```
package design;


public abstract class Actions {
	static protected Session _session;
	static protected MainView _ui;
	static protected void changeSession(String path)
	{
		_session.chgStudy(path);
	}
	static protected void addUiObserver(MainView ui)
	{
		_session.addObserver(ui);
	}

	public abstract void initAction(Object params);
}
```

```
package design;

public class Button implements EventObj {

}
```

```java
/**
 *
 */
package design;

/**
 * @author artur
 */
public class ChgState extends Actions {
	public void initAction(Object params){
		//Actions._session.changeState((int)params);
	}
}
```

```
/**
 *
 */
package design;

import java.util.ArrayList;
/**
 * @author artur
 */
public class Collection {

	Collection(String studyPath)
	{

	}
	/**
	 * @uml.property  name="_imageContainer"
	 * @uml.associationEnd  multiplicity="(1 1)" aggregation="composite" 
inverse="collection:design.ImageContainer"
	 */
	private ImageContainer _imageContainer;
	private ArrayList<String> _allImagePaths = new ArrayList<String>();

	/**
	 * Getter of the property <tt>_imageContainer</tt>
	 * @return  Returns the _imageContainer.
	 * @uml.property  name="_imageContainer"
	 */
	public ImageContainer get_imageContainer() {
		return _imageContainer;
	}

	/**
	 * Setter of the property <tt>_imageContainer</tt>
	 * @param _imageContainer  The _imageContainer to set.
	 * @uml.property  name="_imageContainer"
	 */
	public void set_imageContainer(ImageContainer _imageContainer) {
		this._imageContainer = _imageContainer;
	}

	}
```

```
package design;

public interface Dirs {
}
```

```
package design;


public class DoNext extends Actions {
	public  void initAction(Object params){
		_session.doNext();
	}
}
```

```
package design;


public class DoPrev extends Actions {
	public void initAction(Object params){
		_session.doPrevious();
	}
}
```

```
package design;

public interface EventObj {



}
```

```
/**
 *
 */
package design;

import java.awt.GridLayout;
import java.awt.image.BufferedImage;
import java.util.ArrayList;
import java.util.Iterator;

import javax.swing.ImageIcon;
import javax.swing.JLabel;
import javax.swing.JPanel;

/**
 * @author artur
 */
public class FourWins extends JPanel {

	/**
	 *
	 */
	private static final long serialVersionUID = -7169147603191135658L;

	public FourWins()
	{
		super();
	}
	public FourWins(ArrayList<BufferedImage> list)
	{
		super();
		Iterator<BufferedImage> itr=list.iterator();
		//if (list.size()!=4)  throw new IllegalArgumentException("Number of images "+list.size());
		setLayout(new GridLayout(2, 2, 0, 0));
		int i=0;
		while(itr.hasNext()&&i<4)
		{
			JLabel label=new JLabel(new ImageIcon((BufferedImage)itr.next()));
			label.setHorizontalAlignment(JLabel.CENTER);
			label.setVerticalAlignment(JLabel.CENTER);
			add(label);
			i++;
		}

		//if(list.size()!=1) throw new
	}
}
```

```
/**
 *
 */
package design;

import java.awt.image.BufferedImage;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Collection;
import java.util.Iterator;

/**
 * @author artur
 */
public class ImageContainer {
	ImageContainer(String path)
	{
		_imgPull = new LocalImg(path);
		_imgContainer=new ArrayList<BufferedImage>();
	}
	private LoadImg _imgPull;
	private ArrayList<BufferedImage> _imgContainer;

	public ArrayList<BufferedImage>  doNext(int currentIndex, int currentState) {
		// put in throw here for empty, incorrect inputs, etc --mikey
		_imgContainer.clear();
		for (int i = currentIndex; i < currentIndex+currentState; ++i) {
			try {
				_imgContainer.add(_imgPull.load(i));
			}catch (IOException e) {
				break;
			}
		}
		return _imgContainer;
	}

	public ArrayList<BufferedImage> doPrevious(int currentIndex, int currentState) {
		_imgContainer.clear();
		for (int i = currentIndex; i >= currentIndex-currentState; --i) {
			try {
				_imgContainer.add(_imgPull.load(i));
			}catch (IOException e) {
				break;
			}
		}
		return _imgContainer;
	}

	/*public void changeState(int currentIndex, int toState,String dirpath) {
		for (int i = 0; i < toState; ++i) {
			if (_allImages.size() <= currentIndex + i ) {
				break;
			} else {
```

```
                    _imageContainer.addLoadImg(_allImages[currentIndex + i]);
                }
            }                           _imgContainer.
            return _imageContainer.loadImgToArray();
        }*/
        public int findSize(String path)
        {
            return _imgPull.getSize(path);
        }
}
```

```
package design;

import java.awt.image.BufferedImage;
import java.io.IOException;


public interface LoadImg {
	public void init(String path);
	public abstract BufferedImage load(int index) throws IOException;
	public abstract int getSize(String path);
}
```

```java
/**
 *
 */
package design;

import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Arrays;

import javax.imageio.ImageIO;

/**
 * @author artur
 */
public class LocalImg implements LoadImg {
	ArrayList<File> _imagesPaths;
	public LocalImg(String path)
	{
		_imagesPaths=new ArrayList<File>();
		init(path);
	}
	public void init(String path)
	{
		File dir = new File(path);
		if (dir.isDirectory())
		{
			File[] tab=dir.listFiles();
			Arrays.sort(tab);
			_imagesPaths=new ArrayList<File>(Arrays.asList(tab));
		}
	}
	public BufferedImage load(int index) throws IOException {
		BufferedImage myPicture;
		File imgpath = _imagesPaths.get(index);
		myPicture = ImageIO.read(imgpath);
		return myPicture;
	}

	@Override
	public int getSize(String path) {
		return _imagesPaths.size();
	}


}
```

```
/**
 *
 */
package design;
import javax.swing.SwingUtilities;
/**
 * @author artur
 */
public class Main {

	/**
	 * @param args
	 */
	public static void main(String[] args) {
		// TODO Auto-generated method stub
		SwingUtilities.invokeLater(new Runnable() {
            @Override
            public void run() {
                MainView ex = new MainView();
                ex.setVisible(true);
            }
        });
	}


}
```

```
package design;

import java.util.ArrayList;
import java.util.Collection;
import java.util.Iterator;


public class MainController {

	/**
	 * @uml.property  name="_mainView"
	 * @uml.associationEnd  multiplicity="(1 1)" inverse="_mainController:design.MainView"
	 */
	//private MainView _mainView = new design.MainView();

	/**
	 * Getter of the property <tt>_mainView</tt>
	 * @return  Returns the _mainView.
	 * @uml.property  name="_mainView"
	 */
	//public MainView get_mainView() {
	//	return _mainView;
	//}

	/**
	 * Setter of the property <tt>_mainView</tt>
	 * @param _mainView  The _mainView to set.
	 * @uml.property  name="_mainView"
	 */
	//public void set_mainView(MainView _mainView) {
	//	this._mainView = _mainView;
	//}

	/**
	 * @uml.property name="_actions"
	 * @uml.associationEnd multiplicity="(0 -1)" ordering="true" aggregation="shared"
inverse="mainController:design.Actions"
	 */
	private ArrayList _actions;

	/**
	 * Returns the element at the specified position in this list.
	 * @param index  index of element to return.
	 * @return  the element at the specified position in this list.
	 * @see java.util.List#get(int)
	 * @uml.property  name="_actions"
	 */
	public ArrayList get_actions() {
		return _actions;
	}

	/**
	 * Returns the element at the specified position in this list.
 * @param index  index of element to return.
 * @return  the element at the specified position in this list.
 * @see java.util.List#get(int)
 * @uml.property  name="_actions"
 */
public Actions get_actions(int i) {
	return (Actions) _actions.get(i);
}

/**
 * Returns an iterator over the elements in this list in proper sequence.
 * @return  an iterator over the elements in this list in proper sequence.
 * @see java.util.List#iterator()
 * @uml.property  name="_actions"
 */
public Iterator _actionsIterator() {
	return _actions.iterator();
}

/**
 * Returns <tt>true</tt> if this list contains no elements.
 * @return  <tt>true</tt> if this list contains no elements.
 * @see java.util.List#isEmpty()
 * @uml.property  name="_actions"
 */
public boolean is_actionsEmpty() {
	return _actions.isEmpty();
}

/**
 * Returns <tt>true</tt> if this list contains the specified element.
 * @param element  element whose presence in this list is to be tested.
 * @return  <tt>true</tt> if this list contains the specified element.
 * @see java.util.List#contains(Object)
 * @uml.property  name="_actions"
 */
public boolean contains_actions(Actions actions) {
	return _actions.contains(actions);
}

/**
 * Returns <tt>true</tt> if this list contains all of the elements of the specified collection.
 * @param elements  collection to be checked for containment in this list.
 * @return  <tt>true</tt> if this list contains all of the elements of the specified collection.
 * @see java.util.List#containsAll(Collection)
 * @uml.property  name="_actions"
 */
public boolean containsAll_actions(Collection _actions) {
	return this._actions.containsAll(_actions);
}

/**
 * Returns the number of elements in this list.
 * @return  the number of elements in this list.
```

```
	 * @see java.util.List#size()
	 * @uml.property  name="_actions"
	 */
	public int _actionsSize() {
		return _actions.size();
	}

	/**
	 * Returns an array containing all of the elements in this list in proper sequence.
	 * @return  an array containing all of the elements in this list in proper sequence.
	 * @see java.util.List#toArray()
	 * @uml.property  name="_actions"
	 */
	public Actions[] _actionsToArray() {
		return (Actions[]) _actions.toArray(new Actions[_actions.size()]);
	}

	/**
	 * Returns an array containing all of the elements in this list in proper sequence; the runtime type of the returned
array is that of the specified array.
	 * @param a  the array into which the elements of this list are to be stored.
	 * @return  an array containing all of the elements in this list in proper sequence.
	 * @see java.util.List#toArray(Object[])
	 * @uml.property  name="_actions"
	 */
	public Actions[] _actionsToArray(Actions[] _actions) {
		return (Actions[]) this._actions.toArray(_actions);
	}

	/**
	 * Inserts the specified element at the specified position in this list (optional operation)
	 * @param index  index at which the specified element is to be inserted.
	 * @param element  element to be inserted.
	 * @see java.util.List#add(int,Object)
	 * @uml.property  name="_actions"
	 */
	public void add_actions(int index, Actions actions) {
		_actions.add(index, actions);
	}

	/**
	 * Appends the specified element to the end of this list (optional operation).
	 * @param element  element to be appended to this list.
	 * @return  <tt>true</tt> (as per the general contract of the <tt>Collection.add</tt> method).
	 * @see java.util.List#add(Object)
	 * @uml.property  name="_actions"
	 */
	public boolean add_actions(Actions actions) {
		return _actions.add(actions);
	}

	/**
	 * Removes the element at the specified position in this list (optional operation).
	 * @param index  the index of the element to removed.
 * @return  the element previously at the specified position.
 * @see java.util.List#remove(int)
 * @uml.property  name="_actions"
 */
public Object remove_actions(int index) {
	return _actions.remove(index);
}

/**
 * Removes the first occurrence in this list of the specified element  (optional operation).
 * @param element  element to be removed from this list, if present.
 * @return  <tt>true</tt> if this list contained the specified element.
 * @see java.util.List#remove(Object)
 * @uml.property  name="_actions"
 */
public boolean remove_actions(Actions actions) {
	return _actions.remove(actions);
}

/**
 * Removes all of the elements from this list (optional operation).
 * @see java.util.List#clear()
 * @uml.property  name="_actions"
 */
public void clear_actions() {
	_actions.clear();
}

/**
 * Setter of the property <tt>_actions</tt>
 * @param _actions  the _actions to set.
 * @uml.property  name="_actions"
 */
public void set_actions(ArrayList _actions) {
	this._actions = _actions;
}

/**
 * @uml.property  name="_session"
 * @uml.associationEnd  multiplicity="(1 1)" aggregation="composite" inverse="mainController:design.Session"
 */
private Session _session;

/**
 * Getter of the property <tt>_session</tt>
 * @return  Returns the _session.
 * @uml.property  name="_session"
 */
public Session get_session() {
	return _session;
}

/**
 * Setter of the property <tt>_session</tt>
```

```
	 * @param _session  The _session to set.
	 * @uml.property  name="_session"
	 */
	public void set_session(Session _session) {
		this._session = _session;
	}



}
```

```
package design;


import java.awt.BorderLayout;
import java.awt.CardLayout;
import java.awt.FlowLayout;
import java.awt.event.KeyEvent;
import java.util.ArrayList;
import java.util.Collection;
import java.util.Iterator;
import java.util.Observable;
import java.util.Observer;

import javax.imageio.ImageIO;
import javax.swing.GroupLayout;
import javax.swing.GroupLayout.Alignment;
import javax.swing.JButton;
import javax.swing.JComponent;
import javax.swing.JFileChooser;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JMenu;
import javax.swing.JMenuBar;
import javax.swing.JMenuItem;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.SwingConstants;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;
import java.util.ArrayList;

/**
 * @author artur
 *
 */
public class MainView extends JFrame implements Observer{
	private String path;

	//int Width=800;
	//int Heigth=600;

	private JComponent _fileOpen;
	private JComponent _fileExit;
	private JComponent _btnPrev;
	private JComponent _btnNext;
	private JComponent _btnOne;
	private JComponent _btnFour;
	private JComponent _imgContainer;
	private Actions _next;
	private Actions _prev;
	private Actions _init;
	private Actions _chgState;

	/**
	 * @uml.property  name="_imgWindow"
	 * @uml.associationEnd  multiplicity="(1 1)" aggregation="composite" 
inverse="mainView:design.WindowFactory"
	 */
	private WindowFactory _imgWindow;

	/**
	 * @uml.property  name="_mainController"
	 * @uml.associationEnd  multiplicity="(1 1)" inverse="_mainView:design.MainController"
	 */
	private MainController _mainController = new design.MainController();


	public MainView()
	{
		_imgWindow = new Win4Fact();
		_next = new DoNext();
		_prev = new DoPrev();
		_chgState = new ChgState();
		_imgContainer = new JPanel();
		_imgContainer.setLayout(new CardLayout(0, 0));
		/*ArrayList<BufferedImage> list = new ArrayList<BufferedImage>();
				try {
				list.add(ImageIO.read(new 
File("/home/artur/Downloads/MedImageViewerStudies/axial_head_mri/head01.jpg")));
				list.add(ImageIO.read(new 
File("/home/artur/Downloads/MedImageViewerStudies/axial_head_mri/head02.jpg")));
				list.add(ImageIO.read(new 
File("/home/artur/Downloads/MedImageViewerStudies/axial_head_mri/head03.jpg")));
				list.add(ImageIO.read(new 
File("/home/artur/Downloads/MedImageViewerStudies/axial_head_mri/head04.jpg")));
				} catch (IOException e) {
					// TODO Auto-generated catch block
					e.printStackTrace();
				}*/
		JFileChooser fileChooser = new JFileChooser();
		fileChooser.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
		int returnVal = fileChooser.showOpenDialog(null);
		if (returnVal == JFileChooser.APPROVE_OPTION)
		{
			//_init = new InitStudy(fileChooser.getSelectedFile().toString(),this);
		}

		initGUI();
		//ArrayList<BufferedImage> list = new ArrayList<BufferedImage>();
		//try {
		//list.add(ImageIO.read(new 
File("/home/artur/Downloads/MedImageViewerStudies/axial_head_mri/head01.jpg")));
		//list.add(ImageIO.read(new
File("/home/artur/Downloads/MedImageViewerStudies/axial_head_mri/head02.jpg")));
            //list.add(ImageIO.read(new
File("/home/artur/Downloads/MedImageViewerStudies/axial_head_mri/head03.jpg")));
            //list.add(ImageIO.read(new
File("/home/artur/Downloads/MedImageViewerStudies/axial_head_mri/head04.jpg")));
            //_imgWindow = new Win4Fact(list);
            //_imgContainer = _imgWindow.getWindow();
            //initGUI();
            //} catch (IOException e) {
            //      // TODO Auto-generated catch block
            //      e.printStackTrace();
            //}

      }

      /**
       * Initiate the OneState button
       *
       */
      private void initbtnOne(){
            _btnOne = new JButton("One");
            ((JButton)_btnOne).setVerticalAlignment(SwingConstants.BOTTOM);
            ((JButton)_btnOne).setHorizontalAlignment(SwingConstants.LEFT);
            ((JButton)_btnOne).addActionListener(new ActionListener(){
            @Override
            public void actionPerformed(ActionEvent arg0){
             evActionbtnOne();
            }
            });
      }
      private void evActionbtnOne(){
            _imgWindow = new Win1Fact();
            _chgState.initAction(1);
            ((JButton) _btnOne).disable();
            ((JButton) _btnFour).enable();
      }
      /**
       * Initiate the FourState button
       *
       */
      private void initbtnFour(){
            _btnFour = new JButton("Four");
            ((JButton)_btnFour).setVerticalAlignment(SwingConstants.BOTTOM);
            ((JButton)_btnFour).setHorizontalAlignment(SwingConstants.LEFT);
            ((JButton)_btnFour).addActionListener(new ActionListener(){
            @Override
            public void actionPerformed(ActionEvent arg0){
             evActionbtnFour();
            }
            });
      }
      private void evActionbtnFour(){
            _imgWindow = new Win4Fact();
            _chgState.initAction(4);
```

```
        ((JButton) _btnFour).disable();
        ((JButton) _btnOne).enable();

}
/**
 * Initiate the Previous button item
 *
 */
private void initbtnPrev(){
        _btnPrev = new JButton("Prev");
        ((JButton)_btnPrev).addActionListener(new ActionListener(){
        @Override
        public void actionPerformed(ActionEvent arg0){
                evActionbtnPrev();
        }
        });
}
private void evActionbtnPrev(){
        _prev.initAction(-1);
}
/**
 * Initiate the Next button item
 *
 */
private void initbtnNext(){
        _btnNext = new JButton("Next");
        ((JButton)_btnNext).addActionListener(new ActionListener(){
        @Override
        public void actionPerformed(ActionEvent arg0){
                evActionbtnNext();
        }
        });
}
private void evActionbtnNext(){
        _next.initAction(-1);
}

/**
 * Initiate the _fileOpen menu item
 *
 */
private void fileOpenInit(){
        _fileOpen= new JMenuItem("Open");
    ((JMenuItem)_fileOpen).setMnemonic(KeyEvent.VK_O);
_fileOpen.setToolTipText("Open new study");
evActionFileOpen();
}
/**
 * Set default action for fileOpen item
 * Overriding actionPerformed method
 *
 */
private void evActionFileOpen()
{
        ((JMenuItem)_fileOpen).addActionListener(new ActionListener(){

        @Override
        public void actionPerformed(ActionEvent arg0){
                JFileChooser fileChooser = new JFileChooser();
                fileChooser.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
                int returnVal = fileChooser.showOpenDialog(null);
                if (returnVal == JFileChooser.APPROVE_OPTION)
                {
                        initStudy(fileChooser.getSelectedFile().toString());
                }
        }
});
}
public void initStudy(String path)
{
//_init = new InitStudy(path,this);
}
/**
 * Initiate the _fileExit menu item
 *
 */
private void fileExitInit(){
        _fileExit = new JMenuItem("Exit");
        ((JMenuItem)_fileExit).setMnemonic(KeyEvent.VK_E);
_fileExit.setToolTipText("Exit application");
((JMenuItem)_fileExit).addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent event) {
        System.exit(0);
    }
});
}

@Override
public void update(Observable arg0, Object arg1) {
        // TODO Auto-generated method stub
        if (arg1 instanceof ArrayList<?>)
        {
                _imgWindow.update((ArrayList<BufferedImage>)arg1);
                _imgContainer.removeAll();
                _imgContainer.add(_imgWindow.getWindow());
                //revalidate();
                repaint();
                //getContentPane().add(_imgContainer, BorderLayout.CENTER);
                //repaint();
        }
}
private void initGUI()
{
        //setResizable(false);
                        setTitle("Medical Image Viewing Console");
                        setSize(800, 600);
                    //setExtendedState(Frame.MAXIMIZED_BOTH);
                        //setSize(Width,Heigth);
                //setSize(xSize,ySize);
                setLocationRelativeTo(null);

                setDefaultCloseOperation(EXIT_ON_CLOSE);

                ///////////////////////////////////////////Menu Bar generation/////////////////////////////////////

              JMenuBar menubar= new JMenuBar();
              //ImageIcon icon = new ImageIcon(getClass().getResource("exit.png"));

              JMenu file=new JMenu("File");
              file.setMnemonic(KeyEvent.VK_F);

              fileOpenInit();
              fileExitInit();

            file.add(_fileOpen);
            file.add(_fileExit);
            menubar.add(file);
            setJMenuBar(menubar);

            ///////////////////////////////Next, Prev, Image Container//////////////////////////////////////:

            //Previous
            JPanel panel = new JPanel();
                getContentPane().add(panel, BorderLayout.WEST);

                initbtnPrev();
                GroupLayout gl_panel = new GroupLayout(panel);
                gl_panel.setHorizontalGroup(
                    gl_panel.createParallelGroup(Alignment.LEADING)
                        .addComponent(_btnPrev, GroupLayout.DEFAULT_SIZE,
GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                );
                gl_panel.setVerticalGroup(
                    gl_panel.createParallelGroup(Alignment.LEADING)
                        .addGroup(gl_panel.createSequentialGroup()
                            .addGap(100)
                            .addComponent(_btnPrev, GroupLayout.DEFAULT_SIZE,
GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addGap(95))
                );
                panel.setLayout(gl_panel);

                //Next
                JPanel panel_4 = new JPanel();
                getContentPane().add(panel_4, BorderLayout.EAST);

                initbtnNext();
                GroupLayout gl_panel_4 = new GroupLayout(panel_4);
                gl_panel_4.setHorizontalGroup(
                    gl_panel_4.createParallelGroup(Alignment.LEADING)
                        .addGroup(gl_panel_4.createSequentialGroup()
                            .addComponent(_btnNext, GroupLayout.DEFAULT_SIZE,
GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                        );
                        gl_panel_4.setVerticalGroup(
                                gl_panel_4.createParallelGroup(Alignment.LEADING)
                                        .addGroup(gl_panel_4.createSequentialGroup()
                                                .addGap(100)
                                                .addComponent(_btnNext, GroupLayout.DEFAULT_SIZE,
GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                                .addGap(95))
                        );
                        panel_4.setLayout(gl_panel_4);

                        //Image Container
                        //JPanel panel_1 = new JPanel();
                        getContentPane().add(_imgContainer, BorderLayout.CENTER);
                        //JScrollPane scrollPane = new JScrollPane();
                        //panel_1.add(scrollPane);
                        //
                        //panel_1.add(_imgContainer);


                        ////////////////////////////////State Buttons////////////////////////////////
                        JPanel panel_3 = new JPanel();
                        panel_3.setBorder(null);
                        getContentPane().add(panel_3, BorderLayout.SOUTH);

                        initbtnOne();
                        initbtnFour();
                        panel_3.setLayout(new FlowLayout(FlowLayout.LEFT, 5, 5));
                        panel_3.add(_btnOne);
                        panel_3.add(_btnFour);
        }

}
```

```
package design;


public class Menu implements EventObj {

}
```

```
/**
 *
 */
package design;

import java.awt.Color;
import java.awt.Graphics;
import java.awt.Image;
import java.awt.image.BufferedImage;
import java.util.ArrayList;

import javax.swing.ImageIcon;
import javax.swing.JLabel;

/**
 * @author artur
 */
public class OneWins extends JLabel {

	/**
	 *
	 */
	private static final long serialVersionUID = 1L;

	public OneWins()
	{
		super();
	}
	public OneWins(ArrayList<BufferedImage> list)
	{
		super();
		//Image img = resize(list.get(0));
		setIcon(new ImageIcon(list.get(0)));
		setHorizontalAlignment(JLabel.CENTER);
	setVerticalAlignment(JLabel.CENTER);
		//if(list.size()!=1) throw new
	}
	public void update(ArrayList<BufferedImage> list)
	{
		setIcon(new ImageIcon(list.get(0)));
	}
	public Image resize(BufferedImage img)
	{
		Image scaledImage = img.getScaledInstance((int)800, (int)600, Image.SCALE_SMOOTH);
	BufferedImage imageBuff = new BufferedImage((int)800, (int)600, BufferedImage.TYPE_INT_RGB);
	Graphics g = imageBuff.createGraphics();
	g.drawImage(scaledImage, 0, 0, new Color(0,0,0), null);
	g.dispose();
	return scaledImage;
	}
}
```

```
/**
 *
 */
package design;

import java.awt.image.BufferedImage;
import java.io.IOException;

/**
 * @author artur
 */
public class RemoteImg implements LoadImg {

	public void init(String path){
	}
	public  BufferedImage load(int index) throws IOException{
		return new BufferedImage(0, 0, 0);
	}
	public  int getSize(String path){
		return 0;
	}
}
```

```
/**
 *
 */
package design;
import java.util.Observable;
/**
 * @author artur
 */
public class Session extends Observable{


	public void chgStudy(String path)
	{
		_dirPath=path;
		collection=new ImageContainer(path);
		nbrImages = collection.findSize(path);
		currentImage = 0;
		displayState = 4;
		doNext();
		if (nbrImages==0)notifyObservers(-1);
	}
	/**
	 * @uml.property name="imageCont"
	 * @uml.associationEnd multiplicity="(1 1)" aggregation="shared" inverse="session:design.ImageContainer"
	 */
	private int displayState;
	private ImageContainer collection;
	private int currentImage;
	private int nbrImages;
	String _dirPath;

	/**
	 * Getter of the property <tt>imageCont</tt>
	 * @return  Returns the imageCont.
	 * @uml.property  name="imageCont"
	 */
	public ImageContainer getCollection() {
		return collection;
	}

	/**
	 * Setter of the property <tt>imageCont</tt>
	 * @param imageCont  The imageCont to set.
	 * @uml.property  name="imageCont"
	 */
	public void setCollection(ImageContainer  imageCont) {
		this.collection = imageCont;
	}

	public void doNext() {
		int overload=(currentImage+displayState)-nbrImages;
		if (overload<displayState){
			setChanged();
```

```
				if(overload<=0){
				notifyObservers(collection.doNext(currentImage, displayState));
				currentImage+=displayState;
				}
				else{
				notifyObservers(collection.doNext(currentImage, displayState-overload));
				currentImage=nbrImages-1;
				}
			}
		}

		public void doPrevious() {
			if (currentImage-2*displayState>=0)
			{
				setChanged();
				currentImage=currentImage-2*displayState;
				if(currentImage>=displayState){
				notifyObservers(collection.doPrevious(currentImage, displayState));
				currentImage=currentImage+displayState;
				}else{
				notifyObservers(collection.doPrevious(currentImage, currentImage));
				currentImage=0;
				}
			}
		}

		public void changeState(int toState) {
			currentImage = currentImage-displayState;
			if (currentImage<0)currentImage=0;
			if (currentImage+toState>=nbrImages)currentImage=nbrImages-toState-1;
			displayState = toState;
			setChanged();
			notifyObservers(collection.doNext(currentImage, toState));
		}

		@Override
		public synchronized boolean hasChanged() {
			// TODO Auto-generated method stub
			return super.hasChanged();
		}

		@Override
		public void notifyObservers() {
			// TODO Auto-generated method stub
			super.notifyObservers();
		}


	}
```

```
/**
 *
 */
package design;

/**
 * @author artur
 */
public class URI implements Dirs {

}
```

```java
/**
 *
 */
package design;

import java.awt.image.BufferedImage;
import java.util.ArrayList;
import java.util.List;

import javax.swing.JComponent;
import javax.swing.JLabel;

/**
 * @author artur
 */
public class Win1Fact implements WindowFactory {

	Win1Fact()
	{
		_oneWins = new design.OneWins();
	}

	Win1Fact(ArrayList<BufferedImage> list)
	{
		_oneWins = new design.OneWins(list);
	}
	public void update(ArrayList<BufferedImage> list)
	{
		_oneWins = new design.OneWins(list);
	}
	/**
	 * @uml.property  name="_oneWins"
	 * @uml.associationEnd  multiplicity="(1 1)" inverse="win1Fact:design.OneWins"
	 */
	private JComponent _oneWins;

	/**
	 * Getter of the property <tt>_oneWins</tt>
	 * @return  Returns the _oneWins.
	 * @uml.property  name="_oneWins"
	 */
	public JComponent getWindow() {
		return _oneWins;
	}

	/**
	 * Setter of the property <tt>_oneWins</tt>
	 * @param _oneWins  The _oneWins to set.
	 * @uml.property  name="_oneWins"
	 */
	public void set_oneWins(OneWins _oneWins) {
		this._oneWins = _oneWins;
	}
}
```

```
/**
 *
 */
package design;

import java.awt.image.BufferedImage;
import java.util.ArrayList;

import javax.swing.JComponent;
import javax.swing.JLabel;

/**
 * @author artur
 */
public class Win4Fact implements WindowFactory {
	public Win4Fact()
	{
		fourWins=new FourWins();
	}

	public Win4Fact(ArrayList<BufferedImage> list)
	{
		fourWins=new FourWins(list);
	}
	public void update(ArrayList<BufferedImage> list)
	{
		fourWins=new FourWins(list);
	}
	/**
	 * @uml.property  name="fourWins"
	 * @uml.associationEnd  multiplicity="(1 1)" inverse="win4Fact:design.FourWins"
	 */
	private JComponent fourWins;

	/**
	 * Getter of the property <tt>fourWins</tt>
	 * @return  Returns the fourWins.
	 * @uml.property  name="fourWins"
	 */
	public JComponent getWindow() {
		return fourWins;
	}

	/**
	 * Setter of the property <tt>fourWins</tt>
	 * @param fourWins  The fourWins to set.
	 * @uml.property  name="fourWins"
	 */
	public void setFourWins(FourWins fourWins) {
		this.fourWins = fourWins;
	}

}
```

```
package design;

import javax.swing.JPanel;


public abstract class Window extends JPanel {
	private static final long serialVersionUID = -8687907176608557245L;

}
```

```
/**
 *
 */
package design;

import java.awt.image.BufferedImage;
import java.util.ArrayList;

import javax.swing.JComponent;
import javax.swing.JLabel;

/**
 * @author artur
 */
public interface WindowFactory {
	public abstract JComponent getWindow();
	public abstract void update(ArrayList<BufferedImage> list);

}
```