Organising your code in MVC

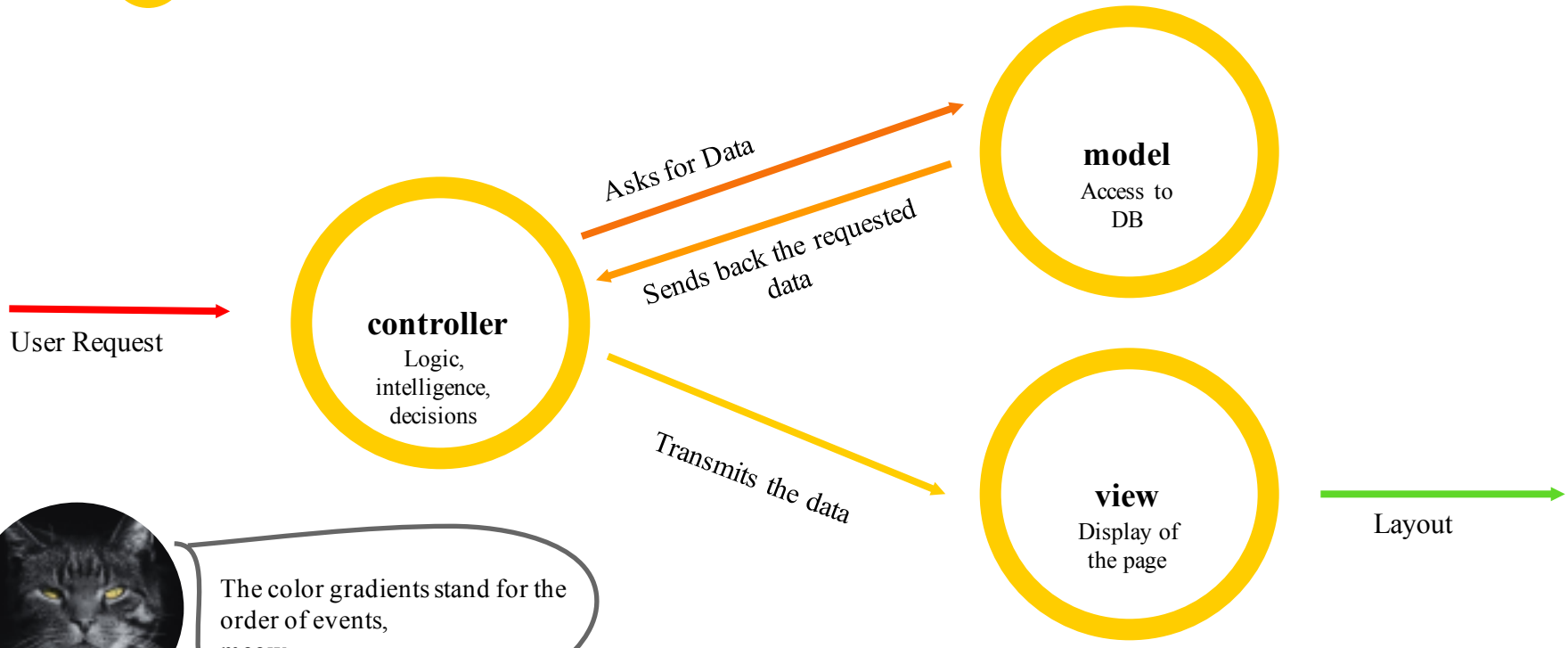
Huh?

M => Model

V => View

C => Controller

How does it work?



The color gradients stand for the order of events,
meow

In practical terms...

The controller is the switcher who will activate the right switches to get the user to his destination.

From the User's perspective, he or she requests a page from the controller and returns a view according to certain parameters (does he or she have permission to access this page for example?)

But why?

- To better organize its code strategically: the view only takes care of the display, the controller takes care of the user inputs and transmits them to the model.
- To work with a widely used convention
- To be able to work more effectively as a team
- To communicate as events between the view, model and controller
- It allows a composite structure: the view will be composed of several elements superimposed on each other, generated by interactions between the user and the application.

Concrete example

MVC applied to a restaurant

Restaurant as MVC

I ordered many Salmon!

controller
waiter

model
cook

view
clients

DB
Freezer, shelves

Order transmitted

The Dishes are ready!

Order given

Dishes sent to service

Is there any Salmon ?

YESS!

Concrete example

MVC applied to an MP3 player

MP3 Player MVC

Via the interface, the user can add MP3s, manage playlists, rename tracks, modify the volume.

The MP3 Player saves information, adds, deletes tracks, …

The MP3 Player starts the playback, stops the music, displays song information, volume of sound, ..

MP3 Player MVC

View: the different interfaces, different displays requested by the user

Model : where the logic of the application is stored, the states (reading, pause, ..) the data (riffs, playlist compositions, …) to manage and play the songs

Controller : When the user clicks [read] button :

- the controller asks the model to play the selected song,
- the model informs the view of the song being played,
- the graphical interface displays information about the song being played.

Now it's your turn!

Make a little website.

If the user is not logged in, the homepage displays "*Howdy the unknown*". If the user is logged in, the homepage displays "*Howdy*" + the user name.

Make a small menu with the "home" and "admin" buttons.
If you (/ the user) are not logged in, by clicking on "admin", you will be redirected to the home page. If you are logged in, you arrive on a page with the message "*welcome to the secret of the Gods !*".

In the controller: you manage the user's redirections (login condition).
In the model: you manage session variable data.
In the view: you control the display.

FR Ressources

- Adopter le MVC avec exemples (Openclassrooms)
- Bonnes pratiques à connaître (openclassrooms)

EN Ressources

- MVC controller example
- Simple PHP MVC example