

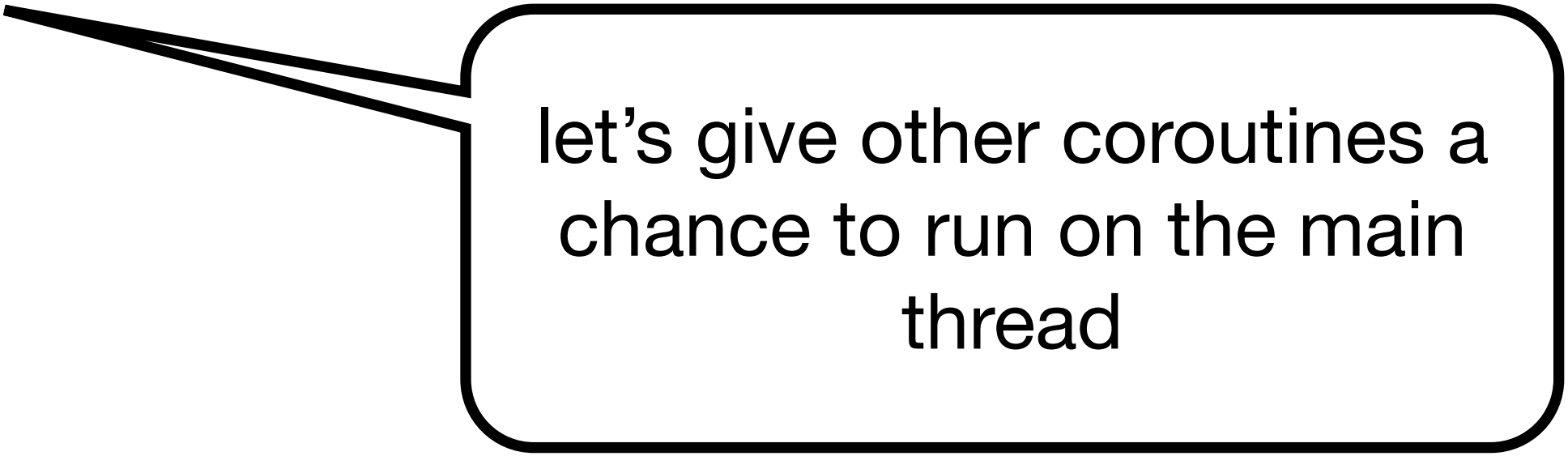
Cooperative Cancellation

3

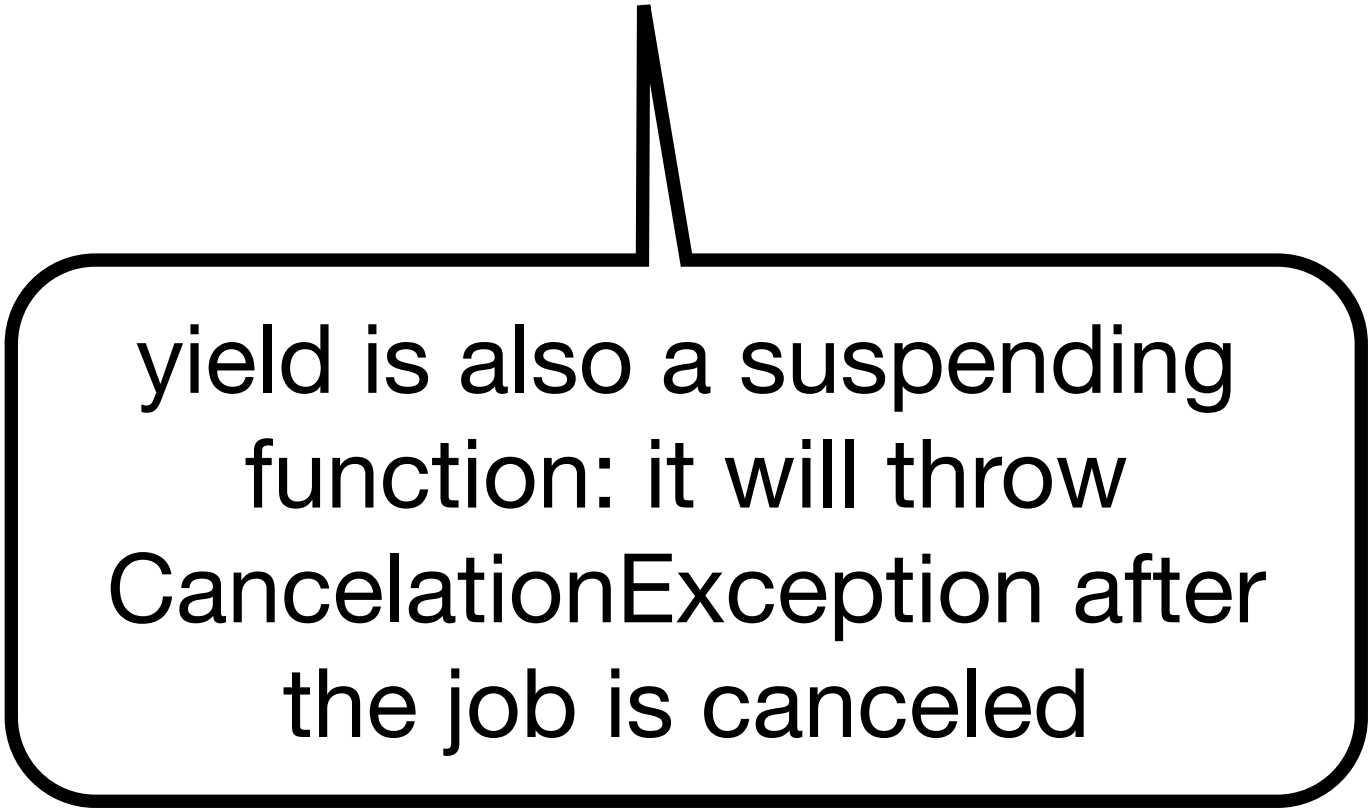
4

```
fun CoroutineScope.cashDesk() = launch {  
    while (true) {  
        println("How many pretzels?")  
        val count = readLine()?.toIntOrNull()  
        yield()  
        //receive pretzel  
    }  
}
```

```
fun main() = runBlocking {  
    val job = cashDesk()  
    delay(1000)  
    job.cancelAndJoin()  
}
```



let's give other coroutines a
chance to run on the main
thread



yield is also a suspending
function: it will throw
CancellationException after
the job is canceled

Cooperative Cancellation

```
fun main() = runBlocking {  
    val job = cashDesk()  
    delay(1000)  
    job.cancelAndJoin()  
}
```

```
fun CoroutineScope.cashDesk() = launch {  
    while (true) {  
        println("How many pretzels?")  
        val count = readLine()?.toIntOrNull()  
        yield()  
        //receive pretzel  
    }  
}
```

yield is also a suspending function: it will throw CancellationException after the job is canceled

let's give other coroutines a chance to run on the main thread

Cooperative Cancellation

```
fun main() = runBlocking {  
    val job = cashDesk()  
    delay(1000)  
    job.cancelAndJoin()  
}
```

```
fun CoroutineScope.cashDesk() = launch(IO) {  
    while (isActive) {  
        println("How many pretzels?")  
        val count = readLine()?.toIntOrNull()  
        //receive pretzel  
    }  
}
```