

Przetwarzanie Strumieni Danych
Zajęcia Zintegrowane 2
Filip Pawłowski 310867

Uruchomiono maszynę wirtualną, w odpowiednich katalogach przekopiowano i wypakowano pliki źródłowe Flinka.

Następnie przy pomocy polecenia: `./bin/start-cluster.sh` uruchomiono cluster przetwarzania danych:

```
psd@ubuntu3:~/flink/flink-1.17.0$ ./bin/start-cluster.sh
Starting cluster.
Starting standalone session daemon on host ubuntu3.
Starting taskexecutor daemon on host ubuntu3.
```

Zweryfikowano działanie systemu przy pomocy przykładowego programu liczącego wyrazy – poleceniem: `./bin/flink run examples/streaming/WordCount.jar`

```
psd@ubuntu3:~/flink/flink-1.17.0$ ./bin/flink run examples/streaming/WordCount.jar
Executing example with default input data.
Use --input to specify file input.
Printing result to stdout. Use --output to specify output path.
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.flink.api.java.ClosureCleaner (file:/home/psd/flink/flink-1.17.0/lib/flink-dist-1.17.0.jar) to field java.lang.String.value
WARNING: Please consider reporting this to the maintainers of org.apache.flink.api.java.ClosureCleaner
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
Job has been submitted with JobID ad3fec835626006d67d9d8b7de3acd11
Program execution finished
Job with JobID ad3fec835626006d67d9d8b7de3acd11 has finished.
Job Runtime: 1762 ms
```

O sukcesie świadczy rezultat sprawdzenia statusu przy pomocy `tail log/flink-psd-taskexecutor-0-ubuntu3.out`:

```
psd@ubuntu3:~/flink/flink-1.17.0$ tail log/flink-psd-taskexecutor-0-ubuntu3.out
(nymph,1)
(in,3)
(thy,1)
(orisons,1)
(be,4)
(all,2)
(my,1)
(sins,1)
(remember,1)
(d,4)
```

Poprawne wykonanie zadań zostało również potwierdzone przy pomocy GUI Flinka pod adresem **localhost:8081** przy użyciu przeglądarki Firefox:

The screenshot shows the Apache Flink Web Dashboard in a Firefox browser window. The dashboard displays a table of completed jobs:

Job Name	Start Time	Duration	End Time	Tasks	Status
WordCount	2025-04-10 14:44:44	647ms	2025-04-10 14:44:45	2 / 2	FINISHED
WordCount	2025-04-10 14:41:34	1s	2025-04-10 14:41:36	2 / 2	FINISHED

Below the dashboard, a terminal window shows the execution of a Flink job:

```
psd@ubuntu3: ~/flink/flink-1.17.0
psd@ubuntu3: ~/flink/flink-1.17.0
Executing example with default input data.
Use --input to specify file input.
Printing result to stdout. Use --output to specify output path.
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.flink.api.java.ClosureCleaner (file:/home/psd/flink/flink-1.17.0/lib/flink-dist-1.17.0.jar) to fi
eld java.lang.String.value
WARNING: Please consider reporting this to the maintainers of org.apache.flink.api.java.ClosureCleaner
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
Job has been submitted with JobID c20b943a1283f24fe1d93893401abedc
Program execution finished
Job with JobID c20b943a1283f24fe1d93893401abedc has finished.
Job Runtime: 647 ms
psd@ubuntu3:~/flink/flink-1.17.0$
```

W następnym kroku należało uruchomić przykładowy kod z instrukcji laboratoryjnej przy użyciu Flink'a. W tym celu przygotowano środowisko wykonawcze poprzez instalację narzędzia Maven oraz środowiska języka Java:

```
psd@ubuntu3:~/flink/flink-1.17.0$ mvn --version
Apache Maven 3.6.3
Maven home: /usr/share/maven
Java version: 11.0.26, vendor: Ubuntu, runtime: /usr/lib/jvm/java-11-openjdk-amd64
Default locale: pl_PL, platform encoding: UTF-8
OS name: "linux", version: "6.8.0-52-generic", arch: "amd64", family: "unix"
psd@ubuntu3:~/flink/flink-1.17.0$ java --version
openjdk 11.0.26 2025-01-21
OpenJDK Runtime Environment (build 11.0.26+4-post-Ubuntu-1ubuntu122.04)
OpenJDK 64-Bit Server VM (build 11.0.26+4-post-Ubuntu-1ubuntu122.04, mixed mode, sharing)
```

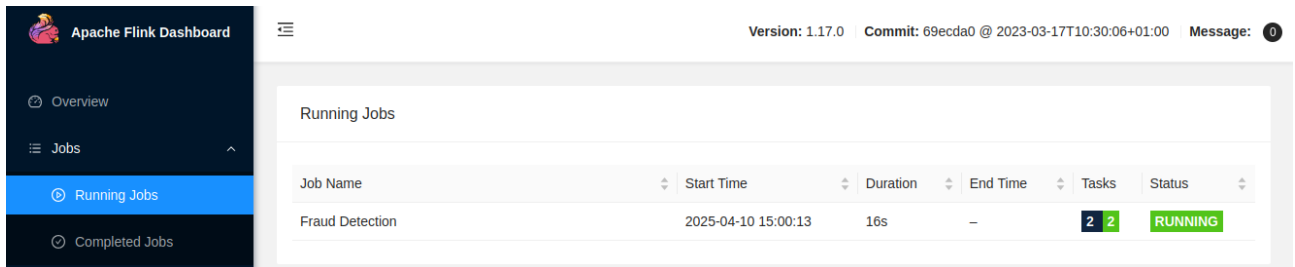
Zgodnie z instrukcjami na oficjalnej stronie Flink'a ([Fraud Detection with the DataStream API | Apache Flink](#)) wygenerowano projekt Fraud Detection z przykładowym plikiem pom.xml, zawierającym w zależnościach między innymi Flinka w wersji 1.17.2.

```
psd@ubuntu3:~/flink/flink-1.17.0/my_job$ mvn archetype:generate \
-DarchetypeGroupId=org.apache.flink \
-DarchetypeArtifactId=flink-walkthrough-datastream-java \
-DarchetypeVersion=1.17.2 \
-DgroupId=frauddetection \
-DartifactId=frauddetection \
-Dversion=0.1 \
-Dpackage=spendreport \
-DinteractiveMode=false
[INFO] Scanning for projects...
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-clean-plugin/2.5/maven-clean-plugin-2.5.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-clean-plugin/2.5/maven-clean-plugin-2.5.pom (3.9 kB at 5.3 kB/s)
```

Następnie przy pomocy polecenia **mvn package** zbudowano plik **.jar**, który został uruchomiony w środowisku Flink'a przy pomocy polecenia **./bin/flink run**.

```
psd@ubuntu3:~/flink/flink-1.17.0/my_job/frauddetection$ ~/flink/flink-1.17.0/bin/flink run target/frauddetection-0.1.jar
Job has been submitted with JobID f50513212634f2234dd116c798236c11
```

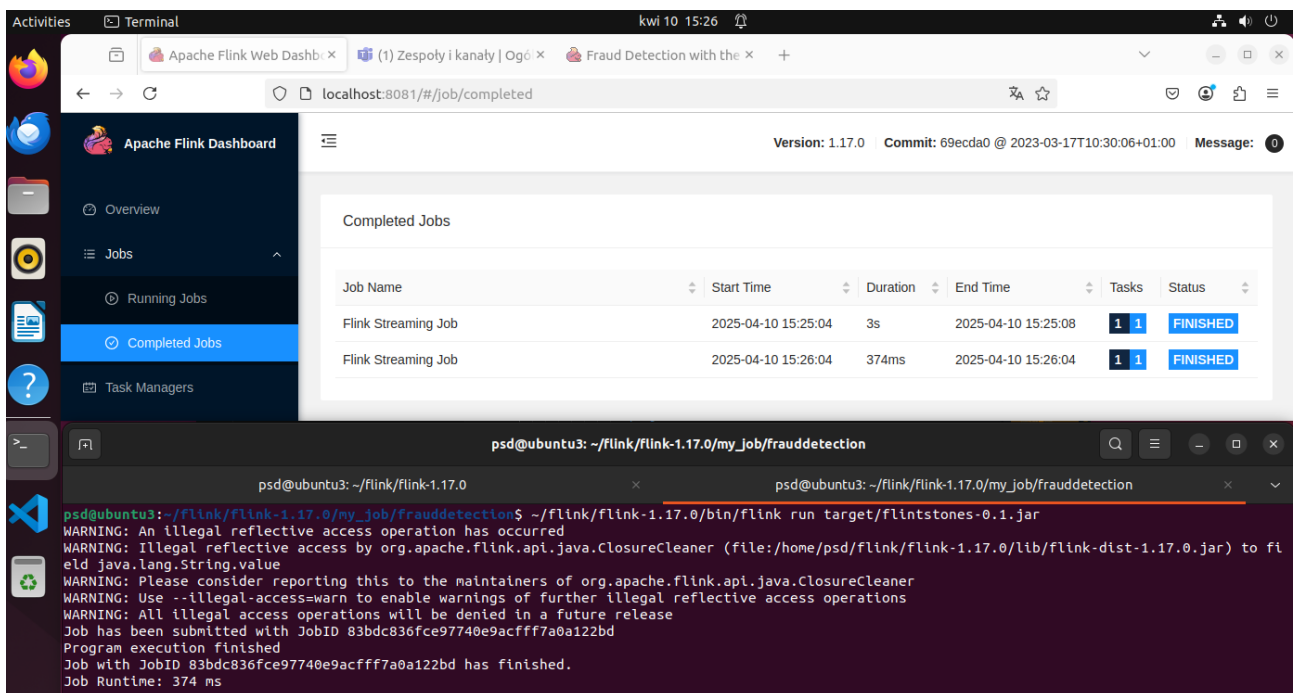
Zadanie było również widocznie w GUI pod portem 8081:



The screenshot shows the Apache Flink Dashboard interface. The left sidebar has a menu with 'Overview', 'Jobs', 'Running Jobs', and 'Completed Jobs'. The 'Running Jobs' tab is selected. The main panel displays a table of running jobs.

Job Name	Start Time	Duration	End Time	Tasks	Status
Fraud Detection	2025-04-10 15:00:13	16s	-	2 / 2	RUNNING

Podobne kroki wykonano dla przykładowego kodu z instrukcji do zajęć nr 2. Klasa o nazwie *Example* została uruchomiona:



The screenshot shows a desktop environment with a terminal window and a web browser displaying the Apache Flink Dashboard. The browser shows the 'Completed Jobs' tab with two finished jobs. The terminal window shows the command to run a Flink job and its output.

Completed Jobs Table:

Job Name	Start Time	Duration	End Time	Tasks	Status
Flink Streaming Job	2025-04-10 15:25:04	3s	2025-04-10 15:25:08	1 / 1	FINISHED
Flink Streaming Job	2025-04-10 15:26:04	374ms	2025-04-10 15:26:04	1 / 1	FINISHED

Terminal Output:

```
psd@ubuntu3: ~/flink/flink-1.17.0/my_job/frauddetection
psd@ubuntu3: ~/flink/flink-1.17.0
psd@ubuntu3: ~/flink/flink-1.17.0$ ./bin/flink run target/flintstones-0.1.jar
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.flink.api.java.ClosureCleaner (file:/home/psd/flink/flink-1.17.0/lib/flink-dist-1.17.0.jar) to field java.lang.String.value
WARNING: Please consider reporting this to the maintainers of org.apache.flink.api.java.ClosureCleaner
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
Job has been submitted with JobID 83bdc836fce97740e9acfff7a0a122bd
Program execution finished
Job with JobID 83bdc836fce97740e9acfff7a0a122bd has finished.
Job Runtime: 374 ms
```

Następnie przygotowano własny kod operujący na Sensorach wykrywających temperatury poniżej 0 stopni. Klasę **FreezeDetector.java**:

```
src > main > java > spendreport > J FreezeDetector.java
```

```
1  /*
12 * Unless required by applicable law or agreed to in writing, software
13 * distributed under the License is distributed on an "AS IS" BASIS,
14 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
15 * See the License for the specific language governing permissions and
16 * limitations under the License.
17 */
18
19 package spendreport;
20
21 import org.apache.flink.streaming.api.functions.KeyedProcessFunction;
22 import org.apache.flink.util.Collector;
23 import org.apache.flink.walkthrough.common.entity.Alert;
24
25 /**
26  * Skeleton code for implementing a fraud detector.
27  */
28 public class FreezeDetector extends KeyedProcessFunction<Integer, FreezeDetectionJob.Sensor, Alert> {
29
30     @Override
31     public void processElement(
32         FreezeDetectionJob.Sensor sensor,
33         Context context,
34         Collector<Alert> collector) throws Exception {
35
36         if (sensor.getTemperature() < 0 ) {
37
38             Alert alert = new Alert();
39             alert.setId(sensor.getId());
40
41             collector.collect(alert);
42         }
43
44     }
45 }
46
```

Oraz klasę **FreezeDetectionJob.java**:

src > main > java > spendreport > J FreezeDetectionJob.java

```
1 package spendreport;
2
3 import org.apache.flink.streaming.api.datastream.DataStream;
4 import org.apache.flink.streaming.api.environment.StreamExecutionEnvironment;
5 import org.apache.flink.walkthrough.common.sink.AlertSink;
6 import org.apache.flink.walkthrough.common.entity.Alert;
7 import org.apache.flink.walkthrough.common.source.TransactionSource;
8
9 public class FreezeDetectionJob {
10
11     public static void main(String[] args) throws Exception {
12         StreamExecutionEnvironment env = StreamExecutionEnvironment.getExecutionEnvironment();
13
14         DataStream<Sensor> temperatures = env.fromElements(
15             new Sensor(1255465, -1),
16             new Sensor(1277465, 35),
17             new Sensor(1277465, 35),
18             new Sensor(1277465, 35),
19             new Sensor(1277465, -123214),
20             new Sensor(1277465, 0),
21             new Sensor(1277465, 5),
22             new Sensor(1277465, 35),
23             new Sensor(1248865, -12))
24             .name("temperatures");
25
26         DataStream<Alert> alerts = temperatures
27             .keyBy(Sensor::getId)
28             .process(new FreezeDetector())
29             .name("freeze-detector");
30
31         alerts
32             .addSink(new AlertSink())
33             .name("send-alerts");
34
35         alerts.print();
36
37         env.execute("Freeze Detection");
38     }
39
40     public static class Sensor {
41         public Integer id;
42         public Integer temperature;
43
44         public Sensor() {}
45
46         public Sensor(Integer id, Integer temperature) {
47             this.id = id;
48             this.temperature = temperature;
49         }
50
51         public String toString() {
52             return this.id.toString() + ": temperature " + this.temperature.toString();
53         }
54
55         public Integer getTemperature() {
56             return this.temperature;
57         }
58
59         public Integer getId() {
60             return this.id;
61         }
62     }
63 }
64
65
66
```

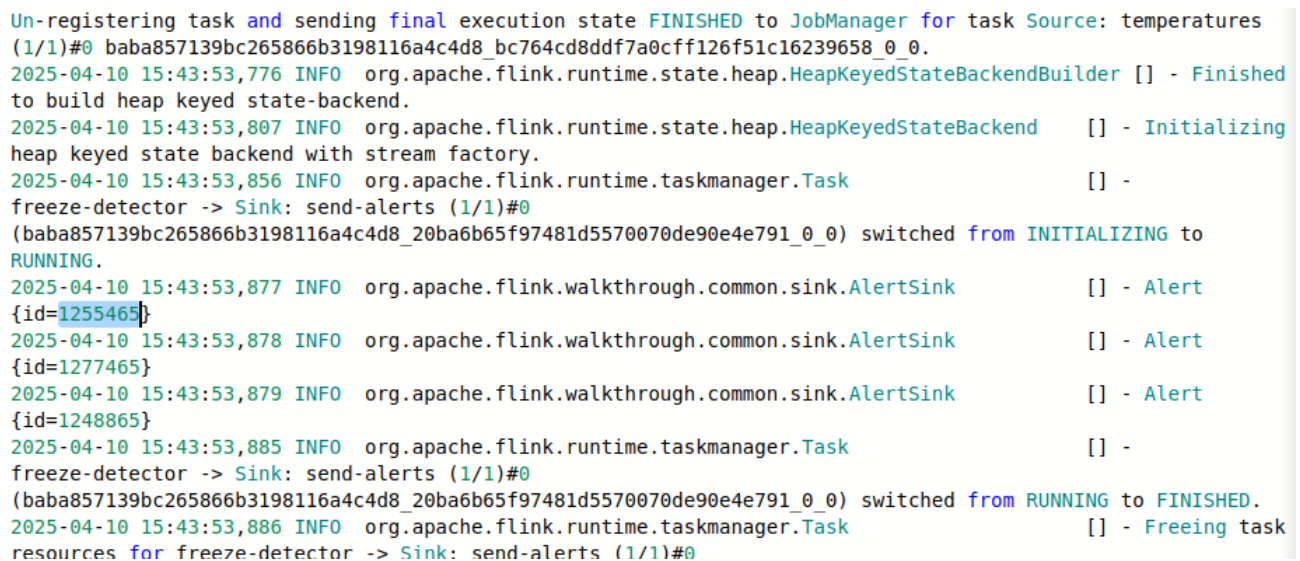
Po ponownym uruchomieniu przy użyciu Flink’a zweryfikowano ich działanie na podstawie logów w zakładce GUI o nazwie JobManager. Dla poprzedniego programu przykładowego z instrukcji pojawiły się wpisy reprezentujące Flintstonów. Dla własnego programu pojawiły się natomiast wpisy z identyfikatorami sensorów, dla których temperatury odnotowane nie przekroczyły 0 stopni:



The screenshot shows the 'Log List' tab in the Flink JobManager GUI. The breadcrumb navigation indicates the path 'Log List / flink-psd-taskexecutor-0-ubuntu3.out'. The log entries are as follows:

```
1 WARNING: An illegal reflective access operation has occurred
2 WARNING: Illegal reflective access by org.jboss.netty.util.internal.ByteBufferUtil (file:/tmp/
  flink-rpc-akka_fa8ad7c1-95e3-4ea5-9af7-58da79b7bec3.jar) to method java.nio.DirectByteBuffer.cleaner()
3 WARNING: Please consider reporting this to the maintainers of org.jboss.netty.util.internal.ByteBufferUtil
4 WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
5 WARNING: All illegal access operations will be denied in a future release
6 Fred: age 35
7 Wilma: age 35
8 Fred: age 35
9 Wilma: age 35
10 Alert{id=1255465}
11 Alert{id=1277465}
12 Alert{id=1248865}
13
```

Alerty te były również widoczne w zakładce TaskManager:



The screenshot shows the TaskManager logs. The log entries are as follows:

```
Un-registering task and sending final execution state FINISHED to JobManager for task Source: temperatures
(1/1)#0 baba857139bc265866b3198116a4c4d8_bc764cd8ddf7a0cff126f51c16239658_0_0.
2025-04-10 15:43:53,776 INFO org.apache.flink.runtime.state.heap.HeapKeyedStateBackendBuilder [] - Finished
to build heap keyed state-backend.
2025-04-10 15:43:53,807 INFO org.apache.flink.runtime.state.heap.HeapKeyedStateBackend [] - Initializing
heap keyed state backend with stream factory.
2025-04-10 15:43:53,856 INFO org.apache.flink.runtime.taskmanager.Task [] -
freeze-detector -> Sink: send-alerts (1/1)#0
(baba857139bc265866b3198116a4c4d8_20ba6b65f97481d5570070de90e4e791_0_0) switched from INITIALIZING to
RUNNING.
2025-04-10 15:43:53,877 INFO org.apache.flink.walkthrough.common.sink.AlertSink [] - Alert
{id=1255465}
2025-04-10 15:43:53,878 INFO org.apache.flink.walkthrough.common.sink.AlertSink [] - Alert
{id=1277465}
2025-04-10 15:43:53,879 INFO org.apache.flink.walkthrough.common.sink.AlertSink [] - Alert
{id=1248865}
2025-04-10 15:43:53,885 INFO org.apache.flink.runtime.taskmanager.Task [] -
freeze-detector -> Sink: send-alerts (1/1)#0
(baba857139bc265866b3198116a4c4d8_20ba6b65f97481d5570070de90e4e791_0_0) switched from RUNNING to FINISHED.
2025-04-10 15:43:53,886 INFO org.apache.flink.runtime.taskmanager.Task [] - Freeing task
resources for freeze-detector -> Sink: send-alerts (1/1)#0
```