Documento de Diseño

Taller 1 Lenguajes y Gramáticas

Autores: Samuel Emperador
Jose Medina
Sebastian Velasquez
David Loreto

Curso Teoria de la Computación

Institución: Potificia Universidad Javeriana

Fecha: 7 de septiembre de 2025

Índice

1 Introducción

EspañolScript es un lenguaje de programación interpretado, de propósito general y de paradigma imperativo, diseñado con sintaxis en español para facilitar el aprendizaje de conceptos de programación. Su objetivo principal es ofrecer una sintaxis clara y cercana al idioma del estudiante, preservando las estructuras esenciales de un lenguaje imperativo para favorecer la comprensión de fundamentos como tipos de datos, expresiones, control de flujo y funciones.

2 Características del Lenguaje

2.1 Tipos de Datos

- **Enteros**: Números enteros sin decimales (ej.: `42`, `-15`, `0`).
- **Decimales**: Números con punto decimal (ej.: `3.14`, `-2.5`, `0.0`).
- **Booleanos**: Valores lógicos `verdadero` y `falso`.
- **Cadenas**: Texto delimitado por comillas dobles (ej.: `"Hola mundo"`).

Nota: No se permiten conversiones automáticas entre tipos.

2.2 Identificadores

- Longitud máxima: 10 caracteres.
- Deben iniciar con letra (a–z, A–Z) o guion bajo (_).
- Pueden contener letras, dígitos (0–9) y guiones bajos.
- Ejemplos válidos: `contador`, `_temp`, `valor1`, `mi_var`.
- Ejemplos inválidos: `1numero`, `variable-larga-nombre`.

2.3 Palabras Reservadas

Categoría	Palabras	Descripción breve
Tipos y constantes	`entero`, `decimal`, `booleano`, `cadena`, `verdadero`, `falso`	Declaración de tipos primitivos y constantes lógicas.
Control de flujo	`si`, `sino`, `para`, `mientras`, `hacer`, `romper`, `continuar`	Estructuras condicionales y de iteración.
Funciones y E/S	`funcion`, `retornar`, `imprimir`, `leer`	Definición y retorno de funciones; entrada/salida por consola.
Lógicos	`y`, `o`, `no`	Operadores lógicos de conjunción, disyunción y negación.

Tabla 1: Palabras reservadas de *EspañolScript* organizadas por categoría.

2.4 Operadores

Categoría	Operadores
Aritméticos	`+` (suma), `-` (resta), `*` (multiplicación), `/` (división), `%` (módulo), `^` (potencia)
Relacionales	`==` (igual), `!=` (diferente), `<` (menor que), `<=` (menor o igual), `>` (mayor que), `>=` (mayor o igual)
Lógicos	`y` (AND), `o` (OR), `no` (NOT)

Tabla 2: Operadores de *EspañolScript* por categoría.

2.5 Estructuras de Control

Condicional

```
si (condicion) {
    // codigo
} sino {
    // codigo alternativo
```

```
}
```

Bucles

```
// Bucle for estilo C
para (inicializacion; condicion; incremento) {
    // codigo del bucle
}

// Bucle while
mientras (condicion) {
    // codigo del bucle
}
```

Funciones

```
funcion nombre_funcion(parametro1, parametro2) {
    // codigo de la funcion
    retornar valor;
}
```

2.6 Entrada y Salida

- `imprimir(expresion)`: Muestra valores en pantalla.
- `leer()`: Lee entrada del usuario (retorna `cadena`).

3 Gramática del Lenguaje

Un programa en EspañolScript consiste en una secuencia de declaraciones y sentencias. Las expresiones respetan la precedencia matemática estándar:

1. Paréntesis `( , )`
2. Potencia `^`
3. Multiplicación, División, Módulo `*, /, %`
4. Suma, Resta `+, -`

5. Relacionales `<`, `<=`, `>`, `>=`, `==`, `!=`
6. Lógicos `no`, `y`, `o`

4 Limitaciones

1. Sin conversión de tipos: no hay *casting* automático entre tipos.
2. Longitud de identificadores: máximo 10 caracteres.
3. Sin estructuras de datos complejas: no hay arreglos, objetos o `structs`.
4. Sin manejo de archivos: sólo entrada/salida por consola.
5. Recursión limitada por memoria disponible.
6. Sin importación de módulos: todo el código reside en un único archivo.
7. Sin manejo de excepciones: sólo errores básicos.

4.1 Manejo de Errores

El intérprete detecta y reporta, al menos, los siguientes tipos de errores:

- **Léxicos**: caracteres no válidos.
- **Sintácticos**: estructura incorrecta.
- **Semánticos básicos**: tipos incompatibles, variables no declaradas.
- **Ejecución**: división por cero, variables no inicializadas.

5 Ejemplos de Código

5.1 Cálculo del Área de un Círculo

```
decimal radio = 5.0;
decimal pi = 3.14159;
decimal area = pi * radio ^ 2;
imprimir("El  rea  del c rculo es: " + area);
```

5.2 Sucesión de Fibonacci (50 términos)

```
entero a = 0;
entero b = 1;
imprimir(a);
imprimir(b);
para (entero i = 2; i < 50; i = i + 1) {
    entero c = a + b;
    imprimir(c);
    a = b;
    b = c;
}
```

6 Arquitectura del Intérprete

1. **Análisis Léxico**: Tokenización (p.ej., con ANTLR).
2. **Análisis Sintáctico**: Generación del AST (p.ej., con ANTLR).
3. **Análisis Semántico**: Verificación de tipos y declaraciones (patrón *Visitor*).
4. **Interpretación**: Ejecución del código a partir del AST.

7 Conclusiones

EspañolScript cumple con el objetivo de servir como un lenguaje sencillo para iniciarse en programación imperativa usando sintaxis en español. El diseño favorece la claridad sobre la complejidad, evitando características avanzadas (módulos, estructuras complejas, excepciones) y concentrándose en tipos básicos, control de flujo y funciones. Esto facilita la comprensión de los conceptos fundamentales y su transferencia posterior a otros lenguajes.