JavaLite

Frances Cao (fc2679)
Mateo Maturana (mm5589)
Hongfei Chen (hc3222)
Ian Chen (yc3936)

February 3, 2021

1 Overview

While Java is one of the most popular programming languages for beginners, it could be challenging for new programmers to get familiar with its excessive syntax and strict object-oriented programming rules. Aiming to design a more beginner-friendly language, we propose JavaLite, a partially object-oriented language derived from Java. JavaLite has simplified syntax compared to Java and incorporates functional programming. Further, JavaLite supports more built-in functionalities for non-primitive data types, including String and Array. These built-in methods are inspired by Python and the goal is to provide a more intuitive way of String/Array manipulation.

For example, compare the following HelloWorld programs in Java and Python.

```
public class HelloWorld {
  public static void main (String[] args) {
    System.out.println("Hello, World!");
  }
}
```

```
print("Hello, World!")
```

JavaLite provides an implementation of Java that is easier to learn.

2 Language Details

2.1 Comments

```
// this is a single line comment
/* multi line comments
can be made like this */
```

2.2 Types

Types

Type	Description
int	32-bit integer (signed)
double	64-bit floating point number (signed)
char	16-bit character/letter
boolean	8-bit true/false value
string	array of chars

2.3 Strings

Strings are initialized to a specific value as follows.

```
string str = "hello";
```

One key difference is that strings in JavaLite are mutable.

2.3.1 .upper()

```
string str = "hello";
str.upper();
// str = "HELLO"
```

2.3.2 .lower()

```
string str = "HeLlO";
str.lower();
// str = "hello"
```

2.3.3 .substring(int a, int b)

```
string str = "hello";
string str1 = str.substring(0, 2);
// str1 = "he"
```

2.3.4 .indexOf(char c)

```
string str = "hello";
int a = str.indexOf('h');
int b = str.indexOf('l');
// a = 0, b = 2
```

2.4 Arrays

Arrays can be initialized as follows.

```
<Type>[] arr = <Type>[size];
```

Arrays can also be created with values as follows.

```
<Type>[] arr = [<Type> val1, <Type> val2, <Type> val3];
```

Elements of an array can be accessed using the [] operation. For example,

```
int[] arr = int[5];
arr[0] = 10;
arr[1] = arr[0];
// arr = [10, 10, 0, 0, 0]
```

2.4.1 .indexOf(T e)

```
int[] arr = [1, 2, 3];
print(arr.indexOf(2)); // 1
```

2.4.2 .pop()

```
int[] arr = [1,1,2];
int a = arr.pop();
// arr = [1, 1], a = 2
```

2.4.3 .append()

```
int[] arr = [1,1];
arr.append(2);
// arr = [1, 1, 2]
```

2.4.4 .length()

```
int[] arr = [1, 1, 1];
int length = arr.length();
// length = 3
```

2.5 Operations

Operations

	int	double	char	boolean	string	Array
$<$	✓	✓	✓			
$>$	✓	✓	✓			
$==$	✓	✓	✓	✓	✓	✓
$<=$	✓	✓	✓			
$>=$	✓	✓	✓			
$!=$	✓	✓	✓	✓	✓	✓
$+$	✓	✓			✓(concat)	✓(concat)
$-$	✓	✓				
$*$	✓	✓				
$/$	✓	✓				
%	✓					
$+=$	✓	✓				
-=	✓	✓				
$++$	✓	✓				
$--$	✓	✓				
[]						✓

2.6 Print

JavaLite uses print to print things to the terminal/console. Print can accept inputs of any type and arrays.

```
print("hi"); // "hi"
print(10); // 10
int[] arr = [1, 2, 3];
print(arr); // [1, 2, 3]
```

2.7 Keywords

int, char, boolean, string, print, for, while, if, else, class, this, return, true, false

3 Example Programs

3.1 Hello World

```
print("Hello, world!");
```

3.2 GCD

```
int gcd(int a, int b) {
  int divisor;
  int dividend;
  if (a > b) {
    dividend = a;
    divisor = b;
  }
  else {
    dividend = b;
    divisor = a;
  }
  while (divisor != 0) {
    int remainder = dividend % divisor;
    dividend = divisor;
    divisor = remainder;
  }
  return dividend;
}
```

3.3 Person Object

```
class Person {
  string name;
  // constructor is assumed to be void
  constructor (string name) {
    this.name = name;
  }
```

```
  void changeName(string newName) {
    this.name = newName;
  }
}

Person me = Person("Adam");
me.changeName("Mark");
print(me.name);
```

3.4 Functional Hello World

```
void sayHello() {
  print("Hello, world!");
}

sayHello();
```