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

import java.util.concurrent.atomic.AtomicReference;

import scala.Option;

import com.google.common.base.Preconditions;

import com.twitter.finagle.Service;

import com.twitter.finagle.SimpleFilter;

import com.twitter.finagle.context.Contexts;

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

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

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

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

import com.twitter.util.Future;

/\*\*

\* A filter that sets the clientId in the local context, to be usd later by SearchPayloadSizeFilter.

\*/

public class SearchPayloadSizeLocalContextFilter

extends SimpleFilter<EarlybirdRequest, EarlybirdResponse> {

private static final SearchCounter CLIENT\_ID\_CONTEXT\_KEY\_NOT\_SET\_COUNTER = SearchCounter.export(

"search\_payload\_size\_local\_context\_filter\_client\_id\_context\_key\_not\_set");

@Override

public Future<EarlybirdResponse> apply(EarlybirdRequest request,

Service<EarlybirdRequest, EarlybirdResponse> service) {

// In production, the SearchPayloadSizeFilter.CLIENT\_ID\_CONTEXT\_KEY should always be set

// (by ThriftServer). However, it's not set in tests, because tests do not start a ThriftServer.

Option<AtomicReference<String>> clientIdOption =

Contexts.local().get(SearchPayloadSizeFilter.CLIENT\_ID\_CONTEXT\_KEY);

if (clientIdOption.isDefined()) {

AtomicReference<String> clientIdReference = clientIdOption.get();

Preconditions.checkArgument(clientIdReference.get() == null);

clientIdReference.set(request.getClientId());

} else {

CLIENT\_ID\_CONTEXT\_KEY\_NOT\_SET\_COUNTER.increment();

}

return service.apply(request);

}

}