package com.twitter.search.common.search.termination;

import java.io.IOException;

import java.util.Arrays;

import com.google.common.base.Preconditions;

import org.apache.lucene.index.IndexReader;

import org.apache.lucene.search.IndexSearcher;

import org.apache.lucene.search.Query;

import org.apache.lucene.search.ScoreMode;

import org.apache.lucene.search.Weight;

/\*\*

\* Query implementation that can timeout and return non-exhaustive results.

\*/

public class TerminationQuery extends Query {

private final Query inner;

private final QueryTimeout timeout;

public TerminationQuery(Query inner, QueryTimeout timeout) {

this.inner = Preconditions.checkNotNull(inner);

this.timeout = Preconditions.checkNotNull(timeout);

}

@Override

public Weight createWeight(

IndexSearcher searcher, ScoreMode scoreMode, float boost) throws IOException {

Weight innerWeight = inner.createWeight(searcher, scoreMode, boost);

return new TerminationQueryWeight(this, innerWeight, timeout);

}

@Override

public Query rewrite(IndexReader reader) throws IOException {

Query rewritten = inner.rewrite(reader);

if (rewritten != inner) {

return new TerminationQuery(rewritten, timeout);

}

return this;

}

public QueryTimeout getTimeout() {

return timeout;

}

@Override

public int hashCode() {

return Arrays.hashCode(new Object[] {inner, timeout});

}

@Override

public boolean equals(Object obj) {

if (!(obj instanceof TerminationQuery)) {

return false;

}

TerminationQuery terminationQuery = TerminationQuery.class.cast(obj);

return Arrays.equals(new Object[] {inner, timeout},

new Object[] {terminationQuery.inner, terminationQuery.timeout});

}

@Override

public String toString(String field) {

return inner.toString(field);

}

}