Using the Java 1.5 ScheduledExecutorService for scheduling repeating tasks

Published April 10th, 2012

How to use the java.util.concurrent Executor classes as a replacement for Timer and TimerTask

The Java 1.4 Timer and TimerTask classes relied on Object.wait(long) for their timing. Java 1.5 introduced the new Executor classes for more advanced scheduling of delayed or repeating tasks and the Future object to represent the scheduled task and allow it to be cancelled.

```
import java.util.concurrent.Executors;
import java.util.concurrent.ScheduledExecutorService;
import java.util.concurrent.ScheduledFuture;
import java.util.concurrent.TimeUnit;

public class SchedulerTaskExample
{
  private ScheduledExecutorService scheduler;
  private ScheduledFuture<?> future;

  public SchedulerTaskExample()
  {
    // create a scheduler with one thread to process the scheduled tasks
    scheduler = Executors.newScheduledThreadPool(1);

    Runnable task = new Runnable()
    {
      @Override
      public void run()
      {
        System.out.println("Scheduled task invoked!");
      }
    };

    // invoke task every 30 seconds after a 10 second initial delay
    future = scheduler.scheduleWithFixedDelay(task, 10, 30, TimeUnit.SECONDS);
  }

  public void shutdown()
  {
    if (future != null)
    {
      // true = allow running tasks to complete, false = interrupt them
      future.cancel(true);
```

```
    }
  }
}
```