ASP.NET MVC: Structuring Controllers

Asked 16 years, 3 months ago Modified 5 years, 9 months ago

Viewed 1k times

10

So I'm embarking on an `ASP.NET MVC` project and while the experience has been a good one overall, I'm not quite as pleased with the spaghetti mess that my controllers have become. I've looked around online (CodeCampServer, etc...) and they all seem to suffer the same issue wherein controller methods violate SRP (single responsibility principle) pretty consistently - such as a controller method that simply renders the view if the request is a `GET` but updates the `model` if it's a `POST`. Now I've got controller methods responsible for multiple logical routes throughout the application - say it checks for which `button` was clicked on the form and acts accordingly. I could redirect each button click to a different form action using JavaScript, but something doesn't feel right there either... The other big issue is the proliferation of magic `strings - ViewData["foo"] = blah;` Long story short, how do you guys `structure` your `controller` logic? One giant model object per view? Lots of little `controller methods` and `JavaScript` is the router? My goal is maintainable code - as features get piled on I'm starting to slide down that slippery slope...

c# asp.net-mvc

Share

Improve this question

Follow

edited Mar 19, 2019 at 10:19

ArunPratap
5,010 ● 7 ● 28 ● 45

asked Aug 30, 2008 at 17:44

matt
1,186 ● 1 ● 10 ● 13

2 Answers

Sorted by: Highest score (default)

9

ASP.NET Preview 5 (available on CodePlex) has an answer for this: the [AcceptVerbs] attribute. Phil Haack has a blog post discussion how it's used.

As for the view data magic key question, it's an interesting problem. If you think of a view as being a bunch of semi-independent components (especially in light of the new partial view support), then making a strongly-typed model becomes less ideal, as the several pieces of the view should be relatively independent of one another.

Share Improve this answer

Follow

edited Aug 30, 2008 at 18:59

answered Aug 30, 2008 at 18:52

Brad Wilson
70.4k ● 9 ● 77 ● 85

0

How are different people handling this issue? I know that i just spent a couple hours reviewing the jumble inside of the model folder. I'm finding creating folders is helpful in reducing the visual clutter, using matching namespaces helps alot too.

But my controllers are monoliths at the moment. the trouble is that i've been focused on learning to this point in the project (still lots to sort out as well).

I'm getting a good handle on MVC now, so it is time to review the complexity and consider modifying the controllers up into better named and cleaner functions.

Are other people breaking their controllers up into sub controllers? (If there is such a thing)

Share Improve this answer Follow

answered Nov 13, 2008 at 13:12
Andrew Harry
13.9k ● 18 ● 71 ● 102

The trick to coding your controllers is to look at them and say 'if each action method is getting over 20 or 30 lines or some relatively small number, how can i reduce it?', Basically, keep it DRY and rethink what you are doing and move that logic into some proper SERVICE layer, which can be reused.
– Pure.Krome Nov 13, 2008 at 13:16

Also, download ROB CONERY's StoreFront MVC Starter Kit and see how he's layered his solution -> very uber awesomesauce. (google it, i can't be bothered finding the link at this late (early?!) hour of the morning... hth! – Pure.Krome Nov 13, 2008 at 13:18