package com.twitter.search.earlybird\_root;

import java.util.concurrent.TimeUnit;

import com.google.common.base.Preconditions;

import com.twitter.search.common.decider.SearchDecider;

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

import com.twitter.search.common.root.LoggingSupport;

import com.twitter.search.earlybird.common.EarlybirdRequestPostLogger;

import com.twitter.search.earlybird.common.EarlybirdRequestPreLogger;

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

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

public class EarlybirdServiceLoggingSupport extends

LoggingSupport.DefaultLoggingSupport<EarlybirdRequest, EarlybirdResponse> {

private static final int LATENCY\_WARN\_THRESHOLD\_MS = 100;

private static final Timer DUMMY\_TIMER;

private final EarlybirdRequestPreLogger requestPreLogger;

private final EarlybirdRequestPostLogger requestPostLogger;

static {

DUMMY\_TIMER = new Timer(TimeUnit.MILLISECONDS);

DUMMY\_TIMER.stop();

}

public EarlybirdServiceLoggingSupport(SearchDecider decider) {

requestPreLogger = EarlybirdRequestPreLogger.buildForRoot(decider.getDecider());

requestPostLogger = EarlybirdRequestPostLogger.buildForRoot(LATENCY\_WARN\_THRESHOLD\_MS,

decider.getDecider());

}

@Override

public void prelogRequest(EarlybirdRequest req) {

requestPreLogger.logRequest(req);

}

@Override

public void postLogRequest(

EarlybirdRequest request,

EarlybirdResponse response,

long latencyNanos) {

Preconditions.checkNotNull(request);

Preconditions.checkNotNull(response);

response.setResponseTimeMicros(TimeUnit.NANOSECONDS.toMicros(latencyNanos));

response.setResponseTime(TimeUnit.NANOSECONDS.toMillis(latencyNanos));

requestPostLogger.logRequest(request, response, DUMMY\_TIMER);

}

@Override

public void logExceptions(EarlybirdRequest req, Throwable t) {

ExceptionHandler.logException(req, t);

}

}