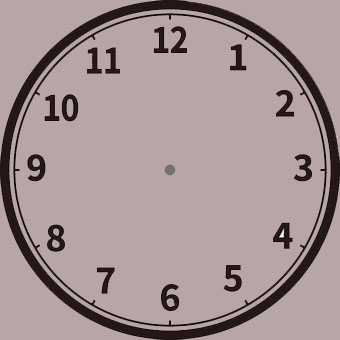
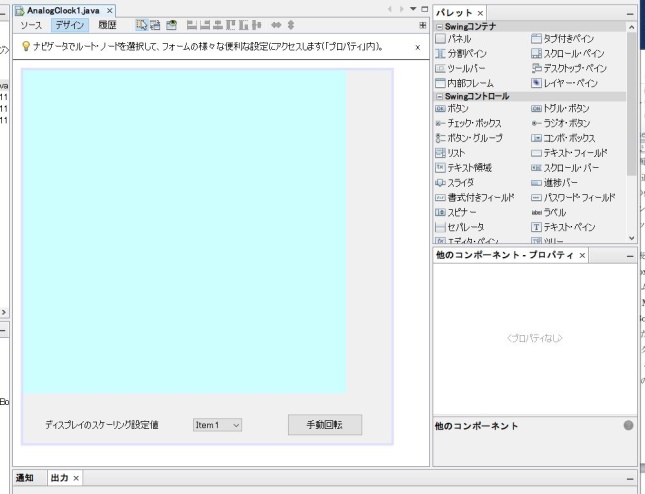
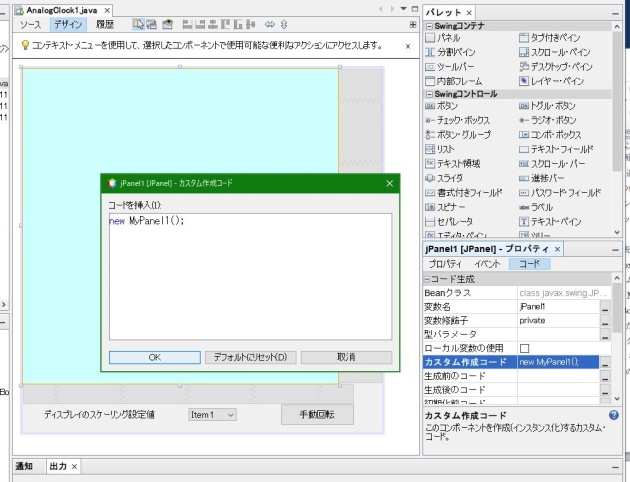
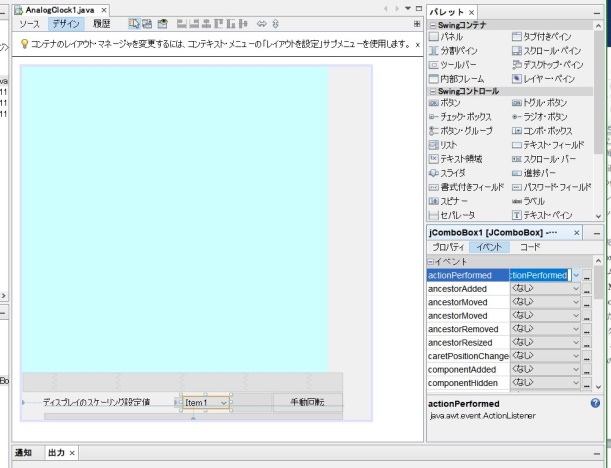
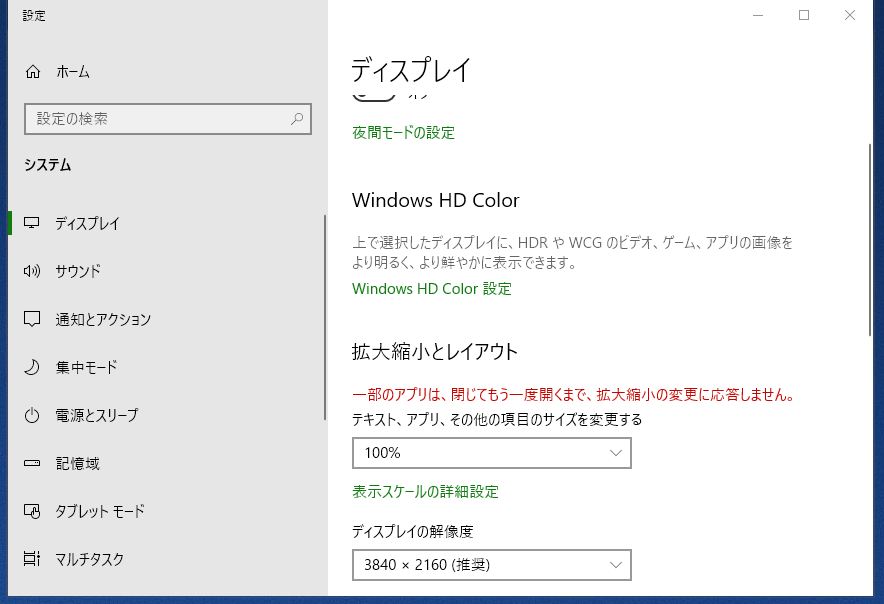
# アナログ時計の作り方

1. 針と文字盤の画像の準備  
   500x500ドットのサイズで秒針、長針、短針の画像を用意する。針以外の背景部分はどれも透明属性にする。そのため画像の形式はPNG形式にする。背景部部を透明にしないと3種類の針を重ねて表示させることができない。  
   具体的には、画像の中心位置(250x250)に針の回転中心となる部分を配置する。また、同じサイズで文字盤画像も用意する。中心位置に色を変えた点を打っておく。こちらは背景色を透明にする必要はない。(透明でも構わない)　文字盤の中心は正確に250x250の位置に来るように調整する。  
   長針と秒針も用意する。
2. Netbeansで自分のプロジェクトを用意し、「ファイルの新規作成」でAnalogClock1というファイル名で新規作成。
3. デザイン画面でフレームのサイズを大きくしておく(600x600ドット程度以上)。
4. JPanelを追加する。サイズを500x500ドットにする。パネルを識別しやすいように、水色にする。
5. jPanel1の位置を調整し、フレームの左上角とパネルの左上角を一致させる。
6. Swingコントロールからボタンを一つとコンボボックスを一つパネルの下あたりに配置する。
7. コンボボックスの左に、ラベルを配置し、「ディスプレイのスケーリング設置値」という文字に設定する。
8. ボタンの表面の文字列は「手動回転」にする。
9. jComboBox1を選択し、右側のウィンドウの「コード」を選択する。
10. 「カスタム生成コード」の右端の・が3つ並んだボタンをクリック、「コードを挿入...」ダイアログに、new MyPanel1(); と入力し[OK]をクリック。
11. jComboBox1を選択し、プロパティの右の「イベント」タブを選択、actionPerformedの右側の・が3つ並んだボタンをクリックし、jComboBox1ActionPerformedが表示されるダイアログで、[追加]ボタンをクリックする。
12. jButton1をダブルクリックする。ソース画面に切り替わるので、必要なコードを入力する。
13. この資料のプログラム通りに入力すると、パネル内に表示される時計が中央から少しずれて表示される。この位置調整はvoid createAdjustArray( )メソッドの中の配列ajVal[ ]に入れる値によって調整される。この配列はクラス AdjustValue のインスタンスを入れてあり、Windowsのディスプレイ設定のスケール値に合わせて値を調整する。また、フレームに配置するJPanelの位置によって時計の表示がずれるという問題および針画像の回転の回転中心がずれる問題を修正するようになっている。この問題は、スケールの設定値(100%、125%、150%、200%等)によって変化するため、それぞれの値に合わせた値をJComboBoxの操作で切り替えるようにしている。
14. 時計の回転の角度計算方法は、各自考えてみること。アンダーラインがついていて空欄になっている部分に適切なプログラムを入力する。

//参考プログラム。　針の回転角度計算部分は空欄になっている。まずは自分で考えてみること。

import java.awt.Graphics;

import java.awt.Graphics2D;

import java.awt.Image;

import java.awt.Point;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.MouseEvent;

import java.awt.event.MouseListener;

import java.awt.event.MouseMotionListener;

import java.awt.geom.AffineTransform;

import java.awt.image.BufferedImage;

import java.io.IOException;

import java.io.InputStream;

import java.util.Calendar;

import java.util.TimeZone;

import javax.imageio.ImageIO;

import javax.swing.JPanel;

public class AnalogClock1 extends javax.swing.JFrame implements ActionListener{

BufferedImage[] bufImgChar = new BufferedImage[3];//針画像用Imageを入れるための配列

BufferedImage bufferedImageBack=null;//文字盤用Image

InputStream inputStream=null; //画像ファイル読み込み用

long cnt=0;

int posx=100,posy=100;

int imageFileNum=3;//秒針、長針、短針

int nscale=3;//ディスプレイのスケール値(100%=0, 125%=1, 150%=2、200%=3)

int panelPosX=0, panelPosY=0;

AdjustValue[] ajVal= new AdjustValue[6];//Affin変換時の画像のずれを調整するためのデータ

String[] imageFileName ={"秒針1.png", "長針1.png", "短針1.gif"};//動画用画像ファイル

String backGroundImageFileName ="文字盤1.png";//背景画像ファイル

javax.swing.Timer timer1, timer2;

Calendar cal = Calendar.getInstance(TimeZone.getTimeZone("Japan"));

/\*\*

\* Creates new form AnalogClock1

\*/

public AnalogClock1() {

initComponents();

//System.out.println("x="+jPanel1.getBounds().x+" Y="+jPanel1.getBounds().y);

jComboBox1.setModel(new javax.swing.DefaultComboBoxModel(new String[] {

"100%", "125%", "150%" ,"200%"

}));

jComboBox1.setSelectedIndex(nscale);

try {

for(int i=0;i<imageFileNum;i++){

inputStream = this.getClass().getResourceAsStream(imageFileName[i]);

bufImgChar[i] = ImageIO.read(inputStream);

}

inputStream = this.getClass().getResourceAsStream(backGroundImageFileName);

bufferedImageBack=ImageIO.read(inputStream); //背景img

} catch (IOException e) {

e.printStackTrace();

}

for(int i=0;i<6;i++){

ajVal[i]=new AdjustValue();

}

createAdjustArray();

timer1 = new javax.swing.Timer(200 , this); //0.5秒ごとにタイマー起動

timer1.setActionCommand("timer1");//actionPerformed内での識別用文字列

timer1.start();//時計タイマーは直ちにスタート

}

void createAdjustArray(){

panelPosX=jPanel1.getBounds().x;

panelPosY=jPanel1.getBounds().y;

System.out.println("x="+jPanel1.getBounds().x+" Y="+jPanel1.getBounds().y);

//100%

ajVal[0].centerX=ajVal[0].centerY=250;

ajVal[0].sizeAdj=450;

ajVal[0].affinX=275;

ajVal[0].affinY=265;

ajVal[0].scaleValue=100;

ajVal[0].shiftX=50;

ajVal[0].shiftY=40;

//125%

ajVal[1].centerX=ajVal[1].centerY=250;

ajVal[1].sizeAdj=600;

ajVal[1].affinX=340;

ajVal[1].affinY=330;

ajVal[1].scaleValue=125;

ajVal[1].shiftX=40;

ajVal[1].shiftY=30;

//150%

ajVal[2].centerX=ajVal[2].centerY=250;

ajVal[2].sizeAdj=700;

ajVal[2].affinX=420;

ajVal[2].affinY=400;

ajVal[2].scaleValue=150;

ajVal[2].shiftX=70;

ajVal[2].shiftY=50;

// 200%

ajVal[3].centerX=ajVal[3].centerY=250;

ajVal[3].sizeAdj=900;

ajVal[3].affinX=550; //回転中心のx座標

ajVal[3].affinY=530; //回転中心のy座標

ajVal[3].scaleValue=250;//時計の大きさ

ajVal[3].shiftX=100;//水平方向の移動量

ajVal[3].shiftY=80;//垂直方向の移動量

}

public void actionPerformed(ActionEvent e){ //Timerがトリガーされるとここが呼び出される

String cmd = e.getActionCommand();//timer1とtimer2の識別用文字列

if(cmd.equals("timer1")){

long now = System.currentTimeMillis();

//long now=cal.getTimeInMillis();

now+=32400000;//GMTをJSTに変換( 32400000ミリ秒加える)

cnt=now/1000;//ミリ秒を秒単位にする

repaint();//画面表示を更新

}else if(cmd.equals("timer2")){

//現在は未使用

}

}

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

timer1.stop();

cnt+=5;

repaint();

}

private void jComboBox1ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

nscale =jComboBox1.getSelectedIndex(); //選択されている項目の番号をnscaleに返す

repaint();

}

double getDegreeOfSec(long sec){ //秒から秒針の角度を求める

System.out.println("Second:"+sec+"Sec="+((long)(sec % 60))\*6.0);

return 　　　　　　　　 ;

}

double getDegreeOfMin(long sec){//秒から秒針の角度を求める

System.out.println("Min="+(sec % 3600)/10);

return 　　　　　　　　 ;

}

double getDegreeOfHour(long sec){//短針の角度(24時間制)

System.out.println("Hr="+(sec % (3600\*12))/120);

return 　　　　　　　　 ;

}

//jPanel1のクラス

class MyPanel1 extends JPanel{

Image imageLong=null,imageShort=null,imageSec=null;//長針，短針，秒針用Image

Image imageBack=null;//文字盤用Image

public MyPanel1(){ //コンストラクタ

}

public void paintComponent( Graphics g ){

Graphics2D g2 = (Graphics2D) g;

super.paintComponent(g);

double rot=0.0;

AffineTransform at = new AffineTransform();//回転のためのクラス

AffineTransform atorg = new AffineTransform();//回転のためのクラス

atorg.rotate(0, 0, 0);//描画座標を初期値に戻す

AffineTransform af = new AffineTransform();

g2.setTransform(af);

g2.drawImage(bufferedImageBack, ajVal[nscale].shiftX, ajVal[nscale].shiftY, ajVal[nscale].sizeAdj,ajVal[nscale].sizeAdj, this);

//System.out.println("width,height="+jPanel1.getWidth()+", "+jPanel1.getHeight());

//秒針

rot=Math.toRadians(getDegreeOfSec(cnt));

af.rotate(rot,ajVal[nscale].affinX, ajVal[nscale].affinY);

g2.setTransform(af);//パネルに回転座標を当てはめる

g2.drawImage(bufImgChar[0], ajVal[nscale].shiftX, ajVal[nscale].shiftY, ajVal[nscale].sizeAdj,ajVal[nscale].sizeAdj,this);//描画

af.rotate(-rot,ajVal[nscale].affinX, ajVal[nscale].affinY);

g2.setTransform(af);//座標を戻す

//長針

rot=Math.toRadians(getDegreeOfMin(cnt));

af.rotate(rot,ajVal[nscale].affinX, ajVal[nscale].affinY);

g2.setTransform(af);//パネルに回転座標を当てはめる

//g2.drawImage(bufImgChar[0], 250, 250,this);//描画

g2.drawImage(bufImgChar[1], ajVal[nscale].shiftX, ajVal[nscale].shiftY, ajVal[nscale].sizeAdj,ajVal[nscale].sizeAdj,this);//描画

af.rotate(-rot,ajVal[nscale].affinX, ajVal[nscale].affinY);

g2.setTransform(af);//座標を戻す

//短針

rot=Math.toRadians(getDegreeOfHour(cnt));

af.rotate(rot,ajVal[nscale].affinX, ajVal[nscale].affinY);

g2.setTransform(af);//パネルに回転座標を当てはめる

//g2.drawImage(bufImgChar[0], 250, 250,this);//描画

g2.drawImage(bufImgChar[2],ajVal[nscale].shiftX, ajVal[nscale].shiftY, ajVal[nscale].sizeAdj,ajVal[nscale].sizeAdj,this);//描画

af.rotate(-rot,ajVal[nscale].affinX, ajVal[nscale].affinY);

g2.setTransform(af);//座標を戻す

}

}// MyPanel1クラスの定義ここまで

//表示位置や回転中心位置の修正用データ格納用クラス

class AdjustValue{

int scaleValue;

int centerX, centerY, affinX, affinY;

int sizeAdj;

int affinAdjX,affinAdjY;

int shiftX, shiftY;

AdjustValue(){

scaleValue=125;

centerX=250; centerY=250; affinX=310; affinY=370; //for125%

sizeAdj=500;

affinAdjX=affinAdjY=(sizeAdj-500)/2;

shiftX=60; shiftY=120;

}

AdjustValue(int scale, int afnX, int afnY, int szAdj, int afnAdjX, int afnAdjY, int shX, int shY){

scaleValue=scale;

centerX=250; centerY=250;

affinX=afnX; affinY=afnY;

sizeAdj=szAdj;

affinAdjX=afnAdjX;

affinAdjY=afnAdjY;

shiftX=shX; shiftY=shY;

}

}

//通常、ここから下はNetbeansによって自動的に作られており、自分で手動入力はしない。

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(AnalogClock1.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(AnalogClock1.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(AnalogClock1.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(AnalogClock1.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new AnalogClock1().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton jButton1;

private javax.swing.JComboBox<String> jComboBox1;

private javax.swing.JLabel jLabel2;

private javax.swing.JPanel jPanel1;

// End of variables declaration

}