Curs 1: Introducere in Java

Cuprins


Java
ORACLE
Status: Installing Java
3 Billion Devices "Run" Java
versioning issues, slowness, unplanned downtime, crashes,
support tickets, abandonware, unclear code, insecure
web apps, unexplained outtages, interfacing problems, bugs
ORACLE

#oow15 #Java20

5 28 39

Replies

TerrorBite @TerrorBite · Jun 12, 2020

Replying to @Oracle

Don't be silly. Three billion devices run Java. It says so right in the installer. There have always been three billion devices running Java. When one falls, another takes its place.
Three billion devices.

Java.

1 4 54

James Gosling,

inventatorul Java

Istoric Java

1991: firma Sun Microsystems finanteaza proiectul Green, coordonat de James Gosling

1995: sunt finalizate specificatiile noului limbaj denumit initial Oak, apoi Java

1996: **Java 1.0** apare prima implementare publica

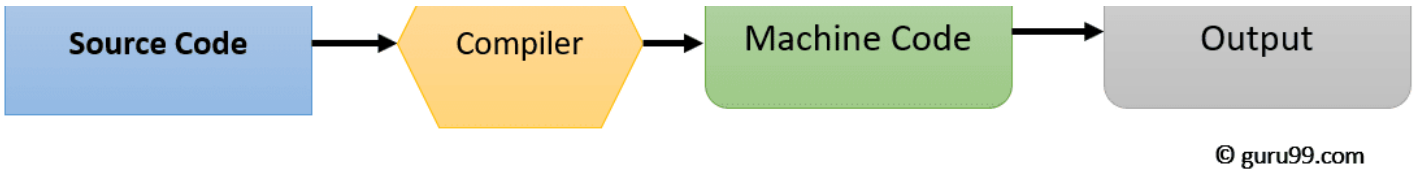
1997: **Java 1.1**: JDBC (Java Database Connectivity) si JIT (Just in Time) pentru JVM (Java Virtual Machine)

2009: Sun Microsystems este cumparata de Oracle

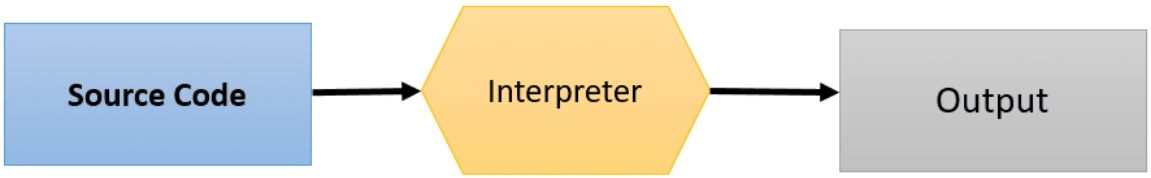
2014: **Java 8**: lambda expresii si programare functionala

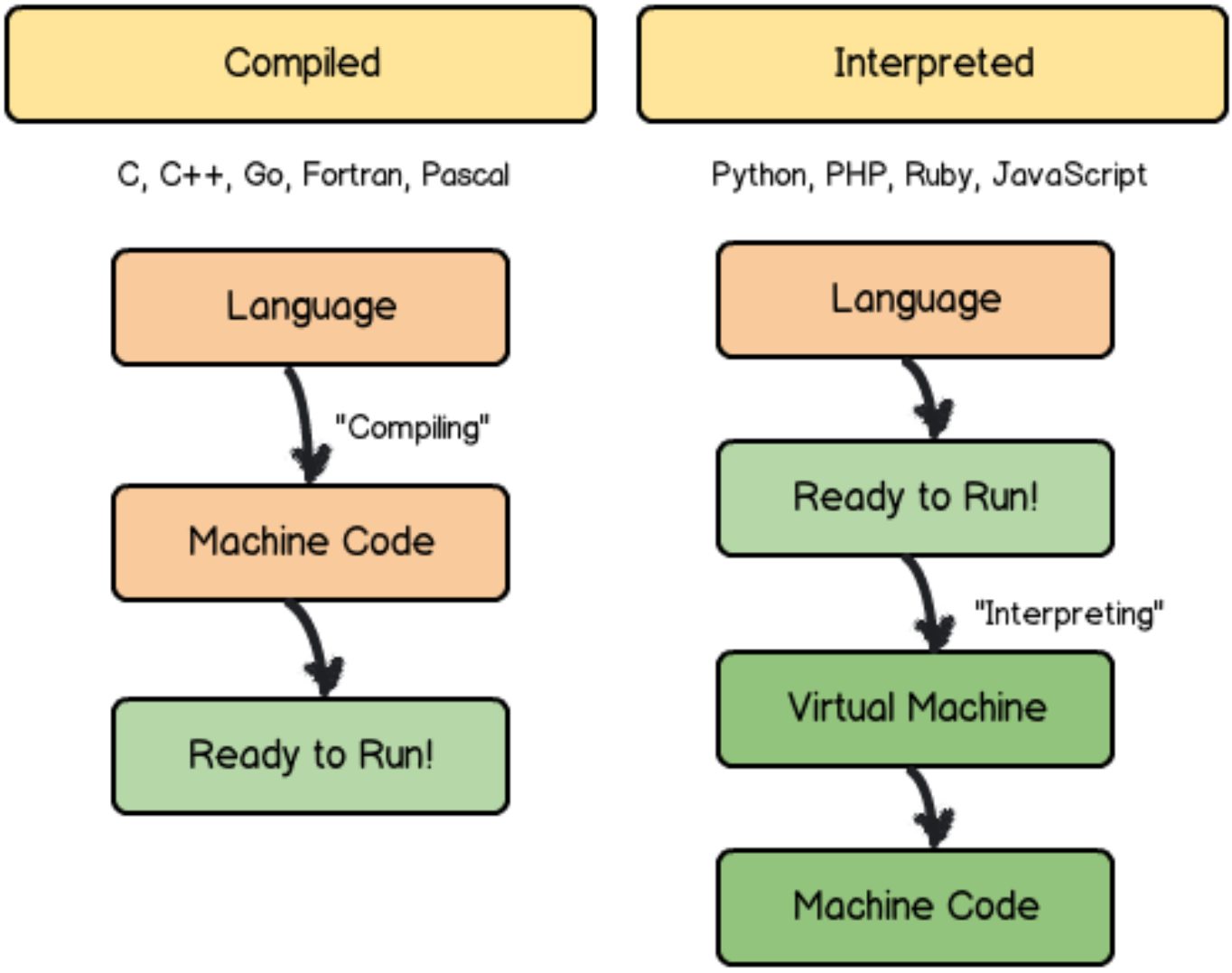
2021: **Java 15**

Java - limbaj compilat si interpretat!

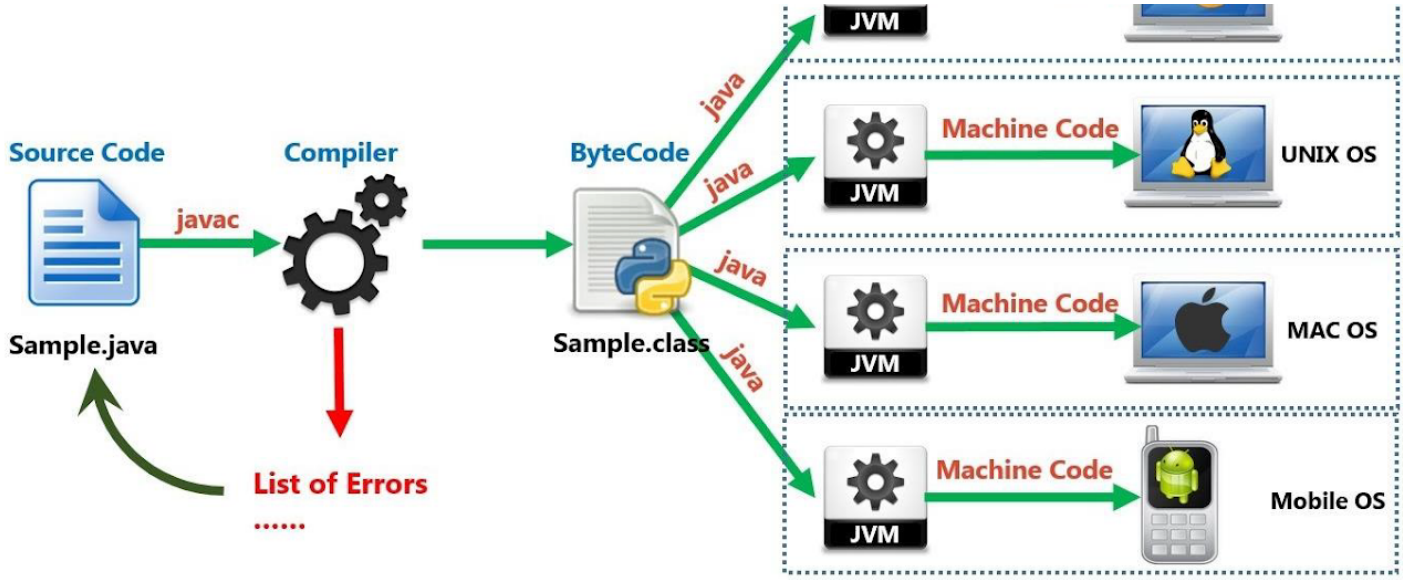


© guru99.com





Java - Write Once, Run Anywhere



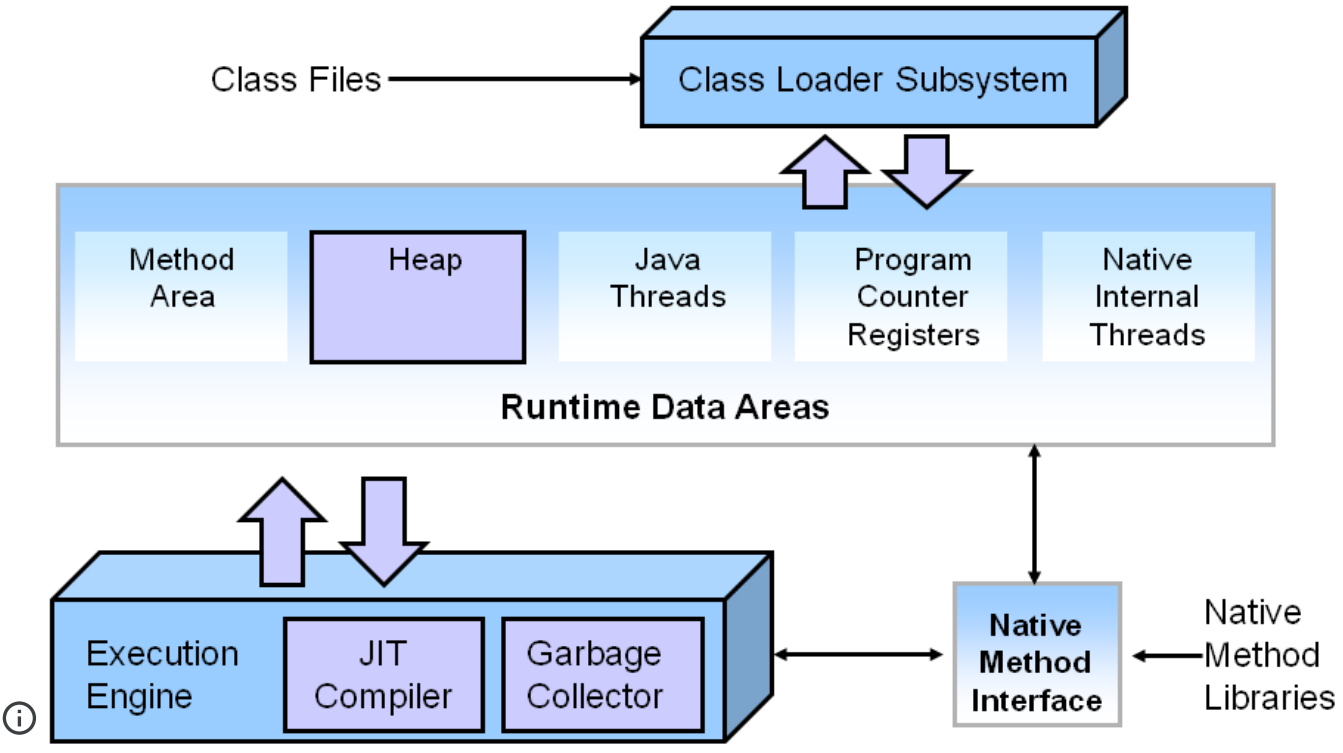
Java Bytecode = instructiuni pentru JVM

C++/Java	X86 ASM	Java bytecode (mnemonics)	Java bytecode (hexadecimal)

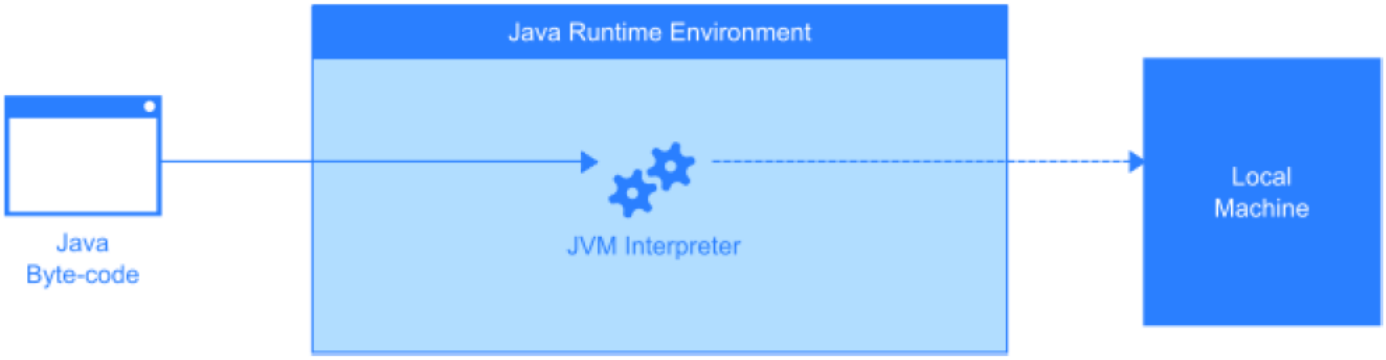
int add(int a, int b)
{
return a+b;
}
	mov eax, byte[ebp-4]	iload_0	0x1A

	mov edx, byte[ebp-8]	iload_1	0x1B
	add eax, edx	iadd	0x60
	ret	ireturn	0xAC

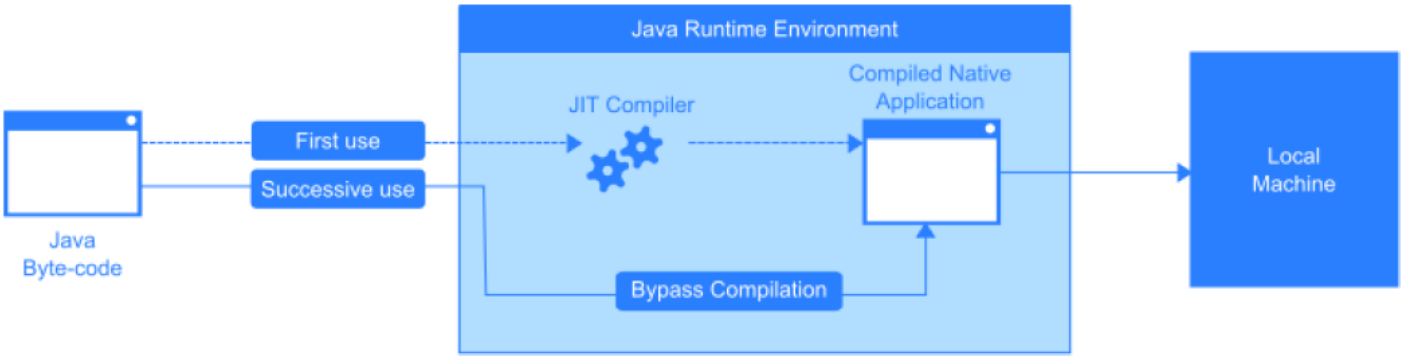
Java Virtual Machine (JVM)



JVM Interpreter : interpreteaza si ruleaza bytecode-ul.



JIT Compiler: este chemat pentru bytecode-ul care se executa frecvent si il transforma in cod masina nativ specific procesorului gazda.



Caracteristicile limbajului Java

Orientat pe obiecte

- Orice program contine cel putin o clasa.
- Nu mai exista functii independente, acestea devin metode.

Simplu

- Java mentine o sintaxa de tip C++ din care sunt eliminate concepte precum:
  - pointeri
  - supraincarcarea operatorilor
  - mostenirea multipla
  - struct/union

Robust

- Management automat al memoriei (Garbage Collector)
- Strong data typed = toate datele cu care lucram sunt dintre tipurile predefinite
- Mecanism de tratare al exceptiilor try-throw-catch

- Masina virtuala Java asigura permisiunile si securitatea
- Sandbox. Aplicatiile pot rula in JVM-uri distincte, izolate. Esecul unei aplicatii nu compromite intreg sistemul.

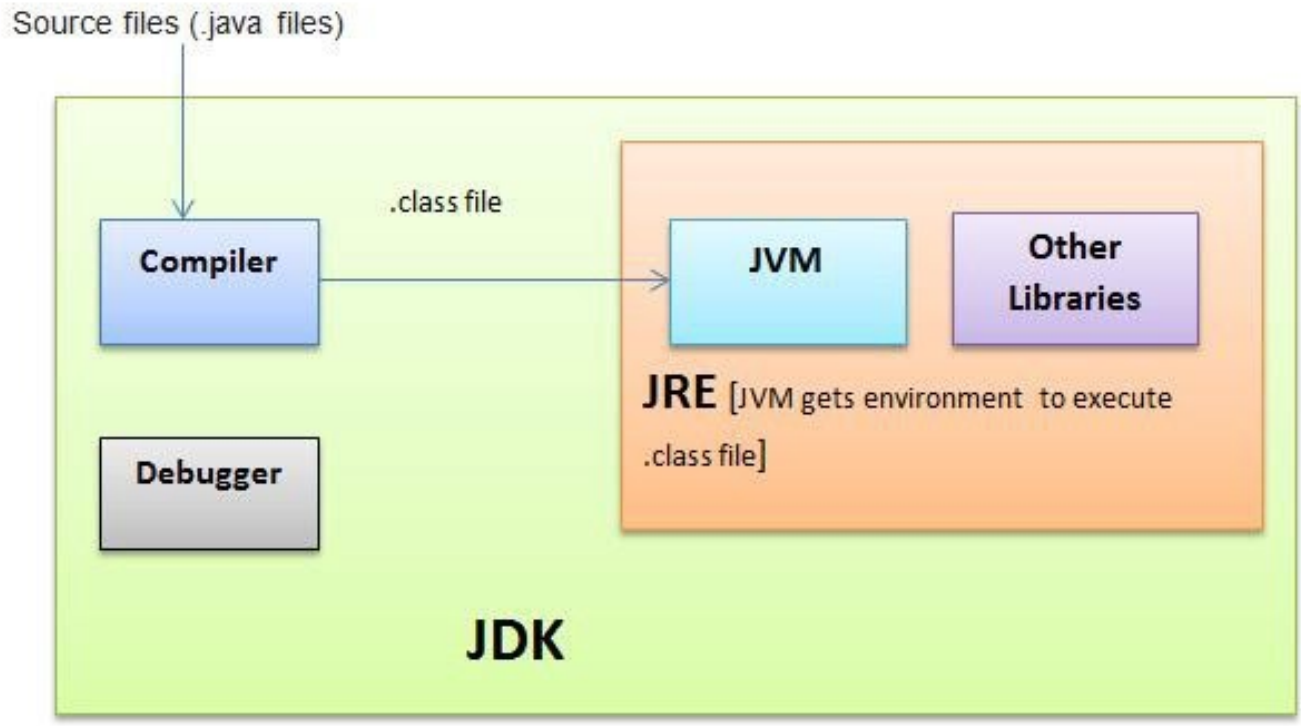
Paralel si Distribuit

- Suport nativ pentru fire de executie (thread-safe)
- Biblioteci cu primitive specifice pentru Java Threads.

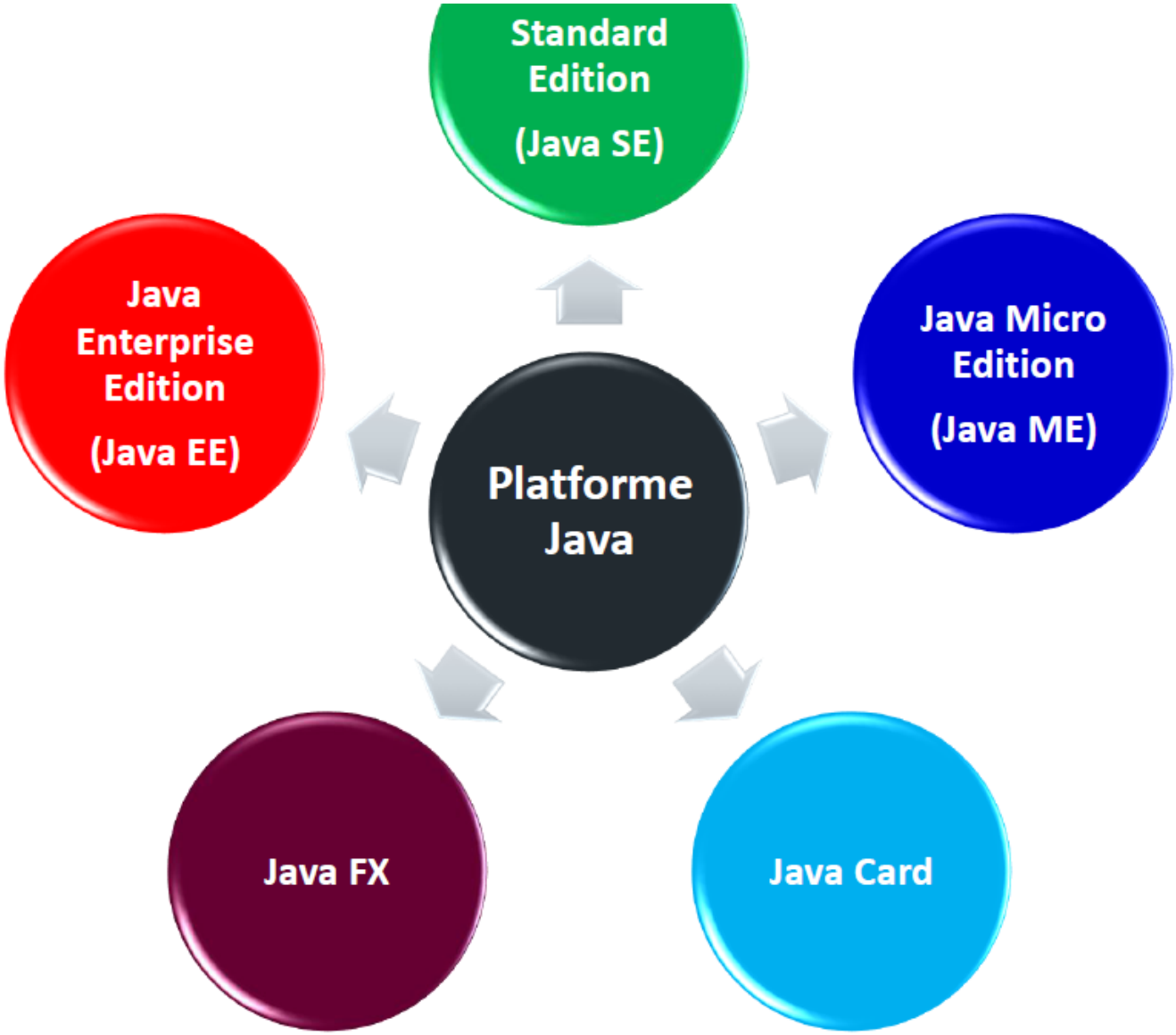
Componentele software Java

JRE - Java Runtime Environment. Contine JVM masina virtuala java si toate componentele necesare rularii de programe Java compilate in bytecode (.class).

JDK - Java Development Kit. Contine JRE si compilatorul Java care produce .class din surse .java.



Platformele Java



Java SE - Standard Edition - varianta pentru aplicatii tip desktop

Java EE - Enterprise Edition - cu extra librarii pentru aplicatii multistratificate client-server

Java ME - Micro Edition - cu librarii modificate pentru hardware cu memorie limitata, telefoane mobile etc.

Java FX - pentru Rich-Internet-Applications dar si aplicatii desktop

Java Card = pentru smartcarduri

Setul de caractere

Unicode

- 65536 simboluri = un caracter se reprezinta pe 2 octeti
- Este compatibil cu ASCII: primele 256 caractere sunt cele din ASCII.
- ⓘ te structurat în blocuri: Basic, Latin, Greek, Arabic, Gothic, Currency, Mathematical, Arrows, Musical, etc.

Intregi

- baza 10, de exemplu 123
- baza 16, de exemplu **0x**7B sau **0X**7B
- baza 8, de exemplu **0**173
- baza 2, de exemplu **0b**1111011 sau **0B**1111011

Numere cu virgula mobila

- 123.0 //8 octeti double
- 123.0**f** //4 octeti float

Boolean

- true sau false

Caractere

- 'A' sau in unicode '\u0041' unde 0041 este codul in baza 16

Siruri de caractere

- "Sir de caractere"

null

- valoarea implicita a oricarei referinte neinitializate

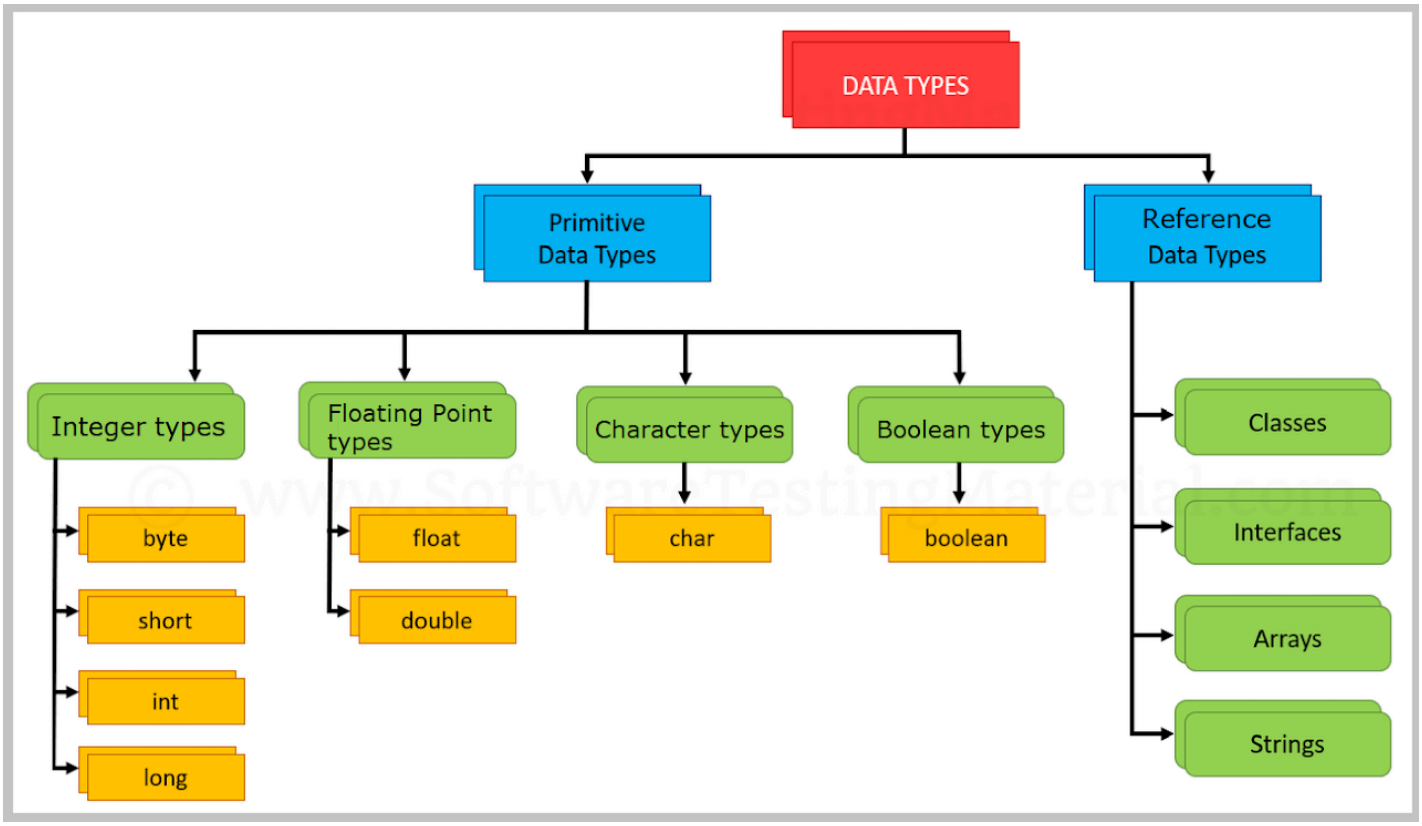
Cuvinte cheie

Java Keywords

abstract	assert	boolean	break	byte
case	catch	char	class	continue
default	do	double	else	enum
extends	final	finally	float	for
if	implements	import	instanceof	int
interface	long	native	new	package
private	protected	public	return	short
static	strictfp	super	switch	synchronized
this	throw	throws	transient	try
void	volatile	while		

Keywords that are not currently used

...st	goto



Tipuri de date primitive

Fiecare tip de date primitiv are o clasa infasuratoare (wrapper) ce contine diverse campuri (dimensiune maxima, minima etc.) si metode care permit crearea de obiecte corespunzatoare tipurilor primitive.

Primitive Types					
Type Name	Wrapper class	Value	Range	Size	Default Value
`byte`	`java.lang.Byte`	integer	−128 through +127	8-bit (1-byte)	`0`
`short`	`java.lang.Short`	integer	−32,768 through +32,767	16-bit (2-byte)	`0`
`int`	`java.lang.Integer`	integer	−2,147,483,648 through +2,147,483,647	32-bit (4-byte)	`0`
`long`	`java.lang.Long`	integer	−9,223,372,036,854,775,808 through +9,223,372,036,854,775,807	64-bit (8-byte)	`0`
`float`	`java.lang.Float`	floating point number	±1.401298E−45 through ±3.402823E+38	32-bit (4-byte)	`0.0`
`double`	`java.lang.Double`	floating point number	±4.94065645841246E−324 through ±1.79769313486232E+308	64-bit (8-byte)	`0.0`
`boolean`	`java.lang.Boolean`	Boolean	`true` or `false`	8-bit (1-byte)	`false`
`char`	`java.lang.Character`	UTF-16 code unit (BMP character or a part of a surrogate pair)	`'\u0000'` through `'\uFFFF'`	16-bit (2-byte)	`'\u0000'`

Clase infasuratoare

Long.**MAX_VALUE** //2^63 -1
Long.**SIZE** //64
Long.**BYTES** //8

Constructori

Long long = new Long(1234**l**);
Long long = new Long(**"**1234**"**);

Metode

Long.**parseLong**("12345") //12345l

Long.**compare**(1234l, 12345l) //-1
Long long = new Long(1234**l**);
long.**compareTo**(12345l) //-1

Long.**toString**(long); //"12345"
Long long = new Long(12345**l**);
long.**toString**() //"12345"

Long.**valueOf**(1234l) // Long obj
Long.**valueOf**("1234") // Long obj

Operatori

Precedence	Operator	Operand type	Description
1	++,	Arithmetic	Increment and decrement
1	+, -	Arithmetic	Unary plus and minus
1	~	Integral	Bitwise complement
1	!	Boolean	Logical complement
1	(*type*)	Any	Cast
2	*, /, %	Arithmetic	Multiplication, division, remainder
3	+, -	Arithmetic	Addition and subtraction
3	+	String	String concatenation
4	<<	Integral	Left shift
4	>>	Integral	Right shift with sign extension
4	>>>	Integral	Right shift with no extension
5	<, <=, >, >=	Arithmetic	Numeric comparison
5	instanceof	Object	Type comparison
6	==, !=	Primitive	Equality and inequality of value
6	==, !=	Object	Equality and inequality of reference
7	&	Integral	Bitwise AND
7	&	Boolean	Boolean AND
8	^	Integral	Bitwise XOR
8	^	Boolean	Boolean XOR
9	|	Integral	Bitwise OR
9	|	Boolean	Boolean OR
10	&&	Boolean	Conditional AND
11	||	Boolean	Conditional OR
12	?:	N/A	Conditional ternary operator
13	=	Any	Assignment

Decision

if-then
if-then-else
switch

Loop-ing

while
do-while
for

Branch-ing

break
continue
return

Biblioteci standard