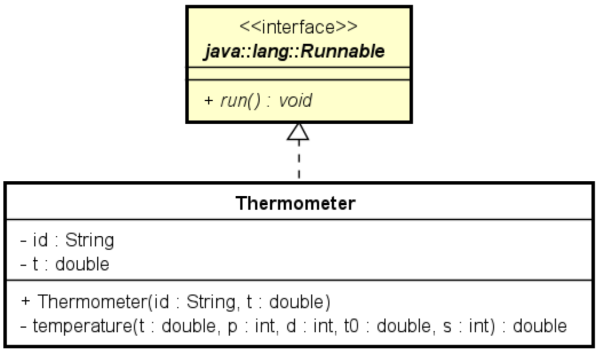
Exercises, Threads SDJ2

Exercise: Simulating the temperature in a room

The purpose for this exercise is to simulate a thermometer (transducer) measuring indoor temperatures. The following method may be used to simulate the temperature in a room (with or without a heater):

```
/**
 * Calculating the temperature measured in one of two locations.
 * This includes a term from a heater (depending on location and
 * heaters power), and a term from an outdoor heat loss.
 * Values are only valid in the outdoor temperature range [-20; 20]
 * and when s, the number of seconds between each measurements are
 * between 4 and 8 seconds.
 *
 * @param t  the last measured temperature
 * @param p  the heaters power {0, 1, 2 or 3} where 0 is turned off,
 *    1 is low, 2 is medium and 3 is high
 * @param d  the distance between heater and measurements {1 or 7}
 *    where 1 is close to the heater and 7 is in the opposite corner
 * @param t0 the outdoor temperature (valid in the range [-20; 20])
 * @param s the number of seconds since last measurement [4; 8]
 * @return the temperature
 */
public double temperature(double t, int p, int d, double t0, int s)
{
  double tMax = Math.min(11 * p + 10, 11 * p + 10 + t0);
  tMax = Math.max(Math.max(t, tMax), t0);
  double heaterTerm = 0;
  if (p > 0)
  {
    double den = Math.max((tMax * (20 - 5 * p) * (d + 5)), 0.1);
    heaterTerm = 30 * s * Math.abs(tMax - t) / den;
  }
  double outdoorTerm = (t - t0) * s / 250.0;
  t = Math.min(Math.max(t - outdoorTerm + heaterTerm, t0), tMax);
  return t;
}
```

Implement a Runnable class Thermometer exactly as shown in the class diagram - with the following notes:


<<interface>>
java::lang::Runnable
+ run() : void
Thermometer
- id : String
- t : double
+ Thermometer(id : String, t : double)
- temperature(t : double, p : int, d : int, t0 : double, s : int) : double

- Copy/paste method temperature as shown and change the visibility to private
- Instance variables id representing the name of the thermometer (e.g. "t1"), and t representing the current temperature.
- A constructor initialising both instance variables
- A run method (from interface Runnable) with an *infinite* loop, in which you
  - Update temperature t calling method temperature. Use the last measured temperature t and d=1, p=0, t0=0 and s=6 (i.e. distance to a heater is 1, heater power is 0, i.e. turned off, outdoor temperature is 0 and number of seconds between each measurement is 6).
  - Print out the temperature t (and the id)
  - Sleep for 6 seconds (6000 milliseconds)

Implement another class with a `main` method, in which you

- Create a `Thermometer` object. Use “t1” for id and 15 for the initial temperature
- Create a thread with the `Thermometer` as argument, and start the thread

Run the application and observe that the temperature slowly drops from 15 towards 0 (over time the indoor temperature drops to the outdoor temperature when there is no heater).

Extra: Change the second argument calling method `temperature` (in the `run` method) to `p=2` (i.e. a heater turned on to power position 2) and observe that the temperature now increases from 15

(Extra: Simulating the temperature in a room – two thermometers)

Modify the previous exercise to include two thermometers in different positions from a heater

Modify class `Thermometer`:

- An extra instance variable, `d`, of type `int`
- The constructor taking also `d` as argument to initialise this one too
- Use the variable `d` as argument when calling `temperature` in the `run` method
- Keep the value p=2 (i.e. a heater’s power position is 2) when calling `temperature`

Modify the `main` method:

- Create two `Thermometer` objects, one with d=1 and one with d=7. Give different ids for the two.
- Create two threads and start both.

Run the application and observe that the temperature raises faster for the thermometer with distance d=1 that the one with distance d=7 (in other words, that the thermometer closest to the heater reacts faster and is less dependent on the outdoor temperature)