ASP.NET MVC – Part 2

Agenda

- More on Views

Tools

- Visual Studio 2013

Pre-requisite

- ASP.NET MVC – Part 1

Version	Last updated	Comments	Modified By
V1.0	08-05-2016		Bilal Shahzad

Brief Introduction

Shared folder

In Views, you can have a folder named "Shared" (by convention) and can add views in it which are shared by whole application. For example, Layouts & Error pages can be placed in "Shared" folder.

Note: You can use any name for this folder. We'll use different name of this folder (e.g. MyShared).

Layouts

Layouts are also views (HTML files) but other views can use these as Layout files. Pages in application are normally same in layout perspective. We can define shared stuff (e.g. header, footer, CSS files, JS files) in Layout file and then specific view will provide content related that page only.
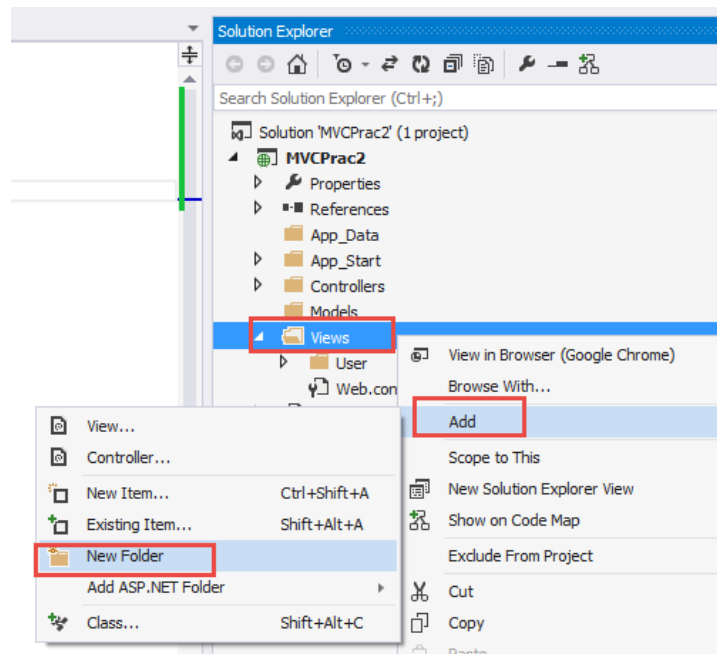
_ViewStart file

The _ViewStart file can be used to define common view code that you want to execute at the start of each View's rendering. For example, we can set our "Layout" property here. In other words, whenever a view is processed, _ViewStart file is executed before that.
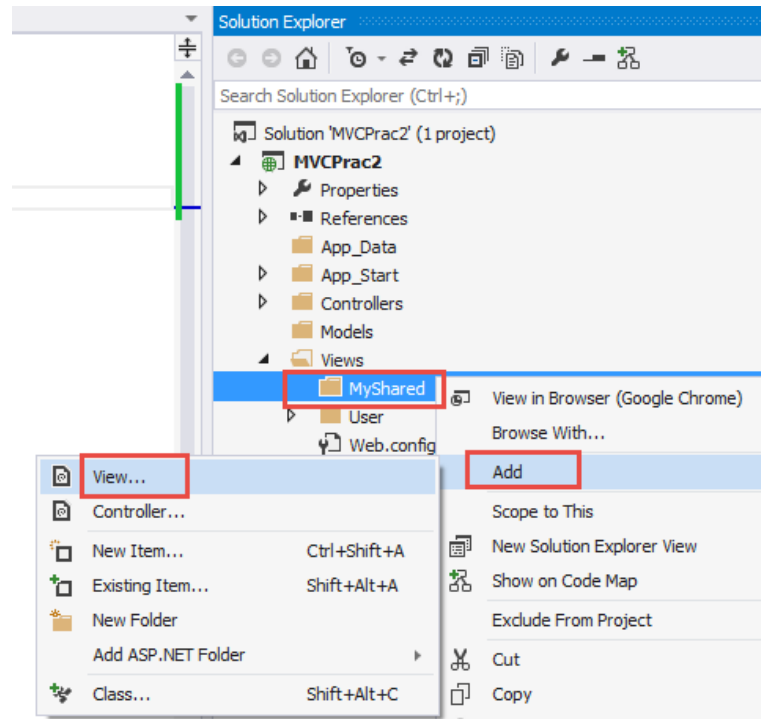
If "_ViewStart" file exists in a folder (e.g. Views/User), it will be only for views of that specific folder. In other words, when a view is process, engine searches if that folder contains any "_ViewStart" file. If yes, it executes that "_ViewStart" file and then executes the view. If no, it searches in "Views" folder if there is "_ViewStart" file. If yes, it executes that "_ViewStart" file and then executes the view.

Step by Step Walkthrough

1- Let's create a folder in "Views" to hold shared content. Right Click on Views => Add => New Folder. Name it "MyShared".

2- Now add a view (e.g. Layout) as shown below. Name it "_MyLayout" and click "Add".
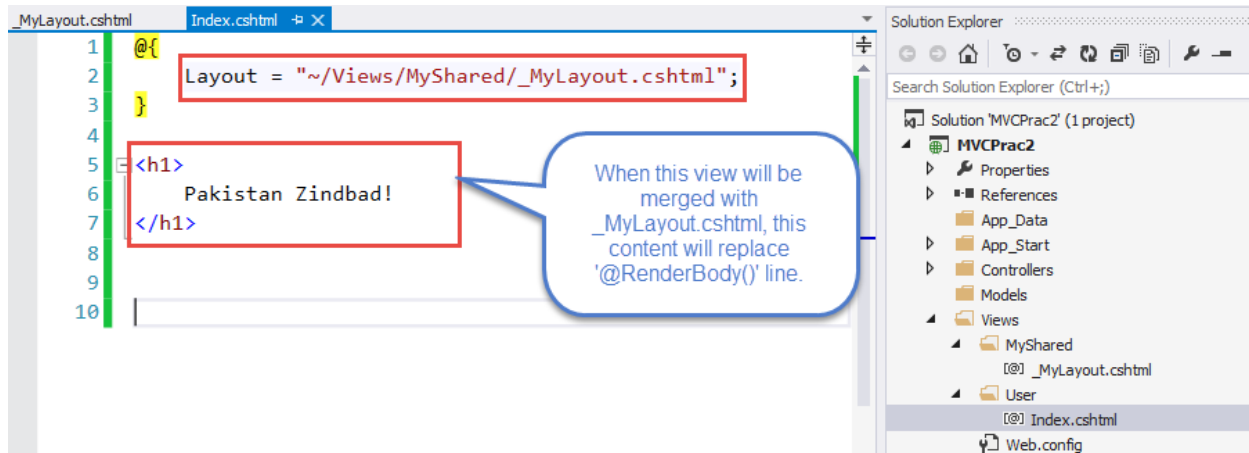
3- Open "_MyLayout.cshtml" file and you will see basic HTML in it. Add "@RenderBody()" line where you want to place HTML of your child view. At run time, when request for a view will come, content of that view is merged with Layout view.

4- Now open "Views/User/Index.cshtml" file and change the content of the file as shown in following screenshot. Here we are specifying the "Layout" file this view should use. Note that here we are going to give only that content which will replace '@RenderBody()' line in Layout.
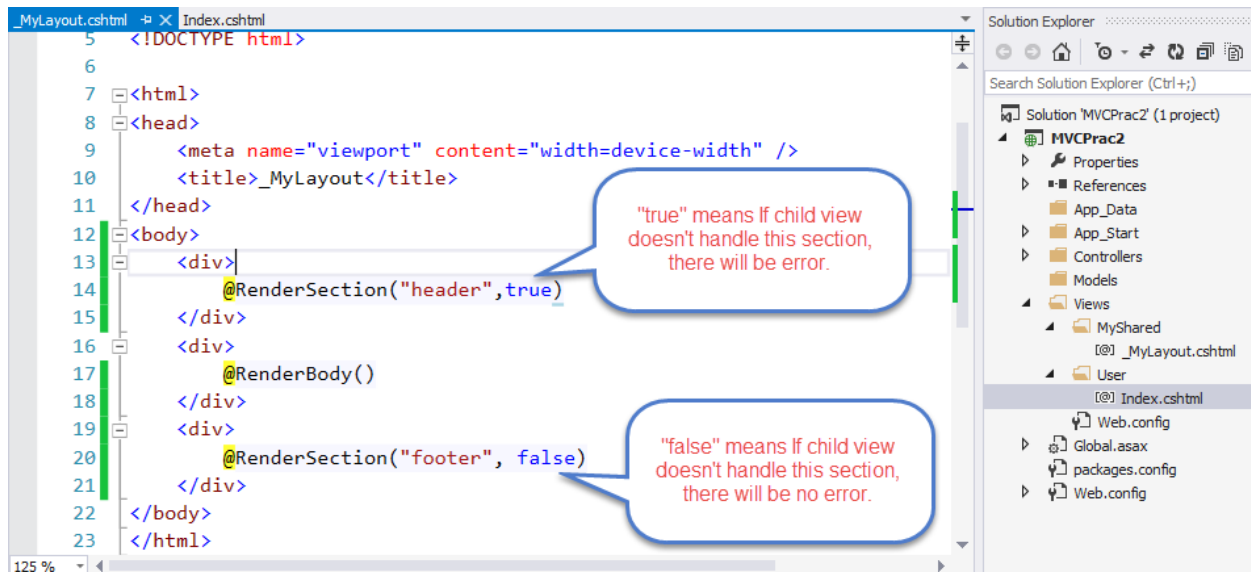

5- Run the project and check if view produces same output. Also inspect the html in browser to see if you see only "index.cshtml" content or a merged result.

6- We can also define sections in Layout which child view (we should not call it child though) may fill. Let's define two sections in our Layout file. Here we'll use '@RenderSection()' method.
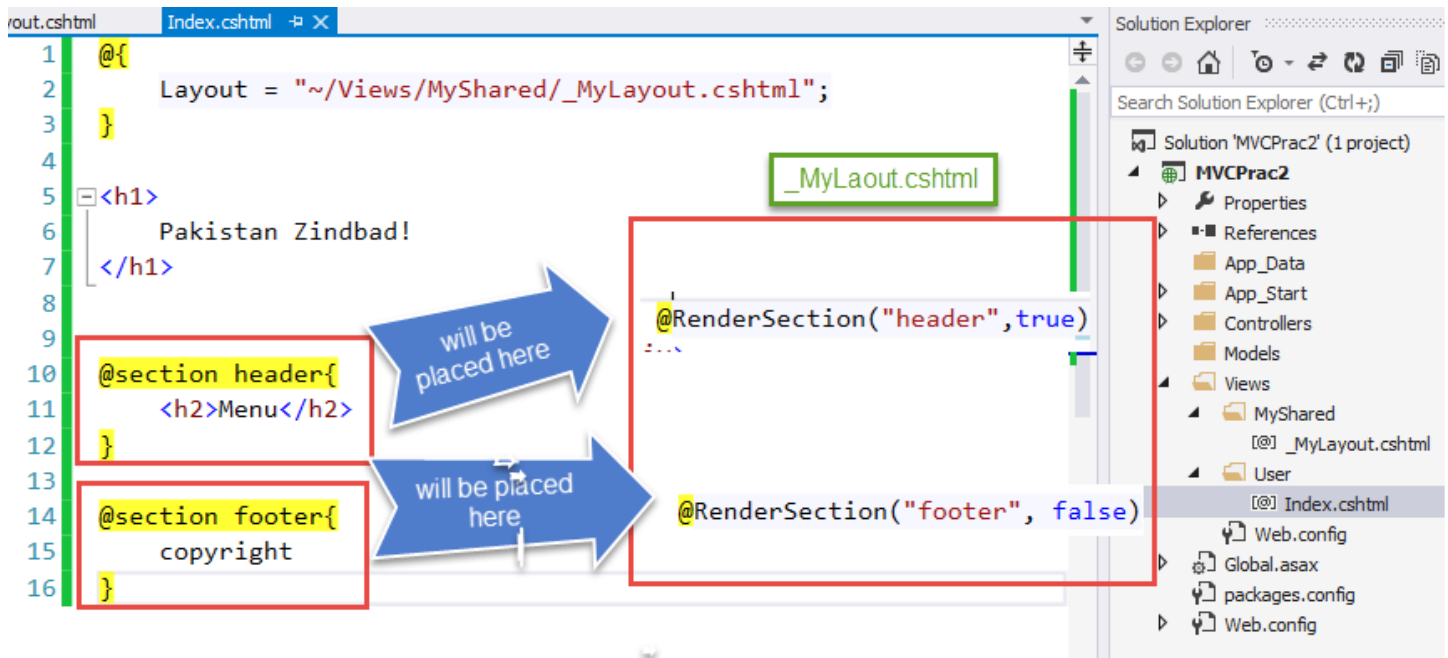
```html
5    <!DOCTYPE html>
6
7    <html>
8    <head>
9        <meta name="viewport" content="width=device-width" />
10       <title>_MyLayout</title>
11   </head>
12   <body>
13       <div>
14           @RenderSection("header",true)
15       </div>
16       <div>
17           @RenderBody()
18       </div>
19       <div>
20           @RenderSection("footer", false)
21       </div>
22   </body>
23   </html>
```

"true" means If child view doesn't handle this section, there will be error.

"false" means If child view doesn't handle this section, there will be no error.

7- Now make changes in "Views/User/Index.cshtml" file as shown below.

```
1    @{
2        Layout = "~/Views/MyShared/_MyLayout.cshtml";
3    }
4
5    <h1>
6        Pakistan Zindbad!
7    </h1>
8
9
10   @section header{
11       <h2>Menu</h2>
12   }
13
14   @section footer{
15       copyright
16   }
```
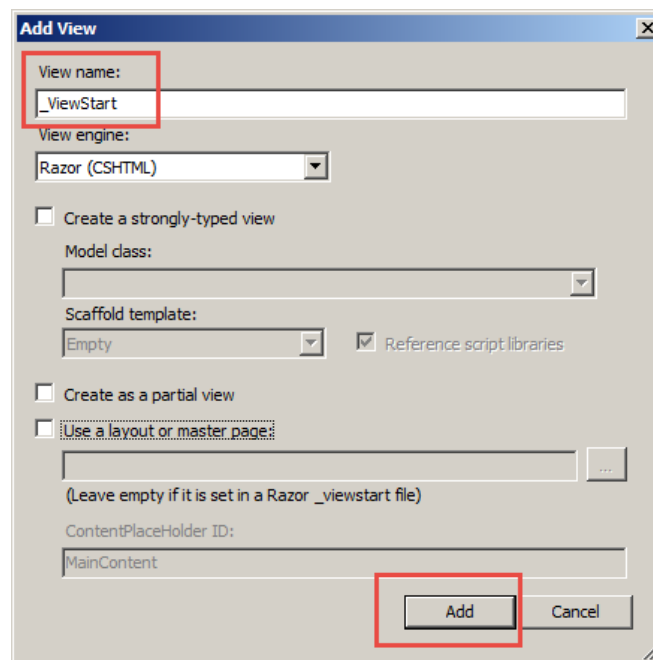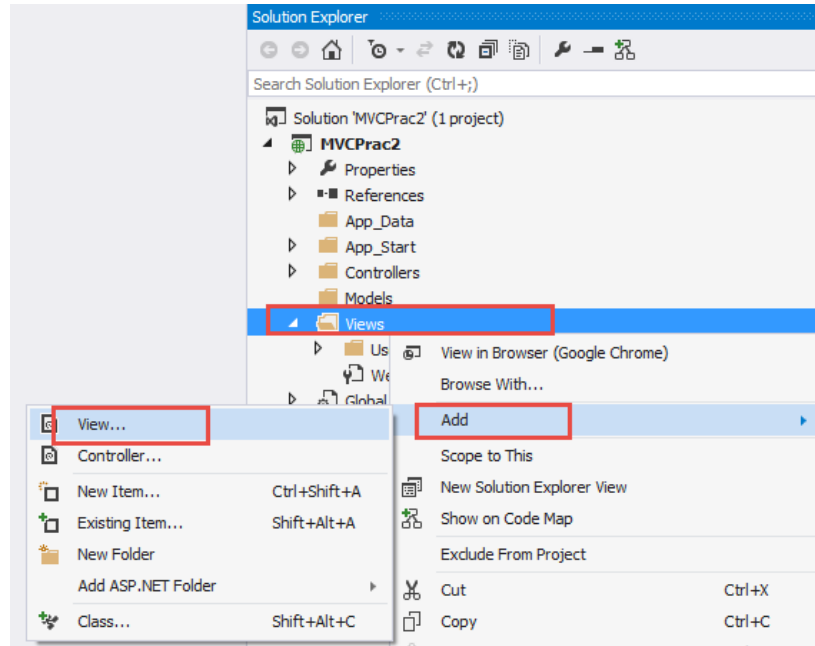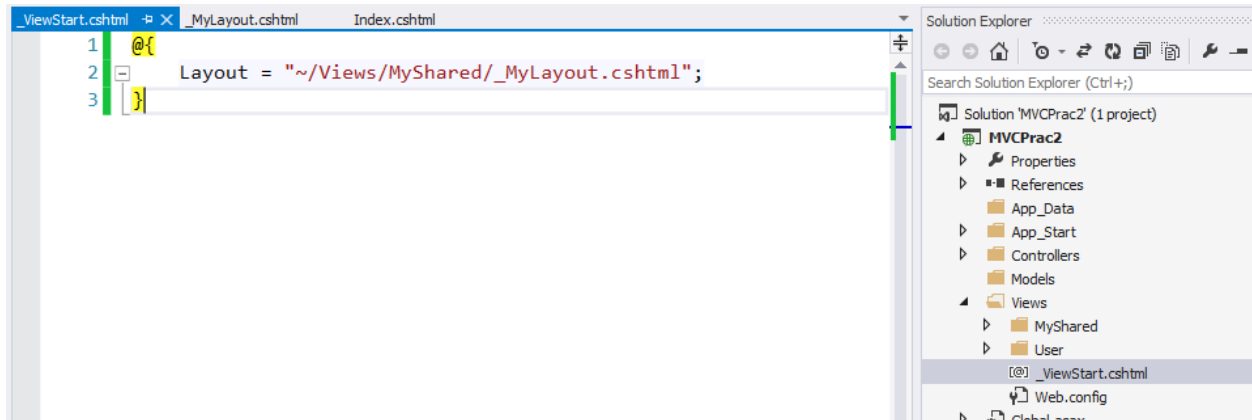
_MyLaout.cshtml

will be placed here → @RenderSection("header",true)
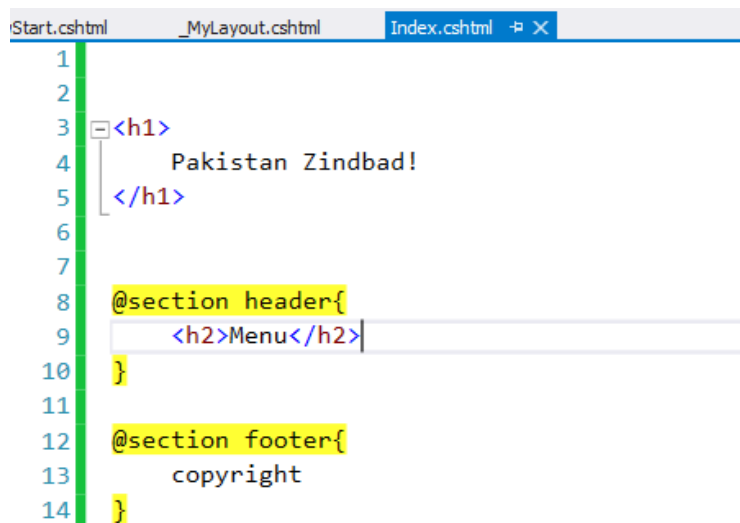
will be placed here → @RenderSection("footer", false)

8- Run the project and check if output is as expected.
9- Now let's talk about "_ViewStart"file. Detail of this file is already shared in introduction section. Suppose we don't want to set "Layout" property in our views and want to set it by default. We know that "_ViewStart" file code executes before the execution of our views so if we'll set "Layout" property there, it will be set when our "view" will be executed. Let's create "_ViewStart.cshtml" file in our views folder. Name should be **"_ViewStart"** according to the convention. Right Click on "Views" folder => Add => View.

10- Now open "_ViewStart.cshtml" file and remove existing HTML and set layout property as shown below.

11- Also remove Layout property from your "Index.cshtml" view. In cases, where you want to set different layout than "_ViewStart", you may set "Layout" property in your views.

.

12- Run the project and check if output is as expected.

Tasks for Practice

Once you are done with above tutorial. Try to work on these tasks.

1- Create a new Controller Student and actions "Home" & "Index" in it.
2- Create a new Layout file with name of your rollno.
3- In your "Student/Home.cshtml" file, refer Layout file created in step 2.
4- Create one more view "Student/Index.cshtml"file but don't set Layout property.
5- Create "_ViewStart.cshtml" file in "Student" folder and set Layout property to your rollno layout.
6- Run the project, access following links and see what is happening.
   a. /Student/Home
   b. /Student/Index
7- Explore what are "nested layouts" in ASP.NET MVC.

Useful Links

http://www.asp.net/mvc/overview/views