package com.twitter.servo.util

/\*\*

\* Provides functions for computing prescribed feature availability based

\* on some runtime condition(s). (e.g. watermark values)

\*/

object Availability {

/\*\*

\* Stay at 100% available down to a high watermark success rate. Then

\* between high and low watermarks, dial down availability to a provided

\* minimum. Never go below this level because we need some requests to

\* track the success rate going back up.

\*

\* NOTE: watermarks and minAvailability must be between 0 and 1.

\*/

def linearlyScaled(

highWaterMark: Double,

lowWaterMark: Double,

minAvailability: Double

): Double => Double = {

require(

highWaterMark >= lowWaterMark && highWaterMark <= 1,

s"highWaterMark ($highWaterMark) must be between lowWaterMark ($lowWaterMark) and 1, inclusive"

)

require(

lowWaterMark >= minAvailability && lowWaterMark <= 1,

s"lowWaterMark ($lowWaterMark) must be between minAvailability ($minAvailability) and 1, inclusive"

)

require(

minAvailability > 0 && minAvailability < 1,

s"minAvailability ($minAvailability) must be between 0 and 1, exclusive"

)

{

case sr if sr >= highWaterMark => 1.0

case sr if sr <= lowWaterMark => minAvailability

case sr =>

val linearFraction = (sr - lowWaterMark) / (highWaterMark - lowWaterMark)

minAvailability + (1.0 - minAvailability) \* linearFraction

}

}

}