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

import com.twitter.product\_mixer.core.model.common.UniversalNoun

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

/\*\*

\* [[DropSelector]] detects duplicates by looking for candidates with the same key. A key can be

\* anything but is typically derived from a candidate's id and class. This approach is not always

\* appropriate. For example, two candidate sources might both return different sub-classes of

\* [[com.twitter.product\_mixer.component\_library.model.candidate.BaseTweetCandidate]] resulting in

\* them not being treated as duplicates.

\*/

trait DeduplicationKey[Key] {

def apply(candidate: ItemCandidateWithDetails): Key

}

/\*\*

\* Use candidate id and class to determine duplicates.

\*/

object IdAndClassDuplicationKey extends DeduplicationKey[(String, Class[\_ <: UniversalNoun[Any]])] {

def apply(item: ItemCandidateWithDetails): (String, Class[\_ <: UniversalNoun[Any]]) =

(item.candidate.id.toString, item.candidate.getClass)

}

/\*\*

\* Use candidate id to determine duplicates.

\* This should be used instead of [[IdAndClassDuplicationKey]] in order to deduplicate across

\* different candidate types, such as different implementations of

\* [[com.twitter.product\_mixer.component\_library.model.candidate.BaseTweetCandidate]].

\*/

object IdDuplicationKey extends DeduplicationKey[String] {

def apply(item: ItemCandidateWithDetails): String = item.candidate.id.toString

}