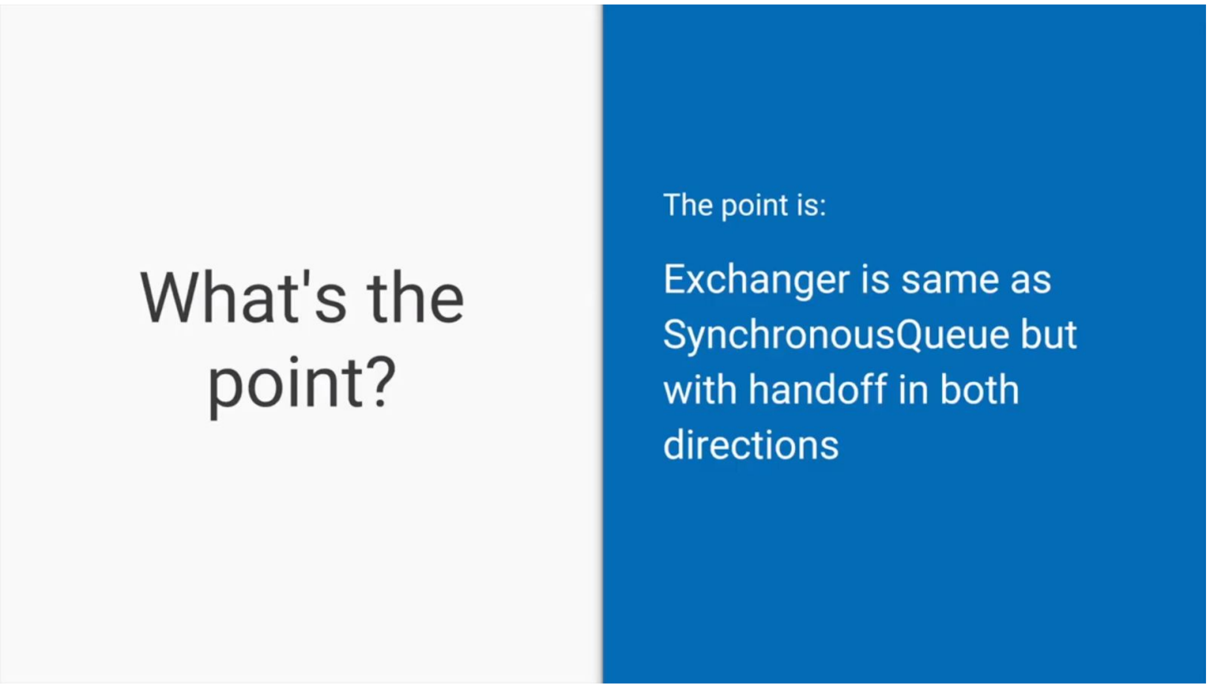
Exchanger Class in Java

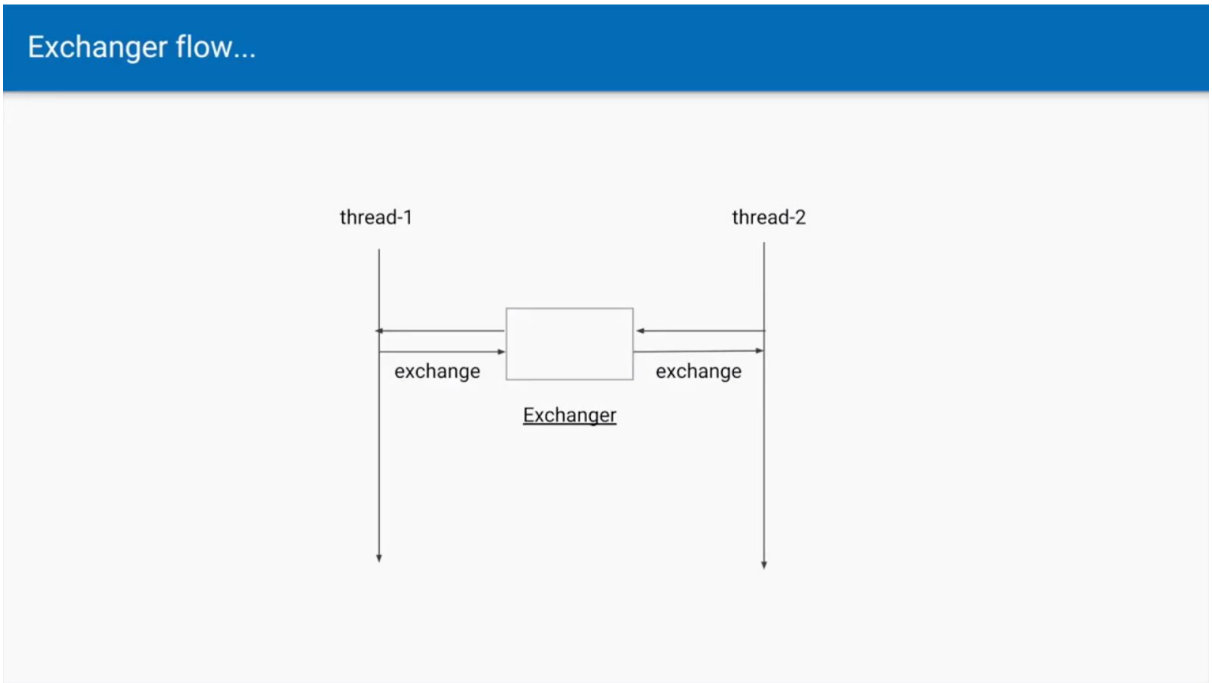
Exchanger is the most interesting **synchronization class** of Java. It facilitates the **exchange of elements between a pair of threads by creating a synchronization point**. It simplifies the exchange of data between two threads. Its operation is simple: it simply waits until two separate threads call its **exchange()** method. When that occurs, it exchanges the data supplied by the threads. It can also be viewed as a **bidirectional SynchronousQueue**. It is a generic class that is declared as below.



What's the point?

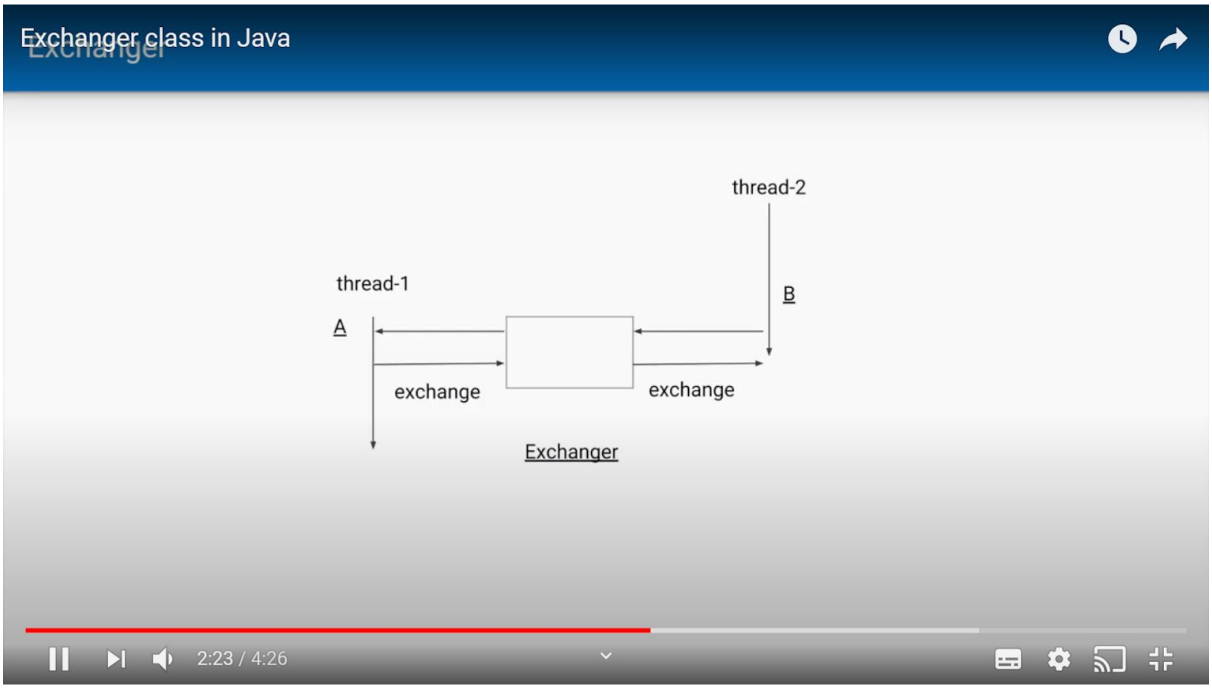
The point is:

Exchanger is same as SynchronousQueue but with handoff in both directions

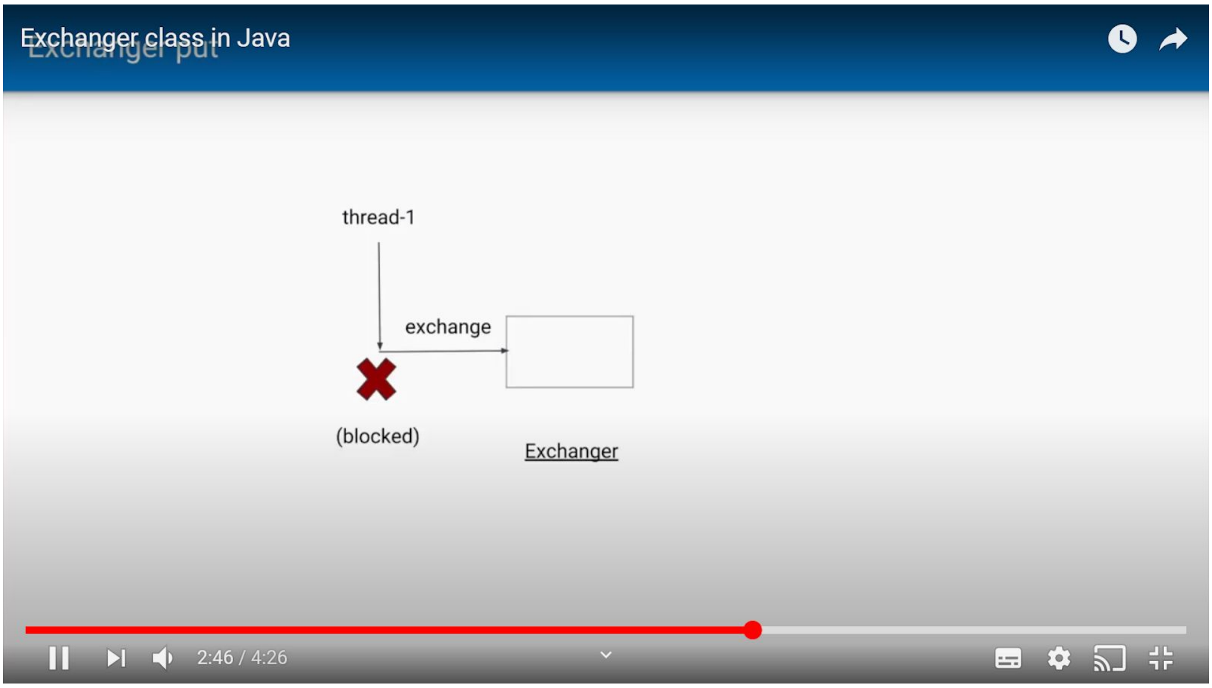


Exchanger flow...

Similar Synchronous Queue, if there is just one thread which wants to exchange, then it is blocked until another thread is there which calls the exchange method. This is similar to Synchronous queue as in that also, if there is no consumer, the producer does not put anything in the queue. It is just that in Exchanger, instead of producing and consuming, the 2 threads are exchanging data.



Exchanger class in Java



Class Syntax:

```
Exchanger<V>
```

Here, V specifies the type of data being exchanged.

Methods:

exchange(V x)– When invoked this function causes the current thread to suspend its execution and wait for another thread to call its exchange method. When another thread calls its exchange method, the threads exchange their data and the execution resumes.

Syntax:

```
public V exchange(V x)

throws InterruptedException
```

exchange(V x, long timeout, TimeUnit unit)– When invoked this function causes the current thread to suspend its execution and wait for another thread to call its exchange method. When another thread calls its exchange method, the threads exchange their data and the execution resumes. The thread waits only for the duration specified by the timeout argument and in case if timeout duration elapses, a TimeoutException is thrown.

Syntax:

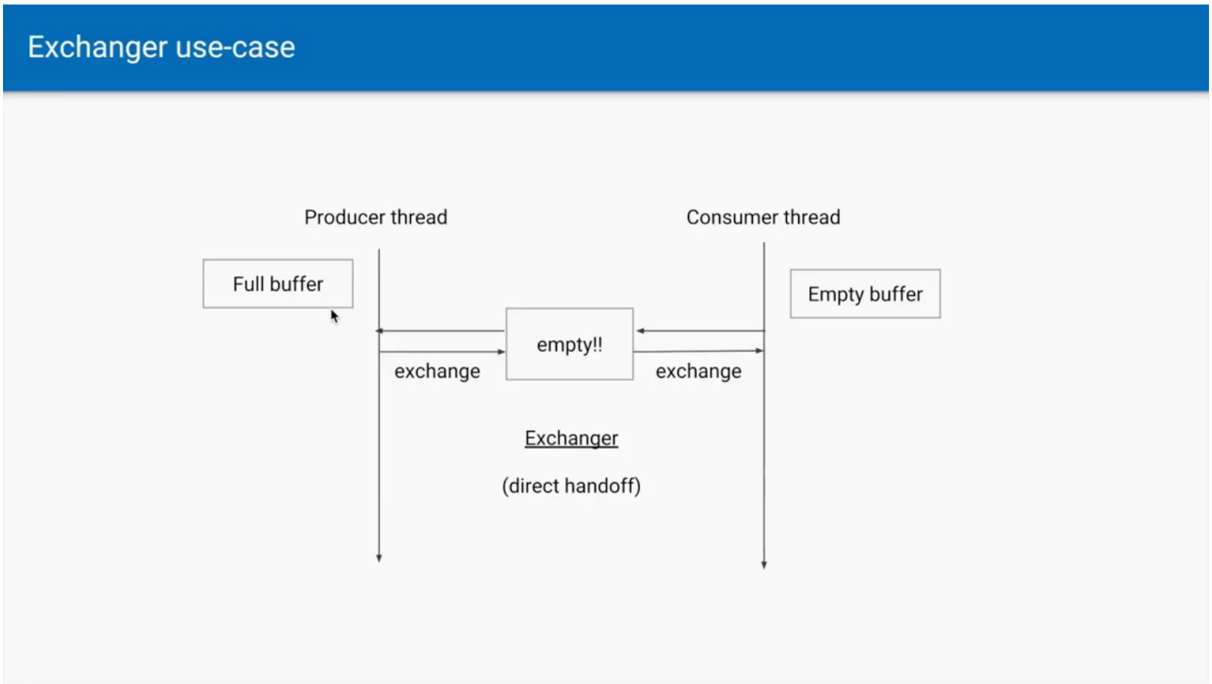
```
public V exchange(V x, long timeout, TimeUnit unit)
throws InterruptedException, TimeoutException
```

USECASE:

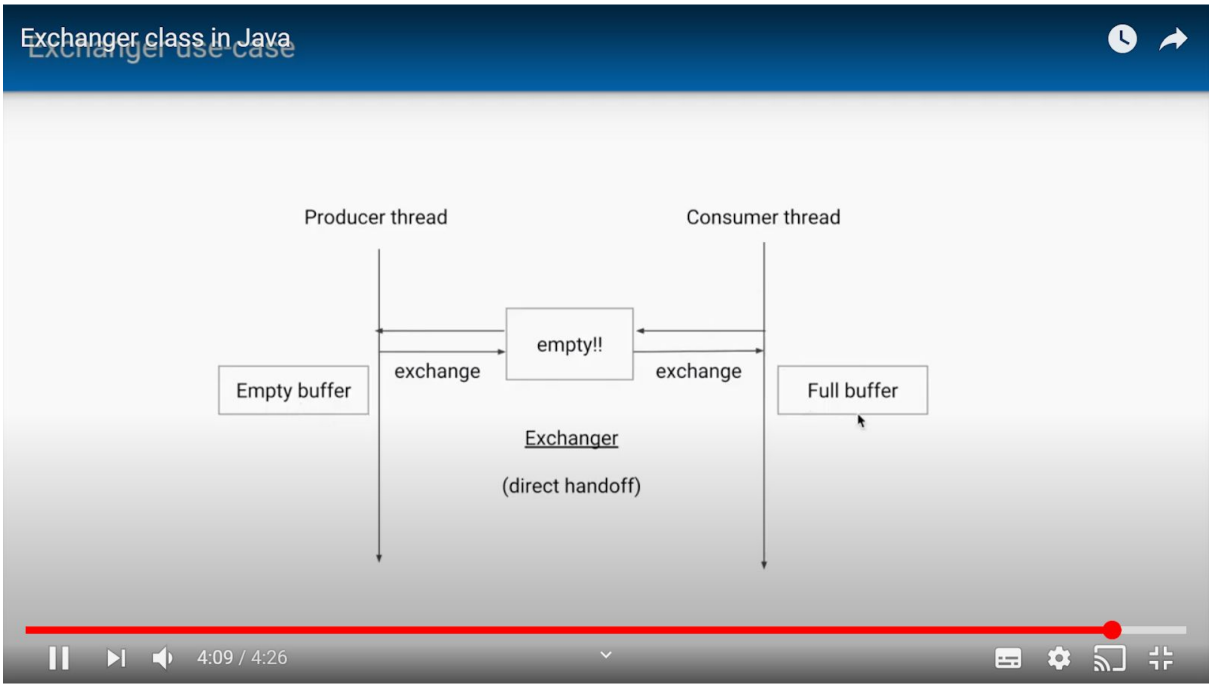
Exchanger class in Java

Use case...

Producer and Consumer exchanging full and empty buffer



Exchanger use-case



In this case, when the producer thread's buffer is full and consumer's is empty, they can exchange so that consumer can continue with further processing and producer can continue with further filling up the buffer again.