Assignment 9

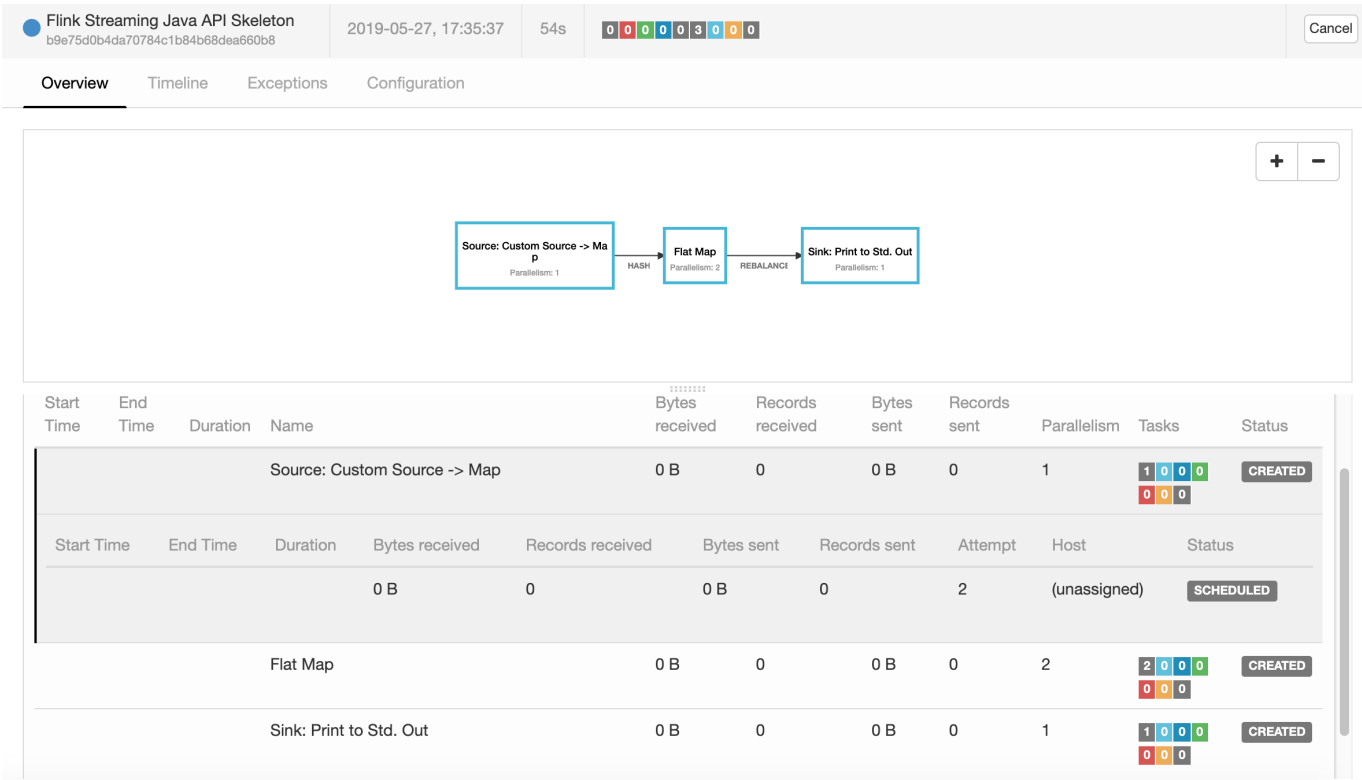
I. Cause a failure and set a restart strategy

Task 1

By killing the TaskManager:

```
Xivid@Xivids-MBP ~ ps aux | grep TaskManager                                    ✓ 17:34:47
Xivid            32214   2.9  2.1  6987352 359424 s000  S     5:33PM   0:08.58 /usr/bin/java -XX:+
UseG1GC -Xms922M -Xmx922M -XX:MaxDirectMemorySize=8388607T -Dlog.file=/Users/Xivid/repo/eth-dspa-2
019/project/bin/flink-1.8.0/log/flink-Xivid-taskexecutor-2-Xivids-MBP.local.log -Dlog4j.configurat
ion=file:/Users/Xivid/repo/eth-dspa-2019/project/bin/flink-1.8.0/conf/log4j.properties -Dlogback.c
onfigurationFile=file:/Users/Xivid/repo/eth-dspa-2019/project/bin/flink-1.8.0/conf/logback.xml -cl
asspath /Users/Xivid/repo/eth-dspa-2019/project/bin/flink-1.8.0/lib/log4j-1.2.17.jar:/Users/Xivid/
repo/eth-dspa-2019/project/bin/flink-1.8.0/lib/slf4j-log4j12-1.7.15.jar:/Users/Xivid/repo/eth-dspa
-2019/project/bin/flink-1.8.0/lib/flink-dist_2.12-1.8.0.jar::: org.apache.flink.runtime.taskexecut
or.TaskManagerRunner --configDir /Users/Xivid/repo/eth-dspa-2019/project/bin/flink-1.8.0/conf
Xivid            32863   0.0  0.0  4258648    300 s001  R+    5:34PM   0:00.00 grep --color=auto -
-exclude-dir=.bzr --exclude-dir=CVS --exclude-dir=.git --exclude-dir=.hg --exclude-dir=.svn TaskMa
nager
Xivid@Xivids-MBP ~ kill -9 32214                                                ✓ 17:34:51
```

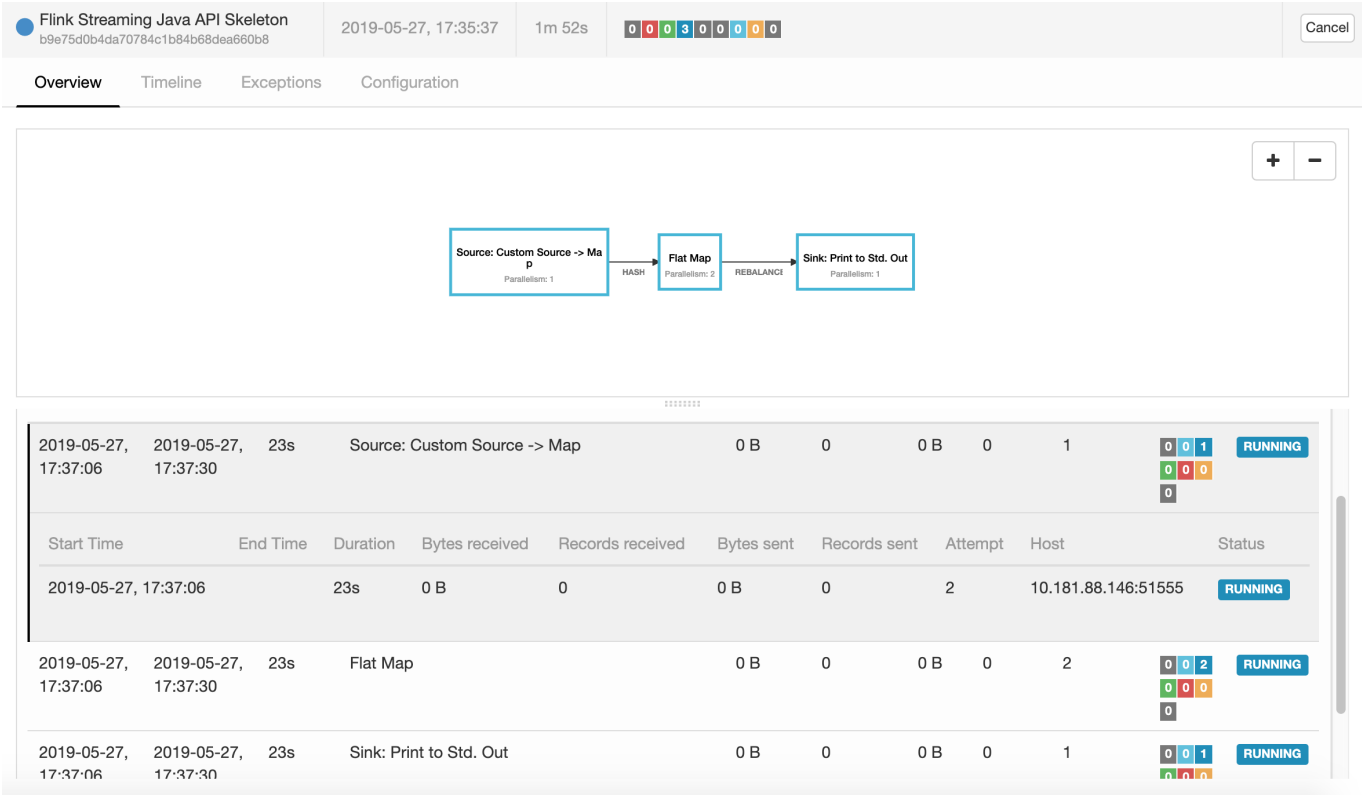
The task status becomes `CREATED`, and `attempt` is incremented:



After starting a new task manager:

```
Xivid@Xivids-MBP ~ cd repo/eth-dspa-2019/project/bin/flink-1.8.0                ✓ 17:34:56
Xivid@Xivids-MBP ~/repo/eth-dspa-2019/project/bin/flink-1.8.0 master ● ? ./bin/taskmanage
r.sh start
Starting taskexecutor daemon on host Xivids-MBP.local.
Xivid@Xivids-MBP ~/repo/eth-dspa-2019/project/bin/flink-1.8.0 master ● ?
```

The status becomes `RUNNING` again, with `attempt = 2`.



Task 2

The output is sinked to `stdout`, which is a file in Flink's `log` folder. Everytime we kill and start a new task manager, we automatically get a new output filename (named by the new task executor's id).

By comparing the following files:

- flink-Xivid-taskexecutor-0-student-net-cx-3057.ethz.ch.out
- flink-Xivid-taskexecutor-1-student-net-cx-3057.ethz.ch.out
- flink-Xivid-taskexecutor-2-student-net-cx-3057.ethz.ch.out

The observation is that file `i+1` contains the full content of file `i` in this header, plus more new results (because the producer is always producing new data).

II. Use managed state and checkpoints

Main function of the streaming application:

```
public static void main(String[] args) throws Exception {
    // set up the streaming execution environment
    final StreamExecutionEnvironment env =
StreamExecutionEnvironment.getExecutionEnvironment();
    RocksDBStateBackend backend = new
RocksDBStateBackend("file:///Users/Xivid/repo/eth-dspa-2019/session-
9/rocks.db", true);
    env.setStateBackend(backend);
    // Set a fixed delay restart strategy with a maximum of 5 restart
attempts
```

```
    // and a 1s interval between retries
    env.setRestartStrategy(RestartStrategies.fixedDelayRestart(5, 1000));

    // Take a checkpoint every 10s
    env.enableCheckpointing(10000);

    Properties kafkaProps = new Properties();
    kafkaProps.setProperty("zookeeper.connect", "localhost:2181");
    kafkaProps.setProperty("bootstrap.servers", "localhost:9092");
    kafkaProps.setProperty("group.id", "test-consumer-group");
    kafkaProps.setProperty("enable.auto.commit", "false");
    // always read the Kafka topic from the start
    kafkaProps.setProperty("auto.offset.reset", "earliest");
    DataStream<Tuple2<String, Integer>> edits = env
            .addSource(new FlinkKafkaConsumer011<>("wiki-edits",
                    new CustomDeserializationSchema(), kafkaProps))
            .setParallelism(1)
            .map(new MapFunction<WikipediaEditEvent, Tuple2<String,
Integer>>() {
                @Override
                public Tuple2<String, Integer> map(WikipediaEditEvent
event) {
                    return new Tuple2<>(
                            event.getUser(), event.getByteDiff());
                }
            });

    DataStream<Tuple2<String, Integer>> results = edits
        // group by user
        .keyBy(0)
        .flatMap(new ComputeDiffs());
    results.print().setParallelism(1);

    // execute program
    env.execute("Flink Streaming Java API Skeleton");
}
```

The flat map function `ComputeDiffs`:

```
// Keep track of user byte diffs by key
public static final class ComputeDiffs extends RichFlatMapFunction<
            Tuple2<String, Integer>, Tuple2<String, Integer>> {

    // actually it would suffice to use ValueState<Integer>, because we
don't really need to store the user name (which is the key)
    private transient ValueState<Tuple2<String, Integer>> diffs;

    @Override
    public void open(Configuration parameters) throws Exception {
        ValueStateDescriptor<Tuple2<String, Integer>> descriptor =
                new ValueStateDescriptor<Tuple2<String, Integer>>(
                        "diffs",
```

```
                        TypeInformation.of(new TypeHint<Tuple2<String,
Integer>>() {}),
                        Tuple2.of("", 0)
                );
        diffs = getRuntimeContext().getState(descriptor);
    }

    @Override
    public void flatMap(Tuple2<String, Integer> in,
                        Collector<Tuple2<String, Integer>> out) throws
Exception {
        String user = in.f0;
        int diff = in.f1;
        Tuple2<String, Integer> currentDiff = diffs.value();

        // the key should always be the same here, so this can be
eliminated ...
        currentDiff.f0 = user;
        currentDiff.f1 += diff;

        // ... and we can out.collect(new Tuple2<String, Integer> (user,
new diff value))
        out.collect(currentDiff);
    }
}
```

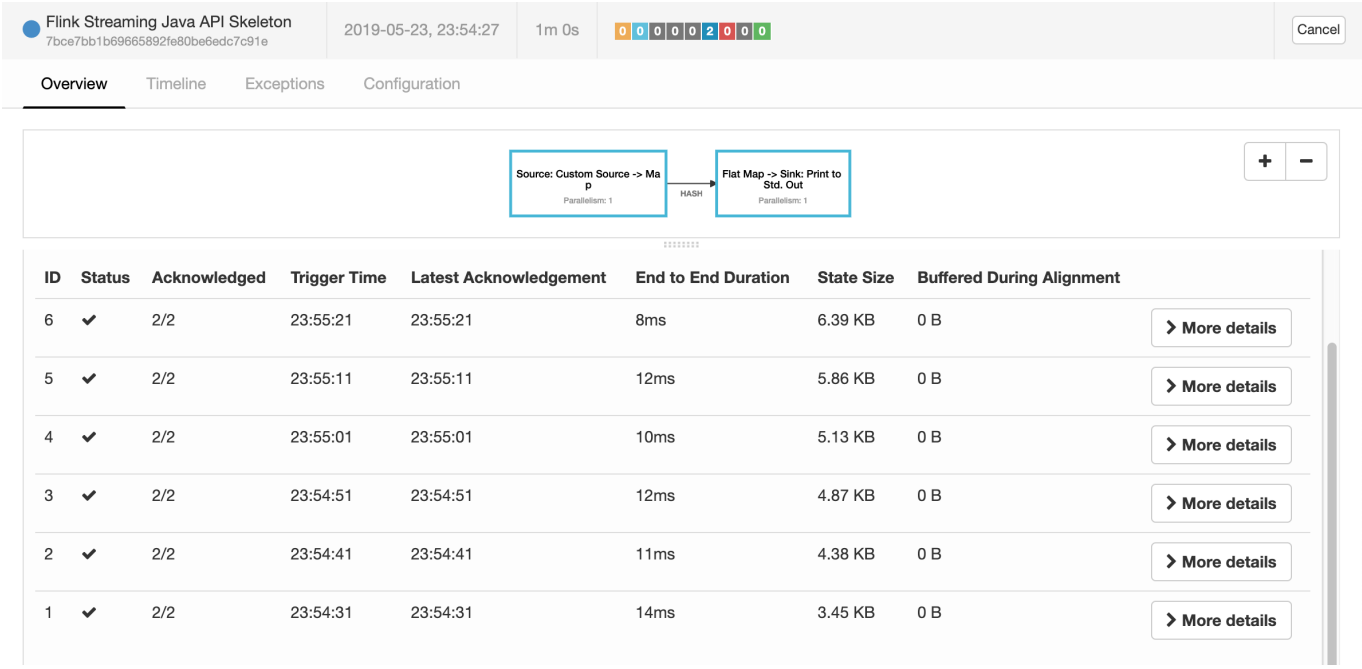
Task 1

Using ValueState is enough, because it is scoped to the key of the input element, which is the user name. It is also possible to use MapState, but it will be an over-kill because there will only be one key.

Task 2

The state backend is changed to RocksDB as stated by lines 4~5 of the above code. The checkpoints are triggered every 10 seconds. The size keeps growing, as shown in the below figure, because the state size get

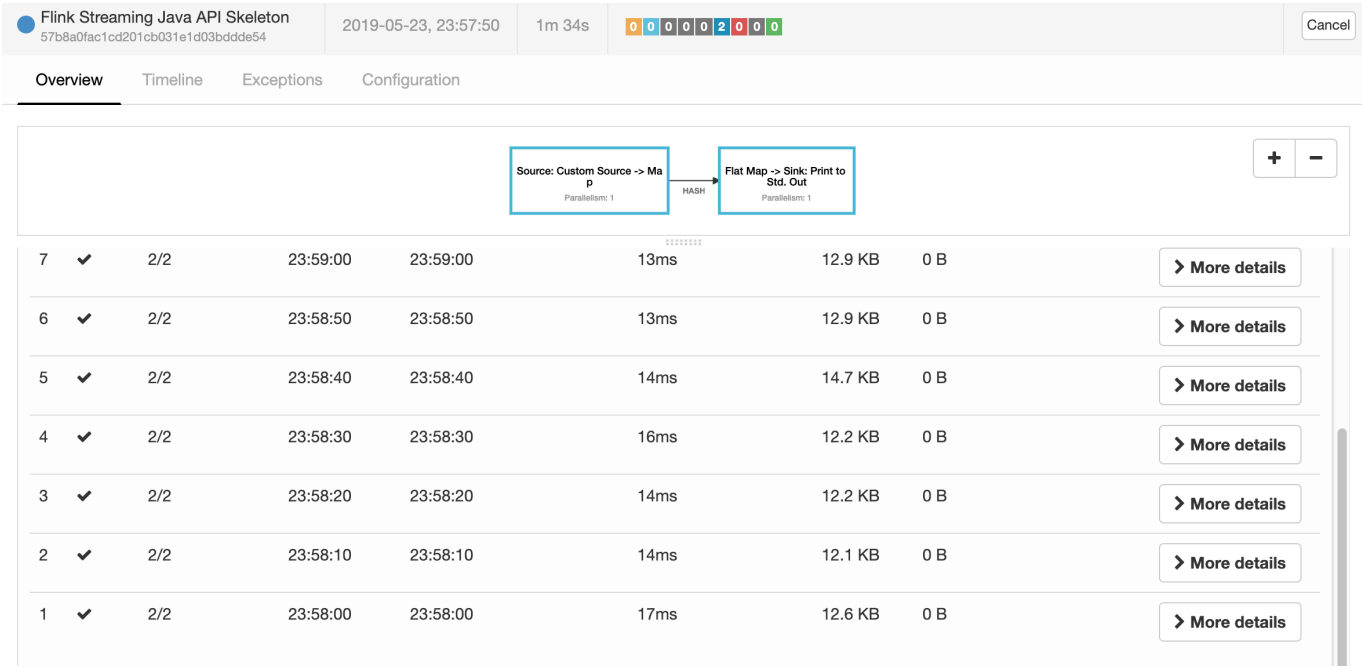
larger and larger. The end-to-end durations are around 10ms.



ID	Status	Acknowledged	Trigger Time	Latest Acknowledgement	End to End Duration	State Size	Buffered During Alignment	
6	✔	2/2	23:55:21	23:55:21	8ms	6.39 KB	0 B	More details
5	✔	2/2	23:55:11	23:55:11	12ms	5.86 KB	0 B	More details
4	✔	2/2	23:55:01	23:55:01	10ms	5.13 KB	0 B	More details
3	✔	2/2	23:54:51	23:54:51	12ms	4.87 KB	0 B	More details
2	✔	2/2	23:54:41	23:54:41	11ms	4.38 KB	0 B	More details
1	✔	2/2	23:54:31	23:54:31	14ms	3.45 KB	0 B	More details

Task 3

The size does not monotonously increase, because only changes are recorded.



7	✔	2/2	23:59:00	23:59:00	13ms	12.9 KB	0 B	More details
6	✔	2/2	23:58:50	23:58:50	13ms	12.9 KB	0 B	More details
5	✔	2/2	23:58:40	23:58:40	14ms	14.7 KB	0 B	More details
4	✔	2/2	23:58:30	23:58:30	16ms	12.2 KB	0 B	More details
3	✔	2/2	23:58:20	23:58:20	14ms	12.2 KB	0 B	More details
2	✔	2/2	23:58:10	23:58:10	14ms	12.1 KB	0 B	More details
1	✔	2/2	23:58:00	23:58:00	17ms	12.6 KB	0 B	More details

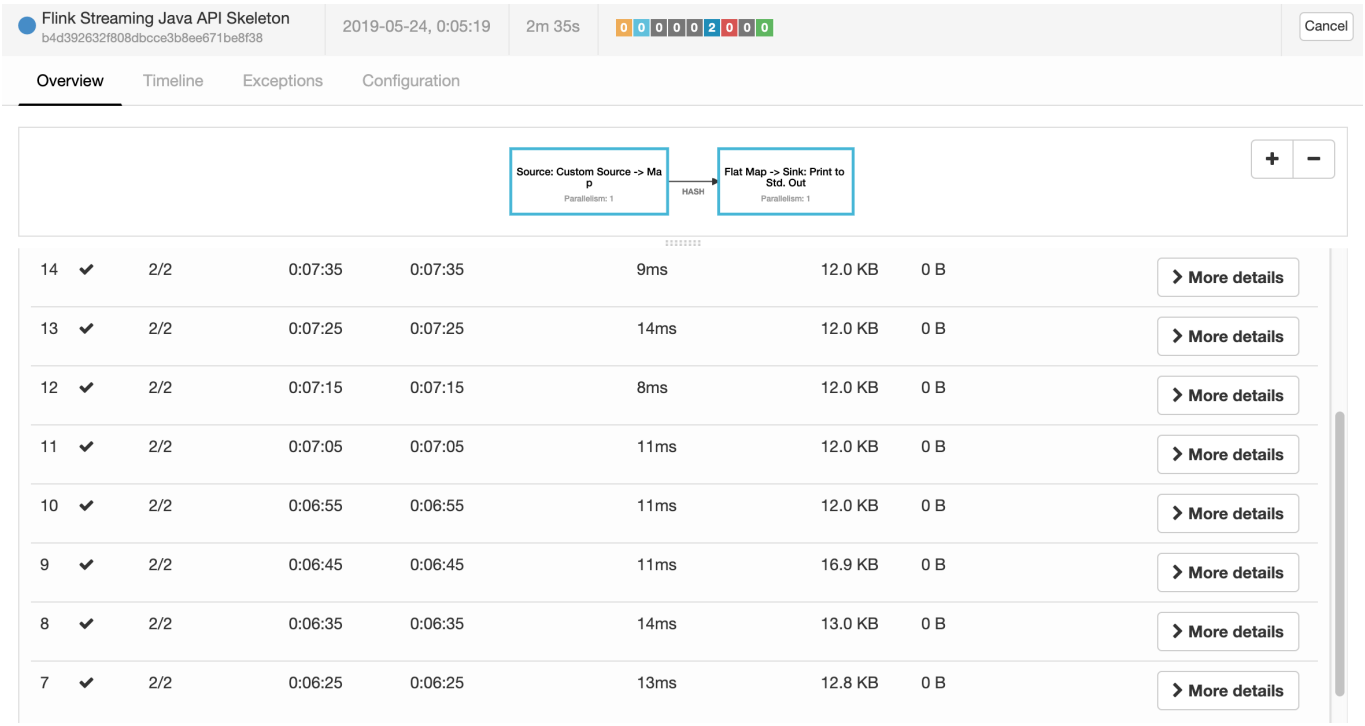
Overview History Summary Configuration Details for Checkpoint 6

ID	Status	Acknowledged	Trigger Time	Latest Acknowledgement	End to End Duration	State Size	Buffered During Alignment	Discarded	Path
6	✔ Completed	2/2 (100%)	23:58:50	23:58:50	13ms	12.9 KB	0 B	Yes	file:/Users/Xivid/repo/eth-dspa-2019/session-9/rocks.db/57b8a0fac1cd201cb031e1d03bddde54/chk-6

Operators

Name	Acknowledged	Latest Acknowledgment	End to End Duration	State Size	Buffered During Alignment	
Source: Custom Source -> Map	1/1 (100%)	23:58:50	9ms	1.21 KB	0 B	Show Subtasks
Flat Map -> Sink: Print to Std. Out	1/1 (100%)	23:58:50	13ms	11.7 KB	0 B	Show Subtasks

But the sizes are slightly larger than the non-incremental case, with a minimum size of 12KB when there is no new message at all (checkpoints 10~14 in the screenshot below). I assume this is the overhead for certain metadata.



III. Re-configure the application from a savepoint

Task 1

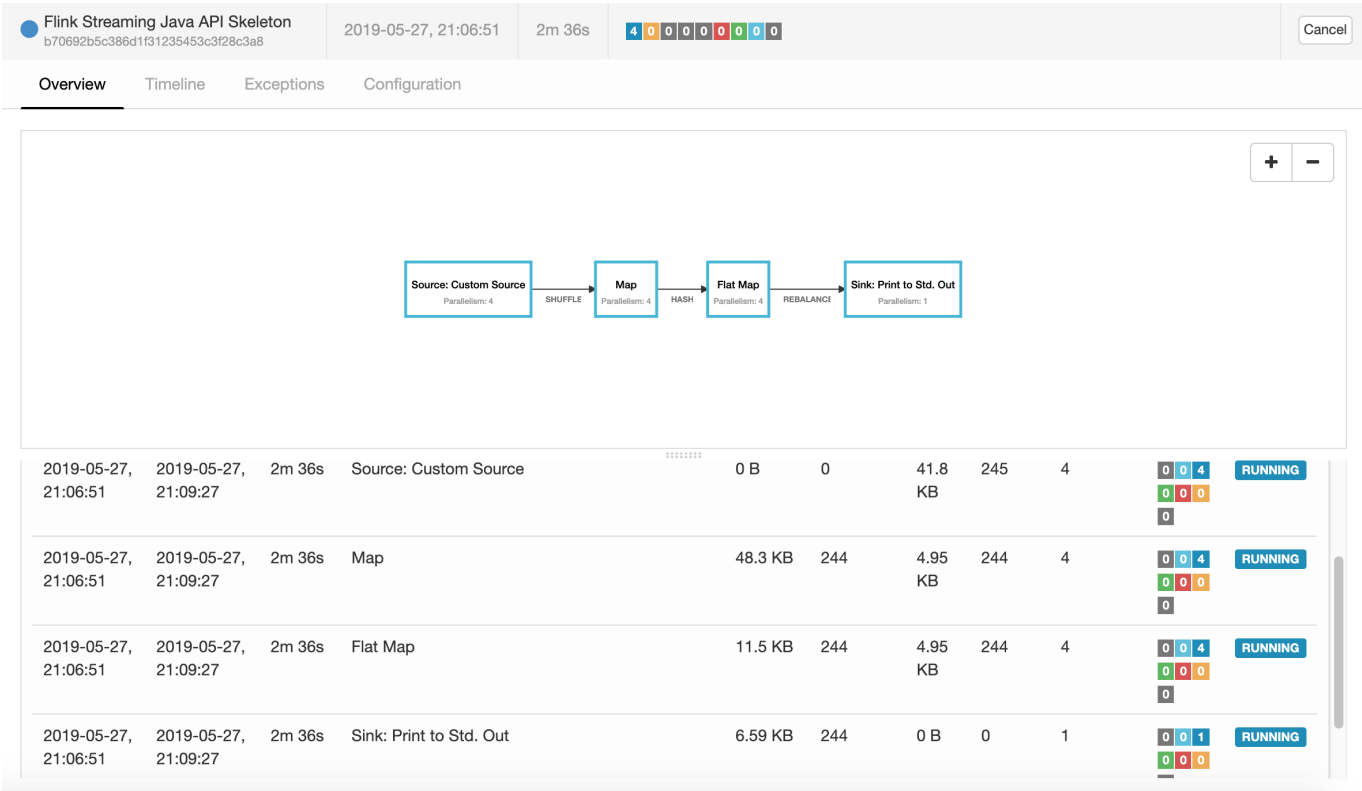
```
public static void main(String[] args) throws Exception {
    // set up the streaming execution environment
    final StreamExecutionEnvironment env =
StreamExecutionEnvironment.getExecutionEnvironment();
    RocksDBStateBackend backend = new
RocksDBStateBackend("file:///Users/Xivid/repo/eth-dspa-2019/session-
9/rocks.db", true);
    env.setStateBackend(backend);
    // Set a fixed delay restart strategy with a maximum of 5 restart
attempts
    // and a 1s interval between retries
    env.setRestartStrategy(RestartStrategies.fixedDelayRestart(5, 1000));

    // Take a checkpoint every 10s
    env.enableCheckpointing(10000);

    Properties kafkaProps = new Properties();
    kafkaProps.setProperty("zookeeper.connect", "localhost:2181");
    kafkaProps.setProperty("bootstrap.servers", "localhost:9092");
    kafkaProps.setProperty("group.id", "test-consumer-group");
    kafkaProps.setProperty("enable.auto.commit", "false");
    // always read the Kafka topic from the start
    kafkaProps.setProperty("auto.offset.reset", "earliest");
    DataStream<Tuple2<String, Integer>> edits = env
            .addSource(new FlinkKafkaConsumer011<>("wiki-edits",
```

```
                    new CustomDeserializationSchema(), kafkaProps))
            .uid("source")
            .shuffle()
            .map(new MapFunction<WikipediaEditEvent, Tuple2<String,
Integer>>() {
                @Override
                public Tuple2<String, Integer> map(WikipediaEditEvent
event) {
                    return new Tuple2<>(
                            event.getUser(), event.getByteDiff());
                }
            })
            .uid("map");

    DataStream<Tuple2<String, Integer>> results = edits
        // group by user
        .keyBy(0)
        .flatMap(new ComputeDiffs()).setParallelism(2)
        .uid("flatmap");
    results.print().setParallelism(1);

    // execute program
    env.execute("Flink Streaming Java API Skeleton");
}

// Keep track of user byte diffs by key
public static final class ComputeDiffs extends RichFlatMapFunction<
        Tuple2<String, Integer>, Tuple2<String, Integer>> {

    // actually it would suffice to use ValueState<Integer>, because we
don't really need to store the user name (which is the key)
    private transient ValueState<Tuple2<String, Integer>> diffs;

    @Override
    public void open(Configuration parameters) throws Exception {
        ValueStateDescriptor<Tuple2<String, Integer>> descriptor =
                new ValueStateDescriptor<Tuple2<String, Integer>>(
                        "diffs",
                        TypeInformation.of(new TypeHint<Tuple2<String,
Integer>>() {}),
                        Tuple2.of("", 0)
                );
        diffs = getRuntimeContext().getState(descriptor);
    }

    @Override
    public void flatMap(Tuple2<String, Integer> in,
                        Collector<Tuple2<String, Integer>> out) throws
Exception {
        String user = in.f0;
        int diff = in.f1;
        Tuple2<String, Integer> currentDiff = diffs.value();

        // the key should always be the same here, so this can be
```

```
eliminated ...
        currentDiff.f0 = user;
        currentDiff.f1 += diff;

        // ... and we can out.collect(new Tuple2<String, Integer> (user,
new diff value))
        out.collect(currentDiff);
    }
}
```

In `flink-conf.yaml` set the parameter: `state.savepoints.dir: file:///Users/Xivid/repo/eth-dspa-2019/flink-checkpoints`.

The job was initially started with parallelism 4.



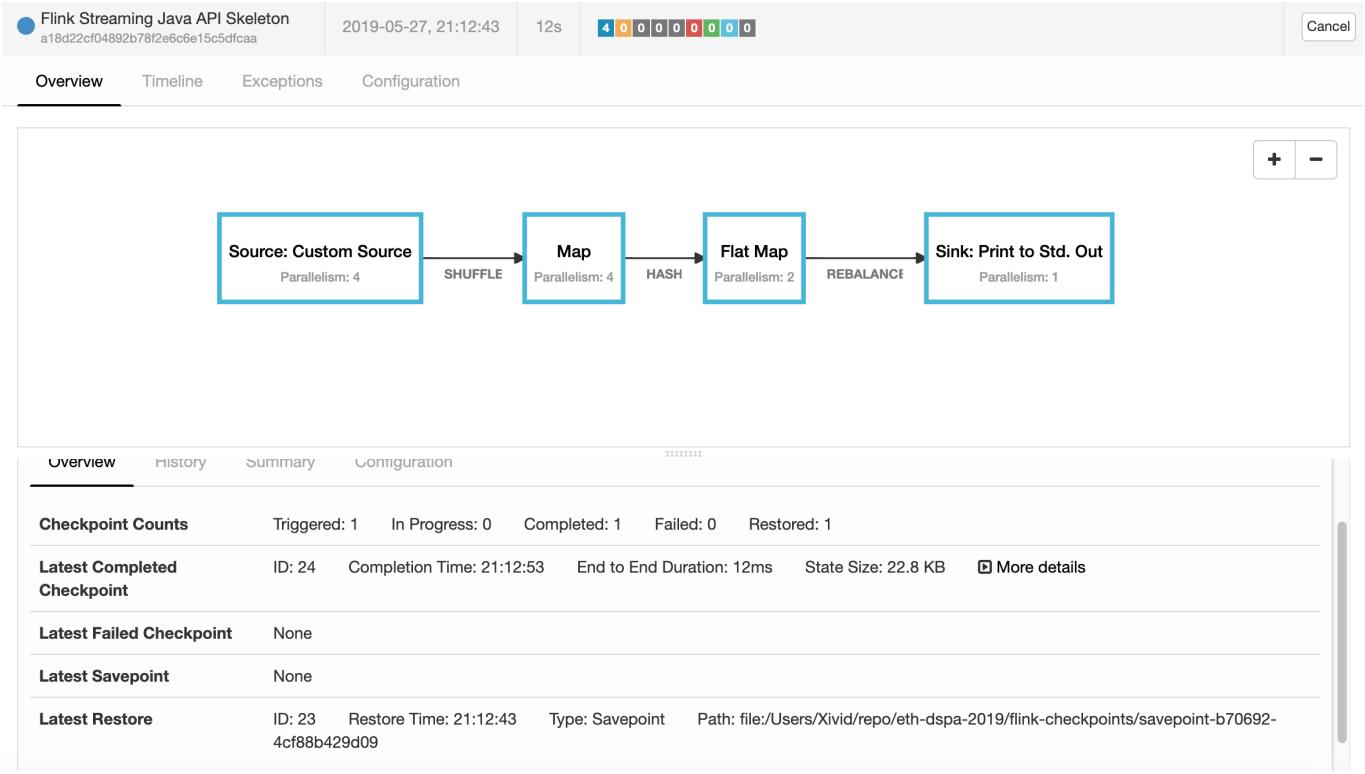
First, cancel the job with a savepoint:

```
Xivid@Xivids-MBP ~/repo/eth-dspa-2019/project/bin/flink-1.8.0 master + ● ? ./bin/flink ca
ncel -s b70692b5c386d1f31235453c3f28c3a8
Cancelling job b70692b5c386d1f31235453c3f28c3a8 with savepoint to default savepoint directory.
Cancelled job b70692b5c386d1f31235453c3f28c3a8. Savepoint stored in file:/Users/Xivid/repo/eth-dsp
a-2019/flink-checkpoints/savepoint-b70692-4cf88b429d09.
```

Then, restore from the savepoint

```
Xivid@Xivids-MBP ~/repo/eth-dspa-2019/project/bin/flink-1.8.0 master + ● ? ./bin/flink ru
n -s /Users/Xivid/repo/eth-dspa-2019/flink-checkpoints/savepoint-b70692-4cf88b429d09 ../../../sess
ion-9/wiki-edits/target/wiki-edits-0.1.jar
Starting execution of program
```

Restored with parallelism 2 for flatMap:



Task 2

To output MB instead of KB:

```
public static void main(String[] args) throws Exception {
    // set up the streaming execution environment
    final StreamExecutionEnvironment env =
StreamExecutionEnvironment.getExecutionEnvironment();
    RocksDBStateBackend backend = new
RocksDBStateBackend("file:///Users/Xivid/repo/eth-dspa-2019/session-
9/rocks.db", true);
    env.setStateBackend(backend);
    // Set a fixed delay restart strategy with a maximum of 5 restart
attempts
    // and a 1s interval between retries
    env.setRestartStrategy(RestartStrategies.fixedDelayRestart(5, 1000));

    // Take a checkpoint every 10s
    env.enableCheckpointing(10000);

    Properties kafkaProps = new Properties();
    kafkaProps.setProperty("zookeeper.connect", "localhost:2181");
    kafkaProps.setProperty("bootstrap.servers", "localhost:9092");
    kafkaProps.setProperty("group.id", "test-consumer-group");
    kafkaProps.setProperty("enable.auto.commit", "false");
    // always read the Kafka topic from the start
    kafkaProps.setProperty("auto.offset.reset", "earliest");
    DataStream<Tuple2<String, Integer>> edits = env
            .addSource(new FlinkKafkaConsumer011<>("wiki-edits",
                    new CustomDeserializationSchema(), kafkaProps))
```

```
            .uid("source")
            .shuffle()
            .map(new MapFunction<WikipediaEditEvent, Tuple2<String,
Integer>>() {
                @Override
                public Tuple2<String, Integer> map(WikipediaEditEvent
event) {
                    return new Tuple2<>(
                            event.getUser(), event.getByteDiff());
                }
            })
            .uid("map");

    DataStream<Tuple2<String, Double>> results = edits
        // group by user
        .keyBy(0)
        .flatMap(new ComputeDiffs()).setParallelism(2)
        .uid("flatmap");
    results.print().setParallelism(1);

    // execute program
    env.execute("Flink Streaming Java API Skeleton");
}

// Keep track of user byte diffs in a HashMap
public static final class ComputeDiffs extends RichFlatMapFunction<
        Tuple2<String, Integer>, Tuple2<String, Double>> {

    // user -> diffs
    private transient ValueState<Tuple2<String, Integer>> diffs;

    @Override
    public void open(Configuration parameters) throws Exception {
        ValueStateDescriptor<Tuple2<String, Integer>> descriptor =
                new ValueStateDescriptor<Tuple2<String, Integer>>(
                        "diffs",
                        TypeInformation.of(new TypeHint<Tuple2<String,
Integer>>() {}),
                        Tuple2.of("", 0)
                );
        diffs = getRuntimeContext().getState(descriptor);
    }

    @Override
    public void flatMap(Tuple2<String, Integer> in,
                        Collector<Tuple2<String, Double>> out) throws
Exception {
        String user = in.f0;
        int diff = in.f1;
        Tuple2<String, Integer> currentDiff = diffs.value();

        currentDiff.f0 = user;
        currentDiff.f1 += diff;
```

```
            out.collect(new Tuple2<String, Double>(currentDiff.f0,
    currentDiff.f1 / 1024.0));
        }
    }
```

Now restore again:

```
Xivid@Xivids-MBP ~/repo/eth-dspa-2019/project/bin/flink-1.8.0 master + ● ? ./bin/flink ru
n -s /Users/Xivid/repo/eth-dspa-2019/flink-checkpoints/savepoint-b70692-4cf88b429d09 ../../../sess
ion-9/wiki-edits/target/wiki-edits-0.1.jar
Starting execution of program
```

Compare the output before and after:

```
 1 (Gogolghosh,-13)
 2 (Agent00x,27)
 3 (Reports bot,0)
 4 (Agent00x,3403)
 5 (Yintan,-30)
 6 (Agent00x,143)
 7 (Girth Summit,32)
 8 (.254.5.98,1)
 9 (MrClog,-5)
10 (Francoisdjvr,-7)
11 (0.222.141.121,0)
12 (Iamreallygoodatcheckers,1)
13 (MLisDreaming,21)
14 (Taj4gt,0)
15 (Awmcphee,70)
16 (Girth Summit,1366)
17 (Tewapack,-7)
18 (Jpcase,138)
19 (AnomieBOT,-3)
20 (David Eppstein,76)
21 (Tom.Reding,22)
22 (Moughera,32)
23 (KylieTastic,-96)
24 (Rahuldottech,88)
25 (Tom.Reding,22)
26 (Tom.Reding,55)
27 (.113.201.172,9)
28 (Tom.Reding,22)
```

```
2441 (Francoisdjvr,-0.0068359375)
2442 (Reports bot,0.0)
2443 (Yintan,-0.029296875)
2444 (WikiCleanerBot,-9.765625E-4)
2445 (Bcp67,-0.0322265625)
2446 (Reports bot,0.0)
2447 (Rosiestep,0.01171875)
2448 (MethodMaster101,0.01171875)
2449 (Tassedethe,0.0048828125)
2450 (Fifaddicted,0.05078125)
2451 (Wgolf,0.0302734375)
2452 (Every-leaf-that-trembles,0.486328125)
2453 (2.251.178.119,0.0146484375)
2454 (01:4A:C001:7828:A9EB:D500:1D6C:AAE1,0.0205
     078125)
2455 (Wallyfromdilbert,-0.3671875)
2456 (Tewapack,-0.052734375)
2457 (Pozzi.c,0.205078125)
2458 (Froid,-0.140625)
2459 (Wbm1058,0.0791015625)
2460 (Hebrides,0.052734375)
2461 (4.221.19.247,0.2783203125)
2462 (Yoninah,0.162109375)
2463 (6.198.189.109,0.0576171875)
2464 (Tassedethe,0.0048828125)
2465 (Loginnigol,0.0)
2466 (Spicemix,0.0)
2467 (Aellanki67,0.0283203125)
```

The counters are correctly printed in KB, including those restored from the savepoint. (The order is different due to the parallelism of the FlatMap operator.)