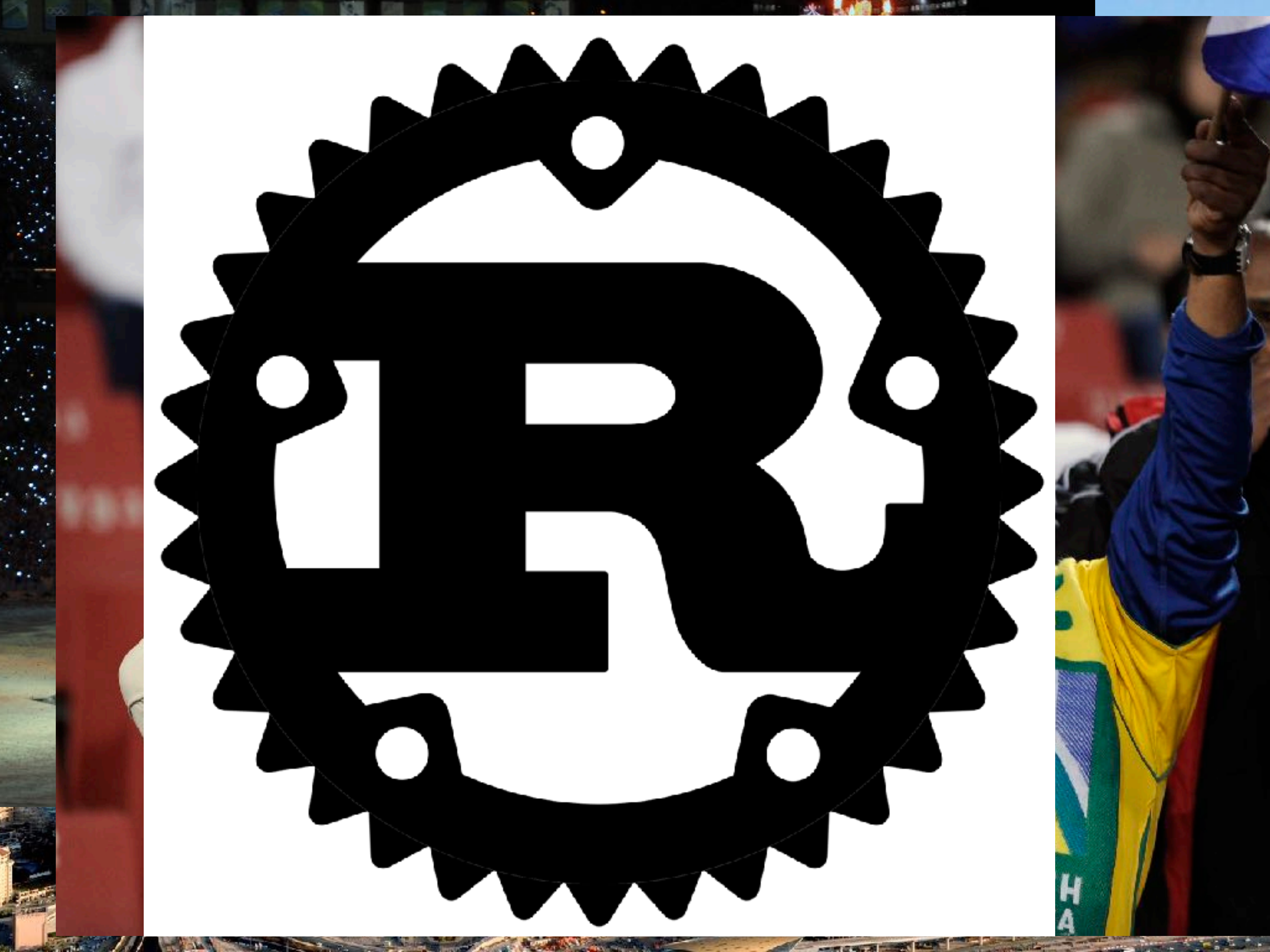
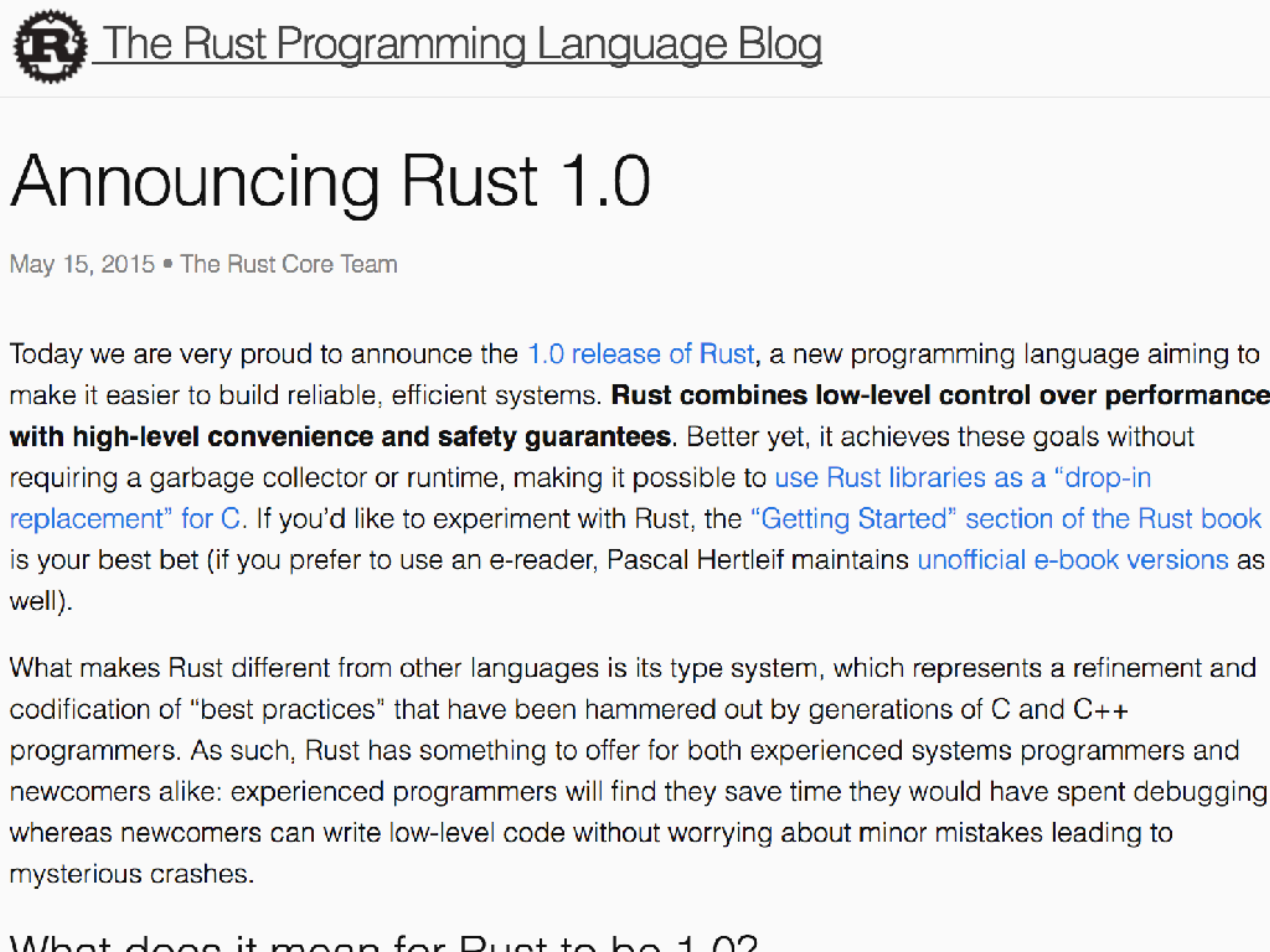
Behind the scenes of Rust

Alex Crichton





The Rust Programming Language Blog

Announcing Rust 1.0

May 15, 2015 • The Rust Core Team

Today we are very proud to announce the 1.0 release of Rust, a new programming language aiming to make it easier to build reliable, efficient systems. **Rust combines low-level control over performance with high-level convenience and safety guarantees**. Better yet, it achieves these goals without requiring a garbage collector or runtime, making it possible to use Rust libraries as a “drop-in replacement” for C. If you’d like to experiment with Rust, the “Getting Started” section of the Rust book is your best bet (if you prefer to use an e-reader, Pascal Hertleif maintains unofficial e-book versions as well).

What makes Rust different from other languages is its type system, which represents a refinement and codification of “best practices” that have been hammered out by generations of C and C++ programmers. As such, Rust has something to offer for both experienced systems programmers and newcomers alike: experienced programmers will find they save time they would have spent debugging whereas newcomers can write low-level code without worrying about minor mistakes leading to mysterious crashes.

What does it mean for Rust to be 1.0?

DUPONT
Tyvek
HomeWrap
Made in the USA

Tooling

Infrastructure

Community

We’re not in 1970

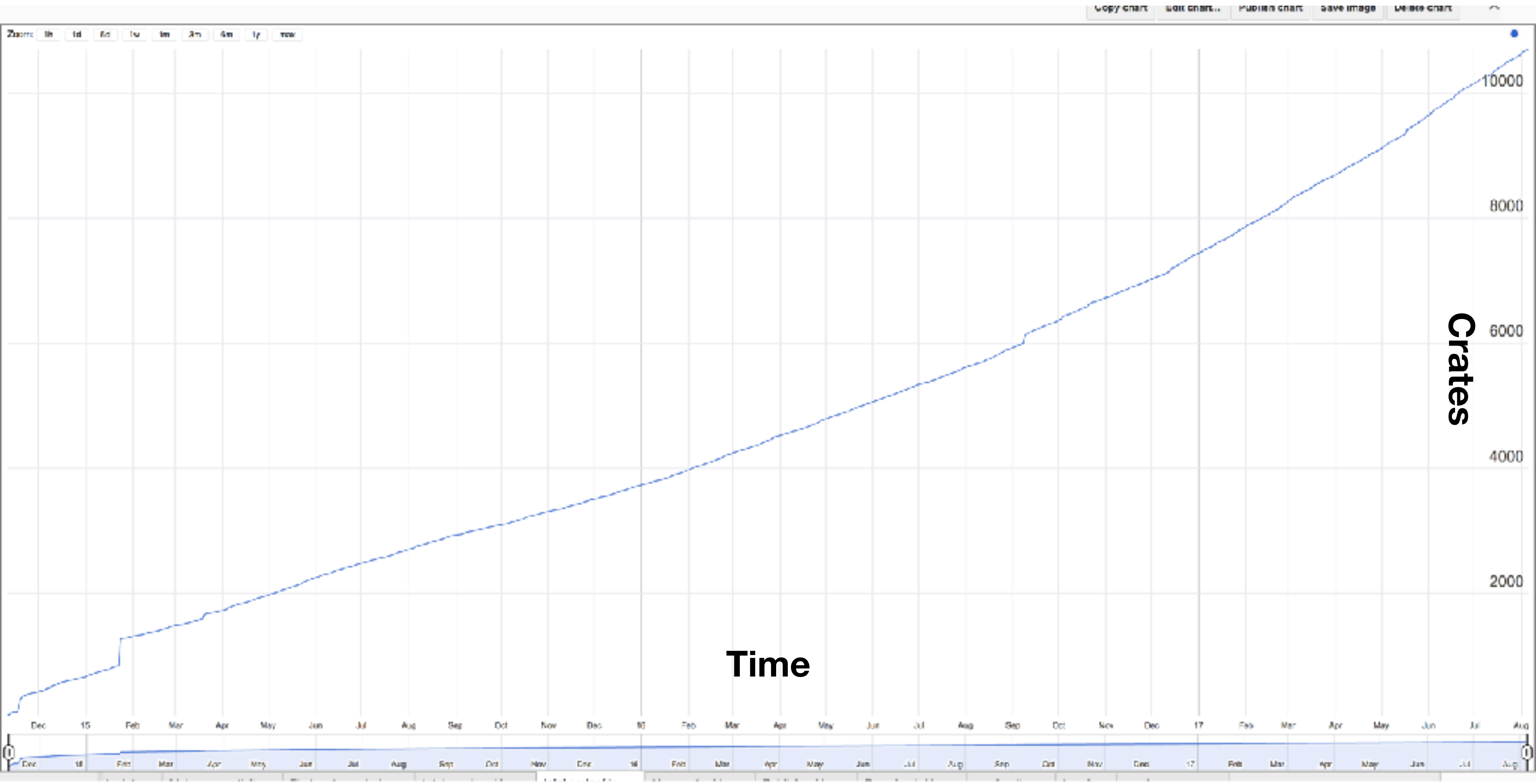
- Developers have high expectations for tooling support
- Developers also expect a lot of tools
- Tools alone can often spawn entire ecosystems

Tools in Rust

- cargo
- rustdoc
- rustup
- rustc

- gdb
- RLS
- rustfmt

Why Cargo?


Copy chart
Edit chart...
Publish chart
Save image
Delete chart
Zoom: 1h 1d 5d 1w 1m 3m 6m 1y max
10000
8000
6000
4000
2000
Crates
Time
Dec 15 Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec 16 Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec 17 Feb Mar Apr May Jun Jul Aug

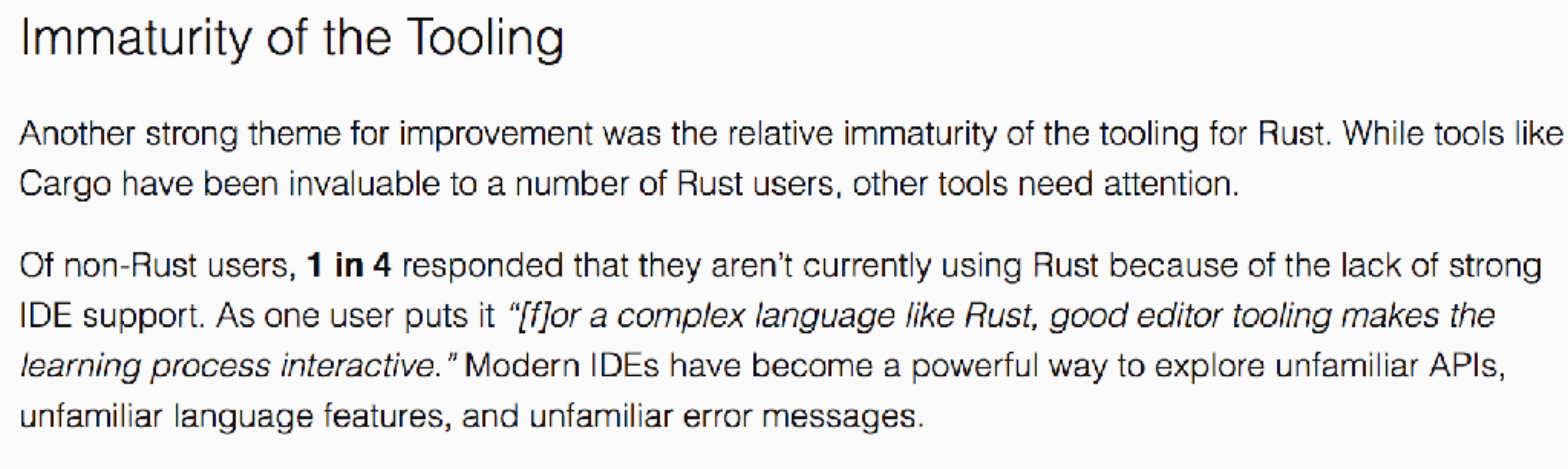
Why Cargo?

- Sharing code is *critical* for a young language
- A package manager from day one is *hugely* beneficial
- Despite blocking 1.0 on Cargo, Cargo accelerated stabilization of libstd through aggressive pruning

How has Cargo helped?

- Small standard library far easier to port and maintain
- “Dependency hell” is almost a thing of the past
- Language features like custom derive far easier with Cargo

What else was missing?

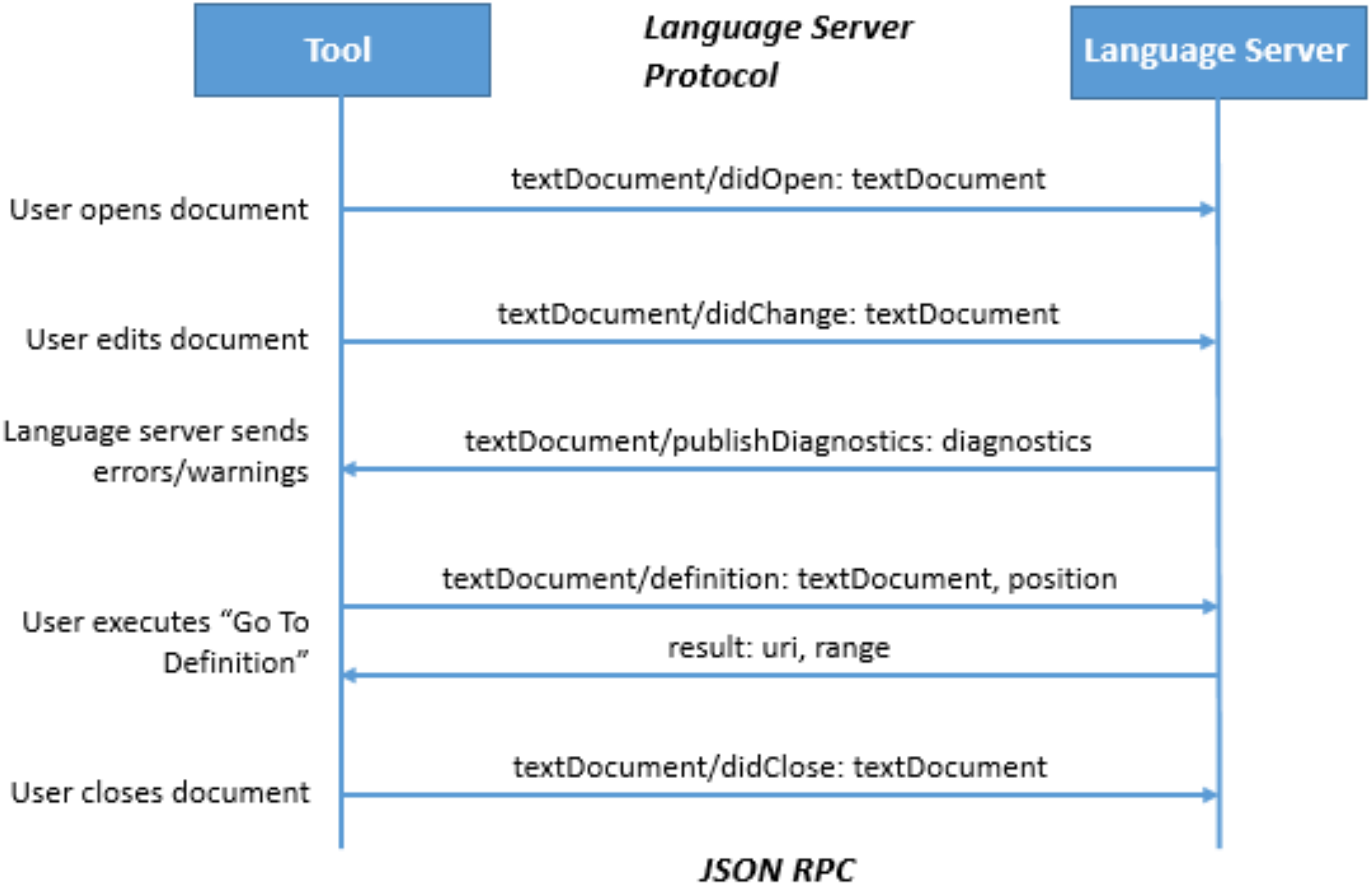


Immaturity of the Tooling

Another strong theme for improvement was the relative immaturity of the tooling for Rust. While tools like Cargo have been invaluable to a number of Rust users, other tools need attention.

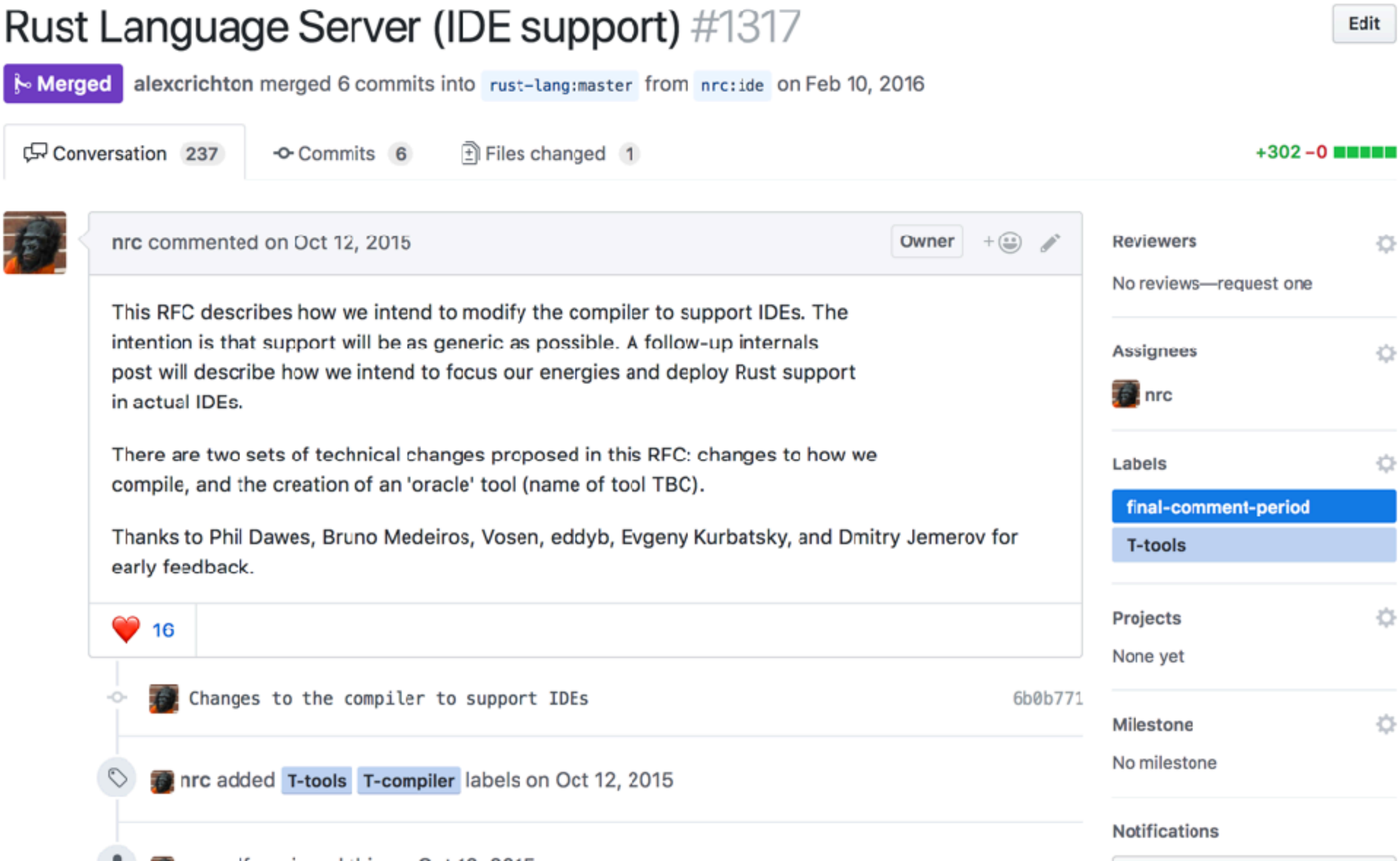
Of non-Rust users, **1 in 4** responded that they aren't currently using Rust because of the lack of strong IDE support. As one user puts it *"[f]or a complex language like Rust, good editor tooling makes the learning process interactive."* Modern IDEs have become a powerful way to explore unfamiliar APIs, unfamiliar language features, and unfamiliar error messages.

Enter the LSP



Tool
Language Server Protocol
Language Server
User opens document
textDocument/didOpen: textDocument
User edits document
textDocument/didChange: textDocument
Language server sends errors/warnings
textDocument/publishDiagnostics: diagnostics
User executes "Go To Definition"
textDocument/definition: textDocument, position
result: uri, range
User closes document
textDocument/didClose: textDocument
JSON RPC

Enter the RLS



Rust Language Server (IDE support) #1317

Merged alexcrichton merged 6 commits into `rust-lang:master` from `nrc:ide` on Feb 10, 2016

nrc commented on Oct 12, 2015

This RFC describes how we intend to modify the compiler to support IDEs. The intention is that support will be as generic as possible. A follow-up internals post will describe how we intend to focus our energies and deploy Rust support in actual IDEs.

There are two sets of technical changes proposed in this RFC: changes to how we compile, and the creation of an 'oracle' tool (name of tool TBC).

Thanks to Phil Dawes, Bruno Medeiros, Vosen, eddyb, Evgeny Kurbatsky, and Dmitry Jemerov for early feedback.

❤️ 16

`Changes to the compiler to support IDEs` 6b0b771

nrc added T-tools T-compiler labels on Oct 12, 2015

Reviewers: No reviews—request one

Assignees: nrc

Labels: final-comment-period, T-tools

Projects: None yet

Milestone: No milestone

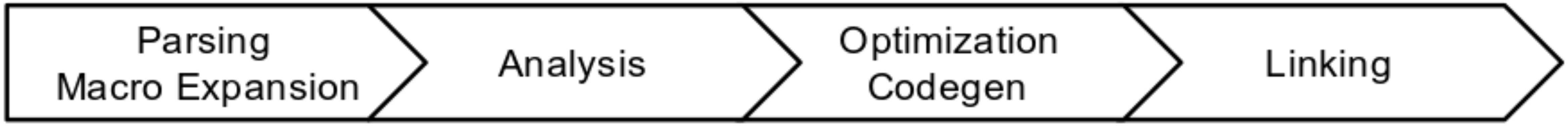
RLS requirements

- Needs to support common IDE queries like
  - go to definition
  - find all uses
  - renaming variables
- Can’t reimplement all of rustc
- Responses must be *fast*

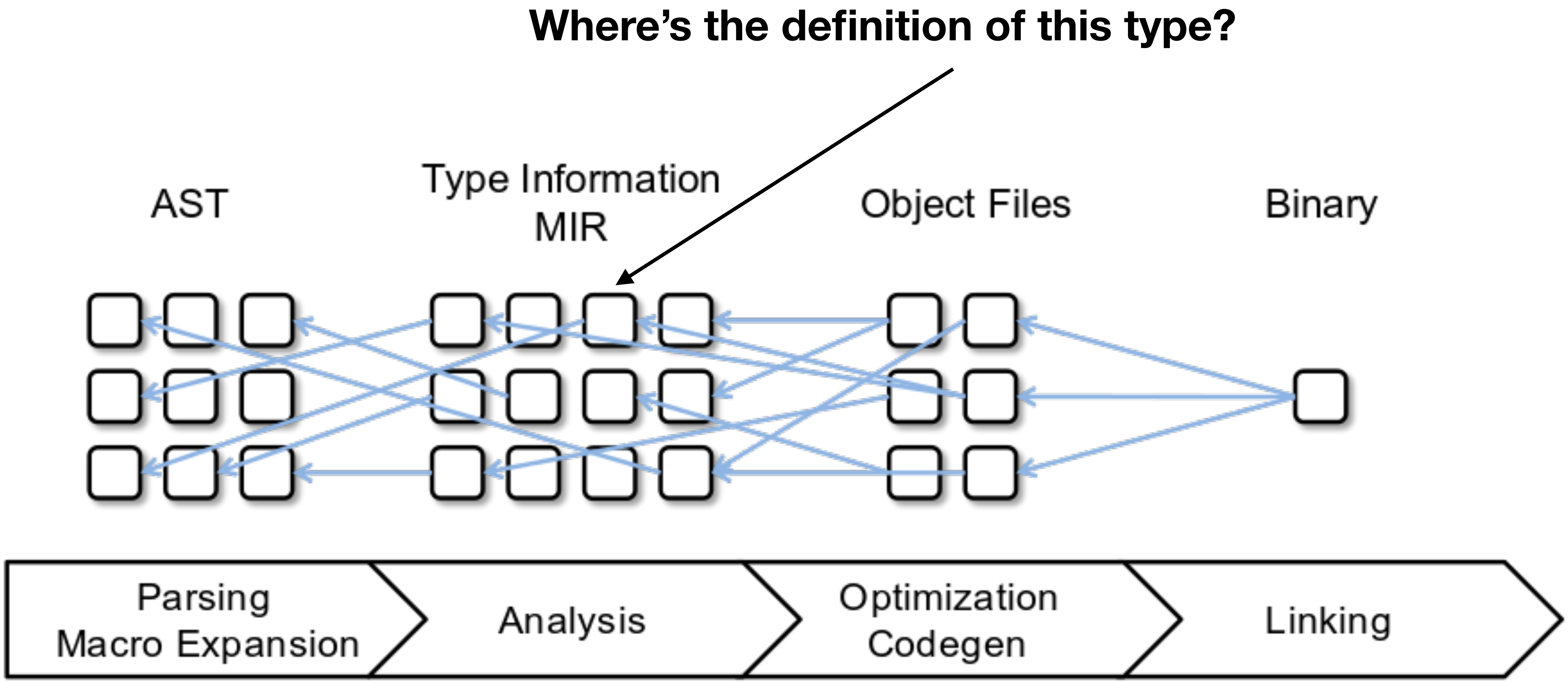
“You must use rustc”

“You must not use rustc”

Getting rustc fast



Getting rustc fast



Polishing the RLS

```
rustup component add rls-preview
```

Tools in Rust

- cargo
- rustdoc
- rustup
- rustc
- gdb
- RLS
- rustfmt

Tooling

Infrastructure

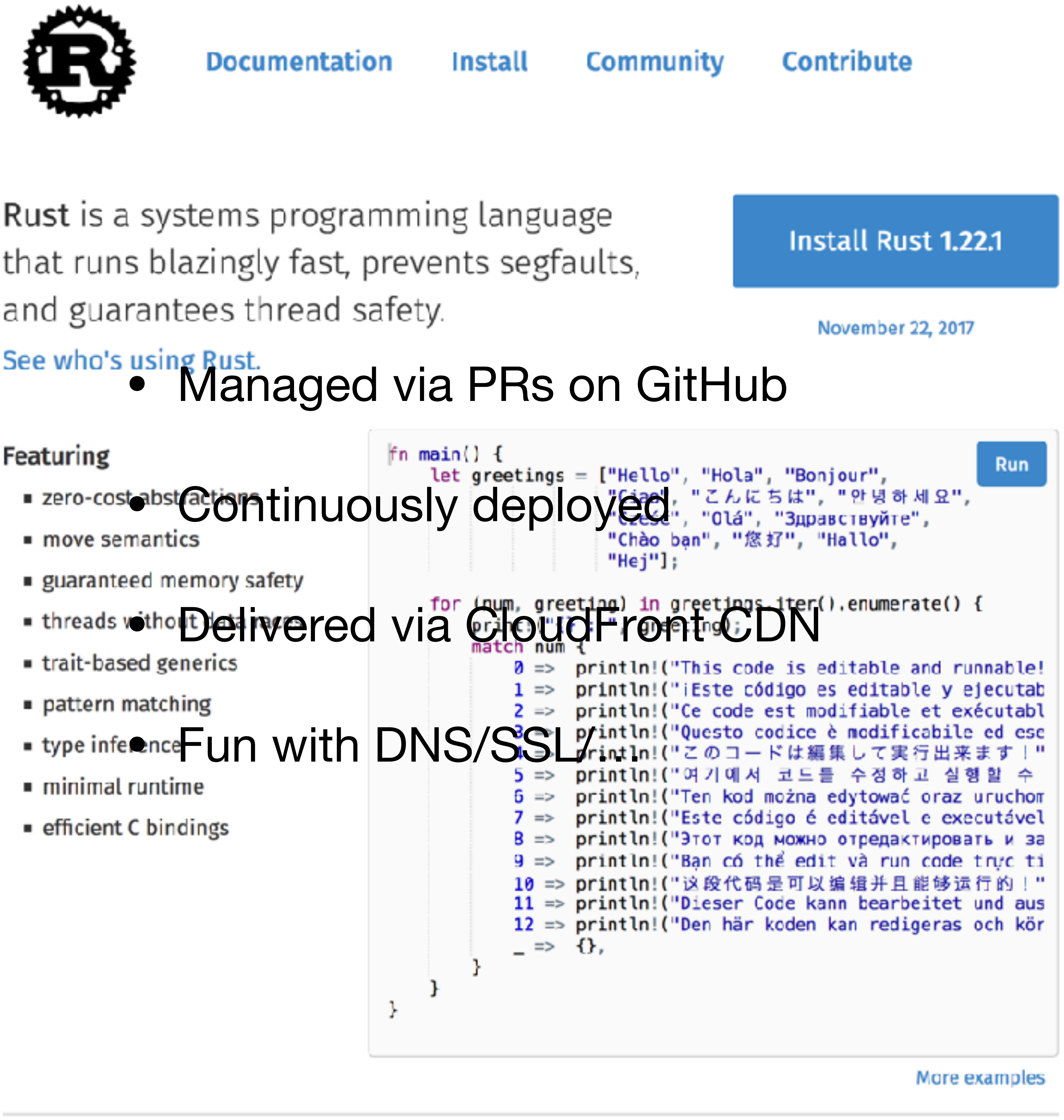
Community

We’re still not in 1970

- Developers expect their tools to not break
- Developers expect low friction when managing tools
- Vast majority of users will always be new ones

Infrastructure of Rust

- Continuous Integration
- {www,doc,play}.rust-lang.org
- crates.io
- Rapid release cycle
- “Dealing with GitHub”
- AWS services, CDNs, storage, etc



Rust is a systems programming language that runs blazingly fast, prevents segfaults, and guarantees thread safety.

See who's using Rust.

November 22, 2017

Featuring

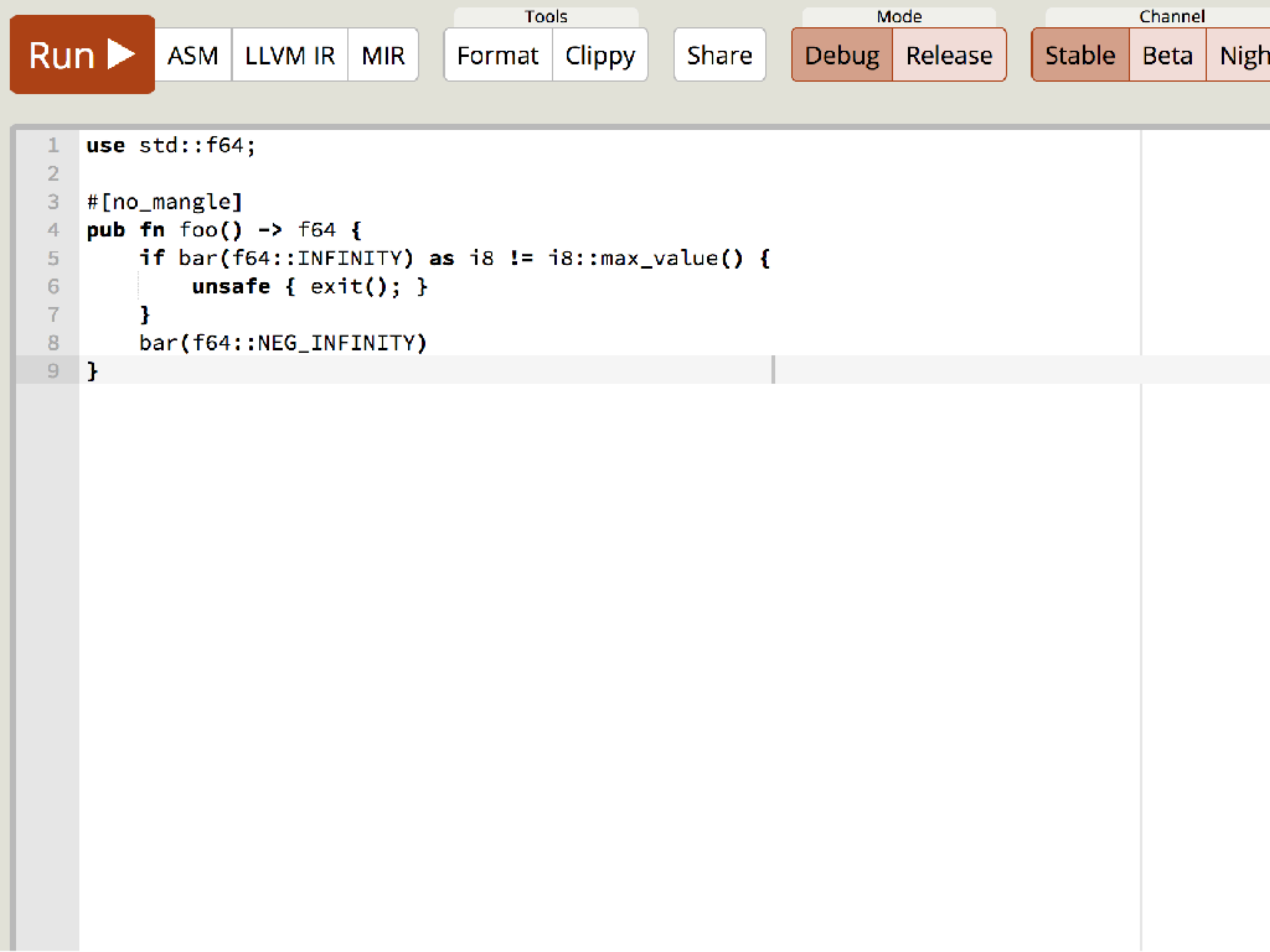
- zero-cost abstractions
- move semantics
- guaranteed memory safety
- threads without data races
- trait-based generics
- pattern matching
- type inference
- minimal runtime
- efficient C bindings

```
fn main() {
    let greetings = ["Hello", "Hola", "Bonjour",
                     "Ciao", "こんにちは", "안녕하세요",
                     "Cześć", "Olá", "Здравствуйте",
                     "Chào bạn", "您好", "Hallo",
                     "Hej"];

    for (num, greeting) in greetings.iter().enumerate() {
        print!("{} : ", greeting);
        match num {
            0 =>  println!("This code is editable and runnable!
            1 =>  println!("¡Este código es editable y ejecutab
            2 =>  println!("Ce code est modifiable et exécutabl
            3 =>  println!("Questo codice è modificabile ed esc
            4 =>  println!("このコードは編集して実行出来ます！"
            5 =>  println!("여기에서 코드를 수정하고 실행할 수
            6 =>  println!("Ten kod można edytować oraz uruchom
            7 =>  println!("Este código é editável e executável
            8 =>  println!("Этот код можно отредактировать и за
            9 =>  println!("Bạn có thể edit và run code trực ti
            10 => println!("这段代码是可以编辑并且能够运行的！"
            11 => println!("Dieser Code kann bearbeitet und aus
            12 => println!("Den här koden kan redigeras och kör
            _ =>  {},
        }
    }
}
```

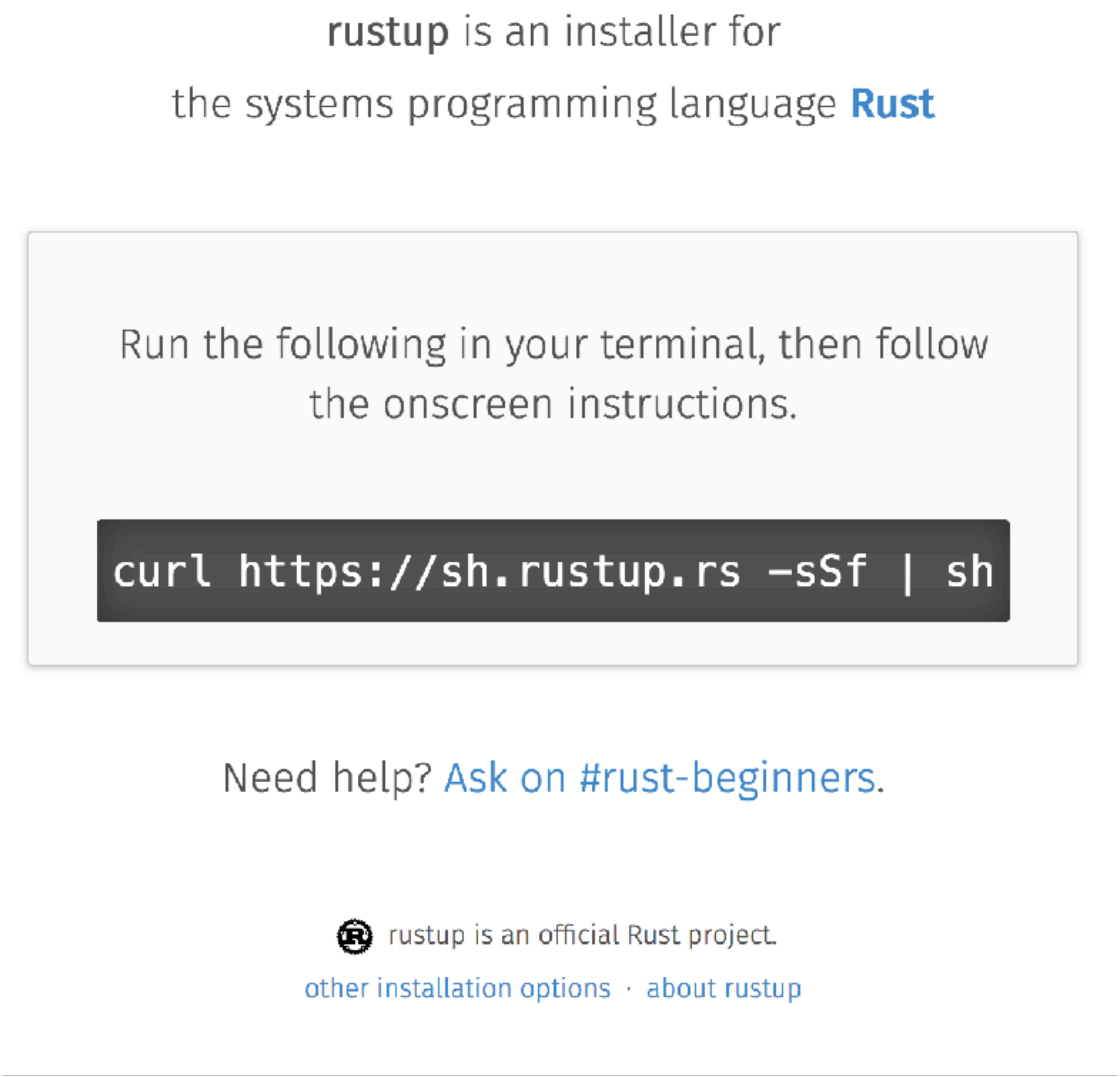
More examples

- Managed via PRs on GitHub
- Continuously deployed
- Delivered via CloudFront CDN
- Fun with DNS/SSL/...



```rust
use std::f64;

#[no_mangle]
pub fn foo() -> f64 {
    if bar(f64::INFINITY) as i8 != i8::max_value() {
        unsafe { exit(); }
    }
    bar(f64::NEG_INFINITY)
}
```



rustup is an installer for

the systems programming language **Rust**

Run the following in your terminal, then follow the onscreen instructions.

```
curl https://sh.rustup.rs -sSf | sh
```

Need help? Ask on #rust-beginners.

rustup is an official Rust project.

other installation options · about rustup

`rustup update`

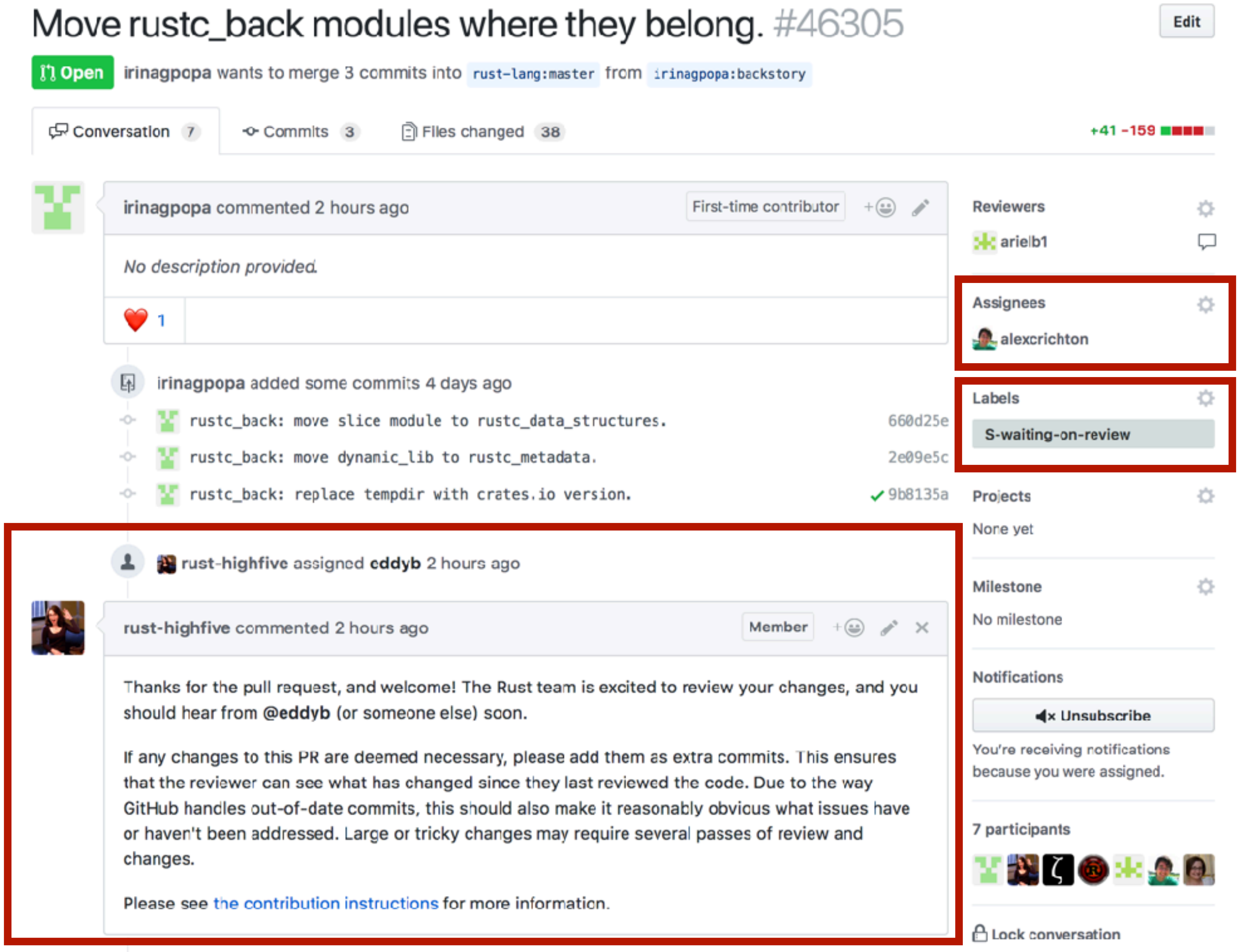
- Also delivered via CDN
- Delivers over 50 targets to compile to
- Crazy platform-specific logic in `rustup` itself

cargo build

- crates.io crates delivered via CDN
- crates.io itself written in Rust
- Deployed via Heroku

You’ve found a bug!

- All rustc/Rust language development happens on GitHub
- GitHub is what most know and love, makes it easiest for new contributors
- Let’s send a PR…



Move rustc_back modules where they belong. #46305

Edit

Open irinagpopa wants to merge 3 commits into `rust-lang:master` from `irinagpopa:backstory`

irinagpopa commented 2 hours ago · First-time contributor

No description provided.

❤️ 1

irinagpopa added some commits 4 days ago

```
rustc_back: move slice module to rustc_data_structures.      660d25e
rustc_back: move dynamic_lib to rustc_metadata.               2e09e5c
rustc_back: replace tempdir with crates.io version.         ✓ 9b8135a
```

rust-highfive assigned **eddyb** 2 hours ago

rust-highfive commented 2 hours ago · Member

Thanks for the pull request, and welcome! The Rust team is excited to review your changes, and you should hear from **@eddyb** (or someone else) soon.

If any changes to this PR are deemed necessary, please add them as extra commits. This ensures that the reviewer can see what has changed since they last reviewed the code. Due to the way GitHub handles out-of-date commits, this should also make it reasonably obvious what issues have or haven't been addressed. Large or tricky changes may require several passes of review and changes.

Please see the contribution instructions for more information.

Reviewers
arielb1

Assignees
alexcrichton

Labels
S-waiting-on-review

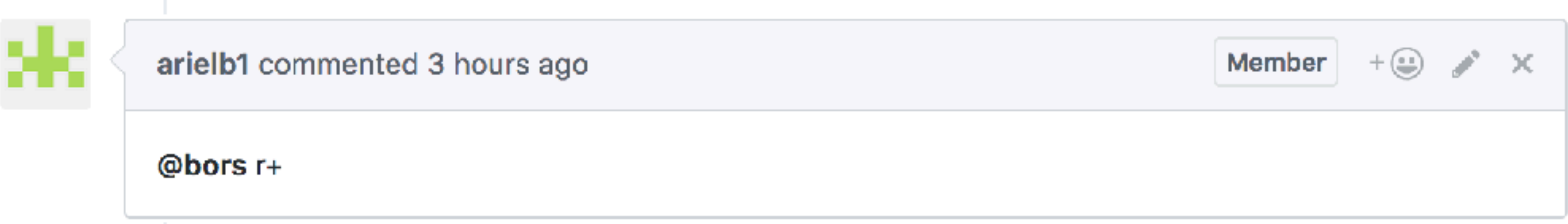
Projects
None yet

Milestone
No milestone

Notifications

You're receiving notifications because you were assigned.

7 participants



arielb1 commented 3 hours ago

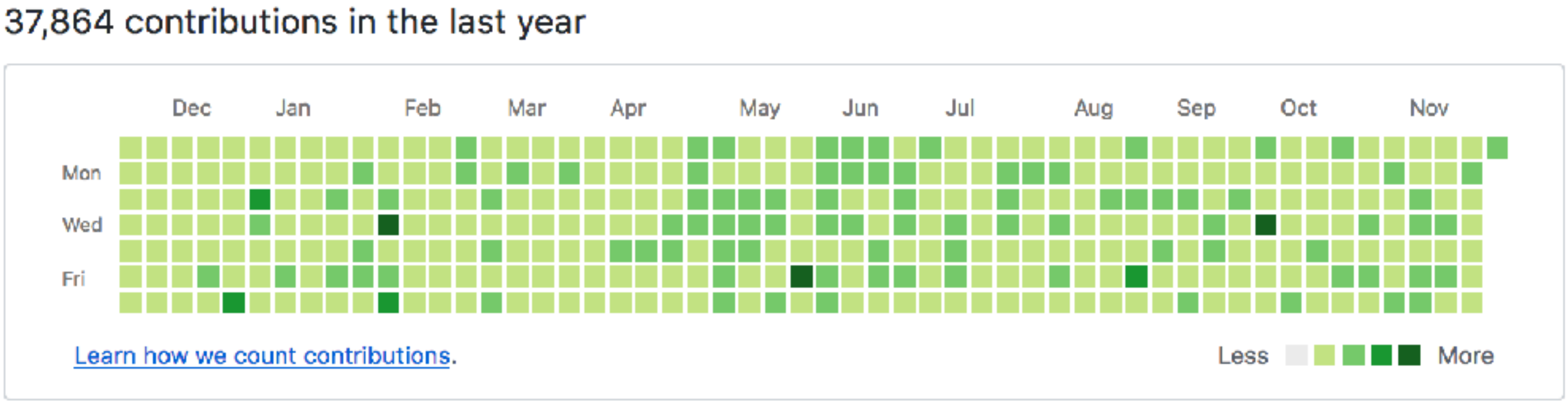
@bors r+

bors: Mandatory homage

The Last of the Masters

*The title character, **Bors, a 200-year-old "government integration robot"**—and the last in existence—awakens after a routine maintenance check to learn that his motor system is in a state of decline. An artificially intelligent machine who displays a degree of emotion and even psychological complexity, he is informed by Fowler, a personal mechanic, that his body has begun to break down due to age.*

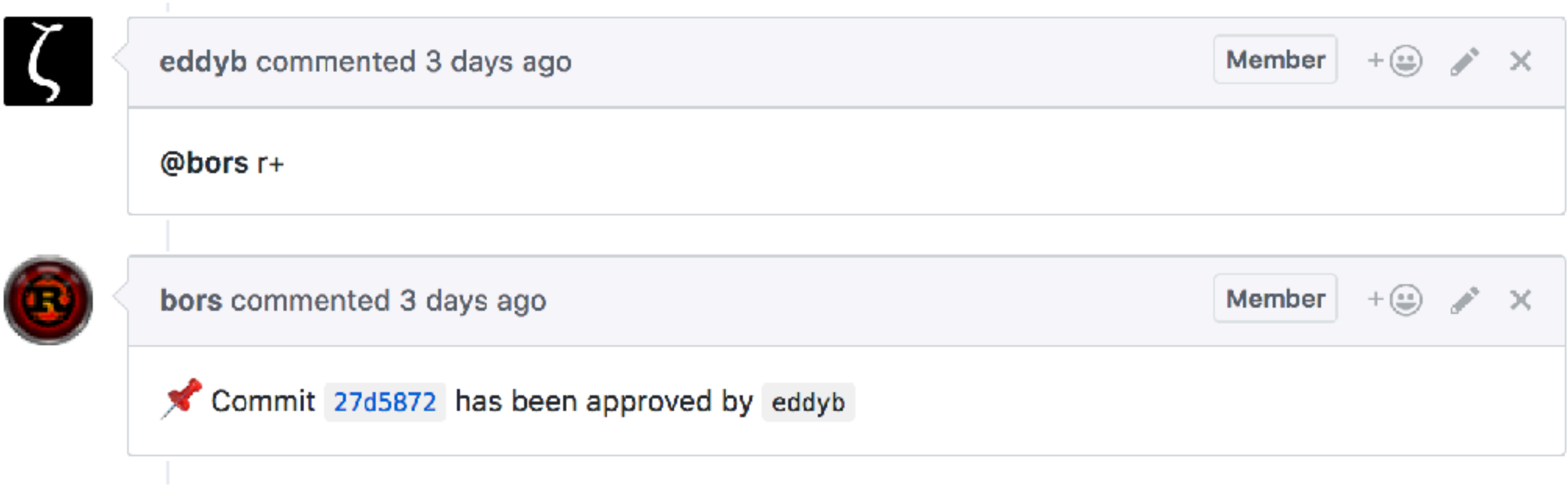
bors never sleeps



37,864 contributions in the last year

Learn how we count contributions.

bors in action



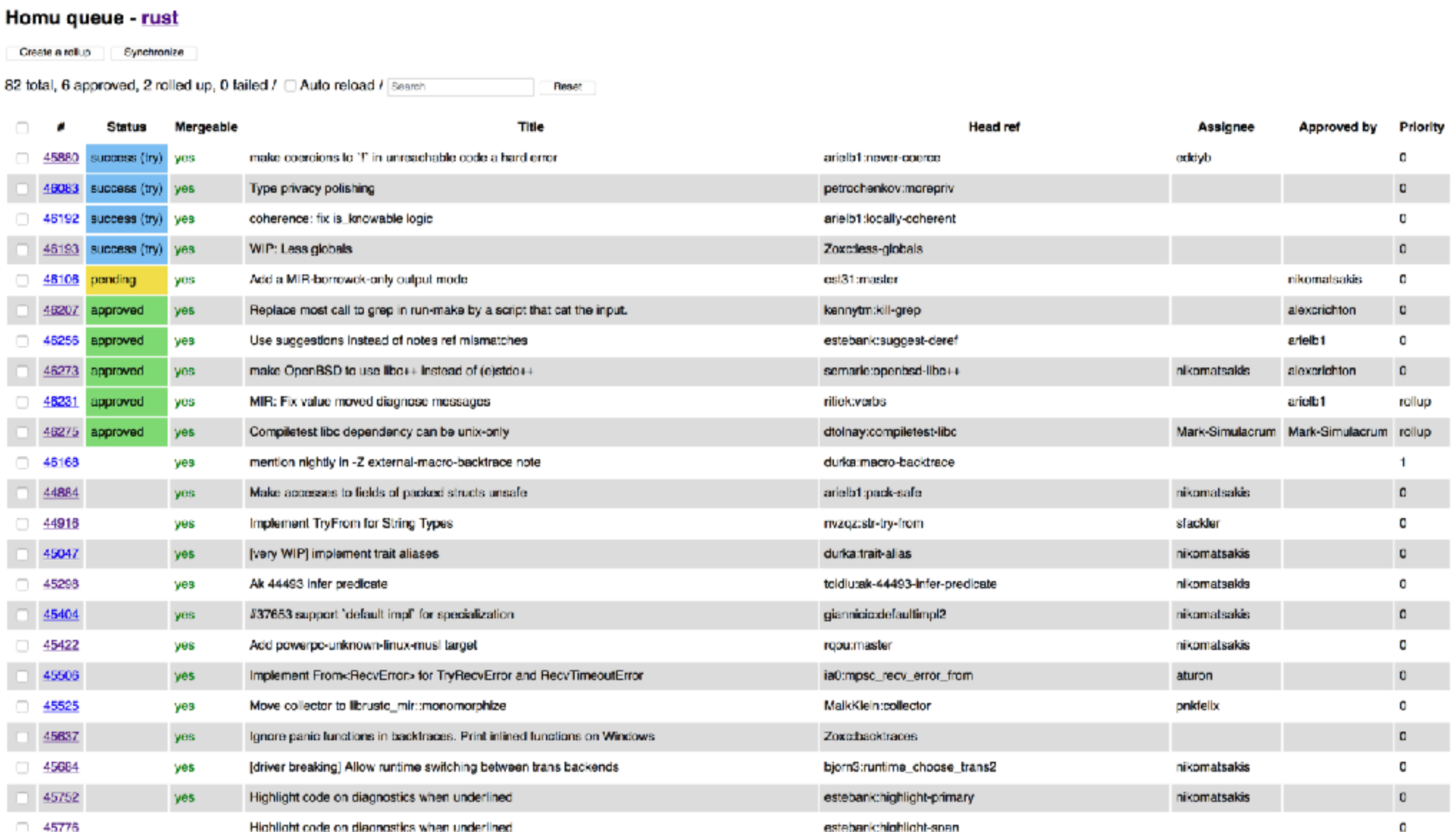
eddyb commented 3 days ago

@bors r+

bors commented 3 days ago

📌 Commit 27d5872 has been approved by eddyb

bors in action

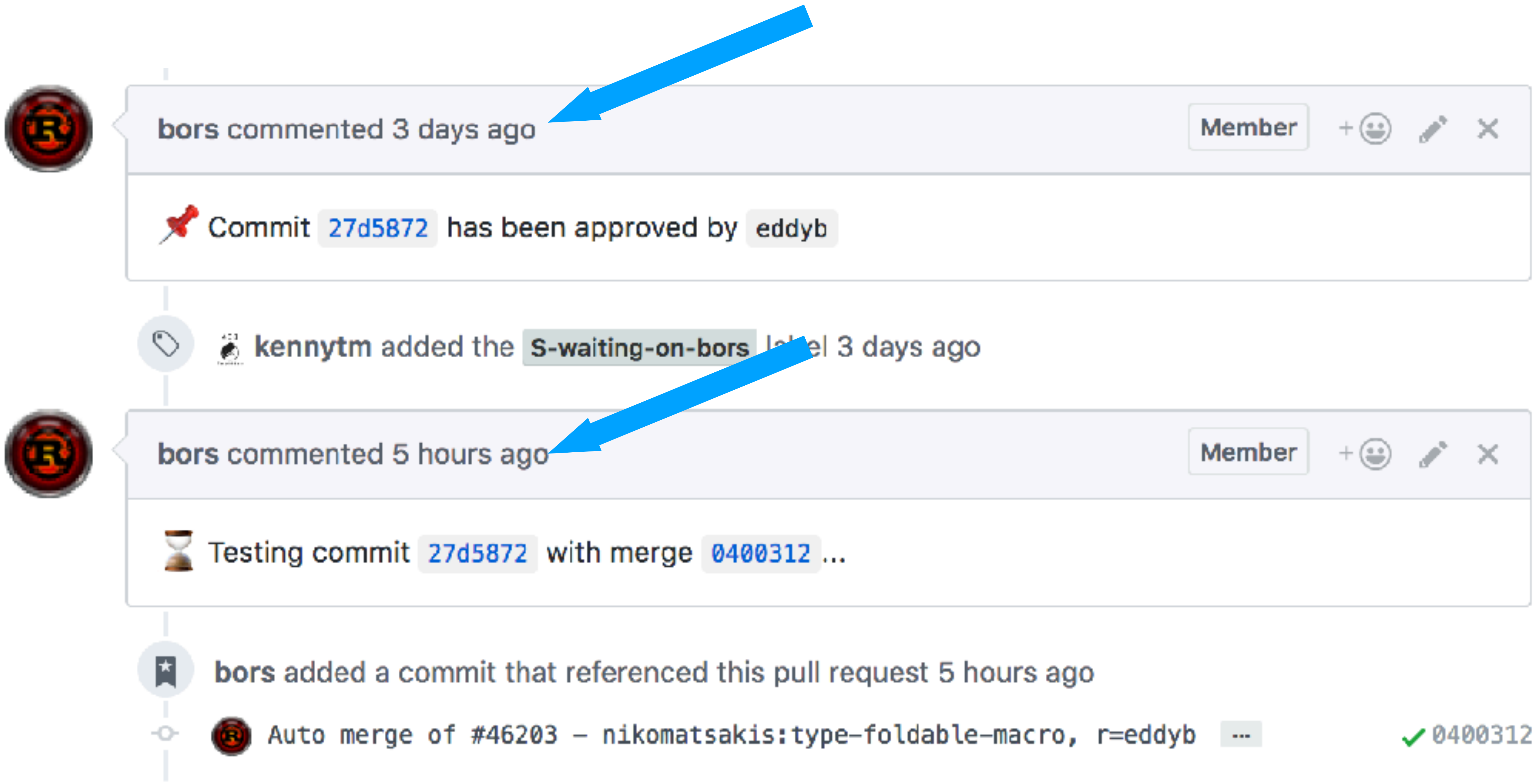


Homu queue - rust

82 total, 6 approved, 2 rolled up, 0 failed / ☐ Auto reload /

☐	#	Status	Mergeable	Title	Head ref	Assignee	Approved by	Priority
☐	45880	success (try)	yes	make coercions to '!' in unreachable code a hard error	arielb1:never-coerce	eddyb		0
☐	46083	success (try)	yes	Type privacy polishing	petrochenkov:morepriv			0
☐	46192	success (try)	yes	coherence: fix is_knowable logic	arielb1:locally-coherent			0
☐	46193	success (try)	yes	WIP: Less globals	Zoxc:less-globals			0
☐	46108	pending	yes	Add a MIR-borrowck-only output mode	est31:master		nikomatsakis	0
☐	46207	approved	yes	Replace most call to grep in run-make by a script that cat the input.	kennytm:kill-grep		alexcrichton	0
☐	46255	approved	yes	Use suggestions instead of notes ref mismatches	estebank:suggest-deref		arielb1	0
☐	46273	approved	yes	make OpenBSD to use libc++ instead of (e)stdc++	semarie:openbsd-libc++	nikomatsakis	alexcrichton	0
☐	46231	approved	yes	MIR: Fix value moved diagnose messages	ritiek:verbs		arielb1	rollup
☐	46275	approved	yes	Compiletest libc dependency can be unix-only	dtolnay:compiletest-libc	Mark-Simulacrum	Mark-Simulacrum	rollup
☐	46168		yes	mention nightly in -Z external-macro-backtrace note	durka:macro-backtrace			1
☐	44884		yes	Make accesses to fields of packed structs unsafe	arielb1:pack-safe	nikomatsakis		0
☐	44918		yes	Implement TryFrom for String Types	mvzqz:str-try-from	sfackler		0
☐	45047		yes	[very WIP] implement trait aliases	durka:trait-alias	nikomatsakis		0
☐	45298		yes	Ak 44493 infer predicate	toidiu:ak-44493-infer-predicate	nikomatsakis		0
☐	45404		yes	#37653 support `default impl` for specialization	giannicic:defaultimpl2	nikomatsakis		0
☐	45422		yes	Add powerpc-unknown-linux-musl target	rqou:master	nikomatsakis		0
☐	45506		yes	Implement From<RecvError> for TryRecvError and RecvTimeoutError	ia0:mpsc_recv_error_from	aturon		0
☐	45525		yes	Move collector to librustc_mir::monomorphize	MaikKlein:collector	pnkfelix		0
☐	45637		yes	Ignore panic functions in backtraces. Print inlined functions on Windows	Zoxc:backtraces			0
☐	45684		yes	[driver breaking] Allow runtime switching between trans backends	bjorn3:runtime_choose_trans2	nikomatsakis		0
☐	45752		yes	Highlight code on diagnostics when underlined	estebank:highlight-primary	nikomatsakis		0
☐	45776			Highlight code on diagnostics when underlined	estebank:highlight-span			0

bors in action



bors commented 3 days ago

📌 Commit 27d5872 has been approved by eddyb

kennytm added the S-waiting-on-bors label 3 days ago

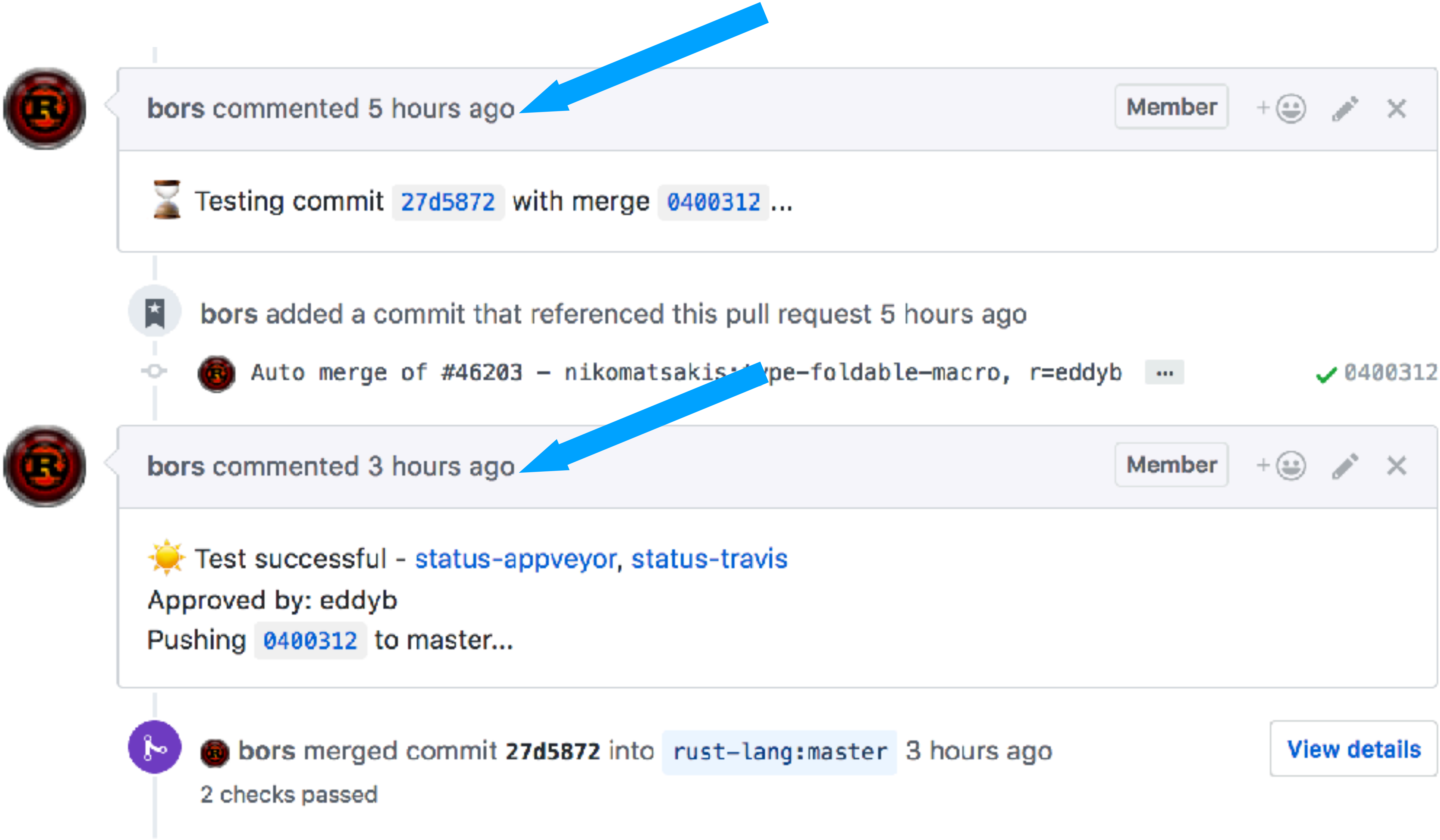
bors commented 5 hours ago

⌛ Testing commit 27d5872 with merge 0400312...

bors added a commit that referenced this pull request 5 hours ago

Auto merge of #46203 - nikomatsakis:type-foldable-macro, r=eddyb ··· ✓ 0400312

bors in action



bors commented 5 hours ago

⌛ Testing commit `27d5872` with merge `0400312`...

bors added a commit that referenced this pull request 5 hours ago

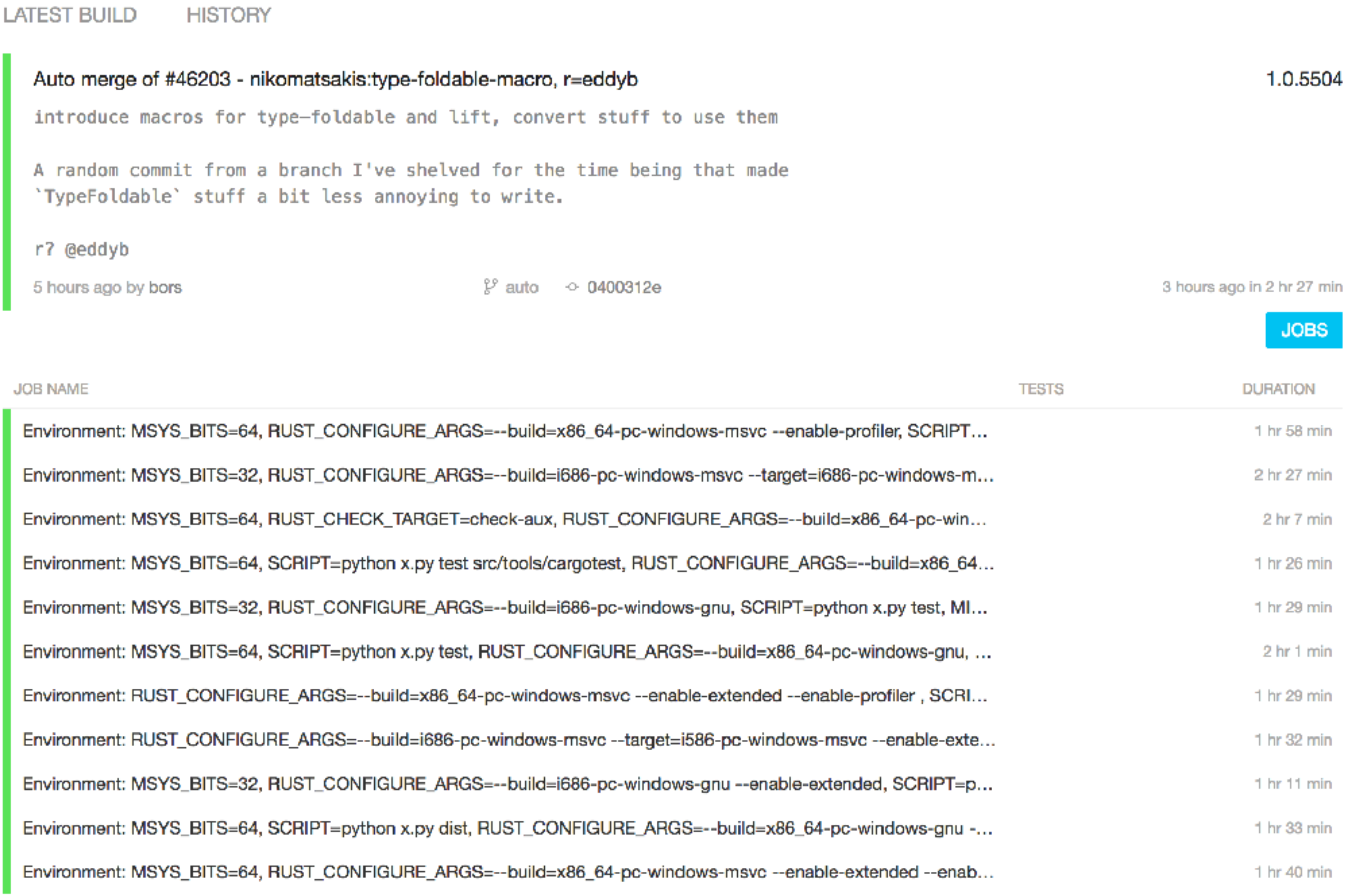
`Auto merge of #46203 - nikomatsakis:type-foldable-macro, r=eddyb` ··· ✓ 0400312

bors commented 3 hours ago

☀️ Test successful - status-appveyor, status-travis
Approved by: eddyb
Pushing `0400312` to master...

bors merged commit **27d5872** into `rust-lang:master` 3 hours ago

2 checks passed



Auto merge of #46203 - nikomatsakis:type-foldable-macro, r=eddyb 1.0.5504

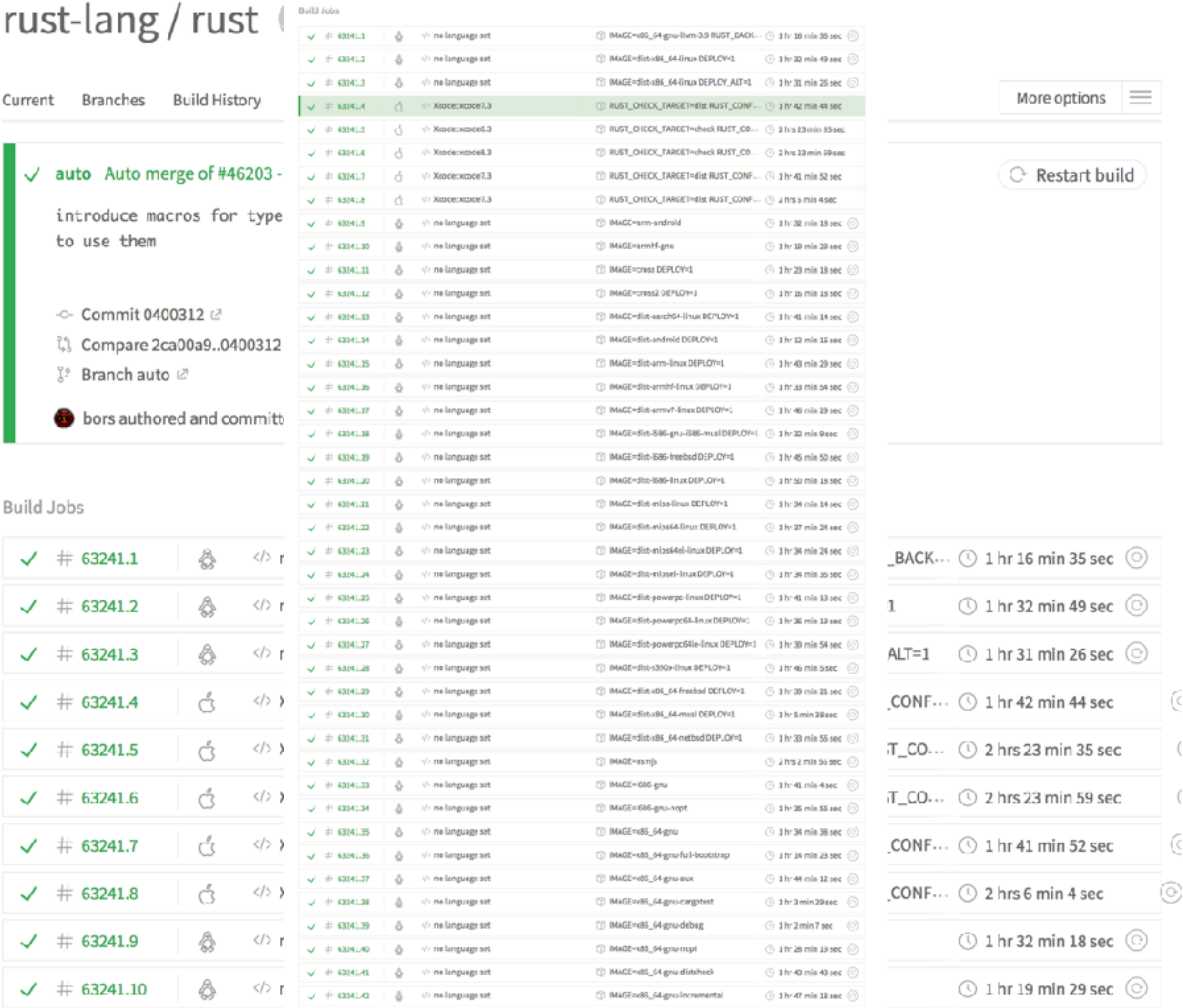
```
introduce macros for type-foldable and lift, convert stuff to use them

A random commit from a branch I've shelved for the time being that made
`TypeFoldable` stuff a bit less annoying to write.

r? @eddyb
```

5 hours ago by bors auto 0400312e 3 hours ago in 2 hr 27 min

JOB NAME	TESTS	DURATION
Environment: MSYS_BITS=64, RUST_CONFIGURE_ARGS=--build=x86_64-pc-windows-msvc --enable-profiler, SCRIPT…		1 hr 58 min
Environment: MSYS_BITS=32, RUST_CONFIGURE_ARGS=--build=i686-pc-windows-msvc --target=i686-pc-windows-m…		2 hr 27 min
Environment: MSYS_BITS=64, RUST_CHECK_TARGET=check-aux, RUST_CONFIGURE_ARGS=--build=x86_64-pc-win…		2 hr 7 min
Environment: MSYS_BITS=64, SCRIPT=python x.py test src/tools/cargotest, RUST_CONFIGURE_ARGS=--build=x86_64…		1 hr 26 min
Environment: MSYS_BITS=32, RUST_CONFIGURE_ARGS=--build=i686-pc-windows-gnu, SCRIPT=python x.py test, MI…		1 hr 29 min
Environment: MSYS_BITS=64, SCRIPT=python x.py test, RUST_CONFIGURE_ARGS=--build=x86_64-pc-windows-gnu, …		2 hr 1 min
Environment: RUST_CONFIGURE_ARGS=--build=x86_64-pc-windows-msvc --enable-extended --enable-profiler , SCRI…		1 hr 29 min
Environment: RUST_CONFIGURE_ARGS=--build=i686-pc-windows-msvc --target=i586-pc-windows-msvc --enable-exte…		1 hr 32 min
Environment: MSYS_BITS=32, RUST_CONFIGURE_ARGS=--build=i686-pc-windows-gnu --enable-extended, SCRIPT=p…		1 hr 11 min
Environment: MSYS_BITS=64, SCRIPT=python x.py dist, RUST_CONFIGURE_ARGS=--build=x86_64-pc-windows-gnu -…		1 hr 33 min
Environment: MSYS_BITS=64, RUST_CONFIGURE_ARGS=--build=x86_64-pc-windows-msvc --enable-extended --enab…		1 hr 40 min



rust-lang / rust

Current Branches Build History

More options

✓ auto Auto merge of #46203 -

introduce macros for type
to use them

Commit 0400312
Compare 2ca00a9..0400312
Branch auto

bors authored and committ

Restart build

Build Jobs

✓	# 63241.1	</> r	_BACK...	1 hr 16 min 35 sec
✓	# 63241.2	</> r	1	1 hr 32 min 49 sec
✓	# 63241.3	</> r	ALT=1	1 hr 31 min 26 sec
✓	# 63241.4	</> X	_CONF...	1 hr 42 min 44 sec
✓	# 63241.5	</> X	T_CO...	2 hrs 23 min 35 sec
✓	# 63241.6	</> X	T_CO...	2 hrs 23 min 59 sec
✓	# 63241.7	</> X	_CONF...	1 hr 41 min 52 sec
✓	# 63241.8	</> X	_CONF...	2 hrs 6 min 4 sec
✓	# 63241.9	</> r		1 hr 32 min 18 sec
✓	# 63241.10	</> r		1 hr 19 min 29 sec

Build Jobs

✓	# 63241.1	</> no language set	IMAGE=x86_64-gnu-llvm-3.9 RUST_BACK...	1 hr 16 min 35 sec
✓	# 63241.2	</> no language set	IMAGE=dist-x86_64-linux DEPLOY=1	1 hr 32 min 49 sec
✓	# 63241.3	</> no language set	IMAGE=dist-x86_64-linux DEPLOY_ALT=1	1 hr 31 min 26 sec
✓	# 63241.4	</> Xcode:xcode7.3	RUST_CHECK_TARGET=dist RUST_CONF...	1 hr 42 min 44 sec
✓	# 63241.5	</> Xcode:xcode8.3	RUST_CHECK_TARGET=check RUST_CO...	2 hrs 23 min 35 sec
✓	# 63241.6	</> Xcode:xcode8.3	RUST_CHECK_TARGET=check RUST_CO...	2 hrs 23 min 59 sec
✓	# 63241.7	</> Xcode:xcode7.3	RUST_CHECK_TARGET=dist RUST_CONF...	1 hr 41 min 52 sec
✓	# 63241.8	</> Xcode:xcode7.3	RUST_CHECK_TARGET=dist RUST_CONF...	2 hrs 6 min 4 sec
✓	# 63241.9	</> no language set	IMAGE=arm-android	1 hr 32 min 18 sec
✓	# 63241.10	</> no language set	IMAGE=armhf-gnu	1 hr 19 min 29 sec
✓	# 63241.11	</> no language set	IMAGE=cross DEPLOY=1	1 hr 23 min 18 sec
✓	# 63241.12	</> no language set	IMAGE=cross2 DEPLOY=1	1 hr 16 min 15 sec
✓	# 63241.13	</> no language set	IMAGE=dist-aarch64-linux DEPLOY=1	1 hr 41 min 14 sec
✓	# 63241.14	</> no language set	IMAGE=dist-android DEPLOY=1	1 hr 12 min 16 sec
✓	# 63241.15	</> no language set	IMAGE=dist-arm-linux DEPLOY=1	1 hr 43 min 29 sec
✓	# 63241.16	</> no language set	IMAGE=dist-armhf-linux DEPLOY=1	1 hr 33 min 54 sec
✓	# 63241.17	</> no language set	IMAGE=dist-armv7-linux DEPLOY=1	1 hr 46 min 29 sec
✓	# 63241.18	</> no language set	IMAGE=dist-i586-gnu-i586-musl DEPLOY=1	1 hr 22 min 9 sec
✓	# 63241.19	</> no language set	IMAGE=dist-i686-freebsd DEPLOY=1	1 hr 45 min 50 sec
✓	# 63241.20	</> no language set	IMAGE=dist-i686-linux DEPLOY=1	1 hr 50 min 15 sec
✓	# 63241.21	</> no language set	IMAGE=dist-mips-linux DEPLOY=1	1 hr 34 min 14 sec
✓	# 63241.22	</> no language set	IMAGE=dist-mips64-linux DEPLOY=1	1 hr 27 min 24 sec
✓	# 63241.23	</> no language set	IMAGE=dist-mips64el-linux DEPLOY=1	1 hr 34 min 24 sec
✓	# 63241.24	</> no language set	IMAGE=dist-mipsel-linux DEPLOY=1	1 hr 34 min 35 sec
✓	# 63241.25	</> no language set	IMAGE=dist-powerpc-linux DEPLOY=1	1 hr 41 min 13 sec
✓	# 63241.26	</> no language set	IMAGE=dist-powerpc64-linux DEPLOY=1	1 hr 26 min 19 sec
✓	# 63241.27	</> no language set	IMAGE=dist-powerpc64le-linux DEPLOY=1	1 hr 39 min 54 sec
✓	# 63241.28	</> no language set	IMAGE=dist-s390x-linux DEPLOY=1	1 hr 46 min 5 sec
✓	# 63241.29	</> no language set	IMAGE=dist-x86_64-freebsd DEPLOY=1	1 hr 39 min 21 sec
✓	# 63241.30	</> no language set	IMAGE=dist-x86_64-musl DEPLOY=1	1 hr 5 min 28 sec
✓	# 63241.31	</> no language set	IMAGE=dist-x86_64-netbsd DEPLOY=1	1 hr 33 min 55 sec
✓	# 63241.32	</> no language set	IMAGE=asmjs	2 hrs 2 min 56 sec
✓	# 63241.33	</> no language set	IMAGE=i686-gnu	1 hr 41 min 4 sec
✓	# 63241.34	</> no language set	IMAGE=i686-gnu-nopt	1 hr 26 min 55 sec
✓	# 63241.35	</> no language set	IMAGE=x86_64-gnu	1 hr 34 min 38 sec
✓	# 63241.36	</> no language set	IMAGE=x86_64-gnu-full-bootstrap	1 hr 16 min 23 sec
✓	# 63241.37	</> no language set	IMAGE=x86_64-gnu-aux	1 hr 44 min 12 sec
✓	# 63241.38	</> no language set	IMAGE=x86_64-gnu-cargotest	1 hr 3 min 29 sec
✓	# 63241.39	</> no language set	IMAGE=x86_64-gnu-debug	1 hr 2 min 7 sec
✓	# 63241.40	</> no language set	IMAGE=x86_64-gnu-nopt	1 hr 28 min 19 sec
✓	# 63241.41	</> no language set	IMAGE=x86_64-gnu-distcheck	1 hr 40 min 43 sec
✓	# 63241.42	</> no language set	IMAGE=x86_64-gnu-incremental	1 hr 47 min 18 sec

Release trains

- Nightly, beta, stable channels
- Automatic nightlies each night
- Stable/beta updated once every 6 weeks
- Stable/beta receive bug fixes with “backports”

Infrastructure of Rust

- Continuous Integration
- {www,doc,play}.rust-lang.org
- crates.io
- Rapid release cycle
- “Dealing with GitHub”
- AWS services, CDNs, storage, etc

Tooling

Infrastructure

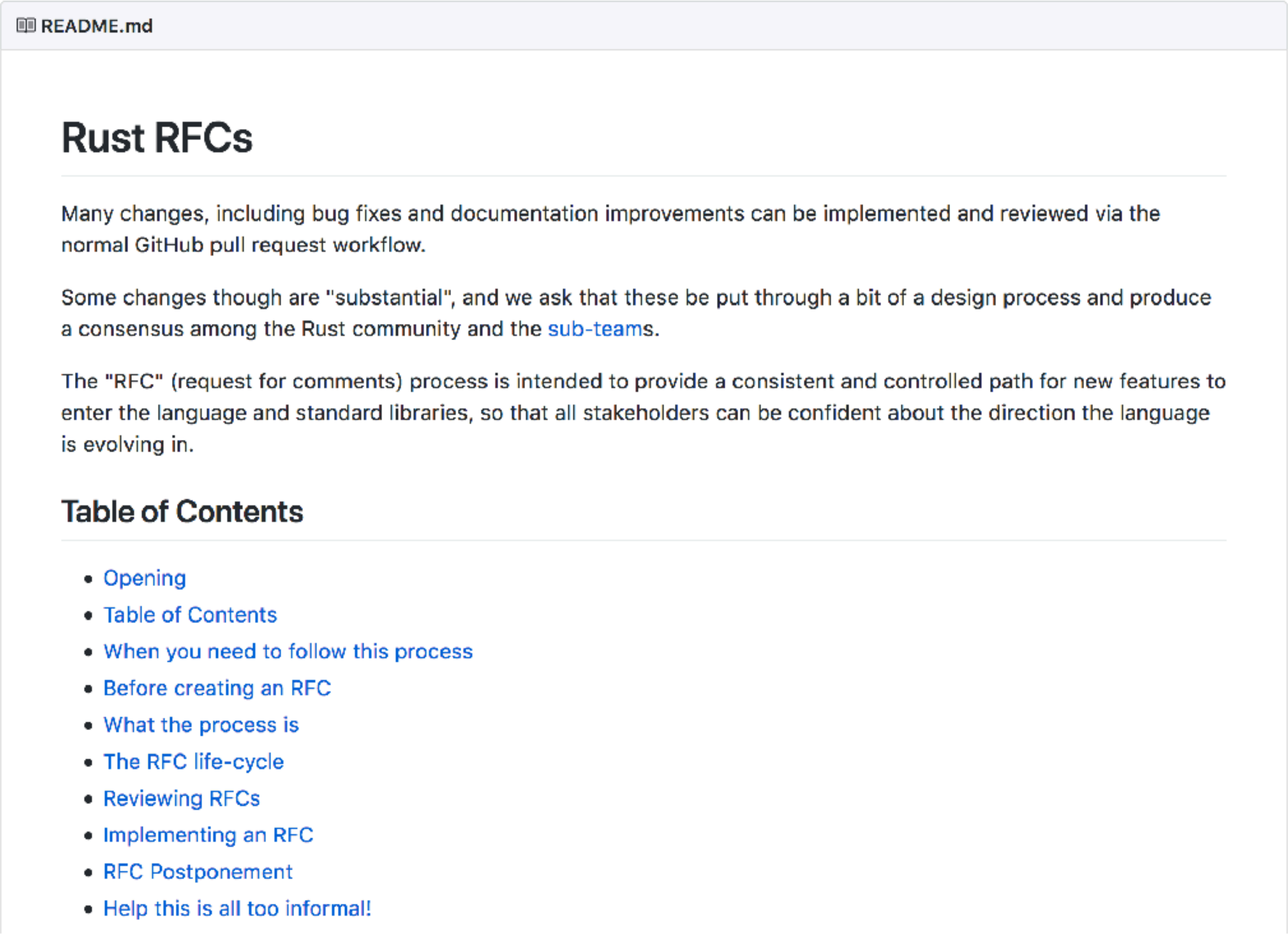
Community

We get it, it’s not 1970

- Development does not happen in isolation any more
- Everyone’s got thoughts (often great ones!)
- Early stage projects live and die by their communities

Rust’s Community

- Governance of Rust itself
- RFC process
- Internals/users forum, IRC
- Community team
- Conferences



Rust RFCs

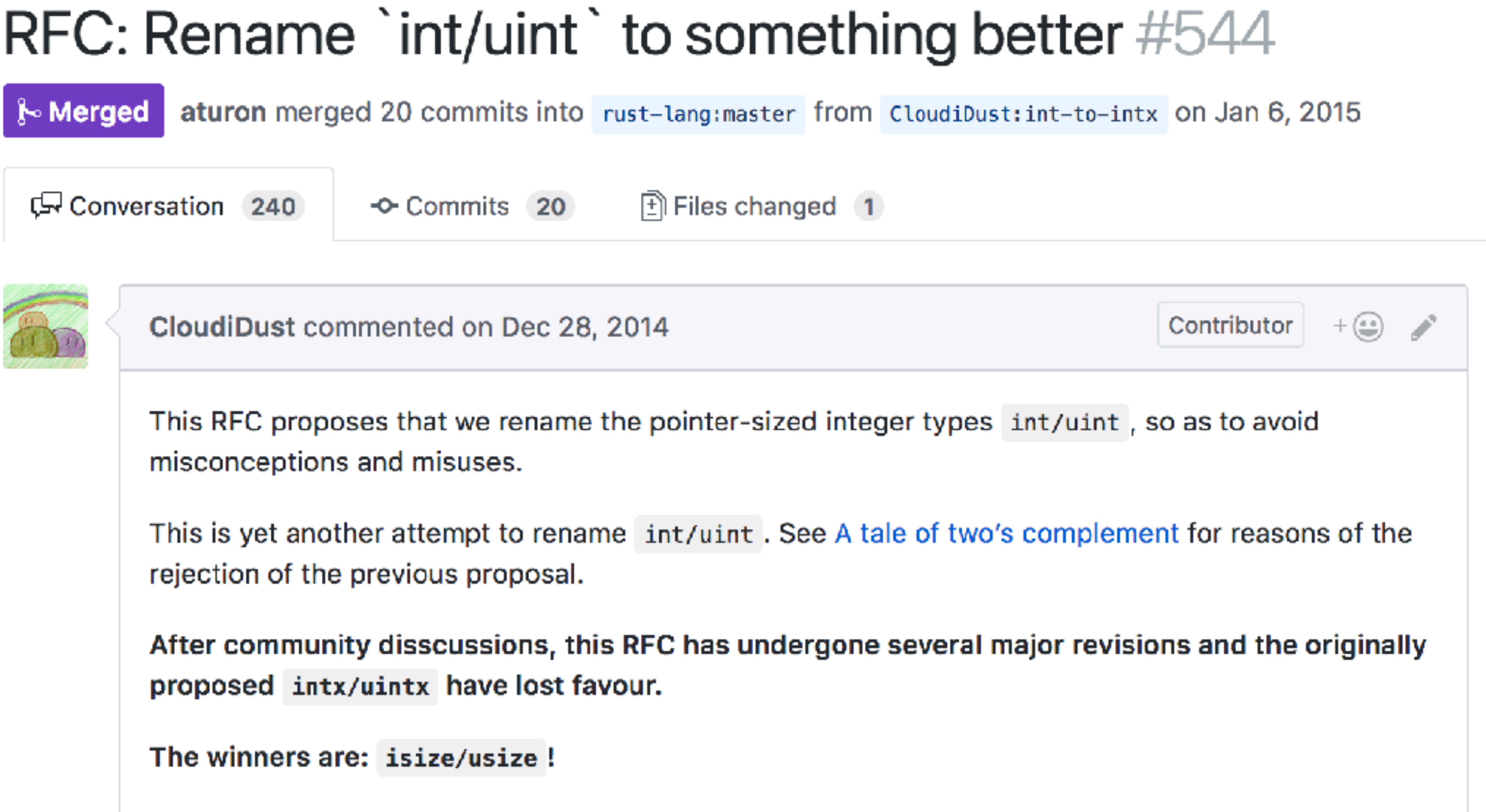
Many changes, including bug fixes and documentation improvements can be implemented and reviewed via the normal GitHub pull request workflow.

Some changes though are "substantial", and we ask that these be put through a bit of a design process and produce a consensus among the Rust community and the sub-teams.

The "RFC" (request for comments) process is intended to provide a consistent and controlled path for new features to enter the language and standard libraries, so that all stakeholders can be confident about the direction the language is evolving in.

Table of Contents

To `int` or not to `int`?



RFC: Rename `int/uint` to something better #544

aturon merged 20 commits into `rust-lang:master` from `CloudiDust:int-to-intx` on Jan 6, 2015

CloudiDust commented on Dec 28, 2014

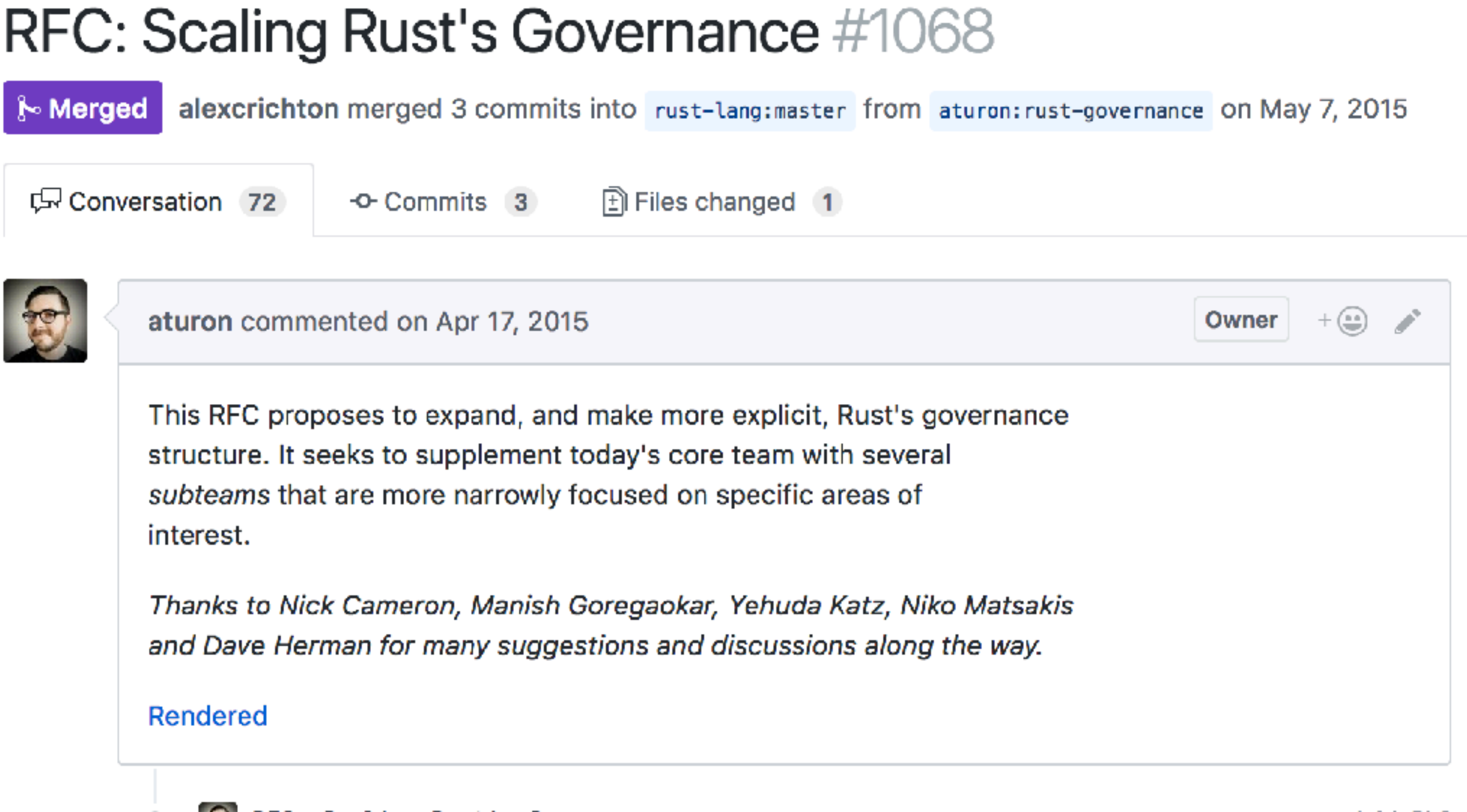
This RFC proposes that we rename the pointer-sized integer types `int/uint`, so as to avoid misconceptions and misuses.

This is yet another attempt to rename `int/uint`. See A tale of two's complement for reasons of the rejection of the previous proposal.

After community disscussions, this RFC has undergone several major revisions and the originally proposed `intx/uintx` have lost favour.

The winners are: `isize/usize`!

Reducing entropy



RFC: Scaling Rust's Governance #1068

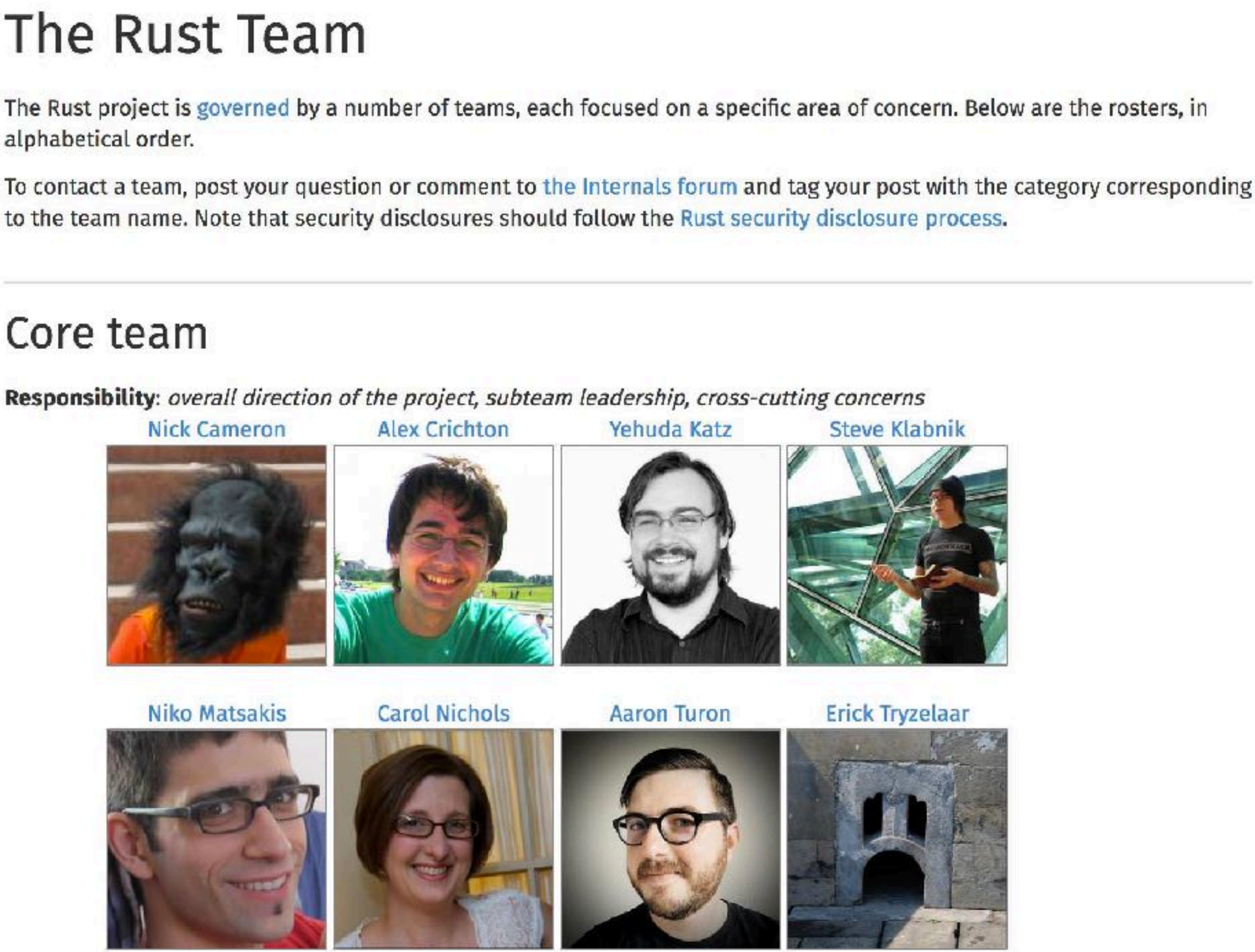
Merged alexcrichton merged 3 commits into `rust-lang:master` from `aturon:rust-governance` on May 7, 2015

aturon commented on Apr 17, 2015

This RFC proposes to expand, and make more explicit, Rust's governance structure. It seeks to supplement today's core team with several *subteams* that are more narrowly focused on specific areas of interest.

Thanks to Nick Cameron, Manish Goregaokar, Yehuda Katz, Niko Matsakis and Dave Herman for many suggestions and discussions along the way.

Rendered



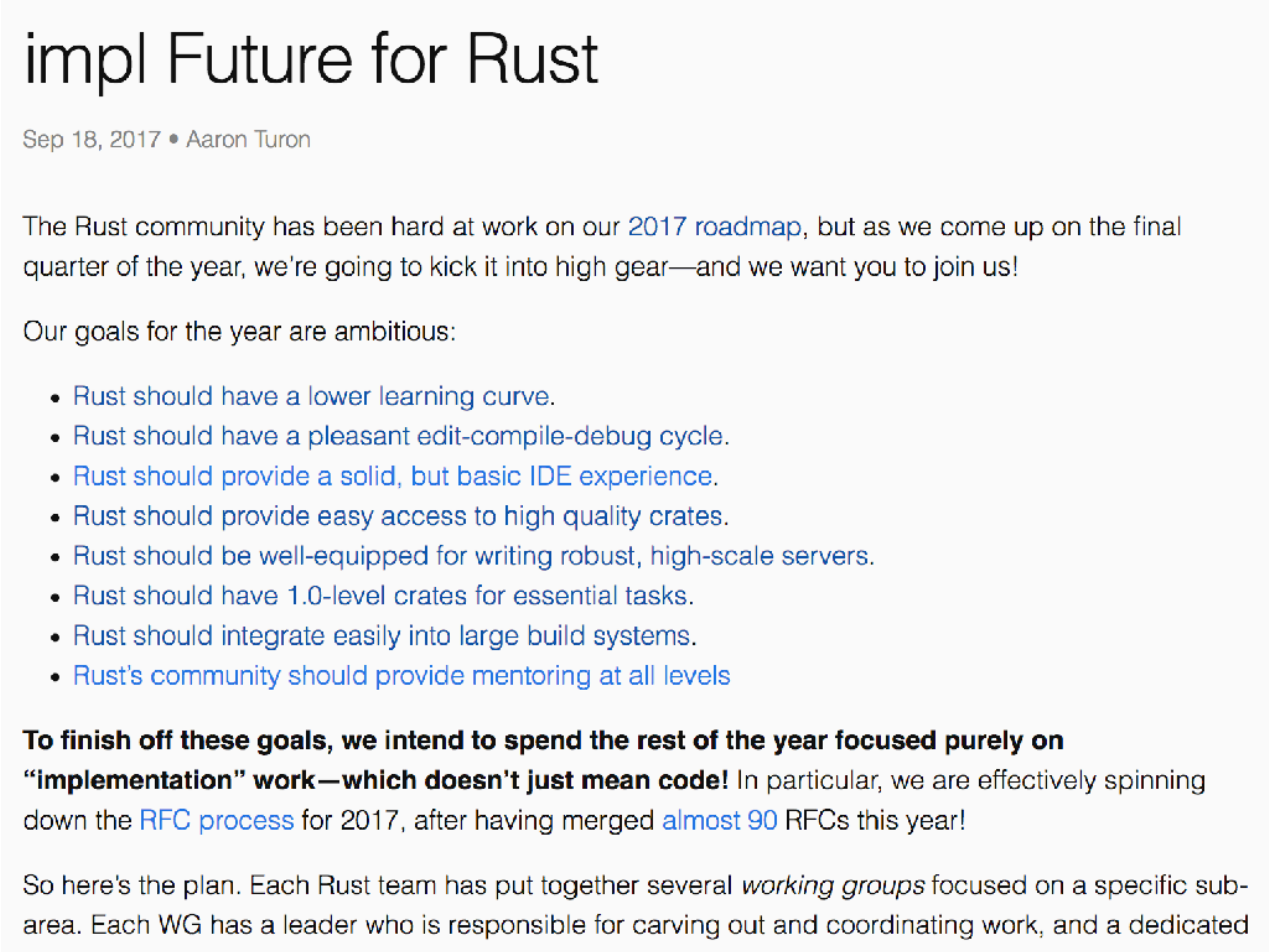
The Rust Team

The Rust project is governed by a number of teams, each focused on a specific area of concern. Below are the rosters, in alphabetical order.

To contact a team, post your question or comment to the Internals forum and tag your post with the category corresponding to the team name. Note that security disclosures should follow the Rust security disclosure process.

Core team

Responsibility: *overall direction of the project, subteam leadership, cross-cutting concerns*



impl Future for Rust

Sep 18, 2017 • Aaron Turon

The Rust community has been hard at work on our 2017 roadmap, but as we come up on the final quarter of the year, we're going to kick it into high gear—and we want you to join us!

Our goals for the year are ambitious:

- Rust should have a lower learning curve.
- Rust should have a pleasant edit-compile-debug cycle.
- Rust should provide a solid, but basic IDE experience.
- Rust should provide easy access to high quality crates.
- Rust should be well-equipped for writing robust, high-scale servers.
- Rust should have 1.0-level crates for essential tasks.
- Rust should integrate easily into large build systems.
- Rust's community should provide mentoring at all levels

To finish off these goals, we intend to spend the rest of the year focused purely on "implementation" work—which doesn't just mean code! In particular, we are effectively spinning down the RFC process for 2017, after having merged almost 90 RFCs this year!

So here's the plan. Each Rust team has put together several *working groups* focused on a specific sub-area. Each WG has a leader who is responsible for carving out and coordinating work, and a dedicated