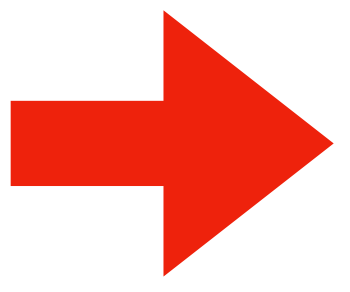
```
class EspressoMachine(scope: CoroutineScope): CoroutineScope by scope {
    ...
}
```

```
fun main() = runBlocking {
    ...
    val espressoMachine = EspressoMachine(this)
    val time = measureTimeMillis {
        coroutineScope {
            launch(CoroutineName("barista-1")) {
                processOrders(ordersChannel, espressoMachine)
            }
            launch(CoroutineName("barista-2")) {
                processOrders(ordersChannel, espressoMachine)
            }
        }
    }
    log("time: $time ms")
}

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



https://jagstalk.page.link/examples

https://jagstalk.page.link/examples

```
class EspressoMachine(scope: CoroutineScope): CoroutineScope by scope {
    ...
}

fun main() = runBlocking {
    ...
    val espressoMachine = EspressoMachine(this)
    val time = measureTimeMillis {
        coroutineScope {
            launch(CoroutineName("barista-1")) {
                processOrders(ordersChannel, espressoMachine)
            }
            launch(CoroutineName("barista-2")) {
                processOrders(ordersChannel, espressoMachine)
            }
        }
    }
    log("time: $time ms")
}

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