package com.twitter.search.common.search;

import java.io.IOException;

import java.util.List;

import javax.annotation.Nullable;

import org.apache.lucene.index.LeafReaderContext;

import org.apache.lucene.search.Collector;

import org.apache.lucene.search.LeafCollector;

import org.apache.lucene.search.Scorable;

import org.apache.lucene.search.ScoreMode;

import com.twitter.common.util.Clock;

import com.twitter.search.common.query.thriftjava.CollectorParams;

/\*\*

\* A {@link com.twitter.search.common.search.TwitterEarlyTerminationCollector}

\* that delegates actual hit collection to a sub collector.

\*/

public final class DelegatingEarlyTerminationCollector

extends TwitterEarlyTerminationCollector {

private final Collector subCollector;

private LeafCollector subLeafCollector;

/\*\* Creates a new DelegatingEarlyTerminationCollector instance. \*/

public DelegatingEarlyTerminationCollector(Collector subCollector,

CollectorParams collectorParams,

TerminationTracker terminationTracker,

@Nullable QueryCostProvider queryCostProvider,

int numDocsBetweenTimeoutChecks,

Clock clock) {

super(

collectorParams,

terminationTracker,

queryCostProvider,

numDocsBetweenTimeoutChecks,

clock);

this.subCollector = subCollector;

}

@Override

public void setScorer(Scorable scorer) throws IOException {

super.setScorer(scorer);

subLeafCollector.setScorer(scorer);

}

@Override

protected void doCollect() throws IOException {

subLeafCollector.collect(curDocId);

}

@Override

protected void doFinishSegment(int lastSearchedDocID) throws IOException {

if (subCollector instanceof TwitterCollector) {

((TwitterCollector) subCollector).finishSegment(lastSearchedDocID);

}

}

@Override

public void setNextReader(LeafReaderContext context) throws IOException {

super.setNextReader(context);

subLeafCollector = subCollector.getLeafCollector(context);

}

@Override

public ScoreMode scoreMode() {

return subCollector.scoreMode();

}

@Override

public List<String> getDebugInfo() {

return null;

}

}