package com.twitter.search.earlybird.factory;

import java.util.concurrent.Callable;

import java.util.concurrent.ScheduledExecutorService;

import java.util.concurrent.ScheduledFuture;

import java.util.concurrent.TimeUnit;

import com.google.common.annotations.VisibleForTesting;

import com.twitter.common.util.concurrent.ForwardingExecutorService;

/\*\*

\* This delegate type is intended for QueryCacheUpdater because it uses multiple threads to

\* create query cache during startup and then switch later to use single thread to update the

\* cache.

\*/

public abstract class QueryCacheUpdaterScheduledExecutorService<T extends ScheduledExecutorService>

extends ForwardingExecutorService<T> implements ScheduledExecutorService {

public QueryCacheUpdaterScheduledExecutorService(T executor) {

super(executor);

}

/\*\*

\* Sets the number of worker threads in this executor service to an appropriate value after the

\* earlybird startup has finished. While earlybird is starting up, we might want this executor

\* service to have more threads, in order to parallelize more some start up tasks. But once

\* earlybird is up, it might make sense to lower the number of worker threads.

\*/

public abstract void setWorkerPoolSizeAfterStartup();

@Override

public ScheduledFuture<?> schedule(Runnable command, long delay, TimeUnit unit) {

return delegate.schedule(command, delay, unit);

}

@Override

public ScheduledFuture<?> scheduleAtFixedRate(

Runnable command, long initialDelay, long period, TimeUnit unit) {

return delegate.scheduleAtFixedRate(command, initialDelay, period, unit);

}

@Override

public ScheduledFuture<?> scheduleWithFixedDelay(

Runnable command, long initialDelay, long delay, TimeUnit unit) {

return delegate.scheduleWithFixedDelay(command, initialDelay, delay, unit);

}

@Override

public <V> ScheduledFuture<V> schedule(Callable<V> callable, long delay, TimeUnit unit) {

return delegate.schedule(callable, delay, unit);

}

@VisibleForTesting

public T getDelegate() {

return delegate;

}

}