package com.twitter.product\_mixer.core.functional\_component.marshaller.response.urt.item.tombstone

import com.twitter.product\_mixer.core.model.marshalling.response.urt.item.tombstone.DisconnectedRepliesAncestor

import com.twitter.product\_mixer.core.model.marshalling.response.urt.item.tombstone.DisconnectedRepliesDescendant

import com.twitter.product\_mixer.core.model.marshalling.response.urt.item.tombstone.Inline

import com.twitter.product\_mixer.core.model.marshalling.response.urt.item.tombstone.NonCompliant

import com.twitter.product\_mixer.core.model.marshalling.response.urt.item.tombstone.TombstoneDisplayType

import com.twitter.product\_mixer.core.model.marshalling.response.urt.item.tombstone.TweetUnavailable

import com.twitter.timelines.render.{thriftscala => urt}

import javax.inject.Inject

import javax.inject.Singleton

@Singleton

class TombstoneDisplayTypeMarshaller @Inject() () {

def apply(tombstoneDisplayType: TombstoneDisplayType): urt.TombstoneDisplayType =

tombstoneDisplayType match {

case TweetUnavailable => urt.TombstoneDisplayType.TweetUnavailable

case DisconnectedRepliesAncestor => urt.TombstoneDisplayType.DisconnectedRepliesAncestor

case DisconnectedRepliesDescendant => urt.TombstoneDisplayType.DisconnectedRepliesDescendant

case Inline => urt.TombstoneDisplayType.Inline

case NonCompliant => urt.TombstoneDisplayType.NonCompliant

}

}