TypeChecker Project Documentation

FITRIANA PRASARI DEWI/MT2024901

September 28, 2025

1 Overview

This project implements a simple type checker for a toy programming language. It supports:

- Variable declarations (e.g., `int x = 5;`)
- Assignments (e.g., `x = x + 1;`)
- Literals (integers, booleans)
- Binary expressions (arithmetic and comparison)

The type checker ensures programs are type-safe.

2 Project Structure

```
TypeChecker/
          pom.xml
          src
              main
                    java
                        org/example/typchecker
                            Main.java
                            ast/
                                  Program.java
                                  VarDecl.java
                                  Assign.java
                                  BinaryExpr.java
                                  VarRef.java
                                  IntLiteral.java
                                  BoolLiteral.java
                            semantic/
                                TypeChecker.java
                                TypeError.java
              test
                  java
                      org/example/typchecker
                          TypeCheckerTest.java
```

3 Components

3.1 Main.java

Entry point of the project. Runs two cases: success and error.

3.2 AST Classes

- Program: Represents the entire program.
- VarDecl: Variable declaration with type and initializer.
- Assign: Assignment to an existing variable.
- BinaryExpr: Binary operations like `+` and `==`.
- VarRef: Reference to a declared variable.
- IntLiteral: Integer constant.
- BoolLiteral: Boolean constant.

3.3 Semantic Classes

- TypeChecker: Walks the AST and enforces typing rules.
- TypeError: Exception thrown on typing violations.

4 Typing Rules

- VarDecl: Adds variable to environment.
- Assign: Checks declaration and type match.
- BinaryExpr: `ADD` requires ints, `EQ` requires same type.
- VarRef: Checks variable exists.
- Literals: Fixed type (int or bool).

5 Example Programs

5.1 Success Program

```
int x = 1 + 2;
bool b = true;
x = x + 3;
b = (x == 6);
```

Type check result: Pass.

5.2 Error Program

```
int x = 5;
x = true;   // Type error: assigning bool to int
```

Type check result: Error.

5.3 Undeclared Variable Error

```
y = 10;   // y not declared
```

Type check result: Error.

6 Tests

Located in `TypeCheckerTest.java`.

6.1 Test Cases

- testCorrectProgram: ensures well-typed programs pass.
- testTypeErrorAssign: ensures type errors are caught.

Run tests with:

```
mvn clean test
```

7 Usage

7.1 Run the Demo

```
mvn clean compile exec:java -Dexec.mainClass="org.example.typchecker.
   Main"
```

7.2 Run Tests

```
mvn test
```

8 Dependencies

- Java 11+
- JUnit Jupiter 5.10.2
- Maven Surefire Plugin 3.0.0-M9

9 Future Extensions

- Functions and function calls
- Nested scopes
- More types (strings, arrays)
- Type inference
- Richer error reporting

10 Conclusion

This project demonstrates the basics of static type checking: AST construction, type rule enforcement, and error handling.