

## Feedback — Week 1 Quiz

[Help](#)

You submitted this quiz on **Fri 16 May 2014 9:31 PM PDT**. You got a score of **39.00** out of **39.00**.

### Question 1

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

Your Answer	Score	Explanation
<input type="checkbox"/> Make the program easier to debug	✓ 1.00	
<input type="checkbox"/> Make the program behave more deterministically with respect to runtime execution order	✓ 1.00	
<input checked="" type="checkbox"/> Improve perceived responsiveness	✓ 1.00	
<input checked="" type="checkbox"/> Enhance performance on multi-core platforms	✓ 1.00	
<input checked="" type="checkbox"/> Simplify program structure relative to event-driven programming	✓ 1.00	
Total	5.00 / 5.00	

#### Question Explanation

See the Section 1 Module 1 Part 1 video

### Question 2

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

Your Answer	Score	Explanation
<input checked="" type="checkbox"/> The structure of its control flow is obscured in both time and space	✓ 1.00	
<input checked="" type="checkbox"/> It's hard to optimize its performance	✓ 1.00	
<input type="checkbox"/> It's not portable across operating systems	✓ 1.00	
<input type="checkbox"/> It's behavior is non-deterministic on multi-core hardware	✓ 1.00	
Total	4.00 / 4.00	

#### Question Explanation

See the Section 1 Module 1 Part 1 video

## Question 3

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

Your Answer	Score	Explanation
<input checked="" type="checkbox"/> Use of low-level application programming interfaces (APIs)	✓ 1.00	
<input type="checkbox"/> Ensuring that threads are given proper access to system resources	✓ 1.00	
<input checked="" type="checkbox"/> Limitations with debugging environments and debugging tools	✓ 1.00	
<input type="checkbox"/> Deadlocks resulting from "circular waiting"	✓ 1.00	
<input type="checkbox"/> Race conditions in critical sections due to lack of synchronization mechanisms	✓ 1.00	
Total	5.00 / 5.00	

#### Question Explanation

See the Section 1 Module 1 Part 2 video

## Question 4

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

Your Answer	Score	Explanation
<input type="checkbox"/> Using the POSIX Pthreads API (defined using the C programming language) to program concurrent applications	✓ 1.00	
<input checked="" type="checkbox"/> Scheduling the arrival and departure of airplanes based on limited resources, such as gates and runways	✓ 1.00	
<input checked="" type="checkbox"/> Ensuring applications running concurrently on an Android device don't corrupt raw contact entries in the SQLite Contacts database	✓ 1.00	
<input type="checkbox"/> Casting void pointers to whatever structure is used to pass data between a caller and callee in the Pthreads environment	✓ 1.00	
Total	4.00 / 4.00	

### Question Explanation

See the Section 1 Module 1 Part 2 video

## Question 5

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

Your Answer	Score	Explanation
<input checked="" type="checkbox"/> A program counter	✓ 1.00	
<input type="checkbox"/> Static data areas	✓ 1.00	
<input type="checkbox"/> The run-time heap	✓ 1.00	
<input checked="" type="checkbox"/> A run-time stack	✓ 1.00	

Total

4.00 / 4.00

### Question Explanation

See the Section 1 Module 2 Part 1 video

## Question 6

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

Your Answer	Score	Explanation
<input checked="" type="checkbox"/> 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	✓ 1.00	
<input type="checkbox"/> 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	✓ 1.00	
<input checked="" type="checkbox"/> Extend the Thread class, override its run() hook method, and call start() on an instance of the extended Thread class	✓ 1.00	
Total	3.00 / 3.00	

### Question Explanation

See the Section 1 Module 2 Part 1 video

## Question 7

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

Your Answer	Score	Explanation
<input type="checkbox"/> The only reliable and portable way to terminate a Java Thread is to call its stop() method	✓ 1.00	

☐ The use of a volatile boolean "stop" flag automatically wakeups blocking wait(), join(), and sleep() calls ✓ 1.00

☒ If user code in a Java Thread calls wait(), join(), or sleep() these methods check if they've been interrupted and throw the InterruptedException ✓ 1.00

☐ 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 ✓ 1.00

Total 4.00 / 4.00

#### Question Explanation

See the Section 1 Module 2 Part 2 video

## Question 8

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

Your Answer	Score	Explanation
-------------	-------	-------------

<input checked="" type="checkbox"/> When the Android Linux scheduler selects a Thread to execute it transitions to the Running state	<span>✓</span> 1.00	
--	---------------------	--

<input type="checkbox"/> When a Thread's run() hook method returns the Thread transitions to the Runnable state	<span>✓</span> 1.00	
---	---------------------	--

<input type="checkbox"/> When a Java program calls sleep() the Thread transitions to the Blocked state	<span>✓</span> 1.00	
--	---------------------	--

<input type="checkbox"/> When a Java program creates a Thread object it's initially in the Runnable state	<span>✓</span> 1.00	
---	---------------------	--

Total 4.00 / 4.00

#### Question Explanation

See the Section 1 Module 2 Part 2 video

## 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?

Your Answer	Score	Explanation
<input type="checkbox"/> The ArrayList class should not be used in concurrent Java programs under any circumstances	✓ 1.00	
<input type="checkbox"/> The ArrayList class should only be used in concurrent Java programs running on a single-core computer	✓ 1.00	
<input checked="" type="checkbox"/> If multiple threads access an ArrayList instance concurrently then it may be corrupted due to race conditions	✓ 1.00	
Total	3.00 / 3.00	

### Question Explanation

See the Section 1 Module 2 Part 3 video

## Question 10

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

Your Answer	Score	Explanation
<input checked="" type="checkbox"/> After the run() methods of both PlayPingPongThread objects return, the calls to their join() methods in the main Thread will also return	✓ 1.00	
<input type="checkbox"/> 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	✓ 1.00	
<input type="checkbox"/> Using Java Semaphores and CountdownLatches will make	✓ 1.00	

the program alternate printing "ping" and "pong" correctly, but will make the performance unacceptably slow

Total

3.00 /

3.00

#### Question Explanation

See the Section 1 Module 2 Part 3 video