package com.twitter.timelineranker.parameters.entity\_tweets

import com.twitter.servo.decider.DeciderGateBuilder

import com.twitter.servo.decider.DeciderKeyName

import com.twitter.timelineranker.decider.DeciderKey

import com.twitter.timelineranker.parameters.entity\_tweets.EntityTweetsParams.\_

import com.twitter.timelines.configapi.decider.DeciderUtils

import com.twitter.timelines.configapi.BaseConfig

import com.twitter.timelines.configapi.BaseConfigBuilder

import com.twitter.timelines.configapi.FeatureSwitchOverrideUtil

import com.twitter.timelines.configapi.Param

object EntityTweetsProduction {

val deciderByParam: Map[Param[\_], DeciderKeyName] = Map[Param[\_], DeciderKeyName](

EnableContentFeaturesHydrationParam -> DeciderKey.EntityTweetsEnableContentFeaturesHydration

)

}

case class EntityTweetsProduction(deciderGateBuilder: DeciderGateBuilder) {

val booleanDeciderOverrides = DeciderUtils.getBooleanDeciderOverrides(

deciderGateBuilder,

EnableContentFeaturesHydrationParam

)

val booleanFeatureSwitchOverrides = FeatureSwitchOverrideUtil.getBooleanFSOverrides(

EnableTokensInContentFeaturesHydrationParam,

EnableTweetTextInContentFeaturesHydrationParam,

EnableConversationControlInContentFeaturesHydrationParam,

EnableTweetMediaHydrationParam

)

val intFeatureSwitchOverrides = FeatureSwitchOverrideUtil.getBoundedIntFSOverrides(

MaxFollowedUsersParam

)

val config: BaseConfig = new BaseConfigBuilder()

.set(booleanDeciderOverrides: \_\*)

.set(booleanFeatureSwitchOverrides: \_\*)

.set(intFeatureSwitchOverrides: \_\*)

.build()

}