package com.twitter.product\_mixer.component\_library.selector.sorter

import com.twitter.product\_mixer.core.model.common.presentation.CandidateWithDetails

object SorterFromOrdering {

def apply(ordering: Ordering[CandidateWithDetails], sortOrder: SortOrder): SorterFromOrdering =

SorterFromOrdering(if (sortOrder == Descending) ordering.reverse else ordering)

}

/\*\*

\* Sorts candidates based on the provided [[ordering]]

\*

\* @note the [[Ordering]] must be transitive, so if `A < B` and `B < C` then `A < C`.

\* @note sorting randomly via `Ordering.by[CandidateWithDetails, Double](\_ => Random.nextDouble())`

\* is not safe and can fail at runtime since TimSort depends on stable sort values for

\* pivoting. To sort randomly, use [[RandomShuffleSorter]] instead.

\*/

case class SorterFromOrdering(

ordering: Ordering[CandidateWithDetails])

extends SorterProvider

with Sorter {

override def sort[Candidate <: CandidateWithDetails](candidates: Seq[Candidate]): Seq[Candidate] =

candidates.sorted(ordering)

}