Flink:

# 什么是Flink

# [Flink 应用场景](https://flink.apache.org/usecases.html)

* 1. flink功能

support for stream and batch processing 对流和批处理的支持

sophisticated state management 复杂状态管理

event-time processing semantics 事件-时间处理语义

exactly-once consistency guarantees for state 状态的仅此一次的持续性保证

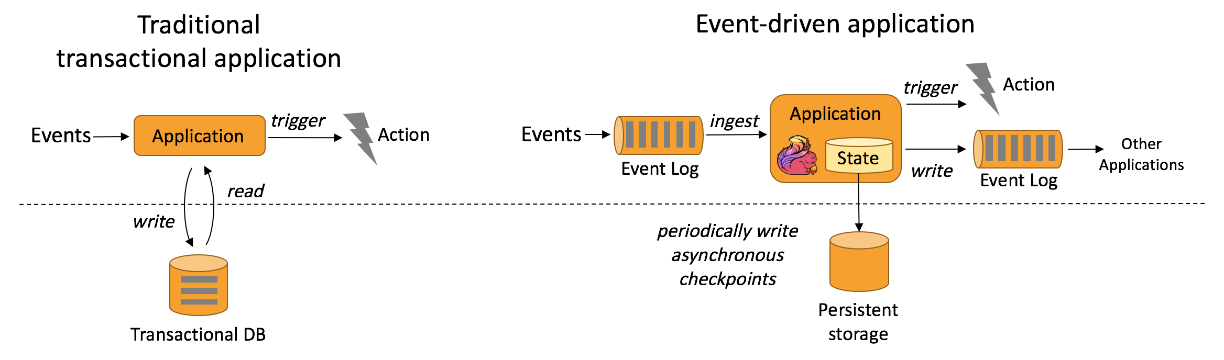
* 1. Flink 最常见的应用场景：
     1. Event-driver Application (事件驱动)

1. 什么是事件驱动型应用

事件驱动型应用是一种有状态的应用，他从一个或多个事件流提取数据，并根据提取到来的数据触发计算、状态更新或者外部action。

事件驱动型应用是一次计算和数据存储层分离的传统应用设计的演变。在此架构中，应用程序能从远程传统交易数据库中读取数据也能将数据持久化到传统库中。

相反，事件驱动型应用是基于有状态的流处理应用，在此设计中，数据和计算协同定位，这将产生局部的数据（基于内存或磁盘）访问。通过阶段性地将将 将checkpoint写入远程持久化存储来实现容错。以下图片说明了传统的应用程序体系结构和事件驱动型应用之间的区别。



1. 事件驱动型应用的优势

本地数据访问使其具有更高的吞吐和更低的延迟

定期地异步、增量式远程持久化checkpoint存储，对正常事件处理影响较小。

减少数据的改变或服务扩容时的协调工作

1. Flink如何支持事件驱动型应用程序的？

事件驱动型应用会受制于底层流处理系统对时间和状态的把控能力。

* + 1. Data Analytics Application（数据分析）
    2. Data Pipeline Application （）

Event-driver Application (事件驱动)

欺诈检测

异常检测

基于规则的告警

业务流程监控

Web应用程序（社交网络）

CEP：复杂事件处理

Data Analytics Application（数据分析）

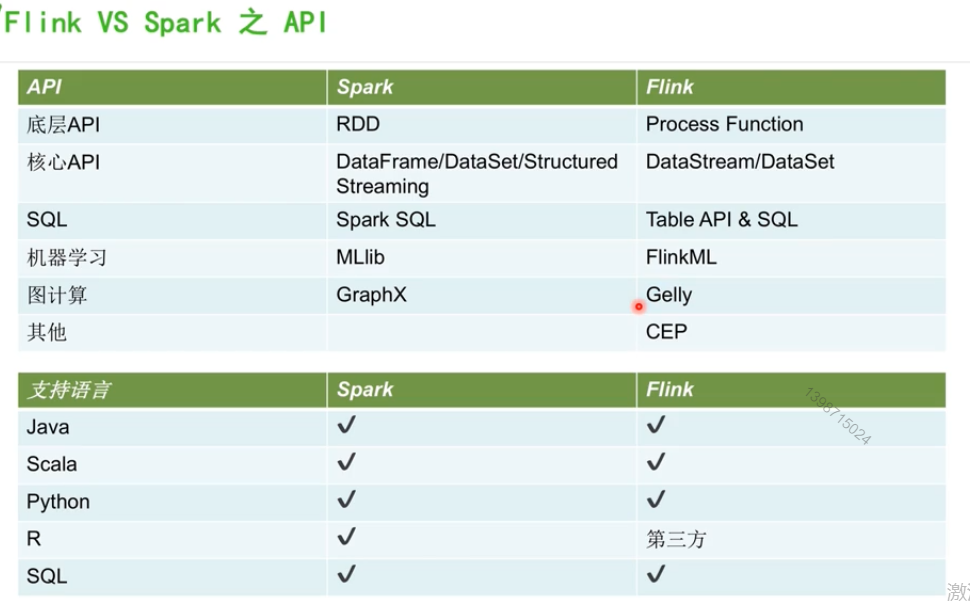
周期性查询

连续性查询

Data Pipeline Application （）

Periodic ETL

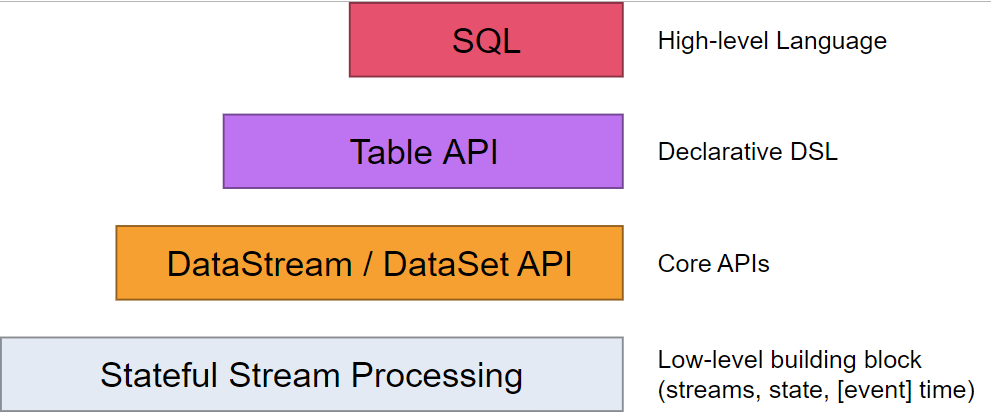
Data Pipeline



# [Flink docs](https://ci.apache.org/projects/flink/flink-docs-release-1.9/)

**Concepts:**

1. Dataflow Programming Model
   1. Levels of Abstraction

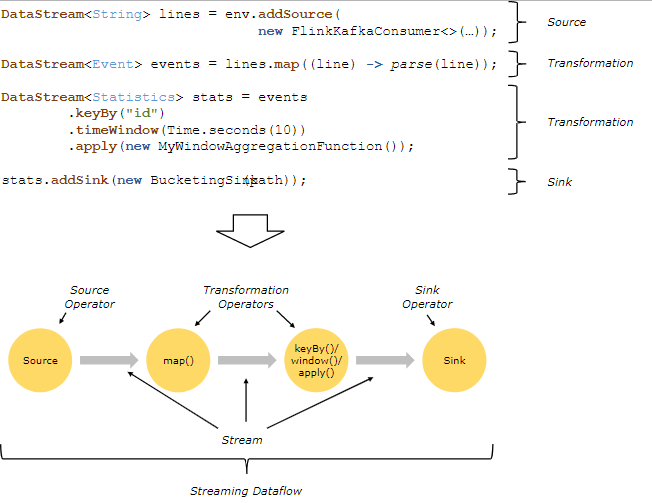


* 1. Programs and Dataflows

流和转换是Flink的基石（请注意：flink的DataSet API中使用的DataSets其内部也是流）。概念上，stream就是数据记录流，transformation 就是一个把一个或多个输入流streams作为输入，然后把产生的一个或多个输出流作为结果的算子。

程序执行时，Flink程序被映射成streaming dataflows，streaming dataflows 由 streams 和transformation 算子组成。每个dataflow以一个或多个source开始而以一个或多个sink结束。Dataflow有点类似于任意的有向无环图（DAG图）。尽管环的特定形式是迭代结构，但是，为了简单起见，大多数情况下我们掩盖这种情况。

通常程序的转换和数据流间的算子是意义对应的。然而，有时，一个转换也有可能有多个转换算子组成。



* 1. Parallel Dataflows

Flink程序的本质还是并行和分布式，在程序运行时，一个stream有一个或多个流分区（stream partitions），每个operator算子也有一个或多个operator子任务（operator subtasks）。算子的subtasks 间彼此独立，并可能在不同的机器或容器上执行不同的线程。

算子的subtasks的数量就是对应算子的并行度。而一个stream的并行度也取决于产生它的算子。同一个程序的不同算子也可能有不同程度的并行度

流可以在两个算子间以一对一（或转发）或者重新分配的模式传输数据，

一对一的流：

重新分配的流：

* 1. Windows
  2. Time
  3. Stateful Operations
  4. Checkpoints for Fault Tolerance
  5. Batch on Streaming