what is the difference between a view model and a data transfer object?

Asked 10 years, 5 months ago Active 4 years ago Viewed 13k times

49

20

I'm basing this question on Fowler PoEAA. Given your familiarity with this text, aren't the ViewModels used in ASP.NET MVC the same as DTOs? Why or why not? Thank you.

asp.net-mvc design-patterns software-design poeaa

edited Jan 19 '16 at 19:07
andrew.fox
5,038 4 41 59

asked Sep 16 '09 at 7:13

mkelley33
4,758 7 41 64

4 Answers

83

They serve a similar purpose (encapsulating data for another layer of the application) but they do it differently and for different reasons.

- The purpose of a DTO is to reduce the number of calls between tiers of an application, especially when those calls are expensive (e.g. distributed systems). DTOs are almost always trivially serializable, and almost never contain any behaviour.

  For example, you're developing an e-Commerce site. `CreateCustomer` and `AddCustomerAddress` are separate operations at the database level, but you might for performance reasons want to aggregate their data into a `NewCustomerWithAddressDto` so that your client only needs to make one round-trip to the server, and doesn't need to care that the server might be doing a bunch of different things with the parcel of data.

- The term "ViewModel" means slightly different things in different flavours of MV*, but its purpose is mainly separation of concerns. Your Model is frequently optimised for some purpose other than presentation, and it's the responsibility of the ViewModel to decouple your View from the Model's implementation details. Additionally, most MV* patterns advise making your Views as "dumb" as possible, and so the ViewModel sometimes takes responsibility for presentation logic.

  For example, in the same e-Commerce application, your `CustomerModel` is the wrong "shape" for presentation on your "New Customer" View. For starters, your View has two form fields for your user to enter and confirm their password, and your `CustomerModel` doesn't contain a password field at all! Your `NewCustomerViewModel` will contain those fields and might, depending on

your flavour of MV*, be responsible for some presentation logic (e.g. to show/hide parts of the view) and basic validation (e.g. ensuring that both password fields match).

edited Feb 23 '16 at 14:08

answered Sep 16 '09 at 7:28

Iain Galloway
15.3k 4 44 71

1 That's an excellent explanation! Up until now the only view models I had seen only had getters and setters so I was like: wow that's so much like a DTO. Thanks for clearing this up for me. – mkelley33 Sep 16 '09 at 11:48

16

The purpose is different:

- DTO's are used to transfer data
- ViewModels are used to show data to an end user.

So normally ViewModels contain the presentation data, witch is in a lot of cases similar to what is in a DTO, but with some differences. Think of representation of enums, localization, currency, date formats, ... This is because normally there should be no logic in your view.

answered Sep 16 '09 at 7:18

Bavo
351 1 6

15

DTOs in MVVM and MVP are usually **Very Dumb Objects** and are basically just a bunch of property setters and getters. ViewModels on the other hand can have some behaviour.

A practical positive side effect of having DTOs is allow easier serialization. If you have a rather complex object in, say C#, you will often find yourself having to selectively turn things off that you don't want serialized. This can get rather ugly and DTOs simplify this process.

answered Sep 16 '09 at 7:52

Igor Zevaka
65.1k 23 99 124

3 +1, the key difference is that DTOs are stupid (and thus trivially serializable, which is their *job*), and ViewModels can contain logic that would have otherwise gone into your view (which is *their* job). – Iain Galloway Sep 17 '09 at 8:22

@Igor Zevaka Can you please explain what do you mean by behavior? – Mohit Shah Mar 11 '16 at 7:53

1

A View Model and a Data Transfer object has similarities and differences.

Similar: Transfer data in a record (object instance, perhaps serialized) to a receiver, whether a view or a service

Difference: A View Model is intended to be sent to a View, where it will be displayed, with formatting. A View Model also sends back data to a controller. A DTO is usually not intended for presentation. It is intended to send raw data.

answered Jan 24 '13 at 19:33

Brother Bill
11 1