Should I migrate to ASP.NET MVC?

Asked 16 years, 3 months ago Modified 12 years, 5 months ago

Viewed 10k times

52

I just listened to the StackOverflow team's 17th podcast, and they talked so highly of ASP.NET MVC that I decided to check it out.

But first, I want to be sure it's worth it. I already created a base web application (for other developers to build on) for a project that's starting in a few days and wanted to know, based on your experience, if I should take the time to learn the basics of MVC and re-create the base web application with this model.

Are there really big pros that'd make it worthwhile?

EDIT: It's not an existing project, it's a project about to start, so if I'm going to do it it should be now...

I just found this

> It does not, however, use the existing post-back model for interactions back to the server.
> Instead, you'll route all end-user interactions to a Controller class instead - which helps ensure clean separation of concerns and testability (**it**

> **also means no viewstate or page lifecycle with MVC based views**).

How would that work? No viewstate? No events?

asp.net-mvc

Share

Improve this question

Follow

edited Jul 13, 2012 at 6:40

tereško

58.4k ● 25 ● 100 ● 150

asked Aug 27, 2008 at 13:20

juan

81.8k ● 52 ● 164 ● 198

20 Answers

Sorted by: Highest score (default)

65

If you are quite happy with WebForms today, then maybe ASP.NET MVC isn't for you.

I have been frustrated with WebForms for a really long time. I'm definitely not alone here. The smart-client, stateful abstraction over the web breaks down severely in complex scenarios. I happen to love HTML, Javascript, and CSS. WebForms tries to hide that from me. It also has some really complex solutions to problems that are really not that complex. Webforms is also inherently difficult to test, and while you can use MVP, it's not a

great solution for a web environment...(compared to MVC).

MVC will appeal to you if... - you want more control over your HTML - want a seamless ajax experience like every other platform has - want testability through-and-through - want meaningful URLs - HATE dealing with postback & viewstate issues

And as for the framework being Preview 5, it is quite stable, the design is mostly there, and upgrading is not difficult. I started an app on Preview 1 and have upgraded within a few hours of the newest preview being available.

Share Improve this answer

Follow

answered Sep 9, 2008 at 21:00

Ben Scheirman

40.9k ● 21 ● 103 ● 139

30

It's important to keep in mind that MVC and WebForms are not competing, and one is not better than the other. They are simply different tools. Most people seem to approach MVC vs WebForms as "one must be a better hammer than the other". That is wrong. One is a hammer, the other is a screwdriver. Both are used in the process of putting things together, but have different strengths and weaknesses.

If one left you with a bad taste, you were probably trying to use a screwdriver to pound a nail. Certain problems are cumbersome with WebForms that become elegant and simple with MVC, and vice-versa.

Share Improve this answer

Follow

answered Aug 27, 2008 at 17:09

Rex M

144k ● 34 ● 291 ● 315

Right on Rex. This should have be the recurring answer.
– David Apr 26, 2009 at 3:34

I can't see an advantage of WebForms that MVC doesn't have. What can I do with WebForms better than MVC?
– gdoron Jan 26, 2012 at 21:10

1 Rapid Application development – Sai Avinash Sep 3, 2013 at 14:54

11

I have used ASP.NET MVC (I even wrote a HTTPModule that lets you define the routes in web.config), and I still get a bitter taste in my mouth about it.

It seems like a giant step backwards in organization and productivity. Maybe its not for some, but I've got webforms figured out, and they present no challenge to me as far as making them maintainable.

That, and I don't endorse the current "TEST EVERYTHING" fad...

Share Improve this answer

Follow

answered Aug 27, 2008 at 14:35

FlySwat

175k ● 75 ● 248 ● 314

11

ASP.NET MVC basically allows you to separate the responsibility of different sections of the code. This enable you to test your application. You can test your Views, Routes etc. It also does speed up the application since now there is no ViewState or Postback.

BUT, there are also disadvantages. Since, you are no using WebForms you cannot use any ASP.NET control. It means if you want to create a GridView you will be running a for loop and create the table manually. If you want to use the ASP.NET Wizard in MVC then you will have to create on your own.

It is a nice framework if you are sick and tired of ASP.NET webform and want to perform everything on your own. But you need to keep in mind that would you benefit from creating all the stuff again or not?

In general I prefer Webforms framework due to the rich suite of controls and the automatic plumbing.

Share Improve this answer

Follow

answered Sep 9, 2008 at 20:50

azamsharp

20.1k ● 38 ● 146 ● 228

9

I would create a test site first, and see what the team thinks, but for me I wouldn't go back to WebForms after using MVC.

Some people don't like code mixed with HTML, and I can understand that, but I far prefer the flexibility over things like Page Lifecycle, rendering HTML and biggy for me - no viewstate cruft embedded in the page source.

Some people prefer MVC for better testibility, but personally most of my code is in the middle layer and easily tested anyway...

Share Improve this answer

Follow

answered Aug 27, 2008 at 13:31

JamesSugrue

15k ● 10 ● 62 ● 93

6

@Juan Manuel Did you ever work in classic ASP? When you had to program all of your own events and "viewstatish" items (like a dropdown recalling its selected value after form submission)?

If so, then ASP.NET MVC will not feel that awkward off the bat. I would check out Rob Conery's Awesome Series "MVC Storefront" where he has been walking through the framework and building each expected component for a storefront site. It's really impressive and easy to follow along (catching up is tough because Rob has been reall active and posted A LOT in that series).

Personally, and quite contrary to Jeff Atwood's feelings on the topic, I rather liked the webform model. It was totally different than the vbscript/classic ASP days for sure but keeping viewstate in check and writing your own CSS friendly controls was enjoyable, actually.

Then again, note that I said "liked". ASP.NET MVC is really awesome and more alike other web technologies out there. It certainly is easier to shift from ASP.NET MVC to RAILS if you like to or need to work on multiple platforms. And while, yes, it is very stable obviously (this very site), if your company disallows "beta" software of any color; implementing it into production at the this time might be an issue.

Share Improve this answer Follow

edited Aug 27, 2008 at 19:48

answered Aug 27, 2008 at 13:57

Ian Patrick Hughes
10.4k ● 3 ● 32 ● 37

5

@Jonathan Holland I saw that you were voted down, but that is a VERY VALID point. I have been reading some posts around the intertubes where people seem to be confusing ASP.NET MVC *the framework* and MVC *the pattern*.

MVC in of itself is a **DESIGN PATTERN**. If all you are looking for is a "separation of concerns" then you can

certainly achieve that with webforms. Personally, I am a big fan of the MVP pattern in a standard n-tier environment.

If you really want TOTAL control of your mark-up in the ASP.NET world, then MVC the ramework is for you.

Share Improve this answer

Follow

answered Aug 27, 2008 at 15:44

Ian Patrick Hughes

10.4k ● 3 ● 32 ● 37

5

If you are a professional ASP.NET developer, and have some time to spare on learning new stuff, I would certainly recommend that you spend some time trying out ASP.NET MVC. It may not be the solution to all your problems, and there are lots of projects that may benefit more from a traditional webform implementation, but while trying to figure out MVC you will certainly learn a lot, and it might bring up lots of ideas that you can apply on your job.

One good thing that I noticed while going through many blog posts and video tutorials while trying to develop a MVC pet-project is that most of them follow the current best practices (TDD, IoC, Dependency Injection, and to a lower extent POCO), plus a lot of JQuery to make the experience more interesting for the user, and that is stuff that I can apply on my current webform apps, and that I wasn't exposed in such depth before.

The ASP.NET MVC way of doing things is so different from webforms that it will shake up a bit your mind, and that for a developer is very good!

OTOH for a total beginner to web development I think MVC is definitely a better start because it offers a good design pattern out of the box and is closer to the way that the web really works (HTML is stateless, after all). On MVC you decide on every byte that goes back and forth on the wire (at least while you don't go crazy on html helpers). Once the guy gets that, he or she will be better equipped to move to the "artificial" facilities provided by ASP.NET webforms and server controls.

Share Improve this answer Follow

edited Nov 24, 2008 at 22:39

answered Nov 24, 2008 at 22:27

rodbv

5,254 ● 4 ● 33 ● 31

> Sounds like that's exactly what Classic ASP did, you code everything yourself, all the HTML, all the DOM manipulation with Javascript, etc, no viewstate, server controls, etc... back to the future anyone? – joedotnot Dec 16, 2009 at 5:47

5

If you like to use server controls which do a lot of work for you, you will NOT like MVC because you will need to do a lot of hand coding in MVC. If you like the GridView, expect to write one yourself or use someone else's.

MVC is not for everyone, specially if you're not into unit testing the GUI part. If you're comfortable with web forms, stay with it. Web Forms 4.0 will fix some of the current shortcomings like the ID's which are automatically assigned by ASP.NET. You will have control of these in the next version.

Share Improve this answer

Follow

answered Feb 16, 2009 at 21:06

Abdu

16.5k ● 13 ● 63 ● 84

4

Unless the developers you are working with are familiar with MVC pattern I wouldn't. At a minimum I'd talk with them first before making such a big change.

Share Improve this answer

Follow

answered Aug 27, 2008 at 13:24

Booji Boy

4,582 ● 4 ● 41 ● 45

> Actually, the new ASP.NET MVC framework abstracts most of the MVC "pattern" stuff away from the developer. My $0.02 (albeit unsolicited). – TheHolyTerrah Mar 25, 2009 at 19:35

4

I'm trying to make that same decision about ASP.NET MVC, Juan Manuel. I'm now waiting for the right bite-sized project to come along with which I can experiment. If the experiment goes well--my gut says it will--then I'm going to architect my new large projects around the framework.

With ASP.NET MVC you lose the viewstate/postback model of ASP.NET Web Forms. Without that abstraction, you work much more closely with the HTML and the HTTP POST and GET commands. I believe the UI programming is somewhat in the direction of classic ASP.

With that inconvenience, comes a greater degree of control. I've very often found myself fighting the psuedo-session garbage of ASP.NET and the prospect of regaining complete control of the output HTML seems very refreshing.

It's perhaps either the best--or the worst--of both worlds.

Share Improve this answer

Follow

edited May 23, 2017 at 10:29

Community Bot

1 ● 1

answered Aug 27, 2008 at 13:41

Zack Peterson

57.3k ● 80 ● 210 ● 280

4

5 Reasons You Should Take a Closer Look at ASP.NET MVC

Share Improve this answer

Follow

answered Apr 7, 2009 at 12:17

Konstantin Tarkus

38.3k ● 14 ● 136 ● 121

2

> I dont´t know ASP.NET MVC, but I am very familiar with MVC pattern. I don´t see another way to build professional applications without MVC. And it has to be MVC model 2, like Spring or Struts. By the way, how you people were building web applications without MVC? When you have a situation that some kind of validation is necessary on every request, as validating if user is authenticated, what is your solution? Some kind of include(validate.aspx) in every page?

Have you never heard of N-Tier development?

Share Improve this answer

Follow

answered Aug 27, 2008 at 14:31

FlySwat

175k ● 75 ● 248 ● 314

2

Ajax, RAD (webforms with ajax are anti-RAD very often), COMPLETE CONTROL (without developing whole bunch of code and cycles). webforms are good only to bind some grid and such and not for anything else, and one more really important thing - performance. when u get stuck into the web forms hell u will switch on MVC sooner or later.

Share Improve this answer

Follow

answered Jun 11, 2009 at 17:52

Vjeran

141 ● 1 ● 1 ● 6

1

I wouldn't recommend just making the switch on an existing project. Perhaps start a small "demo" project that the team can use to experiment with the technology and (if necessary) learn what they need to and demonstrate to management that it is worthwhile to make the switch. In the end, even the dev team might realize they aren't ready or it's not worth it.

Whatever you do, be sure to document it. Perhaps if you use a demo project, write a postmortem for future reference.

Share Improve this answer

Follow

answered Aug 27, 2008 at 13:27

Thomas Owens

116k ● 99 ● 317 ● 436

1

I dont´t know ASP.NET MVC, but I am very familiar with MVC pattern. I don´t see another way to build professional applications without MVC. And it has to be MVC model 2, like Spring or Struts. By the way, how you people were building web applications without MVC? When you have a situation that some kind of validation is necessary on every request, as validating if user is authenticated, what is your solution? Some kind of include(validate.aspx) in every page?

Share Improve this answer

Follow

answered Aug 27, 2008 at 13:35

Ivan Bosnic

1,996 ● 5 ● 21 ● 32

1

No, you shouldn't. Feel free to try it out on a new project, but a lot of people familiar with ASP.NET webforms aren't loving it yet, due to having to muck around with raw HTML + lots of different concepts + pretty slim pickings on documentation/tutorials.

Share Improve this answer

Follow

answered Aug 27, 2008 at 13:46

ryw

9,635 ● 5 ● 29 ● 34

I have to disagree with the lack of tutorials, sorry. www.Asp.net/MVC has a good a mount of tutorials that when put together cover pretty much all the pieces to the puzzle. I didn't vote you down because I can't argue against the raw HTML part that's for sure, of course it offers more control and normal un tampered tag id's no more ctl00_ etc... – David Apr 26, 2009 at 3:30

0

Is the fact that ASP.net MVC is only in 'Preview 5' be a cause for concern when looking into it?

I know that StackOverflow was created using it, but is there a chance that Microsoft could implement significant changes to the framework before it is officially out of beta/alpha/preview release?

Share Improve this answer

Follow

answered Aug 27, 2008 at 14:22

Matthew Ruston

4,322 ● 7 ● 39 ● 47

0

If you are dead set on using an MVC framework, then I would rather set out to use Castle project's one...

When that's said I personally think WebControls have a lot of advantages, like for instance being able to create event driven applications which have a stateful client and so on. Most of the arguments against WebControls are constructed because of lack of understanding the WebControl model etc. And not because they actually are truly bad...

MVC is not a Silver Bullet, especially not Microsoft MVC...

Share Improve this answer

Follow

answered Nov 25, 2008 at 12:46

Thomas Hansen

5,513 ● 1 ● 25 ● 28

3 I guess that Jeff Atwood, who built this very site using ASP.NET MVC, would beg to differ ;) As long as people don't see webforms and ASP.NET MVC as mutually exclusive skills, but 2 good tools that every pro ASP.NET dev should have in their toolboxes, it's all fine. The more tools the better! – rodbv Nov 25, 2008 at 21:42

0

I have seen some implementation of MVC framework where for the sake of testability, someone rendered the whole HTML in code. In this case the view is also a testable code. But I said, my friend, putting HTML in code is a maintenance nightmare and he said well I like everything compiled and tested. I didn't argue, but later found that he did put this HTML into resource files and the craziness continued...

Little did he realized that the whole idea of separating View also solved the maintenance part. It outweighs the testability in some applications. We do not need to test the HTML design if we are using WYSWYG tool. WebForms are good for that reason.

I have often seen people abusing postback and viewstate and blaming it on the ASP .NET model.

Remember the best webpages are still the .HTMLs and that's where is the Power of ASP .NET MVC.

Share Improve this answer

Follow

answered Feb 27, 2009 at 21:51

Otpi

141 ● 2 ● 2 ● 5