# 第10章　理解反应式编程

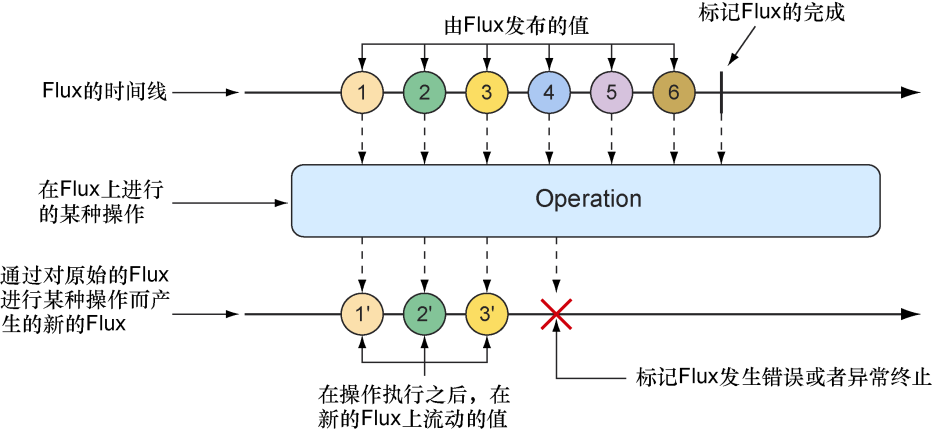


图10.1　描绘Flux基本流程的弹珠图

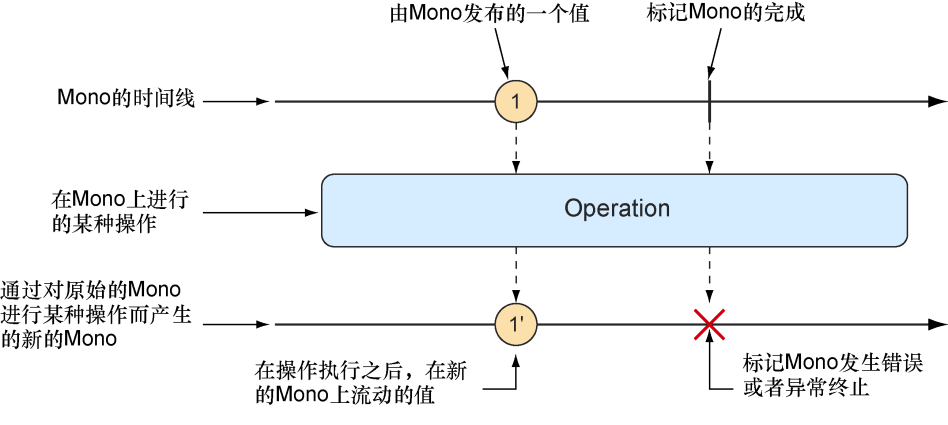


图10.2　描绘Mono基本流程的弹珠图

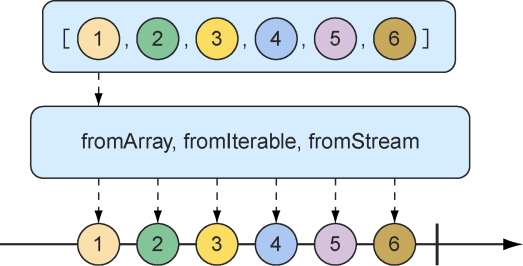


图10.3　可以根据数组、Iterable 或者Java Stream创建Flux

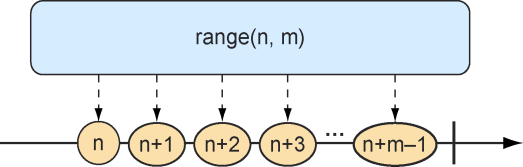


图10.4　从区间创建的Flux会以类似计数器的方式发布消息

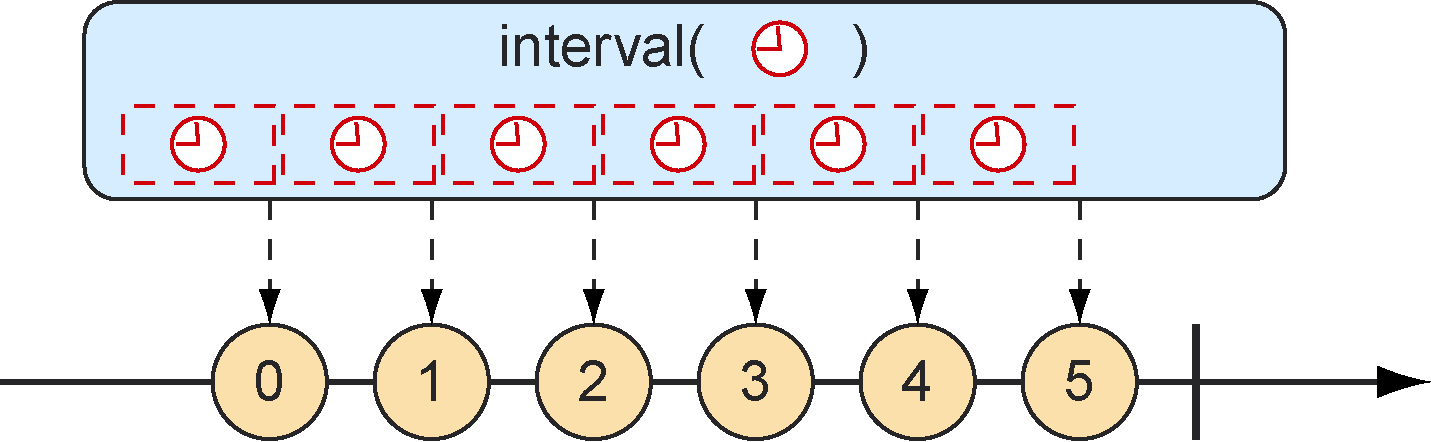


图10.5　根据指定间隔创建的Flux会周期性地发布条目

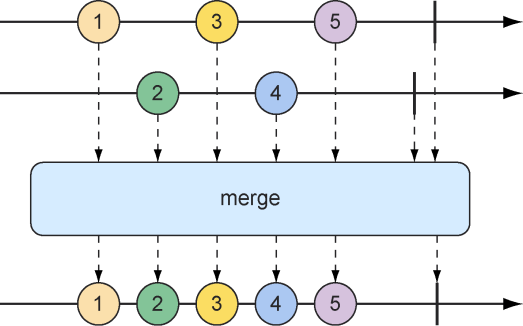


图10.6　合并两个Flux流（它们的消息将会交错合并为一个新的Flux）

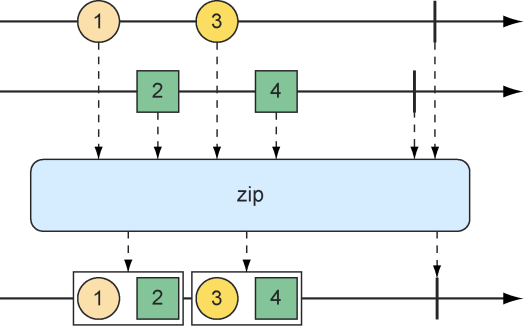


图10.7　通过zip()方法合并两个Flux流

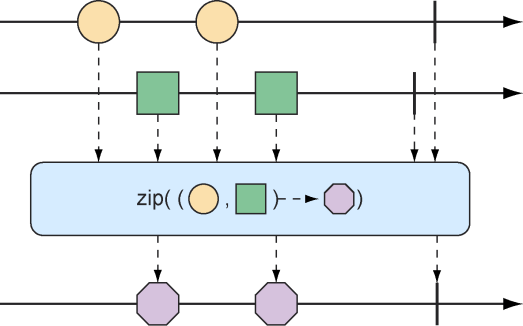


图10.8　zip操作的另一种形式（从每个传入Flux中各取一个元素，  
然后创建消息对象，并产生这些消息组成的Flux）

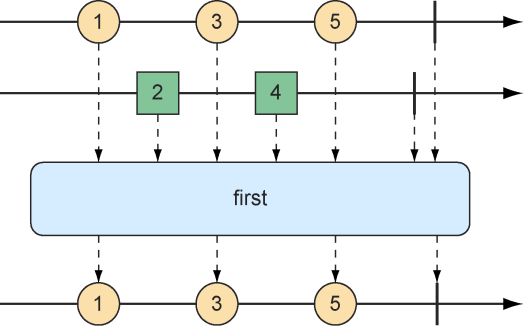


图10.9　first操作将会选择第一个发布消息的Flux并只发布该Flux的值

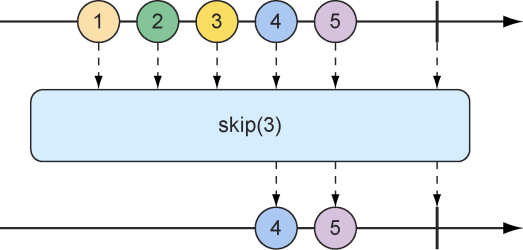


图10.10　skip操作跳过指定数目的消息并将剩下的消息继续在结果Flux上进行传递

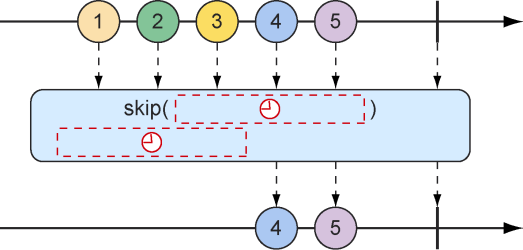


图10.11　skip操作的另一种形式

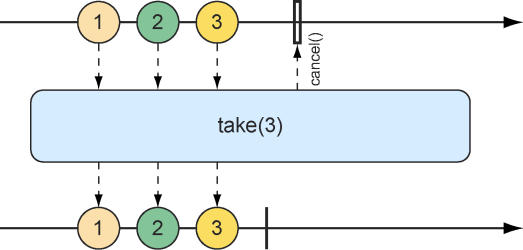


图10.12　take操作只发布传入Flux中前面指定数目的数据项

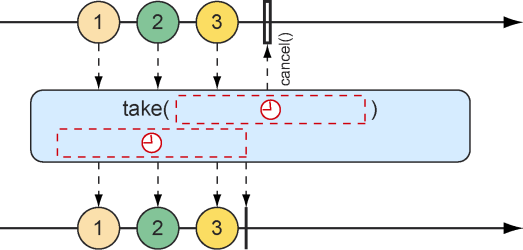


图10.13　take操作的另一种形式（在指定的时间过期之前，一直将消息传递给结果Flux）

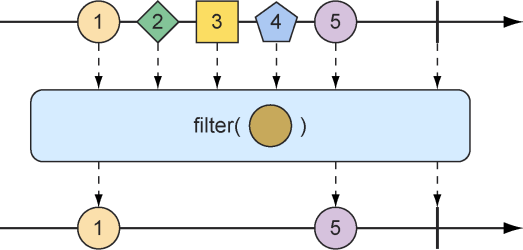


图10.14　可以对传入的Flux进行过滤，这样结果Flux将只会发布满足指定Predicate的消息

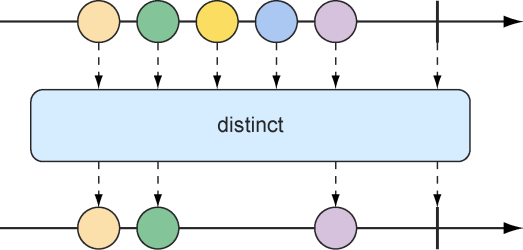


图10.15　distinct操作将会过滤掉重复的消息

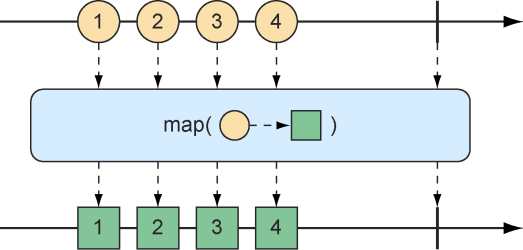


图10.16　map 操作将传入的消息转换为结果流上的新消息

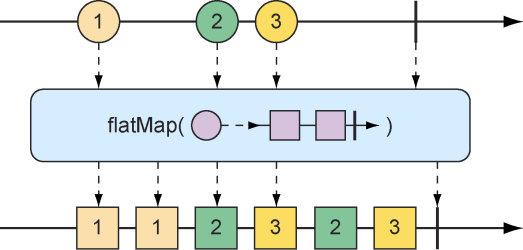


图10.17　flatMap操作使用一个中间的Flux来实现异步转换

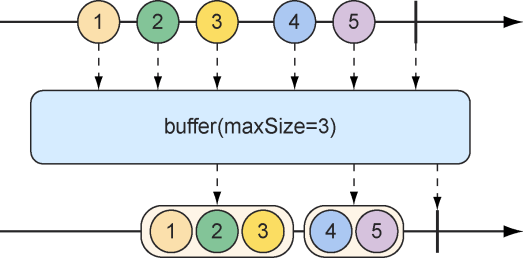


图10.18　buffer操作会产生一个新的包含列表Flux（具备最大长度限制的列表，  
包含从传入的Flux中收集来的数据）

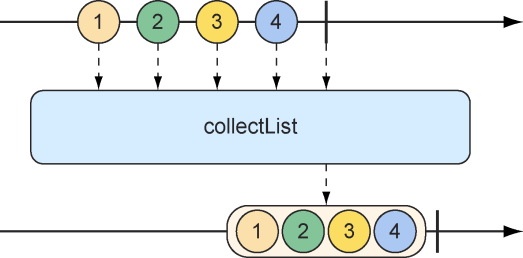


图10.19　collectList操作将产生一个包含传入Flux发布的所有消息的Mono

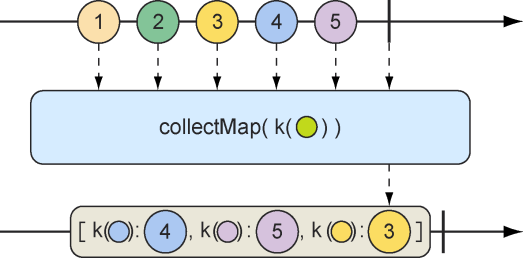


图10.20　collectMap操作将会产生一个Mono（包含了由传入Flux所发出的消息产生的Map，这个Map的key是从传入消息的某些特征衍生而来的）

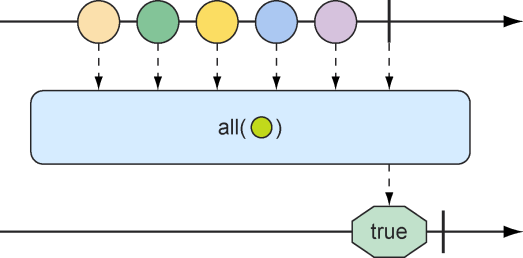


图10.21　可以使用all()方法来确保Flux中的所有消息都满足某些条件

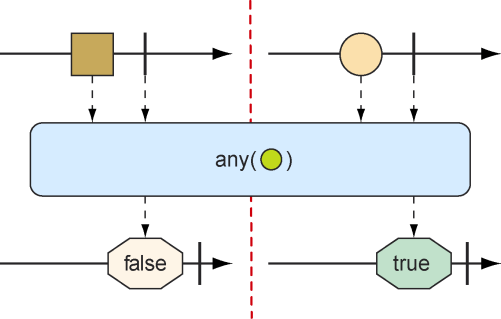


图10.22　可以使用any()方法来确保Flux中至少有一个消息满足某些条件