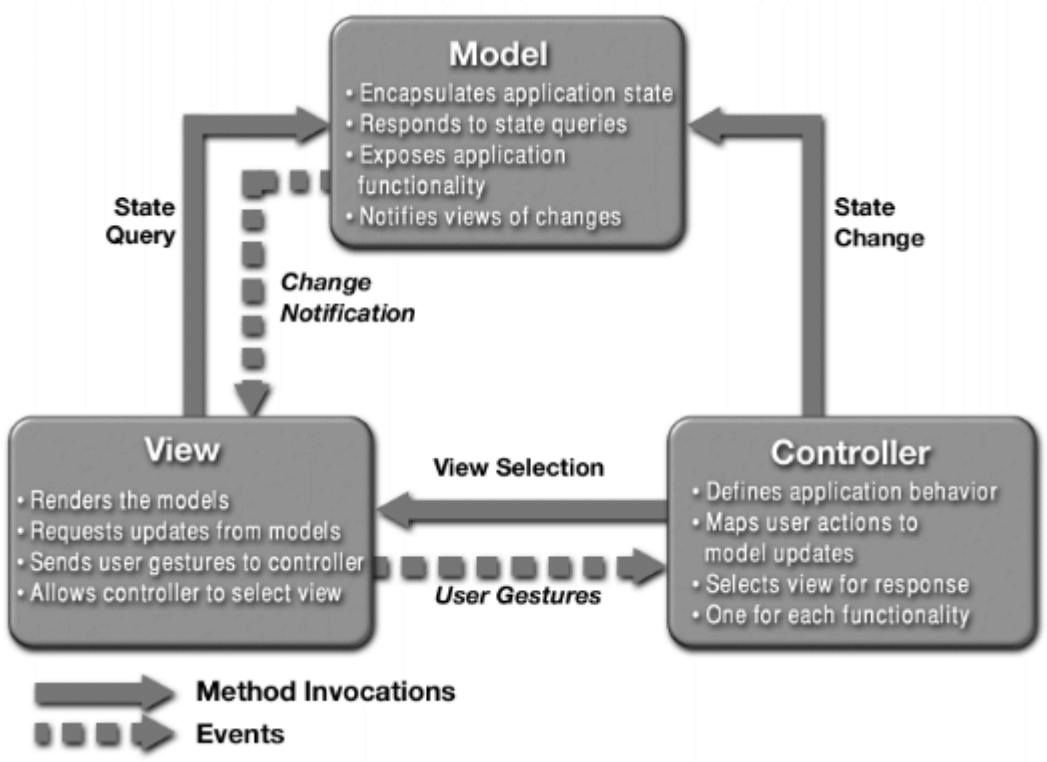
2023/2024(1)
EF234302 Object Oriented Programming

Lecture #9c

MVC Design Pattern

Misbakhul Munir IRFAN SUBAKTI
司馬伊凡
Мисбакхул Мунир Ирфан Субакти

MVC (Model-View-Controller) – Common Implementation –



MVC (Model-View-Controller)

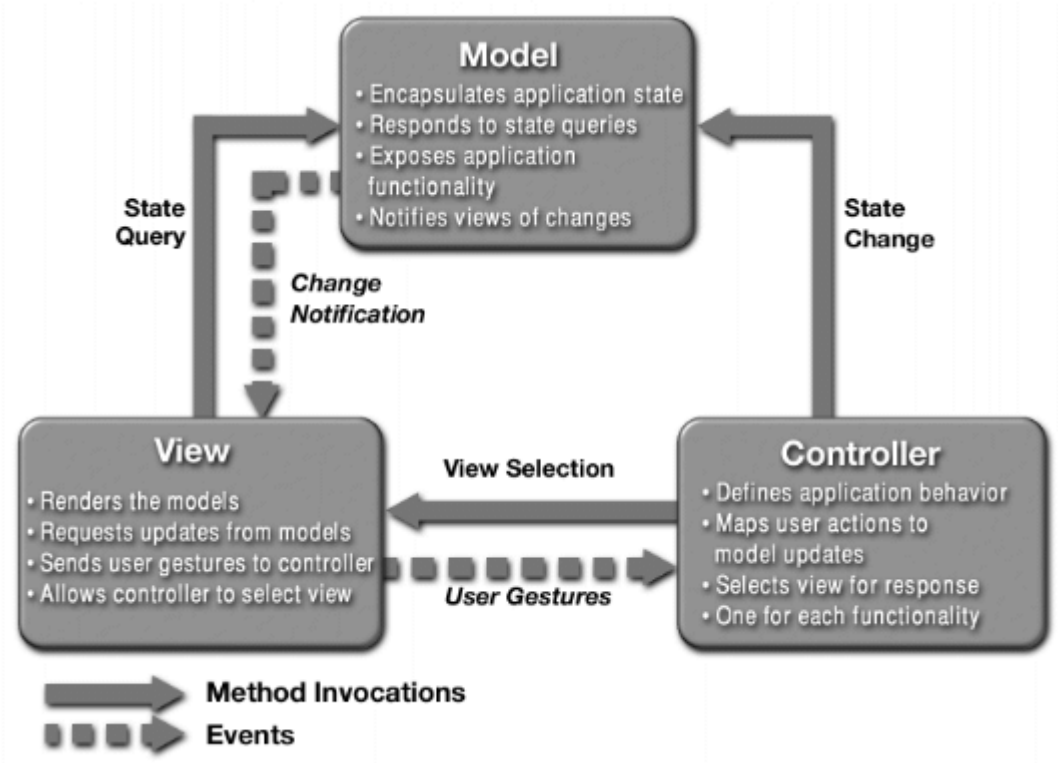
- MVC Design Pattern
  - Java app with a GUI consists of several components.
  - MVC pattern: popular software design pattern for this type of software
  - It separates the application logic from the user interface and the control between the *user interface* and the application *logic*
- Model → represents data and the rules that govern access to and updates of this data. In enterprise software, a model often serves as a software approximation of a real-world process.
- View → renders the contents of a model. It specifies exactly how the model data should be presented. If the model data changes, the view must update its presentation as needed.
- Controller → translates the user's interactions with the view into actions that the model will perform.

MVC (continued)

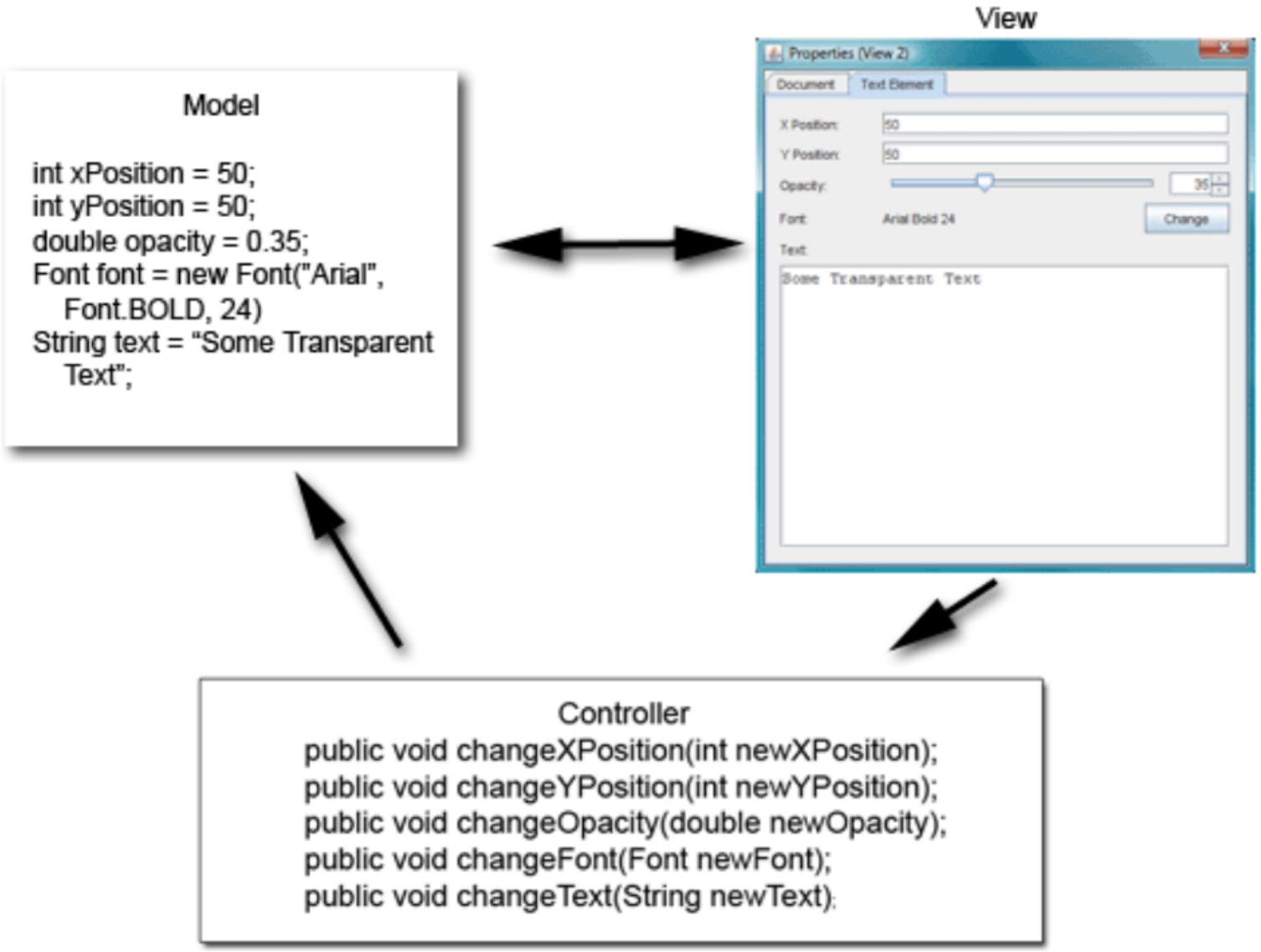
- In a Java application, the model, view, and controller can be implemented in separate classes, or in a single class. The communications between the model and the view can be implemented in several ways as well.
- Tutorial:

  https://www.oracle.com/technical-resources/articles/javase/application-design-with-mvc.html

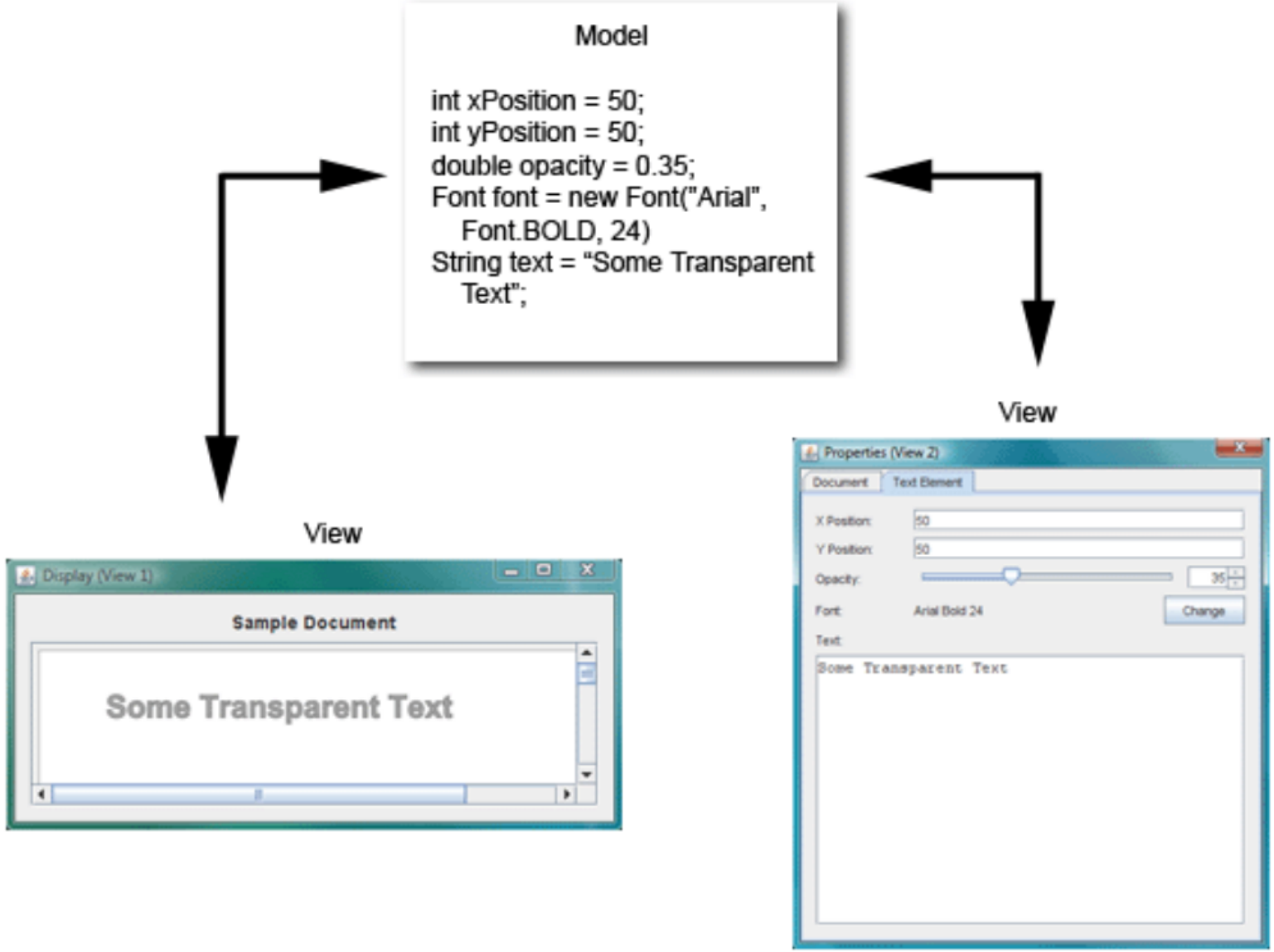
MVC (Model-View-Controller) – Common Implementation –



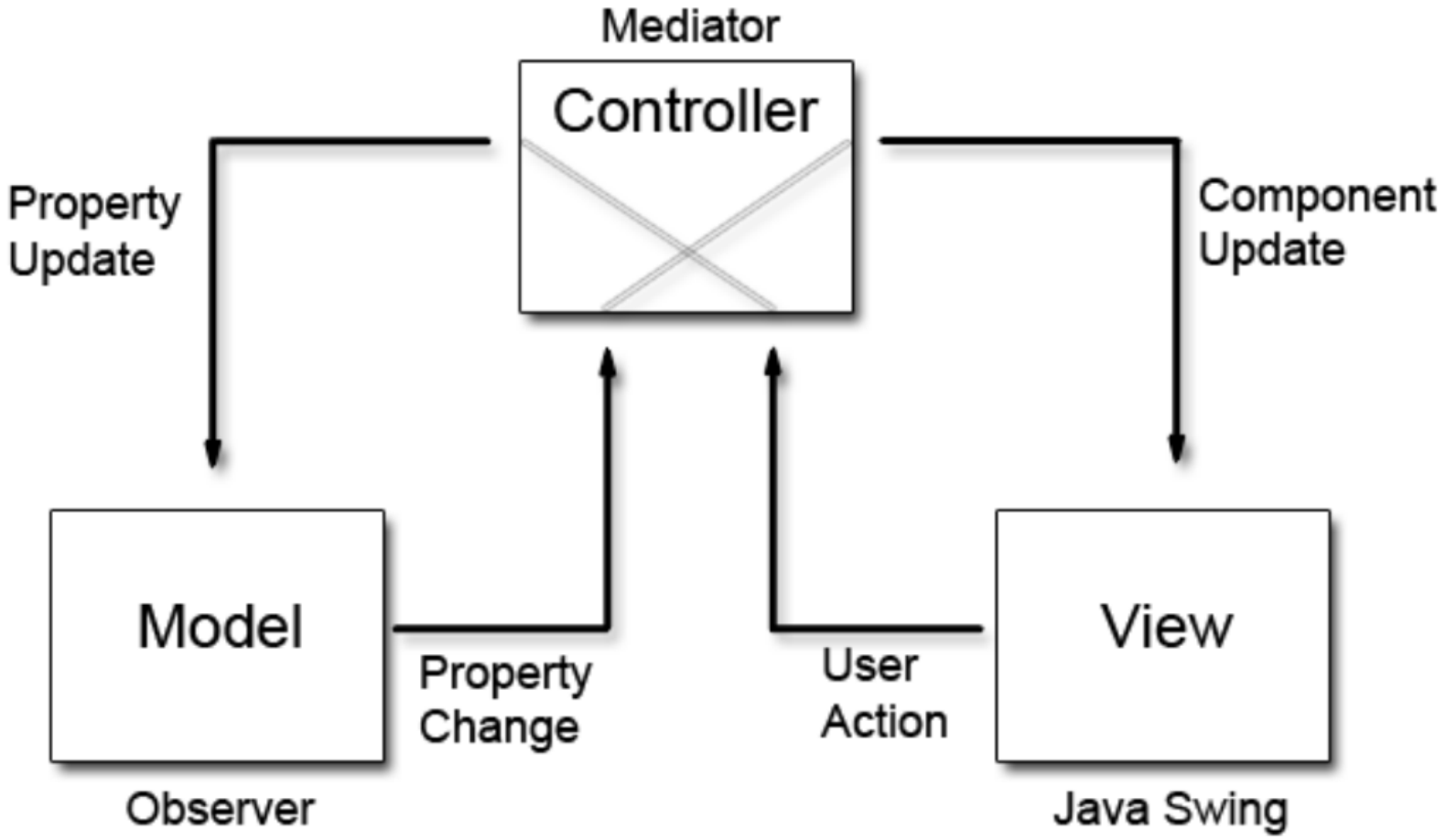
MVC: Diagram


View
Model
int xPosition = 50;
int yPosition = 50;
double opacity = 0.35;
Font font = new Font("Arial",
Font.BOLD, 24)
String text = "Some Transparent
Text";
Properties (View 2)
Document
Text Element
X Position:
50
Y Position:
50
Opacity:
35
Font:
Arial Bold 24
Change
Text:
Some Transparent Text
Controller
public void changeXPosition(int newXPosition);
public void changeYPosition(int newYPosition);
public void changeOpacity(double newOpacity);
public void changeFont(Font newFont);
public void changeText(String newText);

Same model → multiple views



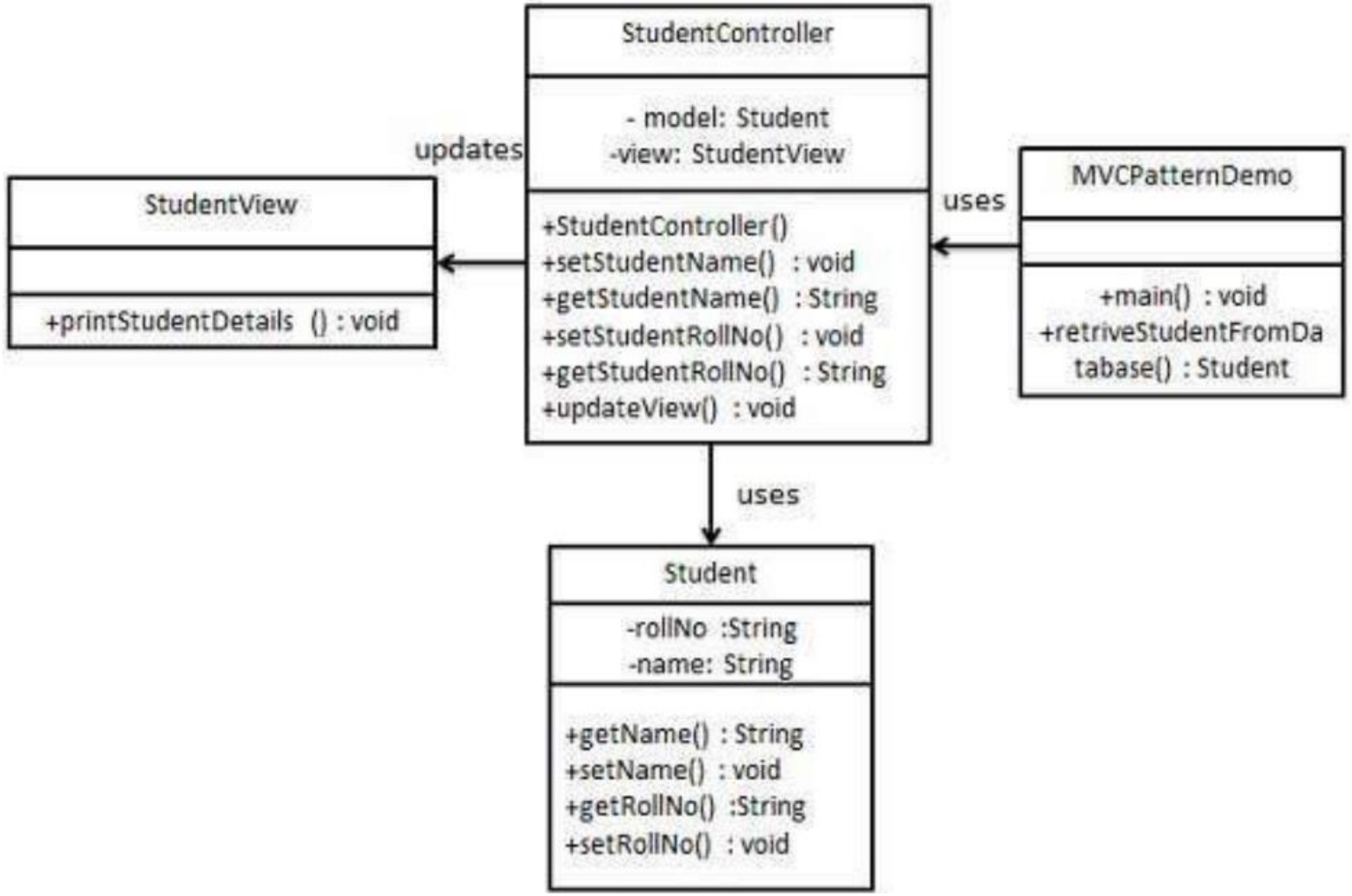
Controller: mediator between model & view



MVC: Example 1

- https://www.tutorialspoint.com/design_pattern/mvc_pattern.htm

MVC: Student


StudentController
- model: Student
-view: StudentView
+StudentController()
+setStudentName() : void
+getStudentName() : String
+setStudentRollNo() : void
+getStudentRollNo() : String
+updateView() : void
updates
StudentView
+printStudentDetails () : void
uses
MVCPatternDemo
+main() : void
+retriveStudentFromDa
tabase() : Student
uses
Student
-rollNo :String
-name: String
+getName() : String
+setName() : void
+getRollNo() :String
+setRollNo() : void

Student: Model-View-Controller

Model.java

```java
package student;
public class Model {
    private String rollNo;
    private String name;
    public String getRollNo() {
        return rollNo;
    }
    public void setRollNo(String rollNo) {
        this.rollNo = rollNo;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
}
```

Controller.java

```java
package student;
public class Controller {
    private Model model;
    private View view;
    public Controller(Model model, View view) {
        this.model = model;
        this.view = view;
    }
    public void setStudentName(String name) {
        model.setName(name);
    }
    public String getStudentName() {
        return model.getName();
    }
    public void setStudentRollNo(String rollNo) {
        model.setRollNo(rollNo);
    }
    public String getStudentRollNo() {
        return model.getRollNo();
    }
    public void updateView() {
        view.printStudentDetails(model.getName(),
                model.getRollNo());
    }
}
```

View.java

```java
package student;
public class View {
    public void printStudentDetails(String studentName,
            String studentRollNo) {
        System.out.println("Student: ");
        System.out.println("Name: " + studentName);
        System.out.println("Roll No: " + studentRollNo);
    }
}
```

Problems | Javadoc | Declaration | Search | Console

```
<terminated> Main (6) [Java Application] D:\Program\Java\JDK\bin\javaw.e
Student:
Name: Robert
Roll No: 10
Student:
Name: John
Roll No: 10
```

Main.java

```java
package student;
public class Main {
    public static void main(String[] args) {
        // Fetch student record based on his roll no from the database
        Model model = retriveStudentFromDatabase();
        // Create a view: to write student details on console
        View view = new View();
        Controller controller = new Controller(model, view);
        controller.updateView();
        // Update model data
        controller.setStudentName("John");
        controller.updateView();
    }
    private static Model retriveStudentFromDatabase() {
        Model student = new Model();
        student.setName("Robert");
        student.setRollNo("10");
        return student;
    }
}
```

MVC: Example 2

Model.java

```java
package counter;
import java.util.Observable;
public class Model extends Observable {
    private int value;
    public Model() {
        value = 0;
    }
    public int getValue() {
        return value;
    }
    public void setValue(int value) {
        this.value = value;
        setChanged();
        notifyObservers();
    }
    public void incValue() {
        value++;
        setChanged();
        notifyObservers(value);
    }
    public void decValue() {
        value--;
        setChanged();
        notifyObservers(value);
    }
}
```

MyPanel.java

```java
package counter;
import java.awt.GridLayout;
import javax.swing.*;
public class MyPanel extends JPanel {
    private JLabel label = new JLabel("Counter:");
    private JTextField counter = new JTextField(6);
    protected JButton up = new JButton("Up");
    protected JButton down = new JButton("Down");
    public MyPanel() {
        setSize(400, 150);
        GridLayout layout = new GridLayout(2, 0);

        // Horizontal gap between components
        layout.setHgap(20);
        // Vertical gap between components
        layout.setVgap(10);

        setLayout(layout);
        counter.setText("0");
        add(label);
        add(counter);
        add(up);
        add(down);
        counter.setEditable(false);
        up.setActionCommand("UP");
        down.setActionCommand("DOWN");
    }
    public void setCounter(String value) {
        counter.setText(value);
    }
}
```

View.java

```java
package counter;
import java.awt.Dimension;
import java.awt.Toolkit;
import java.awt.event.*;
import java.util.Observable;
import java.util.Observer;
import javax.swing.*;
public class View implements Observer, ActionListener {
    private MyPanel panel = new MyPanel();
    private Model model = new Model();
    public void init(String title, int value) {
        JFrame.setDefaultLookAndFeelDecorated(true);
        JFrame frame = new JFrame(title);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        model.setValue(value); // Set the value in Model
        panel.setCounter(Integer.toString(value)); // Set the value in View
        // Make the window to appear in the middle of the screen
        Dimension screen = Toolkit.getDefaultToolkit().getScreenSize();
        frame.setLocation((int) screen.getWidth() / 2 - frame.getWidth() / 2,
                (int) screen.getHeight() / 2 - frame.getHeight() / 2);
        model.addObserver(this);
        panel.up.addActionListener(this);
        panel.down.addActionListener(this);
        frame.setContentPane(panel);
        frame.pack();
        frame.setVisible(true);
    }
    public void update(Observable o, Object arg) {
        if (o instanceof Model) {
            panel.setCounter("" + arg);
        }
    }
    public void actionPerformed(ActionEvent e) {
        String s = e.getActionCommand();
        if (s.equals("UP")) {
            model.incValue();
        } else if (s.equals("DOWN")) {
            model.decValue();
        }
    }
}
```

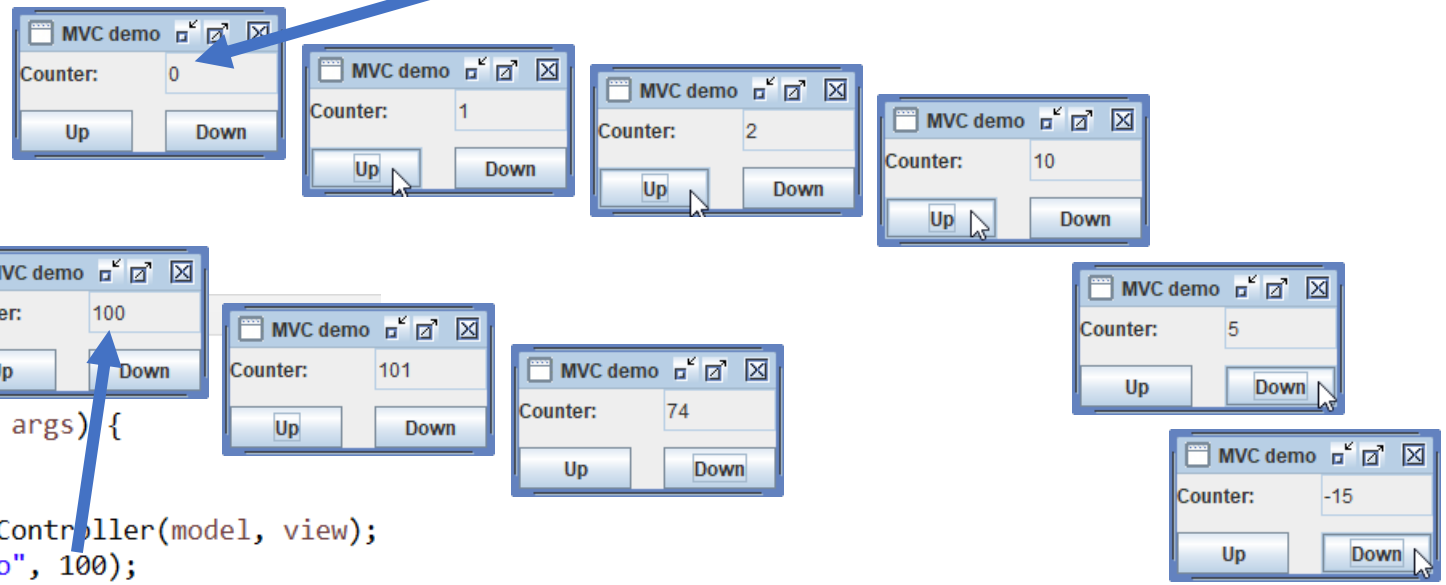
MVC: Example 2 (continued)

Controller.java

```java
package counter;
public class Controller {
    private Model model;
    private View view;
    public Controller(Model model, View view) {
        this.model = model;
        this.view = view;
    }
    public void setModelValue(int value) {
        model.setValue(value);
    }
    public int getModelValue() {
        return model.getValue();
    }
    public void initView(String title, int value) {
        view.init(title, value);
    }
}
```

Main.java

```java
package counter;
public class Main {
    public static void main(String[] args) {
        Model model = new Model();
        View view = new View();
        Controller controller = new Controller(model, view);
        controller.initView("MVC demo", 0);
    }
}
```



Main.java

```java
package counter;
public class Main {
    public static void main(String[] args) {
        Model model = new Model();
        View view = new View();
        Controller controller = new Controller(model, view);
        controller.initView("MVC demo", 100);
    }
}
```