Balabizo Programming Language Documentation

1 Introduction

Balabizo is a modern programming language designed to explore and demonstrate core principles of language development and interpreter design. It features a fully functional interpreter and a clean, expressive syntax, making it an ideal tool for learning and experimenting with programming languages.

Behavior: When code is successfully compiled and executed, the interpreter displays the message `"Life is good"`. If there is an error, the message `"You are being called Balabizo"` along with the error specification is shown.

Contents

2 Features

- **Interpreter Implementation:** Handles syntax parsing, abstract syntax tree (AST) generation, and execution.
- **Expressive Syntax:** Designed to be intuitive and easy to understand.
- **Modular Design:** Codebase is organized into well-defined modules to ensure maintainability and scalability.
- **Version Control:** Managed via GitHub for tracking changes and collaboration.

3 Components

3.1 Interpreter

The core of Balabizo is its interpreter, which performs the following tasks:

- **Lexical Analysis:** Tokenizes the input source code.
- **Parsing:** Converts tokens into an abstract syntax tree (AST).
- **Execution:** Evaluates the AST and executes the code.

4 Installation

4.1 Prerequisites

- **Compiler:** Ensure you have a JDK installed.

4.2 Steps

1. **Clone the Repository:**

```
    git clone https://github.com/yourusername/balabizo.git

```

2. **Navigate to the Project Directory:**

```
    cd balabizo

```

3. **Build the Project:**

```
    mkdir build
    cd build
    cmake ..
    make

```

4. **Run the Interpreter:**

```
    ./balabizo <path-to-your-script.bz>

```

5 Usage

1. **Write a Script:** Create a file with the `.bz` extension. For example, `example.bz`.
2. **Run the Script:**

```
    ./balabizo example.bz

```

3. **Check Output:** The interpreter will output the results of executing your script. Success is indicated by `"Life is good"` and errors by `"You are being called Balabizo"` with error details.

6 Language Syntax

6.1 Variables

Variables are used to store data. They are declared using the `var` keyword.

```
var x = 10;
var name = "Balabizo";
```

6.2 Control Structures

6.2.1 If Statement

The `if` statement is used to execute code conditionally.

```
if (x > 5) {
    print "x is greater than 5";
} else {
    print "x is 5 or less";
}
```

6.2.2 While Loop

The `while` loop executes code as long as a condition is true.

```
while (x > 0) {
    print x;
    x = x - 1;
}
```

6.2.3 For Loop

The `for` loop is used to repeat code a specific number of times.

```
for (var i = 0; i < 10; i = i + 1) {
    print(i);
}
```

6.3 Functions

Functions are used to encapsulate code for reuse.

```
fun greet(name) {
    print "Hello, " + name;
}

greet("World");
```

6.4 Expressions

Expressions are used to compute values.

6.4.1 Arithmetic Operations

Basic arithmetic operations include addition, subtraction, multiplication, and division.

```
var a = 5;
var b = 3;
var sum = a + b;
var product = a * b;
print sum;  // Output: 8
print product;  // Output: 15
```

6.4.2 Logical Operations

Logical operations include `and`, `or`, and `not`.

```
var a = true;
var b = false;
var result = a and b;
print result;  // Output: false
```

6.5 Balabizo full program

```
fun fibonacci(n) {
  var a = 0;
  var b = 1;
  if (n == 0) return a;
  if (n == 1) return b;
  for (var i = 2; i <= n; i = i + 1) {
    var temp = a + b;
    a = b;
    b = temp;
  }
  return b;
}
for (var i = 0; i < 10; i = i + 1) {
  print "fibonacci(" + i + ") = " + fibonacci(i);
}
```

7 Choices of Implementation

The design and implementation of the Balabizo programming language incorporate several key choices:

- **Lexical Analysis:** Tokenizes input code into a stream of tokens for easier parsing.
- **Parsing:** Constructs an abstract syntax tree (AST) from tokens, facilitating code analysis and execution.
- **Interpreter Design:** Executes code directly from the AST, focusing on simplicity and performance.
- **Error Handling:** Provides clear error messages and user-friendly feedback for debugging.
- **Modularity:** Codebase is divided into modules for better organization and maintainability.

8 Development and Contribution

Contributions to the Balabizo project are welcome. To contribute:

- **Fork the Repository:** Create your own fork and make changes.
- **Submit a Pull Request:** Open a pull request with your proposed changes.
- **Open Issues:** Report bugs or request features by opening an issue.

9 Licensing

This project is licensed under the **MIT License**. Please refer to the `LICENSE` file for details.

10 Contact

For questions or further information, please reach out to `abdelrahmanelnagar123@gmail.com` or connect with me on **LinkedIn**.