

# Знакомство с Java

Часть 1 - About Java

## О себе



- Закончил МФТИ ФБМФ по специальности “Системная биология и биоинформатика”
- Знаю больше 10 языков программирования
- Основные языки: Python, Swift, Java, C++
- Работаю в Цосивт в стартапах
- Основной род деятельности – разработка информационных систем

# Немного истории

---

Java is C++  
without guns and  
knives



# Java

- Дата выпуска 1995 г.
- Первое название Oak
- Задумывался как язык для разнообразных бытовых устройств
- Разработана компанией SUN. На данный момент принадлежит компании Oracle
- Названа в честь марки кофе



2011 - выпуск Kotlin

2008 - первая версия Android

2003 - первая версия Scala

1991 - появление Python

1991 - релиз Linux

1972 - был создан язык C

2009 - был представлен GO

2007 - Iphone

**1995 - первая версия Java**

1989 - появление C++

1985 - выпуск Microsoft Windows 1.0



**Patrick  
Naughton**

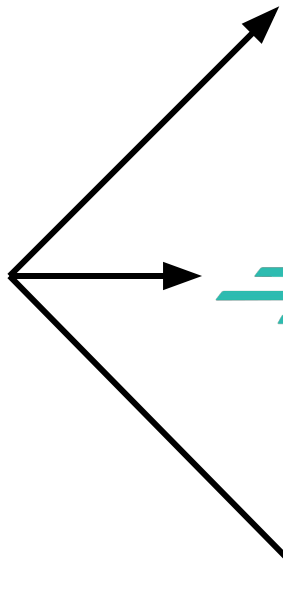


**James  
Gosling**



# Где используется?

1. Серверные приложения
2. Backend веб-приложений
3. Android разработка
4. Встраиваемые системы
5. Трейдинговые и банковские системы
6. Электронные информационные системы государственного уровня
7. Десктоп приложения



# Особенности Java

1. Простота реализация ООП
2. Надежность и безопасность
3. Независимость от платформы и переносимость кода
4. Высокая производительность
5. Интерпретируемость, поточность, динамичность
6. Встроенный механизм поддержки многопоточности
7. Большое количество готовых пакетов
8. Огромное сообщество пользователей

# Особенности Java

---

# ООП подход

- Java – объектно ориентированный язык
- Основная причина создание Java вместо использования популярного на тот момент C++ – неудобство написания классов в C++
- Идея была создать язык, на котором можно было написать классы легкие понятные и читаемые

```
public class HelloWorld{  
    public static void main(String[] args) {  
        ...  
    }  
}
```

# Надежный

- Java крайне типизированный язык
- Каждая мелочь должна быть объявлена заранее с явным указанием типа
- Чем более язык типизирован тем более он надежен. Чем больше правил в написании кода, тем больше ошибок runtime переходят в ошибки компиляции

# Независимость от платформы

- Изначально Java задумывался как язык программирования для бытовой техники
- Код java файлов транслируется в специальный bytecode, который понимают только специальные Java-машины (JVM)
- Для каждой системы существует своя машина для работы программы, что делает возможным запускать код под любой системой
- JVM живет в оперативной памяти компьютера и работает, как software, который обеспечивает нормальную работу java программ
- Таким образом java не только язык для написания программ, а целая платформа

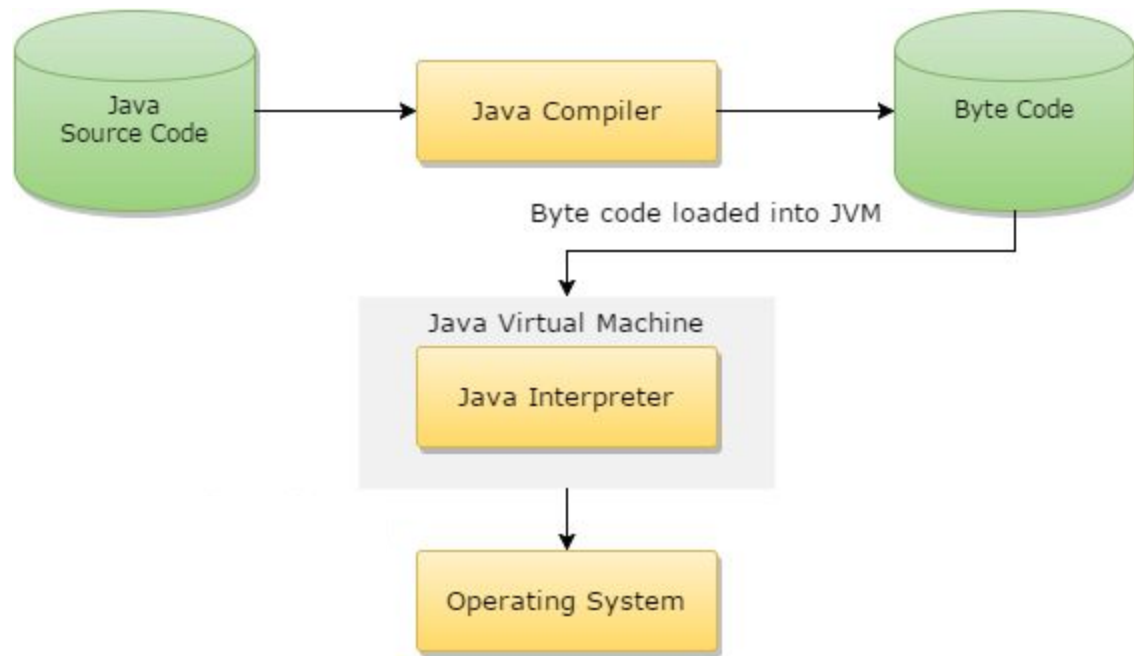


Diagram of JVM



# JVM

Java virtual machine – основная составляющая исполняющей системы java. Виртуальная машина исполняет bytecode, который до этого был создан компилятором (например javac) из исходного текстового файла java с расширением .java

На разных системах можно использовать один и тот же bytecode из одного и того же исходного файла

Compile once,  
run anywhere

# TIOBE Index for October 2018

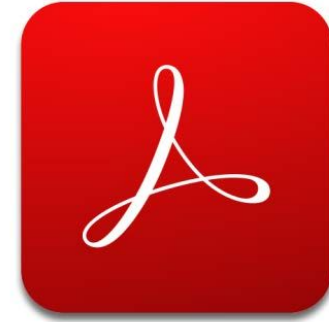
Oct 2018	Oct 2017	Change	Programming Language	Ratings	Change
1	1		Java	17.801%	+5.37%
2	2		C	15.376%	+7.00%
3	3		C++	7.593%	+2.59%
4	5	⬆	Python	7.156%	+3.35%
5	8	⬆	Visual Basic .NET	5.884%	+3.15%
6	4	⬇	C#	3.485%	-0.37%
7	7		PHP	2.794%	+0.00%
8	6	⬇	JavaScript	2.280%	-0.73%
9	-	⬆	SQL	2.038%	+2.04%
10	16	⬆	Swift	1.500%	-0.17%

# Масштабируемость

- Если хочется сделать хороший продукт – используйте PHP
- Если хочется сделать современный продукт с большим количеством функционала с легкостью внедрения новых фич – используйте Python
- Если хочется сделать продукт, который будет стабильно работать при любой нагрузке на веб-сервер – используйте Java



ebay



amazon

ORACLE®  
DATABASE



# Установка Java

---

# Разные платформы Java

- Java SE – Java Standard edition включает в себя все необходимое для разработки приложений на java
- Java EE – Java Enterprise Edition включает в себя больше чем обычная версия. Была создана для разработки на уровне предприятий и больших компаний
- Java ME – Java Micro Edition включает в себя урезанную версию java для разработки на мобильных устройствах
- Java Card – для смарт карт и устройствах крайне ограниченных в ресурсах

[Java SE](#)[Java EE](#)[Java ME](#)[Java SE Subscription](#)[Java Embedded](#)[Java Card](#)[Java TV](#)[Community](#)[Java Magazine](#)[Overview](#)[Downloads](#)[Documentation](#)[Community](#)[Technologies](#)[Training](#)

## Java SE Downloads



Java Platform (JDK) 11

### Java Platform, Standard Edition

#### Java SE 11.0.1(LTS)

Java SE 11.0.1 is the latest release for the Java SE 11 Platforms

[Learn more](#) ▶

- [Installation Instructions](#)
- [Release Notes](#)
- [Oracle JDK License](#)
- [Java SE Licensing Information User Manual](#)
  - Includes Third Party Licenses
- [Certified System Configurations](#)
- [Readme](#)

#### Oracle JDK

[DOWNLOAD](#) ↓

### Java SDKs and Tools

- ↓ [Java SE](#)
- ↓ [Java EE and Glassfish](#)
- ↓ [Java ME](#)
- ↓ [Java Card](#)
- ↓ [NetBeans IDE](#)
- ↓ [Java Mission Control](#)

### Java Resources

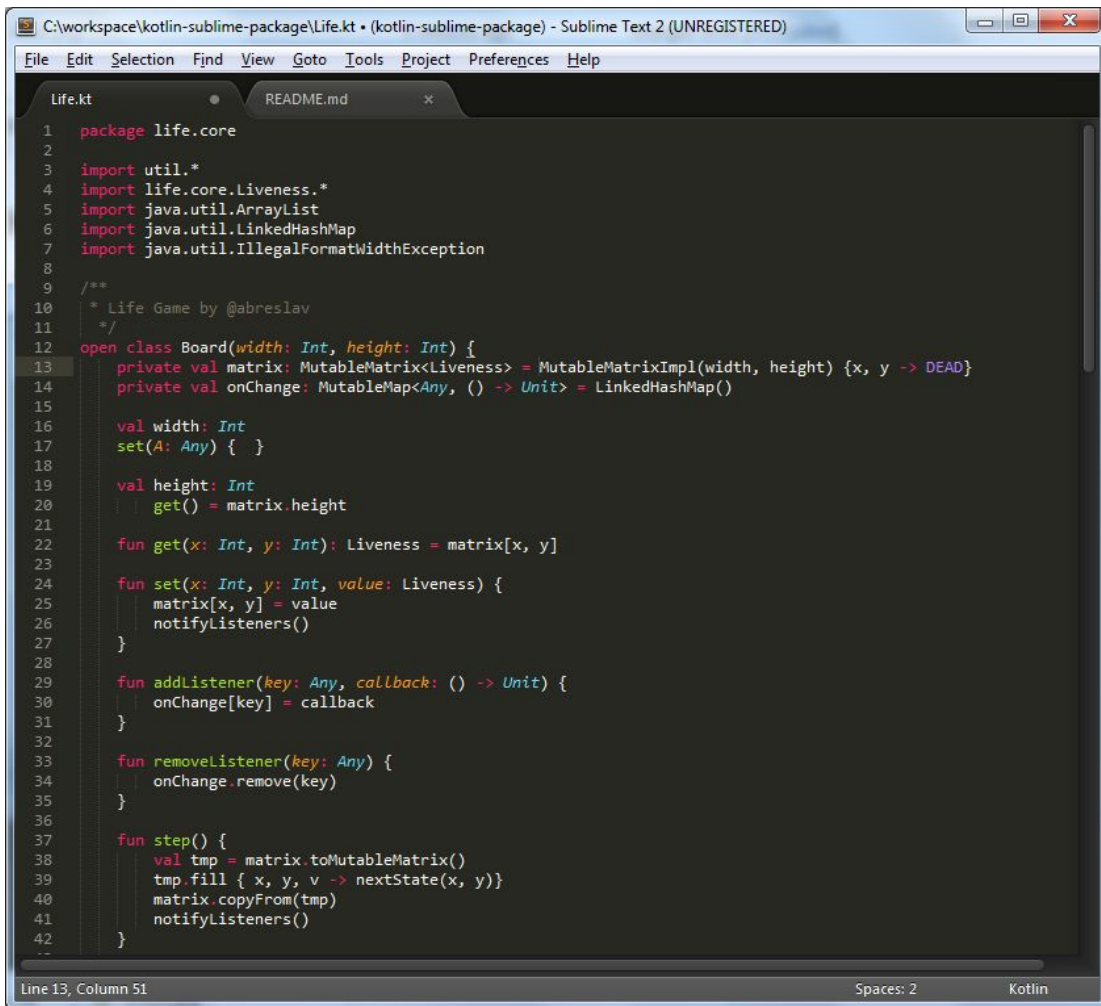
- ↓ [Java APIs](#)
- ↓ [Technical Articles](#)
- ↓ [Demos and Videos](#)
- ↓ [Forums](#)
- ↓ [Java Magazine](#)
- ↓ [Developer Training](#)
- ↓ [Tutorials](#)
- ↓ [Java.com](#)



# Sublime Text 3

<https://www.sublimetext.com/>

3



```
1 package life.core
2
3 import util.*
4 import life.core.Liveness.*
5 import java.util.ArrayList
6 import java.util.LinkedHashMap
7 import java.util.IllegalFormatWidthException
8
9 /**
10  * Life Game by @abreslav
11  */
12 open class Board(width: Int, height: Int) {
13     private val matrix: MutableMatrix<Liveness> = MutableMatrixImpl(width, height) {x, y -> DEAD}
14     private val onChange: MutableMap<Any, () -> Unit> = LinkedHashMap()
15
16     val width: Int
17     set(A: Any) { }
18
19     val height: Int
20     get() = matrix.height
21
22     fun get(x: Int, y: Int): Liveness = matrix[x, y]
23
24     fun set(x: Int, y: Int, value: Liveness) {
25         matrix[x, y] = value
26         notifyListeners()
27     }
28
29     fun addListener(key: Any, callback: () -> Unit) {
30         onChange[key] = callback
31     }
32
33     fun removeListener(key: Any) {
34         onChange.remove(key)
35     }
36
37     fun step() {
38         val tmp = matrix.toMutableMatrix()
39         tmp.fill { x, y, v -> nextState(x, y)}
40         matrix.copyFrom(tmp)
41         notifyListeners()
42     }
43 }
```

Line 13, Column 51

Spaces: 2

Kotlin