package com.twitter.product\_mixer.component\_library.decorator.urt.builder.item.message

import com.twitter.product\_mixer.component\_library.decorator.urt.builder.item.message.CompactPromptCandidateUrtItemStringCenterBuilder.CompactPromptClientEventInfoElement

import com.twitter.product\_mixer.component\_library.model.candidate.CompactPromptCandidate

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

import com.twitter.product\_mixer.core.functional\_component.decorator.urt.builder.CandidateUrtEntryBuilder

import com.twitter.product\_mixer.core.functional\_component.decorator.urt.builder.metadata.BaseClientEventInfoBuilder

import com.twitter.product\_mixer.core.functional\_component.decorator.urt.builder.metadata.BaseFeedbackActionInfoBuilder

import com.twitter.product\_mixer.core.functional\_component.decorator.urt.builder.metadata.BaseStr

import com.twitter.product\_mixer.core.functional\_component.decorator.urt.builder.richtext.BaseRichTextBuilder

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

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

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

object CompactPromptCandidateUrtItemStringCenterBuilder {

val CompactPromptClientEventInfoElement: String = "message"

}

case class CompactPromptCandidateUrtItemStringCenterBuilder[-Query <: PipelineQuery](

clientEventInfoBuilder: BaseClientEventInfoBuilder[Query, CompactPromptCandidate],

feedbackActionInfoBuilder: Option[

BaseFeedbackActionInfoBuilder[Query, CompactPromptCandidate]

] = None,

headerTextBuilder: BaseStr[Query, CompactPromptCandidate],

bodyTextBuilder: Option[BaseStr[Query, CompactPromptCandidate]] = None,

headerRichTextBuilder: Option[BaseRichTextBuilder[Query, CompactPromptCandidate]] = None,

bodyRichTextBuilder: Option[BaseRichTextBuilder[Query, CompactPromptCandidate]] = None)

extends CandidateUrtEntryBuilder[Query, CompactPromptCandidate, MessagePromptItem] {

override def apply(

query: Query,

compactPromptCandidate: CompactPromptCandidate,

candidateFeatures: FeatureMap

): MessagePromptItem =

MessagePromptItem(

id = compactPromptCandidate.id.toString,

sortIndex = None, // Sort indexes are automatically set in the domain marshaller phase

clientEventInfo = clientEventInfoBuilder(

query,

compactPromptCandidate,

candidateFeatures,

Some(CompactPromptClientEventInfoElement)),

feedbackActionInfo = feedbackActionInfoBuilder.flatMap(

\_.apply(query, compactPromptCandidate, candidateFeatures)),

isPinned = None,

content = CompactPromptMessageContent(

headerText = headerTextBuilder.apply(query, compactPromptCandidate, candidateFeatures),

bodyText = bodyTextBuilder.map(\_.apply(query, compactPromptCandidate, candidateFeatures)),

primaryButtonAction = None,

secondaryButtonAction = None,

action = None,

headerRichText =

headerRichTextBuilder.map(\_.apply(query, compactPromptCandidate, candidateFeatures)),

bodyRichText =

bodyRichTextBuilder.map(\_.apply(query, compactPromptCandidate, candidateFeatures))

),

impressionCallbacks = None

)

}