

JoshNewlands - 1602713

by Joshua Newlands

FILE	12810_JOSHUA_NEWLANDS_JOSHNEWLANDS_- _1602713_178145_1724001317.PDF (127.91K)		
TIME SUBMITTED	15-APR-2019 11:44 PM (UTC+0100)	WORD COUNT	1773
SUBMISSION ID	104832485	CHARACTER COUNT	9070



Advanced Web Part 2

M6X01086

Lecturer: Mike Dacey

Josh Newlands, 1602713
josh.newlands@outlook.com

Abstract

Frameworks help web developers create websites by guiding them with how they should approach their code. MVC is a web development model that tells the developer to follow the principles of Models, Views, and Controllers. This report explores the concepts behind the MVC framework and argues the case for the approach to development taken by Symfony2.

Table of Contents

1. Introduction	1
2.MVC model	2
2.1 Model	2
2.2 View	3
2.3 Controller	3
2.4 Template pattern	3
3.Symfony2	4
References	5

1. Introduction

Using a framework for web development allows developers to build websites that take advantage of the resource and tools in the framework [1]. Complex web applications that do not follow a framework can be difficult to maintain and expand their functionality [2]. As well as frameworks there are software design patterns to follow which tell a developer how to approach the development of software. An advantage of following a framework in the development of web applications is that code will be easier to maintain and work with.

2.MVC model

There are different frameworks that use the MVC model such as Spring, Ruby on Rails, Django, and ASP.NET MVC [3] [4]. MVC helps to structure code by separating the application into three parts: Models; Views; and Controllers [5]. The MVC model places a focus on abstraction. Abstraction is the representation of essential features without the complex code of the features tightly wound together.

Abstraction is used to represent features in an application so that each feature is much less dependent on the other. Through this idea a loosely coupled architecture can be created that allows for each feature to be tested independently as well as making it easier to change in future versions, without having massive impact on the rest of the application code. Without abstraction you can end up repeating code and it becomes difficult to change later.

Fowler in his book “Patterns of Enterprise Application Architecture” talks about the idea that in complex websites there are essential features such as security, internationalization, and providing particular views for certain users and that if the code for these features is scattered across objects then it can lead to code duplication and difficulty in future changes.

What Fowler talks about here also refers to the idea that abstraction is important in complex applications, the idea that input controller behaviour is spread across multiple objects shows that the code would be tightly coupled.

When creating a loosely coupled architecture the developer must look at the separation and isolation features. Some of the features that come into consideration would be the view that is provided to the user, the data that is manipulated, and the connection to the database. When these features are considered for decoupling the model, view, and controller architecture begins to form.[8]

2.1 Model

The model is responsible for managing the data [5].

The model is responsible for getting and keeping data in the application, it responds to requests from the view through the controller, it also updates the view. Model classes are kept in the model folder within the MVC structure.

Having the code for connecting to databases and holding application data within a model class keeps these separate from “user” side code and works as a form of encapsulation. The isolation of this code is also beneficial to keep data connection code from being repeated as well as being less impactful if the connection needs to change to a different database in the future. [8]

2.2 View

The view is responsible for displaying data provided by the model [4]. By using views we can control the format of how data is being displayed. The view also works as the user interface that can relay requests to the controllers; this way behaviour can change during runtime. The views may be changed through the use of actions or commands set up in the controller.

Within a typical mvc layout the views are generally all kept within the view folder where they then would have a controller dedicated with the similar name as the view within the controller folder. A Front controller would point when a request comes in to display the view whilst any requests from that view would typically go to the page controller of the similar name. [8]

2.3 Controller

The controller handles the data in the application [2].

The controller will receive the user input, construct the appropriate model, and then pass that data to the view to be displayed to the user [6]. The controller is usually where browser requests get handled. There are some patterns that the controllers may be set up in.

A front controller would be a controller that is set up to handle all of the requests for the website pages. The front controller would also then handle common tasks to the web application that would include session handling and input filtering, as well as catching. In order to be truly loosely coupled a layer of abstraction between the controllers input and processes may be necessary.

Page controllers would then be located typically in the controller folder for each page that requires one. These page controllers handle the requests that come from the view that the controller is responsible for, and processes the requests as necessary for the model.[8]

Having a front controller responsible for the logic of the actions allows for a centralized control to navigate the users direction through the website. This can be used subsequently to restrict and guide the user, whilst catching any attempts to visit pages without authorization or without going through the correct views to do so. This creates both a boost to security as well as control, the actions allow the changing of the address to be customised as well to give error feedback as well as encapsulating the names of view pages.

2.4 Template pattern

Fowler also leads on to another web application development pattern, the template pattern. This pattern was developed in order to create a form of inversion of control, what this means is that rather than the code being statically run it instead creates an opportunity for the code to change depending on the information gathered by the code during run time. This makes it possible to create run time behavioural changes. This kind of pattern is often used when building automation code that can dynamically respond to changes based on circumstance at runtime.

The way that the template pattern works is by creating an abstract class that is a placeholder for subclasses, the two subclasses would have functions that are similar and in the same order but do different things the abstract class would then group these by putting them under the same signature to be called depending on runtime conditions. This is a safer way of runtime polymorphism, where rather than polymorphic overriding which would have a bigger impact on the code, the change is done with small controlled impact.



3.Symfony2

Symfony2 is a framework that is flexible and provides developers with tools for creating web applications. Whilst symfony2 is like many other frameworks a mvc web framework that creates a starting point for developers to use tried and tested common solutions whilst remaining loosely coupled, it separates itself from other frameworks. Symfony2 Does this by only giving tools and integrations for the views and controllers whilst allowing the developers to have the freedom of choosing to build their own models. Symfony2 makes a point on insisting not to create any dependencies on any ORM whilst still giving the developers the freedom to choose an independent ORM to plugin.[7]

Symfony2 reduces the need for testing by providing a set of solutions to common web application problems readily available and thoroughly tested. Alongside this symfony2 has available plug in such as twig which is the server side php template, this is fully unit tested, secure, and documented.

Symfony2 also allows for the integration of a NoSQL database such as MongoDB, this type of database can be popular in web architecture due to its cheaper set up costs and fast

performance. MongoDB can be set up with Symfony2 with the use of the Doctrine ORM plugin.

The advantages of using symfony2 as a framework is that due to the unit tested solutions that it provides to common problems it can cut the time required for a development stage, giving the developers both the freedom of a non dependant application as well as a head start in the development stage. This reduces cost for the company whilst still maintaining a highly complex and competent coded application. In a comparison to other frameworks such as zend and symfony(v1.4) it takes up less memory and is faster.

Symfony2 also maintains a level of reusability that further allows it to cut development time. Despite having reusable code and a fast framework symfony2 still gives the developers the freedom and flexibility of meeting their clients unique needs thanks to the loosely coupled nature of the framework and the lack of dependency on a ORM.[7]

References

- [1] G. Yoon, K. Kim and K. Lee, "Design and implementation of geo-spatial image processing system using OGC WPS 2.0 and Web framework on Openstack cloud," in *2016 4th International Workshop on Earth Observation and Remote Sensing Applications (EORSA)*, Guangzhou, 2016.
 - [2] A. Singh, P. Chawla, K. Singh and A. K. Singh, "Formulating an MVC Framework for Web Development in JAVA," in *2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI)*, Tirunelveli, 2018.
 - [3] X. Li, L. Huang, Z. Lin and Z. Sai, "Research of Routing System which applied in ASP.NET MVC Application," in *The 2009 International Symposium on Computer Science and Computational Technology*, 2009.
 - [4] P. Gupta and M. Govil, "MVC Design Pattern for the Multi Framework Distributed Applications Using XML," *Spring and Struts Framework IJCSE*, vol. 2, no. 4, pp. 1047-1051, 2010.
 - [5] X. Li and N. Liu, "Research on L-MVC Framework," in *2016 17th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT)*, Guangzhou, 2016.
 - [6] W. Romsaiyud, "Applying MVC data model on hadoop for delivering the business intelligence," in *2014 Twelfth International Conference on ICT and Knowledge Engineering*, Bangkok, 2014.
 - [7] Q. J. Hu, "The research on intelligent system of digital publishing based on the framework of Symfony2.0," in *2014 IEEE 5th International Conference on Software Engineering and Service Science*, Beijing, 2014.
 - [8] I. Sarker and K. Apu, "MVC Architecture Driven Design and Implementation of Java Framework for Developing Desktop Application", *International Journal of Hybrid Information Technology*, vol. 7, no. 5, pp. 317-322, 2014. Available: 10.14257/ijhit.2014.7.5.29 [Accessed 30 March 2019].
-

FINAL GRADE

35/100

GENERAL COMMENTS

Instructor

Investigation of Fowler (30) 22

Investigation of Symfony (20) 13

Quality (20) 0

View handling (10) 0

Database handling (15) 0

Software testing (5) 0

LATE CAPPED 40

In discussing Fowler, I liked the explanation of MVC you provided but more evaluation of the benefits/drawbacks of front and page controllers should have been provided.

The discussion of Symfony was a bit vague, and could have done with some examples and more detailed review of the technology.

No code provided.

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5



Comment 1

Actually, in the context the Template View is approach I was expecting you to discuss. However, this pattern may be useful in web development so I will give you marks.