Rust: structs, methods, generics, traits, lifetimes

Madhavan Mukund, S P Suresh

Programming Language Concepts

Lecture 09, 11 February 2025

Structs, Methods

Objects

|

Tree

Interfaces

Callbacks

Iterators

Generics (Polymorphism)

Rust is not O-O

Inheritance

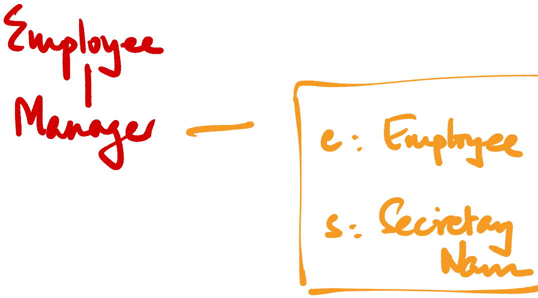
Why?

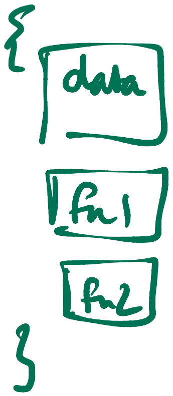
C1

|

C2

Structs, Methods





Structs, Methods

```
impl Rectangle{
  fn area(&self) {
     — self.width
  }
}
```

(*self).width

let rect1 = Rectangle{..}

rect1.area()

Not needed X (&rect1).area()

"Rectangle.area(&rect1)"

Generics

functions over multiple types $<T>$ fn(--)

data structures over multiple types Node$<T>$

Java - type variables S, T . .

$<T>$ $\forall T$

Rust is the same

Traits

Rust for "interface" – capability

Copy trait

$x = \boxed{y}$

type

If the type has Copy trait

by value ?

by reference ?

Lifetimes

Dangling pointer example

$x = \text{example}$

```
fn example {
    create s
    return &s
}
```