Register | Login | Logout | Contact Us

Java-Success.com

Industrial strength Java/JEE Career Companion to open more doors

search here … Go

Home | Java FAQs | 600+ Java Q&As | Career | Tutorials | Member | Why?

Can u Debug? | Java 8 ready? | Top X | Productivity Tools | Judging Experience?

Home › member-paid › 05: Q27 – Q32 Scala idioms Q&As on Option-Some-None, Try-Success-Failure and Either-Left-Right

05: Q27 – Q32 Scala idioms Q&As on Option-Some-None, Try-Success-Failure and Either-Left-Right

Posted on August 16, 2016 by Arulkumaran Kumaraswamipillai

0 Like Share Tweet 0 G+1 Share

Java has idioms like Try-Catch-Finally, Scala has idioms like **1)** "Option-Some-None", **2)** Either-Left-Right **3)** Try-Success-Failure.

Q27. How does Scala try to solve the problem of getting rid of null values & the "NullPointerException"?

A27. Scala provides its own type for representing optional values with "Option[A]", which is a **trait**. So, instead of returning one object when a function succeeds and null when

600+ Full Stack Java/JEE Interview Q&As ♥Free ♦FAQs

open all | close all

⊞ Ice Breaker Interview
⊞ Core Java Interview Q
⊞ JEE Interview Q&A (3
⊞ Pressed for time? Jav
⊞ SQL, XML, UML, JSC
⊞ Hadoop & BigData Int
⊞ Java Architecture Inte
⊟ Scala Interview Q&As
⊟ Scala way of codin
01: Coding Scala
01: ♥ Q1 – Q6 Sca
02: Q6 – Q12 Scala
03: Q13 – Q18 Sca
04: Q19 – Q26 Sca
05: Q27 – Q32 Sca
06: Q33 – Q40 Sca
07: Q41 – Q48 Sca
08: Q49 – Q58 Sca
09: Q59 – Q65 Hig
10: Q66 – Q70 Pat
11: Q71 – Q77 – Sc

it fails, your function should instead return an instance of an **Option**, where the Option object is either "**Some[A]**" or "**None[A]**".

"Some" and "None" are children of "Option".

```
object IdiomOptionSomeNone extends App {

  def toInt(input: String): Option[Int] = {
    try {
      Some(input.toInt)
    } catch {
      case e: NumberFormatException => None
    }
  }

  println(toInt("12").getOrElse(-1)) //12

  println(toInt("Hello").getOrElse(-1)) //-1
}
```

Q28. What will be the output of the following code?

```
object OptionTest extends App {

  val x:Option[Int] = Option(5)
  val y:Option[Int] = None


  val result = for {
     a:Int <- x
     b:Int <- y
  } yield {
     a + b
  }

  println("result=" + result)
  println("result=" + result.getOrElse(0))
}
```

A28. "**for…yield**" in Scala is known as the "for **comprehension**", which is a syntactic sugar for composing multiple operations like forach, map, flatMap, and filter.

Output:

12: Q78 – Q80 Rec
⊞ Spring, Hibernate, & I
⊞ Testing & Profiling/Sa
⊞ Other Interview Q&A 1
⊞ ▶ Free Java Interviev

16 Technical Key Areas

open all | close all
⊞ Best Practice (6)
⊞ Coding (26)
⊞ Concurrency (6)
⊞ Design Concepts (7)
⊞ Design Patterns (11)
⊞ Exception Handling (3
⊞ Java Debugging (21)
⊞ Judging Experience In
⊞ Low Latency (7)
⊞ Memory Managemen
⊞ Performance (13)
⊞ QoS (8)
⊞ Scalability (4)
⊞ SDLC (6)
⊞ Security (13)
⊞ Transaction Managen

80+ step by step Java Tutorials

open all | close all
⊞ Setting up Tutorial (6)
⊞ Tutorial - Diagnosis (2
⊞ Akka Tutorial (9)
⊞ Core Java Tutorials (2
⊞ Hadoop & Spark Tuto
⊞ JEE Tutorials (19)
⊞ Scala Tutorials (1)

```
result=None
result=0

```

The following “flatMap” and “filter” combinators

```

object ScalaForYield extends App {

  val listOfLists = List(List(1, 2, 3), List(4,

  val oddNumbers = listOfLists.flatMap { x =>  x
                              .filter {x => (x %

  println(oddNumbers); //List(1, 3, 5, 7, 9)
}

```

can be written using “for…yield” comprehension.

```

object ScalaForYield extends App {

  val listOfLists = List(List(1, 2, 3), List(4,

  val oddNumbers = for {
     list <- listOfLists   //flatMap
     eachItem <- list if(eachItem % 2 != 0)  //
  } yield eachItem


  println(oddNumbers); //List(1, 3, 5, 7, 9)
}


```

Q29. How do you handle errors more cleanly in Scala without the try..catch idioms?

A29. Having “try…catch” idiom as shown in “Q27” all over your code base can become ugly very quickly and does not go that well with functional programming. This approach will not also work well in an actor based multi-threaded code as the actor system work by receiving a message denoting the error condition.

This is where the semantics of “**try-success-failure**” come to the rescue.

⊞ Spring & HIbernate Tu
⊞ Tools Tutorials (19)
⊞ Other Tutorials (45)

100+ Java pre-interview coding tests

open all | close all
⊞ Can you write code? (
⊞ ♦ Complete the given
⊞ Converting from A to I
⊞ Designing your classe
⊞ Java Data Structures
⊞ Passing the unit tests
⊞ What is wrong with th
⊞ Writing Code Home A
⊞ Written Test Core Jav
⊞ Written Test JEE (1)

How good are your …..?

open all | close all
⊞ Career Making Know-
⊞ Job Hunting & Resum

An instance of “**Try[T]**” can be a successful computation denoting it as “**Success[T]**” by wrapping a value of type “T” or an an instance of “**Failure[T]**“, wrapping an exception. Here is an example of trying to divide a number by 0.

```scala

import scala.util.{ Try, Success, Failure }

object TrySuccessFailureExample extends App {
  def divide(x: Int, y: Int): Try[Int] = {

    val problem = Try(x / y)
    problem match {
      case Success(v) => Success(v)
      case Failure(e) => Failure(e)
    }
  }

  println(divide(5, 0)); //Failure(java.lang.Ari
  println(divide(6, 2)); //Success(3)
}

```

Q30. Can you write code to read a file named “C:\\temp\\test.txt”?

A30. The code needs to handle “Success” and “Failure” scenarios as shown below.

```scala

import scala.util.{ Try, Success, Failure }
import scala.io.Source

object ReadFile extends App {

  val fileName = "C:\\temp\\test.txt"

  //read the file & handle Success & Failure sce
  readFile(fileName) match {
    case Success(lines) => lines.foreach { x =>
    case Failure(ex) => ex.printStackTrace()
  }

  //read the file and store each line as List[St
  def readFile(filename: String): Try[List[Strin
    Try(Source.fromFile(filename).getLines.toLis
  }

}

```

Using Try/Success/Failure is an alternative to using Option/Some/None or Either/Left/Right.

Q31. How does Either-Left-Right idiom defer from the Try-Success-Failure idiom?

A31. “Either” is a container type like “Try” and “Option”. Unlike the “Try[T]” and “Option[T]” , it takes not only one, but two type parameters as in “**Either[A, B]**“. So, one of the possible usages of “Either” is for exception handling as shown below.

```scala

import scala.util.{ Try, Success, Failure }
import scala.io.Source
import java.nio.file.{ Paths, Files }

object ReadFile extends App {

  val fileName = "C:\\temp\\test.txt"

  readFile(fileName) match {
    case Left(lines) => lines.foreach { x => pri
    case Right(ex) => ex.printStackTrace()
  }

  def readFile(filename: String): Either[List[St
    if (Files.exists(Paths.get(filename))) {
      Left(Source.fromFile(filename).getLines.to
    } else {
      Right(new RuntimeException(filename + " is
    }
  }
}


```

Another usage for the “Either” is handling two possible values as shown below.

```scala

object EitherLeftOrRightExample extends App {

    def divide(x:Int, y:Int):Either[String, Int]
        if (y == 0) Left("Can't divide by 0")
        else Right(x/y)
    }

    println(divide(5,0)); //Left(Can't divide by
    println(divide(6,2)); //Right(3)
}

```

Q32. What will be the output of the following Scala code using the “for” **comprehension**? What operations will be internally performed as "for comprehension" is just a syntactic suagar?

```

import java.nio.file.Files
import java.nio.file.Paths
import scala.io.Source

object ReadFile extends App {

  val fileName = "c:/temp/test.txt"

  val result = for {
     line <-  Source.fromFile(fileName).getLines
     character <- line

  } yield character.toUpper

  println(result)
}

```

Where the "c:/temp/test.txt" contains

```

hello Scala!!!

```

A32. The output is:

```

List(H, E, L, L, O,  , S, C, A, L, A, !, !, !)

```

All capitalised, and as a "List[Char]".

Under the covers, it executes a "**map**" to convert a list of strings to "string", and a "**flatMap**" to convert all the strings to a list of chars.

```

import java.nio.file.Files
import java.nio.file.Paths
import scala.io.Source

object ReadFile extends App {

  val fileName = "/home/ucubo/test.txt"

  if (Files.exists(Paths.get(fileName))) {
    val result = Source.fromFile(fileName).getLi
      .flatMap(x => x)
  }
}

```

Popular Member Posts

♦ 11 Spring boot interview questions & answers

850 views

♦ Q11-Q23: Top 50+ Core on Java OOP Interview Questions & Answers

769 views

001A: ♦ 7+ Java integration styles & patterns interview questions & answers

399 views

18 Java scenarios based interview Questions and Answers

387 views

♦ 7 Java debugging interview questions & answers

308 views

01b: ♦ 13 Spring basics Q8 – Q13 interview questions & answers

305 views

01: ♦ 15 Ice breaker questions asked 90% of the time in Java job interviews with hints

297 views

♦ 10 ERD (Entity-Relationship Diagrams) Interview Questions and Answers

294 views

♦ Q24-Q36: Top 50+ Core on Java classes, interfaces and generics interview questions & answers

246 views

001B: ♦ Java architecture & design concepts interview questions & answers

204 views

Bio | **Latest Posts**

Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

About Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

‹ ♥♦ Key Areas driven Java resumes are more impressive

SSH Login without password so that tasks can be automated via shell scripts ›

Posted in member-paid, Scala Interview Q&As

☀ 6 Aspects that can motivate you to fast-track your career & go places ☀ Are you reinventing yourself as a Java developer? ☀ 8 tips to safeguard your Java career against offshoring ☀ My top 5 career mistakes

Prepare to succeed

☀ Turn readers of your Java CV go from “Blah blah” to “Wow”? ☀ How to prepare for Java job interviews? ☀ 16 Technical Key Areas ☀ How to choose from multiple Java job offers?

Select Category ▼

© Disclaimer

The contents in this Java-Success are copy righted. The author has the right to correct or enhance the current content without any prior notice.

These are general advice only, and one needs to take his/her own circumstances into consideration. The author will not be held liable for any damages caused or alleged to be caused either directly or indirectly by these materials and resources. Any trademarked names or labels used in this blog remain the property of their respective trademark owners. No guarantees are made regarding the accuracy or usefulness of content, though I do make an effort to be accurate. Links to external sites do not imply endorsement of the linked-to sites.

↑