모각프 5주차 결과

2021.08.27

18 박준서

ARGOS

INDEX

ARGOS

5주차 결과

진행 상황

- 프로젝트
- 공부

ARGOS

프로젝트

- 안드로이드 프로젝트의 jniLibs 디렉토리에 abi와 그에 맞는 lib.so 추가
- 잘 동작하는지 테스트

ARGOS

rust의 소유권 개념에 대해 공부

소유권 : rust의 메모리 관리 방법으로, 런타임에 동작하며 자동으로 참조되지 않는 메모리를 해제하는 garbage collector나 수동으로 직접 메모리를 관리하는 방법과는 다르게, 컴파일러가 컴파일 시점에 소유권을 검사하여 메모리를 관리하고, 따라서 소유권과 관련된 기능은 프로그램의 실행 성능에는 영향을 끼치지 않는다.

소유권 규칙

- rust의 모든 값은 owner라는 변수를 가지고 있다.
- 특정 시점에서 값의 owner는 단 하나이다.
- Owner가 scope에서 벗어나면, 그 값은 제거된다.

공부

ARGOS

move

```rust
let s1 = String::from("string");
let s2 = s1;
println!("{} {}", s2, s1);
```

```
error[E0382]: borrow of moved value: `s1`
 --> src\main.rs:4:27
  |
2 |     let s1 = String::from("string");
  |         -- move occurs because `s1` has type `String`, which does not implement the `Copy` trait
3 |     let s2 = s1;
  |              -- value moved here
4 |     println!("{} {}", s2, s1);
  |                           ^^ value borrowed here after move

error: aborting due to previous error

For more information about this error, try `rustc --explain E0382`.
```

ARGOS

참조

```rust
let s1 = String::from("string");
let s2 = &s1;
s2.push_str("2");
println!("{} {}", s2, s1);
```

```
error[E0596]: cannot borrow `*s2` as mutable, as it is behind a `&` reference
 --> src\main.rs:4:5
  |
3 |     let s2 = &s1;
  |              --- help: consider changing this to be a mutable reference: `&mut s1`
4 |     s2.push_str("2");
  |     ^^ `s2` is a `&` reference, so the data it refers to cannot be borrowed as mutable

error: aborting due to previous error

For more information about this error, try `rustc --explain E0596`.
```

ARGOS

참조가 필요한 경우

```rust
fn main() {
    let s1 = String::from("string");
    let len = my_func(s1);
    println!("{} {}", s1, len);
}

fn my_func(s: String) -> usize {
    s.len()
}
```

```
error[E0382]: borrow of moved value: `s1`
 --> src\main.rs:4:23
  |
2 |     let s1 = String::from("string");
  |         -- move occurs because `s1` has type `String`, which does not implement the `Copy` trait
3 |     let len = my_func(s1);
  |                       -- value moved here
4 |     println!("{} {}", s1, len);
  |                       ^^ value borrowed here after move

error: aborting due to previous error

For more information about this error, try `rustc --explain E0382`.
```

Q & A

Thank You for Listening