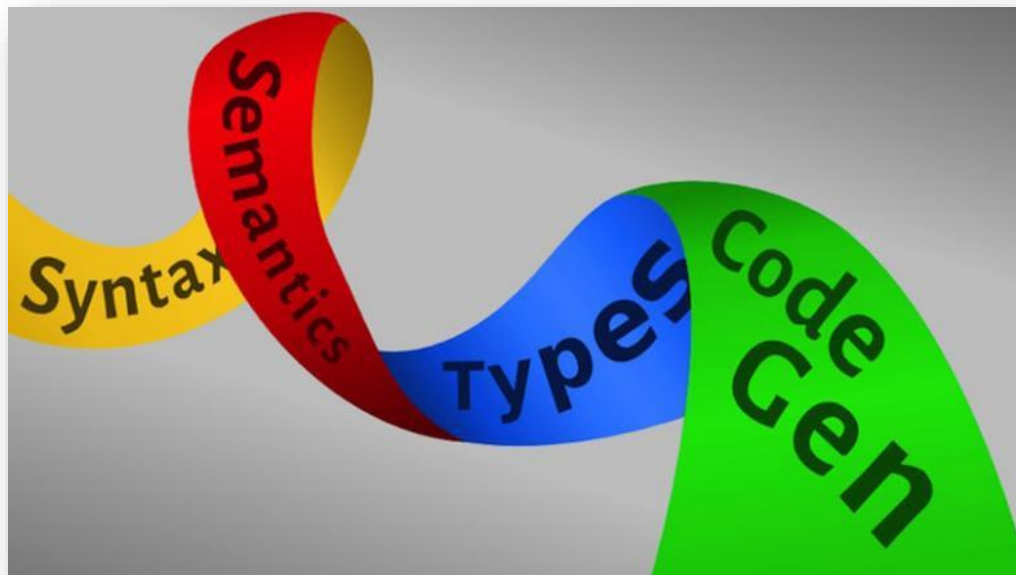


COMPILERS & OPERATING SYSTEMS



Ferhat Bozlak

DHI2V.Sb_18
427517

Inleiding

Voor het vak Compilers & Operating systems hebben we de opdracht gekregen om een eigen taal te ontwikkelen in Java. De ontwikkelde programmeertaal heeft ApollyScript als naam. De syntax van ApollyScript is vergelijkbaar met de syntax van Java. In dit document worden de features van de taal, en de implementatie beschreven.

Inhoudsopgave

Inleiding.....	1
Frameworks/libraries.....	2
Parser	2
Tests	2
Features	3
Variabelen	3
Logische operatoren	3
Wiskundige operatoren	3
If-statement	3
Loops.....	3
Scanners	4
Methoden	4
Klassenstructuur	5
Nodes	5
Scopes	7
Exceptions	7
Tests	8
Good weather tests.....	8
Bad weather tests	8
Resultaat	8
Conclusie	9

Frameworks/libraries

Parser

ApollyScript maakt gebruik van Antlr (ANother Tool for Language Recognition) voor het genereren van een parser. De parser zet input om tot een parse tree. De nodes van de parse tree kunnen vervolgens bezocht worden d.m.v. de gegenereerde visitor van Antlr. De nodes in de parse tree representeren de regels die in de grammatica zijn opgesteld.

Tests

Voor het testen van de taal, is er gebruik gemaakt van JUnit 4. Halverwege het ontwikkelproces is er gedowngraded van JUnit 5 naar JUnit 4. Dit vanwege het feit dat een library nodig was, die JUnit 5 niet ondersteunt. Dit is de System Rules library, wat noodzakelijk is voor het simuleren van scanner input.

Features

De eisen van de opdracht zijn toegepast in de taal. Hieronder bevindt zich een korte beschrijving van alle features in ApollyScript.

Variabelen

ApollyScript ondersteunt de volgende variabelen:

<i>Data type</i>	<i>Voorbeeld</i>
int	int intValue = 20;
boolean	boolean boolVariable = false;
Text	textVariable = "this is a text";
double	doubleVariable = 100.55;

De String variabele van Java is hernoemd naar 'Text', omdat het naar mijn mening duidelijker is.

Logische operatoren

Alle vier de datatypen kunnen logische operatoren gebruiken om waarden met elkaar te vergelijken, mits de typen overeenkomen. Dit zijn de operatoren: '==', '!=', '&&' en '||'.

Wiskundige operatoren

Integers en doubles kunnen gebruik maken van wiskundige operatoren om berekeningen uit te voeren. Dit zijn de operatoren: '+', '-', '*' en '/';

If-statement

If-else statements zijn aanwezig in de taal. De syntax is niet veranderd.

Loops

ApollyScript ondersteunt twee soorten loops: de while loop, en de repeat loop.

De while loop is de loop die we kennen in Java. De syntax hiervan is hetzelfde.

De repeat loop is een loop die je kunt gebruiken om een stukje code een bepaald aantal keren te kunnen herhalen, door simpelweg mee te geven hoe vaak het herhaald moet worden.

```
repeat 10 times {  
  print ("repeat loop");  
}
```

Scanners

De scanner is beschikbaar voor alle vier de datatypes. Zodra een scanner wordt aangeroepen, wordt het object opgeslagen, zodat deze niet een tweede keer aangemaakt hoeft te worden.

<i>Data type</i>	<i>Voorbeeld</i>
int	insertInt()
boolean	insertBoolean()
Text	insertText()
double	insertDouble()

Methoden

De taal ondersteunt methoden. De methoden die aangemaakt worden zijn automatisch statisch. De gebruiker kan ervoor kiezen om één van de datatypes te returnen als waarde. Ook is het mogelijk om niets te returnen.

```
int power (int a, int b) {  
    return a * b;  
}  
  
void printResult (int result) {  
    print(result);  
}  
  
printResult(power(20, 50));
```

Klassenstructuur

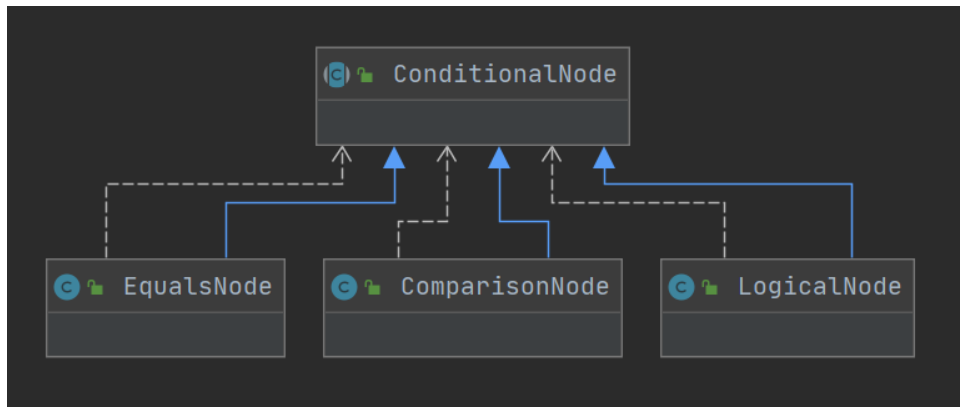
Voor de implementatie van ApollyScript, is object georiënteerd programmeren van uiterst belang. Dit om het toevoegen van features in de toekomst prettiger te maken.

Nodes

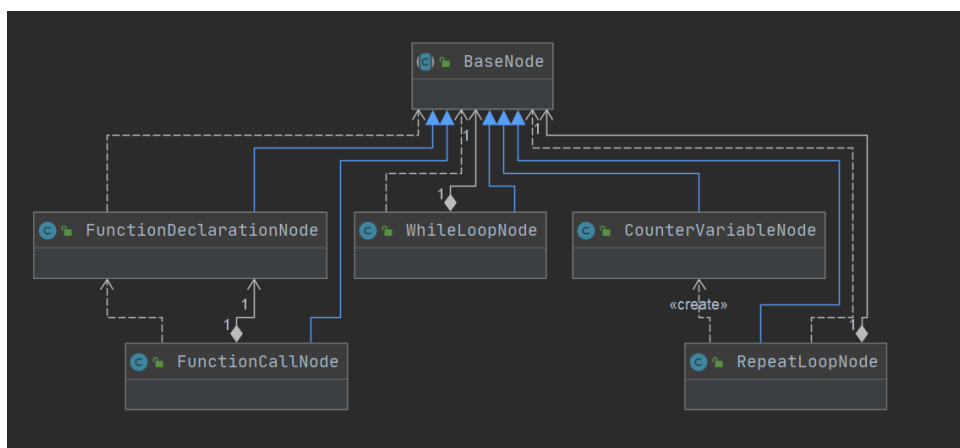
Er is gekozen om voor elke type node een klas aan te maken. Deze klassen erven uiteindelijk allemaal van de BaseNode. De BaseNode is een abstracte class met daarin twee abstracte methoden. Dit zijn 'generateJasminCode' en 'checkType'. De code generatie en de typechecker zijn beide geëncapsuleerd in een object. Meerdere visitor klassen zijn dus niet nodig. Vanwege de hoeveelheid node-klassen worden meerdere klassendiagrammen weergegeven.

De hiërarchie van de verschillende nodes is in onderstaande afbeeldingen te zien.

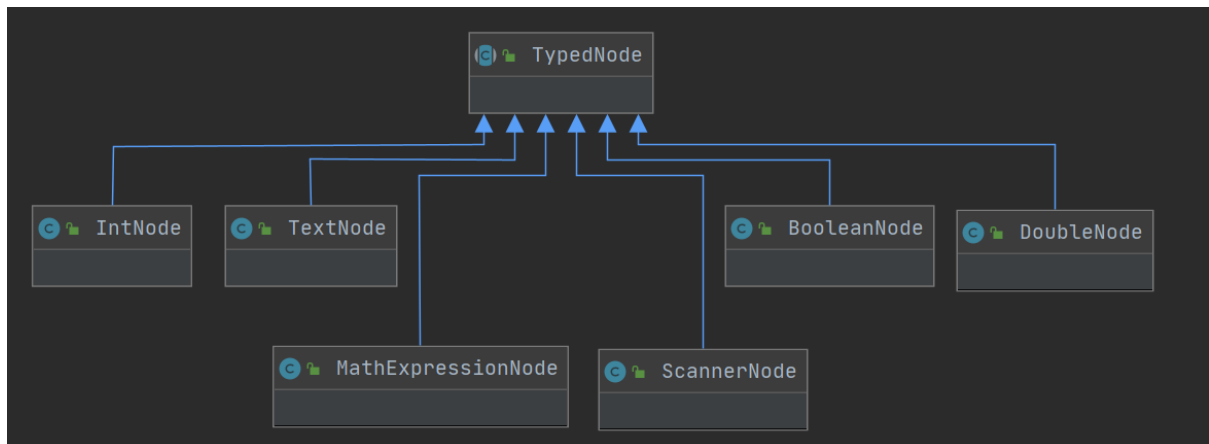
De conditional nodes zijn de vergelijkingsexpressies die een linker- en rechter node bevatten. In de super worden de child nodes gecheckt op type en wordt de bytecode van beide gegenereerd.



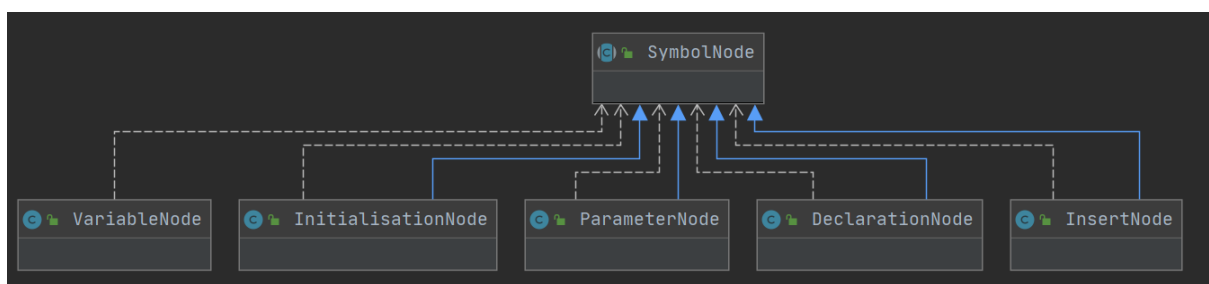
De onderstaande klassen dienen voor de functies en loops



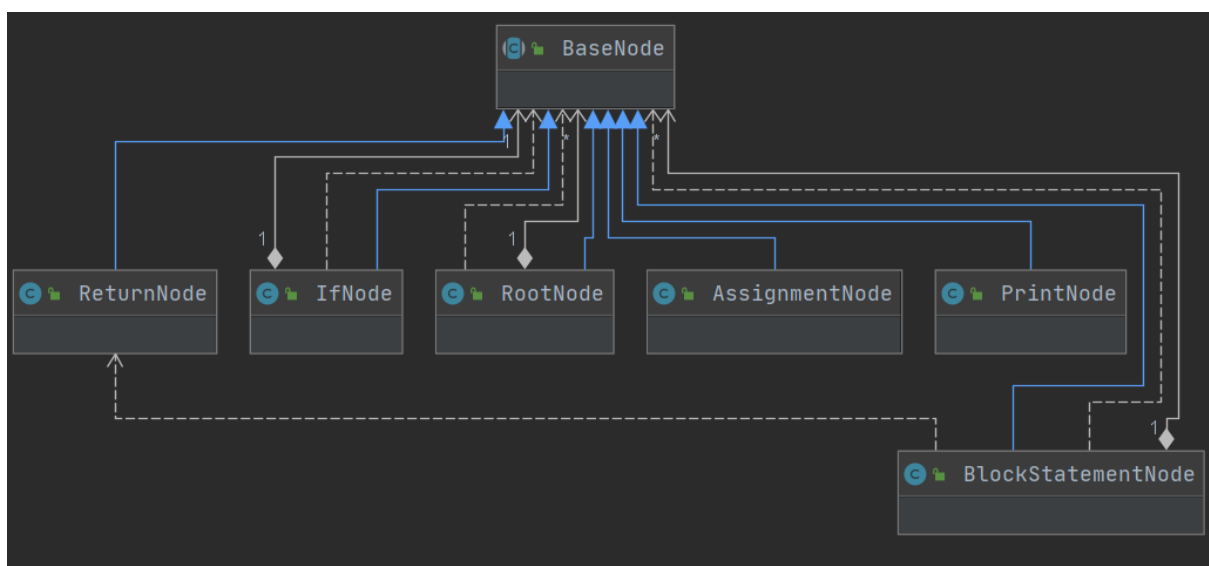
De type nodes zijn de nodes die een abstracte methode bevatten voor het opvragen van de type variabele.



De symbol nodes zijn de nodes die een index hebben. Deze index is nodig om in de bytecode aan te geven welk variabele geladen moet worden. De symbol node erft weer van de TypedNode.



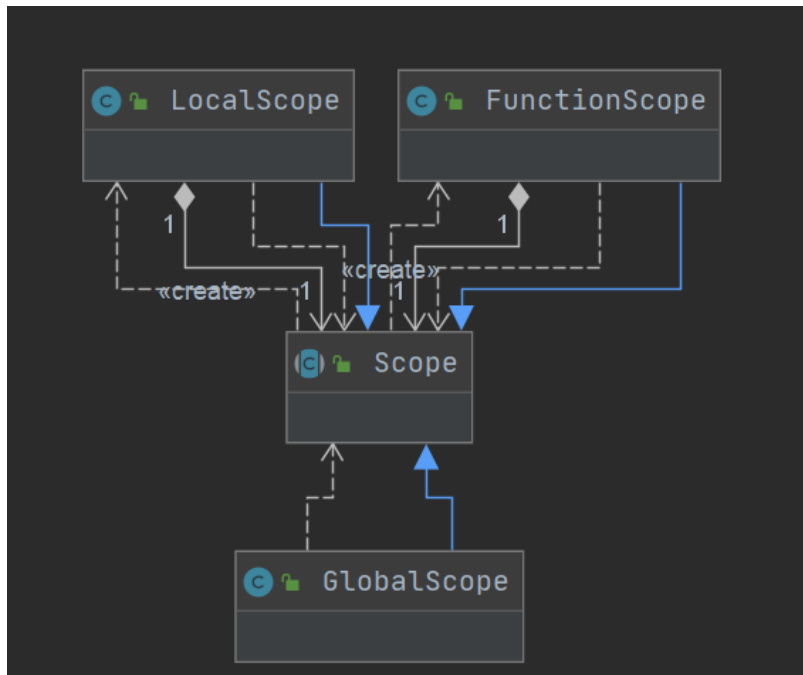
Dit zijn de overige nodes die niet gegroepeerd zijn



Scopes

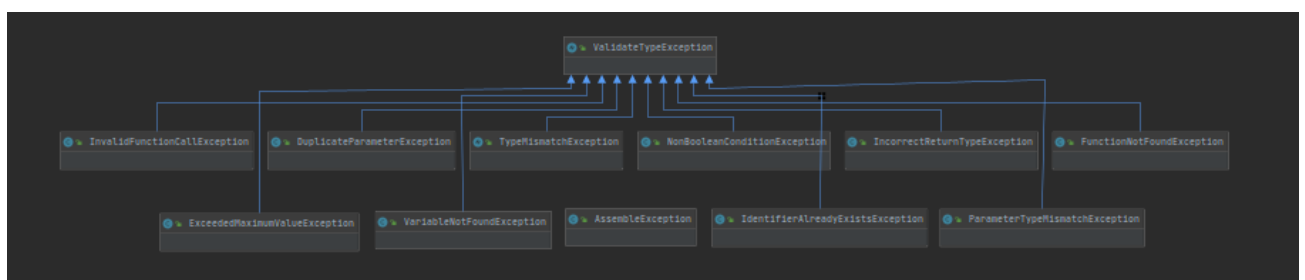
Vanwege het feit dat meerdere typen scopes nodig zijn voor de implementatie van ApollyScript, is er gekozen om voor elke type scope een aparte klasse aan te maken.

De drie verschillende scopes erven van de Scope klasse. Elke scope heeft een andere implementatie voor het opzoeken van een variabele en het opvragen van het aantal opgeslagen variabelen.



Exceptions

Dit zijn de excepties die gegoooid worden bij het detecteren van een fout in de input.



Tests

Om er zeker van te zijn dat de juiste code wordt gegenereerd, zijn er tests geschreven. Dit zijn zowel good- als bad weather tests.

Good weather tests

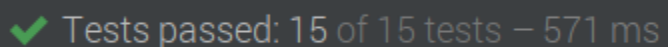
Voor de good weather tests zijn de gegeven voorbeeldprogramma's op Blackboard omgezet naar ApollyScript syntax om vervolgens getest te worden. Naast de vijf gegeven voorbeeldinput, is er één geschreven voor het testen van de double variabele, en als laatste test een compleet voorbeeldprogramma. Een aantal van deze programma's maken gebruik van een scanner. De input wordt toegediend m.b.v. de System Rules library.

Bad weather tests

De bad weather tests simuleren foutieve input van de gebruiker. Bij het detecteren van incorrecte input wordt er een custom exception gegooid met een beschrijving van de fout. De exceptie klassen hebben ieder een korte beschrijving van het fout als naam.

Resultaat

Er zijn 7 good weather tests geschreven, en 8 bad weather tests. Uit het resultaat van de tests is er geconcludeerd dat ApollyScript naar behoren werkt.

A dark grey rectangular banner with a green checkmark icon on the left and the text "Tests passed: 15 of 15 tests – 571 ms" in a light grey font.

✓ Tests passed: 15 of 15 tests – 571 ms

Conclusie

Ik vond dit een interessant vak en heb er veel van geleerd. Ik heb er aardig wat tijd in gestoken en ben vrij tevreden met het resultaat. De requirements zijn naar mijn mening behaald, en de geschreven tests slagen allemaal.