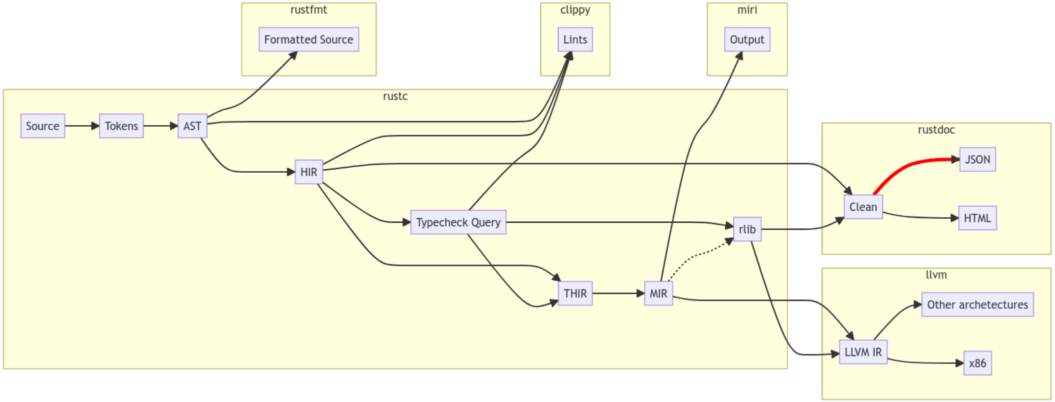
Rustdoc JSON

<alona.page/talks/rustdoc-json-2023-09-08.pdf>

Alona Enraght-Moony

2023-09-08

Rustdoc JSON: A small part of a large system



Rust: The 10,000 foot view

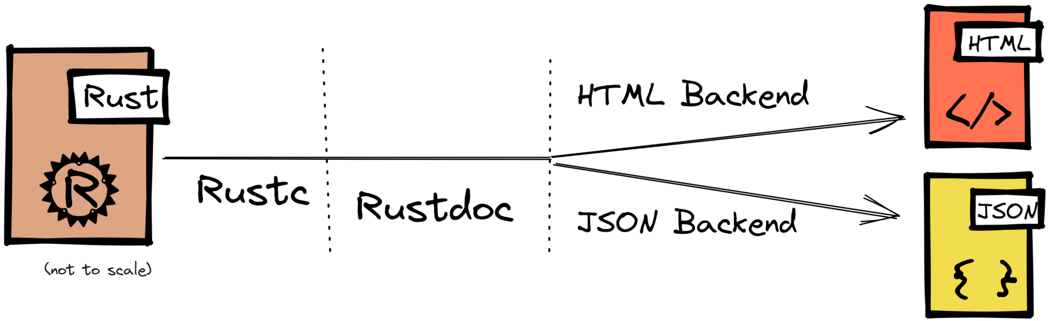
- Originally out of Mozilla, now large multi-org team
- Realty interesting in langugе design, but not the subject of this talk
- Right language for many problems, but **not the one's we face**

Rustdoc: The 10,000 foot view

- Documentation generator for rust
- Sits on top of `rustc`
- Kind of like `godoc`/`pkgsite`
- Kind of like `perldoc`
- `docs.rs` is an equivalent to `pkg.go.dev`/`metacpan.org`

Rustdoc JSON: The 10,000 foot view

- If rustdoc is a rust → HTML compiller, then rustdoc-json is a rust→JSON compiller.
- **Core Insight**: Computers also need docs, but for need a different format.
- Who uses this:
  - roogle
  - `cargo public-api`:
  - `cargo check-external-types`
  - `cargo semver-checks`



The easy case: Crates, Modules and Structs

```
pub struct Foo;
pub mod bar {
    pub struct Baz;
}
```

A simple schema for our simple language

```
enum Item {
    Struct {name: String},
    Module {name: String, items: Vec<Item>},
}
```

```
{
    "kind": "module",
    "name": "somelib",
    "items": [
        {"kind": "struct", "name": "Foo"},
        {
            "kind": "module",
            "name": "bar",
            "items": [{"kind": "struct", "name": "Baz"}]
        }
    ]
}
```

A simple standard library

```rust
pub mod collections {
    pub mod vec { pub struct Vec; }
    pub mod hash_map { pub struct HashMap; }
    pub mod hash_set { pub struct HashSet; }
}
```

- End up having `std::collections::hash_set::HashSet`
- But we want `std::collections::HashSet`

Andrew Koenig/Butler Lampson/David Wheeler to the rescue!

“We can solve any problem by introducing an extra level of indirection.”

```rust
pub mod collections {
    pub mod vec { pub struct Vec; }
    pub mod hash_map { pub struct HashMap; }
    pub mod hash_set { pub struct HashSet; }

    pub use vec::Vec;
    pub use hash_map::HashMap;
    pub use hash_set::HashSet;
}
```

- `std::collections::HashSet` and `std::collections::hash_set::HashSet` now both valid paths.

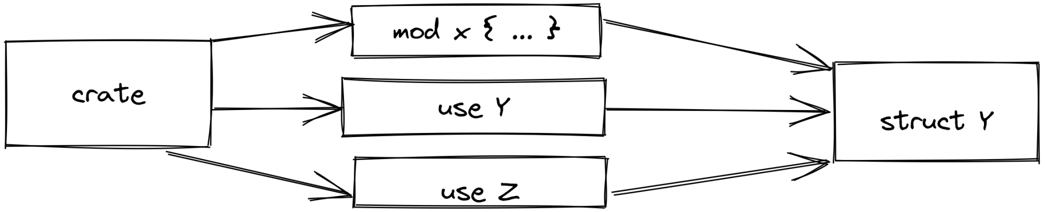
Surely this won't explode into a mountain of complexity

```
pub mod x {
    pub struct Y;
}
pub use x::Y;
pub use x::Y as Z;
```

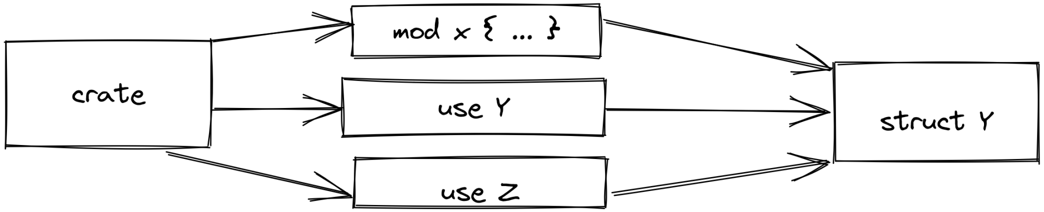
- `somelib::x::Y`, `somelib::Y` and `somelib::Z` all resolve to same item.

Oh no, that's a graph!

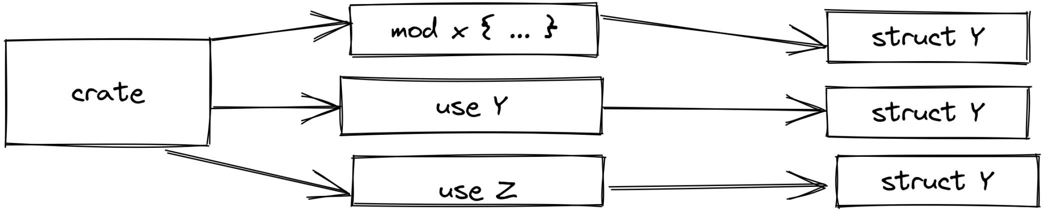
```
pub mod x {
    pub struct Y;
}
pub use x::Y;
pub use x::Y as Z;
```



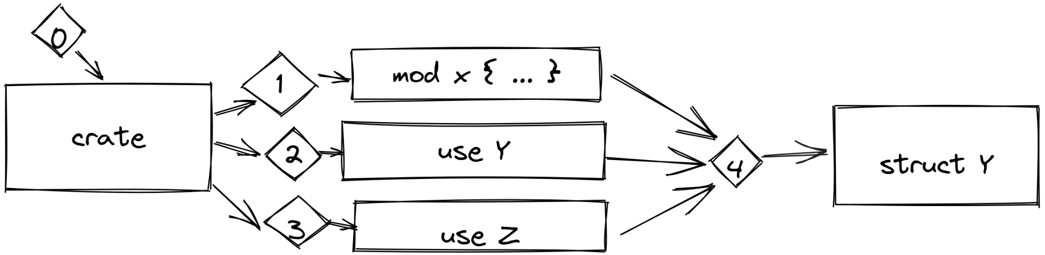
Money doesn't grow on trees, but JSON does



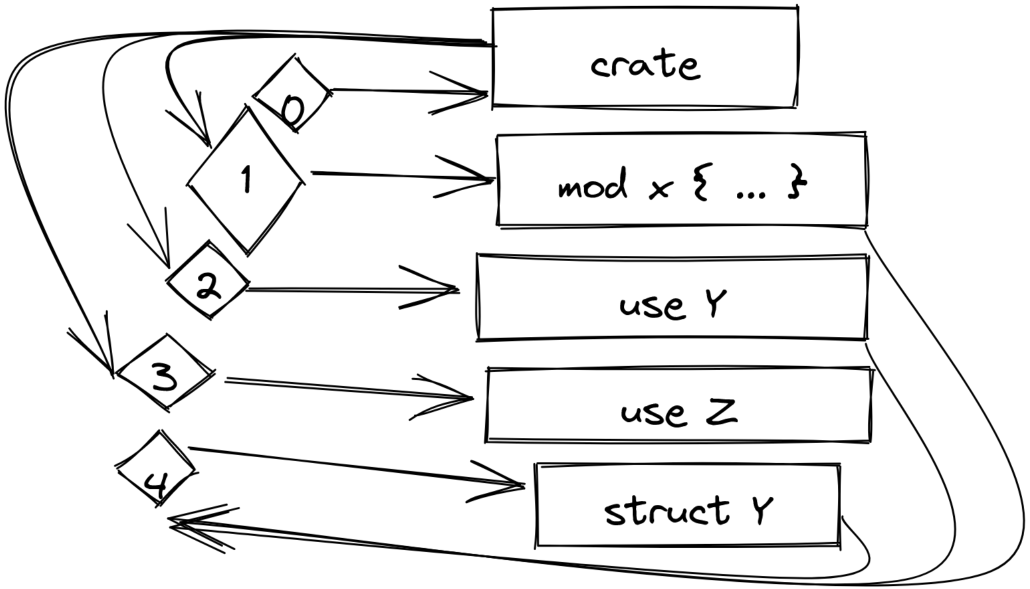
becomes



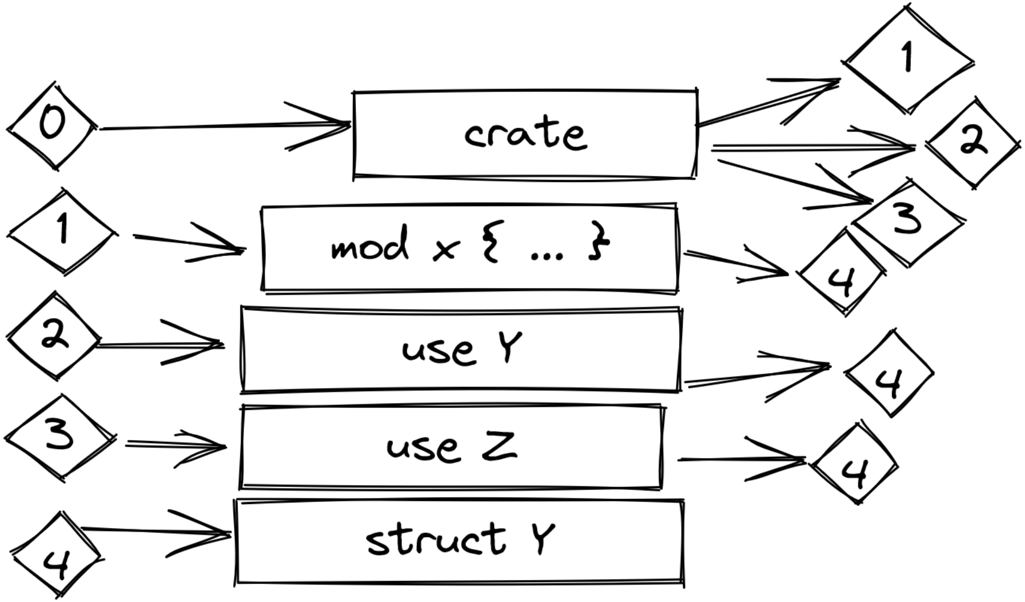
What if we try indirection again?



Rotate a graph in your mind



And like that, we have a map



Obvious JSON Output

```
{
"index": {
 "0": {
  "inner": {"items": ["1", "2", "3"]},
  "kind": "module",
  "name": "cratename"
 },
 "1": {"kind": "module", "name": "x",  "inner": {"items": ["4"]}},
 "2": {"kind": "import", "name": null, "inner": {"id": "4", "name": "Y"}},
 "3": {"kind": "import", "name": null, "inner": {"id": "4", "name": "Z"}},
 "4": {"kind": "struct", "name": "Y",   "inner": {}}
},
"root": "0"
}
```

Other things not covered

- `cfg`/`cfg(doc)`: feature/target dependant API's
- Projections/Normailzation
- Cross-crate ID resolution
- Macros
- Versioning/Evolution
- Testing
- Unnamable Types
- Infinatly long paths
- Maintentenence/Stewardship/Bus Factor
- Stabilization/Unstable features

Conclussion

- Design decissions have unforseen consequences.
- Someone always pays for the complexity.

Thanks

Alex Kladov, Didrik Nordström, Guillaume Gomez, Jacob Hoffman-Andrews, Joseph Ryan, Jynn Nelson, León Orell Valerian Liehr, Luca Palmieri, Martin Nordholts, Michael Goulet, Michael Howell, Noah Lev, QuietMisdreavus, Rune Tynan, Tyler Mandry, Urgau
Excalidraw icons by xxxDeveloper. MIT Licensed.

Links

- Implementation: github.com/rust-lang/rust/tree/master/src/librustdoc/json
- Public API: docs.rs/rustdoc-types/latest/rustdoc_types/
- RFC (now outdated in specifics): rust-lang.github.io/rfcs/2963-rustdoc-json.html
- Bugs: github.com/rust-lang/rust/issues?q=is:Aopen+is:issue+label:A-rustdoc-json

Bonus Slides: Infinatly Long Paths

```rust
pub mod cx {
    pub use super::cx as cx;
    pub struct Cx;
}
```

- `cx::Cx`, `cx::cx::Cx`, `cx::cx::cx::Cx`, .. are all valid.
- So can't map `Path` -> `Item`, would be infinatly large.
- Push complexity to consumers.

Bonus Slides: Unnamable Types

```rust
mod private {
    pub struct Bar;
}
pub fn get_bar() -> private::Bar { private::Bar }
```