

CLAIMS

1. A method performed at least in part on at least one machine, comprising:

 multiplexing an async task that is identified in at least two task chains, including sharing results of the async task between a listener set comprising at least two dependent lower-level listeners, maintaining identity information of each dependent lower-level listener in association with the async task, preventing the cancellation of one listener from cancelling the async task as long as at least one listener remains dependent on the async task, based upon the identity information, wherein the listener comprises a listener task including listener task code that performs the listener task, and wrapping the listener task code in cancel-checking code that checks, before the listener task code executes, whether the listener task is intended to be cancelled, and if intended to be cancelled, not allowing at least some of the listener task code to execute.

2. The method of claim 1 wherein preventing the cancellation of one listener from cancelling the async task comprises receiving a cancel request corresponding to a cancelled lower-level listener, accessing the identity information, modifying the identity information including removing the cancelled lower-level listener from the set of listeners dependent on the async task, and determining from the identity information that at least one non-cancelled listener remains dependent on the async task.

3. The method of claim 1 further comprising, notifying each listener of the listener set upon completion the async task.

4. The method of claim 3 further comprising, queuing each listener of the listener set to run as another async task.

5. The method of claim 1 further comprising, optimizing the at least two task chains to make the async task a shared task.

6. The method of claim 1 further comprising, associating a task chain with a name.

7. The method of claim 6 further comprising, cancelling a pending task chain based upon the name associated with the pending task chain.

8. The method of claim 6 further comprising, cancelling a pending task chain based upon the name associated with the pending task chain matching a name of a new task chain to be run instead of the pending task chain.

9. The method of claim 1 further comprising, associating a task chain with a name and a current cancel version number that is based upon a cancel call for that task chain, maintaining the current cancel version number as a saved cancel version number in association with the name, and while running a task chain with a matching name, evaluating the saved cancel version number with a current cancel version number and cancelling the task chain if the saved cancel version number does not match the current cancel version number.

10. A method performed at least in part on at least one machine, comprising:

 multiplexing an async task that is identified in at least two task chains, including sharing results of the async task between a listener set comprising at least two dependent lower-level listeners, maintaining identity information of each dependent lower-level listener in association with the async task, preventing the cancellation of one listener from cancelling the async task as long as at least one listener remains dependent on the async task, based upon the identity information, associating a task chain with a name and a current cancel version number that is based upon a cancel call for that task chain, maintaining the current cancel version number as a saved cancel version number in association with the name, and while running a task chain with a matching name, evaluating the saved cancel version number with a current cancel version number and cancelling the task chain if the saved cancel version number does not match the current cancel version number.

11. The method of claim 10 wherein preventing the cancellation of one listener from cancelling the async task comprises receiving a cancel request corresponding to a cancelled lower-level listener, accessing the identity information, modifying the identity information including removing the cancelled lower-level listener from the set of listeners dependent on the async task, and determining from the identity information that at least one non-cancelled listener remains dependent on the async task.

12. The method of claim 10 further comprising, notifying each listener of the listener set upon completion the async task.

13. The method of claim 12 further comprising, queuing each listener of the listener set to run as another async task.

14. The method of claim 10 further comprising, optimizing the at least two task chains to make the async task a shared task.

15. The method of claim 10 further comprising, wrapping async task code that performs the async task in cancel-checking code.

16. The method of claim 15 further comprising, running the async task, including executing the cancel-checking code to determine whether the async task is intended to be cancelled, and if intended to be cancelled, not executing at least some of the async task code.

17. The method of claim 10 wherein the listener comprises a listener task including listener task code that per-

forms the listener task, and further comprising wrapping the listener task code in cancel-checking code that checks, before the listener task code executes, whether the listener task is intended to be cancelled, and if intended to be cancelled, not allowing at least some of the listener task code to execute.

18. The method of claim 10 further comprising, associating a task chain with a name.

19. The method of claim 18 further comprising, cancelling a pending task chain based upon the name associated with the pending task chain.

20. The method of claim 18 further comprising, cancelling a pending task chain based upon the name associated with the pending task chain matching a name of a new task chain to be run instead of the pending task chain.