Solace Specification Document

Noe Garcia

12 November, 2022

1 Solace Overview

This document contains the specification lists and requirements for the Solace Language. Solace is a statically typed object oriented programming language with a focus on quick construction and easy to read and write syntax. Unlike languages such as Java, not all structures are required to be objects. The following document defines syntax, grammar, and lexical aspects of the langauge. As the definition of the language grows, this document will be updated. Solace is not a pure object oriented programming language like smalltalk. Solace is a small project designed and constructed by one person who is learning more as they are building the language.

2 Types

This section documents the available types within Solace. Types are broken into Atomic, Composite, and Domain-specific.

2.1 Atomic Types

reserved words	description
void	null return type
null	base type representing no data type
int	integer data type, defaults to int32
float	floating point number type, defaults to float 64
char	character type
string	string type
bool	bolean type
:sym	symbolic type

2.2 Composite Types

type name	description
arrays	designating an array object, contains same type collections.
maps	contain key/value pairs of designated types
class	designating a class object definition

2.3 Domain-Specific

There is nothing to note here right now. As Solace if flushed out more information will be added. Solace is meant to be general purpose at this point, so nothin domain specific will be listed.

3 Lexical Rules

Like in many languages, Solace has rules as to what constitutes legal syntax for things such as variable declarations, class naming, and function and method naming.

Solace defines a number of reserved words that are used in defining types, function definitions, class definitions, data structures, and much more. The reserved words chosen for this language are meant to be similar to other languages, such as Python and C. A lot of those reserved words are defined for type declarations shown above, or the legal operators for the language defined below. The following are the reserved words for the language that do not fit in under the other sections, however, are just as crucial to the language definition.

3.1 Reserved Words

Reserved word	description
main	used to declare the main function of the program
class	used to declare a class structure
use	used to include packages/libraries
module	used to define the module the file belongs to
return	keyword used to return value(s) from function

4 Operators

The following showcase some of the legal oporators for Solace

Oporator character(s)	description
+	addition operation
-	subtraction operation
	multiplication operation
/	division operation
%	modulo operation
¿	greater than comparison
¡	less than comparison
¿=	greater than or equal to comparison
¡=	less than or equal to comparison
==	equal to comparison
!=	not equal to comparison

5 Syntax

This section outlines an overview on the basic syntax of the Solace language. The main goal of the syntax is to remain simple but flexible enough to build interesting programs. The following showcase a simple program that defines a factorial function and the main function of the program.

```
module Fibonacci

main: void ()
{
int n = 5;
printc(fibonacci(n));
}

fibonacci: int (int n)
{
if (n <= 1) {
return 1;
}
return fibonacci(n-1) + fibonacci(n-2)
}
```

6 Summary

Solace is a simple object oriented programming language built as a simple project language. This language is meant to be simple to read and write programs with, circumventing the verbose nature of traditional object oriented languages. Solace is written utilizing Yacc, Bison, and C. The goal right now is getting the language up and off the ground by implementing the specifications defined above. From there, the language will be evaluated and this document will be updated on where the language will build from there.