Java EE Support Patterns

Cloud, Java, Middleware, APM & Tools tutorials

java.lang.NoClassDefFoundError: How to resolve – Part 3

7/10/2012 PIERRE-HUGUES CHARBONNEAU [5 COMMENTS]

This is part 3 of our java.lang.NoClassDefFoundError troubleshooting series. As I mentioned in my first article, there are many possible problems that can lead to java.lang.NoClassDefFoundError such as a wrong Java runtime classpath. This article will describe one of the most common causes of this problem: failure of Java class static initializer blocks or static variables.

A sample Java program will be provided and I encourage you to compile and run this example from your workstation in order to properly replicate and understand this type of problem.

Java static initializer revisited

The Java programming language provides you with the capability to “statically” initialize variables or a block of code. This is achieved via the “static” variable identifier or the usage of a static {} block at the header of a Java class. Static initializers are guaranteed to be executed only once in the JVM life cycle and are Thread safe by design which make their usage quite appealing for static data initialization such as internal object caches, loggers etc.

What is the problem? I will repeat again, static initializers are guaranteed to be executed only once in the JVM life cycle…This means that such code is executed at the class loading time

FOLLOW US

YouTube

RSS and Email subscription:

Subscribe

RECENT ARTICLES

Oracle WebLogic Native IO & Java Muxers
This article will provide the complete root cause analysis details...

Oracle Open World and Java One 2016 summary later this week
This post is to inform you that I will publish...

Java 8 Performance Optimization - DZone Refcard Update

and never executed again until you restart your JVM. Now what happens if the code executed at that time (@Class loading time) terminates with an unhandled Exception?

Welcome to the java.lang.NoClassDefFoundError problem case #2!

NoClassDefFoundError problem case 2 – static initializer failure

This type of problem is occurring following the failure of static initializer code combined with successive attempts to create a new instance of the affected (non-loaded) class.

Sample Java program

** A tutorial video is now available: http://www.youtube.com/watch?v=hlg6RstHhiM

The following simple Java program is split as per below:

- The main Java program `NoClassDefFoundErrorSimulator`
- The affected Java class `ClassA`
- `ClassA` provides you with a ON/OFF switch allowing you the replicate the type of problem that you want to study

This program is simply attempting to create a new instance of `ClassA` 3 times (one after each other). It will demonstrate that an initial failure of either a static variable or static block initializer combined with successive attempt to create a new instance of the affected class triggers java.lang.NoClassDefFoundError.

```
#### NoClassDefFoundErrorSimulator.java
package org.ph.javaee.tools.jdk7.training2;

/**
 * NoClassDefFoundErrorSimulator
 * @author Pierre-Hugues Charbonneau
 *
```

I am happy to inform you that I published recently...

DZone's Guide to Building and Deploying Applications on the Cloud

This post is to inform you that DZone has just...

Java 8 - CPU Flame Graph

Brendan Gregg and Martin Spier from Netflix recently shared a...

ABOUT THE AUTHOR

PIERRE-HUGUES CHARBONNEAU

P-H is an IT Architect currently working for CGI Inc. in Canada. He has 15 years+ of experience developing and troubleshooting Java & Web enterprise systems. Email: phcharbonneau@hotmail.com.

VIEW MY COMPLETE PROFILE

BLOG ARCHIVE

► 2018 (1)

► 2016 (3)

► 2015 (6)

► 2014 (4)

```
 */
public class NoClassDefFoundErrorSimulator {

    /**
     * @param args
     */
    public static void main(String[] args) {
        System.out.println("java.lang.NoClassDefFoundError Simulator - Training 2");
        System.out.println("Author: Pierre-Hugues Charbonneau");
        System.out.println("http://javaeesupportpatterns.blogspot.com\n\n");

        try {
            // Create a new instance of ClassA (attempt #1)
            System.out.println("FIRST attempt to create a new instance of ClassA...\n");
            ClassA classA = new ClassA();

        } catch (Throwable any) {
            any.printStackTrace();
        }

        try {
            // Create a new instance of ClassA (attempt #2)
            System.out.println("\nSECOND attempt to create a new instance of ClassA...\n");
            ClassA classA = new ClassA();

        } catch (Throwable any) {
            any.printStackTrace();
        }

        try {
            // Create a new instance of ClassA (attempt #3)
            System.out.println("\nTHIRD attempt to create a new instance of ClassA...\n");
            ClassA classA = new ClassA();
```

- ► 2013 (18)
- ▼ 2012 (40)
  - ► December (2)
  - ► November (4)
  - ► October (2)
  - ► September (2)
  - ► August (3)
  - ▼ July (4)
    - Oracle Service Bus – Stuck Thread Case Study
    - 5 tips for proper Java Heap size
    - java.lang.NoClassDefFoundError: How to resolve – P...
    - How to analyze Thread Dump – Part 5: Thread Stack ...
  - ► June (4)
  - ► May (1)
  - ► April (2)
  - ► March (4)
  - ► February (8)
  - ► January (4)
- ► 2011 (52)
- ► 2010 (1)

```
        } catch (Throwable any) {
            any.printStackTrace();
        }

        System.out.println("\n\ndone!");
    }
}
```

```
#### ClassA.java
package org.ph.javaee.tools.jdk7.training2;

/**
 * ClassA
 * @author Pierre-Hugues Charbonneau
 *
 */
public class ClassA {

    private final static String CLAZZ = ClassA.class.getName();
    // Problem replication switch ON/OFF
    private final static boolean REPLICATE_PROBLEM1 = true; // static variable initializer
    private final static boolean REPLICATE_PROBLEM2 = false; // static block{} initializer

    // Static variable executed at Class loading time
    private static String staticVariable = initStaticVariable();

    // Static initializer block executed at Class loading time
    static {

        // Static block code execution...
        if (REPLICATE_PROBLEM2) throw new IllegalStateException("ClassA.static{}: Internal Error!");
    }

    public ClassA() {
        System.out.println("Creating a new instance of "+ClassA.class.getName()+"...");
    }

    /**
     *
     * @return
     */
    private static String initStaticVariable() {

        String stringData = "";

        if (REPLICATE_PROBLEM1) throw new IllegalStateException("ClassA.initStaticVariable(): Internal Error!");

        return stringData;
    }
}
```

Problem reproduction

In order to replicate the problem, we will simply “voluntary” trigger a failure of the static initializer code. Please simply enable the problem type that you want to study e.g. either static variable or static block initializer failure:

```
    // Problem replication switch ON (true) / OFF (false)
    private final static boolean REPLICATE_PROBLEM1 = true; // static variable initializer
    private final static boolean REPLICATE_PROBLEM2 = false; // static block{} initializer
```

Now, let's run the program with both switch at OFF (both boolean values at false)

Baseline (normal execution)

```
java.lang.NoClassDefFoundError Simulator - Training 2
Author: Pierre-Hugues Charbonneau
http://javaeesupportpatterns.blogspot.com



FIRST attempt to create a new instance of ClassA...

Creating a new instance of org.ph.javaee.tools.jdk7.training2.ClassA...

SECOND attempt to create a new instance of ClassA...

Creating a new instance of org.ph.javaee.tools.jdk7.training2.ClassA...

THIRD attempt to create a new instance of ClassA...

Creating a new instance of org.ph.javaee.tools.jdk7.training2.ClassA...

done!
```

For the initial run (baseline), the main program was able to create 3 instances of `ClassA` successfully with no problem.

Problem reproduction run (static variable initializer failure)

```
java.lang.NoClassDefFoundError Simulator - Training 2
Author: Pierre-Hugues Charbonneau
http://javaeesupportpatterns.blogspot.com
```

```
FIRST attempt to create a new instance of ClassA...

java.lang.ExceptionInInitializerError
        at org.ph.javaee.tools.jdk7.training2.NoClassDefFoundErrorSimulator.main(NoClassDefFoundErrorSimulator.java:21)
Caused by: java.lang.IllegalStateException: ClassA.initStaticVariable(): Internal Error!
        at org.ph.javaee.tools.jdk7.training2.ClassA.initStaticVariable(ClassA.java:37)
        at org.ph.javaee.tools.jdk7.training2.ClassA.<clinit>(ClassA.java:16)
        ... 1 more

SECOND attempt to create a new instance of ClassA...

java.lang.NoClassDefFoundError: Could not initialize class org.ph.javaee.tools.jdk7.training2.ClassA
        at org.ph.javaee.tools.jdk7.training2.NoClassDefFoundErrorSimulator.main(NoClassDefFoundErrorSimulator.java:30)

THIRD attempt to create a new instance of ClassA...

java.lang.NoClassDefFoundError: Could not initialize class org.ph.javaee.tools.jdk7.training2.ClassA
        at org.ph.javaee.tools.jdk7.training2.NoClassDefFoundErrorSimulator.main(NoClassDefFoundErrorSimulator.java:39)


done!
```

Problem reproduction run (static block initializer failure)

```
java.lang.NoClassDefFoundError Simulator - Training 2
Author: Pierre-Hugues Charbonneau
http://javaeesupportpatterns.blogspot.com


FIRST attempt to create a new instance of ClassA...

java.lang.ExceptionInInitializerError
```

```
        at
org.ph.javaee.tools.jdk7.training2.NoClassDefFoundErrorSimulator.main(NoClassDefFoundErrorSimulator.java:21)
Caused by: java.lang.IllegalStateException: ClassA.static{}: Internal Error!
        at org.ph.javaee.tools.jdk7.training2.ClassA.<clinit>(ClassA.java:22)
        ... 1 more

SECOND attempt to create a new instance of ClassA...

java.lang.NoClassDefFoundError: Could not initialize class org.ph.javaee.tools.jdk7.training2.ClassA
        at
org.ph.javaee.tools.jdk7.training2.NoClassDefFoundErrorSimulator.main(NoClassDefFoundErrorSimulator.java:30)

THIRD attempt to create a new instance of ClassA...

java.lang.NoClassDefFoundError: Could not initialize class org.ph.javaee.tools.jdk7.training2.ClassA
        at
org.ph.javaee.tools.jdk7.training2.NoClassDefFoundErrorSimulator.main(NoClassDefFoundErrorSimulator.java:39)


done!
```

What happened? As you can see, the first attempt to create a new instance of ClassA did trigger a `java.lang.ExceptionInInitializerError`. This exception indicates the failure of our static initializer for our static variable & bloc which is exactly what we wanted to achieve.

The key point to understand at this point is that this failure did prevent the whole class loading of `ClassA`. As you can see, attempt #2 and attempt #3 both generated a java.lang.NoClassDefFoundError, why? Well since the first attempt failed, class loading of `ClassA` was prevented. Successive attempts to create a new instance of `ClassA` within the current ClassLoader did generate java.lang.NoClassDefFoundError over and over since `ClassA` was not found within current ClassLoader.

As you can see, in this problem context, the NoClassDefFoundError is just a *symptom* or *consequence* of another problem. The original problem is the ExceptionInInitializerError triggered following the failure of the static initializer code. This clearly demonstrates the importance of proper error handling and logging when using Java static initializers.

Recommendations and resolution strategies

Now find below my recommendations and resolution strategies for NoClassDefFoundError problem case 2:

- Review the java.lang.NoClassDefFoundError error and identify the missing Java class
- Perform a code walkthrough of the affected class and determine if it contains static initializer code (variables & static block)
- Review your server and application logs and determine if any error (e.g. ExceptionInInitializerError) originates from the static initializer code
- Once confirmed, analyze the code further and determine the root cause of the initializer code failure. You may need to add some extra logging along with proper error handling to prevent and better handle future failures of your static initializer code going forward

Please feel free to post any question or comment.
The part 4 is now available.

Posted in: ClassLoader,Java

Newer Post Home Older Post

5 comments:

Shimray says:
July 20, 2012 at 5:54 AM

Yes, thank you for the explaination.

Reply

P-H says:
August 23, 2012 at 12:32 PM

Hi all,

Please note that the part 4 is now available and provides detail on class loader related problems for parent first delegation model.

http://javaeesupportpatterns.blogspot.com/2012/08/javalangnoclassdeffounderror-parent.html

Thanks.
P-H

Reply

Anonymous says:
December 20, 2012 at 9:27 AM

Extremely helpful. Thanks for posting!

Reply

Anonymous says:
June 16, 2014 at 1:07 AM

I am getting the following error when I run my main class.
NoClassDefFoundError: abc.abc.abc.JavaClass (initialization failure).

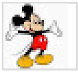
My code which tries to create an instance and fails is as follows:
JavaClass jc= JavaClass.getInstance();

and getInstance() is defined as follows:

```
public static JavaClass inst;
public static JavaClass getInstance() {
if (inst == null)
inst = new JavaClass();
return inst;
}
```

Please guide.

Reply



Raja Suba says:

May 10, 2017 at 3:10 AM

Very useful one ! Thank you so much for such a nice post :)

Reply

Post a Comment

Enter your comment...

Comment as: Amitabh Mandal

Sign out

Publish

Preview

□ Notify me