**Национальный исследовательский университет информационных технологий, механики и оптики**

**Факультет Программной Инженерии и Компьютерной Техники**

**Лабораторная работа №5**

**По дисциплине:**

**«Информатика»**

Работу выполнила:

Студентка группы P3112

Сенина Мария Михайловна

Преподаватель:

Малышева Татьяна Алексеевна

Санкт-Петербург

2020

Оглавление

1. [*1* Squirrel 3](#_Toc55754529)

[1.1 Общая характеристика и история создания 3](#_Toc55754530)

[Введение 3](#_Toc55754531)

[История 3](#_Toc55754532)

[1.2 Система типов и особенности архитектуры языка 3](#_Toc55754533)

[Система типов 3](#_Toc55754534)

[Архитектура Squirrel 4](#_Toc55754535)

[1.3 Примеры кода и интересные факты 5](#_Toc55754536)

[Примеры кода 5](#_Toc55754537)

[Интересные факты 6](#_Toc55754538)

1. [*2* Pure 6](#_Toc55754539)

[2.1 Общая характеристика и история создания 6](#_Toc55754540)

[Введение 6](#_Toc55754541)

[История 7](#_Toc55754542)

[2.2 Система типов и особенности архитектуры языка 7](#_Toc55754543)

[Система типов 7](#_Toc55754544)

[Архитектура Pure 7](#_Toc55754545)

[2.3 Примеры кода и интересные факты 8](#_Toc55754546)

[Примеры кода 8](#_Toc55754547)

[Интересные факты 8](#_Toc55754548)

1. [*3* Elixir 9](#_Toc55754549)

[3.1 Общая характеристика и история создания 9](#_Toc55754550)

[Введение 9](#_Toc55754551)

[История 9](#_Toc55754552)

[3.2 Система типов и особенности архитектуры языка 9](#_Toc55754553)

[Система типов 9](#_Toc55754554)

[Архитектура Elixir 10](#_Toc55754555)

[3.3 Примеры кода и интересные факты 10](#_Toc55754556)

[Примеры кода 10](#_Toc55754557)

[Интересные факты 10](#_Toc55754558)

# **Squirrel**

## Общая характеристика и история создания

### Введение

Squirrel — переводится с английского, как «белка» (см. Рисунок 1 Эмблема Squirrel). Это объектно-ориентированный интерпретируемый язык программирования с динамической типизацией, по синтаксису близкий к языкам C++, C#. Разработан Альберто Демикелисом, для упрощения написания скриптов, обрабатывающих данные в реальном времени, т.е. имеющих ограничения по памяти устройств. Встраиваемый язык, например, для видеоигр или встроенных систем. Официальный сайт языка. [1]

Рисунок 1 Эмблема Squirrel

### История

«Впервые опубликован в 2003 году под лицензией zlib/libpng. Позднее, в ноябре 2010-го, лицензия была изменена на MIT для публикации на Google Code.» [2]. В процессе разработки Демкелис опирался на опыт работы с языком Lua, главной проблемой которого была в реализации сборщика мусора. Его работа сильно влияла на производительность. Решая эту проблему Демкелис реализовал сборщик мусора на основе принципа подсчёта ссылок, который стал основой для нового Lua-подобного языка – Squirrel. Сейчас разработка ядра завершена, а изменения, вносимые автором в язык – не большие и по большей части касаются документации.

## Система типов и особенности архитектуры языка

### Система типов

Squirrel – язык с динамическим определением типов данных. (Список типов данных см. Таблица 1 Типы данных в Squirrel [2])

«Язык с динамическим определением [типов данных](https://ru.wikipedia.org/wiki/%D0%A2%D0%B8%D0%BF_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) — тип [переменной](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B5%D0%BC%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) определяется значением, которое она хранит в данный момент времени и может меняться при присваивании нового значения.» [3]

Таблица 1 Типы данных в Squirrel [2]

|  |  |
| --- | --- |
| ***Название*** | ***Значение*** |
| **null** | Неопределённое значение. Переменные типа Null могут иметь только одно значение — null |
| **bool** | Логический тип, может иметь значение true или false |
| **integer** | Целые числа (32 бита или больше) |
| **float** | Вещественные числа (32 бита или больше) |
| **string** | Текстовые строки — может хранить строки в кодировке ASCII (1 байт) или unicode (2 байта) |
| **table** | Таблица — может хранить набор значений, представленных парами ключ/значение (называемых «слоты») |
| **array** | Массив — последовательность значений переменной длины, индексируемых целыми числами. Индексы начинаются с 0 |
| **function** | Функция — значение типа функция можно «вызвать», передав ему параметры и получив результат выполнения |
| **generator** | Генератор — функция, выполнение которой может быть приостановлено и возобновлено с помощью операторов yield/resume |
| **class** | Как и таблица, классы имеют набор слотов, однако, классы могут быть унаследованы, а на их основе можно создавать объекты. После создания класса к нему можно добавлять слоты в любой момент |
| **instance** | Объект, созданный из класса. Так же содержит слоты, однако добавлять новые слоты в объект нельзя |
| **thread** | Поток выполнения |
| **userdata** | Пользовательские данные — участки памяти и указатели, которые используются приложением, использующим Squirrel |

### Архитектура Squirrel

Автор языка при создании Squirrel вдохновлялся такими языками как Phyton, JavaScript и Lua (им в особенности), поэтому между ними есть много общего.

***Основные особенности языка:***

* Делегация
* Объектно-ориентированный
* Поддерживает функциональное программирование
* Ограниченная лексическая область видимости
* В языке есть генераторы, наподобие генераторов в Phyton
* Поддерживает многопоточность
* Можно реализовать рекурсию
* Присутствует обработка исключений
* Автоматическое управление памятью (есть сборщик мусора)
* Компилирует программы в байт-код, при этом виртуальная машина и компилятор – лёгкие, что позволяет использовать Squirrel, как встраиваемый язык
* Есть JIT компилятор
* Мощный API встраивания

## Примеры кода и интересные факты

### Примеры кода

Поиск факториала числа [2]:

1. function factorial(x)
2. {
3. if (x == 0) {
4. return 1;
5. }
6. else {
7. return x \* factorial(x-1);
8. }
9. }

Таблицы [4]:

1. local myTable = {
2. firstValue: "This is the first value",
3. secondValue: 100
4. }

Классы и наследование [2]:

1. class BaseVector {
2. constructor(...)
3. {
4. if(vargv.len() >= 3) {
5. x = vargv[0];
6. y = vargv[1];
7. z = vargv[2];
8. }
9. }
10. x = 0;
11. y = 0;
12. z = 0;
13. }
15. class Vector3 extends BaseVector {
16. function \_add(other)
17. {
18. if(other instanceof ::Vector3)
19. return ::Vector3(x+other.x,y+other.y,z+other.z);
20. else
21. throw "wrong parameter";
22. }
23. function Print()
24. {
25. ::print(x+","+y+","+z+"\n");
26. }
27. }
29. local v0 = Vector3(1,2,3)
30. local v1 = Vector3(11,12,13)
31. local v2 = v0 + v1;
32. v2.Print();

### Интересные факты

В качестве основного языка его использует пакет MirthKit, позволяющий создавать кроссплатформенные 2D-игры с открытым исходным кодом. На Squirrel написаны такие игры как: «Vice City Multiplayer (мод для Grand Theft Auto: Vice City); Left 4 Dead 2; Portal 2; Alien Swarm; OpenTTD; Counter-Strike: Global Offensive; Shadow Warrior; War Thunder; Mafia 2 Online; GTA IV: Multiplayer; Gothic 2 Online» [2], а среди неигровых приложений Squirrel используют: Code::Blocks и WhatsApp для Windows.

# **Pure**

## Общая характеристика и история создания

### Введение

Pure - функциональный язык программирования с динамической типизацией, основанный на переписывании терминов. Он имеет средства для пользовательского синтаксиса оператора, макросов, арифметики произвольной точности (числа с множественной точностью) и компиляции в собственный код через LLVM.

Рисунок 2 Эмблема Pure

«Pure — это бесплатное программное обеспечение с открытым исходным кодом, распространяемое (в основном) по лицензии GNU Lesser General Public License версии 3 или более поздней.» [5]

Этот язык в основном предназначен математических приложений научных вычислений, но его интерактивная среда интерпретатора, интерфейс C и растущий набор дополнительных модулей позволяют использовать его и в других сферах, например, в искусственном интеллекте, символьных вычислениях и обработке мультимедиа в режиме реального времени.

### История

Язык программирования Pure появился 2008 году, написанный, Альбертом Графом (Германия), как улучшение его предыдущего языка Q. Pure сильно расширяет его функционал. В Pure становятся доступны локальные функции с лексической областью видимости, эффективная поддержка векторов и матриц, а также встроенный интерфейс. Для него существует JIT компилятор, сильно ускоряющий работу скриптов на Pure по сравнению с Q.

## Система типов и особенности архитектуры языка

Pure - язык динамической типизации, но даже для такого класса языков примитивных типов в нём достаточно мало (см. Таблица 2 Примитивные типы Pure). Например, нет логического типа.

### Система типов

Таблица 2 Примитивные типы Pure [6]

|  |  |
| --- | --- |
| ***Название*** | ***Значение*** |
| **int** | Целые числа |
| **bigint** | Целые числа произвольной точности (GMP bigints) |
| **string** | Строки |
| **matrix** | Матрицы |
| **pointer** | Мнимые числа |

### Архитектура Pure

Общая группа языков, к которой можно отнести Pure, состоит из таких языков, как Q, Haskell, Lisp, Алиса и MATLAB.

***Основные особенности Pure*** [5]***:***

* Ссылочная прозрачность
* Понимание списка
* Защита
* Сборка мусора
* Функциональное программирование
* Отложенное вычисление
* Динамическая типизация
* JIT компиляция

## Примеры кода и интересные факты

### Примеры кода

Hello world [6]:

1. > using system;
2. > puts "Hello, world!";

Hello, world!

 Вывод трёх случайных чисел [6]:

1. > using math;
2. > random, random, random;

-795755684,581869302,-404620562

Функции [6]:

1. > square x = x\*x;
2. > square 7;

49

### Интересные факты

«Плагины Pure доступны для электронной таблицы Gnumeric и графического мультимедийного программного обеспечения Miller Puckette Pure Data, которые позволяют расширять эти программы функциями, написанными на языке Pure. Интерфейсы также предоставляются в виде библиотечных модулей для GNU Octave, OpenCV , OpenGL , научной библиотеки GNU , FAUST , SuperCollider и liblo (для Open Sound Control (OSC)).» [5]

# **Elixir**

## Общая характеристика и история создания

### Введение

Elixir – в переводе с английского – «эликсир, снадобье» (см. Рисунок 3 Эмблема Elixir). Это функциональный, распределённый язык программирования общего назначения, работающий на виртуальной машине Erlang (BEAM). Построен поверх Erlang, что обеспечивает распределённость, отказоустойчивость, исполнение в режиме мягкого реального времени, метапрограммирование с макросами и полиморфизмом, реализованным через протоколы. Использует Erlang/OTP для работы с деревьями процессов. Официальный сайт Elixir [7].

Рисунок 3 Эмблема Elixir

### История

Создан в 2012 году Жозе Валимом, целью которого было включить более высокую расширяемость и производительность в Erlang VM, сохраняя совместимость с инструментами и экосистемой Erlang. То есть сейчас это один из основных языков на базе виртуальной машины Erlang.

## Система типов и особенности архитектуры языка

### Система типов

В Elixir строгая типизация и примитивные типы данных Elixir совпадают с типами данных Erlang:

***Примитивные типы данных Elixir*** [8]***:***

* Целые произвольной точности
* Числа с плавающей запятой
* Атомы
* Интервальные типы (англ. ranges)
* Регулярные выражения
* Идентификаторы процессов (PID) и портов
* Ссылки
* Кортежи
* Списки
* Отображения (англ. maps)
* Бинарные данные (англ. binaries)
* Функции

### Архитектура Elixir

Архитектура языка зависима от того, как потом интерпретируется и компилируется код. Поэтому, конечно, наибольшее влияние на Elixir имеет его VM Erlang. Хотя так же язык похож на языки Ruby, Clojure.

***Особенности архитектуры:***

* Компилируемый в байт-код Erlang
* Функциональное программирование
* Строгая типизация
* Метапрограммирование, дающее возможность манипуляций с абстрактным деревом
* Полиморфизм
* Центральный контроль хранения и передачи информации в программах
* Параллельное программирование без разделения ресурсов

## Примеры кода и интересные факты

### Примеры кода

Hello world [8]:

1. IO.puts "Hello World!"

 Модули [8]:

1. defmodule Fun do
2. def fib(0), do: 0
3. def fib(1), do: 1
4. def fib(n), do: fib(n-2) + fib(n-1)
5. end

### Интересные факты

***Интересные проекты на Elixir*** [8]***:***

* «Mix — это инструмент автоматизации сборки, который предоставляет задачи для создания, компиляции и тестирования проектов на Elixir, управления его зависимостями и многое другое.
* Phoenix — это среда веб-разработки, написанная на Elixir, которая реализует шаблон контроллера представления модели (MVC) на стороне сервера.
* Nerves — это платформа, фреймворк и инструментальная среда для создания встроенных систем и устройств.
* Ecto — это оболочка базы данных и генератор запросов для Elixir.

Список использованной литературы

|  |  |
| --- | --- |
| [1] | «Squirrel Programming Language,» [В Интернете]. Available: http://www.squirrel-lang.org/#work. |
| [2] | «Squirrel,» 23 05 2017. [В Интернете]. Available: https://ru.wikipedia.org/wiki/Squirrel. |
| [3] | «Squirrel,» Национальная библиотека им. Н. Э. Баумана, [В Интернете]. Available: https://ru.bmstu.wiki/Squirrel. |
| [4] | «Introduction to Squirrel Programming,» 28 01 2017. [В Интернете]. Available: https://github.com/mickelson/attract/wiki/Introduction-to-Squirrel-Programming#squirrel-tables . |
| [5] | «Чистый (язык программирования) - Pure (programming language),» [В Интернете]. Available: https://ru.qaz.wiki/wiki/Pure\_(programming\_language). |
| [6] | A. Gräf, «Pure Language and Library Documentation,» в *Pure Language and Library Documentation*, 2020. |
| [7] | «Официальный сайт Elixir,» [В Интернете]. Available: https://elixir-lang.org/. |
| [8] | «Elixir (язык программирования),» 30 05 2019. [В Интернете]. Available: https://en.wikipedia.org/wiki/Elixir\_(programming\_language). |

|  |  |
| --- | --- |
| ***Опросник*** |  |
| Кто вы? | Выберите элемент. |
| Во что компилируется код, написанный на Elixir? | Место для ввода текста. Байт-код Erlang |
| Как зовут автора языка Elixir? | Жозе Валим  Жазу Валям  Жоз Лявим |
| Какие особенности архитектуры относятся к Elixir? | ООП  Многопоточность  Динамическая типизация  Функциональное программирование |