Compiler Documentation

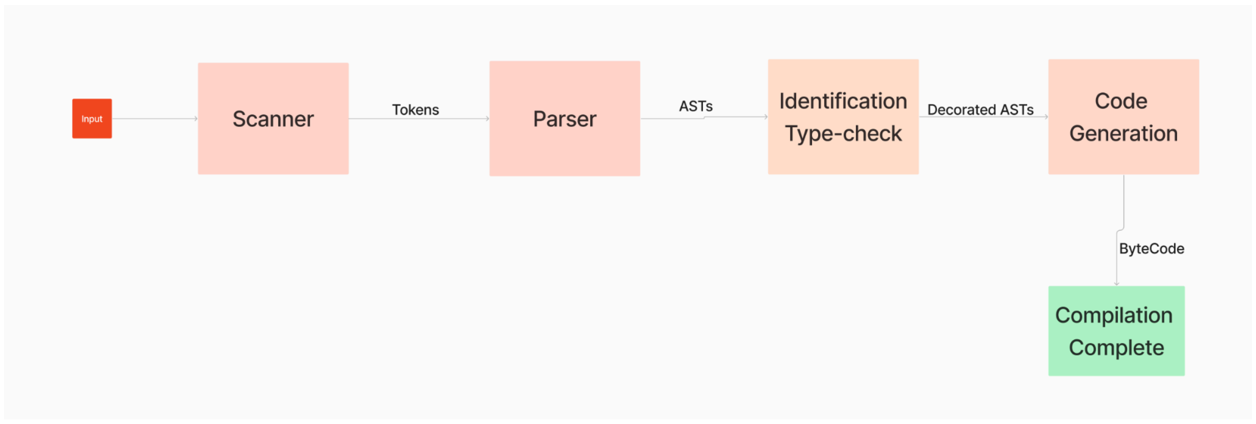

Figure 1: Visual Representation of miniJava Compiler Process

Given an input, the scanner returns tokens which are then parsed by recursive descent to create ASTs (It is assumed the reader is familiar with Abstract Syntax Trees, see ASTChanges.txt for further information). The ASTs undergo an identification traversal where Scoped Identification occurs and then a type-checking traversal is performed. Note, the order of the traversals does not matter, however both identification and type-checking traversals must be performed for contextual analysis. Using the decorated ASTs, code generation occurs that targets x86/x64 processors and the Linux operating system.