package com.twitter.tweetypie

package hydrator

import com.twitter.tweetypie.additionalfields.AdditionalFields

import com.twitter.tweetypie.core.\_

import com.twitter.tweetypie.repository.TweetQuery

import com.twitter.tweetypie.thriftscala.\_

/\*\*

\* A hydrator that scrubs tweet fields that weren't requested. Those fields might be

\* present because they were previously requested and were cached with the tweet.

\*/

trait UnrequestedFieldScrubber {

def scrub(tweetResult: TweetResult): TweetResult

def scrub(tweetData: TweetData): TweetData

def scrub(tweet: Tweet): Tweet

}

object UnrequestedFieldScrubber {

def apply(options: TweetQuery.Options): UnrequestedFieldScrubber =

if (!options.scrubUnrequestedFields) NullScrubber

else new ScrubberImpl(options.include)

private object NullScrubber extends UnrequestedFieldScrubber {

def scrub(tweetResult: TweetResult): TweetResult = tweetResult

def scrub(tweetData: TweetData): TweetData = tweetData

def scrub(tweet: Tweet): Tweet = tweet

}

class ScrubberImpl(i: TweetQuery.Include) extends UnrequestedFieldScrubber {

def scrub(tweetResult: TweetResult): TweetResult =

tweetResult.map(scrub(\_))

def scrub(tweetData: TweetData): TweetData =

tweetData.copy(

tweet = scrub(tweetData.tweet),

sourceTweetResult = tweetData.sourceTweetResult.map(scrub(\_)),

quotedTweetResult =

if (!i.quotedTweet) None

else tweetData.quotedTweetResult.map(qtr => qtr.map(scrub))

)

def scrub(tweet: Tweet): Tweet = {

val tweet2 = scrubKnownFields(tweet)

val unhandledFields = i.tweetFields -- AdditionalFields.CompiledFieldIds

if (unhandledFields.isEmpty) {

tweet2

} else {

tweet2.unsetFields(unhandledFields)

}

}

def scrubKnownFields(tweet: Tweet): Tweet = {

@inline

def filter[A](fieldId: FieldId, value: Option[A]): Option[A] =

if (i.tweetFields.contains(fieldId)) value else None

tweet.copy(

coreData = filter(Tweet.CoreDataField.id, tweet.coreData),

urls = filter(Tweet.UrlsField.id, tweet.urls),

mentions = filter(Tweet.MentionsField.id, tweet.mentions),

hashtags = filter(Tweet.HashtagsField.id, tweet.hashtags),

cashtags = filter(Tweet.CashtagsField.id, tweet.cashtags),

media = filter(Tweet.MediaField.id, tweet.media),

place = filter(Tweet.PlaceField.id, tweet.place),

quotedTweet = filter(Tweet.QuotedTweetField.id, tweet.quotedTweet),

takedownCountryCodes =

filter(Tweet.TakedownCountryCodesField.id, tweet.takedownCountryCodes),

counts = filter(Tweet.CountsField.id, tweet.counts.map(scrub)),

deviceSource = filter(Tweet.DeviceSourceField.id, tweet.deviceSource),

perspective = filter(Tweet.PerspectiveField.id, tweet.perspective),

cards = filter(Tweet.CardsField.id, tweet.cards),

card2 = filter(Tweet.Card2Field.id, tweet.card2),

language = filter(Tweet.LanguageField.id, tweet.language),

spamLabels = None, // unused

contributor = filter(Tweet.ContributorField.id, tweet.contributor),

profileGeoEnrichment =

filter(Tweet.ProfileGeoEnrichmentField.id, tweet.profileGeoEnrichment),

conversationMuted = filter(Tweet.ConversationMutedField.id, tweet.conversationMuted),

takedownReasons = filter(Tweet.TakedownReasonsField.id, tweet.takedownReasons),

selfThreadInfo = filter(Tweet.SelfThreadInfoField.id, tweet.selfThreadInfo),

// additional fields

mediaTags = filter(Tweet.MediaTagsField.id, tweet.mediaTags),

schedulingInfo = filter(Tweet.SchedulingInfoField.id, tweet.schedulingInfo),

bindingValues = filter(Tweet.BindingValuesField.id, tweet.bindingValues),

replyAddresses = None, // unused

obsoleteTwitterSuggestInfo = None, // unused

escherbirdEntityAnnotations =

filter(Tweet.EscherbirdEntityAnnotationsField.id, tweet.escherbirdEntityAnnotations),

spamLabel = filter(Tweet.SpamLabelField.id, tweet.spamLabel),

abusiveLabel = filter(Tweet.AbusiveLabelField.id, tweet.abusiveLabel),

lowQualityLabel = filter(Tweet.LowQualityLabelField.id, tweet.lowQualityLabel),

nsfwHighPrecisionLabel =

filter(Tweet.NsfwHighPrecisionLabelField.id, tweet.nsfwHighPrecisionLabel),

nsfwHighRecallLabel = filter(Tweet.NsfwHighRecallLabelField.id, tweet.nsfwHighRecallLabel),

abusiveHighRecallLabel =

filter(Tweet.AbusiveHighRecallLabelField.id, tweet.abusiveHighRecallLabel),

lowQualityHighRecallLabel =

filter(Tweet.LowQualityHighRecallLabelField.id, tweet.lowQualityHighRecallLabel),

personaNonGrataLabel =

filter(Tweet.PersonaNonGrataLabelField.id, tweet.personaNonGrataLabel),

recommendationsLowQualityLabel = filter(

Tweet.RecommendationsLowQualityLabelField.id,

tweet.recommendationsLowQualityLabel

),

experimentationLabel =

filter(Tweet.ExperimentationLabelField.id, tweet.experimentationLabel),

tweetLocationInfo = filter(Tweet.TweetLocationInfoField.id, tweet.tweetLocationInfo),

cardReference = filter(Tweet.CardReferenceField.id, tweet.cardReference),

supplementalLanguage =

filter(Tweet.SupplementalLanguageField.id, tweet.supplementalLanguage),

selfPermalink = filter(Tweet.SelfPermalinkField.id, tweet.selfPermalink),

extendedTweetMetadata =

filter(Tweet.ExtendedTweetMetadataField.id, tweet.extendedTweetMetadata),

communities = filter(Tweet.CommunitiesField.id, tweet.communities),

visibleTextRange = filter(Tweet.VisibleTextRangeField.id, tweet.visibleTextRange),

spamHighRecallLabel = filter(Tweet.SpamHighRecallLabelField.id, tweet.spamHighRecallLabel),

duplicateContentLabel =

filter(Tweet.DuplicateContentLabelField.id, tweet.duplicateContentLabel),

liveLowQualityLabel = filter(Tweet.LiveLowQualityLabelField.id, tweet.liveLowQualityLabel),

nsfaHighRecallLabel = filter(Tweet.NsfaHighRecallLabelField.id, tweet.nsfaHighRecallLabel),

pdnaLabel = filter(Tweet.PdnaLabelField.id, tweet.pdnaLabel),

searchBlacklistLabel =

filter(Tweet.SearchBlacklistLabelField.id, tweet.searchBlacklistLabel),

lowQualityMentionLabel =

filter(Tweet.LowQualityMentionLabelField.id, tweet.lowQualityMentionLabel),

bystanderAbusiveLabel =

filter(Tweet.BystanderAbusiveLabelField.id, tweet.bystanderAbusiveLabel),

automationHighRecallLabel =

filter(Tweet.AutomationHighRecallLabelField.id, tweet.automationHighRecallLabel),

goreAndViolenceLabel =

filter(Tweet.GoreAndViolenceLabelField.id, tweet.goreAndViolenceLabel),

untrustedUrlLabel = filter(Tweet.UntrustedUrlLabelField.id, tweet.untrustedUrlLabel),

goreAndViolenceHighRecallLabel = filter(

Tweet.GoreAndViolenceHighRecallLabelField.id,

tweet.goreAndViolenceHighRecallLabel

),

nsfwVideoLabel = filter(Tweet.NsfwVideoLabelField.id, tweet.nsfwVideoLabel),

nsfwNearPerfectLabel =

filter(Tweet.NsfwNearPerfectLabelField.id, tweet.nsfwNearPerfectLabel),

automationLabel = filter(Tweet.AutomationLabelField.id, tweet.automationLabel),

nsfwCardImageLabel = filter(Tweet.NsfwCardImageLabelField.id, tweet.nsfwCardImageLabel),

duplicateMentionLabel =

filter(Tweet.DuplicateMentionLabelField.id, tweet.duplicateMentionLabel),

bounceLabel = filter(Tweet.BounceLabelField.id, tweet.bounceLabel),

selfThreadMetadata = filter(Tweet.SelfThreadMetadataField.id, tweet.selfThreadMetadata),

composerSource = filter(Tweet.ComposerSourceField.id, tweet.composerSource),

editControl = filter(Tweet.EditControlField.id, tweet.editControl),

developerBuiltCardId = filter(

Tweet.DeveloperBuiltCardIdField.id,

tweet.developerBuiltCardId

),

creativeEntityEnrichmentsForTweet = filter(

Tweet.CreativeEntityEnrichmentsForTweetField.id,

tweet.creativeEntityEnrichmentsForTweet

),

previousCounts = filter(Tweet.PreviousCountsField.id, tweet.previousCounts),

mediaRefs = filter(Tweet.MediaRefsField.id, tweet.mediaRefs),

isCreativesContainerBackendTweet = filter(

Tweet.IsCreativesContainerBackendTweetField.id,

tweet.isCreativesContainerBackendTweet),

editPerspective = filter(Tweet.EditPerspectiveField.id, tweet.editPerspective),

noteTweet = filter(Tweet.NoteTweetField.id, tweet.noteTweet),

// tweetypie-internal metadata

directedAtUserMetadata =

filter(Tweet.DirectedAtUserMetadataField.id, tweet.directedAtUserMetadata),

tweetypieOnlyTakedownReasons =

filter(Tweet.TweetypieOnlyTakedownReasonsField.id, tweet.tweetypieOnlyTakedownReasons),

mediaKeys = filter(Tweet.MediaKeysField.id, tweet.mediaKeys),

tweetypieOnlyTakedownCountryCodes = filter(

Tweet.TweetypieOnlyTakedownCountryCodesField.id,

tweet.tweetypieOnlyTakedownCountryCodes

),

underlyingCreativesContainerId = filter(

Tweet.UnderlyingCreativesContainerIdField.id,

tweet.underlyingCreativesContainerId),

unmentionData = filter(Tweet.UnmentionDataField.id, tweet.unmentionData),

blockingUnmentions = filter(Tweet.BlockingUnmentionsField.id, tweet.blockingUnmentions),

settingsUnmentions = filter(Tweet.SettingsUnmentionsField.id, tweet.settingsUnmentions)

)

}

def scrub(counts: StatusCounts): StatusCounts = {

@inline

def filter[A](fieldId: FieldId, value: Option[A]): Option[A] =

if (i.countsFields.contains(fieldId)) value else None

StatusCounts(

replyCount = filter(StatusCounts.ReplyCountField.id, counts.replyCount),

favoriteCount = filter(StatusCounts.FavoriteCountField.id, counts.favoriteCount),

retweetCount = filter(StatusCounts.RetweetCountField.id, counts.retweetCount),

quoteCount = filter(StatusCounts.QuoteCountField.id, counts.quoteCount),

bookmarkCount = filter(StatusCounts.BookmarkCountField.id, counts.bookmarkCount)

)

}

def scrub(media: MediaEntity): MediaEntity = {

@inline

def filter[A](fieldId: FieldId, value: Option[A]): Option[A] =

if (i.mediaFields.contains(fieldId)) value else None

media.copy(

additionalMetadata =

filter(MediaEntity.AdditionalMetadataField.id, media.additionalMetadata)

)

}

}

}