Introduction to MVC (Model–View–Controller)

IT Learning &
Outsourcing Center

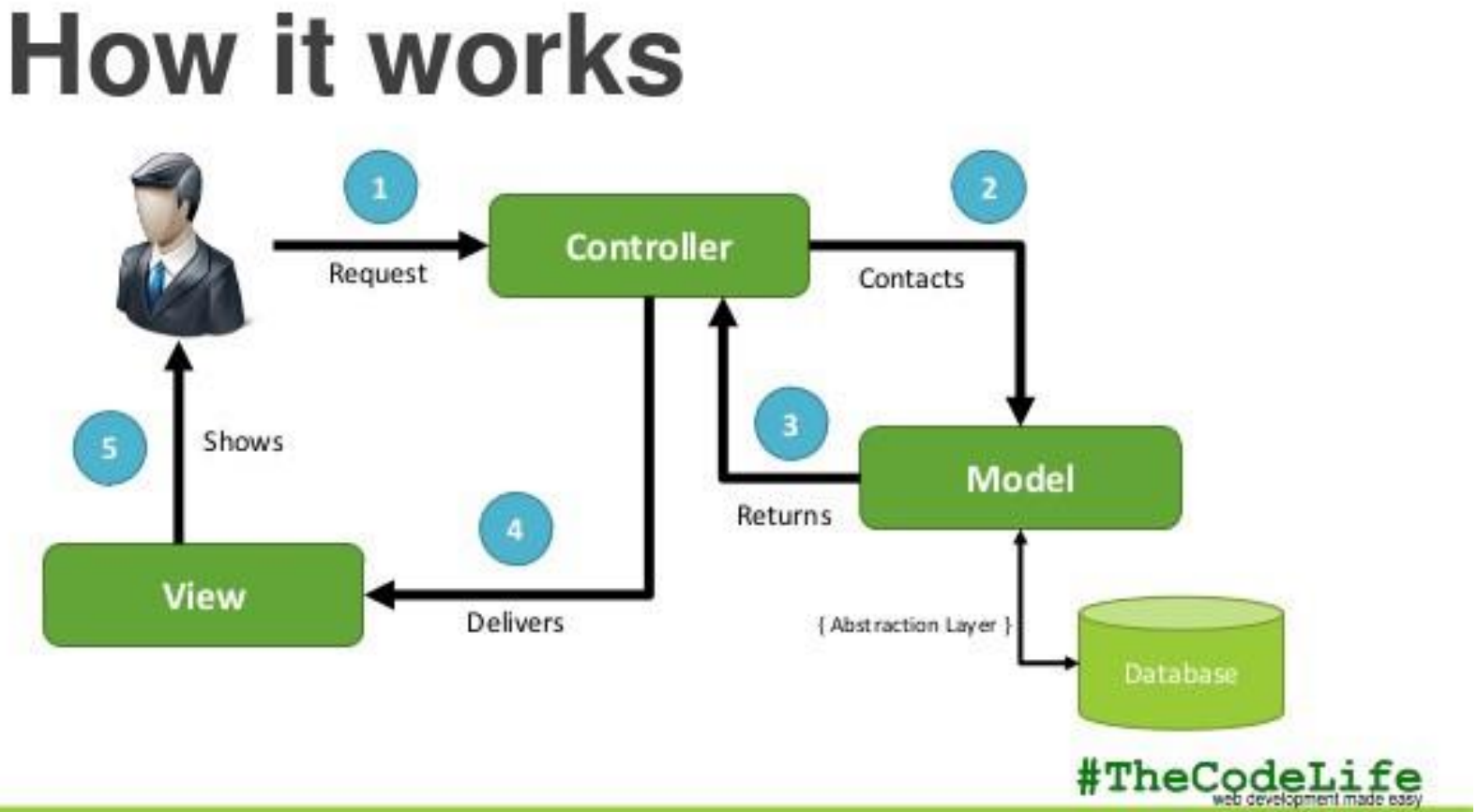
www.pragmatic.bg

Lector: Stefan Vadev
Skype: stefanvadev77
E-mail: stefan.vadev@gmail.com
Facebook:
https://www.facebook.com/stefan.vadev.7

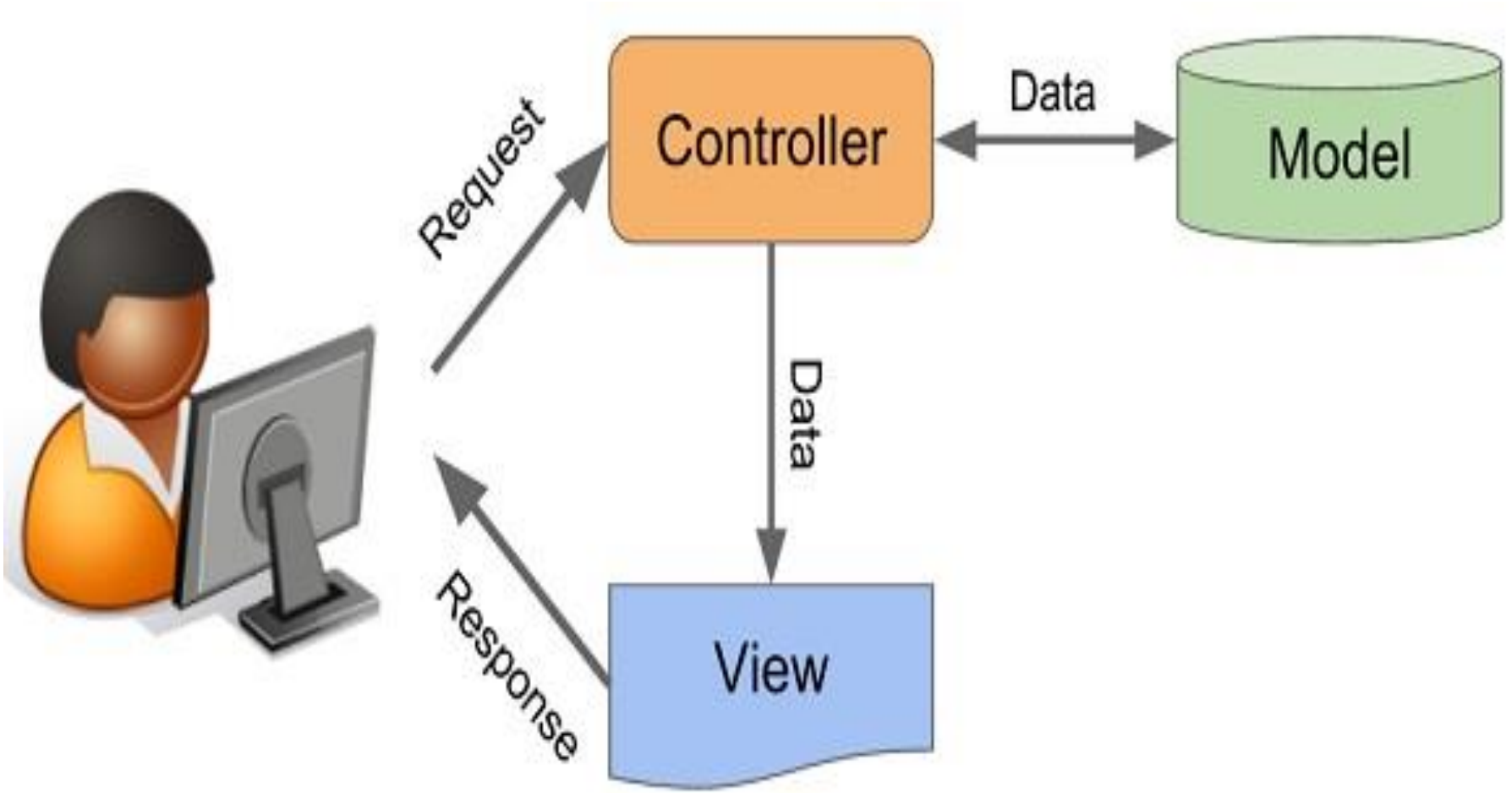
Summary

- What is MVC
- Spring MVC

MVC - Overview



MVC - Overview



MVC – Model View Controller

In a typical application you will find these three fundamental parts:

Data (Model)
An interface to view and modify the data (View)
Operations that can be performed on the data (Controller)

The MVC pattern, in a nutshell, is this:
The **model** represents the data, and does nothing else. The model does NOT depend on the controller or the view.

The **view** displays the model data, and sends user actions (e.g. button clicks) to the controller. The view can:

be independent of both the model and the controller; or
actually **be** the controller, and therefore depend on the model.

The **controller** provides model data to the view, and interprets user actions such as button clicks. The controller depends on the view and the model. In some cases, the controller and the view are the same object.

MVC – Model

Models represent knowledge. A model could be a single object (rather uninteresting), or it could be some structure of objects.
There should be a one-to-one correspondence between the model and its parts on the one hand, and the represented world as perceived by the owner of the model on the other hand.

MVC – View

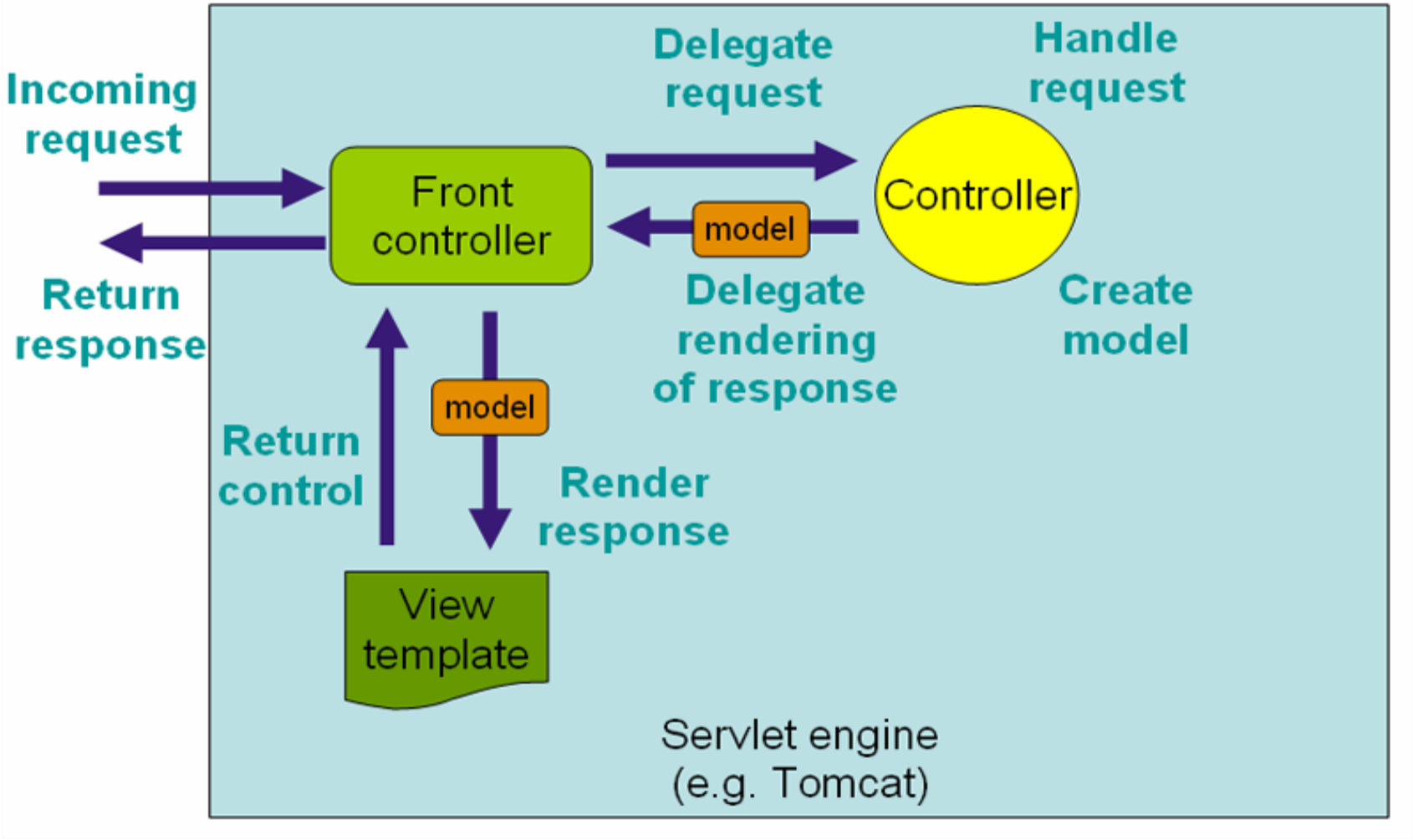
A view is a (visual) representation of its model. It would ordinarily highlight certain attributes of the model and suppress others. It is thus acting as a *presentation filter*.

A view is attached to its model (or model part) and gets the data necessary for the presentation from the model by asking questions. It may also update the model by sending appropriate messages. All these questions and messages have to be in the terminology of the model, the view will therefore have to know the semantics of the attributes of the model it represents.

MVC – Controller

A controller is the link between a user and the system. It provides the user with input by arranging for relevant views to present themselves in appropriate places on the screen. It provides means for user output by presenting the user with menus or other means of giving commands and data. The controller receives such user output, translates it into the appropriate messages and pass these messages on to one or more of the views.

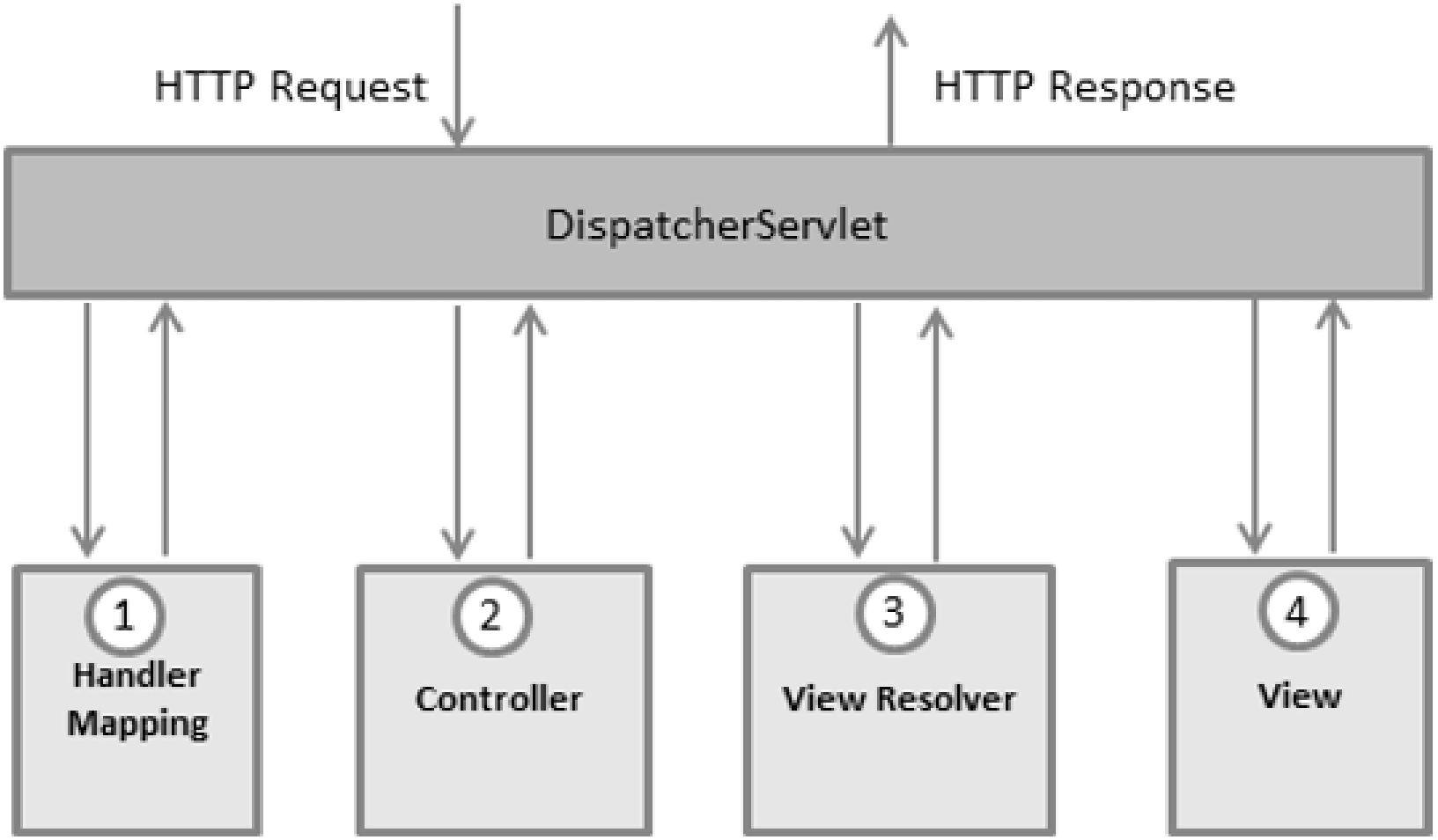
Spring MVC


Incoming request
Return response
Front controller
Delegate request
Handle request
Controller
model
Delegate rendering of response
Create model
model
Return control
Render response
View template
Servlet engine (e.g. Tomcat)

How Spring MVC works

The Spring Web model-view-controller (MVC) framework is designed around a ***DispatcherServlet*** that handles all the HTTP requests and responses. The request processing workflow of the Spring Web MVC ***DispatcherServlet*** is illustrated in the following diagram:

How Spring MVC works



How Spring MVC works

After receiving an HTTP request, *DispatcherServlet*consults the *HandlerMapping* to call the appropriate *Controller*.

The *Controller* takes the request and calls the appropriate service methods based on used GET or POST method. The service method will set model data based on defined business logic and returns view name to the *DispatcherServlet*.

The *DispatcherServlet* will take help from *ViewResolver* to pickup the defined view for the request.

Once view is finalized, The *DispatcherServlet*passes the model data to the view which is finally rendered on the browser.

Spring MVC

More info about (Spring) MVC here:

https://docs.spring.io/spring/docs/current/spring-framework-reference/web.html

https://www.tomdalling.com/blog/software-design/model-view-controller-explained/

https://blog.codinghorror.com/understanding-model-view-controller/

http://www.baeldung.com/spring-mvc-tutorial

https://www.tutorialspoint.com/spring/spring_web_mvc_framework.htm

http://websystique.com/spring-4-mvc-tutorial/

Summary

- What is MVC
- Spring MVC