ASP. NET MVC

**Action Filtering in ASP.NET MVC Applications**

* Article
* 02/21/2011
* 2 minutes to read

In ASP.NET MVC, controllers define action methods that usually have a one-to-one relationship with possible user interactions, such as clicking a link or submitting a form. For example, when the user clicks a link, a request is routed to the designated controller, and the corresponding action method is called.

Sometimes you want to perform logic either before an action method is called or after an action method runs. To support this, ASP.NET MVC provides action filters. Action filters are custom attributes that provide a declarative means to add pre-action and post-action behavior to controller action methods.

**MVC Action Filter Types**

ASP.NET MVC provides the following types of action filters:

* Authorization filter, which makes security decisions about whether to execute an action method, such as performing authentication or validating properties of the request. The [AuthorizeAttribute](https://learn.microsoft.com/en-us/previous-versions/aspnet/dd460317(v=vs.100)) class is one example of an authorization filter.
* Action filter, which wraps the action method execution. This filter can perform additional processing, such as providing extra data to the action method, inspecting the return value, or canceling execution of the action method.
* Result filter, which wraps execution of the [ActionResult](https://learn.microsoft.com/en-us/previous-versions/aspnet/dd493064(v=vs.100)) object. This filter can perform additional processing of the result, such as modifying the HTTP response. The [OutputCacheAttribute](https://learn.microsoft.com/en-us/previous-versions/aspnet/dd492556(v=vs.100)) class is one example of a result filter.
* Exception filter, which executes if there is an unhandled exception thrown somewhere in action method, starting with the authorization filters and ending with the execution of the result. Exception filters can be used for tasks such as logging or displaying an error page. The [HandleErrorAttribute](https://learn.microsoft.com/en-us/previous-versions/aspnet/dd470778(v=vs.100)) class is one example of an exception filter.

**How To Apply an Action Filter**

Typically, an action filter is an attribute that implements the abstract [FilterAttribute](https://learn.microsoft.com/en-us/previous-versions/aspnet/dd493051(v=vs.100)) class. Some action filters, such as [AuthorizeAttribute](https://learn.microsoft.com/en-us/previous-versions/aspnet/dd460317(v=vs.100)) and [HandleErrorAttribute](https://learn.microsoft.com/en-us/previous-versions/aspnet/dd470778(v=vs.100)), implement the [FilterAttribute](https://learn.microsoft.com/en-us/previous-versions/aspnet/dd493051(v=vs.100)) class directly. These action filters are always called before the action method runs.

Other action filters, such as [OutputCacheAttribute](https://learn.microsoft.com/en-us/previous-versions/aspnet/dd492556(v=vs.100)), implement the abstract [ActionFilterAttribute](https://learn.microsoft.com/en-us/previous-versions/aspnet/dd470536(v=vs.100)) class, which enables the action filter to run either before or after the action method runs.

You can use the action filter attribute to mark any action method or controller. If the attribute marks a controller, the action filter applies to all action methods in that controller.

The following example shows the default implementation of the HomeController class. In the example, the HandleError attribute is used to mark the controller. Therefore, the filter applies to both action methods in the controller.