package com.twitter.servo.cache

import com.twitter.util.Future

/\*\*

\* A cache wrapper that makes the underlying cache transparent to

\* certain keys.

\*/

class KeyFilteringCache[K, V](val underlyingCache: Cache[K, V], keyPredicate: K => Boolean)

extends CacheWrapper[K, V] {

override def get(keys: Seq[K]): Future[KeyValueResult[K, V]] =

underlyingCache.get(keys filter keyPredicate)

override def getWithChecksum(keys: Seq[K]): Future[CsKeyValueResult[K, V]] =

underlyingCache.getWithChecksum(keys filter keyPredicate)

override def add(key: K, value: V) =

if (keyPredicate(key)) {

underlyingCache.add(key, value)

} else {

Future.True

}

override def checkAndSet(key: K, value: V, checksum: Checksum) =

if (keyPredicate(key)) {

underlyingCache.checkAndSet(key, value, checksum)

} else {

Future.True

}

override def set(key: K, value: V) =

if (keyPredicate(key)) {

underlyingCache.set(key, value)

} else {

Future.Done

}

override def replace(key: K, value: V) =

if (keyPredicate(key)) {

underlyingCache.replace(key, value)

} else {

Future.True

}

override def delete(key: K) =

if (keyPredicate(key)) {

underlyingCache.delete(key)

} else {

Future.True

}

}