The "Mobile Cloud Computing with Android" Specialization Learn More ×

Week 1 Quiz

Help

The **due date** for this quiz is **Sun 6 Jul 2014 9:32 AM PDT**.

- [x] **In accordance with the Coursera Honor Code, I (Pablo Perotti) certify that the answers here are my own work.**

Thank you!

Question 1

Which of the following are motivations for concurrency described in these videos?

- [x] Simplify program structure relative to event-driven programming
- [x] Enhance performance on multi-core platforms
- [x] Improve perceived responsiveness
- [ ] Make the program easier to debug
- [x] Make the program behave more deterministically with respect to runtime execution order

Question 2

According to the videos, which of the following are reasons why purely event-driven software is hard to program?

- [x] It's hard to optimize its performance
- [ ] It's not portable across operating systems
- [x] It's behavior is non-deterministic on multi-core hardware
- [ ] The structure of its control flow is obscured in both time and space

Question 3

Which of the following are examples of "accidental complexities" as described in the videos?

- [✓ Use of low-level application programming interfaces (APIs)]
- Deadlocks resulting from "circular waiting"
- [✓ Limitations with debugging environments and debugging tools]
- Race conditions in critical sections due to lack of synchronization mechanisms
- Ensuring that threads are given proper access to system resources

Question 4

Which of the following are examples of inherent complexities related to synchronization and scheduling presented in these videos?

- [✓ Scheduling the arrival and departure of airplanes based on limited resources, such as gates and runways]
- Casting void pointers to whatever structure is used to pass data between a caller and callee in the Pthreads environment
- Using the POSIX Pthreads API (defined using the C programming language) to program concurrent applications
- [✓ Ensuring applications running concurrently on an Android device don't corrupt raw contact entries in the SQLite Contacts database]

Question 5

Which of the following implementation elements are unique to each thread, according to the videos?

- [✓ A program counter]
- [✓ A run-time stack]
- Static data areas
- The run-time heap

Question 6

Which of the following are ways that a program can give a Java Thread some code to run, according to the videos?

- ☐ Extend the Thread class, override its run() hook method, and explicitly call run() from application code to start the Thread without having to call its start() method explicitly
- ☑ Extend the Thread class, override its run() hook method, and call start() on an instance of the extended Thread class
- ☑ Implement the Runnable interface, override its run() hook method, pass the Runnable object to the constructor of a new Thread object, and call start() on the Thread object

Question 7

Which of the following statements are true according to the videos?

- ☐ The only reliable and portable way to terminate a Java Thread is to call its stop() method
- ☑ Java the Thread interrupt() method behaves like traditional hardware & operating system interrupts, i.e., it automatically terminates a Thread regardless of what it is doing
- ☑ If user code in a Java Thread calls wait(), join(), or sleep() these methods check if they've been interrupted and throw the InterruptedException
- ☐ The use of a volatile boolean "stop" flag automatically wakeups blocking wait(), join(), and sleep() calls

Question 8

Which of the following statements about a Java Thread's lifecycle are correct, according to the videos?

- [✓] When a Thread's run() hook method returns the Thread transitions to the Runnable state
- [ ] When a Java program creates a Thread object it's initially in the Runnable state
- [ ] When the Android Linux scheduler selects a Thread to execute it transitions to the Running state
- [ ] When a Java program calls sleep() the Thread transitions to the Blocked state

Question 9

Which of the following are the consequences of the Java ArrayList class implementation not being synchronized in the BuggyQueue example, according to the videos?

- [✓] If multiple threads access an ArrayList instance concurrently then it may be corrupted due to race conditions
- [✓] The ArrayList class should not be used in concurrent Java programs under any circumstances
- [ ] The ArrayList class should only be used in concurrent Java programs running on a single-core computer

Question 10

Which of the following statements about the PingPongWrong program are correct, according to the material presented in the videos?

- [ ] Although this program doesn't work properly in the Java console environment, it will work correctly on Android due to Android's multi-threaded design restrictions
- [✓] Using Java Semaphores and CountDownLatches will make the program alternate printing "ping" and "pong" correctly, but will make the performance unacceptably slow
- [✓] After the run() methods of both PlayPingPongThread objects return, the calls to their join() methods in the main Thread will also return

✓ **In accordance with the Coursera Honor Code, I (Pablo Perotti) certify that the answers here are my own work.** **Thank you!**

Submit Answers Save Answers