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

import com.google.inject.Provides

import com.twitter.conversions.DurationOps.\_

import com.twitter.conversions.PercentOps.\_

import com.twitter.finagle.mtls.authentication.ServiceIdentifier

import com.twitter.finagle.thrift.ClientId

import com.twitter.finagle.thriftmux.MethodBuilder

import com.twitter.finatra.mtls.thriftmux.modules.MtlsClient

import com.twitter.inject.Injector

import com.twitter.inject.thrift.modules.ThriftMethodBuilderClientModule

import com.twitter.stitch.tweetypie.TweetyPie

import com.twitter.tweetypie.thriftscala.TweetService

import com.twitter.util.Duration

import javax.inject.Singleton

/\*\*

\* Implementation with reasonable defaults for an idempotent TweetyPie Thrift and Stitch client.

\*

\* Note that the per request and total timeouts are meant to represent a reasonable starting point

\* only. These were selected based on common practice, and should not be assumed to be optimal for

\* any particular use case. If you are interested in further tuning the settings in this module,

\* it is recommended to create local copy for your service.

\*/

object TweetyPieClientModule

extends ThriftMethodBuilderClientModule[

TweetService.ServicePerEndpoint,

TweetService.MethodPerEndpoint

]

with MtlsClient {

override val label: String = "tweetypie"

override val dest: String = "/s/tweetypie/tweetypie"

@Singleton

@Provides

def providesTweetypieStitchClient(tweetService: TweetService.MethodPerEndpoint): TweetyPie =

new TweetyPie(tweetService)

/\*\*

\* TweetyPie client id must be in the form of {service.env} or it will not be treated as an

\* unauthorized client

\*/

override protected def clientId(injector: Injector): ClientId = {

val serviceIdentifier = injector.instance[ServiceIdentifier]

ClientId(s"${serviceIdentifier.service}.${serviceIdentifier.environment}")

}

override protected def configureMethodBuilder(

injector: Injector,

methodBuilder: MethodBuilder

): MethodBuilder =

methodBuilder

.withTimeoutPerRequest(200.milliseconds)

.withTimeoutTotal(400.milliseconds)

.idempotent(1.percent)

override protected def sessionAcquisitionTimeout: Duration = 500.milliseconds

}