Distinction Task 01: Custom Program

Related Learning Outcomes

ULO1 – Explain the OO Principles

ULO2 – Use OO Language and Library

ULO3 – Design, Develop and Test using an IDE

ULO4 – Communicate using UML Diagrams

I have drawn a high level overview of the main classes in a UML diagram

ULO5 – Describe Elements of Good OO Design

Program Description:

The program will create an intermediate representation (as an Abstract Syntax Tree) for a simple made up programming language. The built tree will be printed to file. If time permits, I may emit C code. The language will allow declaring and assigning strings and integers, calling functions, and simple arithmetic (+, -, *, /)

Sample code:
number : s32 // declare a variable of type signed 32bit int named 'number'
number = 42 // assign a value to variable
foo() // call a function
foo(a, b, c) // call a function with args
number = 2 + 5 // assign expression result to variable
number = foo()

Hello World
message : string
message = "Hello World!"
number : s32
number = 45
print("%s %i", message, number) // prints "Hello World!45" to console

Screenshots:

AST_Node <>
+Parse (Token Stream) <virtual>

Ast Node_Assignment
-lhs : AstNode Identifier
-rhs : AstNode Expression
+Parse (Token Stream)

Ast Node Declaration
-declared_id : AstNode Identifier
-type : AstNode Identifier
+Parse (Token Stream)

AstNode Expression
+Parse (Token Stream)

Ast Node Function Call
-iden : AstNode Identifier
-args : List of AstNodeExp
+Parse (Token Stream)

AstNode Identifier
-id : String
+Parse (Token Stream)

Ast Node Literal Number
-value : Int
+Parse (Token Stream)

Token Stream
-tokens : List of Token
-Lex File (File)
-Peek() -> Token
-Eat() -> Token

Struct
Token
+Value : String
+file : String
+line : Integer
+column : Integer