context class loader

Asked 15 years, 10 months ago Modified 5 years, 9 months ago

Viewed 3k times

5

What is the difference between?

Class.getClassLoader() and Thread.getContextClassLoader()?

java

Share

Improve this question

Follow

asked Feb 23, 2009 at 11:45

michal2

possible duplicate of Difference between thread's context class loader and normal classloader – sleske Jan 14, 2014 at 13:29

2 Answers

Sorted by: Highest score (default)

7

From this thread:

> `Class.getClassLoader()` returns the `ClassLoader` that loaded the class it is invoked on.
>
> `Thread.getContextClassLoader()` returns the `ClassLoader` set as the context `ClassLoader` for the `Thread` it is invoked on, which can be different from the `ClassLoader` that loaded the Thread class itself if the Thread's `setContextClassLoader(ClassLoader)` method has been invoked.
>
> This can be used to allow the object starting a thread to specify a `ClassLoader` that objects running in the thread should use, but the cooperation of some of those objects is required for that to work.

So if you are wondering why there are 3 API calls

```
Thread<instance>.getContextClassLoader()
Thread<instance>.getClassLoader()
Class<instance>.getClassLoader()
```

, you can find in this Find a way out of the ClassLoader maze an answer.

> Why do thread context classloaders exist in the first place? They were introduced in J2SE without much fanfare. A certain lack of proper guidance and documentation from Sun Microsystems likely explains why many developers find them confusing.
>
> In truth, context classloaders provide a back door around the classloading delegation scheme also introduced in J2SE. Normally, all classloaders in a JVM are organized in a hierarchy such that every classloader (except for the primordial classloader that bootstraps the entire JVM) has a single parent.
>
> `[...]` To make matters worse, certain application servers set context and current classloaders to different ClassLoader instances that have the same classpaths and yet are not related as a delegation parent and child.

Share Improve this answer Follow

edited Jun 20, 2020 at 9:12

Community Bot
1 ● 1

answered Feb 23, 2009 at 11:49

VonC
1.3m ● 558 ● 4.7k ● 5.6k

3

Class.getClassLoader() returns the classloader used to load that particular class.

Thread.getContextClassLoader() gets the ClassLoader set as **context** for that thread, which will be used to load needed classes.

This makes a difference for example in Tomcat. For java.lang.String the classloader will be the top level JVM classloader which won't have all classes on your classpath, but the Thread context classloader should do.

edited Mar 19, 2019 at 11:57
Hearen
7,778 ● 4 ● 57 ● 73

answered Feb 23, 2009 at 11:48
Nick Fortescue
44.1k ● 26 ● 108 ● 136