**ListenableFuture из Guava**

C помощью метода ***Futures.allAsList()*** мы можем объединить ***много*** экземпляров ***ListenableFuture*** в ***один*** ***ListenableFuture***, который завершается только после успешного завершения всех объединенных потоков, каждый из которых выполняет свою задачу:

ExecutorService executorService = Executors. newFixedThreadPool (2); // ( пул на 2 потока, с автоматическим восстановлением потока)  
ListeningExecutorService listeningExecutorService = MoreExecutors.listeningDecorator(executorService);  
  
ListenableFuture<String> future1 = listeningExecutorService.submit(() -> "Hello");

ListenableFuture<String> future2 = listeningExecutorService.submit(() -> "World");  
  
String greeting = Futures.allAsList(future1, future2).get().stream().collect(Collectors.joining(" "));

assertEquals("Hello World", greeting);

**https://www.baeldung.com/guava-futures-listenablefuture**

Одним из способов объединения нескольких фьючерсов является использование метода Futures.allAsList(). Это позволяет нам собирать результаты всех фьючерсов, если все они успешны, в порядке предоставленных фьючерсов. Если одно из этих фьючерсов терпит неудачу, то весь результат — неудавшееся будущее. Механизм обработки для удачного и неудачного результата:

ListenableFuture<List<String>> configsTask = Futures.successfulAsList(task1, task2, task3);

Futures.addCallback(configsTask, new FutureCallback<List<String>>() {

@Override

public void onSuccess(@Nullable List<String> configResults) {

// handle results. If task2 failed, then configResults.get(1) == null

}

@Override

public void onFailure(Throwable t) {

// handle failure

}

}, listeningExecutor);