Seanca 5

Shared Views

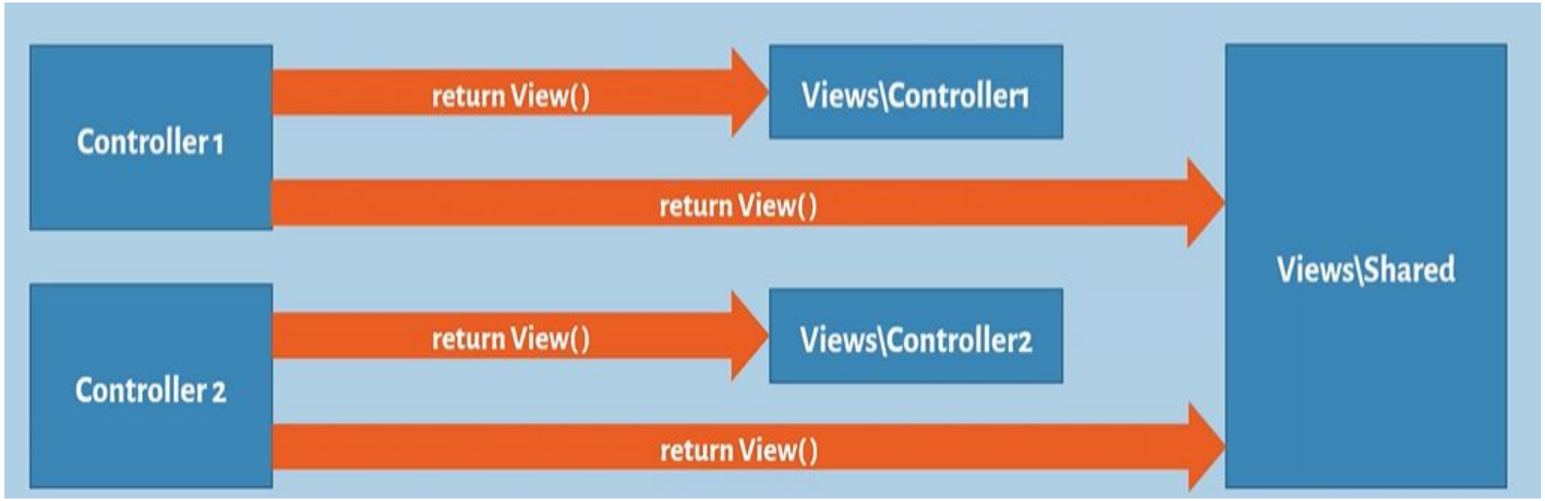
- Shared views are present in the “Views\Shared” folder.
- Shared views can be called from any controller of the same MVC project.

 For example, there are two controllers called “Controller1Controller” and “Controller2Controller”. The views that are in the “Views\Controller1” folder are accessible from the “Controller1Controller” only. The views that are in the “Views\Controller2” folder are accessible from the “Controller2Controller” only. The views that are present in the “Views\Shared” folder are accessible from both “Controller1Controller” and also from “Controller2Controller”.

- “Views\Shared” folder contains the common views of all the controllers.
- **The name “Views\Shared” is a fixed name, we can’t change it.**

When you call a view, ASP.NET MVC first checks in “Views\Controllername” folder and then it checks in “Views\Shared” folder.

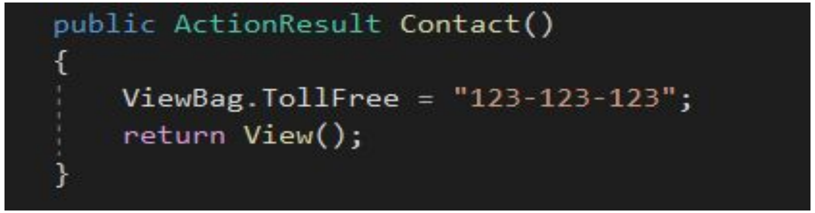
Shared Views



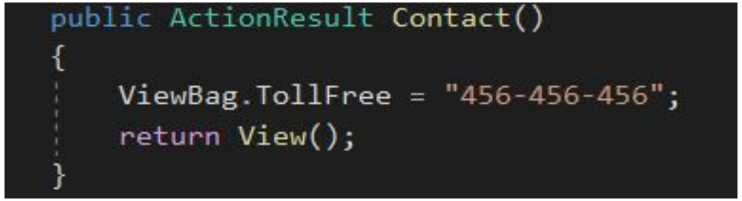
Passing data to shared views

- We can use ViewBag to pass data to the shared view, from the controller, we can pass different values from controllers that call this view

Example: We pass different value for admin and home controller



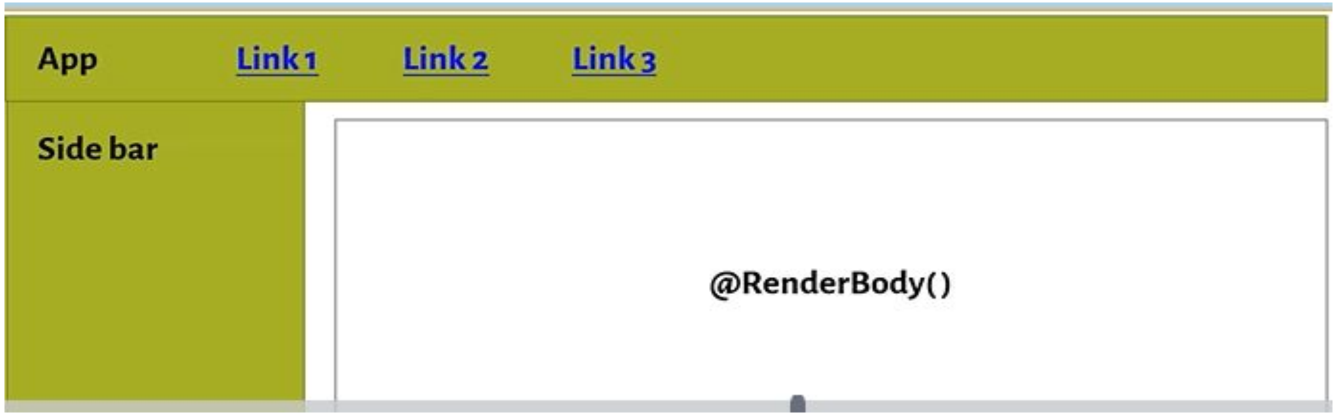
```
public ActionResult Contact()
{
    ViewBag.TollFree = "123-123-123";
    return View();
}
```



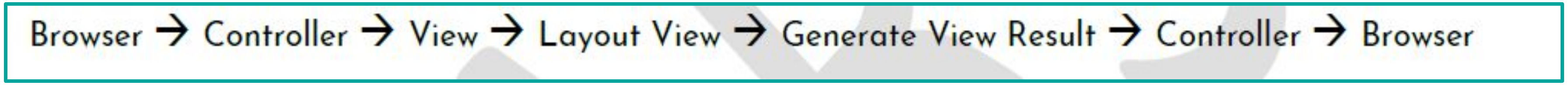
```
public ActionResult Contact()
{
    ViewBag.TollFree = "456-456-456";
    return View();
}
```

Layout views with bootstrap

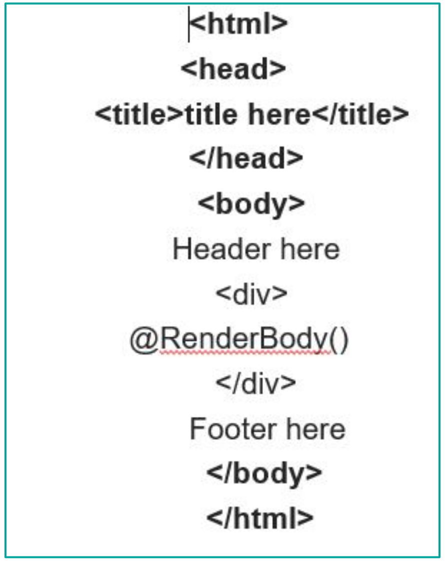
- Layout Views (or) Layout Pages are just like master pages in asp.net web forms.
- Layout pages contain “page template”.
- Layout pages contain the common content (such as header, footer, side bar etc.) that should be displayed in every page.
- First we create a layout page and then we have to create normal views **based on the layout pages**



Execution Flow


Browser → Controller → View → Layout View → Generate View Result → Controller → Browser

Syntax of layout page:



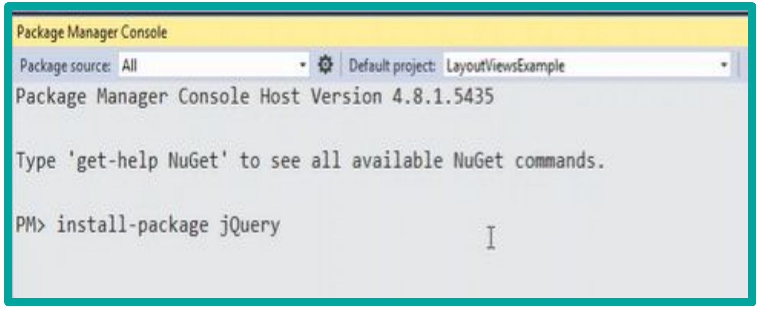
```
<html>
<head>
<title>title here</title>
</head>
<body>
Header here
<div>
@RenderBody()
</div>
Footer here
</body>
</html>
```

Bootstrap

- Open source front-end framework
- It contains css, html, and javascript
- It creates responsive websites and applications
- Bootstrap includes responsive, mobile, grid system that appropriately scales up to 12 columns as the device or viewport size increases. It includes predefined classesf or easy layout options
- From the online documentation we can see components it provides and use them (https://getbootstrap.com/docs/4.0/components/buttons/)

Installing Bootstrap

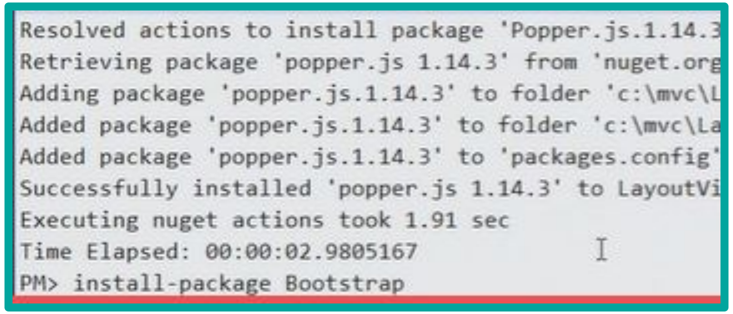
- First we need to install JQuery and Bootstrap from **Package Manager Console**



```
Package Manager Console
Package source: All   Default project: LayoutViewsExample
Package Manager Console Host Version 4.8.1.5435

Type 'get-help NuGet' to see all available NuGet commands.

PM> install-package jQuery
```

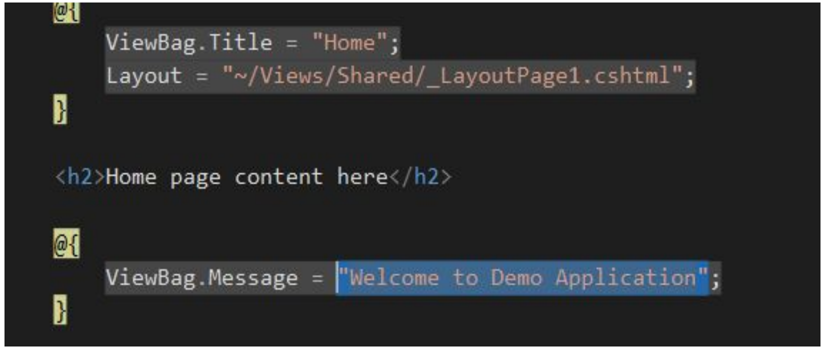


```
Resolved actions to install package 'Popper.js.1.14.3
Retrieving package 'popper.js 1.14.3' from 'nuget.org
Adding package 'popper.js.1.14.3' to folder 'c:\mvc\L
Added package 'popper.js.1.14.3' to folder 'c:\mvc\La
Added package 'popper.js.1.14.3' to 'packages.config'
Successfully installed 'popper.js 1.14.3' to LayoutVi
Executing nuget actions took 1.91 sec
Time Elapsed: 00:00:02.9805167
PM> install-package Bootstrap
```

Sharing data from view to layout view

- We can share data from view to layout view with a ViewBag object

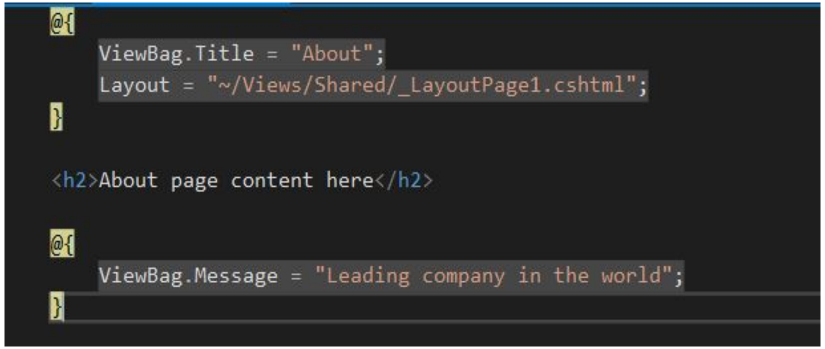
Example: In Index View we want to show a message in the part of layout view**: "Welcome to Demo Application“**, in About view we want that the title shows: **“Leading company in the world”**



```
@{
    ViewBag.Title = "Home";
    Layout = "~/Views/Shared/_LayoutPage1.cshtml";
}

<h2>Home page content here</h2>

@{
    ViewBag.Message = "Welcome to Demo Application";
}
```

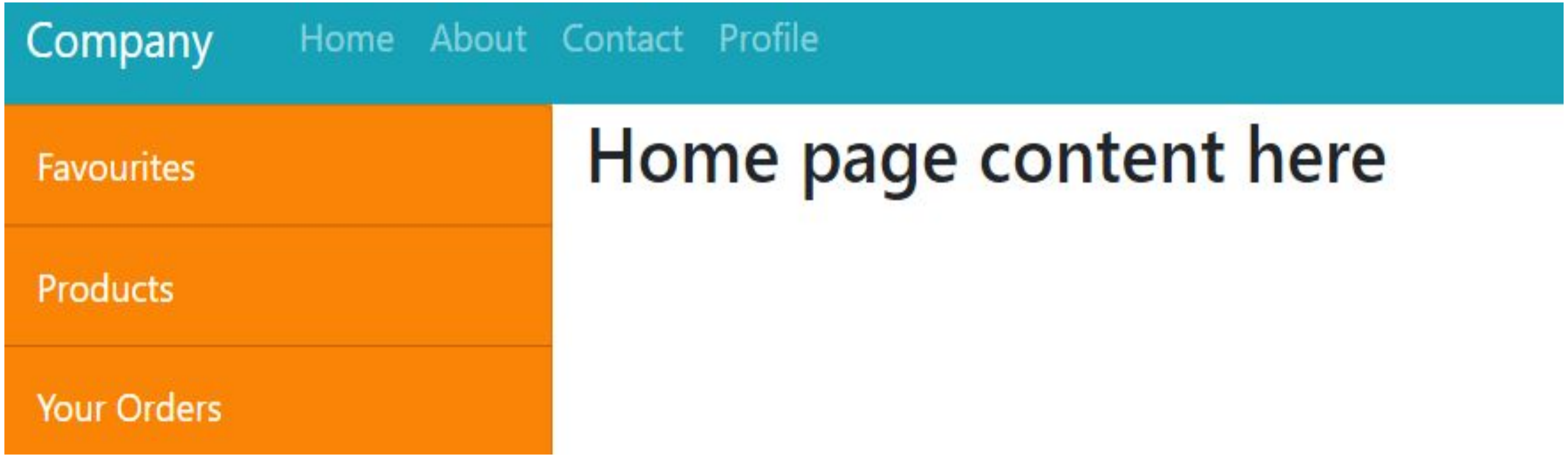


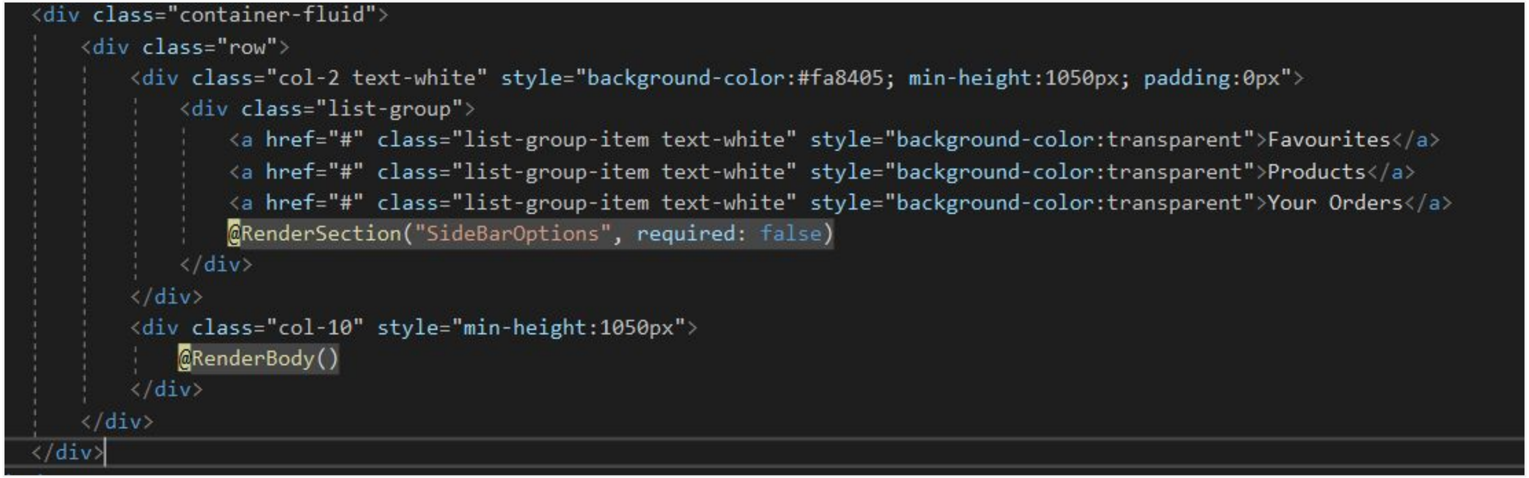
```
@{
    ViewBag.Title = "About";
    Layout = "~/Views/Shared/_LayoutPage1.cshtml";
}

<h2>About page content here</h2>

@{
    ViewBag.Message = "Leading company in the world";
}
```

Creating SideBar in Layout View





```
<div class="container-fluid">
    <div class="row">
        <div class="col-2 text-white" style="background-color:#fa8405; min-height:1050px; padding:0px">
            <div class="list-group">
                <a href="#" class="list-group-item text-white" style="background-color:transparent">Favourites</a>
                <a href="#" class="list-group-item text-white" style="background-color:transparent">Products</a>
                <a href="#" class="list-group-item text-white" style="background-color:transparent">Your Orders</a>
                @RenderSection("SideBarOptions", required: false)
            </div>
        </div>
        <div class="col-10" style="min-height:1050px">
            @RenderBody()
        </div>
    </div>
</div>
```

Sections in LayoutView

- Sections are like “content place holders” in asp.net web forms.
- Sections are used to display view-specific content in the layout view.
- Sections are defined in the normal view and rendered in the layout view.

Syntax to define a section in the view

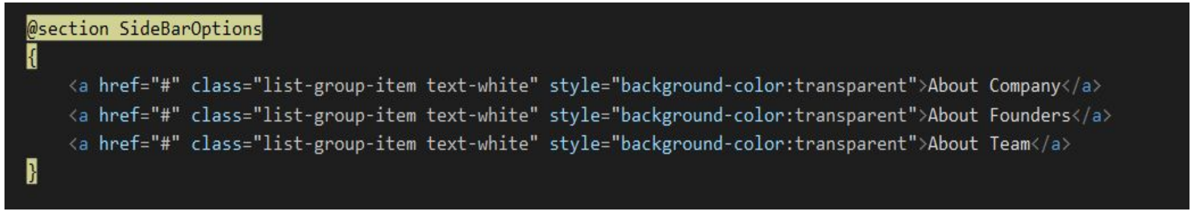
```
  @section sectionnamehere
  {
   Content here
   }
```

Syntax to render (display) the section in the layout view

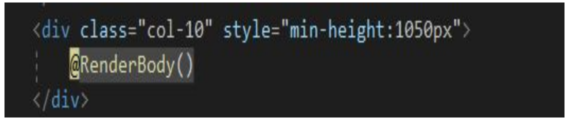
@RenderSection("sectionnamehere", required: false)

Sections Example

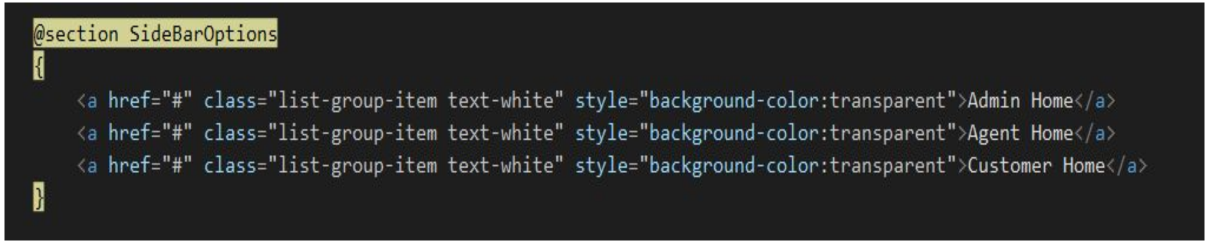
- We will show different menu items for the sidebar menu, for different controllers



```
@section SideBarOptions
{
    <a href="#" class="list-group-item text-white" style="background-color:transparent">About Company</a>
    <a href="#" class="list-group-item text-white" style="background-color:transparent">About Founders</a>
    <a href="#" class="list-group-item text-white" style="background-color:transparent">About Team</a>
}
```



```
<div class="col-10" style="min-height:1050px">
    @RenderBody()
</div>
```

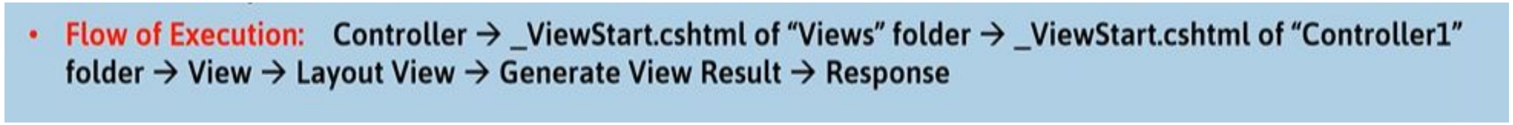


```
@section SideBarOptions
{
    <a href="#" class="list-group-item text-white" style="background-color:transparent">Admin Home</a>
    <a href="#" class="list-group-item text-white" style="background-color:transparent">Agent Home</a>
    <a href="#" class="list-group-item text-white" style="background-color:transparent">Customer Home</a>
}
```

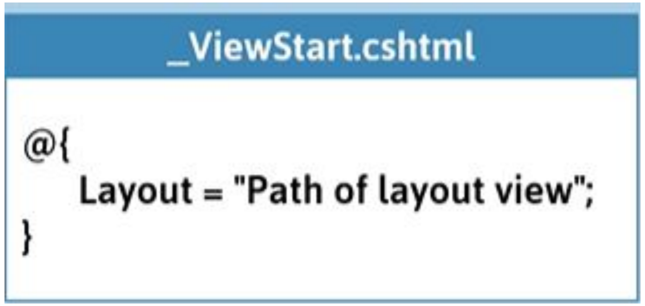
_ViewStart.cshtml

- The “_ViewStart.cshtml” (fixed name) is a special file that can be present either in “Views” folder or in “Views\controllername” folder.
- If it is present in “Views” folder, it specifies the path of the default layout view **of all the views in the entire project.**
- If it is present in “Views\controllername” folder, it specifies the path of layout view **of all the views in the same folder only.**
- Before executing the view, ASP.NET MVC calls the “_ViewStart.cshtml” file automatically, if it is is present.
- You can override the setting of “_ViewStart.cshtml”, in any view, by re-assigning the value of “Layout” property.
- We can’t write any extra code in the “_ViewStart.cshtml” file, except setting a layout view.

Execution Flow



- Flow of Execution: Controller → _ViewStart.cshtml of "Views" folder → _ViewStart.cshtml of "Controller1" folder → View → Layout View → Generate View Result → Response



_ViewStart.cshtml

```
@{
    Layout = "Path of layout view";
}
```

Example

Change the layout view solution:

- Create the _ViewStart.cshtml and add the layout view name
- Remove the layout line from the views

❖ Advantage: If you need to change the layout of your website, you only need to change the _ViewStart.chtml file and change the layout view name

MultipleLayout Views

- We can have multiple layout views in an application, by creating different layout views

Example:

We can specify a global layout view for the project, a layout view for Home Controller, and even override the layout for specific view under the same controller

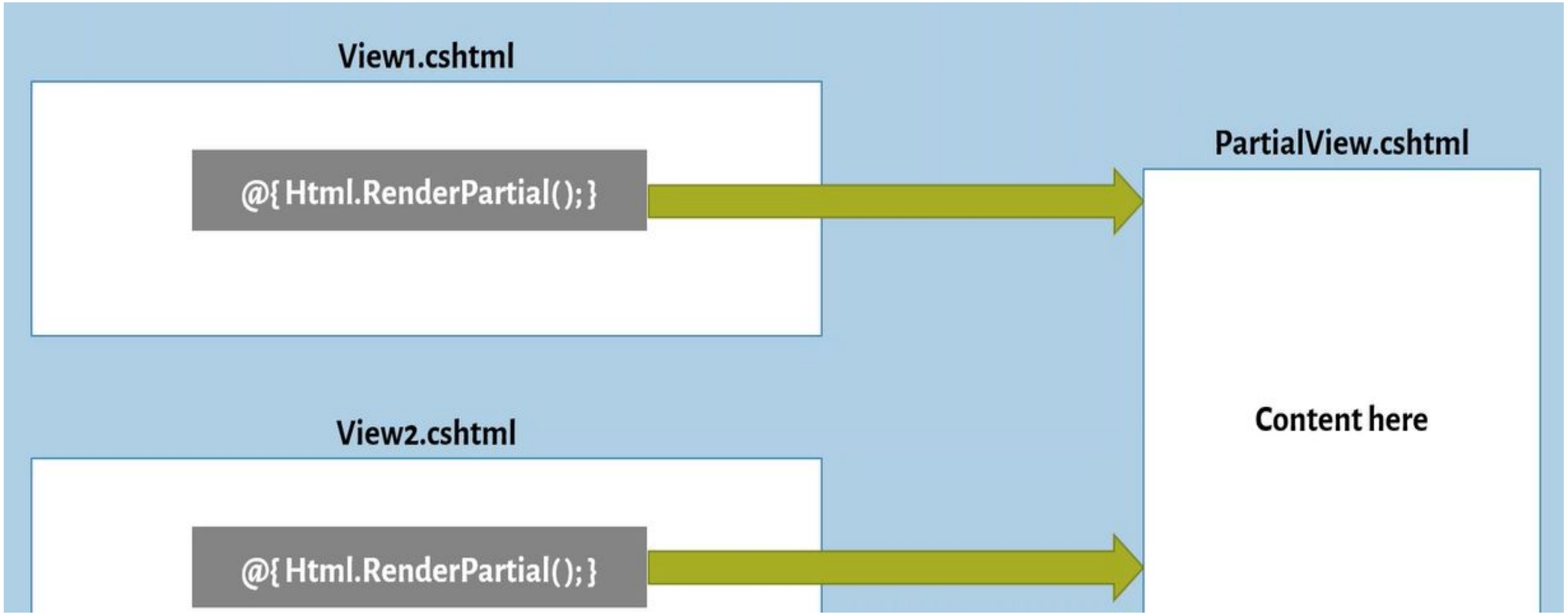
1. Create two layout views: layout 1, and layout 2
2. Apply layout 1 to the project
3. Apply layout 2 to home controller
4. Change layout of a view inside home controller, to layout 1

Partial Views

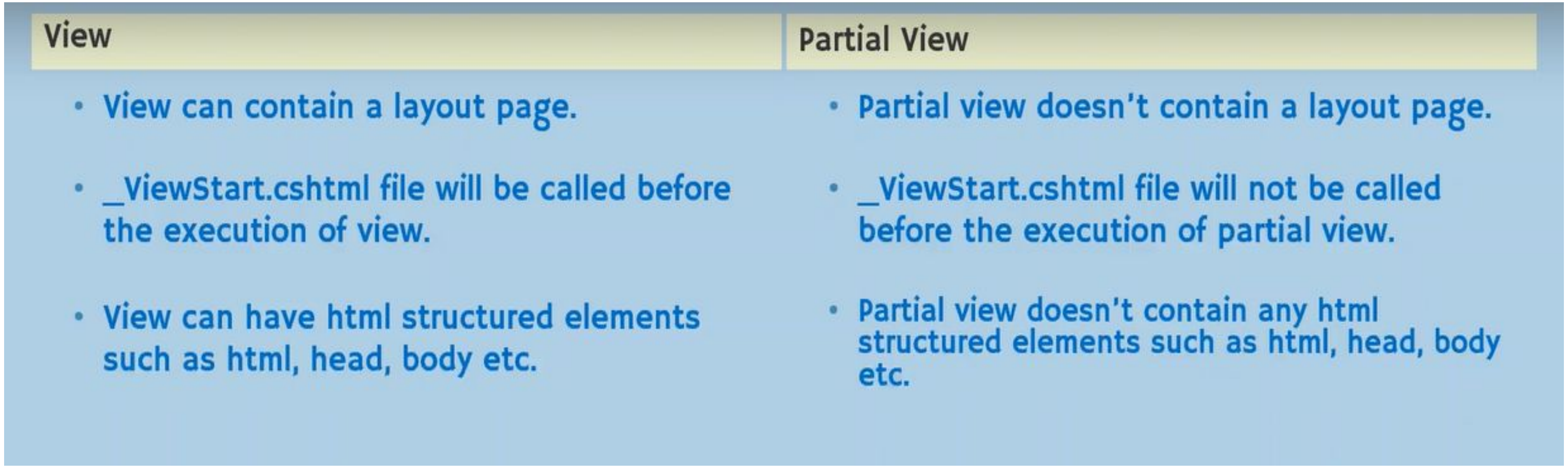
Assume that, you have a specific part of the web page repeated in many web pages. Then you can make that part of the page as a separate file called “partial view” and then you can call the same in any view, anywhere.

- Partial views are “part of the page”.
- Partial views are “re-usable code block”.
- Partial views can be called in one or more views.
- Partial views can be present in “Views\controllername” folder or in “Views\Shared” folder.
- If partial view is present in “Views\controllername” folder, it can be called only within the views of the same folder.
- If partial view is present in “Views\Shared” folder, it can be called in all the views of entire project.

Partial Views



Views vs Partial Views

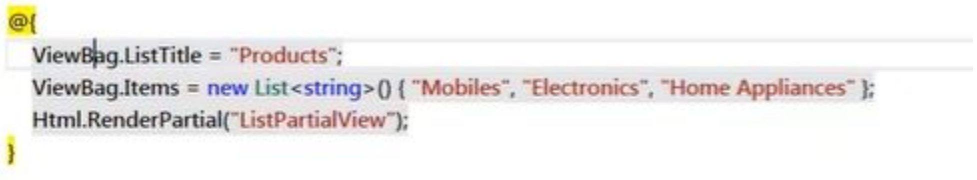


View	Partial View
• View can contain a layout page.	• Partial view doesn't contain a layout page.
• _ViewStart.cshtml file will be called before the execution of view.	• _ViewStart.cshtml file will not be called before the execution of partial view.
• View can have html structured elements such as html, head, body etc.	• Partial view doesn't contain any html structured elements such as html, head, body etc.

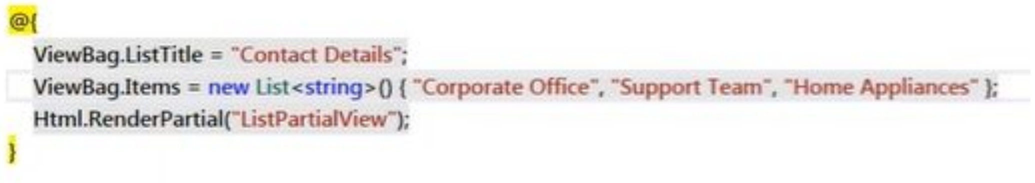
Example

- Create a partial view ListPartialView that will display a list of objects. It will be used by two different views, one will display a list of products, and the other will display a list of contact details

Passing the data:



```
@{
    ViewBag.ListTitle = "Products";
    ViewBag.Items = new List<string>() { "Mobiles", "Electronics", "Home Appliances" };
    Html.RenderPartial("ListPartialView");
}
```



```
@{
    ViewBag.ListTitle = "Contact Details";
    ViewBag.Items = new List<string>() { "Corporate Office", "Support Team", "Home Appliances" };
    Html.RenderPartial("ListPartialView");
}
```

Assignment

1) In the Product List Solution, add a Category Controller and view that will display a list of categories.
2) Use a partial view for both Products and Categories, that returns a list of objects.

Assignment

- Create a sidebar in the List of Products project, with products and categories items. When clicking at this sidebar, the list of products or categories should be displayed.

Assignment

- Create a section in a layout view to display a message in the header:” List of products”, and “List of categories”, based on the view.

QUESTIONS