

Parallel Execution | Channels

Channels

- ◆ One-way communication between threads
 - Message passing
 - ***Sender*** and ***Receiver***
- ◆ Can have limited or unlimited capacity
- ◆ ***crossbeam-channel*** crate
 - Use docs.rs website to view documentation for crates

[*dependencies*]

crossbeam-channel = "*"

■ Message Passing

- ◆ *enum* commonly used for messages
 - ***match*** allows easy message handling
- ◆ Guaranteed in-order delivery
- ◆ Can be blocking or non-blocking
 - Block on *Sender*: Channel full
 - Block on *Receiver*: No messages
 - Behavior determined by function, not by channel

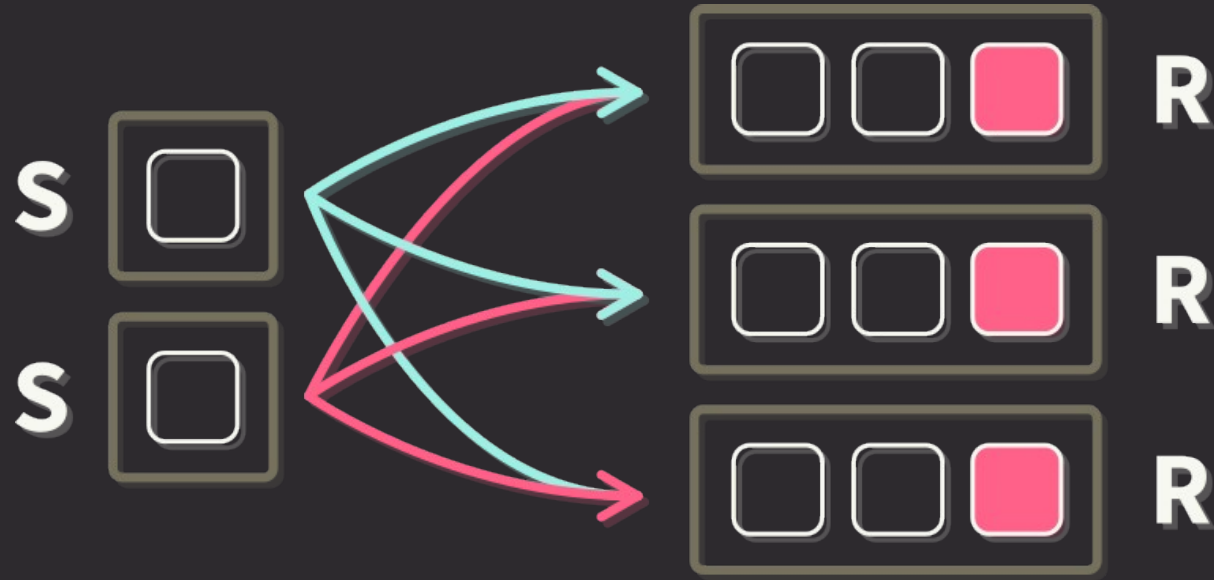
Channel Operation

Sender

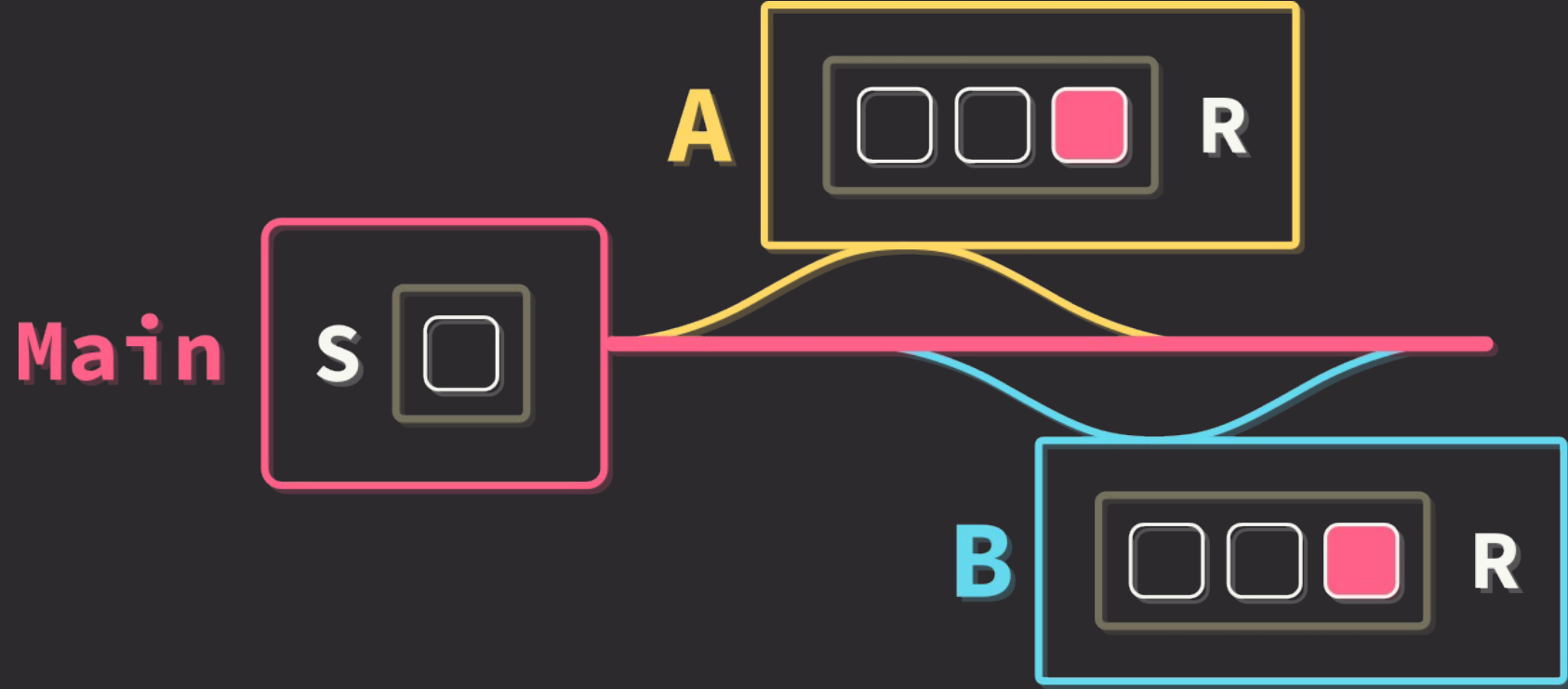
Receiver



Channel Operation



Channels & Threads



Example

```
use crossbeam_channel::unbounded;
```

```
let (sender, receiver) = unbounded();
```

```
sender.send("Hello, channel!");
```

```
match receiver.recv() {  
    Ok(msg) => println!("{}", msg),  
    Err(e)  => println!("{:?}", e),  
}
```

Sender<type>
Receiver<type>

Threaded Example

```
use crossbeam_channel::unbounded;
use std::thread;

let (s, r) = unbounded();

let handle = thread::spawn(move || match r.recv() {
    Ok(msg) => println!("Thread: {}", msg),
    Err(e) => println!("{:?}", e),
});

s.send("Hello from main!");
handle.join();
```

Thread: Hello from main!

Multi-threaded Example

```
let (s, r1) = unbounded();
let r2 = r1.clone();

let handle1 = thread::spawn(move || match r1.recv() {
    Ok(msg) => println!("Thread1: {}", msg),
    Err(e) => println!("{:?}", e),
});

let handle2 = thread::spawn(move || match r2.recv() {
    Ok(msg) => println!("Thread2: {}", msg),
    Err(e) => println!("{:?}", e),
});

s.send("Hello from main!")?;
s.send("Hello from main!")?;
handle1.join();
handle2.join();
```

■ Result

Thread1: Hello from main!

Thread2: Hello from main!

■ Recap

- ◆ Channels offer unidirectional communication
- ◆ Composed of ***Send*** and ***Receive*** ends
 - Ends can be ***cloned*** and sent to threads
- ◆ Channel operations can be blocking or non-blocking
- ◆ Any data can be sent across a channel
 - ***enum*** is useful because of variants