

DropとFutureと

未 来
Dropのfuture

keen TechFeed公認エキスパート (Rust)

#tfcon

Drop

FTechFeed
CONFERENCE
2022

値のライフタイムが終わったときに走る

panic以外の色々な処理が走る

BufWriterのflushとか

```
pub trait Drop {  
    fn drop(&mut self);  
}
```

#tfcon

Dropが呼ばれるタイミング

FTechFeed
CONFERENCE
2022

```
let mut value = Some(Data::new(1));  
value = Some(Data::new(2));  
value.insert(Data::new(3));  
if value.is_none() {  
    panic!("unexpected");  
}  
let _ = Data::new(4);
```

Dropが呼ばれるタイミング

FTechFeed
CONFERENCE
2022

再代入

```
let mut value = Some(Data::new(1));  
value = Some(Data::new(2));  
value.insert(Data::new(3));  
if value.is_none() {  
    panic!("unexpected");  
}  
let Data::new(4);
```

内部で再代入する関数

panic

let _ =

スコープの終わり

async

```
async fn some_function() {  
}
```



脱糖

```
fn some_function() -> impl Future<Output = ()> {  
}
```

awaitはasyncの中でしか使えない

asyncの中で同期IOはよくない

FutureとDrop

FTechFeed
CONFERENCE
2022

Future内でもdropが走る

```
async fn wait_then_drop_stream(_stream: TlsStream) {  
    time::sleep(Duration::from_secs(10)).await;  
}
```

#tfcon

FutureとDrop

Future内でもdropが走る

```
async fn wait_then_drop_stream(_stream: TlsStream) {  
    time::sleep(Duration::from_secs(10)).await;  
    _stream.drop()  
}
```

このdrop、asyncにしたい？

-> async drop (議論中の機能)

async dropの難しさ

FTechFeed
CONFERENCE
2022

1. dropとasync dropどっち呼ぶの？
2. syncの中でasync drop呼んでいいの？
3. async dropのキャンセルとか

トレイトオブジェクトやasyncの状態管理など他にも色々

drop or async drop?

FTechFeed
CONFERENCE
2022

```
struct Hoge { /* ... */ }  
impl Drop for Hoge { /* ... */ }  
impl AsyncDrop for Hoge { /* ... */ }
```

```
fn func() {  
    let hoge = Hoge { /* ... */ };  
}  
async fn func() {  
    let hoge = Hoge { /* ... */ };  
}
```

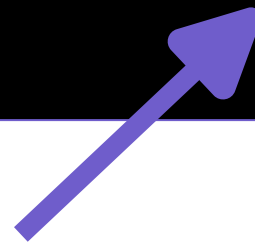
async drop in sync

FTechFeed
CONFERENCE
2022

```
fn just_drop(_stream: TlsStream) {  
}
```



```
fn just_drop(_stream: TlsStream) {  
    _stream.async_drop().await  
}
```



pollできないけどいいの？

禁止するとOption::insertとかも使えなくなる

Futureの実行

FTECHFEED
CONFERENCE
2022

```
let conn = pool.get_conn().await?;  
let mut user = get_user(conn).await?;  
user.name = new_name;  
save_user(conn, &user).await?;
```



#tfcon

Futureの実行

FTECHFEED
CONFERENCE
2022

```
let conn = pool.get_conn().await?;  
let mut user = get_user(conn).await?;  
user.name = new_name;  
save_user(conn, &user).await?;
```

Futureの実行

FTechFeed
CONFERENCE
2022

```
pool.get_conn()
```

```
let conn = ... ?  
    get_user(conn)
```

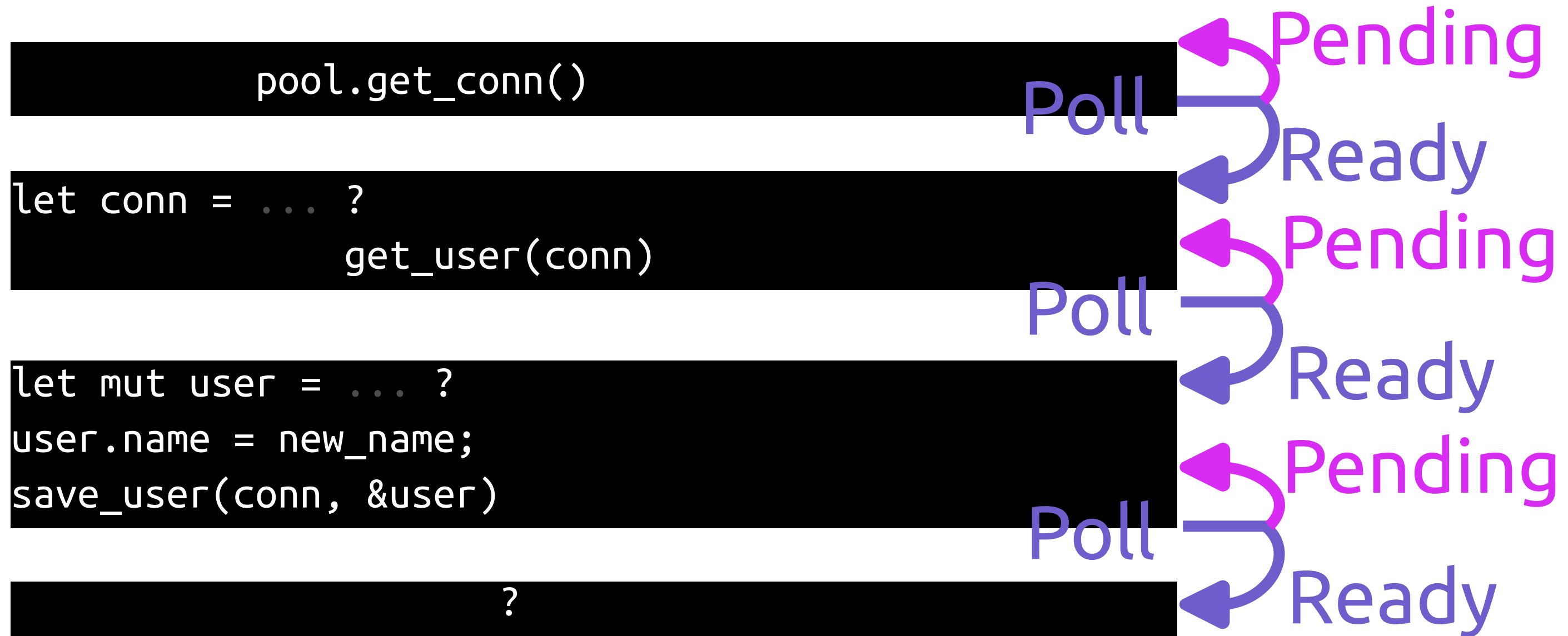
```
let mut user = ... ?  
user.name = new_name;  
save_user(conn, &user)
```

```
?
```

#tfcon

Futureの実行

FTECHFEED
CONFERENCE
2022



```
pub enum Poll<T> {  
    Ready(T),  
    Pending,  
}
```

#tfcon

Futureのdrop

FTechFeed
CONFERENCE
2022

Futureはデータなのでdropされうる

pollを呼ばずにdropできる

Futureのdrop = Futureのキャンセルと決められてる

つまりawait = キャンセル可能なポイント

Drop と Future と Drop

FTechFeed
CONFERENCE
2022


```
async fn func(mut stream: TlsStream) {  
    let new_stream = ...;  
    stream = new_stream;  
    // ...  
}
```

#tfcon

Drop と Future と Drop

FTechFeed
CONFERENCE
2022

```
async fn func(mut stream: TlsStream) {  
    let new_stream = ...;  
    stream = new_stream;  
    // ...  
}  
stream.async_drop().await
```



DropとFutureとDrop

FTECHFEED
CONFERENCE
2022

```
async fn func(mut stream: TlsStream) {  
    let new_stream = ...;  
    stream = new_stream;  
    // ...  
}  
stream.async_drop().await
```



何もないところからawaitがでてきた

= キャンセル可能になった

async dropするところにはawait書かせる？

そもそも全てのFutureがキャンセル可能でよい？

DropとFutureとDropの未来

FTechFeed
CONFERENCE
2022

async dropは望まれてる機能だよ

でも合理的な設計が難しいよ

将来どうなるんでしょうね。



#tfcon

参考資料

FTECHFEED
CONFERENCE
2022

Async destructors, async genericity and completion futures - Sabrina Jewson
<https://sabinajewson.org/blog/async-drop>

asyncdrop/async-dest.md at main · vzvezda/asyncdrop
<https://github.com/vzvezda/asyncdrop/blob/main/article/async-dest.md>

Destructors - The Rust Reference
<https://doc.rust-lang.org/reference/destructors.html>

#tfcon