Grafischer Darstellungsrechner

Authors: Priller Patrick, Mairhofer David, Pernthaler Daniel

stpripat@bx.fallmerayer.it, stmaidav@bx.fallmerayer.it, stperdan@bx.fallmerayer.it

Dokumantation

Informatik
Klasse 4AT

Professor: Rainer Ulrich

Brixen, den 03. Mai 2023

Inhaltsverzeichnis

1 Themenbeschreibung **3**
1.1 Projektziele . . . 3
1.2 Unsere Vision . . . 3
1.3 Potenziell nutzbare APIs . . . 4
1.3.1 mxParser . . . 4
1.3.2 JScience . . . 4
1.4 Technologien und Tools . . . 5
1.5 Programmiersprache und Plattform . . . 5
1.6 APIs und Bibliotheken . . . 5
1.7 Entwicklungsumgebung und Tools . . . 5

2 Vorgehensmodell: Kanban **7**
2.1 Grundlagen von Kanban . . . 7
2.2 Anwendung von Kanban . . . 8
2.3 Vorteile und Erfahrungen . . . 8

3 Anforderungsanalyse **9**
3.1 Funktionale Anforderungen . . . 9
3.2 Nicht funktionale Anforderungen . . . 9
3.3 Aktivitätsdiagramm . . . 9

4 Use-Case-Diagramm **10**
4.1 Erklärung . . . 10
4.2 Diagramm . . . 10

5 Systemarchitektur **11**
5.1 Übersicht . . . 11
5.2 Frontend . . . 11
5.3 Backend . . . 11
5.4 Klassendiagramm . . . 12

6 Umsetzung (Programmierung) **14**
6.1 Backend-Entwicklung . . . 14
6.2 Frontend-Entwicklung . . . 14
6.2.1 UI-Prototyp . . . 14
6.3 Integration von APIs . . . 14
6.4 Teststrategie . . . 14

7 Benutzung (User Guide) **14**
7.1 Installation und Einrichtung . . . 14
7.2 Bedieung der Anwendung . . . 14
7.3 Beispiele . . . 14

8 Fazit **14**
8.1 Zusammenfassung . . . 14
8.2 Erreichte Ziele . . . 14
8.3 Zukünftige Erweiterungen . . . 14

9 Anhang **14**
9.1 Arbeitstagebuch (Planung) 14
9.2 Arbeitstagebuch (Umsetzung) 14
9.3 Quellcode 14
9.4 Screenshots 14
9.5 Lizenzinformationen 14

1 Themenbeschreibung

1.1 Projektziele

In diesem Projekt verfolgen wir das Ziel, eine benutzerfreundliche und leistungsfähige grafische Funktionen-Plotting-Anwendung zu entwickeln, die es Benutzern ermöglicht, mathematische Funktionen und ihre Ableitungen zu visualisieren und zu analysieren. Um dieses Ziel zu erreichen, haben wir uns folgende spezifische Ziele gesetzt:

- Entwicklung einer benutzerfreundlichen Oberfläche, die es Benutzern ermöglicht, Funktionen einfach einzugeben und zu bearbeiten.
- Unterstützung einer breiten Palette von mathematischen Funktionen, einschließlich linearer, quadratischer, exponentieller und trigonometrischer Funktionen.
- Integration von APIs wie mxParser und JScience, um die Funktionalität und Benutzerfreundlichkeit der Anwendung zu erweitern.
- Implementierung von Funktionen zur Berechnung und Anzeige von Ableitungen und Kurvendiskussionen.
- Bereitstellung von Werkzeugen zur Anpassung der Darstellung von Funktionen und des Plotbereichs.
- Ermöglichung von Zoomen und Verschieben des Plotbereichs, um die Anzeige von Funktionen flexibel zu gestalten.
- Entwicklung eines effektiven Testverfahrens, um sicherzustellen, dass die Anwendung korrekt funktioniert und benutzerfreundlich ist.
- Erstellung einer umfassenden Dokumentation, die den Entwicklungsprozess, die Systemarchitektur und die Benutzung der Anwendung beschreibt.

Durch das Erreichen dieser Ziele wollen wir eine leistungsstarke und benutzerfreundliche Plotter-Anwendung schaffen, die sowohl für den Bildungsbereich als auch für professionelle Anwender wertvoll ist und die Visualisierung und Analyse von mathematischen Funktionen vereinfacht.

1.2 Unsere Vision

Unsere Vision für das Plotter-Projekt war es, eine benutzerfreundliche und leistungsfähige Java-basierte grafische Funktionen-Plotting-Anwendung zu entwickeln, die es Benutzern ermöglicht, mathematische Funktionen und ihre Ableitungen zu visualisieren. Wir wollten eine breite Palette von Funktionen unterstützen, einschließlich linearer, quadratischer, exponentieller und trigonometrischer Funktionen. Dabei sollte die Anwendung die mxParser-Bibliothek verwenden, um mathematische Ausdrücke zu analysieren und ihre Werte zu berechnen.

Zu Beginn des Projekts haben wir uns darauf konzentriert, die Kernfunktionen der Anwendung zu entwickeln, wie das Plotten von Funktionen in einer benutzerfreundlichen Oberfläche, das Berechnen und Anzeigen von Ableitungen von Funktionen, das Anwenden von Konstanten und Variablen auf Funktionen,

das Ein- und Ausblenden von Funktionen im Graphen und das Anpassen des Erscheinungsbilds des Plotbereichs. Zudem haben wir die Möglichkeit geschaffen, die Ansicht des Plotbereichs zu vergrößern und zu verkleinern.

Wir wollten sicherstellen, dass die Anwendung einfach zu installieren und auszuführen ist, weshalb wir uns für den Einsatz von Maven zur Erstellung des Projekts entschieden haben. Um das Projekt schnell und einfach zu nutzen, haben wir detaillierte Anweisungen zur Installation der erforderlichen Abhängigkeiten, zum Erstellen des Projekts mit Maven und zum Ausführen der Anwendung bereitgestellt.

Insgesamt war unsere Vision, ein leistungsstarkes und benutzerfreundliches Plotter-Tool zu entwickeln, das sowohl für den Bildungsbereich als auch für professionelle Anwender wertvoll ist und die Visualisierung und Analyse von mathematischen Funktionen vereinfacht.

1.3 Potenziell nutzbare APIs

In unserem Plotter-Projekt haben wir verschiedene APIs in Betracht gezogen, um die Funktionalität und Benutzerfreundlichkeit der Anwendung zu erweitern. Zwei der vielversprechendsten APIs, die wir untersucht haben, sind mxParser und JScience.

1.3.1 mxParser

mxParser ist eine vielseitige und leistungsfähige Bibliothek zur Analyse und Berechnung mathematischer Ausdrücke. Die Bibliothek ist in Java geschrieben und unterstützt eine breite Palette von Funktionen, einschließlich algebraischer, trigonometrischer, exponentieller und logarithmischer Funktionen. Wir haben mxParser in unserem Plotter-Projekt verwendet, um mathematische Ausdrücke zu analysieren und ihre Werte zu berechnen. Die offizielle Website von mxParser bietet umfassende Dokumentation und Beispiele zur Verwendung der Bibliothek: `http://mathparser.org/`

1.3.2 JScience

JScience ist eine umfangreiche Java-Bibliothek, die verschiedene wissenschaftliche Disziplinen abdeckt, darunter Mathematik, Physik, Chemie und Informatik. Die Bibliothek bietet eine Reihe von Funktionen, die für unser Plotter-Projekt nützlich sein könnten, wie z.B. Unterstützung für komplexe Zahlen, Einheitenkonversion und lineare Algebra. JScience könnte verwendet werden, um die Funktionalität unserer Anwendung zu erweitern und die Unterstützung für zusätzliche mathematische Funktionen und Operationen zu verbessern. Weitere Informationen zu JScience finden Sie auf der offiziellen Website: `http://jscience.org/`

Durch die Integration dieser APIs in unser Plotter-Projekt können wir die Benutzererfahrung und die Funktionalität unserer Anwendung weiter verbessern und eine breitere Palette von mathematischen Funktionen und Operationen unterstützen.

1.4 Technologien und Tools

Bei der Entwicklung des Plotters haben wir eine Reihe von Technologien und Tools verwendet, um eine solide und benutzerfreundliche Anwendung zu erstellen. In diesem Abschnitt werden die wichtigsten Technologien und Tools vorgestellt, die wir im Projekt eingesetzt haben:

1.5 Programmiersprache und Plattform

- **Java**: Wir haben uns für die Programmiersprache Java entschieden, da sie plattformübergreifend, objektorientiert und gut geeignet für die Entwicklung von grafischen Anwendungen ist. Java bietet eine breite Palette von Bibliotheken und Frameworks, die die Implementierung der benötigten Funktionen erleichtern.
- **JavaFX**: Für die Erstellung der grafischen Benutzeroberfläche haben wir JavaFX verwendet, ein modernes Framework für die Entwicklung von Desktop-Anwendungen in Java. JavaFX ermöglicht die Erstellung von ansprechenden und interaktiven Benutzeroberflächen und bietet viele nützliche UI-Komponenten, die für unser Projekt relevant sind.

1.6 APIs und Bibliotheken

- **mxParser**: Diese leistungsfähige Bibliothek wird verwendet, um mathematische Ausdrücke zu analysieren und ihre Werte zu berechnen. mxParser unterstützt eine Vielzahl von Funktionen, was es uns ermöglicht, eine breite Palette von mathematischen Funktionen in unserer Anwendung zu unterstützen.
- **JScience**: JScience ist eine umfangreiche Java-Bibliothek, die verschiedene wissenschaftliche Disziplinen abdeckt. Durch die Integration von JScience in unser Projekt könnten wir die Funktionalität unserer Anwendung erweitern und zusätzliche mathematische Funktionen und Operationen unterstützen.

1.7 Entwicklungsumgebung und Tools

- **IntelliJ IDEA**: Wir haben die IntelliJ IDEA-Entwicklungsumgebung verwendet, um unseren Code zu schreiben, zu testen und zu debuggen. IntelliJ IDEA bietet eine Vielzahl von Funktionen, die die Entwicklung von Java- und JavaFX-Anwendungen erleichtern, einschließlich Code-Vervollständigung, Refactoring-Tools und Integration mit Versionskontrollsystemen wie Git.
- **Maven**: Maven ist ein Build-Management-Tool, das wir verwendet haben, um den Build-Prozess zu automatisieren und die Abhängigkeiten unseres Projekts zu verwalten. Maven erleichtert die Integration von externen Bibliotheken wie mxParser und JScience und stellt sicher, dass unser Projekt auf verschiedenen Plattformen und Systemen konsistent gebaut und ausgeführt werden kann.

- **Git**: Für die Versionskontrolle und das Teamwork haben wir Git verwendet. Git ermöglicht es uns, Änderungen am Code nachzuvollziehen, verschiedene Versionen der Anwendung zu verwalten und die Zusammenarbeit zwischen Teammitgliedern zu erleichtern.

Durch die Kombination dieser Technologien und Tools konnten wir eine leistungsfähige und benutzerfreundliche Anwendung entwickeln, die es Benutzern ermöglicht, mathematische Funktionen effizient zu visualisieren und zu analysieren.

2 Vorgehensmodell: Kanban

2.1 Grundlagen von Kanban

Kandan ist ein visuelles Projektmanagement-System, das ursprünglich in der Automobilindustrie entwickelt wurde und auf der Kanban-Methode basiert. Für unser Plotter-Projekt wurde Kandan gewählt, da es eine hohe Flexibilität bietet und es uns ermöglicht, einen guten Überblick über den Fortschritt des Projekts zu behalten.

Kandan besteht aus einer Reihe von Spalten, die verschiedene Phasen des Projekts repräsentieren, z. B. SZu erledigen", În Arbeitünd "Fertig". Innerhalb dieser Spalten werden Aufgaben als Karten dargestellt, die Informationen zu Priorität, Verantwortlichen und Fälligkeitsdatum enthalten. Karten werden von einer Spalte zur nächsten verschoben, wenn sie von einer Phase zur nächsten fortschreiten.

Dieses System erleichtert die Verfolgung des Fortschritts und hilft dabei, Engpässe oder Verzögerungen schnell zu identifizieren. Es fördert außerdem eine kontinuierliche Verbesserung, indem es Teams dazu anregt, regelmäßig den Workflow zu überprüfen und anzupassen.

Die folgende Abbildung zeigt ein Beispiel für ein Kandan-Board:

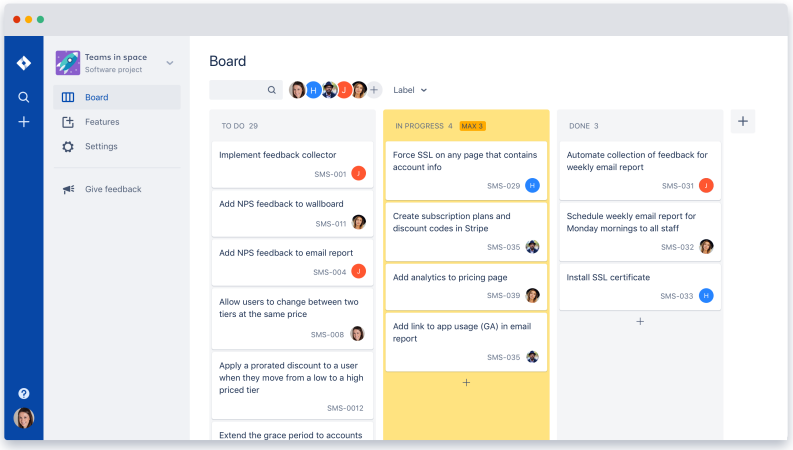

Abbildung 1: Beispiel eines Kandan-Boards

Insgesamt bietet Kandan eine einfache und effektive Möglichkeit, den Fortschritt unseres Plotter-Projekts zu verwalten und sicherzustellen, dass wir kontinuierlich Verbesserungen vornehmen und auf unsere Ziele hinarbeiten.

2.2 Anwendung von Kanban

Im Verlauf unseres Projekts haben wir uns für das Vorgehensmodell Kanban entschieden, um den Entwicklungsprozess zu organisieren und zu steuern. Kanban ist eine agile Methode, die auf kontinuierlicher Verbesserung und Flexibilität basiert. Es hilft, den Arbeitsfluss zu visualisieren, Engpässe zu identifizieren und die Produktivität des Teams zu steigern.

Wir haben ein Kanban-Board verwendet, um unsere Aufgaben und den Fortschritt im Projekt darzustellen. Unser Kanban-Board war in mehrere Spalten unterteilt, die den verschiedenen Phasen des Arbeitsablaufs entsprechen, z.B. 'To Do', 'In Progress', 'Review' und 'Done'. Jede Aufgabe wurde als Karte auf dem Board dargestellt, und wir haben die Karten zwischen den Spalten verschoben, um den Fortschritt der Aufgaben zu visualisieren.

Um den Entwicklungsprozess effektiv zu gestalten, haben wir unser Programm in kleinere Teile aufgeteilt, die wir Vorgänge (in Jira, dem von uns verwendeten Projektmanagement-Tool, genannt) nennen. Dies hat dazu beigetragen, dass wir uns auf bestimmte Aspekte der Anwendung konzentrieren und diese unabhängig voneinander entwickeln konnten.

In regelmäßigen Stand-up-Meetings haben wir den Status unserer Aufgaben überprüft, Engpässe identifiziert und die Prioritäten im Projekt angepasst. Dadurch konnten wir sicherstellen, dass die wichtigsten Aufgaben stets im Fokus standen und effektiv bearbeitet wurden.

2.3 Vorteile und Erfahrungen

Durch die Anwendung von Kanban in unserem Projekt haben wir verschiedene Vorteile erfahren:

- **Flexibilität**: Kanban hat es uns ermöglicht, schnell auf Änderungen im Projektumfeld oder in den Anforderungen zu reagieren. Dadurch konnten wir unsere Prioritäten anpassen und sicherstellen, dass wir uns auf die wichtigsten Aspekte des Projekts konzentrierten.
- **Transparenz**: Durch die Visualisierung des Arbeitsablaufs auf dem Kanban-Board konnten alle Teammitglieder den aktuellen Status der Aufgaben und den Fortschritt des Projekts einsehen. Dies hat zu einer besseren Kommunikation und Zusammenarbeit im Team geführt.
- **Effizienz**: Die Anwendung von Kanban und die Aufteilung des Programms in kleinere Vorgänge haben dazu beigetragen, Engpässe im Arbeitsablauf zu identifizieren und zu beseitigen. Dies hat die Produktivität des Teams gesteigert und dazu geführt, dass Aufgaben schneller und effizienter abgeschlossen wurden.
- **Kontinuierliche Verbesserung**: Die regelmäßigen Stand-up-Meetings und die Möglichkeit, den Arbeitsablauf anzupassen, haben dazu beigetragen, kontinuierliche Verbesserungen im Projektprozess zu fördern. Dadurch konnten wir mögliche Probleme frühzeitig erkennen und Lösungen entwickeln, um diese zu beheben.

Insgesamt hat die Anwendung von Kanban in unserem Projekt dazu beigetragen, einen effizienten und flexiblen Entwicklungsprozess zu gestalten, der es uns ermöglichte, eine erfolgreiche und benutzerfreundliche Anwendung zu entwickeln.

3 Anforderungsanalyse

3.1 Funktionale Anforderungen

3.2 Nicht funktionale Anforderungen

3.3 Aktivitätsdiagramm

4 Use-Case-Diagramm

4.1 Erklärung

Das Use-Case-Diagramm beschreibt die Funktionalität des Plotters und zeigt, wie Benutzer mit der Anwendung interagieren können. Der Benutzer kann in ein Textfeld seine gewünschte Funktion eingeben und mithilfe von Buttons Sonderzeichen in das Textfeld schreiben. Der Benutzer hat auch die Möglichkeit, verschiedene Buttons zu verwenden, um die Funktion zu bearbeiten, wie zum Beispiel die Sichtbarkeit, Ableitung, Kurvendiskussion oder die Funktion zu löschen. Zusätzlich kann der Benutzer den Anzeigebereich der Funktion mithilfe von Zoomen oder Verschieben verändern.

4.2 Diagramm

Das folgende Diagramm zeigt das Use-Case-Diagramm für unseren Plotter:

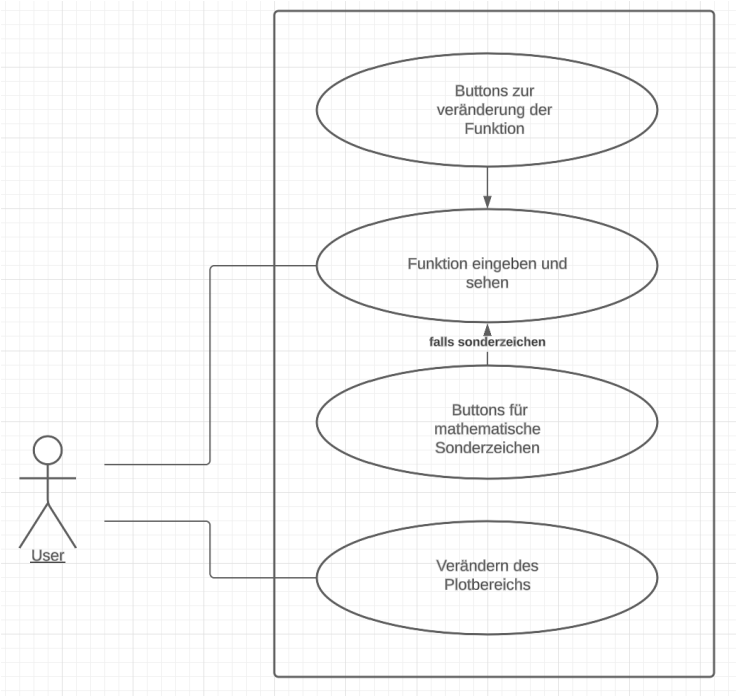

Abbildung 2: Use-Case-Diagramm des Plotters

Das Diagramm veranschaulicht die verschiedenen Aktionen, die ein Benutzer ausführen kann, um mit der Anwendung zu interagieren und mathematische Funktionen zu visualisieren und zu analysieren.

5 Systemarchitektur

5.1 Übersicht

Die Systemarchitektur des Projekts ist in zwei Hauptkomponenten unterteilt: das Frontend und das Backend. Das Frontend ist für die Benutzeroberfläche und die Interaktionen mit dem Benutzer verantwortlich, während das Backend die Berechnungen und die Verarbeitung der eingegebenen mathematischen Funktionen übernimmt. Zusammen bieten sie eine effektive und benutzerfreundliche Anwendung zum Plotten von mathematischen Funktionen und ihren Ableitungen.

5.2 Frontend

Das Frontend der Anwendung ist mit JavaFX implementiert, einer leistungsstarken und flexiblen Bibliothek für die Erstellung grafischer Benutzeroberflächen. Es bietet eine Vielzahl von UI-Komponenten und Steuerelementen, die für die Implementierung der Anwendung erforderlich sind, einschließlich Szenen, Gruppen, Schaltflächen, Eingabefelder und mehr. Das Frontend ist für die Organisation und Darstellung der Benutzeroberfläche verantwortlich, einschließlich der Eingabefelder für Funktionen, Schaltflächen für mathematische Operationen und Anzeige der Funktionsplots im Plotbereich.

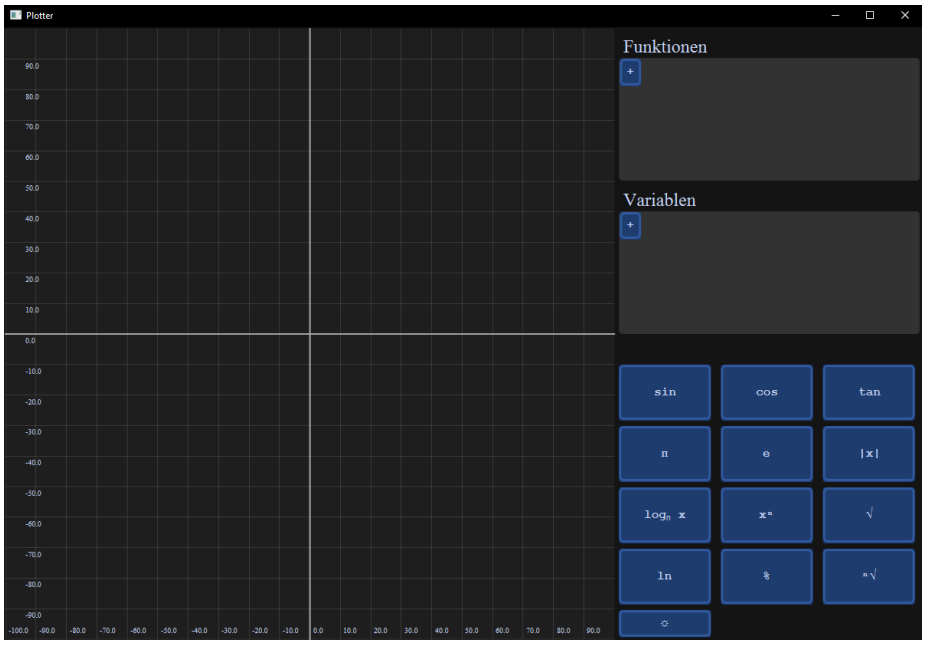

Abbildung 3: Screenshot der Benutzeroberfläche der Anwendung

5.3 Backend

Das Backend der Anwendung ist für die Verarbeitung der eingegebenen mathematischen Funktionen, Berechnung der Ableitungen und Aktualisierung des Plotbereichs verantwortlich. Es verwendet die mxparser-Bibliothek, um mathematische Ausdrücke zu parsen und auszuwerten, und die Hauptklassen der Anwendung, um die grafischen Darstellungen der Funktionen und ihrer Ableitungen zu erzeugen. Die Klassen DerivativeCalculator, PlotFunction und PlotArea sind die Hauptkomponenten des Backends und arbeiten zusammen, um die gewünschte Funktionalität bereitzustellen.

5.4 Klassendiagramm

Das Klassendiagramm zeigt die Struktur und die Beziehungen zwischen den Hauptklassen der Anwendung. Es hilft dabei, die Verantwortlichkeiten jeder Klasse und die Interaktionen zwischen den Klassen besser zu verstehen.

- **DerivativeCalculator**: Diese Klasse ist verantwortlich für die Berechnung der Ableitungen der eingegebenen mathematischen Funktionen. Sie verwendet die mxparser-Bibliothek und interagiert mit der PlotFunction-Klasse, um die Ableitungen zu berechnen und bereitzustellen.
- **Layout**: Diese Klasse ist ein benutzerdefiniertes JavaFX Group-Objekt, das die Benutzeroberfläche der Anwendung organisiert. Es enthält die Komponenten wie Eingabefelder, Schaltflächen und den Plotbereich und stellt sicher, dass diese Elemente auf dem Bildschirm richtig angeordnet sind.
- **PlotArea**: Diese Klasse ist ein benutzerdefiniertes JavaFX Group-Objekt, das den Plotbereich der Anwendung darstellt. Es verwaltet das Hinzufügen, Entfernen und Aktualisieren von Funktionen, Variablen und Gitterlinien im Plotbereich.
- **PlotFunction**: Die PlotFunction-Klasse verwaltet die grafische Darstellung einer mathematischen Funktion. Sie berechnet die Positionen der Liniensegmente, die die Kurve der Funktion approximieren, und steuert die Sichtbarkeit der Funktion im Plotbereich.
- **Plotter**: Die Plotter-Klasse ist die Hauptklasse der JavaFX-Anwendung und steuert den gesamten Programmablauf. Sie verwendet die mxparser-Bibliothek, um die eingegebenen mathematischen Funktionen zu analysieren und auszuwerten.
- **ScrollPaneFunctionsElement**: Diese Klasse stellt ein einzelnes Funktionselement in der ScrollPane dar, einschließlich eines Textfelds zur Eingabe der Funktion und Schaltflächen zum Berechnen der Ableitung, Löschen der Funktion, Steuern der Sichtbarkeit und Anzeigen der Funktionsanalyse.
- **ScrollPaneVariablesElement**: Diese Klasse stellt ein einzelnes Variablenelement in der ScrollPane dar, einschließlich eines Textfelds zur Eingabe des Variablenwerts und einer Schaltfläche zum Löschen der Variable.

Die Klassen interagieren miteinander, um die Hauptfunktionalitäten der Anwendung bereitzustellen. Die Plotter-Klasse agiert als zentraler Koordinator, der die anderen Klassen steuert und verwaltet, um das Plotten von Funktionen, Berechnung von Ableitungen und Aktualisierung der Benutzeroberfläche zu ermöglichen.
Im Folgenden finden Sie das Klassendiagramm für das Projekt:

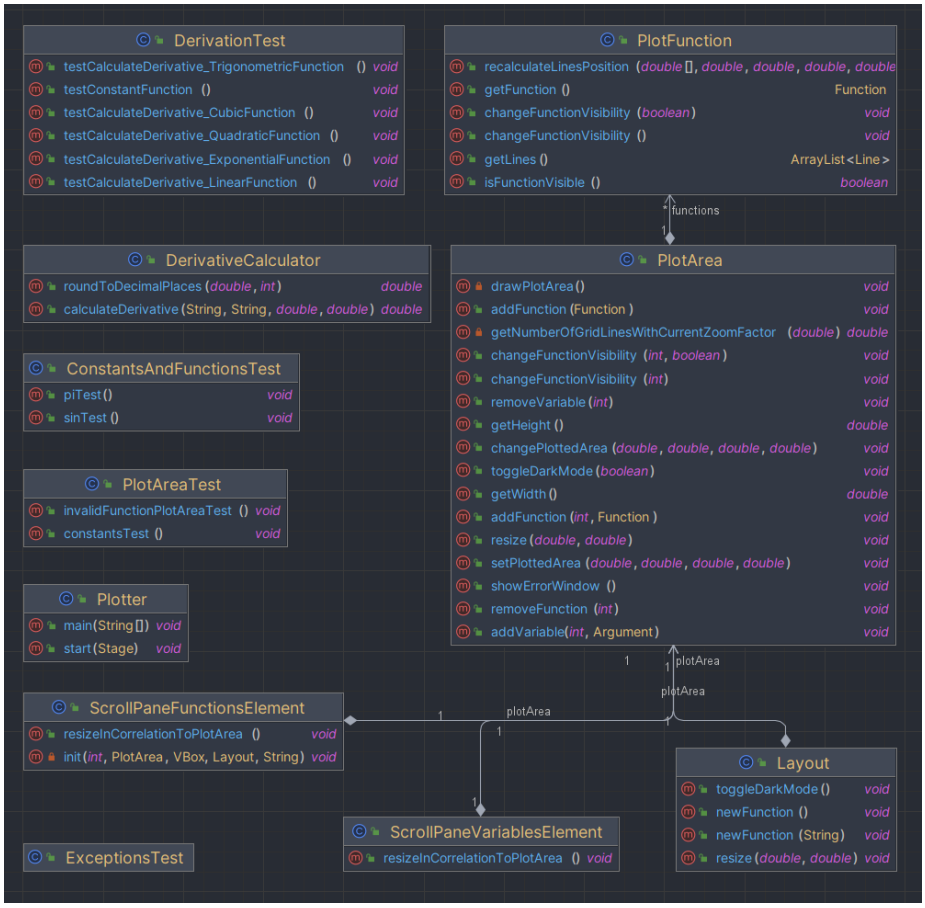

Abbildung 4: Klassendiagramm der Hauptklassen

6 Umsetzung (Programmierung)

6.1 Backend-Entwicklung

6.2 Frontend-Entwicklung

6.2.1 UI-Prototyp

6.3 Integration von APIs

6.4 Teststrategie

7 Benutzung (User Guide)

7.1 Installation und Einrichtung

7.2 Bedieung der Anwendung

7.3 Beispiele

8 Fazit

8.1 Zusammenfassung

8.2 Erreichte Ziele

8.3 Zukünftige Erweiterungen

9 Anhang

9.1 Arbeitstagebuch (Planung)

9.2 Arbeitstagebuch (Umsetzung)

9.3 Quellcode

9.4 Screenshots

9.5 Lizenzinformationen