

# Rust governance

Kuba Beránek



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

**Kuba Beránek**





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

# Kuba Beránek

- PhD student, teacher @ VSB-TUO





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

# Kuba Beránek

- PhD student, teacher @ VSB-TUO
- Researcher @ IT4Innovations





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

# Kuba Beránek

- PhD student, teacher @ VSB-TUO
- Researcher @ IT4Innovations
- HPC, distributed systems, code optimization, machine learning,...





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

# Kuba Beránek

- PhD student, teacher @ VSB-TUO
- Researcher @ IT4Innovations
- HPC, distributed systems, code optimization, machine learning,...
- Rust user & contributor



# Compiler performance working group

Improving rustc compilation performance (build times)

## Members



**Mark Rousskov**

GitHub: [Mark-Simulacrum](#)

Team leader



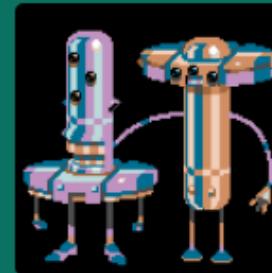
**Jakub Beránek**

GitHub: [Kobzol](#)



**Rémy Rakic**

GitHub: [lqd](#)



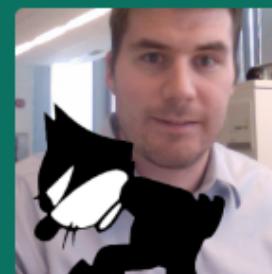
**Michael Woerister**

GitHub: [michaelwoerister](#)



**Nicholas Nethercote**

GitHub: [nnethercote](#)



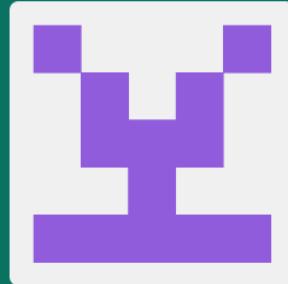
**Felix Klock**

GitHub: [pnkfelix](#)

# Parallel rustc working group

Making parallel compilation the default for rustc

## Members



**Camille Gillot**

GitHub: [cjgillot](#)

Team leader



**Jakub Beránek**

GitHub: [Kobzol](#)



**Sparrow Li**

GitHub: [SparrowLii](#)



**bjorn3**

GitHub: [bjorn3](#)

nek

woerister

[woerister](#)

**Nicholas  
Nethercote**

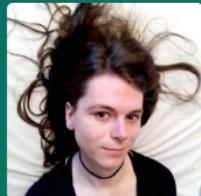
GitHub: [nnethercote](#)

# Parallel Rust working group

## Binary size working group

Improving the binary size of Rust programs and libraries

### Members



**Mara Bos**

GitHub: [m-ou-se](#)

Team leader



**Thom Chiovoloni**

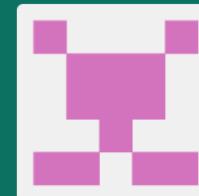
GitHub: [thomcc](#)

Team leader



**David Wood**

GitHub: [davidtwco](#)



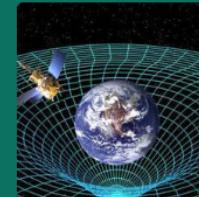
**h1467792822**

GitHub: [h1467792822](#)



**Jakub Beránek**

GitHub: [Kobzol](#)



**Gary Guo**

GitHub: [nbdd0121](#)



**Nicholas Nethercote**

GitHub: [nnethercote](#)



**Wayne Wu**

GitHub: [wain303009](#)



**Jubilee**

GitHub: [workingjubilee](#)

nek

perister  
woerister

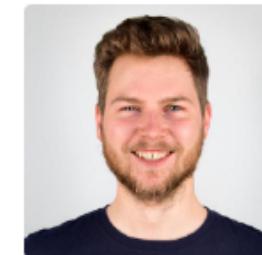
# Parallel Rust working group

## Binary size working group

### Infrastructure team

Managing the infrastructure supporting the Rust project itself, including CI, releases, bots, and metrics

#### Members



**Jan David Nose**

GitHub: [jdno](#)  
Team leader



**Jake Goulding**

GitHub: [shepmaster](#)  
Team leader



**kennytm**

GitHub: [kennytm](#)



**Jakub Beránek**

GitHub: [Kobzol](#)



**Mark Rousskov**

GitHub: [Mark-Simulacrum](#)



**Pietro Albini**

GitHub: [pietroalbini](#)

# **What is governance?**

# **What is governance?**

Decision-making process

# What is governance?

Decision-making process  
designed to evolve <X>

# Governance modes for languages

# Governance modes for languages

- Company-backed - Kotlin, C#

# Governance modes for languages

- Company-backed - Kotlin, C#
- Design by committee - C, C++

# Governance modes for languages

- Company-backed - Kotlin, C#
- Design by committee - C, C++
- BDFL - ex-Python, Ruby

# Governance modes for languages

- Company-backed - Kotlin, C#
- Design by committee - C, C++
- BDFL - ex-Python, Ruby
- Open RFC process - Rust, Python

# Rust governance history



2006

Started as a personal project  
by Graydon Hoare (@Mozilla)

2006

Started as a personal project  
by Graydon Hoare (@Mozilla)

“

I think I named it after fungi...  
that is "over-engineered for survival".

Graydon Hoare ,”



# Project Servo

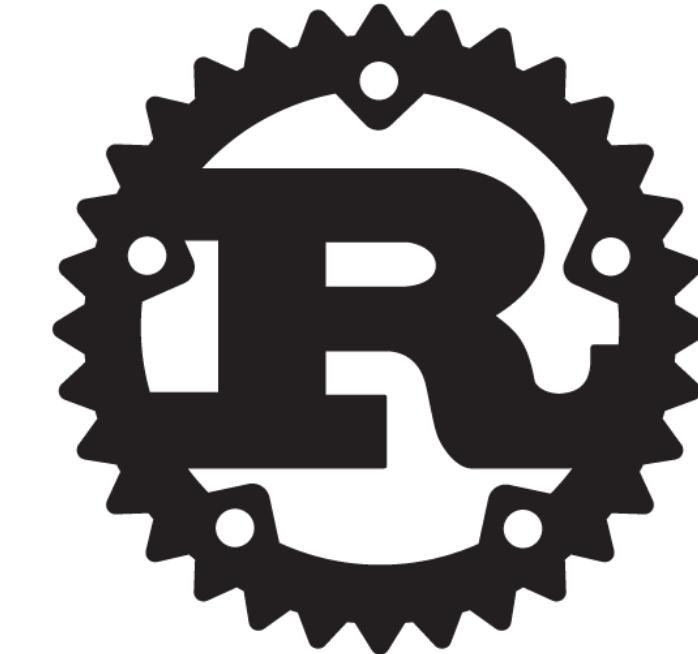
Technology from the past  
come to save the future  
from itself

Mozilla Annual Summit, July 2010  
[<graydon@mozilla.com>](mailto:<graydon@mozilla.com>)

2006

2010

# Project Servo



Technology from the past  
come to save the future  
from itself

Mozilla Annual Summit, July 2010  
[<graydon@mozilla.com>](mailto:graydon@mozilla.com)



Patrick Walton, Niko Matsakis join Graydon



Patrick Walton, Niko Matsakis join Graydon  
Communication on mailing lists, later IRC

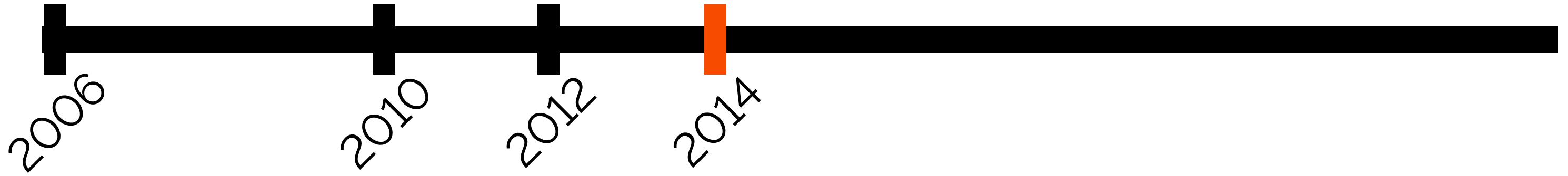


Graydon steps down as BDFL

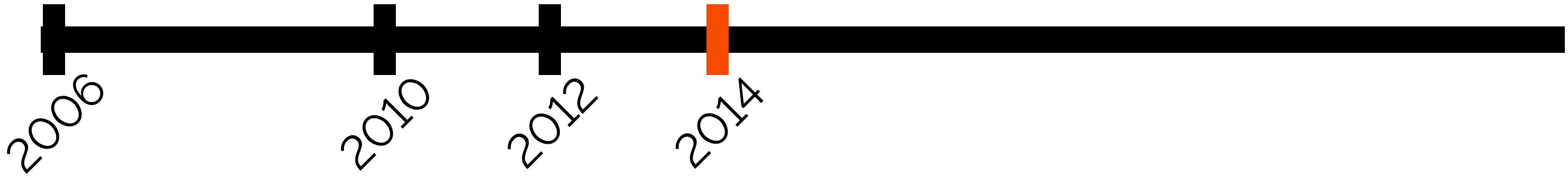


Graydon steps down as BDFL

Core team is created



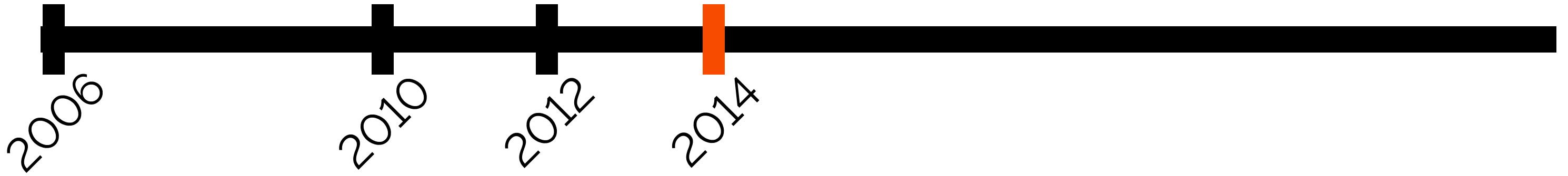
RFC process created (inspired by PEP)



- Start Date: 2014-03-11
- RFC PR: [rust-lang/rfcs#1](#)
- Rust Issue: [rust-lang/rust#8122](#)

## Summary

This is an RFC to make all struct fields private by default. This includes both tuple structs and structural structs.



- Start Date: 2014-03-11
- RFC PR: [rust-lang/rfcs#2](#), [rust-lang/rfcs#6](#)
- Rust Issue: N/A

## Summary

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.



Rust 1.0 released



RFC #1068: introduction of (sub)teams



- Feature Name: not applicable
- Start Date: 2015-02-27
- RFC PR: [rust-lang/rfcs#1068](#)
- Rust Issue: N/A

## Summary

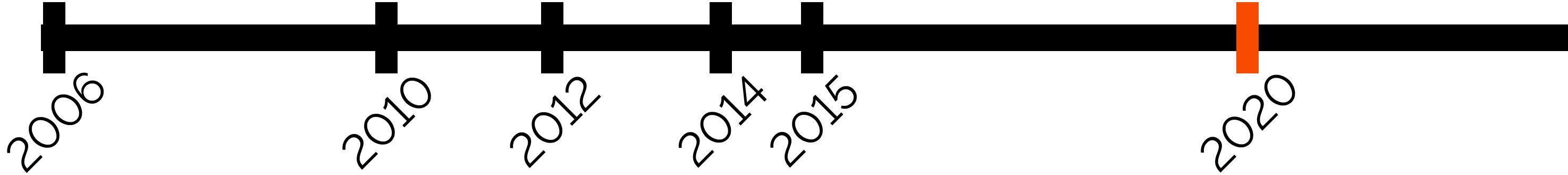
This RFC proposes to expand, and make more explicit, Rust's governance structure. It seeks to supplement today's core team with several *subteams* that are more narrowly focused on specific areas of interest.

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



## Initial subteams:

- Language
- Library
- Compiler
- Tooling and infrastructure
- Moderation

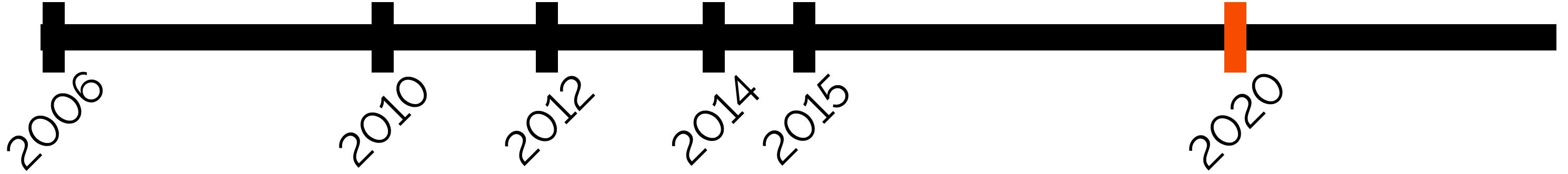


LEADERSHIP

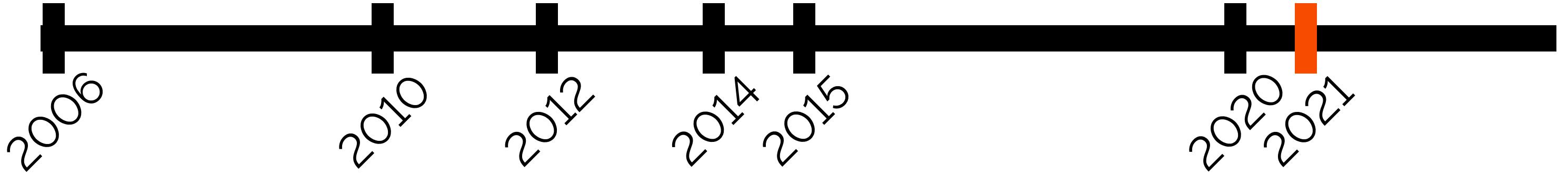
# Changing World, Changing Mozilla

AUGUST 11, 2020

MITCHELL BAKER



Core team plans to create a Rust Foundation

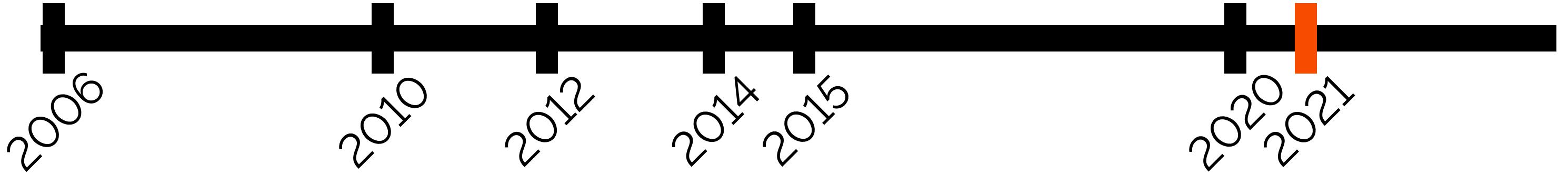


# Hello World!

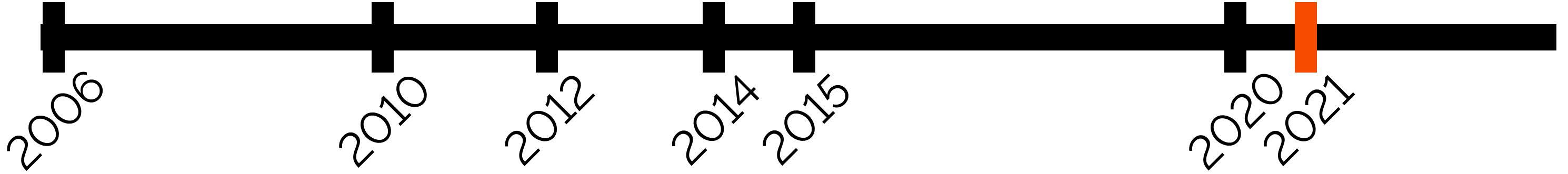
Announcing the Rust Foundation to the World

by Ashley Williams, Interim Executive Director, Core Team Member

08 Feb 2021

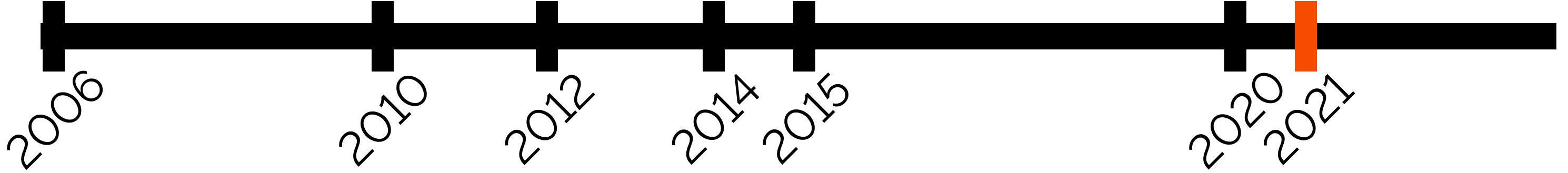


## Rust Foundation goals:



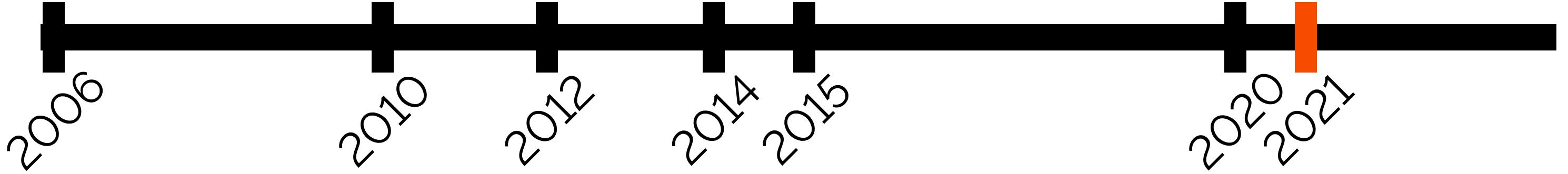
## Rust Foundation goals:

- Create an official Rust entity



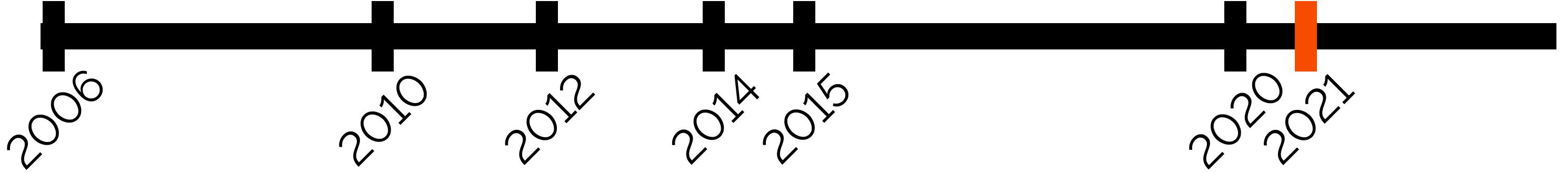
## Rust Foundation goals:

- Create an official Rust entity
- Legal, bank account, domain ownership, ...



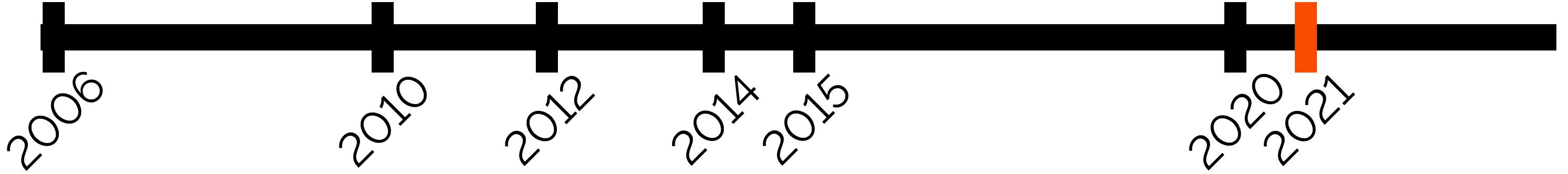
## Rust Foundation goals:

- Create an official Rust entity
- Legal, bank account, domain ownership, ...
- Support maintainers



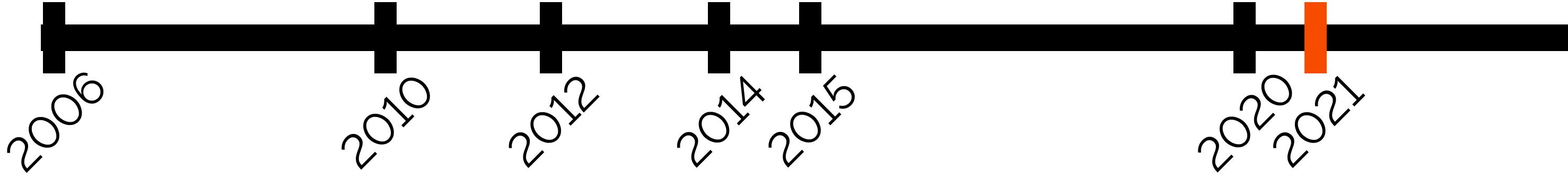
## Rust Foundation goals:

- Create an official Rust entity
- Legal, bank account, domain ownership, ...
- Support maintainers
- Facilitate sponsorship



## Rust Foundation goals:

- Create an official Rust entity
- Legal, bank account, domain ownership, ...
- Support maintainers
- Facilitate sponsorship
- Mozilla, Google, AWS, Microsoft, Huawei



# mod team resignation #671

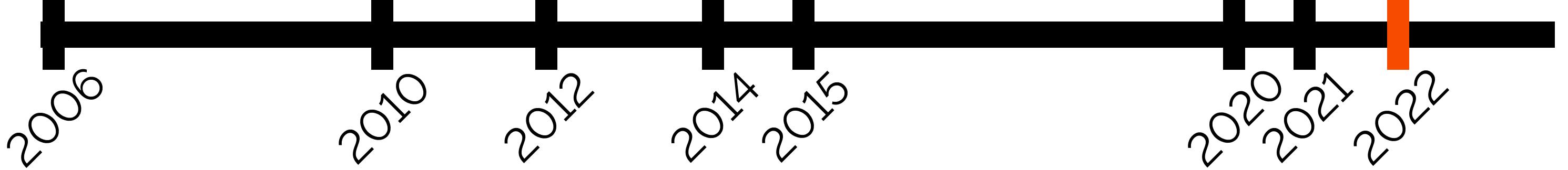
 Merged

aidanhs merged 1 commit into `rust-lang:master` from `BurntSushi:ag/mod-team-update`   
on Nov 22, 2021

Conversation 10 Commits 1 Checks 0 Files changed 1

 BurntSushi commented on Nov 22, 2021 • edited   
Member

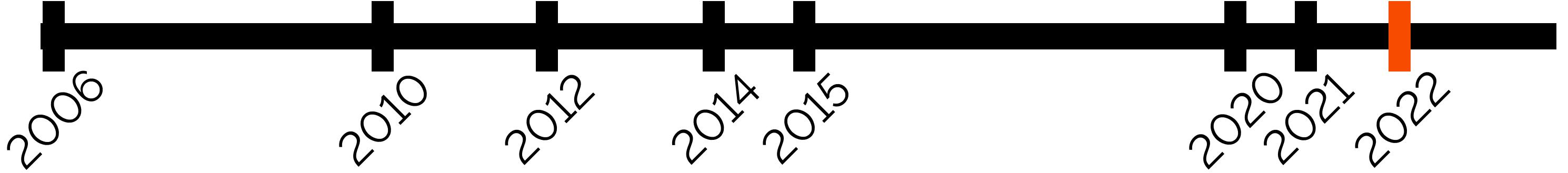
The entire moderation team resigns, effective immediately. This resignation is done in protest of the Core Team placing themselves unaccountable to anyone but themselves.



# Governance Update

May 19, 2022 · Ryan Levick and Mara Bos

Last month, the core team, all the leads of top-level teams, the moderators, and the project representatives on the Rust Foundation board jointly sent out an update to all Rust project members on investigations happening into improvements to the governance of the Rust project.

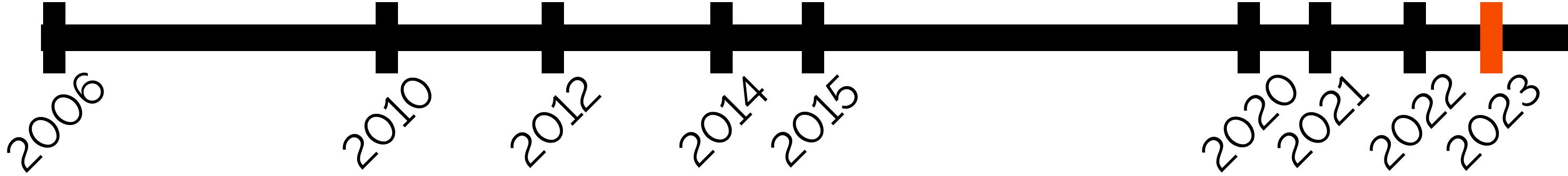


# Governance Update

May 19, 2022 · Ryan Levick and Mara Bos

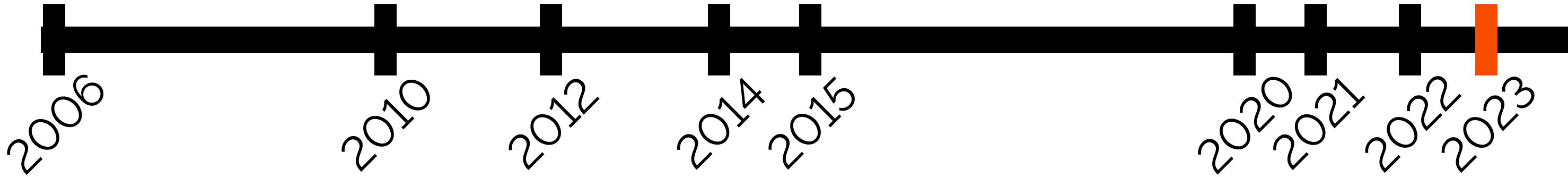
Last month, the core team, all the leads of top-level teams, the moderators, and the project representatives on the Rust Foundation board jointly sent out an update to all Rust project members on investigations happening into improvements to the governance of the Rust project.

Created a private "leadership chat" to solve the issues



**Rust Foundation so sorry for scaring  
the C out of you with trademark  
crackdown talk**

Should have wrapped proposed rules on name and logo use in unsafe {} ?



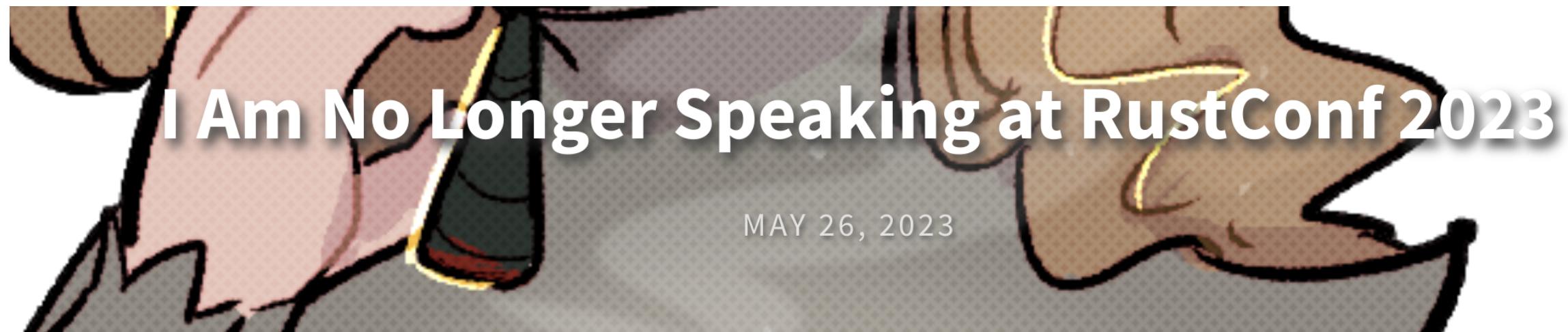
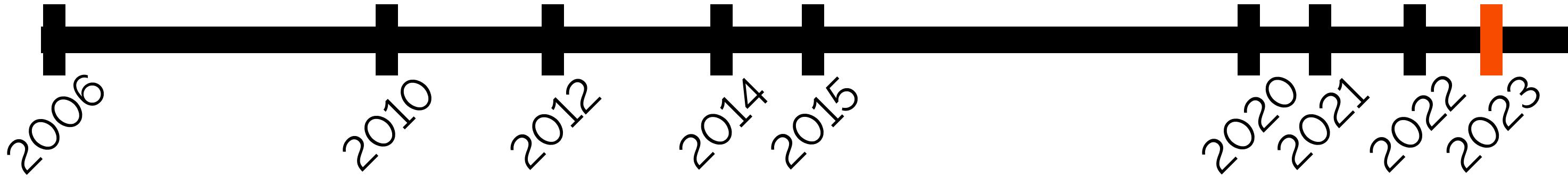
## CrabLang

[Home](#) [Discord](#) [Twitter](#) [GitHub](#)

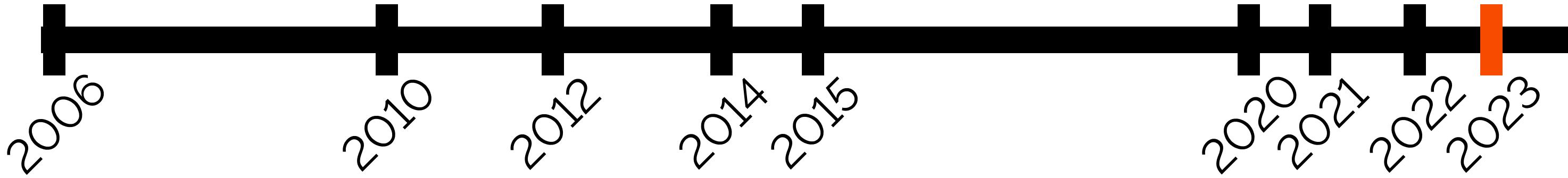
# The Official Home of CrabLang!

A community fork of a language named after a plant fungus. All of the memory-safe features you love, now with 100% less bureaucracy!





I will no longer be speaking at [RustConf 2023](#) about [A \(Possible\) Future for Compile-Time Programming](#).



# Introducing the Rust Leadership Council

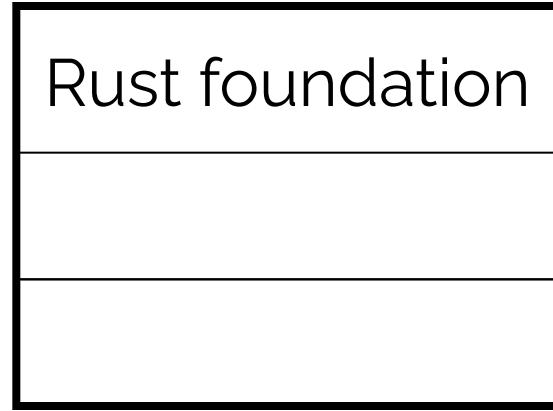
June 20, 2023 · Leadership Council on behalf of [Leadership Council](#)

As of today, [RFC 3392](#) has been merged, forming the new top level governance body of the Rust Project: the Leadership Council. The creation of this Council marks the end of both the Core Team and the interim Leadership Chat.

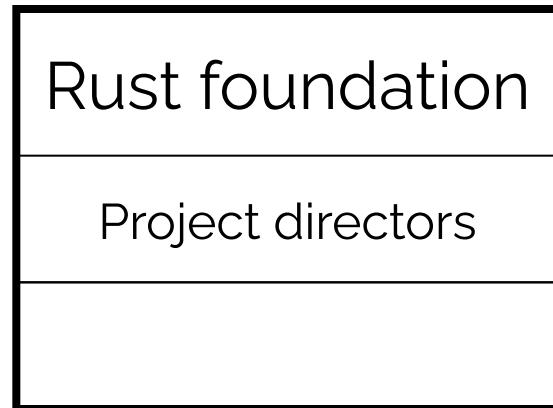
# Rust governance diagram



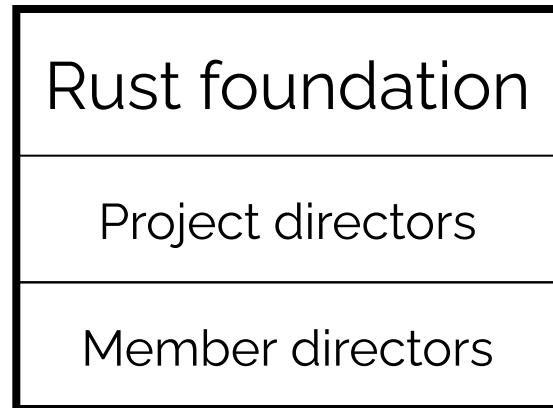
# Rust governance diagram



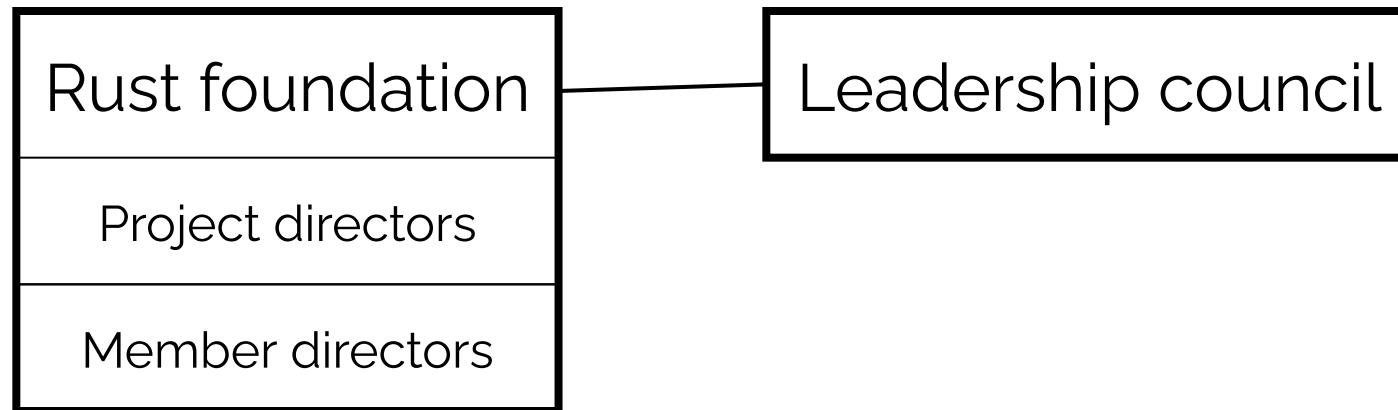
# Rust governance diagram



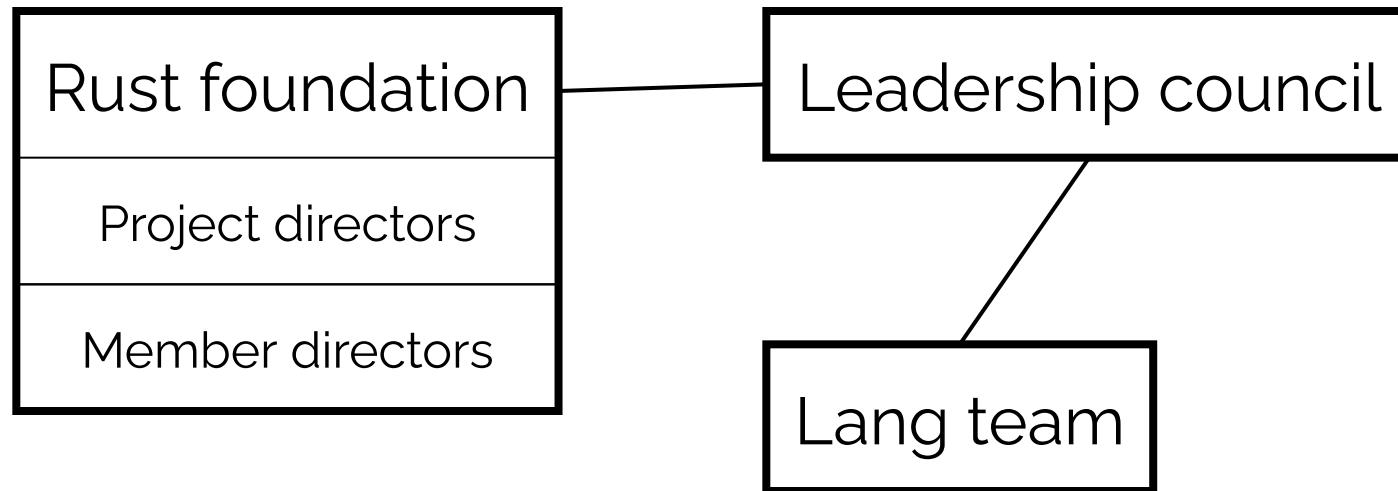
# Rust governance diagram



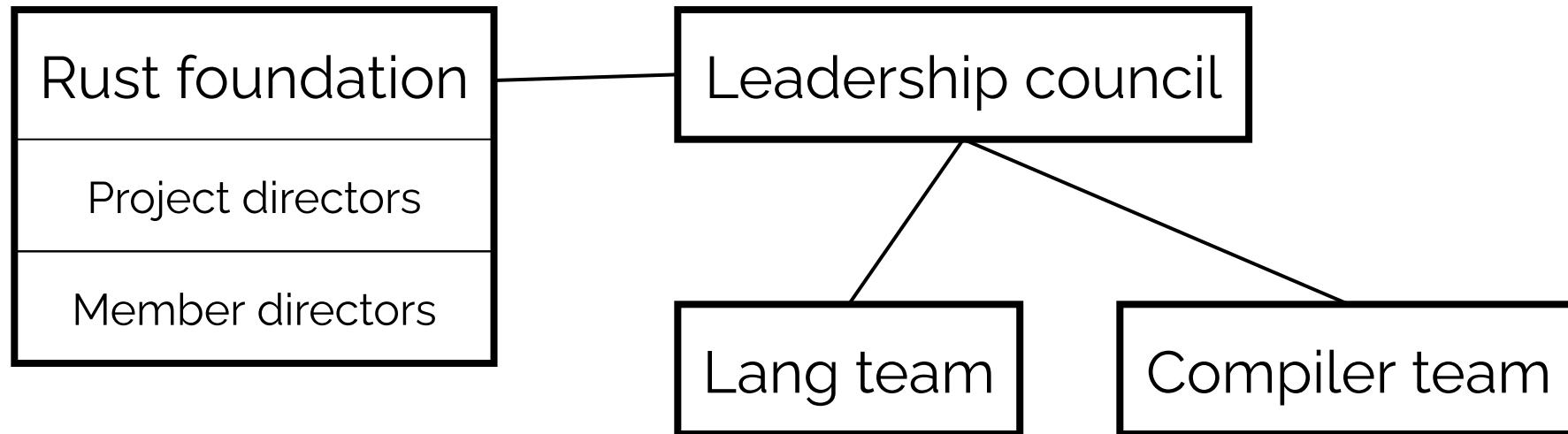
# Rust governance diagram



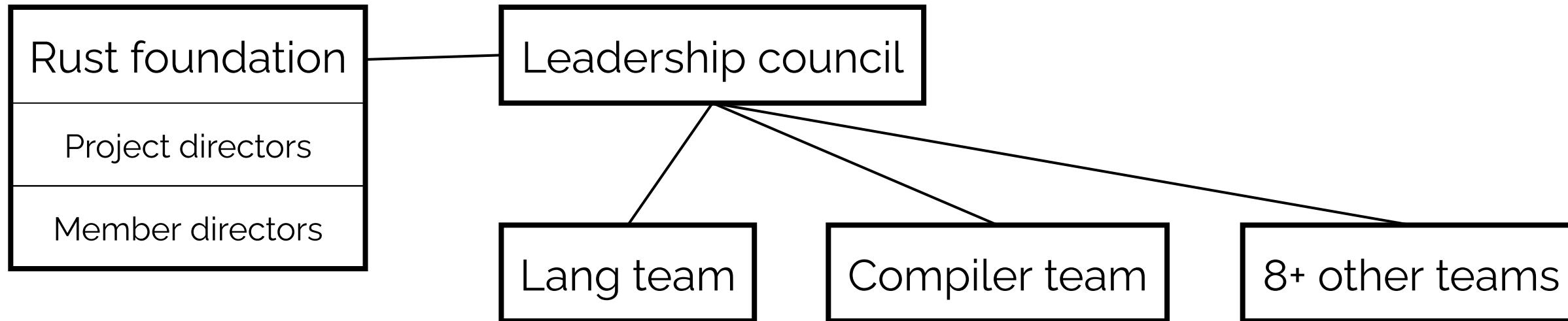
# Rust governance diagram



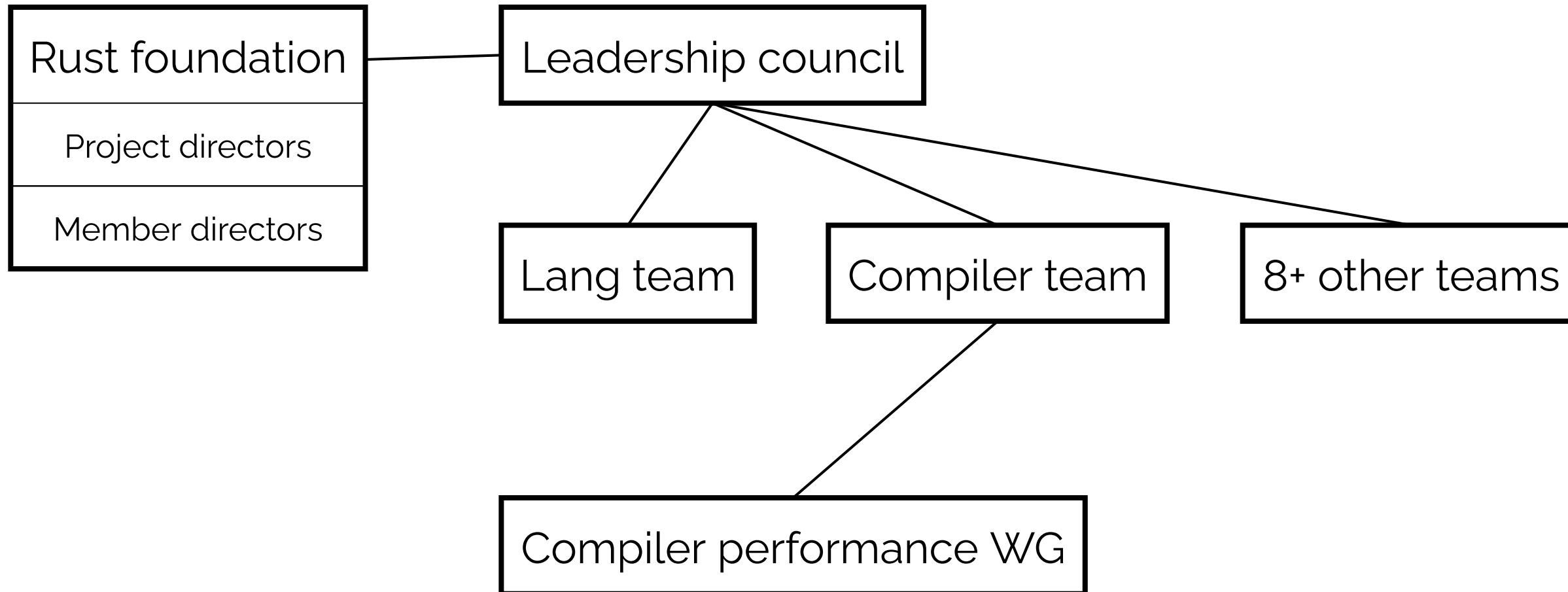
# Rust governance diagram



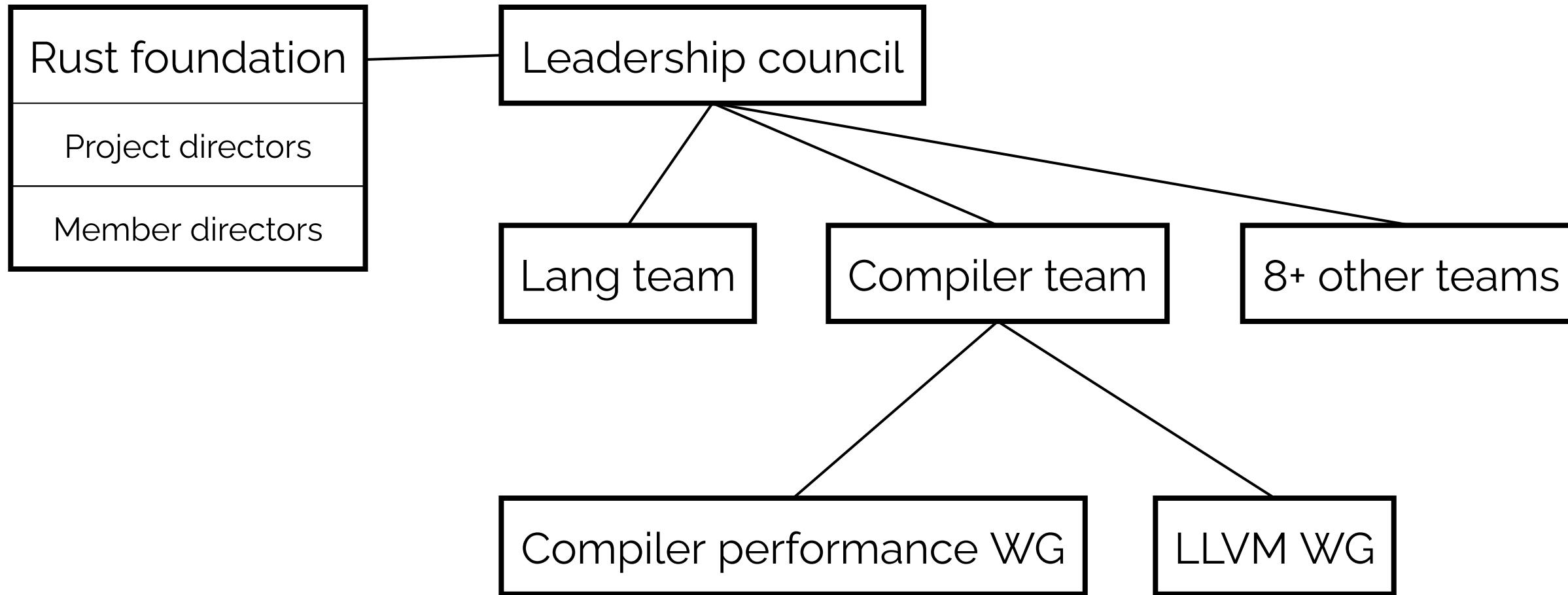
# Rust governance diagram



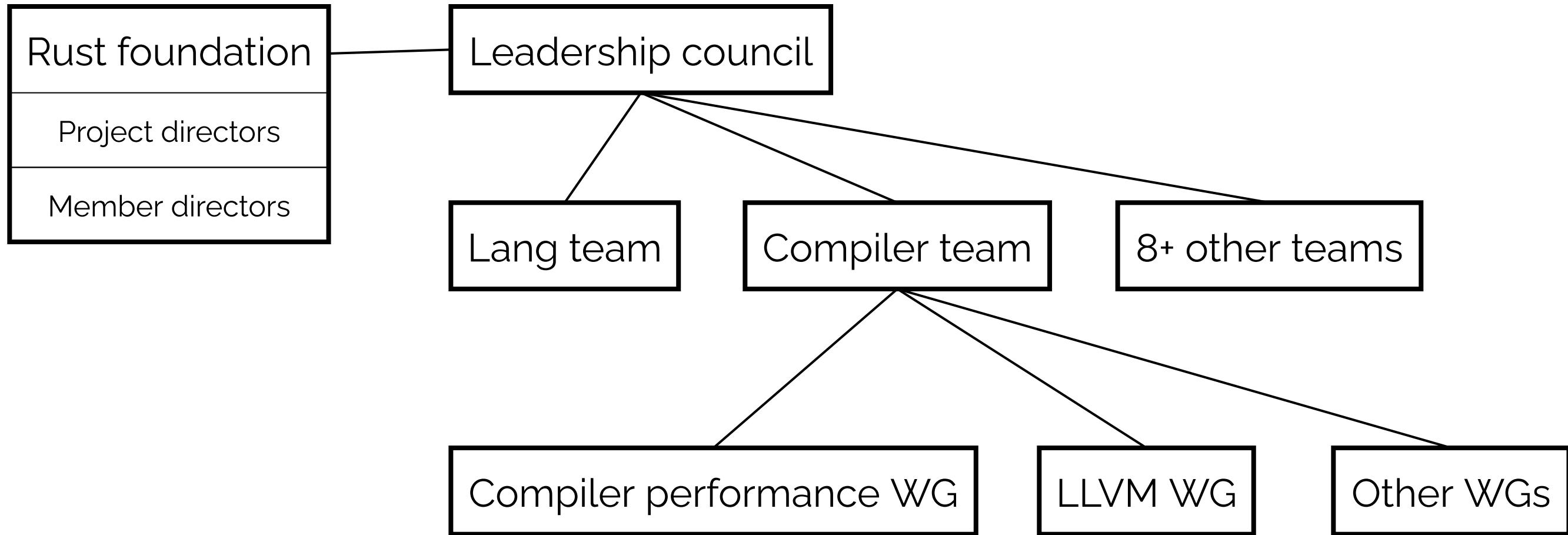
# Rust governance diagram



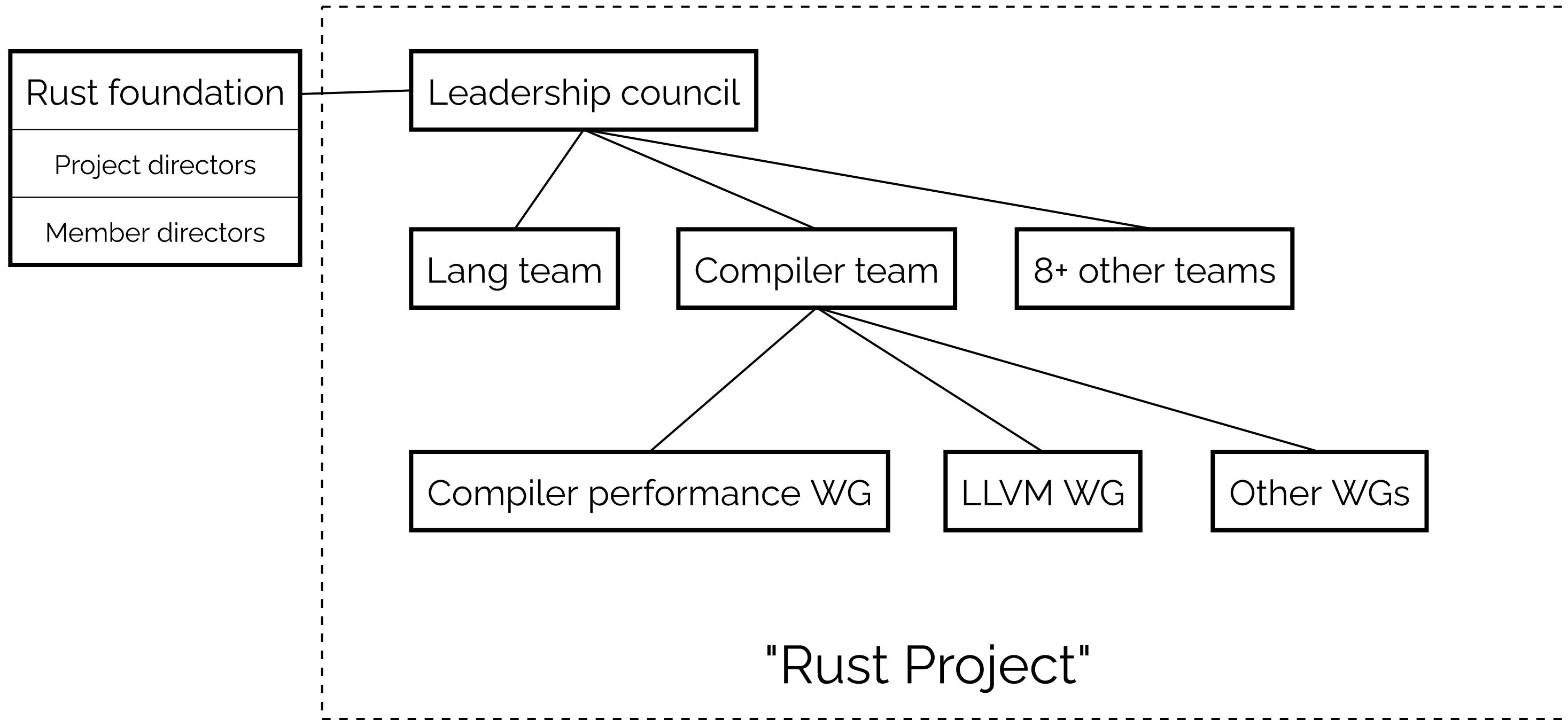
# Rust governance diagram



# Rust governance diagram



# Rust governance diagram



## Leadership council

Charged with the success of the Rust Project as whole, consisting of representatives from top-level teams

[MEMBERS & CONTACTS](#)

## Compiler team

Developing and managing compiler internals and optimizations

[MEMBERS & CONTACTS](#)

## Crates.io team

Managing operations, development, and official policies for crates.io

[MEMBERS & CONTACTS](#)

## Dev tools team

Contributing to and creating the Rust development tools

[MEMBERS & CONTACTS](#)

## Infrastructure team

Managing the infrastructure supporting the Rust project itself, including CI, releases, bots, and metrics

[MEMBERS & CONTACTS](#)

## Language team

Designing and helping to implement new language features

[MEMBERS & CONTACTS](#)

## Library team

Managing and maintaining the Rust standard library and official rust-lang crates

[MEMBERS & CONTACTS](#)

## Moderation team

Helping uphold the code of conduct and community standards

[MEMBERS & CONTACTS](#)

## Release team

Tracking regressions and stabilizations, and producing Rust releases

[MEMBERS & CONTACTS](#)





140 teams



140 teams  
46 working groups



140 teams  
46 working groups  
480+ team/WG members

# Automating team member permissions

**<https://github.com/rust-lang/team>**





# Automating team member permissions

**<https://github.com/rust-lang/team>**

arm.toml	Split Arm team to distinguish maintainers.	3 months ago
bootstrap.toml	make onur-ozkan co-lead of bootstrap team	5 months ago
cargo.toml	Add devtools alumni (#1137)	last month
clippy.toml	Backfill all currently listed alumni's teams (#1134)	last month
cloud-compute.toml	Give dev-desktop permission to t-libs	2 months ago

# Automating team member permissions



<https://github.com/rust-lang/team>

```
1  name = "wg-compiler-performance"
2  subteam-of = "compiler"
3  kind = "working-group"
4
5  [people]
6  leads = ["Mark-Simulacrum"]
7  members = [
8      "wesleywiser", "Mark-Simulacrum", "rylev",
9      "tgnotttingham", "pnkfelix", "michaelwoerister",
10     "Kobzol", "lqd", "nnethercote",
11 ]
12
13 [permissions]
14 perf = true
15 bors.rust.try = true
16 bors.measureme.review = true
```

# Automating team member permissions

<https://github.com/rust-lang/team>





# How do changes happen?



## How do changes happen?

- MCP (Major Change Proposal)



## How do changes happen?

- MCP (Major Change Proposal)
- ACP (API Change Proposal)



## How do changes happen?

- MCP (Major Change Proposal)
- ACP (API Change Proposal)
- FCP (Final Comments Period)



## How do changes happen?

- MCP (Major Change Proposal)
- ACP (API Change Proposal)
- FCP (Final Comments Period)
- **RFC (Request For Comments)**



# RFC process



# RFC process

## 1. Discuss on Zulip/Rust forum

🔒 Pre-RFC: `#[derive(Default)]` on enums with a `#[default]` attribute

language design

jhpratt 2 🖊 Apr '21

I've extracted and slightly modified some text from [Centril's draft RFC](#) from a while back that proposed this and much more. He has granted permission for me to use the text.

- Feature Name: `derive_enum_default`
- Start Date: 2021-04-07
- RFC PR: TODO
- Rust Issue: TODO



# RFC process

1. Discuss on Zulip/Rust forum
2. Write RFC

A screenshot of a GitHub pull request page. The title of the PR is "#[derive(Default)] on enums with a #[default] attribute #3". A purple button indicates it is "Merged" by pnkfelix on Jul 27, 2021. Below the title, there are tabs for Conversation (73), Commits (1), Checks (0), and Files changed (1). A comment from jhpratt dated Apr 12, 2021, is shown, containing code examples and reactions. The code example shows an enum Option<T> with a #[default] attribute:

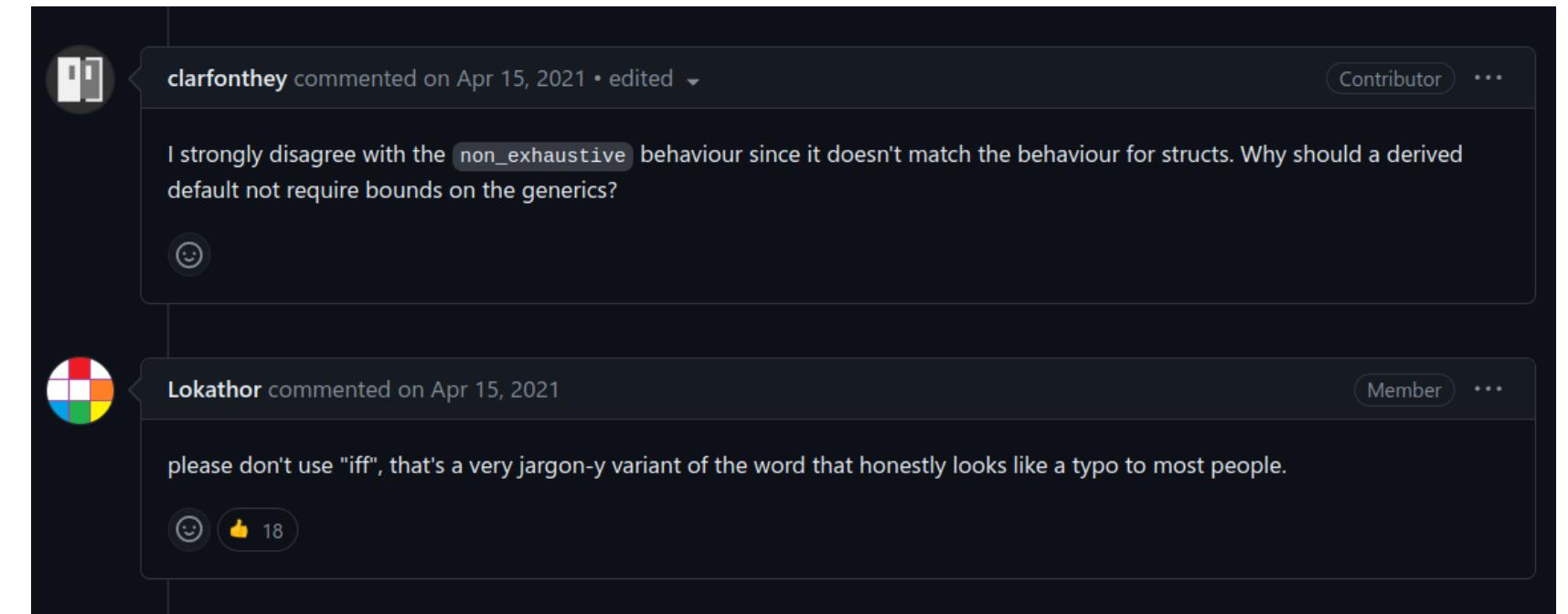
```
[#derive(Default)]
enum Option<T> {
    #[default]
    None,
    Some(T),
}
```

The comment has 48 upvotes, 9 comments, and 13 likes.



# RFC process

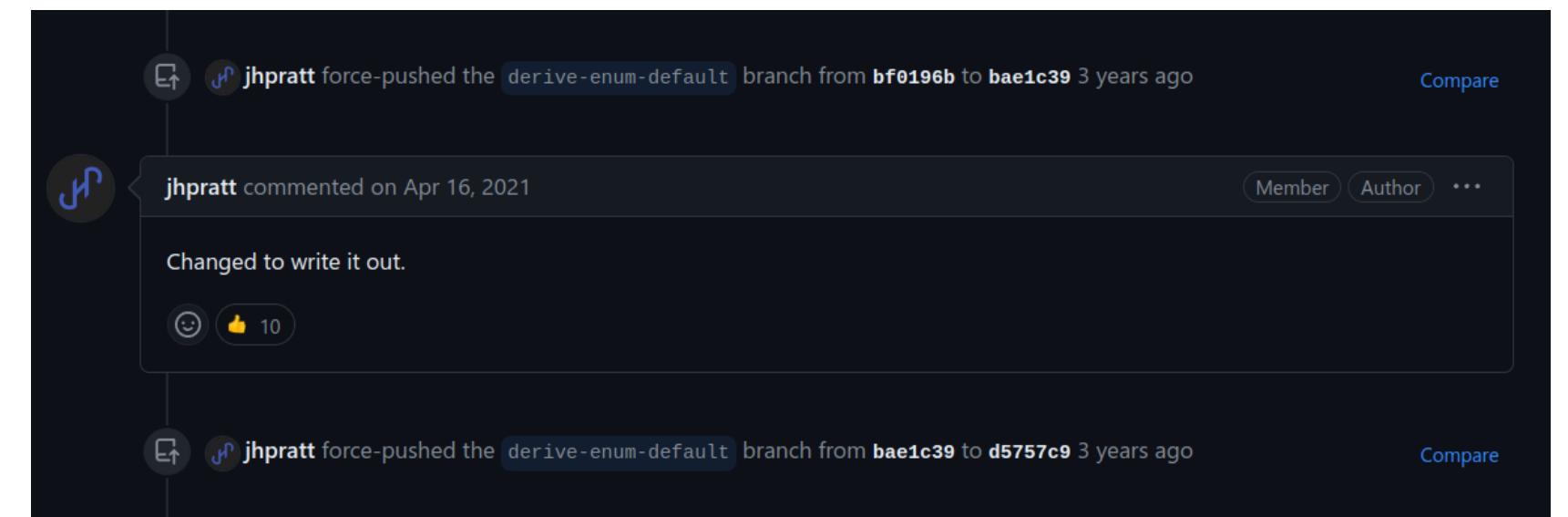
1. Discuss on Zulip/Rust forum
2. Write RFC
3. Receive comments





# RFC process

1. Discuss on Zulip/Rust forum
2. Write RFC
3. Receive comments
4. Modify RFC





# RFC process

1. Discuss on Zulip/Rust forum
2. Write RFC
3. Receive comments
4. Modify RFC
5. If concerns, goto 3.



# RFC process

1. Discuss on Zulip/Rust forum
2. Write RFC
3. Receive comments
4. Modify RFC
5. If concerns, goto 3.
6. Vote

A screenshot of a Zulip message from the bot account `rfcbot`. The message was commented on Jun 21, 2021, and edited by `joshtriplett`. It contains a list of team members who have been tagged to review the proposal:

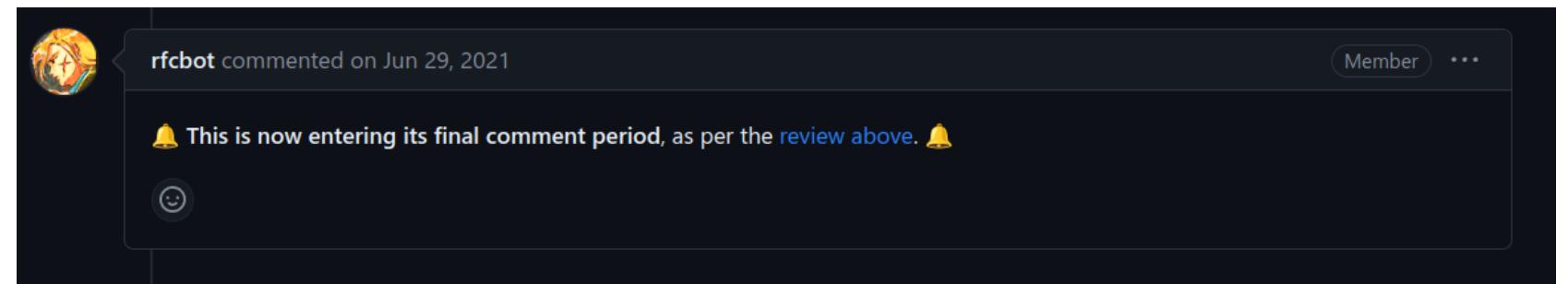
- `@Amanieu`
- `@BurntSushi`
- `@cramertj`
- `@dtolnay`
- `@joshttriplett`
- `@m-ou-se`
- `@nikomatsakis`
- `@pnkfelix`
- `@scottmcm`
- `@sfackler`
- `@yaahc`

No concerns currently listed.



# RFC process

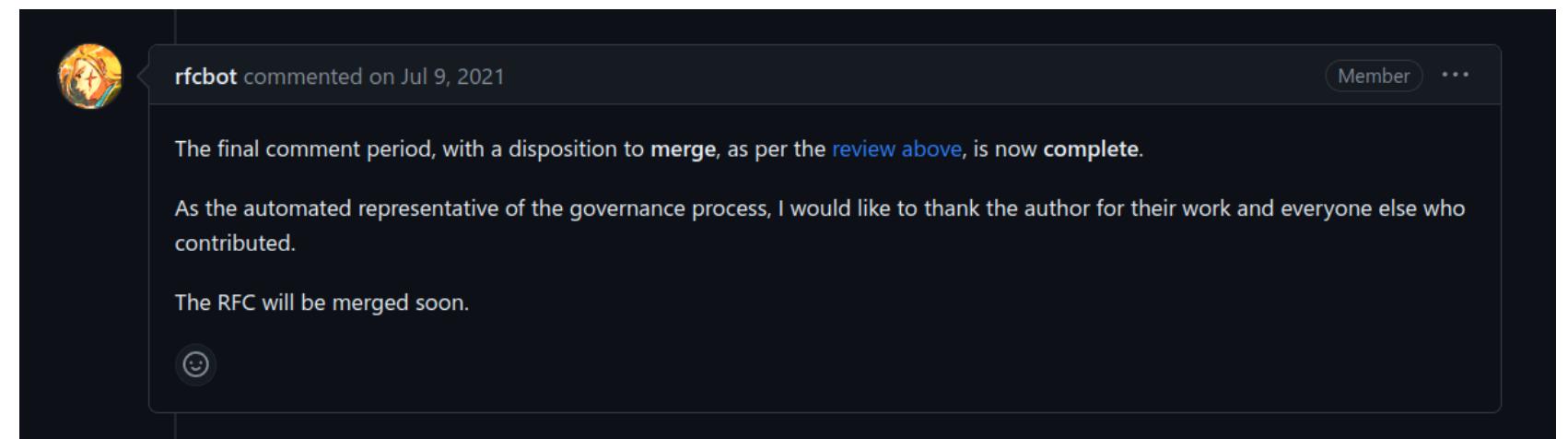
1. Discuss on Zulip/Rust forum
2. Write RFC
3. Receive comments
4. Modify RFC
5. If concerns, goto 3.
6. Vote
7. FCP (Final Comment Period)





# RFC process

1. Discuss on Zulip/Rust forum
2. Write RFC
3. Receive comments
4. Modify RFC
5. If concerns, goto 3.
6. Vote
7. FCP (Final Comment Period)
8. Wait 10 days





# RFC process

1. Discuss on Zulip/Rust forum
2. Write RFC
3. Receive comments
4. Modify RFC
5. If concerns, goto 3.
6. Vote
7. FCP (Final Comment Period)
8. Wait 10 days
9. Implement the feature





# RFC process

1. Discuss on Zulip/Rust forum
2. Write RFC
3. Receive comments
4. Modify RFC
5. If concerns, goto 3.
6. Vote
7. FCP (Final Comment Period)
8. Wait 10 days
9. Implement the feature
10. Stabilize the feature

Stabilize `derive_default_enum` #94457

Merged bors merged 2 commits into `rust-lang:master` from `jhpratt:stabilize-derive_default_enum` on Apr 15, 2022

Conversation 20 Commits 2 Checks 0 Files changed 16

**jhpratt** commented on Feb 28, 2022

This stabilizes `#[feature(derive_default_enum)]`, as proposed in [RFC 3107](#) and tracked in [#87517](#). In short, it permits you to `#[derive(Default)]` on `enum`s, indicating what the default should be by placing a `#[default]` attribute on the desired variant (which must be a unit variant in the interest of forward compatibility).

@rustbot label +S-waiting-on-review +T-lang

24 12



# RFC process

1. Discuss on Zulip/Rust forum
2. Write RFC
3. Receive comments
4. Modify RFC
5. If concerns, goto 3.
6. Vote
7. FCP (Final Comment Period)
8. Wait 10 days
9. Implement the feature
10. Stabilize the feature
11. ????



# RFC process

1. Discuss on Zulip/Rust forum
2. Write RFC
3. Receive comments
4. Modify RFC
5. If concerns, goto 3.
6. Vote
7. FCP (Final Comment Period)
8. Wait 10 days
9. Implement the feature
10. Stabilize the feature
11. ????
12. Profit



# RFC process

1. Discuss on Zulip/Rust forum
2. Write RFC
3. Receive comments
4. Modify RFC
5. If concerns, goto 3.
6. Vote
7. FCP (Final Comment Period)
8. Wait 10 days
9. Implement the feature
10. Stabilize the feature
11. ????
12. Profit

RFC acceptance: 4 months



# RFC process

1. Discuss on Zulip/Rust forum
  2. Write RFC
  3. Receive comments
  4. Modify RFC
  5. If concerns, goto 3.
  6. Vote
  7. FCP (Final Comment Period)
  8. Wait 10 days
  9. Implement the feature
  10. Stabilize the feature
  11. ????
  12. Profit
- Implementation + approval: 1 month



# RFC process

1. Discuss on Zulip/Rust forum
  2. Write RFC
  3. Receive comments
  4. Modify RFC
  5. If concerns, goto 3.
  6. Vote
  7. FCP (Final Comment Period)
  8. Wait 10 days
  9. Implement the feature
  10. Stabilize the feature
  11. ????
  12. Profit

— Stabilization: 9 months



# Changing Cargo defaults (FCP)

t-cargo > Setting `strip=debuginfo` by default when `debug=0` •)

 **Jakub Beránek**

Hi, I wanted to ask about enabling stripping debuginfo symbols by default when no debuginfo is requested.

When a Rust helloworld program is compiled in release mode (without debuginfo), the resulting binary currently has about 4.5 MiB! (on Linux). After stripping debug symbols from it (`strip --strip-debug`), the size is only ~440 KiB, so about 4 MiB of that is debuginfo, which comes from the Rust `stdlib`. This is IMO a bad default and leaves a bad impression on new users that Rust binaries are bloated.



# Changing Cargo defaults (FCP)

 **Kobzol** commented 3 weeks ago • edited Member ...

I would like to propose to change Cargo defaults so that `debug = 0` implies `strip = "debuginfo"`. This has been discussed on Zulip several times, most recently [here](#).

## Motivation

Currently, when `debug = 0` is used in a Cargo profile, Cargo promises that there will be no debuginfo in the resulting binary. However, this is not true, because the standard library contains debuginfo, and this debuginfo is propagated into the resulting binary by default, unless it is stripped.



# Changing Cargo defaults (FCP)

 **rfcbot** commented 3 weeks ago • edited by arlosi ▾ Member ...

Team member [@weihanglo](#) has proposed to merge this. The next step is review by the rest of the tagged team members:

- [@Eh2406](#)
- [@Muscraft](#)
- [@arlosi](#)
- [@ehuss](#)
- [@epage](#)
- [@joshtriplett](#)
- [@weihanglo](#)

No concerns currently listed.

Once a majority of reviewers approve (and at most 2 approvals are outstanding), this will enter its final comment period. If you spot a major issue that hasn't been raised at any point in this process, please speak up!

See [this document](#) for info about what commands tagged team members can give me.





# Changing Cargo defaults (FCP)

## Strip debuginfo when debuginfo is not requested #13257

Open

Kobzol wants to merge 5 commits into `rust-lang:master` from `Kobzol:profile-strip-debuginfo`

Conversation 2

Commits 5

Checks 19

Files changed 6



Kobzol commented last week • edited

Member

Reviewers

No reviews

Still in progress?

### What does this PR try to resolve?

This PR implements [this proposal](#). It contains a detailed description of the change.

As a summary, this PR modifies Cargo so that if the user doesn't set `strip` explicitly, and debuginfo is not enabled for any package being compiled, Cargo will implicitly set `strip = "debuginfo"`, to strip pre-existing debuginfo coming from the standard library. This reduces the default size of release binaries considerably (~4.5 MiB => ~450 KiB for helloworld on Linux x64).

Assignees



Labels



# Changing Cargo defaults (FCP)

## Strip debuginfo when debuginfo is not requested #13257

Merged bors merged 5 commits into rust-lang:master from Kobzol:profile-strip-debuginfo 1 hour ago

Conversation 7 Commits 5 Checks 19 Files changed 6

Kobzol commented last week • edited Member ...

**What does this PR try to resolve?**

This PR implements [this proposal](#). It contains a detailed description of the change.

As a summary, this PR modifies Cargo so that if the user doesn't set `strip` explicitly, and debuginfo is not enabled for any package being compiled, Cargo will implicitly set `strip = "debuginfo"`, to strip pre-existing debuginfo coming from the standard library. This reduces the default size of release binaries considerably (~4.5 MiB => ~450 KiB for helloworld on Linux x64).

**Reviewers**  
No reviews

**Assignees**  
 ehuss

**Labels**  
 A-build-exec  
 S-waiting-on



# Changing Cargo defaults (FCP)

Less than a month



# Changing Cargo defaults (FCP)

## Primary benchmarks ↓

	Benchmark	Profile	Scenario	Backend	% Change	Significance Threshold ?	Significance Factor ?
▶	helloworld	opt	full	llvm	-88.92%	0.17%	509.82x
▶	helloworld	opt	incr-unchanged	llvm	-88.92%	0.17%	508.90x
▶	helloworld	opt	incr-full	llvm	-88.92%	0.17%	508.90x
▶	helloworld	opt	incr-patched: println	llvm	-88.92%	0.17%	508.90x
▶	serde_derive-1.0.136	opt	incr-unchanged	llvm	-60.61%	0.11%	559.85x
▶	serde_derive-1.0.136	opt	incr-full	llvm	-60.61%	0.11%	559.85x
▶	serde_derive-1.0.136	opt	incr-patched: println	llvm	-60.52%	0.11%	555.62x
▶	serde_derive-1.0.136	opt	full	llvm	-59.51%	0.16%	376.52x
▶	bitmaps-3.1.0	opt	incr-full	llvm	0.06%	0.00%	31.75x
▶	bitmaps-3.1.0	opt	incr-unchanged	llvm	0.06%	0.00%	31.50x
▶	bitmaps-3.1.0	opt	incr-patched: println	llvm	0.06%	0.00%	31.50x
▶	bitmaps-3.1.0	opt	full	llvm	0.06%	0.00%	20.62x



# Changing Cargo defaults (FCP)

## Primary benchmarks ↓

Benchmark	Profile	Scenario	Backend	% Change	Significance Threshold ?	Significance Factor ?
▶ helloworld	opt	incr-unchanged	llvm	-47.35%	0.50%	95.05x
▶ helloworld	opt	full	llvm	-45.97%	0.51%	89.66x
▶ helloworld	opt	incr-patched: println	llvm	-45.72%	0.54%	84.36x
▶ helloworld	opt	incr-full	llvm	-43.68%	0.60%	72.38x
▶ serde_derive-1.0.136	opt	incr-unchanged	llvm	-7.59%	0.13%	60.08x
▶ serde_derive-1.0.136	opt	incr-patched: println	llvm	-1.54%	1.20%	1.28x
▶ serde_derive-1.0.136	opt	incr-full	llvm	-0.66%	0.24%	2.77x



# How to handle conflicts?



# Great int debate (2014)



## Great int debate (2014)

“

We have been reading these threads and have also done a lot of internal experimentation, and we believe we've come to a final decision on the fate of integers in Rust.

Core team (2014) , ,



# No new rationale



## No new rationale

decisions must be made only on the basis of rationale  
already debated in public (to a steady state)



# Awaiting a solution (2018/2019)



# Awaiting a solution (2018/2019)

```
await! (fut) ;  
await fut;  
await { fut } ;  
fut.await;  
fut.await();  
fut.await!;  
fut@await;
```



# Resolve await syntax #57640

Closed

cramertj opened this issue on Jan 15, 2019 · 512 comments



cramertj commented on Jan 15, 2019 • edited by scottmcm

Member

...

Before commenting in this thread, please check [#50547](#) and try to check that you're not duplicating arguments that have already been made there.



arthureroberer

Apr '19

the Orthogonality argument convinced me  
im now on the postfix camp

7



drwwlkr

Apr '19

The orthogonality argument feels really unconvincing since it requires breaking what people have already learned about field access syntax.

Having magical implicit fields feels strictly worse than having to type extra parens.



Darksonn · 4 yr. ago  
tokio · rust-for-linux

I also remember someone making a survey asking people which syntax they wanted, and the Rust team decided to not follow the majority of that survey. I am happy they took the decision to  
~~do that.~~

▲ 203 ▾ Reply Share ...



othermike · 4 yr. ago

Yeah, I'm not sure surveys of the general userbase are that useful in language design. I can remember plenty of examples of younger-me voting for a faster horse.

▲ 138 ▾ Reply Share ...



crabbytag · 4 yr. ago

| a faster horse

I googled this. It's based on a quote attributed to Henry Ford

| If I had asked people what they wanted, they would have said faster horses.

▲ 114 ▾ Reply Share ...



sepease · 4 yr. ago

I objected.

I was wrong.



304



Reply

Share

...



disintegore · 4 yr. ago

I fought the crab and the crab won



172



Reply

Share

...



Narishma · 4 yr. ago



dpc\_pw · 4 yr. ago

The crab always wins.



12



Reply

Share

...



# Unresolved conflicts



# Unresolved conflicts

- Burnout



## Unresolved conflicts

- Burnout
- One party leaves



## Unresolved conflicts

- Burnout
- One party leaves
- Most work done => most voting rights?



## Unresolved conflicts

- Burnout
- One party leaves
- Most work done => most voting rights?
- Transparency

# How to get something done?



- Poke someone to do it



# How to get something done?

## Enable LTO for rustc\_driver.so #101403

Merged **bors** merged 5 commits into `rust-lang:master` from `bjorn3:dylib_lto` on Oct 23, 2022

Conversation 83    Commits 5    Checks 0    Files changed 15

**bjorn3** commented on Sep 4, 2022 • edited Member ...

Alternative to #97154

This enables LTO'ing dylibs behind a feature flag and uses this feature for compiling `rustc_driver.so`.

35



# How to get something done?

## Instruction count

This is a highly reliable metric that was used to determine the overall result at the top of this comment.

	mean <sup>[1]</sup>	range	count <sup>[2]</sup>
Regressions ✗ (primary)	-	-	0
Regressions ✗ (secondary)	-	-	0
Improvements ✓ (primary)	-4.2%	[-9.6%, -0.4%]	230
Improvements ✓ (secondary)	-4.0%	[-9.5%, -0.4%]	257
All ✗✓ (primary)	-4.2%	[-9.6%, -0.4%]	230



# How to get something done?

- Poke someone to do it
- Summarize the current state



# How to get something done?



CHANGED PŘED 9 MĚSÍCI

Like 4

Bookmark

Subscribed ▾

## Better support of Docker layer caching in Cargo

This document serves as a summary of motivation, existing workarounds, prior art and possible solutions for the Cargo issue [#2644](#) (dubbed “original issue” in the rest of the document), which is currently the most commented and upvoted issue on the Cargo repository. The original statement of the issue is “provide a Cargo command for building only the dependencies of a project”, however this description does not adequately describe the primary problem, therefore I will mostly talk about **“modifying Cargo to better support Docker layer caching”**, which is the primary use-case, as described below.



# How to get something done?

- Poke someone to do it
- Summarize the current state
- Do it :-)



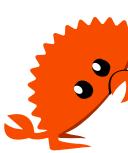
# How to get something done?

 t-cargo > Setting `strip=debuginfo` by default when `debug=0` •)

 **Jakub Beránek**

Hi, I wanted to ask about enabling stripping debuginfo symbols by default when no debuginfo is requested.

When a Rust helloworld program is compiled in release mode (without debuginfo), the resulting binary currently has about 4.5 MiB! (on Linux). After stripping debug symbols from it (`strip --strip-debug`), the size is only ~440 KiB, so about 4 MiB of that is debuginfo, which comes from the Rust `stdlib`. This is IMO a bad default and leaves a bad impression on new users that Rust binaries are bloated.



# How to get something done?

## Strip debuginfo when debuginfo is not requested #13257

Merged bors merged 5 commits into `rust-lang:master` from `Kobzol:profile-strip-debuginfo` 1 hour ago

Conversation 7 Commits 5 Checks 19 Files changed 6

Kobzol commented last week • edited Member ...

**What does this PR try to resolve?**

This PR implements [this proposal](#). It contains a detailed description of the change.

As a summary, this PR modifies Cargo so that if the user doesn't set `strip` explicitly, and debuginfo is not enabled for any package being compiled, Cargo will implicitly set `strip = "debuginfo"`, to strip pre-existing debuginfo coming from the standard library. This reduces the default size of release binaries considerably (~4.5 MiB => ~450 KiB for helloworld on Linux x64).

**Reviewers**  
No reviews

**Assignees**  
 ehuss

**Labels**  
 A-build-exec  
 S-waiting-on



# How to get something done?

t-compiler > Automatically convert `clone` calls to `clone\_from` · · ·

 **Jakub Beránek**

Hi, have there been any attempts to automatically (e.g. via a MIR opt) convert `a = b.clone();` into `a.clone_from(&b);`?



# How to get something done?

Add [assigning\\_clones](#) lint #12077

[Open](#) Kobzol wants to merge 3 commits into [rust-lang:master](#) from [Kobzol:assigning-clones](#) [Diff](#)

[Conversation 3](#) [Commits 3](#) [Checks 7](#) [Files changed 16](#)

 **Kobzol** commented 2 weeks ago [Member](#) ...

This PR is a "revival" of #10613 (with @kpreid's permission).

I tried to resolve most of the unresolved things from the mentioned PR:



# How to get something done?



**Jakub Beránek**

Hi, I was wondering what's the status of the 2022 survey data. Concretely:

- 1) Is there a way to let the compiler team/wg-performance know what is the status of the "slow compile times" challenge response from the 2022 survey? In the 2021 survey, I think that it was a single % that told us how many people consider slow compile times to be a problem.
- 2) Can I help with extracting this kind of data from the survey, and/or publishing the anonymized results in some raw form somewhere? I'm sure that it would be useful to people.



# How to get something done?

## Bootstrap questions for the 2023 survey #234

Merged · Kobzol merged 25 commits into `rust-lang:main` from `Kobzol:survey-2023` on Oct 9, 2023

Conversation 70 · Commits 25 · Checks 0 · Files changed 1

 **Kobzol** commented on Sep 27, 2023 • edited · Member · ...

I copy pasted the 2022 questions and modified some small details (mostly Rust versions and dates), based on the previous diff from the 2021 to the 2022 surveys.

Then I went through the ~20 newest opened issues on this repository and tried to incorporate some "low hanging fruit" into the 2023 survey. I did the changes in individual commits, so that we can discuss and cherry pick them separately.

Helps with #202  
Fixes: #224  
Fixes: #225  
Fixes: #228  
Fixes: #231

2



# How to get something done?

 foundation > Google Summer of Code and idea list •)

 **Jakub Beránek** EDITED

Hi! There is a rather famous project called Google Summer of Code (GSOC), where Google sponsors students to work on open-source projects over summer, under the mentorship of contributors to said projects. I think that engaging in this project could be beneficial for Rust, as it could help us bring new contributors to the project, and also motivate us to better formalize a list of ideas for self-contained projects that (new) contributors could work on. Such an ideas list would be nice to have also e.g. for the Rust Foundation contributor grants, or just to give new contributors an overview on what they could start working on.

One of the requirements of the organization application is to create a list of ideas for projects (the project should last ~8-22 weeks) that the students could work on (over the span of summer), with some basic description of what are the projects about, and who could mentor them. The students then submit actual project proposals based on this idea list.

I'm writing here because it is required for an organization that manages the OSS project (so in our case, the Rust Foundation), to apply for GSOC. GSOC 2024 is currently waiting for applications from the organizations, the applications will be opened from January 22 until February 6, 2024. But it happens every year, so even if we don't make it this year, we could decide to join e.g. next year.

What do you think about this?



# How to get something done?



OWNED THIS NOTE



CHANGED PŘED 2 DNY



Like



Bookmark



Subscribed ▾

## Rust GSoC 2024 project list

This document contains a list of projects offered by the Rust Project for [Google Summer of Code \(2024\)](#).

**The project list should be prepared by 22. 01. 2024.**

GSOC mentions that at least 4 ideas should be in the list. Optimal number is 8-10+. Note that the projects serve as inspiration for the students. They will actually write down and propose their own project topic (that could be heavily inspired by something from our list). The main thing is having a mentor available for the project.



# How to join the fray?



## How to join the fray?

- Lurk, observe, engage @ Zulip
  - <https://rust-lang.zulipchat.com>



## How to join the fray?

- Lurk, observe, engage @ Zulip
  - <https://rust-lang.zulipchat.com>
- Find easy issues @ GitHub
  - <https://github.com/rust-lang/rust/labels/E-easy>



## How to join the fray?

- Lurk, observe, engage @ Zulip
  - <https://rust-lang.zulipchat.com>
- Find easy issues @ GitHub
  - <https://github.com/rust-lang/rust/labels/E-easy>
- Study



## How to join the fray?

- Lurk, observe, engage @ Zulip
  - <https://rust-lang.zulipchat.com>
- Find easy issues @ GitHub
  - <https://github.com/rust-lang/rust/labels/E-easy>
- Study
  - Forge: <https://forge.rust-lang.org/>



## How to join the fray?

- Lurk, observe, engage @ Zulip
  - <https://rust-lang.zulipchat.com>
- Find easy issues @ GitHub
  - <https://github.com/rust-lang/rust/labels/E-easy>
- Study
  - Forge: <https://forge.rust-lang.org/>
  - Rustc dev guide: <https://rustc-dev-guide.rust-lang.org/>



## How to join the fray?

- Lurk, observe, engage @ Zulip
  - <https://rust-lang.zulipchat.com>
- Find easy issues @ GitHub
  - <https://github.com/rust-lang/rust/labels/E-easy>
- Study
  - Forge: <https://forge.rust-lang.org/>
  - Rustc dev guide: <https://rustc-dev-guide.rust-lang.org/>
  - Stdlib dev guide: <https://std-dev-guide.rust-lang.org/>



imgflip.com

TOMMY TIGG



# Thank you for your attention

Slides are available here:



Slides were created with [github.com/spirali/elsie](https://github.com/spirali/elsie)