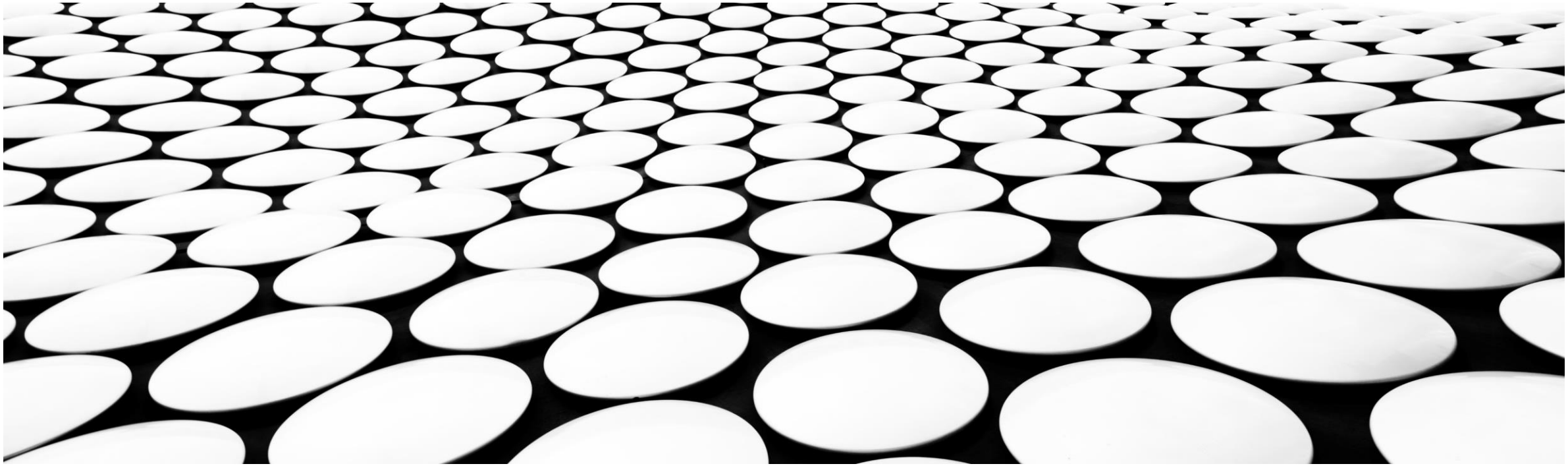


服务端开发-反应式编程基础

陶召胜

两种不同的编程范式

- 命令式编程，imperative
- 反应式编程，Reactive Programming

Reactive Programming解决什么问题

- IO密集型场景
- 同步阻塞模型，阻塞线程多，CPU利用率不高，性能下降
- 管理多线程，意味着更高的复杂性
- 学习文档
  - ✓ 剖析Reactor 模型：https://mp.weixin.qq.com/s?__biz=MzIwNDAyOTI2Nw==&mid=2247483716&idx=1&sn=91e7c3f7a46b6d054b8a938cefd3120d&chksm=96c72d78a1b0a46e6f3058c6c895496caab199184376d817a310fbd73620d55dd2bbc434b8d1&token=1026451003&lang=zh_CN#rd
  - ✓ Java NIO 底层原理：https://www.toutiao.com/article/6887439886178058759/
  - ✓ Netty介绍：https://www.zhihu.com/question/607575828/answer/3157903032

Reactor项目

```
<dependency>
    <groupId>io.projectreactor</groupId>
    <artifactId>reactor-core</artifactId>
</dependency>
```

- Reactive Streams：Netflix、Lightbend和Pirotal于2013年开始制定的一种规范，旨在提供无阻塞回压的异步流处理标准
- Reactor：Spring Pivotal团队提供的响应式编程的Java实现，其它类似实现：RxJava
  - ✓ 函数式、声明式，描述数据会流经的管道或流
- Spring WebFlux：启用基于响应式编程的Web应用程序的开发。提供类似于Spring MVC的编程模型

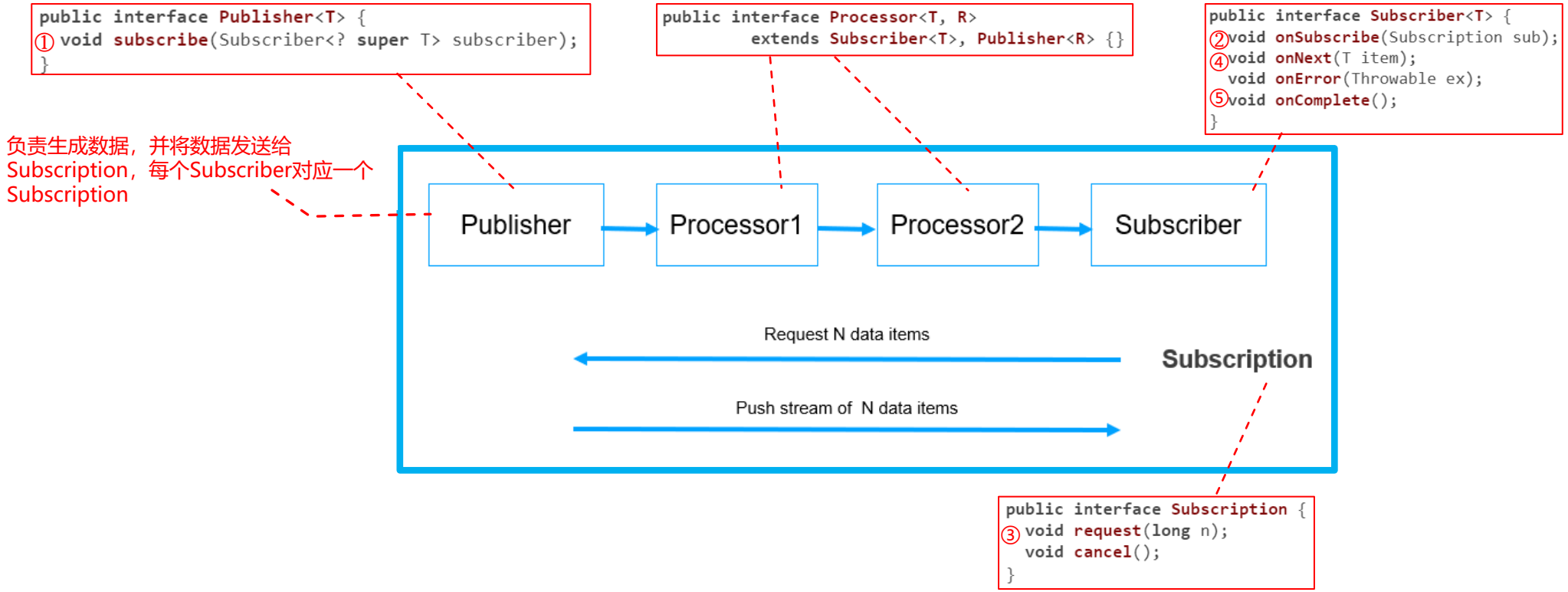
Java的stream与反应式的流区别

- Java的stream通常都是同步的，并且只能处理有限的数据集，本质上来说，它们只是使用函数来对集合进行迭代的一种方式
- JDK9中的 Flow API对应反应式流

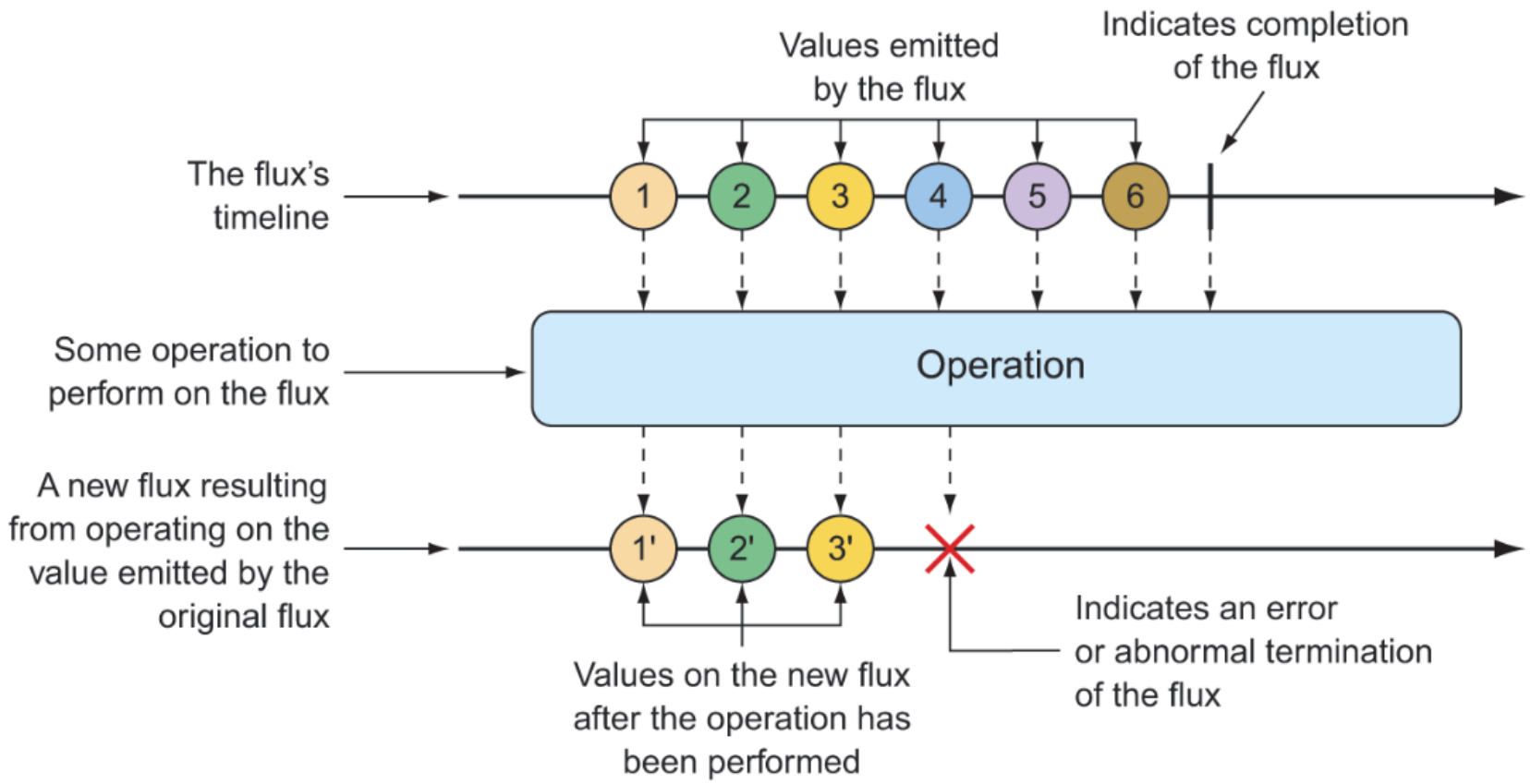
```
3   public final class Flow {
4
5       private Flow() {}
6
7       @FunctionalInterface
8       public static interface Publisher<T> {
9           public void subscribe(Subscriber<? super T> subscriber);
10      }
11
12      public static interface Subscriber<T> {
13          public void onSubscribe(Subscription subscription);
14          public void onNext(T item);
15          public void onError(Throwable throwable);
16          public void onComplete();
17      }
18
19      public static interface Subscription {
20          public void request(long n);
21          public void cancel();
22      }
23
24      public static interface Processor<T,R> extends Subscriber<T>, Publisher<R> {
25      }
26  }
```

反应式流规范定义的4个接口

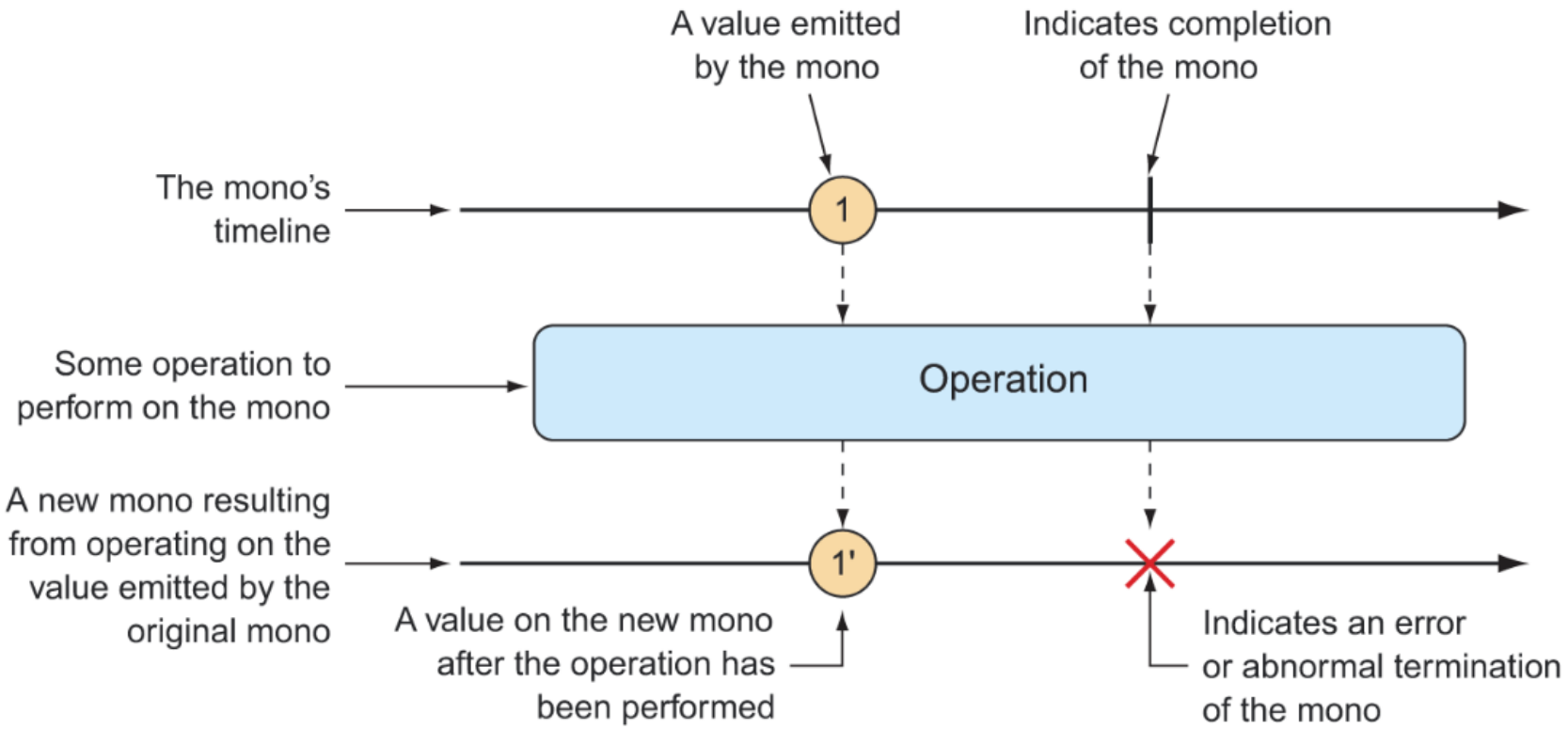
- org.reactivestreams.*



反应式流图（Flux）



反应式流图（Mono）


A value emitted by the mono
Indicates completion of the mono
The mono's timeline
1
Some operation to perform on the mono
Operation
A new mono resulting from operating on the value emitted by the original mono
1'
A value on the new mono after the operation has been performed
Indicates an error or abnormal termination of the mono

两个基本概念：Flux 和 Mono

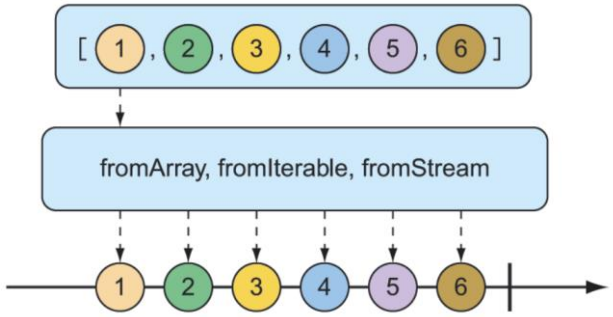
- Flux：包含 0 到 N 个元素的异步序列
- Mono：包含 0 或者 1 个元素的异步序列
- 消息：正常的包含元素的消息、序列结束的消息和序列出错的消息
- 操作符（Operator）：对流上元素的操作

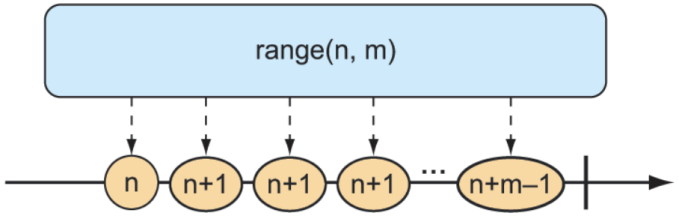
操作类型

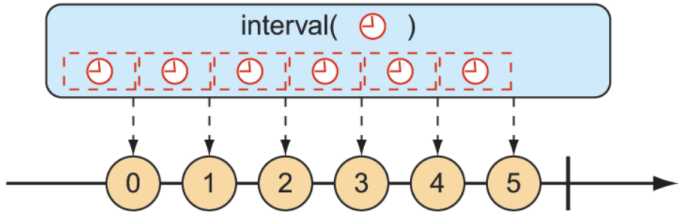
- 创建操作
- 组合操作
- 转换操作
- 逻辑操作

创建Flux

- Flux的静态方法
- 根据对象创建，just方法
- 根据集成创建，数组、Iterable、Java Stream
- range
- interval

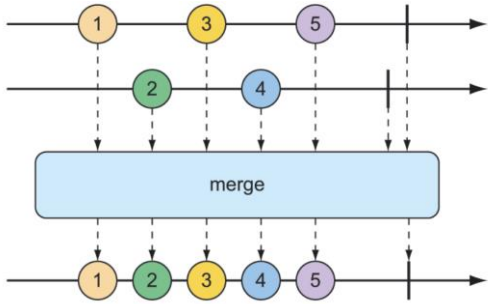


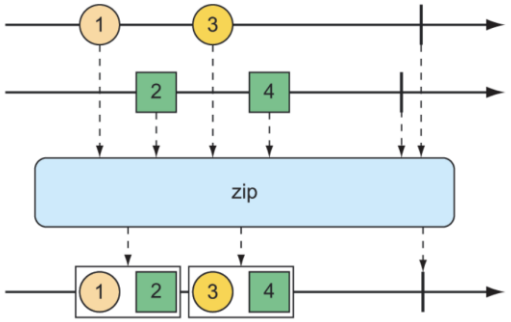


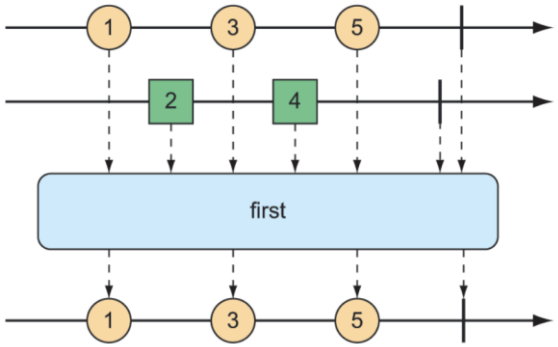


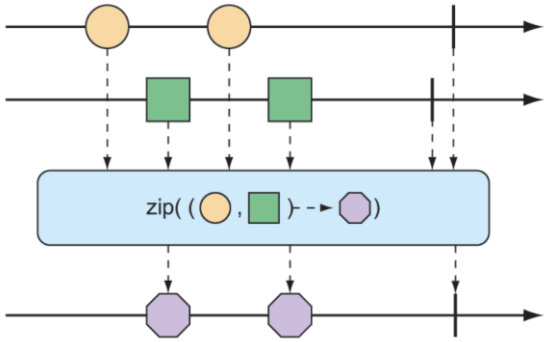
组合Flux流

- mergeWith
- zip
- zip，提供合并函数
- first



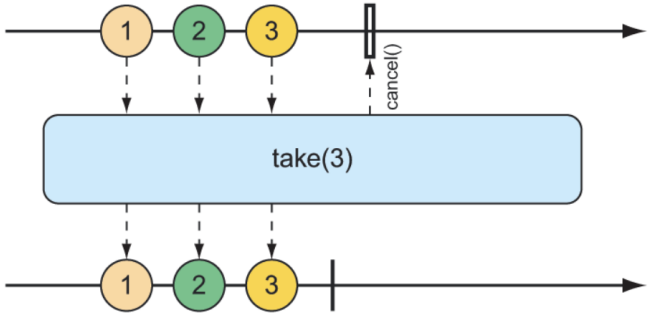


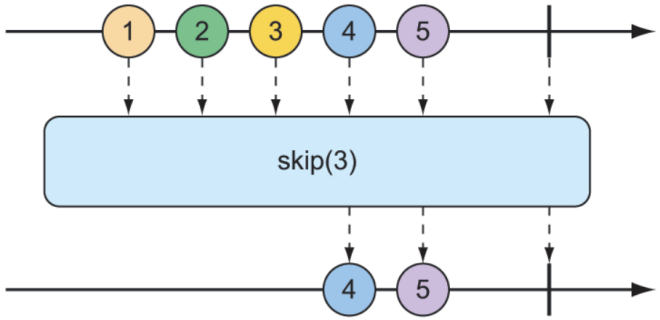


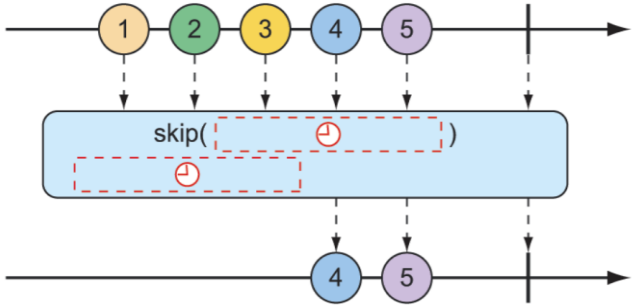


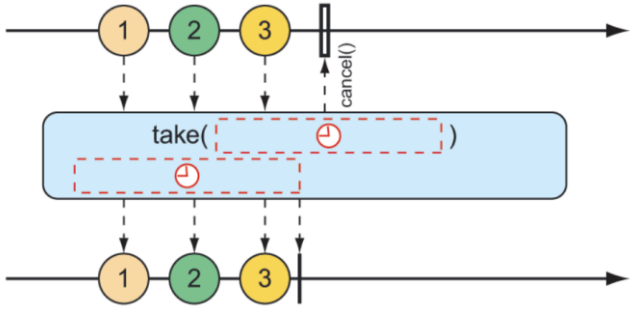
过滤Flux流1

- skip指定个数
- skip指定时长
- take指定个数
- take指定时长



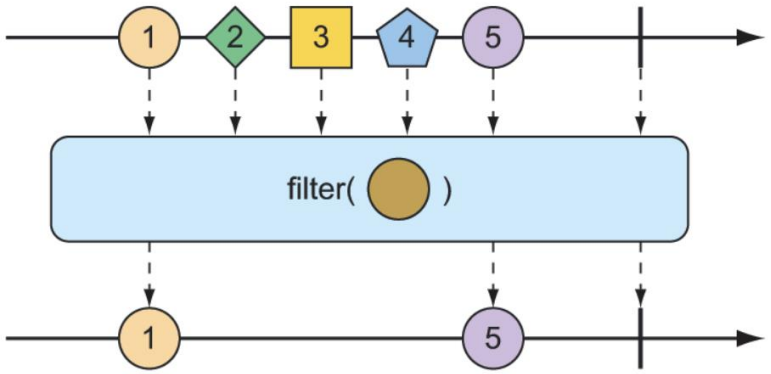


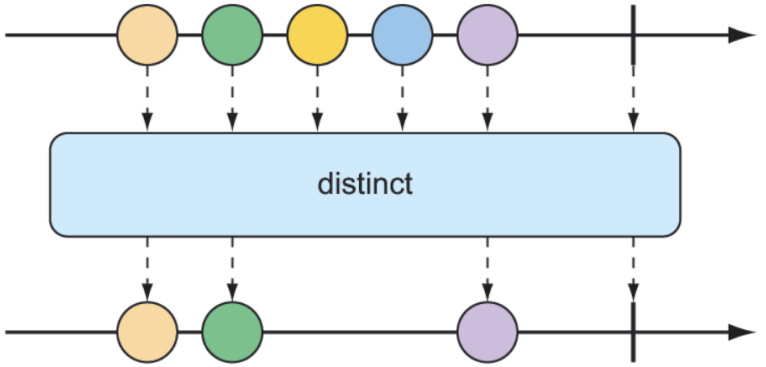




过滤Flux流2

- filter，需要提供Predicate
- distinct，只发布源Flux中尚未发布过的数据项

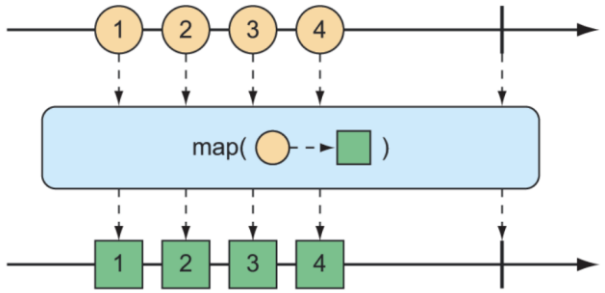


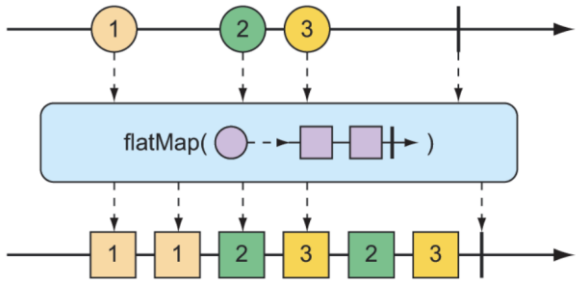


转换Flux流1

- map，同步
- flatMap，异步

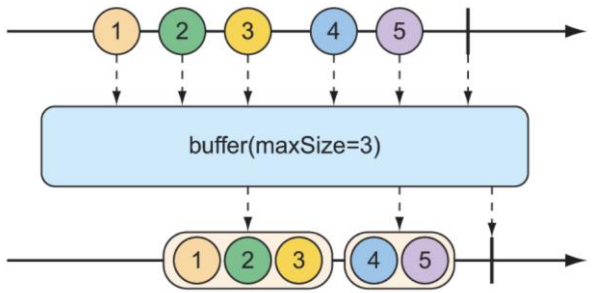
- 并发模型（Schedulers方法）
  - ✓ .immediate()
  - ✓ .single()
  - ✓ .newSingle()
  - ✓ .elastic()
  - ✓ .parallel()

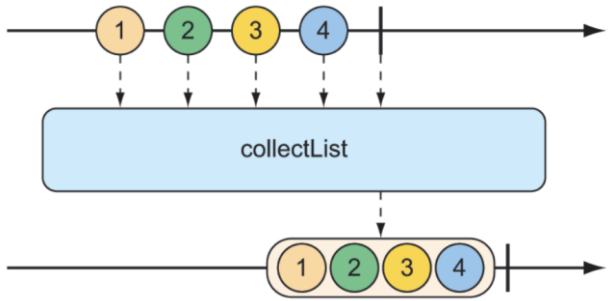


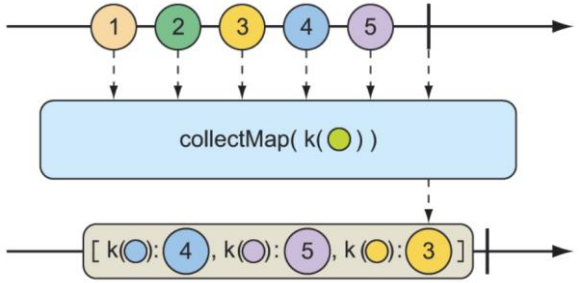


转换Flux流2

- buffer，缓冲数据，bufferAndFlatMap
- collectList，同：buffer不带参数则缓冲所有数据到列表
- collectMap，需要提供生成key的函数

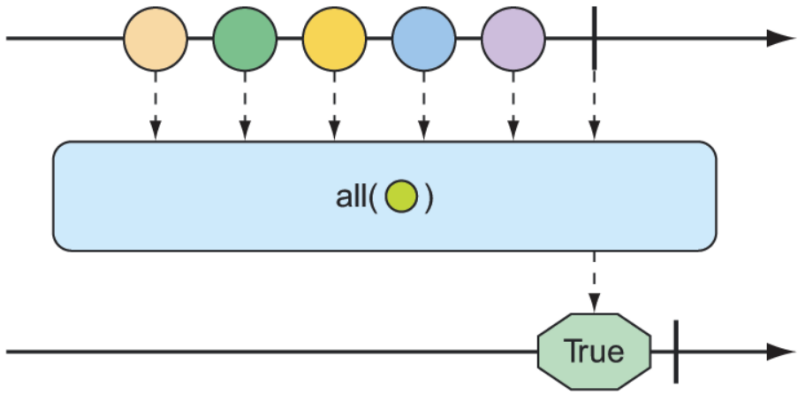


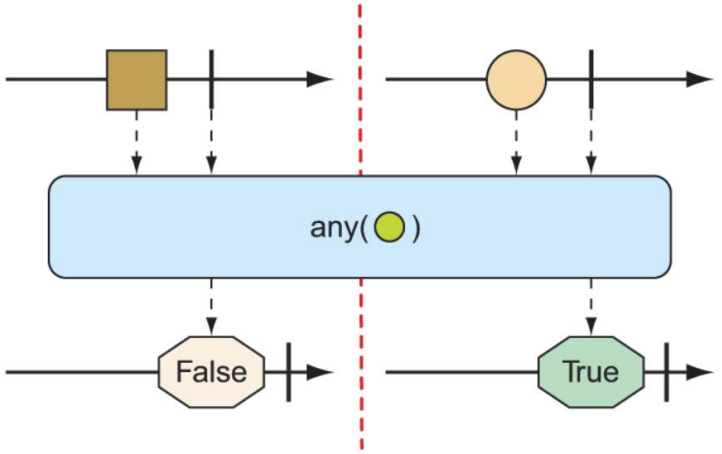




对流执行逻辑操作

- all，需要提供Predicate函数，注意返回类型Mono<Boolean>
- any，需要提供Predicate函数，注意返回类型Mono<Boolean>





谢谢观看！