package com.twitter.product\_mixer.component\_library.feature\_hydrator.query.async

import com.twitter.ml.featurestore.lib.EntityId

import com.twitter.product\_mixer.core.feature.Feature

import com.twitter.product\_mixer.core.feature.featuremap.FeatureMap

import com.twitter.product\_mixer.core.feature.featurestorev1.BaseFeatureStoreV1QueryFeature

import com.twitter.product\_mixer.core.functional\_component.common.alert.Alert

import com.twitter.product\_mixer.core.functional\_component.feature\_hydrator.AsyncHydrator

import com.twitter.product\_mixer.core.functional\_component.feature\_hydrator.QueryFeatureHydrator

import com.twitter.product\_mixer.core.functional\_component.feature\_hydrator.featurestorev1.FeatureStoreV1DynamicClientBuilder

import com.twitter.product\_mixer.core.functional\_component.feature\_hydrator.featurestorev1.FeatureStoreV1QueryFeatureHydrator

import com.twitter.product\_mixer.core.model.common.identifier.FeatureHydratorIdentifier

import com.twitter.product\_mixer.core.model.common.identifier.PipelineStepIdentifier

import com.twitter.product\_mixer.core.pipeline.PipelineQuery

import com.twitter.stitch.Stitch

/\*\*

\* A [[QueryFeatureHydrator]] with [[AsyncQueryFeatureHydrator]] that hydrated asynchronously for features

\* to be before the step identified in [[hydrateBefore]]

\*

\* @param hydrateBefore the [[PipelineStepIdentifier]] step to make sure this feature is hydrated before.

\* @param queryFeatureHydrator the underlying [[QueryFeatureHydrator]] to run asynchronously

\* @tparam Query The domain model for the query or request

\*/

case class AsyncQueryFeatureHydrator[-Query <: PipelineQuery] private[async] (

override val hydrateBefore: PipelineStepIdentifier,

queryFeatureHydrator: QueryFeatureHydrator[Query])

extends QueryFeatureHydrator[Query]

with AsyncHydrator {

override val identifier: FeatureHydratorIdentifier = FeatureHydratorIdentifier(

"Async" + queryFeatureHydrator.identifier.name)

override val alerts: Seq[Alert] = queryFeatureHydrator.alerts

override val features: Set[Feature[\_, \_]] = queryFeatureHydrator.features

override def hydrate(query: Query): Stitch[FeatureMap] = queryFeatureHydrator.hydrate(query)

}

/\*\*

\* A [[FeatureStoreV1QueryFeatureHydrator]] with [[AsyncHydrator]] that hydrated asynchronously for features

\* to be before the step identified in [[hydrateBefore]]. We need a standalone class for feature store,

\* different from the above as FStore hydrators are exempt from validations at run time.

\*

\* @param hydrateBefore the [[PipelineStepIdentifier]] step to make sure this feature is hydrated before.

\* @param queryFeatureHydrator the underlying [[QueryFeatureHydrator]] to run asynchronously

\* @tparam Query The domain model for the query or request

\*/

case class AsyncFeatureStoreV1QueryFeatureHydrator[Query <: PipelineQuery] private[async] (

override val hydrateBefore: PipelineStepIdentifier,

featureStoreV1QueryFeatureHydrator: FeatureStoreV1QueryFeatureHydrator[Query])

extends FeatureStoreV1QueryFeatureHydrator[

Query

]

with AsyncHydrator {

override val identifier: FeatureHydratorIdentifier = FeatureHydratorIdentifier(

"Async" + featureStoreV1QueryFeatureHydrator.identifier.name)

override val alerts: Seq[Alert] = featureStoreV1QueryFeatureHydrator.alerts

override val features: Set[BaseFeatureStoreV1QueryFeature[Query, \_ <: EntityId, \_]] =

featureStoreV1QueryFeatureHydrator.features

override val clientBuilder: FeatureStoreV1DynamicClientBuilder =

featureStoreV1QueryFeatureHydrator.clientBuilder

}

object AsyncQueryFeatureHydrator {

/\*\*

\* A [[QueryFeatureHydrator]] with [[AsyncQueryFeatureHydrator]] that hydrated asynchronously for features

\* to be before the step identified in [[hydrateBefore]]

\*

\* @param hydrateBefore the [[PipelineStepIdentifier]] step to make sure this feature is hydrated before.

\* @param queryFeatureHydrator the underlying [[QueryFeatureHydrator]] to run asynchronously

\* @tparam Query The domain model for the query or request

\*/

def apply[Query <: PipelineQuery](

hydrateBefore: PipelineStepIdentifier,

queryFeatureHydrator: QueryFeatureHydrator[Query]

): AsyncQueryFeatureHydrator[Query] =

new AsyncQueryFeatureHydrator(hydrateBefore, queryFeatureHydrator)

/\*\*

\* A [[FeatureStoreV1QueryFeatureHydrator]] with [[AsyncHydrator]] that hydrated asynchronously for features

\* to be before the step identified in [[hydrateBefore]]. We need a standalone class for feature store,

\* different from the above as FStore hydrators are exempt from validations at run time.

\*

\* @param hydrateBefore the [[PipelineStepIdentifier]] step to make sure this feature is hydrated before.

\* @param queryFeatureHydrator the underlying [[QueryFeatureHydrator]] to run asynchronously

\* @tparam Query The domain model for the query or request

\*/

def apply[Query <: PipelineQuery](

hydrateBefore: PipelineStepIdentifier,

featureStoreV1QueryFeatureHydrator: FeatureStoreV1QueryFeatureHydrator[Query]

): AsyncFeatureStoreV1QueryFeatureHydrator[Query] =

new AsyncFeatureStoreV1QueryFeatureHydrator(hydrateBefore, featureStoreV1QueryFeatureHydrator)

}