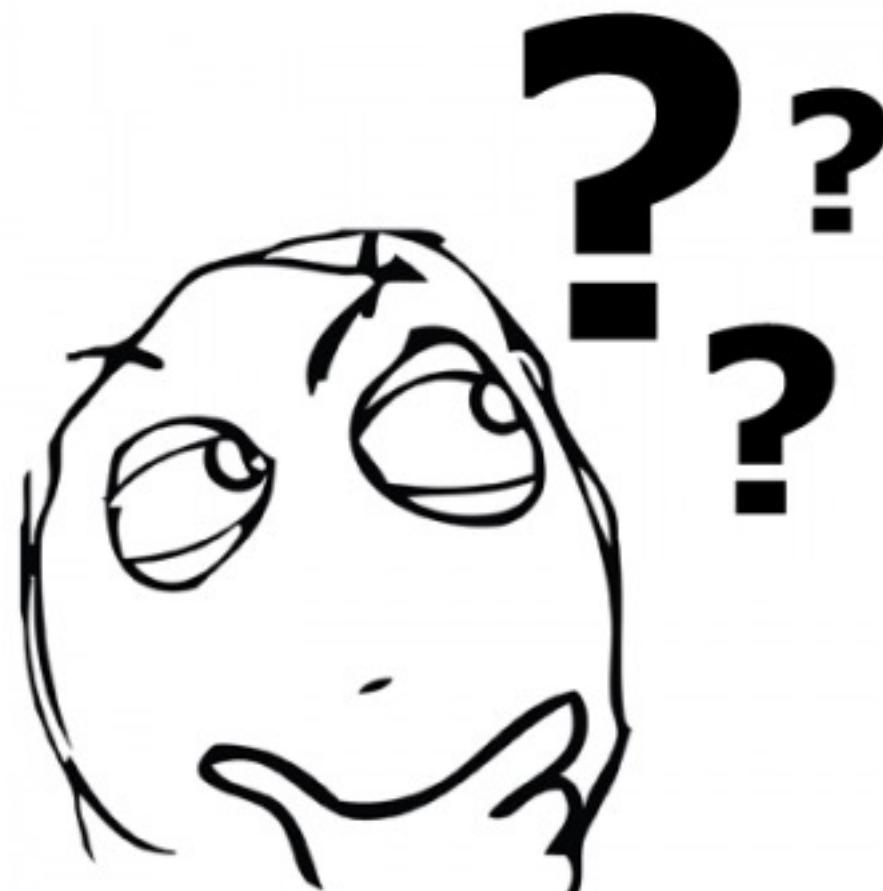


Estrutura de Dados

Aula 1

GUSTAVO FORTUNATO PUGA

Algoritmo



Dicionário

Definições de [Oxford Languages](#) · [Saiba mais](#)

Pesquise uma palavra



algoritmo

substantivo masculino

1. **MATEMÁTICA**

sequência finita de regras, raciocínios ou operações que, aplicada a um número finito de dados, permite solucionar classes semelhantes de problemas.

2. **INFORMÁTICA**

conjunto das regras e procedimentos lógicos perfeitamente definidos que levam à solução de um problema em um número finito de etapas.

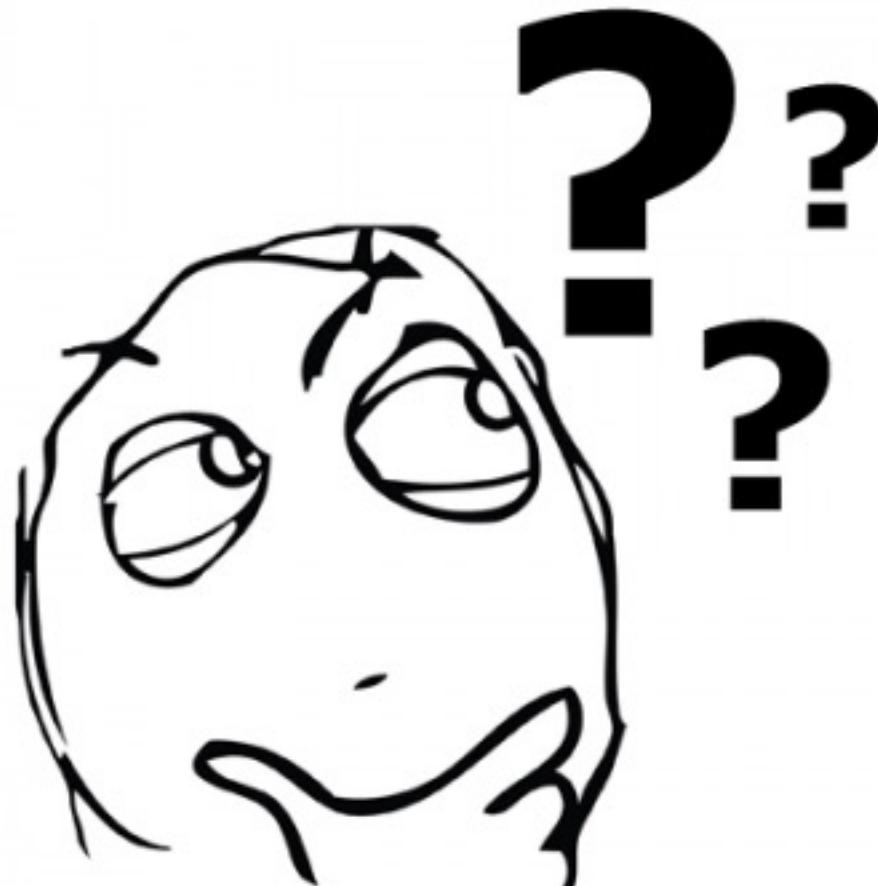
Origem

○ ETIM lat.medv. *algorismus*, com infl. do gr. *arithmós* 'número'



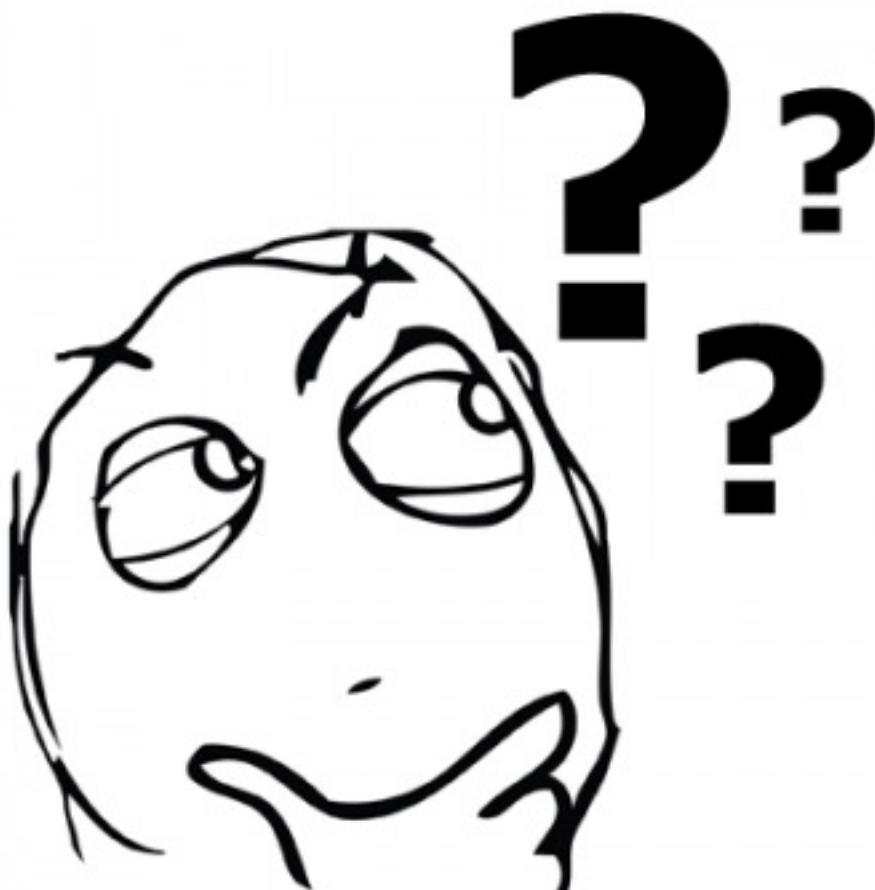
-

Software



• Software

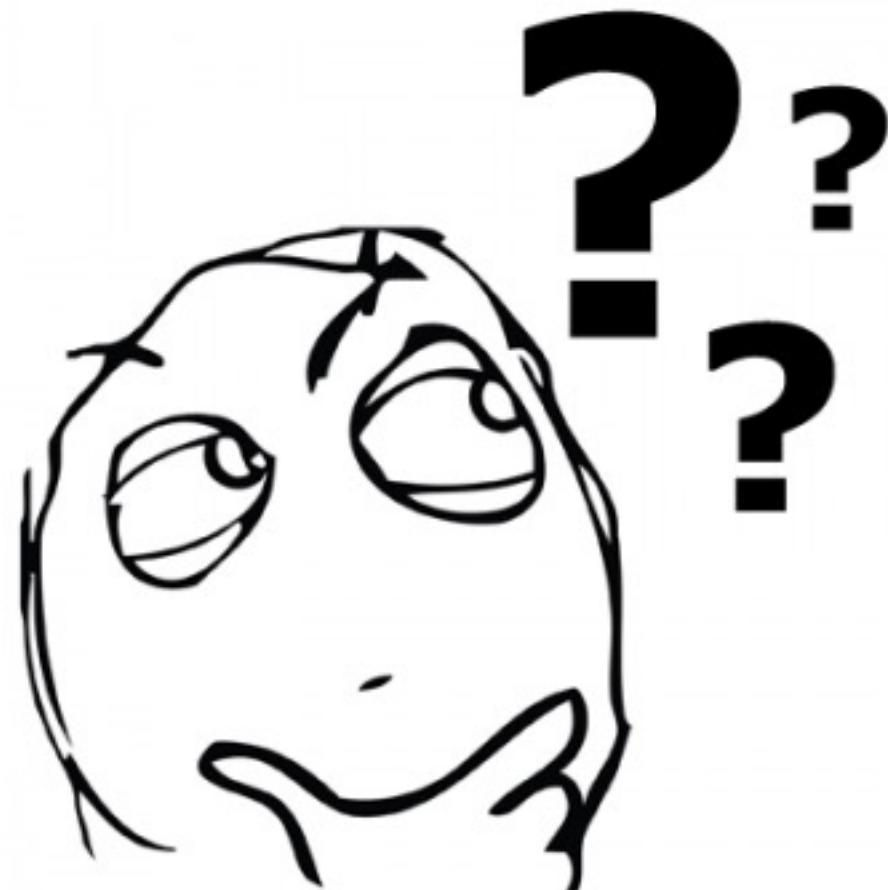
BASICAMENTE A TRADUÇÃO COMPUTACIONAL
DE UM ALGORITMO



SEQUÊNCIA DE INSTRUÇÕES QUE
MANIPULAM, REDIRECIONAM OU
MODIFICAM DADOS OU
ACONTECIMENTOS

-

Estrutura de dados



-

Estrutura de dados



O ramo da computação que estuda os diversos mecanismos de organização de dados para atender aos diferentes requisitos de processamento.

Definem a organização, métodos de acesso e opções de processamento para a informação manipulada pelo programa

Estrutura de dados



<https://www.youtube.com/watch?v=EfF1M7myAyY>





JAVA



[https://canaltech.com.br/software/java-o-que-o-cafezinho-tem-a-
ver-com-a-origem-da-linguagem-de-programacao-207312/](https://canaltech.com.br/software/java-o-que-o-cafezinho-tem-a-ver-com-a-origem-da-linguagem-de-programacao-207312/)



-

JAMES GOSLING: O pai do Java

Em 1991 a Sun cria um time para desenvolver inovações tecnológicas.



-

Projeto Green: Linguagem OAK



A ideia era de que essa linguagem fosse usada em pequenos dispositivos, como TVs, videocassetes, aspiradores, liquidificadores



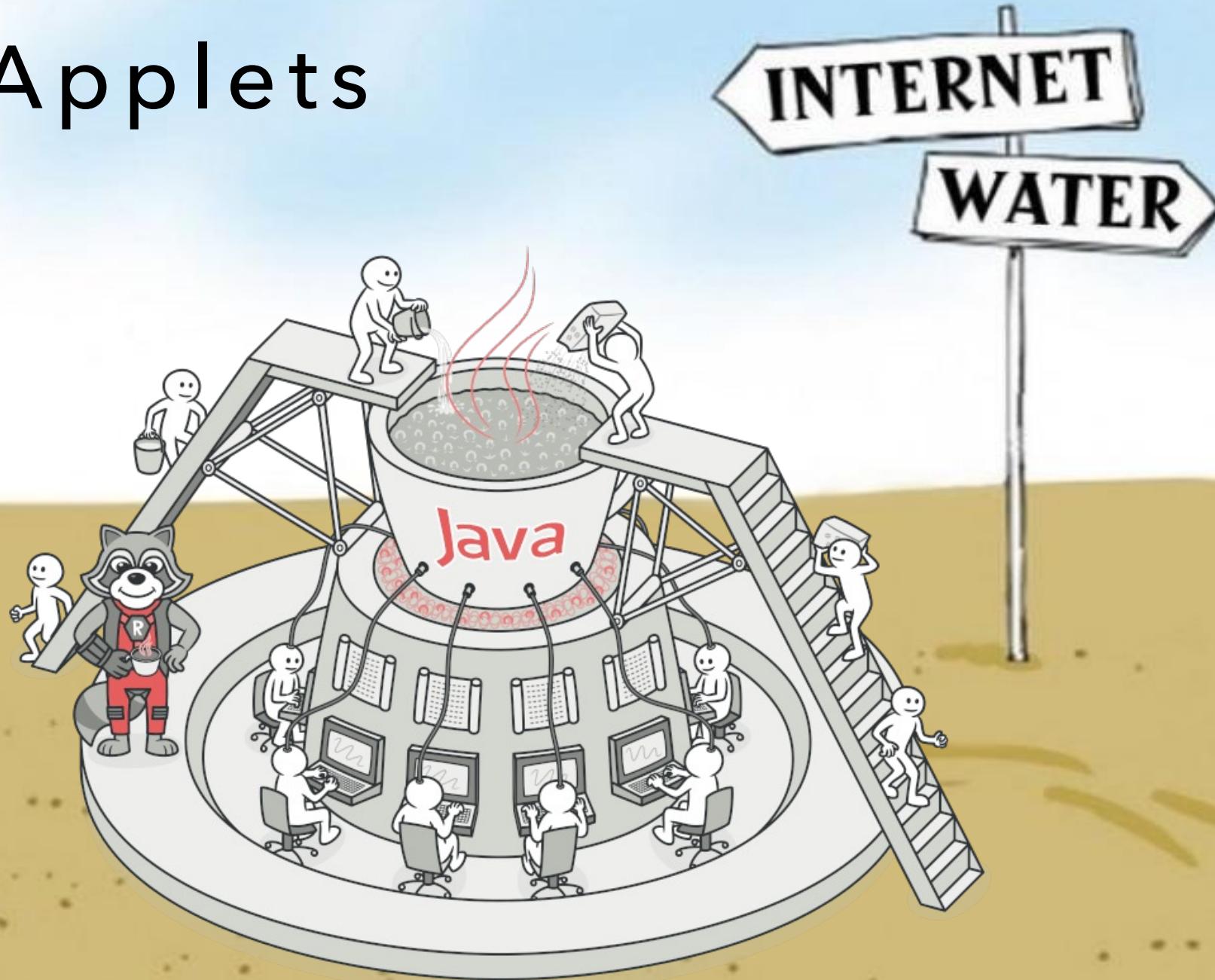
1995

A década de 90 ficou conhecida
como o "boom da internet"

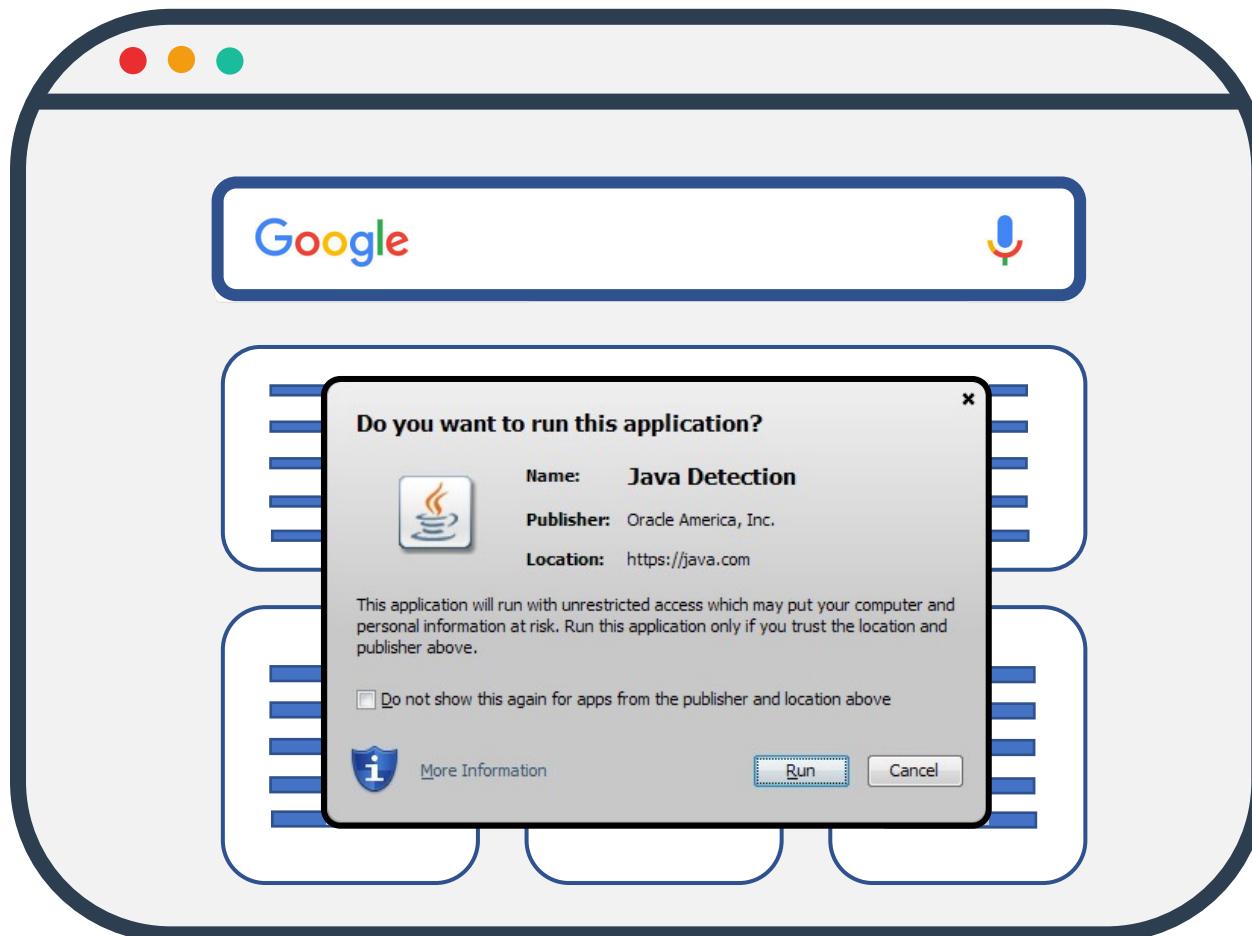


-

Applets



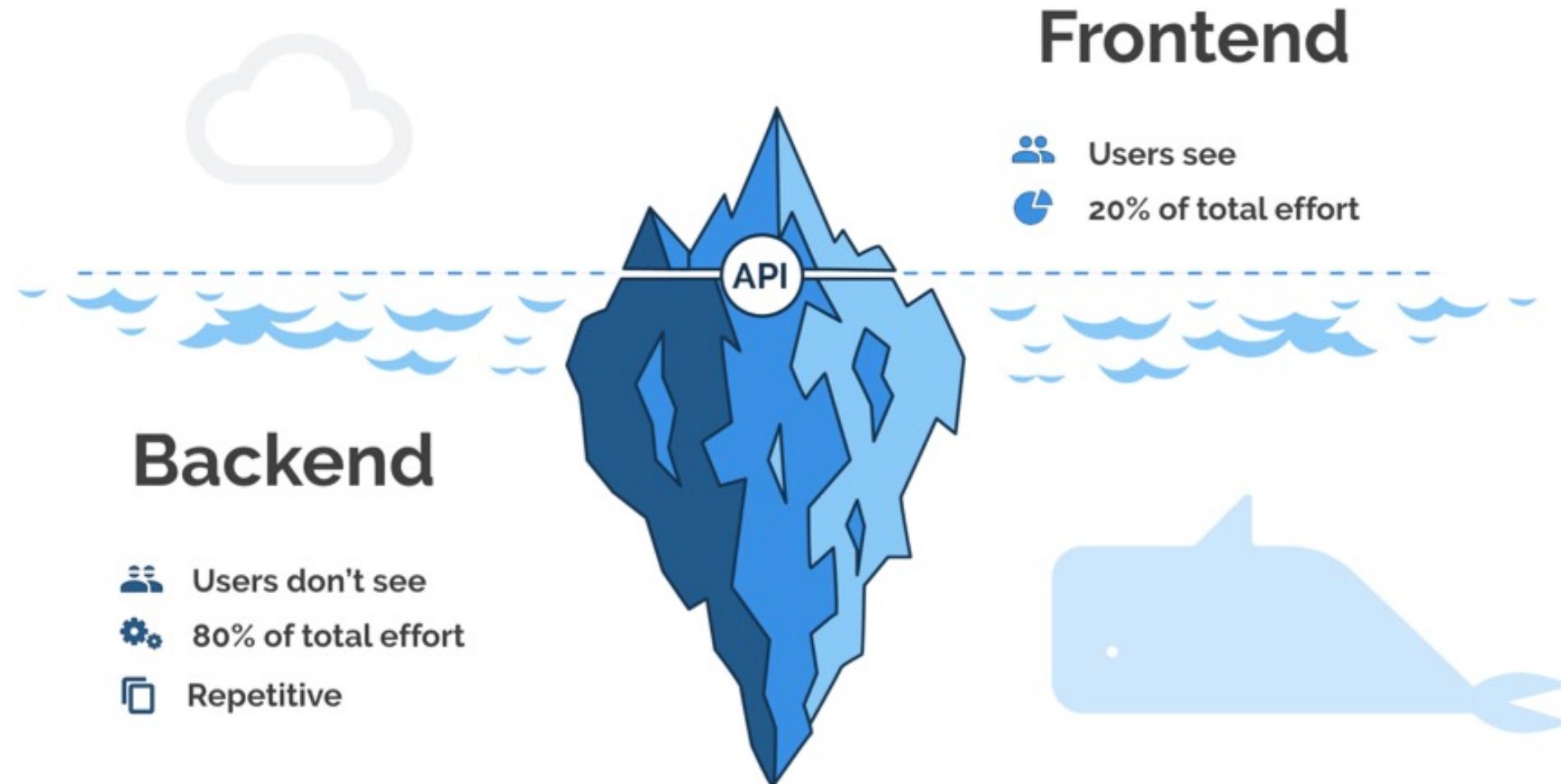
Applets



-

Java

Embora tenha sido idealizado com um propósito e lançado com outro, o Java ganhou destaque no lado do servidor.



Java

Em 2009, a Oracle comprou a Sun, fortalecendo a marca. A Oracle sempre foi, junto à IBM, uma das empresas que mais investiram e fizeram negócios por meio do uso da plataforma Java. Em 2014, surge a versão Java 8 com mudanças interessantes na linguagem.

ORACLE



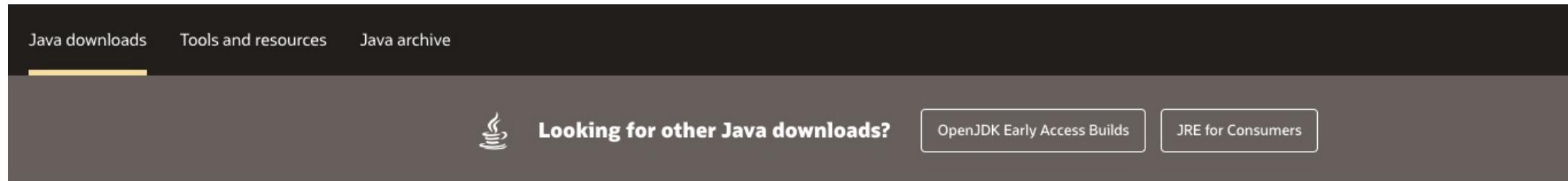
Java

O Java foi criado pela antiga Sun Microsystems, pertence a Oracle e é mantido por meio de um comitê denominado JCP (<http://www.jcp.org>)



Java

- JVM: apenas a virtual machine. Esse download não existe, pois ela sempre vem acompanhada.
- JRE: Java Runtime Environment. Ambiente de execução Java, formado pela JVM e bibliotecas, tudo que você precisa para executar uma aplicação Java.
- **JDK: Java Development Kit. Para desenvolvedor, faremos o download do JDK do Java SE (Standard Edition). Ele é formado pela JRE somado às ferramentas como o compilador.**



Java 17 available now

Java 17 LTS is the latest long-term support release for the Java SE platform. JDK 17 binaries are free to use in production and free to redistribute, at no cost, under the [Oracle No-Fee Terms and Conditions](#).

[Learn about Java SE Subscription](#)

JDK 17 will receive updates under these terms, until at least September 2024.

Java SE Development Kit 17.0.2 downloads

Thank you for downloading this release of the Java™ Platform, Standard Edition Development Kit (JDK™). The JDK is a development environment for building applications and components using the Java programming language.

The JDK includes tools for developing and testing programs written in the Java programming language and running on the Java platform.



Java

Em 2018 há mudanças na distribuição e suporte da JDK da Oracle

Java SE / OpenJDK / Build do OpenJDK da Oracle / JDK da Oracle

A comunidade do [OpenJDK](#) cria e mantém a Implementação de Referência (Reference Implementation (RI)) código-aberto (GPLv2+CE) da Especificação do **Java SE** como governado pelo [Java Community Process](#) (JCP) e que é definido como uma Java Specification Request (JSR) guarda-chuva para cada release futuro.

Existem implementações do Java SE de diversos provedores (como Azul, Eclipse, IBM, Red Hat, Oracle, SAP, e outros), o mais comum sendo o **JDK da Oracle (Oracle JDK)**.

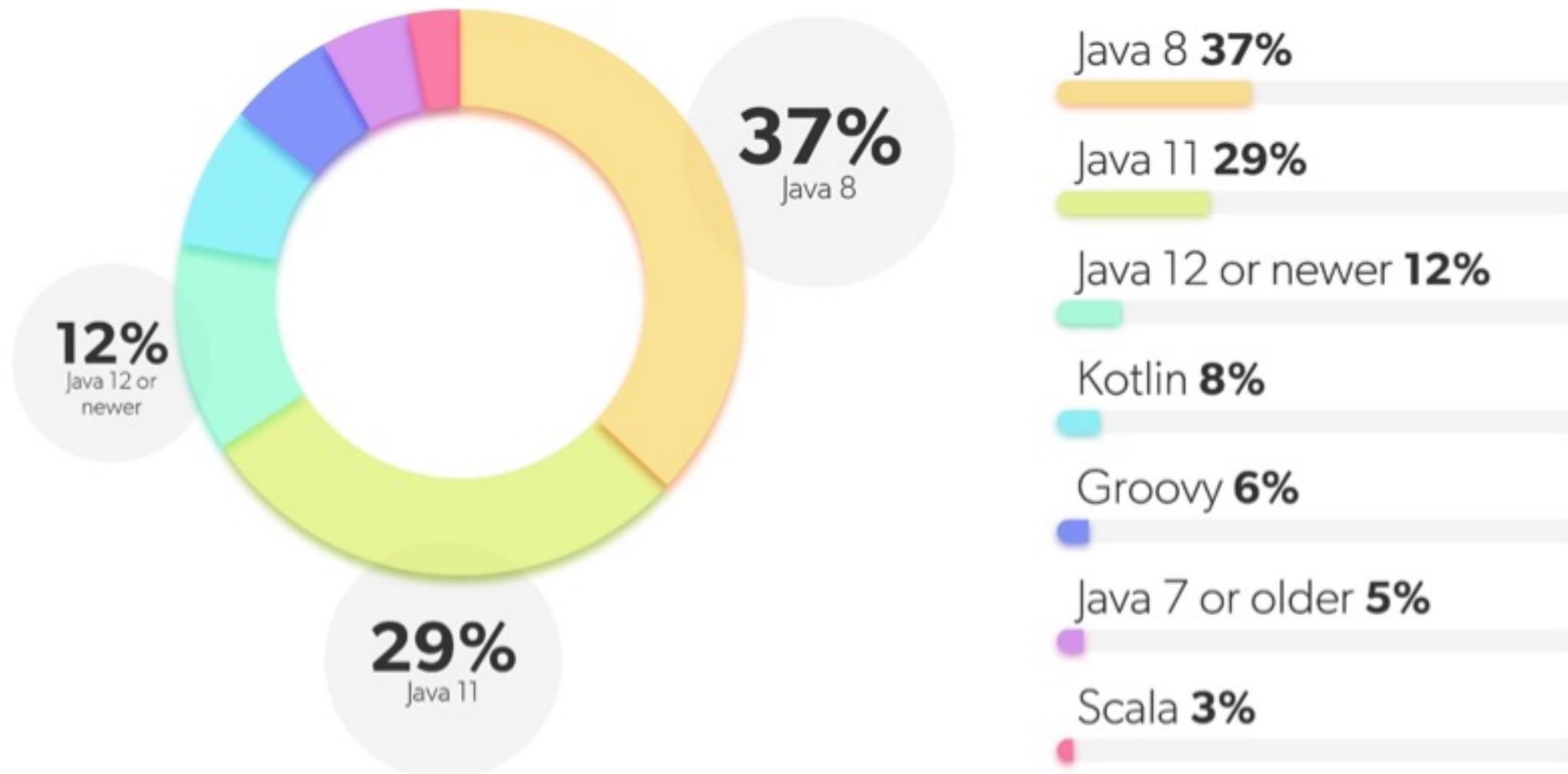
Oracle JDK 8 está no [processo](#) de "*Fim de Atualizações Públicas*" o que significa que não haverá mais atualizações gratuitas para fins comerciais ao final de Janeiro de 2019. Porém, desde o Java SE 9, a Oracle está disponibilizando também o [builds OpenJDK](#) que são livres para uso comercial, e há também builds gratuitos do [OpenJDK](#) de outras empresas como [AdoptOpenJDK](#), Azul, IBM, Red Hat, Linux distros e outros.

Essas empresas provaram que suas implementações atendem a todos os requisitos da especificação Java SE ao passarem pelos testes de Technology Compatibility Kit (TCK).

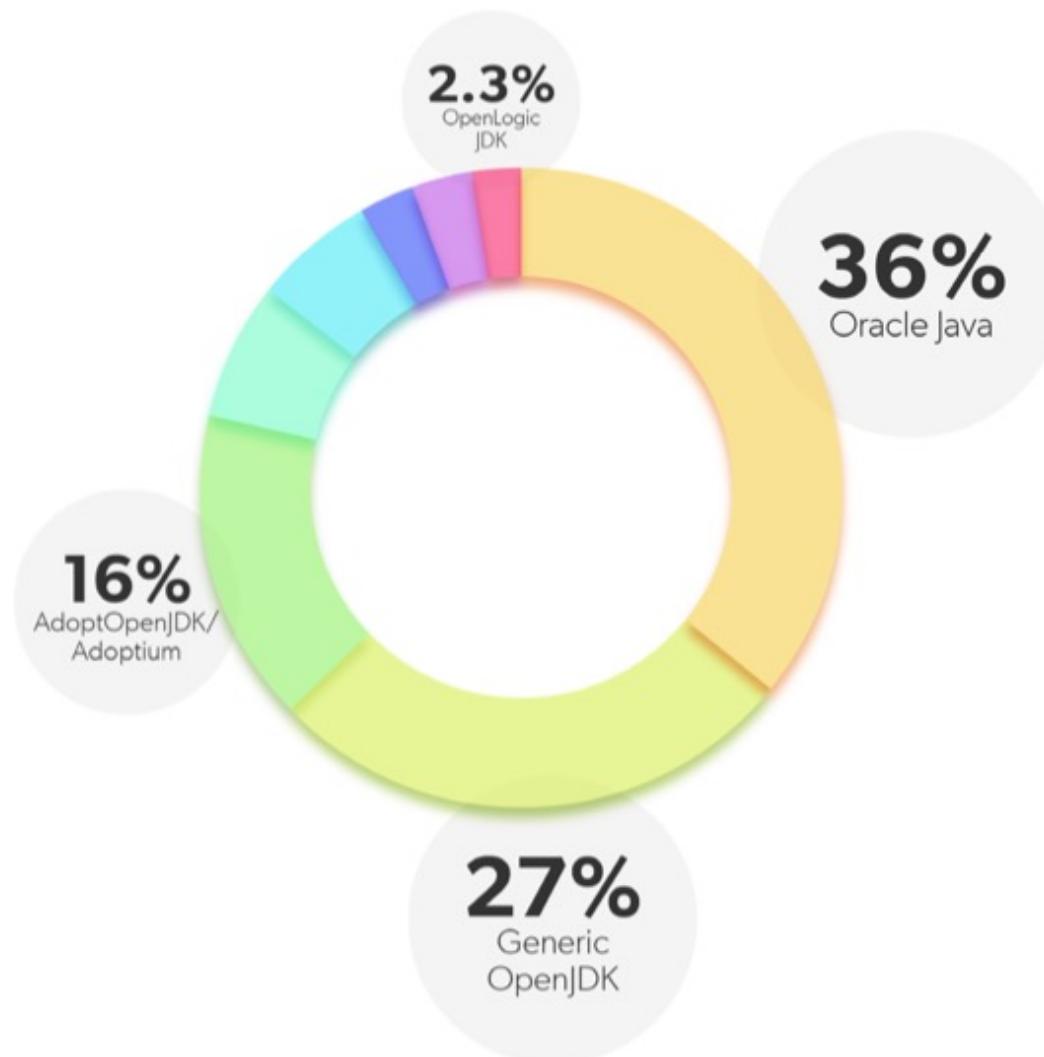




Which JDK Programming Languages are You Using in Your Main Application?



What JRE/JDK Distribution Do You Use?



Oracle Java **36%**

Generic OpenJDK **27%**

AdoptOpenJDK/Apoptium **16%**

Amazon Corretto **7%**

Azul Zulu **6%**

GraalVM **3%**

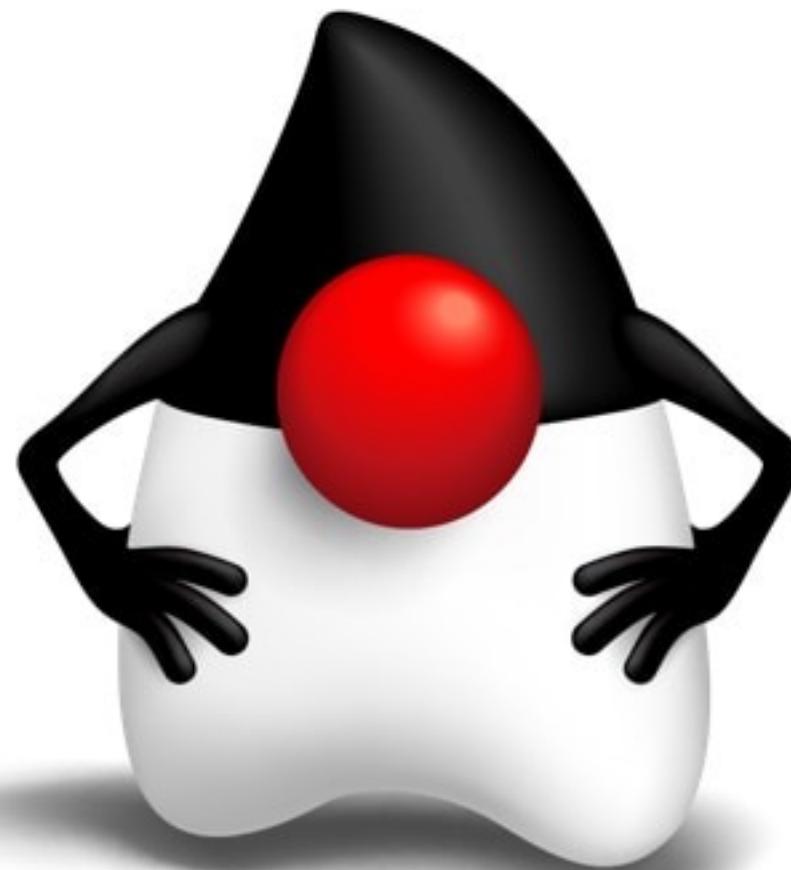
Other **3%**

OpenLogic JDK **2.3%**



-

Porque aprender Java?



<https://www.oracle.com/java/duke/>





Porque aprender Java?

Três índices:

- Tiobe:
 - Popularidade das linguagens de programação. Atualizado mês a mês. As avaliações são baseadas no número de engenheiros especializados no mundo, cursos e fornecedores de software. Motores de busca populares como Google, Bing, Yahoo!, Wikipedia, Amazon, YouTube e Baidu são usados para calcular as classificações.
 - Conta o número de páginas da web com o nome da linguagem.
 - O índice não é sobre a melhor linguagem de programação ou o maior número de linhas de código gerados.





Porque aprender Java?

- PYPL:
 - Criado analisando a frequência com que os tutoriais das linguagens são pesquisados no Google.
 - Os dados brutos vêm do Google Trends.
- IEEE Spectrum: Ranking utiliza 12 métricas ponderadas de 10 fontes de dados.





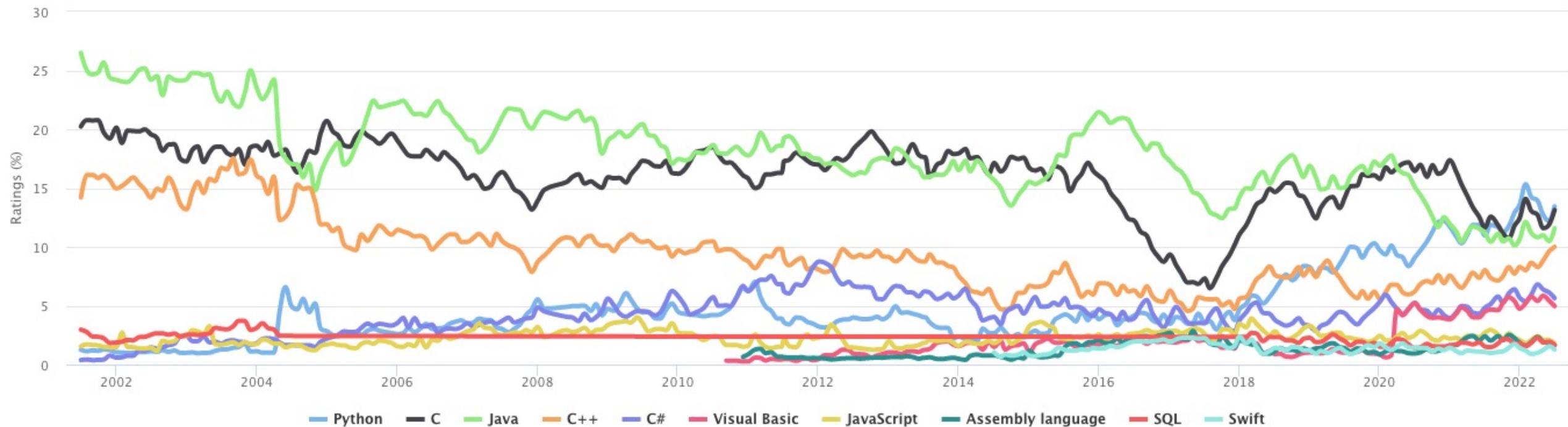
Jul 2022	Jul 2021	Change	Programming Language	Ratings	Change
1	3	Difference compared to last year	Python	13.44%	+2.48%
2	1		C	13.13%	+1.50%
3	2		Java	11.59%	+0.40%
4	4		C++	10.00%	+1.98%
5	5		C#	5.65%	+0.82%
6	6		Visual Basic	4.97%	+0.47%
7	7		JavaScript	1.78%	-0.93%
8	9		Assembly language	1.65%	-0.76%
9	10		SQL	1.64%	+0.11%
10	16		Swift	1.27%	+0.20%
11	8		PHP	1.20%	-1.38%
12	13		Go	1.14%	-0.03%





TIOBE Programming Community Index

Source: www.tiobe.com



<https://www.tiobe.com/tiobe-index/>





PYPL Popularity of Programming Language

Worldwide, Aug 2022 compared to a year ago:

Rank	Change	Language	Share	Trend
1		Python	28.11 %	-2.6 %
2		Java	17.35 %	-0.9 %
3		JavaScript	9.48 %	+0.2 %
4		C#	7.08 %	+0.1 %
5		C/C++	6.19 %	-0.3 %
6		PHP	5.47 %	-0.8 %
7		R	4.35 %	+0.6 %
8	↑↑	TypeScript	2.79 %	+1.1 %
9	↑↑	Swift	2.09 %	+0.5 %
10	↓↓	Objective-C	2.03 %	+0.2 %

<https://pypl.github.io/>



Rank	Language	Type		Score
1	Python▼	🌐	💻	100.0
2	Java▼	🌐	📱	95.4
3	C▼	📱	💻	94.7
4	C++▼	📱	💻	92.4
5	JavaScript▼	🌐		88.1
6	C#▼	🌐	📱	82.4
7	R▼		💻	81.7
8	Go▼	🌐	💻	77.7
9	HTML▼	🌐		75.4
10	Swift▼	📱	💻	70.4

Choose a Ranking

IEEE Spectrum
Trending

Jobs
Open
Custom

Create Custom Ranking

Language Types

Web
🌐
Enterprise
💻

Mobile
📱
Embedded
⚙️

(Click to hide)



Fonte:
IEEE SPECTRUM



Rank	Language	Type	Score
1	Python	🌐💻⚙️	100.0
2	C	📱💻⚙️	96.0
3	Java	🌐📱💻	95.9
4	JavaScript	🌐	89.6
5	C++	📱💻⚙️	88.3
6	Go	🌐💻	87.3
7	R	💻	85.7
8	HTML	🌐	81.3
9	C#	🌐📱💻⚙️	79.8
10	SQL	💻	71.9

Choose a Ranking

IEEE Spectrum
Trending

Jobs
Open
Custom

Create Custom Ranking

Language Types

Web
🌐

Enterprise
💻

Mobile
📱

Embedded
⚙️

(Click to hide)



Fonte:

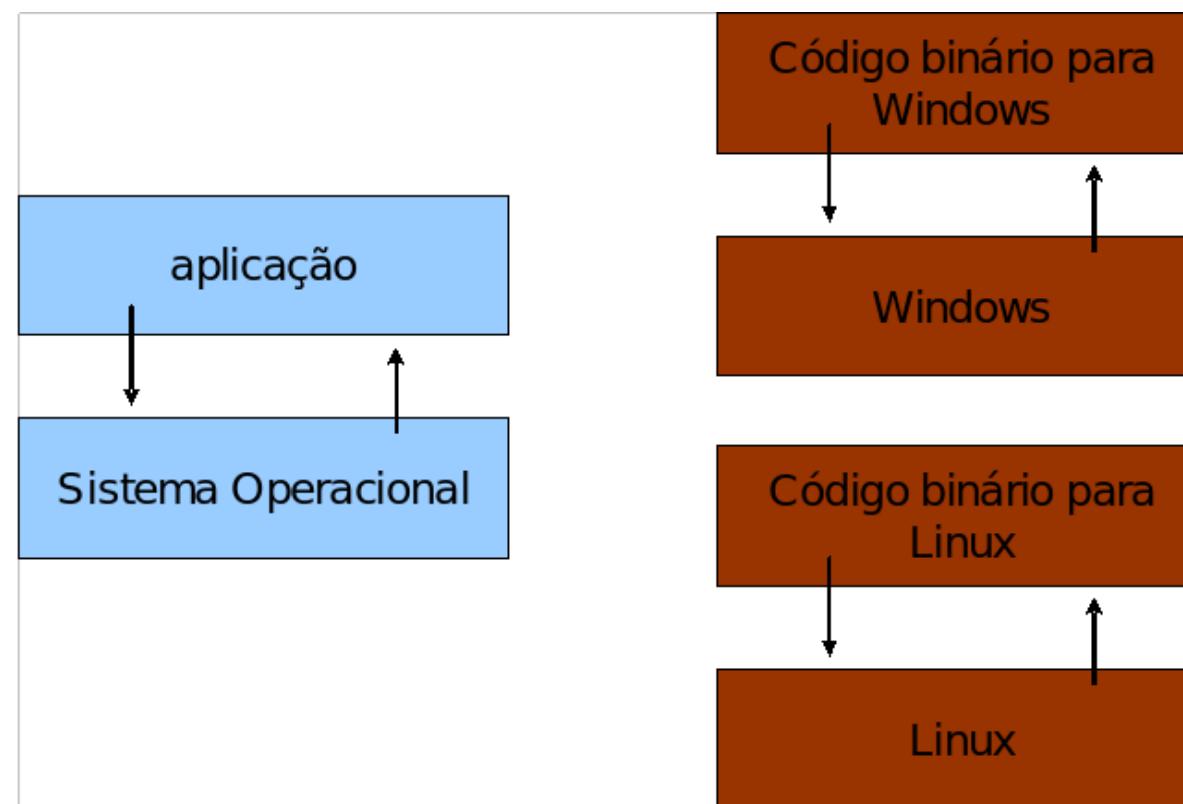


<https://spectrum.ieee.org/top-programming-languages/>



-

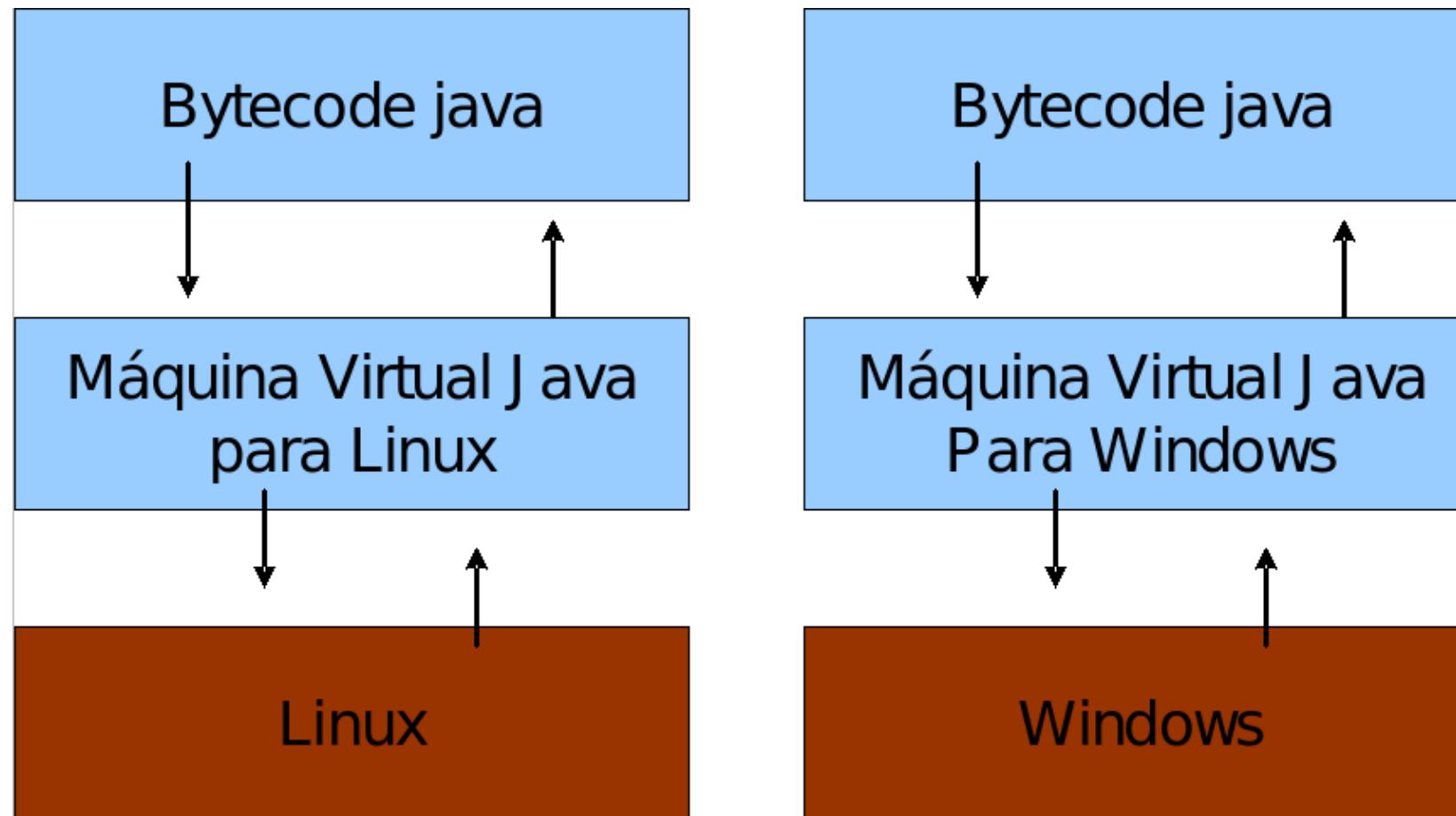
Maquina Virtual



-

Máquina Virtual

WRITE ONCE, RUN ANYWHERE





Como instalar o JDK

- [https://www.devmedia.com.br/installacao-e-configuracao-do-pacote-jdk/23749#:~:text=Para%20instalar%20o%20JDK%20no,32%20ou%2064%20bits\)%20utiliza](https://www.devmedia.com.br/installacao-e-configuracao-do-pacote-jdk/23749#:~:text=Para%20instalar%20o%20JDK%20no,32%20ou%2064%20bits)%20utiliza)
- https://docs.oracle.com/javase/8/docs/technotes/guides/install/install_overview.html
- <https://docs.oracle.com/en/java/javase/18/install/overview-jdk-installation.html#GUID-8677A77F-231A-40F7-98B9-1FD0B48C346A>





Java é lento?

- Hotspot: tecnologia que a JVM utiliza para detectar pontos quentes da sua aplicação.
 - Quando a JVM julgar necessário, ela vai compilar trechos de códigos para instruções realmente nativas da plataforma, para melhorar a performance da aplicação.
- Esse compilador é o JIT: Just inTime Compiler, que aparece bem na hora em que precisa.
- A JVM, compila dinamicamente durante a execução.
 - Identifica a performance do código e pode otimizar mais se necessário ou ainda mudar a estratégia de otimização.
 - Por isso as JVMs mais recentes, em alguns casos, chegam a ganhar de códigos C compilados com o GCC 3.x.





O foco da plataforma

Aplicações de médio a grande porte, em que o time de desenvolvedores tem várias pessoas e sempre pode vir a mudar e crescer



Criar projetos, mesmo com o auxílio de IDEs e ferramentas poderosas, será mais trabalhoso que muitas linguagens script ou de alta produtividade. Porém, seguindo as boas práticas e recomendações sobre design orientado a objetos, será mais fácil e rápido fazer alterações no sistema desde que você.





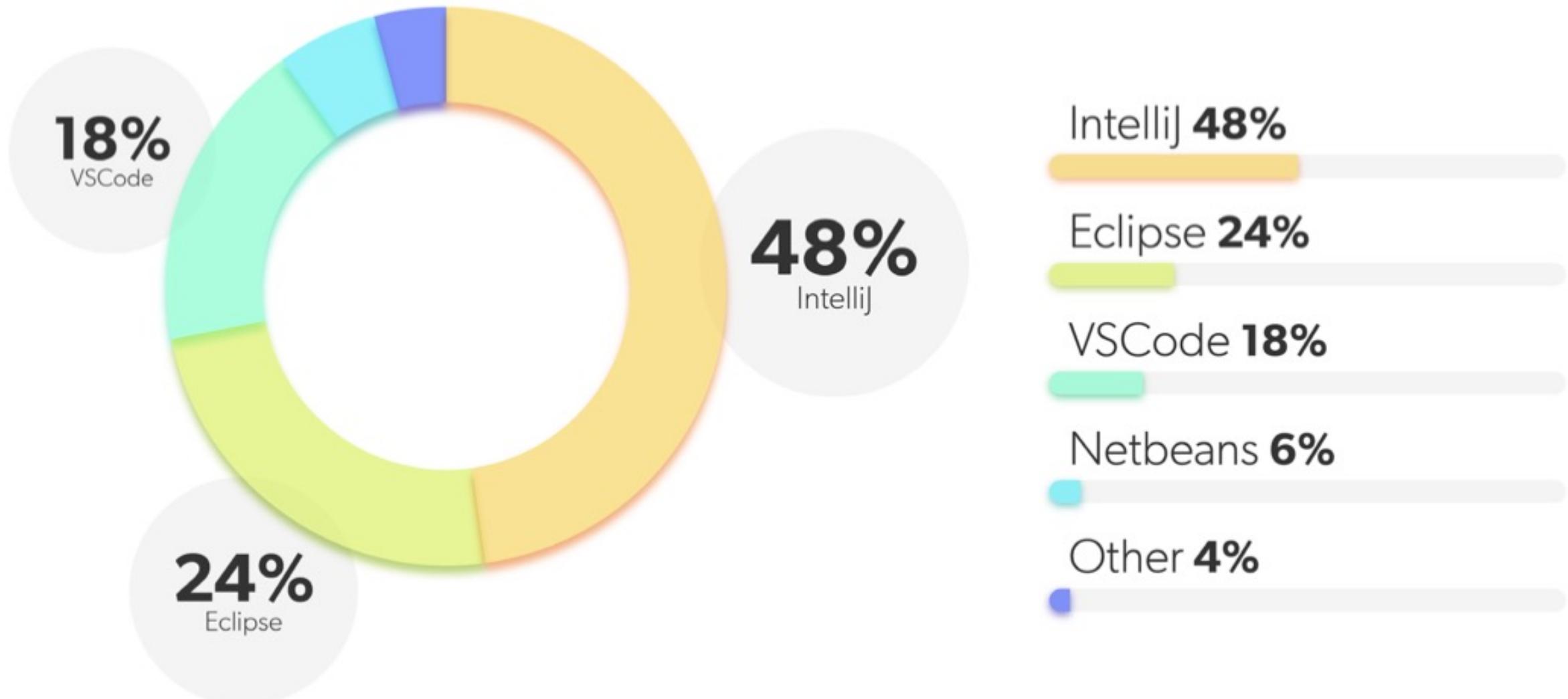
O ecossistema do Java é enorme

- Enorme quantidade de bibliotecas gratuitas
 - relatórios, os gráficos,
 - os sistemas de busca,
 - a geração de código de barra,
 - a manipulação de XML,
 - os tocadores de vídeo,
 - os manipuladores de texto,
 - a persistência transparente,
 - a impressão

É possível criar aplicações sofisticadas usando diversos recursos sem precisar pagar nada.



What Developer IDE Do You Use Professionally?



-

Meu Primeiro Programa

Crie um arquivo chamado **MeuPrimeiroPrograma.java**

Nossa aplicação irá imprimir um texto simples



-

Meu Primeiro Programa

O Java é **CASE SENSITIVE**

Cuidado com maiúsculas e minúsculas.

```
1 public class MeuPrimeiroPrograma {  
2     public static void main(String[] args) {  
3         System.out.println("Estou vivo!!!");  
4     }  
5 }
```

A linguagem é bastante burocrática



-

Meu Primeiro Programa

Toda aplicação Java começa por um ponto de entrada: o método **main**

```
1 public class MeuPrimeiroPrograma {  
2     public static void main(String[] args) {  
3         System.out.println("Estou vivo!!!");  
4     }  
5 }
```



-

Meu Primeiro Programa

Abra o terminal e digite:

```
javac MeuPrimeiroPrograma.java
```

Será gerado um arquivo `MeuPrimeiroPrograma.class` (bytecode)



-

Meu Primeiro Programa

Visualize o arquivo do bytecode

```
javap -c MeuPrimeiroPrograma
```



Bytecode

- O bytecode pode ser revertido para o .java original
 - Haverá perda de comentários e nomes de variáveis locais
- Para comercializar:
 - Use um ofuscador no código ele irá embaralhar classes, métodos e um monte de outros recursos
 - Exemplo
 - <http://proguard.sf.net>
 - <https://osbsoftware.com.br/produto/java-obfuscator>





Ofuscador

- Não impossibilita a engenharia reversa mas é dificulta
 - Torna o código fonte gerado pela engenharia reversa “menos” útil para hackers e concorrentes
- Não impede um atacante de desvendar como seu software funciona ou copiar seu código binário.





Ofuscador

- Não impossibilita a engenharia reversa mas é dificulta
 - Torna o código fonte gerado pela engenharia reversa “menos” útil para hackers e concorrentes
- Não impede um atacante de desvendar como seu software funciona ou copiar seu código binário.



-

Nome do arquivo Java

Experimente alterar o nome do arquivo para:

MinhaClasseMarota.java



-
-

Nome do arquivo Java

```
> javac MinhaClasseMarota.java
```

MinhaClasseMarota.java:5: error:

```
class MeuPrimeiroPrograma is public, should be declared in a file  
named MeuPrimeiroPrograma.java
```

```
public class MeuPrimeiroPrograma {
```

 ^

1 error



• Regras para criação dos arquivos e código pertencente ao arquivo

- Não é necessário ter uma classe declarada com o mesmo nome do arquivo.
- Cada arquivo .java deve conter apenas uma classe pública.
- Caso uma classe tenha o modificador de acesso public, ela deve ter o mesmo nome do arquivo .java.
- Podemos ter mais de uma classe declarada em um mesmo arquivo.



</>

Para testar

Experimente:

- Remover o modificador public da classe (compile e rode)

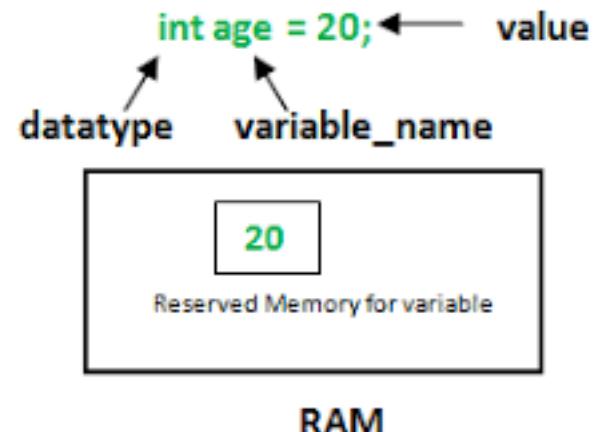
```
1 class MeuPrimeiroPrograma {  
2     public static void main(String[] args) {  
3         System.out.println("Estou vivo!!!");  
4     }  
5 }
```



-

Variáveis

Endereços de memória que tem um espaço ou tamanho definido de acordo com o tipo de dado que será guardado





Variáveis em Java

FORTEMENTE TIPADA

Toda variável tem um tipo que não pode ser alterado uma vez que é declarado

tipoDaVariavel nomeDaVariavel





Tipos de variáveis em Java

2 tipos de dados divididos por: **valor** e **referência**.

- **Variáveis primitivas:** boolean, byte, char, short, int, long, float e double
- **Variáveis de referência:** Tipos de Objetos
 - Strings,
 - Arrays Primitivos,
 - Objetos





Variáveis primitivas

Tipos de informação mais usuais e básicos.

- Boolean: Não é um valor numérico
 - Só admite os valores: `true` ou `false`
- Char: Usa o código UNICODE e ocupa cada caractere 16 bits
 - Guarda apenas um carácter
 - Deve estar entre aspas simples: 'a'
 - Vazio não é um caractere
 - Não guarda um código como ''





Variáveis primitivas

- Inteiros: diferem nas precisões e podem ser positivos ou negativos
 - Byte: 1 byte
 - Short: 2 bytes
 - Int: 4 bytes
 - Long: 8 bytes
- Ponto flutuante: diferem nas precisões e podem ser positivos ou negativos
 - Float: 4 bytes.
 - Double: 8 bytes.





Variáveis primitivas

TIPO	TAMANHO
boolean	1 bit
byte	1 byte
short	2 bytes
char	2 bytes
int	4 bytes
float	4 bytes
long	8 bytes
double	8 bytes



-

Variáveis primitivas

O valor de um tipo primitivo sempre é **copiado**

```
int i = 5; // i recebe uma cópia do valor 5;
int j = i; // j recebe uma cópia do valor de i;
i = i + 1; // i vira 6, j continua 5.
```



Casting

MEIO PELO QUAL UM VALOR É "**TRANSFORMADO**" DE UM
TIPO PARA OUTRO

Quando existe a necessidade de arredondar números de ponto flutuante para armazená-los em números inteiros, é preciso "**ORDENAR**" que o número quebrado seja **moldado (casted)** como um número inteiro para fazer que não ocorra erro de compilação.





Casting

PARA:	byte	short	char	int	long	float	double
DE:							
byte	----	<i>Impl.</i>	(char)	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>
short	(byte)	----	(char)	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>
char	(byte)	(short)	----	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>
int	(byte)	(short)	(char)	----	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>
long	(byte)	(short)	(char)	(int)	----	<i>Impl.</i>	<i>Impl.</i>
float	(byte)	(short)	(char)	(int)	(long)	----	<i>Impl.</i>
double	(byte)	(short)	(char)	(int)	(long)	(float)	----





Promoção

1. Se dois valores tiverem tipos de dados diferentes, será promovido automaticamente o menor tipo de dados para o maior.
2. Se um dos valores é inteiro e o outro é ponto flutuante, automaticamente será promovido o valor inteiro para o tipo de dados do valor de ponto flutuante.
3. Os tipos de dados menores, ou seja, *byte*, *short* e *char*, são promovidos primeiramente a qualquer momento para *int* no momento em que forem usados com um operador aritmético binário, mesmo que nenhum dos operandos seja *int*.
4. Após toda a promoção ter ocorrido e os operandos tiverem o mesmo tipo de dados, o valor resultante terá o mesmo tipo de dados da ultima promoção.



Discussão

Qual o tipo de *result*?

```
public class Promocao {  
    public static void main(String[] args) {  
        byte a = 10;  
        short b = 21;  
        int c = 32;  
        long d = 43L;  
        float e = 54.5F;  
        double f = 65.4D;  
        double result = (a * b / c * d / e * f);  
        System.out.println(result);  
    }  
}
```





Resposta

double

result = 309.5999959945679





Exercício

Fazer um programa em "Java" que receba o valor em graus Fahrenheit e imprime no vídeo o correspondente em graus Celsius usando as fórmulas:

- a) Usar uma variável *double* para ler o valor em Fahrenheit e a fórmula

$$C = (f-32.0) * (5.0/9.0)$$

- b) Usar uma variável *int* para ler o valor em Fahrenheit e a fórmula

$$C = (f-32)*(5/9)$$



-

Fazendo uma pausa nas Variáveis



Antes de entrar na orientação a objetos vamos aprender as estruturas de controle no Java



-

Estruturas condicionais: if ... else

IF é a estrutura condicional que executa a afirmação, dentro do bloco, se determinada condição for **verdadeira**. Se for **falsa**, executa as afirmações dentro de **ELSE**.

```
if (condicaoBooleana) {  
    codigo;  
}
```

Condição booleana é qualquer expressão que retorne true ou false





Condicionais aceitos

- MENOR QUE: $a < b$
- MENOR OU IGUAL: $a \leq b$
- MAIOR QUE: $a > b$
- MAIOR OU IGUAL: $a \geq b$
- IGUAL: $a == b$
- DIFERENTE DE: $a != b$



-

Estruturas condicionais: if else

Complementar ao **if/else** temos o operador **else if**, que possibilita adicionar uma nova condição à estrutura de decisão para atender a lógica sendo implementada.

```
if (condicaoBooleana1) {  
    // bloco de código 1  
} else if (condicaoBooleana2) {  
    // bloco de código 2  
} else {  
    // bloco de código 3  
}
```



-

Operador ternário

Similar ao do **if/else**, mas que é codificado em apenas uma linha.

```
(condiçãoBooleana) ? código 1 : código 2;
```

Ao avaliar a expressão booleana, caso ela seja verdadeira, o código 1, declarado após o ponto de interrogação (?) será executado; do contrário, o programa irá executar o código 2, declarado após os dois pontos (:).



-

Estruturas condicionais: switch

Possibilita diversos desvios, ou seja, de acordo com o resultado de uma condição pode-se executar um desvio entre os vários possíveis.



Estruturas condicionais: switch

```
char operacao = '+';
switch (operacao) {
    case '+':
        //Instruções;
        break;

    case '-':
        //Instruções;
        break;

    case '*':
        //Instruções;
        break;

    default
        //Instruções;
        break;
}
```

Tipos de valores aceitos: **char, byte, int, short ou um tipo enumerado**

Tipos enumerados não serão abordados agora.



-

Laços de Repetições: while

Usado quando não se sabe quantas vezes um determinado bloco de instruções precisa ser repetido

A execução das instruções continua até que uma condição seja verdadeira.

A **condição** a ser analisada para a execução do laço de repetição **deverá retornar um valor booleano**.





Laços de Repetições: while

Se a primeira verificação da condição for falsa, o programa simplesmente irá "pular" para a execução da próxima instrução após o laço

```
while (teste condicional){  
    //comandos; > serão executados enquanto o teste condicional for igual a verdadeiro  
}
```



-

Laços de Repetições: do...while

Semelhante ao while, porém os comandos dentro do bloco serão **executados ao menos uma vez**

```
do{  
    //comandos;  
} while (condicao);
```



-

Laços de Repetições: for

Usado **quando se sabe** quantas vezes é necessário repetir um bloco de instruções

```
for (valor inicial; teste booleano; incremento){  
    //Lógica  
}
```



-

Laços de Repetições: for

Na primeira parte da instrução for, podemos declarar e inicializar uma ou mais variáveis (separadas por vírgula)

```
for (int a = 0, b = 1, c = 2; a < 5; a++, b++, c++) {  
}
```

A inicialização acontece sempre antes dos outros comandos e apenas uma vez no laço de repetição





Para testar

Experimente: Esse código funciona?

```
public class TesteFor {  
    public static void main(String[] args) {  
        boolean condicao = false;  
        for (; !condicao;) {  
            System.out.println(condicao);  
            condicao = true;  
            System.out.println(condicao);  
        }  
    }  
}
```



-

Controle de loops: Break e Continue

Apesar das condições booleanas nos laços, em algum momento,

pode ser necessário parar o loop por algum motivo especial sem que
o resto do laço seja executado.





Controle de loops: Break

Usado em laços de repetição **while**, **do/while**, **for** e com os comandos **switch/case**.

```
public class TesteBreak {  
  
    public static void main(String[] args) {  
  
        for (int i = 1; i < 10; i++) {  
            if (i == 5) {  
                break;  
            }  
            System.out.println("Valor de i = " + i);  
        }  
    }  
}
```

Quando usado em laço de repetição, causa uma interrupção imediata do mesmo, continuando a execução do programa na próxima linha após o laço. Isso ocorre caso a condição imposta seja atendida.





Controle de loops: Continue

Usado **somente em laços de repetição**. Quando executado, o laço volta imediatamente para o teste de condição do laço de repetição.

```
public class TesteContinue {  
  
    public static void main(String[] args) {  
  
        int i = 1;  
  
        while (i < 5){  
            if (i == 4){  
                continue;  
            }  
            System.out.println("Valor de i = " + i);  
        }  
  
    }  
  
}
```

Normalmente usado em
conjunto com o
condicional if.



-

Escopo de variáveis

NOME DADO AO TRECHO DE CÓDIGO EM QUE AQUELA VARIÁVEL EXISTE E O LUGAR ONDE É POSSÍVEL ACESSÁ-LA

É possível declarar variáveis a qualquer momento

As variáveis declaradas num bloco só valem dentro dele.



-

Escopo de variáveis

```
// aqui, a variável i não existe.  
int i = 5;  
// a partir daqui, ela existe.  
while (condicao) {  
    // o i ainda vale aqui.  
    int j = 7;  
    // o j passa a existir.  
}  
// aqui, o j não existe mais, porém o i continua dentro do escopo.
```



</>

Exercício

A Sequência de Fibonacci tem como primeiros termos os números 0 e 1 e, a seguir, cada termo subsequente é obtido pela soma dos dois termos predecessores. Nenhuma outra sequência de números foi tão estudada e possui aplicações em áreas tão distintas, como por exemplo Biologia, Arquitetura, Arte e outras. A razão entre dois números consecutivos da sequência converge para um valor constante de 1,618... conhecido como número de ouro.

- a) Imprima os 30 primeiros números da sequência de Fibonacci
- b) Imprima os primeiros números da série de Fibonacci menores que 100

Regra:

$$F_n = F_{n - 1} + F_{n - 2}$$





Mas e a matéria?

- Programação Orientada a Objetos
- UML
- Boas práticas

Composição da nota:

20% (atividades e questionários) + 60% trabalho + 20% prova final





Próxima aula

Para aprender mais:

🔗 <https://loiane.training/curs.../java-basico>

🔗 <https://www.caelum.com.br/a.../apostila-jav...-objetos/>

🔗 <https://www.alura.com.br/.../artigos/poo-...-orientada-a-...-objetos>

✓ Atributos e métodos

✓ Java



Bibliografia

BARNES, David J.; KOLLING, Michael. **Programação orientada a objetos com Java.** 4. ed. São Paulo: Prentice Hall - Br, 2009.

BOENTE, Alfredo. **Aprendendo a programar em C++: usando classes e objetos.** Rio de Janeiro: Brasport, 2004. ISBN 9788574521749.

CORREIA, Carlos Henrique; TAFNER, Malcon Anderson. **Análise orientada a objetos.** 2. ed. Florianópolis: Visual Books, 2006.

DEITEL, Paul; DEITEL, Harvey. **Java: como programar.** 10. ed. São Paulo: Pearson, 2017. ISBN 9788543004792. Disponível em: <https://ifsp.bv3.digitalpages.com.br/users/publications/9788543004792>. Acesso em: 14 jun. 2019.

Sierra, Kathy. **Use A Cabeça Java.** S.l: Alta Books, 2009. ISBN: 8576081733





**INSTITUTO
FEDERAL**

São Paulo

Câmpus
São Paulo

Obrigado!

Gustavo Fortunato Puga

gustavo.puga@ifsp.edu.br

