Understand MVC

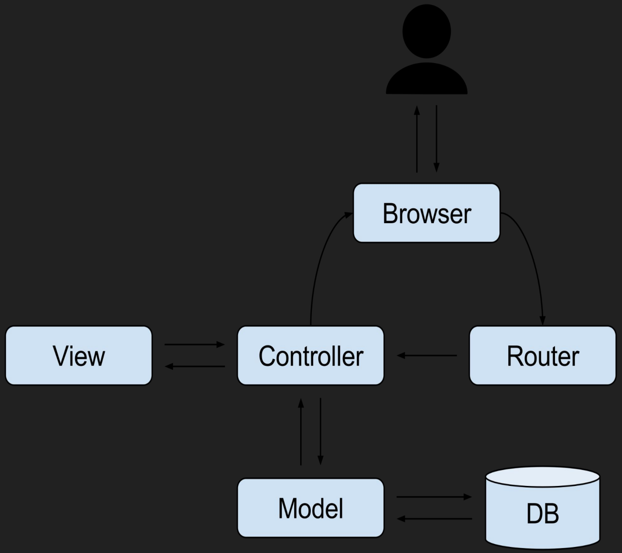
By Julian Martinez

Reasoning for use of analogies

Sometimes learning new things for me is overwhelming and stressful, so to augment that stress, I like to use analogies to objects or situations I am more comfortable with to better understand as well as feel more comfortable with the new subject material.

Spaceship Analogy

I was looking at the image for a flowchart-like depiction of the MVC and it really stood out to me it's set up almost exactly like a spaceship.



It’s a bit of a stretch for some components, but mostly the MVC

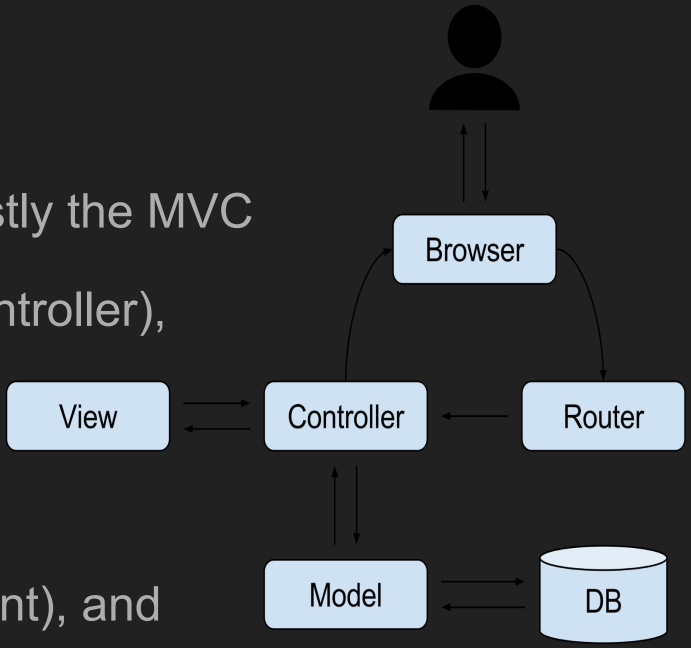
consists of, like a spaceship, the control center (controller),

the view (the viewport/window),

the router (navigation system),

model (the engine), data base (storage compartment), and

it’s a bit of a stretch but you can liken the browser to outer space.



Controller

Hinted by its name, this controls the important parts of the MVC to send commands to the other parts of the app by listening for events from the user input.

But more importantly, it is connected to all the other architecture.

Analogy: The control center is connected to the spaceship navigation with a nav system (router) and engines (model) to get you to the destination.

Router

As alluded to in the controller description, the router is very much like the nav system of a spaceship because it decides where to navigate in the application based on user input.

Model

It’s a bit of a stretch but the model is like an engine because you could say that it drives the application forward by encapsulating data and logic, but one of its main purposes is to separate data processing from storage.

Data Base (DB)

This one is self explanatory: the data base acts as a storage center for data, so you could liken it to a storage compartment on a spaceship.

View

The view can be likened to, and indeed is sometimes referred to as in web development, as the viewport.

Analogy: You can look outside the viewport and see outer space (the application’s state)

Browser

The browser is the environment that the MVC operates in, so it can be likened to outer space.

Why use MVC?

The main purpose for the MVC paradigm is to allow developers to more cohesively work on an application together without causing bugs when one part of the application is changed or updated.

The MVC allows modularity, which enables developers to swap out entire sections of the MVC as needed without affecting the whole application’s codebase.