package com.twitter.search.ingester.pipeline.twitter.filters;

import java.util.EnumSet;

import java.util.Set;

import com.twitter.decider.Decider;

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

import com.twitter.search.common.relevance.entities.TwitterMessage;

import com.twitter.search.common.relevance.entities.TwitterMessageUtil;

public class IngesterValidMessageFilter {

public static final String KEEP\_NULLCAST\_DECIDER\_KEY =

"ingester\_all\_keep\_nullcasts";

public static final String STRIP\_SUPPLEMENTARY\_EMOJIS\_DECIDER\_KEY\_PREFIX =

"valid\_message\_filter\_strip\_supplementary\_emojis\_";

protected final Decider decider;

public IngesterValidMessageFilter(Decider decider) {

this.decider = decider;

}

/\*\*

\* Evaluate a message to see if it matches the filter or not.

\*

\* @param message to evaluate

\* @return true if this message should be emitted.

\*/

public boolean accepts(TwitterMessage message) {

return TwitterMessageUtil.validateTwitterMessage(

message, getStripEmojisFields(), acceptNullcast());

}

private Set<TwitterMessageUtil.Field> getStripEmojisFields() {

Set<TwitterMessageUtil.Field> stripEmojisFields =

EnumSet.noneOf(TwitterMessageUtil.Field.class);

for (TwitterMessageUtil.Field field : TwitterMessageUtil.Field.values()) {

if (DeciderUtil.isAvailableForRandomRecipient(

decider,

STRIP\_SUPPLEMENTARY\_EMOJIS\_DECIDER\_KEY\_PREFIX + field.getNameForStats())) {

stripEmojisFields.add(field);

}

}

return stripEmojisFields;

}

protected final boolean acceptNullcast() {

return DeciderUtil.isAvailableForRandomRecipient(decider, KEEP\_NULLCAST\_DECIDER\_KEY);

}

}