Triton recruitment

Alentaris

Java – Technical interview answers

07/05/2013

Stéphane Robert

# Exercice 1 : Unit testing

Asked class is implemented here :

main/java : com.triton.DiskFreeSpaceUtils.java

We can see it in action in TestCase here :

test/java : com.triton.ExcerciceDisk.java

We can see it in action in Maven command line :

mvn test -Dtest=ExcerciceDisk

This implementation uses only JDK but we could use Jakarta IOUtils that is often used as dependencies in many projects.

# Exercice 2 : Spring

# Exercice 3 : Fib

Fibonacci has been implemented by an abstract class with 2 sub classes :

main/java : com.triton.AbstractFibonacci.java

main/java : com.triton.FibonacciRecurcive.java

main/java : com.triton.FibonacciNonRecurcive.java

We can see it in action in TestCase here :

test/java : com.triton.ExcerciceFib.java

We can see it in action in Maven command line :

mvn test -Dtest=ExcerciceFib

Advantages of using non-recursive function is mainly performance. It drastically reduces heap space and avoid problems like : **java.lang.OutOfMemoryError: Java heap space**. Because each call to a function in recursive mode is stored before application could give complete result.

Memory usage and performances should be improved using non-recursive function.

To mesure this improvement in our particular Fibonacci example, I wrote the 2 implementations and compare them in JUnit TestCase.

# Exercice 4 : Incrementing an integer

# Exercice 5 : StwAndo

StwAndo has been implemented in a TestCase here :

test/java : com.triton.ExcerciceStwAndo.java

We can see it in action in Maven command line :

mvn test -Dtest=ExcerciceStwAndo

# Exercice 6 : Algorithm

NumberToWords has been implemented by an abstract class with 1 sub class :

main/java : com.triton.AbstractNumberToWords.java

main/java : com.triton.EnglishNumberToWords.java

It is a first step if we want to implement other languages.

We can see it in action in TestCase here :

test/java : com.triton.ExcerciceNumberToWords.java

We can see it in action in Maven command line :

mvn test -Dtest=ExcerciceNumberToWords

# Exercice 7 : Java

To implement asked functionnality I wrote a class that is a wrapper over a java.util.Map. It is as generic as possible in the way you provide your own Map<K,V> (where K and V can be any type of classes) in wrapper’s constructor.

This technique uses no external libray. But we could also implement it by AOP. In this case, there would be less code to write, but it would need more dependencies.

MapWithStat has been implemented by 1 classe :

main/java : com.triton.MapWithStat.java

It’s a first step if we want to implement other languages.

We can see it in action in TestCase here :

test/java : com.triton.ExcerciceMap.java

We can see it in action in Maven command line :

mvn test -Dtest=ExcerciceMap

# Exercice 8 : Code sample