**Filters**

Filters are basically use to perform any logic before or after an action method is being called so according to the need of project a single or multiple filters can be applied in a controller. There is a provision of built-in filter and custom filters in MVC.

Filter general meaning though which something has to pass before actual processing take place it change the flow of execution.

**List of common Built-in Action Filters**

|  |  |  |
| --- | --- | --- |
| **S.no** | **Name of Filter** | **Purpose** |
| 1. | OutputCache | It will cache the last state and proceed according to the condition specified in outputcache |
| 2. | Authorize | Only authorized user will access the action method no anonymous user are permitted |
| 3. | ValidateInput | Turn on or off request validation |
| 4. | ValidateAntiForgeryToken | Help to prevent the cross site request forgeries. |
| 5. | HandleError | Using in Excepiton handling cases |

***Example of OutputCache Action Filter***

[OutputCache(Duration=15)]

public ActionResult Index()

{

string dt = DateTime.Now.ToLongTimeString();

ViewBag.dtime = dt;

return View();

}

***Example of OutputCache Authorize Filter***

[Authorize]

public ActionResult Process()

{

return View();

}

For understanding the concept of validate input one thing very important that tags cannot be submit in the form due to high security reasons because due to tags some unauthorized data can be access using script tags by default validate input is true

But manually it can be set as false.

***Example of OutputCache ValidateInputFilter***

[HttpPost]

[ValidateInput(false)]

public ActionResult Process(string txtname)

{

ViewBag.result = txtname;

return View();

}

**Very Important ActionFilter**

**Some illegal user can copy the code from one site and paste or integrate the code in their site and can submit the data in my site so we can prevent this kind of copied by ValidateAntiForgeryToken**

**STEAL CODE**

**<form action="http://localhost:1336/Home/Process" method="post">**

**Enter Project Name <input type="text" name="txtname" />**

**<input type="submit" id="btn" value="submit" />**

**</form>**

**So it a two step process to prevent**

**Step1**

[HttpPost]

[ValidateInput(false)]

[ValidateAntiForgeryToken()]

public ActionResult Process(string txtname)

{

ViewBag.result = txtname;

return View();

}

**Step2**

<form action="/Home/Process" method="post">

@Html.AntiForgeryToken();

Enter Project Name <input type="text" name="txtname" />

<input type="submit" id="btn" value="submit" />

</form>