

- `evictAfter()`方法：定义要在窗口函数之后应用的移除逻辑。在应用窗口函数之前移除的元素不会被窗口函数处理。

Flink 默认提供了以下 3 个移除器。

- `CountEvictor`：从窗口中保留用户指定数量的元素，并从窗口缓冲区的开头丢弃其余的元素。
- `DeltaEvictor`：采用 `DeltaFunction` 和阈值，计算窗口缓冲区中最后一个元素与其余每个元素之间的增量，并删除增量大于或等于阈值的元素。
- `TimeEvictor`：采用以毫秒为单位的间隔作为参数。对于给定的窗口，它将在其元素中找到最大时间戳 `max_ts`，并删除所有时间戳小于 `max_ts` 的元素。

在默认情况下，所有预先实现的移除程序均在窗口函数应用之前应用其逻辑。



指定移除器可以防止任何预聚合，因为在应用计算之前必须将窗口中的所有元素传递给移除器。

Flink 不保证窗口内元素的排序。移除器从窗口中删除的元素不一定是最先到达的。

7.2.13 处理迟到数据

在使用“事件时间”窗口时，可能会发生元素迟到的情况，具体的表现是，Flink 用于跟踪“事件时间”进度的水位线已经超过了元素所属窗口的结束时间戳。

在默认情况下，当水位线超过窗口末端时将删除迟到的元素。但是，Flink 允许为窗口算子指定最大允许延迟——在删除指定元素之前可以延迟的时间，其默认值为 0。

在使用某些触发器时，延迟但未掉落的元素可能会导致窗口再次触发，事件时间触发器就存在这种情况。

Flink 保持窗口的状态，直到允许的延迟过期为止。一旦发生这种情况，Flink 将删除该窗口并删除其状态。

在使用全局窗口分配器时，不需要考虑任何数据延迟，因为全局窗口的结束时间戳是 `Long.MAX_VALUE`。

可以使用 `allowedLateness()`方法指定延迟，其使用方法如下所示：

```
// 加载或创建源数据
DataStream<T> input = ...;
// 转换数据
input
    // 键控流转换算子
    .keyBy(<key selector>)
    // 窗口转换算子
```