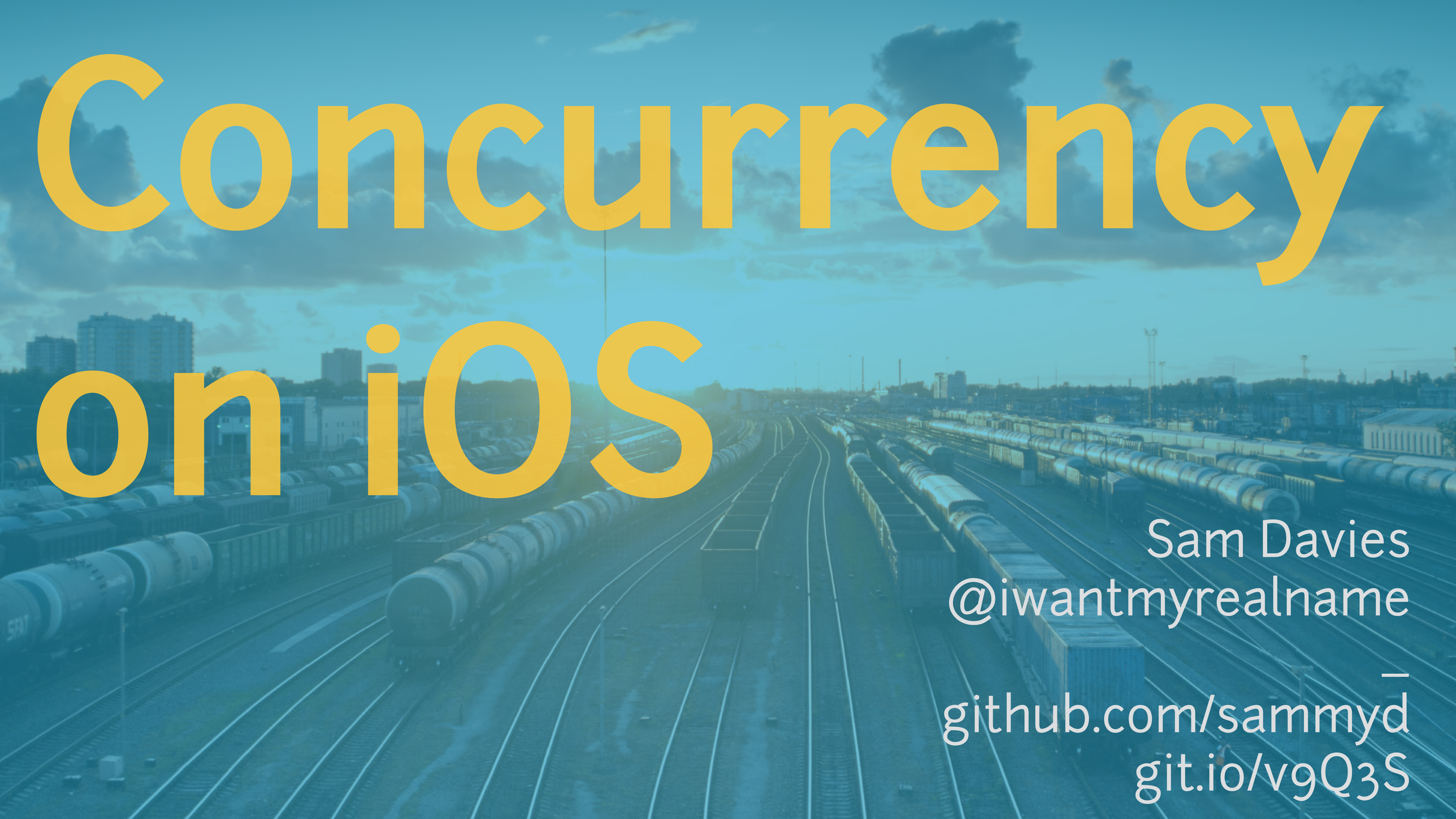
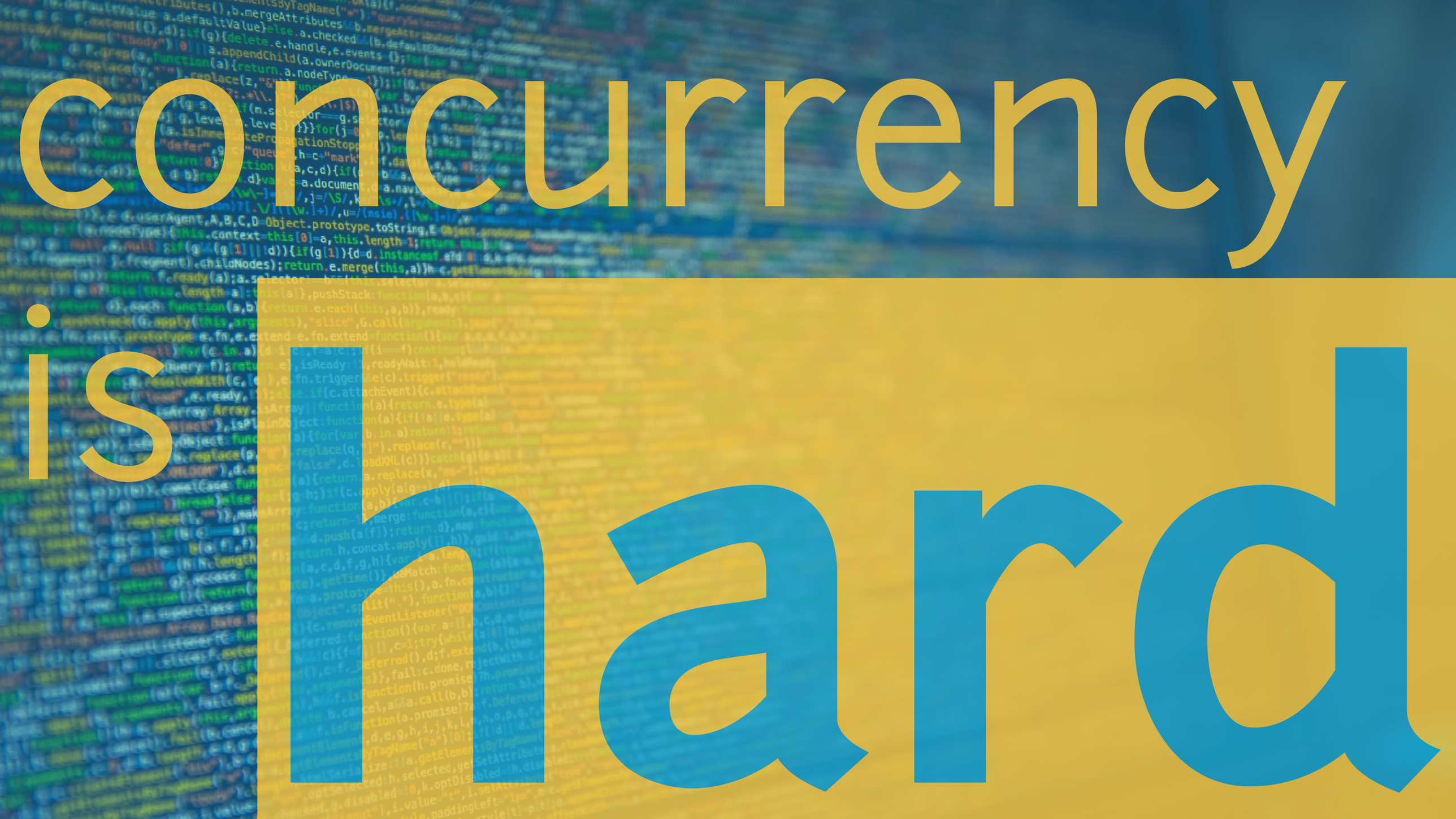


Concurrency on iOS

Sam Davies
@iwantmyrealname

github.com/sammyd_
git.io/v9Q3S


concurrency
is
hard


what is
concurrency?


why use
concurrency?

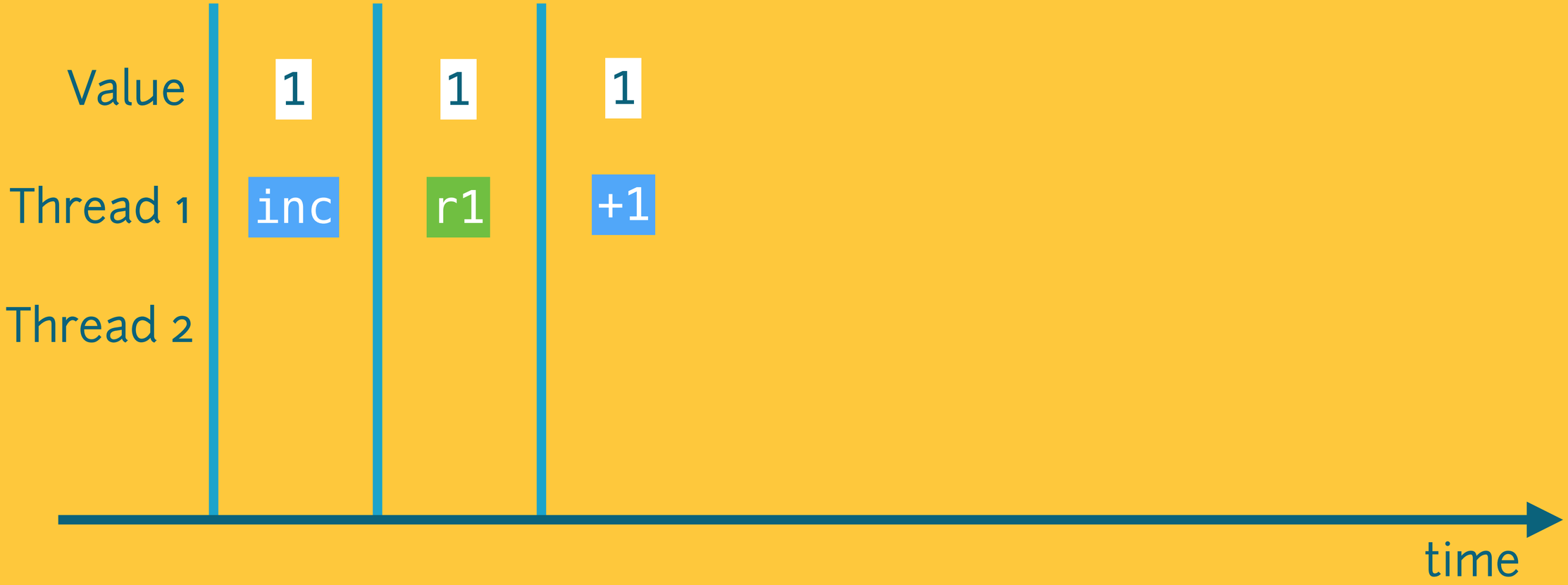

common
concurrency
problems


race condition

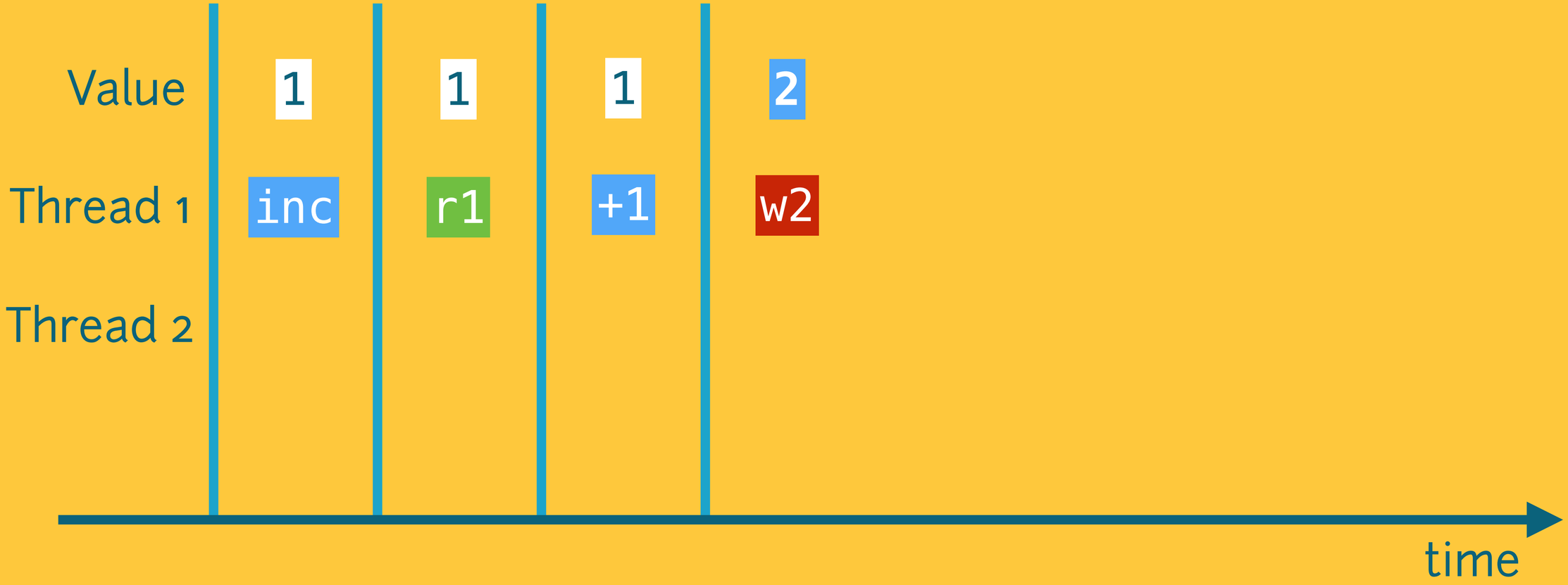
Race Condition

Race Condition

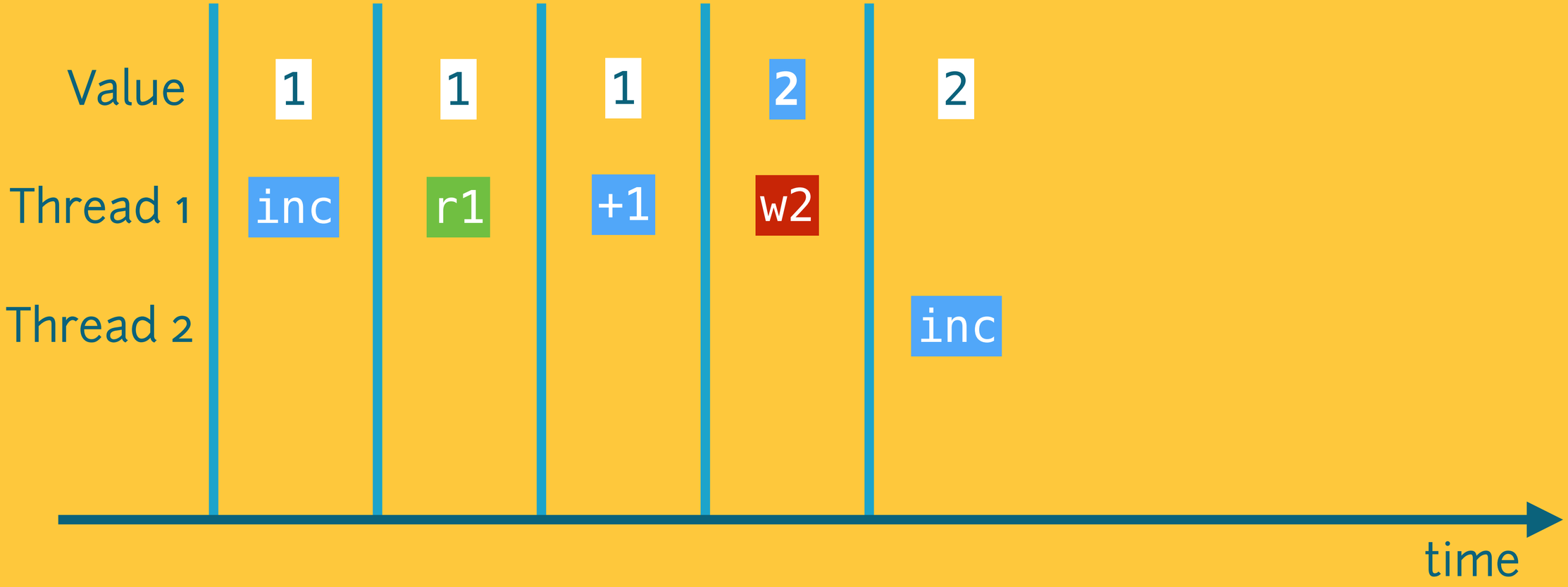
Race Condition



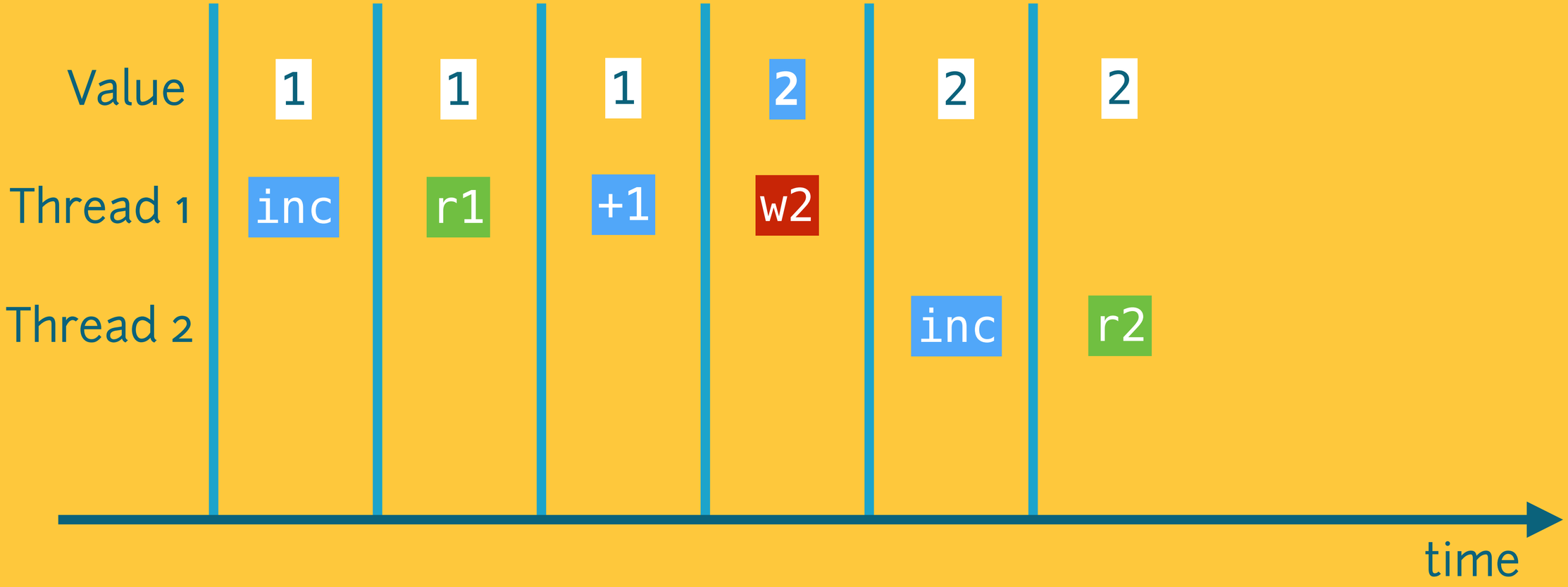
Race Condition



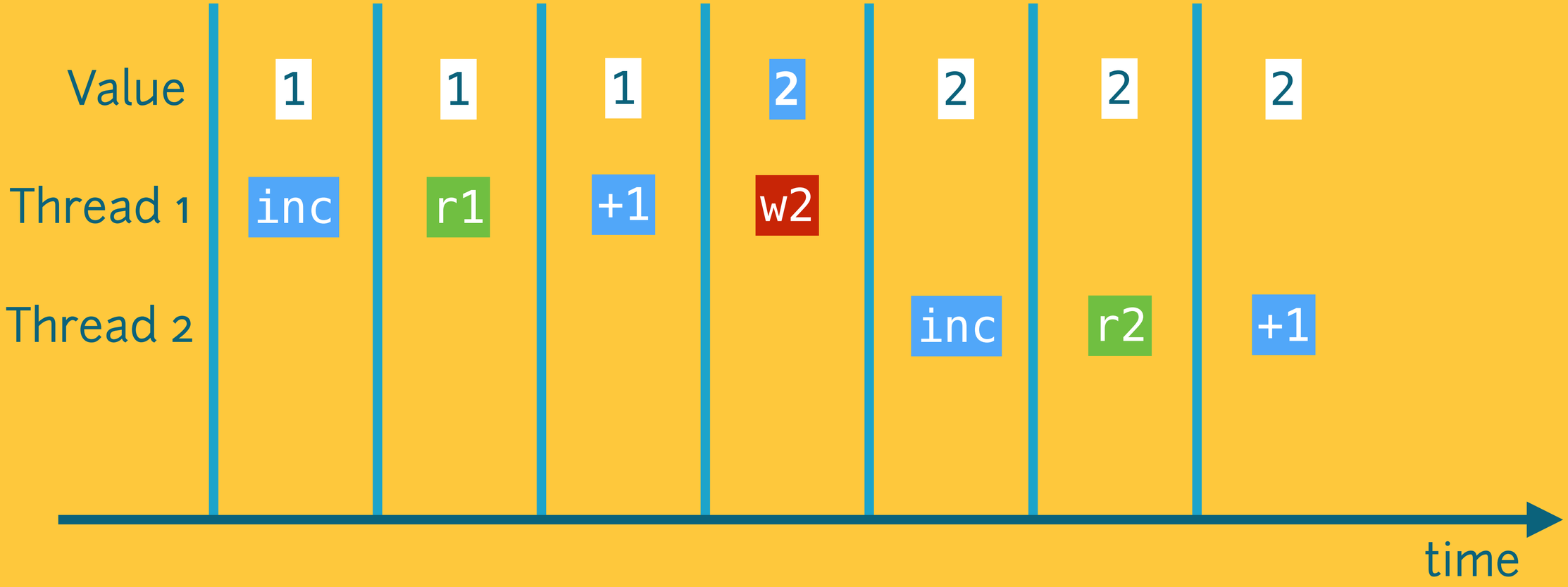
Race Condition



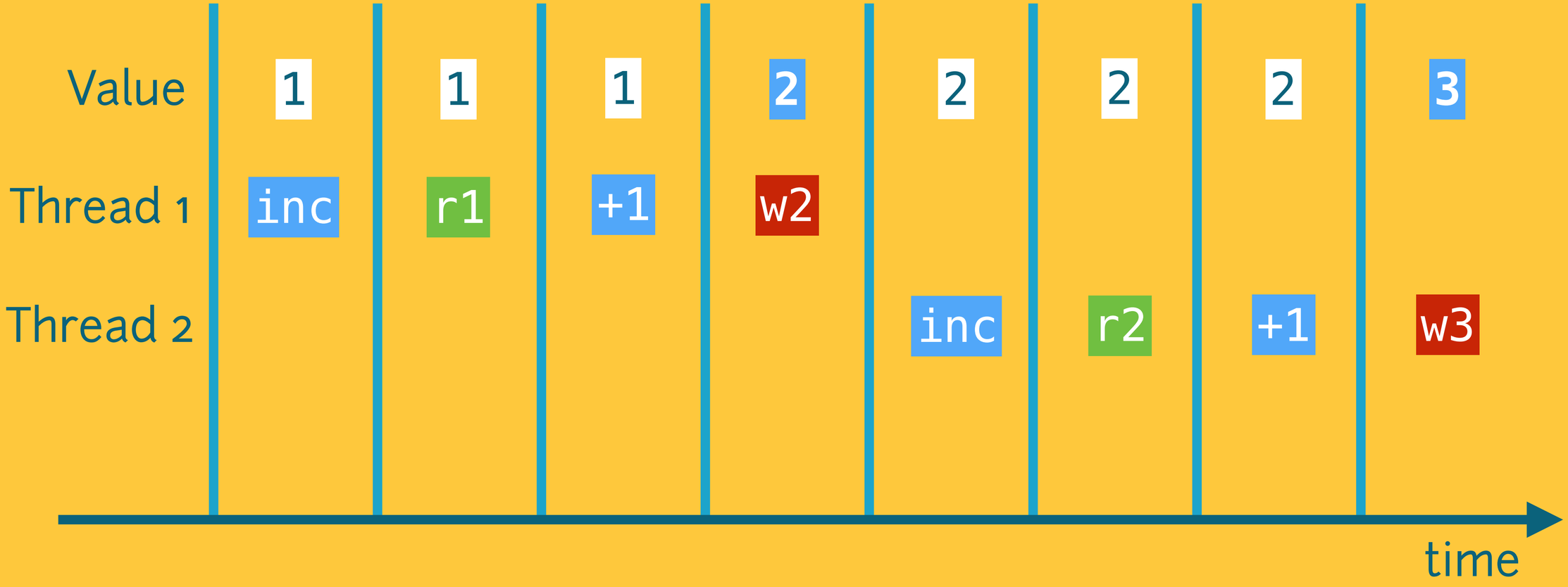
Race Condition



Race Condition

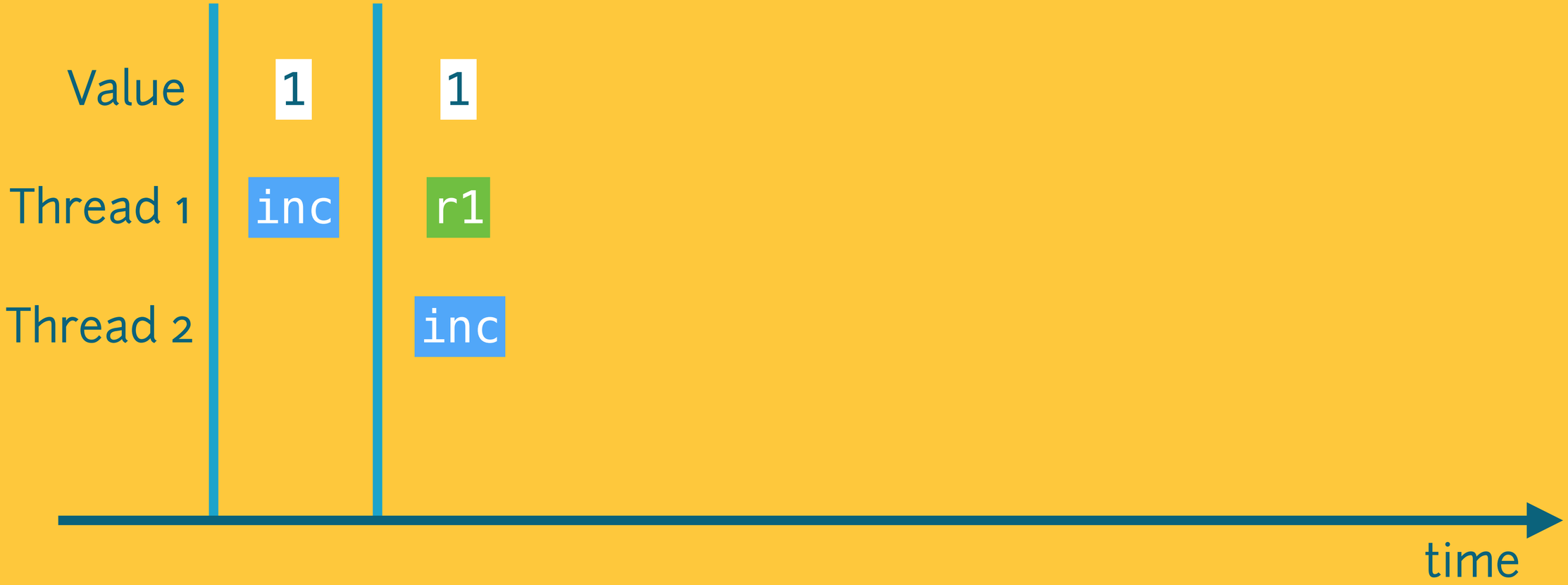


Race Condition

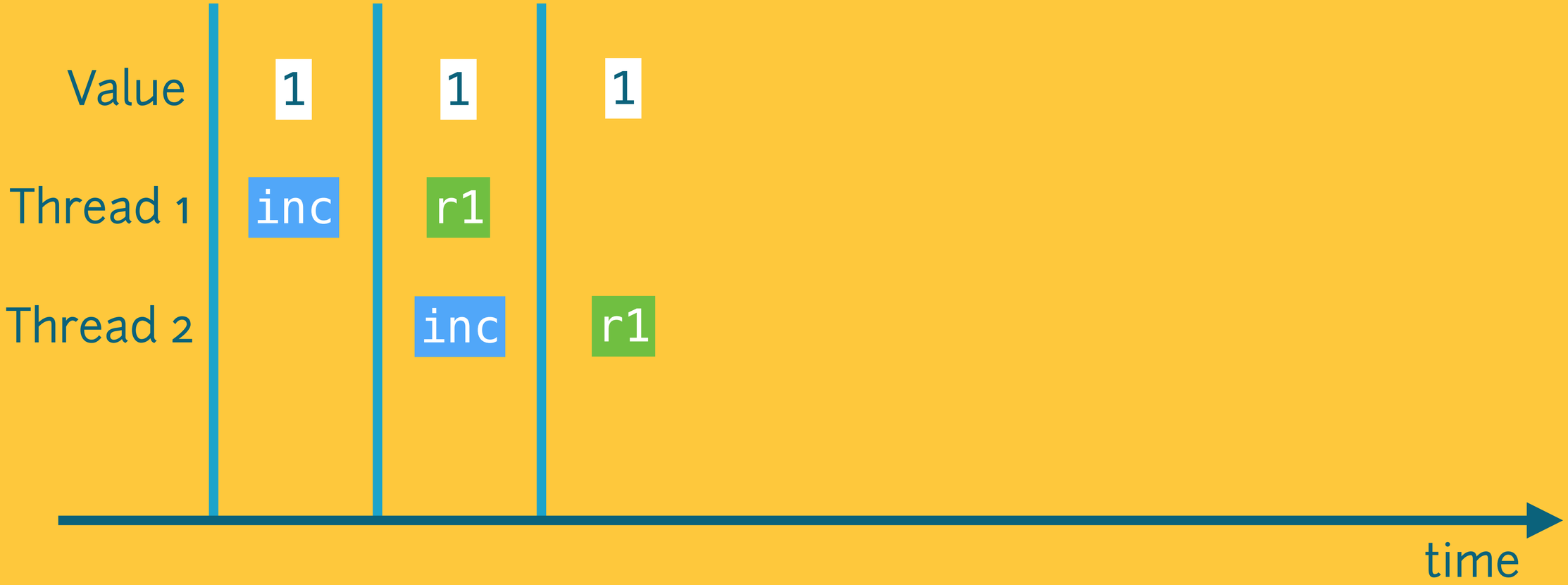


Race Condition

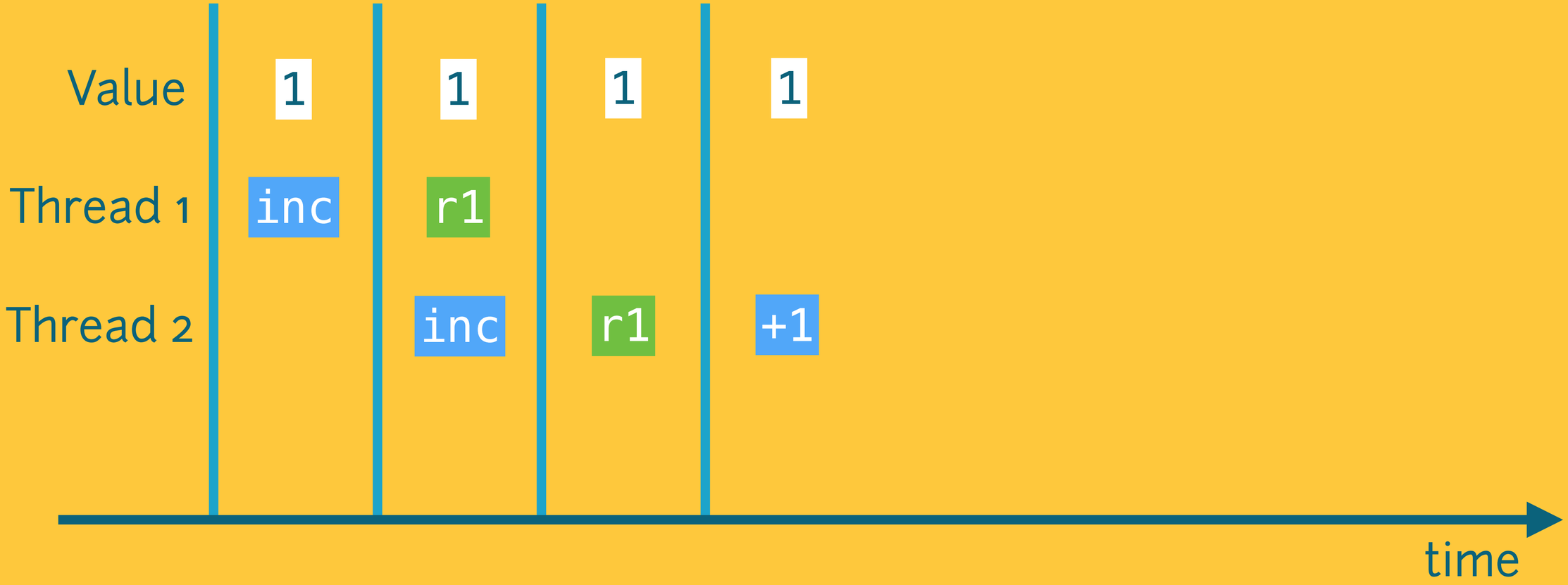
Race Condition



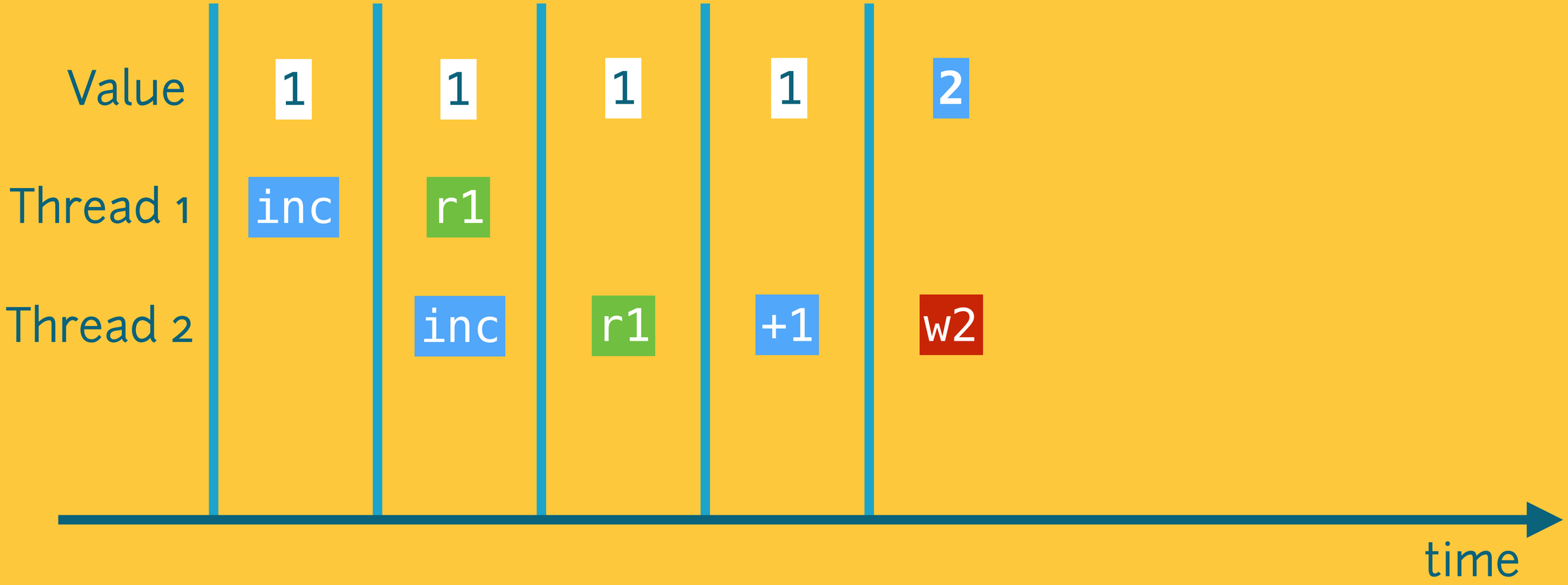
Race Condition



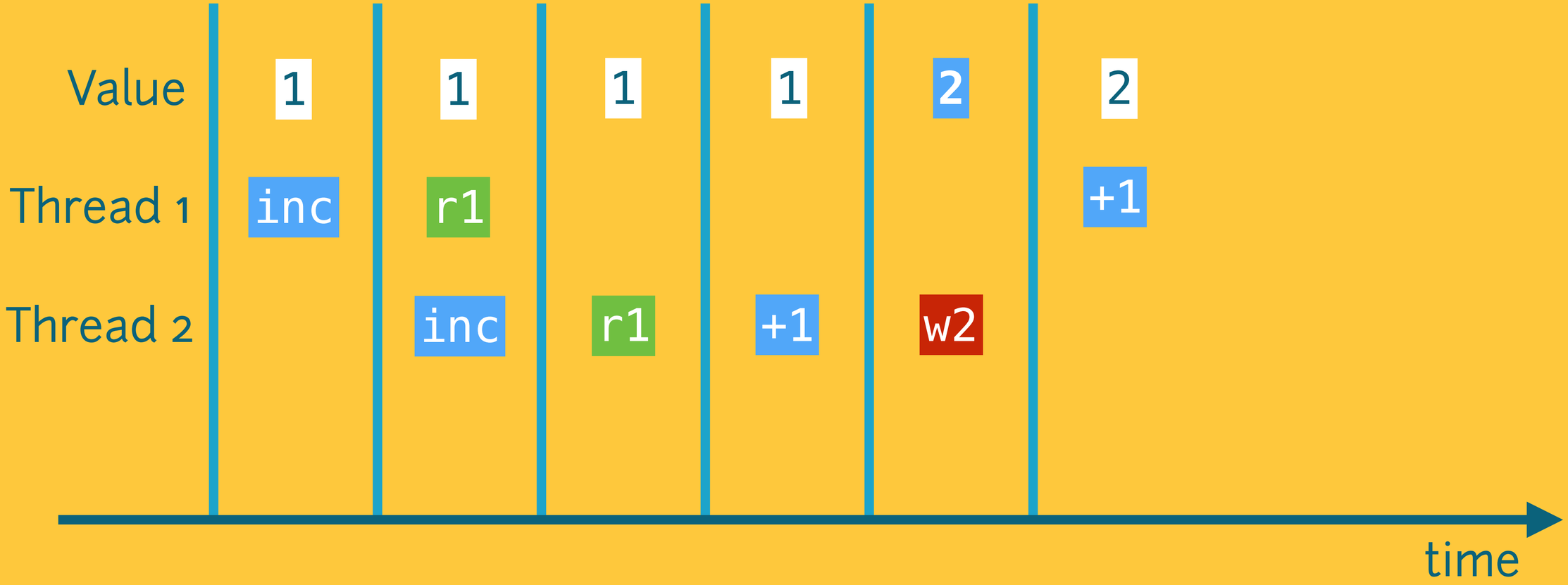
Race Condition



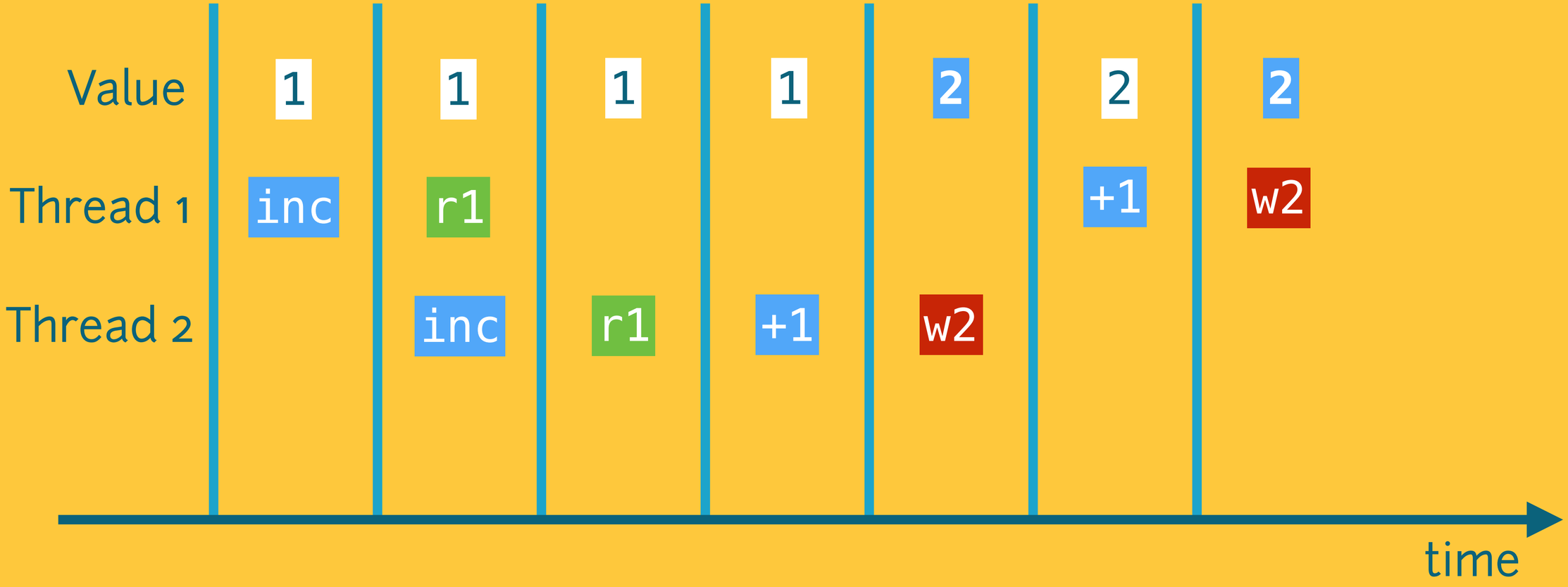
Race Condition



Race Condition



Race Condition

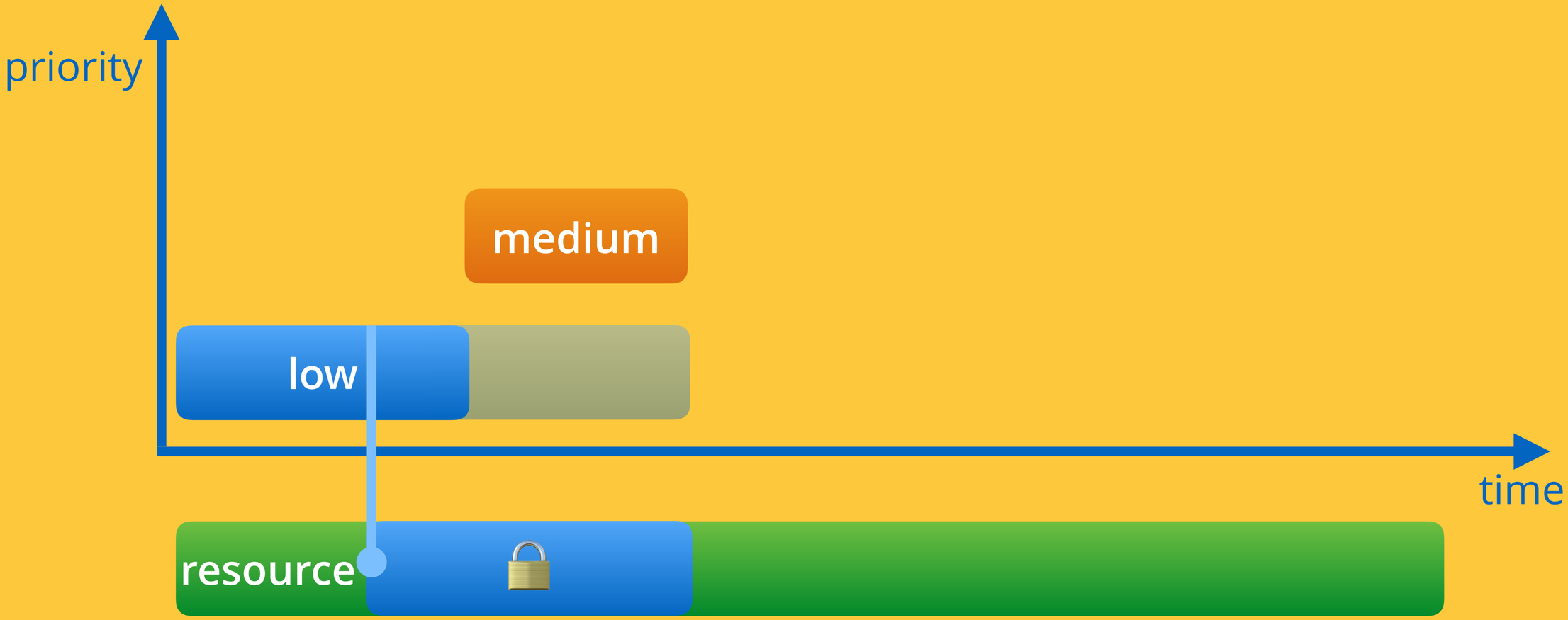


priority
inversion

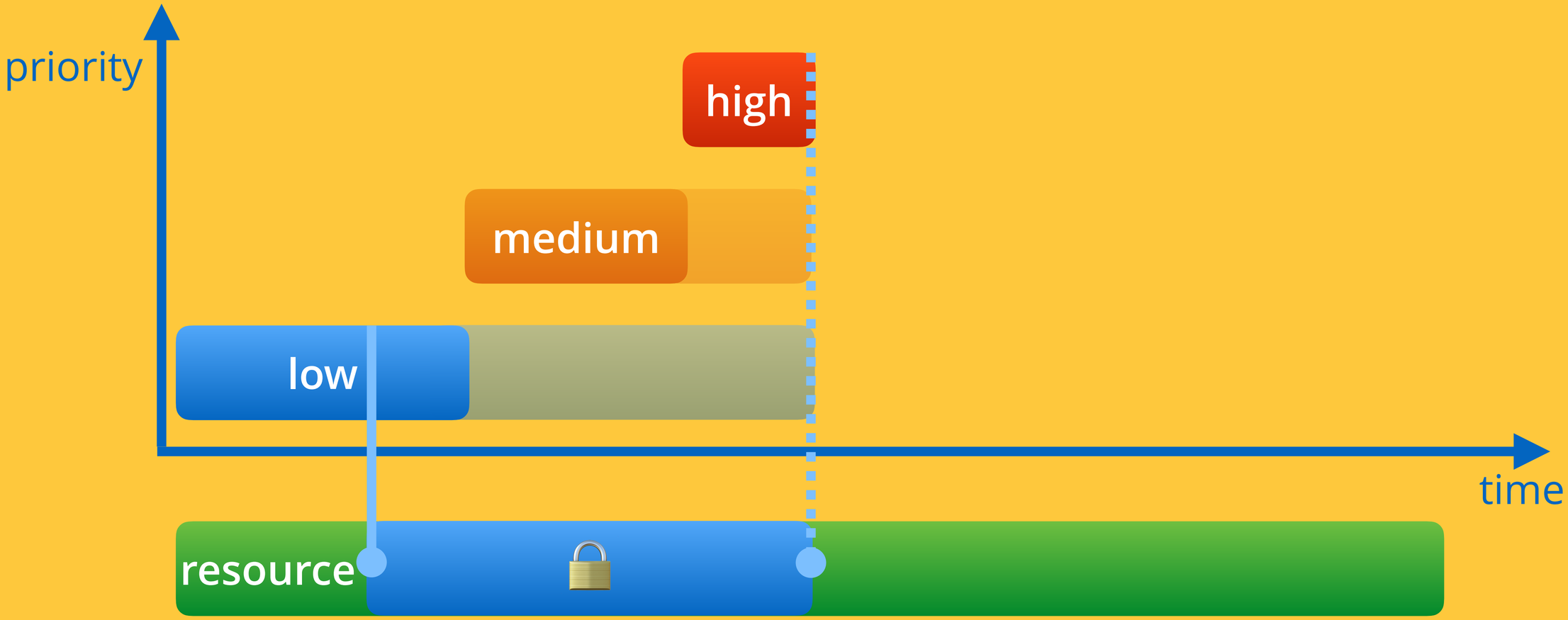
Priority Inversion

Priority Inversion

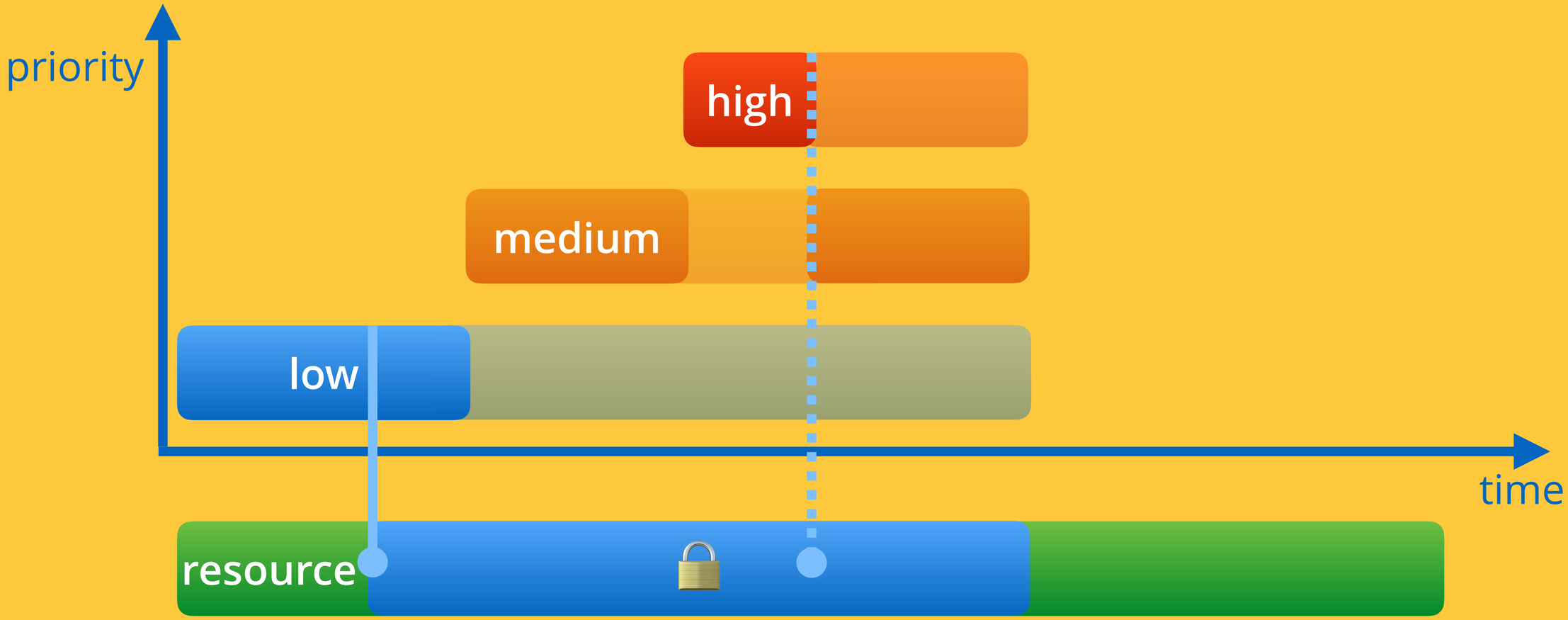
Priority Inversion



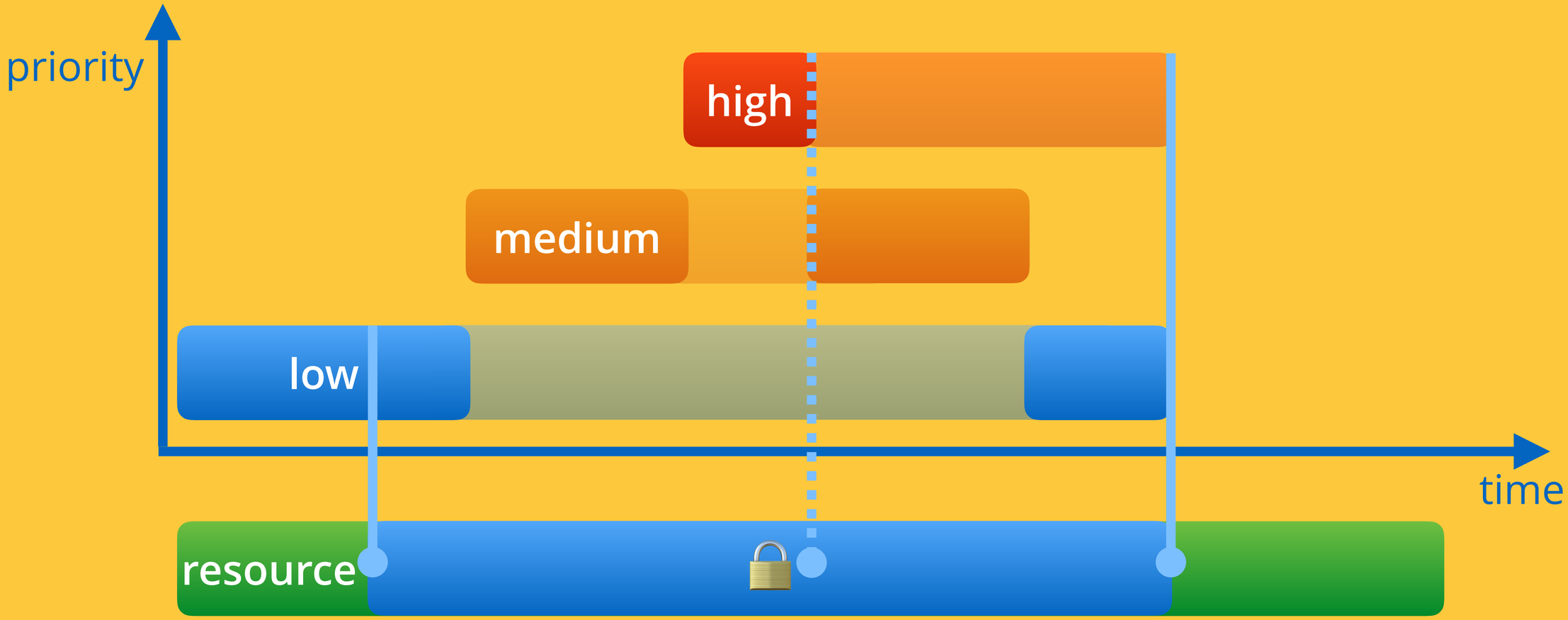
Priority Inversion



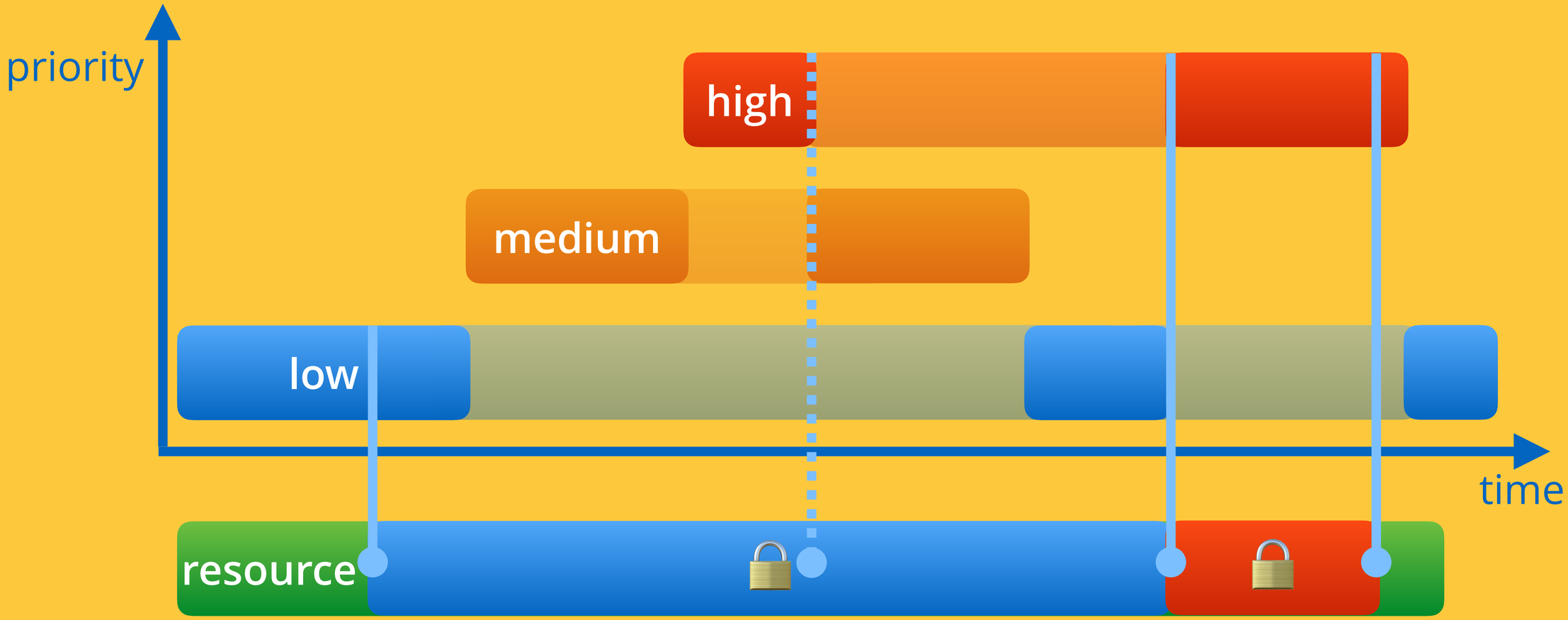
Priority Inversion



Priority Inversion

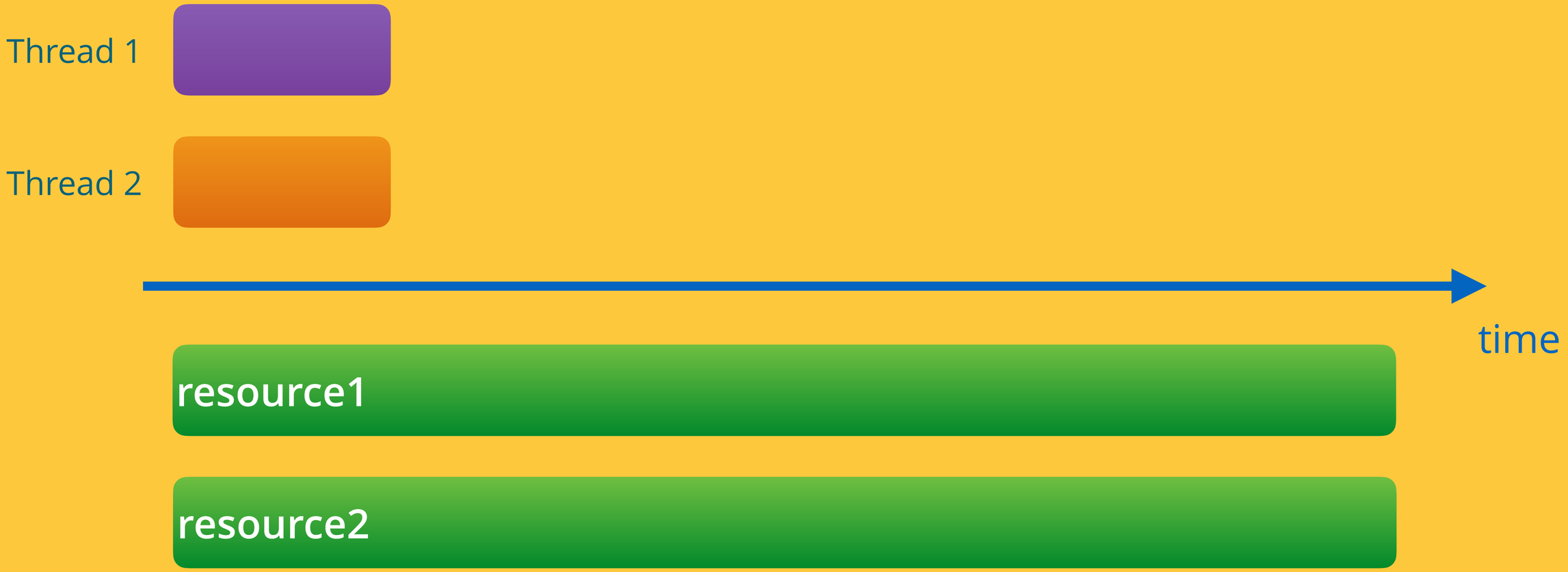


Priority Inversion

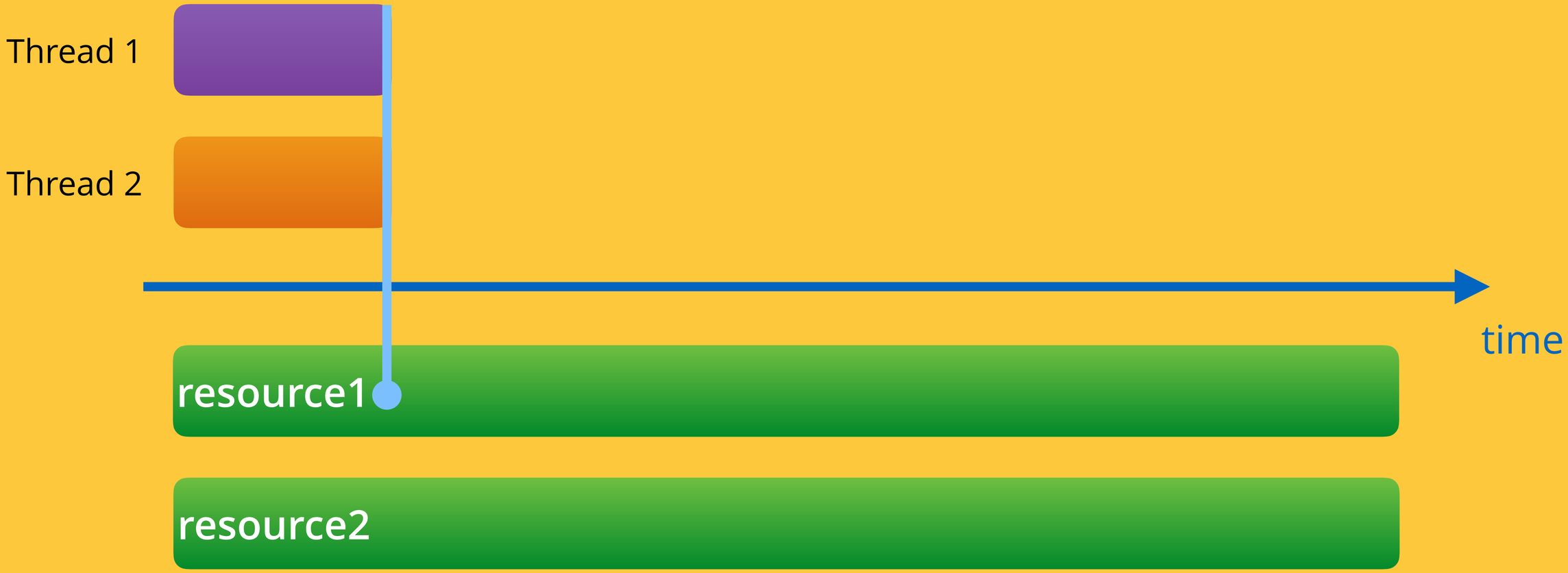



deadlock

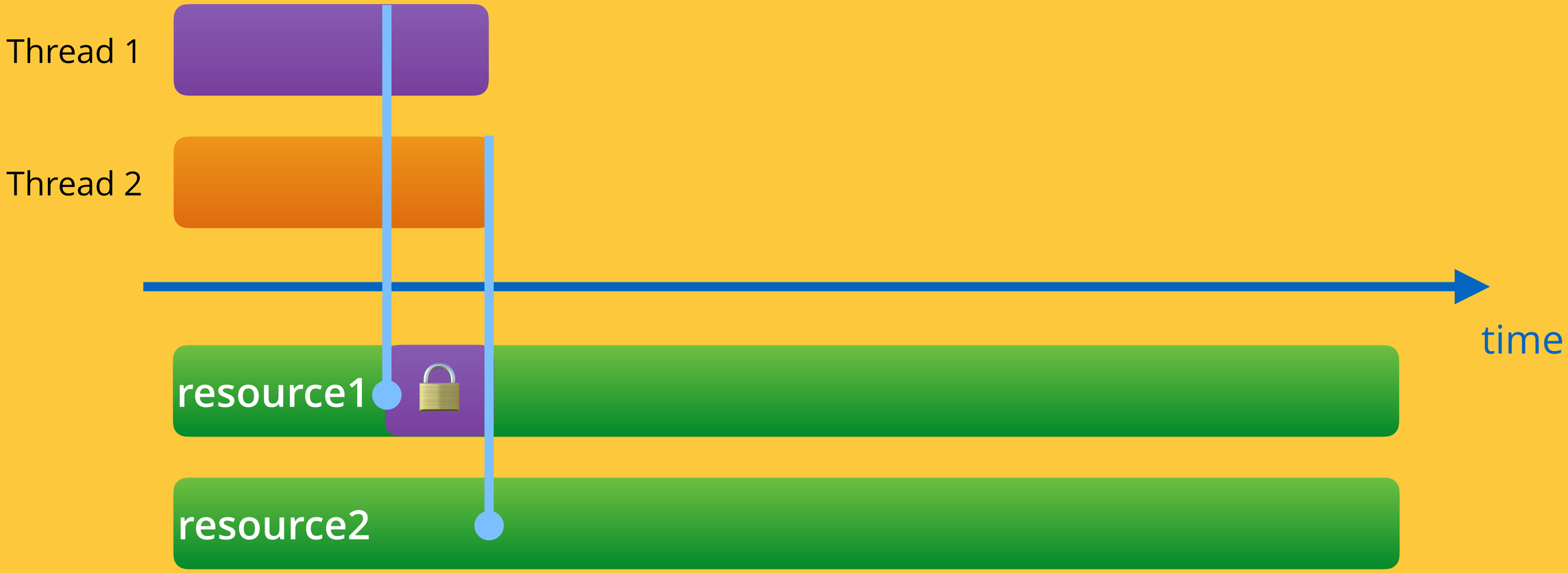
Deadlock



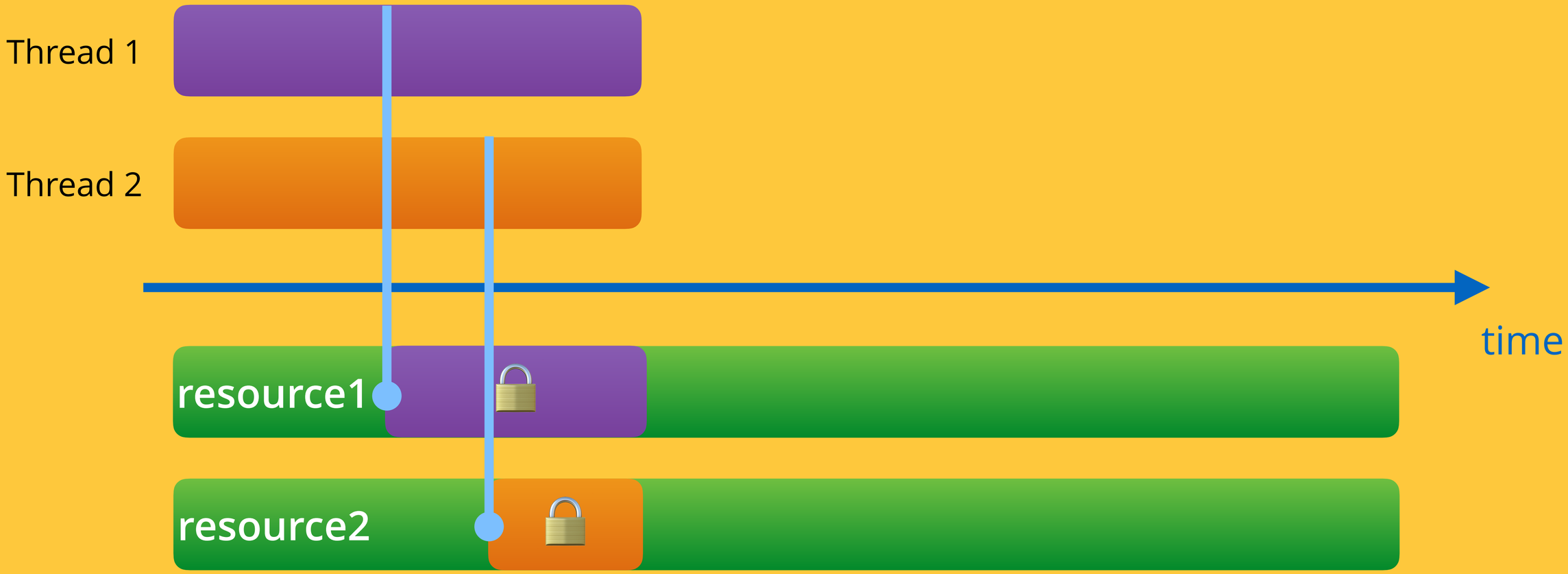
Deadlock



Deadlock

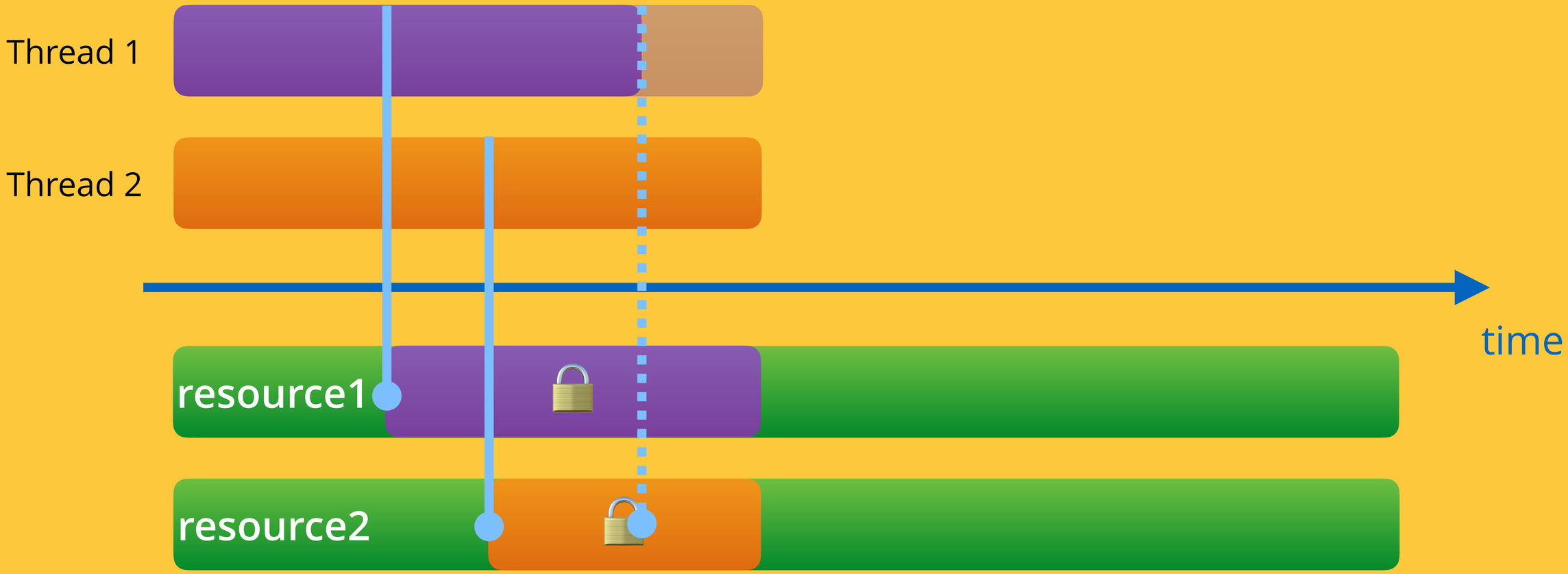


Deadlock

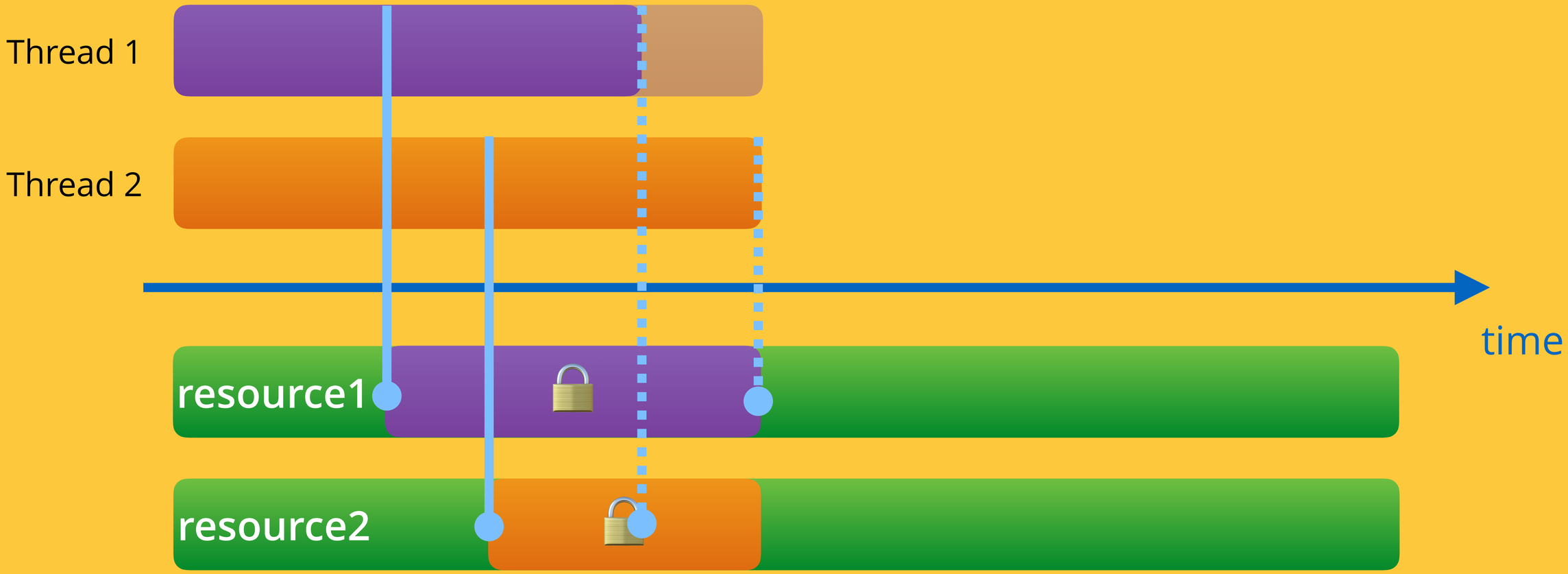


Deadlock

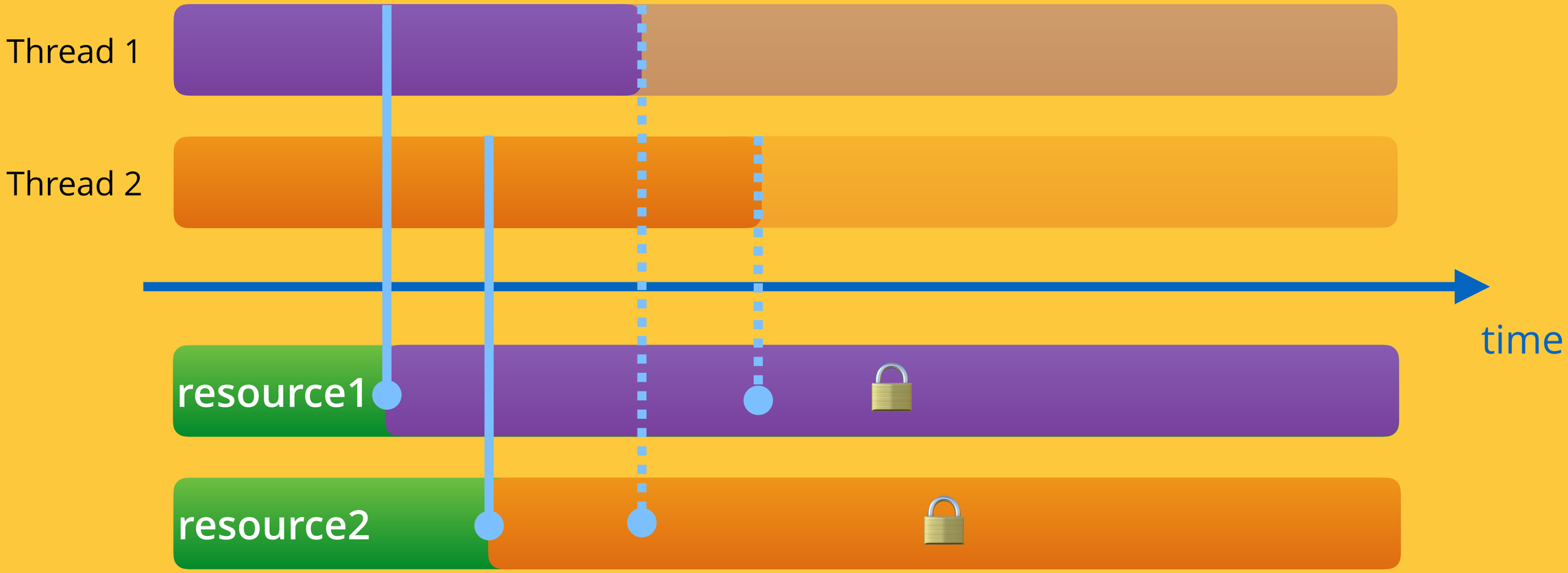
Deadlock



Deadlock



Deadlock



queueueueueueue paradigm

Tasks & Queues

Queue 

Thread 1

Thread 2

Tasks & Queues

Queue 

Thread 1 

Thread 2

Tasks & Queues

Queue

Thread 1

Thread 2

Tasks & Queues

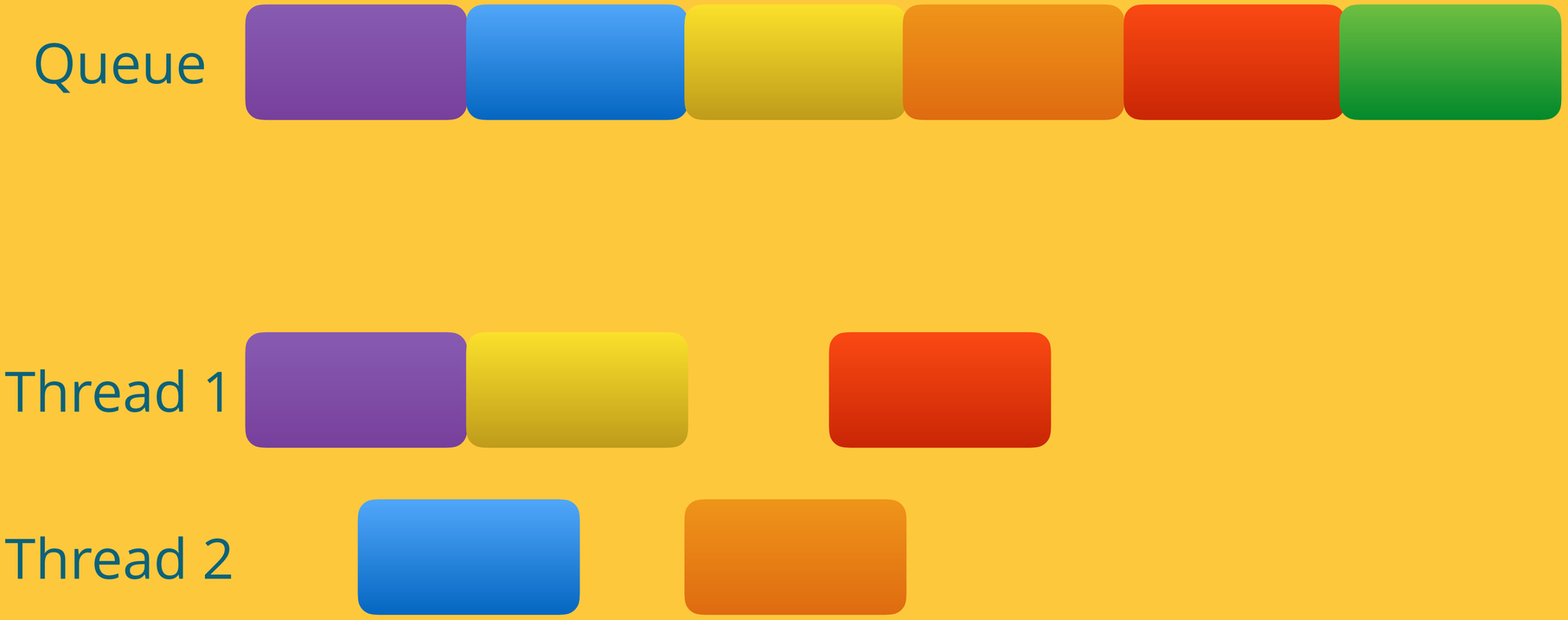
Queue

Thread 1

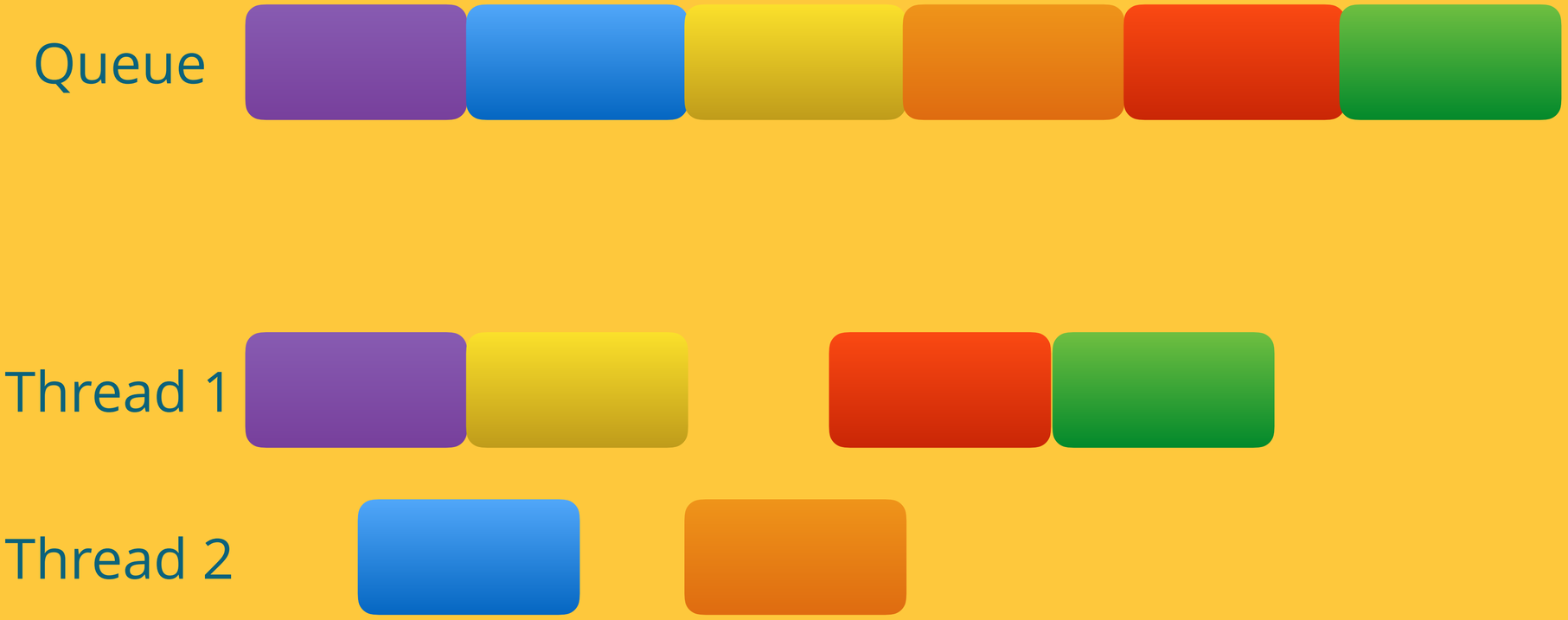
Thread 2

Tasks & Queues

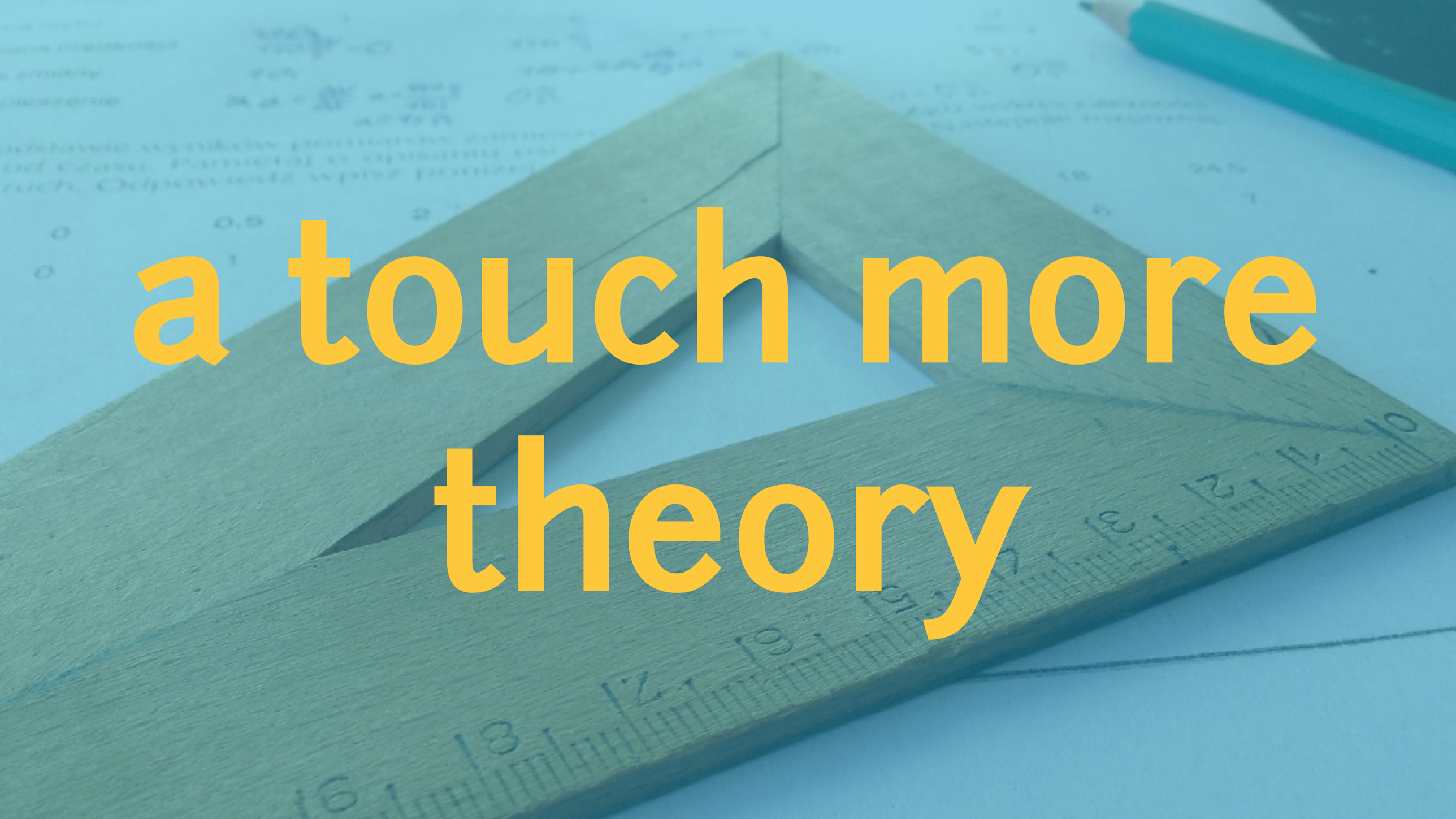
Tasks & Queues



Tasks & Queues




feeling
brave?

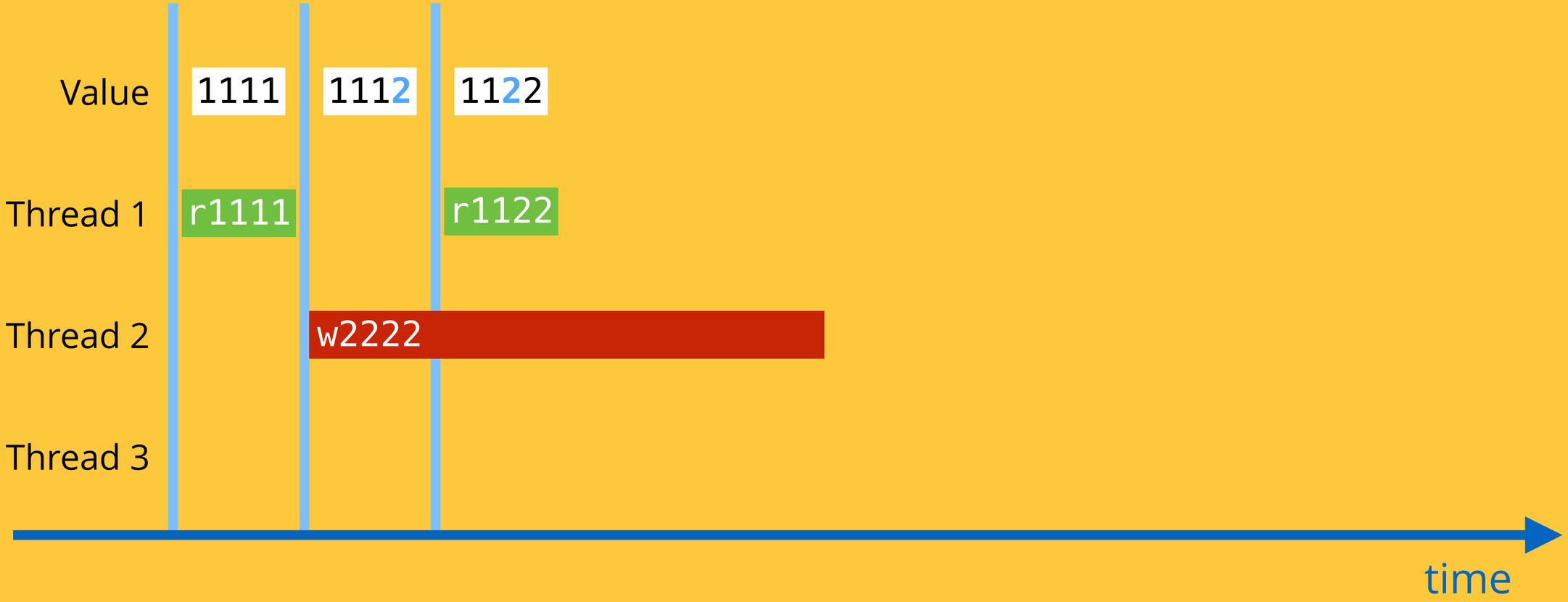

a touch more
theory

Thread Safety

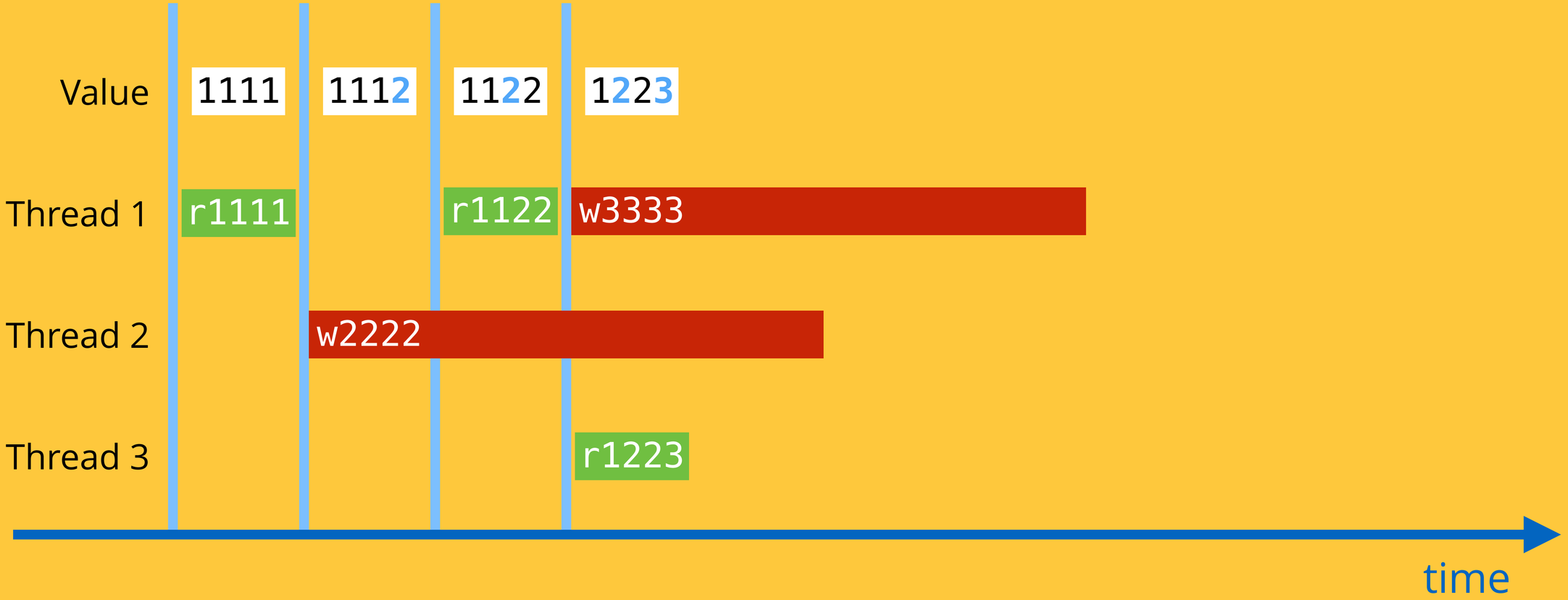
Thread Safety

Thread Safety

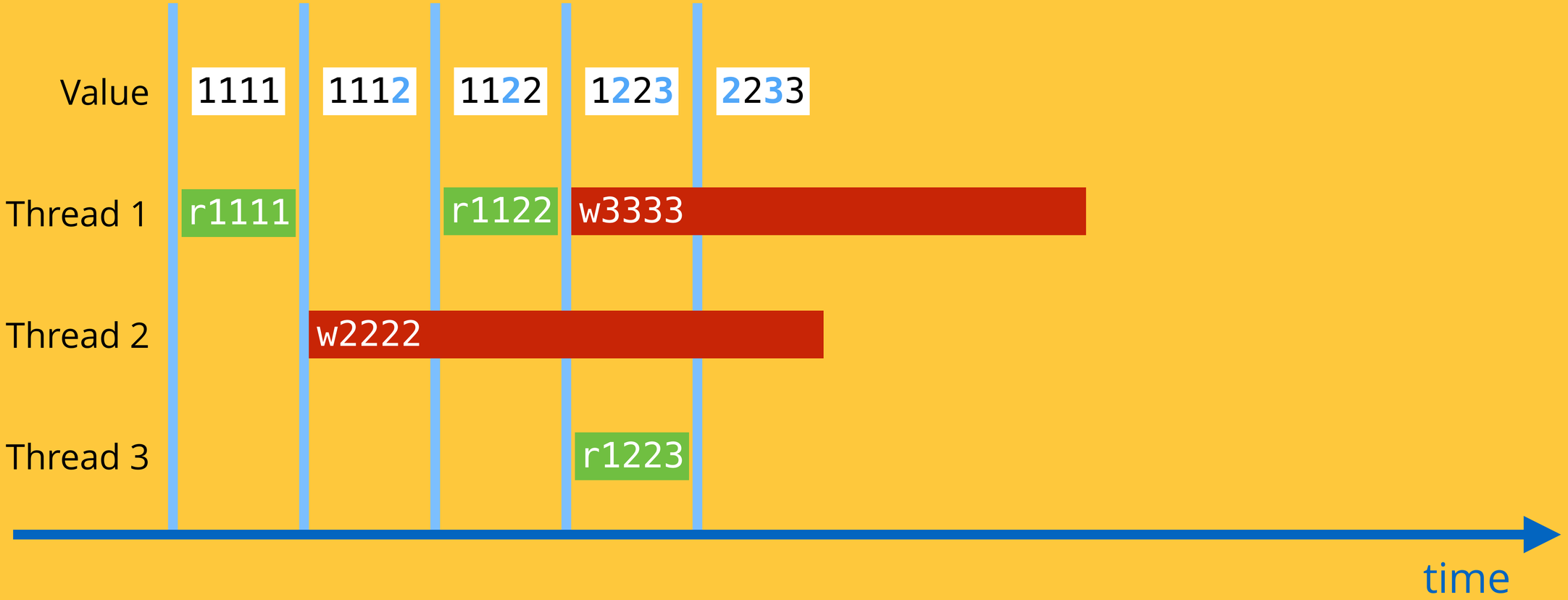
Thread Safety



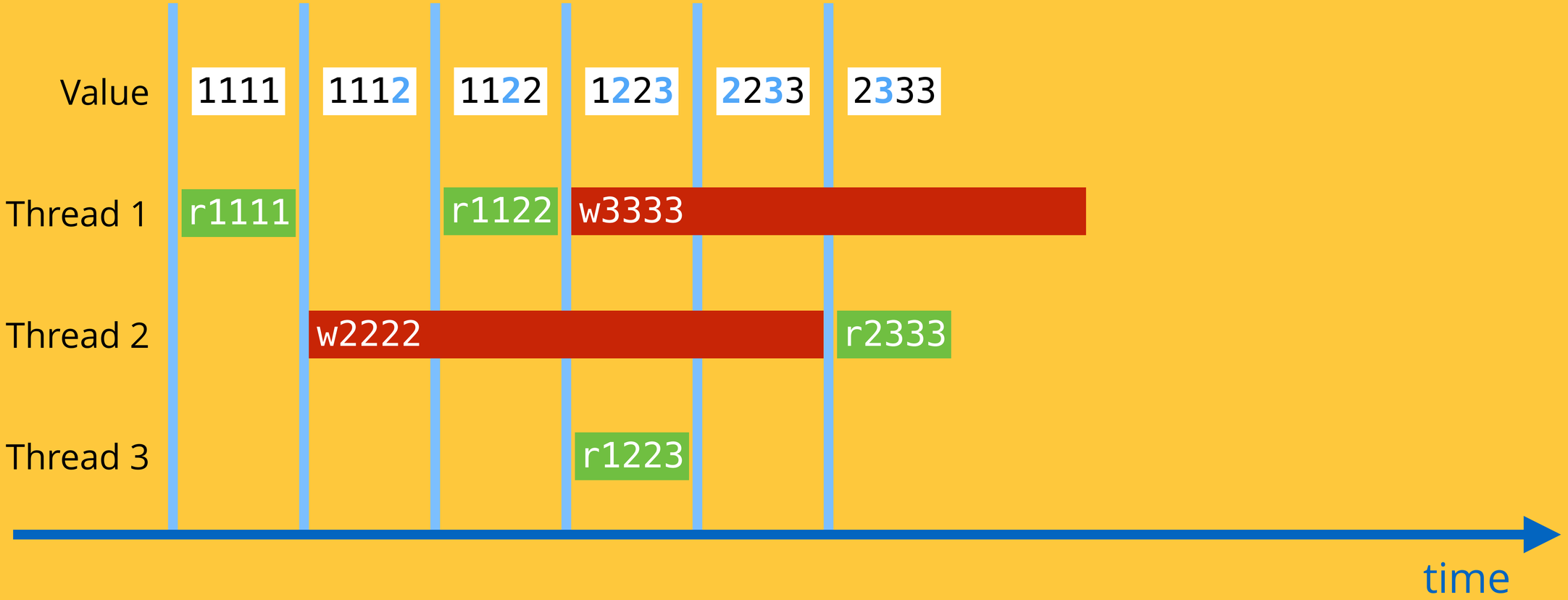
Thread Safety



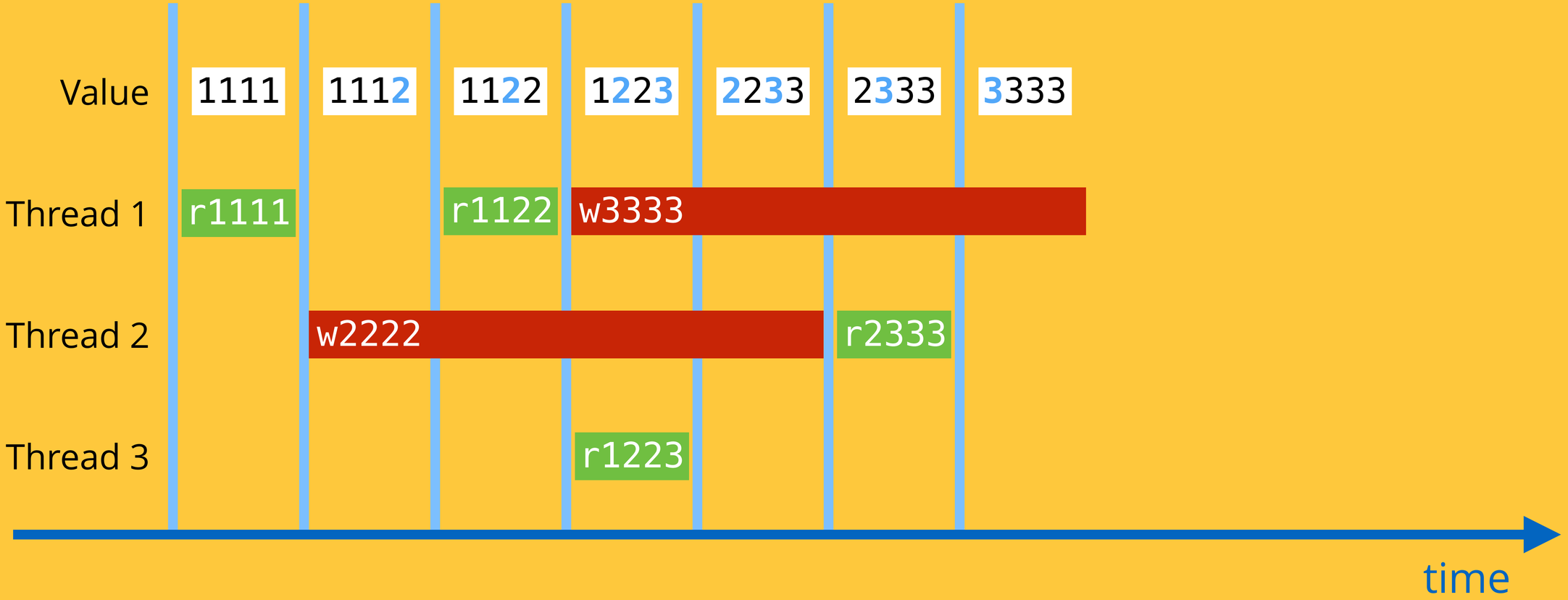
Thread Safety



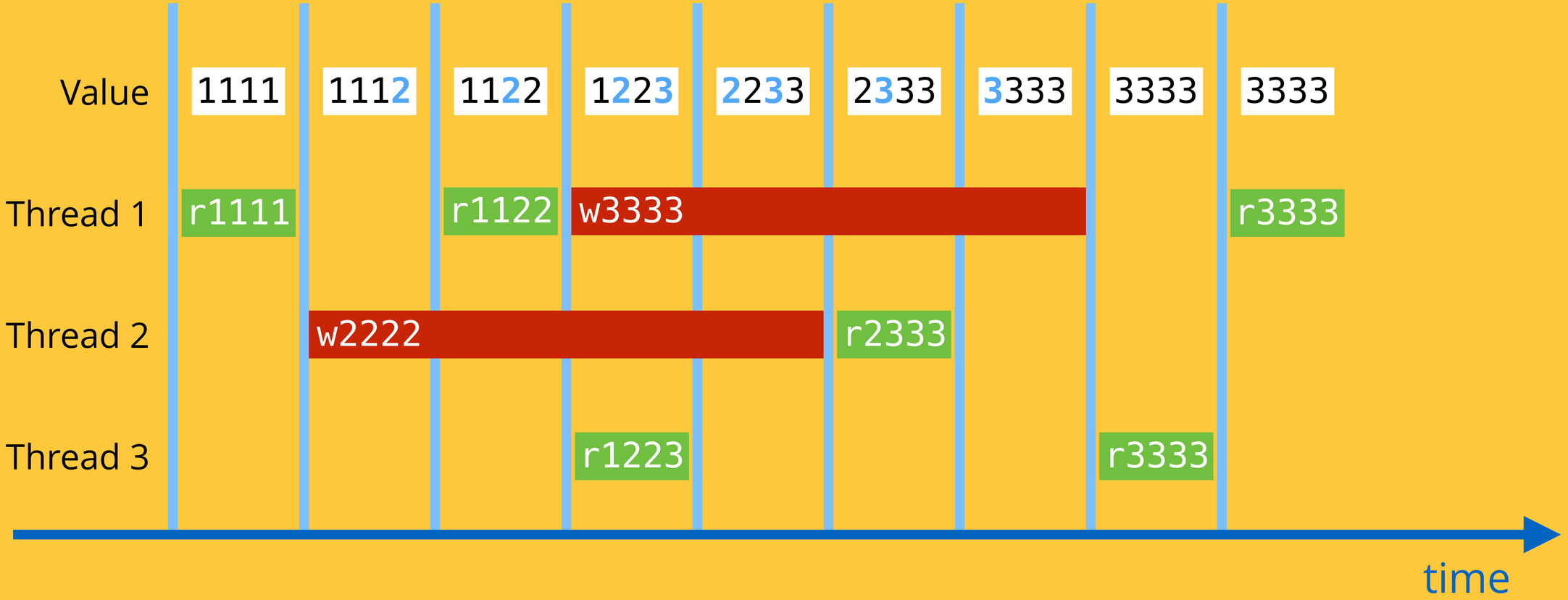
Thread Safety



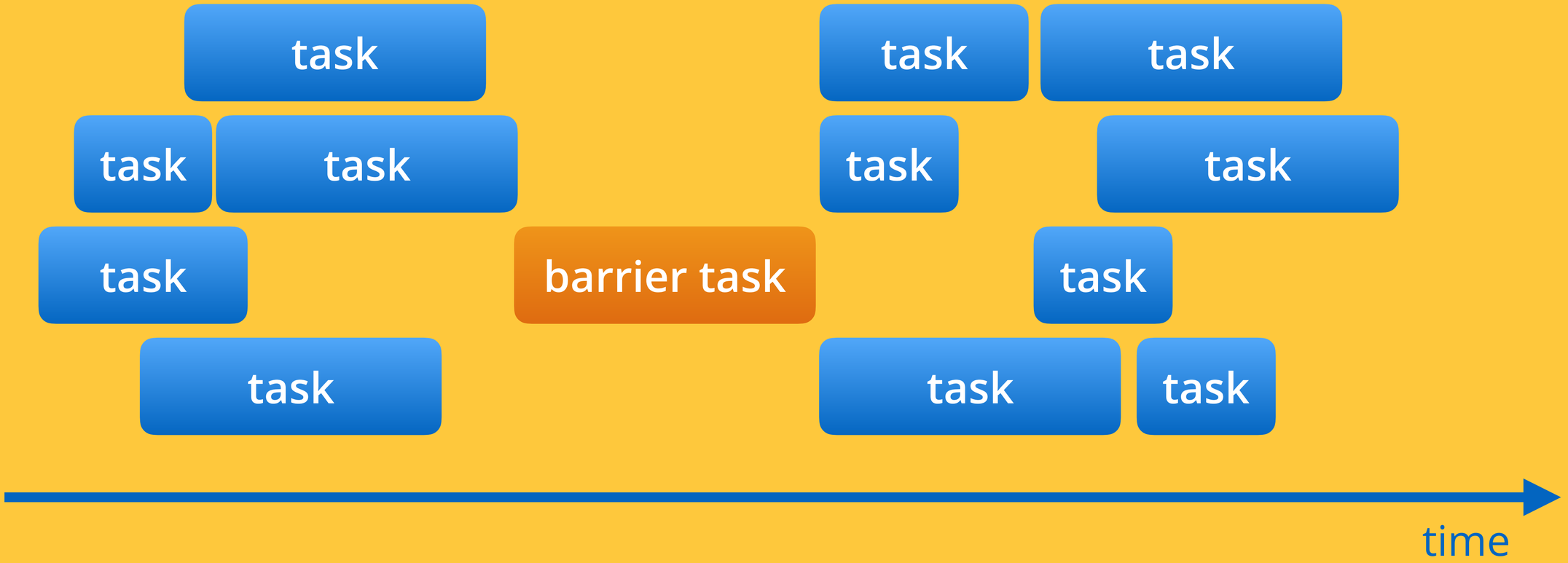
Thread Safety



Thread Safety



Dispatch Barrier



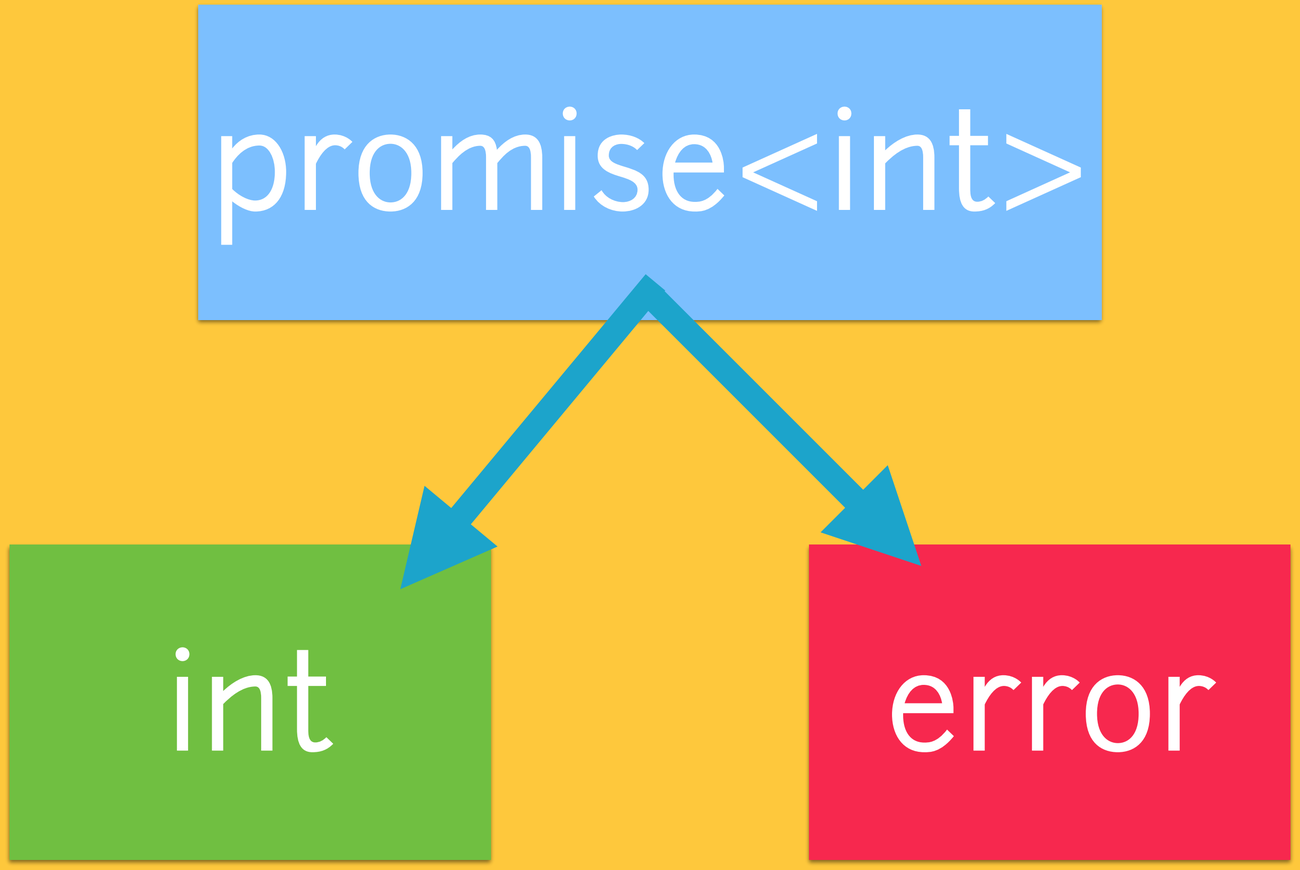
Dispatch Barrier


alternatives
to

promises

```
networkRequest(url, callback: { data in
  decompressor(data, callback: { image in
    imageResizer(image, callback: { small in
      filterImage(small, callback: { filtered in
        displayImage(filtered)
      })
    })
  })
})
```

promises



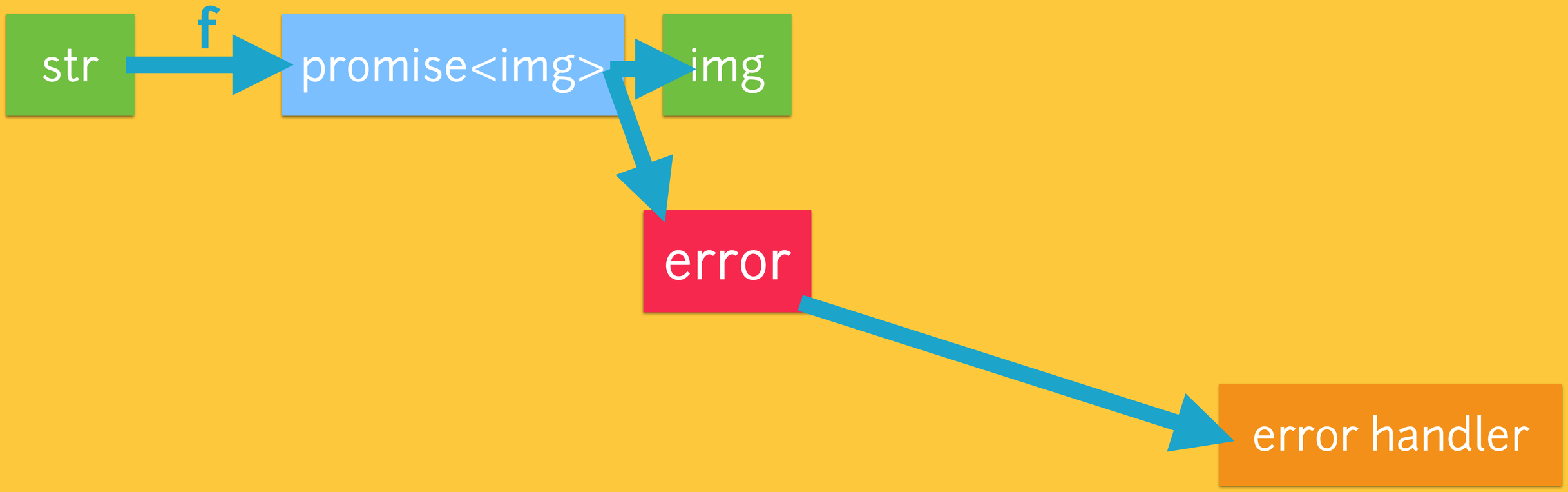
promises

str

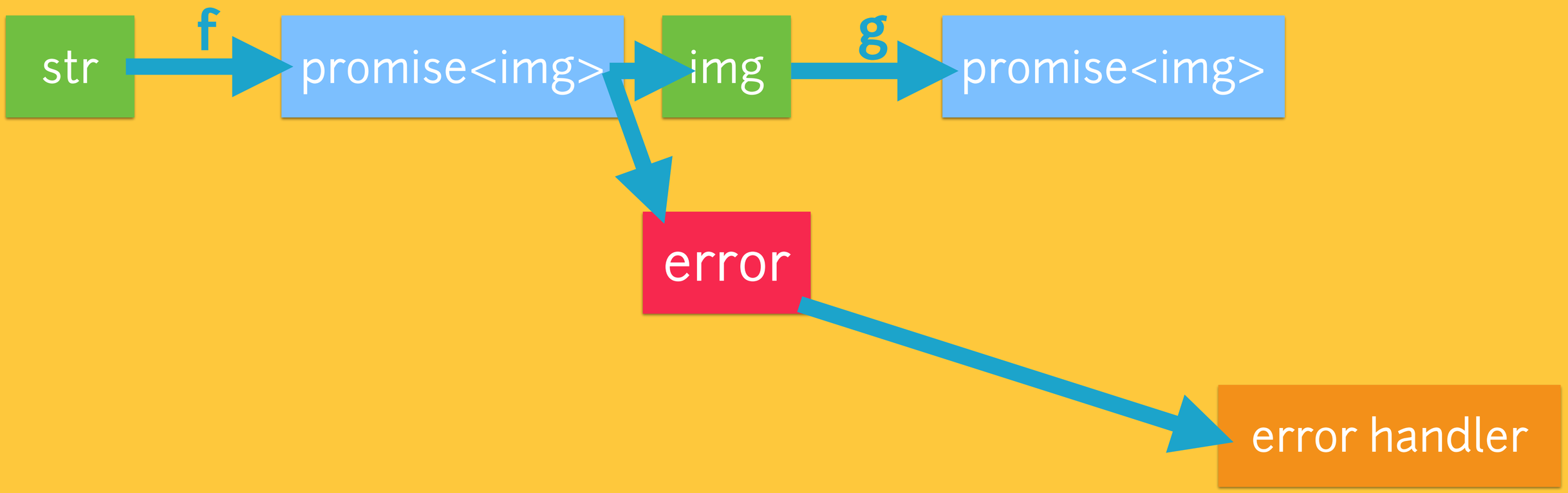
promises

promises

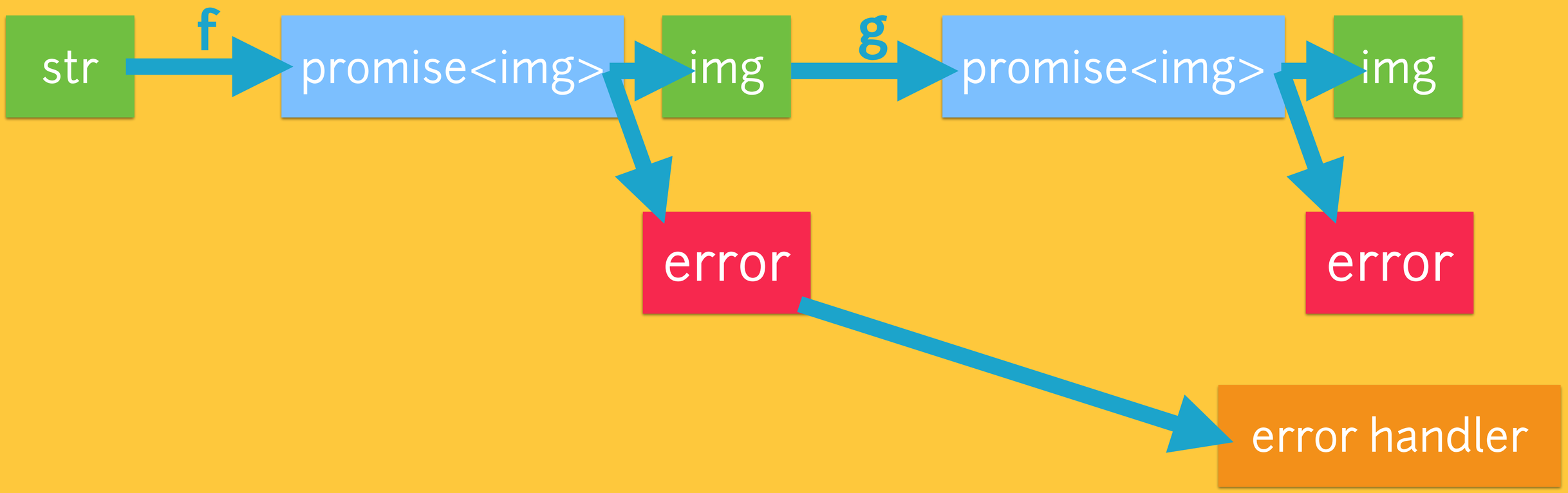
promises



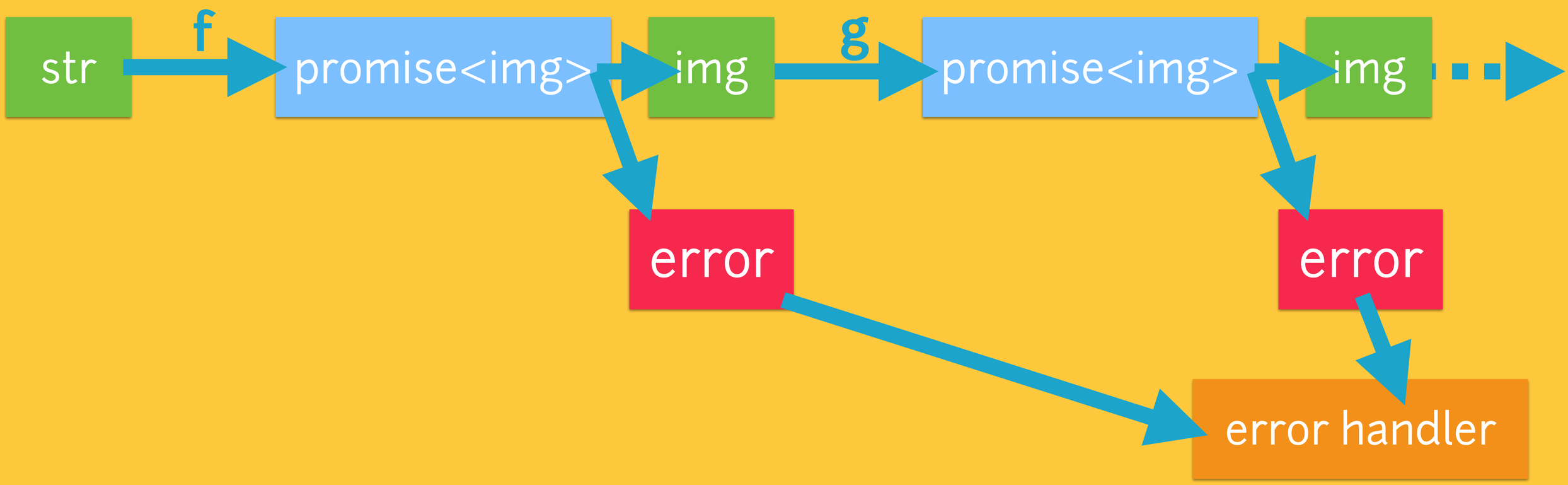
promises



promises


str
f
promise<img>
img
g
promise<img>
img
error
error
error handler

promises


str
f
promise<img>
img
g
promise<img>
img
error
error
error handler

promises

```
networkRequest(url)
  .then(decompressor)
  .then(imageResizer)
  .then({ (image) in
    return filterImage(image)
  }).then(displayImage)
  .catch({ (error) in print(error) })
```

imagine the
“promise pipeline”
accepted a
sequence of values…

that's reactive programming


conclusion


concurrency
is
hard


you will get
it
wrong

- videos.raywenderlich.com
- git.io/v9Q3S
- @iwantmyrealname