Δίκτυα Υπολογιστών II

Στεφανίδης Ιωάννης

AEM: 9587

Source code

Github

```java
/**
 * Αυτή η εφαρμογή χρησιμοποιεί το ITHAKI API που έφτιαξα.Εδώ απλά γίνονται
 * κάποια loops για τα χρονικά διαστήματα που ζητάει η εργασία.
 *
 * Also check out the simple shell script I made for getting codes from Ithaki
 *
 * @see <a href="https://github.com/johnstef99/networks_2/">GitHub</a>
 * */
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileWriter;
import java.io.IOException;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.text.DecimalFormat;
import java.util.Scanner;

import ithaki_api.CAMERAS;
import ithaki_api.ITHAKI;
import ithaki_api.Image;
import ithaki_api.IthakiCopterPacket;
import ithaki_api.Packet;
import ithaki_api.Sound;
import ithaki_api.VehiclePacket;

public class userApplication {
  // GLOBAL
  static int SERVER_PORT;
  static int CLIENT_PORT;
  static String ECHO_CODE;
static String IMG_CODE;
static String SOUND_CODE;
static String VEHICLE_CODE;
static String resultsDir = "../results/session2/";
static DatagramSocket SEND_SOCKET;
static DatagramSocket RECIEVE_SOCKET;
static InetAddress SERVER_ADDRESS;

public static void main(String[] args) throws IOException {
  System.out.println("===========================");
  System.out.println("           ITHAKI          ");
  System.out.println("===========================\n");

  getCodes();
  ITHAKI ithaki = new ITHAKI(SERVER_PORT, CLIENT_PORT, ECHO_CODE,
      IMG_CODE, SOUND_CODE, VEHICLE_CODE);

  echo(ithaki, 4 * 60, true);
  echo(ithaki, 4 * 60, false);
  images(ithaki);
  temperatures(ithaki);
  sound(ithaki);
  System.out.println("Open ithaki copter jar file and press a key to "
      +"continue..");
  System.in.read();
  telemetry(ithaki, 60);
  vehicle(ithaki, 4 * 60);
}

/**
 * Getting ithaki codes from file named "codes"
 */
private static void getCodes() {
  File codesFile = new File("./codes");
  try {
    Scanner reader = new Scanner(codesFile);
    int line = 0;
    while (reader.hasNextLine()) {
      switch (line) {
        case 1:
          CLIENT_PORT = Integer.parseInt(reader.nextLine());
          System.out.println("client port: " + CLIENT_PORT);
          break;
        case 3:
          SERVER_PORT = Integer.parseInt(reader.nextLine());
          System.out.println("server port: " + SERVER_PORT);
          break;
        case 4:
          ECHO_CODE = reader.nextLine();
          System.out.println("ECHO_CODE: " + ECHO_CODE);
          break;
        case 5:
          IMG_CODE = reader.nextLine();
          System.out.println("IMG_CODE: " + IMG_CODE);
          break;
        case 6:
          SOUND_CODE = reader.nextLine();
          System.out.println("SOUND_CODE: " + SOUND_CODE);
          break;
        case 7:
          System.out.println("COPTER_CODE: " + reader.nextLine());
          break;
        case 8:
          VEHICLE_CODE = reader.nextLine();
          System.out.println("VEHICLE_CODE: " + VEHICLE_CODE);
          break;
        default:
          reader.nextLine();
          break;
      }
      line++;
    }
    reader.close();
  } catch (FileNotFoundException e) {
    e.printStackTrace();
  }
}

/**
 * Gets the sound samples from ithaki
 *
 * @param ithaki
 */
private static void sound(ITHAKI ithaki) {
  Sound song1_aq = ithaki.getSound(300, 3, true);
  song1_aq.writeToFile(resultsDir + "song3");
  song1_aq.play();

  Sound song2_aq = ithaki.getSound(300, 4, true);
  song2_aq.writeToFile(resultsDir + "song4");
  song2_aq.play();
  Sound song1 = ithaki.getSound(300, 3, false);
  song1.writeToFile(resultsDir + "song3");
  song1.play();

  Sound gen = ithaki.getSound(300, 0, false);
  gen.writeToFile(resultsDir + "generator");
}

/**
 * Get echo packets
 *
 * @param ithaki    Ithaki's API instant
 * @param runTime   How many second to get packets
 * @param withDelay Whether to have delay or not
 */
private static void echo(ITHAKI ithaki, int runTime, boolean withDelay) {
  String delay = "_NO_DELAY";
  if (withDelay)
    delay = "_DELAY";
  File echo_packets_file = new File(resultsDir + ECHO_CODE + delay + ".txt");
  try {
    if (echo_packets_file.createNewFile()) {
      System.out.println("File created: " + echo_packets_file.getName());
    } else {
      System.out.println(echo_packets_file.getName() + " already exist");
    }
    FileWriter echo_writer = new FileWriter(echo_packets_file, false);
    double startTime = System.currentTimeMillis();
    System.out.println("Progress\tPacket");
    DecimalFormat per = new DecimalFormat("#0.00");
    for (double now = System.currentTimeMillis();
        now < startTime + runTime * 1000;
        now = System.currentTimeMillis()) {
      Packet aPacket = ithaki.getPacket(withDelay, -1);
      double progress = ((now - startTime) / (runTime * 1000)) * 100;
      if (aPacket != null) {
        System.out.println(per.format(progress) + "%\t" + aPacket.toString());
        echo_writer.write(String.valueOf(aPacket.responseTime) + "\n");
      }
    }
    System.out.println("100%\tGetting echo packets finished");
    System.out.println("Exported to file: " + echo_packets_file.getName());
    echo_writer.close();
  } catch (IOException e) {
    System.out.println("Error creating " + echo_packets_file.getName());
    e.printStackTrace();
  }
}

/**
 * Gets images from both cameras
 *
 * @param ithaki
 */
private static void images(ITHAKI ithaki) {
  Image e1 = ithaki.getImage(CAMERAS.FIX);
  e1.writeToFile(resultsDir + "E1.jpg");
  Image e2 = ithaki.getImage(CAMERAS.PTZ);
  e2.writeToFile(resultsDir + "E2.jpg");
}

/**
 * Get temperature from the only working sensor 0
 *
 * @param ithaki
 */
private static void temperatures(ITHAKI ithaki) {
  Packet temPacket = ithaki.getPacket(true, 0);
  File temp_file = new File(resultsDir + ECHO_CODE + "_TEMP.txt");
  try {
    if (temp_file.createNewFile()) {
      System.out.println("File created: " + temp_file.getName());
    } else {
      System.out.println(temp_file.getName() + " already exist");
    }
    FileWriter temp_writer = new FileWriter(temp_file, false);
    temp_writer.write(temPacket.toString() + "\n");
    temp_writer.close();
  } catch (IOException e) {
    System.out.println("Error creating " + temp_file.getName());
    e.printStackTrace();
  }
}

/**
 * Get telemetry packets
 *
 * @param ithaki  Ithaki's API instant
 * @param runTime How many second to get packets
 */
private static void telemetry(ITHAKI ithaki, int runTime) {
  File copter_packets_file = new File(resultsDir + "ITHAKICOPTER.txt");
```

```java
  try {
    if (copter_packets_file.createNewFile()) {
      System.out.println("File created: " + copter_packets_file.getName());
    } else {
      System.out.println(copter_packets_file.getName() + " already exist");
    }
    FileWriter echo_writer = new FileWriter(copter_packets_file, false);
    double startTime = System.currentTimeMillis();
    System.out.println("Progress\tPacket");
    DecimalFormat per = new DecimalFormat("#0.00");
    for (double now = System.currentTimeMillis();
        now < startTime + runTime * 1000;
        now = System.currentTimeMillis()) {
      IthakiCopterPacket aPacket = ithaki.getTelemetry();
      double progress = ((now - startTime) / (runTime * 1000)) * 100;
      System.out.println(per.format(progress) + "%\t" + aPacket.toString());
      echo_writer.write(aPacket.toJson() + "\n");
    }
    System.out.println("100%\tGetting telemetry packets finished");
    System.out.println("Exported to file: " + copter_packets_file.getName());
    echo_writer.close();
  } catch (IOException e) {
    System.out.println("Error creating " + copter_packets_file.getName());
    e.printStackTrace();
  }
}

/**
 * Get vehicle packets
 *
 * @param ithaki  Ithaki's API instant
 * @param runTime How many second to get packets
 */
private static void vehicle(ITHAKI ithaki, int runTime) {
  File vehicle_packets_file = new File(resultsDir + "V" + VEHICLE_CODE
      + ".json");
  try {
    if (vehicle_packets_file.createNewFile()) {
      System.out.println("File created: " + vehicle_packets_file.getName());
    } else {
      System.out.println(vehicle_packets_file.getName() + " already exist");
    }
    FileWriter vehicle_writer = new FileWriter(vehicle_packets_file, false);
    vehicle_writer.write("[\n");
    double startTime = System.currentTimeMillis();
    System.out.println("Progress\tPacket");
```

```
      DecimalFormat per = new DecimalFormat("#0.00");
      for (double now = System.currentTimeMillis();
          now < startTime + runTime * 1000;
          now = System.currentTimeMillis()) {
        VehiclePacket aPacket = ithaki.getVehiclePacket();
        double progress = ((now - startTime) / (runTime * 1000)) * 100;
        System.out.println(per.format(progress) + "%\t" + aPacket.toString());
        vehicle_writer.write(String.valueOf(aPacket.toJson()) + ",\n");
      }
      vehicle_writer.write("]");
      vehicle_writer.close();
      System.out.println("100%\tGetting vehicle packets finished");
      System.out.println("Exported to file: " + vehicle_packets_file.getName());
      vehicle_writer.close();
    } catch (IOException e) {
      System.out.println("Error creating " + vehicle_packets_file.getName());
      e.printStackTrace();
    }

  }
}
```