




```
private suspend fun processOrders(orders: ReceiveChannel<Menu.Cappuccino>,
    espressoMachine: EspressoMachine) {
    orders.consumeEach {
        val groundBeans = grindCoffeeBeans(it.beans())
        val espressoDeferred = async {
            espressoMachine.pullEspressoShot(groundBeans)
        }
        val steamedMilkDeferred = async {
            espressoMachine.steamMilk(it.milk())
        }
        val cappuccino = makeCappuccino(
            it, espressoDeferred.await(), steamedMilkDeferred.await())
        log("serve: $cappuccino")
    }
}
```



<http://jagstark.page.link/examples>

<https://jagstalk.page.link/examples>

```
private suspend fun processOrders(orders: ReceiveChannel<Menu.Cappuccino>,
    espressoMachine: EspressoMachine) {
    orders.consumeEach {
        val groundBeans = grindCoffeeBeans(it.beans())
        val espressoDeferred = async {
            espressoMachine.pullEspressoShot(groundBeans)
        }
        val steamedMilkDeferred = async {
            espressoMachine.steamMilk(it.milk())
        }
        val cappuccino = makeCappuccino(
            it, espressoDeferred.await(), steamedMilkDeferred.await())
        log("serve: $cappuccino")
    }
}
```



<https://jagstalk.page.link/examples>

```
private suspend fun processOrders(orders: ReceiveChannel<Menu.Cappuccino>,
    espressoMachine: EspressoMachine) {
    orders.consumeEach {
        val groundBeans = grindCoffeeBeans(it.beans())
        coroutineScope {
            val espressoDeferred = async {
                espressoMachine.pullEspressoShot(groundBeans)
            }
            val steamedMilkDeferred = async {
                espressoMachine.steamMilk(it.milk())
            }
            val cappuccino = makeCappuccino(
                it, espressoDeferred.await(), steamedMilkDeferred.await())
        }
        log("serve: $cappuccino")
    }
}
```