Alejandro Reyes Bautista – Q13134663

Model View Controller Pattern Presentation: SCRIPT

Slide 1:

This presentation is about how the Model View Controller pattern is being used to assign responsibilities, in the context of the Athletics club Demonstration Program

Slide 2:

First we have the model. In java the model is composed by classes.

This level defines where the data is stored. The model doesn’t know anything about views and controllers. Whenever there are changes done in the model it will automatically notify observers than the changes are made.

On the left we can see the domain model of the Athletics Club System. Each element is a container of the data that is supposed to belong to the entity it represents. In example, the Event class is the model that stores all the data which belong to the entity event: id, type, gender, ageGroup, transport, date...That means that when we need to get or set any concrete data, we have to interact with that element.

Slide 3:

Afterward we have the view, that it would be the user interface.

This level creates an interface to show the actual output to the user. The controller tells the view what to display to the user. It handles requests from the user and informs controller.

This is an example of view for the demonstration program main page. There are different actions; for event we have add an event, view an event, update an event and pick teams; for training actions we have add a training, view a training update a training and add a training record and for membership actions we get add, view an update a member. The view could be different, but all of them must allow the user to achieve all these tasks.

Slide 4:

Last we find the controller. Controller is a level which acts like a brain of the entire MVC system. It is the link between a user and the system. Understand both input and output, converting them into the appropriate messages.

Here we are one of the controller methods that we have implemented. It shows the list of events by search type (age group, type or gender for example)

When the user clicks the button, the method binded is called. This method gets the input (the search type), interacts with the model to retrieve the data and displays the output in the right elements.

Slide 5:

The MVC pattern exists in all the steps of the ICONIX process.

In the Robustness Diagram, View, Controller and Model are separated in boundaries, controllers and entities. That means Robustness diagram divide responsibilities in the manner it identifies distinct kind of objects.

Boundary objects (in red) represent different screens of the same view.

Control objects (in orange) represent “the brain” that addresses business logic. Each one is a task that the controller has to do.

Entity objects (in green) represent the distinct objects that compose the model.

Slide 6:

The sequence diagram also fits the MVC pattern about assigning responsibilities. Every part, takes it owns responsibility, being isolated. The View is just the interface, and the controller decides what to do when a user action is triggered (to display another screen, validate the input of the user or create a new Event using these details, for example). The model just stores the data (add event method only add a new event to the list).

Slide 7:

To finalize, the main advantages of the MVC model in the project have been the faster development process, because one programmer could work on the view while another can work on the controller to create business logic, the ability to provide multiple views and the fact that the modification does not affect the entire model.