package com.twitter.product\_mixer.core.model.common.identifier

import com.fasterxml.jackson.databind.annotation.JsonSerialize

/\*\*

\* A non-empty immutable stack of [[ComponentIdentifier]]s

\*

\* [[ComponentIdentifierStack]] does not support removing [[ComponentIdentifier]]s,

\* instead a [[ComponentIdentifierStack]] should be used by adding new [[ComponentIdentifier]]s

\* as processing enters a given `Component`, then discarded after.

\* Think of this as similar to a let-scoped variable, where the let-scope is the given component.

\*/

@JsonSerialize(using = classOf[ComponentIdentifierStackSerializer])

class ComponentIdentifierStack private (val componentIdentifiers: List[ComponentIdentifier]) {

/\*\* Make a new [[ComponentIdentifierStack]] with the [[ComponentIdentifier]] added at the top \*/

def push(newComponentIdentifier: ComponentIdentifier): ComponentIdentifierStack =

new ComponentIdentifierStack(newComponentIdentifier :: componentIdentifiers)

/\*\* Make a new [[ComponentIdentifierStack]] with the [[ComponentIdentifier]]s added at the top \*/

def push(newComponentIdentifiers: ComponentIdentifierStack): ComponentIdentifierStack =

new ComponentIdentifierStack(

newComponentIdentifiers.componentIdentifiers ::: componentIdentifiers)

/\*\* Make a new [[ComponentIdentifierStack]] with the [[ComponentIdentifier]]s added at the top \*/

def push(newComponentIdentifiers: Option[ComponentIdentifierStack]): ComponentIdentifierStack = {

newComponentIdentifiers match {

case Some(newComponentIdentifiers) => push(newComponentIdentifiers)

case None => this

}

}

/\*\* Return the top element of the [[ComponentIdentifierStack]] \*/

val peek: ComponentIdentifier = componentIdentifiers.head

/\*\* Return the size of the [[ComponentIdentifierStack]] \*/

def size: Int = componentIdentifiers.length

override def toString: String =

s"ComponentIdentifierStack(componentIdentifiers = $componentIdentifiers)"

override def equals(obj: Any): Boolean = {

obj match {

case componentIdentifierStack: ComponentIdentifierStack

if componentIdentifierStack.eq(this) ||

componentIdentifierStack.componentIdentifiers == componentIdentifiers =>

true

case \_ => false

}

}

}

object ComponentIdentifierStack {

/\*\*

\* Returns a [[ComponentIdentifierStack]] from the given [[ComponentIdentifier]]s,

\* where the top of the stack is the left-most [[ComponentIdentifier]]

\*/

def apply(

componentIdentifier: ComponentIdentifier,

componentIdentifierStack: ComponentIdentifier\*

) =

new ComponentIdentifierStack(componentIdentifier :: componentIdentifierStack.toList)

}