edhouse

Výhody Rustu, o kterých 'nikdo nemluví'

github.com/marekpsenka/rust-advantages

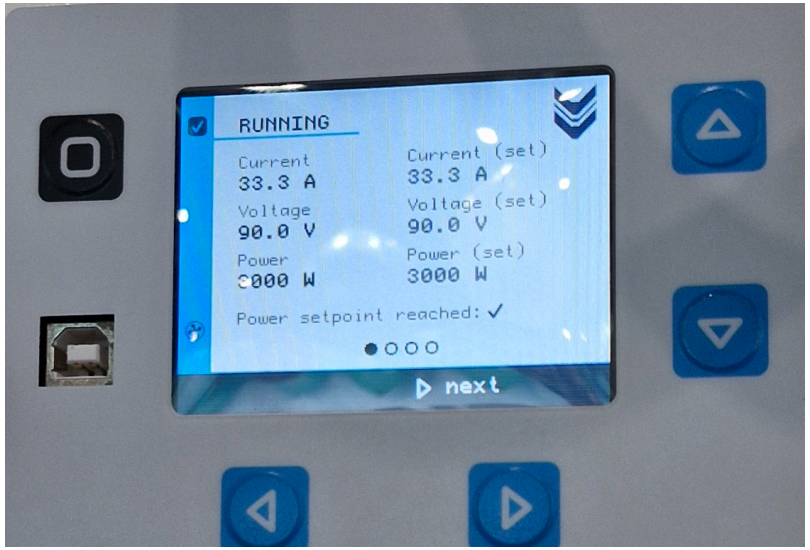
edhouse

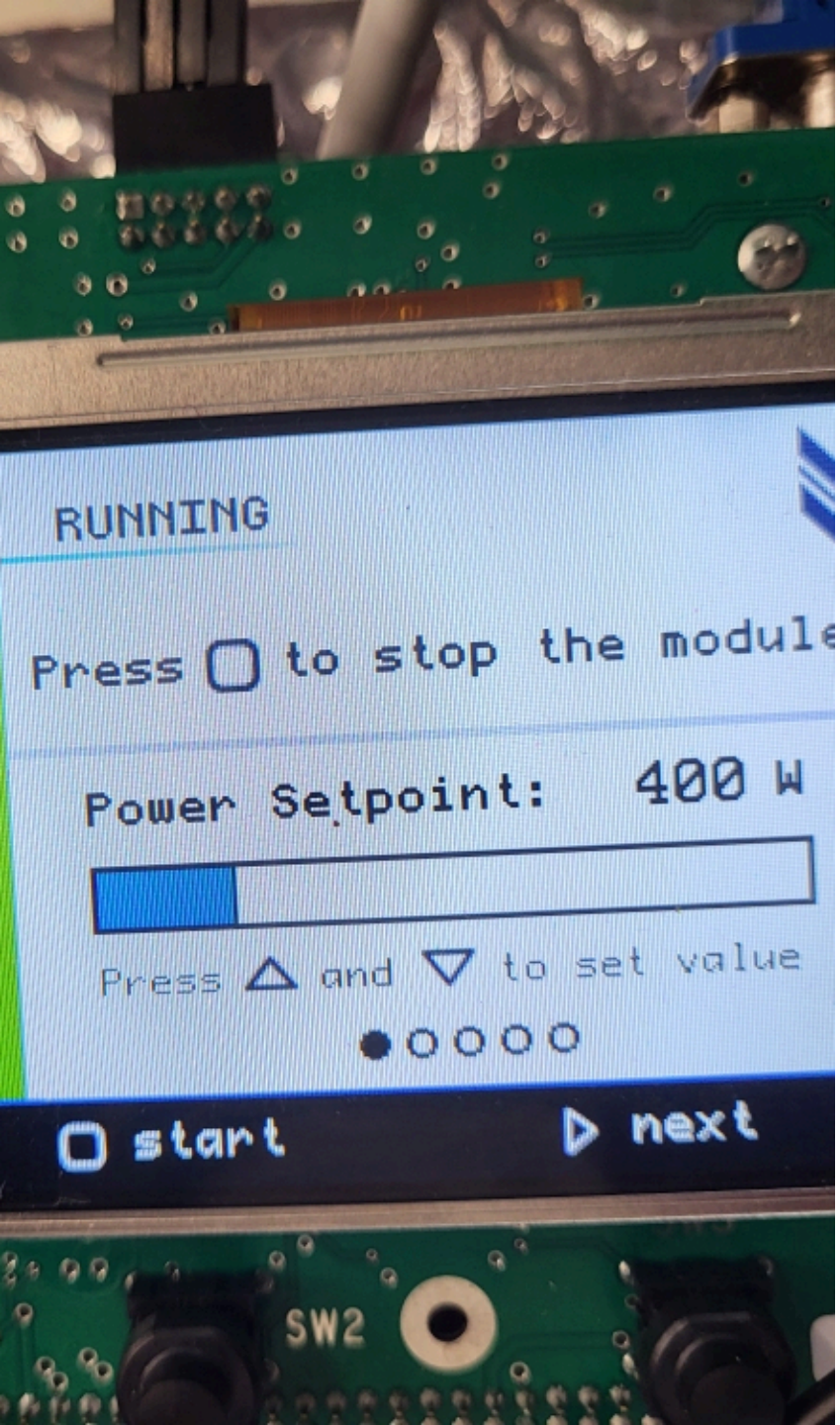
Marek Pšenka

- Technický vedoucí v Edhouse
- 7 let zkušeností
- Většinu kariéry jsem pracoval s C++ a C#
- Rust používám již dva roky

Generátor vodíku H2Gem

- Zařízení pro výrobu zeleného vodíku
- Kolegové v Edhouse vyvinuli kompletní firmware
- Rust jim významně pomohl se spolehlivostí





H2Gem technicky

Řešené úlohy:

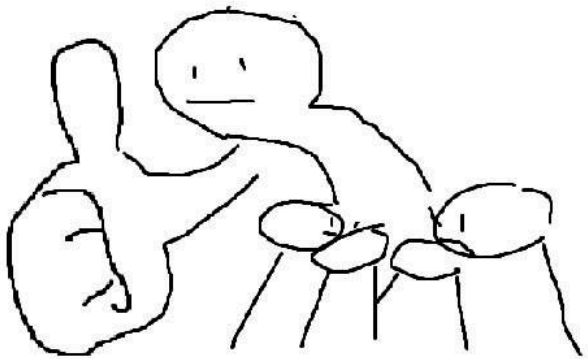
- komunikace a řízení zdroje elektrické energie
- komunikace se senzory a nadřazeným systémem
- zobrazení a vstupy na/z grafického displeje
- vše na platformě STM32.

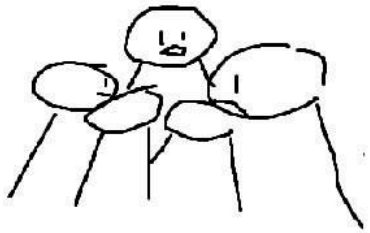
Role Rustu:

- Celé řešení, včetně ovladačů pomocí RTIC
- žádné runtime chyby v průběhu vývoje a testování
- rychlejší obrátky na HW

Rust is the best language ever.
We should rewrite everything in Rust.
Rust is the future.

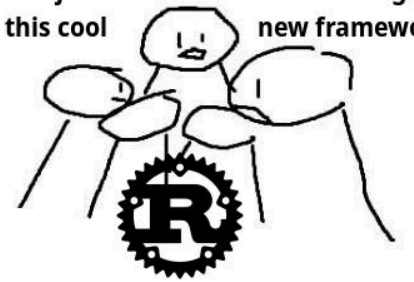
Other Developers working
with other languages

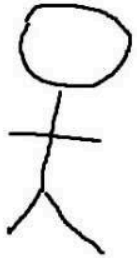


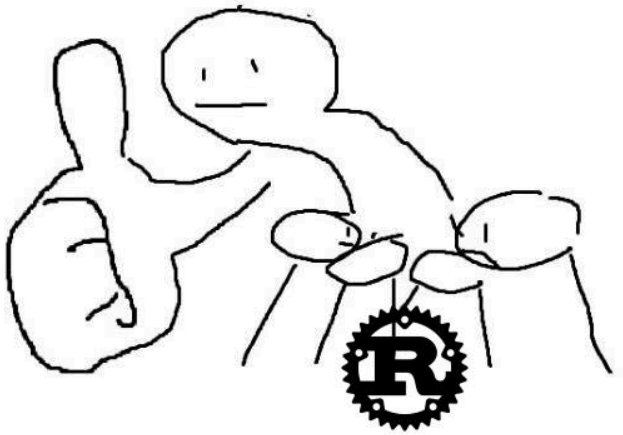


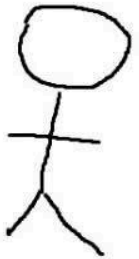
Rust isn't the right tool for everything, stop loving Rust so much

So I just made this website using this cool new framework

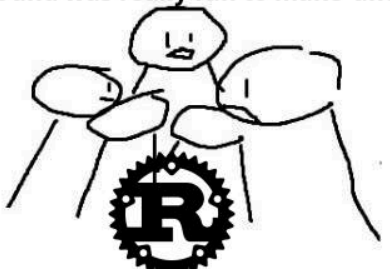








Then I made a CLI tool that helps me a lot and was really fun to make and...



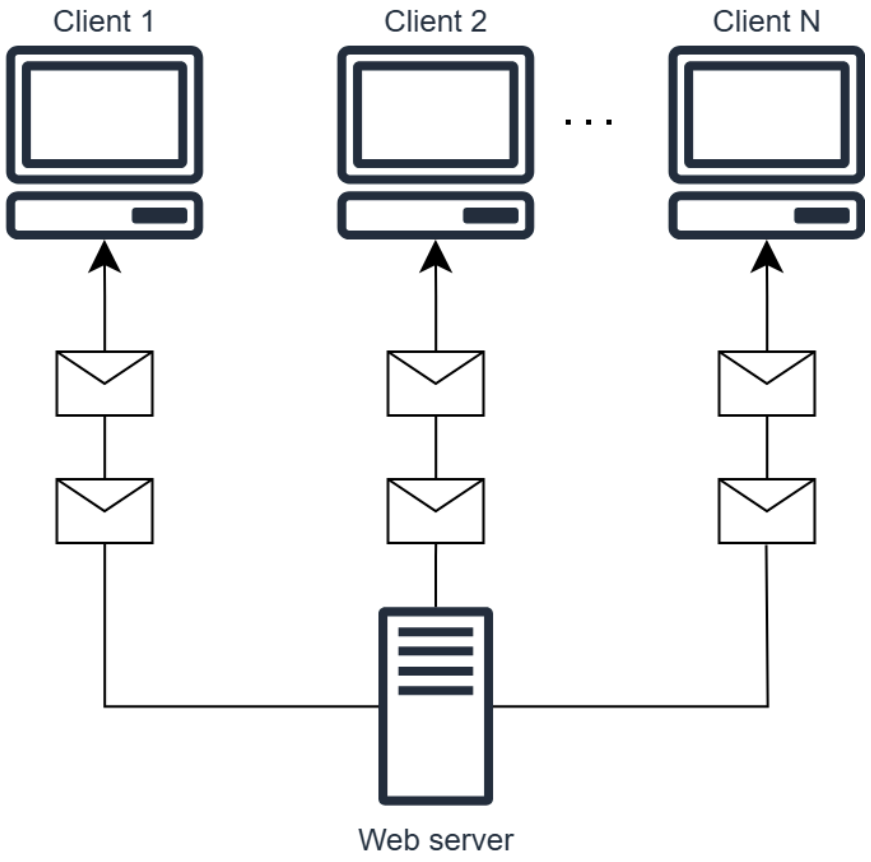
Tato přednáška

Záměrně se vyhneme srovnání s jinými jazyky

Obejdeme nejčastěji skloňované přednosti = výkon a paměťovou bezpečnost

Zaměříme se na přednosti, o kterých 'nikdo nemluví'

- Souběžnost bez obav (Fearless Concurrency)
- Živý ekosystém a komunita
- Silná makra and generiky
- Práce s chybami



Server-sent Events (SSE)

```
c:\code\rust-advantages>cargo run
    Finished `dev` profile [unoptimized + debuginfo] target(s) in 0.09s
     Running `target\debug\example-server.exe`
http://localhost:3000
```

```
client> curl -N http://localhost:3000/events

event: beep
data: {"counter_value":7}

event: beep
data: {"counter_value":8}

...
```

Souběžnost bez obav (Fearless Concurrency)

Co můžeme neohroženě (fearlessly) napsat jinde?

```
function f(integer& n)
{
    ++n;
}

function main () {
    integer n = 0;
    thread my_thread(f, &n);
    my_thread.join();
    print(n);
}
```

Rust potřebuje víc, aby zůstal v klidu (fear-less)

```rust
fn f(n_container: Arc<Mutex<i32>>) {
    let mut n = n_container.lock().expect("Lock is not poisoned");
    *n += 1;
}

fn main() {
    let n_container = Arc::new(Mutex::new(0i32));
    let container_clone = n_container.clone();
    let my_thread = std::thread::spawn(move || {
        f(container_clone);
    });
    _ = my_thread.join();
    let n = n_container.lock().expect("Lock not poisoned");
    println!("{}", *n);
}
```

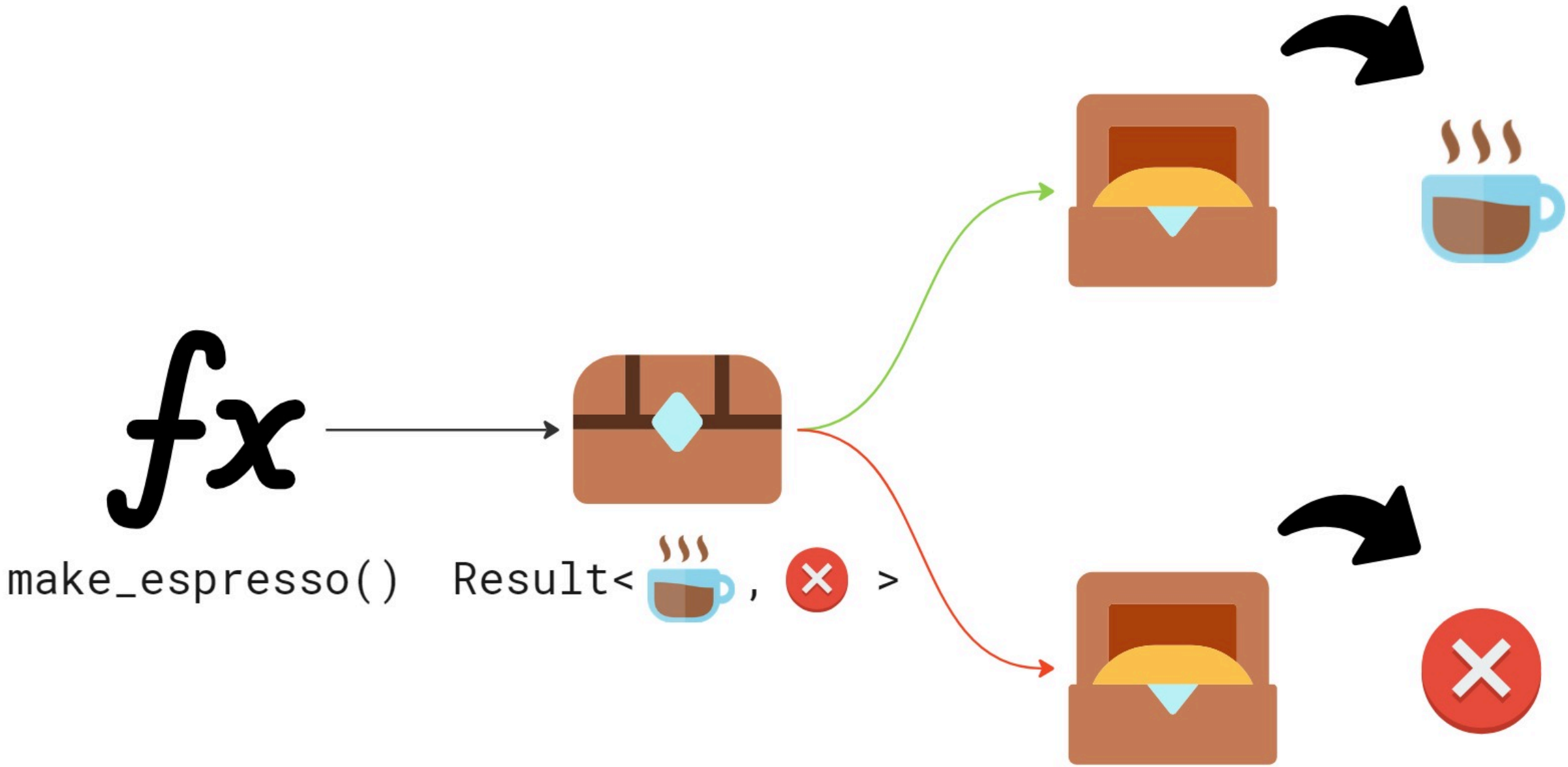
```rust
pub struct CoffeeMachine {
    water_tank_volume: f64,
    available_coffee_beans: f64,
}

impl CoffeeMachine {
    pub fn make_espresso(&self) -> Result<Espresso, String> {
        if self.water_tank_volume < 25.0 {
            Err("Not enough water in tank".to_string())
        } else if self.available_coffee_beans < 7.0 {
            Err("Not enough coffee beans".to_string())
        } else {
            Ok(Espresso {})
        }
    }
}
```

edhouse

```rust
#[test]
fn error_returned_when_making_espresso_without_beans() {
    let machine = CoffeeMachine {
        water_tank_volume: 300.0,
        available_coffee_beans: 2.0,
    };

    let result = machine.make_espresso();
    assert!(result.is_err());
    assert_eq!(result, Err("Not enough coffee beans".to_string()));
}
```


fx
make_espresso()
Result<
,
>

edhouse

Filozofie

Myšlenka vyhradit prostor pro chybové informace v návratové hodnotě není nová

```c
int main(void)
{
    FILE *f = fopen("non_existent", "r");
    if (f == NULL) {
        perror("fopen() failed");
    } else {
        fclose(f);
    }
}
```

```
fopen() failed: No such file or directory
```

edhouse

Rust nám to usnadňuje

```rust
pub enum Result<T, E> {
    Ok(T),
    Err(E),
}
```

```rust
fn open_nonexistent_file() {
    match std::fs::File::open("non_existent") {
        Ok(file) => drop(file),
        Err(err) => println!("open() failed: {}", err),
    }
}
```

```
open() failed: The system cannot find the file specified. (os error 2)
```

edhouse

Porovnej

```c
int main(void) {
    FILE *f = fopen("non_existent", "r");
    if (f == NULL) {
        perror("fopen() failed");
    } else {
        fclose(f);
    }
}
```

```rust
fn open_nonexistent_file() {
    match std::fs::File::open("non_existent") {
        Ok(file) => drop(file),
        Err(err) => println!("open() failed: {}", err),
    }
}
```

edhouse

Jiná strategie - výjimky v C# - nejsou vidět

```
public static System.IO.FileStream Open (string path, System.IO.FileMode mode);
```

Kde se dozvím jak vypadá chyba? **V dokumentaci**:

> ArgumentNullException
> PathTooLongException
> (...)

Rust je explicitní. Dozvím se to **v kódu**:

```
pub fn open<P: AsRef<Path>>(path: P) -> std::Result<T, std::io::Error>;
```

Vyjímky střílí

```
void OpenNonexistentFile() {
    File.Open("non_existent", FileMode.Open);
}

OpenNonexistentFile();

DoSomethingElse();
```

```
C:\code\rust-error-handling\_examples_cs>dotnet run
Unhandled exception. System.IO.FileNotFoundException: Could not find file 'non_existent'.
(...)
```

Porovnej

edhouse

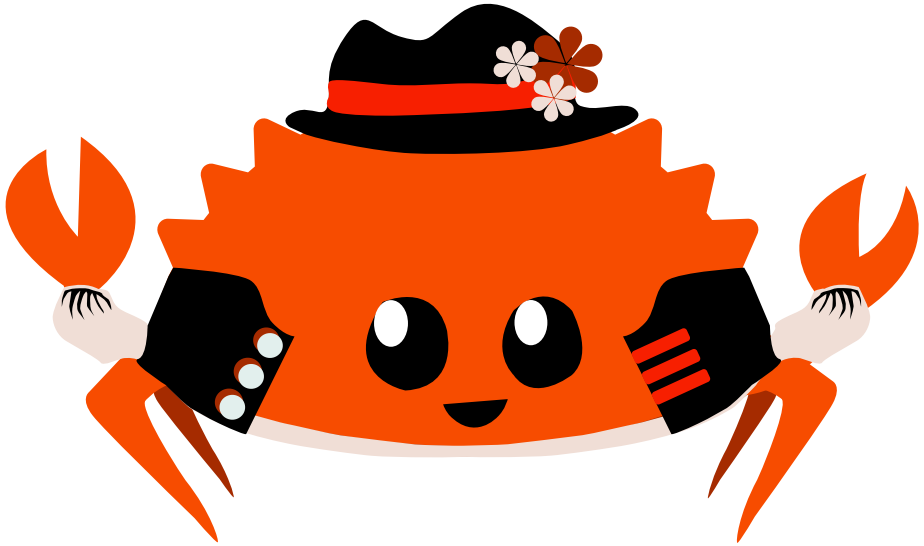
```
void OpenNonexistentFile() {
    try
    {
        File.Open("non_existent", FileMode.Open);
    }
    catch (Exception e) {
        Console.WriteLine($"{e}");
    }
}
```

```
fn open_nonexistent_file() {
    match std::fs::File::open("non_existent") {
        Ok(file) => drop(file),
        Err(err) => println!("open() failed: {}", err),
    }
}
```

edhouse

Shrnutí

- Rust nám na zákaznických projektech pomáhá psát spolehlivý kód
- Myšlenka vyhradit prostor pro chybové informace v návratové hodnotě není nová
- Rust nám to usnadňuje standardním typem `Result<T, E>`
- Příklad alternativní strategie jsou výjimky.
- Nejsou ale vidět a střílí - nutná bdělost



Rust Moravia