درس ۴

Rust

مالکیت

```rust
main.rs
let s2 = s1;   // move
let r = &s2;   // borrow
let m = &mut x;
```

یک کورس رایگان از MyWebUniversity

برنامه‌نویسی Rust

درس ۴

مالکیت و قرض گرفتن

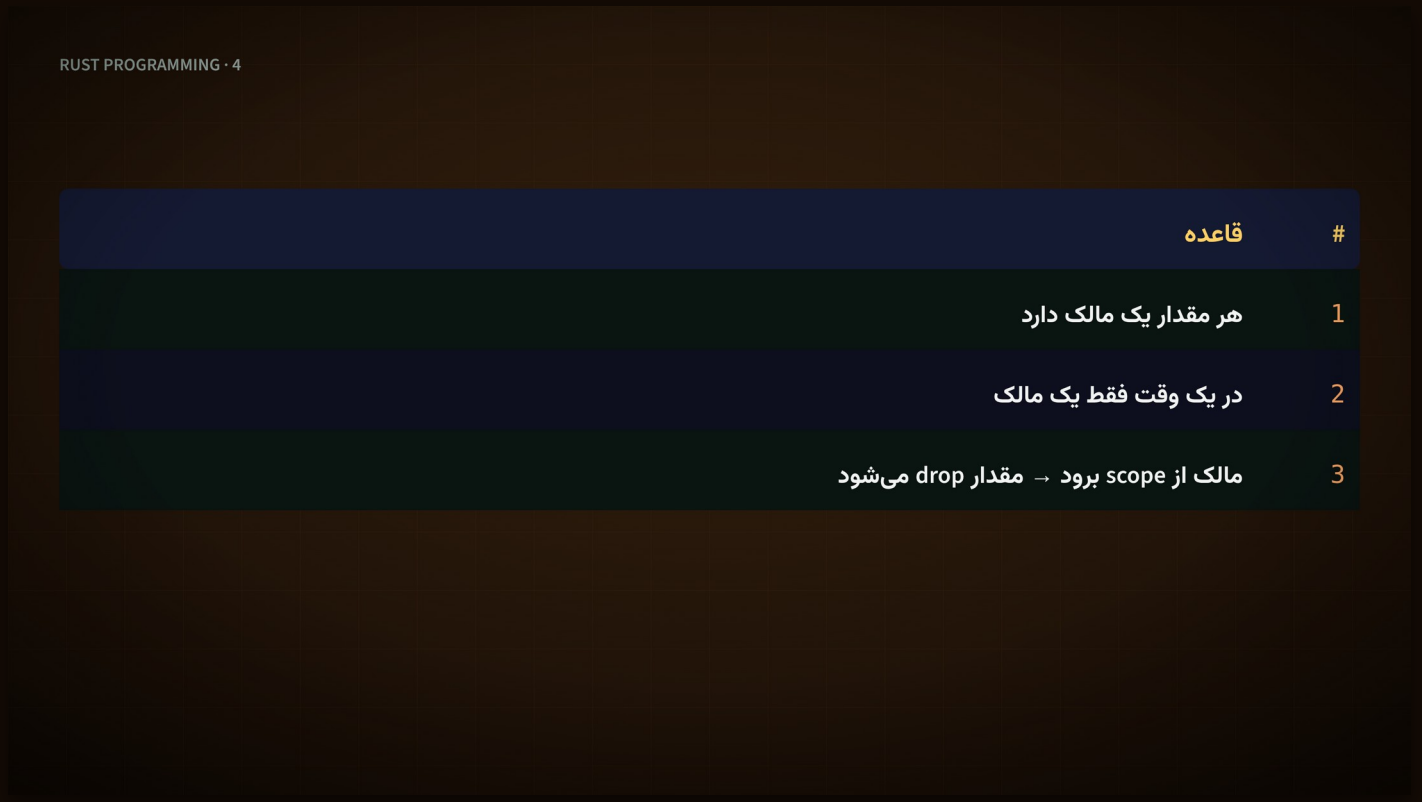
سه قاعدهٔ مالکیت، move، clone، reference ها و borrow checker



قاعده‌های مالکیت

هر مقدار دقیقاً یک مالک دارد. وقتی مالک از scope بیرون می‌رود، مقدار drop می‌شود و حافظهٔ ...

این سه قاعده اجازه می‌دهند Rust حافظه را هنگام کمپایل مدیریت کند، بدون garbage collector و ...

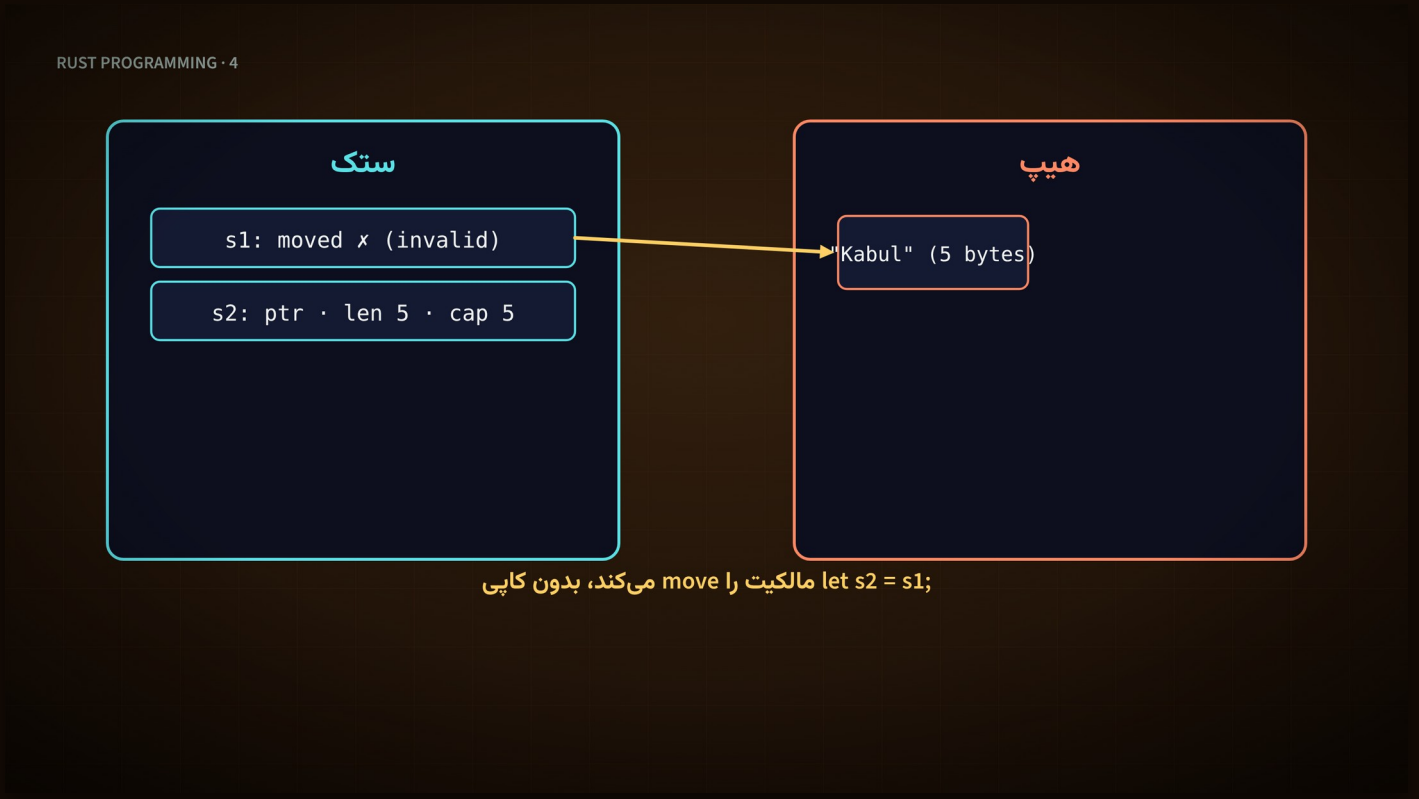


01

move

دادن یک String به متغیر دیگر آن را move می‌کند. متغیر اول دیگر معتبر نیست، پس حافظه هرگز دو ...

نوع‌های ساده مثل integer ها و bool ها Copy هستند، پس به جای move کاپی می‌شوند.



02

clone

وقتی واقعاً دو کاپی مستقل لازم دارید، clone را صدا کنید. دادهٔ heap را هم کاپی می‌کند، پس ...

دیدن clone در کود یک نشانه است: اینجا یک کاپی عمیق انجام می‌شود.

main.rs

```rust
let s1 = String::from("Kabul");
let s2 = s1.clone();           // deep copy
println!("{s1} {s2}");        // both valid

let a = 5;
let b = a;                     // i32 is Copy
println!("{a} {b}");
```

03

قرض گرفتن با reference ها

به جای move کردن یک مقدار به یک تابع، آن را با & قرض بدهید. تابع آن را قرض می‌گیرد، و ...

یک reference از نوع &mut اجازه می‌دهد قرض‌گیرنده مقدار را تغییر دهد.

main.rs

```rust
fn length(s: &String) -> usize {
    s.len()                      // borrows, does not own
}
fn shout(s: &mut String) {
    s.push_str("!");
}
let mut city = String::from("Herat");
let n = length(&city);
shout(&mut city);                // "Herat!"
```

04

borrow checker

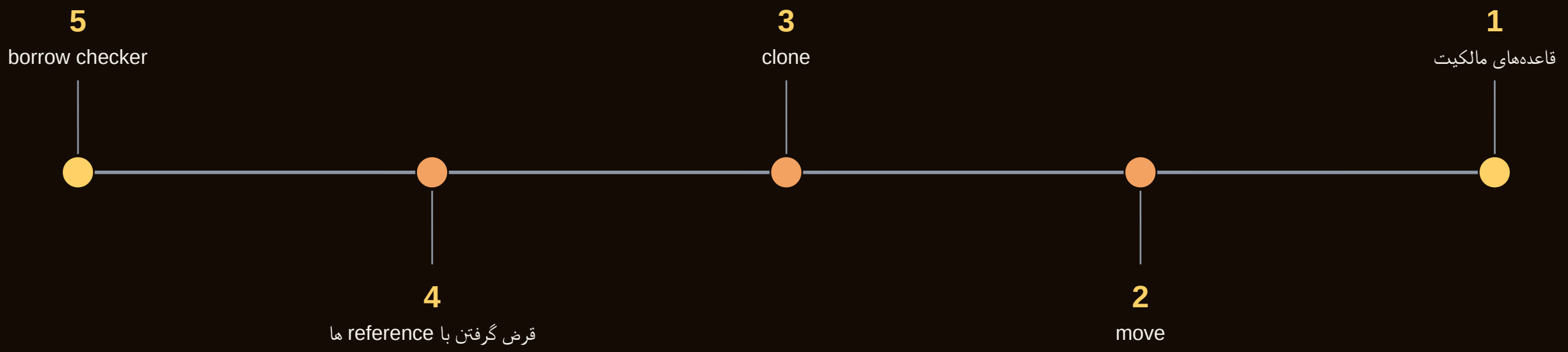
قاعده: در هر لحظه می‌توانید یا reference های شریک زیاد، یا دقیقاً یک reference قابل تغییر ...

همین یک قاعده، که هنگام کمپایل بررسی می‌شود، باگ‌های data race و use-after-free را کاملاً ...

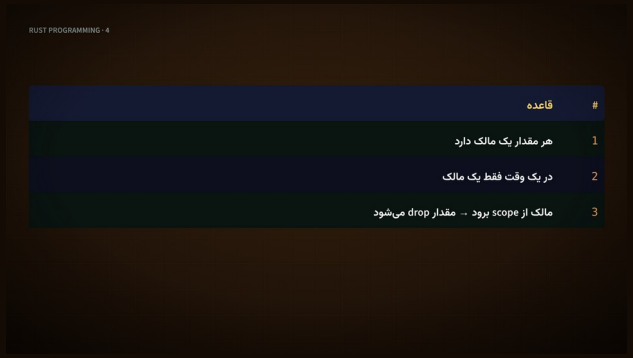


05

نقشهٔ درس


5
borrow checker
3
clone
1
قاعده‌های مالکیت
4
قرض گرفتن با reference ها
2
move

منابع و مطالعهٔ بیشتر





تشکر از تماشای شما

MyWebUniversity.com

آموزش رایگان تکنالوژی از سال ۲۰۰۰ — انگلیسی و دری

@MyWebUniversity-Dari · @MyWebUniversity