**MVC** (Model-View-Controller) is a pattern in software design commonly used to implement user interfaces, data, and controlling logic. It emphasizes a separation between the software's business logic and display. This "separation of concerns" provides for a better division of labor and improved maintenance.

Some other design patterns are based on MVC, such as MVVM (Model-View-Viewmodel), MVP (Model-View-Presenter), MVT (Model-View-Template) and MVW (Model-View-Whatever).

The three parts of the MVC software-design pattern can be described as follows:

1. ***Model:*** *Manages data and business logic.*
2. ***View:*** *Handles layout and display.*
3. ***Controller:*** *Routes commands to the model and view parts*

**[Model View Controller example](https://developer.mozilla.org/en-US/docs/Glossary/MVC" \l "model_view_controller_example" \o "Permalink to Model View Controller example)**

Imagine a simple shopping list app. All we want is a list of the name, quantity and price of each item we need to buy this week. Below we'll describe how we could implement some of this functionality using MVC.



**MVC Vs 3 tier**

At first glance, the three tiers may seem similar to the MVC (Model View Controller) concept; however, topologically they are different.

A fundamental rule in a three-tier architecture is the client tier never communicates directly with the data tier; in a three-tier model all communication must pass through the middleware tier. Conceptually the three-tier architecture is linear.

However, the MVC architecture is triangular: the View sends updates to the Controller, the Controller updates the Model, and the View gets updated directly from the Model.