Flink四种选择Key的方法

在Flink中有许多函数需要我们为其指定key，比如groupBy，Join中的where等。如果我们指定的Key不对，可能会出现一些问题，正如下面的程序：

```
package com.iteblog.flink

import org.apache.flink.api.scala.{ExecutionEnvironment, _}
import org.apache.flink.util.Collector

////////////////////////////////////////////////////////////////////////
 User: 过往记忆
 Date: 2017-03-13
 Time: 22:59
 bolg: https://www.iteblog.com
 本文地址：https://www.iteblog.com/archives/2069
 过往记忆博客，专注于hadoop、hive、spark、shark、flume的技术博客，大量的干货
 过往记忆博客微信公共帐号：iteblog_hadoop
////////////////////////////////////////////////////////////////////////
object GroupCombine {
  def main(args: Array[String]): Unit = {
    val env = ExecutionEnvironment.getExecutionEnvironment
    val input: DataSet[String] = env.fromElements("a", "b", "c", "a")
    val combinedWords: DataSet[(String, Int)] = input
      .groupBy(0)
      .combineGroup {
        (words, out: Collector[(String, Int)]) =>
          var key: String = null
          var count = 0
          for (word <- words) {
            key = word
            count += 1
          }
          out.collect((key, count))
      }

    combinedWords.print()
  }
}
```

上面的代码没有任何语法错误，但是我们编译运行这个程序，就会出现以下的异常信息：

```
Exception in thread "main" org.apache.flink.api.common.InvalidProgramException: Specifying keys via field positions is only valid for tuple data types. Type: String
 at org.apache.flink.api.common.operators.Keys$ExpressionKeys.<init>(Keys.java:232)
 at org.apache.flink.api.common.operators.Keys$ExpressionKeys.<init>(Keys.java:223)
 at org.apache.flink.api.scala.DataSet.groupBy(DataSet.scala:875)
 at com.iteblog.flink.GroupCombine$.main(GroupCombine.scala:14)
 at com.iteblog.flink.GroupCombine.main(GroupCombine.scala)
 at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
 at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:57)
 at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
 at java.lang.reflect.Method.invoke(Method.java:601)
 at com.intellij.rt.execution.application.AppMain.main(AppMain.java:147)
```

很显然，上面groupBy(0)函数需要我们指定一个key，但是我们为其指定了值为0，而0仅仅对tuple类型的数据有效，所以才导致上面的异常。Flink中指定key的值主要有以下四种方法：

- 指定key表达式（key expressions）
- 指定key选择函数
- 一个或多个字段位置键（field position keys），这个仅仅对Tuple类型的DataSet有效)
- Case Class中的字段

下面我将一一介绍这四种方法的使用。

指定key expressions

键表达式指定DataSet中元素的一个或多个字段。
每个键表达式是公共字段的名称或getter方法。点符号可以用于表示整个对象。"*"
key表达式表示选择所有的字段。具体使用如下：

```
// some ordinary POJO
class WC(val word: String, val count: Int) {
  def this() {
    this(null, -1)
  }
  // [...]
}

val words: DataSet[WC] = // [...]
val wordCounts = words.groupBy("word").reduce {
  (w1, w2) => new WC(w1.word, w1.count + w2.count)
```

```
}
```

这个例子中，需要指定key的函数是groupBy，我们将WC样本类中的word字段作为key表达式，所以上面的代码将会对WC样本类中的word字段进行分组。

指定Key选择函数

key选择器函数从DataSet中的元素提取键值。如下：

```
// some ordinary POJO
class WC(val word: String, val count: Int) {
  def this() {
    this(null, -1)
  }
  // [...]
}

val words: DataSet[WC] = // [...]
val wordCounts = words.groupBy { _.word } reduce {
  (w1, w2) => new WC(w1.word, w1.count + w2.count)
}
```

我们指定了_.word选择函数，这样上面的groupBy函数将会对WC样本类中的word字段进行分组。

指定Field Position

如果你的DataSet中存储的元素类型是Tuple，那么我们可以指定Tuple中的Field Position，使用如下：

```
val tuples = DataSet[(String, Int, Double)] = // [...]
// group on the first and second Tuple field
val reducedTuples = tuples.groupBy(0, 1).reduce { ... }
```

正如上面代码，我们将Tuple中的第一和第二个field作为key传入groupBy，这样只要Tuple中第一和第二个field相同的元素将会分组到一起。

指定Case Class中的Fields

如果你的DataSet中存储的元素类型是样本类（Case Class），那么我们是可以直接指定Case Class中Field的名字，如下：

```
case class MyClass(val a: String, b: Int, c: Double)
val tuples = DataSet[MyClass] = // [...]
// group on the first and second field
val reducedTuples = tuples.groupBy("a", "b").reduce { ... }
```

如何解决文章开头的问题

现在我们已经学习了四种指定key的方法，那么文章中最开始那个例子该如何指定我们的key呢？很好办，这里最少有三种方法解决：

通过指定key expressions实现

因为我们的DataSet中存储的类型是String，我们可以指定“*”来选择元素中的所有字段，如下：

```
.groupBy("*")
```

通过指定key选择器

```
.groupBy(x => x)
```

通过map转换实现

我们可以对DataSet进行转换，将原DataSet中的元素全部转换成Tuple1，然后就可以通过指定Field Position来实现了，如下：

```
package com.iteblog.flink

import org.apache.flink.api.scala.{ExecutionEnvironment, _}
import org.apache.flink.util.Collector

object GroupCombine {
```

```
  def main(args: Array[String]): Unit = {
    val env = ExecutionEnvironment.getExecutionEnvironment
    val input = env.fromElements("a", "b", "c", "a").map(Tuple1(_))
    val combinedWords: DataSet[(String, Int)] = input
      .groupBy(0)
      .combineGroup {
        (words, out: Collector[(String, Int)]) =>
          var key: String = null
          var count = 0
          for (word <- words) {
            key = word._1
            count += 1
          }
          out.collect((key, count))
      }

    combinedWords.print()
  }
}
```

当然你也可以将原DataSet转换成case class，实现和转换成Tuple1类似，我就不介绍了。