International Journal of Engineering & Technology, 7 (3.24) (2018) 721-723

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

A Functional Programming Language in Tamil: The Use Of Native Language for Programming

S. Hemavathi[1]*,K.Jayasakthi velmurugan[2]

[1] Faculty, Computer Science and Engineering, Sri Sai Ram Engineering College, Chennai, Tamil Nadu, India
[2]Faculty, Computer Science and Engineering, Jeppiaar Engineering College, Chennai, Tamil Nadu, India
**Corresponding author E-mail: hemavathi.cse@sairam.edu.in*

Abstract

Learning a new programming language has always been a daunting task for the students who feel difficult to understand the program. Most of the programming languages that are used for teaching purpose is present in English language. Learning in mother tongue is enjoyable and thus create technically sound citizens for the development of society which increases the learning curve. The purpose of this work is to create a programming language which uses Tamil characters to create programs with current technology. The goal is to make programming easier for students whose native language is Tamil. The system will provide all the necessary tools for learners to create programs and understanding the concepts which are widespread among the available languages. It includes all the phases of compiler along with the creation of set of keywords. The system can be implemented by creating an interpreter that will run the code and will execute on console like all widely used programming languages. It will implement concepts that are rudimentary basics which are present in other programming languages. User can interact with the interpreter either using a REPL (Read Evaluate Print Loop) or by writing the program in a text-file and run it using the command line argument. The interpreter will be available as an installable package for running the program. Thus, students can learn programming easier in their native language and get exposed to the concepts of programming which can be enhanced for future technologies.

***Keywords**: Programming; language, Tamil; learning, functional programming, learning difficulties, native language;*

1. Introduction

The technology is growing at a very fast pace and newer type of technology are flourishing every day. In order to maintain this growth and advancements more it requires a larger community with students and developers. Learning a programming language helps people to contribute and implement their idea using it. One of the most commonly overlooked barrier is the language to that is used for programming. Almost all of the programming languages that are available today are understandable only to people who knows English. If this impediment is removed then even more people with various background can use technology and accomplish many things.

We are creating a programming language that helps people to program in Tamil language in order to break this language barrier [3]. The language has a functional programming design with less syntactic sugar [1]. The functional programming [6] is a programming paradigm where the functions are treated as first-class members. I.e. a function can be created, passed as an argument, returned as a value. It has closure and anonymous function creation features which can be used.

2. Ease of Use

2.1 Repl

The REPL Read-Evaluate-Print-Loop also known as language shell is a text based interactive platform in which users

enters an input which is the program and upon each entry it will produce some information to the user. As the name suggests the REPL reads the input and sends the input to eval which refers to evaluation. The evaluation is done based on the previous lines of code and resources available at the current entry point. The print is different from ordinary printing to the console as the print part returns the value of the entered code or some warnings and errors if they are present in the entered lines of code. The language shell also supports the POSIX signals in the Linux platform where users can send signals that can stop the current operation, kill the program or send signals to the parent process that invoked the currently executing process.

2.2 Reader and Printer

The reader accepts code written in Tamil characters and can store them in the same format. The printer writes the output to console and also prints the errors and warnings if any of them are present. It supports UTF-8 encoding format which has complete character specification for Tamil characters.

Both the reader and printer require a terminal which supports the UTF-8 format otherwise the user cannot read the characters. The reader also understands the POSIX signals that are sent to the application. It can print the output either in the terminal or to a text file based upon the choice of the user.

3. Existing System

3.1 Ezhil Language

Ezhil, in Tamil language script, is a compact, open source, interpreted, programming language, originally designed to enable native-Tamil speaking students, K-12 age-group to learn computer programming [4], and enable learning numeracy and computing, outside of linguistic expertise in predominately English language-based computer systems. In this programming language, Tamil keywords and language-grammar are chosen to easily enable the native Tamil speaker write programs in the system. It allows easy representation of computer program closer to the Tamil language logical constructs equivalent to the conditional, branch and loop statements in modern English based programming languages. It was the first freely available programming language in the Tamil language and one of many known non-English-based programming languages. The language was officially announced in July 2009, while it has been developed since late 2007.

The syntax of Ezhil is broadly similar to that of BASIC, blocks of code are run in sequential order, or via functions definitions, in a common control flow structures include while, and if. The termination of function block and statement blocks should have the termination keyword, similar to END in BASIC. Declarations are not necessary as it is a dynamic typed language, though type conversions must be made explicitly.

3.2. Swaram Language

Swaram is a full-fledged static-typed programming language, with a feature set resembling C-programming language. It has inspired some of the choice of keywords in the Ezhil language. To its credit Swaram is the first programming language in Tamil, in the true sense with a JIT compiler from source and a virtual machine (VM). In reporting, the authors justify the need for a complete language rather than plain pre-processors, and other syntactic sugar. It is not publicly available, which severely limits language development, system use, community support and improvement. It is strongly typed and allows mixed English & Tamil identifiers.

4. Proposed System

4.1. Tamil Programming Language

The proposed system is an Interpreter which is available for major Platforms as an installable package. Users of this Interpreter doesn’t require English proficiency [3]. Just Tamil knowledge is enough. The programming Language implemented using this Interpreter will have minimum learning curve and it is easy to understand and to do coding with[5].

The created programming Language will be available as an easy installable package. It is also free of cost and is made available with GNU public license, so that anyone who wishes to modify the Language can do so and streamline it to his convenience, wish & will. Since it doesn’t require English knowledge, teachers can find it very easy to teach the same to Tamil Medium Students [2][3]. The Interpreter will be made available as an interactive Programming Language, such that when a user gives an input, it will provide information about how the given input was processed and whether it contains any error or not.

4.2 Advantages of the Proposed System

It has an automatic garbage collector which captures unused ports and non-referenced memory locations. To avoid fragmentation in the memory, it has a built-in vector pool compaction function the re-organizes the stored memory areas such that the free memory space is made available as a whole. It can send POSIX signals to the Kernel in Linux Operating System, so that users can view the information about the process and manage it externally. It has complete support for UTF-8 Character Encoding Scheme that allows users to enter the program in Tamil Characters. It is available free of cost and hence can be used by anyone. The Syntax of this Language is Simple, easy to understand and can be learned quickly [6]. Since it has less Syntactic Sugar, students who are learning this language can easily learn other languages too, without much trouble, as they are not tied up to any language specific features [6].

5. System Design

5.1. Components of the System

Both the source code and the libraries are given by the user to the interpreter for evaluation. Inclusion of libraries are done by the user and any external libraries can also be used in it. The program is both written in a file and executed by the interpreter or can be given through the REPL loop. The code that is entered in the REPL will not be saved whereas the code written in a file can be saved and used later.The lexical analyzer reads the given input and removes any commented lines present in it. It also checks for any errors and separates the lines into tokens which are used later. It treats all the white space as the same and waits till the expression to be completed before reading it. The syntax analyzer checks the syntax such that if there are any unbalanced brackets or not. It also checks whether if any keywords are being used as identifiers which will result in improper execution of the code. Semantic analysis does checks the types of operations made on the data. If any mathematical operation is done between the string and an integer it will throw an error regarding the type mismatch.

The program environment contains the declaration and definition of functions and data structures. It uses the dynamic binding which is more common and easily understood by the learners. Whenever a variable is out of scope it then recycled by the garbage collector. Each of the element stored in the environment stored is referred to as a node. Each node has a name, type and a value. Since the programming language is a functional programming all of the functions are treated the same as the variables. Lambda functions are used to create anonymous functions which can be unnamed and passed as a value or can also be named and used as an ordinary function.

The garbage collector is run automatically whenever the memory is low and calls the memory allocator to add some storage to the interpreter. Since the memory space is dynamically allocated the main memory required to run the interpreter with small programs are relatively very low. It does not wait for the memory to be completely filled before the new allocation. When the low level of free space memory is detected then additional memory is being added to the free space.

5.2 Parser

The parser is the front end of the interpreter. It takes the typed code and converts it into an internal tree structure based upon the given input. The parser is a top down parser that is, all of the text is parsed from the top. A parser is a program that interprets the physical bit stream of an incoming message and creates an internal logical representation of the message in a tree structure. The parser also regenerates a bit stream for an outgoing message from the internal message tree representation. It is called when the bit stream that represents an input message is converted to the internal form that can be handled by the broker; this invocation of the parser is known as parsing. The internal form, a logical tree structure, is described in Logical tree structure. It is described as a tree because messages are typically hierarchical in structure.

5.3 Syntax Analyzer

The syntax analyzer has two main purposes. It checks if the program to be compiled is syntactically correct. It converts the program, which is given as a string of characters, into an abstract syntax tree, which is a representation of the program that is much easier to use for the code generator.

Syntax Analysis or Parsing is the second phase, i.e. after lexical analysis. It checks the syntactical structure of the given input, i.e. whether the given input is in the correct syntax (of the language in which the input has been written) or not. It does so by building a data structure, called a Parse tree or Syntax tree. The parse tree is constructed by using the predefined Grammar of the language and the input string. If the given input string can be produced with the help of the syntax tree (in the derivation process), the input string is found to be in the correct syntax. Headings, or heads, are organizational devices that guide the reader through your paper. There are two types: component heads and text heads.

5.4 Semantic Analyzer

The syntax analyzer has two main purposes. It checks if the program to be compiled is syntactically correct. It converts the program, which is given as a string of characters, into an abstract syntax tree, which is a representation of the program that is much easier to use for the code generator. Syntax Analysis or Parsing is the second phase, i.e. after lexical analysis. It checks the syntactical structure of the given input, i.e. whether the given input is in the correct syntax (of the language in which the input has been written) or not. It does so by building a data structure, called a Parse tree or Syntax tree. The parse tree is constructed by using the predefined Grammar of the language and the input string.

If the given input string can be produced with the help of the syntax tree (in the derivation process), the input string is found to be in the correct syntax. If there are any type mismatch present, If any undeclared variables are use, If the reserved identifiers are misused, If there are multiple declaration of variable in a scope, Accessing an out of scope variable Actual and formal parameter mismatch.

5.5 Evaluator and Garbage Collector

The evaluator is the core of the interpreter. If the given code contains no errors then all of the major work is done by the evaluator. It is first applied to all of the arguments and then after all the nodes are being reduced the function is applied to the main function. It also uses the symbol table to do the evaluation and creation of the functions. Garbage collector runs in a constant space and does not require any additional space for the garbage collection. It does the memory management and also the ports that are used.

6.Conclusion

In this work, programming was taught to students using their native language (Tamil) along with English. Students have expressed positive sentiments about using both Tamil and English for teaching programming. The students in the experimental group have expressed strong positive sentiments than the control group about our intervention. We attribute this increased positivity in sentiments to the usage of the native language within the classroom. Even though this work was done specifically in India, the results of our work are applicable for teaching CS more effectively in many other countries, where English is not the native language.

References

[1] Anabela Gomes and António José Mendes, "Studies and proposals about initial programming learning", 2010 IEEE Frontiers in Education Conference (FIE).

[2] Adalbert Gerald Soosai Raj, Kasama Ketsuriyonk, Jignesh M. Patel and Richard Halverson, "What Do Students Feel about Learning Programming Using Both English and Their Native Language? ", 2017 International Conference on Learning and Teaching in Computing and Engineering (LaTICE).

[3] Yogendra Pal and Sridhar Iyer, "Effect of Medium of Instruction on Programming ability Acquired through Screencast ",2015 International Conference on Learning and Teaching in Computing and Engineering.

[4] Sze Yee Lye and Joyce Hwee Ling Koh, "Computers in Human Behavior ", National Institute of Education, Nanyang Technological University, Singapore, 1 Nanyang Walk, Singapore 637616, Singapore.

[5] Walter Cazzola and Diego Mathias Olivares ,"Gradually Learning Programming Supported by a Growable Programming Language", 2015 IEEE 39th Annual Computer Software and Applications Conference.

[6] Kire Trivodaliev, Biljana Risteska Stojkoska, Marija Mihova, Mile Jovanov and Slobodan Kalajdziski," Teaching computer programming: The macedonian case study of functional programming", 2017 IEEE Global Engineering Education Conference (EDUCON).