#6 Class Diagram

Disadur dari soal tahun lalu, oleh David Adiutama

Tugas 1 - Snake: SourceCode to ClassDiagram

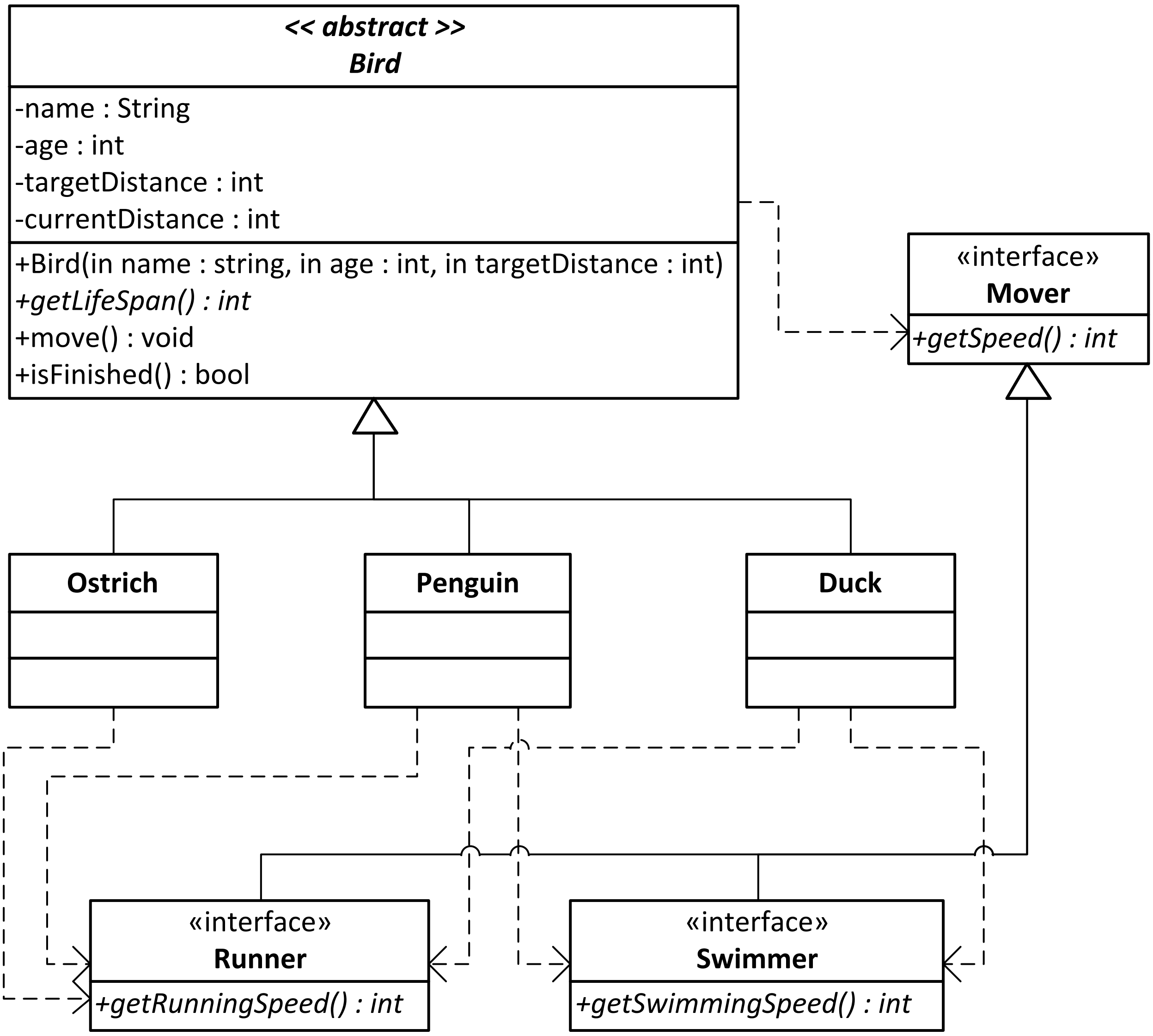
```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.util.*;

class Snake extends JFrame implements KeyListener, Runnable {

    JPanel p1, p2;
    JButton[] lb = new JButton[200];
    JButton bonusfood;
    JTextArea t;
    int x = 500, y = 250, gu = 2, directionx = 1, directiony = 0, speed = 50,
        difference = 0, oldx, oldy, score = 0;
    int[] lbx = new int[300];
    int[] lby = new int[300];
    Point[] lbp = new Point[300];
    Point bfp = new Point();
    Thread myt;
    boolean food = false, runl = false, runr = true, runu = true, rund = true, bonusflag = true;
    Random r = new Random();
    JMenuBar mymbar;
    JMenu game, help, level;

    public void initializeValues() {
        gu = 3;
        lbx[0] = 100;
        lby[0] = 150;
        directionx = 10;
        directiony = 0;
        difference = 0;
        score = 0;
        food = false;
        runl = false;
        runr = true;
        runu = true;
        rund = true;
        bonusflag = true;
    }

    Snake() {
        super("Snake");
        setSize(500, 330);
        //Create Menue bar with functions
        creatbar();
        //initialize all variables
        initializeValues();
        // Start of UI design
        p1 = new JPanel();
        p2 = new JPanel();
        // t will view the score
        t = new JTextArea("Score ==>" + score);
        t.setEnabled(false);
        t.setBackground(Color.BLACK);
        // snake have to eat bonousfood to growup
        bonusfood = new JButton();
        bonusfood.setEnabled(false);
        // will make first snake
        createFirstSnake();

        p1.setLayout(null);
        p2.setLayout(new GridLayout(0, 1));
        p1.setBounds(0, 0, x, y);
        p1.setBackground(Color.blue);
        p2.setBounds(0, y, x, 30);
        p2.setBackground(Color.RED);

        p2.add(t); // will contain score board
        // end of UI design
        getContentPane().setLayout(null);
        getContentPane().add(p1);
        getContentPane().add(p2);

        show();
        setDefaultCloseOperation(EXIT_ON_CLOSE);

        addKeyListener(this);
        // start thread
        myt = new Thread(this);
        myt.start(); // go to run() method
    }

    public void createFirstSnake() {
        // Initially the snake has small length 3
        for (int i = 0; i < 3; i++) {
            lb[i] = new JButton("lb" + i);
            lb[i].setEnabled(false);
            p1.add(lb[i]);
            lb[i].setBounds(lbx[i], lby[i], 10, 10);
            lbx[i + 1] = lbx[i] - 10;
            lby[i + 1] = lby[i];
        }
    }

    public void creatbar() {
        mymbar = new JMenuBar();

        game = new JMenu("Game");

        JMenuItem newgame = new JMenuItem("New Game");
        JMenuItem exit = new JMenuItem("Exit");

        newgame.addActionListener(
                new ActionListener() {

                    public void actionPerformed(ActionEvent e) {
                        reset();
                    }
                });

        exit.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {
                System.exit(0);
            }
        });

        game.add(newgame);
        game.addSeparator();
        game.add(exit);

        mymbar.add(game);

        level = new JMenu("Level");

        mymbar.add(level);

        setJMenuBar(mymbar);
    }

    void reset() {
        initializeValues();
        p1.removeAll();

        myt.stop();

        createFirstSnake();
        t.setText("Score==>" + score);

        myt = new Thread(this);
        myt.start();
    }

    void growup() {
        lb[gu] = new JButton();
        lb[gu].setEnabled(false);
        p1.add(lb[gu]);

        int a = 10 + (10 * r.nextInt(48));
        int b = 10 + (10 * r.nextInt(23));

        lbx[gu] = a;
        lby[gu] = b;
        lb[gu].setBounds(a, b, 10, 10);

        gu++;
    }
    // this method contains the logic to move the snake. player will define the derection
    // this method just forward the snake to the right derection which deriction is pressed
    // by the player.
    void moveForward() {
        for (int i = 0; i < gu; i++) {
            lbp[i] = lb[i].getLocation();
        }

        lbx[0] += directionx;
        lby[0] += directiony;
        lb[0].setBounds(lbx[0], lby[0], 10, 10);

        for (int i = 1; i < gu; i++) {
            lb[i].setLocation(lbp[i - 1]);
        }

        if (lbx[0] == x) {
            lbx[0] = 10;
        } else if (lbx[0] == 0) {
            lbx[0] = x - 10;
        } else if (lby[0] == y) {
            lby[0] = 10;
        } else if (lby[0] == 0) {
            lby[0] = y - 10;
        }

        if (lbx[0] == lbx[gu - 1] && lby[0] == lby[gu - 1]) {
            food = false;
            score += 5;
            t.setText("Score==>" + score);
            if (score % 50 == 0 && bonusflag == true) {
                p1.add(bonusfood);
                bonusfood.setBounds((10 * r.nextInt(50)), (10 * r.nextInt(25)), 15, 15);
                bfp = bonusfood.getLocation();
                bonusflag = false;
            }
        }

        if (bonusflag == false) {
            if (bfp.x <= lbx[0] && bfp.y <= lby[0] && bfp.x + 10 >= lbx[0] && bfp.y + 10 >= lby[0]) {
                p1.remove(bonusfood);
                score += 100;
                t.setText("Score ==>" + score);
                bonusflag = true;
            }
        }

        if (food == false) {
            growup();
            food = true;
        } else {
            lb[gu - 1].setBounds(lbx[gu - 1], lby[gu - 1], 10, 10);
        }

        for (int i = 1; i < gu; i++) {
            if (lbp[0] == lbp[i]) {
                t.setText("GAME OVER     " + score);
                try {
                    myt.join();
                } catch (InterruptedException ie) {
                }
                break;
            }
        }
        p1.repaint();
        show();
    }

    public void keyPressed(KeyEvent e) {
        // snake move to left when player pressed left arrow
        if (runl == true && e.getKeyCode() == 37) {
            directionx = -10; // means snake move right to left by 10pixels
            directiony = 0;
            runr = false;     // run right(runr) means snake cant move from left to right
            runu = true;      // run up   (runu) means snake can move from down to up
            rund = true;      // run down (run down) means snake can move from up to down
        }
        // snake move to up when player pressed up arrow
        if (runu == true && e.getKeyCode() == 38) {
            directionx = 0;
            directiony = -10; // means snake move from down to up by 10 pixel
            rund = false;     // run down (run down) means snake can move from up to down
            runr = true;      // run right(runr) means snake can move from left to right
            runl = true;      // run left (runl) means snake can move from right to left
        }
        // snake move to right when player pressed right arrow
        if (runr == true && e.getKeyCode() == 39) {
            directionx = +10; // means snake move from left to right by 10 pixel
            directiony = 0;
            runl = false;
            runu = true;
            rund = true;
        }
        // snake move to down when player pressed down arrow
        if (rund == true && e.getKeyCode() == 40) {
            directionx = 0;
            directiony = +10; // means snake move from left to right by 10 pixel
            runu = false;
            runr = true;
            runl = true;
        }
    }

    public void keyReleased(KeyEvent e) {
    }

    public void keyTyped(KeyEvent e) {
    }

    public void run() {
        for (;;) {
            // Move the snake move forword. In the start of the game snake move left to right,
            // if player press up, down, right or left arrow snake change its direction according to
            // pressed arrow
            moveForward();
            try {
                Thread.sleep(speed);
            } catch (InterruptedException ie) {
            }
        }
    }
}
```

P0601xxyyy.png

Pada tugas ini, Anda diminta untuk membuat kelas diagram dari source code diatas (Snake Game with GUI). Anda dapat mencoba menjalankan programnya dan memainkan, dan pelajarilah source code nya.
Kelas diagram yang dibuat dilengkapi dengan visibility masing-masing atribut & methodnya. Tambahkan pula kelas-kelas lain beserta method yang digunakannya.

Anda diminta untuk membuatnya dengan tool Astah, lalu export juga sebagai file PNG.

Tugas 2 - BirdRace: ClassDiagram to SourceCode



Implementasi method `getLifeSpan` dan `getRunning/SwimmingSpeed` pada masing-masing inheritance dari kelas Bird, mengembalikan nilai berdasarkan tabel berikut:

	Ostrich	Penguin	Duck
LifeSpan	40 years	20 years	10 years
RunningSpeed	65 kph	4 kph	8 kph
SwimmingSpeed	-	10 kph	3 kph

sedangkan method `getSpeed` dari masing-masing inheritance kelas Bird berupa perhitungan rata-rata dari penjumlahan RunningSpeed & SwimmingSpeed

Kelas abstrak **Bird** yang memiliki atribut sebagai berikut:

+ `age:int` → umur unggas saat ini
+ `targetDistance:int` → jarak yang harus ditempuh
+ `currentDistance:int` → jarak yang sudah ditempuh sampai saat ini

dan method-method sebagai berikut:

+ `isFinished():boolean` → mengembalikan status apakah jarak yang sudah ditempuh sudah melampaui jarak yang harus ditempuh.

+ `move():void` → menambah atribut `currentDistance` dengan `getSpeed` relatif terhadap umur dan masa hidup berdasarkan grafik sinusoidal dengan rumus:

$$currentDistance = currentDistance + speed \times \sin\left(\frac{\pi \times age}{lifeSpan}\right)$$

Buatlah kelas tester bernama **BirdRace.java** yang memiliki
array of Bird dengan 3 objek
dengan jarak lintasan yang harus ditempuh masing-masing 250km dan dengan spesifikasi berikut:

Tipe objek	**Name**	**Age**
Ostrich	Ostrich20	20
Penguin	Penguin10	10
Duck	Duck5	5

Spesifikasi output

Tampilkan jarak masing-masing pada setiap jamnya;
bila sudah mencapai titik akhir, tampilkan juga posisi peringkatnya.

Contoh sebagian output:

```
Hour 1
Ostrich20 at distance 65 km
Penguin10 at distance 7 km
Duck5 at distance 27 km

Hour 2
Ostrich20 at distance 130 km
Penguin10 at distance 14 km
Duck5 at distance 10 km
...
Hour 4
Ostrich20 finished at 1
Penguin10 at distance 28 km
Duck5 at distance 108 km
...
Hour 36
Penguin10 finished at 2
Duck5 at distance 180 km
...
Hour 50
Duck5 finished at 3
```

P0602xxyyy.jar – BirdRace (Mover.java, Runner.java, Swimmer.java, Bird.java, Ostrich.java, Penguin.java, Duck.java, BirdRace.java)

Kumpulkan file T06Bxxyyy.jar yang mengandung kode program javanya.
(jangan lupa untuk menghapus nilai "Exclude From JAR File" di Project Properties -> Build -> Packaging.)