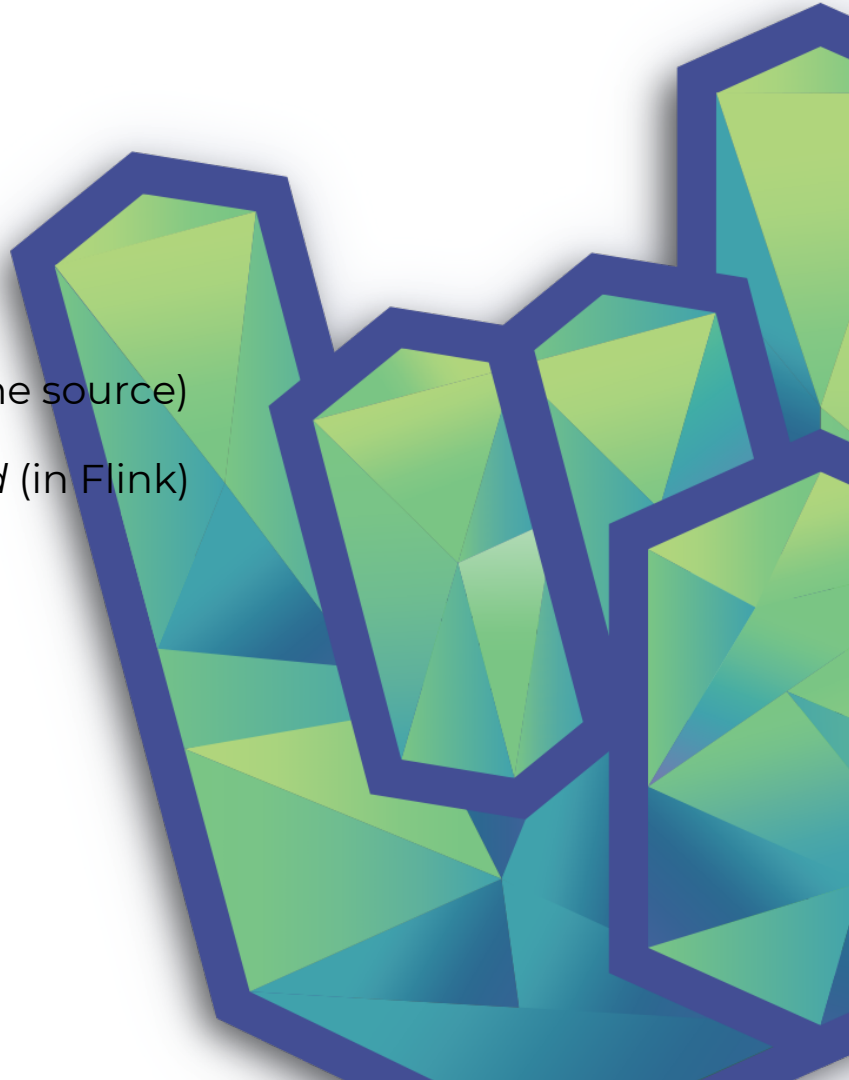
Time-Based Transformations

Time

Event time = when the event was *created* (at the source)

Processing time = when the event was *handled* (in Flink)



Time Is Everything

Event time = when the event was *created* (at the source)

- we don't care about Flink's internal time
- we need to care about handling late data (watermarks)
- same events on different runs produce the same results

```
val groupedEventsByWindow = shoppingCartEvents
  .assignTimestampsAndWatermarks(
    WatermarkStrategy
      .forBoundedOutOfOrderness(java.time.Duration.ofMillis(500)) // max delay < 500 millis
      .withTimestampAssigner(new SerializableTimestampAssigner[ShoppingCartEvent] {
        override def extractTimestamp(element: ShoppingCartEvent, recordTimestamp: Long) =
          element.time.toEpochMilli
      })
  )
```

Processing time = when the event was *handled* (in Flink)

- we don't care about when the event was created
- different runs produce different results

Custom Watermarks

Powerful APIs to emit watermarks

- when handling an element
- periodically

```
class BoundedOutOfOrdernessGenerator(maxDelay: Long) extends WatermarkGenerator[ShoppingCartEvent] {
  var currentMaxTimestamp: Long = 0L

  // maybe emit watermark on a particular event
  override def onEvent(event: ShoppingCartEvent, eventTimestamp: Long, output: WatermarkOutput) =
    currentMaxTimestamp = Math.max(currentMaxTimestamp, event.time.toEpochMilli)


  // Flink can also call onPeriodicEmit regularly - maybe emit a watermark at these times
  override def onPeriodicEmit(output: WatermarkOutput) =
    output.emitWatermark(new Watermark(currentMaxTimestamp - maxDelay - 1))
}
```

```
env.getConfig.setAutoWatermarkInterval(1000L) // call onPeriodicEmit every 1s
```

Flink rocks