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

import com.google.common.base.Preconditions;

import com.google.common.cache.CacheBuilder;

import com.google.common.cache.CacheLoader;

import com.google.common.cache.LoadingCache;

import com.google.common.collect.ImmutableMap;

import com.twitter.common.util.Clock;

import com.twitter.finagle.Service;

import com.twitter.finagle.SimpleFilter;

import com.twitter.search.common.clientstats.RequestCounters;

import com.twitter.search.common.clientstats.RequestCountersEventListener;

import com.twitter.search.common.util.FinagleUtil;

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

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

import com.twitter.search.earlybird\_root.common.EarlybirdRequestContext;

import com.twitter.search.earlybird\_root.common.EarlybirdRequestType;

import com.twitter.util.Future;

public class RequestTypeCountFilter

extends SimpleFilter<EarlybirdRequestContext, EarlybirdResponse> {

private final ImmutableMap<EarlybirdRequestType, RequestCounters> typeCounters;

private final RequestCounters allRequestTypesCounter;

private final ImmutableMap<EarlybirdRequestType, LoadingCache<String, RequestCounters>>

perTypePerClientCounters;

/\*\*

\* Constructs the filter.

\*/

public RequestTypeCountFilter(final String statSuffix) {

ImmutableMap.Builder<EarlybirdRequestType, RequestCounters> perTypeBuilder =

ImmutableMap.builder();

for (EarlybirdRequestType type : EarlybirdRequestType.values()) {

perTypeBuilder.put(type, new RequestCounters(

"request\_type\_count\_filter\_" + type.getNormalizedName() + "\_" + statSuffix));

}

typeCounters = perTypeBuilder.build();

allRequestTypesCounter =

new RequestCounters("request\_type\_count\_filter\_all\_" + statSuffix, true);

ImmutableMap.Builder<EarlybirdRequestType, LoadingCache<String, RequestCounters>>

perTypePerClientBuilder = ImmutableMap.builder();

// No point in setting any kind of expiration policy for the cache, since the stats will

// continue to be exported, so the objects will not be GCed anyway.

CacheBuilder<Object, Object> cacheBuilder = CacheBuilder.newBuilder();

for (final EarlybirdRequestType requestType : EarlybirdRequestType.values()) {

CacheLoader<String, RequestCounters> cacheLoader =

new CacheLoader<String, RequestCounters>() {

@Override

public RequestCounters load(String clientId) {

return new RequestCounters("request\_type\_count\_filter\_for\_" + clientId + "\_"

+ requestType.getNormalizedName() + "\_" + statSuffix);

}

};

perTypePerClientBuilder.put(requestType, cacheBuilder.build(cacheLoader));

}

perTypePerClientCounters = perTypePerClientBuilder.build();

}

@Override

public Future<EarlybirdResponse> apply(

EarlybirdRequestContext requestContext,

Service<EarlybirdRequestContext, EarlybirdResponse> service) {

EarlybirdRequestType requestType = requestContext.getEarlybirdRequestType();

RequestCounters requestCounters = typeCounters.get(requestType);

Preconditions.checkNotNull(requestCounters);

// Update the per-type and "all" counters.

RequestCountersEventListener<EarlybirdResponse> requestCountersEventListener =

new RequestCountersEventListener<>(

requestCounters, Clock.SYSTEM\_CLOCK, EarlybirdSuccessfulResponseHandler.INSTANCE);

RequestCountersEventListener<EarlybirdResponse> allRequestTypesEventListener =

new RequestCountersEventListener<>(

allRequestTypesCounter, Clock.SYSTEM\_CLOCK,

EarlybirdSuccessfulResponseHandler.INSTANCE);

RequestCountersEventListener<EarlybirdResponse> perTypePerClientEventListener =

updatePerTypePerClientCountersListener(requestContext);

return service.apply(requestContext)

.addEventListener(requestCountersEventListener)

.addEventListener(allRequestTypesEventListener)

.addEventListener(perTypePerClientEventListener);

}

private RequestCountersEventListener<EarlybirdResponse> updatePerTypePerClientCountersListener(

EarlybirdRequestContext earlybirdRequestContext) {

EarlybirdRequestType requestType = earlybirdRequestContext.getEarlybirdRequestType();

LoadingCache<String, RequestCounters> perClientCounters =

perTypePerClientCounters.get(requestType);

Preconditions.checkNotNull(perClientCounters);

String clientId = ClientIdUtil.formatFinagleClientIdAndClientId(

FinagleUtil.getFinagleClientName(),

ClientIdUtil.getClientIdFromRequest(earlybirdRequestContext.getRequest()));

RequestCounters clientCounters = perClientCounters.getUnchecked(clientId);

Preconditions.checkNotNull(clientCounters);

return new RequestCountersEventListener<>(

clientCounters, Clock.SYSTEM\_CLOCK, EarlybirdSuccessfulResponseHandler.INSTANCE);

}

}