Unit 7
Model-View-Controller

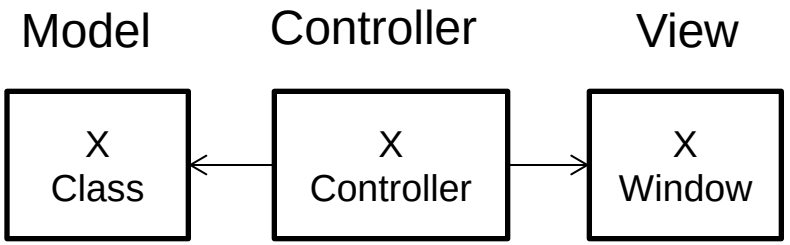
Software Analysis and Design Project

Universidad Autónoma de Madrid

Model-View-Controller (MVC)

- Model (domain classes)
- View (classes that inherit from JComponent, like JButton…)
- Controllers (classes that implement interfaces of the EventListener type, like ActionListener, MouseListener…)

1. The controller is registered in the interface
2. The user performs some action in the interface
3. The controller treats the event:
   1. Model is changed (if needed)
   2. New view is generated

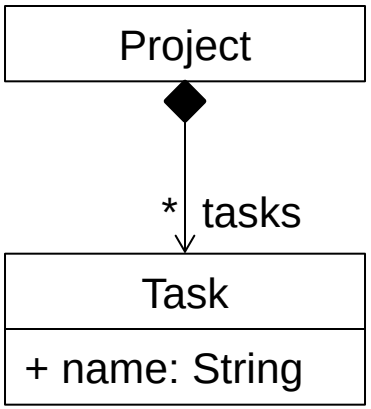


4. The new view accesses the data to be used from the model

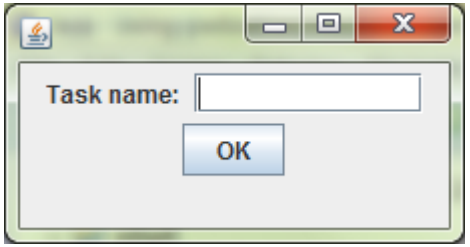
Model

```
public class Project {

   private List<Task> tasks = new ArrayList<Task>();
   // ...

   public void addTask(Task t) {
      tasks.add(t);
   }
}
```

```
public class Task {

   private String name;
   // ...

   public Task (String name) {
      this.name = name;
   }

   public String getName() {
      return name;
   }
}
```



View




```
public class TaskCreation extends JPanel {

   private JTextField taskName;
   private JButton    oKButton;
   // ...

   public TaskCreation () {
      // assign layout
      // ...

      // create components
      JLabel label = new JLabel("Task name: ");
      taskName     = new JTextField(10);
      oKButton     = new MyButton("OK");

      // Add components to panel
      this.add(label);
      this.add(taskName);
      this.add(oKButton);
   }

   // Assign controller to button
   public void setControlador(ActionListener c) {
      oKButton.addActionListener(c);
   }

   // Get the name of a task from the JTextField
   public String getTaskName () {
      return taskName.getText();
   }
}
```

Controller

```
public class TaskCreationControl implements ActionListener {

   private TaskCreation view;
   private Project model;

   public TaskCreationControl(TaskCreation view, Project model) {
      this.view  = view;
      this.model = model;
   }

   @Override
   public void actionPerformed(ActionEvent arg0) {

      // validate view values
      String taskName = view.getTaskName();
      if (taskName.equals("")) {
         JOptionPane.showMessageDialog(view, "You must type a name.", "Error", JOptionPane.ERROR_MESSAGE);
         return;
      }

      // Modify model
      Task task = new Task(taskName);
      model.addTask(task);

      // show new view
      ProjectDetail newView = this.getPanelProjectDetail();
      newView.update(task);
      newView.setVisible(true);
      view.setVisible     (false); // A different possibility: use CardLayout
   }
   // ...
}
```

Putting everything together…

```
// view
TaskCreation view= new TaskCreation();

// model
Project model = new Project();

// controller
TaskCreationController controller = new TaskCreationController(view, model);

// Associate controller to view
view.setController(controller);
```