LESSON 10

main.rs

```
trait Area { fn area(&self) -> f64; }
impl Area for Circle { ... }
fn f(x: &impl Area) {}
```

Rust

Traits & Generics

Learn it all @MyWebUniversity



A FREE MYWEBUNIVERSITY COURSE

RUST PROGRAMMING

LESSON 10

Traits and Generics

Defining traits, default methods, generics, trait bounds, impl Trait and dyn Trait

Traits

main.rs

```rust
trait Describe {
    fn name(&self) -> String;
    fn describe(&self) -> String {          // default method
        format!("This is {}", self.name())
    }
}
impl Describe for Student {
    fn name(&self) -> String { self.name.clone() }
}
```

- A trait defines behavior that types can share, like an interface. Implement it for a type …
- Traits can provide default method bodies that implementers may override.

01

Standard Traits: Display and From

main.rs

```rust
use std::fmt;

impl fmt::Display for Student {
    fn fmt(&self, f: &mut fmt::Formatter) -> fmt::Result {
        write!(f, "{} ({:.1})", self.name, self.score)
    }
}
println!("{s}");           // Mariam (95.0)
let c: Celsius = Fahrenheit(98.6).into();
```

- Implement Display to control how your type prints with curly braces. Implement From to ...
- Iterator, Default, PartialOrd and Hash are other traits you will meet often.

02

Generics and Trait Bounds

main.rs

```rust
fn largest<T: PartialOrd + Copy>(items: &[T]) -> T {
    let mut max = items[0];
    for &x in items {
        if x > max { max = x; }
    }
    max
}
largest(&[3, 9, 2]);          // 9
largest(&[1.5, 0.2]);         // 1.5
```

- Generic functions work for many types. A trait bound, T colon PartialOrd, says what the …
- Rust generates specialized code for each type you use, so generics cost nothing at run time.

03

impl Trait and where Clauses

main.rs

```rust
fn print_all(items: &[impl Describe]) {
    for i in items { println!("{}", i.describe()); }
}

fn evens(limit: u32) -> impl Iterator<Item = u32> {
    (0..limit).filter(|n| n % 2 == 0)
}

fn report<T>(x: T) where T: Describe + fmt::Display { /* ... */ }
```

- impl Trait in a parameter accepts any type with that trait. In a return position, it hides …
- where clauses move long bounds after the signature to keep it readable.

04

Trait Objects: dyn Trait

generics: impl Trait

- one type per call
- decided at compile time
- fastest

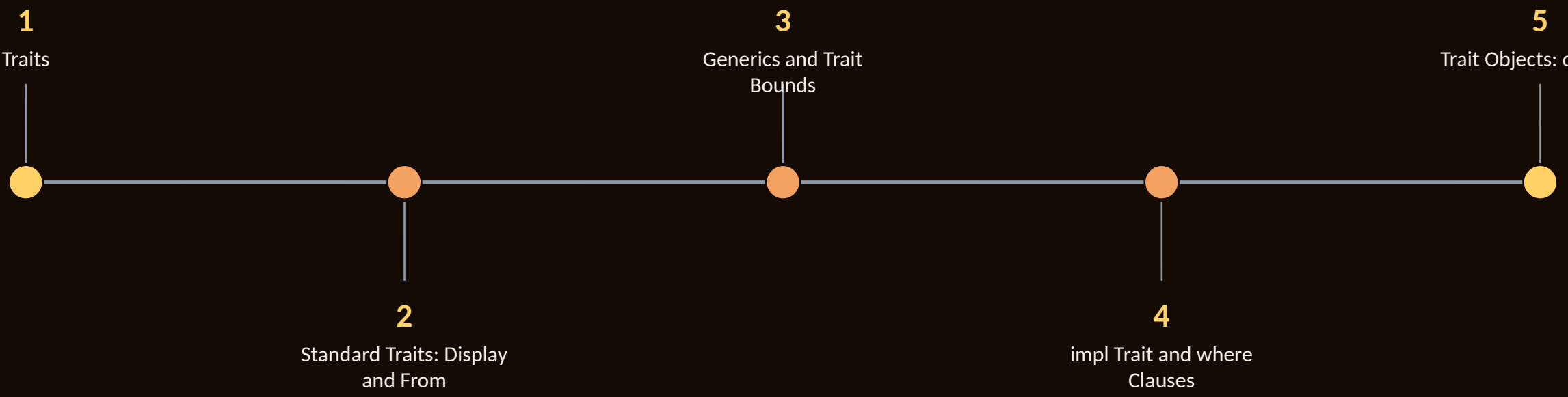
dyn Trait

- Vec<Box<dyn Shape>>
- mixed types together
- decided at run time

- Generics pick the type at compile time. When you need a list of different types together, …
- Method calls go through a table at run time. It is slightly slower but very flexible.

05

Lesson Map


1
Traits
2
Standard Traits: Display and From
3
Generics and Trait Bounds
4
impl Trait and where Clauses
5
Trait Objects: dyn Trait

Sources and Further Reading

main.rs

```rust
trait Describe {
    fn name(&self) -> String;
    fn describe(&self) -> String {          // default method
        format!("This is {}", self.name())
    }
}
impl Describe for Student {
    fn name(&self) -> String { self.name.clone() }
}
```

RUST PROGRAMMING · 10

main.rs

```rust
fn largest<T: PartialOrd + Copy>(items: &[T]) -> T {
    let mut max = items[0];
    for &x in items {
        if x > max { max = x; }
    }
    max
}
largest(&[3, 9, 2]);          // 9
largest(&[1.5, 0.2]);         // 1.5
```

Thank you for watching

MyWebUniversity.com

Free technical education since 2000 — English and Dari

@MyWebUniversity · @MyWebUniversity-Dari