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

import javax.inject.Inject;

import javax.inject.Named;

import com.twitter.search.common.caching.Cache;

import com.twitter.search.common.caching.filter.CacheFilter;

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

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

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

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

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

/\*\*

\* A filter that:

\* - Strips the request of all personalization fields, normalizes it and looks it up in the cache.

\* If it finds a response with 0 results in the cache, it returns it.

\* - Caches the response for a personalized query, whenever the response has 0 results. The cache

\* key is the normalized request with all personalization fields stripped.

\*

\* If a query (from a logged in or logged out user) returns 0 results, then the same query will

\* always return 0 results, for all users. So we can cache that result.

\*/

public class RelevanceZeroResultsCacheFilter

extends CacheFilter<EarlybirdRequestContext, EarlybirdRequest, EarlybirdResponse> {

/\*\* Creates a filter that caches relevance requests with 0 results. \*/

@Inject

public RelevanceZeroResultsCacheFilter(

@RelevanceCache Cache<EarlybirdRequest, EarlybirdResponse> cache,

SearchDecider decider,

@Named(SearchRootModule.NAMED\_NORMALIZED\_SEARCH\_ROOT\_NAME) String normalizedSearchRootName) {

super(cache,

new RelevanceZeroResultsQueryCachePredicate(decider, normalizedSearchRootName),

new RelevanceZeroResultsCacheRequestNormalizer(),

new RelevanceZeroResultsCachePostProcessor(),

new RelevanceZeroResultsServicePostProcessor(cache),

new EarlybirdRequestPerClientCacheStats("relevance\_zero\_results"));

}

}