JAVA

ПЛАТФОРМЕНО-НЕЗАВИСИМИ ПРОГРАМНИ ЕЗИЦИ

Лекция 1. Обектно-ориентирано програмиране

доц. д-р инж. Румен П. Миронов

Въведение

Основни проблеми

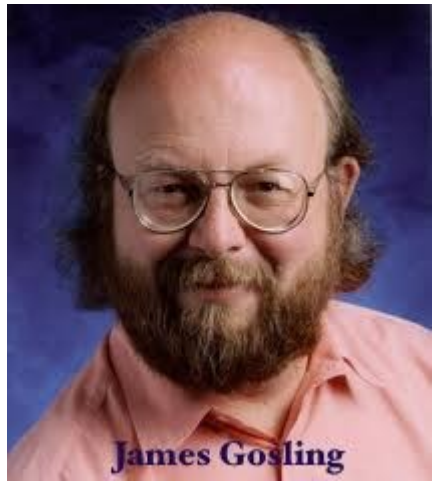
- Методи и подходи за решаване на проблемите
- Организация на структурите от данни
- Разработване на алгоритми за отделните задачи
- Разработване на програми
  - Избор на езици за програмиране
  - Избор на среда за програмиране
- Изпълнение на програмите

Въведение

Развитие на езиците за програмиране

- Машинен код и асемблери
- Процедурно програмиране
- Обектно-ориентирано програмиране
- Компонентно програмиране
- Аспектно-ориентирано програмиране
- Script програмиране
- Мета-програмиране

1. Исторически сведения за езика JAVA



Програмният език Java е разработен от James Gosling, който още се нарича бащата на Java, през 1995 год..

- James Gosling, Mike Sheridan и Patrick Naughton стартират Java Language Project през June 1991. Тимът от инженери от фирмата SUN Microsystems се нарича Green Team;
- първоначално е разработен за малки вградени (embedded systems) системи в електронни устройства като Set-Top Boxes;
- по това време се нарича „Greentalk" от James Gosling и разширението на файловете му е .gt;
- след това го наричат „Oak“ и го разработват като част от Green Project;

Дъбът е символ на сила и е избран за национално дърво в много страни като САЩ, Франция, Германия, Румъния и др.

- през 1995 г „Oak“ е преименуван на "Java", тъй като вече е търговска марка на фирмата Oak Technologies, а името е изключително уникално и се предпочита от членовете на екипа;

Java е остров в Индонезия, където е произведено първото кафе (наречено Java кафе). Това е вид еспресо зърна. Името Java е избрано от Джеймс Гослинг, докато пие чаша кафе в близост до офиса му

1. Исторически сведения за езика JAVA

Основните принципи при създаването на езика за програмиране Java са:

- Simple – прост за разбиране
- Robust – здрав, устойчив
- Portable - преносим
- Platform-independent – платформено независим
- Secured – работи в защитена среда
- High Performance – високо производителен
- Multithreaded - многонишков
- Architecture Neutral – независим по отношение на архитектурата
- Object-Oriented – обектно-ориентиран
- Interpreted - интерпретируем
- Dynamic - динамичен

1. Исторически сведения за езика JAVA

Основните версии на езика за програмиране Java са:

JDK Alpha and Beta (1995)
JDK 1.0 (23rd Jan 1996)
JDK 1.1 (19th Feb 1997)
J2SE 1.2 (8th Dec 1998)
J2SE 1.3 (8th May 2000)
J2SE 1.4 (6th Feb 2002)
J2SE 5.0 (30th Sep 2004)
Java SE 6 (11th Dec 2006)
Java SE 7 (28th July 2011)
Java SE 8 (18th Mar 2014)
Java SE 9 (21st Sep 2017)
Java SE 10 (20th Mar 2018)
Java SE 11 (September 2018)
Java SE 12 (March 2019)
Java SE 13 (September 2019)
Java SE 14 (Mar 2020)
Java SE 15 (September 2020)
Java SE 16 (Mar 2021)
Java SE 17 (September 2021)
Java SE 18 (to be released by March 2022)

Основните съкращения на базовите версии са:

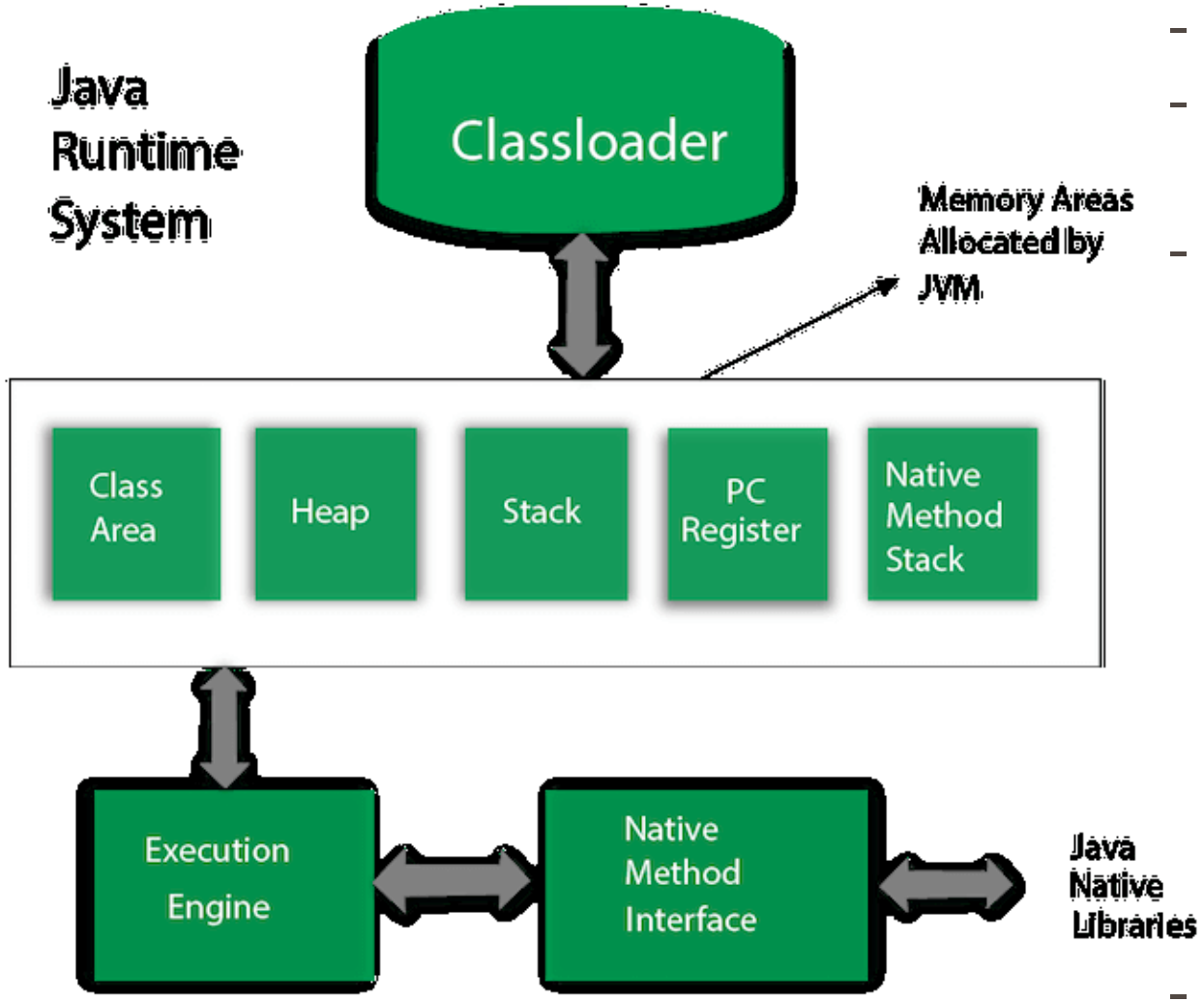
J2SE (Java 2 Platform, Standard Edition), J2EE (Java 2 Platform, Enterprise Edition), J2ME(Java 2 Platform, Micro Edition)

2. Основни компоненти на JAVA

Основните компоненти на Java включват:

- **Java Virtual Machine** (JVM) - изпълнява следните основни задачи:
  - Loads code - зарежда кода на програмата;
  - Verifies code - проверява кода на програмата;
  - Executes code - изпълнява кода на програмата;
  - Provides runtime environment – осигурява среда за изпълнение.
- **Java Runtime Environment** (JRE) е набор от софтуерни инструменти, които се използват за разработване на Java приложения. Използва се за осигуряване на среда за изпълнение. Това е физическото внедряване на JVM. Той съдържа набор от библиотеки + други файлове, които JVM използва по време на изпълнение.
- **Java Development Kit** (JDK) е среда за разработка на софтуер, която се използва за разработване на Java приложения и аплети. Той съдържа JRE + инструменти за разработка.

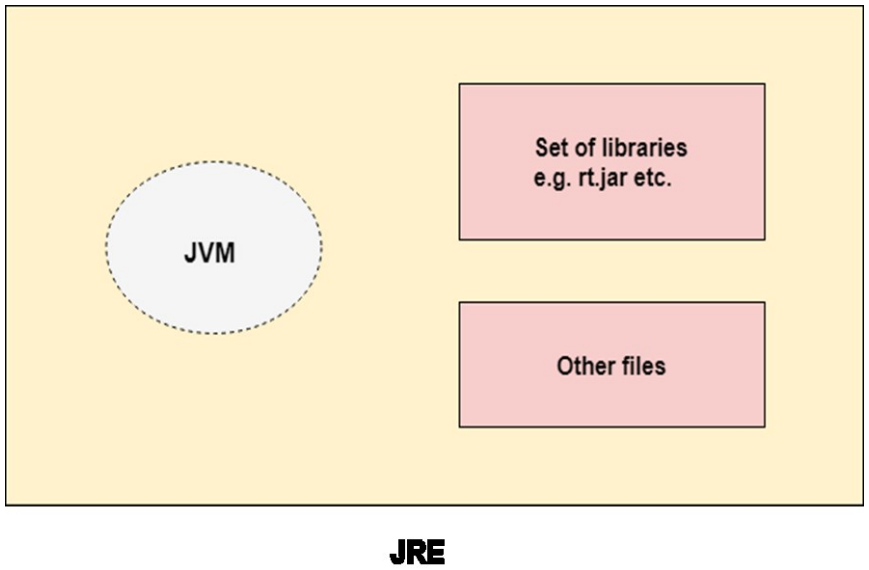
2. Основни компоненти на JAVA

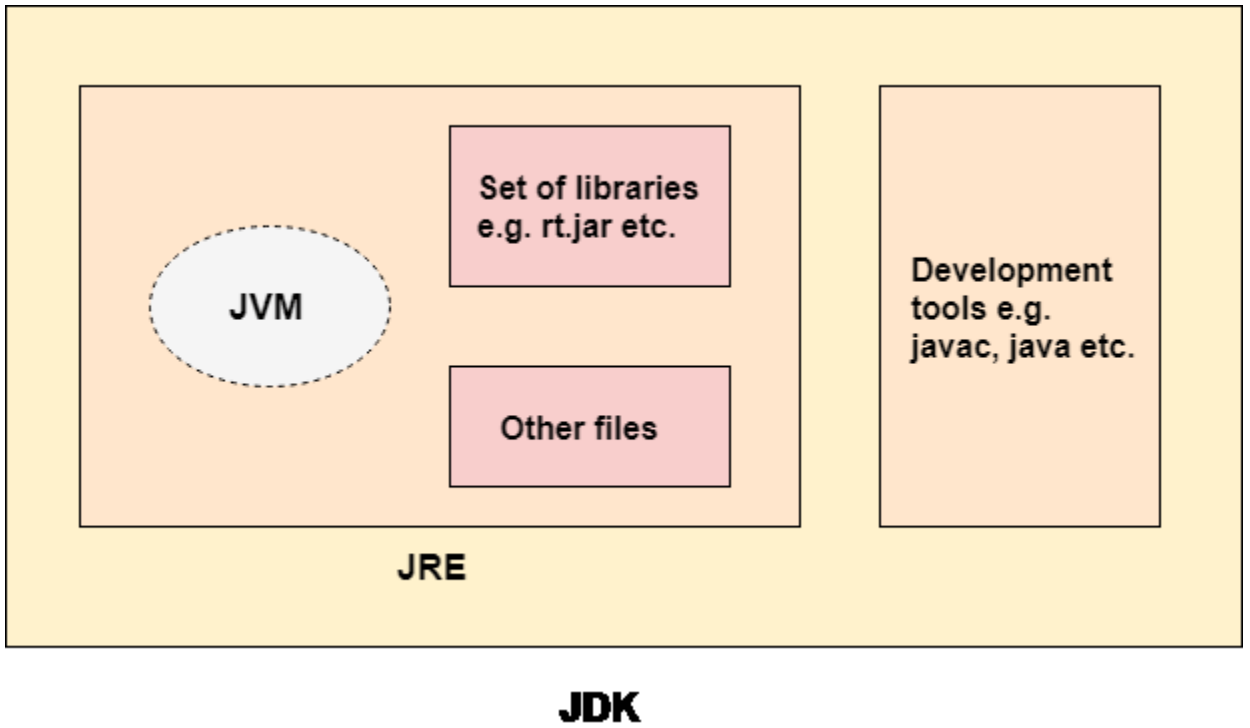


Execution Engine съдържа:

- virtual processor
- Interpreter – прочита bytecode потока и изпълнява инструкциите.
- Just-In-Time (JIT) compiler - Използва се за подобряване на производителността. JIT компилира части от байт кода, които имат подобна функционалност едновременно и по този начин намалява времето, необходимо за компилиране. Терминът "компилатор" се отнася до транслатор от набора от инструкции на виртуалната машина на Java (JVM) към набора от инструкции на конкретен процесор.
- Java Native Interface (JNI) - осигурява връзката с приложения на други езици.

2. Основни компоненти на JAVA





JDK съдържа частна виртуална машина на Java (JVM) и няколко други ресурси като: интерпретатор/зареждащ инструмент (java), компилатор (javac), архиватор (jar), генератор на документация (javadoc) и др., за да се завърши разработката на Java приложенията.

3. Компилиране на кода на програмата на JAVA

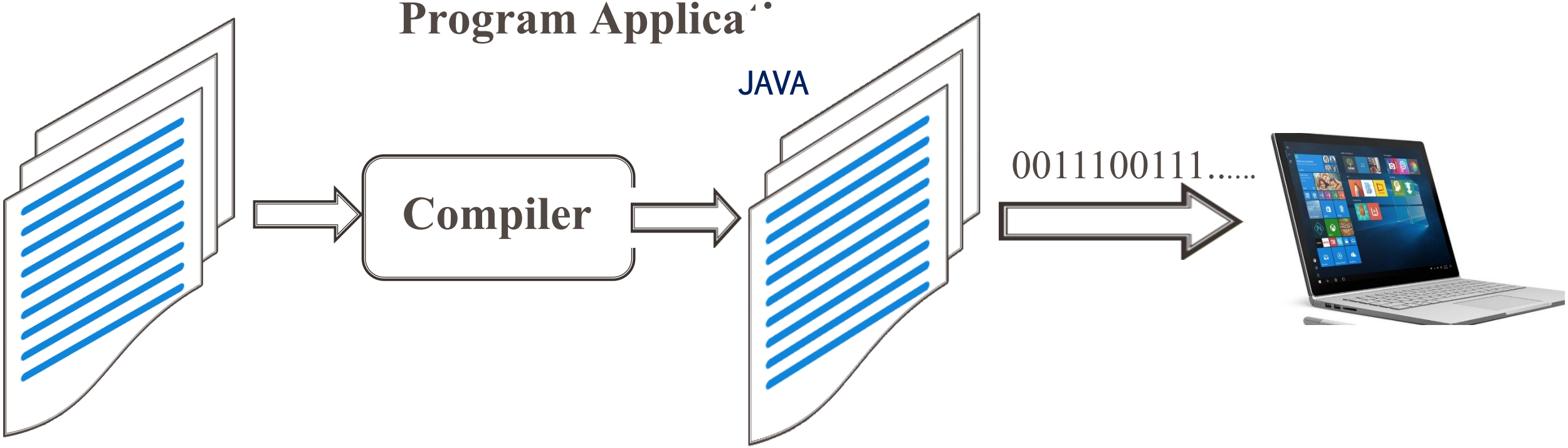
Java source code (.java)

Compilation

Java bytecode (.class)

Java Virtual Machine

Program Applica‘‘



Simple.java

Simple.class

Simple program

Обектно Ориентирано Програмиране

Object-Oriented Programming е методология или парадигма за изграждане на програми чрез използване на класове и обекти. То опростява разработката и поддръжката на софтуера, като осигурява някои основни концепции:

- Abstraction
  - Object
  - Class
- Encapsulation
- Inheritance
- Polymorphism

Обектно Ориентирано Програмиране

Каква е разликата между обектно-ориентиран език за програмиране и обектно-базиран език за програмиране?

Обектно базираният език за програмиране притежава всички характеристики на ООП с изключение на наследяването. JavaScript и VBScript са примери за обектно-базирани езици за програмиране.

Предимства на ООП пред процедурно-ориентираните езици за програмиране:

- OOPs улесняват разработката и поддръжката, докато в програмния език, ориентиран към процедурите, не е лесно да се управлява, ако кодът расте с увеличаване на размера на проекта.
- OOPs осигуряват скриване на данните, докато в процедурно-ориентиран език за програмиране глобалните данни могат да бъдат достъпни от всяко място.

3. Структура на програмата на JAVA

- Клас
- Членове на класа
  - член-променливи (свойства, атрибути)
  - член-функции (методи на класа)
- Капсулиране на данните (Encapsulation)
  - Видимост
    - пълна квалификация – package/namespace, class/type, instance/reference
    - степени на видимост – public, private, protected
- Наследяване (Inheritance)
- Полиморфизъм
- Обект (инстанция на класа)

3. Структура на програмата на JAVA

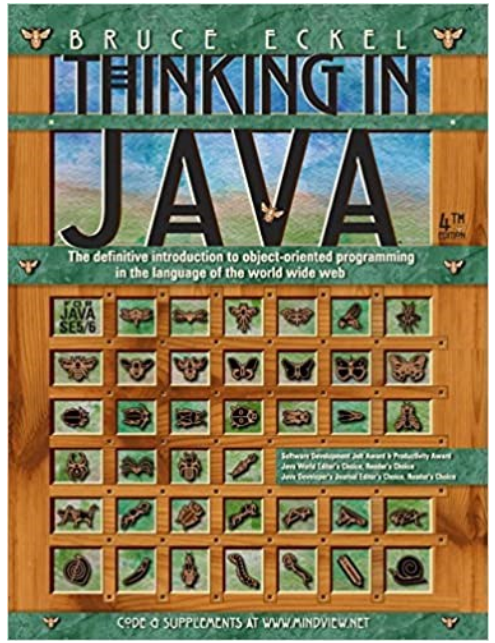
Simple.java

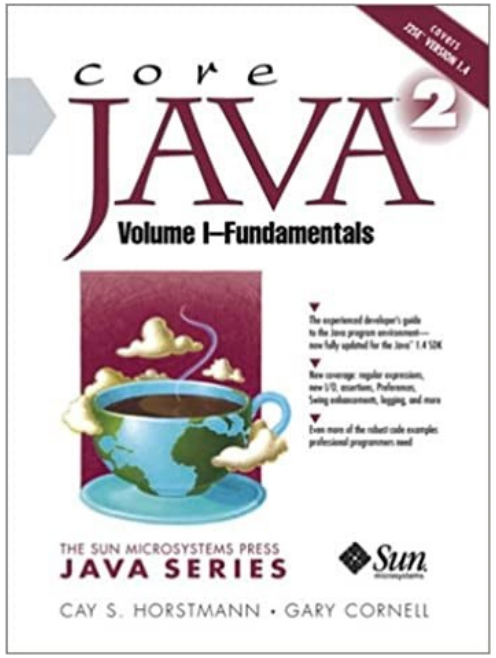
```
1. public class Simple {
2.     public static void main(String [] args) {
3.         System.out.println("Hello Java");
4.     }
5. }
```

Compile by: javac Simple.java
Run by: java Simple

Литература

Bruce Eckel. Thinking in Java. 4th Edition, Pearson, 2006. ISBN-10: 0131872486, ISBN-13: 978-0131872486.

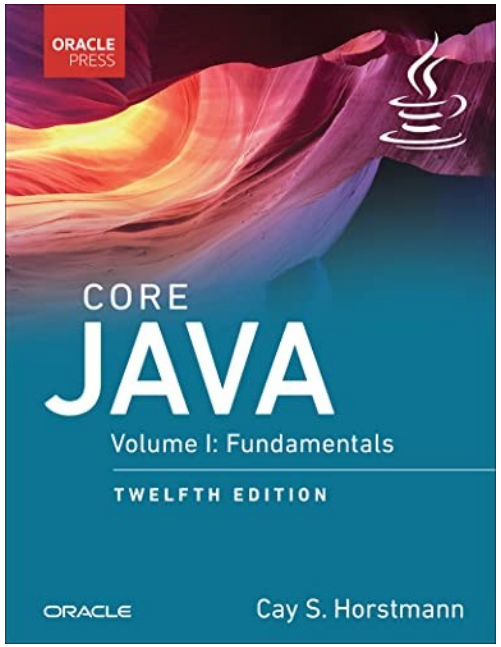


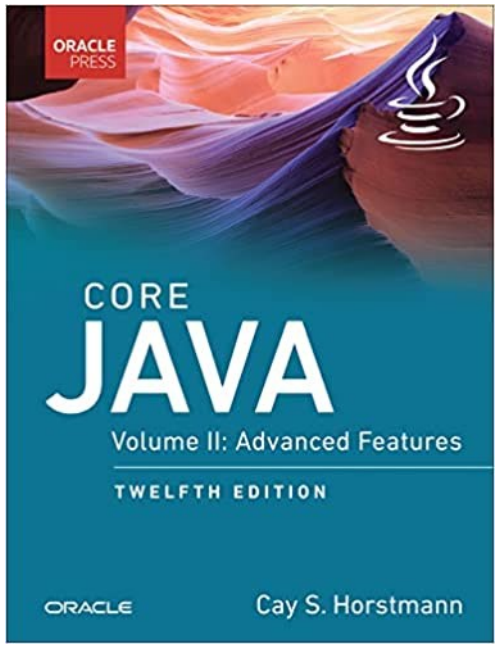


Cay S. Horstmann, Gary Cornell. Core Java 2, Volume I: Fundamentals. 6th Edition, Prentice Hall, 2003. ISBN-10: 0130471771, ISBN-13: 978-0130471772.

Литература

Cay S. Horstmann. Core Java, Volume I: Fundamentals. 12th Edition, Oracle Press, 2021. ISBN-10: 0137673620, ISBN-13: 978-0137673629.

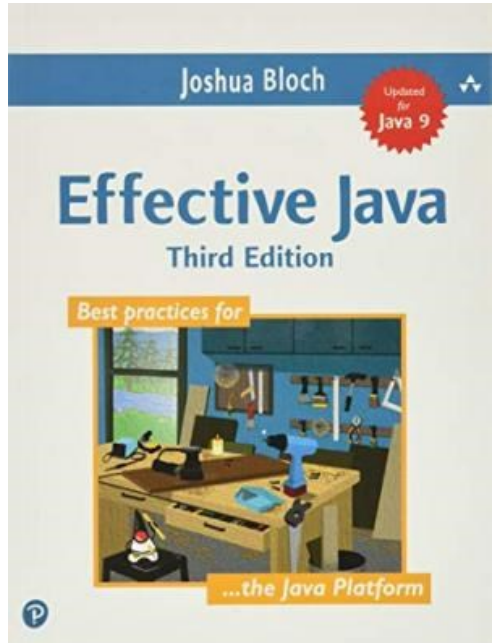




Cay S. Horstmann. Core Java, Volume 2: Advanced Features. 12th Edition, Oracle Press, 2022. ISBN-10: 0137871074, ISBN-13: 978-0137871070.

Литература

Joshua Bloch. Effective Java. 3rd Edition, Addison-Wesley, 2017. ISBN-10: 0134685997, ISBN-13: 978-0134685991.



Java® Platform, Standard Edition & Java Development Kit

https://docs.oracle.com/en/java/javase/19/docs/api/index.html

Java Tutorial

https://www.javatpoint.com/history-of-java

https://www.w3schools.com/java/

Robust

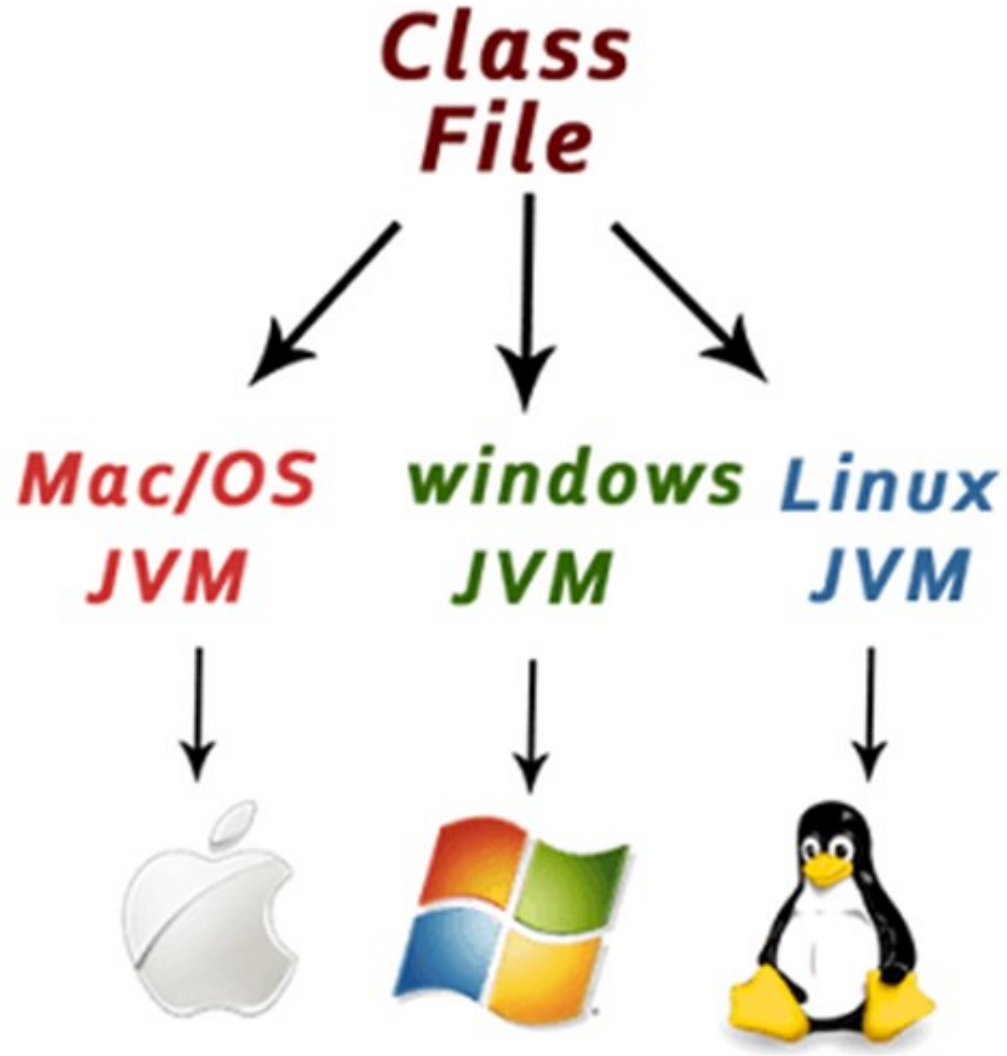
Английското значение на Robust е силен. Java е стабилна, защото притежава следните свойства:

- Той използва средства за управление на паметта.
- Липсват указатели, които избягват проблеми със сигурността.
- Java осигурява автоматично “събиране на боклука” (garbage collection), което работи на Java Virtual Machine, за да се отърве от обектите, които вече не се използват от Java приложенията.
- В Java има обработка на изключения и механизъм за проверка на типа.

Всички тези точки правят Java стабилна.

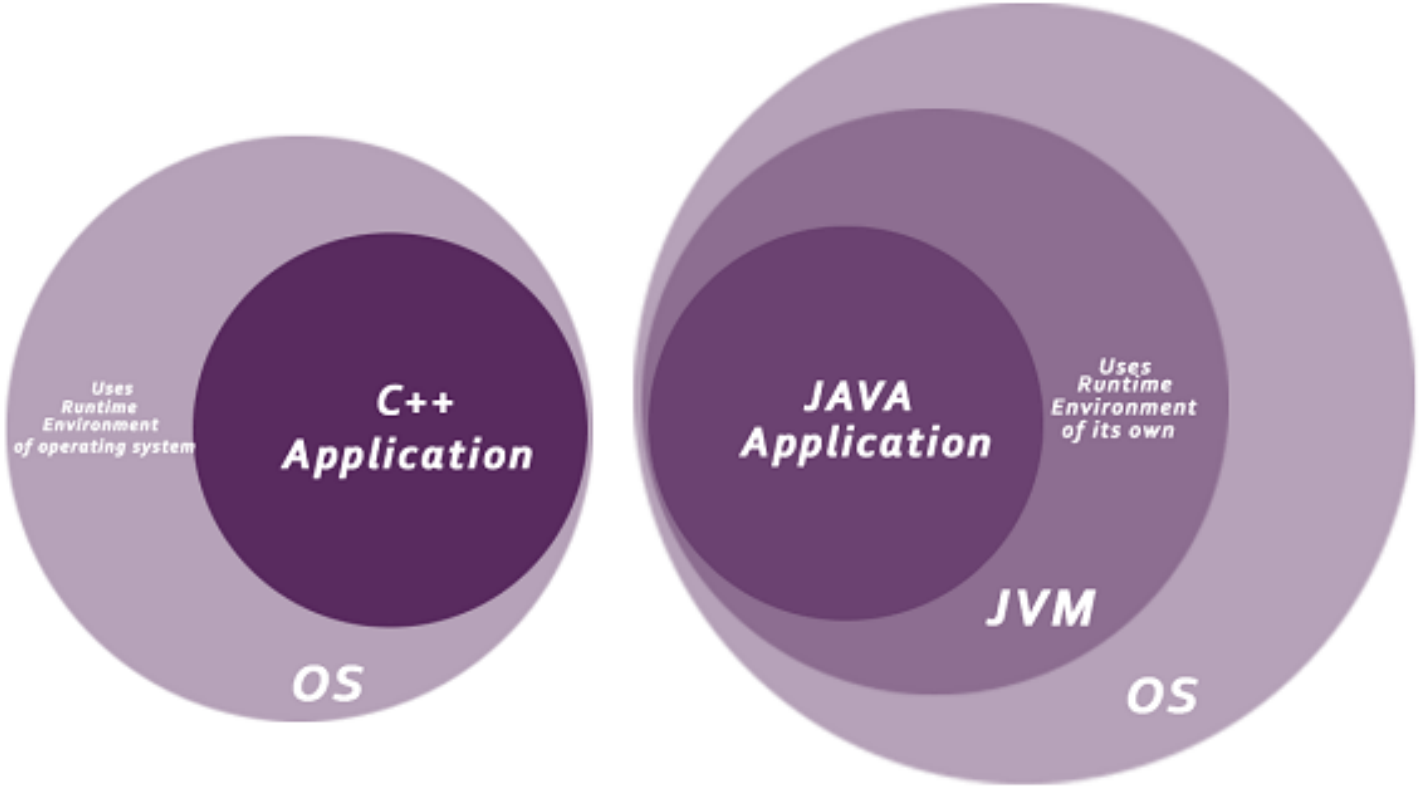
Return

Платформено независим



Return

Secured



Return

Architecture Neutral

Java е неутрална по отношение на архитектурата, защото няма функции, зависещи от изпълнението, например размерът на примитивните типове е фиксиран.

В програмният език C, типът за данни int заема 2 bytes от паметта за 32-bit архитектури и 4 bytes от паметта за 64-bit архитектури.

В Java типът за данни int заема 4 байта памет както за 32, така и за 64-битови архитектури в Java.

Return


Java™