package com.twitter.search.common.query;

import java.io.IOException;

import java.util.Set;

import org.apache.lucene.index.LeafReaderContext;

import org.apache.lucene.index.Term;

import org.apache.lucene.search.ConstantScoreScorer;

import org.apache.lucene.search.DocIdSetIterator;

import org.apache.lucene.search.Explanation;

import org.apache.lucene.search.Query;

import org.apache.lucene.search.Scorer;

import org.apache.lucene.search.ScoreMode;

import org.apache.lucene.search.Weight;

/\*\*

\* An abstract Weight implementation that can be used by all "filter" classes (Query instances that

\* should not contribute to the overall query score).

\*/

public abstract class DefaultFilterWeight extends Weight {

public DefaultFilterWeight(Query query) {

super(query);

}

@Override

public void extractTerms(Set<Term> terms) {

}

@Override

public Explanation explain(LeafReaderContext context, int doc) throws IOException {

Scorer scorer = scorer(context);

if ((scorer != null) && (scorer.iterator().advance(doc) == doc)) {

return Explanation.match(0f, "Match on id " + doc);

}

return Explanation.match(0f, "No match on id " + doc);

}

@Override

public Scorer scorer(LeafReaderContext context) throws IOException {

DocIdSetIterator disi = getDocIdSetIterator(context);

if (disi == null) {

return null;

}

return new ConstantScoreScorer(this, 0.0f, ScoreMode.COMPLETE\_NO\_SCORES, disi);

}

@Override

public boolean isCacheable(LeafReaderContext ctx) {

return false;

}

/\*\*

\* Returns the DocIdSetIterator over which the scorers created by this weight need to iterate.

\*

\* @param context The LeafReaderContext instance used to create the scorer.

\*/

protected abstract DocIdSetIterator getDocIdSetIterator(LeafReaderContext context)

throws IOException;

}