**Лабораторная работа 4. Разработка представлений**

В этой работе вы разработаете представление с использованием шаблонных страниц и таблицы стилей.

***Упражнение 1. Создание формы для ввода данных с помощью HTML Helpers***

В этом упражнении вы примените HTML хэлперы для генерации HTML разметки.

1. Откройте страницу InputData.cshtml. Внутри тега <div> замените разметку для ввода данных с помощью метода-хелпера Html.BeginForm и двух html-хелперов для ввода информации пользователем (второе поле использует лямбда-выражение):

@using (Html.BeginForm())

{

<table>

<tr>

<td><p>Введите имя:</p></td>

<td>@Html.TextBox("FirstName")</td>

</tr>

<tr>

<td><p>Введите фамилию:</p></td>

<td>@Html.TextBoxFor(x => x.LastName)</td>

</tr>

<tr><td><input type="submit" value="Отправить" /> </td><td></td></tr>

</table>

}

Оба варианта ввода данных идентичен, но использование лямбда-выражений в строго типизированном хелпере считается предпочтительным.

2. Запустите приложение. Проверьте работу страницы ввода данных.

***Упражнение 2. Применение шаблонных страниц***

В этом упражнении вы используете шаблонные страницы (макеты) для задания общего форматирования сайта. Автономные представления, разработанные в предыдущих упражнениях удобны для простых приложений, но реальный проект может иметь множество представлений, и макеты являются эффективными шаблонами, которые содержат разметку, используемую для создания логичности и постоянства в веб приложении.

1. Откройте любой файл представления, например, Index.cshtml и найдите в начале разметки следующее Razor-выражение:

@{

Layout = null;

}

Этот блок кода устанавливает значение свойства Layout на null, в результате чего представление является автономным, и оно будет показывать все содержимое, которое должно вернуться клиенту.

**Создание макета**

2. Для создания макета откройте контекстное меню папки **Views** в обозревателе решения, укажите команду **Add** (Добавить) и далее **New Item** (Создать новый).

3. В отрывшемся окне добавления нового элемента выберите страницу макета Razor, укажите имя \_BasicLayout.cshtml и нажмите кнопку **Add** (Добавить).

Обратите внимание на первый символ имени – символ подчеркивания (\_). Имена макетов, которые являются поддерживающими файлами должны начинаться с подчеркивания.

4. Изучите содержимое созданного макета, обратите внимание на вызов метода @RenderBody, который вставляет содержимое представления, указанного методом действия в разметку макета.

5. Добавьте следующий код для придания общей наглядности:

<title>@ViewBag.Title</title>

...

<body>

<h1>ОАО "Добро пожаловать"</h1>

<div style="padding: 20px; border: solid medium green; font-size: 20pt">

@RenderBody()

</div>

</body>

6. Сохраните изменения.

**Применение макета**

1. Откройте файл представления Index.cshtml.

2. Чтобы применить макет к представлению установите значение макетной страницы свойству Layout и укажите значение для свойства ViewBag.Title, которое будет использоваться в качестве контента элемента title в HTML документе, отправленном обратно пользователю (это не обязательно, но желательно):

@{

Layout = "~/Views/\_BasicLayout.cshtml";

ViewBag.Title = "Главная";

}

3. Удалите все элементы, которые обеспечивают структуру полной HTML страницы:

@{

Layout = "~/Views/\_BasicLayout.cshtml";

ViewBag.Title = "Главная";

}

<div>

@ViewBag.Greeting, спасибо, что зашли

<p>

@Html.ActionLink("Введите имя", "InputData")

</p>

</div>

В итоге, то что осталось ориентировано на показе данных из объекта модели представления пользователю.

4. Запустите приложение и убедитесь, что теперь к первой странице применена стилизация шаблона.

5. Примените макет \_BasicLayout.cshtml к остальным представлениям Hello.cshtml, InputData.cshtml и ListPerson.cshtml.

6. Запустите приложение и убедитесь, что теперь ко всем страницам применена стилизация шаблона.

// Результат

// ОАО "Добро пожаловать"

**Использование макета \_ViewStart**

Жесткая привязка к шаблону в разметке страниц бывает неудобной при практическом применении. Есть возможность для страницы пропустить указание шаблона и в таком случае будет применятся шаблон по умолчанию \_ViewStart.

1. Добавьте в папку **Views** новый файл макета с именем \_ViewStart.cshtml.

2. Замените его содержимое на присвоение свойству Layout готового макета:

@{

Layout = "~/Views/\_BasicLayout.cshtml";

}

3. Во всех представлениях удалите строку присвоения шаблона.

4. Запустите приложение и убедитесь, что ко всем страницам применена стилизация шаблона по умолчанию, а так как он ссылается на старый шаблон, то визуализация страниц не должна измениться.

// Выполнено.

***Упражнение 3. Использование CSS***

В этой части упражнения вы реализуете стилизацию приложения, настроив файл стилей для приложения.

1. Добавьте в проект сайта папку **Content**.

2. В папку **Content** добавьте файл **Таблица стилей** с именем Site.css.

3. Вставьте в файл код настройки стилей:

h1 {

margin: 0px;

padding: 2px 0;

font-size: 30px;

font-weight: bold;

}

body {

font-family: "Trebuchet MS", Verdana, Helvetica, Sans-Serif;

padding: 20px;

border: solid medium green;

font-size: 15pt

}

button[type=submit] {

margin-top: 15px;

padding: 5px;

width: 336px;

}

4. В файле \_BasicLayout.cshtml вставьте код подключения стилей с помощью метода Url.Content(путь к файлу) и удалите стиль, ранее указанный в элементе <div>:

...

<title>@ViewBag.Title</title>

**<link href="@Url.Content("~/Content/Site.css")" rel="stylesheet" type="text/css" />**

...

<div>

@RenderBody()

</div>

5. Запустите приложение и убедитесь, что ко всем страницам применена стилизация шаблона в соответствии с таблицей стилей.

// поменялся размер зелёной рамки – стал растягиваться вслед за полем