SSC - Concurrency and Multi-threading Interview questions

Shan He

School for Computational Science
University of Birmingham

Module 06-19321: SSC
Outline of Topics

Interview questions
Interview questions

- Q1: What is the difference between Process and Thread?
- Q2: What are the benefits of multi-threaded programming?
- Q3: What is context-switching in multi-threading?
- Q4: Can main() finishes before other threads it created?
- Q5: How can we create a Thread in Java?
- Q6: What are different states in lifecycle of Thread?
Interview questions

- Q7: Can we call run() method of a Thread class?
- Q8: What do you understand about Thread Priority?
- Q9: You have thread T1, T2 and T3, how will you ensure that thread T2 run after T1 and thread T3 run after T2?
- Q11: Can you restart a thread?
Interview questions

From *Java Interview Questions - Concurrency*

- **Q12:** Describe synchronization in respect to multithreading.
  
  **Answer:** From lecture notes “Synchronisation (I) - Issues in multithreading”: Threads communicate by sharing the same memory space, e.g., access to fields and the objects reference fields refer to. It is efficient, but makes two kinds of errors possible:
    
    - Thread interference
    - Memory consistency errors

  We need synchronization to solve the above two problems.

- **Q13:** What is memory consistency errors?
  
  **Answer:** From lecture notes “Synchronisation (I)” different threads have inconsistent views of the same data.
Interview questions

From *Java Interview Questions - Concurrency*

- **Q14:** What is a deadlock?
  - **Answer:** From slide “Synchronisation (II)”: a situation threads are blocked forever, waiting for each other:
    - Can be two threads waiting for another to release a lock
    - Can be more than two processes are waiting for resources in a circular chain

- **Q15:** What is liveness in concurrent programming?
  - **Answer:** From slide “Synchronisation (II)”: Liveness property in concurrent programming means something good eventually happens. More specifically, the ability of an application to execute in a timely manner.
Interview questions

From Java Interview Questions - Concurrency

- Q16: What are atomic operations?
- Answer: From slide “Synchronisation (I)”: an action either happens completely, or it doesn’t happen at all.

- Q17: Explain different ways of using a thread.
- Answer: From the web page: “You can inherit from Thread class or implement the Runnable interface.” which can be found from my lecture notes. But we also know from Lecture notes “High Level Concurrency”, we can also use Java Executor Framework.
Interview questions

From *Java Interview Questions - Concurrency*

- Q16: Why would you use a synchronized block instead of synchronized method?
- Answer: From slide “Synchronisation (II)”:  
  - A synchronised method synchronises on the method’s object instance or the class: cannot synchronise some parts of your code → coarse grained, lack of granular control over lock.
  - Thread synchronises on the whole object instead of a smaller critical section → slow;

However, I missed this “Also, only with blocks you can set an intrinsic lock on an external object.”
Interview questions

From Java Interview Questions - Concurrency

- Q17: Explain different ways of using a thread.
  - Answer: From the web page: “You can inherit from Thread class or implement the Runnable interface.” which can be found from my lecture notes. But we also know from Lecture notes “High Level Concurrency”, we can also use Java Executor Framework.

- Q18: What’s the difference between the methods `sleep()` and `wait()`?
  - Answer: `sleep()` temporarily ceases execution of the thread for a specified time. `wait()` ceases execution until `notify` is called on the object or `notifyAll` is called by another thread.
Q19: How would you describe Producer Consumer problem in Java?

Answer: From my lecture notes “Producer Consumer Model and Thread Coordination”:

- Two threads: the producer and the consumer
- A shared buffer: a fixed-size queue.
- The produce: generating a piece of data, putting it into the buffer and start again.
- The consumer: removing the data continuously from the buffer one piece at a time
- Requirement: the producer won’t try to add data into the buffer if it’s full and that the consumer won’t try to remove data from an empty buffer.
- Everyday examples everywhere: rotating sushi bar
Interview questions

From Top 50 Java Interview Questions for Investment Banking Domain

Q19: What is difference between Callable and Runnable?
Answer: From my lecture notes “High Level Concurrency”:
Runnable vs Callable:
  ▶ Similarity: both are designed for classes whose instances are potentially executed by another thread.
  ▶ Difference: Runnable does not return a result and cannot throw a exception.
  ▶ Callable = Runnable objects with a return value.

Note: The answer on the web: Callable “must be executed using a ExecutorService” is incorrect!
Try this example
Exam questions

- Q1: Explain what is a thread pool? What are the advantages of thread pools? (2009)
- Answer: From my lecture notes “High Level Concurrency”:
  - Thread pool: a managed collection of worker threads that are created and waiting to perform tasks, also contains a job queue which holds tasks waiting to get executed.
  - Benefits of thread pools:
    - Improved performance when executing large numbers of tasks: reuses worker threads to reduce per-task invocation overhead.
    - A means of bounding the resources consumed by threads when executing a collection of tasks.
    - No management of the life cycle of threads. You just need to focus on the tasks that you want the threads to perform, instead of creating, managing and coordinating threads.
2. The following code is part of an airline booking system that is designed to be used in a single threaded application.

```java
public class SeatCounter{
    private int count = 0;

    public int getSeatCount(){
        return count;
    }

    public void bookSeat(){
        count++;
    }

    public void unBookSeat(){
        count--;
    }
}
```
Exam questions

(a) Describe a potential problem with this code when used with multiple concurrent threads. Provide an example that demonstrates the problem. [6%]

(b) Fix the code so that it works when used with multiple threads. [3%]