Zero-effort Structural Logging

London Scala Users Group, Scala Community Day, 14/12/2019

Septimal Mind Ltd
team@7mind.io

What would be the outcome of the following code?

```scala
val id = "john@doe.com"
val balance = 265
logger.info(s"User id=$id, balance=$balance")
```

```
User id=john@doe.com, balance=265
```

There may be a bit different code as well:

```scala
val id = "+13023072835"
val balance = 42
logger.info(s"User id=$id, balance=$balance")
```

```
User id=+13023072835, balance=42
```

let's assume we've collected many logs and now we want let's assume we've collected many of these logs. . .

. . . and we want to filter logs lines for users. . .

. . . who registered with their email

```
grep -E -o "[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+[A-Za-z]2,6" file.txt
```

Oops. . .

Can we do it better?

We need structural logs

```
{
"id": "john@doe.com",
"id.type": "email",
"balance":"42"
}
```

Almost all the structural logging libraries are very inconvenient. . .

```scala
object Example {
  val LOG = FluentLoggerFactory
    .getLogger("fluentd.test")

  // ...

  val data =
      new HashMap[String, String]()
  data.put("id", userId)
  data.put("balance", userBalance.toString)
  LOG.log("user balance", data)
}
```

But the code...

```scala
val user = "JohnDoe"
logger.debug(s"Received a message from $user")
```

. . . is always structured!

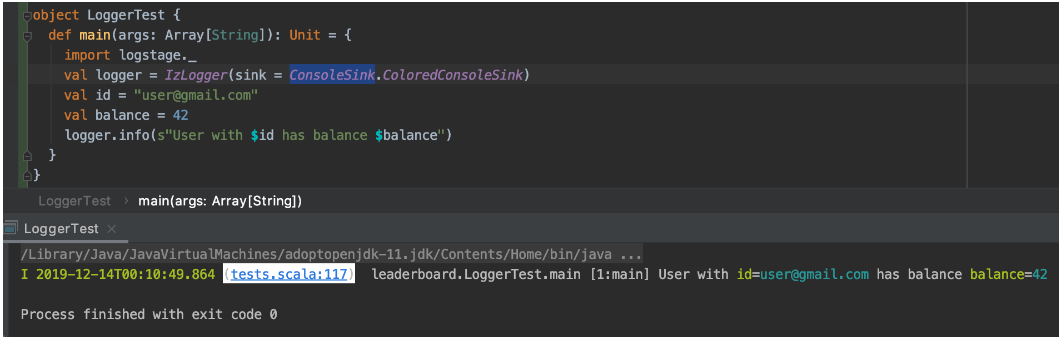
```
Expr(Apply(Select(
  Apply(
    Select(Select(Ident("scala"), scala.StringContext),
      TermName("apply"))
      , List(Literal(Constant("Received a message from "))
          , Literal(Constant(""))
        )
  ),
  TermName("s")
  )
, List(Ident(TermName("user")))
))
```

LOGSTAGE

First-class logging framework for Scala

So, we made a library . . .
which can extract structure . . .

. . . so you may have text for humans

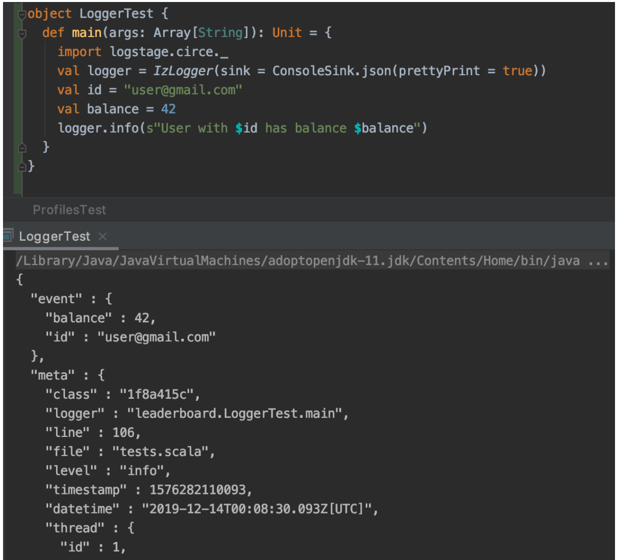


```
object LoggerTest {
  def main(args: Array[String]): Unit = {
    import logstage._
    val logger = IzLogger(sink = ConsoleSink.ColoredConsoleSink)
    val id = "user@gmail.com"
    val balance = 42
    logger.info(s"User with $id has balance $balance")
  }
}
```

```
/Library/Java/JavaVirtualMachines/adoptopenjdk-11.jdk/Contents/Home/bin/java ...
I 2019-12-14T00:10:49.864 (tests.scala:117)  leaderboard.LoggerTest.main [1:main] User with id=user@gmail.com has balance balance=42

Process finished with exit code 0
```

. . . and nice JSON for robots



```
object LoggerTest {
  def main(args: Array[String]): Unit = {
    import logstage.circe._
    val logger = IzLogger(sink = ConsoleSink.json(prettyPrint = true))
    val id = "user@gmail.com"
    val balance = 42
    logger.info(s"User with $id has balance $balance")
  }
}
```

```
/Library/Java/JavaVirtualMachines/adoptopenjdk-11.jdk/Contents/Home/bin/java ...
{
  "event" : {
    "balance" : 42,
    "id" : "user@gmail.com"
  },
  "meta" : {
    "class" : "1f8a415c",
    "logger" : "leaderboard.LoggerTest.main",
    "line" : 106,
    "file" : "tests.scala",
    "level" : "info",
    "timestamp" : 1576282110093,
    "datetime" : "2019-12-14T00:08:30.093Z[UTC]",
    "thread" : {
      "id" : 1,
```

Batteries included

1. `LogF[F[_]]`
2. `LogBIO[F[_, _]]`
3. Better alternative for MDC
4. `slf4j` adapter
5. etc, etc...

Thank you for your attention

docs: `https://izumi.7mind.io`

Septimal Mind is an Irish software consultancy and R&D Company

We're looking for clients, contributors, adopters and colleagues ;)

Contact: team@7mind.io

Github: `https://github.com/7mind`

Slides: `https://github.com/7mind/slides`

Follow us on Twitter

`https://twitter.com/shirshovp`

`https://twitter.com/kai_nyasha`