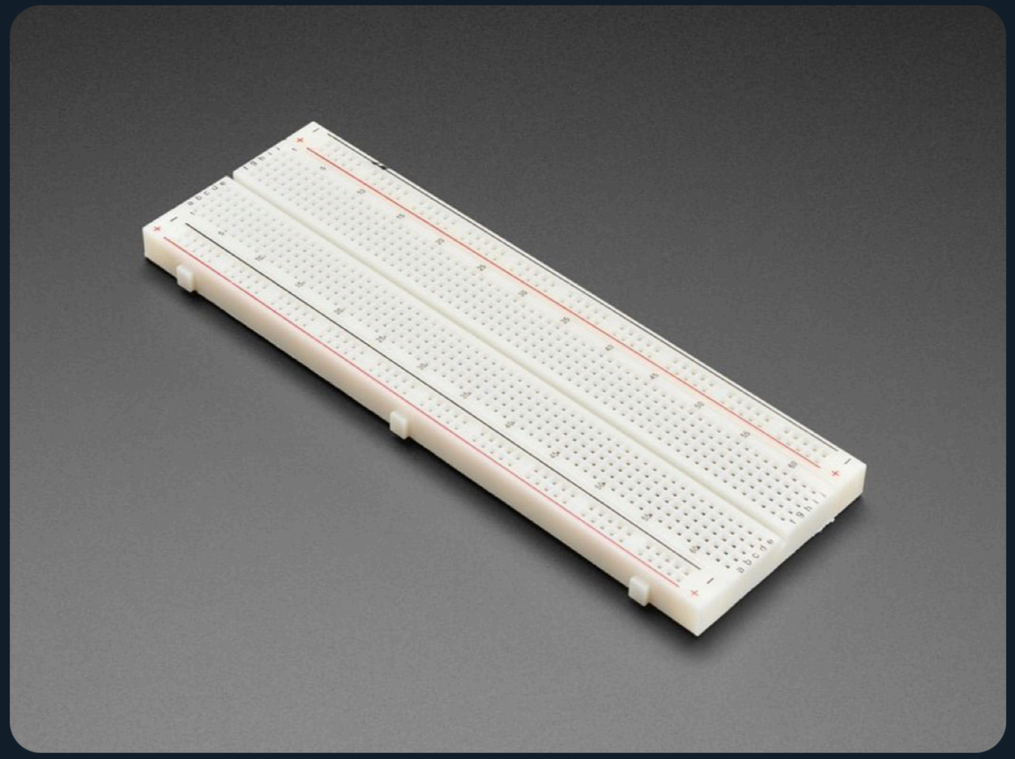
RUNNING RUST APPLICATIONS ABSOLUTELY EVERYWHERE
- ON EMBEDDED, SERVERS, AND THE BROWSER

THE FULLEST STACK

Devendra •ᴗ•
@iotambat

oh you're a full-stack engineer? what's this then?



23:06 · 25.01.23

1.838 Retweets **1.119** Quote Tweets **23,6K** Likes

ANATOL ULRICH, FREELANCE DEV – "RUST MAKES ME SLEEP AT NIGHT"


https://anatol.versteht.es/


spookyvision


pandora9001


@dngrs@chaos.social

BEFORE WE CONTINUE

- a few words from today's sponsor

THIS TALK BROUGHT TO YOU BY

AD BREAK!

AD BREAK!

"IN COLLABORATION WITH"

- James Munns!
- not a sponsor!
- https://onevariable.com/
- find both of us in the #rust-embedded matrix room

BUT WHO ARE YOU?

- let's get to know my audience
- I'll adapt based on your responses

TABLE OF CONTENTS

- what makes Rust the fullest stackest languagest
- what browsers have to do with it
- demo time!
- explain the inner workings
- outlook/nerdsnipe/more demos
- Q&A

STACK? PLATFORM? MICROSERVICE? BROKER? RUNTIME?

- Rust excels at cross compilation
- modern language with types that just work, everywhere*
- "write once, ~~run~~ compile anywhere"
- don't build a runtime when you can use an existing one
- heterogeneous applications need to *communicate*

POSTCARD, A (VERY) QUICK TOUR

- Fast, space efficient and embedded friendly data format
- mostly Rust-only

HOWTO COMPLEX APPLICATION IN ONE (1) SLIDE

- Define one single data model as source of truth
- impls are fine too! (can feature gate based on availability of `std`)
- Never convert or special case anything unless you absolutely have to
- Store state serde-friendly or in a database when that makes more sense
- Now just™ add a (G)UI! In Rust!
- Consider browsers for that

DEMO TIME!

- so you bought a board on AliExpress
- or designed one yourself
- now what?

HOW DOES THIS WORK?

- UI: dioxus & plotters-rs
- communication: postcard-rpc
- connectivity: WebUSB

WHAT'S NOT TO LIKE?

- this sounds like a lot of code
- what can we do about it?

LET'S GET RID OF CODE

- don't repeat yourself: write UI elements *once* and
- we can use postcard's `experimental-derive` feature for type introspection
- postcard-rpc is typed though:

```
pub async fn send_resp<E: Endpoint>(
    &self,
    t: &E::Request,
) -> Result<E::Response, HostErr<WireErr>>
where
    E::Request: Serialize + Schema,
    E::Response: DeserializeOwned + Schema,
```

LET'S GET RID OF CODE FOR GOOD

- postcard-dyn to the rescue
- leverages serde-json's dynamic `Value` type
- add dynamic call in postcard-rpc:

```
pub async fn send_resp_dyn(
    &self,
    req_schema: &'static NamedType,
    req_key: Key,
    resp_schema: &'static NamedType,
    resp_key: Key,
    payload: &serde_json::Value,
)
```

- sprinkle UI hints on top


“Dynamic types? ”
In *my* Rust?
It’s more likely than you think.
FREE PC CHECK!
CONTENTwatch™

MORE DEMO: LET'S TALK BACKEND

- you have some gadget deployed all over the world
- despite your best efforts, crash reports start rolling in
- nothing makes any sense? let's visualize!

REDUCE YOUR DATABASE LOAD BY OVER 9000% WITH THIS ONE WEIRD TRICK

- client-server apps are not always the best interactive experience
- you could add caching, but then you have two problems
- what else could we do?

EVEN MORE DEMO: DSP, LOGGING

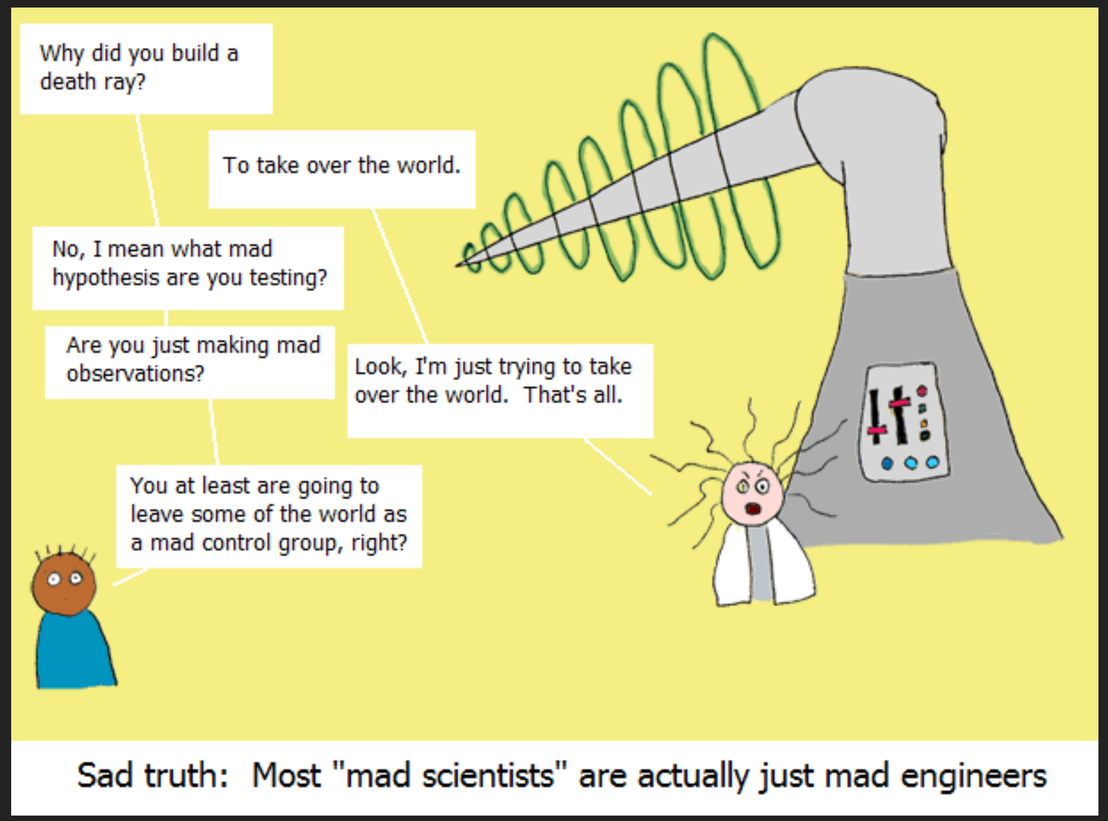
- how about viewing some live FFT data?
- `std`-style logging is a papercut in embedded Rust
- defmt to the rescue! But typically requires a probe and the firmware
- log to a browser, or a central database, the sky is the limit

EVEN MORER DEMOER: HOW ABOUT AN ENTIRE OS

- and it's oops all async

WRAPPING UP

- some things are not quite great yet
- did I get you excited?
- check out Bret Victor's Kill Math and Stop drawing dead fish



THE END

WHAT ARE YOUR QUESTIONS?