erlaubt die abstrakte Methode addSpecificColumns das Hinzufügen

neuer objektspezifischer Spalten. In der Klasse GanttTaskControl, eine Kindklasse von der GanttTable-Control, ermöglicht die Überschreibung der Methode addSpecificColumns das Hinzufügen von Spalten, die für eine Aufgabe spezifisch sind. Insbesondere eine Spalte für die Priorität der Aufgabe und eine weitere, um zu überprüfen, ob sich eine Aufgabe auf dem kritischen Pfad befindet (siehe Listing3.3).

Algorithmus 3.2 Die Methode createGanttTableView in der Klasse GanttTableView

```
1  public abstract class GanttTableControl<T extends Activity> extends
       TableView<T> {
2
3      protected String progress = "Progress";
4      private void createGanttTableView() {
5      TableColumn<T, Double> progressCol = new TableColumn<T, Double>(
           progress);
6          progressCol.setCellValueFactory(new PropertyValueFactory<T,
               Double>(
7              "progress"));
8          progressCol
9              .setCellFactory(ProgressBarTableCell.<T>forTableColumn())
                   ;
10         Callback<TableColumn<T, Double>, TableCell<T, Double>>
               cellFactory = new Callback<TableColumn<T, Double>,
               TableCell<T, Double>>() {
11             public TableCell<T, Double> call(TableColumn<T, Double> p
                   ) {
12             return new TableCell<T, Double>() {
13
14                 private ProgressBar pb = new ProgressBar();
15                 @Override
16                 public void updateItem(Double item, boolean empty) {
17                     if (item != null) {
18                         super.updateItem(item, empty);
19                         if (empty) {
20                             setText(null);
21                             setGraphic(null);
22                         } else {
23                             pb.setProgress(item);
24                             setGraphic(pb);
25                             pb.setStyle("-fx-accent:␣green");
26                             }
27                     }
28                 }
29                 };
30
31             };
32     progressCol.setCellFactory(cellFactory);
33
34     this.getColumns().addAll(nameCol, startCol, endCol, durationCol,
           stateCol, progressCol, descriptionCol);
35
36     // add specific columns
37     addSpecificColumns();
38
39     }
40     public abstract void addSpecificColumns();
41 }
```

Listing 3.3 zeigt die Überschreibung Methode addSpecificColumns der Klasse GanttTableControl in der Klasse GanttTaskControl. Mit der Überschreibung der Methode addSpecificColumns werden zwei neue Spalten hinzugefügt, eine davon für die Priorität der Aufgabe und eine weitere, um herauszufinden, ob sich eine Aufgabe auf dem kritischen Pfad befindet.

Die Eigenschaft Priorität der Klasse GanttTaskControl ist vom Typ Integer. Es steht eine Klasse TaskPriority zur Verfügung, mit der die Prioritätsstufe und die Farbe der Priorität festgelegt werden können. Nach dem Ausfüllen der Tabelle wird einer Zahl eine Prioritätsstufe zugewiesen. Und die definierte Farbe dieser Prioritätsstufe wird in einem Quadrat ausgefüllt. In der Tabelle wird die Priorität der Aufgabe also als Quadrat mit einer Hintergrundfarbe dargestellt. Die Hintergrundfarbe hängt von der Prioritätsstufe der Aufgabe ab.

Der kritische Pfad ist eine Berechnung anhand eines Gantt-, oder Pert- Diagramms die darauf abzielt, die längste Reihenfolge der zu erledigenden Aufgaben zu ermitteln und damit die kürzeste Zeit für die Fertigstellung des Projekts zu bestimmen. Die Aufgaben, die durch die Berechnung des kritischen Pfades identifiziert werden, sind die wichtigsten Aufgaben des Projekts. Sie werden als kritische Aufgaben bezeichnet, da sie für den Erfolg des Projekts entscheidend sind. Im Rahmen dieses Prototyps wird der kritische Pfad nicht berechnet, sondern als Boolean in die Spalte isCriticalCol eingefügt. Die Werte in der Spalte isCriticalCol werden dann durch Checkboxen ersetzt.

Algorithmus 3.3 Die Methode addSpecificColumns der Klasse GanttTaskControl

```
1  public class GanttTaskControl extends GanttTableControl<GanttTask> {
2      @Override
3      public void addSpecificColumns() {
4          // Creating columns
5          TableColumn<GanttTask, Integer> priorityCol = new TableColumn<
               GanttTask, Integer>(priority);
6          priorityCol.setCellValueFactory(new PropertyValueFactory<
               GanttTask, Integer>("priority"));
7          priorityCol.setCellFactory(column -> new TableCell<GanttTask,
               Integer>() {
8           @Override
9           protected void updateItem(Integer item, boolean empty) {
10              Rectangle rect = new Rectangle();
11              rect.setX(20); // setting the X coordinate of upper left
                    //corner of rectangle
12              rect.setY(20); // setting the Y coordinate of upper left
                    //corner of rectangle
13              rect.setWidth(20); // setting the width of rectangle
14              rect.setHeight(20);
15              super.updateItem(item, empty);
16              if (empty) {
17                  setText(null);
18                  setGraphic(null);
19              } else {
20                  switch (item) {
21                      case 3:
22                          rect.setFill(low.getColor());
23                          legendBox.getChildren().add(rect);
24
25                          break;
26                      case 2:
27                          rect.setFill(medium.getColor());
28                          legendBox.getChildren().add(rect);
29                          break;
30                      case 1:
31                          rect.setFill(high.getColor());
32                          legendBox.getChildren().add(rect);
33                          break;
34                  }
35                  setGraphic(rect);
36              }
37          }
38      });
39
40      TableColumn<GanttTask, Boolean> isCriticalCol = new TableColumn<
            GanttTask, Boolean>(isCritical);
41      isCriticalCol.setCellValueFactory(new PropertyValueFactory<
            GanttTask, Boolean>("isCritical"));
42      isCriticalCol.setCellValueFactory(cellData -> cellData.getValue()
            .isCriticalProperty());
43      isCriticalCol.setCellFactory(CheckBoxTableCell.forTableColumn(
            isCriticalCol));
44      isCriticalCol.setCellFactory(column -> new CheckBoxTableCell());
45  }
```

3.3.2.4 Die Klasse GanttChartHbox

Der GanttChartHbox ist durch die Datei GanttChartHbox.java definiert. Der GanttChartHbox ist eine generische Klasse, die der javafx.scene.layout.HBox erweitert. Darin enthalten sind zwei Buttons, ein ListView, Images, welche in ihrer Gesamtheit die GanttChartHbox bilden. Das Klassendiagramm in Abbildung3.5 listet alle Eigenschaften und Methode der Klasse GanttChartHbox auf.

Das Gantt-Diagramme können viel Platz auf dem Bildschirm einnehmen. Daher muss die Ansicht so bewegt und skaliert werden können, damit der Benutzer nicht den Überblick verliert. Ein GanttChartHbox ermöglicht es, Aktionen auf dem Diagramm auszuführen. Um diese Aktionen zu realisieren, verfügt der GanttChartHbox von drei Buttons und eine Liste der Jahre, über die sich die Grafik erstreckt. Wenn ein Jahr in der Liste gedrückt wurde, wird bis zum ersten Tag des betreffenden Jahres gescrollt. Die Now-Taste wird verwendet, um zum heutigen Datum zurückzukehren. Die Earsliest-Taste wird verwendet, um zum ersten Tag des Jahres zu scrollen, in dem das Diagramm beginnt, und Lastest-Taste, um zum ersten Tag des letzten Jahres des Diagramms zu scrollen.

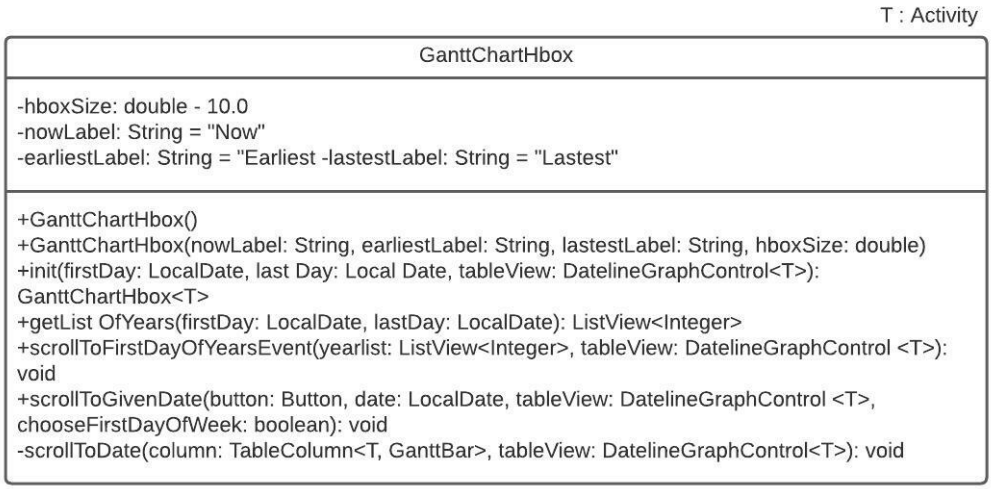


Abbildung 3.5: Übersicht der Klassediagramme des GanttChartHbox aus dem Control-Paket des entwickelten Prototyps.

4 Evaluation

Das Ziel dieser Masterarbeit war es, ein Prototyp zu entwickeln und zu implementieren, der die Erstellung eines Gantt-Diagramms ermöglicht. Zu diesem Zweck sollten zunächst die verschiedenen grundlegenden Technologien und Standards zur Realisierung eines solchen Systems untersucht und beschrieben werden. Der praktische Teil der Arbeit umfasste die Implementierung des Prototyps (Datenmodell und verschiedene Steuerelementen). In diesem Kapitel wird der Prototyp anhand eines Anwendungsfalls vorgestellt.

4.1 Praxisfall

Ein IT-Team möchte ein Gantt-Diagramm verwenden, um die Aufgaben eines neuen Projekts zu visualisieren, bei dem eine Anwendung erstellt werden soll, mit der medizinische Daten auf dem Telefon empfangen werden können.

4.1.1 GanttChart

Um eine Ganttchart-GUI zu entwickeln, muss der Entwickler zunächst die JavaFX-Distributionsdateien und die Datei DlGanttFx.jar (siehe Unterabschnitt3.2.1) zum Classpath seines Java-Projekts hinzufügen. Anschließend können die Klassen GanttTableControl, DatelineGraphControl und GanttChartHbox importieren werden. (Abbildung (4.1)). Anschließend werden die Tableview ganttTaskControl und datelineControl mithilfe der Methode addGanttTask mit Daten ausgefüllt.

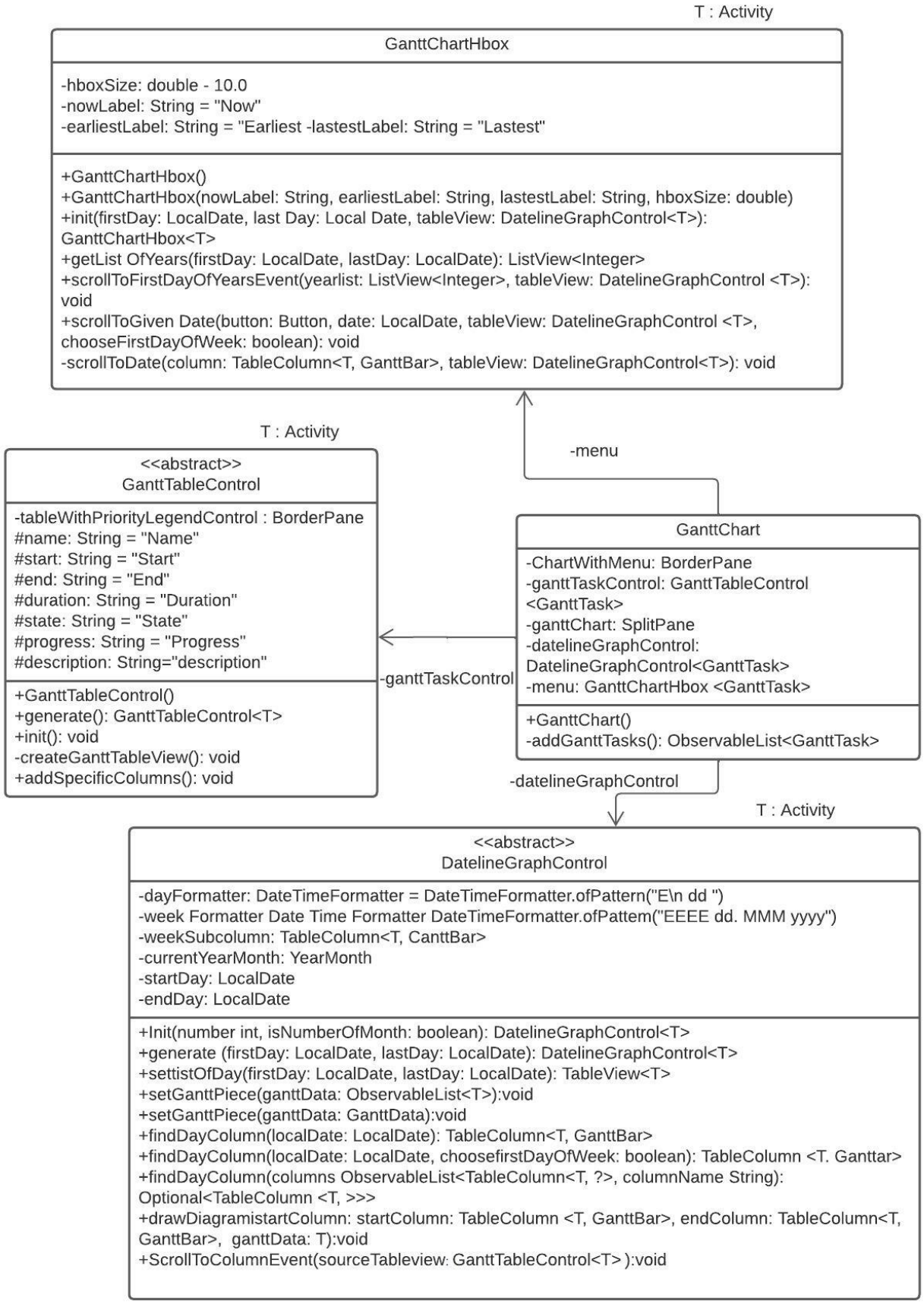


Abbildung 4.1: UML-Diagramm der GanttChart-Steuerelements

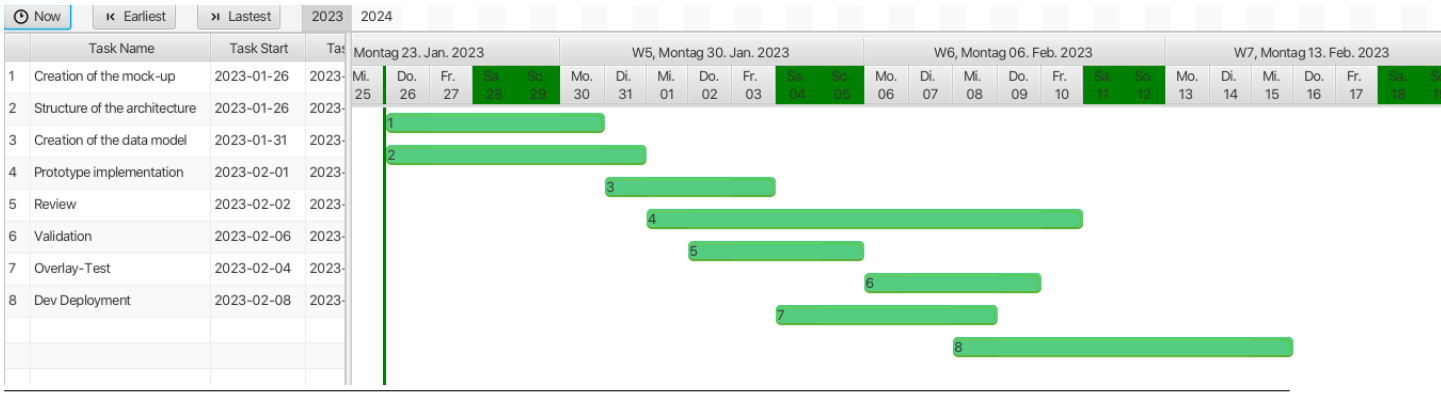
4.1.1.1 Die Strukture

Das GUI besteht aus mehreren untergeordneten Steuerelementen:

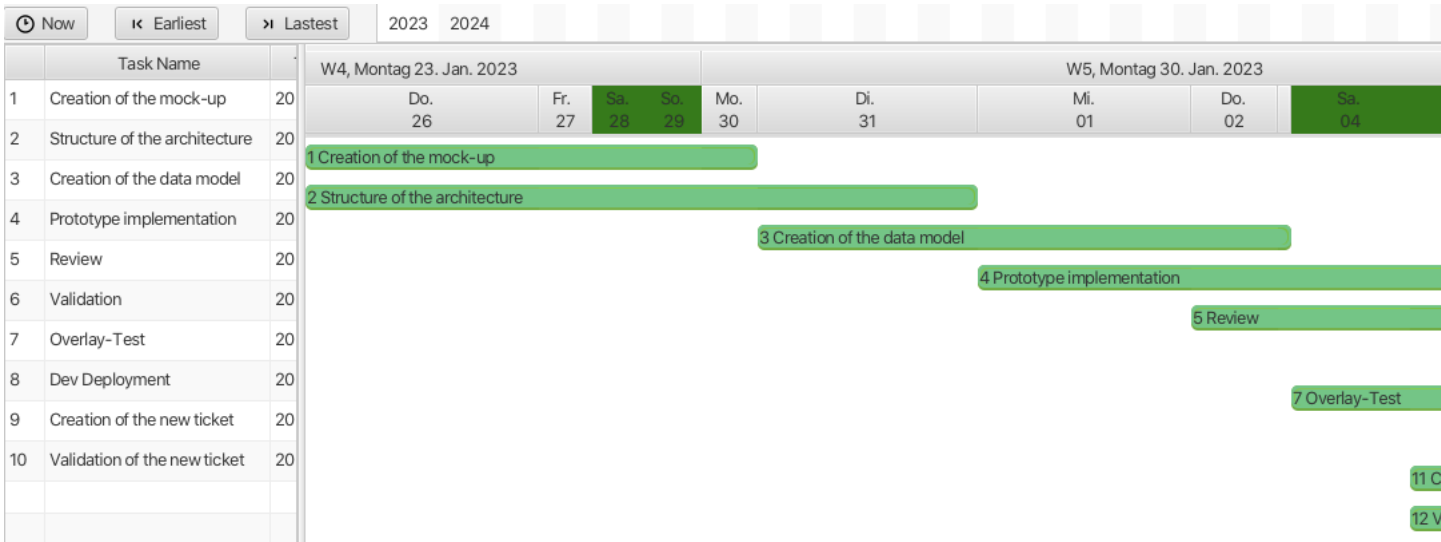
- GanttTableControl: wird auf der rechten Seite angezeigt, um eine grafische Darstellung der Modelldaten anzuzeigen.
- DatelineGraphControl: zeigt die grafische Ansicht und darüber die Zeitleiste an. die zeitleiste zeigt Tage, Wochen, Monate, Jahre.
- GanttChartHbox: wird über dem GanttTableControl und dem DatelineGraphControl angezeigt.

Die Abbildung (4.1) zeigt, wie ein Gantt-Diagramm aussieht. Dem Entwickler steht ein ziemlich flexibles Programm zur Verfügung. Er kann neue Spalten in die Tabelle einfügen, eine Spalte löschen, die Nummer der Aufgabe oder die Nummer und den Namen der Aufgabe im Diagramm anzeigen(4.2) Die gesamte CSS-Datei kann überschrieben werden.

Algorithmus 4.1 Gantt-Diagramm



Algorithmus 4.2 Mit dem Name in Balken Gantt-Diagramm



4.2 Die Steuerelemente

4.2.1 Das GanttTableControl

Das GanttTableControl ist ein generisches JavaFX- Steuerelement zur Darstellung von Activitäten (Task, Resource). Die Aufgabentabelle(siehe 4.2) besteht aus mehreren Spalten. Die erste Spalte "Task Name" zeigt den tatsächlichen Namen der Aufgabe an. Dann die Spalte "Task Start", die das Startdatum der Aufgabe anzeigt, und die Spalte "Task End", die das Enddatum der Aufgabe anzeigt. Dann die Spalte "Task Duration" zeigt die Dauer der Aufgabe, die Spalte "Task State" zeigt Zustand der Aufgabe, der sein kann: RUNNING, HALTED oder TERMINATED. Dann die Spalte "Progress", die uns den Fortschritt der Aufgabe in Echtzeit anzeigt. Die Spalte "Task Description" anzeigt die Beschreibung der Aufgabe, die Spalte "Task Priority", die die Priorität der Aufgabe anzeigt. Die letzte Spalte "IsCritical" zeigt mithilfe einer Checkbox an, ob die Aufgabe kritisch ist oder nicht. Im unteren Bereich des BorderPane gibt es eine Legende, die den Prioritätsgrad jeder Farbe in der Spalte "Task Priority" angibt.

	Task Name	Task Start	Task End	Task Duration	Priority	Task State	Progress	Task Description	IsCritical
1	Creation of the mock-up	2023-01-26	2023-01-30	5		HALTED		design of the Interface	☑
2	Structure of the architecture	2023-01-26	2023-01-31	6		HALTED		Work structuring	☑
3	Creation of the data model	2023-01-31	2023-02-03	4		HALTED		UML-Diagram	☑
4	Prototype implementation	2023-02-01	2023-02-10	10		HALTED			☐
5	Review	2023-02-02	2023-02-05	4		HALTED		test the prototyping	☑
6	Validation	2023-02-06	2023-02-09	4		HALTED		design of the Interface	☑
7	Overlay-Test	2023-02-04	2023-02-08	5		HALTED		Work structuring	☐
8	Dev Deployment	2023-02-08	2023-02-15	8		HALTED		UML-Diagram	☑
9	Creation of the new ticket	2023-01-20	2023-02-11	23		RUNNING		design of the Interface	☑
10	Validation of the new ticket	2023-01-05	2023-01-09	5		TERMINATED		Work structuring	☐

Low Priority Medium Priority Hight Priority

Abbildung 4.2: Tabelle des Gantt-Aufgabe

4.2.2 Das DatelineGraphControl

Das DatelineGraphControl enthält eine Diagramms, in dem die Aktivitäten visualisiert werden. Der DatelineGraphControl ist ein generisches JavaFX- Steuerelement. Die Zeitachse des Diagramms zeigt den Tag, an dem die jeweilige Aufgabe geplant wird. Die Zeitachse ist in zwei Reihen unterteilt: Die erste Reihe zeigt die Wochen an, die zweite die Tage. Die Wochenenden sind farbig und es gibt auch eine farbige Linie, die den aktuellen Tag darstellt.

Abbildung 4.3: Die Zeitachse des DatelineGraphControls

4.2.3 Das GanttChartHbox

Ein Menü ermöglicht es Ihnen, bestimmte Aktionen im Diagramm auszuführen. Mit der Schaltfläche "Now" können Sie zum aktuellen Datum scrollen. Mit dem Button "Earliest" gelangen Sie zum ersten Tag des Diagramms und mit dem Button "Lastest" zum letzten Tag des Diagramms. Durch Anklicken der Jahre in der Liste wird bis zum ersten Tag des jeweiligen Jahres gescrollt.

Abbildung 4.4: Menu des Gantt-Diagramms

5 Zusammenfassung und Fazit

Das erstellte Framework erfüllt alle in der Anforderungsanalyse definierten Bedingungen. Im Rahmen der Arbeit wurde jedoch wegen Zeit und Umfang auf einige Features verzichtet. Das Thema Framework ist ein sehr vielfältiges und komplexes Thema, das hier auf kleinem Raum und aus einer gezielten Perspektive dargestellt wurde. Eines der Ziele dieser Masterarbeit war zunächst die Einführung in JavaFX, das Gantt-Diagramm und die Konzeption eines Opensource-Frameworks. Anschließend wurde JavaFX vorgestellt und die wichtigsten technischen Aspekte erläutert. Neben der JavaFX-Architektur und der Beschreibung der JavaFX-API-Pakete und von FXML lag der Schwerpunkt auf der zugrunde liegenden Logikschicht. Im zweiten Teil wurde ein praktisches Beispiel für die Bearbeitung von Gantt-Projekten auf der Grundlage der vorgestellten Techniken entwickelt. Außerdem wurden die Anforderungen und Konzepte entwickelt, die für die Umsetzung des Praxisbeispiels erforderlich sind. Die Realisierung eines JavaFX-Frameworks orientiert sich an vielen Bereichen der Entwicklung von Benutzeroberflächen. Aufgrund der geringen Einschränkungen durch die System- und Plattformunabhängigkeit haben sie das Potenzial, die Landschaft der Technologien neu zu gestalten. Die schnelle Abfolge neuer JavaFX-Versionen zeigt die Bedeutung ihr Framework für die Entwicklung von GUIs. Mit der Unterstützung von JavaFX-Bibliotheken bietet JavaFX zahlreiche Entwicklungsmöglichkeiten. Bei der Entwicklung eines Prototyps für ein Gantt-Framework erwies sich JavaFX als geeignetes Werkzeug, da es genügend Möglichkeiten zur Umsetzung der entwickelten Konzepte bietet. Die von JavaFX-API nicht unterstützten Funktionen können von den Entwicklern mithilfe der darin enthaltenen Komponenten individuell realisiert oder durch Frameworks von Drittanbietern ergänzt werden.

Der entwickelte Prototyp ermöglicht eine schnelle Implementierung von GanttDiagrammen, kann aber noch verbessert werden. Er muss robuster sein und auch die Leistung muss erheblich gesteigert werden. Neue Funktionen wie Aktivitätsabhängigkeiten, der kritische Pfad und die Ressourcenzuweisung können hinzugefügt werden. Der kritische Pfad eines Projekts könnte berechnet werden und die isCritical Spalte in der Tabelle würde für alle Aktivitäten, die sich auf diesem kritischen Pfad befinden, automatisch ein Häkchen setzen. In der Grafik könnten den Aktivitäten Meilensteine hinzugefügt werden, Aktivitätsabhängigkeiten zwischen Aktivitäten hinzugefügt werden und Aufgaben auf kritischen Pfaden mit einer anderen Farbe dargestellt werden. Um einen besseren Überblick über das Gantt-Diagramm zu erhalten, kann die Zoom-Funktion implementiert werden. Die Klasse Activity könnte um Child-

und Parent-Aktivitäten erweitert werden. Schließlich wird das gesamte Projekt implementiert.

Abbildungsverzeichnis

Tabellenverzeichnis

Algorithmenverzeichnis

Nomenklatur

API	Application Programming Interface
CSS	Cascading Style Sheets
DOM	Document Object Model
FXML	XML-basierte Oberflächenbeschreibungssprache
GPU	Graphics Processing Unit
HTML	HyperText Markup Language
IDE	Integrated Development Environment
JAR	Java Archive
JDK	Java Development Kit
JRE	Java Runtime Environment
JVM	Java Virtual Machine
MVC	Model View Controller
RIAs	Rich Internet Applications
SVG	Scalable Vector Graphics
UI	User Interface
XML	Extensible Markup Language

Literaturverzeichnis

[Gan22] GANTT.COM: *Was ist ein Gantt Diagramm?* 2022. – Angesehen am 24.08.2022

[Gra19] GRANDJEAN, Florian: *Wild Code School.* Juni 2019. – Angehen am 11.11.2022

[GÃŒ18] GÃŒNSTER, Kai: *SchrÃ¶dinger lernt HTML5, CSS und JavaScript.* Rheinwerk Verlag GmbH, 2018 `https://www.ebook.de/de/product/34817960/kai_guenster_schroedinger_lernt_html5_css_und_javascript.html`. – ISBN 9783836268271

[HÃ¶22] HÃ¶LLER, Christoph: *Angular.* Rheinwerk Verlag GmbH, 2022 `https://www.ebook.de/de/product/41867988/christoph_hoeller_angular.html`. – ISBN 9783836282437

[Jai11] JAISWAL, Sonoo: *Javatpoint.* 2011. – Angehen am 13.11.2022

[KN] KOCH, Marianne B. ; NORA: *Rich Internet Applications*

[Lem16] LEMMERMANN, Dirk: *FlexGanttFX Developer Manual.* 2016. – Angehen am 16.11.2022

[Sof11] SOFTSEA.COM: *SwiftGantt.* Februar 2011. – Angehen am 07.11.2022

[SpÃ20] SPÃ€TH, Peter: *Beginning Java MVC 1.0.* Apress, 2020 `https://www.ebook.de/de/product/41241265/peter_spaeth_beginning_java_mvc_1_0.html`. – ISBN 9781484262801

[Ste14] STEYER, Ralph: *EinfÃŒhrung in JavaFX: Moderne GUIs fÃŒr RIAs und Java-Applikationen.* Springer Vieweg. in Springer Fachmedien Wiesbaden GmbH, 2014. – ISBN 978–3–658–02836–7

[tut11a] TUTORIALSPOINT: *JavaFX - Architectur.* Februar 2011. – Angehen am 17.11.2022

[tut11b] TUTORIALSPOINT: *JavaFX-Application Structure.* 2011. – Angehen am 17.11.2022

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit zum Thema

"Konzeption eines JavaFX-Frameworks zur Erstellung von Gantt-Diagrammen - Anforderungen, Architektur und Implementierung eines Prototyps"

vollkommen selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe. Die Arbeit wurde in dieser oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt.

Brandenburg an der Havel, den 26. Januar 2023

Unterschrift

Dallysse Laure Djouhou Tamdjo