```java
//********************************************************************************
//
// File:    Packet.java
// Package: ---
// Unit:    Packet Link
//
//********************************************************************************

import edu.rit.numeric.ListSeries;
import edu.rit.sim.Simulation;
import edu.rit.util.Random;

/**
 * Class Packet provides a packet model in the web simulation. It contains the
 * logic necessary to determine the size of the packet and the amount of time
 * it would take to transmit along a link. It also reports to several instances
 * of ListSeries objects that keep track of the response time of packets based
 * on their size.
 *
 * @author  Alan Kaminsky
 * @author Jimi Ford (jhf3617)
 * @version 5-6-2015
 */
public class Packet
{
    /**
     * size of the packet in bits
     */
    public final int size;

    /**
     * unique identifier across all other packets
     */
    public final int id;

    /**
     * true if this packet is 576 bytes, false otherwise
     */
    public final boolean isLarge;

    // private data member

    private static int idCounter = 0;
    private Simulation sim;
    private double startTime;
    private double finishTime;
    private ListSeries respTimeSeries;
    private ListSeries respTimeLargePackets;
    private ListSeries respTimeSmallPackets;

    private static final int
        SMALL = 40 * Byte.SIZE,
        LARGE = 576 * Byte.SIZE;


    /**
     * Construct a new packet. The packet's response time will be recorded in
     * the ListSeries.
```

```
59      * @param prng a pseudorandom number generator
60      * @param sim the current simulation object
61      * @param series the series to keep track of response times in
62      * @param seriesLargePackets series to keep track of large packet response
63      * times in
64      * @param seriesSmallPackets series to keep track of small packet response
65      * times in
66      */
67     public Packet(Random prng, Simulation sim, ListSeries series,
68             ListSeries seriesLargePackets, ListSeries seriesSmallPackets) {
69         this.id = ++ idCounter;
70         this.sim = sim;
71         this.startTime = sim.time();
72         this.size = prng.nextDouble() < .5 ? SMALL : LARGE;
73         this.isLarge = this.size == LARGE;
74         this.respTimeSeries = series;
75         this.respTimeLargePackets = seriesLargePackets;
76         this.respTimeSmallPackets = seriesSmallPackets;
77     }
78 
79     /**
80      * get the time it would take this packet to transmit along a given link
81      *
82      * @param link the given link to transmit on
83      * @return the time in seconds it would take to transmit on the given link
84      */
85     public double transmitTime(Link link) {
86         if(link.infiniteBitRate) {
87             return 0;
88         }
89         return ((double)size) / link.bitRate;
90     }
91 
92     /**
93      * Mark this request as finished. The request's finish time is set to the
94      * current simulation time. The request's response time is recorded in the
95      * response time series.
96      */
97     public void finish()
98     {
99         finishTime = sim.time();
100        respTimeSeries.add (responseTime());
101        if(isLarge) respTimeLargePackets.add(responseTime());
102        else respTimeSmallPackets.add(responseTime());
103    }
104 
105    /**
106     * Returns this request's response time.
107     *
108     * @return  Response time.
109     */
110    public double responseTime()
111    {
112        return finishTime - startTime;
113    }
114 
115    /**
116     * Returns a string version of this request.
```

```
117      *
118      * @return  String version.
119      */
120     public String toString()
121     {
122         return "Packet " + id;
123     }
124 }
```