
epam
March 2017
ASP.Net MVC View

План занятия

- • Шаблоны представлений
- • Частичные представления
- • Секции
- • Пакеты (bundles)
- • Вспомогательные методы (helpers)
- • Атрибуты HTML

MVC View

- Любое представление может являться шаблоном для других представлений;
- Шаблон также может использовать другое представление в качестве собственного шаблона;
- Путь к шаблону записывается в свойство представления **Layout**;
- Значение **Layout** равное **null** означает, что страница не использует шаблон;
- Место вставки контента в шаблоне обозначается вызовом метода **RenderBody()**.

Подключение и оформление шаблона

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

<h2>Index</h2>
```

```
<!DOCTYPE html>
<html>
<head>
    <title>@ViewBag.Title</title>
    <link href="~/Content/Site.css" rel="stylesheet" />
</head>
<body>
    @RenderBody()

    <script src="~/Scripts/jquery-1.10.2.min.js"></script>
</body>
</html>
```

Порядок формирования страницы

- 1.Формируется **Views/_ViewStart.cshtml**(если есть).
- 2.Формируется указанное контроллером представление. Результат конкатенируется с результатом **_ViewStart.cshtml**.
- 3.Если в **Layout** указано представление (шаблон), формируем его, подставляя полученный ранее результат в место вызова **RenderBody()**.
- 4.Если шаблон содержит **Layout**, повторяем пункт3.

ViewStart и шаблоны

- _ViewStart может содержать серверные вставки, включая установку шаблона представлений по умолчанию в **Layout**.
- Как правило, основной шаблон размещают в **Views/Shared/_Layout.cshtml**.
- Значение **Layout** можно изменить в любой момент при формировании страницы.
- Явно установленное значение **null** отменяет использование шаблона представления.

Частичные представления

- Частичные представления — это обычные представления, встраиваемые в произвольные места любой другой страницы.
- Частичные представления формируются без использования **_ViewStart.cshtml**.
- Частичные представления могут либо опираться на модель основной страницы, либо представлять конкретную модель.

Способы включения частичных представлений

- Непосредственное включение:**Html.Partial**(представление,модель);
- Дочерний action-метод:**Html.Action**(метод,контроллер,параметры);
  - Метод желательно пометить атрибутом **[ChildActionOnly]**;
  - Метод должен вернуть **PartialViewResult** или **ContentResult**.

```
[ChildActionOnly]
public ActionResult News()
{
    return PartialView();
}
```

Шаблоны против частичных представлений

- Шаблон включает ровно одну страницу-тело. Число частичных представлений ничем не ограничено.
- Страница явно привязана к конкретному шаблону, но шаблон не знает, для каких страниц он применяется. Страница явно указывает используемые частичные представления, но они не знают, на каких страницах они будут использоваться.
- Частичные представления могут как работать с моделью базового представления, так и опираться на собственные модели.

Секции

- Секции позволяют делать вставки фрагментов страницы в разные участки шаблона.
- Секции могут быть как обязательными, так и опциональными.
- Страница не может содержать неиспользуемые или повторяющиеся секции.
- Шаблон не может включать одну и ту же секцию дважды.
- Секции частичных представлений добавляются к секциям основной страницы.

Описание и включение секции

- Описание секции в представлении:

```
@section scripts {
    <script src="~/Scripts/bootstrap.js"></script>
}
```

- Включение опциональной секции в шаблон:

```
@RenderSection("scripts", required: false)
```

- Включение обязательной секции в шаблон:

```
@RenderSection("scripts")
```

Пакеты (bundles)

- Пакеты предназначены для гибкого подключения набора однотипных файлов.
- Различают пакеты стилей (StyleBundle) и скриптов (ScriptBundle).
- Поддержка пакетов реализована в NuGet- пакете Microsoft ASP.NET Web Optimization Framework.

Формирование пакета

- Создание пакета:

```
var bundle = new StyleBundle("~/Content/css");
```

- Добавление файлов:

```
bundle.Include(
            "~/Content/bootstrap.css",
            "~/Content/site.css");
```

- Включение пакета в коллекцию:

```
BundleTable.Bundles.Add(bundle);
```

- Подключение пакета:

```
@Styles.Render("~/Content/css")
```

Дополнительные возможности

- Имя подключаемого файла (стиля, …) может включать заместитель версии — {version}.
  - К примеру, jquery-{version}.js захватит как jquery-1.10.2.js, так и jquery-2.1.1.js.
- Имя подключаемого файла может включать универсальный заменитель *
  - * может быть либо в начале, либо в конце имени файла (включая расширение)
  - * должен быть один
  - имя не может состоять только из *
- Можно подключить все файлы из каталога, используя метод IncludeDirectory().

Оптимизация файлов пакета

- Технология пакетов поддерживает оптимизацию включённых в пакет файлов.
- Оптимизация включает в себя минификацию файлов и соединение их в один файл с генерированным уникальным именем.
- Для включения оптимизации нужно:
  - **BundleTable.EnableOptimizations = true**;
  - или убрать атрибут **debug="true"** элемента **<compilation>** в секции **<system.web>** файла конфигурации.

http://www.asp.net/mvc/tutorials/mvc-4/bundling-and-minification

Вспомогательные методы

- Razor поддерживает несколько видов вспомогательных методов:
  - обычный метод, возвращающий MvcHtmlString
  - методы в секции @functions
  - специальный синтаксический блок @helper

Обычные вспомогательные методы

- Для удобства их зачастую делают методами расширения.

```
namespace System.Web.Mvc
{
    public static class HtmlHelperExtensions
    {
        public static MvcHtmlString Submit<T>(
            this HtmlHelper<T> html, string name)
        {
            var format = @"<input type=""submit"" name=""{0}"" />";
            var htmlString = string.Format(format, name);
            return new MvcHtmlString(htmlString);
        }
    }
}
```

```
@Html.Submit("save")
```

Методы в секции @functions

- Применяются, если метод нужен только на этом представлении.

```
@functions {
    MvcHtmlString Submit(string name)
    {
        var format = @"<input type=""submit"" name=""{0}"" />";
        var htmlString = string.Format(format, name);
        return new MvcHtmlString(htmlString);
    }
}
```

```
@Submit("save")
```

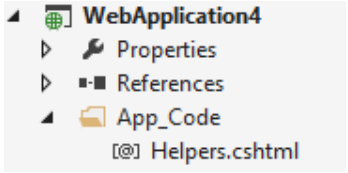
MvcHtmlString _Page_Views_Home_Index_cshtml.Submit(string name)

Специальный блок @helper

- Позволяет использовать синтаксис Razor.

```
@helper Submit(string name) {
    <input type="submit" name="@name" />
}
```

- Набор таких блоков может быть вынесен в отдельный файл в каталог App_Code.




```
@Helpers.Submit("save")
```

Стандартные вспомогательные методы

- Работа с вёрсткой.
  - -Классы: HtmlHelper и HtmlHelper<TModel>
  - -Доступ: свойство представления Html
- Работа с адресами.
  - -Класс: UrlHelper
  - -Доступ: свойство представления Url

Методы HtmlHelper

Метод	Назначение
Action	Рендерингрезультата дочернего метода действия
ActionLink	Текстс гиперссылкой на метод действия
AntiForgeryToken	Специальное поле, препятствующее подмене формы
AttributeEncode	Преобразуетобъект в набор атрибутов
BeginForm	Форма, отправляемая в метод действия
BeginRouteForm	Форма, отправляемаяпо именованному маршруту
CheckBox(For)	Флажок(длясвойства модели)
Display(For)	Отображение свойства модели
DisplayName(For)	Имя свойства модели
DropDownList(For)	Выпадающий список(для свойства модели)
Editor(For)	Редактирование свойства модели
EnumDropDownListFor	Выпадающийсписок для enum
Hidden(For)	Скрытоеполе (для свойства модели)

Методы HtmlHelper

Метод	Назначение
Label(For)	Метка с именем(для свойства модели)
ListBox(For)	Список(для свойства модели)
Partial	Частичное представление
Password(For)	Поле ввода пароля (для свойства модели)
RadioButton(For)	Переключатель(для свойства модели)
Raw	«Сырая» разметка
RouteLink	Текст с гиперссылкой на именованный маршрут
TextArea(For)	Текстовая область (для свойства модели)
TextBox(For)	Текстовое поле (длясвойства модели)
ValidationMessage(For)	Результат валидации поля (для свойства модели)

Методы UrlHelper

Метод	Назначение
Action	Путь к указанному методу действия
Content	Конвертирует виртуальный путь в абсолютный
Encode	Заменяет специальные символы в %xx последовательности
HttpRouteUrl	Путь к указанному именованному маршруту
IsLocalUrl	Проверка, является ли указанный путь локальным
RouteUrl	Путь к указанным параметра маршрута

Атрибуты HTML

- При использовании стандартных методов, генерирующих элементы вёрстки, может потребоваться добавление дополнительных атрибутов тегов:

  -идентификаторы;

  -bootstrap;

  -data-attributes;

  -…

```
<input type="text" name="Login"
       class="form-control" data-id="login" />
```

Способы передачи атрибутов

- •Если метод поддерживает htmlAttributes:

```
@Html.TextBox("Login", "",
```

▲ 3 of 7 ▼ (extension) MvcHtmlString HtmlHelper.TextBox(string name, object value, **object htmlAttributes**)
Returns a text input element by using the specified HTML helper, the name of the form field, the value, and the HTML attributes.
htmlAttributes: *An object that contains the HTML attributes to set for the element.*

```
@Html.TextBox("Login", "",
    new { @class = "form-control", data_id = "login" })
```

- •Если метод не поддерживает htmlAttrbutes, но поддерживает additionalViewData:

```
@Html.Editor("Login", "",
```

▲ 4 of 6 ▼ (extension) MvcHtmlString HtmlHelper.Editor(string expression, string templateName, **object additionalViewData**)
Returns an HTML input element for each property in the object that is represented by the expression, using the specifie
additionalViewData: *An anonymous object that can contain additional view data that will be merged into the System*

```
@Html.Editor("Login", "", new { htmlAttributes =
    new { @class = "form-control", data_id = "login" } })
```

Ссылки

https://www.tutorialspoint.com/asp.net_mvc/asp.net_mvc_views.htm

http://www.asp.net/mvc/tutorials/mvc-4/bundling-and-minification

https://metanit.com/sharp/mvc5/4.1.php

Спасибо за внимание

Контактная информация:
Лектор: **Павел Титов**
EPAM Systems, Inc.
Адрес: Астана,
Email: Pavel_Titov@epam.com
http://www.epam.com