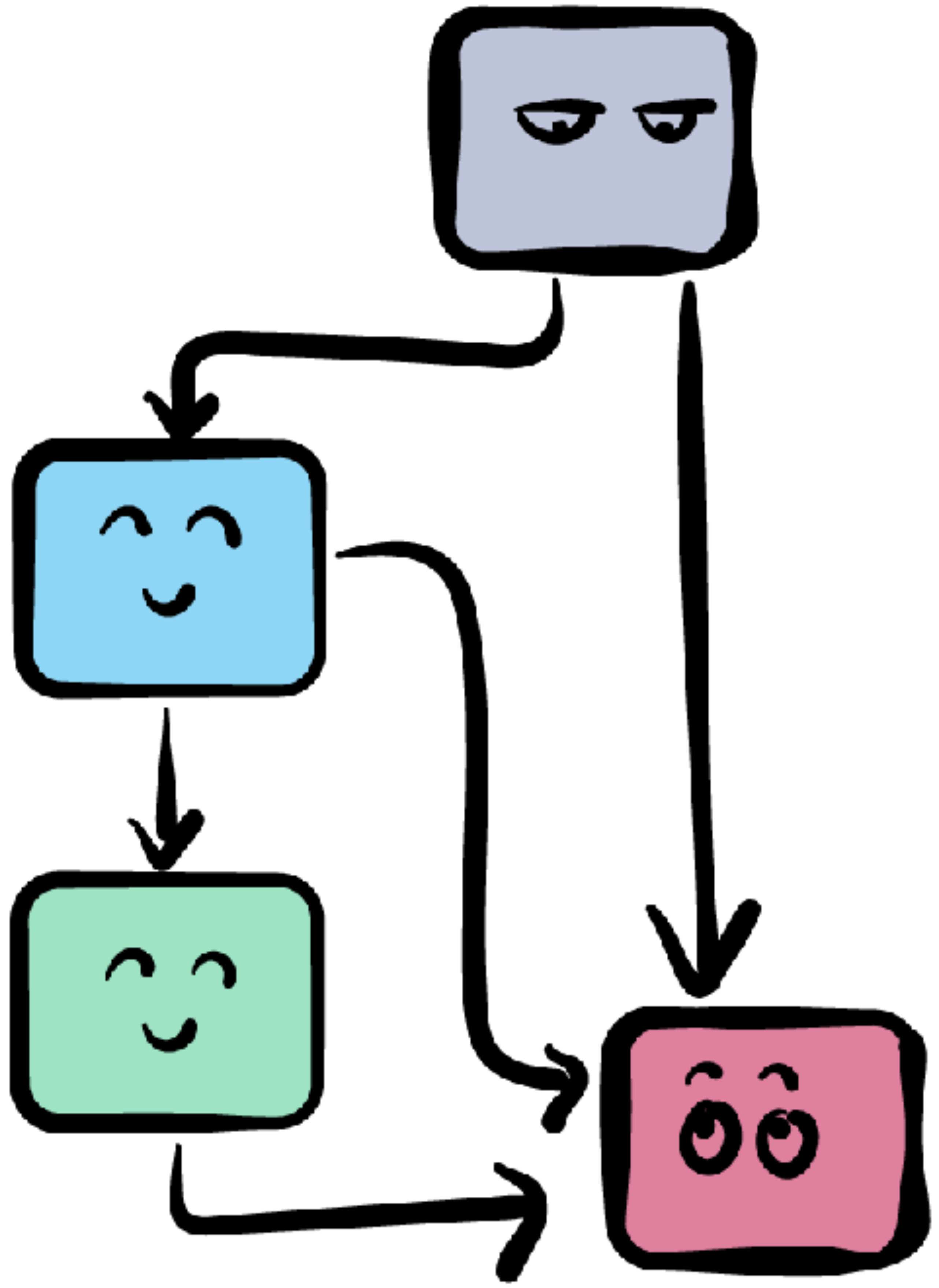
iOS Concurrency with GCD & Operations

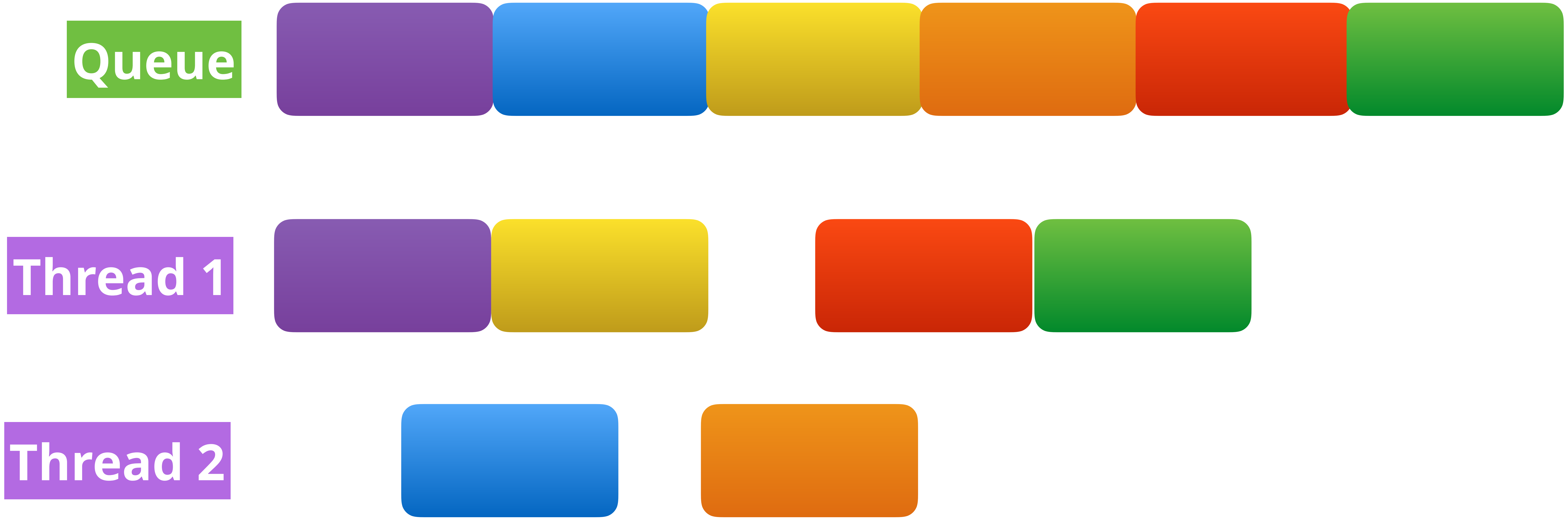


Part 1: Introduction

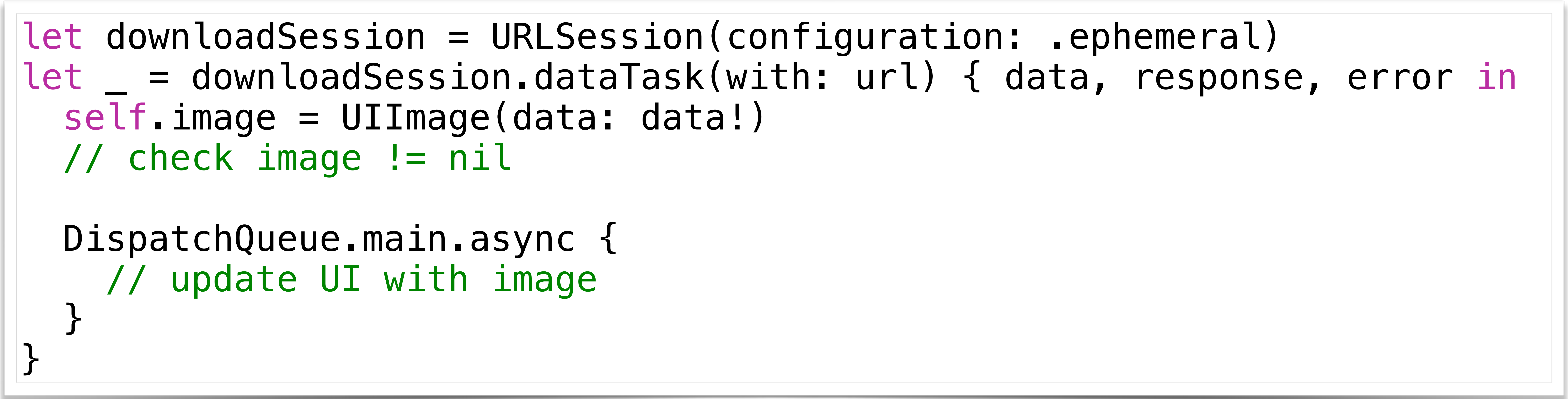
FAQ

- What is concurrency?
- Why use concurrency?
- How do you use concurrency?
- GCD or Operations?
- Where do tasks run?

Queues & Threads

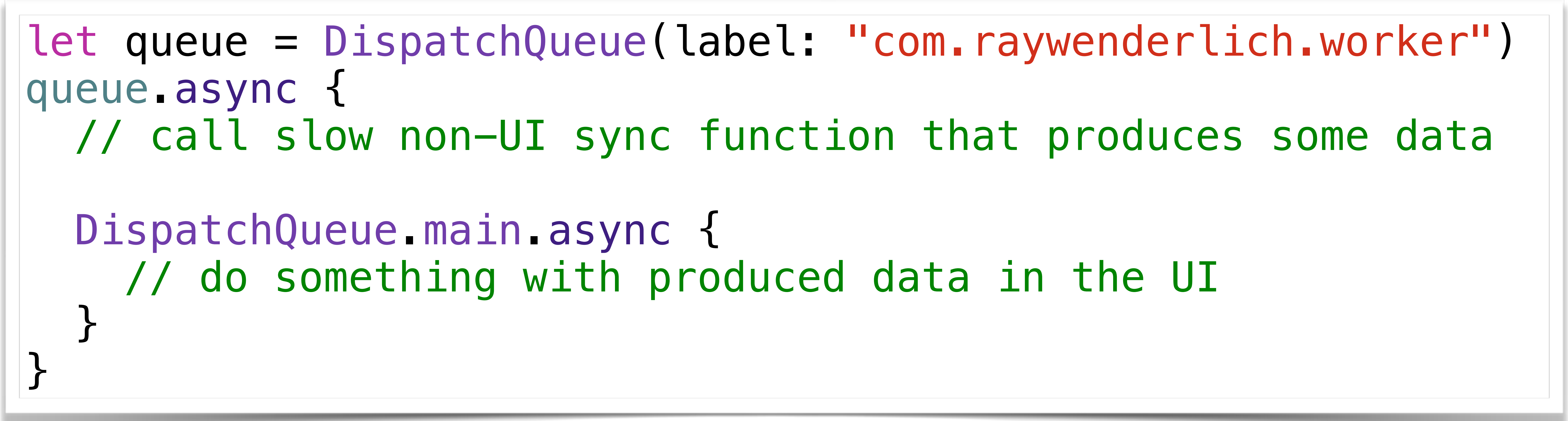


Async All the Things!



```
let downloadSession = URLSession(configuration: .ephemeral)
let _ = downloadSession.dataTask(with: url) { data, response, error in
  self.image = UIImage(data: data!)
  // check image != nil

  DispatchQueue.main.async {
    // update UI with image
  }
}
```



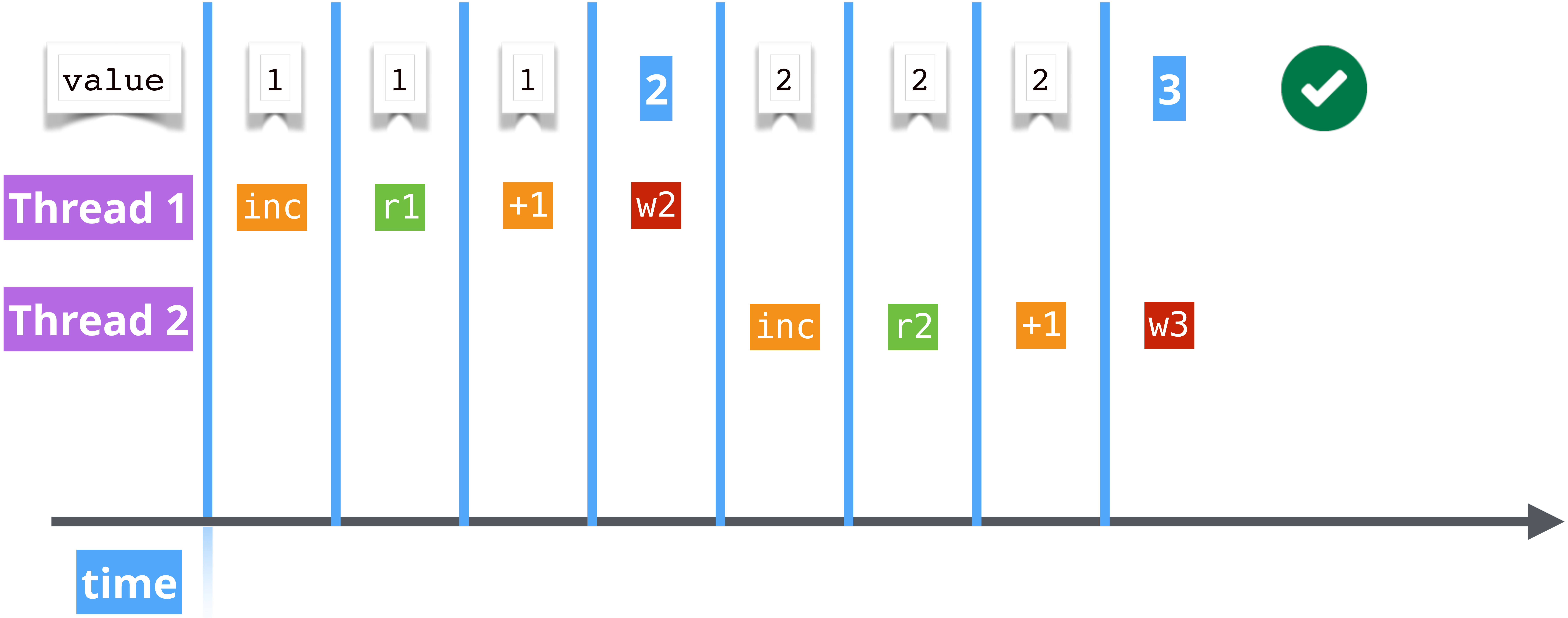
```
let queue = DispatchQueue(label: "com.raywenderlich.worker")
queue.async {
  // call slow non-UI sync function that produces some data

  DispatchQueue.main.async {
    // do something with produced data in the UI
  }
}
```

Concurrency Problems

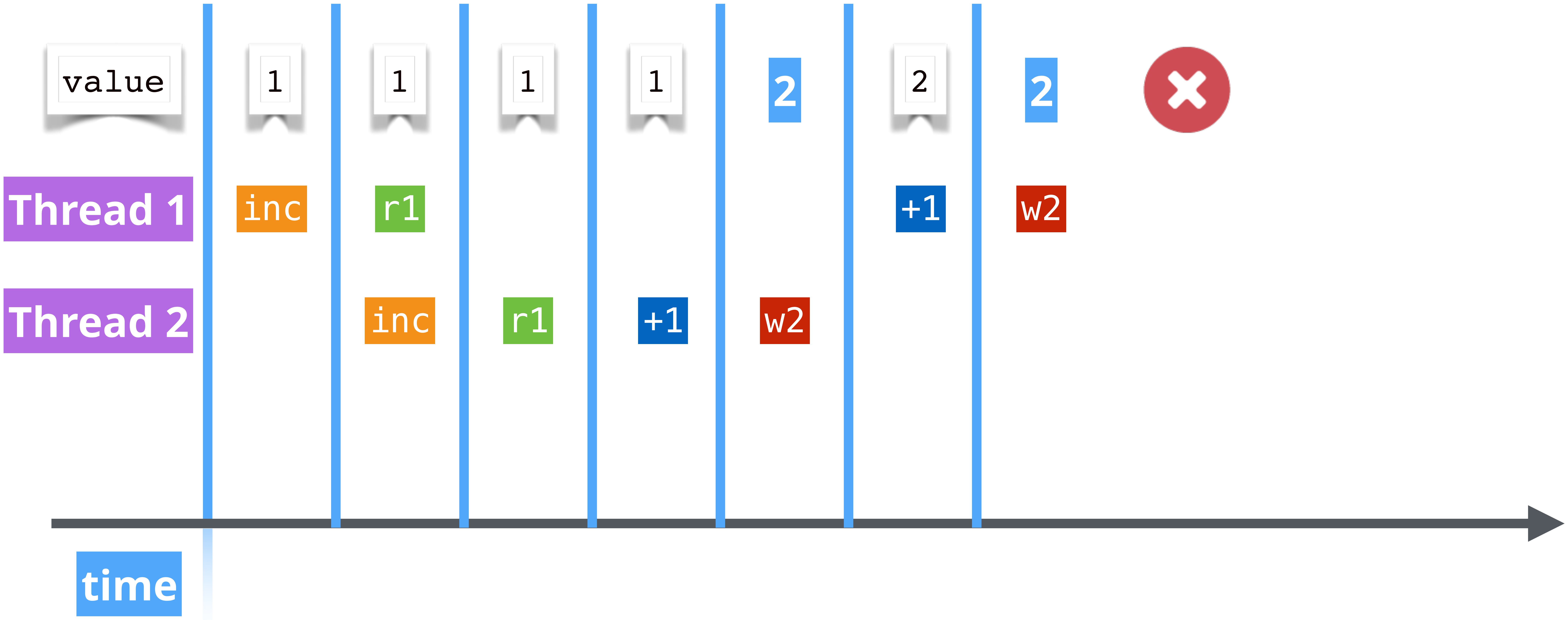
- Race condition
- Priority inversion
- Deadlock

Race Condition



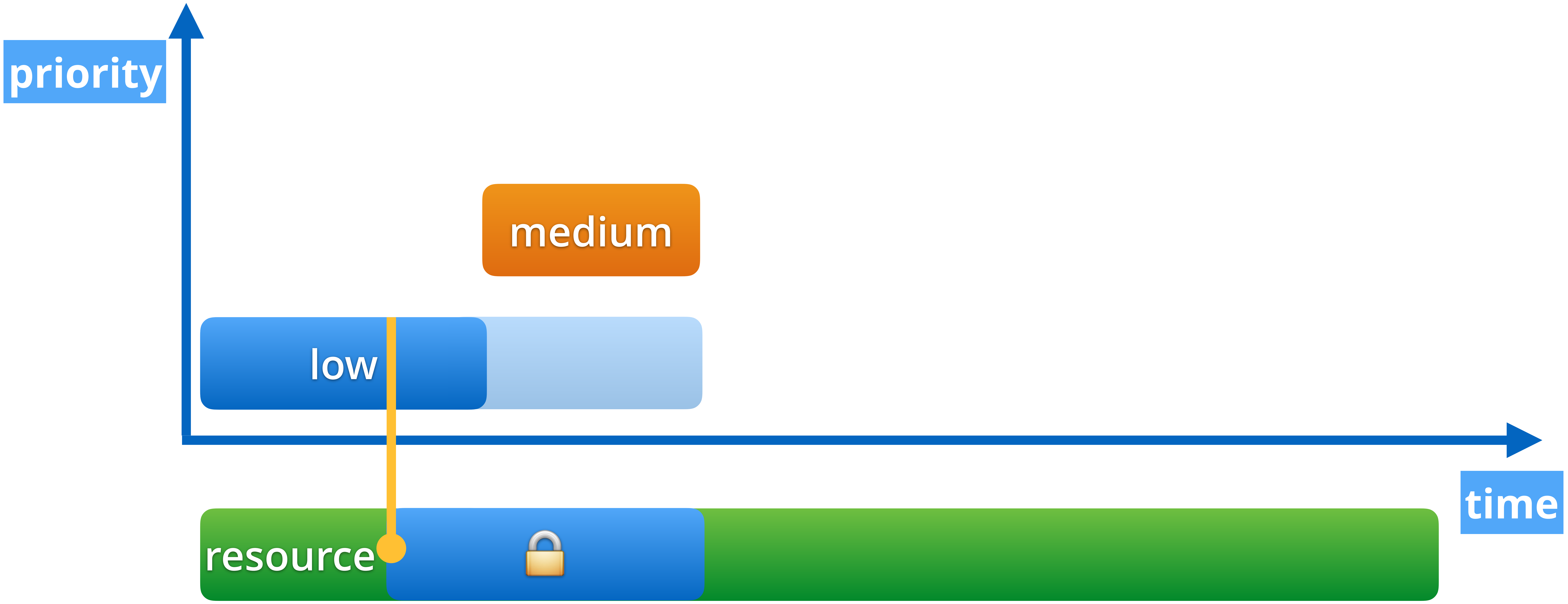
value	1	1	1	2	2	2	2	3	✓
Thread 1	inc	r1	+1	w2					
Thread 2					inc	r2	+1	w3	

Race Condition

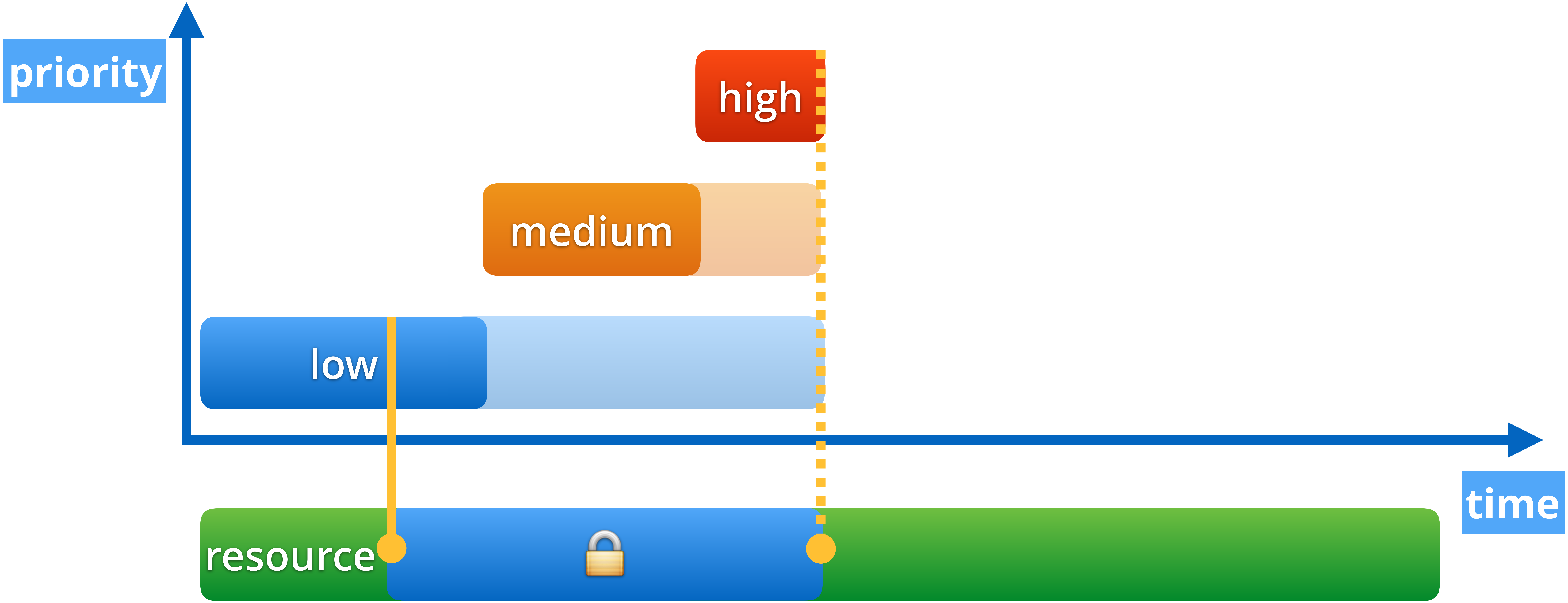


value	1	1	1	1	2	2	2	✖
Thread 1	inc	r1				+1	w2	
Thread 2		inc	r1	+1	w2			

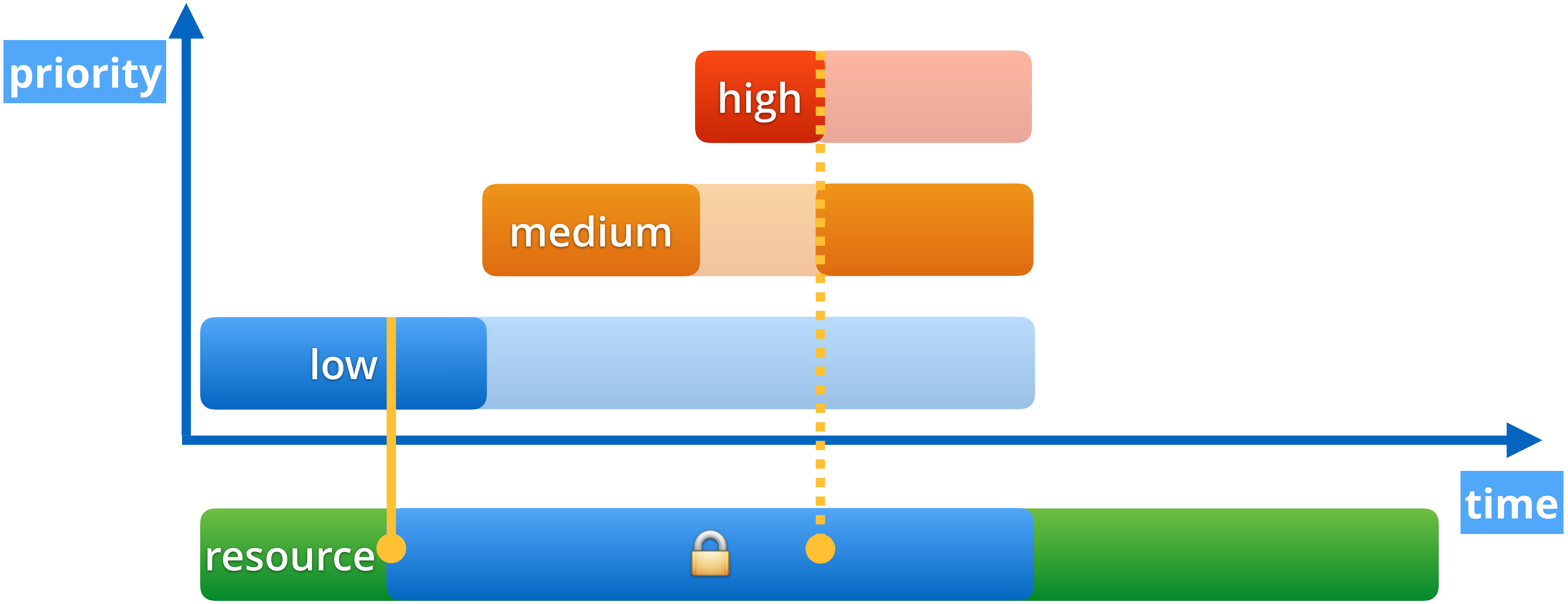
Priority Inversion



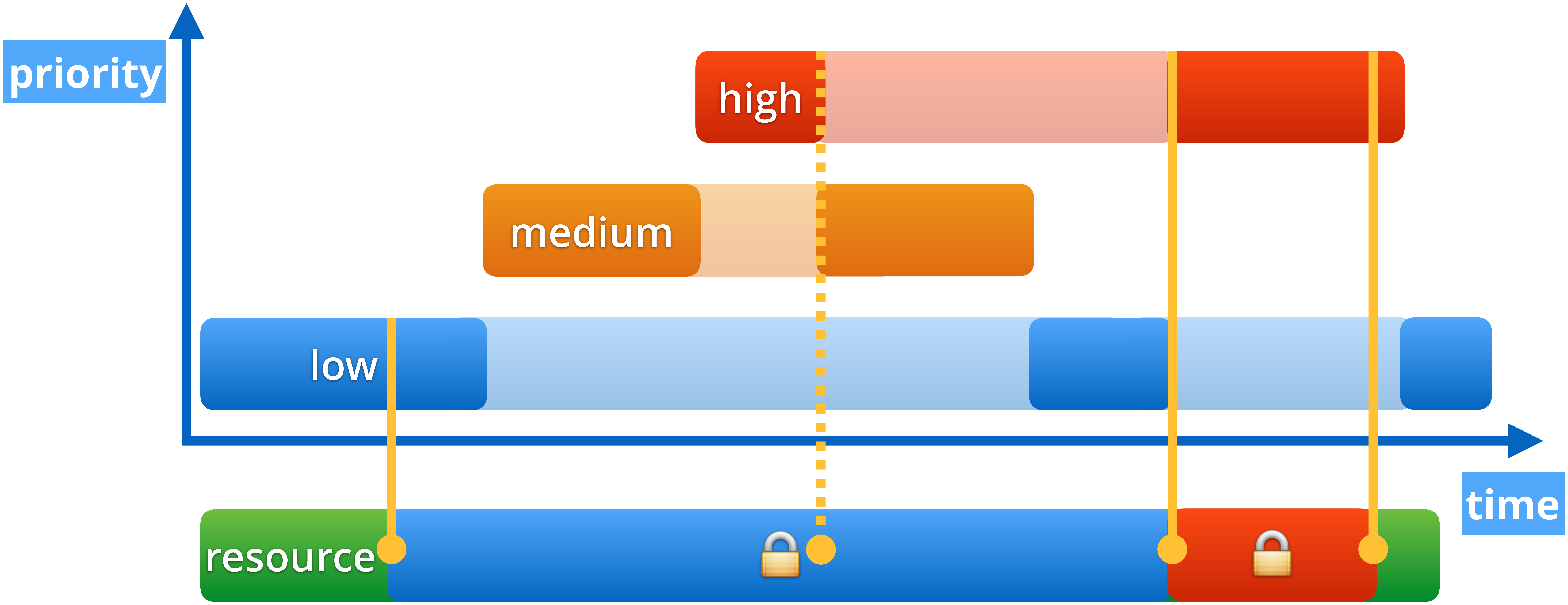
Priority Inversion



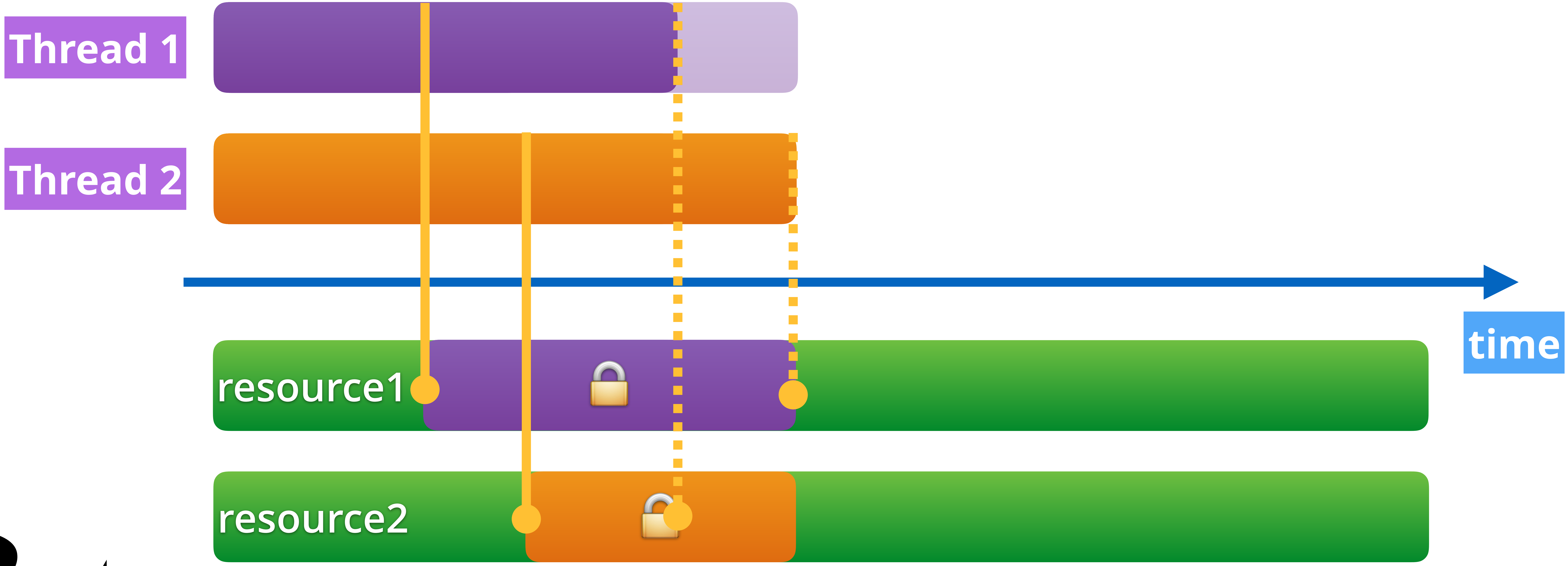
Priority Inversion



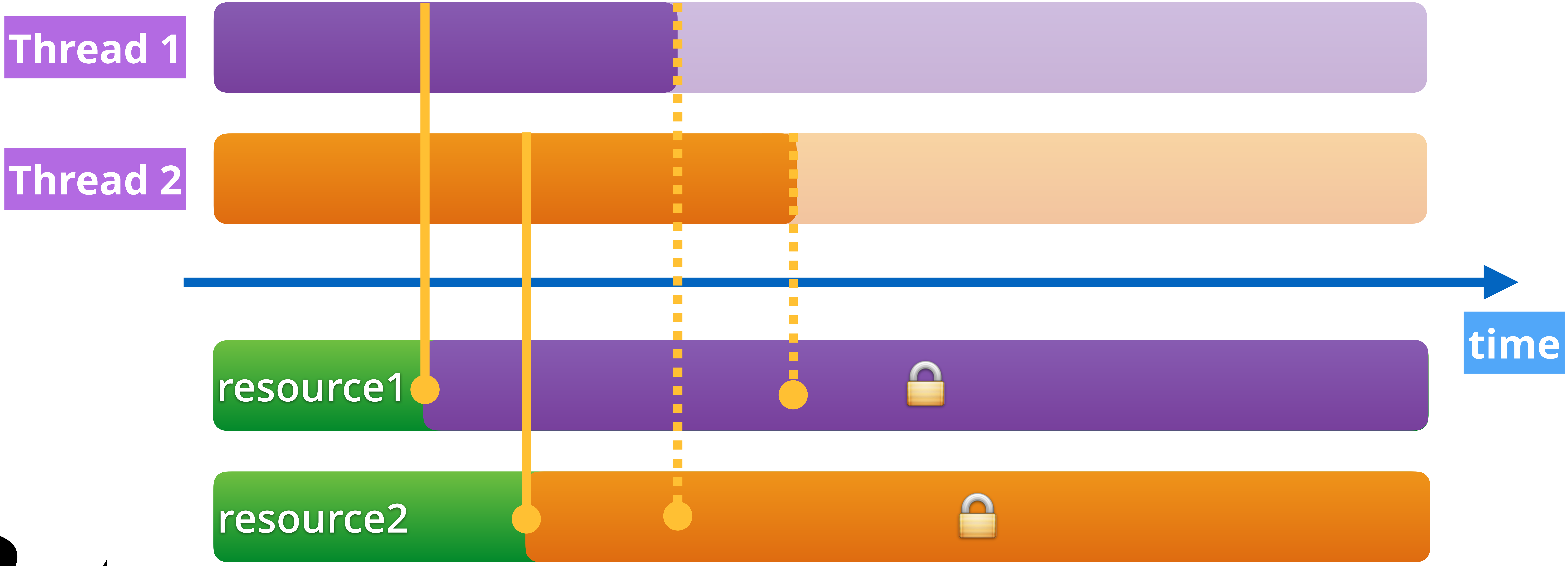
Priority Inversion



Deadlock



Deadlock



Where We're Going

- Terminology
- Use cases:
  - Similar independent tasks; chain of tasks
  - Group of tasks with notification when all have finished
  - Tasks with dependencies; cancelling tasks
- Concurrency solutions

Credits

Sam Davies
@iwantmyrealname

Divyendu Singh
@divyenduz