Ok, let's add some navigation to our application.

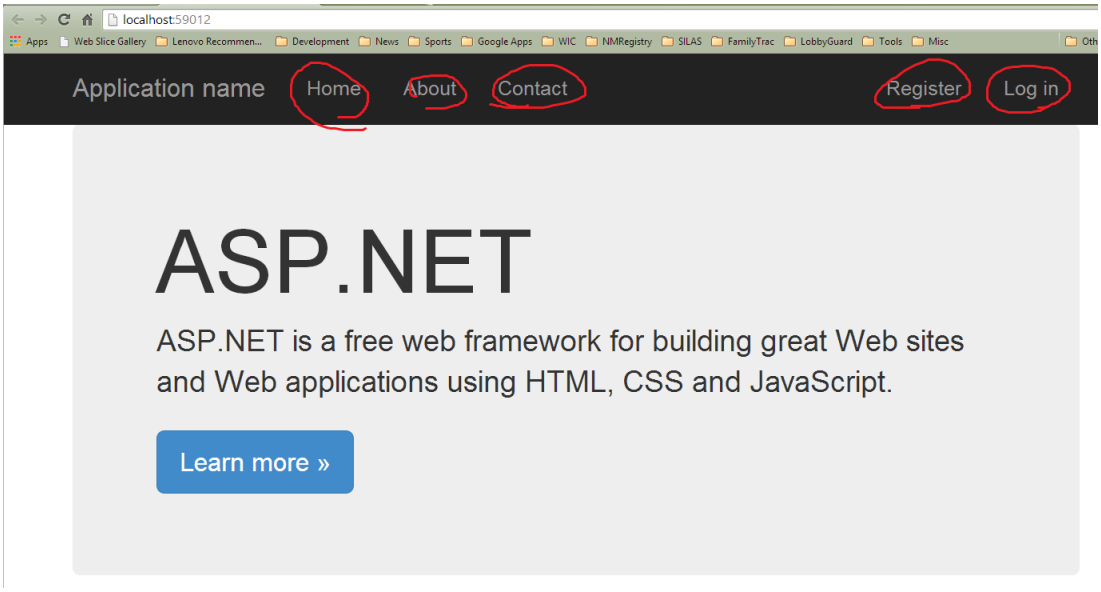
Html Helpers is a nice tool to help us. Let's add some Html.ActionLinks for navigation.

So, what is an HTML Helper Method? In most cases, an Html Helper is just a method that returns a string.

Here is the MSDN reference:

MVC 5 HtmlHelper Methods
http://msdn.microsoft.com/en-us/library/system.web.mvc.htmlhelper_methods(v=vs.118).aspx

If you create an MVC project in Visual Studio, not the empty one that we started with, you get some navigation:



Let's look at what they did. They just added some ActionLinks to the _Layout.cshtml file. Let's steal some of that code, do the same, and then come back and look at what's going on here.

Open _Layout.cshtml, it's in Views => Shared. The nav div and un-numbered list (<ul>) is already present. Let's steal the 3 list items:

```
<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>
```

About and Contact don't work yet, but that's ok.

Let's add a navigation item for Meetings. The signature for this Html.Helper is

(linkText, actionName, controllerName)

```
<li>@Html.ActionLink("Home", "Index", "Home")</li>
<li>@Html.ActionLink("Meetings", "Index", "Meetings")</li>
<li>@Html.ActionLink("About", "About", "Home")</li>
<li>@Html.ActionLink("Contact", "Contact", "Home")</li>
```

Now add the Contact and About controllers and views so that all of the navigation links work.

But wait, it doesn't work. Why?

Because the Microsoft solution placed contact and about in the Home controller, as actions with their own views.

So, the action links have to look like this. Remember, (linkText, actionName, controllerName):

```
<li>@Html.ActionLink("About", "Index", "About")</li>
<li>@Html.ActionLink("Contact", "Index", "Contact")</li>
```

Now, run the project and view source. Here's the HTML that is rendered:

```
<div class="navbar-collapse collapse">
        <ul class="nav navbar-nav">
                <li><a href="/">Home</a></li>
                <li><a href="/Meetings">Meetings</a></li>
                <li><a href="/About">About</a></li>
                <li><a href="/Contact">Contact</a></li>
        </ul>
</div>
```

Almost doesn't seem worth it, but maybe so as we use more complex helpers. Take away, you can always just write the Html, it's just a helper!

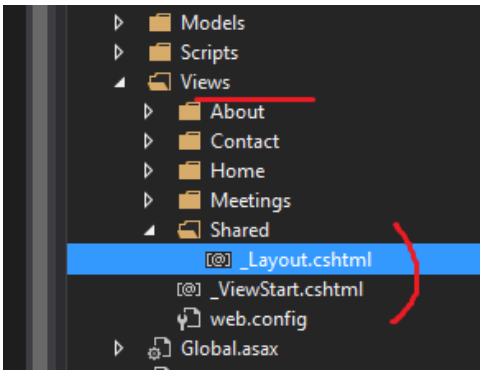
You can also change the <h2> text for Contact and About:

```
@{
   ViewBag.Title = "Contact";
}

<h2>Contact</h2>
```

Ok, test your links, make sure everything works!

Inside the Views folder lives a Shared folder and _Layout.cshtml:



Views here share this layout. And @RenderBody renders the content of your page that is not within any named sections. Since we haven't used @RenderSection yet, then everything in the views is rendered.

Time allowing, let's look at some Html.Helpers that Jim added:

```
@Html.BeginForm()
@Html.AntiForgeryToken()
@Html.ValidationSummary()
@Html.HiddenFor()
@Html.LabelFor()
@Html.EditorFor()
@Html.ValidationMessageFor()
@Html.ActionLink()
```

Note that there are 12 overloaded signatures for the ActionLink on the reference page. Here are 3 basic forms:

```
ActionLink(String, String)
ActionLink(String, String, Object)
ActionLink(String, String, RouteValueDictionary)
```