

# 异步编程

单线程 JavaScript 异步方案





```
document.write('task 1')
```

```
document.write('task 2')
```

```
document.write('task 3')
```

```
document.write('task 4')
```

```
document.write('task 5')
```



```
document.write('task 1')
```

```
document.write('task 2')
```

```
document.write('task 3')
```

```
document.write('task 4')
```

```
document.write('task 5')
```



```
document.write('task 1')
```

```
document.write('task 2')
```

```
document.write('task 3')
```

```
document.write('task 4')
```

```
document.write('task 5')
```



```
document.write('task 1')
```

```
document.write('task 2')
```

```
document.write('task 3')
```

```
document.write('task 4')
```

```
document.write('task 5')
```



```
document.write('task 1')
```

```
document.write('task 2')
```

```
document.write('task 3')
```

```
document.write('task 4')
```

```
document.write('task 5')
```



```
document.write('task 1')
```

```
document.write('task 2')
```

```
document.write('task 3')
```

```
document.write('task 4')
```

```
document.write('task 5')
```





```
console.log('foo')
```

```
for (let i = 0; i < 100000; i++) {  
  console.log('耗时操作')  
}
```

```
console.log('等待耗时操作结束')
```



```
console.log('foo')
```

```
for (let i = 0; i < 100000; i++) {  
  console.log('耗时操作')  
}
```

```
console.log('等待耗时操作结束')
```



```
console.log('foo')
```

```
for (let i = 0; i < 100000; i++) {  
  console.log('耗时操作')  
}
```

```
console.log('等待耗时操作结束')
```

# 内容概要

SUMMARY

- 同步模式与异步模式
- 事件循环与消息队列
- 异步编程的几种方式
- Promise 异步方案、宏任务 / 微任务队列
- Generator 异步方案、Async / Await 语法糖

# 同步模式

Synchronous

```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```

```
console.log('global begin')

function bar () {
  console.log('bar task')
}

function foo () {
  console.log('foo task')
  bar()
}

foo()

console.log('global end')
```

Console

Call stack

```
console.log('global begin')

function bar () {
  console.log('bar task')
}

function foo () {
  console.log('foo task')
  bar()
}

foo()

console.log('global end')
```



Call stack

Console



```
console.log('global begin')

function bar () {
  console.log('bar task')
}

function foo () {
  console.log('foo task')
  bar()
}

foo()

console.log('global end')
```



Call stack

(anonymous)

Console

```
console.log('global begin')
```



```
function bar () {  
  console.log('bar task')  
}
```

```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```

Console

Call stack

(anonymous)

```
console.log('global begin')
```



```
function bar () {  
  console.log('bar task')  
}
```

```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```

Console

Call stack

```
console.log('global begin')
```

```
(anonymous)
```

```
console.log('global begin')
```



```
function bar () {  
  console.log('bar task')  
}
```

```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```

#### Console

global begin

#### Call stack

console.log('global begin')

(anonymous)

```
console.log('global begin')
```



```
function bar () {  
  console.log('bar task')  
}
```

```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```

### Console

global begin

### Call stack

(anonymous)

```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```



### Console

global begin

### Call stack

(anonymous)

```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

```
function foo () {  
  console.log('foo task')  
  bar()  
}
```



```
foo()
```

```
console.log('global end')
```

Console

global begin

Call stack

(anonymous)

```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```

Console

global begin

Call stack

(anonymous)



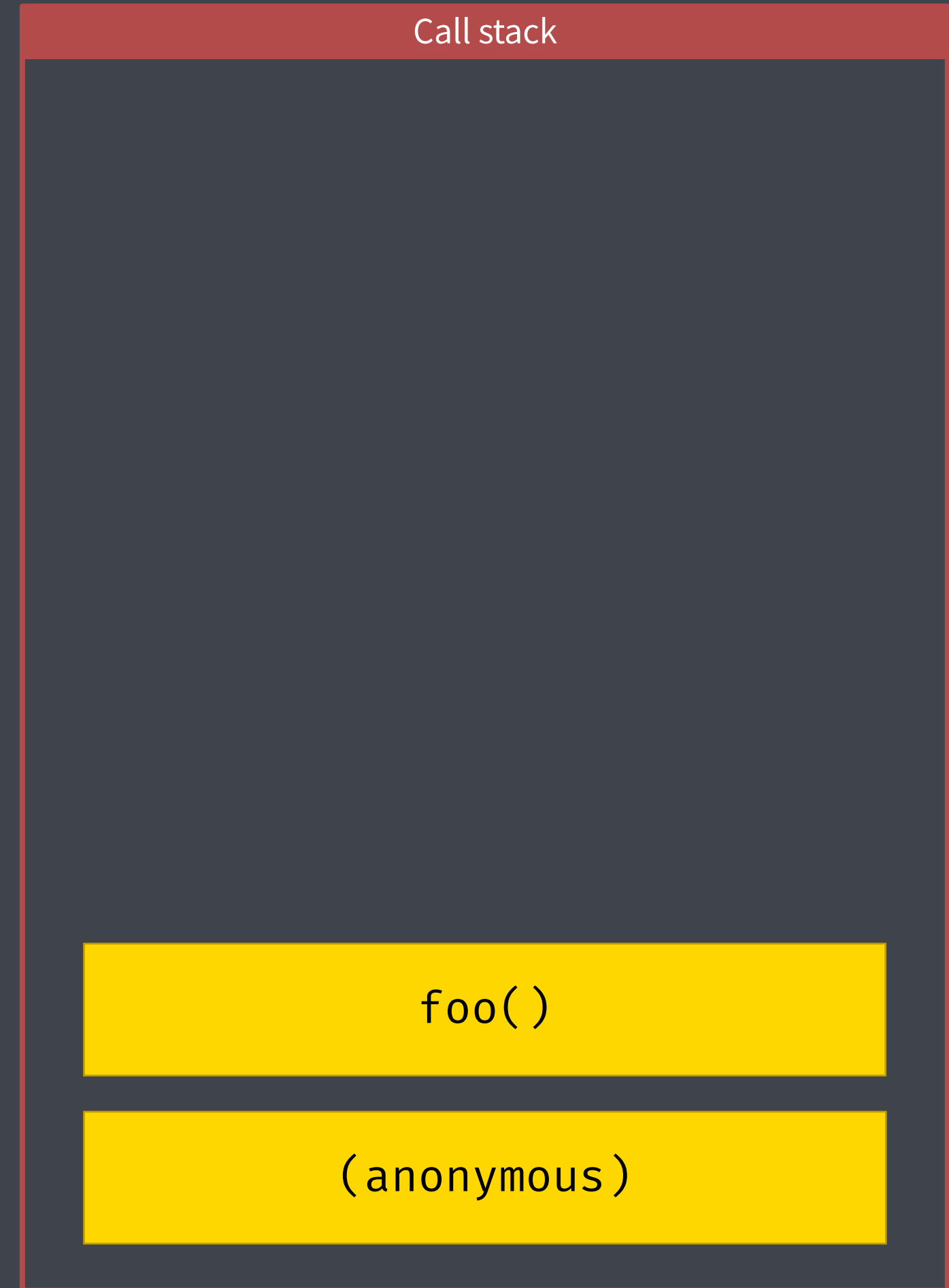
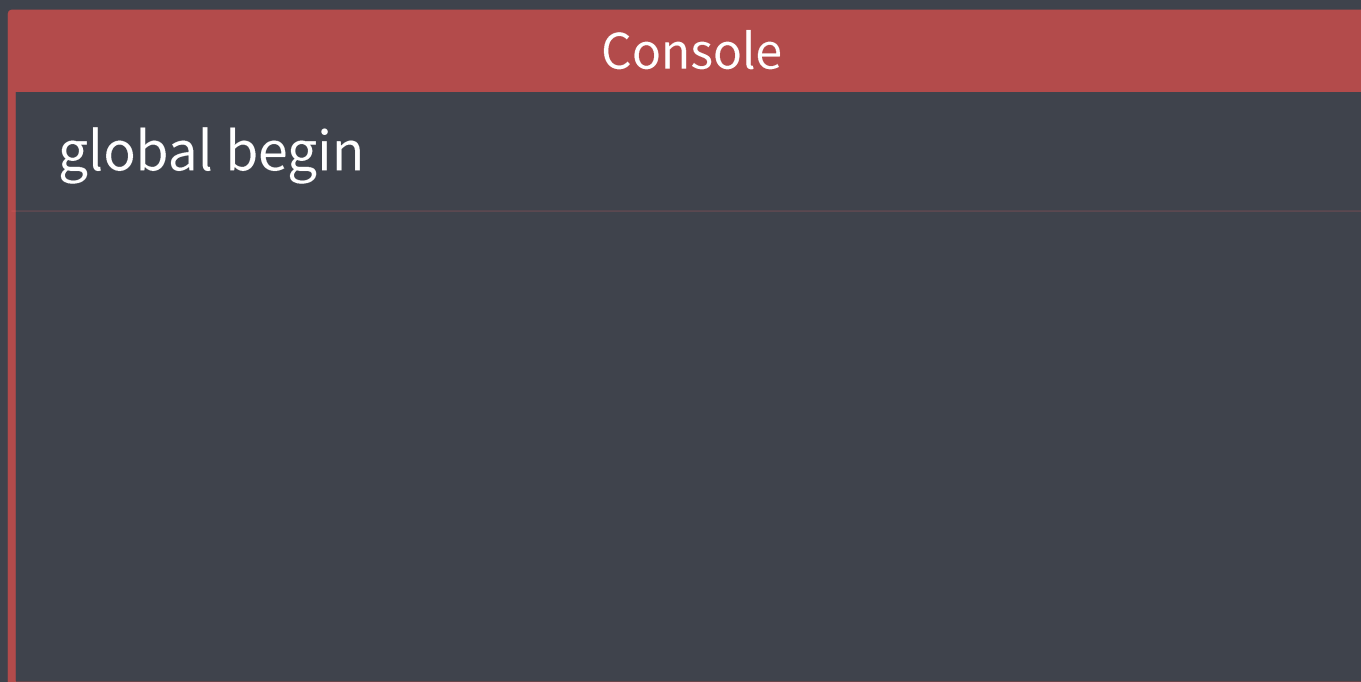
```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```



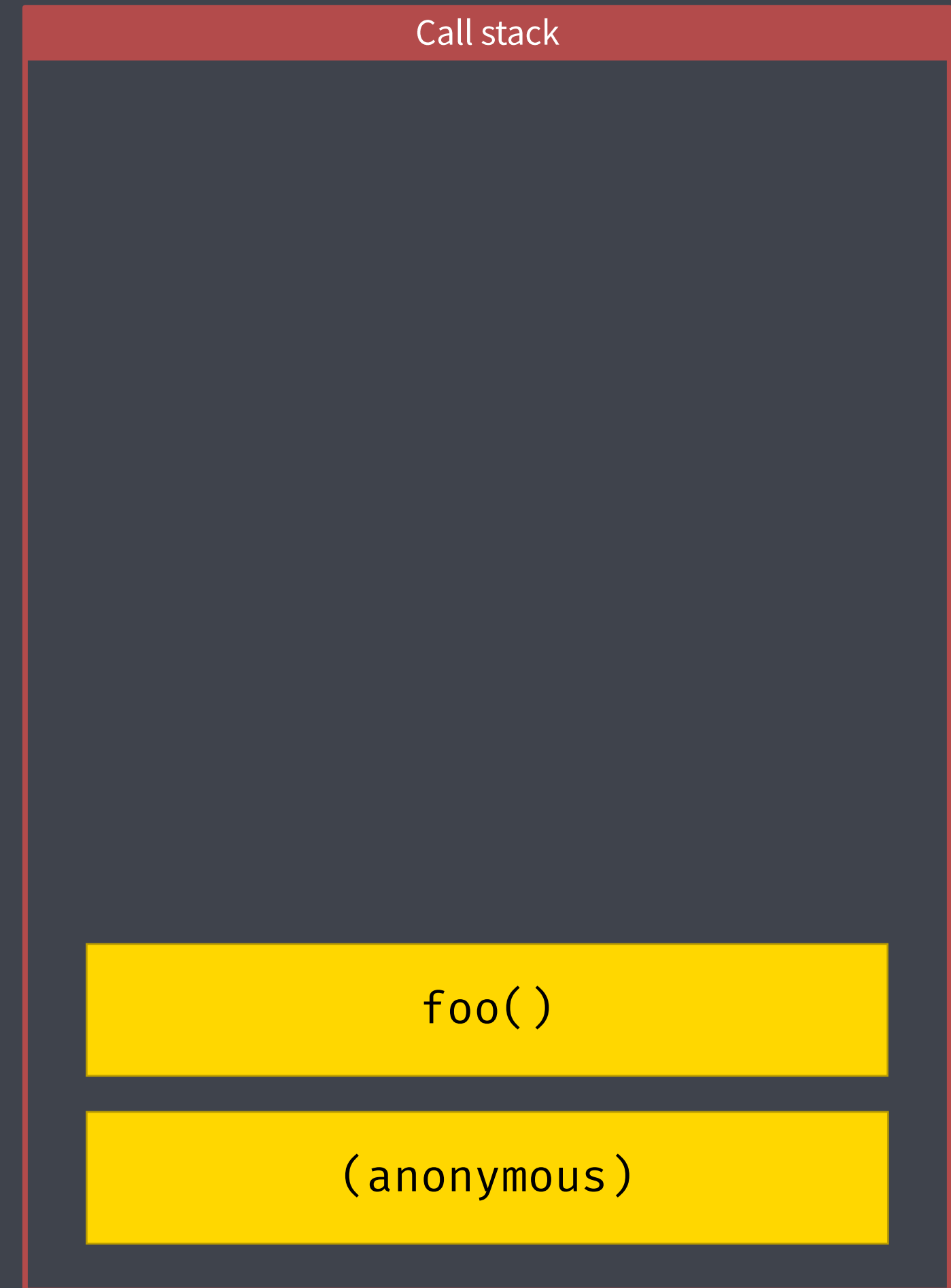
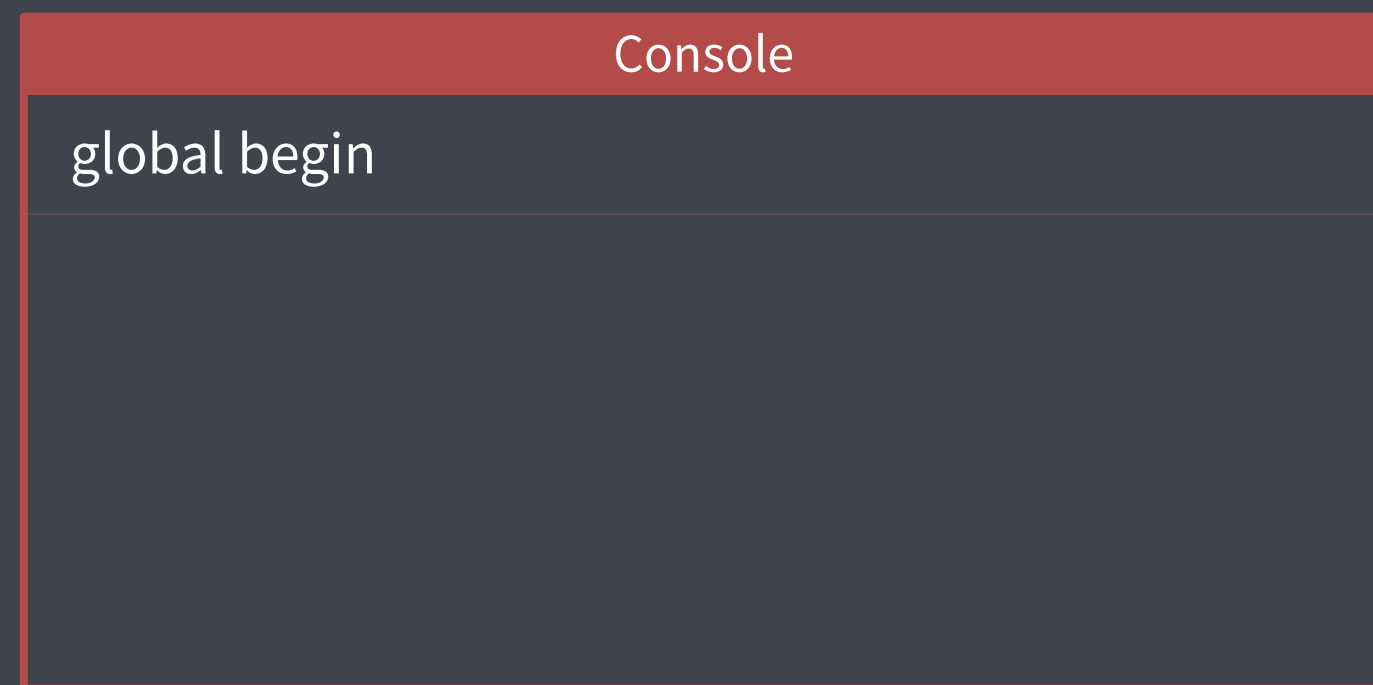
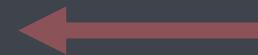
```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```



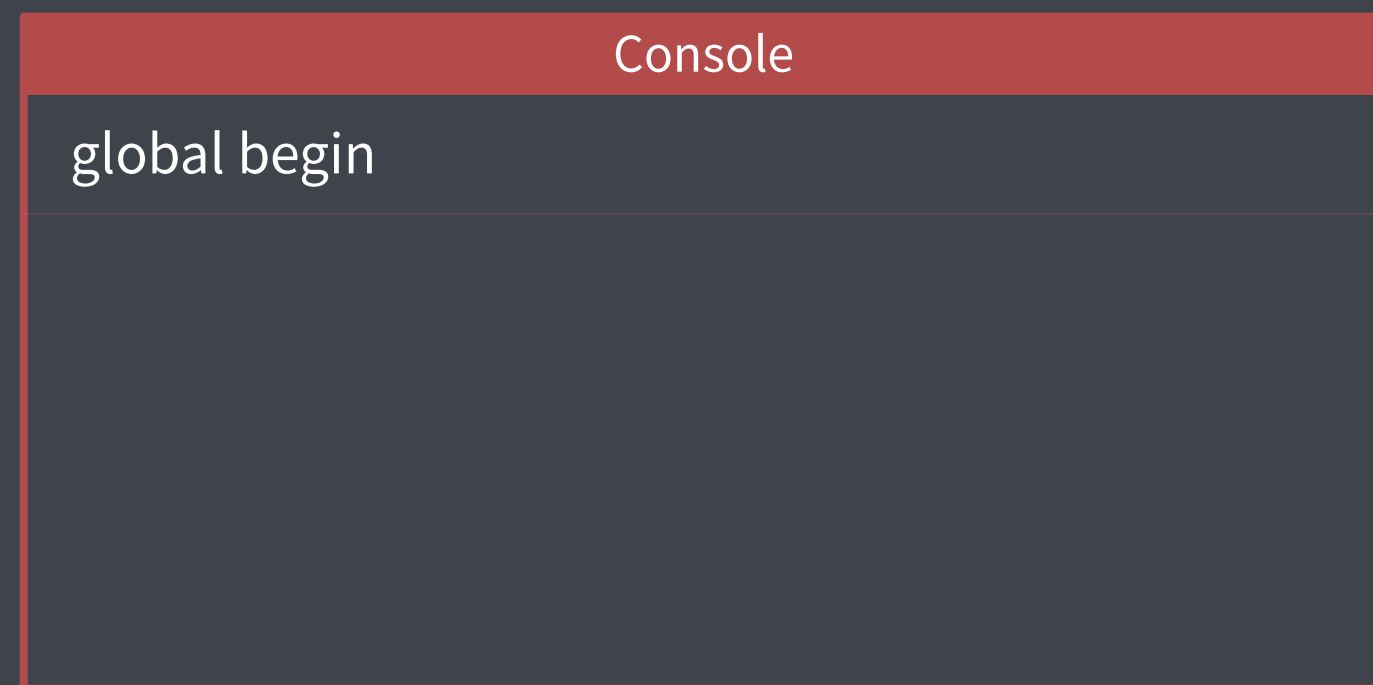
```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```



```
console.log('global end')
```

```
graph TD; subgraph Console; direction TB; L1[global begin]; L2[foo task]; end
```

The diagram illustrates a console output. It features a dark gray rectangular area representing the console, with a red header bar at the top containing the word "Console" in white. Inside the console area, there are two lines of white text: "global begin" on the first line and "foo task" on the second line. The lines are separated by a thin horizontal line.

Call stack

```
graph TD; A[console.log('foo task')] --> B[foo()]; B --> C[(anonymous)];
```

The diagram illustrates a call stack with three frames. The top frame contains the code `console.log('foo task')`. The middle frame contains the code `foo()`. The bottom frame contains the code `(anonymous)`. Arrows indicate the flow of execution from the top frame to the middle frame, and from the middle frame to the bottom frame.

```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

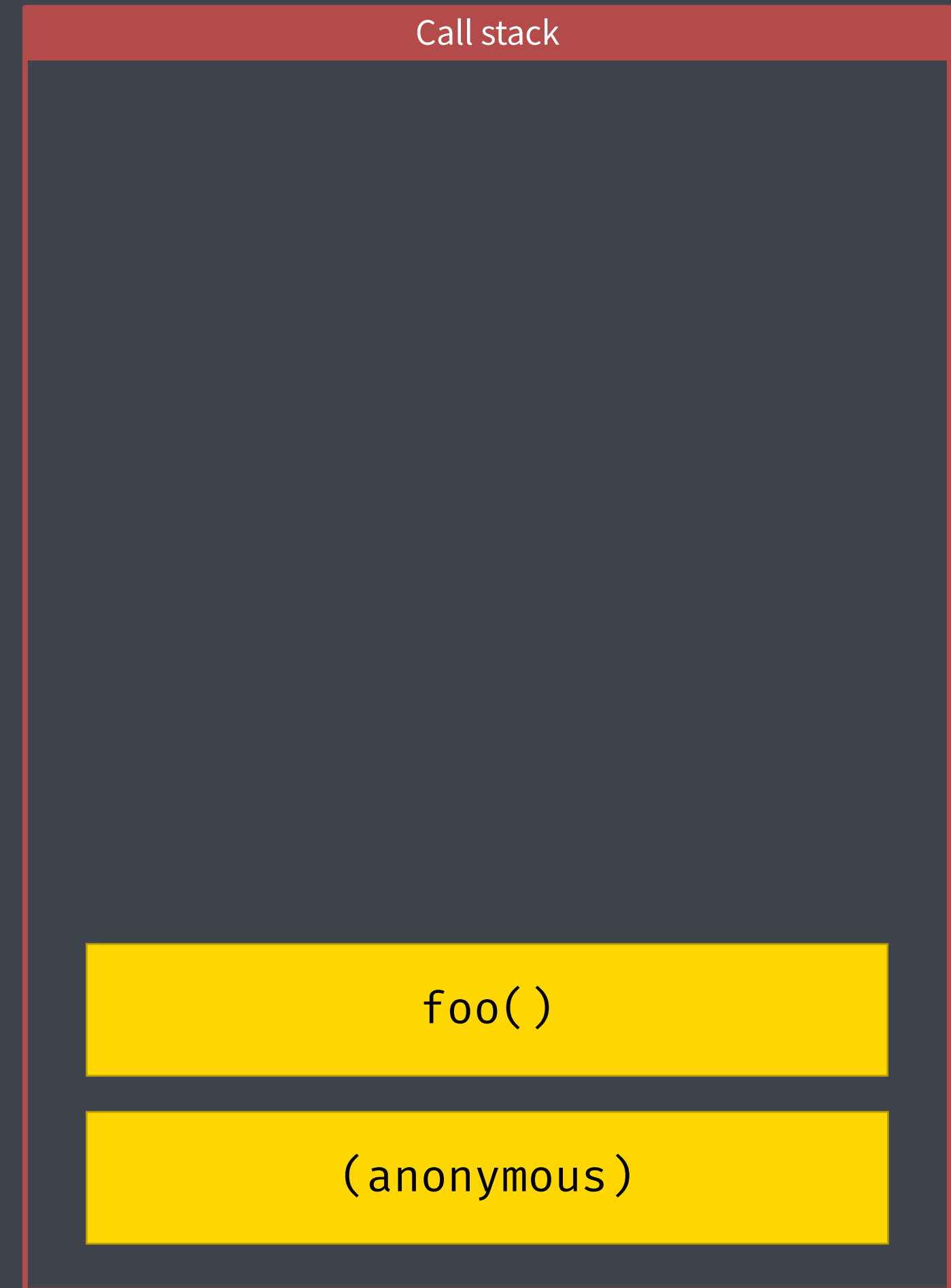
```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```



Console
global begin
foo task



```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

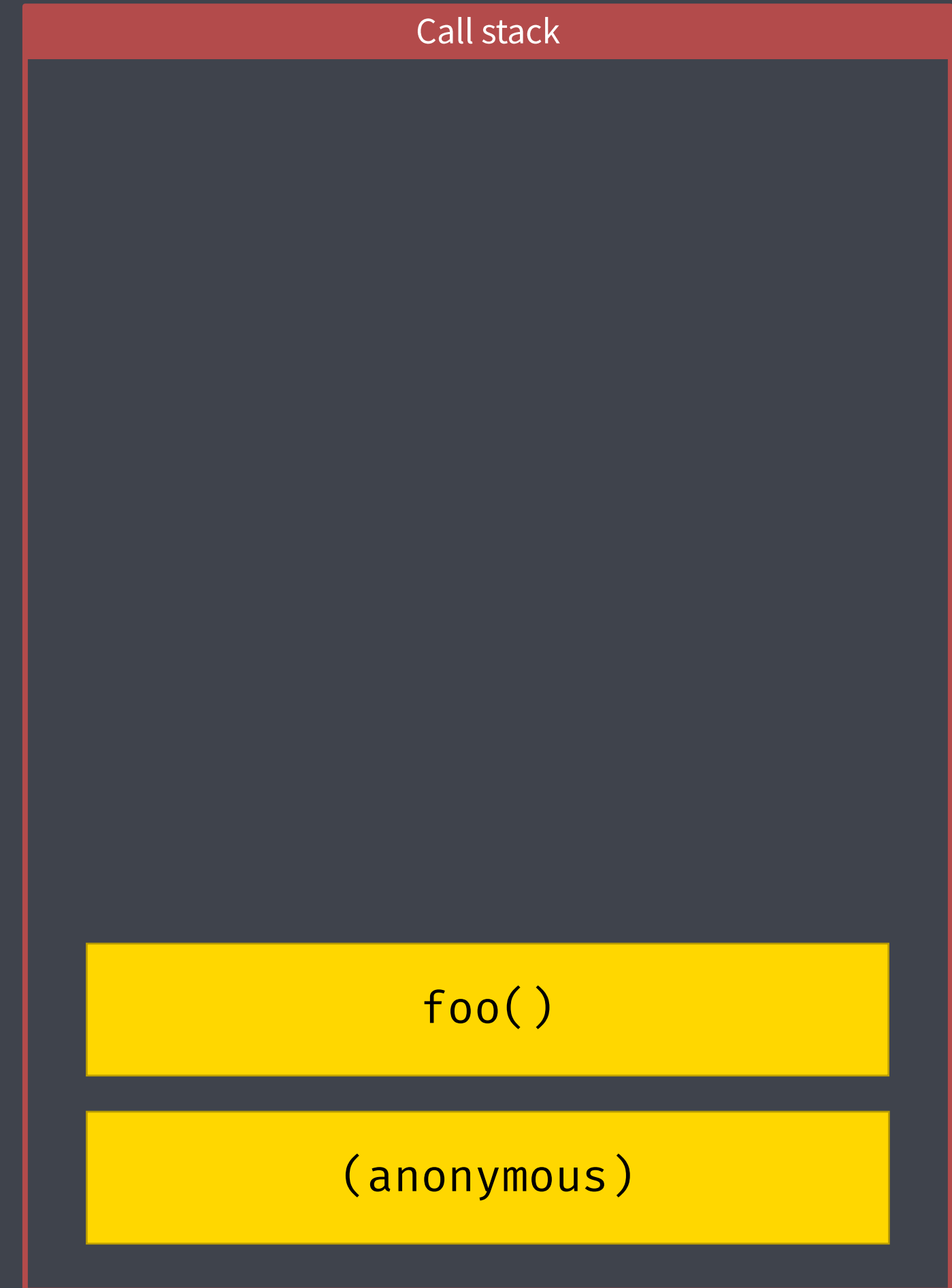
```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```



Console
global begin
foo task



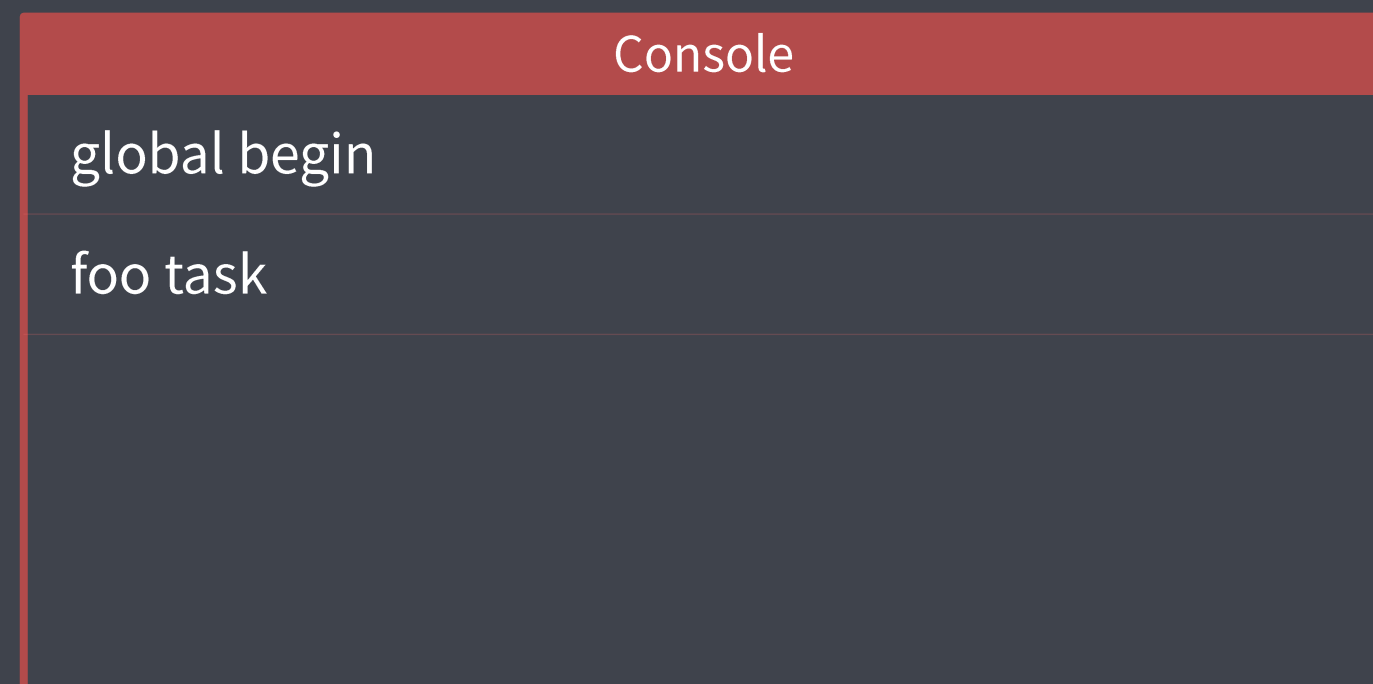
```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```



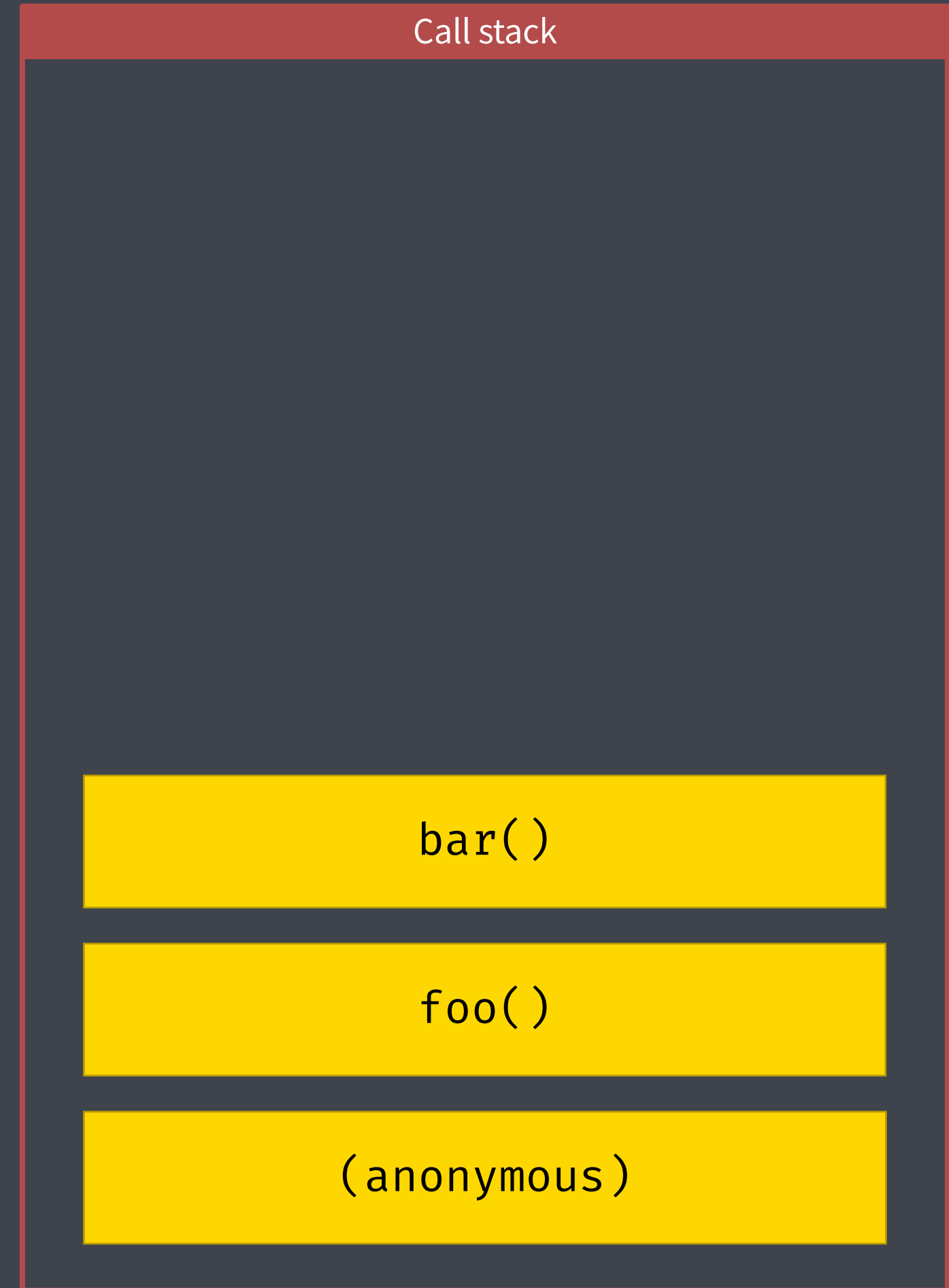
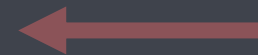
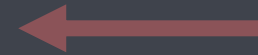
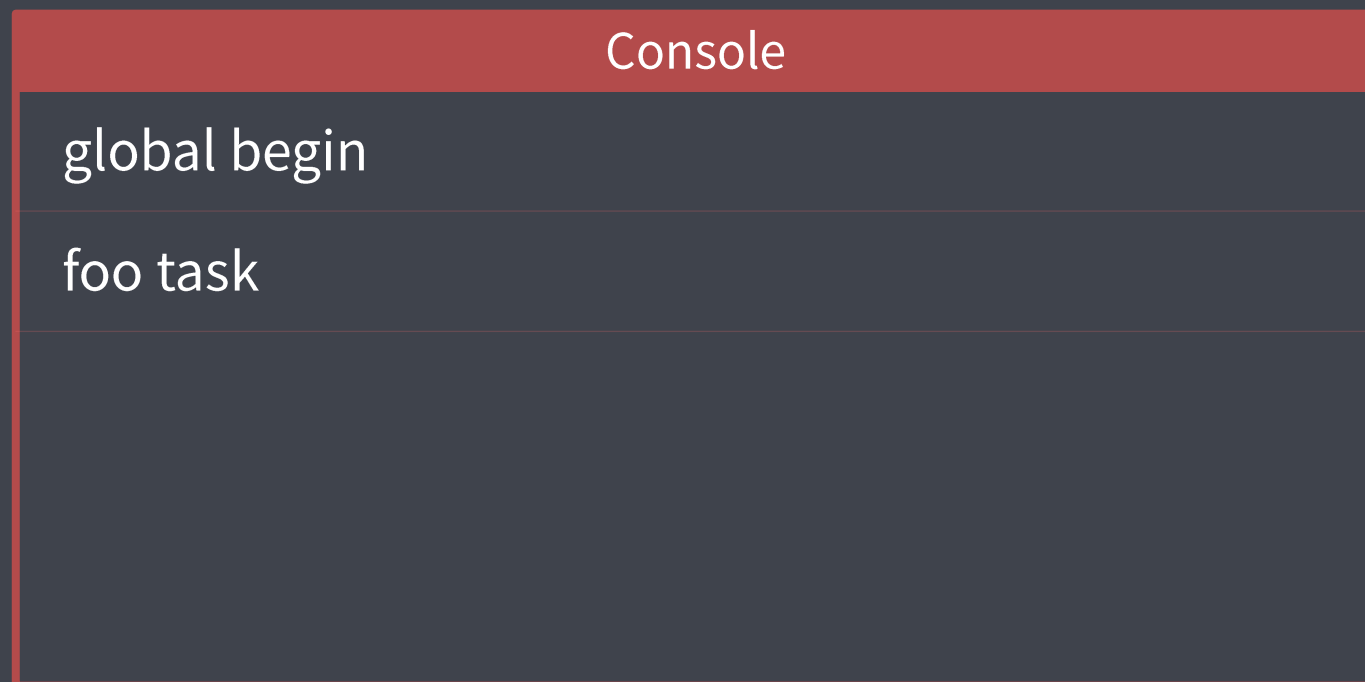
```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```





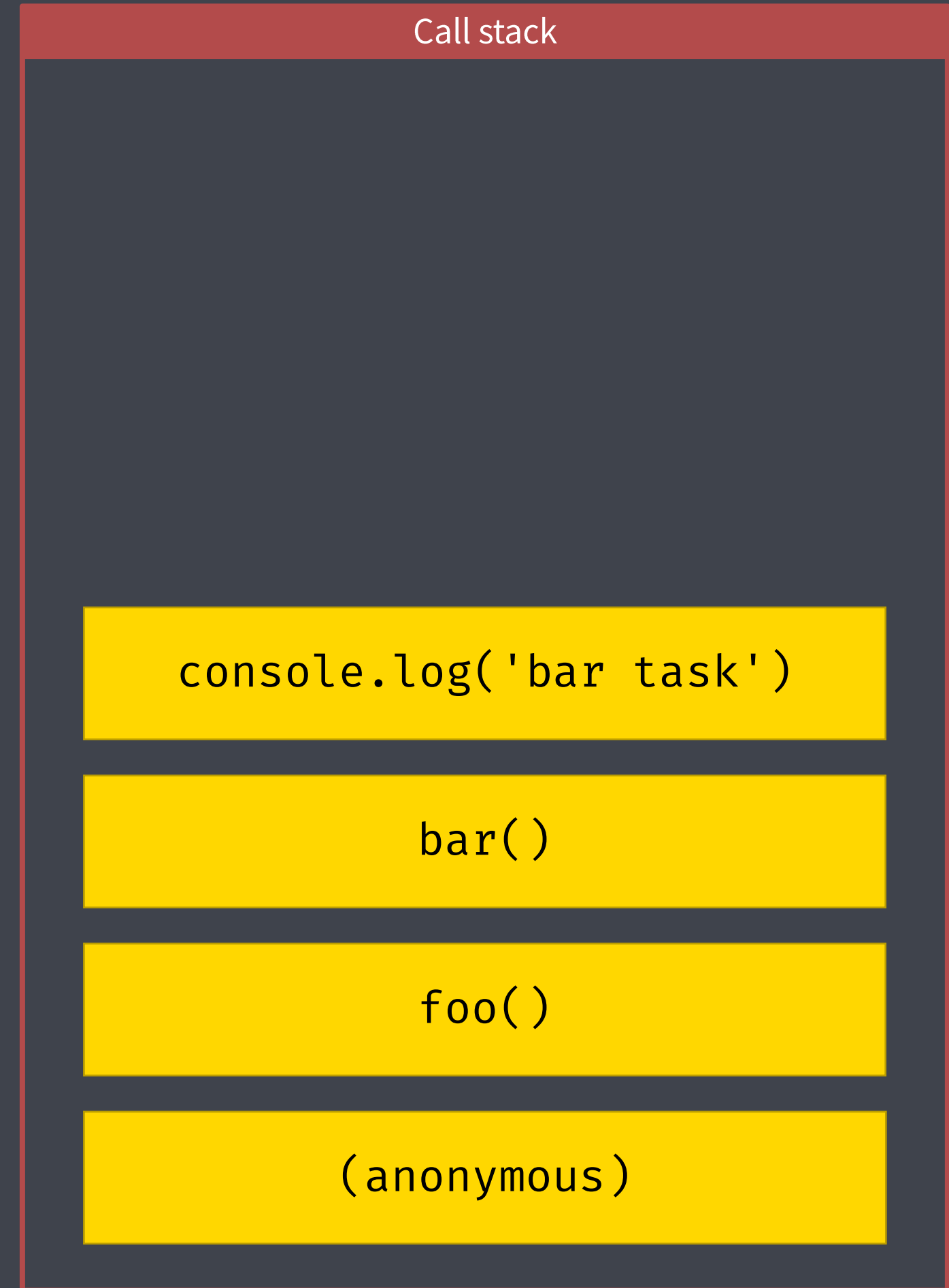
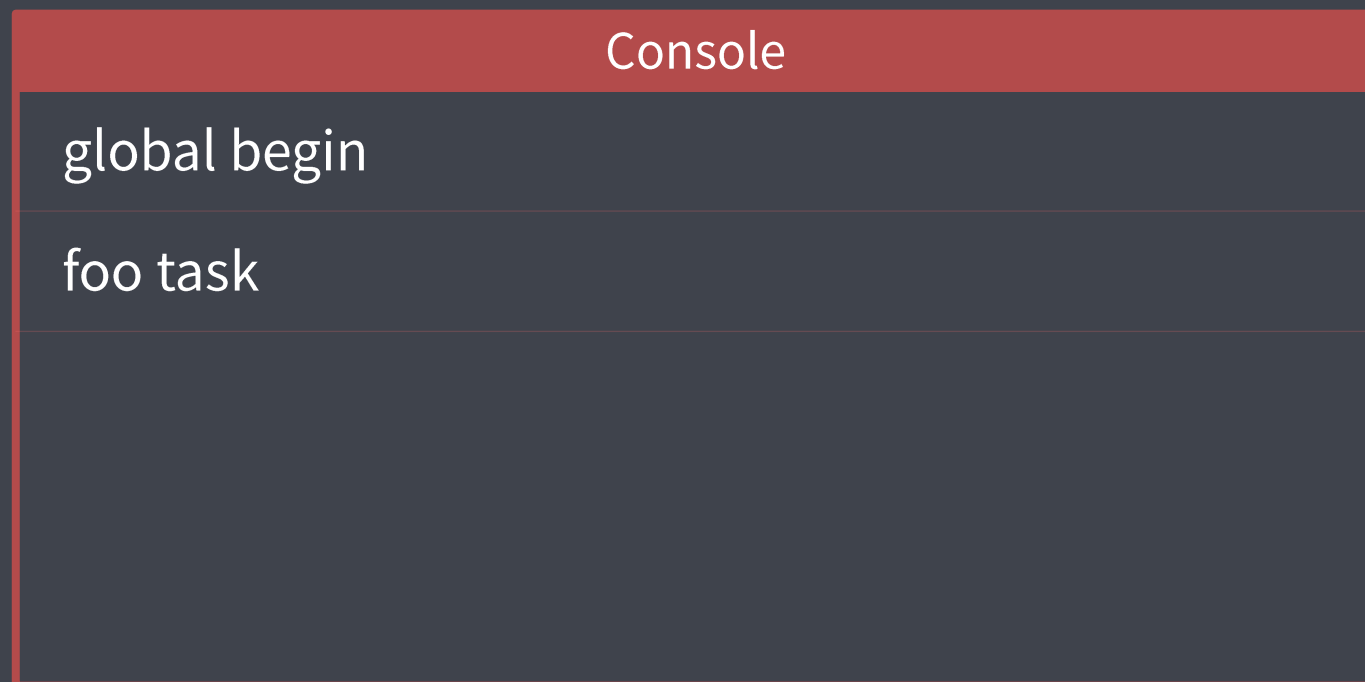
```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```



```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```



### Console

global begin

foo task

bar task

### Call stack

console.log('bar task')

bar()

foo()

(anonymous)

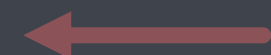
```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

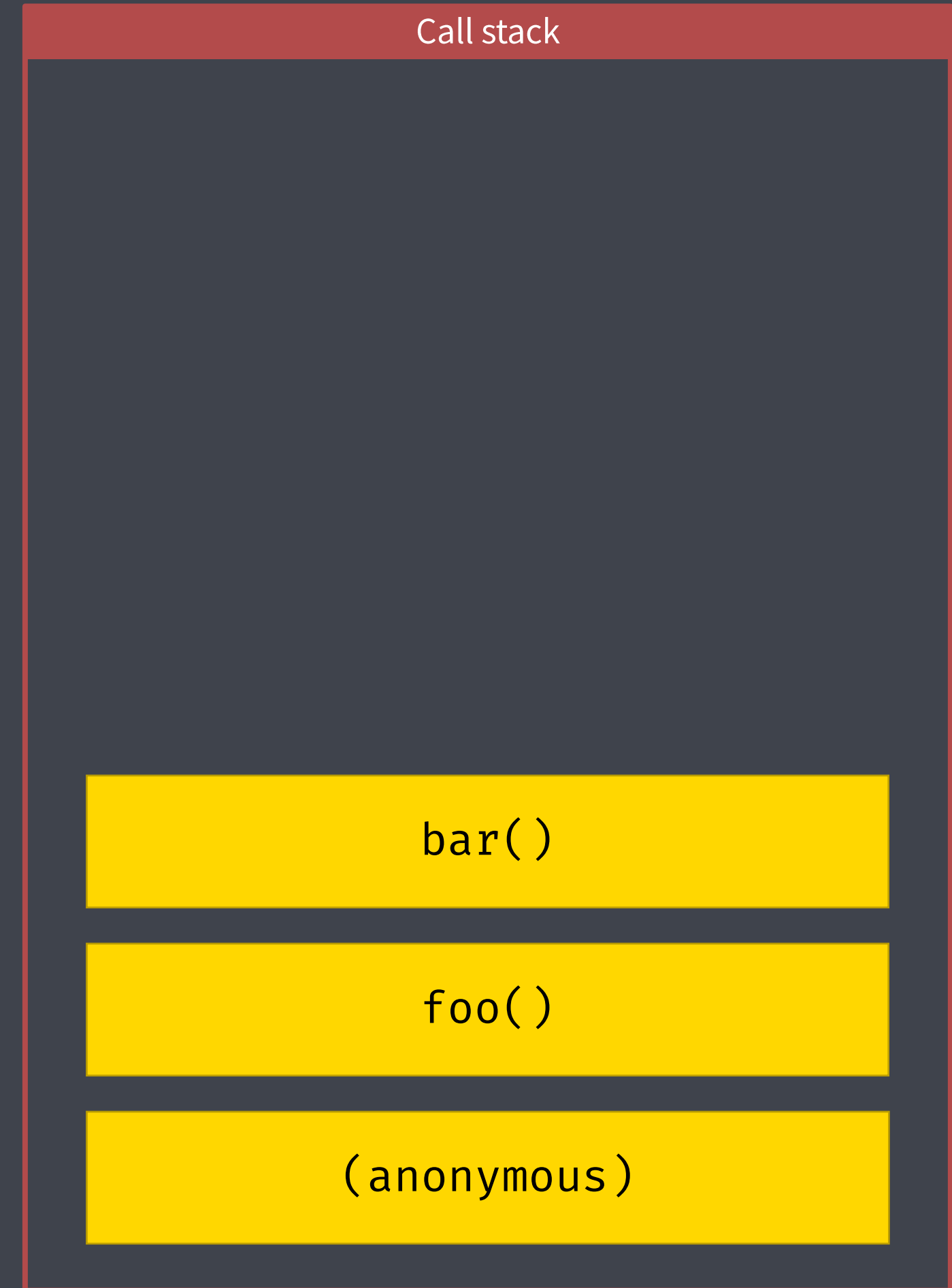
```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```



Console
global begin
foo task
bar task



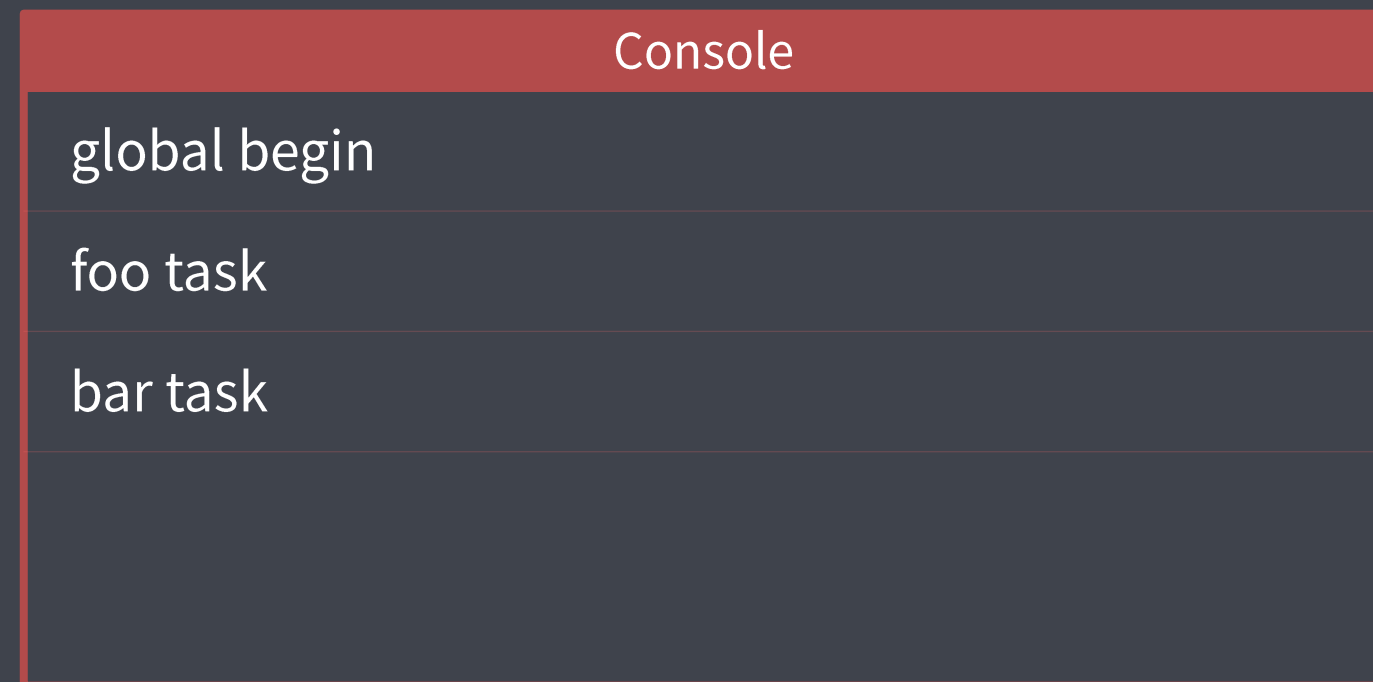
```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```



```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

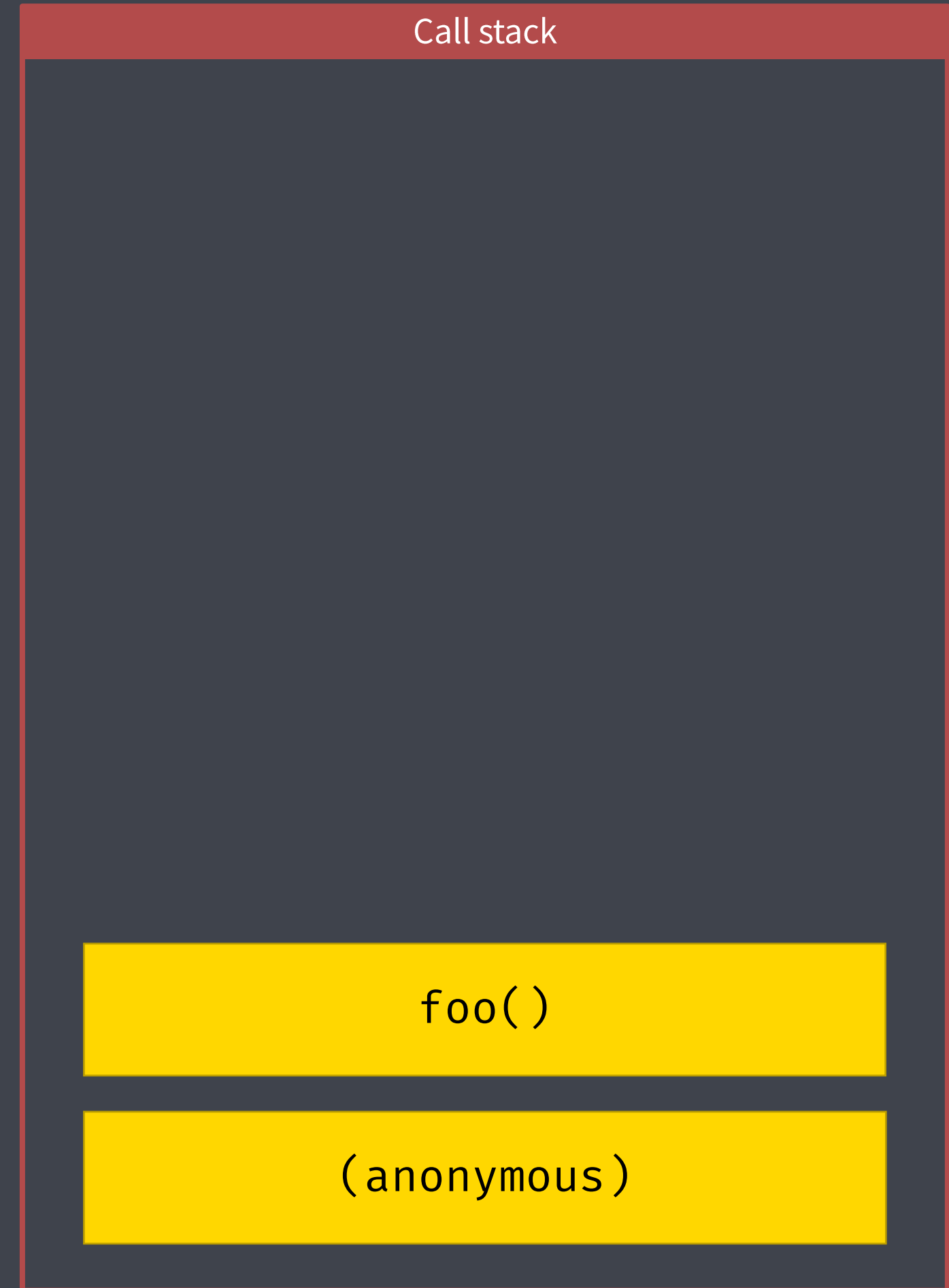
```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```



Console
global begin
foo task
bar task



```
console.log('global begin')
```

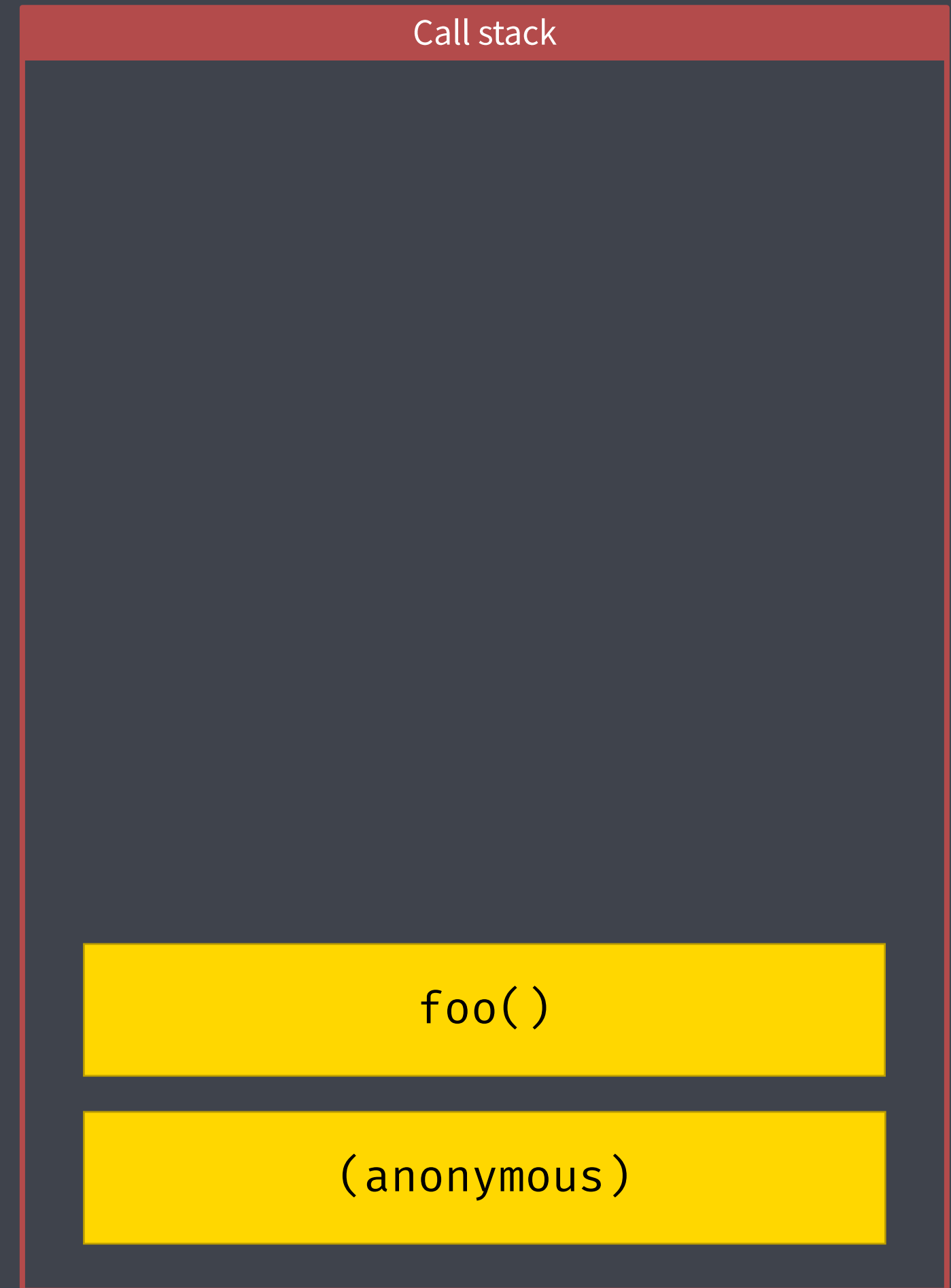
```
function bar () {  
  console.log('bar task')  
}
```

```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```

Console
global begin
foo task
bar task



```
console.log('global begin')
```

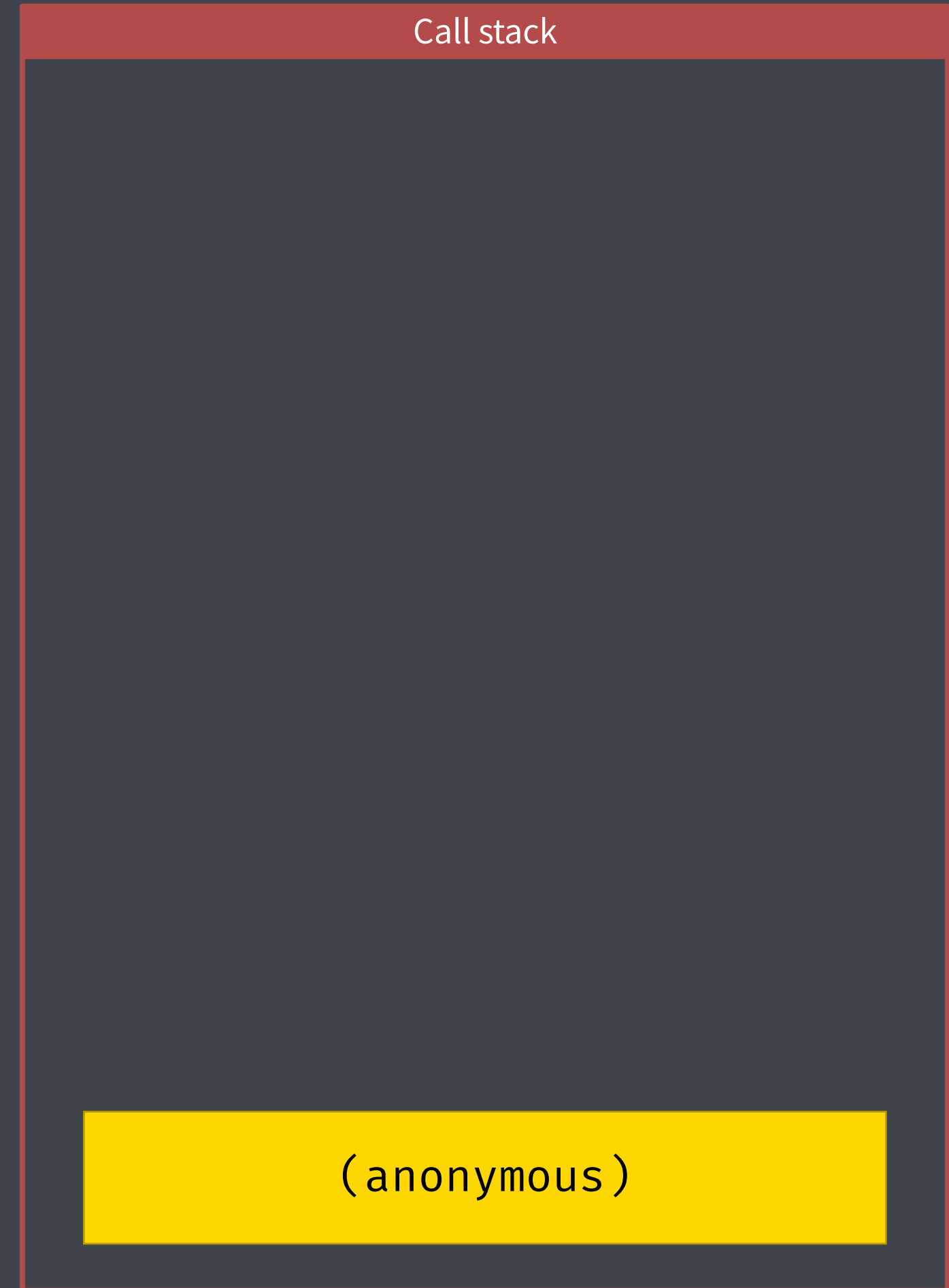
```
function bar () {  
  console.log('bar task')  
}
```

```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```

Console
global begin
foo task
bar task



```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

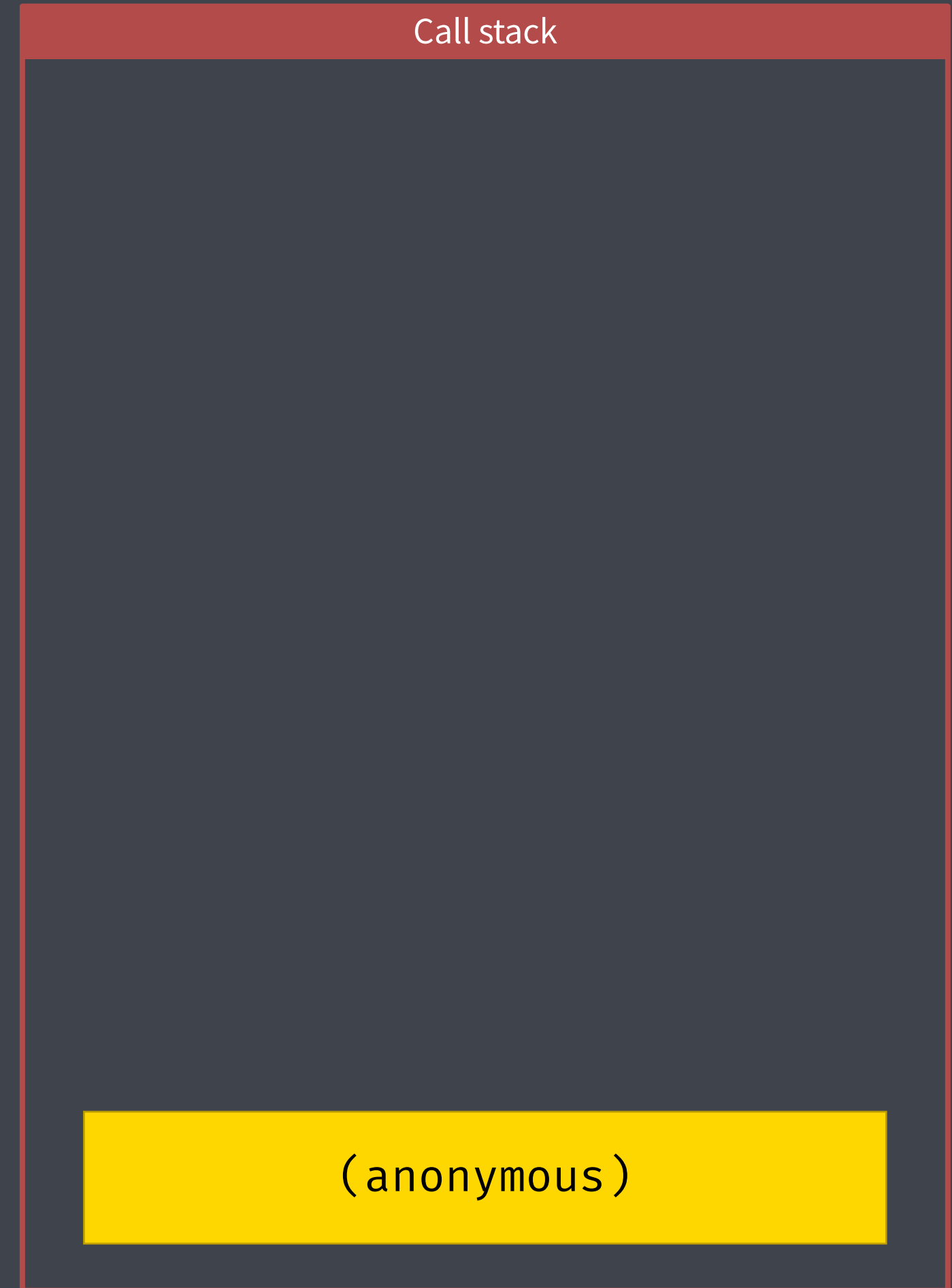
```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```



Console
global begin
foo task
bar task





```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

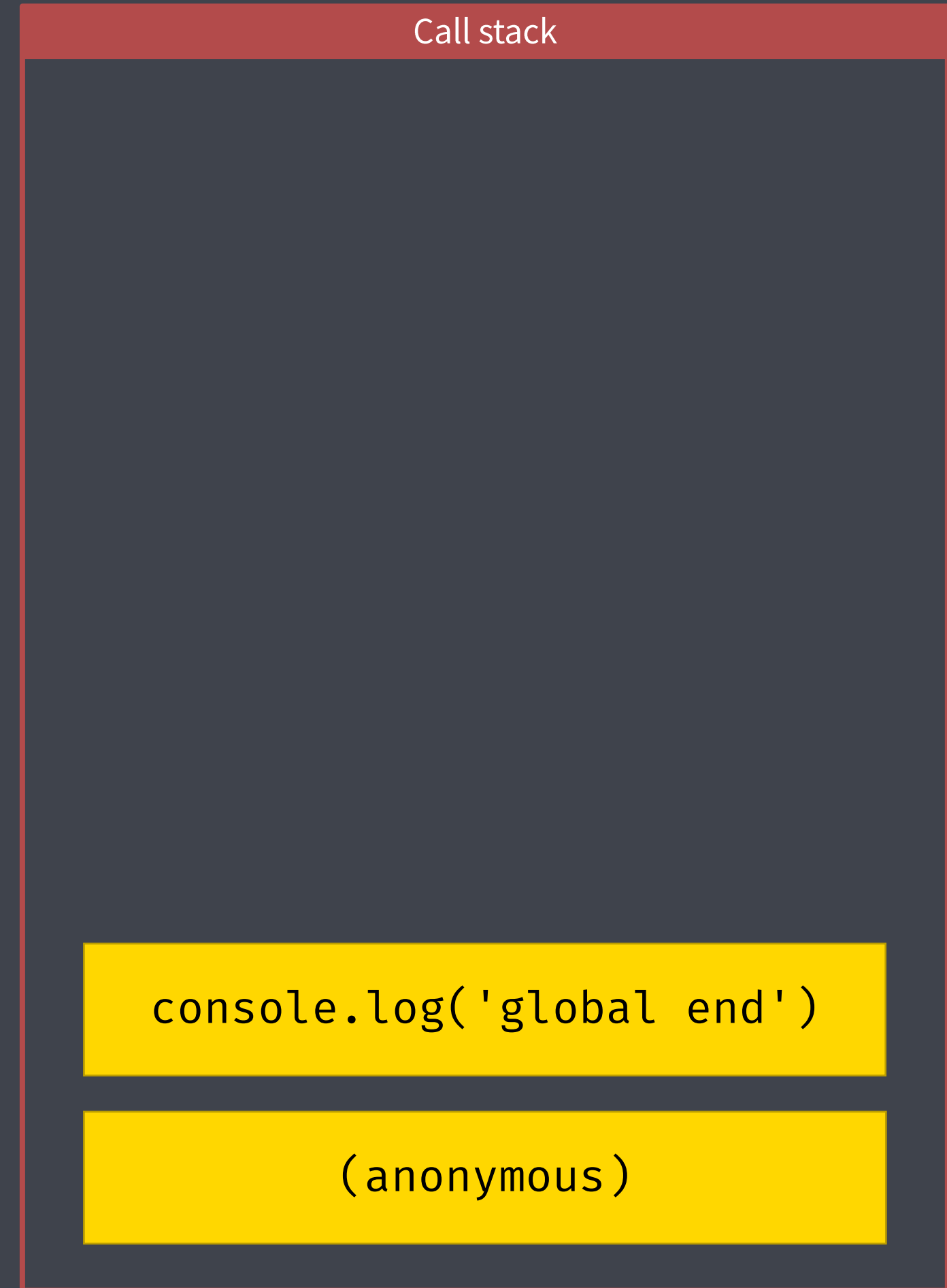
```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```



Console
global begin
foo task
bar task



```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

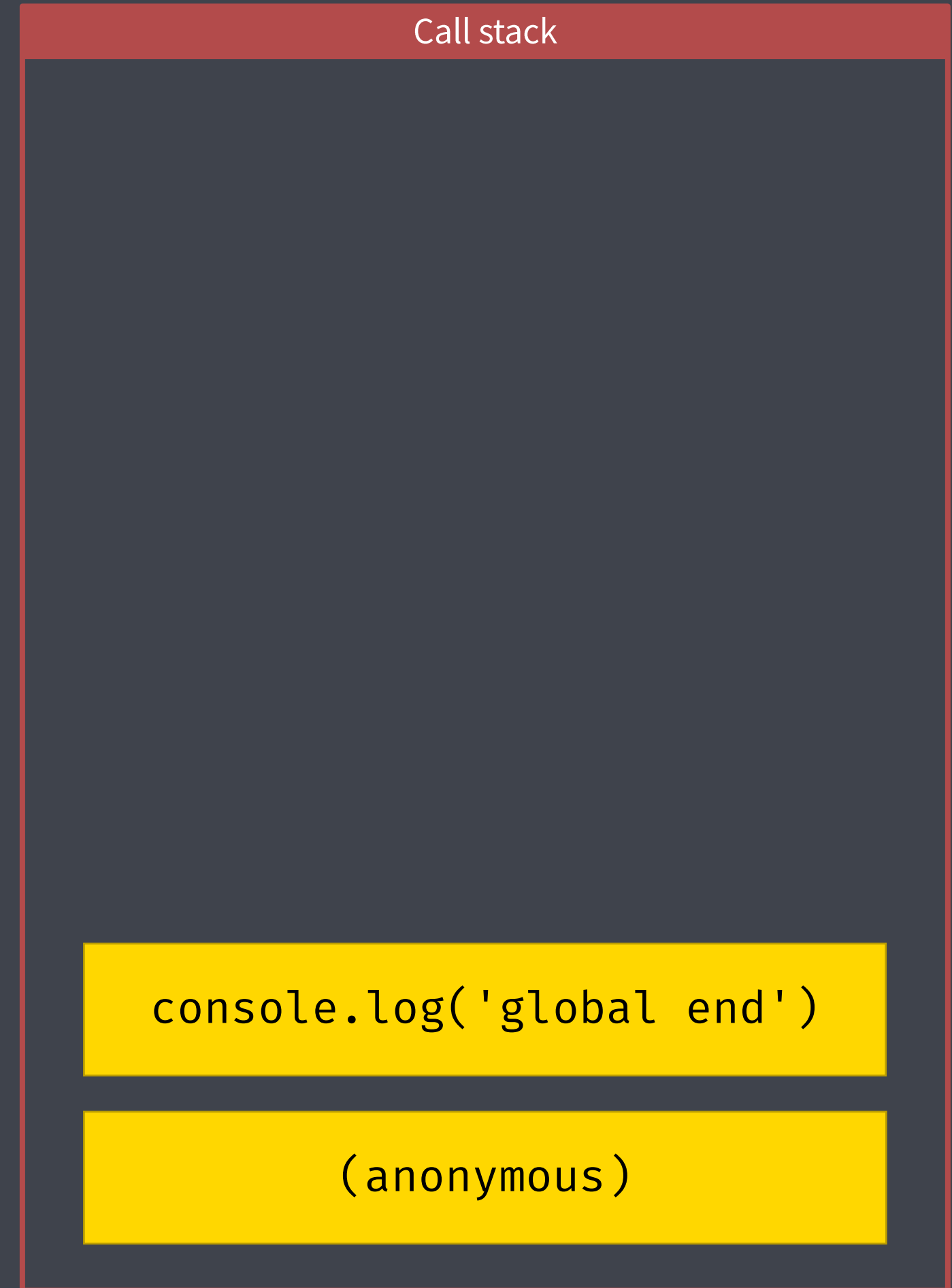
```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```



Console
global begin
foo task
bar task
global end



```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

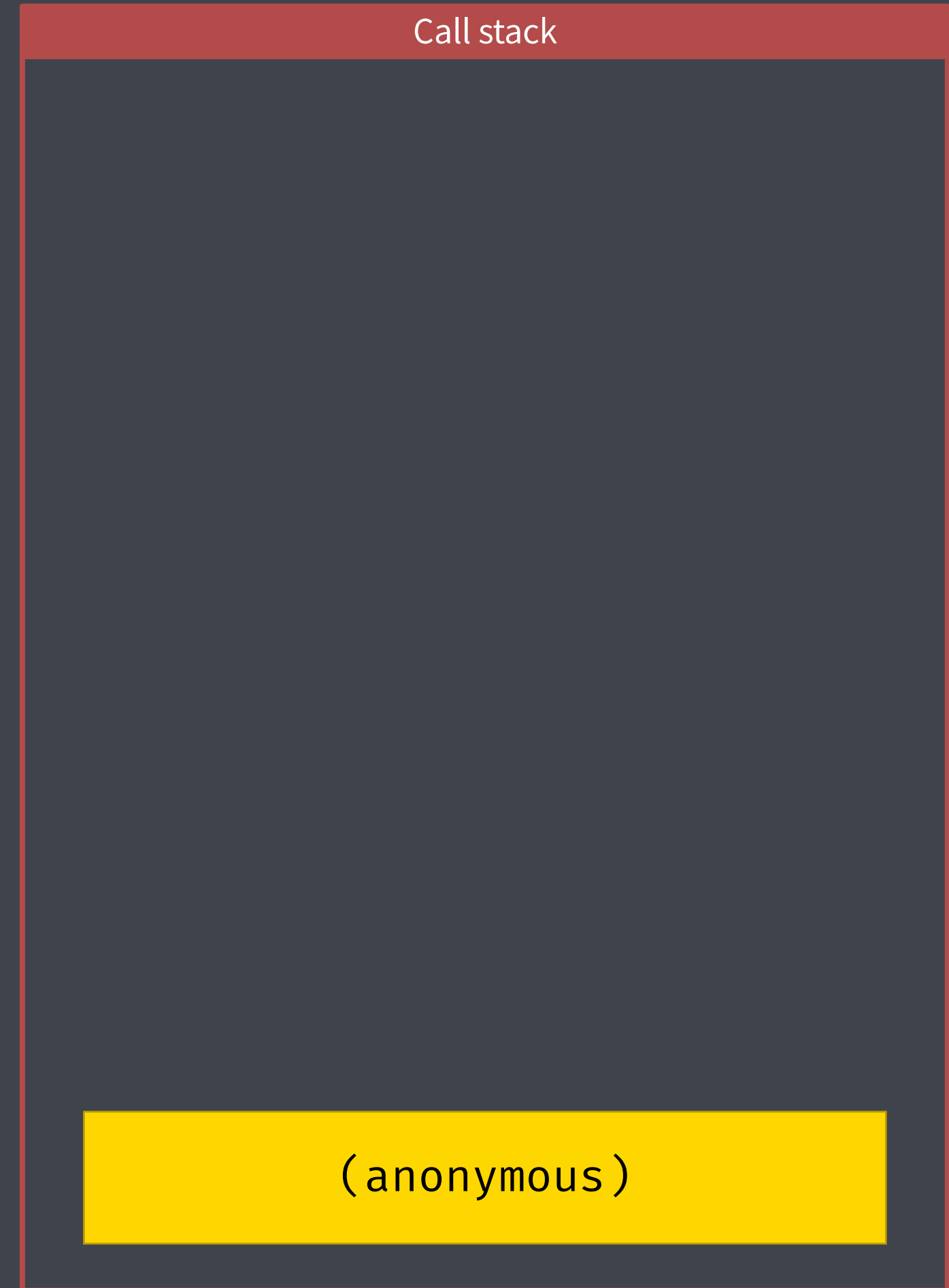
```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

```
foo()
```

```
console.log('global end')
```



Console
global begin
foo task
bar task
global end



```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

foo()

```
console.log('global end')
```

## Console

global begin

foo task

bar task

global end

## Call stack

(anonymous)

```
console.log('global begin')
```

```
function bar () {  
  console.log('bar task')  
}
```

```
function foo () {  
  console.log('foo task')  
  bar()  
}
```

foo()

```
console.log('global end')
```

## Console

global begin

foo task

bar task

global end

## Call stack

# 异步模式

Asynchronous

```
console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)

console.log('global end')
```

```

console.log('global begin')

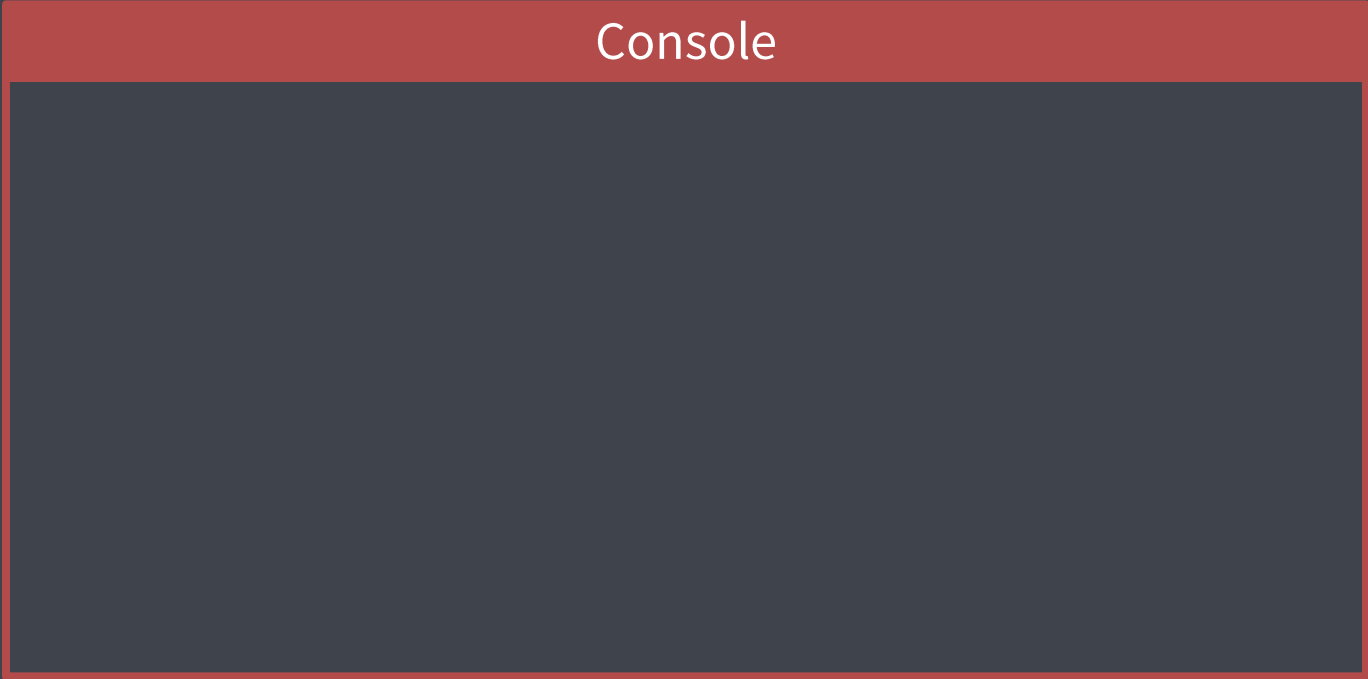
setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

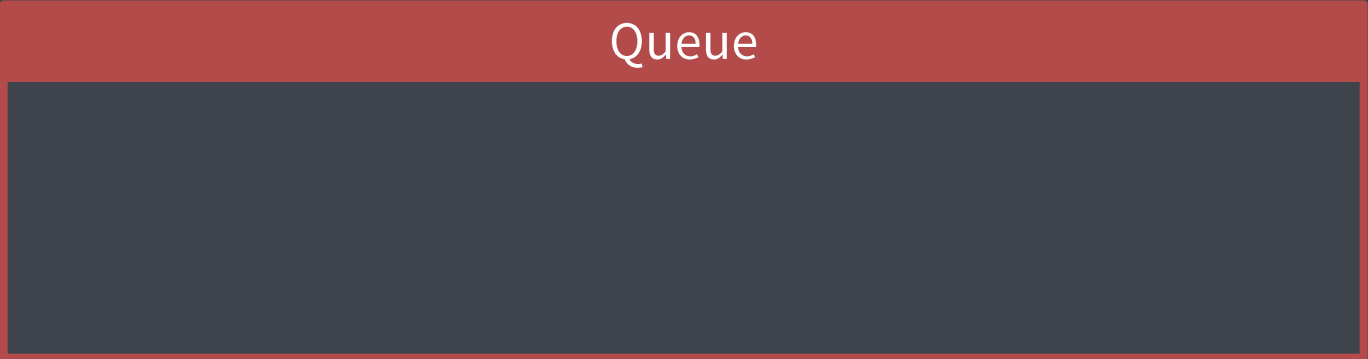
  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)

console.log('global end')

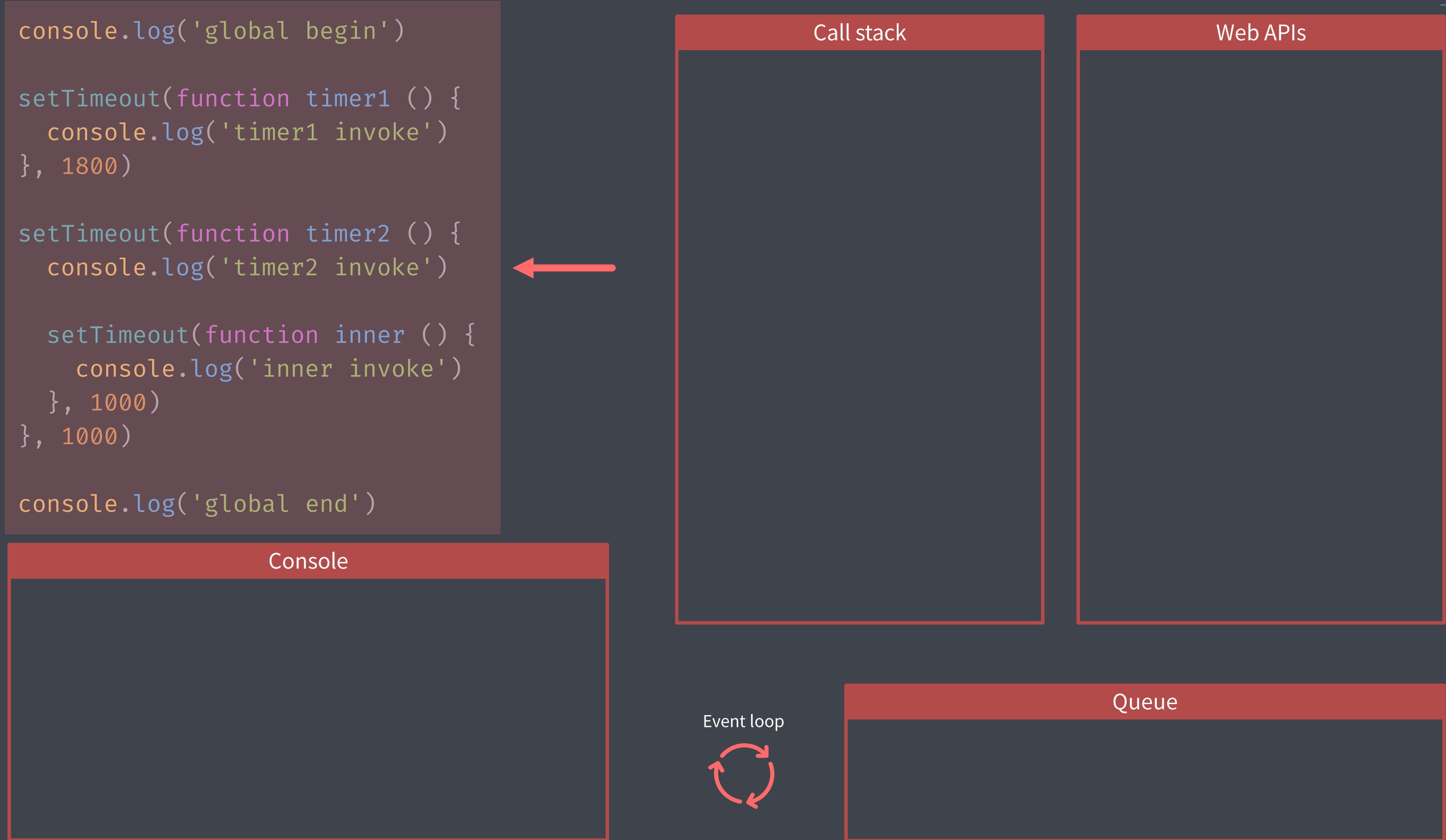
```

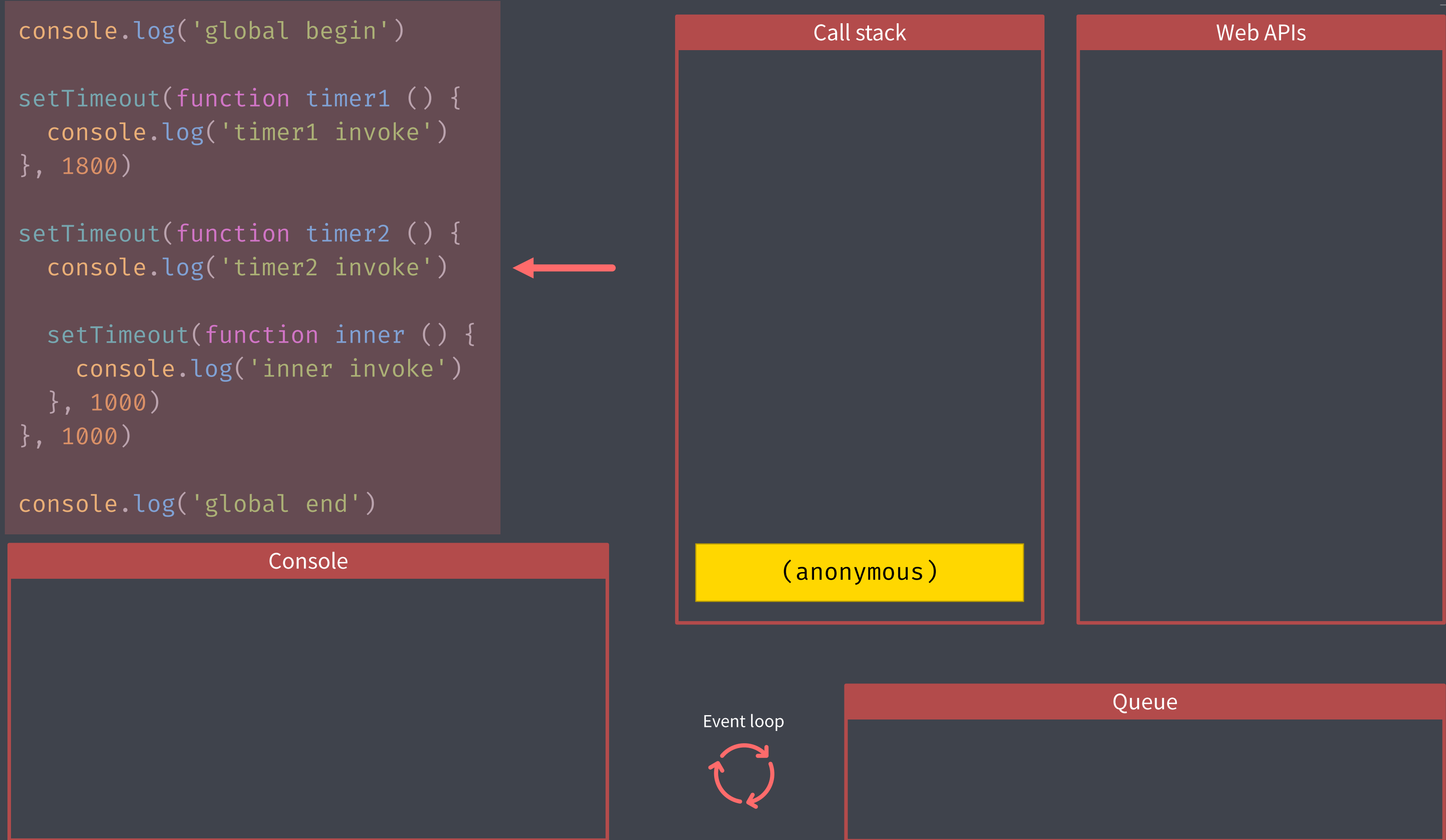


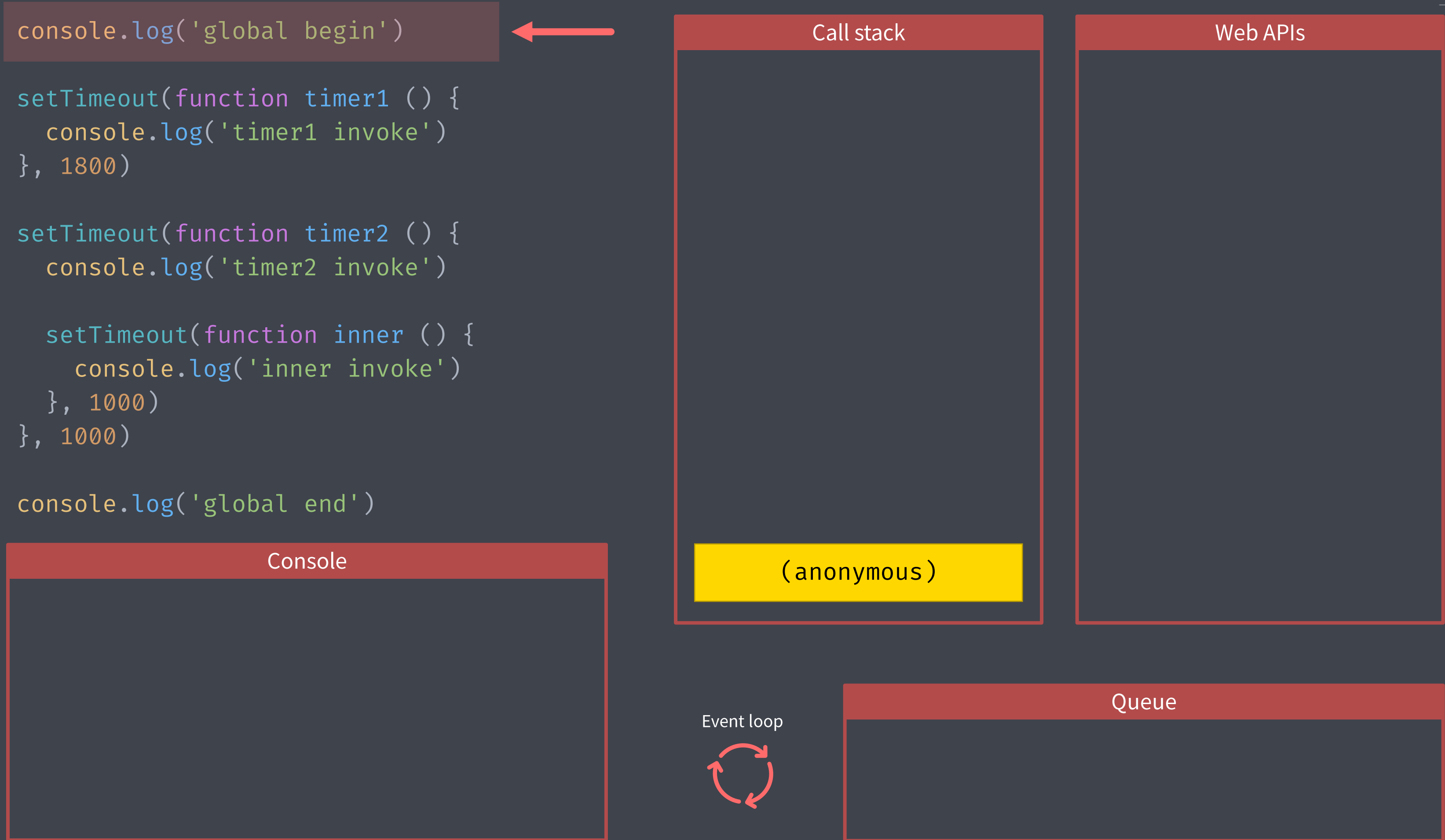
Event loop

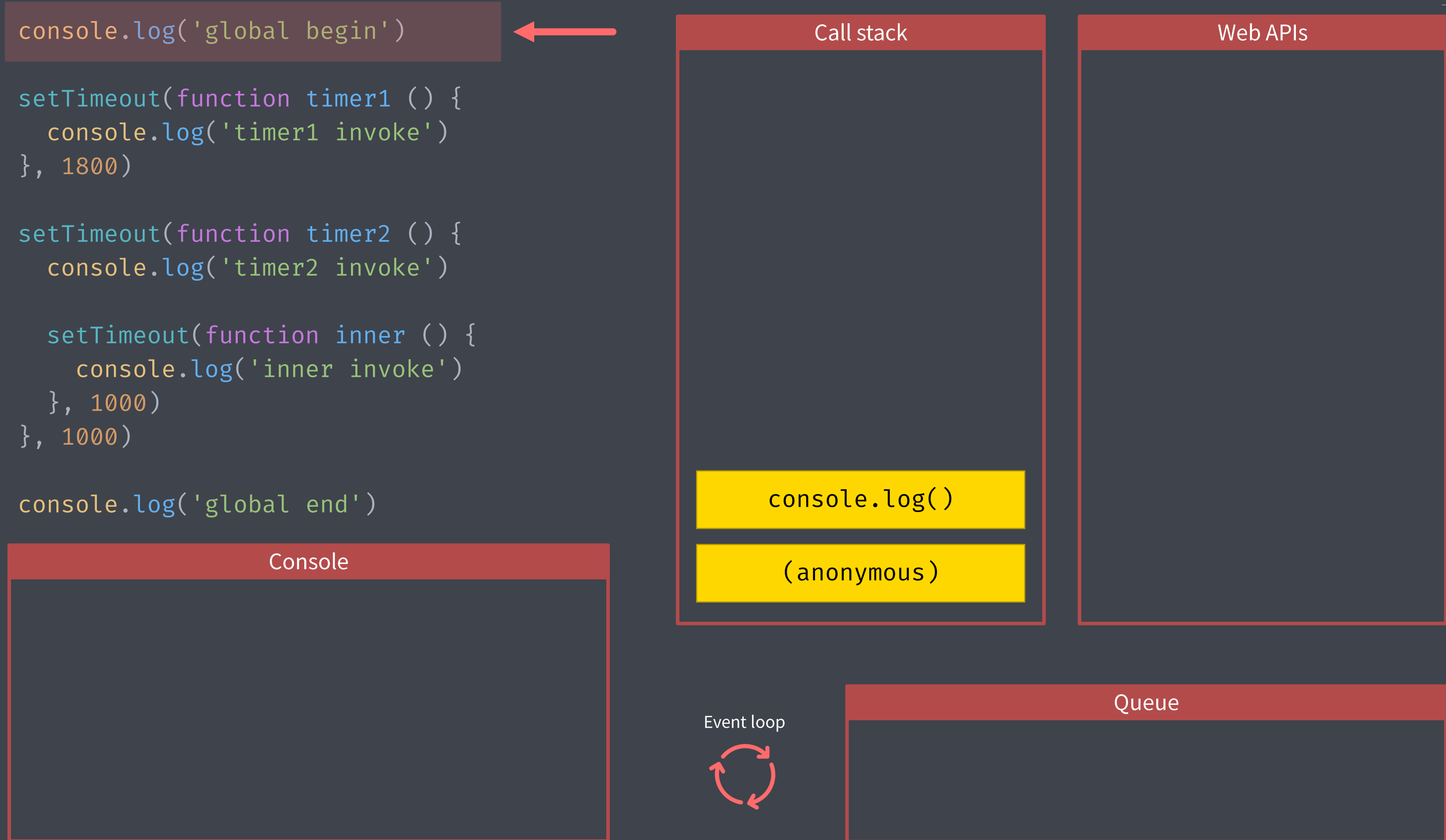


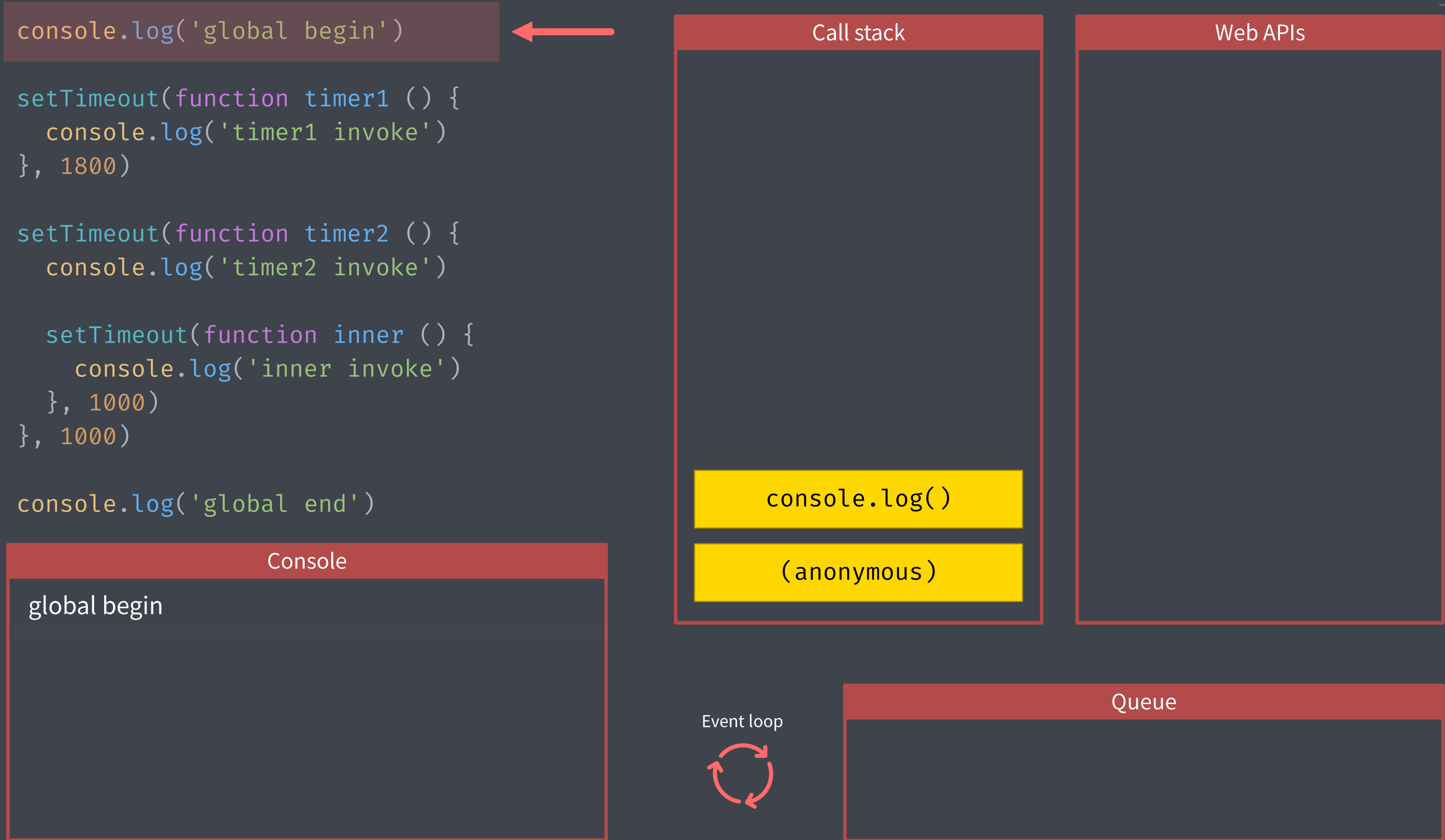


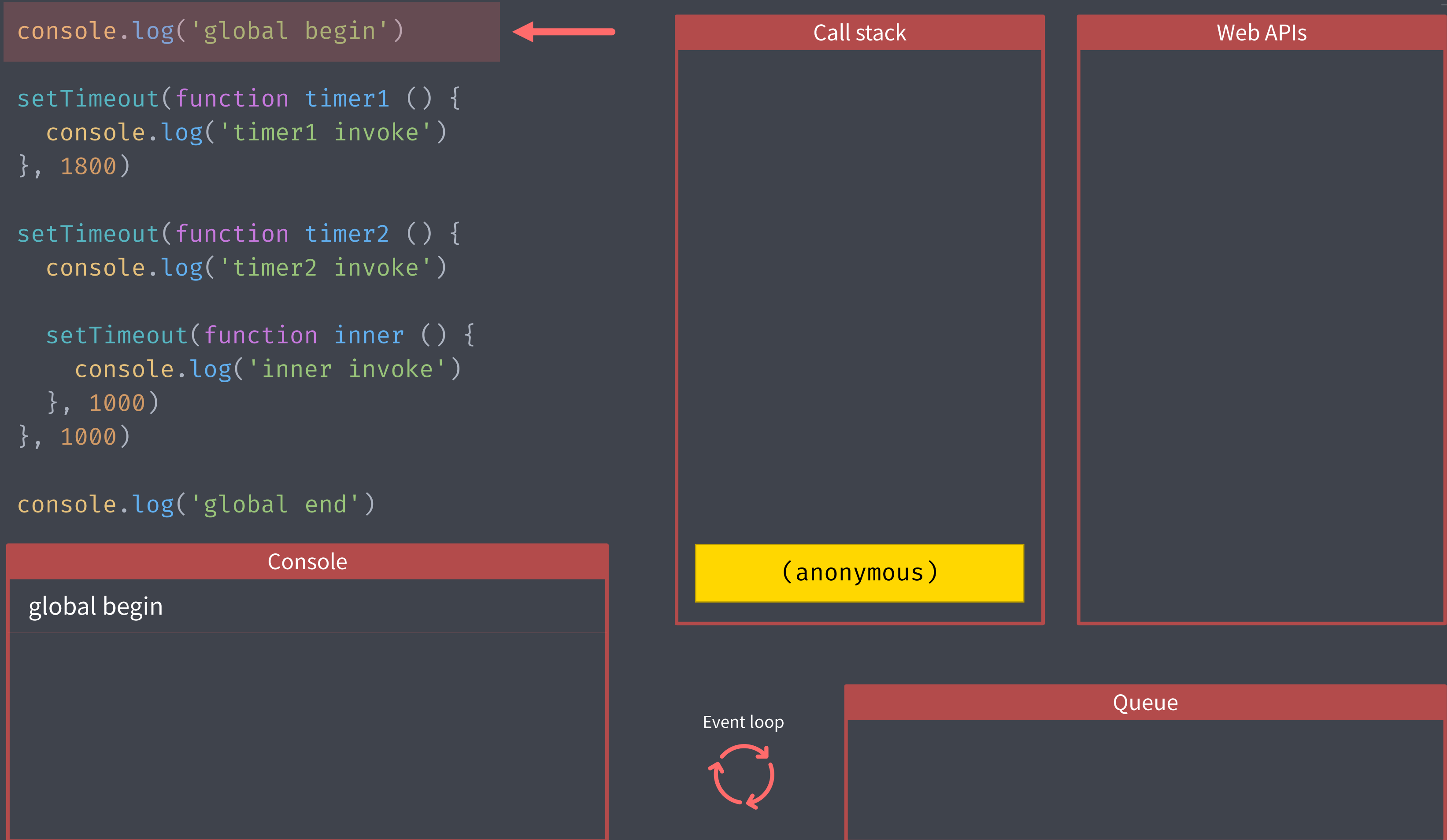












```
console.log('global begin')
```

```
setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)
```

```
setTimeout(function timer2 () {
  console.log('timer2 invoke')
```

```
  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)
```

```
console.log('global end')
```

Console

global begin

Call stack

(anonymous)

Web APIs

Event loop



Queue

```
console.log('global begin')
```

```
setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)
```

```
setTimeout(function timer2 () {
  console.log('timer2 invoke')
```

```
  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)
```

```
console.log('global end')
```

Console

global begin

Call stack

setTimeout(timer1)

(anonymous)

Web APIs

Event loop



Queue



```
console.log('global begin')
```

```
setTimeout(function timer1 () {  
  console.log('timer1 invoke')  
}, 1800)
```

```
setTimeout(function timer2 () {  
  console.log('timer2 invoke')
```

```
  setTimeout(function inner () {  
    console.log('inner invoke')  
  }, 1000)  
}, 1000)
```

```
console.log('global end')
```

### Console

global begin

### Call stack

setTimeout(timer1)

(anonymous)

### Web APIs

⌚ 1.8s timer1

Event loop



### Queue

```
console.log('global begin')
```

```
setTimeout(function timer1 () {  
  console.log('timer1 invoke')  
}, 1800)
```

```
setTimeout(function timer2 () {  
  console.log('timer2 invoke')
```

```
  setTimeout(function inner () {  
    console.log('inner invoke')  
  }, 1000)  
}, 1000)
```

```
console.log('global end')
```

Console

global begin

Call stack

(anonymous)

Web APIs

⌚ 1.8s timer1

Event loop



Queue

```
console.log('global begin')
```

```
setTimeout(function timer1 () {  
  console.log('timer1 invoke')  
}, 1800)
```

```
setTimeout(function timer2 () {  
  console.log('timer2 invoke')  
  
  setTimeout(function inner () {  
    console.log('inner invoke')  
  }, 1000)  
}, 1000)
```

```
console.log('global end')
```

#### Console

global begin

#### Call stack

(anonymous)

#### Web APIs

⌚ 1.8s timer1

Event loop



#### Queue

```
console.log('global begin')
```

```
setTimeout(function timer1 () {  
  console.log('timer1 invoke')  
}, 1800)
```

```
setTimeout(function timer2 () {  
  console.log('timer2 invoke')  
  
  setTimeout(function inner () {  
    console.log('inner invoke')  
  }, 1000)  
}, 1000)
```

```
console.log('global end')
```

#### Console

global begin

#### Call stack

setTimeout(timer2)

(anonymous)

#### Web APIs

⌚ 1.8s timer1

Event loop



#### Queue

```
console.log('global begin')
```

```
setTimeout(function timer1 () {  
  console.log('timer1 invoke')  
}, 1800)
```

```
setTimeout(function timer2 () {  
  console.log('timer2 invoke')  
  
  setTimeout(function inner () {  
    console.log('inner invoke')  
  }, 1000)  
}, 1000)
```

```
console.log('global end')
```

### Console

global begin

### Call stack

setTimeout(timer2)

(anonymous)

### Web APIs

⌚ 1.8s timer1

⌚ 1s timer2

Event loop



### Queue

```
console.log('global begin')
```

```
setTimeout(function timer1 () {  
  console.log('timer1 invoke')  
}, 1800)
```

```
setTimeout(function timer2 () {  
  console.log('timer2 invoke')  
  
  setTimeout(function inner () {  
    console.log('inner invoke')  
  }, 1000)  
}, 1000)
```

```
console.log('global end')
```

#### Console

global begin

#### Call stack

(anonymous)

#### Web APIs

⌚ 1.8s timer1

⌚ 1s timer2

Event loop



#### Queue

```
console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)
```

```
console.log('global end')
```

### Console

global begin

### Call stack

(anonymous)

### Web APIs

⌚ 1.8s timer1

⌚ 1s timer2

Event loop



### Queue

```
console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)
```

```
console.log('global end')
```

Console

global begin

Call stack

console.log()

(anonymous)

Web APIs

⌚ 1.8s timer1

⌚ 1s timer2

Event loop



Queue



```
console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)
```

```
console.log('global end')
```

#### Console

global begin

global end

#### Call stack

console.log()

(anonymous)

#### Web APIs

⌚ 1.8s timer1

⌚ 1s timer2

Event loop



#### Queue

```
console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)
```

```
console.log('global end')
```

#### Console

global begin

global end

#### Call stack

(anonymous)

#### Web APIs

⌚ 1.8s timer1

⌚ 1s timer2

Event loop



#### Queue

```
console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)

console.log('global end')
```

#### Console

global begin

global end

#### Call stack

(anonymous)

#### Web APIs

⌚ 1.8s timer1

⌚ 1s timer2

Event loop



#### Queue

```
console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)

console.log('global end')
```

Console

global begin

global end

Call stack

Web APIs

⌚ 1.8s timer1

⌚ 1s timer2

Event loop



Queue

```
console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)

console.log('global end')
```

#### Console

global begin

global end

#### Call stack

#### Web APIs

⌚ 1.8s timer1

⌚ 1s timer2

#### Event loop



#### Queue

timer2()

timer1()

```
console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)

console.log('global end')
```

Console

global begin

global end

Call stack

Web APIs

Event loop



Queue

timer2()

timer1()

```

console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)

console.log('global end')

```

Console

global begin

global end

Call stack

timer2()

Web APIs

Event loop



Queue

timer1()

```
console.log('global begin')
```

```
setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)
```

```
setTimeout(function timer2 () {
  console.log('timer2 invoke')

```

```
  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)
```

```
console.log('global end')
```

#### Console

global begin

global end

#### Call stack

timer2()

#### Web APIs

Event loop



#### Queue

timer1()



```
console.log('global begin')
```

```
setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)
```

```
setTimeout(function timer2 () {
  console.log('timer2 invoke')
  // ←
```

```
  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)
```

```
console.log('global end')
```

### Console

global begin

global end

### Call stack

console.log()

timer2()

### Web APIs

Event loop



### Queue

timer1()

```
console.log('global begin')
```

```
setTimeout(function timer1 () {  
  console.log('timer1 invoke')  
}, 1800)
```

```
setTimeout(function timer2 () {  
  console.log('timer2 invoke')  
}, 1000)
```

```
  setTimeout(function inner () {  
    console.log('inner invoke')  
  }, 1000)  
}, 1000)
```

```
console.log('global end')
```

#### Console

global begin

global end

timer2 invoke

#### Call stack

console.log()

timer2()

#### Web APIs

Event loop



#### Queue

timer1()

```
console.log('global begin')
```

```
setTimeout(function timer1 () {  
  console.log('timer1 invoke')  
}, 1800)
```

```
setTimeout(function timer2 () {  
  console.log('timer2 invoke')  
}, 1000)
```

```
  setTimeout(function inner () {  
    console.log('inner invoke')  
  }, 1000)  
}, 1000)
```

```
console.log('global end')
```

#### Console

global begin

global end

timer2 invoke

#### Call stack

timer2()

#### Web APIs

Event loop



#### Queue

timer1()

```
console.log('global begin')
```

```
setTimeout(function timer1 () {  
  console.log('timer1 invoke')  
}, 1800)
```

```
setTimeout(function timer2 () {  
  console.log('timer2 invoke')
```

```
    setTimeout(function inner () {  
      console.log('inner invoke')  
    }, 1000)  
  }, 1000)
```

```
console.log('global end')
```

#### Console

global begin

global end

timer2 invoke

#### Call stack

timer2()

#### Web APIs

Event loop



#### Queue

timer1()

```
console.log('global begin')
```

```
setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)
```

```
setTimeout(function timer2 () {
  console.log('timer2 invoke')
```

```
  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)
```

```
console.log('global end')
```

### Console

global begin

global end

timer2 invoke

### Call stack

setTimeout(inner)

timer2()

### Web APIs

Event loop



### Queue

timer1()

```
console.log('global begin')
```

```
setTimeout(function timer1 () {  
  console.log('timer1 invoke')  
}, 1800)
```

```
setTimeout(function timer2 () {  
  console.log('timer2 invoke')
```

```
    setTimeout(function inner () {  
      console.log('inner invoke')  
    }, 1000)  
  }, 1000)
```

```
console.log('global end')
```

#### Console

global begin

global end

timer2 invoke

#### Call stack

setTimeout(inner)

timer2()

#### Web APIs

⌚ 1s inner

#### Event loop



#### Queue

timer1()

```
console.log('global begin')
```

```
setTimeout(function timer1 () {  
  console.log('timer1 invoke')  
}, 1800)
```

```
setTimeout(function timer2 () {  
  console.log('timer2 invoke')
```

```
    setTimeout(function inner () {  
      console.log('inner invoke')  
    }, 1000)  
  }, 1000)
```

```
console.log('global end')
```

#### Console

global begin

global end

timer2 invoke

#### Call stack

timer2()

#### Web APIs

⌚ 1s inner

#### Event loop



#### Queue

timer1()

```
console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

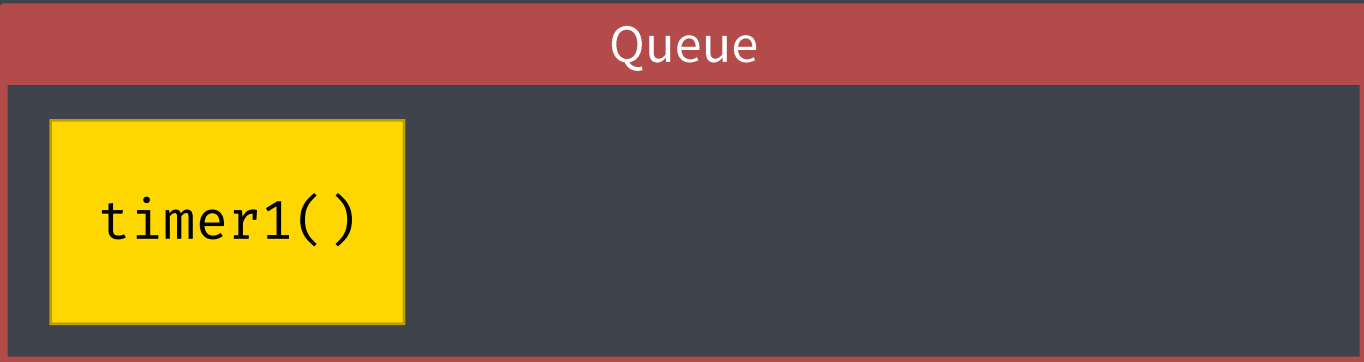
  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)

console.log('global end')
```

Console
global begin
global end
timer2 invoke



Event loop





```
console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

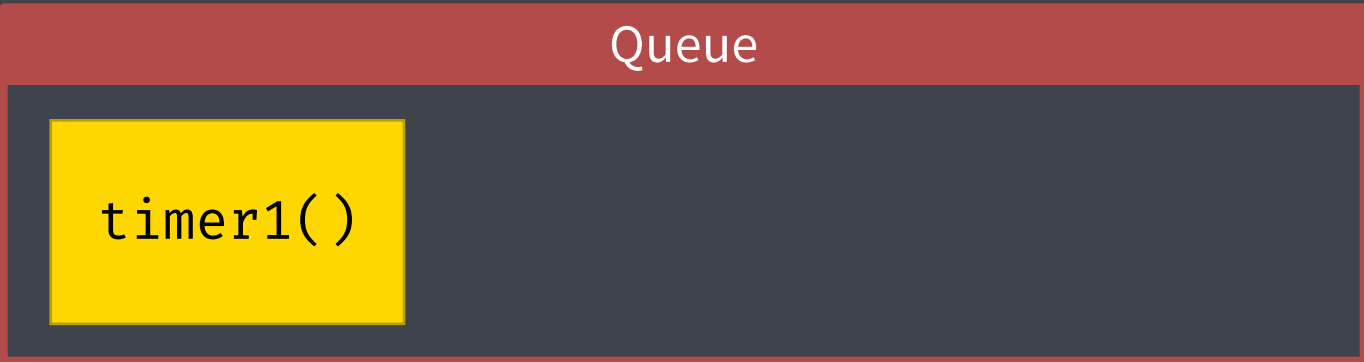
  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)

console.log('global end')
```

Console
global begin
global end
timer2 invoke



Event loop



```
console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

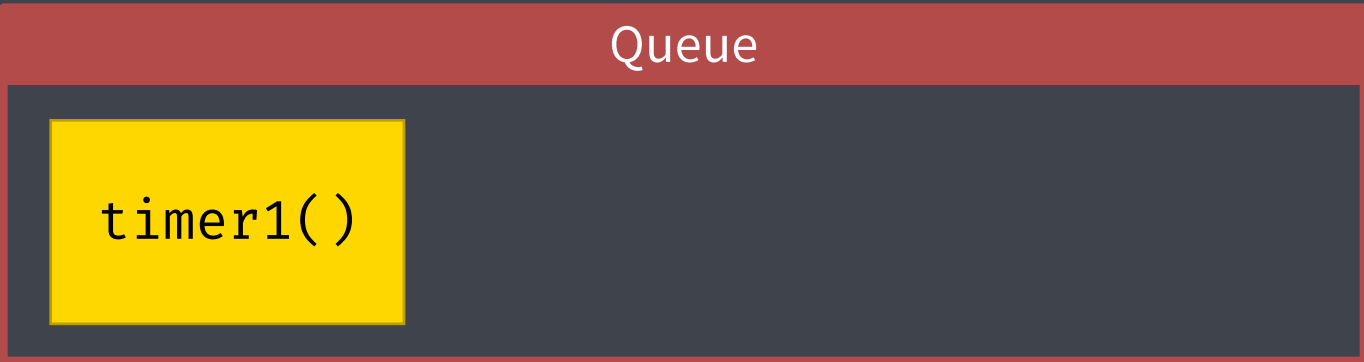
  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)

console.log('global end')
```

Console
global begin
global end
timer2 invoke



Event loop



```
console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

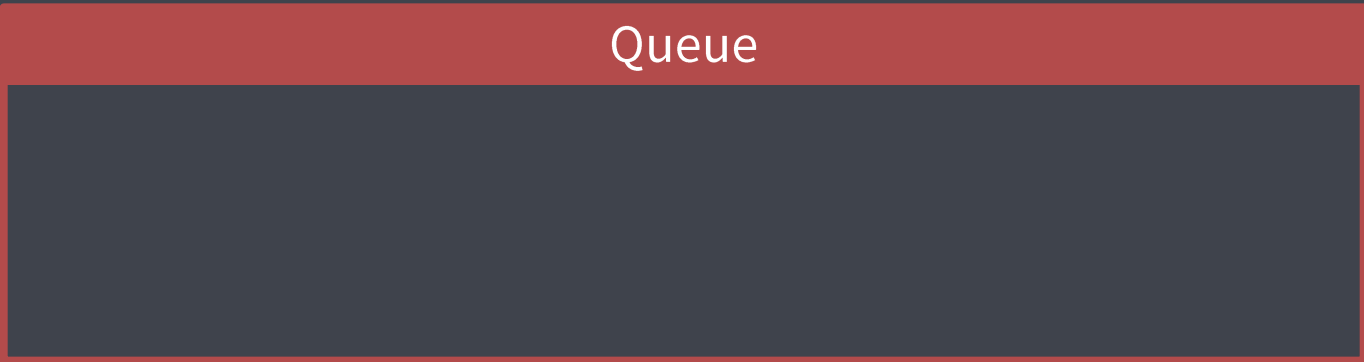
  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)

console.log('global end')
```

Console
global begin
global end
timer2 invoke



Event loop



```
console.log('global begin')
```

```
setTimeout(function timer1 () {  
  console.log('timer1 invoke')  
}, 1800)
```

```
setTimeout(function timer2 () {  
  console.log('timer2 invoke')
```

```
  setTimeout(function inner () {  
    console.log('inner invoke')  
  }, 1000)  
}, 1000)
```

```
console.log('global end')
```

#### Console

global begin

global end

timer2 invoke

#### Call stack

timer1()

#### Web APIs

⌚ 1s inner

Event loop



#### Queue

```
console.log('global begin')
```

```
setTimeout(function timer1 () {  
  console.log('timer1 invoke')  
}, 1800)
```

```
setTimeout(function timer2 () {  
  console.log('timer2 invoke')
```

```
    setTimeout(function inner () {  
      console.log('inner invoke')  
    }, 1000)  
  }, 1000)
```

```
console.log('global end')
```

#### Console

global begin

global end

timer2 invoke

#### Call stack

console.log()

timer1()

#### Web APIs

⌚ 1s inner

Event loop



#### Queue

```
console.log('global begin')
```

```
setTimeout(function timer1 () {  
  console.log('timer1 invoke')  
}, 1800)
```

```
setTimeout(function timer2 () {  
  console.log('timer2 invoke')
```

```
  setTimeout(function inner () {  
    console.log('inner invoke')  
  }, 1000)  
}, 1000)
```

```
console.log('global end')
```

#### Console

global begin

global end

timer2 invoke

timer1 invoke

#### Call stack

console.log()

timer1()

#### Web APIs

⌚ 1s inner

#### Event loop



#### Queue

```
console.log('global begin')
```

```
setTimeout(function timer1 () {  
  console.log('timer1 invoke')  
}, 1800)
```

```
setTimeout(function timer2 () {  
  console.log('timer2 invoke')
```

```
  setTimeout(function inner () {  
    console.log('inner invoke')  
  }, 1000)  
}, 1000)
```

```
console.log('global end')
```

#### Console

global begin

global end

timer2 invoke

timer1 invoke

#### Call stack

timer1()

#### Web APIs

⌚ 1s inner

Event loop



#### Queue

```
console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

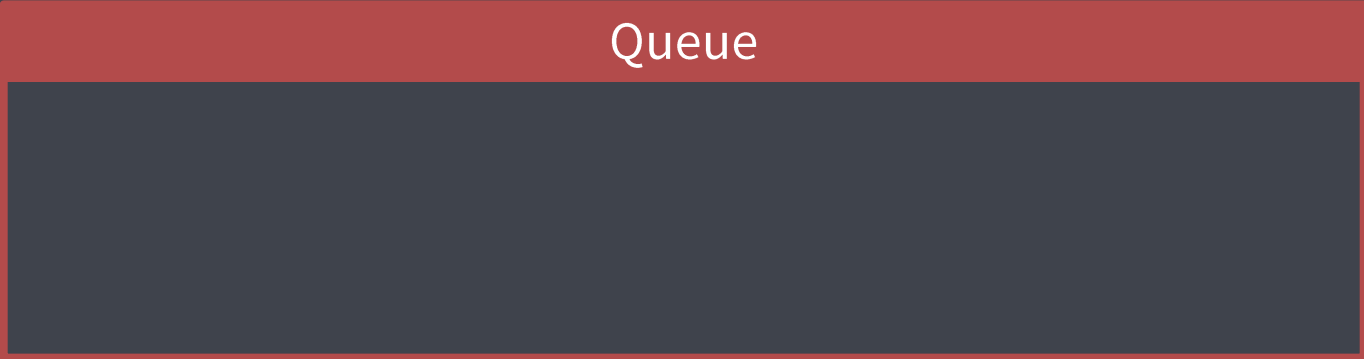
  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)

console.log('global end')
```

Console
global begin
global end
timer2 invoke
timer1 invoke



Event loop





```
console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

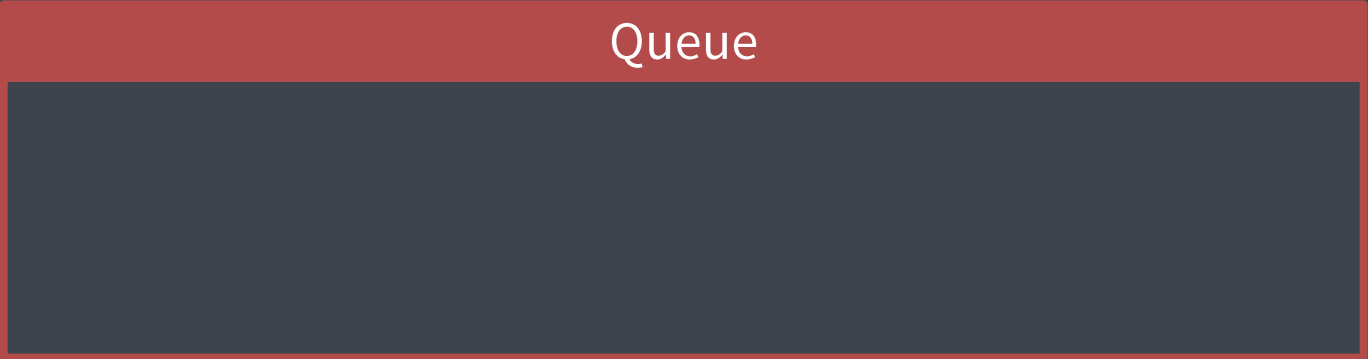
  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)

console.log('global end')
```

Console
global begin
global end
timer2 invoke
timer1 invoke



Event loop



```
console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

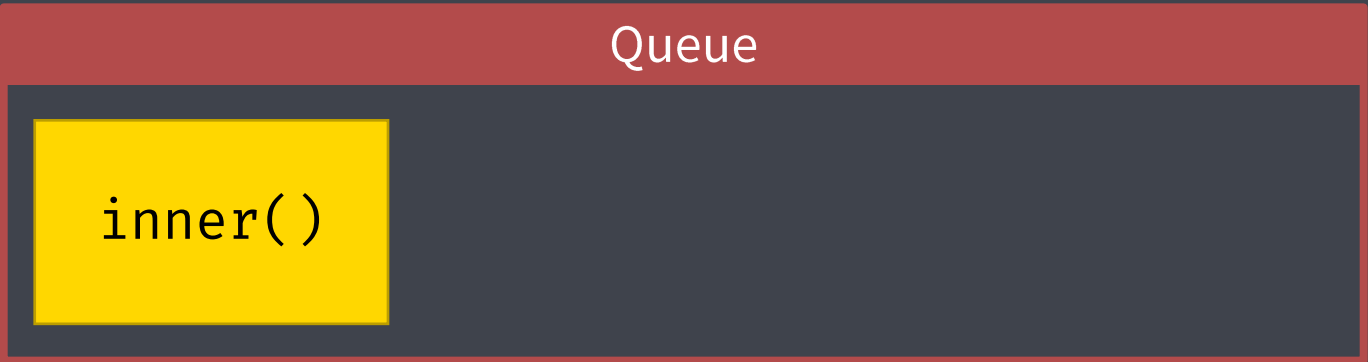
  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)

console.log('global end')
```

Console
global begin
global end
timer2 invoke
timer1 invoke



Event loop



```
console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

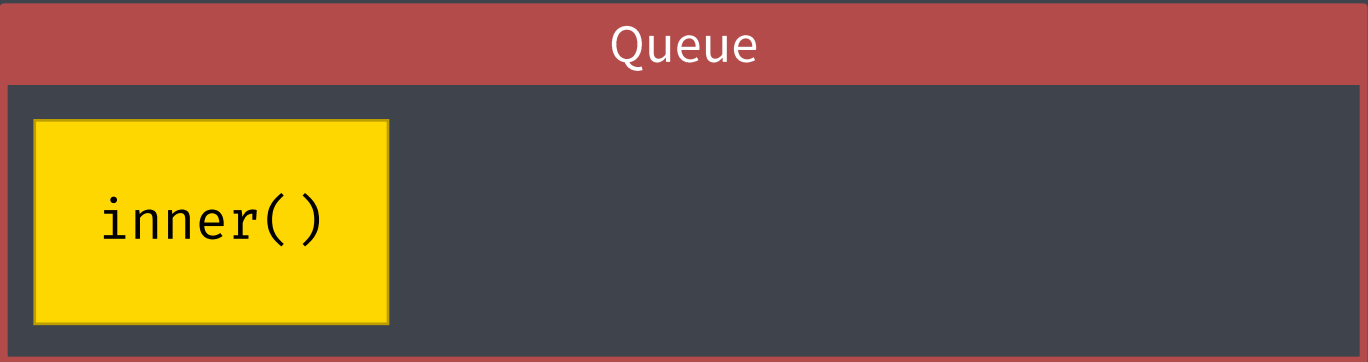
  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)

console.log('global end')
```

Console
global begin
global end
timer2 invoke
timer1 invoke



Event loop



```

console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)

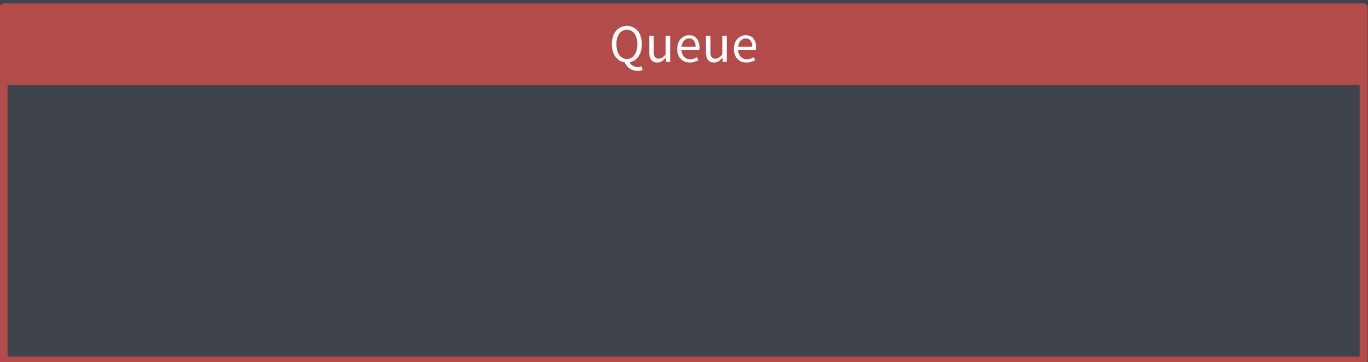
console.log('global end')

```

Console
global begin
global end
timer2 invoke
timer1 invoke



Event loop



```
console.log('global begin')
```

```
setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)
```

```
setTimeout(function timer2 () {
  console.log('timer2 invoke')
```

```
  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)
```

```
console.log('global end')
```

### Console

global begin

global end

timer2 invoke

timer1 invoke

### Call stack

inner()

### Web APIs

Event loop



### Queue

```
console.log('global begin')
```

```
setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)
```

```
setTimeout(function timer2 () {
  console.log('timer2 invoke')
```

```
  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)
```

```
console.log('global end')
```

#### Console

global begin

global end

timer2 invoke

timer1 invoke

#### Call stack

console.log()

inner()

#### Web APIs

Event loop



#### Queue

```
console.log('global begin')
```

```
setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)
```

```
setTimeout(function timer2 () {
  console.log('timer2 invoke')
```

```
  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)
```

```
console.log('global end')
```

### Console

global begin

global end

timer2 invoke

timer1 invoke

inner invoke

### Call stack

console.log()

inner()

### Web APIs

Event loop



### Queue

```
console.log('global begin')
```

```
setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)
```

```
setTimeout(function timer2 () {
  console.log('timer2 invoke')
```

```
  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)
```

```
console.log('global end')
```

### Console

global begin

global end

timer2 invoke

timer1 invoke

inner invoke

### Call stack

inner()

### Web APIs

Event loop



### Queue



```
console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)

console.log('global end')
```

#### Console

global begin

global end

timer2 invoke

timer1 invoke

inner invoke

#### Call stack

inner()

#### Web APIs

Event loop



#### Queue

```
console.log('global begin')

setTimeout(function timer1 () {
  console.log('timer1 invoke')
}, 1800)

setTimeout(function timer2 () {
  console.log('timer2 invoke')

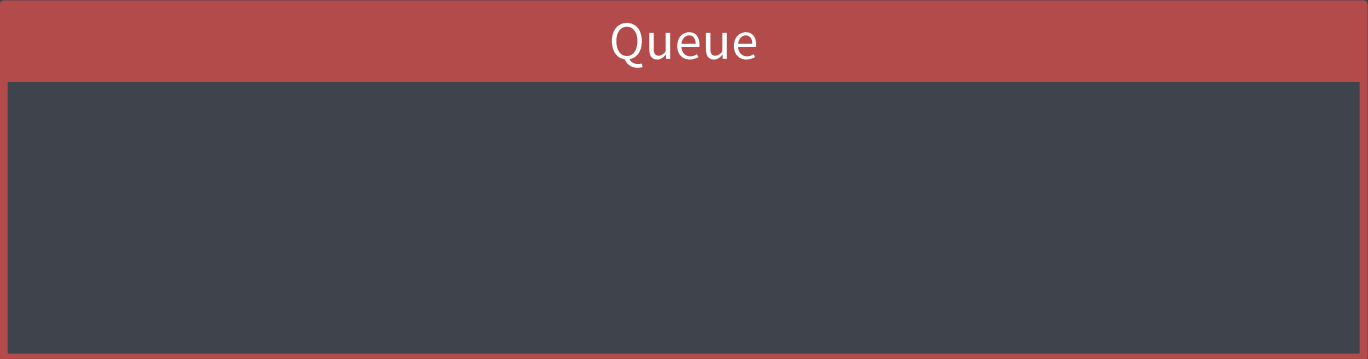
  setTimeout(function inner () {
    console.log('inner invoke')
  }, 1000)
}, 1000)

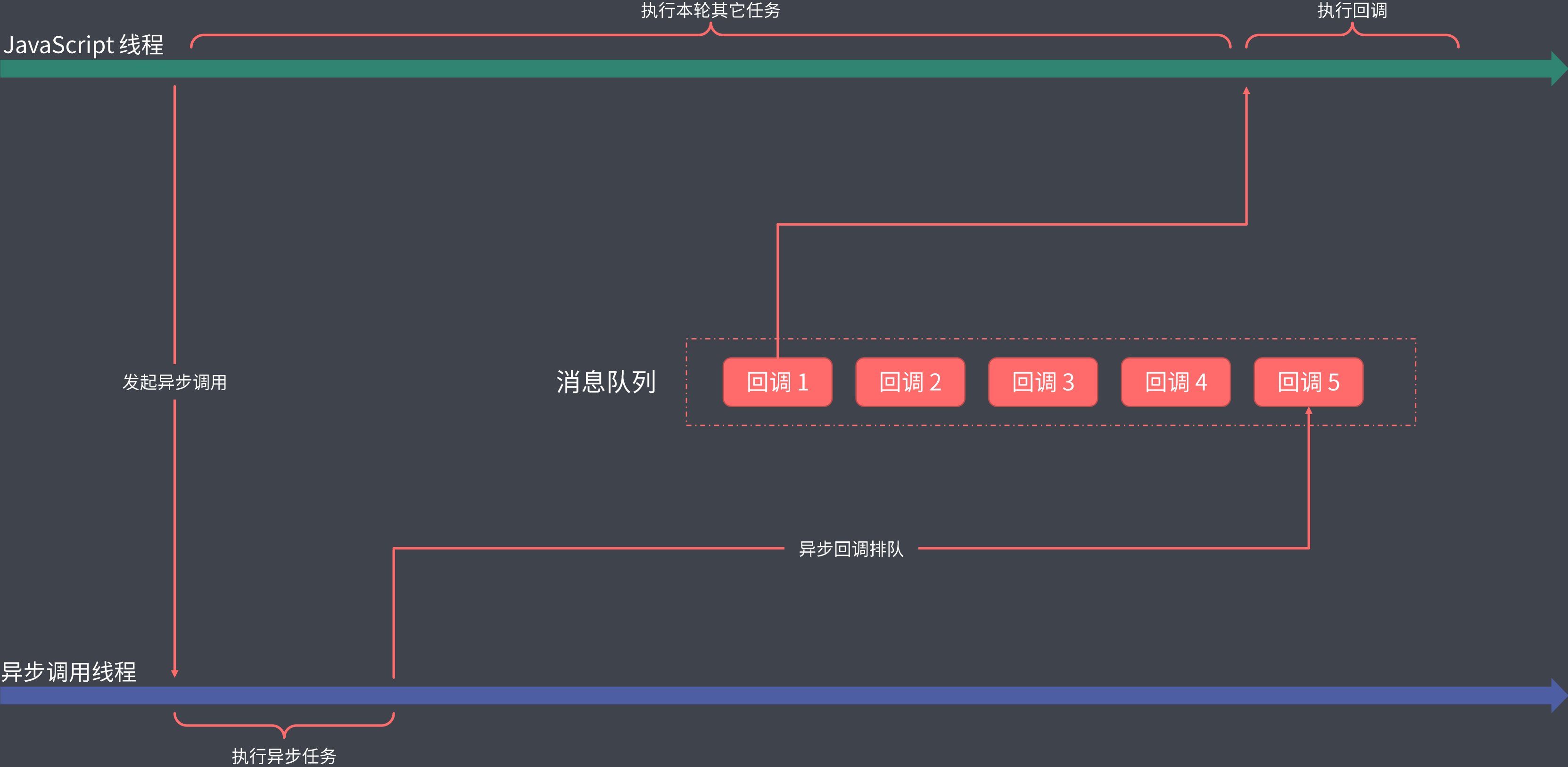
console.log('global end')
```

Console
global begin
global end
timer2 invoke
timer1 invoke
inner invoke



Event loop





# 回调函数

所有异步编程方案的根基





知道该怎么刷

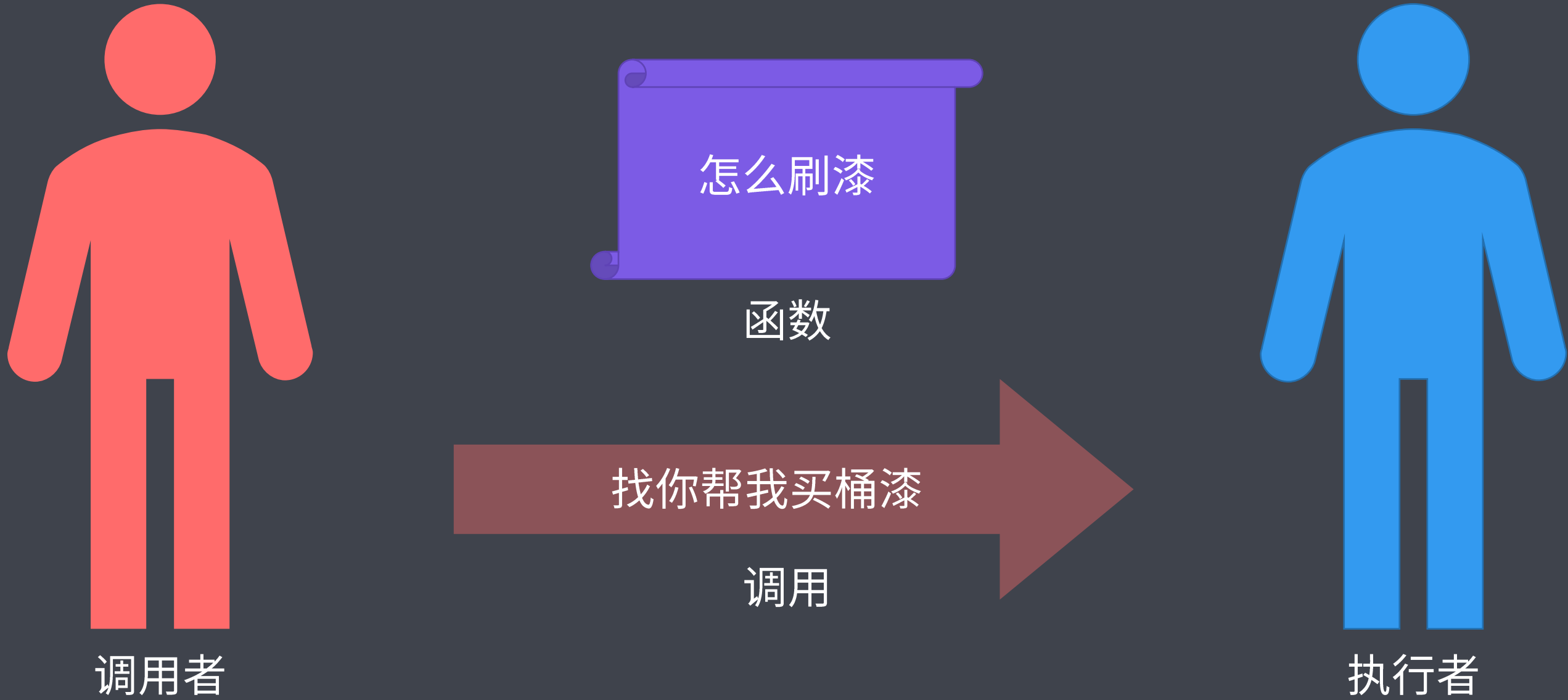


没有油漆












```
function foo (callback) {  
  setTimeout(function () {  
    callback()  
  }, 3000)  
}
```

```
foo(function () {  
  console.log('这就是一个回调函数')  
  console.log('调用者定义这个函数，执行者执行这个函数')  
  console.log('其实就是调用者告诉执行者异步任务结束后应该做什么')  
})
```

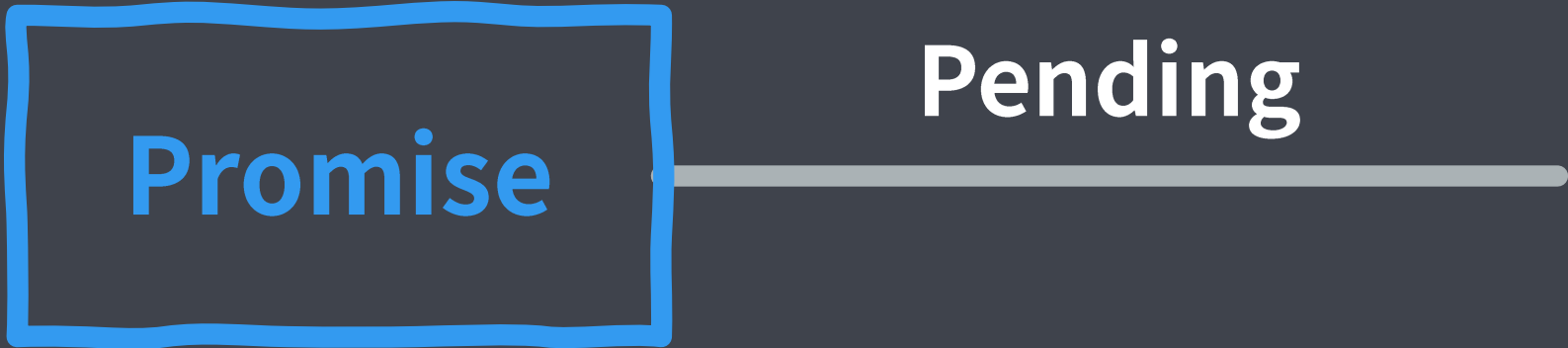
# Promise

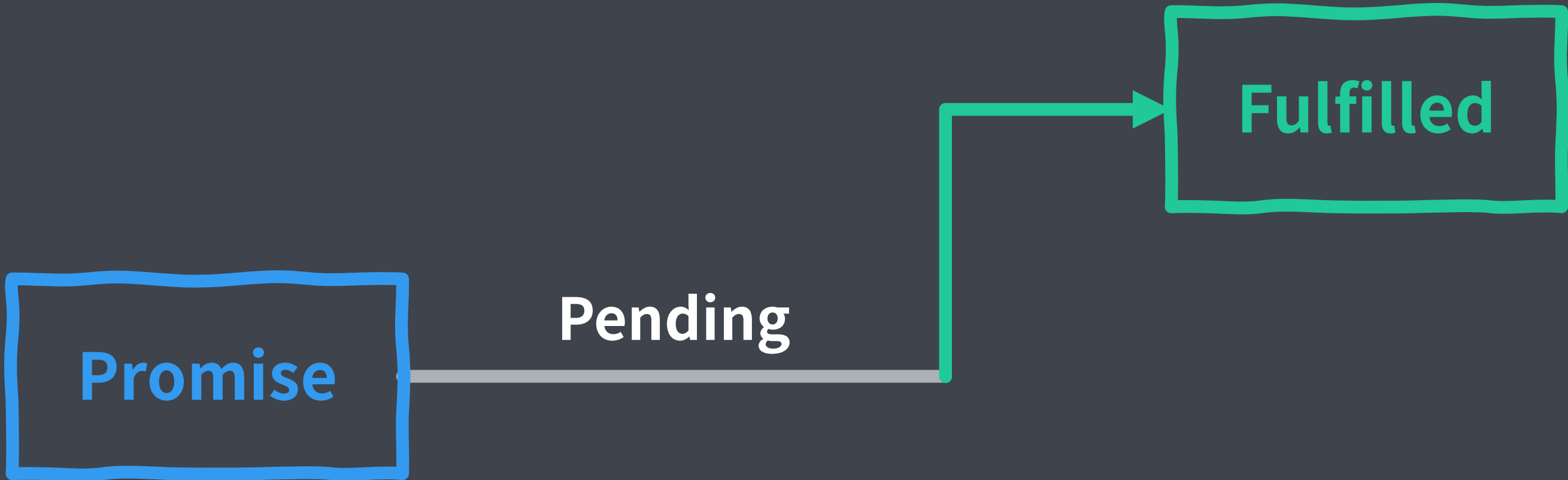
一种更优的异步编程统一方案



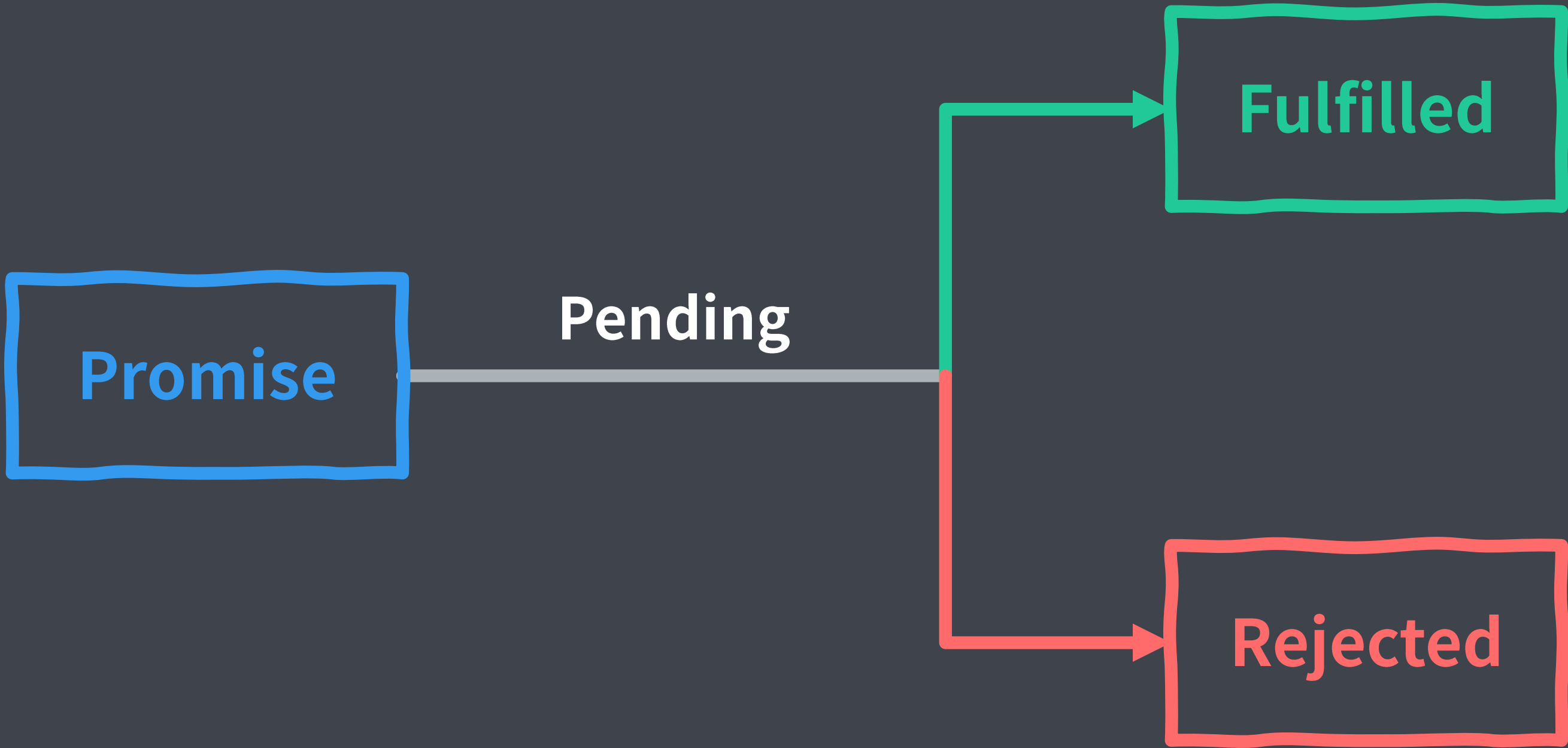
```
$.get('/url1', function (data1) {  
  $.get('/url2', data1, function (data2) {  
    $.get('/url3', data2, function (data3) {  
      $.get('/url4', data3, function (data4) {  
        $.get('/url5', data4, function (data5) {  
          $.get('/url6', data5, function (data6) {  
            $.get('/url7', data6, function (data7) {  
              // 略微夸张了一点点  
            })  
          })  
        })  
      })  
    })  
  })  
})
```

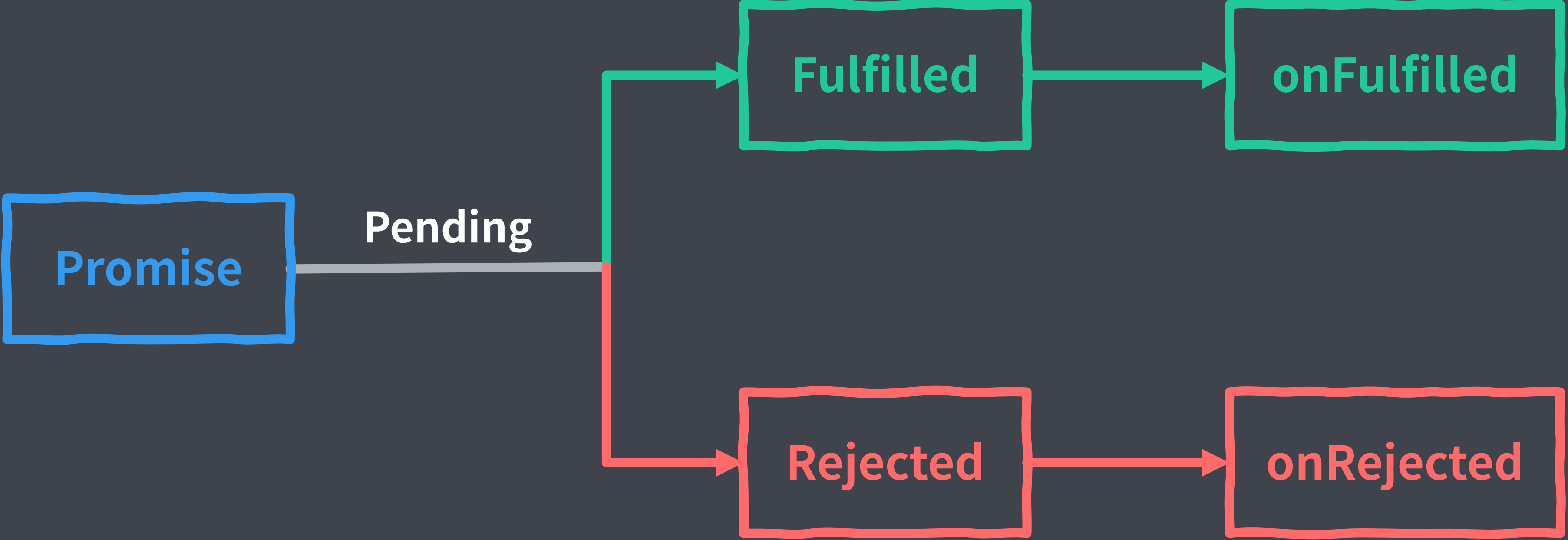
Promise











# Promise

## 基本用法

# Promise

使用案例

# Promise

常见误区

# Promise

链式调用

# Promise

异常处理

# Promise

静态方法



# Promise

并行执行

# Promise

执行时序 / 宏任务 vs. 微任务

# Generator 异步方案

回顾 Generator 函数

# Generator 异步方案

体验 Generator 函数异步方案

# Generator 异步方案

递归执行 Generator 函数

# Async / Await 语法糖

语言层面的异步编程标准