package com.twitter.unified\_user\_actions.enricher.hcache

import com.google.common.cache.Cache

import com.twitter.cache.FutureCache

import com.twitter.cache.guava.GuavaCache

import com.twitter.finagle.stats.NullStatsReceiver

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.util.Future

/\*\*

\* A local cache implementation using GuavaCache.

\* Underneath it uses a customized version of the EvictingCache to 1) deal with Futures, 2) add more stats.

\*/

class LocalCache[K, V](

underlying: Cache[K, Future[V]],

statsReceiver: StatsReceiver = NullStatsReceiver) {

private[this] val cache = new GuavaCache(underlying)

private[this] val evictingCache: FutureCache[K, V] =

ObservedEvictingCache(underlying = cache, statsReceiver = statsReceiver)

def getOrElseUpdate(key: K)(fn: => Future[V]): Future[V] = evictingCache.getOrElseUpdate(key)(fn)

def get(key: K): Option[Future[V]] = evictingCache.get(key)

def evict(key: K, value: Future[V]): Boolean = evictingCache.evict(key, value)

def set(key: K, value: Future[V]): Unit = evictingCache.set(key, value)

def reset(): Unit =

underlying.invalidateAll()

def size: Int = evictingCache.size

}