IT-Academy

Разработка представлений

МОДУЛЬ 6

IT-Academy

План

- View
- Razor Engine
- Layout
- Html Helpers
- Url Helpers
- Templates

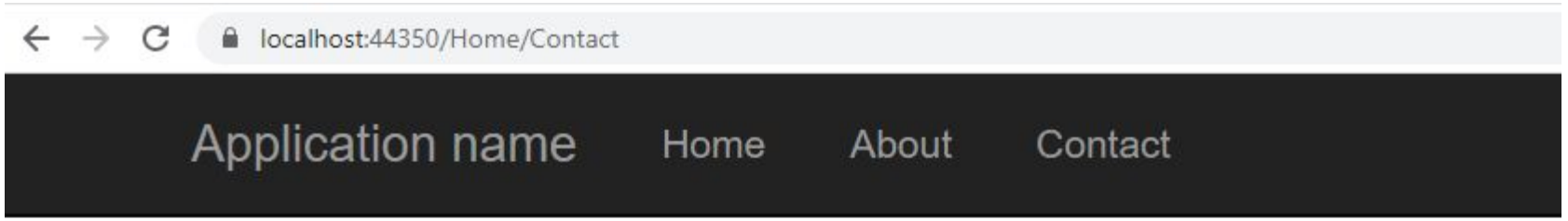
IT-Academy

View

Представление отвечает за предоставление пользовательского интерфейса (UI) пользователю. После того, как контроллер выполнил соответствующую логику для запрошенного URL, он делегирует отображение представлению.

IT-Academy

```
@{
    ViewBag.Title = "Contact";
}
<h2>@ViewBag.Title.</h2>
<h3>@ViewBag.Message</h3>

<address>
    One Microsoft Way<br />
    Redmond, WA 98052-6399<br />
    <abbr title="Phone">P:</abbr>
    425.555.0100
</address>

<address>
    <strong>Support:</strong>   <a href="mailto:Support@example.com">Support@example.com</a><br />
    <strong>Marketing:</strong> <a href="mailto:Marketing@example.com">Marketing@example.com</a>
</address>
```


localhost:44350/Home/Contact
Application name
Home
About
Contact
Contact.
Your contact page.
One Microsoft Way
Redmond, WA 98052-6399
P: 425.555.0100
Support: Support@example.com
Marketing: Marketing@example.com
© 2020 - My ASP.NET Application

IT-Academy

IT-Academy

```
public ActionResult About()
{
    ViewBag.Message = "Your application description page.";

    return View();
}
```

ViewResult

/Views/Имя_контроллера/Имя_представления.cshtml

/Views/Shared/Имя_представления.cshtml

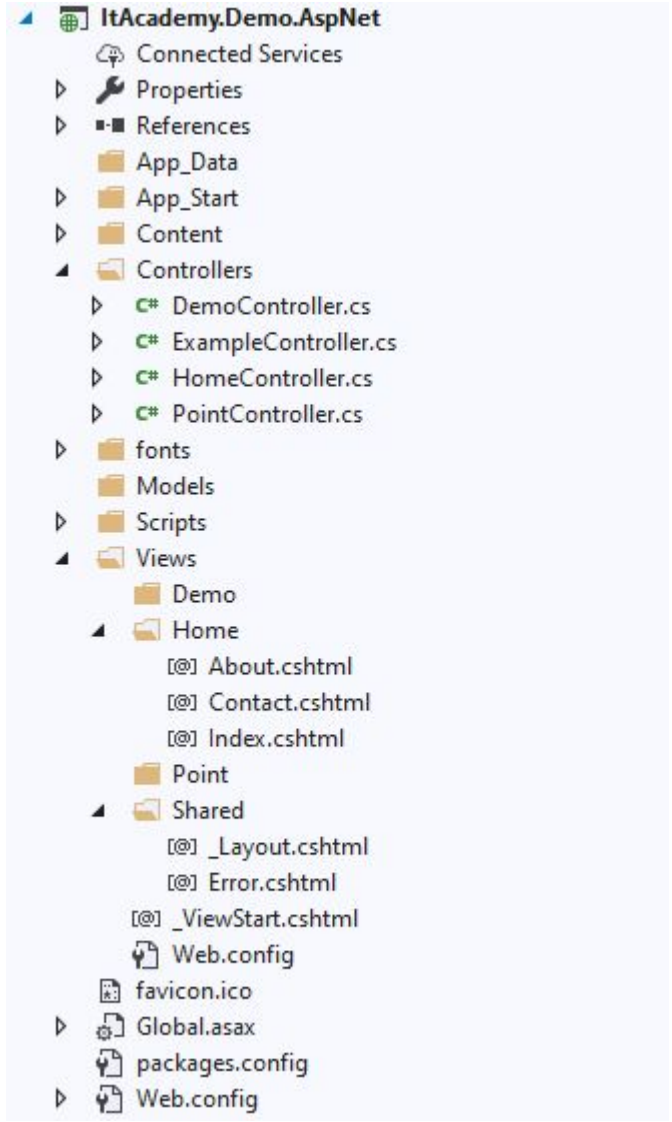
IT-Academy

ViewResult

- View():
  использует название вызывающего экшена.
- View(string viewName):
  переопределяет имя представления.
- View(object model):
  передает модель.
- View(string viewName, object model):
  переопределяет имя представления и передает в него модель.

IT-Academy

```
public ActionResult About()
{
    ViewBag.Message = "Your application description page.";

    return View(5);
}

public ActionResult Contact()
{
    ViewBag.Message = "Your contact page.";

    return View("Index");
}

public ActionResult Contact()
{
    ViewBag.Message = "Your contact page.";

    return View("Index", 5);
}
```


ItAcademy.Demo.AspNet
Connected Services
Properties
References
App_Data
App_Start
Content
Controllers
DemoController.cs
ExampleController.cs
HomeController.cs
PointController.cs
fonts
Models
Scripts
Views
Demo
Home
About.cshtml
Contact.cshtml
Index.cshtml
Point
Shared
_Layout.cshtml
Error.cshtml
_ViewStart.cshtml
Web.config
favicon.ico
Global.asax
packages.config
Web.config

IT-Academy

IT-Academy

View

- Html
- Razor syntax
- Html Helpers

IT-Academy

Razor View Engine

```
<footer>
    <p>Year @DateTime.Now.Year </p>


    @{
        int year = DateTime.Now.Year;
        int nextYear = year + 1;
    }

    <p>Year @year, Next year @nextYear</p>
</footer>
```

Razor View Engine

Вывод текста в блоке кода

```
<footer>
    <p>Year @DateTime.Now.Year </p>


    @{
        int year = DateTime.Now.Year;
        int nextYear = year + 1;
        <text>Year @year, Next year @nextYear</text>
    }
</footer>
```

```
<footer>
    <p>Year @DateTime.Now.Year </p>


    @{
        int year = DateTime.Now.Year;
        int nextYear = year + 1;
        <p>Year @year, Next year @nextYear</p>
    }
</footer>
```

Razor View Engine

Вывод текста в блоке кода

```
<footer>
    <p>Year @DateTime.Now.Year </p>


    @{
        int year = DateTime.Now.Year;
        int nextYear = year + 1;
        @:Year @year, Next year @nextYear
    }
</footer>
```

Razor View Engine

функции

```
@functions{
    public int GetSum(int a, int b)
    {
        return a + b;
    }
}
<div>Sum: @GetSum(1, 2)</div>
```

Razor View Engine

- @ для написания server side code.
- @{* code * } для написания блока server side code.
- @: для отображения текста из блока кода.
- <text></<text> для отображения текста из блока кода.
- @if{ }
- @for
- @model позволяет использовать модель во всем view.

Передача данных в представление

IT-Academy

- View Model
- ViewBag
- ViewData

- TempData

```
TempData.Keep();
```

Передача данных в представление

ViewBag, ViewData

IT-Academy

- ViewData["CurrentTime"] = DateTime.Now;
- ViewBag.CurrentTime = DateTime.Now;

- ViewBag.CurrentTime эквивалент ViewData["CurrentTime"]

Передача данных в представление

ViewBag, ViewData

IT-Academy

- @Html.TextBox("name", ViewBag.Name) will fail

- @Html.TextBox("name", ViewData["Name"])

  или

- @Html.TextBox("name", (string)ViewBag.Name)

Передача данных в представление

IT-Academy

```
public ActionResult About()
{
    TempData["currentDateTime"] = DateTime.Now.ToString();
    ViewBag.Message = "Your application description page.";
    var viewModel = new Point();

    return View(viewModel);
}
```

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

@model ItAcademy.Demo.AspNet.Controllers.Point

<h2>@ViewBag.Title.</h2>
<h3>@ViewBag.Message</h3>

<h4>@TempData["currentDateTime"]</h4>

<p>@Model.X</p>
<p>@Model.Y</p>

<p>Use this area to provide additional information.</p>
```

IT-Academy

Partial View

- @{Html.RenderPartial("Partial");}

  результат напрямую пишет вывод в выходной поток

- @Html.Partial("Partial")

  возвращает MvcHtmlString

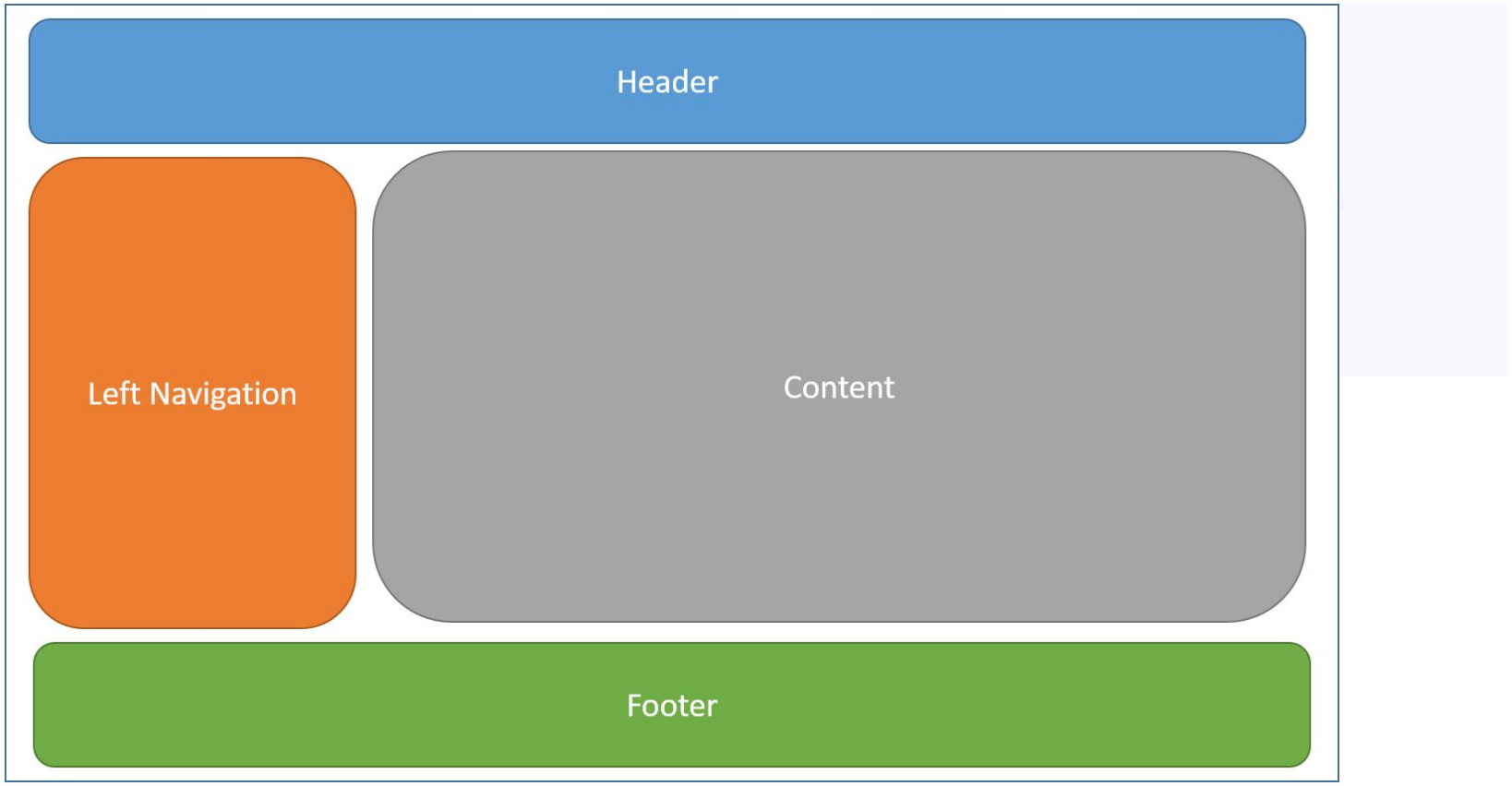
- @{Html.RenderAction(“Menu")}
- @Html.Action("Menu")

IT-Academy

Demo

Layout

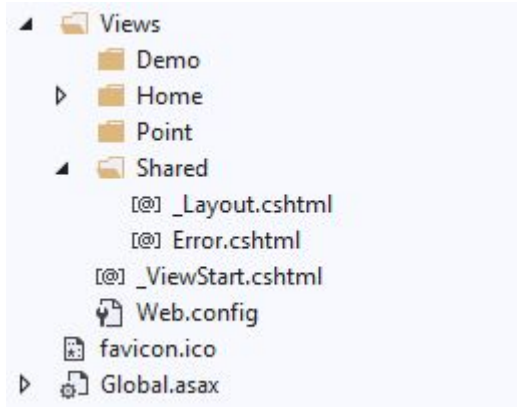
Мастер-страницы



Layout

Мастер-страницы

- _Layout.chtml
  - RenderBody: exactly one
  - RenderSection: zero or more
- _ViewStart.cshtml



```
<!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")
    <script type="text/javascript" src="https://google.com/somescript" ></script>
</head>
<body>
    <>...</>
    <div class="container body-content">
        @RenderBody()
        <hr />
        <>...</>
    </div>

    @Scripts.Render("~/bundles/jquery")
    @Scripts.Render("~/bundles/bootstrap")
    @RenderSection("scripts", required: false)
</body>
</html>
```

IT-Academy

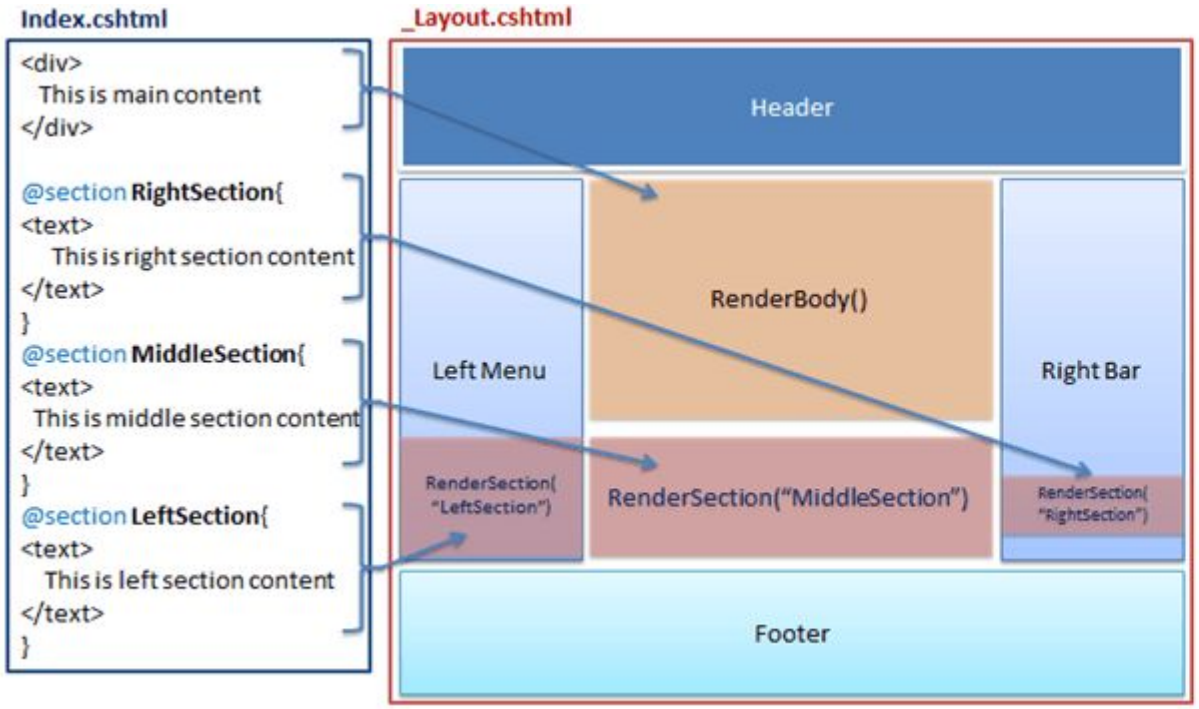
Layout

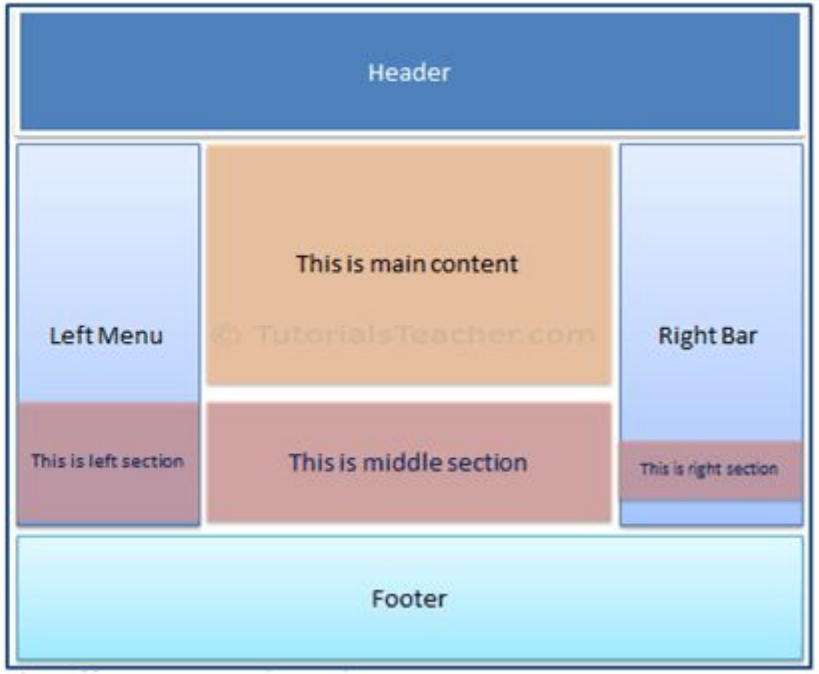
Мастер-страницы

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

<p>Use this area to provide additional information.</p>

@section scripts{
    <script>
        alert('This is from the scripts section.');
    </script>
}
```

Layout

Мастер-страницы





IT-Academy

_ViewStart.cshtml

- выполнение перед каждой Full View

```
@{
    Layout = "~/Views/Shared/_Layout.cshtml";
}
```

Static Data

(Bundling and Minification)

- @Styles.Render("~/Content/css")
- @Scripts.Render("~/bundles/jquery")

IT-Academy

IT-Academy

Demo

IT-Academy

Html Helpers

Html Helper	Strongly Typed Html Helper	Html Control
Html.ActionLink		<a>
Html.RouteLink		<a>

IT-Academy

Html Helpers

Html.ActionLink

```
@Html.ActionLink("Get point.", "Get", "Point")
```

```
<a href="/Point/Get">Go to the home page.</a>
```

Html Helpers

Html.RouteLink

```
@Html.RouteLink("Get point.", "PointRoute")
```

```
<a href="/Point/Get">Go to the home page.</a>
```

```
routes.MapRoute(
    name: "PointRoute",
    url: "point/get",
    defaults: new { controller = "Point", action = "Get", id = UrlParameter.Optional }
);
```

IT-Academy

Url Helpers

- Url.Action
- Url.RouteUrl
- Url.Content

```
@Url.Action("IndexAction")
@Url.RouteUrl("IndexRoute")
@Url.Content("~/Content/css/main.css")
```

```
<a href='@Url.Action("IndexAction")'>Go to the home page</a>
```

Html Helper	Strongly Typed Html Helper	Html Control
Html.TextBox	Html.TextBoxFor	<input type="text">
Html.TextArea	Html.TextAreaFor	<textarea>
Html.CheckBox	Html.CheckBoxFor	<input type="checkbox">
Html.RadioButton	Html.RadioButtonFor	<input type="radio">
Html.DropDownList	Html.DropDownListFor	<select>
Html.Hidden	Html.HiddenFor	<input type="hidden">
Html.Password	Html.PasswordFor	<input type="password">
Html.Label	Html.LabelFor	<label>

IT-Academy

Html Helpers

```
@model Student

@Html.TextBox("StudentName", Model.StudentName, new { @class = "form-control" })

@Html.TextBox("StudentName", "Kate", new { @class = "form-control" })

@Html.TextBoxFor(m => m.StudentName, new { @class = "form-control" })
```

```
<input class="form-control" id="StudentName" name="StudentName" type="text" value="Kate" />
<input class="form-control" id="@nameof(Model.StudentName)"
       name="@nameof(Model.StudentName)" type="text" value="@Model.StudentName" />
```

IT-Academy

Html Helpers

```
@model Student

@Html.DropDownList("StudentGender", new SelectList(Enum.GetValues(typeof(Gender))),
                   "Select Gender", new { @class = "form-control" })

@Html.DropDownListFor(m => m.StudentGender, new SelectList(Enum.GetValues(typeof(Gender))),
                   "Select Gender", new { @class = "form-control" })

@Html.EnumDropDownListFor(m => m.StudentGender,
                   "Select Gender", new { @class = "form-control" })
```

```
<select class="form-control" id="StudentGender" name="StudentGender">
    <option>Select Gender</option>
    <option selected="selected">Male</option>
    <option>Female</option>
</select>
```

Html Helper	Strongly Typed Html Helper	Html Control
Html.Display	Html.DisplayFor	Html text
Html.DisplayText	Html.DisplayTextFor	Html text
Html.Editor	Html.EditorFor	Generates Html controls based on data type of specified model property e.g. textbox for string property, numeric field for int, double or other numeric type.
	Html.DisplayForModel	
	Html.EditorForModel	

IT-Academy

IT-Academy

Html.Editor

Property DataType	Html Element
string	<input type="text" >
int	<input type="number" >
decimal, float	<input type="text" >
boolean	<input type="checkbox" >
Enum	<input type="text" >
DateTime	<input type="datetime" >
string	<input type="text" >

IT-Academy

Html.Editor

```
@model Student

<p>StudentId:       @Html.Editor("StudentId")</p>
<p>Student Name:    @Html.Editor("StudentName")</p>
<p>Age:             @Html.Editor("Age")</p>
<p>Password:        @Html.Editor("Password")</p>
<p>isNewlyEnrolled:@Html.Editor("IsActive")</p>
<p>Gender:          @Html.Editor("StudentGender")</p>
<p>DoB:             @Html.Editor("CreatedDate")</p>

<p>StudentId:       @Html.EditorFor(m => m.StudentId)</p>
<p>Student Name:    @Html.EditorFor(m => m.StudentName)</p>
<p>Age:             @Html.EditorFor(m => m.Age)</p>
<p>Password:        @Html.EditorFor(m => m.Password)</p>
<p>isNewlyEnrolled:@Html.EditorFor(m => m.IsActive)</p>
<p>Gender:          @Html.EditorFor(m => m.StudentGender)</p>
<p>DoB:             @Html.EditorFor(m => m.CreatedDate)</p>
```

IT-Academy

Html.Editor

StudentId: 1

Student Name: Kate

Age: 18

Password: 12345

isNewlyEnrolled: ☑

Gender: Female

DoB: 29/07/2015 00:00:00

Html Helper

Html.BeginForm

Strongly Typed Html Helper Html Control

<form>

IT-Academy

```
@using (Html.BeginForm("Create", "Home", FormMethod.Post))
{
    <p>
        <label>Name</label><br />
        <input type="text" name="name" />
    </p>
    <p>
        <label>Age</label><br />
        <input type="number" name="age" />
    </p>
    <p>
        <input type="submit" value="Save" />
    </p>
}
```

```
<form method="post" action="~/Home/Create">
    <p>
        <label>Name</label><br />
        <input type="text" name="name" />
    </p>
    <p>
        <label>Age</label><br />
        <input type="number" name="age" />
    </p>
    <p>
        <input type="submit" value="Save" />
    </p>
    }

</form>
```

Html Helper	Strongly Typed Html Helper
Html.ValidationMessage	Html.ValidationMessageFor
Html.ValidationSummary	

IT-Academy

Custom Html Helper

in class

```
public static class ListHelper
{
    public static MvcHtmlString CreateList(this HtmlHelper html, string[] items)
    {
        TagBuilder ul = new TagBuilder("ul");
        foreach (string item in items)
        {
            TagBuilder li = new TagBuilder("li");
            li.SetInnerText(item);
            ul.InnerHtml += li.ToString();
        }

        return new MvcHtmlString(ul.ToString());
    }
}
```

```
@Html.CreateList(new string[] { "a", "b" })
```

Custom Html Helper

in view

```
@model Point[]


@helper PointList(IEnumerable<Point> points)
{
    <ul>
        @foreach (var p in points)
        {
            <li>@p.X, @p.Y, @p.Z</li>
        }
    </ul>
}
```

IT-Academy

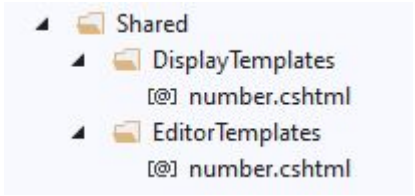
Templates

- Editor
- Display

IT-Academy

Templates

```
[UIHint("number")]
public int X { get; set; }
```




```
@model int

<p class="number">@Model</p>
```

```
@model int

<input type="number" class="number" name="number" value="@Model"/>
```