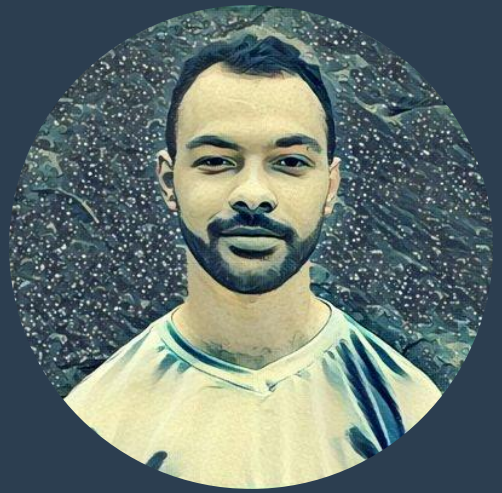
Programação Funcional
não é Modinha

f(x)



Vinicius Reis

Engenheiro de Aplicações @ Decision6

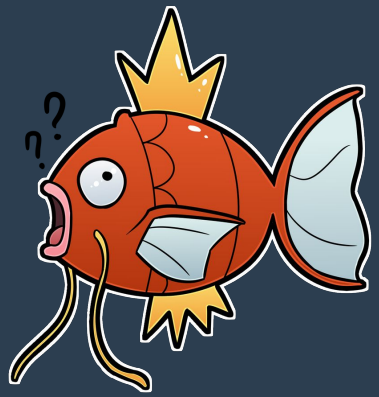
Gravo aulas sobre Vue.js, JavaScript e Laravel para **codecasts.com.br**

@LuizVinicius73

@vinicius73

Vamos falar sobre programação funcional?

Vamos falar sobre programação funcional?



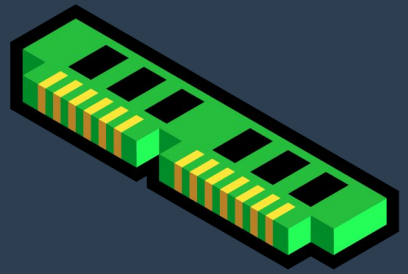
Vamos falar sobre programação funcional?

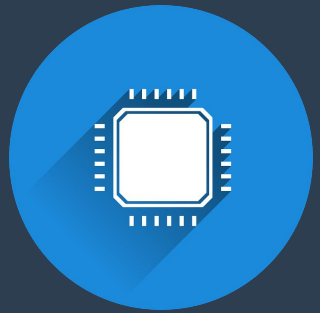
Vamos falar sobre programação funcional?

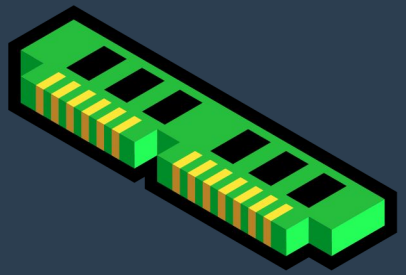
O paradigma funcional não é nenhuma novidade
~1950

“Se é tão velho porque só comecei a ouvir falar disso agora??”

“Se é tão velho porque só comecei a ouvir falar disso agora??”





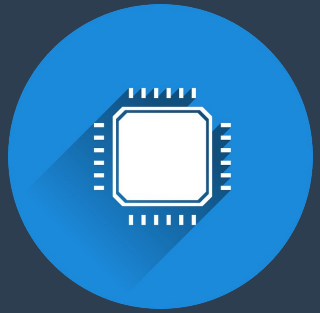


Memória

Memória não era tão barata ou acessível quanto é hoje

Ano	Tecnologia	Valor/Mb (USD)
1957	Flip-Flop	411.041.792
1960	IBM Core memory	5.242.880
1970	IBM Core memory	734.003
1987	DRAM	176
1995	EDO	33
1997	PC66 SDRAM	5
1999/2000	PC133 SDRAM	1
2005	DDR 2	0,15
2011	DDR 3	0,01
2017	DDR 4	0,0046

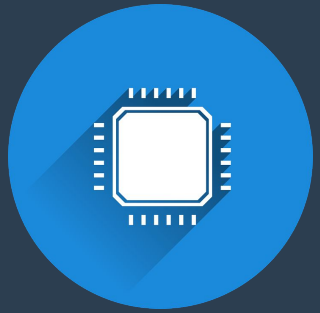
http://www.jcmit.net/memoryprice.htm



Processamento

Antes, quanto mai GHz mais rápido o software seria

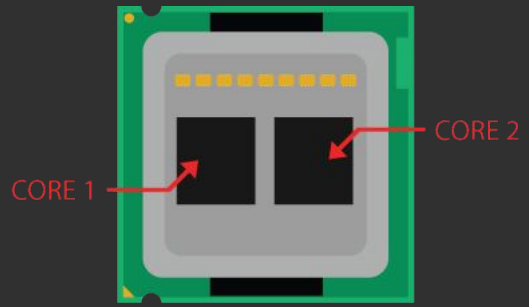
Ano	Modelo	
1978	VAX-11/780	--
1991	HP PA-RISC	0.05 GHz
1993	PowerPC 604	0.1 GHz
1996	Alpha 21164	0.5 GHz
1999	AMD Athlon	1.0 GHz
2000	Intel Pentium 4	2.0 GHz
2005	Intel Pentium D	3.2 GHz
2017	AMD Ryzen	4.1 GHz



Processamento

GHz parou de cresser, mas agora temos múltiplos cores

Ano	Modelo	Cores
2001	IBM POWER4	2
2006	Core 2	2
2008	Core i7	6
2011	Xeon E7	10
2017	Ryzen	32



MULTI-CORE CPU

“E por que estamos ouvindo novamente falar sobre programação funcional?”

“The Free Lunch Is Over: A Fundamental Turn Toward Concurrency in Software”

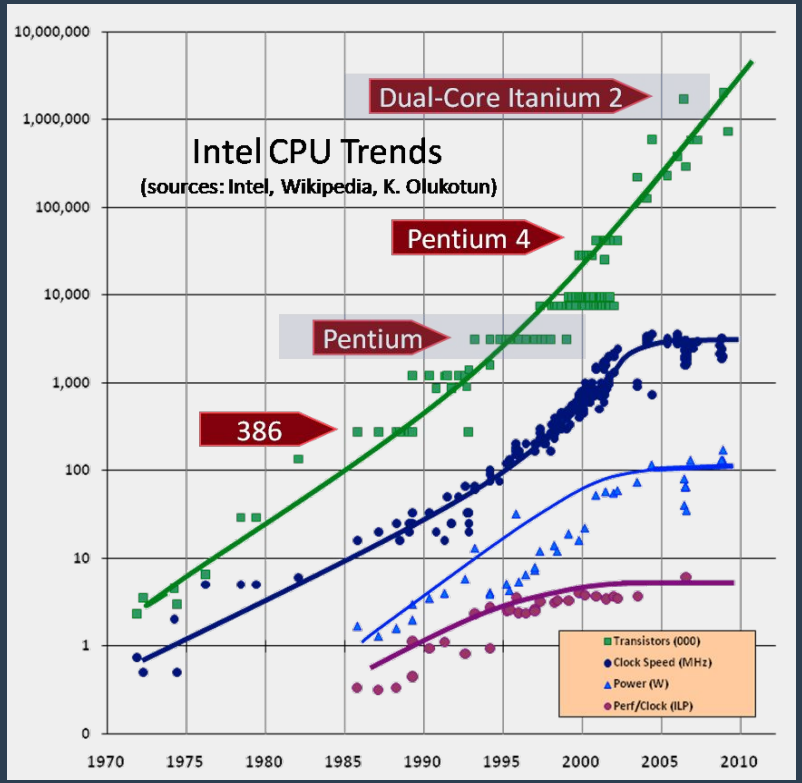
By Herb Sutter

http://www.gotw.ca/publications/concurrency-ddj.htm

The Free Lunch Is Over: A Fundamental Turn Toward Concurrency in Software

Os computadores não são mais os mesmos, os softwares não são mais os mesmos.

É preciso tirar proveito dos múltiplos cores dos processadores para que o software seja eficiente.

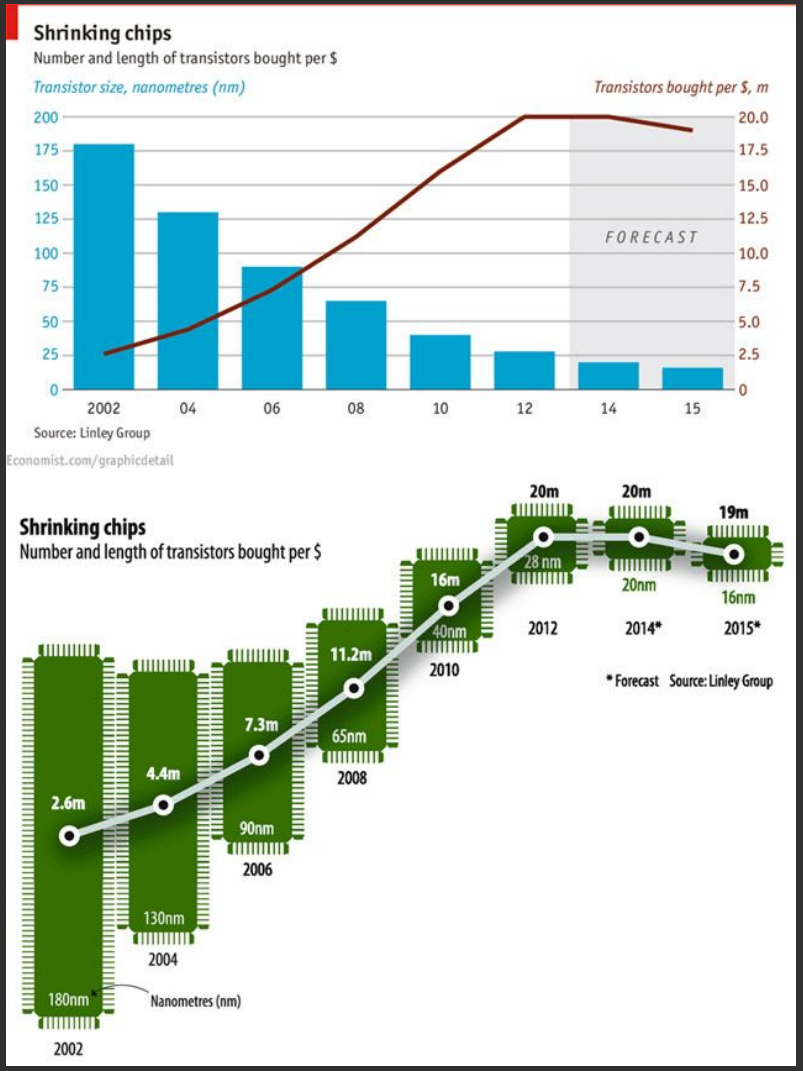


2005

Is This the End of Moore's Law?

Desde 2014 a lei de Moore não está sendo seguida a risca.

https://www.financialsense.com/contributors/marc-chandler/is-this-the-end-of-moore-s-law



Linguagens que suportam o paradigma funcional

Há mais delas do que você imagina

Lisp	1958	Scala	2004
Common Lisp	1984	F#	2005
Erlang	1986	Closure	2007
Haskel	1990	Rust	2010
Python*	1991	Elixir	2011
R	1993	Kotlin*	2011
Lua*	1993	Swift*	2014
JavaScript*	1995	Java 8*	2014
OCaml*	1996	Perl 6*	2015
		Eta	2016

Linguagens que suportam o paradigma funcional

Há mais delas do que você imagina

Lisp	1958	Scala	2004
Common Lisp	1984	F#	2005
Erlang	1986	Closure	2007
Haskel	1990	Rust	2010
Python*	1991	Elixir	2011
R	1993	Kotlin*	2011
Lua*	1993	Swift*	2014
JavaScript*	1995	Java 8*	2014
OCaml*	1996	Perl 6*	2015
		Eta	2016

Linguagens que suportam o paradigma funcional

Há mais delas do que você imagina

Lisp	1958	Scala	2004
Common Lisp	1984	F#	2005
Erlang	1986	Closure	2007
Haskel	1990	Rust	2010
Python*	1991	Elixir	2011
R	1993	Kotlin*	2011
Lua*	1993	Swift*	2014
JavaScript*	1995	Java 8*	2014
OCaml*	1996	Perl 6*	2015
		Eta	2016

Pilares da programação funcional

First-class e higher-order functions

- Funções são tratadas como **valores**
- É possível passar funções como **argumentos**
- É possível receber funções como **resultados**.

First-class e higher-order functions

- Reaproveitamento total de lógicas
- Maior poder de abstração
- Menor complexidade
- Mais expressividade

First-class e higher-order functions

- Funções podem representar qualquer coisa, inclusive valores
- Existem inúmeros partners que podem ser representados a partir desses conceitos
- SOLID não é exclusivo de OOP

Pure functions

- Não possui efeitos colaterais
- Não altera os contextos que têm acesso
- Não altera os argumentos que recebe
- Dado um argumento sempre possui o mesmo resultado

Pure functions

- Previsibilidade
- Atomicidade
- Testes simples e baratos
- Sem surpresas

Pure functions

- É mais intuitivo do que aparenta
- Inúmeros bugs deixam de existir só em adotar esta filosofia
- A previsão pode evitar procedimentos desnecessários

Immutability

- Anda lado a lado com funções puras
- Variáveis não têm seus valores modificados
- Sempre gera novos dados
- Passa-se a pensar em estados e não em variáveis

Immutability

- Previsibilidade
- Código claro e objetivo
- Menos debug
- Maior precisão

Immutability

- Chave da computação paralela e concorrência
- Assim como funções puras, a simples adoção de imutabilidade previne diversos bugs
- É mais intuitivo do que se imagina

Quem esta usando?


Microsoft
Google
XERPA
pagar.me
easy MILE
ejabberd
Telia
docker
AT&T
intel
glyde
LINKQL
Lexmark™
Riot GAMES
mashape
NVIDIA
CITRIX
SQUARE ENIX
fave
6 decision
MOZ
basho
amazon
HostNET
globo.com
ERICSSON
Couchbase
FYNDER
nimblecommerce

Obrigado!

http://bit.ly/modinha-functional-programming