




# Intro to Rust

Rust Lisbon - 7 Jan 2020







# Background and context

CENTRAL EUROPEMIDDLE EASTSCANDINAVIAAFRICAUKITALYSPAINMORE▼

MUST READ: Google kills Xiaomi-Nest integration after user gets images from strangers

## Microsoft: 70 percent of all security bugs are memory safety issues

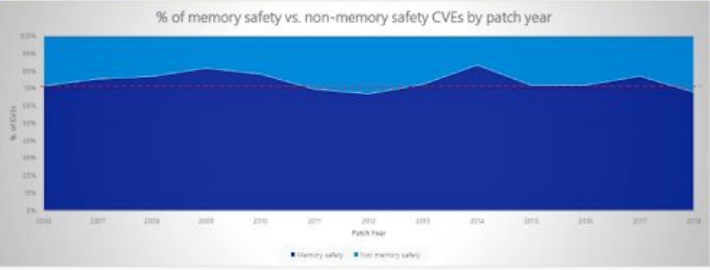
Percentage of memory safety issues has been hovering at 70 percent for the past 12 years.



By Catalin Cimpanu for Zero Day | February 11, 2019 -- 15:48 GMT (15:48 GMT) | Topic: Security

We closely study the root cause trends of vulnerabilities & search for patterns


% of memory safety vs. non-memory safety CVEs by patch year




Legend: Memory safety (dark blue), Non-memory safety (light blue)

Image: Matt Miller

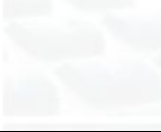
MORE FROM CATALIN CIMPANU



Security  
School management software provider discloses severe security breach



Security  
DeathRansom evolves from joke to actual ransomware



Security  
Company shuts down because of ransomware, leaves 300 without jobs just before holidays

# History and values of Rust

- Founded in 2010 by



- Version 1.0 in 2015
- Current version 1.40

## Values:

- Performance
- Reliability
- Tooling
- Community driven

# Characteristics as language

- Low level language (but with high-level like features)
- Multi-paradigm (draws from imperative, OOP?, procedural, functional...)
- Compiled down to binary/wasm
- Memory safety without garbage collector or runtime (!!!)
- Can be used to build CLI tools, embedded, WASM, network components...

# Main takeaways when programming Rust

“Do as much checking as possible at compile-time”

“Always write safe code” (steep learning curve)

“Immutable by default”

“Only change what is yours to change”

“Explicitly manage memory when copying variables, but implicitly derive variables’ types”

Code time!

# Ownership system

- Central feature of Rust
- Prevents data-race at compile-time!
- Compiler is like your father: very strict but guides you to do the right thing

# Ownership system

*3 rules follow we must:*

1. Each value in Rust has a variable that's called its *owner*.
2. There can only be one owner at a time.
3. When the owner goes out of scope, the value will be dropped.



Code time!

# Ownership system

When dealing with mutable references, at any given time, we can only have **either**:

**One mutable** reference

**or**

**Multiple immutable** references

# Rust vs other languages

Language	Time (sec)	Memory (mb)
C++ Gcc	1.94	1.0
Rust	2.16	4.8
Java	4.03	513.8
LuaJIT	12.61	1.0
Lua 5.1	182.74	1.0
Python	314.79	4.9

[Source](#)

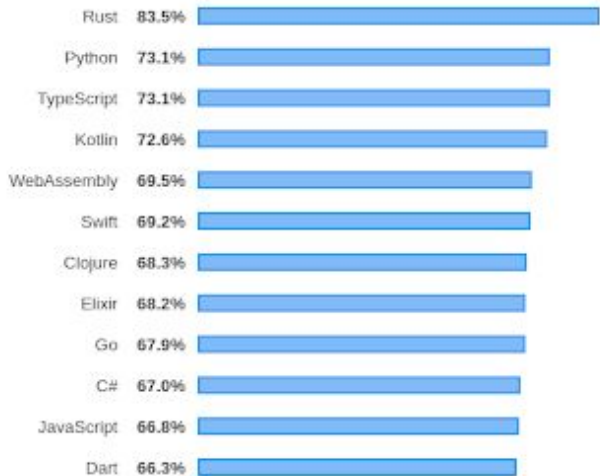
# Rust vs other languages

## Most Loved, Dreaded, and Wanted Languages

Loved

Dreaded

Wanted

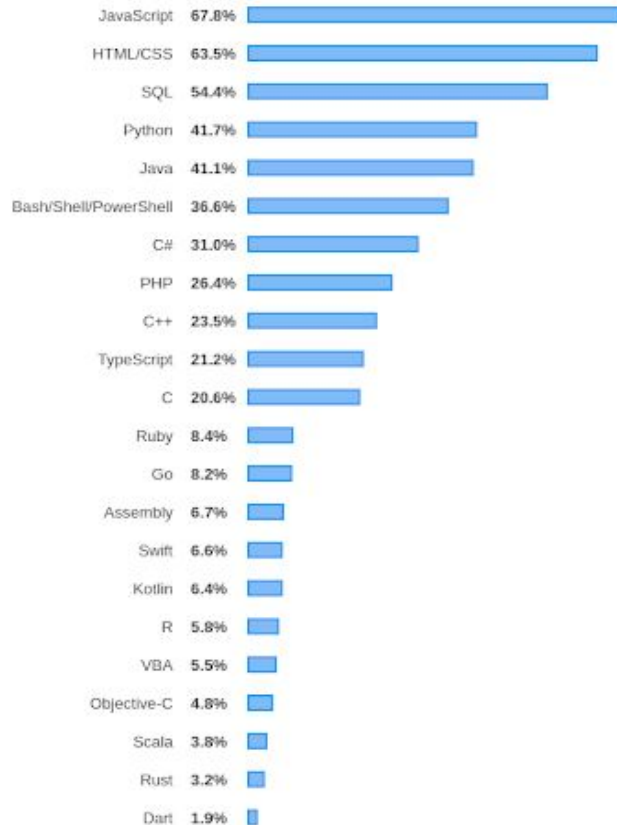


4 years in a row!

## Programming, Scripting, and Markup Languages

All Respondents

Professional Developers



# Rust's ecosystem

- Strong community
- Flourishing [crates](#) ecosystem
- Still a niche language, though adoption rising
- Strong build, toolchain and [docs](#) tooling
- IDE tooling lacking, catching up
- Language is incrementally updating (editions)
- Last addition: *async / await*



Rust Book

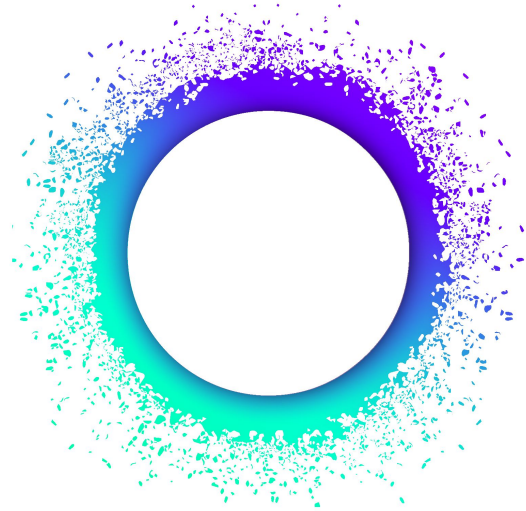


Cargo



WebAssembly

# For decentralized tech



Holochain



Demo time!

# Thank you!

[guillem.cordoba@gmail.com](mailto:guillem.cordoba@gmail.com)

[github.com/guillemcordoba](https://github.com/guillemcordoba)