

# Rapport de Projet

Mini Compilateur Java

**Analyseur Lexical pour le langage JavaScript et Analyseur  
Syntaxique pour l'instruction while**

Réalisé par : Kherbouche Mouloud  
Encadré par : [Mme Nadia Tassoult]

7 décembre 2025

## Grammaire de l'Analyseur Syntaxique

### Grammaire BNF Étendue

```
Program    → S

S          → while ( E ) B

B          → ; | { L }

L          → I L |

I          → ident OP= E ;
           | ident = E ;
           | ident ;
           | ident++ ;
           | ident-- ;
           | ++ident ;
           | --ident ;

E          → E2 (|| E2)*

E2         → E3 (&& E3)*
```

$E3 \rightarrow R ((< | > | <= | >= | == | !=) R)^*$   
 $R \rightarrow T ((+ | -) T)^*$   
 $T \rightarrow F ((* | /) F)^*$   
 $F \rightarrow ( E ) | \text{ident} | \text{nombre}$   
 $OP= \rightarrow = | += | -= | *= | /=$   
 $\text{ident} \rightarrow \text{lettre} (\text{lettre} | \text{chiffre} | \_)^*$   
 $\text{nombre} \rightarrow \text{chiffre}^+ ( . \text{chiffre}^+ )?$

## Explications

- **S** : Instruction **while** complète
- **B** : Bloc (simple ou entre accolades)
- **L** : Liste d'instructions
- **I** : Instructions (affectations, incrémentations)
- **E, E2, E3** : Expressions (logiques, comparaisons)
- **R, T, F** : Expressions arithmétiques avec priorité

## Priorité des Opérateurs

1. Parenthèses ( )
2. \* /
3. + -
4. < > <= >= == !=
5. &&
6. ||

## Méthode d'Analyse

### Descente Récursive

Chaque non-terminal correspond à une méthode Java :

```

public void S() { ... } // while
public void B() { ... } // bloc
public void L() { ... } // liste
public void I() { ... } // instruction
public void E() { ... } // expression
public void E2() { ... } // expression ET
public void E3() { ... } // comparaisons
public void R() { ... } // +/-
public void T() { ... } // *//
public void F() { ... } // facteur

```

## Exemples Acceptés

```
while (x < 10) { x++; }  
while (a + b * 2 < 50) { c += 2; }  
while (i <= 100 && j > 0) { i++; j--; }  
while ((x + y) * z > 10) { total += x; }
```

## Gestion des Erreurs

- Parenthèses manquantes
- Points-virgules oubliés
- Opérateurs invalides
- Expressions mal formées

## Structure du Projet

```
MiniCompilateur/  
  WhileParserSimpleComplet.java  
  test_cases.txt  
  README.md  
  rapport.pdf
```

## Conclusion

L'analyseur syntaxique implémente avec succès une grammaire complète pour l'instruction `while` en Java, avec support des expressions complexes et gestion d'erreurs.