LAAS CNRS

Capitole du Libre 2023

Robotique en Rust

Capitole du Libre 2023

2023-11-18

Guilhem Saurel

cnrs

Université Fédérale Toulouse Midi-Pyrénées

LAAS CNRS

Cette présentation

Disponible à

`https://homepages.laas.fr/gsaurel/talks/robotique-rust.pdf`

Sous license

https://creativecommons.org/licenses/by-sa/4.0/

Cette présentation (suite)

Source

`https://gitlab.laas.fr/gsaurel/talks : robotique-rust.md`

Discussions & Projet accompagnant

https://github.com/nim65s/RobotS

Présentation

Figure 1: IR en robotique humanoïde

Contexte

https://www.youtube.com/watch?v=1-TDT7pREOg

Rust

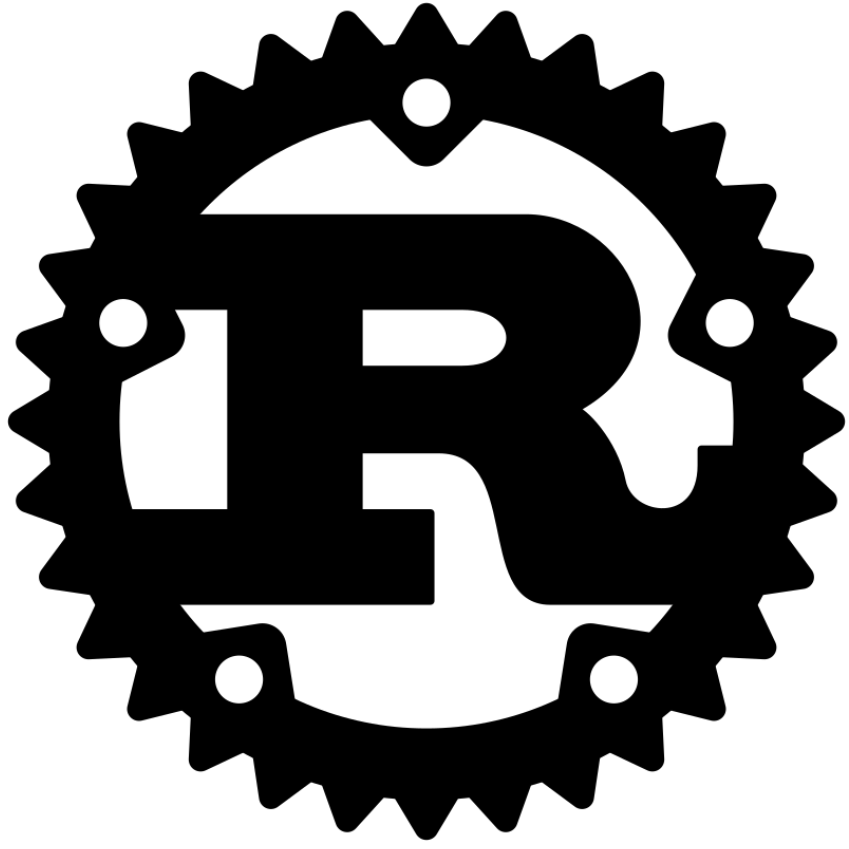

Figure 2: Rust

Structures de données

```rust
struct Pose {
    x: f64,
    y: f64,
}

enum Mode {
    Idle,
    Move(Pose, f64),
}

struct Robot {
    name: String,
    pose: Pose,
    mode: Mode,
}
```

Traits

```
pub trait SetSpeed {
    fn set_speed(&mut self, tgt: f64) -> Result;
}
```

Traits

```rust
pub trait SetSpeed {
    fn set_speed(&mut self, tgt: f64) -> Result;
}

impl SetSpeed for Robot {
    fn set_speed(&mut self, tgt: f64) -> Result {
        match self.mode {
            Mode::Idle => Err(Error::IsIdle),
            Mode::Move(pose, _) = {
                self.mode = Mode::Move(pose, tgt);
                Ok(())
            }
        }
    }
}
```

Async

```rust
#[tokio::main]
async fn main() -> Result {
    let port = 1234;
    tokio::spawn(async move {
        serve(port).await;
    });

    let mut client = Client::connect(port).await?;
    client.set("hello", "world").await?;

    let result = client.get("hello").await?;
    println!("got {result:?}");

    Ok(())
}
```

Channels

```rust
#[tokio::main]
async fn main() {
    let (tx, mut rx) = mpsc::channel(32);

    tokio::spawn(async move {
        tx.send("sending from first handle").await;
    });

    while let Some(message) = rx.recv().await {
        println!("GOT = {}", message);
    }
}
```

Rust Embedded

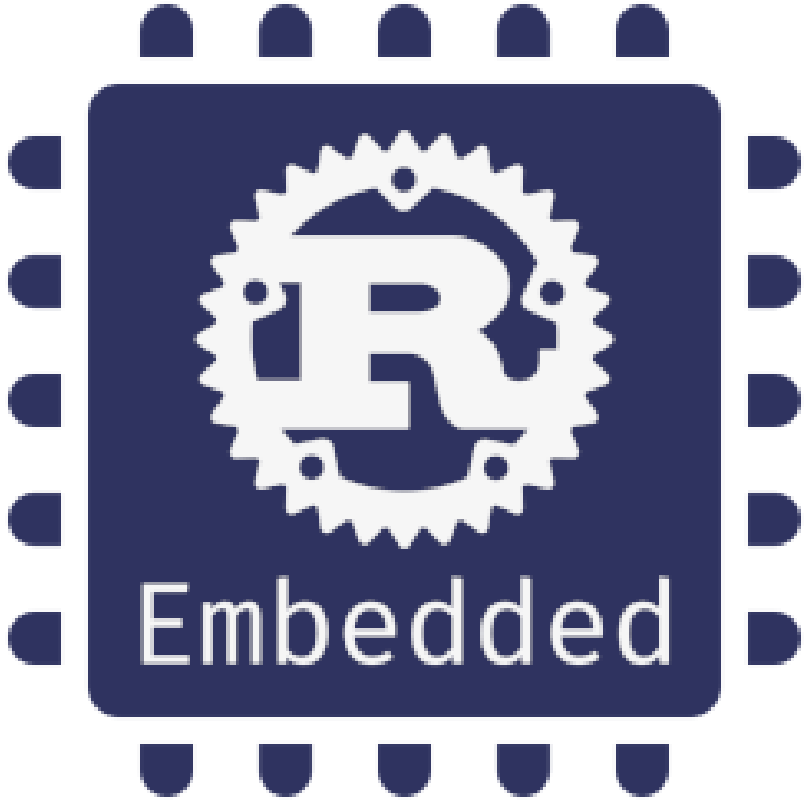

Figure 3: Rust Embedded

Ecosystème

- PAC: Peripheral Access Crates
- HAL: Hardware Abstraction Layers

Les sont implémentés avec les traits fournis par `embedded-hal`, et fonctionnent partout !

Ecosystème

- PAC: Peripheral Access Crates
- HAL: Hardware Abstraction Layers

Les sont implémentés avec les traits fournis par `embedded-hal`, et fonctionnent partout !

ref. https://www.youtube.com/watch?v=vLYit_HHPaY

Frameworks

- RTIC
- Embassy

Web

Leptos

Figure 4: Fast full-stack web apps in Rust

Démo !

LAAS CNRS

toy example

https://github.com/nim65s/RobotS

Questions ?

Merci de votre attention !