

As próximas cinco linguagens para você aprender... e por quê?

Fernando Castor

Centro de Informática – Universidade Federal de Pernambuco

Alguns direitos reservados



Linguagens de Programação

- Servem para **dizer ao computador o que fazer**
- **Independentemente da máquina** subjacente
- **Legíveis** para humanos



Linguagens de Programação

- Servem para **dizer ao computador o que fazer**
- **Independentemente da máquina** subjacente
- **Legíveis** para humanos

Hello World!

```
C      Hello World in Fortran 77

      PROGRAM HELLO
      PRINT*, 'Hello, World!'

      END
```



Linguagens de Programação são **Ferramentas**

Cada uma se presta a um fim

- **Fortran**: processamento numérico
- **Cobol**: descrição de dados e aplicações comerciais
- **Perl**: processamento de cadeias de caracteres



Linguagens de Programação são **Ferramentas**

Cada uma se presta a um fim

- **Fortran:** processamento numérico
- **Cobol:** descrição de dados e aplicações comerciais
- **Perl:** processamento de cadeias de caracteres



Esse fim **pode mudar** com o tempo

Exemplo: **Java**

- 1 Originalmente: projetada para televisão interativa
- 2 No lançamento/primeiros anos: a linguagem da Web
- 3 Hoje: servidores e aplicações distribuídas

Certo, são muitas linguagens, mas...

- 1 Quais são as linguagens de programação mais usadas na prática?
- 2 Se você quisesse aprender cinco linguagens para o futuro, quais deveriam ser?



Parte 1

Quais são as linguagens de programação mais usadas na prática?



Não há um censo oficial sobre uso de linguagens

- Resposta precisa vir de várias fontes
- Com diferentes contextos
- Dados são **difíceis de comparar**
- C# no **CodePlex** vs. C# no **sourceforge**

SORTED BY TAG	
.NET	(1186)
.NET 2.0	(404)
.NET 3.5	(499)
.NET 4.0	(413)
ASP.NET	(831)
ASP.NET MVC	(339)
C#	(2476)
DotNetNuke	(276)

Recently updated x	
Programming Language	Status
Java	(10,384)
C++	(9,260)
C	(6,445)
PHP	(4,817)
Python	(3,813)
C#	(3,300)
JavaScript	(2,713)

O que significam “ser usada” e “na prática”?

- Mais **linhas de código** escritas na linguagem?



O que significam “ser usada” e “na prática”?

- Mais **linhas de código** escritas na linguagem?
- Número de **commits** de programas que usam a linguagem?



O que significam “ser usada” e “na prática”?

- Mais **linhas de código** escritas na linguagem?
- Número de **commits** de programas que usam a linguagem?
- Mais **“projetos”** com linhas de código escritas na linguagem?



O que significam “ser usada” e “na prática”?

- Mais **linhas de código** escritas na linguagem?
- Número de **commits** de programas que usam a linguagem?
- Mais **“projetos”** com linhas de código escritas na linguagem?
 - O que é um “projeto”?
 - **Forks** contam como projetos separados?
 - Se usar apenas **código de terceiros** conta?



O que significam “ser usada” e “na prática”?

- Mais **linhas de código** escritas na linguagem?
- Número de **commits** de programas que usam a linguagem?
- Mais **“projetos”** com linhas de código escritas na linguagem?
 - O que é um “projeto”?
 - **Forks** contam como projetos separados?
 - Se usar apenas **código de terceiros** conta?
- Onde a **linha** da “prática” é **traçada**?



O que significam “ser usada” e “na prática”?

- Mais **linhas de código** escritas na linguagem?
- Número de **commits** de programas que usam a linguagem?
- Mais **“projetos”** com linhas de código escritas na linguagem?
 - O que é um “projeto”?
 - **Forks** contam como projetos separados?
 - Se usar apenas **código de terceiros** conta?
- Onde a **linha** da “prática” é **traçada**?
- É relevante saber se desenvolvedores **falam** muito **sobre**?



Em resumo:
é necessário adotar
diferentes perspectivas.



Felizmente...

- Alguém já fez **parte** do trabalho
- TIOBE Programming Community Index

... is an indicator of the popularity of programming languages. The index is updated once a month. The ratings are based on the number of skilled engineers world-wide, courses and third party vendors. The popular search engines Google, Bing, Yahoo!, Wikipedia, Amazon, YouTube and Baidu are used to calculate the ratings.



De acordo com a TIOBE, o ranking em julho/2012 era este:

Position Jul 2012	Position Jul 2011	Delta in Position	Programming Language	Ratings Jul 2012	Delta Jul 2011	Status
1	2	↑	C	18.331%	+1.05%	A
2	1	↓	Java	16.087%	-3.16%	A
3	6	↑↑↑	Objective-C	9.335%	+4.15%	A
4	3	↓	C++	9.118%	+0.10%	A
5	4	↓	C#	6.668%	+0.45%	A
6	7	↑	(Visual) Basic	5.695%	+0.59%	A
7	5	↓↓	PHP	5.012%	-1.17%	A
8	8	=	Python	4.000%	+0.42%	A
9	9	=	Perl	2.053%	-0.28%	A
10	12	↑↑	Ruby	1.768%	+0.44%	A
11	10	↓	JavaScript	1.454%	-0.79%	A
12	14	↑↑	Delphi/Object Pascal	1.157%	+0.27%	A
13	13	=	Lisp	0.997%	+0.09%	A
14	15	↑	Transact-SQL	0.954%	+0.15%	A
15	25	↑↑↑↑↑↑↑↑	Visual Basic .NET	0.917%	+0.43%	A
16	16	=	Pascal	0.837%	+0.17%	A
17	19	↑↑	Ada	0.689%	+0.14%	B
18	11	↓↓↓↓↓	Lua	0.684%	-0.89%	B
19	21	↑↑	PL/SQL	0.645%	+0.10%	A--
20	26	↑↑↑↑↑	MATLAB	0.639%	+0.19%	B

De acordo com a TIOBE, o ranking em julho/2012 era este:

Position Jul 2012	Position Jul 2011	Delta in Position	Programming Language	Ratings Jul 2012	Delta Jul 2011	Status
1	2	↑	C	18.331%	+1.05%	A
2	1	↓	Java	16.087%	-3.16%	A
3	6	↑↑↑	Objective-C	9.335%	+4.15%	A
4	3	↓	C++	9.118%	+0.10%	A
5	4	↓	C#	6.668%	+0.45%	A
6	7	↑	(Visual) Basic	5.695%	+0.59%	A
7	5	↓↓	PHP	5.012%	-1.17%	A
8	8	=	Python	4.000%	+0.42%	A
9	9	=	Perl	2.053%	-0.28%	A
10	12	↑↑	Ruby	1.768%	+0.44%	A
11	10	↓	JavaScript	1.454%	-0.79%	A
12	14	↑↑	Delphi/Object Pascal	1.157%	+0.27%	A
13	13	=	Lisp	0.997%	+0.09%	A
14	15	↑	Transact-SQL	0.954%	+0.15%	A
15	25	↑↑↑↑↑↑↑↑	Visual Basic .NET	0.917%	+0.43%	A
16	16	=	Pascal	0.837%	+0.17%	A
17	19	↑↑	Ada	0.689%	+0.14%	B
18	11	↓↓↓↓↓	Lua	0.684%	-0.89%	B
19	21	↑↑	PL/SQL	0.645%	+0.10%	A-
20	26	↑↑↑↑↑	MATLAB	0.639%	+0.19%	B

Relembrando:

*... is an indicator of the popularity of programming languages. The index is updated once a month. The ratings are based on the **number of skilled engineers** world-wide, **courses** and **third party vendors**. The **popular search engines** Google, Bing, Yahoo!, Wikipedia, Amazon, YouTube and Baidu are used to calculate the ratings.*



Relembrando:

... is an indicator of the popularity of programming languages. The index is updated once a month. The ratings are based on the **number of skilled engineers** world-wide, **courses** and **third party vendors**. The **popular search engines** Google, Bing, Yahoo!, Wikipedia, Amazon, YouTube and Baidu are used to calculate the ratings.

Problema do ranking do índice da TIOBE:
não menciona diretamente **código** escrito nas linguagens ou **número de repositórios ou projetos** que as usam.

Obtendo informações sobre código

- Repositórios de software de código aberto (*forges*) são um bom lugar para procurar
 - Os maiores incluem dezenas de milhares de projetos
 - E já fornecem algumas estatísticas

Examinaremos os dados de dois desses repositórios:



Dados do SourceForge

- 1 Java (10.386)
- 2 C++ (9.261)
- 3 C (6.446)
- 4 PHP (4.817)
- 5 Python (3.814)
- 6 C# (3.300)
- 7 JavaScript (2.713)
- 8 Perl (1.514)
- 9 Unix Shell (985)
- 10 Visual Basic .NET (811)
- 11 Delphi/Kylix (742)
- 12 Assembly (480)
- 13 Visual Basic (435)
- 14 ActionScript (357)
- 15 Lua (339)

Entre parênteses: número de projetos que declaram usar a linguagem



Dados do SourceForge

- 1 Java (10.386)
- 2 C++ (9.261)
- 3 C (6.446)
- 4 PHP (4.817)
- 5 Python (3.814)
- 6 C# (3.300)
- 7 JavaScript (2.713)
- 8 Perl (1.514)
- 9 **Unix Shell** (985)
- 10 Visual Basic .NET (811)
- 11 Delphi/Kylix (742)
- 12 **Assembly** (480)
- 13 Visual Basic (435)
- 14 **ActionScript** (357)
- 15 Lua (339)

Entre parênteses: número de projetos que declaram usar a linguagem

Assembly, ActionScript e UNIX Shell não aparecem no índice da TIOBE



Dados do SourceForge

- 1 Java (10.386)
- 2 C++ (9.261)
- 3 C (6.446)
- 4 PHP (4.817)
- 5 Python (3.814)
- 6 C# (3.300)
- 7 JavaScript (2,713)
- 8 Perl (1.514)
- 9 **Unix Shell** (985)
- 10 Visual Basic .NET (811)
- 11 Delphi/Kylix (742)
- 12 **Assembly** (480)
- 13 Visual Basic (435)
- 14 **ActionScript** (357)
- 15 Lua (339)

Entre parênteses: número de projetos que declaram usar a linguagem

Assembly, **ActionScript** e **UNIX** Shell não aparecem no índice da TIOBE

Ruby e **Objective-C** aparecem no Top 10 da TIOBE mas não aqui



Dados do Github

Com base em número de repositórios únicos:

- 1 JavaScript (20%)
- 2 Ruby (14%)
- 3 Python (9%)
- 4 Shell (8%)
- 5 Java (8%)
- 6 PHP (7%)
- 7 C (7%)
- 8 C++ (4%)
- 9 Perl (4%)
- 10 Objective-C (3%)

Entre parênteses: percentagem
do total de repositórios



Dados do Github

Com base em número de repositórios únicos:

- 1 JavaScript (20%) – 7
- 2 Ruby (14%) – não aparece
- 3 Python (9%) – 5
- 4 Shell (8%) – 9
- 5 Java (8%) – 1
- 6 PHP (7%) – 4
- 7 C (7%) – 3
- 8 C++ (4%) – 2
- 9 Perl (4%) – 8
- 10 Objective-C (3%) – não aparece

Entre parênteses: percentagem do total de repositórios

Comparando-se as posições das linguagens no Github e no SourceForge.



Dados do Github

Com base em número de repositórios únicos:

- 1 JavaScript (20%) – 7
- 2 Ruby (14%) – não aparece
- 3 Python (9%) – 5
- 4 Shell (8%) – 9
- 5 Java (8%) – 1
- 6 PHP (7%) – 4
- 7 C (7%) – 3
- 8 C++ (4%) – 2
- 9 Perl (4%) – 8
- 10 Objective-C (3%) – não aparece

Entre parênteses: percentagem do total de repositórios

Comparando-se as posições das linguagens no Github e no SourceForge.

Apenas **Shell** não aparece no índice da TIOBE.



E sobre quais os programadores estão falando?

A partir das tags do
StackOverflow.com:

- 1 C# (331964)
- 2 Java (275849)
- 3 PHP (255372)
- 4 JavaScript (239977)
- 5 C++ (138684)
- 6 Python (121864)
- 7 Objective-C (94303)
- 8 SQL (88972)
- 9 C (65463)
- 10 Ruby (50757)
- 11 VB.NET (33043)

Entre parênteses: número de
posts com aquela tag



E sobre quais os programadores estão falando?

A partir das tags do
StackOverflow.com:

- 1 C# (331964)
- 2 Java (275849)
- 3 PHP (255372)
- 4 JavaScript (239977)
- 5 C++ (138684)
- 6 Python (121864)
- 7 Objective-C (94303)
- 8 SQL (88972)
- 9 C (65463)
- 10 Ruby (50757)
- 11 VB.NET (33043)

Entre parênteses: número de
posts com aquela tag

Se levarmos JQuery, Django e
Rails em conta, os números
mudam:

- 1 JavaScript (**445576**)
- 5 Ruby (**162223**)
- 6 Python (**157798**)



Complementarmente: Ohloh.net

<http://www.ohloh.net/languages/compare>

Permite comparação em termos de diferentes
critérios



Top 10 a partir de diversas fontes

Github	SourceForge	TIOBE	stackoverflow	Ohloh (commits)
JavaScript	Java	C	JavaScript	Java
Ruby	C++	Java	C#	C
Python	C	Objective-C	Java	C++
Shell	PHP	C++	PHP	Python
Java	Python	C#	Ruby	JavaScript
PHP	C#	(Visual) Basic	Python	PHP
C	JavaScript	PHP	C++	Ruby
C++	Perl	Python	Objective-C	Shell
Perl	Visual Basic	Perl	SQL	C#
Objective-C	Shell	Ruby	C	Perl

Top 10 a partir de diversas fontes

Github	SourceForge	TIOBE	stackoverflow	Ohloh (commits)
JavaScript	Java	C	JavaScript	Java
Ruby	C++	Java	C#	C
Python	C	Objective-C	Java	C++
Shell	PHP	C++	PHP	Python
Java	Python	C#	Ruby	JavaScript
PHP	C#	(Visual) Basic	Python	PHP
C	JavaScript	PHP	C++	Ruby
C++	Perl	Python	Objective-C	Shell
Perl	Visual Basic	Perl	SQL	C#
Objective-C	Shell	Ruby	C	Perl

Importante: não representam a indústria **como um todo**.

Parte 2

Se você quisesse aprender cinco linguagens para o futuro, quais deveriam ser?



Exercício mental: pense numa
lista com cinco linguagens com
resposta à pergunta



Exercício mental: pense numa lista com cinco linguagens com resposta à pergunta

Que critérios lhe levaram a pensar nessa lista?



Meus critérios

Nada de linguagens “acadêmicas”

Não devem servir apenas como prova de conceito para uma ideia



Meus critérios

Nada de linguagens “acadêmicas”

Não devem servir apenas como prova de conceito para uma ideia

Linguagens que forneçam **lições amplas**

- linguagens que se possa empregar diretamente
- e linguagens que ajudem a usar melhor outras linguagens

Meus critérios

Nada de linguagens “acadêmicas”

Não devem servir apenas como prova de conceito para uma ideia

Linguagens que forneçam **lições amplas**

- linguagens que se possa empregar diretamente
- e linguagens que ajudem a usar melhor outras linguagens

Linguagens de programação para você **aprender**



Meus critérios

Levando em conta certos nichos de aplicação e cenários de uso

- 1 Primeira linguagem de programação para aprender
- 2 Linguagem Pós-Java
- 3 Programação multi-núcleo e sistemas confiáveis
- 4 Desenvolvimento para a Web
- 5 Software de baixo nível e de alto desempenho

1. Primeira Linguagem de Programação para Aprender



Olá, Mundo!

```
class HelloWorld {  
    static public void main( String args[] ) {  
        System.out.println(" Hello World!");  
    }  
}
```

```
#include <stdio.h>
```

```
int main(void) {  
    puts(" Hello World!");  
}
```

```
10 PRINT " Hello World!"
```

```
(defun helloworld ()  
  (print " Hello World!")  
)
```

```
print " Hello World!"
```

```
package main  
import "fmt"  
func main() {  
    fmt.Printf(" Hello World!")  
}
```

Linguagem 1



Linguagem 1





Porquês

- Sintaxe amigável
- Dinamicamente tipificada
- Multiparadigma
- Muito popular na prática
- Usada como primeira linguagem em vários cursos de computação

Alternativas



2. Linguagem Pós-Java

Java é Pop

Github	SourceForge	TIOBE	stackoverflow	Ohloh (commits)
JavaScript	Java	C	JavaScript	Java
Ruby	C++	Java	C#	C
Python	C	Objective-C	Java	C++
Shell	PHP	C++	PHP	Python
Java	Python	C#	Ruby	JavaScript
PHP	C#	(Visual) Basic	Python	PHP
C	JavaScript	PHP	C++	Ruby
C++	Perl	Python	Objective-C	Shell
Perl	Visual Basic	Perl	SQL	C#
Objective-C	Shell	Ruby	C	Perl

- Mais de 10.000 projetos no SourceForge
- Busca por “Java” no Github: 37.000+ repositórios
- Quase 240.000 tags no StackOverflow.com
- Linguagem com mais commits em 2012 (Ohloh.net)

Mas envelheceu mal!



Mas envelheceu mal!



Males da idade



Males da idade



Evolução burocrática

- 4,5 anos entre as versões 6 e 7
- E as coisas mais legais ficaram para a v8.0.

Males da idade



Evolução burocrática

- 4,5 anos entre as versões 6 e 7
- E as coisas mais legais ficaram para a v8.0.

Além disso...

- Java ficou mais **complexa**

Males da idade



Evolução burocrática

- 4,5 anos entre as versões 6 e 7
- E as coisas mais legais ficaram para a v8.0.

Além disso...

- Java ficou mais **complexa**
Isso se reflete em sua especificação:
 - 1a ed.: 539 págs.
 - 2a ed.: 544 págs.
 - 3a ed.: 688 págs.
 - 4a ed.: **928 págs.!**
- Mas ainda não oferece funcionalidades requisitadas por muitos!

Prós de Java:

- Existe código Java para fazer praticamente qualquer coisa
- A JVM é uma VM sofisticada
 - E há implementações dela para diversas arquiteturas e SOs

Contras de Java:

- A linguagem



Em poucas palavras:

Abandona-se a
linguagem Java mas
mantém-se a plataforma Java.





Centro
de Informática
UFPB



The End of an Era: **Scala** Community Arrives, **Java** Deprecated

Posted by [Ryan Slobojan](#) on Apr 01, 2010

Sections [Enterprise Architecture](#), [Operations & Infrastructure](#), [Process & Practices](#), [Architecture & Design](#), [Development](#) Topics [Scala](#), [Ruby](#), [JVM Languages](#), [Functional Programming](#), [FEATURED: Java](#), [Dynamic Languages](#), [Leadership](#), [InfoQ Announcements](#), [Architecture](#), [Change](#), [migration](#), [Legacy Code](#), [FEATURED: Agile](#), [InfoQ](#), [Careers](#)

Share     

 Bookmark this!

It was recently announced that [InfoQ is creating a new Operations community](#), which will be our seventh community. In addition to that, another major change which has been in the works for the last few months at InfoQ is the conversion of the Java community to the [Scala community](#). InfoQ spoke with a prominent Scala expert and members of the former InfoQ Java editorial team to learn more about this change and why it was made.

Describing this transition, [Charles Humble](#), lead editor of the Scala community, said:

As the lead Java editor for InfoQ the decision to drop coverage of Java and concentrate instead on Scala is not one we took lightly. In 1995 Java represented a paradigm shift. By taking ideas that worked well in other languages such as C, C++, Cedar/Mesa, Modula and Simula, and adding built-in distributed programming support, it moved both OO and distributed programming into the mainstream of enterprise computing. Today a similar paradigm shift is occurring as a consequence of the rise of multi-core machines, and it is functional language based. Scala is the Java of functional programming, [described by Neal Gafter](#) as "a strong contender for Java++." For modern enterprise computing Scala represents a much more logical choice than Java, and we expect it to become the dominant language on the JVM this year. We've recently thrown away InfoQ's existing Java code base, re-writing it in Scala, and we expect many of our readers are engaged in similar activities. In view of this we've decided to concentrate our efforts on our Scala coverage, and to drop coverage for the Java language, and related legacy topics such as Java EE and Spring from these pages.

Dean Wampler, Ph.D., the co-author of O'Reilly's "Programming Scala", offered this comment on the sudden industry switch to Scala vs. the less appealing alternatives:

We all know that object-oriented programming is dead and buried. Scala gives you a 'grace period'; you can use its deprecated support for objects until you've ported your code to use [Monads](#).



Porquês

- Compatibilidade retroativa com Java: **roda na JVM**
- Menos verborrágica
 - Inclui **inferência de tipos!**
- Imperativa, OO, funcional e concorrente.



Porquês

- Compatibilidade retroativa com Java: **roda na JVM**
- Menos verborrágica
 - Inclui **inferência de tipos!**
- Imperativa, OO, funcional e concorrente.
- Principais funcionalidades de Java 8 **já estão em Scala**
 - Desde as primeiras versões!
 - E Scala implementa outras que Java nem vislumbra
- Sony, Twitter, LinkedIn e Siemens, entre outras, já usam!

Alternativas



Clojure



3. Programação Multi-Núcleo e Sistemas Confiáveis



<http://is1.minnesotacomputers.net/images/productimages/M605.gif>

http://www.ps3-systems.com/images/playstation_3.jpg

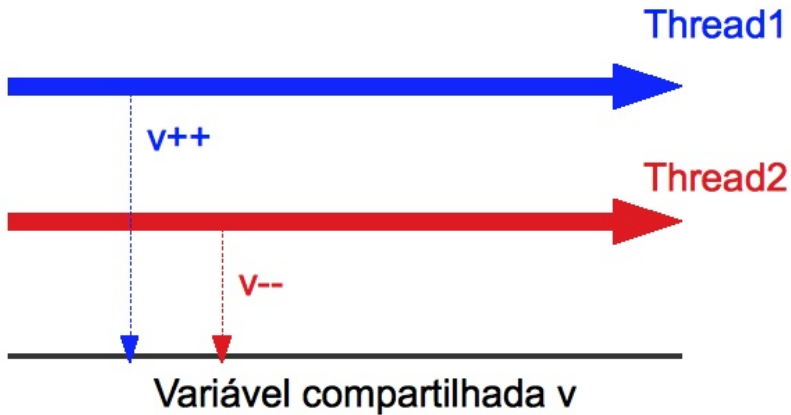
<http://www.zeldauniverse.net/wp-content/uploads/2010/07/oot3dsblack.png>

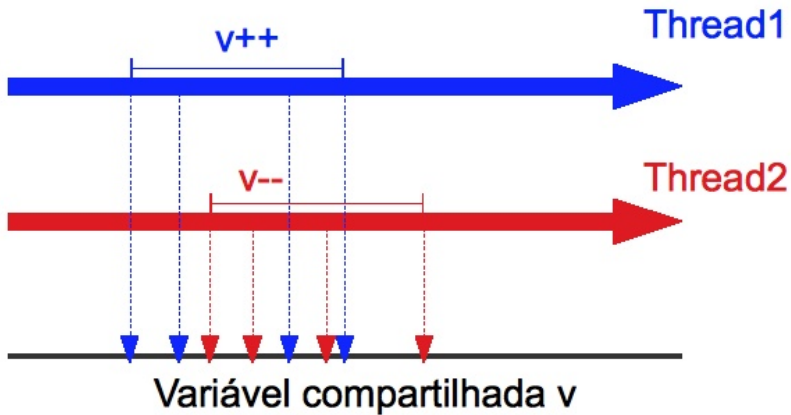
http://www.tudocelular.com/new_files/images/global/Samsung-Galaxy-Note_53032_1.jpg

<http://maxcdn.liewcf.com/blog/wp-content/uploads/apple-ipad-2-wifi.jpg>

Multi-núcleo \Rightarrow
concorrência e paralelismo







Problemas oriundos de programação multi-núcleo

- Condições de corrida
- Falhas de atomicidade
- Deadlocks, livelocks, starvation
- Morte prematura de threads
- Overhead de criação e uso de threads



Três causas fundamentais

- **Estado compartilhado mutável** é a raiz de quase todos os males.
- **Falta** de mecanismos para **tratamento de erros** em cenários concorrentes e distribuídos.
- Emprego de **threads do sistema operacional** como unidades de quebra de trabalho.

Linguagem 3



Linguagem 3





Porquês

- **Filosofia Let It Crash**
- Puramente funcional e sem estado compartilhado
- Processos leves – unidades finas de quebra de trabalho
- Concorrência e distribuição são **inerentes** à linguagem
 - É impossível construir sistemas não-triviais ignorando isso

Nenhuma.

4. Desenvolvimento para a Web

Qual o atual estado da prática de desenvolvimento para a Web?

(de acordo com a Wikipédia)



Lado servidor: ASP, CSP, Server-Side ANSI C, ColdFusion, CGI, Groovy, Grails, Java, Lotus Domino, Perl, PHP, Python (Django), Real Studio Web Edition, Ruby (Rails), Server-Side JavaScript, Mozilla Rhino, Websphere, .NET, (...)

Lado cliente: Ajax, Flash, JavaScript, jQuery, Silverlight, HTML5



JavaScript é um padrão de fato, mas...

- $[] + [] =$



JavaScript é um padrão de fato, mas...

- `[] + [] = string vazio!`
- `[] + {} =`



JavaScript é um padrão de fato, mas...

- `[] + [] = string vazio!`
- `[] + {} = [object Object]?!`
- `{ } + [] =`



JavaScript é um padrão de fato, mas...

- $[] + [] = \text{string vazio!}$
- $[] + \{\} = [\text{object Object}]?!$
- $\{\} + [] = 0!!!$
- $\{\} + \{\} =$

JavaScript é um padrão de fato, mas...

- $[] + [] = \text{string vazio!}$
- $[] + \{\} = [\text{object Object}]?!$
- $\{\} + [] = 0!!!$
- $\{\} + \{\} = \text{NaN!}$
- $\{\} - \{\} =$

JavaScript é um padrão de fato, mas...

- $[] + [] = \text{string vazio!}$
- $[] + \{\} = [\text{object Object}]?!$
- $\{\} + [] = 0!!!$
- $\{\} + \{\} = \text{NaN!}$
- $\{\} - \{\} = \text{NaN!}$
- $\{\} * \{\} =$

JavaScript é um padrão de fato, mas...

- $[] + [] = \text{string vazio!}$
- $[] + \{\} = [\text{object Object}]?!$
- $\{\} + [] = 0!!!$
- $\{\} + \{\} = \text{NaN!}$
- $\{\} - \{\} = \text{NaN!}$
- $\{\} * \{\} = \text{syntax error!}$
- $\text{Array}(16) + 5 =$

JavaScript é um padrão de fato, mas...

- $[] + [] = \text{string vazio!}$
- $[] + \{\} = [\text{object Object}]?!$
- $\{\} + [] = 0!!!$
- $\{\} + \{\} = \text{NaN!}$
- $\{\} - \{\} = \text{NaN!}$
- $\{\} * \{\} = \text{syntax error!}$
- $\text{Array}(16) + 5 = \text{,,,,,,,,,,,,,5}$
- $\text{Array}(16) - 5 =$

JavaScript é um padrão de fato, mas...

- $[] + [] = \text{string vazio!}$
- $[] + \{\} = [\text{object Object}]?!$
- $\{\} + [] = 0!!!$
- $\{\} + \{\} = \text{NaN!}$
- $\{\} - \{\} = \text{NaN!}$
- $\{\} * \{\} = \text{syntax error!}$
- $\text{Array}(16) + 5 = \text{,,,,,,,,,,,,,5}$
- $\text{Array}(16) - 5 = \text{NaN!}$

E o escopo de variáveis?

“Javascript programmers are practically ranked by how well they understand scope.”

<http://stackoverflow.com/questions/500431/javascript-variable-scope>

- Cuidado com o significado de **this**!



Linguagem 4







DART

Porquês

- Menos caos
 - sem protótipos, objeto global ou atributos adicionados dinamicamente
- Possibilidade de usar **anotações de tipo**
- Escopos intuitivos, baseados em blocos
- Compilável para JavaScript



5. Software de Baixo Nível e de Alto Desempenho



Que linguagem você acha que seria boa para construir um driver de dispositivo?



E um escalonador de processos?



E um *shooter* 3D?



Um programa para processar imagens de satélites?



Sendo um pouco mais focado

Tem algo que nenhuma linguagem discutida anteriormente lhe dá
Controle sobre a máquina



Sendo um pouco mais focado

Tem algo que nenhuma linguagem discutida anteriormente lhe dá
Controle sobre a máquina

Que linguagem usar se eu quiser...

- ter acesso direto à memória RAM
 - inclusive para gerenciar memória manualmente
- guardar valores de variáveis explicitamente em registradores
 - em alguns casos, especificando o registrador
- produzir código extremamente eficiente
- e fazer isso tudo podendo de forma portátil?

Linguagem 5



Linguagem 5



Linguagem 5



THE

C

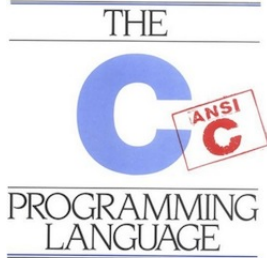


PROGRAMMING
LANGUAGE

BRIAN W. KERNIGHAN
DENNIS M. RITCHIE



Linguagem 5



BRIAN W. KERNIGHAN
DENNIS M. RITCHIE

Porquês

- Porque permite **tudo mencionado no slide anterior**
- Porque é muito **menos complexa** que C++
- Porque Assembly só é uma opção quando é a **única opção**



Mas há outras razões para aprender C

Outros porquês

- Gerenciamento de memória manual
- Tipificação fraca
- Manipulação de ponteiros é
 - fundamental mesmo para as operações mais básicas
 - complexa
 - insegura
- ausência de suporte decente a programação paralela

Resumindo: é importante aprender C porque:



Resumindo: é importante aprender C porque:

- É uma linguagem muito usada na prática



Resumindo: é importante aprender C porque:

- É uma linguagem **muito usada na prática**
- Para vários tipos de aplicação é uma **excelente ferramenta**

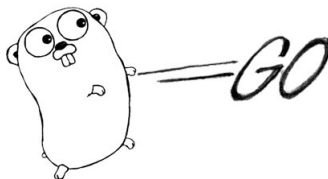


Resumindo: é importante aprender C porque:

- É uma linguagem **muito usada na prática**
- Para vários tipos de aplicação é uma **excelente ferramenta**
- As **características negativas** da linguagem fornecem lições importantes



Alternativas



E sempre haverá **C++**, claro. E Assembly.





Obrigado!

Contato: castor@cin.ufpe.br

Slides em: <http://github.com/fernandocastor>