Matheformelparser

Erzeugt von Doxygen 1.8.6

Die Jul 8 2014 22:55:07

Inhaltsverzeichnis

Kapitel 1

Klassen-Verzeichnis

1.1 Auflistung der Klassen

Hier folgt die Aufzählung aller Klassen, Strukturen, Varianten und Schnittstellen mit einer Kurzbeschreibung:

Kapitel 2

Datei-Verzeichnis

2.1 Auflistung der Dateien

Hier folgt die Aufzählung aller Dateien mit einer Kurzbeschreibung:

Kapitel 3

Klassen-Dokumentation

3.1 FormelparserMinimal Klassenreferenz

Die Formelparserklasse in reduzierter Größe.

Öffentliche Methoden

- FormelparserMinimal ()

  Initialisiert die Membervariablen.
- double parse (String s) throws IOException

  Parst übergebenen Ausdruck und gibt dessen Ergebnis zurrück.

Öffentliche, statische Methoden

- static void main (String args[]) throws IOException

  Ruft Parse() mit den Kommandozeilenparametern auf und meldet aufgetretene Fehler.

Öffentliche Attribute

- boolean isValid

  Gibt an, ob der zuletzt geparste Ausdruck gültig gewesen ist.
- double parseResult

  enthält das Ergebnis des zuletzt geparsten Ausdruckes

Private Methoden

- String tokenToString (int token)

  Hilfsfunktion für Debugausgaben.
- void next () throws IOException

  Liest das nächste Token ein.
- void match (int expected) throws IOException

  Prüft, ob das nächste Token das erwartete ist.
- double parseExpr () throws IOException

  Parst die Ableitungsregel: Expr -> Term (("+" | "-") Term)∗.
- double parseTerm () throws IOException

  Parst die Ableitungsregel: Term -> Fact (("∗" | "/") Fact)∗.

- double parseFact () throws IOException

  Parst die Ableitungsregel: Fact -> (("cos" | "sin" | "tan") Trig)∗ | Trig.

- double parseTrig () throws IOException

  Parst die Ableitungsregel: Trig -> "(" Expr ")" | Number.

Private Attribute

- int lookahead

  Vorschau auf Typ des nächsten Tokens.

- StreamTokenizer tokenizer

  Hilfsobjekt zur Aufsplittung der Eingabe in einzelne Token.

3.1.1 Ausführliche Beschreibung

Die Formelparserklasse in reduzierter Größe.

Formelparser.java - ein Parser zum Auswerten arithmetischer Ausdrücke mittels Recursive-Descent-Parser, der auch rechnet

Autor

Copyright 2014 Norman nospam.schwirz at freenet.de

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA.

Funktionsweise

Der gegebene Ausdruck wird mittels folgender Grammatik geparst:

1. Expr -> Term (("+" | "-") Term)∗
2. Term -> Fact (("∗" | "/") Fact)∗
3. Fact -> (("cos" | "sin" | "tan") Trig)∗ | Trig
4. Trig -> "(" Expr ")" | Number

Aufgrund der verwendeten StreamTokenizer- Klasse sind Leerzeichen nach einigen Operatoren nötig aber auch Klammern können helfen.

- Minuszeichen vor Zahlen werden ansonsten als deren Vorzeichen angesehen.
- Operatoren, die aus Buchstaben bestehen, werden sonst nicht richtig ausgewertet.

3.1.2 Beschreibung der Konstruktoren und Destruktoren

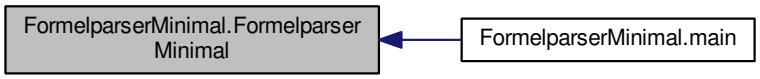
3.1.2.1 FormelparserMinimal.FormelparserMinimal ( )

Initialisiert die Membervariablen.

Benutzt isValid und parseResult.

Wird benutzt von main().

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:



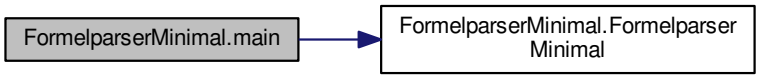
3.1.3 Dokumentation der Elementfunktionen

3.1.3.1 static void FormelparserMinimal.main (String *args[ ]*) throws IOException [static]

Ruft Parse() mit den Kommandozeilenparametern auf und meldet aufgetretene Fehler.

Benutzt FormelparserMinimal().

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:



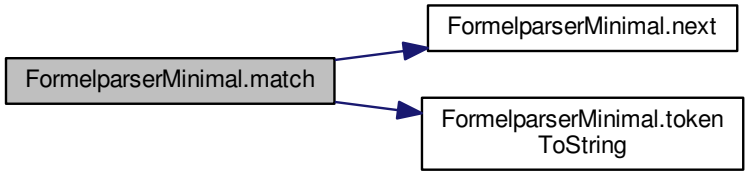
3.1.3.2 void FormelparserMinimal.match (int *expected*) throws IOException [private]

Prüft, ob das nächste Token das erwartete ist.

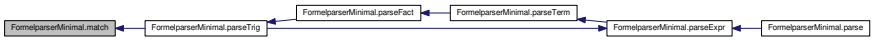
Benutzt lookahead, next() und tokenToString().

Wird benutzt von parseTrig().

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:



Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:



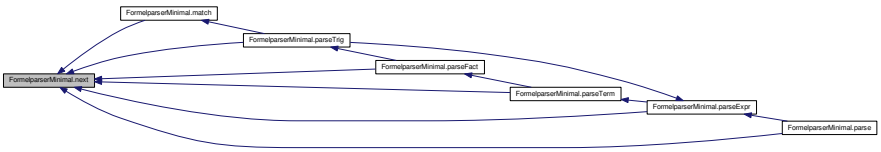
3.1.3.3 void FormelparserMinimal.next ( ) throws IOException [private]

Liest das nächste Token ein.

Benutzt lookahead.

Wird benutzt von match(), parse(), parseExpr(), parseFact(), parseTerm() und parseTrig().

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

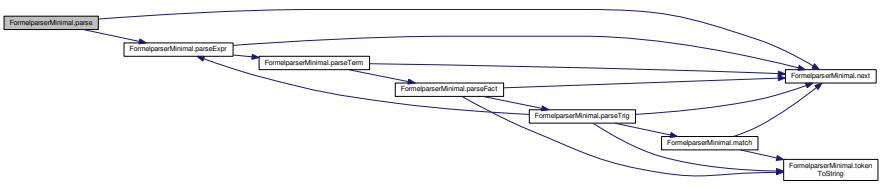


3.1.3.4 double FormelparserMinimal.parse (String *s*) throws IOException

Parst übergebenen Ausdruck und gibt dessen Ergebnis zurrück.

Benutzt isValid, lookahead, next(), parseExpr(), parseResult und tokenizer.

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:



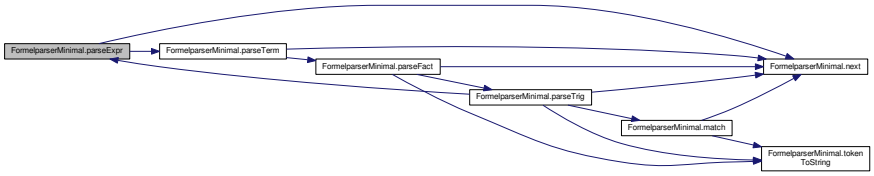
3.1.3.5 double FormelparserMinimal.parseExpr ( ) throws IOException [private]

Parst die Ableitungsregel: Expr -> Term (("+" | "-") Term)∗.

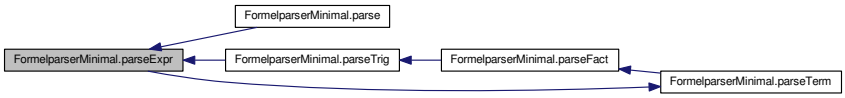
Benutzt lookahead, next() und parseTerm().

Wird benutzt von parse() und parseTrig().

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:



Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:



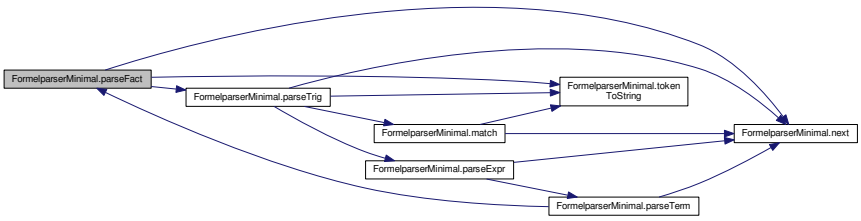
3.1.3.6 double FormelparserMinimal.parseFact ( ) throws IOException [private]

Parst die Ableitungsregel: Fact -> (("cos" | "sin" | "tan") Trig)∗ | Trig.

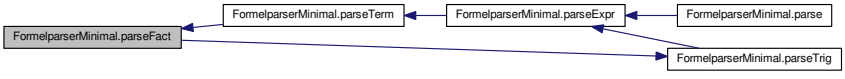
Benutzt lookahead, next(), parseTrig() und tokenToString().

Wird benutzt von parseTerm().

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:



Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:



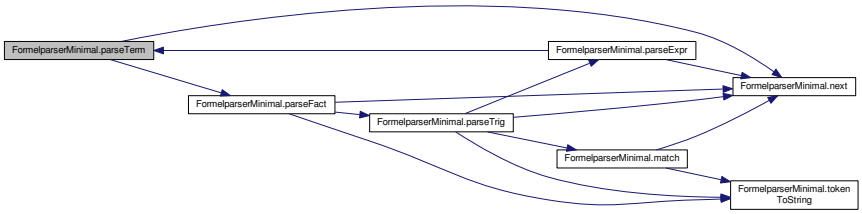
3.1.3.7 double FormelparserMinimal.parseTerm ( ) throws IOException [private]

Parst die Ableitungsregel: Term -> Fact (("∗" | "/") Fact)∗.

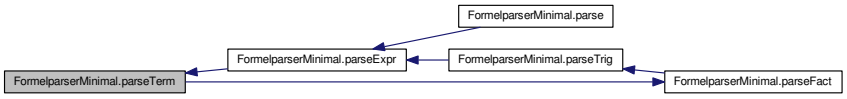
Benutzt lookahead, next() und parseFact().

Wird benutzt von parseExpr().

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:



Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:



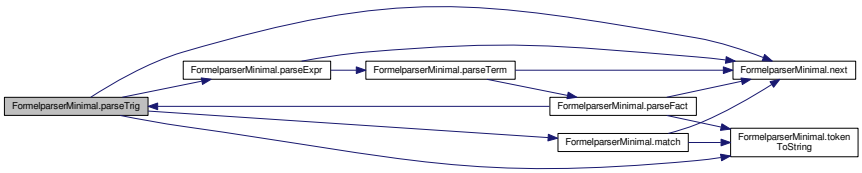
3.1.3.8 double FormelparserMinimal.parseTrig ( ) throws IOException [private]

Parst die Ableitungsregel: Trig -> "(" Expr ")" | Number.

Benutzt lookahead, match(), next(), parseExpr() und tokenToString().

Wird benutzt von parseFact().

Hier ist ein Graph, der zeigt, was diese Funktion aufruft:



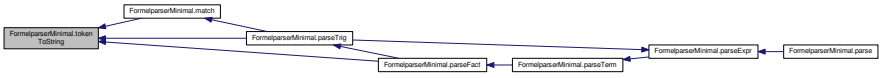
Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:

3.1.3.9 String FormelparserMinimal.tokenToString (int *token*) [private]

Hilfsfunktion für Debugausgaben.

Wird benutzt von match(), parseFact() und parseTrig().

Hier ist ein Graph der zeigt, wo diese Funktion aufgerufen wird:



3.1.4 Dokumentation der Datenelemente

3.1.4.1 boolean FormelparserMinimal.isValid

Gibt an, ob der zuletzt geparste Ausdruck gültig gewesen ist.

Wird benutzt von FormelparserMinimal() und parse().

3.1.4.2 int FormelparserMinimal.lookahead [private]

Vorschau auf Typ des nächsten Tokens.

Wird benutzt von match(), next(), parse(), parseExpr(), parseFact(), parseTerm() und parseTrig().

3.1.4.3 double FormelparserMinimal.parseResult

enthält das Ergebnis des zuletzt geparsten Ausdruckes

Siehe auch

isValid()

Wird benutzt von FormelparserMinimal() und parse().

3.1.4.4 StreamTokenizer FormelparserMinimal.tokenizer [private]

Hilfsobjekt zur Aufsplittung der Eingabe in einzelne Token.

Wird benutzt von parse().

Die Dokumentation für diese Klasse wurde erzeugt aufgrund der Datei:

- FormelparserMinimal.java

Kapitel 4

Datei-Dokumentation

4.1 FormelparserMinimal.java-Dateireferenz

Klassen

- class FormelparserMinimal

  Die Formelparserklasse in reduzierter Größe.

Index