

Blocking



Blocking

Run blocking effects on a dedicated thread pool

Blocking ZIOs cannot (usually) be interrupted

Can use *attemptBlockingInterrupt*

- based on `Thread.interrupt`, a heavy operation
- can also fail if the code catches `InterruptedException`

Best option: *attemptBlockingCancelable*

- have an external flag/switch
- pass a "canceling" effect which manipulates the switch
- on interrupt, the fiber will call the canceling effect

Semantic blocking

- no threads are really blocked, only descheduled

Yielding

```
val blockingZIO = ZIO.attemptBlocking {  
  println("a long computation...")  
  Thread.sleep(10000)  
  42  
}
```

```
def bcEffect(flag: AtomicBoolean): Task[Unit] =  
  ZIO.attemptBlockingCancelable {  
    (1 to 100000).foreach { element =>  
      if (!flag.get()) {  
        println(element)  
        Thread.sleep(100)  
      }  
    }  
  } (ZIO.succeed(cancelFlag.set(true)))
```

ZIO rocks

