Javarevisited

Blog about Java, Programming, Spring, Hibernate, Interview Questions, Books and Online Course Recommendations from Udemy, Pluarlsight etc

Home core java spring hibernate collections multithreading design patterns interview questions coding data structure OOP java 8 books About Me

Java Certifications JDBC jsp-servlet JSON SQL Linux Courses online resources jvm-internals REST Eclipse jQuery Java IO Java XML

SATURDAY, FEBRUARY 4, 2017

Why wait, notify and notifyAll is defined in Object Class and not on Thread class in Java

Why to wait, notify and notifyAll is declared in Object Class instead of Thread is famous core java interview question which is asked during all levels of Java interview ranging from 2 years, 4years to quite senior level position on java development. Beauty of this question is that it reflect what does interviewee knows about wait notify mechanism, how does it sees whole wait and notify feature and whether his understanding is not shallow on this topic. Like Why Multiple inheritance is not supported in Java or why String is final in java there could be multiple answers of *why wait and notify is defined in Object class* and every one could justify there reason.

In my all interview experience I found that wait and notify still remains most confusing for most of Java programmer specially up-to 2 to 3 years and if they asked to write code using wait and notify they often struggle. So if you are going for any Java interview make sure you have sound knowledge of wait and notify mechanism as well as you are comfortable writing code using wait and notify like Produce Consumer problem or implementing Blocking queue etc. by the way This article is in continuation of my earlier article related to wait and notify e.g. Why Wait and notify requires to be called from Synchronized block or method and Difference between wait, sleep and yield method in Java , if you haven't read you may found interesting.

Reason Why Wait , Notify and NotifyAll are in Object Class.

Here are some thoughts on why they should not be in Thread class which make sense to me :

Interview Questions

core java interview question (168)
Coding Interview Question (72)
data structure and algorithm (68)
interview questions (46)
SQL Interview Questions (30)
design patterns (30)
object oriented programming (30)
thread interview questions (30)
collections interview questions (25)
spring interview questions (17)
database interview questions (16)
servlet interview questions (15)
Programming interview question (6)
hibernate interview questions (6)

Best of Javarevisited

How Spring MVC works internally?
How to design a vending machine in Java?
How HashMap works in Java?
Why String is Immutable in Java?
10 Articles Every Programmer Must Read
How to convert lambda expression to method reference in Java 8?
10 Tips to improve Programming Skill

1) Wait and notify is not just normal methods or synchronization utility, more than that they are **communication mechanism between two threads in Java**. And Object class is correct place to make them available for every object if this mechanism is not available via any java keyword like synchronized. Remember synchronized and wait notify are two different area and don’t confuse that they are same or related. Synchronized is to provide mutual exclusion and ensuring thread safety of Java class like race condition while wait and notify are communication mechanism between two thread.

2)**Locks are made available on per Object basis**, which is another reason wait and notify is declared in Object class rather then Thread class.

3) In Java in order to enter critical section of code, Threads needs lock and they wait for lock, they don't know which threads holds lock instead they just know the lock is hold by some thread and they should wait for lock instead of knowing which thread is inside the synchronized block and asking them to release lock. this analogy fits with wait and notify being on object class rather than thread in Java.

These are just my thoughts on **why wait and notify method is declared in Object class rather than Thread in Java** and you have different version than me. In reality its another design decision made by Java designer like not supporting Operator overloading in Java. Anyway please post if you have any other convincing reason *why wait and notify method should be in Object class and not on Thread*.

Update:
@Lipido has made an insightful comment , which is worth adding here. read his comment for full text

> "Java is based on Hoare's monitors idea (http://en.wikipedia.org/wiki/Monitor_%28synchronization%29). In Java all object has a monitor. Threads waits on monitors so, to perform a wait, we need 2 parameters:
> - a Thread
> - a monitor (any object)

10 OOP design principles programmer should know
How Synchronization works in Java?
10 tips to work fast in Linux
5 Books to improve Coding Skills

Java Tutorials

date and time tutorial (21)
FIX protocol tutorial (15)
Java Certification OCPJP SCJP (22)
java collection tutorial (73)
java IO tutorial (28)
Java JSON tutorial (12)
Java multithreading Tutorials (55)
Java Programming Tutorials (18)
Java xml tutorial (16)
JDBC (29)
jsp-servlet (37)
online resources (79)

Categories

SQL (53)
linux (36)
courses (30)
database (30)
Eclipse (28)
JVM Internals (22)
Java Certification OCPJP SCJP (22)
JQuery (16)
REST (16)
Testing (11)
Maven (10)
general (10)

> In the Java design, the thread can not be specified, it is always the current thread running the code. However, we can specify the monitor (which is the object we call wait on). This is a good design, because if we could make any other thread to wait on a desired monitor, this would lead to an "intrusion", posing difficulties on designing/programming concurrent programs. Remember that in Java all operations that are intrusive in another thread's execution are deprecated (e.g. stop())."

Further Learning
Multithreading and Parallel Computing in Java
Java Concurrency in Practice - The Book
Applying Concurrency and Multi-threading to Common Java Patterns
Java Concurrency in Practice Course by Heinz Kabutz

Some **Java articles on Thread** you may like

When to use Thread or Runnable interface in Java?

How to Stop Thread in Java

How to implement Thread in Java

Top 15 Thread interview questions in Java

How to prevent deadlock in Java Code

What does Volatile keyword do in Java

By Javin Paul at February 04, 2017
Labels: core java , core java interview question , Java multithreading Tutorials , thread interview questions
Location: United States

27 comments :

Anonymous said...

why wait and notify method in object because threads wait for lock and lock is implemented on Object level. its as simple as this.

Blog Archive

- ► 2018 (87)
- ▼ 2017 (856)
  - ► December (100)
  - ► November (38)
  - ► October (52)
  - ► September (47)
  - ► August (62)
  - ► July (60)
  - ► June (65)
  - ► May (49)
  - ► April (80)
  - ► March (104)
  - ▼ February (88)
    - Difference between Executor, ExecutorService and E...
    - Why Catching Throwable or Error is bad?
    - How to use ConcurrentHashMap in Java - Example Tut...
    - How to Count Number of Words in String - Java Codi...
    - How to connect to Microsoft SQL Server database us...
    - How to convert Char to String in Java with Example...
    - 10 Example of Lambda Expressions and Streams in Ja...
    - How to get Key From Value in Hashtable, HashMap or...
    - How to Setup Java Remote Debugging in Eclipse - St...

February 5, 2012 at 5:40 PM

guru said...

Recently this questions "why wait and notify method in object class" were asked to my friend and he was quite happy to answer it correctly. its mostly because of locking.

February 5, 2012 at 5:42 PM

Kazi Abdullah Saikat said...

The only reason to have wait/notify/notifyAll methods on Object is to allow locking on a per-object basis.

If you think about *just* a communication mechanism between multiple threads, then it would have been enough to have special lock objects and have these methods on that instead of the general Object class.

February 7, 2012 at 2:20 AM

Javin @ run Java program from command prompt said...

@Anonymous, Thanks for your comment, I respect your opinion but would have been better if you post your convincing reason on why wait and notify is declared in Object Class?

February 7, 2012 at 5:14 AM

Javin @ Java Iterator Example said...

@Kazi, Definitely lock is one of the primary reason why wait and notify is defined in Object class. Point to mention thread communication mechanism was due to separate wait() from Sleep() method which is also used to put Thread on wait or pause but can not be used to communicate between Thread.

February 7, 2012 at 5:17 AM

Javi said...

Instead of having a lock in every object maybe it would have been wiser to create a special class supporting it... Most of the time it's not helpful to see the wait/wait/notify/notifyAll methods every time you use a dot over a reference. Just my two cents.

February 8, 2012 at 8:52 AM

Lipido said...

Java is based on Hoare's monitors idea (http://en.wikipedia.org/wiki/Monitor_%28synchronization%29). In Java all object has a monitor. Threads waits on monitors so, to perform a wait, we need 2 parameters:
- a Thread
- a monitor (any object)

How to Check or Detect Duplicate Elements in Array...

String Deduplication of G1 Garbage collector to Sa...

Data Access Object (DAO) design pattern in Java - ...

How to solve java.util.NoSuchElementException in J...

How to Read File in One Line in JDK 7 or Java 8

How to Format and Display Number to Currency in Ja...

Overriding equals() and hashCode() method in Java ...

5 Differences between COALESCE and ISNULL in SQL S...

Top 20 jQuery Interview Questions and Answers

2 Ways to Combine Arrays in Java - Integer, String...

5 Difference between Constructor and Static Factor...

How to execute native shell commands from Java Pro...

How to set JAVA_HOME environment in Linux, Unix an...

What is Race Condition in multithreading - 2 Examp...

JSTL Set tag examples or <c:set> in JSP - Java J2E...

Top 5 JavaScript Books to Learn - Best of Lot, Mus...

Fixing java.net.BindException: Cannot assign reque...

Difference between throw and throws in Exception h...

How to Swap Two Numbers Without Temp or Third Vari...

In the Java design, the thread can not be specified, it is always the current thread running the code. However, we can specify the monitor (which is the object we call wait on). This is a good design, because if we could make any other thread to wait on a desired monitor, this would lead to an "intrusion", psosing difficulties on designing/programming concurrent programs. Remember that in Java all operations that are intrusive in another thread's execution are deprecated (e.g. stop()).

In addition, I think synchronized is not different. It is very related because it is needed to complete the monitor model:

-OPERATION 1) To acquire a monitor:
- a) synchronized block: synchronized(instance){ code }
- b) synchronized method: the same as putting synchronized(this) as the first line
- b) static synchronized method: the same as putting synchronized(CurrentClass.class){...}

- OPERATION 2) To release a monitor and block: wait operation. Releases the monitor and wait for the arrival of notify. This is because we need to have the objects monitor (synchronized needed). When notify arrives, it will try to acquire the monitor again.

- OPERATION 3) To make other threads to stop waiting on monitor. notify. To explore which threads are waiting and to select one of them, we also need to have the monitor (synchronized required). One thread will be n

February 12, 2012 at 3:00 AM

Javin @ java hashtable example said...

@Lipido, Thanks for your comment.you are amazingly clear on first paragraph on highlighting why wait and notify should be in Object class. I am including your first paragraph on main article for better visibility. on synchronized perspective I am not agreed yes they are not absolutely unrelated but purpose of both are different synchronized is means to acquire and release lock and ensuring atomicity, ordering and visibility while wait and notify is used for thread communication. Once again thanks for adding value.

February 13, 2012 at 5:01 AM

Anuj said...

One more reason why wait and notify are in Object Class and not inside Thread class is because if there would be another thread like Process than it can reuse same wait and notify mechanism for communication. kind of inter process communication. wait and notify are abstraction of inter process communication and that's why declared in Object class.

April 10, 2012 at 7:50 PM

Nomad said...

10 points about volatile modifier or field in Java...
Insertion Sort Algorithm in Java with Example and ...
Top 30 Java Phone Interview Questions Answers for ...
How to Install JDK on Windows - Java Programming...
FIX Protocol tutorials: Difference between Session...
Difference between LinkedList vs ArrayList in Java...
Difference between Comparison (QuickSort) and Non-...
How to disable JUnit Test - @Ignore annotation Exa...
How to encode decode String in Java base64 Encodin...
Difference between UTF-8, UTF-16 and UTF-32 Charac...
How to Create Tabs UI using HTML, CSS, jQuery, JSP...
Repeating groups in FIX Protocol
How to create and call stored procedure in MySQL w...
Top 5 Core Java Books for Beginners - Learn Best o...
How to set Java Path and Classpath in Windows 8 an...
Producer Consumer Design Pattern with Blocking Que...
How to convert XMLGregorianCalendar to Date to XML...
Top 6 SQL Query Interview Questions and Answers
Why non-static variable cannot be referenced from ...

To put it in simple words
Wait , notify , notifyall methods are only required when you want your threads to access a shared resource and a shared resource could be any java object which is on the heap.So, these methods are defined on the core Object class so that each object has control of allowing Threads to wait on it's monitor. Java doesn't have any special object which is used for sharing a common resource. No such data structure is defined.So, onus is given on the Object class to be able to become shared resource providing it will helper methods like wait(),notify() and notifyAll();

June 15, 2012 at 7:17 PM

tofeeq ahmad said...

your tutorial help a lot to me even now i very much aware to java.

June 25, 2012 at 12:16 AM

Niraj Singh said...

Here are my two cents. The Thread-Object communication should be in a publish subscribe model rather than polling model.

In case of polling model the Thread will have the responsibility of constantly observing the Object's state and take necessary actions. This is obviously not desirable as it will have performance overhead of checking time and the again. The onus of communicating the state change should lie with the object itself. And it achieves that by help of wait/notify methods.

January 22, 2013 at 5:52 AM

kumar naidu said...

One more reason why wait notify notifyall in Object class because the Thread class extends Object class and Thread need lock and the locks defined in Object class.

March 2, 2013 at 2:38 AM

kailash said...

this is because we lock object, we wait on object and we notify availability of object

August 19, 2013 at 1:52 AM

vishwa said...

As Object represent the state of a class and as per design perspective, here we wait for object to attain some state after which we can execute some specific operations on the object and take it to different state. In the life cycle of an object many a times object needs to satisfy some precondition before some operations happen on that particular object. And the primary purpose of Thread is to interact with different objects and manage a line of execution , if wait

How to Add Leading Zeros to Integers in Java - Str...

How to Implement Thread in Java with Example

Why Static Code Analysis is Important?

Solving java.lang.ArrayindexOutOfBoundsException: ...

How to Find First and Last element in LinkedList J...

Top 5 Websites for Practicing Data structures and ...

Difference between Right shift and Unsigned right ...

5 JSTL Core IF Tag Examples in JSP - Tutorial

Display tag Pagination, Sorting Example in JSP and...

How to convert JSON String to Java object - Jackso...

fail-safe vs fail-fast Iterator in Java

How to Consume JSON from RESTful Web Service and C...

Constructor vs Init method in Servlet - JEE Interv...

How to Implement Linked List in Java With JUnit Te...

How to remove all special characters from String i...

Introduction of How Android Works for Java Program...

17 Examples of Calendar and Date in Java

How to fix java.lang.ClassNotFoundException: org.a...

How to Fix java.net.ConnectException: Connection r...

Java Comparable Example for Natural Order Sorting

and notify provided in Thread class it means we are waiting for line of execution(not on target object) to achieve some state so that we can perform some operation on target object.

September 23, 2013 at 5:00 AM

Nitin Banarasi said...

For better understanding why wait() and notify() method belongs to Object class, I'll give you a real life example: Suppose a gas station has a single toilet, the key for which is kept at the service desk. The toilet is a shared resource for passing motorists. To use this shared resource the prospective user must acquire a key to the lock on the toilet. The user goes to the service desk and acquires the key, opens the door, locks it from the inside and uses the facilities.

Meanwhile, if a second prospective user arrives at the gas station he finds the toilet locked and therefore unavailable to him. He goes to the service desk but the key is not there because it is in the hands of the current user. When the current user finishes, he unlocks the door and returns the key to the service desk. He does not bother about waiting customers. The service desk gives thekey to the waiting customer. If more than one prospective user turns up while the toilet is locked, they must form a queue waiting for the key to the lock. Each thread has no idea who is in the toilet.

Obviously in applying this analogy to Java, a Java thread is a user and the toilet is a block of code which the thread wishes to execute. Java provides a way to lock the code for a thread which is currently executing it using the synchorinized keywokd, and making other threads that wish to use it wait until the first thread is finished. These other threads are placed in the waiting state. Java is NOT AS FAIR as the service station because there is no queue for waiting threads. Any one of the waiting threads may get the monitor next, regardless of the order they asked for it. The only guarantee is that all threads will get to use the monitored code sooner or later.

Finally the answer to your question: the lock could be the key object or the service desk. None of which is a Thread.

However, these are the objects that currently decide whether the toilet is locked or open. These are the objects that are in a position to notify that the bath room is open (“notify”) or ask people to wait when it is locked wait.

October 17, 2013 at 12:26 PM

ADITYA said...

my view is ..
wait(), notify(), notifyAll() methods are not only related to Thread objects. These are related to all java objects.
For example: If a customer Object wants to withdraw money from his back ac. at the same time another person crediting to that ac. Here if we call wait() method on the first customer object, that he waits till the second person credits the money to his ac. Once the credit completes 2nd person will call notify() method to notify the first.

The conclusion is: We have this requirement on all java objects including Threads. Thats why these methods are available in Object class for global use.

Top 5 Concurrent Collections from JDK 5 and 6 Java...

Top 10 Java wait, notify, Locking and Synchronizat...

What is blocking methods in Java and how do deal w...

How to Read/Write from RandomAccessFile in Java

How to Join Multiple Threads in Java - Thread Join...

FIX Protocol Session or Admin messages tutorial

Difference between Association, Composition and Ag...

Why wait, notify and notifyAll is defined in Objec...

How to disable submit button in HTML JavaScript to...

Introduction to Tibco Hawk as Tibco Tutorial

Top 5 Books to learn REST and RESTful WebServices ...

How to Filter Collections in Java 8 with Streams a...

What is Timer and TimerTask in Java - Tutorial Exa...

Difference between JAX-RS, Restlet, Jersey, RESTEa...

How to Fix Must Override a Superclass Method Error...

Top 50 Programmer Phone Interview Questions with A...

How HashMap works in Java

StringTokenizer Example in Java with Multiple Deli...

3 Example to Compare two Dates in Java

Difference between instance class and local variab...

February 4, 2014 at 10:38 PM

vimal thaker said...

can we avoid wait() , notify() or notifyAll() while using any object ? i was asked question "Suppose my application is single threaded and I am not going to use wait() or notify() any where in application then why should i borrow that load in my application ? " does any option to avoid this load or any other importance of the method other than synchronization.

April 5, 2014 at 6:57 AM

Anonymous said...

@Nitin Banarasi excellent explanation

September 30, 2014 at 8:47 AM

Anonymous said...

@Nitin Banarasi seriously, apt explantion .

January 20, 2015 at 2:36 PM

Chema said...

And why not Java created a special class called Monitor with these methods ?
If I would need an object for holding locks , I would create a new Monitor()

Indeed, when I create a new monitor, I usually do "Object monitor = new Monitor()"

May 11, 2016 at 4:50 AM

Anonymous said...

great explanation with real time example

June 1, 2016 at 1:53 AM

Anonymous said...

those methods can be put in Thread class also(Just assume, nothing to loose). In case of static synchronized method, the lock should be acquired at class level but not object level. How can you support from this aspect.

July 29, 2016 at 8:50 AM

sajal goyal said...

5 ways to check if String is empty in Java - examp...

▶ January (111)

▶ 2016 (135)

References

1. Oracle's Java Tech Network
2. jQuery Documentation
3. Microsoft SQL Server Documentation
4. Java SE 8 API Documentation
5. Spring Documentation
6. Oracle's JAva Certification
7. Spring Security 5 Documentation

Pages

Privacy Policy

had wait and notify been at thread level then lock would also have been at thread level and thread includes multiple objects which would have locked so many objects .To avoid unnecessary locking , its better to have lock on single object which is supposed to be synchronized

January 10, 2017 at 3:37 AM

Anonymous said...

why wait and notify defined in object class not in thread class. If suppose that defined in thread class with lock mechanism then it will work happily or any problem for execution plz explain??.

May 22, 2017 at 10:31 PM

Arun Singh said...

I wud say,
A thread may have more than one object locked. so obj.wait() clears which object's lock has to be released. So the object calls wait() method directly.

July 11, 2017 at 7:17 AM

Unknown said...

Well These methods should be in object class because Thread can call these methods on any JAVA OBJECT.Thats the simple answer

January 12, 2018 at 6:19 AM

Post a Comment

Enter your comment...

Comment as: Amitabh Mandal

Sign out

Publish

Preview

□ Notify me

Newer Post Home Older Post

Subscribe to: Post Comments (Atom)

Search This Blog

Search