

Scala solution with a Haskell taste

How to make your scala more haskellish?

George Leontiev

folone.info

February 27, 2013

Note: I intentionally made it more "interesting" to show more neat scalaz stuff

Brief Overview

Let's look at the types

Main functions

```
object Words
```

```
  wordCount :: String → [(String, Int)]
```

```
  main :: Array String → IO Unit
```

Helper functions

```
acceptedChars :: Char → Boolean
```

```
getFileContent :: String → IO String
```

```
time :: R → IO R
```

Counting routine

```
def wordCount(text: String): List[(String, Int)] =  
  text.filter(acceptedChars)  
    .toLowerCase.split("\\W")  
    .par  
    .groupBy(identity)  
    .map { case (key, value) =>  
      key.trim → value.length  
    }  
    .filterNot { case (key, _) => key.isEmpty }  
    .toList.sortBy { case (_, value) => -value }  
    .seq.toList
```

First attempt

```
def wholeFile(path: String) =  
  for {  
    source ← IO { Source.fromFile(path) }  
    text   = source.mkString  
    _ ←      IO { source.close() }  
    result = wordCount(text)  
  } yield result.take(N).shows
```

First attempt

Works fine, but eats all the heap on a large enough file.

Second attempt

```
def byLine(path: String) =  
  for {  
    source ← IO { Source.fromFile(path) }  
    stream = source.getLines.toStream  
    result = stream.map(wordCount)  
    _ ←      IO { source.close() }  
  } yield result.take(N).shows
```

Second attempt

Just what is this `|+|`?


```
implicit val mapInstances =  
  new Show[List[(String, Int)]]  
    with Monoid[List[(String, Int)]]
```

is the same as

```
instance Show [(String, Int)] where ...  
instance Monoid [(String, Int)] where ...
```

Second attempt

Pretty slow.

Wordcounting

Scoobi <http://nicta.github.com/scoobi/>

Spark <http://spark-project.org/>

Scalding <https://github.com/twitter/scalding/wiki/Type-safe-api-reference>

Turns out, this code will work for these "as is".

TODO finish up: some notes on IO and on Arrows