

State of Racket 2021

Sam Tobin-Hochstadt
(chaperone (eleventh RacketCon))

8.0!



8.1!

8.2!

8.3!

Code


Community


Coming Soon!

Code

Compatibility and consolidation

pkg-build failures on Racket CS #3457

 Open

 62 of 75 tasks

samth opened this issue on Oct 22, 2020 · 54 comments



samth commented on Oct 22, 2020 • edited

racket's sponsor

Member



This issue is for tracking package build failures on Racket CS, particularly those that don't fail on Racket BC.

Updated status (as of 2/23/21):




pkg-build bugs

- ☐ remix <https://plt.eecs.northwestern.edu/pkg-build/server/built/fail/remix.txt> (s-exp->fasl error)
- ☐ sql-sourcery <https://plt.eecs.northwestern.edu/pkg-build/server/built/fail/sql-sourcery.txt> ("current-directory: `exists' access denied for /home/root/racket/")
- ☐ tessellation <https://plt.eecs.northwestern.edu/pkg-build/server/built/fail/tessellation.txt> ("current-directory: `exists' access denied for /home/root/racket/")

Uses C modules

- ☐ socketcan <https://plt.eecs.northwestern.edu/pkg-build/server/built/fail/socketcan.txt>
- ☐ termios <https://plt.eecs.northwestern.edu/pkg-build/server/built/fail/termios.txt>
- ☐ serial <https://plt.eecs.northwestern.edu/pkg-build/server/built/fail/serial.txt> (see `termios`)

Uses BC runtime functions

- ☐ ffi-definer-convention <https://plt.eecs.northwestern.edu/pkg-build/server/built/fail/ffi-definer-convention.txt>, see  [Documentation won't build on Racket CS](#) takikawa/racket-ffi-definer-convention#1
- ☐ video-v0-1 <https://plt.eecs.northwestern.edu/pkg-build/server/built/fail/video-v0-1.txt>  [video-v0-1 does not work on Racket CS](#) videolang/video#66
- ☐ udev <https://plt.eecs.northwestern.edu/pkg-build/server/built/test-fail/udev.txt> (uses BC runtime functions,  Use `unsafe-socket->port` . mordae/racket-udev#1)
- ☐ binutils <https://plt.eecs.northwestern.edu/pkg-build/server/built/test-fail/binutils.txt> (uses BC runtime functions)

Package bugs exposed by Racket CS thread scheduling

Simplification

Easier build process

More platforms

Single repository

Syntax Simplification

 syntax-arm

 syntax-disarm

raco cross

```
$ raco cross --target x86_64-linux dist example-dist example
```

C Code

425978

BC 5.0 (6/2010)

249617

BC 6.0 (2/2014)

244543

BC 7.0 (7/2018)

50773

CS 7.9.0.3 (10/2020)

C Code

425978

BC 5.0 (6/2010)

249617

BC 6.0 (2/2014)

244543

BC 7.0 (7/2018)

50773

CS 7.9.0.3 (10/2020)

50667

CS 8.3 (11/2021)

Community

WELCOME

welcome

announcements

rules

introductions

roles

MODERATION

moderator-only

GENERAL

general

show-and-tell

help

off-topic

resources

drracket

sauron-ide

advent-of-code

gamejam

internals

rhombus

bots

jobs

meta

racketscript

rcon

VOICE CHAT

samth

#3420

announcements

Follow

Announcements about this server, such as channel changes and server events.

November 6, 2021

7:11 PM

Stephen

Racket version 8.3 is now available from

<https://racket-lang.org/>

* Racket removes syntax arming and disarming in favor of a simpler system of protected syntax operations, along with other updates to the syntax system.

* DrRacket has improved support for custom #lang languages.

* Typed Racket improves precision for type-checking of non-polymorphic structures, existential types, and certain binding forms.

* Scribble HTML output gains a button to show / hide the table of contents on mobile platforms.

* Redex's stepper's GUI shows IO-judgment form rule names.

* Many bug fixes!

The following people contributed to this release:

Adam Zaiter, Alex Knauth, Alexis King, Ayman Osman, Ben Greenman, Bob Burger, Bogdan Popa, Brian Adkins, Cameron Moy, Carl Eastlund, Dan Holtby, Dominik Pantůček, Eli Barzilay, Ethan Leba, Fred Fu, Greg Hendershott, Gustavo Massaccesi, J. Ryan Stinnett, Jason Hemann, Jay McCarthy, Jesse Alama, Joel Dueck, John Clements, Jonathan Simpson, Kartik Sabharwal, Laurent Orseau, Lehua Ding, Maciej Barć, Marc Burns, Matthew Flatt, Matthias Felleisen, Michael Ballantyne, Mike Sperber, Noah Ma, Paulo Matos, Pavel Panchekha, Philip McGrath, Robby Findler, Ryan Culpepper, Ryan Sundberg, Sage Gerard, Sam Tobin-Hochstadt, Shu-Hung You, Sorawee Porncharoenwase, Stefan Schwarzer, Stephen De Gabrielle, Vincent St-Amour, William J. Bowman, minor-change, and yjqww6

Feedback Welcome

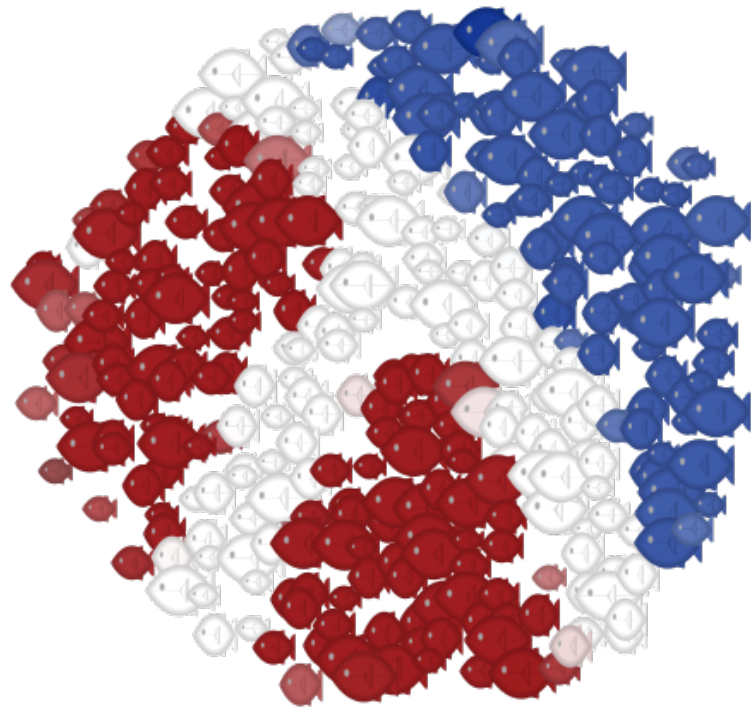
Stephen (Copied from announce in slack)

Message #announcements

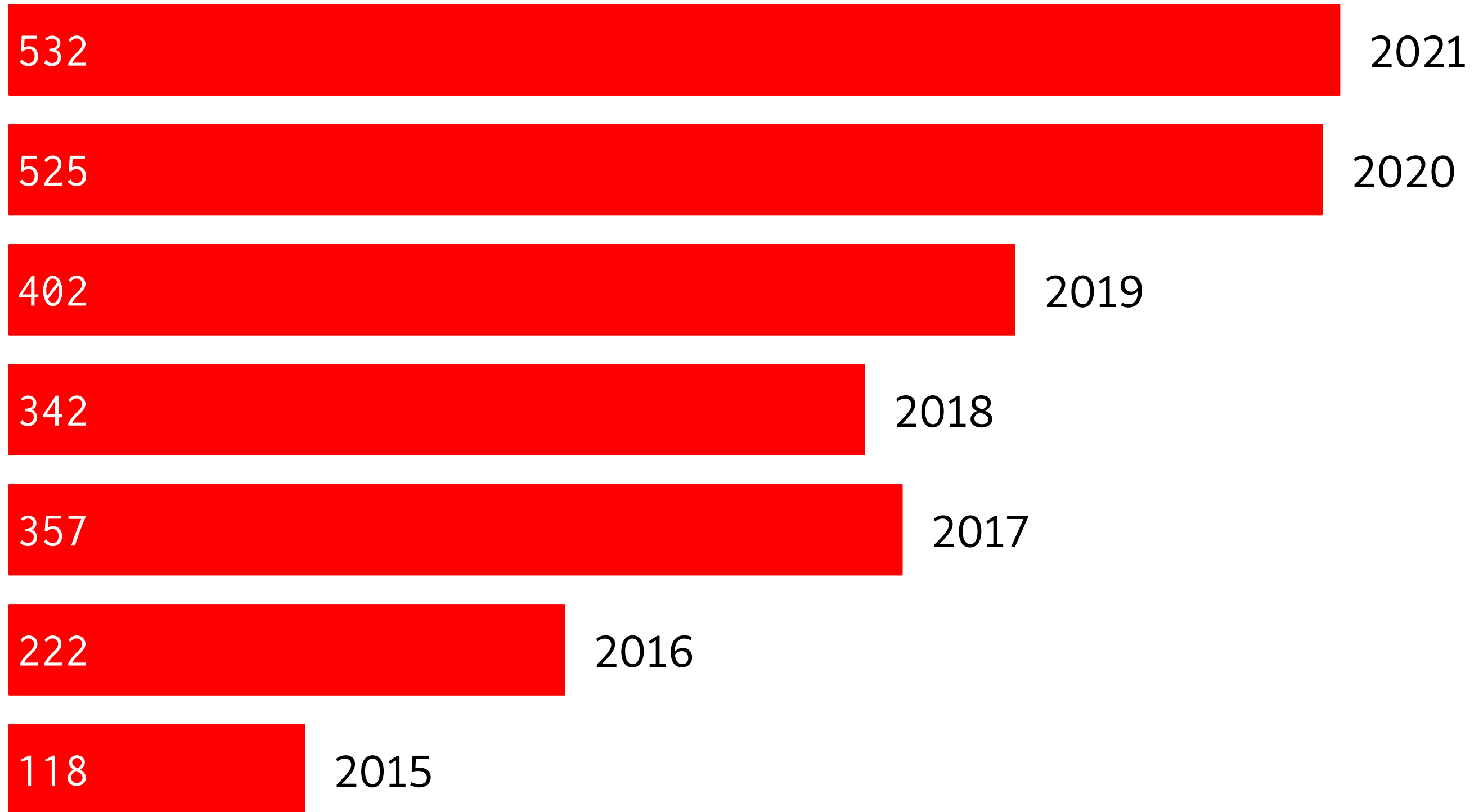
GIF

16

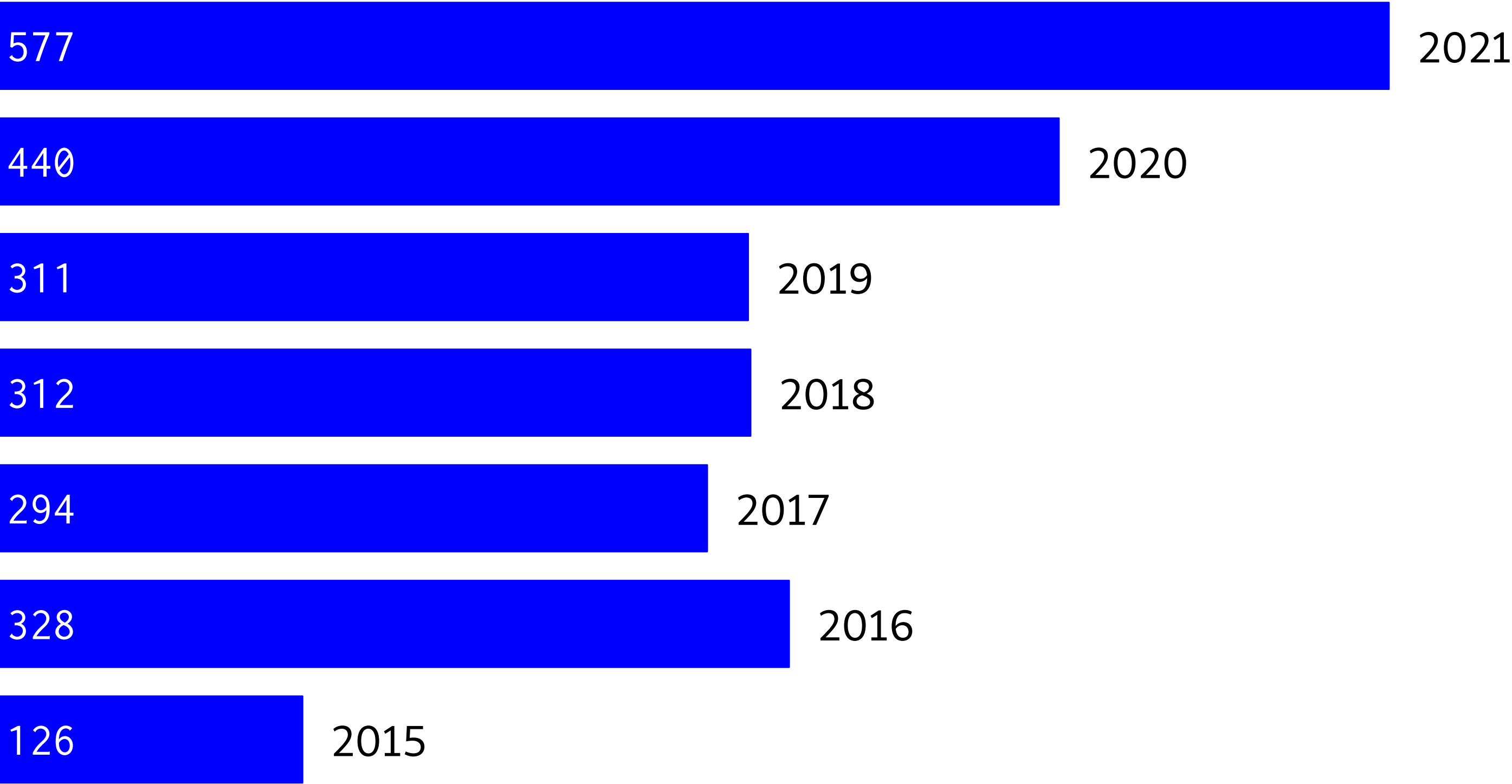
Competitions



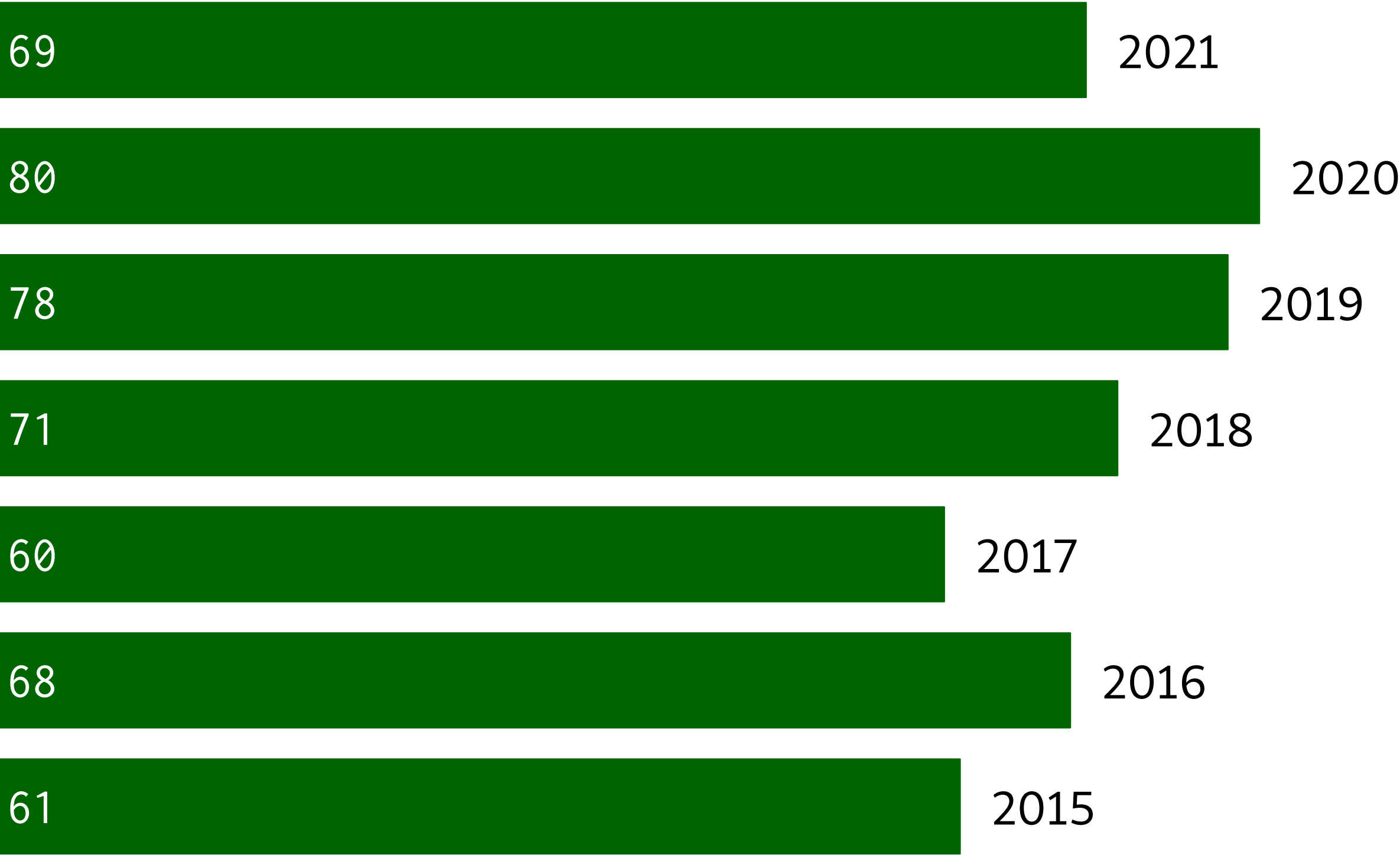
PRs Merged



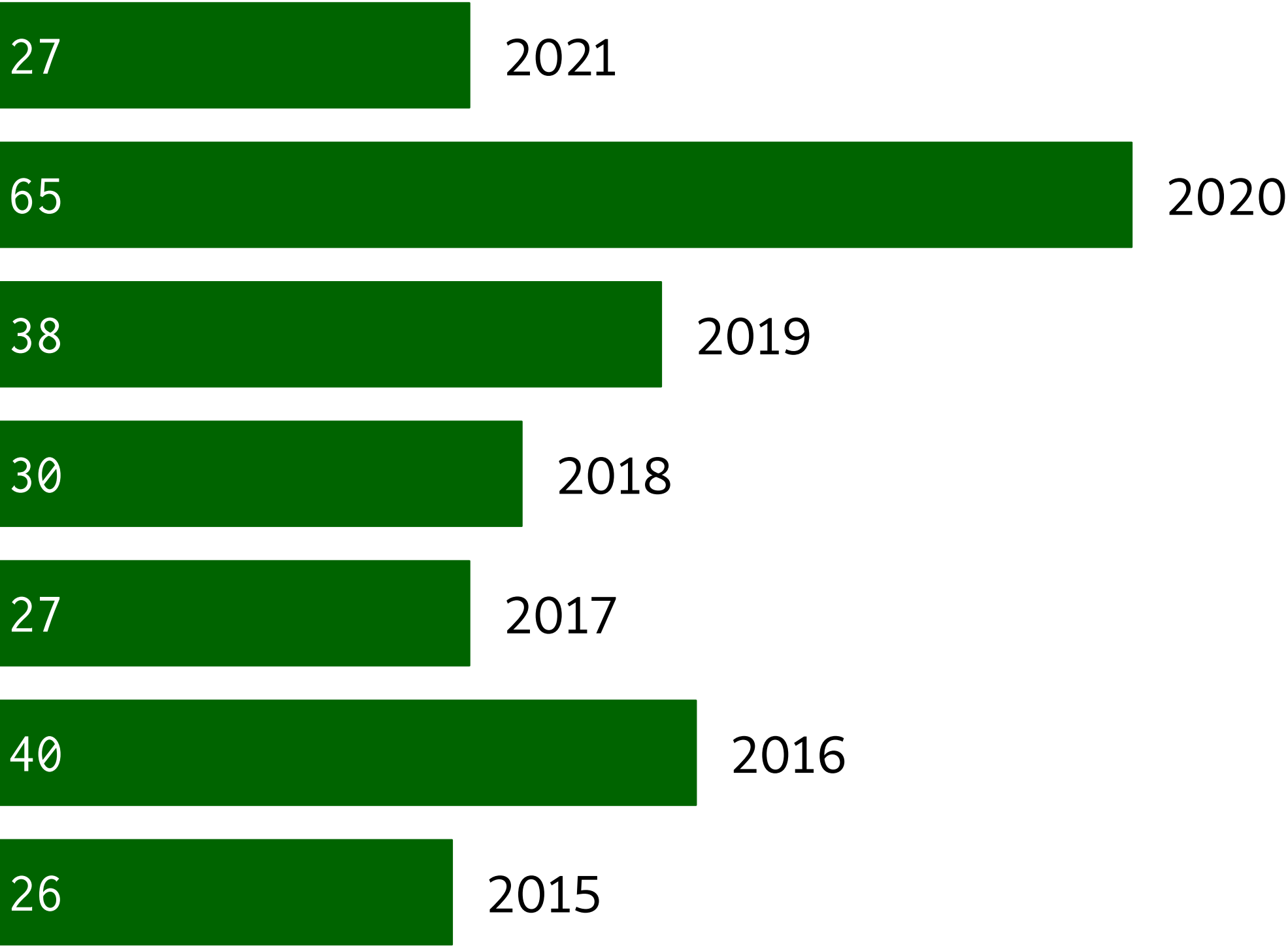
Issues Closed



Contributors to racket/racket



New Contributors to racket/racket



Coming Soon!


```

[samth@huor:~/work/rcon2020 (master) plt] racket
Welcome to Racket v8.3.0.6 [cs].
> (require slideshow/step slideshow/code slideshow/face
    ppict/2 ppict/slideshow2 "title.rkt"
    (only-in slideshow/slide title-size)
    "config.rkt" "helper.rkt"
    (except-in "beamer.rkt" title title-slide) "lib.rkt" racket/gui/base
    "tslide.rkt"

    racket/runtime-path (except-in mzlib/etc identity) unstable/gui/slideshow)
> (require (prefix-in p: racket-poppler))
> (define (page->pict pth #:rotate [r 0] #:scale [factor 1] [page 0])
    (bitmap (scale (rotate (p:page->pict (p:pdf-page (p:open-pdf pth) page)) r) factor)))
> (define (scale-h p)
    (scale p (/ (+ margin margin client-h) (pict-height p))))
> (define (scale-w p)
    (scale p (/ (+ margin margin client-w) (pict-width p))))
> (define (pic fname [r 1])
    (pslide #:go (coord .5 .5 'cc) (scale-h (bitmap fname))))
> █

```


Transient Typed Racket (again)

Summary

This RFC adds two new languages to Typed Racket (TR): shallow and optional. These languages use the same static types as TR to compile a program, but have different opinions about what types mean when a program runs.

Here is a quick comparison, using the declaration `(: str* (Listof String))` to ground the discussion.

(Remember, the value of `str*` can come from untyped code, so asking "what do types mean?" really asks which untyped values can enter this variable and which values lead to a run-time error.)


1. Normal `#lang typed/racket` offers *Deep* types. The whole static type means something at run-time. For `str*`, the value is guaranteed to be a list of only strings.
2. New `#lang typed/racket/shallow` gives weaker, *Shallow* types. Only the outer-most part of the type, the constructor, means something. For `str*`, the value is guaranteed to be a list, but there is no guarantee about the elements. That said, if our typed code takes `(car str*)` then the result has type `String` and is guaranteed to be a string --- otherwise the program raises an error.
3. New `#lang typed/racket/optional` is like the `no-check` languages, but actually type-checks code. These *Optional* types mean nothing at run-time. The value of `str*` could be anything.

For the rest of this RFC, the focus is on Shallow types and how we plan to implement them using the *Transient* semantics. The idea with Transient is to rewrite all typed code with simple assertions about the shape of values.

By contrast to Transient, normal TR gets Deep types through the Guarded (aka Natural) semantics. The idea with Guarded is to keep a strict

Typed Racket with Kinds

Kinding #1143

 Draft

capfredf wants to merge 24 commits into `racket:master` from `capfredf:kinding` 

 Conversation 5

 Commits 24

 Checks 4

 Files changed 50



capfredf commented on Sep 24 • edited ▼

racket's sponsor

Member



a kind system for typed racket

Current status: More than a "proof of concept". The new changes dovetail nicely with the existing infrastructure. Most of the existing tests pass without changes, though some tests has moved or will move from "succeed" to "fail".

Todos:

☒ a lot of code refactoring

☐ a RFC ([rendered](#))



 2



capfredf marked this pull request as draft last month

Rhombus

Example Rhombus/Shrubbery Programs #186

samdphillips started this conversation in **Show and tell**



samdphillips 9 days ago



Hi all,

One activity that almost anyone can work on is writing programs in the current Rhombus/Shubbery prototype and sharing their experience is using it. This feedback can be useful in finding deficiencies in the design, help direct where standard library should go, and show others what a Rhombus future might look like.

If you are not sure what you could write here's a short list of suggestions:

- Porting the server from ["More: Systems Programming with Racket"](#)
- If you have a collection of personal scripts try porting some of them.
- Puzzles and Code Katas

As an example here is an incredibly basic script that I wrote yesterday to install a bunch of packages in a new Racket install I had with some notes.

```
#lang rhombus

/*
1. It's been a few months since I've written and Rhombus. Looks like the keyword syntax changed since last time.
2. Keyword arguments are ~keyword:, but the `~to` in rename doesn't need a `:`?
3. Don't forget to use the prefix.
4. What is the lexical syntax for symbols?
   - Just use the escape hatch it seems. Ok that doesn't work.
   - No lexical syntax for `quote`, so use the `symbol(...)` special form.
   - Oh and if it has hyphen use #{ }
*/

import:
  pkg:
    rename:
```



Thank You