Rust

Rápida, confiável, produtiva -- Escolha três

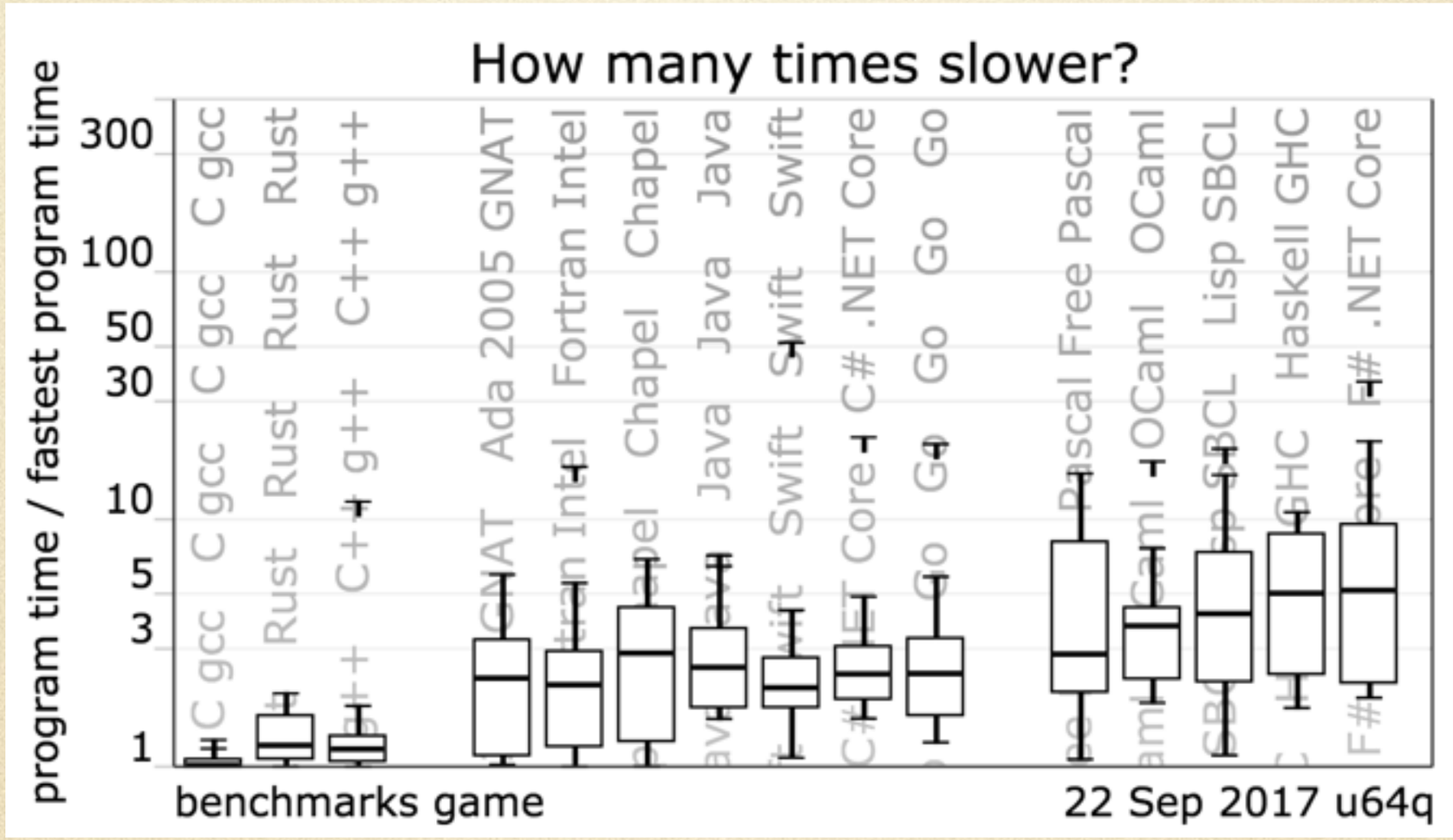
```
fn main() {
    println!("Hello, World!");
}
```

❤️

Loved | Dreaded | Wanted

Loved | Dreaded | Wanted

Benchmarks



Best plaintext responses per second, Dell servers at ServerCentral (238 tests)

Framework	Best performance (higher is better)		Cls	Lng	Plt	FE	Aos	IA	Errors
libreactor	3,987,169	100.0% (97.6%)	Plt	C	lib	Non	Lin	Rea	0
ulib	3,969,896	99.6% (97.2%)	Plt	C++	Non	ULi	Lin	Rea	1
rapidoid-http-fast	3,794,920	95.2% (92.9%)	Plt	Jav	Rap	Non	Lin	Rea	0
tokio-minihttp	3,682,355	92.4% (90.1%)	Mcr	Rus	Rus	tok	Lin	Rea	0

Muitas linguagens

	
Gerenciamento automático de memória	Coletor de lixo Alocação dinâmica
Sem data races	Sem threads
Abstrações	Interpretadores chamadas virtuais ...

Rust

❤️	🏃
Gerenciamento automático de memória	Sem coletor de lixo Alocação na pilha
Sem data races	Paralelismo
Abstrações	De custo zero

Porque não ter um GC?

- GC gerencia memória mas não recursos.
- Minimizar uso de CPU e memória.
- Portabilidade.
- Modelo de execução complexo.
- É um problema em aberto.

Quem

Usa

Veja você mesmo

Ripgrep

DIESEL

Rocket

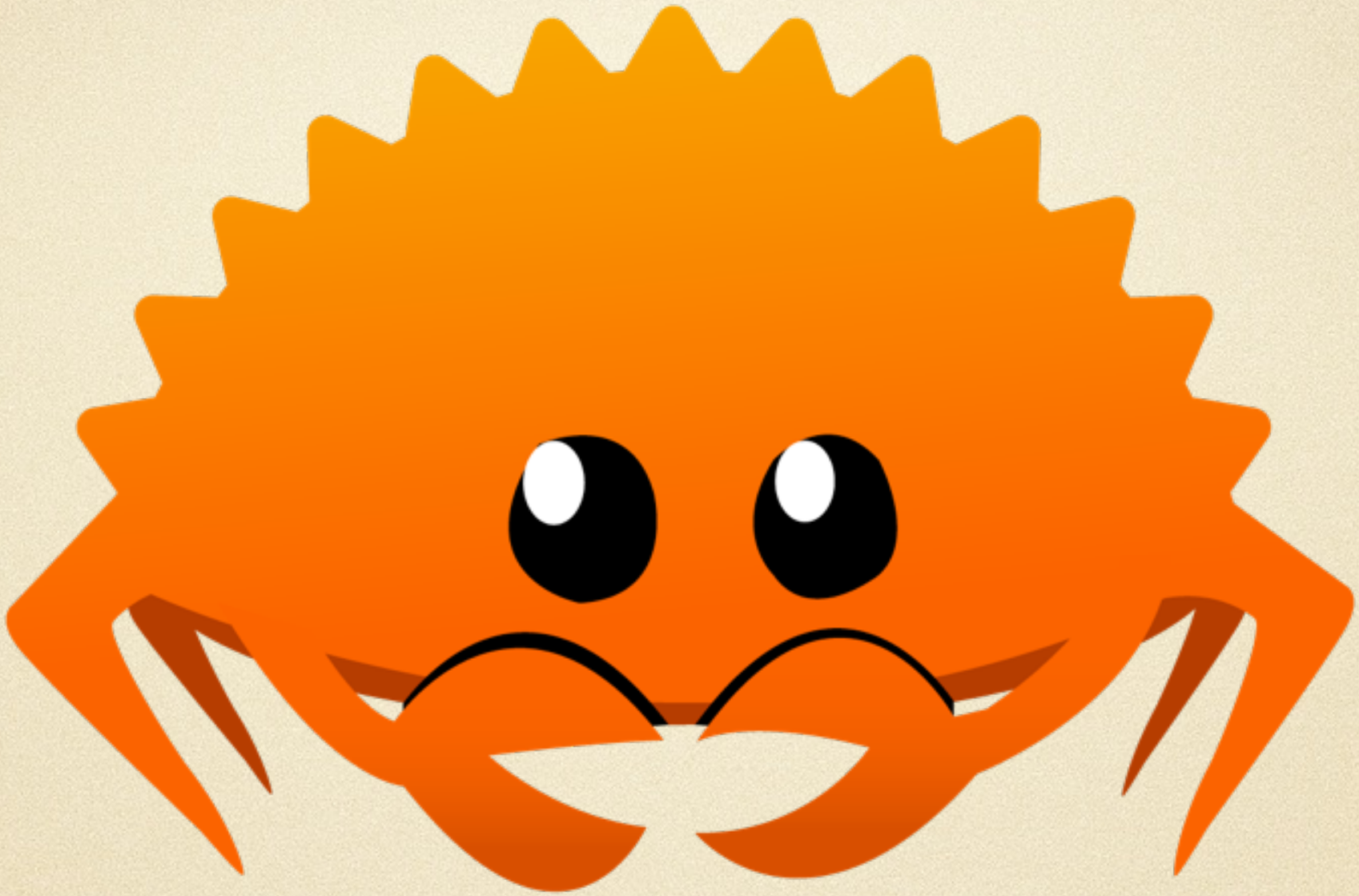
Seus novos amigos

The Book

Rust by Example

Docs.rs

Run ▶



rustup.rs