package com.twitter.search.earlybird\_root.common;

import java.util.concurrent.TimeUnit;

import javax.annotation.Nullable;

import com.google.common.annotations.VisibleForTesting;

import com.google.common.base.Preconditions;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import com.twitter.search.common.metrics.SearchCounter;

import com.twitter.search.common.metrics.SearchTimerStats;

import com.twitter.search.earlybird.thrift.EarlybirdRequest;

import com.twitter.search.earlybird.thrift.EarlybirdResponse;

import com.twitter.search.earlybird.thrift.EarlybirdResponseCode;

import com.twitter.search.queryparser.parser.SerializedQueryParser;

import com.twitter.search.queryparser.query.Query;

import com.twitter.search.queryparser.query.QueryParserException;

import com.twitter.util.Future;

/\*\*

\* Common utils for parsing serialized queries, and handling query parser exceptions.

\*/

public final class QueryParsingUtils {

private static final Logger LOG = LoggerFactory.getLogger(QueryParsingUtils.class);

@VisibleForTesting

public static final SearchCounter QUERYPARSE\_COUNT =

SearchCounter.export("root\_queryparse\_count");

private static final SearchTimerStats QUERYPARSE\_TIMER =

SearchTimerStats.export("root\_queryparse\_time", TimeUnit.NANOSECONDS, false, true);

private static final SearchCounter NO\_PARSED\_QUERY\_COUNT =

SearchCounter.export("root\_no\_parsed\_query\_count");

private QueryParsingUtils() { }

/\*\*

\* Takes an earlybird request, and parses its serialized query (if it is set).

\* Expects the required ThriftSearchQuery to be set on the passed in EarlybirdRequest.

\*

\* @param request the earlybird request to parse.

\* @return null if the request does not specify a serialized query.

\* @throws QueryParserException if querry parsing fails.

\*/

@Nullable

static Query getParsedQuery(EarlybirdRequest request) throws QueryParserException {

// searchQuery is required on EarlybirdRequest.

Preconditions.checkState(request.isSetSearchQuery());

Query parsedQuery;

if (request.getSearchQuery().isSetSerializedQuery()) {

long startTime = System.nanoTime();

try {

String serializedQuery = request.getSearchQuery().getSerializedQuery();

parsedQuery = new SerializedQueryParser().parse(serializedQuery);

} finally {

QUERYPARSE\_COUNT.increment();

QUERYPARSE\_TIMER.timerIncrement(System.nanoTime() - startTime);

}

} else {

NO\_PARSED\_QUERY\_COUNT.increment();

parsedQuery = null;

}

return parsedQuery;

}

/\*\*

\* Creates a new EarlybirdResponse with a CLIENT\_ERROR response code, to be used as a response

\* to a request where we failed to parse a user passed in serialized query.

\*/

public static Future<EarlybirdResponse> newClientErrorResponse(

EarlybirdRequest request,

QueryParserException e) {

String msg = "Failed to parse query";

LOG.warn(msg, e);

EarlybirdResponse errorResponse =

new EarlybirdResponse(EarlybirdResponseCode.CLIENT\_ERROR, 0);

errorResponse.setDebugString(msg + ": " + e.getMessage());

return Future.value(errorResponse);

}

}