package com.twitter.product\_mixer.core.functional\_component.premarshaller

import com.twitter.product\_mixer.core.functional\_component.marshaller.TransportMarshaller

import com.twitter.product\_mixer.core.model.common.Component

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

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

import com.twitter.product\_mixer.core.model.common.presentation.CandidateWithDetails

import com.twitter.product\_mixer.core.model.common.presentation.ModulePresentation

import com.twitter.product\_mixer.core.model.common.presentation.UniversalPresentation

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

/\*\*

\* Transforms the `selections` into a [[DomainResponseType]] object (often URT, Slice, etc)

\*

\* [[DomainMarshaller]]s may contain business logic

\*

\* @note This is different from `com.twitter.product\_mixer.core.marshaller`s

\* which transforms into a wire-compatible type

\*/

trait DomainMarshaller[-Query <: PipelineQuery, DomainResponseType] extends Component {

override val identifier: DomainMarshallerIdentifier

/\*\* Transforms the `selections` into a [[DomainResponseType]] object \*/

def apply(

query: Query,

selections: Seq[CandidateWithDetails]

): DomainResponseType

}

class UnsupportedCandidateDomainMarshallerException(

candidate: Any,

candidateSource: ComponentIdentifier)

extends UnsupportedOperationException(

s"Domain marshaller does not support candidate ${TransportMarshaller.getSimpleName(

candidate.getClass)} from source $candidateSource")

class UndecoratedCandidateDomainMarshallerException(

candidate: Any,

candidateSource: ComponentIdentifier)

extends UnsupportedOperationException(

s"Domain marshaller does not support undecorated candidate ${TransportMarshaller

.getSimpleName(candidate.getClass)} from source $candidateSource")

class UnsupportedPresentationDomainMarshallerException(

candidate: Any,

presentation: UniversalPresentation,

candidateSource: ComponentIdentifier)

extends UnsupportedOperationException(

s"Domain marshaller does not support decorator presentation ${TransportMarshaller

.getSimpleName(presentation.getClass)} for candidate ${TransportMarshaller.getSimpleName(

candidate.getClass)} from source $candidateSource")

class UnsupportedModuleDomainMarshallerException(

presentation: Option[ModulePresentation],

candidateSource: ComponentIdentifier)

extends UnsupportedOperationException(

s"Domain marshaller does not support module presentation ${presentation

.map(p =>

TransportMarshaller

.getSimpleName(presentation.getClass)).getOrElse("")} but was given a module from source $candidateSource")

class UndecoratedModuleDomainMarshallerException(

candidateSource: ComponentIdentifier)

extends UnsupportedOperationException(

s"Domain marshaller does not support undecorated module from source $candidateSource")