package com.twitter.product\_mixer.component\_library.decorator.urt.builder.flexible\_injection\_pipeline

import com.twitter.onboarding.injections.{thriftscala => onboardingthrift}

import com.twitter.product\_mixer.core.model.marshalling.response.urt.cover.CenterCoverHalfCoverDisplayType

import com.twitter.product\_mixer.core.model.marshalling.response.urt.cover.CoverBehaviorDismiss

import com.twitter.product\_mixer.core.model.marshalling.response.urt.cover.CoverBehaviorNavigate

import com.twitter.product\_mixer.core.model.marshalling.response.urt.cover.CoverCta

import com.twitter.product\_mixer.core.model.marshalling.response.urt.cover.CoverCtaBehavior

import com.twitter.product\_mixer.core.model.marshalling.response.urt.cover.CoverHalfCoverDisplayType

import com.twitter.product\_mixer.core.model.marshalling.response.urt.cover.CoverImage

import com.twitter.product\_mixer.core.model.marshalling.response.urt.cover.HalfCoverDisplayType

import com.twitter.product\_mixer.core.model.marshalling.response.urt.icon.\_

import com.twitter.product\_mixer.core.model.marshalling.response.urt.item.message.FollowAllMessageActionType

import com.twitter.product\_mixer.core.model.marshalling.response.urt.item.message.LargeUserFacepileDisplayType

import com.twitter.product\_mixer.core.model.marshalling.response.urt.item.message.MessageAction

import com.twitter.product\_mixer.core.model.marshalling.response.urt.item.message.MessageActionType

import com.twitter.product\_mixer.core.model.marshalling.response.urt.item.message.MessageImage

import com.twitter.product\_mixer.core.model.marshalling.response.urt.item.message.MessageTextAction

import com.twitter.product\_mixer.core.model.marshalling.response.urt.item.message.UserFacepile

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.Bounce

import com.twitter.product\_mixer.core.model.marshalling.response.urt.button.ButtonStyle

import com.twitter.product\_mixer.core.model.marshalling.response.urt.button.Default

import com.twitter.product\_mixer.core.model.marshalling.response.urt.button.Primary

import com.twitter.product\_mixer.core.model.marshalling.response.urt.button.Secondary

import com.twitter.product\_mixer.core.model.marshalling.response.urt.button.Text

import com.twitter.product\_mixer.core.model.marshalling.response.urt.button.Destructive

import com.twitter.product\_mixer.core.model.marshalling.response.urt.button.Neutral

import com.twitter.product\_mixer.core.model.marshalling.response.urt.button.DestructiveSecondary

import com.twitter.product\_mixer.core.model.marshalling.response.urt.button.DestructiveText

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.Callback

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.ClientEventInfo

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.DeepLink

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.Dismiss

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.DismissInfo

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.ExternalUrl

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.FeedbackAction

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.FeedbackActionInfo

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.FollowGeneralContextType

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.GeneralContext

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.ImageAnimationType

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.ImageDisplayType

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.ImageVariant

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.FullWidth

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.Icon

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.IconSmall

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.SocialContext

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.Url

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.UrtEndpoint

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.UrtEndpointOptions

import com.twitter.product\_mixer.core.model.marshalling.response.urt.richtext.Center

import com.twitter.product\_mixer.core.model.marshalling.response.urt.richtext.Natural

import com.twitter.product\_mixer.core.model.marshalling.response.urt.richtext.Plain

import com.twitter.product\_mixer.core.model.marshalling.response.urt.richtext.ReferenceObject

import com.twitter.product\_mixer.core.model.marshalling.response.urt.richtext.RichText

import com.twitter.product\_mixer.core.model.marshalling.response.urt.richtext.RichTextAlignment

import com.twitter.product\_mixer.core.model.marshalling.response.urt.richtext.RichTextEntity

import com.twitter.product\_mixer.core.model.marshalling.response.urt.richtext.RichTextFormat

import com.twitter.product\_mixer.core.model.marshalling.response.urt.richtext.RichTextCashtag

import com.twitter.product\_mixer.core.model.marshalling.response.urt.richtext.RichTextHashtag

import com.twitter.product\_mixer.core.model.marshalling.response.urt.richtext.RichTextList

import com.twitter.product\_mixer.core.model.marshalling.response.urt.richtext.RichTextMention

import com.twitter.product\_mixer.core.model.marshalling.response.urt.richtext.RichTextUser

import com.twitter.product\_mixer.core.model.marshalling.response.urt.richtext.Strong

/\*\*\*

\* Helper class to convert onboarding thrift to product-mixer models

\*/

object OnboardingInjectionConversions {

def convertFeedbackInfo(

feedbackInfo: onboardingthrift.FeedbackInfo

): FeedbackActionInfo = {

val actions = feedbackInfo.actions.map {

case onboardingthrift.FeedbackAction.DismissAction(dismissAction) =>

FeedbackAction(

Dismiss,

prompt = dismissAction.prompt,

confirmation = dismissAction.confirmation,

hasUndoAction = dismissAction.hasUndoAction,

feedbackUrl = dismissAction.feedbackUrl,

childFeedbackActions =

None,

confirmationDisplayType = None,

clientEventInfo = None,

icon = None,

richBehavior = None,

subprompt = None,

encodedFeedbackRequest = None

)

case onboardingthrift.FeedbackAction.UnknownUnionField(value) =>

throw new UnsupportedOperationException(s"Unknown product: $value")

}

FeedbackActionInfo(

feedbackActions = actions,

feedbackMetadata = None,

displayContext = None,

clientEventInfo = None)

}

def convertClientEventInfo(input: onboardingthrift.ClientEventInfo): ClientEventInfo =

ClientEventInfo(

component = input.component,

element = input.element,

details = None,

action = input.action,

entityToken = None)

def convertCallback(callback: onboardingthrift.Callback): Callback =

Callback(callback.endpoint)

def convertImage(image: onboardingthrift.Image): MessageImage =

MessageImage(

Set(convertImageVariant(image.image)),

backgroundColor =

None

)

def convertCoverImage(image: onboardingthrift.Image): CoverImage =

CoverImage(

convertImageVariant(image.image),

imageDisplayType = convertImageDisplayType(image.imageDisplayType),

imageAnimationType = image.imageAnimationType.map(convertImageAnimationType),

)

def convertImageDisplayType(

imageDisplayType: onboardingthrift.ImageDisplayType

): ImageDisplayType =

imageDisplayType match {

case onboardingthrift.ImageDisplayType.Icon => Icon

case onboardingthrift.ImageDisplayType.FullWidth => FullWidth

case onboardingthrift.ImageDisplayType.IconSmall => IconSmall

case onboardingthrift.ImageDisplayType.EnumUnknownImageDisplayType(value) =>

throw new UnsupportedOperationException(s"Unknown product: $value")

}

private def convertImageAnimationType(

imageAnimationType: onboardingthrift.ImageAnimationType

): ImageAnimationType =

imageAnimationType match {

case onboardingthrift.ImageAnimationType.Bounce => Bounce

case onboardingthrift.ImageAnimationType.EnumUnknownImageAnimationType(value) =>

throw new UnsupportedOperationException(s"Unknown product: $value")

}

def convertImageVariant(imageVariant: onboardingthrift.ImageVariant): ImageVariant =

ImageVariant(

url = imageVariant.url,

width = imageVariant.width,

height = imageVariant.height,

palette = None)

def convertButtonAction(

buttonAction: onboardingthrift.ButtonAction

): MessageTextAction =

MessageTextAction(

buttonAction.text,

MessageAction(

dismissOnClick = buttonAction.dismissOnClick.getOrElse(true),

url = getActionUrl(buttonAction),

clientEventInfo = Some(convertClientEventInfo(buttonAction.clientEventInfo)),

onClickCallbacks = buttonAction.callbacks.map(\_.map(convertCallback).toList)

)

)

private def getActionUrl(buttonAction: onboardingthrift.ButtonAction) =

buttonAction.buttonBehavior match {

case onboardingthrift.ButtonBehavior.Navigate(navigate) => Some(navigate.url.url)

case onboardingthrift.ButtonBehavior.Dismiss(\_) => None

case onboardingthrift.ButtonBehavior.UnknownUnionField(value) =>

throw new UnsupportedOperationException(s"Unknown product: $value")

}

def convertRichText(

richText: com.twitter.onboarding.injections.thriftscala.RichText

): RichText = {

val entities = richText.entities.map(entity =>

RichTextEntity(

entity.fromIndex,

entity.toIndex,

entity.ref.map(convertRef),

entity.format.map(convertFormat)))

RichText(

text = richText.text,

entities = entities.toList,

rtl = richText.rtl,

alignment = richText.alignment.map(convertAlignment))

}

private def convertAlignment(alignment: onboardingthrift.RichTextAlignment): RichTextAlignment =

alignment match {

case onboardingthrift.RichTextAlignment.Natural => Natural

case onboardingthrift.RichTextAlignment.Center => Center

case onboardingthrift.RichTextAlignment.EnumUnknownRichTextAlignment(value) =>

throw new UnsupportedOperationException(s"Unknown product: $value")

}

private def convertRef(ref: onboardingthrift.ReferenceObject): ReferenceObject =

ref match {

case onboardingthrift.ReferenceObject.User(user) => RichTextUser(user.id)

case onboardingthrift.ReferenceObject.Mention(mention) =>

RichTextMention(mention.id, mention.screenName)

case onboardingthrift.ReferenceObject.Hashtag(hashtag) => RichTextHashtag(hashtag.text)

case onboardingthrift.ReferenceObject.Cashtag(cashtag) => RichTextCashtag(cashtag.text)

case onboardingthrift.ReferenceObject.TwitterList(twList) =>

RichTextList(twList.id, twList.url)

case onboardingthrift.ReferenceObject.Url(url) => RichTextHashtag(url.url)

case onboardingthrift.ReferenceObject.UnknownUnionField(value) =>

throw new UnsupportedOperationException(s"Unknown product: $value")

}

private def convertFormat(format: onboardingthrift.RichTextFormat): RichTextFormat =

format match {

case onboardingthrift.RichTextFormat.Plain => Plain

case onboardingthrift.RichTextFormat.Strong => Strong

case onboardingthrift.RichTextFormat.EnumUnknownRichTextFormat(value) =>

throw new UnsupportedOperationException(s"Unknown product: $value")

}

// Specific to Message prompt

def convertSocialContext(socialContext: onboardingthrift.RichText): SocialContext =

GeneralContext(

contextType = FollowGeneralContextType,

text = socialContext.text,

url = None,

contextImageUrls = None,

landingUrl = None)

def convertUserFacePile(

userFacepile: onboardingthrift.PromptUserFacepile

): UserFacepile =

UserFacepile(

userIds = userFacepile.userIds.toList,

featuredUserIds = userFacepile.featuredUserIds.toList,

action = userFacepile.action.map(convertButtonAction),

actionType = userFacepile.actionType.map(convertUserFacePileActionType),

displaysFeaturingText = userFacepile.displaysFeaturingText,

displayType = Some(LargeUserFacepileDisplayType)

)

private def convertUserFacePileActionType(

actionType: onboardingthrift.FacepileActionType

): MessageActionType =

actionType match {

case onboardingthrift.FacepileActionType.FollowAll => FollowAllMessageActionType

case onboardingthrift.FacepileActionType.EnumUnknownFacepileActionType(value) =>

throw new UnsupportedOperationException(s"Unknown product: $value")

}

// Specific to Cover

def convertHalfCoverDisplayType(

displayType: onboardingthrift.HalfCoverDisplayType

): HalfCoverDisplayType =

displayType match {

case onboardingthrift.HalfCoverDisplayType.Cover => CoverHalfCoverDisplayType

case onboardingthrift.HalfCoverDisplayType.CenterCover =>

CenterCoverHalfCoverDisplayType

case onboardingthrift.HalfCoverDisplayType.EnumUnknownHalfCoverDisplayType(value) =>

throw new UnsupportedOperationException(s"Unknown product: $value")

}

def convertDismissInfo(dismissInfo: onboardingthrift.DismissInfo): DismissInfo =

DismissInfo(dismissInfo.callbacks.map(\_.map(convertCallback)))

def convertCoverCta(

buttonAction: onboardingthrift.ButtonAction

): CoverCta =

CoverCta(

buttonAction.text,

ctaBehavior = convertCoverCtaBehavior(buttonAction.buttonBehavior),

callbacks = buttonAction.callbacks.map(\_.map(convertCallback).toList),

clientEventInfo = Some(convertClientEventInfo(buttonAction.clientEventInfo)),

icon = buttonAction.icon.map(covertHorizonIcon),

buttonStyle = buttonAction.buttonStyle.map(covertButtonStyle)

)

private def convertCoverCtaBehavior(

behavior: onboardingthrift.ButtonBehavior

): CoverCtaBehavior =

behavior match {

case onboardingthrift.ButtonBehavior.Navigate(navigate) =>

CoverBehaviorNavigate(convertUrl(navigate.url))

case onboardingthrift.ButtonBehavior.Dismiss(dismiss) =>

CoverBehaviorDismiss(dismiss.feedbackMessage.map(convertRichText))

case onboardingthrift.ButtonBehavior.UnknownUnionField(value) =>

throw new UnsupportedOperationException(s"Unknown product: $value")

}

private def covertButtonStyle(bStyle: onboardingthrift.CtaButtonStyle): ButtonStyle =

bStyle match {

case onboardingthrift.CtaButtonStyle.Default => Default

case onboardingthrift.CtaButtonStyle.Primary => Primary

case onboardingthrift.CtaButtonStyle.Secondary => Secondary

case onboardingthrift.CtaButtonStyle.Text => Text

case onboardingthrift.CtaButtonStyle.Destructive => Destructive

case onboardingthrift.CtaButtonStyle.Neutral => Neutral

case onboardingthrift.CtaButtonStyle.DestructiveSecondary => DestructiveSecondary

case onboardingthrift.CtaButtonStyle.DestructiveText => DestructiveText

case onboardingthrift.CtaButtonStyle.EnumUnknownCtaButtonStyle(value) =>

throw new UnsupportedOperationException(s"Unknown product: $value")

}

private def covertHorizonIcon(icon: onboardingthrift.HorizonIcon): HorizonIcon =

icon match {

case onboardingthrift.HorizonIcon.Bookmark => Bookmark

case onboardingthrift.HorizonIcon.Moment => Moment

case onboardingthrift.HorizonIcon.Debug => Debug

case onboardingthrift.HorizonIcon.Error => Error

case onboardingthrift.HorizonIcon.Follow => Follow

case onboardingthrift.HorizonIcon.Unfollow => Unfollow

case onboardingthrift.HorizonIcon.Smile => Smile

case onboardingthrift.HorizonIcon.Frown => Frown

case onboardingthrift.HorizonIcon.Help => Help

case onboardingthrift.HorizonIcon.Link => Link

case onboardingthrift.HorizonIcon.Message => Message

case onboardingthrift.HorizonIcon.No => No

case onboardingthrift.HorizonIcon.Outgoing => Outgoing

case onboardingthrift.HorizonIcon.Pin => Pin

case onboardingthrift.HorizonIcon.Retweet => Retweet

case onboardingthrift.HorizonIcon.Speaker => Speaker

case onboardingthrift.HorizonIcon.Trashcan => Trashcan

case onboardingthrift.HorizonIcon.Feedback => Feedback

case onboardingthrift.HorizonIcon.FeedbackClose => FeedbackClose

case onboardingthrift.HorizonIcon.EyeOff => EyeOff

case onboardingthrift.HorizonIcon.Moderation => Moderation

case onboardingthrift.HorizonIcon.Topic => Topic

case onboardingthrift.HorizonIcon.TopicClose => TopicClose

case onboardingthrift.HorizonIcon.Flag => Flag

case onboardingthrift.HorizonIcon.TopicFilled => TopicFilled

case onboardingthrift.HorizonIcon.NotificationsFollow => NotificationsFollow

case onboardingthrift.HorizonIcon.Person => Person

case onboardingthrift.HorizonIcon.Logo => Logo

case onboardingthrift.HorizonIcon.EnumUnknownHorizonIcon(value) =>

throw new UnsupportedOperationException(s"Unknown product: $value")

}

def convertUrl(url: onboardingthrift.Url): Url = {

val urlType = url.urlType match {

case onboardingthrift.UrlType.ExternalUrl => ExternalUrl

case onboardingthrift.UrlType.DeepLink => DeepLink

case onboardingthrift.UrlType.UrtEndpoint => UrtEndpoint

case onboardingthrift.UrlType.EnumUnknownUrlType(value) =>

throw new UnsupportedOperationException(s"Unknown product: $value")

}

Url(urlType, url.url, url.urtEndpointOptions.map(convertUrtEndpointOptions))

}

private def convertUrtEndpointOptions(

urtEndpointOptions: onboardingthrift.UrtEndpointOptions

): UrtEndpointOptions =

UrtEndpointOptions(

requestParams = urtEndpointOptions.requestParams.map(\_.toMap),

title = urtEndpointOptions.title,

cacheId = urtEndpointOptions.cacheId,

subtitle = urtEndpointOptions.subtitle

)

}