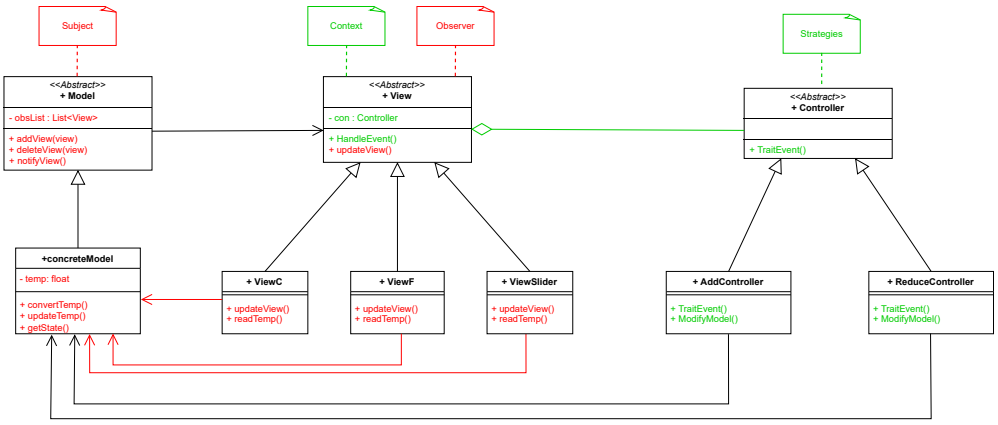
DW N° 4

Exercise 1

Solution[1]

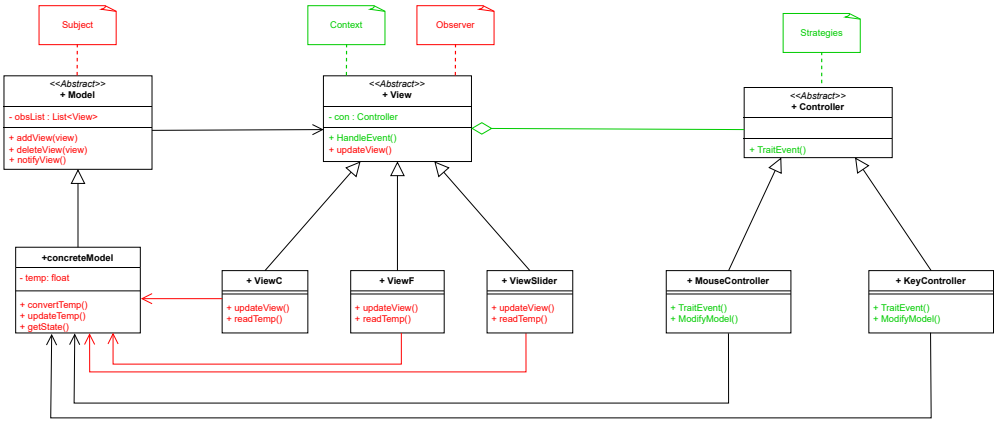


Explication

- **The View**: we have 3 views in this app:
  - `ViewC`: displays the temperature in Celsius with - and + buttons.
  - `ViewF`: displays the temperature in Fahrenheit with - and + buttons.
  - `ViewSlider`: displays the temperature with a slider.

  `readTemp()`: calls the `getState()` method of the `concreteModel`. When the user triggers an event, the `HandleEvent()` method is called, which in turn calls the `TraitEvent()` method of the controller class.
- **Controller**: we have 2 types of events, either increasing or decreasing the temperature. Therefore, we have 2 subclasses: `AddController` and `ReduceController`. `TraitEvent()` calls `ModifyModel()`, which then calls the `updateTemp()` method of `concreteModel`.
- **Model**: after the temperature is updated, the `notifyView()` method loops through all views in the list and calls their `updateView()` method.

Solution$_2$



Note

This solution is very similar to the first one. The only difference is that, instead of having two events for increasing and decreasing the temperature, we now have keyboard and mouse events.

In theory, this should work, but in some tools and languages, it might cause issues if the slider cannot be used with keyboard arrow keys. This would mean that the **strategies aren't interchangeable** , violating the **Strategy** Design Pattern