Язык программирования WLang

Оглавление

[1.Начaло 3](#_Toc167360256)

[1.1 Пример простой программы 3](#_Toc167360257)

[1.2 Функция print 4](#_Toc167360258)

[2.Переменные и арифметические выражения 4](#_Toc167360259)

[2.1 Объявление переменной 4](#_Toc167360260)

[2.2 Типы переменных 4](#_Toc167360261)

[2.3 Цикл for 5](#_Toc167360262)

[2.4 Комментарии 5](#_Toc167360263)

[2.5 Конкатенация строк 5](#_Toc167360264)

[3. Функции 6](#_Toc167360265)

[3.1 Определение функции 6](#_Toc167360266)

[3.2 Вызов функции 6](#_Toc167360267)

[3.3 Оператор return 7](#_Toc167360268)

[4.Конструкторы 7](#_Toc167360269)

[4.1 Конструктор без параметров 7](#_Toc167360270)

[4.2 Правила объявления конструктора 7](#_Toc167360271)

[4.3 Конструктор с параметрами 7](#_Toc167360272)

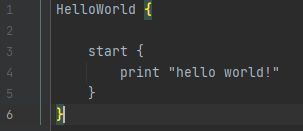
[4.4 Вызов конструктора 8](#_Toc167360273)

[4.5 Конструктор по умолчанию 8](#_Toc167360274)

# 1.Начало

## 1.1 Пример простой программы

WLang-класс, печатающая "hello world!", выглядит так:



В папке out\artifacts\the\_compiler.jar находится собранный проект the\_compiler.jar.

Пример команды для запуска: java -jar the\_compiler.jar D:\SumCalculator.w

По завершении выполнения программы мы получаем скомпилированный класс, который можно запустить по средствам JVM.

Пример команды для запуска скомпилированного класса: java SumCalculator

Если вы все сделали правильно, то программа напечатает

hello world!

Методы в WLang похожи на методы в Java.

Приведенная программа — это класс с методом start. Обычно вы вольны придумывать любые имена для своих методов, но "start" — особое имя: любой класс в WLang начинает свои вычисления с первой команды метода start.

Первая программа на WLang.

|  |  |
| --- | --- |
| HelloWorld { | Название класса, его тело заключается в фигурные скобки |
| start | Определение метода с именем start |
| { | Метод start(как и любая другая) заключаются в фигурные скобки. |
| print “hello world!” | Метод start вызывает библиотечную функцию  print для печати заданной последовательности  символов; |
| }  } | Все блоки кода начинаются и заканчиваются фигурными скобками |

Команды метода заключаются в фигурные скобки {}.

## 1.2 Функция print

Метод start содержит только одну команду

print "hello world!”

Функция вызывается по имени, после которого, перечисляются аргументы. Таким образом,

приведенная выше строка — это вызов функции print с аргументом "hello world!”. Функция

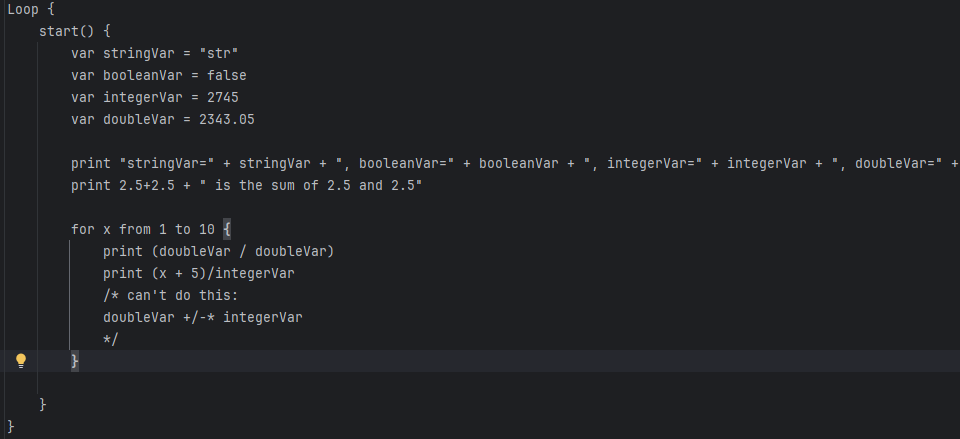
print — это библиотечная функция, которая в данном случае напечатает последовательность символов,

заключенную в двойные кавычки.

Последовательность символов в двойных кавычках, такая как "hello world!”, называется строкой символов, или строковой константой.

# 2.Переменные и арифметические выражения

Приведенная ниже программа четкого смысла не имеет, она лишь показывает, как объявляются переменный и арифметические возможности Wlang.



## 2.1 Объявление переменной

Любая переменная объявляется раньше, чем используется, даже если она является итерационной в цикле for. Для объявления обычной переменной в начале указывается ключевое слово “var”, например:

var stringVar = “str”

## 2.2 Типы переменных

Тип переменной определяется автоматически в зависимости от ее значения. В WLang есть 4 примитивных типа для переменных:

-integer-целочисленный тип, под капотом когда компилятор получает токен NUMBER (строковое представление числа), он подбирает подходящий целочисленный тип, который существует в java, начиная с самого маленького (byte) и в плоть до самого большого (BIGINTEGER).

-double-вещественный тип с плавующей точкой, по аналогии с типом integer подбирается наиболее подходящий, который существует в java.

-boolean- логический тип, имеет всего два значения как и в java(false и true).

-string-строковый тип, обозначается в двойных кавычках (“строка”)

## 2.3 Цикл for

Вся основная арифметика выполняется в цикле for:

for x from 1 to 10 {

Он работает следующим образом, после ключевого слова for пишется название итерационной переменной, после идут начальная граница (from) и конечная граница(to) в которых эта переменная будет итерироваться с шагом 1. Итерационную переменную можно использовать в теле цикла как обычную, но за пределами тела цикла ее существовать не будет.

## 2.4 Комментарии

Две строки в теле цикла for:

/\* can't do this:  
doubleVar +/-\* integerVar  
\*/

Являются комментариями, которые в данном случае, кратко объясняют, что арифметические операции между разными типами (за исключением сложения со строками) переменных в WLang не работают. Все символы, помещенные между /\* и \*/, игнорируются компилятором, и этим можно свободно пользоваться, чтобы сделать программу более понятной.

## 2.5 Конкатенация строк

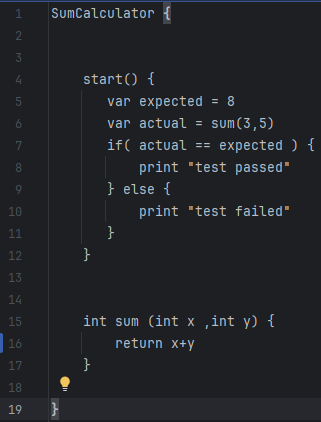
Две строки перед циклом for:

print "stringVar=" + stringVar + ", booleanVar=" + booleanVar + ", integerVar=" + integerVar + ", doubleVar=" + doubleVar  
print 2.5+2.5 + " is the sum of 2.5 and 2.5"

демонстрируют возможность сложения любого типа данных со строкой, результат такой операции всегда строка.

# 3. Функции

Итак, мы имеем функцию sum и главную функцию start, пользующуюся ее услугами, так что вся программа выглядит следующим образом:



## 3.1 Определение функции

Определение любой функции имеет следующий вид:

тип-результата имя-функции (список параметров, если он есть)

{

Тело функции

}

## 3.2 Вызов функции

В 6 строке программы происходит вызов функции sum, главный метод start в ответ получает число и в данном случае сохраняет в переменную actual.

В первой строке определения sum:

int sum (int x ,int y)

указываются типы параметров, имя функции и тип результата. Имена параметров локальны внутри sum, это значит, что они скрыты для любой другой функции, так что остальные подпрограммы могут свободно пользоваться теми же именами для своих целей.

## 3.3 Оператор return

Значение, вычисляемое функцией sum, возвращается в start с помощью инструкции return. За словом return может следовать любое выражение:

return выражение

Функция не обязательно возвращает какое-нибудь значение. Инструкция return без выражения только передает управление в ту программу, которая ее вызвала, не передавая ей никакого результирующего значения. То же самое происходит, если в процессе вычислений мы выходим на конец функции, обозначенный в тексте последней закрывающей фигурной скобкой.

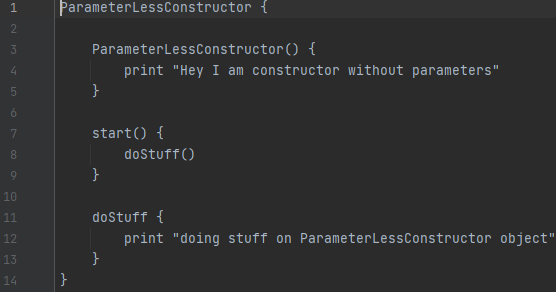
# 4.Конструкторы

В объектно-ориентированной архитектуре Java у классов присутствуют конструкторы для возможности использования функционала одного класса в другом, WLang тоже поддерживает такую возможность

Для начала разберем как объявлять конструкторы, а уже потом взаимодействия между классами.

## 4.1 Конструктор без параметров

Вот пример класса с конструктором без параметров:



## 4.2 Правила объявления конструктора

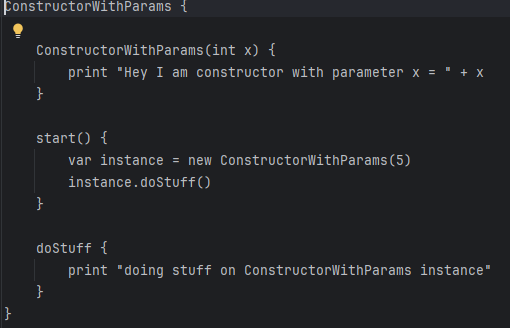
Конструктор отличается от метода несколькими правилами:

-название конструктора совпадает с название класса

-возвращаемый тип всегда null

## 4.3 Конструктор с параметрами

Так как конструктор — это специфичный метод, то в него могут передаваться параметры:



## 4.4 Вызов конструктора

Вызов конструктора в WLang похож на вызов конструктора в Java, сначала пишется ключевое слово “new” затем название конструктора и круглые скобки, куда передаются аргументы.

Конструктор возвращает экземпляр класса, и чтобы им воспользоваться, его можно, например сохранить в переменную, после обратиться к методу этого класса:

var instance = new ConstructorWithParams(5)  
instance.doStuff()

## 4.5 Конструктор по умолчанию

Как и в java если класс не имеет явно прописанного конструктора, то будет создан конструктор по умолчанию без параметров.

Пример класса где будет создан конструктором по умолчанию:

