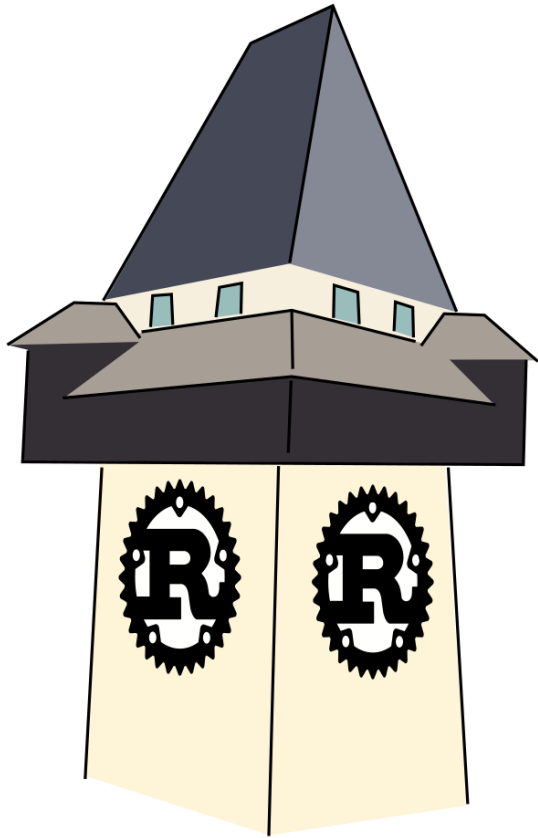


RUST GRAZ – 00

INTRO

Lukas Prokop

July 24, 2019



Data types and Trait `std::iter::Iterator`

25th of July 2019

Rust Graz, lab10

WELCOME BACK!

After one year w/o a talk, we are going to reboot RustGraz.

1. Regular talks each month.
2. Starting with introductory topics in summer 2019.
3. English as preferred language to include internationals.

ABOUT THE MEETUP

ORGANIZATION VIA MEETUP

25
JUL

Thursday, July 25, 2019

Data types and Trait std::iter::Iterator



Hosted by [Lukas Prokop](#)

From [Rust Graz Meetup](#)

Public group

You're going 5 people going



Share: [f](#) [t](#) [in](#)

Organizer tools

Thursday, July 25, 2019
7:00 PM to 10:00 PM
[Add to calendar](#)

lab10 - Incubator & Coworking Space
Strauchergasse 13, 8020 · Graz



Details

We are back! On 25th of July 2019, we are going to start our monthly series on introductory rust topics. In particular, Lukas is going to talk about "Data types and Trait std::iter::Iterator":

- * integers, 128-bit integers
- * usize
- * floating point numbers (IEEE 754)
- * bool, char, string
- * unit, never

WHICH DATE?

survey via framadate.org/TeSBAMxfWoUijdEI

RustGraz date

[Print](#) [Export to CSV](#)

Creator of the poll

Lukas




Description


When shall RustGraz take place regularly? It is fixed to start always at 19:00 and takes place at lab10. Please tick any preferred choices. The final decision will be made by myself (maybe against the majority as I have also some constraints). Please tick before 2019/07/31.

Public link to the poll [🔗](#)

<https://framadate.org/TeSBAMxfWoUijdEI>

Votes



	Second Monday of the month	Second Wednesday of the month	Third Monday of the month	Third Wednesday of the month	Third Thursday of the month	Last Monday of the month	Last Tuesday of the month	Last Wednesday of the month	Last Thursday of the month	
 Your name	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Save

ANNOUNCEMENTS ALSO VIA TWITTER

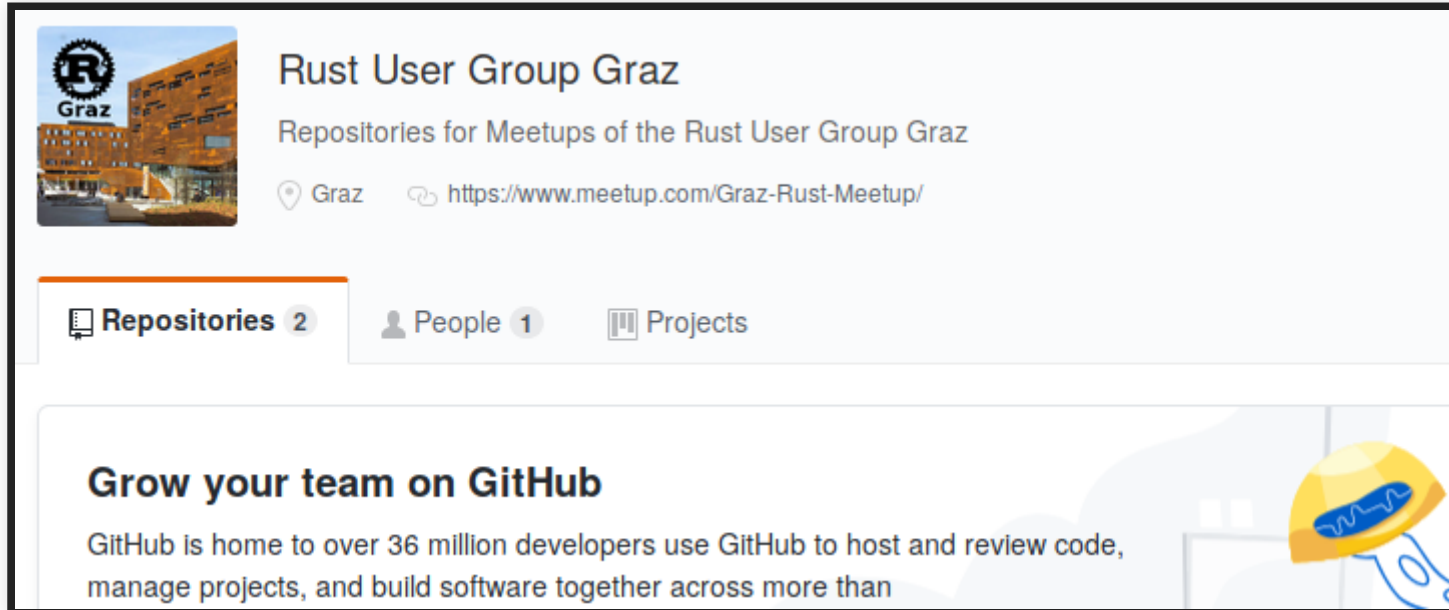


RESOURCES ON GITHUB


github.com/rust-user-group-graz




Today:

[...-graz/00-intro](#), [...-graz/01-getting-started](#),
[...-graz/02-datatypes-and-iterator](#)



The screenshot shows the GitHub profile page for the Rust User Group Graz. The profile header includes a repository icon with the Rust logo and the text "Graz", the name "Rust User Group Graz", and the description "Repositories for Meetups of the Rust User Group Graz". Below this, there is a location pin icon for "Graz" and a link icon for "https://www.meetup.com/Graz-Rust-Meetup/". The main content area has three tabs: "Repositories" (selected, showing 2 items), "People" (showing 1 item), and "Projects". At the bottom, there is a promotional banner for GitHub with the text "Grow your team on GitHub" and a description of GitHub's features, accompanied by a yellow hard hat icon.

 Rust User Group Graz
Repositories for Meetups of the Rust User Group Graz
Graz <https://www.meetup.com/Graz-Rust-Meetup/>

 **Repositories** 2  **People** 1  **Projects**

Grow your team on GitHub
GitHub is home to over 36 million developers use GitHub to host and review code, manage projects, and build software together across more than

ABOUT RUST

Why a meetup about rust? We have a bunch of other meetups going in Graz.



Machine Learning
Graz

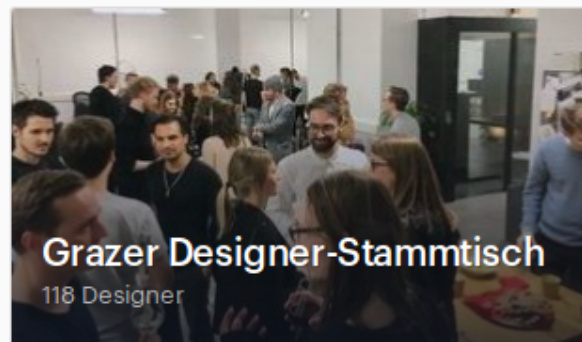
Machine Learning Graz

436 Machine Learners



Python User Group Graz

376 Pythonistas



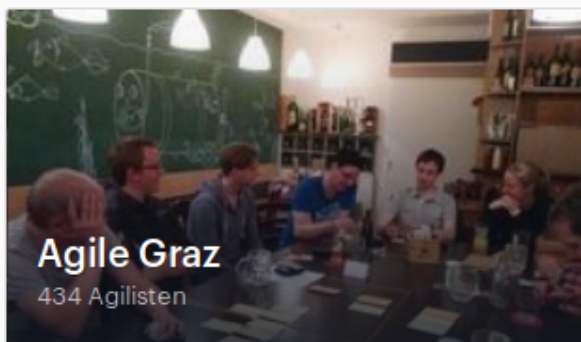
Grazer Designer-Stammtisch

118 Designer



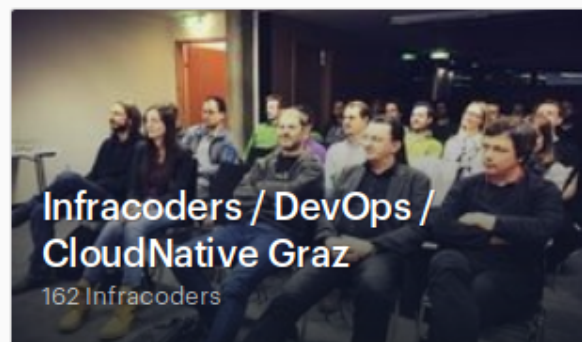
Go Language Usergroup Graz

262 Mitglieder



Agile Graz

434 Agilisten



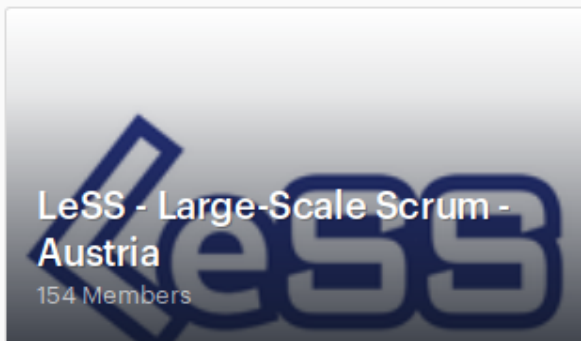
**Infracoders / DevOps /
CloudNative Graz**

162 Infracoders



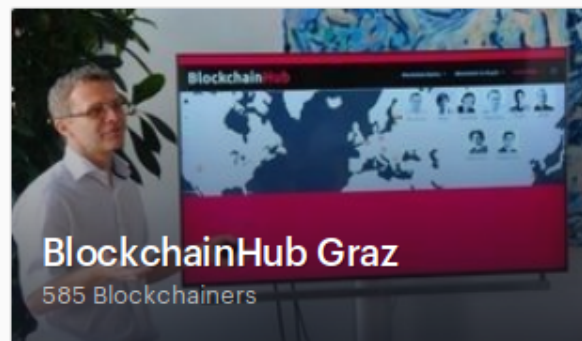
**Microsoft Developer User
Group Graz**

262 .NET-Entwickler



**LeSS - Large-Scale Scrum -
Austria**

154 Members



BlockchainHub Graz

585 Blockchainers

ABOUT THE STATE OF RUST

rust is a *multi-paradigm system* programming language focused on *safety*, especially *safe concurrency* (via [Wikipedia](#)) and a *rich type system* and *ownership model* guaranteeing memory-safety and thread-safety (via [rust-lang.org](#)).

Ok. But does it work?

CURRENT STATE OF RUST: SOME PROJECTS

- [Mozilla Research](#) published rust to improve Firefox ([Firefox Quantum](#), v57, 2×speed, 30% less RAM than Chrome, [75% of Firefox's code rewritten](#), [5 mio. LOCs](#), [servo project](#))

CURRENT STATE OF RUST: SOME PROJECTS

- parity.io develops the “fastest and most advanced Ethereum client” (Parity Ethereum), “a framework for building blockchains and spawning decentralised innovation” (Parity Substrate), “the next-generation platform for connecting independent blockchains together” (Polkadot) and others.

CURRENT STATE OF RUST: SOME PROJECTS

- npm registry uses rust for its CPU-bound bottlenecks
- Several components of the Dropbox core file-storage system were written in Rust to pursue greater datacenter efficiency

CURRENT STATE OF RUST: SOME PROJECTS

- [Redox \(redox-os.org\)](https://redox-os.org) is a Unix-like Operating System written in Rust, aiming to bring the innovations of Rust to a modern microkernel and full set of applications
- “Writing an OS in Rust” (os.phil-opp.com)

[Documentation](#)[Donate](#)[Community](#)[GitLab](#)[RSoC](#)[News](#)[Screenshots](#)

Redox is a Unix-like Operating System written in Rust, aiming to bring the innovations of Rust to a modern microkernel and full set of applications.

[View Releases](#)[Pull from GitLab](#)

- Implemented in Rust
- Microkernel Design
- Includes optional GUI - Orbital
- Supports Rust Standard Library
- MIT Licensed
- Drivers run in Userspace
- Includes common Unix commands
- Custom libc written in Rust (relibc)

CURRENT STATE OF RUST: SOME PROJECTS

- [Jix \(jix.one\)](https://jix.one) implements a SAT solver “[Varisat](#)” in rust and blogs about its design
- [TLS performance: rustls versus OpenSSL](#) (15% quicker to send data, 5% quicker to receive data, 30-70% quicker to resume a client connection, ..., uses less than half the memory of OpenSSL)

CURRENT STATE OF RUST

- For the fourth year in a row, Rust is the **most loved**¹ programming language among our respondents, followed close behind by Python, the fastest-growing major language today ([stackoverflow Developer Survey Results 2019](#))

¹ most loved means “high percentage of developers who are currently using rust express high interest in continuing to do so“

CRITICISM OF RUST

CRITICISM OF RUST

So rust is great and saves the world?

- We saw some positive sides
- Let's talk about some negative sides

Literature:

- [Criticizing the Rust Language, and Why C/C++ Will Never Die](#) by afiskon (translated to [English](#)), abridged version by [Quuxxy](#)
- [Rust is not a good C replacement](#) by Drew DeVault

CRITICIZING THE RUST LANGUAGE, AND WHY C/C++ WILL NEVER DIE

- Since `unsafe` exists, Rust is no better than C++ for safety.
- Safe Rust is not as fast as C++, so you need to write in `unsafe`, thus there's no point.
- C++ has fuzzers and static checkers, and you can write tests.

CRITICIZING THE RUST LANGUAGE, AND WHY C/C++ WILL NEVER DIE

- Rust has five incompatible kinds of pointers (example is that you can't go from having pointers to the heap to pointers to the stack without changing types).
- `Vec<Rc<RefCell<Box<Trait>>>>`
- Rust doesn't insert the necessary Rcs and Boxes for you.

CRITICIZING THE RUST LANGUAGE, AND WHY C/C++ WILL NEVER DIE

- Macros are a crutch and will prevent any good IDEs from existing.
- “cargo actively encourages downloading packages directly from git repositories”
- “C++ doesn't restrict programmers regarding what they can or cannot use.”
- Smart pointers aren't perfect.

CRITICIZING THE RUST LANGUAGE, AND WHY C/C++ WILL NEVER DIE

- No strict description of Rust's semantics.
- “the source of troubles is usually in humans”
- There are no Rust jobs.

RUST IS NOT A GOOD C REPLACEMENT

- “*kitchen sink*” programming language: language solves problems by adding more language features. New features per year:
C → 0.73, Go → 2, C++ → 11.3, Rust → 15
- C is the most portable programming language
- C has a spec
- C has many implementations
- C has a consistent & stable ABI

RUST IS NOT A GOOD C REPLACEMENT

- Cargo is mandatory
- Concurrency is generally a bad thing
- Safety

“Go is the result of C programmers designing a new programming language, and Rust is the result of C++ programmers designing a new programming language”

ABOUT ME

Dec 2017–end 2019

self-employed software developer (python, Go)

Oct 2019+

IAIK PhD student in infosec (interest in RISC-V)

Others

- bachelor's degree math student
- co-organizer of PyGraz and Grazer Linuxtage
- I didn't have time to program rust for too long. Let's get serious in the next months [together]!

DISCLAIMER

I don't think rust should be your first programming language. Thus, I assume familiarity with fundamental programming concepts.