package com.twitter.product\_mixer.core.pipeline.step.selector

import com.twitter.product\_mixer.core.functional\_component.selector.Selector

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

import com.twitter.product\_mixer.core.pipeline.state.HasCandidatesWithDetails

import com.twitter.product\_mixer.core.pipeline.state.HasQuery

import com.twitter.product\_mixer.core.pipeline.step.Step

import com.twitter.product\_mixer.core.service.Executor

import com.twitter.product\_mixer.core.service.selector\_executor.SelectorExecutor

import com.twitter.product\_mixer.core.service.selector\_executor.SelectorExecutorResult

import com.twitter.stitch.Arrow

import javax.inject.Inject

/\*\*

\* A selection step, it takes the input list of candidates with details and the given

\* selectors and executes them to decide which candidates should be selected.

\*

\* @param selectorExecutor Selector Executor

\* @tparam Query Type of PipelineQuery domain model

\* @tparam State The pipeline state domain model.

\*/

case class SelectorStep[

Query <: PipelineQuery,

State <: HasQuery[Query, State] with HasCandidatesWithDetails[State]] @Inject() (

selectorExecutor: SelectorExecutor)

extends Step[State, Seq[

Selector[Query]

], SelectorExecutor.Inputs[

Query

], SelectorExecutorResult] {

override def adaptInput(

state: State,

config: Seq[Selector[Query]]

): SelectorExecutor.Inputs[Query] =

SelectorExecutor.Inputs(state.query, state.candidatesWithDetails)

override def arrow(

config: Seq[Selector[Query]],

context: Executor.Context

): Arrow[SelectorExecutor.Inputs[Query], SelectorExecutorResult] =

selectorExecutor.arrow(config, context)

override def updateState(

input: State,

executorResult: SelectorExecutorResult,

config: Seq[Selector[Query]]

): State = input.updateCandidatesWithDetails(executorResult.selectedCandidates)

// Selection is a bit different to other steps (i.e, other steps, empty means don't change anything)

// where an empty selection list drops all candidates.

override def isEmpty(config: Seq[Selector[Query]]): Boolean = false

}