

COMPILADORES E INTÉRPRETES

Proyecto
Reglas de Sintaxis de MINIJAVA
Segundo Cuatrimestre de 2025

1 Introducción

Este documento describen las reglas de sintaxis de MINIJAVA. Como se menciona en el documento de especificación del proyecto, MINIJAVA es un lenguaje que puede verse como una simplificación de Java, donde por una parte no se consideran elementos avanzados como Genericidad, Excepciones o Hilos, y por otra parte la estructura de su sintaxis es más estricta. En este documento se presentará la estructura sintáctica del lenguaje mediante su gramática. A partir de las producciones de esta gramática se podrán identificar con mayor facilidad aquellos elementos que alejan a MINIJAVA de Java.

2 Gramática de MiniJava

Cualquier programa MINIJAVA sintácticamente válido debe ser producto de la gramática que se presenta en esta sección. La gramática sigue la notación BNF-extendida, donde:

terminal	es un símbolo terminal
<Clase>	es un símbolo no terminal (además la primer letra es una mayúscula)
ϵ	representa la cadena vacía
<X>::=α	representa una producción, con α una secuencia de terminales y no terminales
<X>::=$\alpha \beta$	es una abreviación de <X>::=α y <X>::=β

Producciones BNF

El no terminal de inicio es **<Inicial>** y las producciones de la Gramática de MINIJAVA son:

```
<Inicial> ::= <ListaClases> eof
<ListaClases> ::= <Clase> <ListaClases> |  $\epsilon$ 
<Clase> ::= <ModificadorOpcional> class idClase <HerenciaOpcional> { <ListaMiembros> }
<ModificadorOpcional> ::= abstract | static | final |  $\epsilon$ 
<HerenciaOpcional> ::= extends idClase |  $\epsilon$ 
<ListaMiembros> ::= <Miembro> <ListaMiembros> |  $\epsilon$ 
<Miembro> ::= <Atributo> | <Metodo> | <Constructor>
<Atributo> ::= <Tipo> idMetVar ;
<Metodo> ::= <ModificadorOpcional> <TipoMetodo> idMetVar <ArgsFormales> <BloqueOpcional>
```

```

<Constructor> ::= public idClase <ArgsFormales> <Bloque>

<TipoMetodo> ::= <Tipo> | void

<Tipo> ::= <TipoPrimitivo> | idClase

<TipoPrimitivo> ::= boolean | char | int

<ArgsFormales> ::= ( <ListaArgsFormalesOpcional> )

<ListaArgsFormalesOpcional> ::= <ListaArgsFormales> |  $\epsilon$ 

<ListaArgsFormales> ::= <ArgFormal>
<ListaArgsFormales> ::= <ListaArgsFormales> , <ArgFormal>

<ArgFormal> ::= <Tipo> idMetVar

<BloqueOpcional> ::= <Bloque> | ;

<Bloque> ::= { <ListaSentencias> }

<ListaSentencias> ::= <Sentencia> <ListaSentencias> |  $\epsilon$ 

<Sentencia> ::= ;
<Sentencia> ::= <Asignacion> ;
<Sentencia> ::= <Llamada> ;
<Sentencia> ::= <VarLocal> ;
<Sentencia> ::= <Return> ;
<Sentencia> ::= <If>
<Sentencia> ::= <While>
<Sentencia> ::= <Bloque>

<Asignacion> ::= <Expresion>

<llamada> ::= <Expresion>

<VarLocal> ::= var idMetVar = <ExpresionCompuesta>

<Return> ::= return <ExpresionOpcional>

<ExpresionOpcional> ::= <Expresion> |  $\epsilon$ 

<If> ::= if ( <Expresion> ) <Sentencia>
<If> ::= if ( <Expresion> ) <Sentencia> else <Sentencia>

<While> ::= while ( <Expresion> ) <Sentencia>

<Expresion> ::= <ExpresionCompuesta> <OperadorAsignacion> <ExpresionCompuesta>
<Expresion> ::= <ExpresionCompuesta>

<OperadorAsignacion> ::= =

<ExpresionCompuesta> ::= <ExpresionCompuesta> <OperadorBinario> <ExpresionBasica>
<ExpresionCompuesta> ::= <ExpresionBasica>

<OperadorBinario> ::= || | && | == | != | < | > | <= | >= | + | - | * | / | %

<ExpresionBasica> ::= <OperadorUnario> <Operando>

```

```

<ExpresionBasica> ::= <Operando>

<OperadorUnario> ::= + | ++ | - | -- | !

<Operando> ::= <Primitivo>
<Operando> ::= <Referencia>

<Primitivo> ::= true | false | intLiteral | charLiteral | null

<Referencia> ::= <Primario>
<Referencia> ::= <Referencia> <VarEncadenada>
<Referencia> ::= <Referencia> <MetodoEncadenado>

<Primario> ::= this
<Primario> ::= stringLiteral
<Primario> ::= <AccesoVar>
<Primario> ::= <LlamadaConstructor>
<Primario> ::= <LlamadaMetodo>
<Primario> ::= <LlamadaMetodoEstatico>
<Primario> ::= <ExpresionParentizada>

<AccesoVar> ::= idMetVar

<LlamadaConstructor> ::= new idClase <ArgsActuales>

<ExpresionParentizada> ::= ( <Expresion> )

<LlamadaMetodo> ::= idMetVar <ArgsActuales>

<LlamadaMetodoEstatico> ::= idClase . idMetVar <ArgsActuales>

<ArgsActuales> ::= ( <ListaExpsOpcional> )

<ListaExpsOpcional> ::= <ListaExps> | ε

<ListaExps> ::= <Expresion>
<ListaExps> ::= <Expresion> , <ListaExps>

<VarEncadenada> ::= . idMetVar

<MetodoEncadenado> ::= . idMetVar <ArgsActuales>

```