

Report

by Josh Benjamin Rowland

FILE	2664_JOSH_BENJAMIN_ROWLAND_REPORT_93284_1706511818.DOCX		
TIME SUBMITTED	22-MAR-2018 11:39AM (UTC+0000)	WORD COUNT	1198
SUBMISSION ID	85299898	CHARACTER COUNT	6458

Josh Rowland

P145133

Advanced Web

Advanced Web part 2

Michael Dacey

Part A

The MVC or model-view-controller framework is primarily used to separate components of the web application from the business logic, this applies to any form of MVC framework regardless of the coding language[1]. The page controller pattern can be used in many different scenarios such as improving the reuse of logic across several pages, reducing code duplication from reading and rendering data from a source or when in dynamic web pages where actions can lead to different controller logic[1]. By using the page controller we can accept input from a page then invoke the necessary actions on the model and determine the appropriate view to display the results while also reducing duplicate code and increasing the consistency and testability of the code[1]. The front controller has a similar concept to the page controller as they both aim to reduce code duplication however the front controller also aims for reuse and flexibility which is why they are used in situations such as when you need to centralize the common logic if it is going to be replicated at other points in the application which also improves maintainability of the application, when retrieving data it should be in one place on the front controller not scattered across views and it allows quick and easy testing as the roles are separated and more organised[2]. The developer will want to use a front controller over a page controller in situations where a large application may require the page controllers base class behaviour to extend with more code as time goes on which can make it messy and hard to maintain or when the developer wishes to avoid excessive conditional logic as two examples[2]. To summarise the front controller is used to solve the decentralization problem that exists in page controller by having all the request pass through the single controller[2]. These patterns integrate into MVC because they are both forms of controllers which are key in the MVC architecture as it manages the relationship between the view and model as they responded to input which changes the models output so by using the page and front controllers you can manipulate and control the flow of information on the application while making it easier to maintain, reduce code duplication, increase testability and in the application there should only ever be one front controller that has to be configured into the web server which handles the dispatching which is a massive advantage for a web server that is difficult to configure[3]. Flower talks about template views, saying that they allow the developer to write the presentation and embed markers on the page where dynamic content will be placed however despite its usefulness it can lead to very disorganised code that is very difficult to maintain[3]. The main use of a template view is to use the markers as they will be replaced by the results of a service request which is why it is a popular choice due to its effectiveness and simplicity when use as a server page which allows the developer to build on it even more by adding programming logic into the page[3]. The main reason people use the template view is because it allows the developer to build the content of the page by simply looking at the page structure however it is bad as its very easy to put complicated code into the page which makes maintenance harder the more code that is added, but also that it can be difficult to test as they are designed to work within a web server[3].

Frameworks

There are many PHP frameworks available for building complex applications such as Symfony or the Laravel ecosystem which both aim to make the development process of web applications quicker and easier as it will have code and pages such as an authentication form

which may take several days to develop readily available to add to the application saving massive amounts of time by having these generic components ready for the developer[4][5]. Due to frameworks like the ones mentioned above, making PHP development faster, quicker, more secure and more maintainable they are highly beneficial to developers as it will mean less mistakes while getting development completed faster. In Laravel's case it allows code reuse easily, creation of database relations are easy, many advanced tools and clear documentation however it is slow and has a large learning curve for new developers because the community is small and methods in reverse routing can be complex[6]. Symfony on the other hand has fantastic flexibility and a large ecosystem of components that can greatly improve the web application however file parsing can be difficult and the security feature can be difficult to use for new developers to Symfony[6]. The use of MVC in PHP has the same benefits and drawbacks as it would with any language or framework as the benefits and drawbacks are because of the architecture itself, not the language or framework implemented[7]. The benefits and drawbacks of the MVC architecture are:[7]

- Faster development as there could be one developer working on the views, another the model, and another the controller which would be even faster when using frameworks such as those mentioned above.
- You can have multiple views for the model as MVC separates concerns and reduces code reuse.
- Modification of code won't affect the whole model as the model does not depend on the views.
- MVC supports the use of the asynchronous technique which helps the developers create an application that has minimal load durations.
- The model returns unformatted data which means that the same components can be used with any interface.

However MVC does have its drawbacks such as:[7]

- The knowledge of multiple technologies and techniques is needed to get the most from MVC.
- Best with multiple developers so they can work on a separate part each.
- It can be very complex and difficult to develop when first learning the architecture compared to other methods.

Just by comparing the above it is obvious that the MVC pattern is incredibly useful however it should only be used in application where there should be a clear separation of concerns but it can be confusing for many when it is the right time to use MVC and when it isn't as mentioned by Fowler who says that MVC shouldn't been seen as a pattern because it contains so many different ideas and because the different people who read about MVC and take different ideas from it and then describe said ideas as MVC it causes confusion about what MVC truly is, he compares it to Chinese whispers[8].

References

- [1] "Page Controller." [Online]. Available: <https://msdn.microsoft.com/en-us/library/ff649595.aspx>. [Accessed: 14-Mar-2018].
- [2] "Front Controller." [Online]. Available: <https://msdn.microsoft.com/en-us/library/ff648617.aspx>. [Accessed: 14-Mar-2018].
- [3] M. Fowler *et al.*, "Patterns of Enterprise Application Architecture," 2002.
- [4] "What is Symfony." [Online]. Available: <http://symfony.com/what-is-symfony>. [Accessed: 15-Mar-2018].
- [5] "Laravel - The PHP Framework For Web Artisans." [Online]. Available: <https://laravel.com/>. [Accessed: 15-Mar-2018].
- [6] "Advantages and Disadvantages of most Favoured PHP Frameworks." [Online]. Available: <https://www.agriya.com/blog/pros-cons-favoured-php-frameworks/>. [Accessed: 15-Mar-2018].
- [7] "What is MVC? Advantages and Disadvantages of MVC - Interserver Tips." [Online]. Available: <https://www.interserver.net/tips/kb/mvc-advantages-disadvantages-mvc/>. [Accessed: 13-Jan-2018].
- [8] "GUI Architectures." [Online]. Available: <https://martinfowler.com/eaDev/uiArchs.html>. [Accessed: 19-Mar-2018].

Report

GRADEMARK REPORT

FINAL GRADE

67 / 100

GENERAL COMMENTS

Instructor

Investigation of Fowler (30) 25

Investigation of MVC frameworks (20) 12

Quality of website and techniques used (25) 14

View handling (10) 6

Database handling (15) 10

Good discussion of Fowler, although the template view analysis is a bit weak.

The evaluation of MVCs benefits and drawbacks is fine, but you haven't really compared the frameworks in any detail.

You have provide an application that allows user details to be entered and displays them. This demonstrates a basic understanding of the MVC framework provided.

PAGE 1

PAGE 2



Comment 1

You needed to expand on this to clarify what you mean here

PAGE 3

PAGE 4
