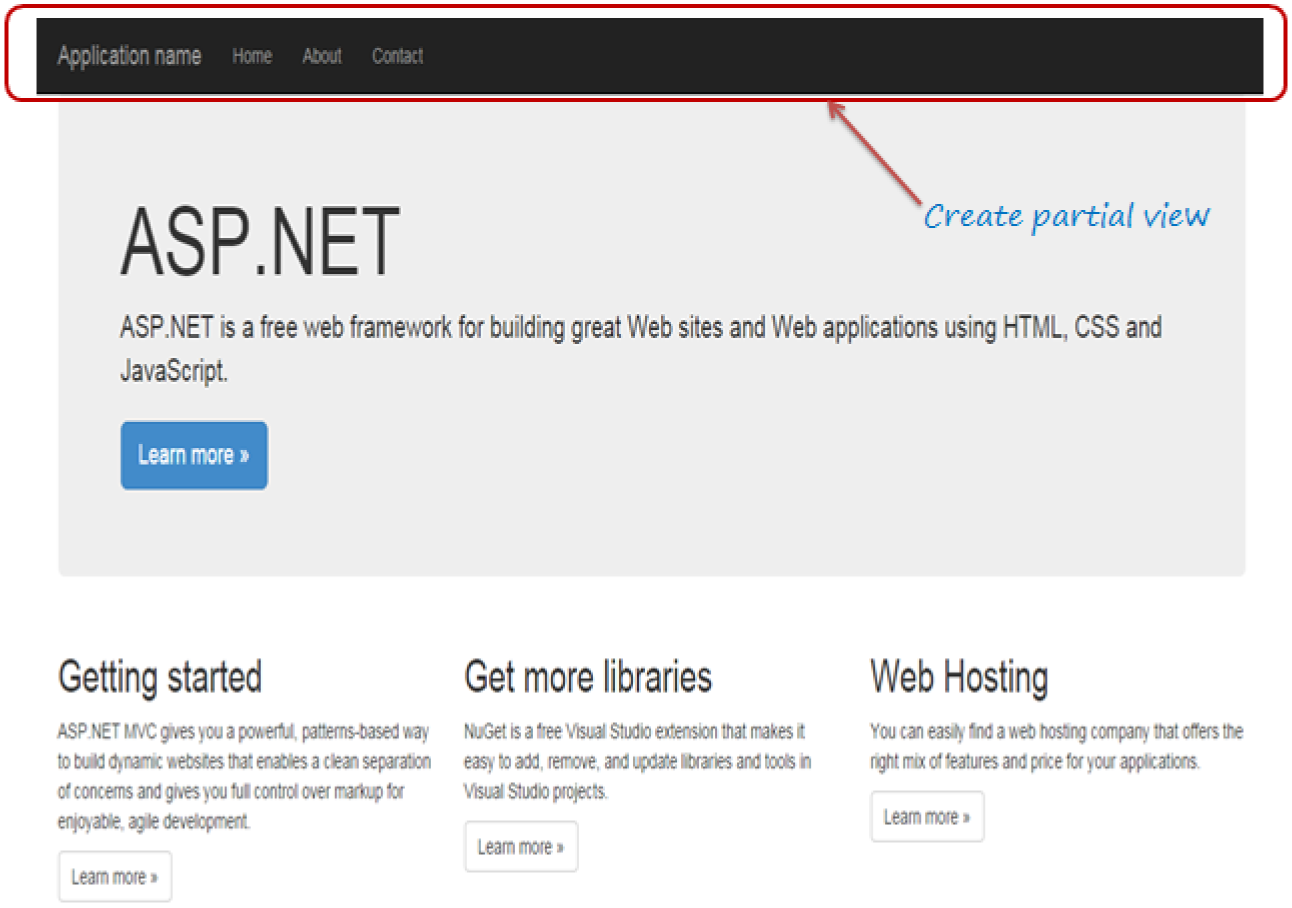
< Previous Next >

Partial View

In this section you will learn about partial views in ASP.NET MVC.

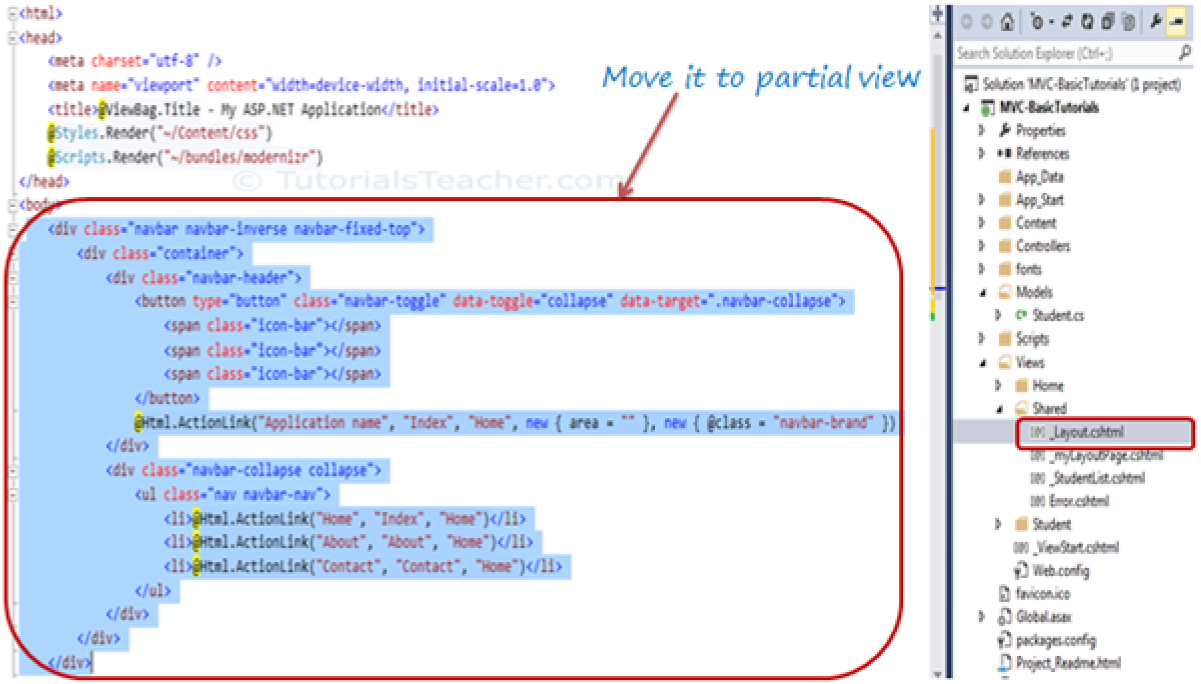
Partial view is a reusable view, which can be used as a child view in multiple other views. It eliminates duplicate coding by reusing same partial view in multiple places. You can use the partial view in the layout view, as well as other content views.

To start with, let's create a simple partial view for the following navigation bar for demo purposes. We will create a partial view for it, so that we can use the same navigation bar in multiple layout views without rewriting the same code everywhere.


Application name
Home
About
Contact
Create partial view
ASP.NET
ASP.NET is a free web framework for building great Web sites and Web applications using HTML, CSS and JavaScript.
Learn more »
Getting started
ASP.NET MVC gives you a powerful, patterns-based way to build dynamic websites that enables a clean separation of concerns and gives you full control over markup for enjoyable, agile development.
Learn more »
Get more libraries
NuGet is a free Visual Studio extension that makes it easy to add, remove, and update libraries and tools in Visual Studio projects.
Learn more »
Web Hosting
You can easily find a web hosting company that offers the right mix of features and price for your applications.
Learn more »

Partial View

The following figure shows the html code for the above navigation bar. We will cut and paste this code in a seperate partial view for demo purposes.



```
<html>
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>@ViewBag.Title - My ASP.NET Application</title>
    @Styles.Render("~/Content/css")
    @Scripts.Render("~/bundles/modernizr")
</head>
<body>
    <div class="navbar navbar-inverse navbar-fixed-top">
        <div class="container">
            <div class="navbar-header">
                <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse">
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                </button>
                @Html.ActionLink("Application name", "Index", "Home", new { area = "" }, new { @class = "navbar-brand" })
            </div>
            <div class="navbar-collapse collapse">
                <ul class="nav navbar-nav">
                    <li>@Html.ActionLink("Home", "Index", "Home")</li>
                    <li>@Html.ActionLink("About", "About", "Home")</li>
                    <li>@Html.ActionLink("Contact", "Contact", "Home")</li>
                </ul>
            </div>
        </div>
    </div>
```

Partial Views

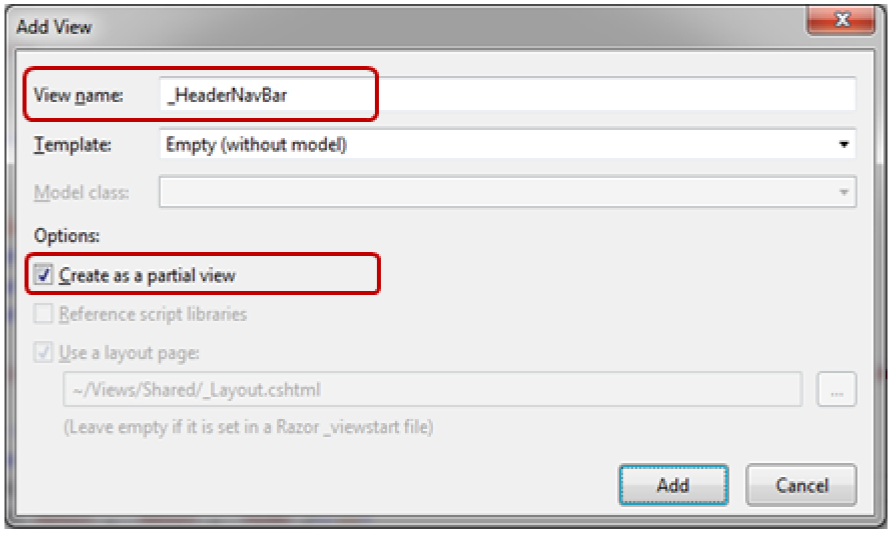
Create a New Partial View

To create a partial view, right click on Shared folder -> select **Add** -> click on **View..**

Note:

If a partial view will be shared with multiple views of different controller folder then create it in the Shared folder, otherwise you can create the partial view in the same folder where it is going to be used.

In the Add View dialogue, enter View name and select "Create as a partial view" checkbox and click Add.



View name: _HeaderNavBar
Template: Empty (without model)
Model class:
Options:
☑ Create as a partial view
☐ Reference script libraries
☑ Use a layout page:
~/Views/Shared/_Layout.cshtml
(Leave empty if it is set in a Razor _viewstart file)

Partial Views

We are not going to use any model for this partial view, so keep the Template dropdown as Empty (without model) and click **Add**. This will create an empty partial view in Shared folder.

Now, you can cut the above code for navigation bar and paste it in _HeaderNavBar.cshtml as shown below:

Example: Partial View _HeaderNavBar.cshtml

```
<div class="navbar navbar-inverse navbar-fixed-top">
    <div class="container">
        <div class="navbar-header">
                    <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-
collapse">
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
            </button>
             @Html.ActionLink("Application name", "Index", "Home", new { area = "" }, new { @class = "navbar-
brand" })
```

```
            </div>
            <div class="navbar-collapse collapse">
                <ul class="nav navbar-nav">
                    <li>@Html.ActionLink("Home", "Index", "Home")</li>
                    <li>@Html.ActionLink("About", "About", "Home")</li>
                    <li>@Html.ActionLink("Contact", "Contact", "Home")</li>
                </ul>
            </div>
        </div>
    </div>
```

Thus, you can create a new partial view. Let's see how to render partial view.

Render Partial View

You can render the partial view in the parent view using html helper methods: Partial() or RenderPartial() or RenderAction(). Each method serves different purposes. Let's have an overview of each method and then see how to render partial view using these methods.

Html.Partial()

@Html.Partial() helper method renders the specified partial view. It accept partial view name as a string parameter and returns MvcHtmlString. It returns html string so you have a chance of modifing the html before rendering.

The following table lists overloads of the Partial helper method:

Helper Method	Description
MvcHtmlString Html.Partial(string partialViewName)	Renders the given partial view content in the referred view.

Helper Method	Description
MvcHtmlString Html.Partial(string partialViewName,object model)	Renders the partial view content in the referred view. Model parameter passes the model object to the partial view.
MvcHtmlString Html.Partial(string partialViewName, ViewDataDictionary viewData)	Renders the partial view content in the referred view. View data parameter passes view data dictionary to the partial view.
MvcHtmlString Html.Partial(string partialViewName,object model, ViewDataDictionary viewData)	Renders the partial view content in the referred view. Model parameter passes the model object and View data passes view data dictionary to the partial view.

Html.RenderPartial()

The RenderPartial helper method is same as the Partial method except that it returns void and writes resulted html of a specified partial view into a http response stream directly.

Helper method	Description
RenderPartial(String partialViewName)	Renders the specified partial view
RenderPartial(String partialViewName, Object model)	Renders the specified partial view and set the specified model object
RenderPartial(String partialViewName, ViewDataDictionary viewData)	Renders the specified partial view, replacing its ViewData property with the specified ViewDataDictionary object.
RenderPartial(String partialViewName, Object model, ViewDataDictionary viewData)	Renders the specified partial view, replacing the partial view's ViewData property with the specified ViewDataDictionary object and set the specified model object

Html.RenderAction()

The RenderAction helper method invokes a specified controller and action and renders the result as a partial view. The specified Action method should return PartialViewResult using the Partial() method.

Name	Description
RenderAction(String actionName)	Invokes the specified child action method and renders the result in the parent view.
RenderAction(String actionName, Object routeValue)	Invokes the specified child action method using the specified parameters and renders the result inline in the parent view.
RenderAction(String actionName, String controllerName)	Invokes the specified child action method using the specified controller name and renders the result inline in the parent view.
RenderAction(String actionName, RouteValueDictionary routeValues)	Invokes the specified child action method using the specified parameters and renders the result inline in the parent view.
RenderAction(String actionName, String controllerName, Object routeValue)	Invokes the specified child action method using the specified parameters and controller name and renders the result inline in the parent view.
RenderAction(String actionName, String controllerName, RouteValueDictionary routeValues)	Invokes the specified child action method using the specified parameters and controller name and renders the result inline in the parent view.

So now, we can use any of the above rending methods to render the _HeaderNavBar partial view into _Layout.cshtml. The following layout view renders partial view using the RenderPartial() method.

Example: Html.RenderPartial()

```
<!DOCTYPE html>
```

```
<html>
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>@ViewBag.Title - My ASP.NET Application</title>
    @Styles.Render("~/Content/css")
    @Scripts.Render("~/bundles/modernizr")
</head>
<body>
    @{
      Html.RenderPartial("_HeaderNavBar");
    }
    <div class="container body-content">
        @RenderBody()

        <hr />
        <footer>
            <p>&copy; @DateTime.Now.Year - My ASP.NET Application</p>
        </footer>
    </div>
        @Scripts.Render("~/bundles/jquery")
        @Scripts.Render("~/bundles/bootstrap")
        @RenderSection("scripts", required: false)
</body>
</html>
```

Note:

RenderPartial returns void, so a semicolon is required at the end and so it must be enclosed in the braces.

The following layout view uses the Partial method to render partial view_HeaderNavBar.cshtml.

Example: Html.Partial()

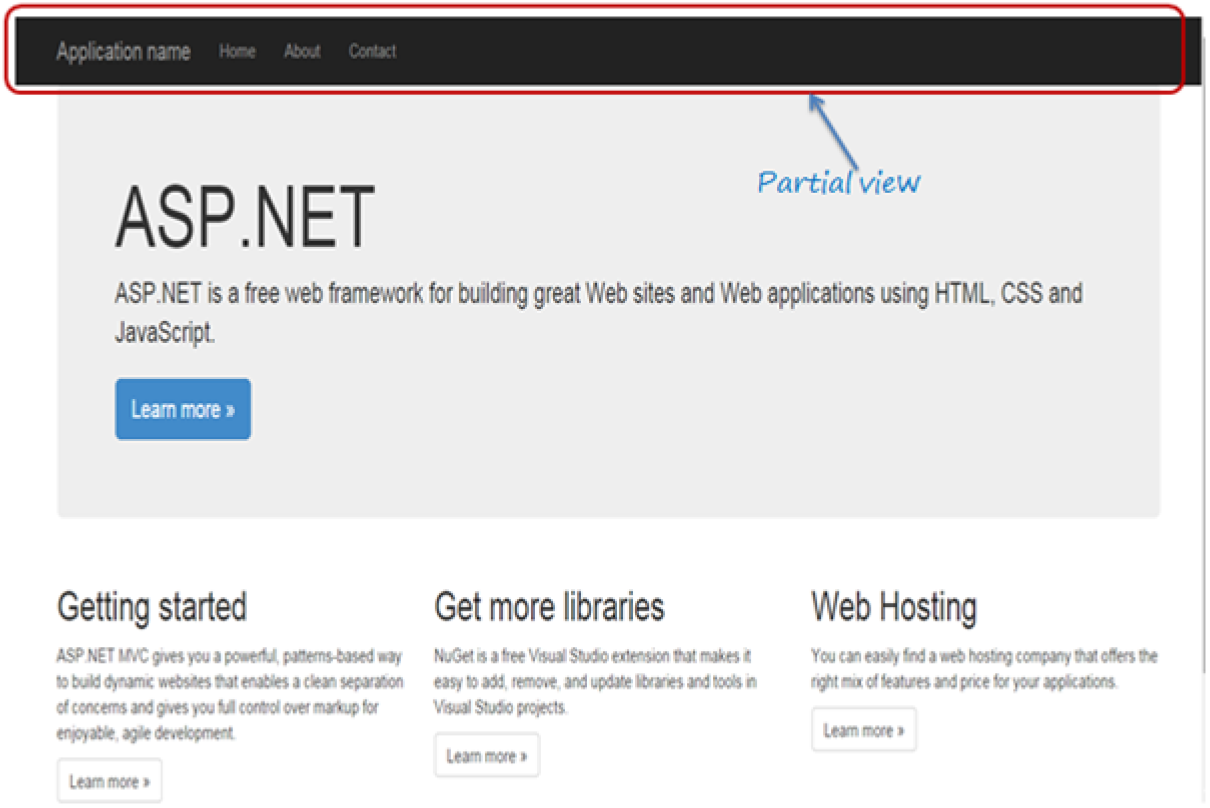
```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>@ViewBag.Title - My ASP.NET Application</title>
    @Styles.Render("~/Content/css")
    @Scripts.Render("~/bundles/modernizr")
</head>
<body>
    @Html.Partial("_HeaderNavBar")
    <div class="container body-content">
        @RenderBody()

        <hr />
        <footer>
            <p>&copy; @DateTime.Now.Year - My ASP.NET Application</p>
        </footer>
    </div>
    @Scripts.Render("~/bundles/jquery")
    @Scripts.Render("~/bundles/bootstrap")
    @RenderSection("scripts", required: false)
</body>
</html>
```

Note:

@Html.Partial() method doesn't need to be in code block because it returns a html string.

You will see following UI in browser when you run the application.



ASP.NET

ASP.NET is a free web framework for building great Web sites and Web applications using HTML, CSS and JavaScript.

Getting started

ASP.NET MVC gives you a powerful, patterns-based way to build dynamic websites that enables a clean separation of concerns and gives you full control over markup for enjoyable, agile development.

Get more libraries

NuGet is a free Visual Studio extension that makes it easy to add, remove, and update libraries and tools in Visual Studio projects.

Web Hosting

You can easily find a web hosting company that offers the right mix of features and price for your applications.

Index.cshtml

So in this way, you can use partial view without any differences in the UI.

Learn the Difference between html.Partial and Html.RenderPartial.

Points to Remember :

1) Partial view is a reusable view, which can be used as a child view in multiple other views.

2) Partial view can be rendered using Html.Partial(), Html.RenderPartial() or Html.RenderAction() method.

Share Tweet

Share Whatsapp

[Previous](#) [Next](#)

TUTORIALS

- ASP.NET Core
- ASP.NET MVC
- IoC
- Web API
- C#
- LINQ
- Entity Framework
- AngularJS 1
- Node.js
- D3.js
- JavaScript
- jQuery
- Sass
- Https

E-MAIL LIST

Subscribe to TutorialsTeacher email list and get latest updates, tips & tricks on C#, .Net, JavaScript, jQuery, AngularJS, Node.js to your inbox.

Email address GO

We respect your privacy.

HOME PRIVACY POLICY TERMS OF USE ADVERTISE WITH US