


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

```
private suspend fun processOrders (orders: Flow<Menu.Cappuccino>,
    espressoMachine: EspressoMachine): Flow<Beverage.Cappuccino> {
    ...
}
```

```
fun main() = runBlocking {  
    val orders = listOf(...)  
    val ordersFlow: Flow<Menu.Cappuccino> = orders.asFlow()  
  
    val espressoMachine = EspressoMachine(this)  
    val time = measureTimeMillis {  
        flowOf(  
            processOrders(ordersFlow, espressoMachine),  
            processOrders(ordersFlow, espressoMachine)  
        )  
    }  
    log("time: $time ms")  
    espressoMachine.destroy()  
}
```




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

```
fun main() = runBlocking {  
    val orders = listOf(...)  
    val ordersFlow: Flow<Menu.Cappuccino> = orders.asFlow()  
  
    val espressoMachine = EspressoMachine(this)  
    val time = measureTimeMillis {  
        flowOf(  
            processOrders(ordersFlow, espressoMachine),  
            processOrders(ordersFlow, espressoMachine)  
                    )  
    }  
    log("time: $time ms")  
    espressoMachine.destroy()  
}
```

```
private suspend fun processOrders(orders: Flow<Menu.Cappuccino>,  
    espressoMachine: EspressoMachine): Flow<Beverage.Cappuccino> {  
    ...  
}
```

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

```
fun main() = runBlocking {  
    val orders = listOf(...)  
    val ordersFlow: Flow<Menu.Cappuccino> = orders.asFlow()  
  
    val espressoMachine = EspressoMachine(this)  
    val time = measureTimeMillis {  
        flowOf(  
            processOrders(ordersFlow, espressoMachine),  
            processOrders(ordersFlow, espressoMachine)  
        )  
         .flattenMerge()  
    }  
    log("time: $time ms")  
    espressoMachine.destroy()  
}
```

```
private suspend fun processOrders(orders: Flow<Menu.Cappuccino>,  
    espressoMachine: EspressoMachine): Flow<Beverage.Cappuccino> {  
    ...  
}
```