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

import java.util.ArrayList;

import java.util.List;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import com.twitter.search.common.futures.Futures;

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

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

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

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

import com.twitter.util.Future;

import com.twitter.util.Try;

/\*\*

\* Responsible for handling requests in superroot.

\*/

public abstract class RequestRouter {

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

/\*\*

\* Saved request and response, to be included in debug info.

\*/

class RequestResponse {

// Where is this request sent to. Freeform text like "realtime", "archive", etc.

private String sentTo;

private EarlybirdRequestContext requestContext;

private Future<EarlybirdResponse> earlybirdResponseFuture;

RequestResponse(String sentTo,

EarlybirdRequestContext requestContext,

Future<EarlybirdResponse> earlybirdResponseFuture) {

this.sentTo = sentTo;

this.requestContext = requestContext;

this.earlybirdResponseFuture = earlybirdResponseFuture;

}

String getSentTo() {

return sentTo;

}

public EarlybirdRequestContext getRequestContext() {

return requestContext;

}

Future<EarlybirdResponse> getEarlybirdResponseFuture() {

return earlybirdResponseFuture;

}

}

/\*\*

\* Forward a request to different clusters and merge the responses back into one response.

\* @param requestContext

\*/

public abstract Future<EarlybirdResponse> route(EarlybirdRequestContext requestContext);

/\*\*

\* Save a request (and its response future) to be included in debug info.

\*/

void saveRequestResponse(

List<RequestResponse> requestResponses,

String sentTo,

EarlybirdRequestContext earlybirdRequestContext,

Future<EarlybirdResponse> earlybirdResponseFuture

) {

requestResponses.add(

new RequestResponse(

sentTo,

earlybirdRequestContext,

earlybirdResponseFuture

)

);

}

Future<EarlybirdResponse> maybeAttachSentRequestsToDebugInfo(

List<RequestResponse> requestResponses,

EarlybirdRequestContext requestContext,

Future<EarlybirdResponse> response

) {

if (requestContext.getRequest().getDebugMode() >= 4) {

return this.attachSentRequestsToDebugInfo(

response,

requestResponses

);

} else {

return response;

}

}

/\*\*

\* Attaches saved client requests and their responses to the debug info within the

\* main EarlybirdResponse.

\*/

Future<EarlybirdResponse> attachSentRequestsToDebugInfo(

Future<EarlybirdResponse> currentResponse,

List<RequestResponse> requestResponses) {

// Get all the response futures that we're waiting on.

List<Future<EarlybirdResponse>> allResponseFutures = new ArrayList<>();

for (RequestResponse rr : requestResponses) {

allResponseFutures.add(rr.getEarlybirdResponseFuture());

}

// Pack all the futures into a single future.

Future<List<Try<EarlybirdResponse>>> allResponsesFuture =

Futures.collectAll(allResponseFutures);

return currentResponse.flatMap(mainResponse -> {

if (!mainResponse.isSetDebugInfo()) {

mainResponse.setDebugInfo(new EarlybirdDebugInfo());

}

Future<EarlybirdResponse> responseWithRequests = allResponsesFuture.map(allResponses -> {

// Get all individual response "Trys" and see if we can extract something from them

// that we can attach to the debugInfo.

for (int i = 0; i < allResponses.size(); i++) {

Try<EarlybirdResponse> responseTry = allResponses.get(i);

if (responseTry.isReturn()) {

EarlybirdResponse attachedResponse = responseTry.get();

// Don't include the debug string, it's already a part of the main response's

// debug string.

attachedResponse.unsetDebugString();

EarlybirdRequestResponse reqResp = new EarlybirdRequestResponse();

reqResp.setSentTo(requestResponses.get(i).getSentTo());

reqResp.setRequest(requestResponses.get(i).getRequestContext().getRequest());

reqResp.setResponse(attachedResponse.toString());

mainResponse.debugInfo.addToSentRequests(reqResp);

}

}

return mainResponse;

});

return responseWithRequests;

});

}

}