# Scenario

1. Reactive thinking не новое изобретение (spreadsheet)
2. Реактивный манифест
3. Существующие решения на java
4. Библиотека изначально была написана Майкрософт, библиотека была портирована на все языки
5. Главные игроки в rxjava
6. Что такое Operator
7. Word and antonym, pull vs push, message-driven push
8. Marble diagrams, map, filter, chaining
9. Пример с пивоварением
10. Как создать данные с Observable
11. Демо с BeerClient
12. Методы с side effects для отладки
13. Методы для работы с ошибками
14. Демо с BeerClientWithFailover
15. Schedulers, многопоточность, subscribeOn – Observable будет делать push на данном конкретном thread, observerOn – Observer будет делать subscribe на данном конкретном thread. Различные стратегии.
16. Параллельная обработка и фокус с flatmap.
17. Демо SubscribeOnObserveOn, ParallelStreams
18. Backpressure, можно избавиться от Spring batch
19. Refactor JpaBoundsOperators.mono

|  |
| --- |
| MonoProcessor<V> monoProcessor = MonoProcessor.create();  input  .publishOn(Schedulers.elastic())  .transform(transformer)  .subscribe(monoProcessor);  return monoProcessor.subscribeOn(Schedulers.elastic()); |

1. Refactor GitterService.

|  |
| --- |
| **gitterMessageSource** = Flux.*from*(gitterClient.getMessages(**null**))  .onBackpressureBuffer()  .publish(1)  .autoConnect(0); |

1. Refactor DefaultMessageService.latest

|  |
| --- |
| return chatClient.stream()  .transform(MessageMapper::toViewModelUnits); |