

## Mão à obra: Juntando resultados

Agora vamos criar a tarefa que é responsável por juntar o resultados dos dois comandos criados no exercício anterior.

1) Crie uma nova classe chamada `JuntaResultadosFutureWSFutureBanco` , que implementa a interface `Callable<Void>` :

```
public class JuntaResultadosFutureWSFutureBanco implements Callable<Void> {

    //não queremos devolver nada, entao usamos um tipo que representa nada: Void
    public Void call() {

        return null;
    }
}
```

2) Nessa classe, receba no construtor as duas instâncias de `Future` e a saída do cliente:

```
private Future<String> futureWS;
private Future<String> futureBanco;
private PrintStream saidaCliente;

public JuntaResultadosFutureWSFutureBanco(Future<String> futureWS, Future<String> futureBanco, I
    this.futureWS = futureWS;
    this.futureBanco = futureBanco;
    this.saidaCliente = saidaCliente;
}
```

3) No método `call()` , utilize os `Future` s para pegar o resultado. Não espere mais de 20s:

```
//dentro do método call
System.out.println("Aguardando resultados do future WS e Banco");

try {
    String numeroMagico = this.futureWS.get(20, TimeUnit.SECONDS);
    String numeroMagico2 = this.futureBanco.get(20, TimeUnit.SECONDS);

    this.saidaCliente.println("Resultado do comando c2: " + numeroMagico + ", " + numeroMagico2);

} catch (InterruptedException | ExecutionException | TimeoutException e) {

    //aqui vem mais
}

System.out.println("Finalizou JuntaResultadosFutureWSFutureBanco");
```

4) Ainda no método `call` , dentro do bloco `catch` cancele os `Future` s e devolva uma mensagem para o cliente:

//dentro do bloco catch:

```
System.out.println("Timeout: Cancelando a execução do comando c2");

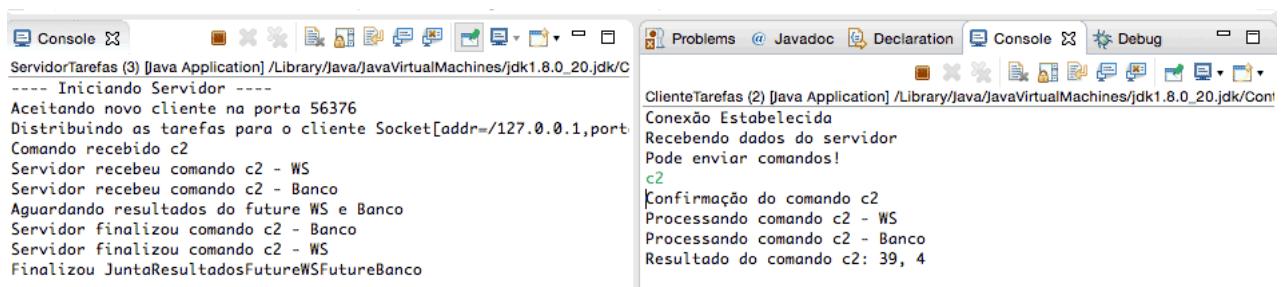
this.futureWS.cancel(true);
this.futureBanco.cancel(true);

this.saidaCliente.println("Timeout na execução do comando c2");
```

5) Vamos voltar para a classe `DistribuirTarefas` no nosso `case "c2"`. Antes do `break`, adicione o código para criar a tarefa e submeter ao pool:

```
Callable<Void> juntaResultados = new JuntaResultadosFutureWSFutureBanco(futureWS, futureBanco, this.threadPool.submit(juntaResultados);
```

6) Tudo deve estar compilando. Agora, para testar, rode primeiro o servidor e depois o cliente. Envie o comando `c2` e espere o resultado (mínimo 15s)! A saída deve ser parecida com:



**ServidorTarefas (3) [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0\_20.jdk/Console**

```
----- Iniciando Servidor -----
Aceitando novo cliente na porta 56376
Distribuindo as tarefas para o cliente Socket[addr=/127.0.0.1, port=56376]
Comando recebido c2
Servidor recebeu comando c2 - WS
Servidor recebeu comando c2 - Banco
Aguardando resultados do future WS e Banco
Servidor finalizou comando c2 - Banco
Servidor finalizou comando c2 - WS
Finalizou JuntaResultadosFutureWSFutureBanco
```

**ClienteTarefas (2) [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0\_20.jdk/Console**

```
Conexão Estabelecida
Recebendo dados do servidor
Pode enviar comandos!
c2
Confirmação do comando c2
Processando comando c2 - WS
Processando comando c2 - Banco
Resultado do comando c2: 39, 4
```