

2RM, École Technologique 2023

2023-05-15

Guilhem Saurel

Available at

[https://homepages.laas.fr/gsaurel/  
rust-rovers.pdf](https://homepages.laas.fr/gsaurel/rust-rovers.pdf)

Under License



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

## Source

[https://gitlab.laas.fr/gsaurel/talks :  
rust-rovers.md](https://gitlab.laas.fr/gsaurel/talks : rust-rovers.md)

## Discussions & Companion Project

<https://github.com/nim65s/RobotS>

# Présentations



Figure 1: IR en robotique humanoïde

Rovers

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

Rust

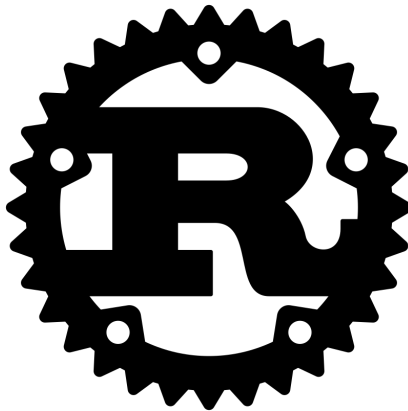


Figure 2: Rust

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

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

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

```

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(())
            }
        }
    }
}

```

```
#[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(())
}
```

```
#[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

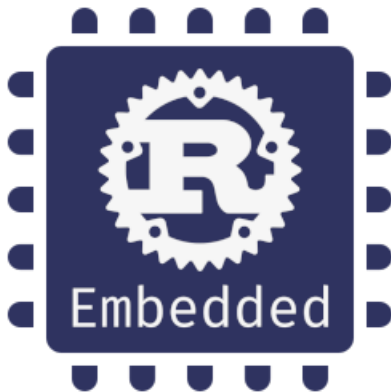


Figure 3: Rust Embedded

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

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

- RTIC
- Embassy

Web

The word "Leptos" is rendered in a stylized font. The letters 'L', 'e', 'p', 't', and 's' are dark blue. The letter 'o' is red and is stylized as a circle with a vertical line extending upwards to a smaller red circle, and a curved line extending downwards and to the left, ending under the 'p'.

Figure 4: Fast web apps in Rust