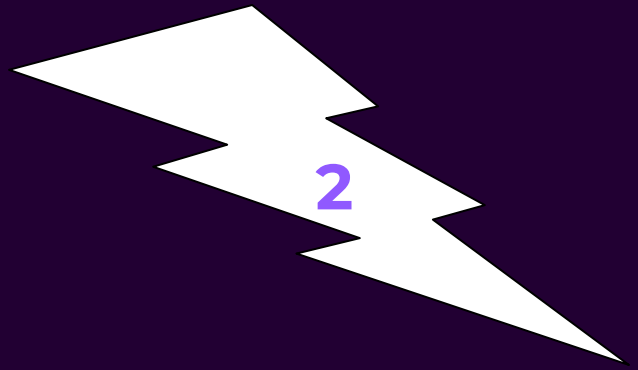
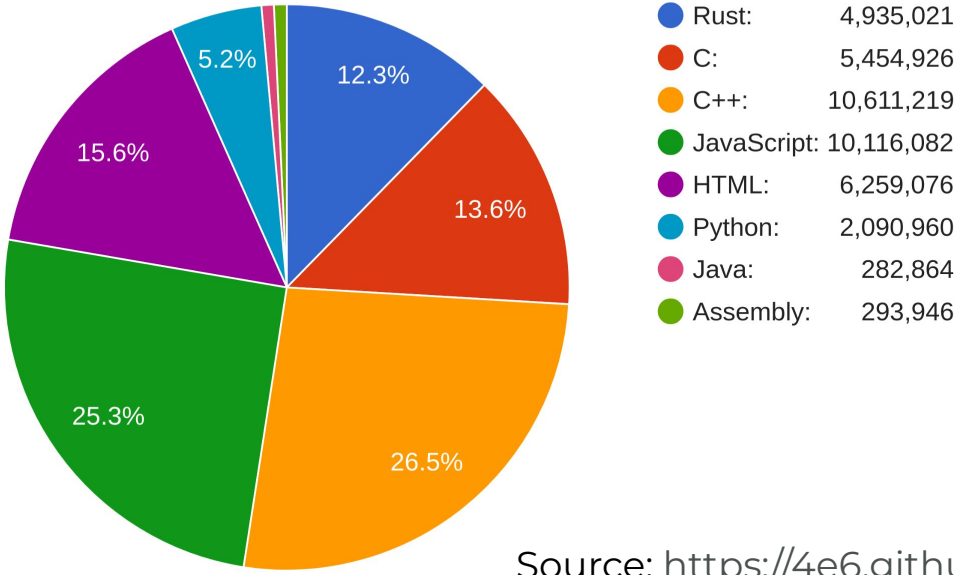


Adding a Rust library to Gecko

Manuel Bucher

Language stats

Firefox languages in SLOC



Source: https://4e6.github.io/firefox-lang-stats/, 2025-07-08

This wouldn’t have happened if it were written in Rust

- Use after free bugs: `kungFuDeathGrip`
- Data races: “Do I need to lock a Mutex?”
- Nullptr dereferences. “Can this pointer I’ve got passed be null?”
- Iterator invalidation
- Other undefined behavior

Other Rusty reasons

- Very well designed language
- Great documentation infrastructure
- Awesome compiler errors

https://lsd-ucsc.github.io/ChoRus/

Case study: RIIR “Copy Clean Link”

Patch: D225390

moz.build: Functionality crate

```
FINAL_LIBRARY = "xul"
```

moz.build: Binding crate with xpcom & cbindgen

```
FINAL_LIBRARY = "xul"
EXPORTS.mozilla += [
    "UrlStrip.h",
]
if CONFIG["COMPILE_ENVIRONMENT"]:
    CbindgenHeader(
        "UrlStrip_ffi.h",
        inputs=["/toolkit/components/antitracking/urlstrip_glue"]
    )
    EXPORTS.mozilla += [
        "!UrlStrip_ffi.h",
    ]
```


Writing bindings

Two ways of calling Rust library code. You can use both :)

cbindgen	Optional dependency to XPCOM
+ Necessary for creating objects + Native C++ bindings - Only callable from C++	+ Callable from JS and C++ + Can use `nsstring`, `nsI…` - can't use `#[test]`[1]

Maybe in future cxx: Bug 1921139 - Allow creating C++ bindings for Rust code with cxx (unlikely)

[1]: https://firefox-source-docs.mozilla.org/testing-rust-code/index.html#rust-tests

Further files for bindings crate

cbindgen.toml: Get inspired by other cbindgen.toml files on searchfox.

UrlStrip.h: Somewhat nicer C++ wrapper around auto generated symbols

```
#include <mozilla/UrlStrip_ffi.h>
struct UrlStrip {
public:
 static void Create(UrlStrip** pThis) { urlstrip_new(pThis); }
 void Delete() { urlstrip_drop(this); }

 void Init() { urlstrip_init(this); };
 void Uninit() { urlstrip_uninit(this); };

 nsresult Strip(const nsACString& aURI, nsCString* aOutURI) {
   return urlstrip_strip(this, &aURI, aOutURI);
 }
 bool CanStrip(const nsACString& aURI) {
   return urlstrip_canstrip(this, &aURI);
 }

private:
 UrlStrip() = delete;
 ~UrlStrip() = delete;
 UrlStrip(const UrlStrip&) = delete;
 UrlStrip& operator=(const UrlStrip&) = delete;
};
```


Other files to touch

toolkit/library/rust/shared/Cargo.toml

```
urlstrip = { path = "../../../components/antitracking/urlstrip" }
urlstrip_glue = { path = "../../../components/antitracking/urlstrip_glue" }
```

toolkit/library/rust/shared/lib.rs

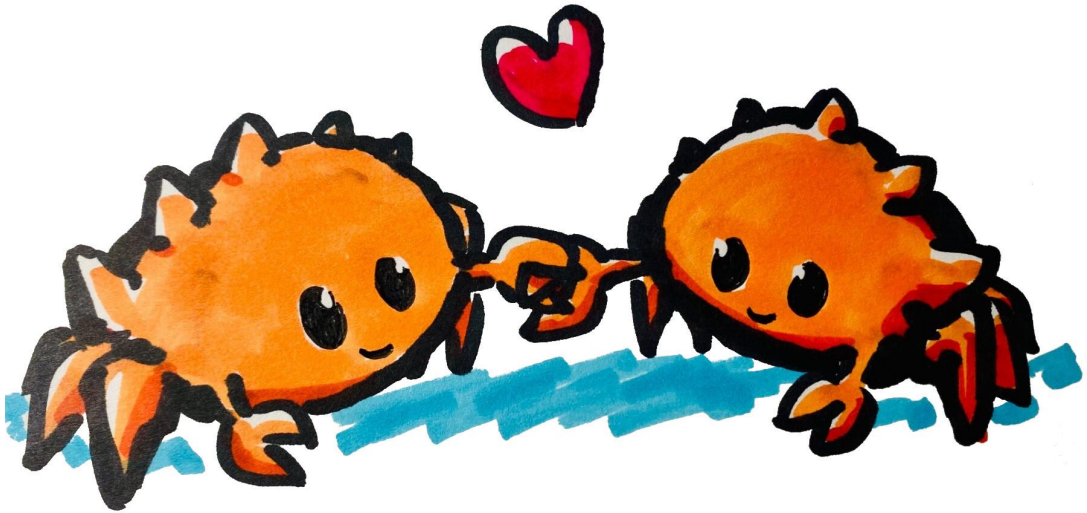
```
extern crate urlstrip;
extern crate urlstrip_glue;
```


Further documents

- https://firefox-source-docs.mozilla.org/build/buildsystem/rust.html
- https://firefox-source-docs.mozilla.org/writing-rust-code/
- https://manuelbucher.com/blog/rust-gecko/

https://rustacean.net/more-crabby-things/rustdocs.png

Questions?



https://wandering.shop/@aldeka/109301243634789356