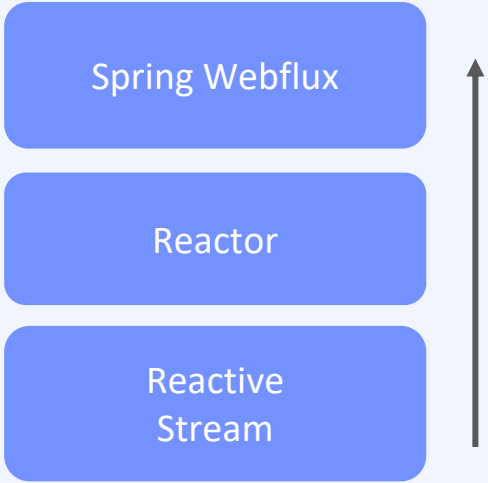
Webflux

5 Reactor 이론

Reactive Stream

(specification)

Reactive Stream
(specification)



Reactor

Reactive Stream

Reactive Streams

Reactive Streams is an initiative to provide a standard for asynchronous stream processing with non-blocking back pressure. This encompasses efforts aimed at runtime environments (JVM and JavaScript) as well as network protocols.

JDK9 `java.util.concurrent.Flow`

The interfaces available in JDK >= 9 java.util.concurrent.Flow, are 1:1 semantically equivalent to their respective Reactive Streams counterparts. This means that there will be a migratory period, while libraries move to adopt the new types in the JDK, however this period is expected to be short - due to the full semantic equivalence of the libraries, as well as the Reactive Streams <-> Flow adapter library as well as a TCK compatible directly with the JDK Flow types.

Read this if you are interested in learning more about `Reactive Streams` for the JVM.

The Problem

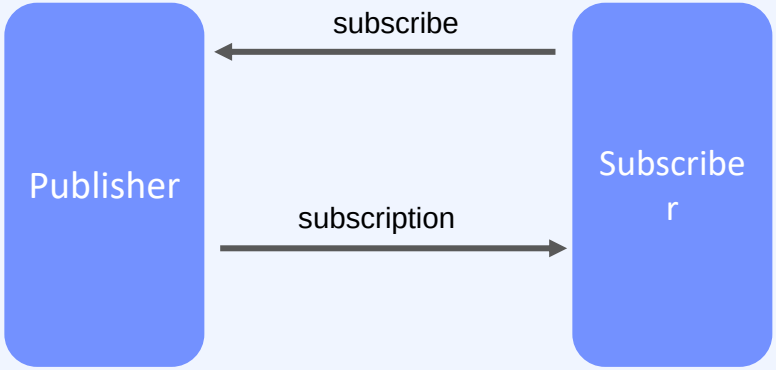
Handling streams of data—especially "live" data whose volume is not predetermined—requires special care in an asynchronous system. The most

https://www.reactive-streams.org/

Reactor

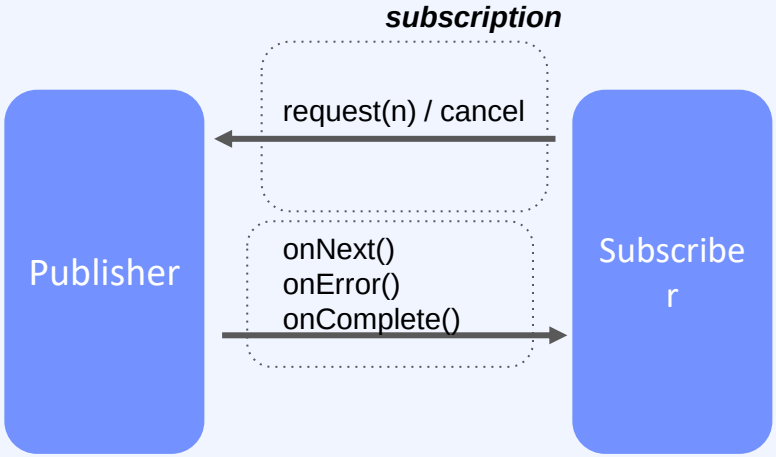
구성요소

1. **stream**
    a. publisher
    b. subscriber
    c. subscription
    d. processor
2. asynchronous
3. back pressure



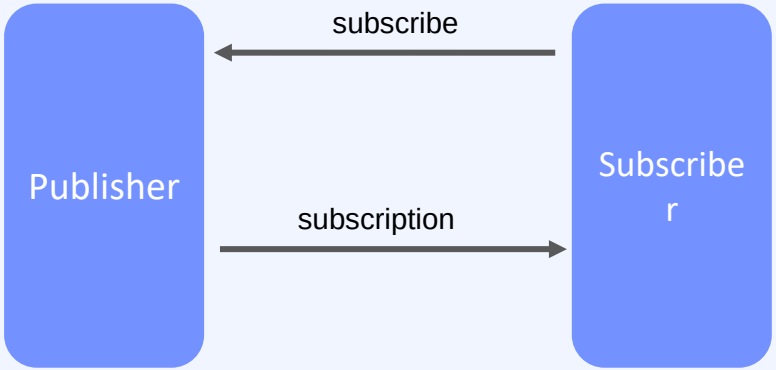
구성요소

1. **stream**
   a. publisher
   b. subscriber
   c. subscription
   d. processor
2. asynchronous
3. back pressure



Reactor

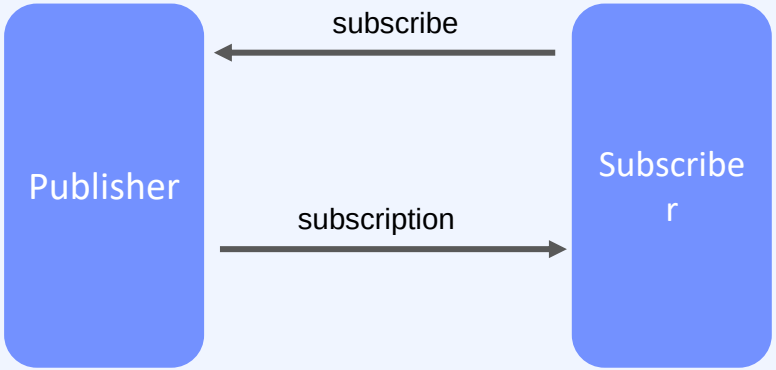
구성요소




```
public interface Publisher<T> {
    public void subscribe(Subscriber<? super T> s);
}
```

Reactor

구성요소

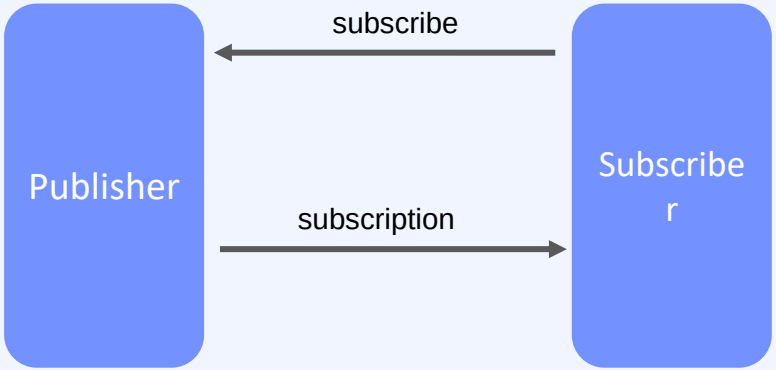



```
public interface Publisher<T> {
    public void subscribe(Subscriber<? super T> s);
}
```

```
public interface Subscriber<T> {
    public void onSubscribe(Subscription s);
    public void onNext(T t);
    public void onError(Throwable t);
    public void onComplete();
}
```

Reactor

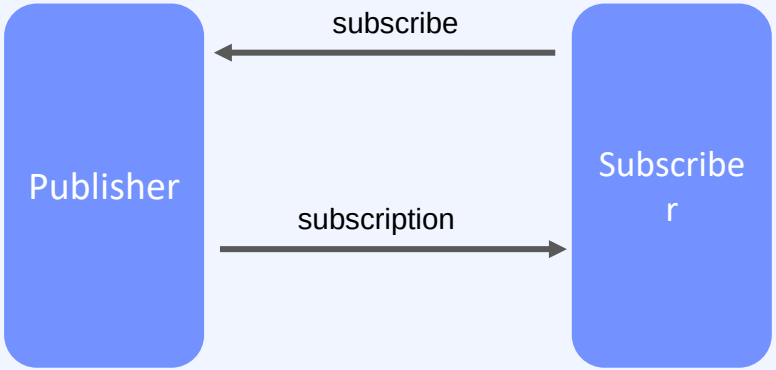
구성요소




```
public interface Subscription {
    public void request(long n);
    public void cancel();
}
```

Reactor

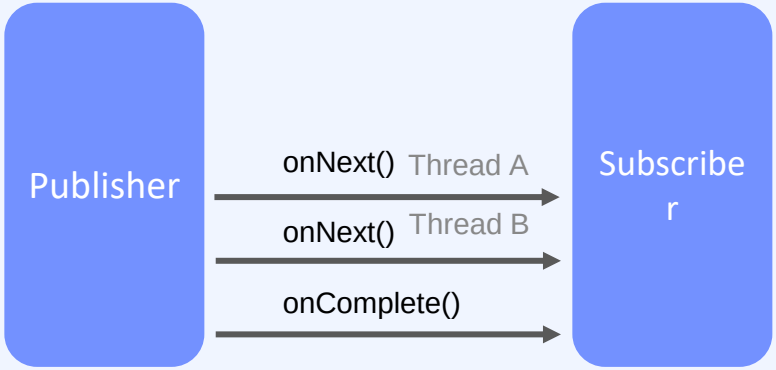
구성요소




```
public interface Processor<T, R> extends Subscriber<T>, Publisher<R> {
}
```

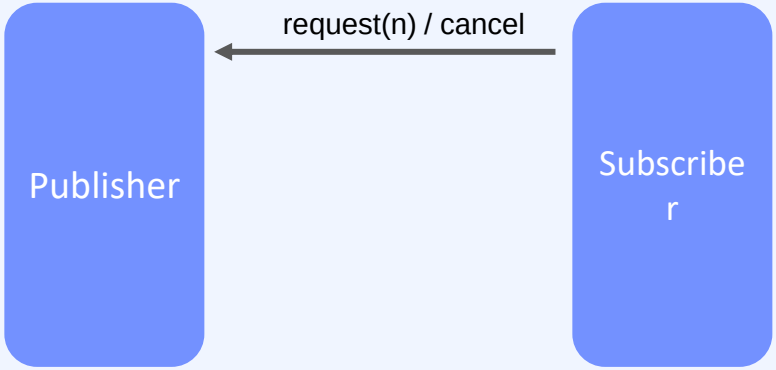
주요 특징

1. stream
   a. publisher
   b. subscriber
   c. subscription
   d. processor
2. **asynchronous**
3. back pressure



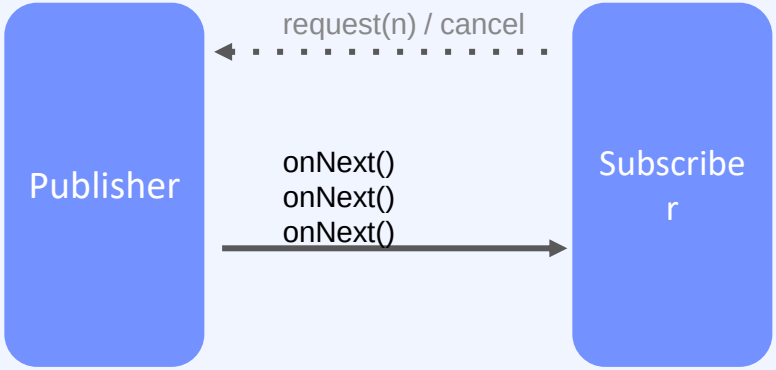
주요 특징

1. stream
    a. publisher
    b. subscriber
    c. subscription
    d. processor
2. asynchronous
3. **back pressure**



주요 특징

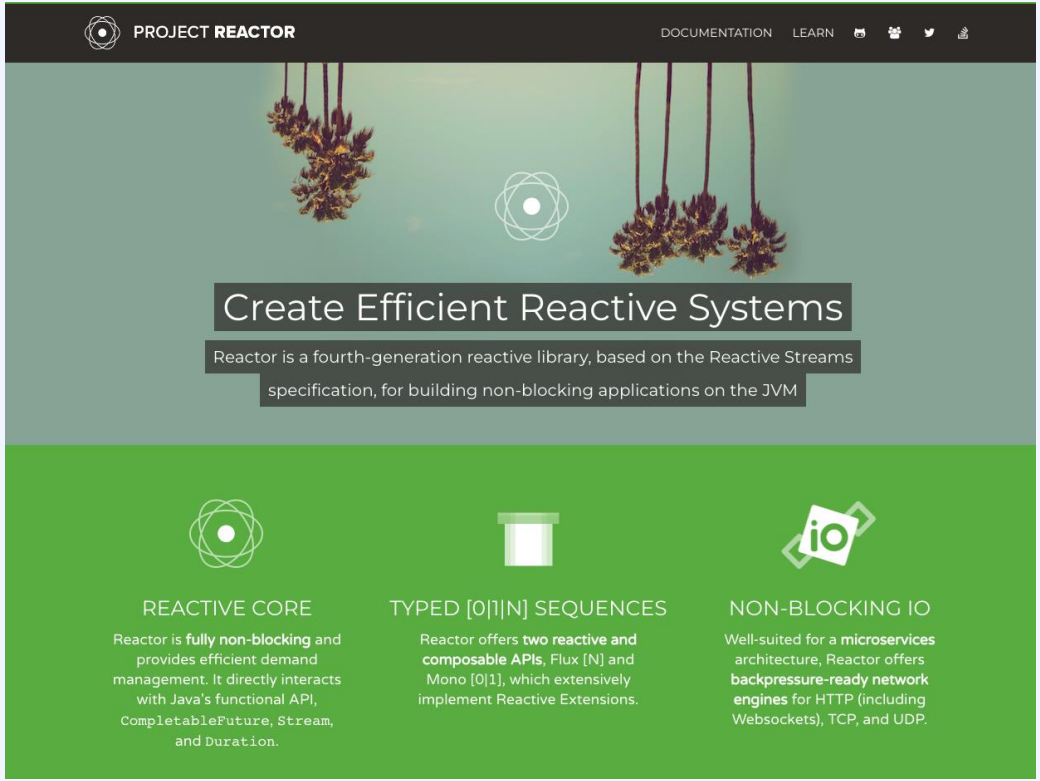
1. stream
    a. publisher
    b. subscriber
    c. subscription
    d. processor
2. asynchronous
3. back pressure



Project reactor

Spring Webflux

reactor



https://projectreactor.io/

마무리

1. Reactive Stream
    a. Asynchronous Stream processing
    b. Nonblocking backpressure
2. Publisher, Subscriber, Subscription, Processor
3. Reactor