package com.twitter.search.earlybird.querycache;

import java.io.File;

import java.io.FileNotFoundException;

import java.io.FileReader;

import java.io.Reader;

import java.util.ArrayList;

import java.util.List;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.yaml.snakeyaml.TypeDescription;

import org.yaml.snakeyaml.Yaml;

import org.yaml.snakeyaml.constructor.Constructor;

import com.twitter.search.common.config.Config;

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

import com.twitter.search.earlybird.common.config.EarlybirdConfig;

// QueryCacheConfig is not thread safe. \*Do not\* attempt to create multiple QueryCacheConfig

// in different threads

public class QueryCacheConfig {

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

private static final String DEFAULT\_CONFIG\_FILE = "querycache.yml";

private final SearchStatsReceiver statsReceiver;

private List<QueryCacheFilter> filters;

public QueryCacheConfig(SearchStatsReceiver statsReceiver) {

this(locateConfigFile(EarlybirdConfig.getString("query\_cache\_config\_file\_name",

DEFAULT\_CONFIG\_FILE)), statsReceiver);

}

// package protected constructor for unit test only

QueryCacheConfig(Reader reader, SearchStatsReceiver statsReceiver) {

this.statsReceiver = statsReceiver;

if (reader == null) {

throw new RuntimeException("Query cache config not loaded");

}

loadConfig(reader);

}

public List<QueryCacheFilter> filters() {

return filters;

}

int getFilterSize() {

return filters.size();

}

private static FileReader locateConfigFile(String configFileName) {

File configFile = null;

String dir = Config.locateSearchConfigDir(EarlybirdConfig.EARLYBIRD\_CONFIG\_DIR, configFileName);

if (dir != null) {

configFile = openConfigFile(dir + "/" + configFileName);

}

if (configFile != null) {

try {

return new FileReader(configFile);

} catch (FileNotFoundException e) {

// This should not happen as the caller should make sure that the file exists before

// calling this function.

LOG.error("Unexpected exception", e);

throw new RuntimeException("Query cache config file not loaded!", e);

}

}

return null;

}

private static File openConfigFile(String configFilePath) {

File configFile = new File(configFilePath);

if (!configFile.exists()) {

LOG.warn("QueryCache config file [" + configFile + "] not found");

configFile = null;

} else {

LOG.info("Opened QueryCacheFilter config file [" + configFile + "]");

}

return configFile;

}

private void loadConfig(Reader reader) {

TypeDescription qcEntryDescription = new TypeDescription(QueryCacheFilter.class);

Constructor constructor = new Constructor(qcEntryDescription);

Yaml yaml = new Yaml(constructor);

filters = new ArrayList<>();

for (Object data : yaml.loadAll(reader)) {

QueryCacheFilter cacheFilter = (QueryCacheFilter) data;

try {

cacheFilter.sanityCheck();

} catch (QueryCacheFilter.InvalidEntryException e) {

throw new RuntimeException(e);

}

cacheFilter.createQueryCounter(statsReceiver);

filters.add(cacheFilter);

LOG.info("Loaded filter from config {}", cacheFilter.toString());

}

LOG.info("Total filters loaded: {}", filters.size());

}

}