

Pixi,

the Missing Companion to Cargo

About Me

-  Julian Hofer
-  Background in Physics
-  Thinks that languages are cool
-  Loves to talk about dependency management



Scientific Rust

- Strengths
 - Speed
 - Thread safety
 - Memory safety
- Weaknesses:
 - Steep learning curve
 - Difficult to depend on libraries *not* written in Rust
 - Lack of high quality libraries



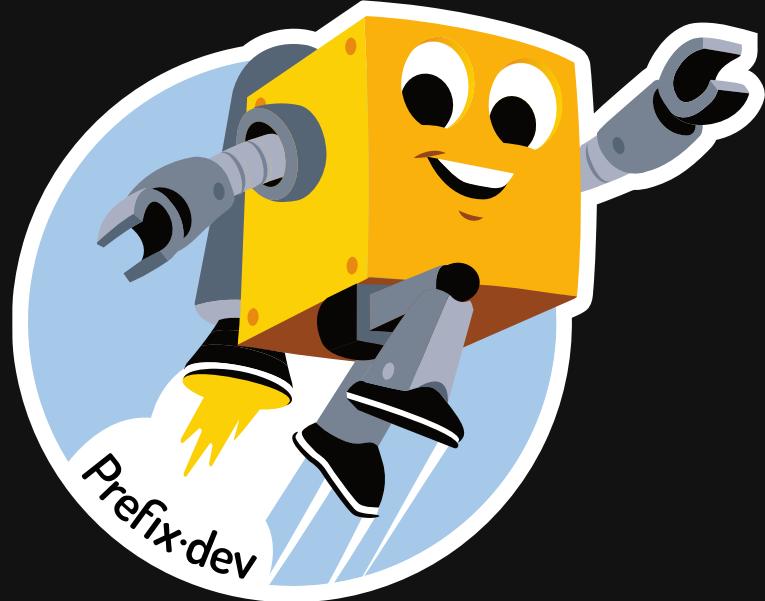
Cargo is Not Enough

- No task runner
 - External tools are needed for pipelines
- No easy way to depend on scientific libraries
 - Rewrite in Rust (not feasible for many projects)
 - Compile them yourself in `build.rs`
 - Where does the compiler come from?
 - Detect the library on your system in `build.rs`
 - Where does the library come from?
 - Use external tool



Introducing Pixi

-  Conda package ecosystem:
 - Cross-platform (Windows, macOS, Linux, ...)
 - Cross-language (C, C++, Fortran, Python, ...)
-  Fast & Open Source
-  Workflow management
-  Multi-environments
-  Reproducible thanks to lock-files



Pixi Workflow

Cargo.toml

```
[package]
edition = "2024"
name = "geos-rs"
version = "0.1.0"

[dependencies]
geos = "10.0"
```

main.rs

```
use geos::{Error, Geom, Geometry};

fn main() -> Result<(), Error> {
    // Create square with length of 5
    let square = Geometry::new_from_wkt(
        "POLYGON ((0 0, 0 5, 5 5, 5 0, 0 0))",
    )?;
    // 5 x 5 should be 25!
    println!("Area : {}", square.area()?);
    Ok(())
}
```

pixi.toml

```
[project]
channels = [ "conda-forge" ]
name = "geos-rs"
platforms = [ "linux-64", "osx-64", "osx-arm64", "win-64" ]

[tasks]
start = "cargo run"

[dependencies]
rust = "~1.87.0"
```

\$ pixi run start

```
thread 'main' panicked at geos-sys-2.0.6/build.rs:137:13:
Could not detect GEOS using pkg-config or geos-config
```

Pixi Workflow

Cargo.toml

```
[package]
edition = "2024"
name = "geos-rs"
version = "0.1.0"

[dependencies]
geos = "10.0"
```

main.rs

```
use geos::{Error, Geom, Geometry};

fn main() -> Result<(), Error> {
    // Create square with length of 5
    let square = Geometry::new_from_wkt(
        "POLYGON ((0 0, 0 5, 5 5, 5 0, 0 0))",
    )?;
    // 5 x 5 should be 25!
    println!("Area : {}", square.area()?);
    Ok(())
}
```

pixi.toml

```
[project]
channels = ["conda-forge"]
name = "geos-rs"
platforms = ["linux-64", "osx-64", "osx-arm64", "win-64"]

[tasks]
start = "cargo run"

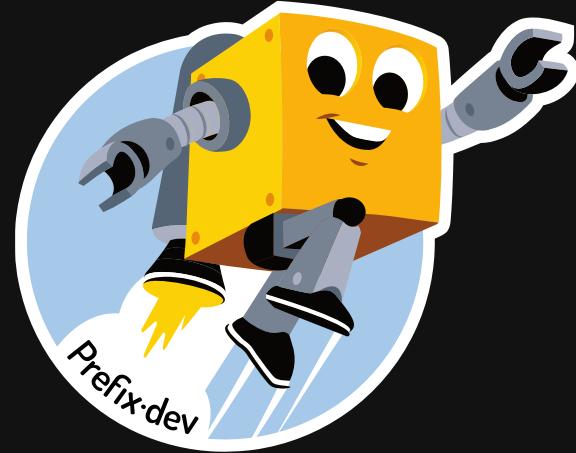
[dependencies]
geos = "~=3.13.0"
pkg-config = "~=0.29.0"
rust = "~=1.87.0"
```

\$ pixi run start

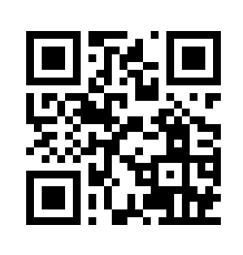
Area : 25

Conclusion

- **Scientific Rust** needs more than just Cargo
- **Conda ecosystem**: Provides binary packages for multiple platforms and programming languages
- **Pixi**: Package and workflow manager, built on the Conda ecosystem



Thank you for your attention!



Pixi Website



LinkedIn



Discord