



# RedisJSON

## A Document DB in Rust

Dr. Christoph Zimmermann

Redis Labs

2. 2. 2020 @ FOSDEM

For Luca



# stat /proc/self

- PhD in reflective operating system architectures
- First crush on Linux: kernel 0.95
- Tech support + more @ FraLUG
- Arch package maintainer
- Hobbies include:
  - SDLC
  - IT security and other forms of black art
  - Community liaison / solution architect @ redislabs

LBW 2020  
@ KRONBERG

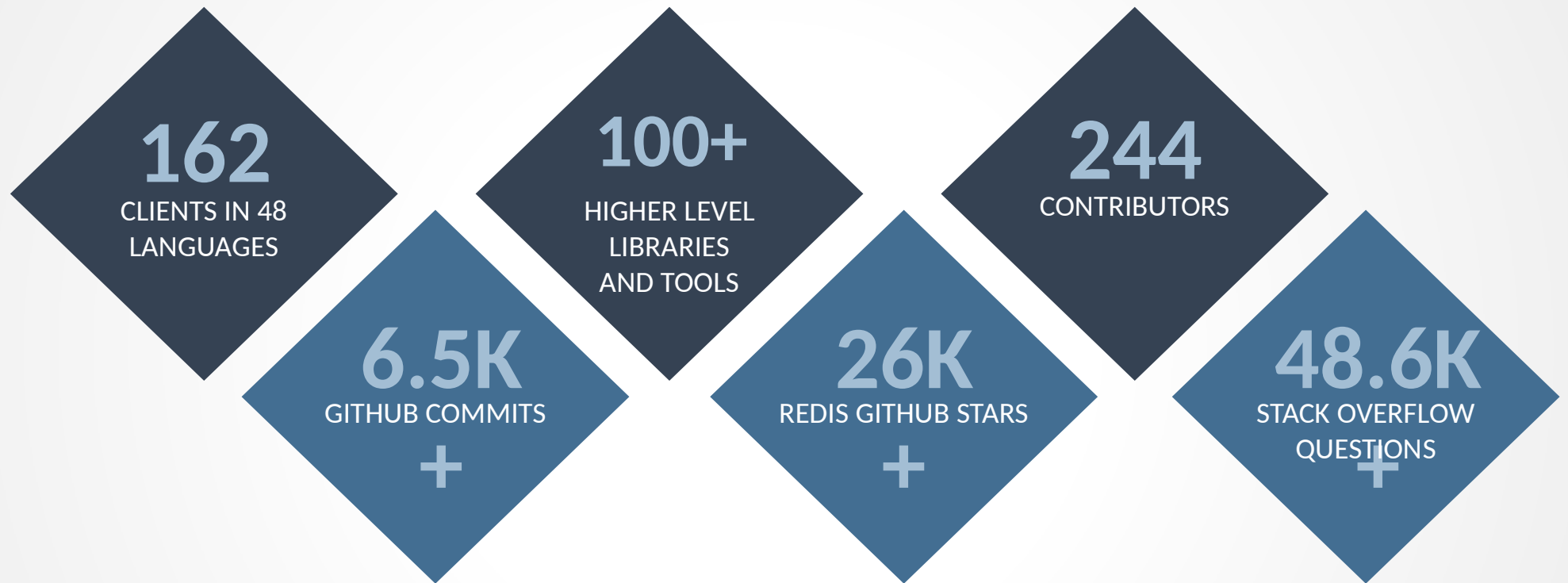


# cat /etc/motd

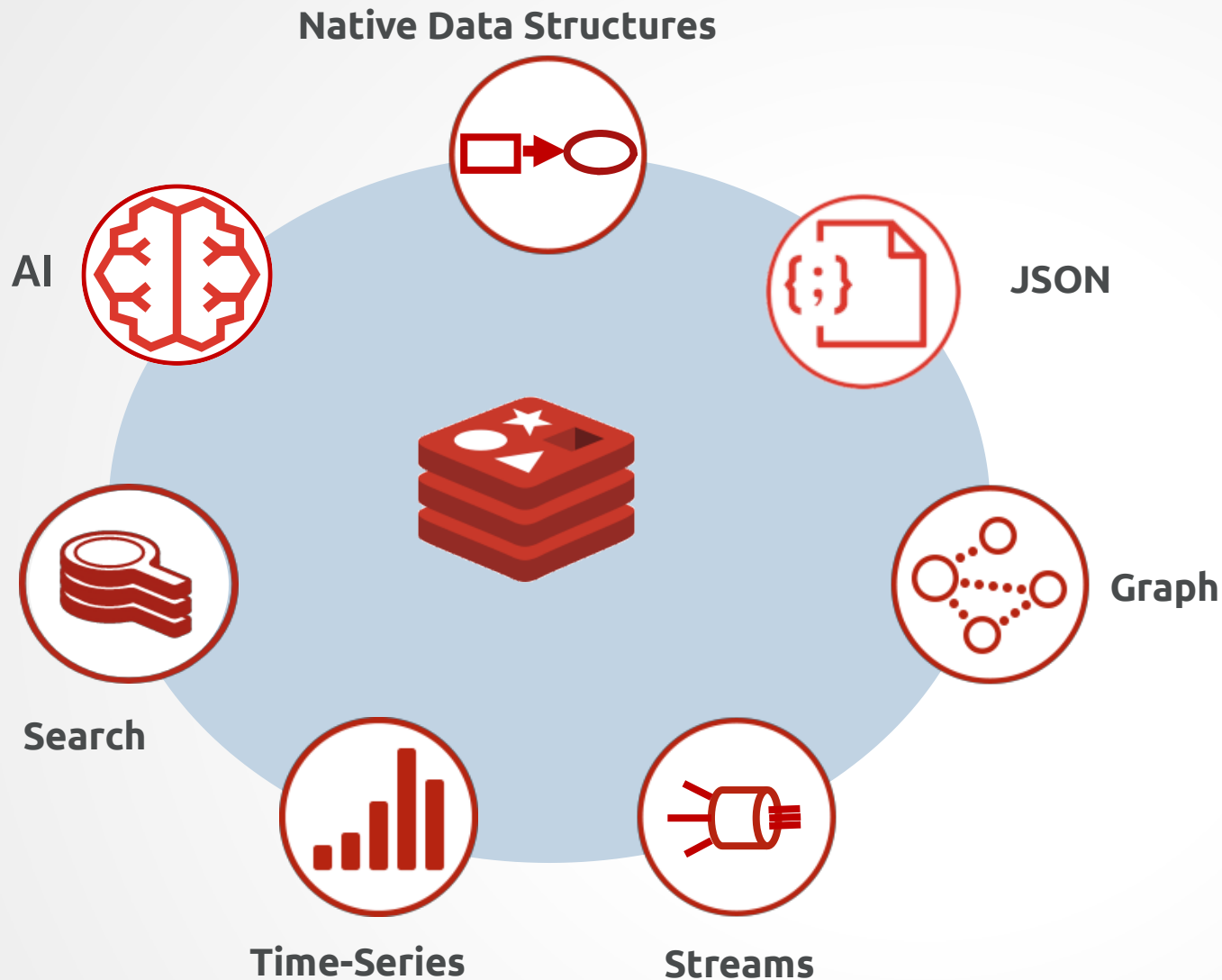
- What is Redis / RedisJSON
- Architecture
- The application perspective
- Summary / outlook



# cat /etc/group



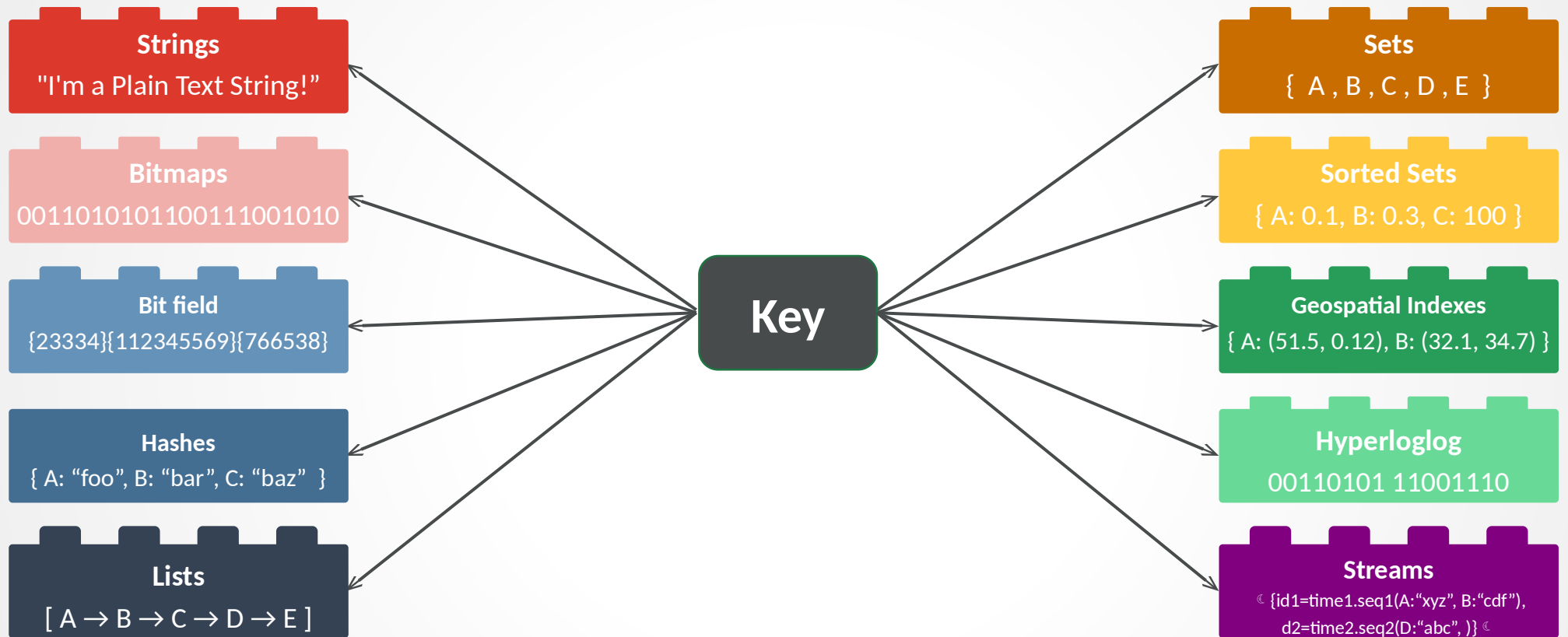
# Is /opt/redis



- Dedicated engine for each data model (vs. API only)
- Models engines can be selectively loaded, according to use case
- All model engines access the same data, eliminating the need for transferring data between them



# type which



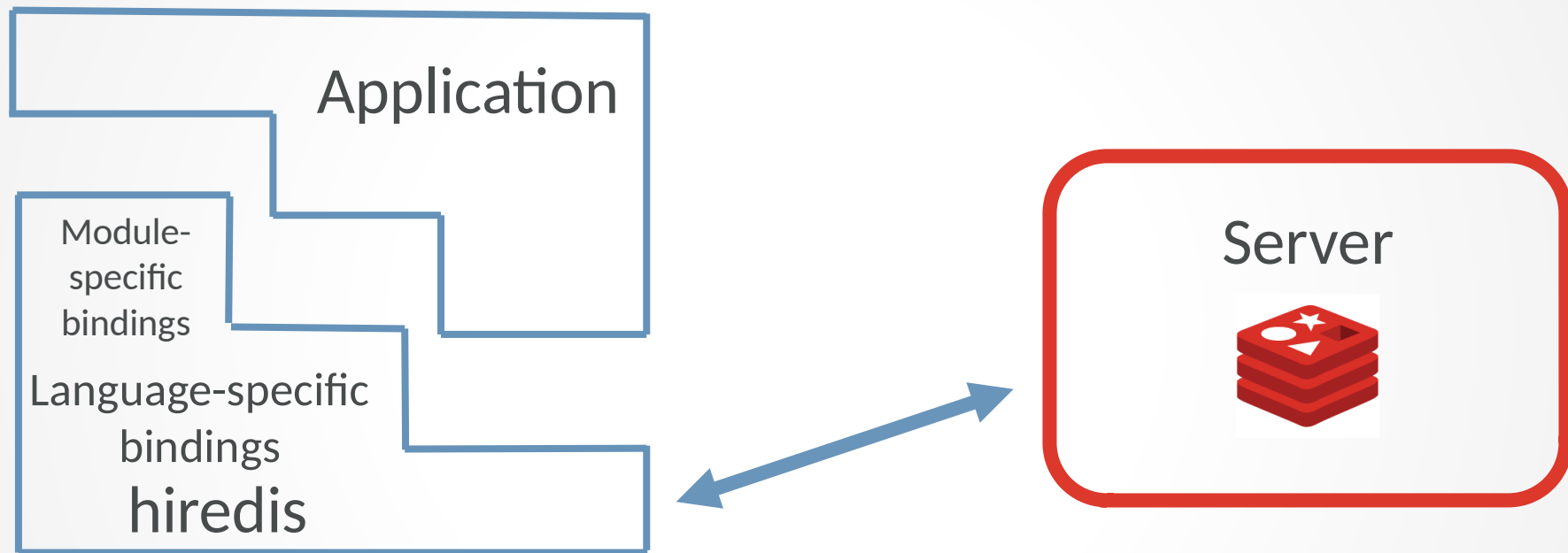
# loadmodule redisjson

- Document DB extension: ECMA-404 compliant module
- Typical commands: `json.set`, `json.get`, `json.arrappend`, `json.arrinsert`
- Navigation via JSONPath:
  - `.foo.bar <=> foo ['bar'] <=> ['foo']['bar']`
  - Wildcard support

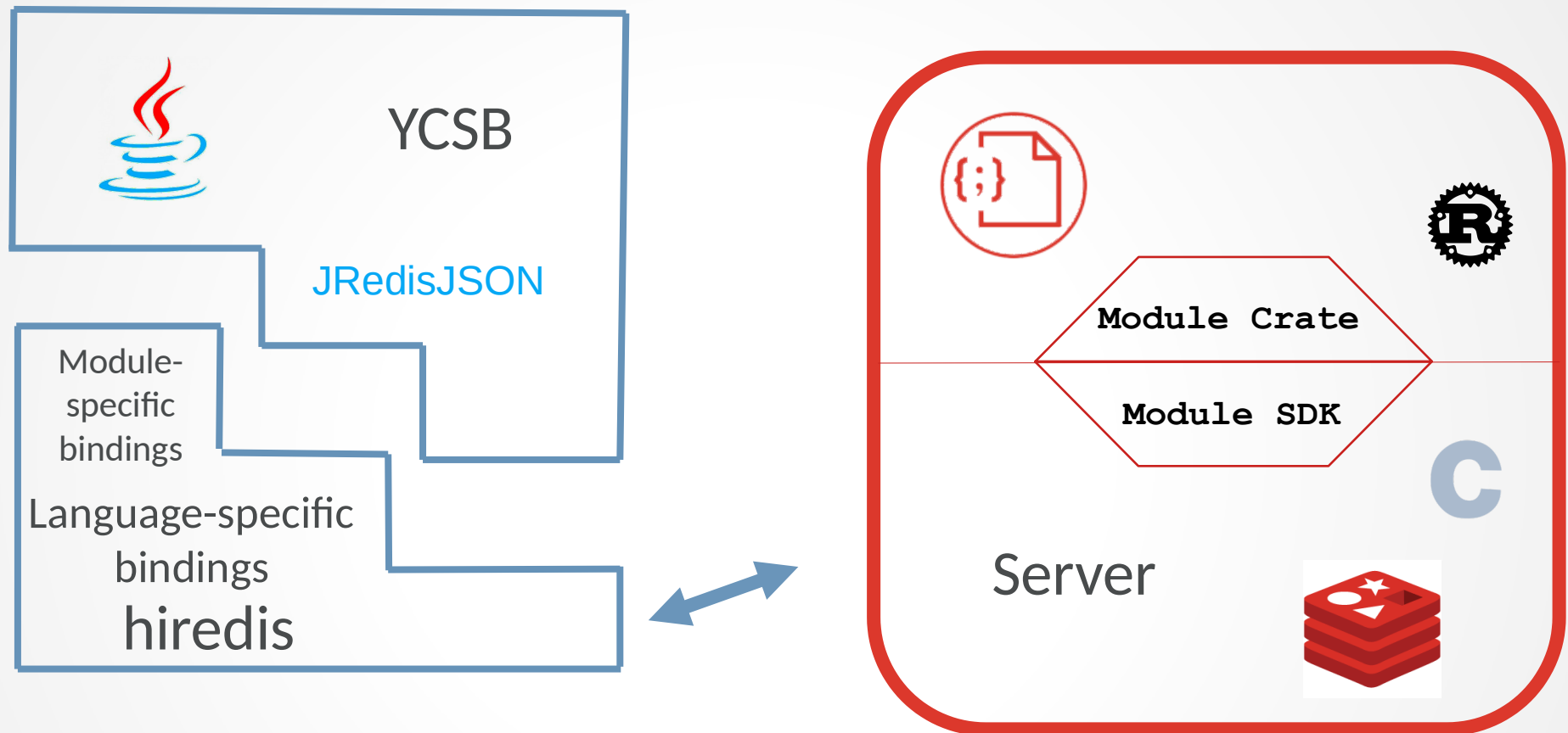




# redis-server -h



# bin/ycsb



# cat /etc/motd

- Original C-based implementation:  $\sim 5.2$  kLoC
- New implementation in Rust:  $\sim 3.2$  kLoC
- Main drivers:
  - Future technology stack
  - Less technical debt => lower QA effort => lower TCO
  - Shorter TTM



# man rustc

- Background:
  - Team with diverse dev background
  - Main reason for C as first choice: in line with server implementation
- Lessons learned:
  - × Steep learning curve
  - ✓ Comprehensive ecosystem
  - ✓ Responsive community
  - ✓ Toolchain support



# info ycsb

- Yahoo Cloud Serving Benchmark: standard Java-based framework
- Pluggable backend for DB integration:
  - Already contains huge variety of NoSQL i/fs
  - Integration of new DB: thin interface for client library
- Relevant workloads: A (R/W: 1/1), B (R/W: .95/.05), F (read-modify-write)



# time ycsb

Workload	A			B			F		
# of threads / database	1	4	8	1	4	8	1	4	8
Redis	62.19	31.41	30.14	53.99	32.04	30.13	53.28	30.28	30.49
RedisJSON	75.27	44.26	42.30	73.73	46.30	43.86	75.71	46.63	41.90
RedisJSON2	81.19	49.88	49.32	83.22	52.25	48.43	73.72	44.66	42.54

## Benchmark spec:

- Stock Eoan on Dell XPS 13 (i7-7560U @ 2.40GHz, 16 GB RAM, 512 GB SSD) with 1M records
- Redis 5.0.5
- RedisJSON (git clone from 1/2/2020), RedisJSON2 (1/3/2020)
- No persistence



- RedisJSON vs. friends:
  - Redis extension => in-memory
  - Focus: performance
  - CAP theorem: flexibility powering variety of use-cases
- Outlook:
  - Integration with RediSearch
  - Functionality improvements



# apropos redis

- [redis.io](https://redis.io/): Redis documentation
- [redisjson.io](https://redisjson.io/): JSON module
- [github.com/RedisJSON](https://github.com/RedisJSON): RedisJSON + RedisJSON2
- [github.com/antirez/redis](https://github.com/antirez/redis): Redis
- [github.com/brianfrankcooper/YCSB](https://github.com/brianfrankcooper/YCSB): Yahoo Cloud Server Benchmark (PR for RedisJSON pending soon :-)
- [university.redislabs.com](https://university.redislabs.com): Redis Labs university





# Questions?



# Thank you!

© 2020 CC-BY

Dr. Christoph Zimmermann

monochromec at <ignore>space</ignore>gmail<dot></dot>com

