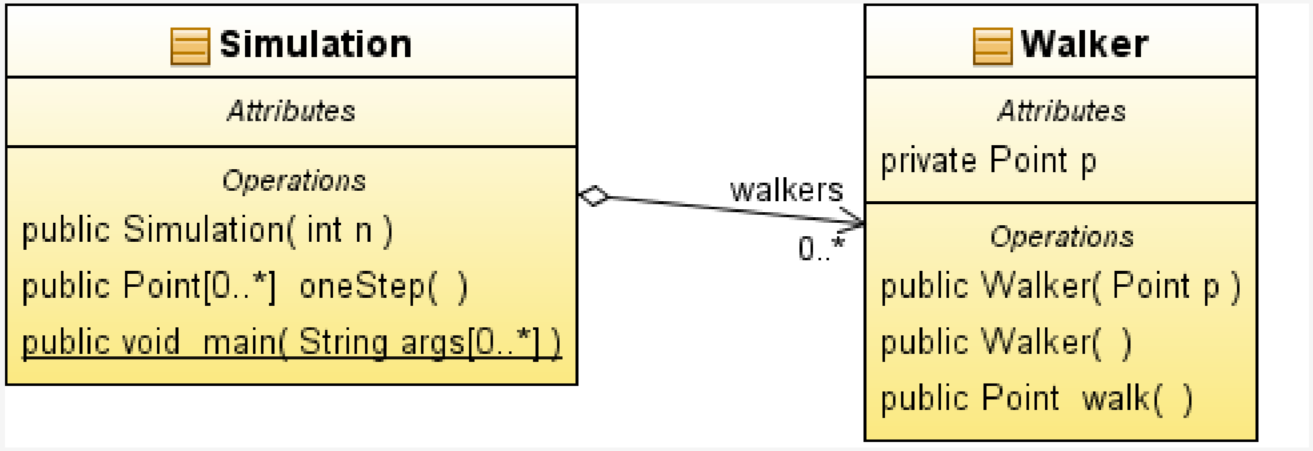
例：簡単な酔歩シミュレーション

シミュレーションの簡単な例

- GUI無しのシミュレーションを作る
- GUIを作る
  - パラメタを設定する
  - デモンストレーションをする

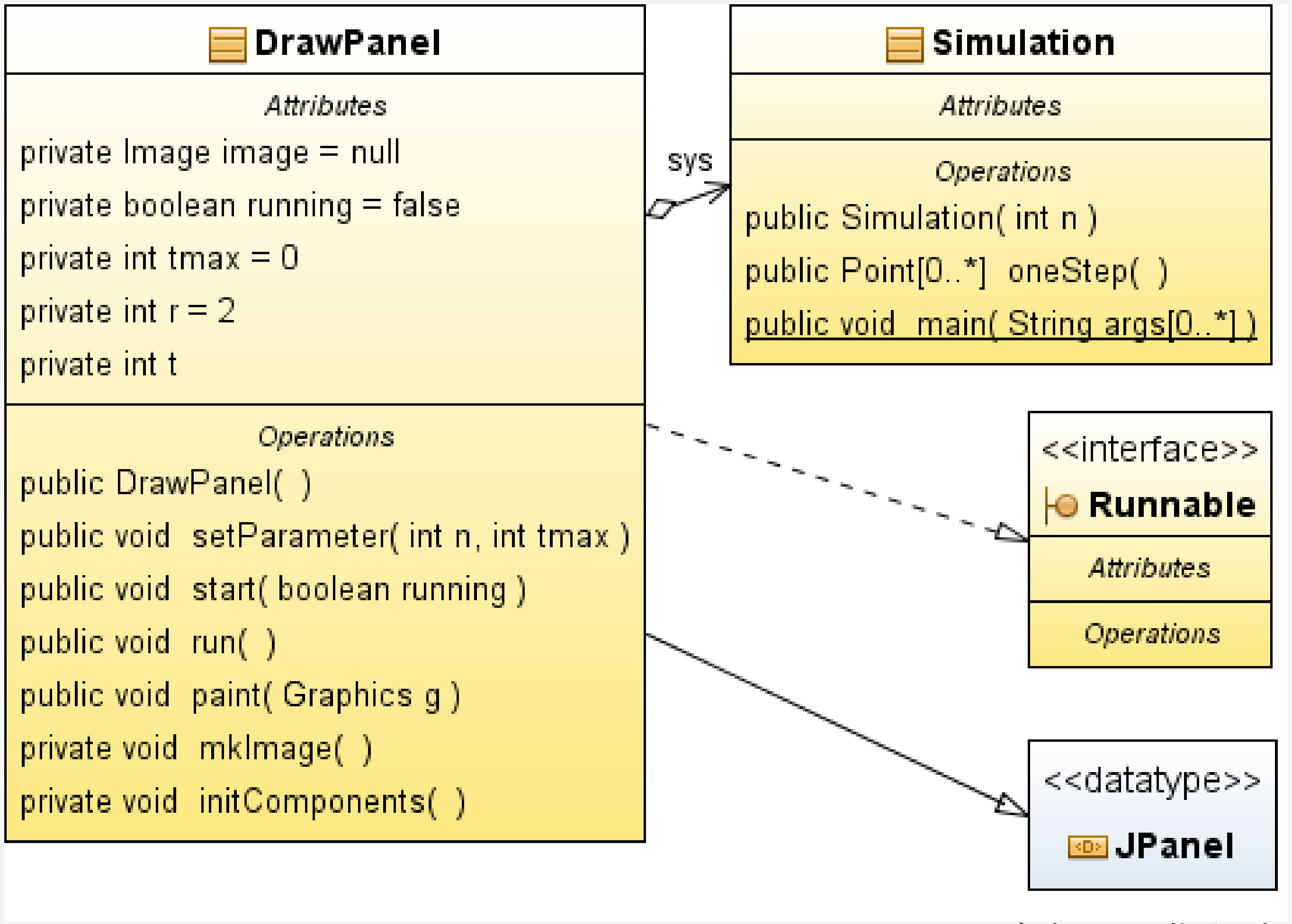
簡単な二次元酔歩

- Walkerは二次元面内で４方向に等確率で移動
    - メソッドmoveで移動し、新しい位置を返す
- Simulationクラス
    - 多数のWalkerを同時に移動
    - メソッドoneStepは一時間ステップ進め、Walkerの新しい位置のリストを返す


Simulation
Attributes
Operations
public Simulation(int n)
public Point[0..*] oneStep( )
public void main(String args[0..*])
walkers
0..*
Walker
Attributes
private Point p
Operations
public Walker(Point p)
public Walker( )
public Point walk( )

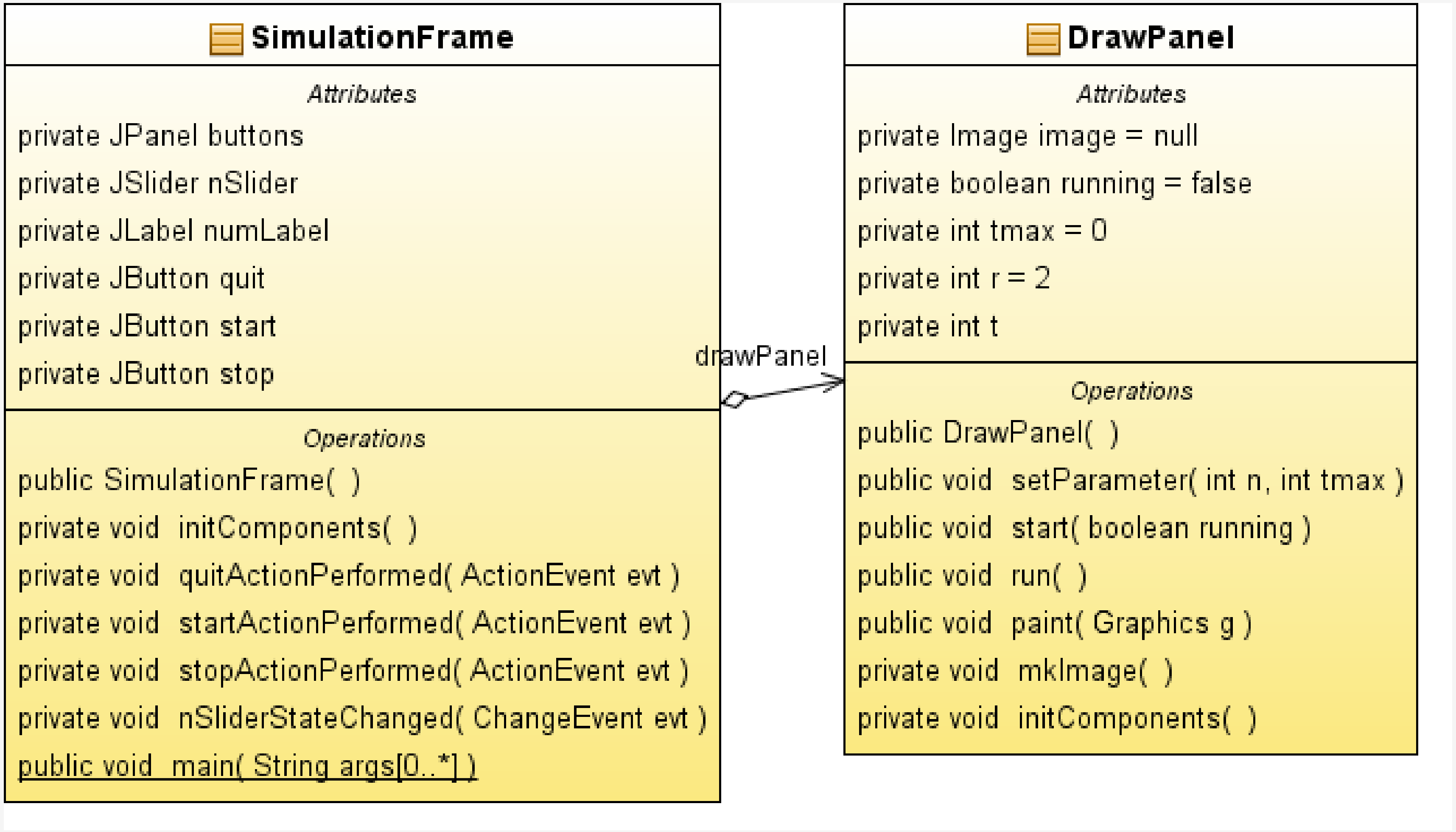
動作を表示するパネル

- Runnableインターフェイスを付ける
    - スレッドとして動作
    - スレッドからの駆動はrunメソッド
- 描画イメージを作る：mkImage
    - イメージ初期化
    - Simulation.oneStepを呼び、位置を取得
    - 位置を表示


DrawPanel
Attributes
private Image image = null
private boolean running = false
private int tmax = 0
private int r = 2
private int t
Operations
public DrawPanel( )
public void setParameter(int n, int tmax)
public void start(boolean running)
public void run( )
public void paint(Graphics g)
private void mkImage( )
private void initComponents( )
sys
Simulation
Attributes
Operations
public Simulation(int n)
public Point[0..*] oneStep( )
public void main(String args[0..*])
<<interface>>
Runnable
Attributes
Operations
<<datatype>>
JPanel

全体構成

- SimulationFrame
  - ボタン（開始、停止、終了）
  - Walker数設定
  - DrawPanelをスレッドで起動


SimulationFrame
Attributes
private JPanel buttons
private JSlider nSlider
private JLabel numLabel
private JButton quit
private JButton start
private JButton stop
Operations
public SimulationFrame( )
private void initComponents( )
private void quitActionPerformed(ActionEvent evt)
private void startActionPerformed(ActionEvent evt)
private void stopActionPerformed(ActionEvent evt)
private void nSliderStateChanged(ChangeEvent evt)
public void main(String args[0..*])
drawPanel
DrawPanel
Attributes
private Image image = null
private boolean running = false
private int tmax = 0
private int r = 2
private int t
Operations
public DrawPanel( )
public void setParameter(int n, int tmax)
public void start(boolean running)
public void run( )
public void paint(Graphics g)
private void mkImage( )
private void initComponents( )

```java
/**
 * Walkerのクラス
 * @author tadaki
 */
package model;

import java.awt.Point;

public class Walker {

    private Point p;//Walkerの位置

    public Walker(Point p) {
        this.p = p;
    }

    public Walker() {
        p = new Point(0, 0);
    }

    /**
     * 一時間ステップの移動
     * @return 新しい位置
     */
    public Point walk() {
        /** 4方向に等確率で移動する **/
        int r = (int) (4 * Math.random());
        int x = 2 * (r % 2) - 1;
        int y = 2 * (r / 2) - 1;
        x += p.x;
        y += p.y;
        p.move(x, y);
        return new Point(p);
    }
}
```

```java
/**
 * 二次元酔歩モデルのシミュレーション
 * @author tadaki
 */
package model;

import java.awt.Point;
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

public class Simulation {
    private List<Walker> walkers=null;//Walkerのリスト

    public Simulation(int n) {
        walkers = Collections.synchronizedList(new ArrayList<Walker>());
        /** Walkerを初期化 */
        for(int i=0;i<n;i++){
            walkers.add(new Walker());
        }
    }

    /**
     * 一時間ステップの動作
     * @return 更新したWalkerの位置の一覧
     */
    public List<Point> oneStep(){
        List<Point> pList =
                Collections.synchronizedList(new ArrayList<Point>());
        for(Walker w:walkers){
            Point p = w.walk();
            pList.add(p);
        }
        return pList;
    }


    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        Simulation sys=new Simulation(100);
        for(int i=0;i<100;i++){
            sys.oneStep();
        }
        List<Point> pList = sys.oneStep();
        for(Point p:pList){
            System.out.print(p.x);
            System.out.print(" ");
            System.out.println(p.y);
        }
    }

}
```

```java
/*
 * DrawPanel.java
 * 酔歩シミュレーションの画面表示
 * Created on 2010/12/17, 9:19:40
 * @author tadaki
 */
package gui;

import java.awt.Color;
import java.awt.Dimension;
import java.awt.Graphics;
import java.awt.Image;
import java.awt.Point;
import java.util.List;

public class DrawPanel extends javax.swing.JPanel implements Runnable {

    private Image image = null;
    private volatile boolean running = false;
    private model.Simulation sys = null;
    private int tmax = 0;
    private int r = 2;
    private int t;

    /** Creates new form DrawPanel */
    public DrawPanel() {
        initComponents();
    }

    /**
     * 酔歩シミュレーションの初期化
     * @param n Walker数
     * @param tmax 時間上限
     */
    public void setParameter(int n, int tmax) {
        this.tmax = tmax;
        sys = new model.Simulation(n);
        running = false;
        t = 0;
    }

    public void start(boolean running) {
        this.running = running;
    }
    public void run() {
        while (running) {
            mkImage();
            repaint();
            if (t > tmax) {
                running = false;
            }
            try {
                Thread.sleep(100);
            } catch (InterruptedException e) {
            }
        }
    }

    @Override
    public void paint(Graphics g) {
        if (image == null) {
            return;
        }
        g.drawImage(image, 0, 0, this);
    }

    /** 描画イメージ作成 **/
    private void mkImage() {
        if (sys == null) {
            return;
        }
        Dimension dimension = getSize();
        image = createImage(dimension.width, dimension.height);
        Graphics g = image.getGraphics();
        g.setColor(getBackground());
        g.fillRect(0, 0, dimension.width, dimension.height);
        g.setClip(0, 0, dimension.width, dimension.height);
        g.translate(dimension.width / 2, dimension.height / 2);

        List<Point> pList = sys.oneStep();

        g.setColor(Color.red);
        for (Point p : pList) {
            g.fillOval(p.x - r, p.y - r, 2 * r, 2 * r);
        }
        t++;
    }
    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {
    // 省略
    }// </editor-fold>//GEN-END:initComponents
    // Variables declaration - do not modify//GEN-BEGIN:variables
    // End of variables declaration//GEN-END:variables
}
```

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

/*
 * SimulationFrame.java
 *
 * Created on 2010/12/17, 9:16:41
 */

package gui;

/**
 *
 * @author tadaki
 */
public class SimulationFrame extends javax.swing.JFrame {

    /** Creates new form SimulationFrame */
    public SimulationFrame() {
        initComponents();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {
    // 省略
    }// </editor-fold>//GEN-END:initComponents

    private void quitActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_quitActionPerformed
        System.exit(0);
    }//GEN-LAST:event_quitActionPerformed

    private void startActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_startActionPerformed
        int n = nSlider.getValue();
        int t= 2*drawPanel.getSize().width;
        drawPanel.setParameter(n,t);
        drawPanel.start(true);
        new Thread(drawPanel).start();
    }//GEN-LAST:event_startActionPerformed

    private void stopActionPerformed(java.awt.event.ActionEvent evt)
 {//GEN-FIRST:event_stopActionPerformed
        drawPanel.start(false);
    }//GEN-LAST:event_stopActionPerformed

    private void nSliderStateChanged(javax.swing.event.ChangeEvent evt)
 {//GEN-FIRST:event_nSliderStateChanged
        int n = nSlider.getValue();
        numLabel.setText("# "+String.valueOf(n));
    }//GEN-LAST:event_nSliderStateChanged

    /**
    * @param args the command line arguments
    */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new SimulationFrame().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JPanel buttons;
    private gui.DrawPanel drawPanel;
    private javax.swing.JSlider nSlider;
    private javax.swing.JLabel numLabel;
    private javax.swing.JButton quit;
    private javax.swing.JButton start;
    private javax.swing.JButton stop;
    // End of variables declaration//GEN-END:variables

 }
```