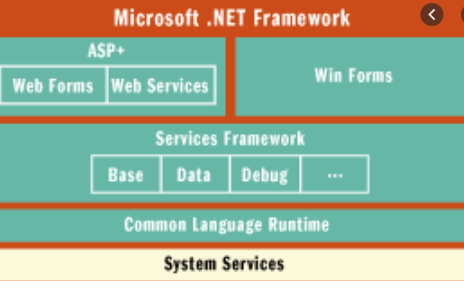
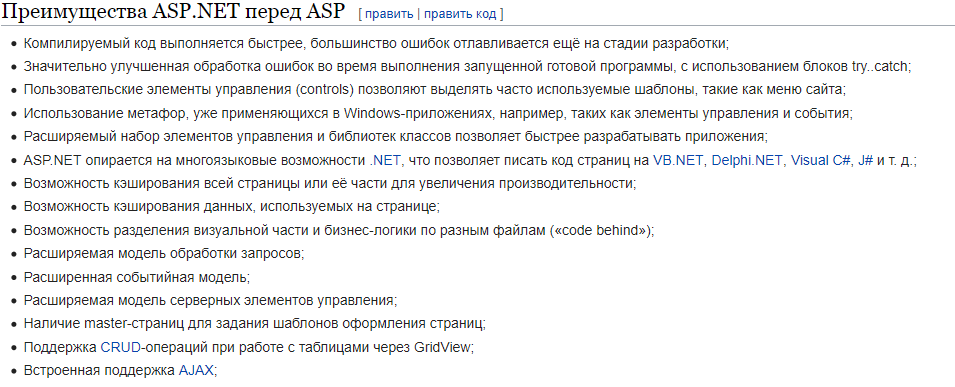
1. ASP.NET(Active Server Pages) – технология создания web-приложений и web-сервисов компании Microsoft. Технология является составной частью Microsoft.NET



Является развитием более старой технологии Microsoft ASP.

ASP.NET существенно отличается от ASP, поскольку основан на платформе .NET.



Счёт версий ASP.NET идёт параллельно версиям .NET Framework.

ASP.NET компилируется также, как и любое приложение .NET.

ASP.NET поддерживает 3 языка C#, VB.NET, MSIL

ASP.NET приложение исполняется CLR

Программная модель ASP.NET основывается на протоколе HTTP и использует его правила для взаимодействия между сервером и браузером. При формировании страницы заложена абстрактная программная модель Web Forms и на ней основана основная часть реализации программного кода.

Solutions Explorer, Refrences

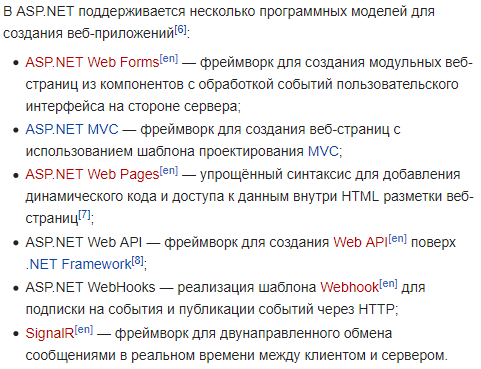
Default.aspx (режимы: конструктор, разделить, исходный код)

Тег Page, атрибут CodeBehind.

Default.aspx.cs, namespace, partial class

Default.aspx.designer.cs

Web.config



Приложение ASP.NET исполняется CLR.

Типы приложений: web form, mvc, web api, asmx-сервисы, wcf-сервисы

Простая и базовая разница заключается в том, что ASMX-сервис предназначен для отправки и получения сообщений используя SOAP только через HTTP. В то же время WCF-сервис может обмениваться сообщениями используя любой формат, (SOAP является стандартным) через любой транспортный протокол (HTTP, TCP/IP, NamedPipes и т.д.)

1. ASMX-сервисы могут быть размещены только в IIS, в то время как WCF-сервисы имеют следующие варианты хостинга: (IIS, WAS, Console Applocation, Windows NT Services)
2. ASMX-сервисы ограничены в поддержке одного HTTP, WCF – поддерживается HTTP, TCP, Named Pipes
3. ASMX-сервисы исп. Для сериализации класс XmlSerializer, WCF-сервисы. DataContractSerializer, который лучше по производительности.

Недостатки XmlSerializer: только публичные поля или свойства могут быть переведены в XML, Только классы, которые реализовывают интервейс IEnumerable могут быть переведены, Классы, которые реализуют IDictionary, такие как Hashtable, не могут быть сериализованы.

1. Что содержится в файле aspx?

Созданная сервером веб-страница, содержащая сценарии, которые обрабатываются на веб-сервере

Другое название – веб форма

Файл active server page extended, содержащий код VbScript или С#, также XHTML-разметку

Aspx.cs, aspx.designer.cs

1. Что содержится в файле aspx.cs?

Часть кода страницы, обработчики событий страницы

1. Как связаны файлы aspx и aspx.cs?
2. Что содержится в файле aspx.designer.cs?
3. Что содержится в файле Web.config?
4. Что означает **partil class**?

Можем иметь несколько файлов с определением одного и того же класса, и при компиляции все эти определения будут скопмилированы в одно (не применяется на делегат и перечисление)

1. Что такое пространство имен (namespace) и как используется?
2. Опишите назначение оператора **using**.
3. Что такое CLR?
4. Что такое MCIL?
5. Что такое JIT-компилятор?
6. Что такое .NET FRAMEWORK?
7. Что такое FCL?
8. Что такое, сборка .NET и каких видов они бывают.
9. Что такое интерфейс (interface) в C#?
10. Как создать ссылку в приложении на DDL-сборку?
11. Какой класс является базовым для класса \_Default?
12. Что означает протокол http является statless-протоколом?
13. Перечислите основные группы статуса ответа http-протокола.
14. Перечислите основные методы запроса http-протокола, в чем основное отличие GET и POST-запросов.
15. Что такое заголовок запроса/ответа и для чего он применяется?
16. Что такое IIS?

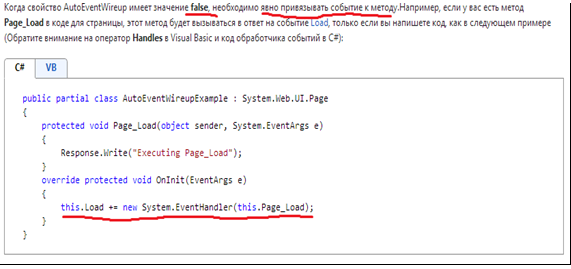
Web-форма: часть веб-приложения, интерпретируется сервером, результат интерпретации: html и js-code. Aspx-page.



Inherits: на основе страницы генерируется класс производный от указанного в Inherits.

CodeBehind: файл, содержащий обработчики событий страницы

AutoEventWireUp, когда значение false, необходимо явно привязывать событие к методу.



VIEWSTATE

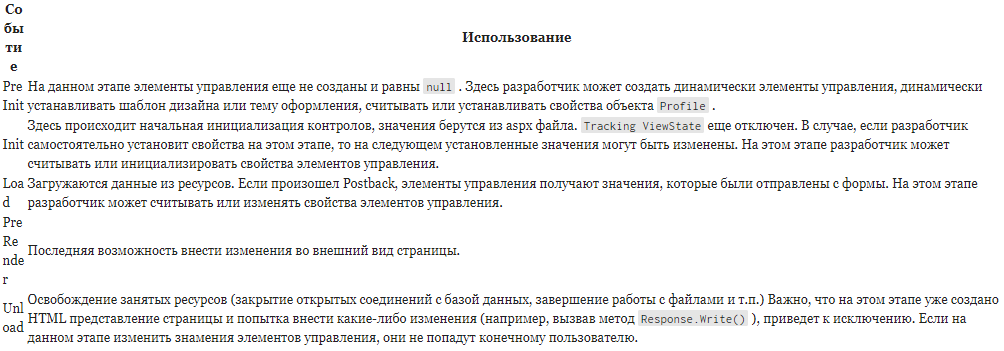


EVENTVALIDATION



Web-приложения: выполняются на сервере

Не поддерживают состояние.



maxPageStateFieldLength – некоторые брандмауэры и прокси-серверы не пропускают скрытые поля больше некоторой длины.

Брандмауэр – Нужен для защиты системы от компьютерных атак.

Прокси-сервер – википедия

События web-form

Init

Load

Prerender

Unload

Disposed

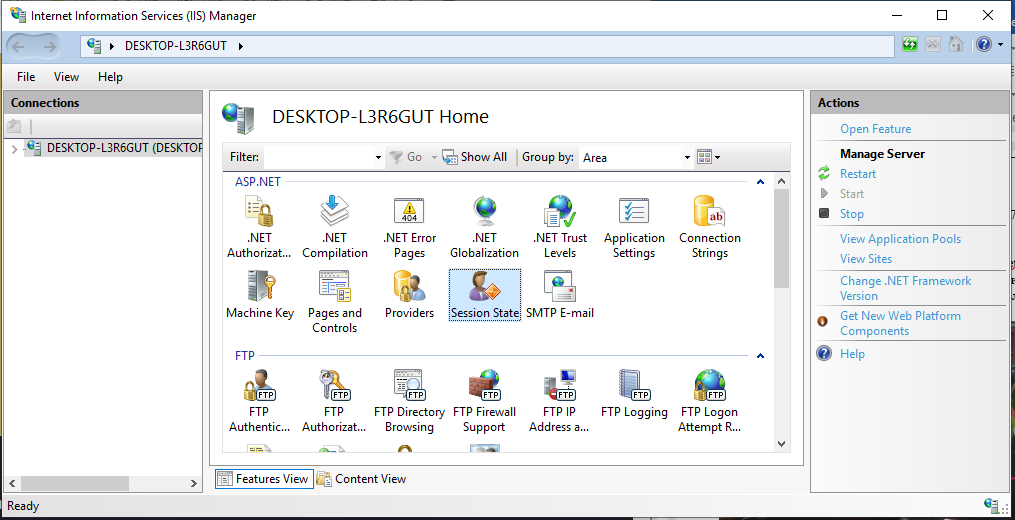
Autopostback – автоматическая обратная отправка данных

1. Что значит: http statless-протокол?
2. Объясните способ сохранения состояния элементов aspx-формы с помощью тега viewstate.
3. Объясните назначение и способ применения параметра maxPageStateFieldLength.
4. Перечислите основные события web-формы их назначение и порядок обработки.
5. Объясните свойство Autopostback, которое может быть установлено для некоторых элементов формы.
6. Перечислите основные системные параметры, которые могут быть получены внутри web-формы.

IIS(Internet Information Services)

Версии 5.x(windows 2000\xp) 6.0(windows 2003), iis 7.0(windows 2008\vista), iis 8.5(windows 2012), iis 10(windows 10, windows 2016)

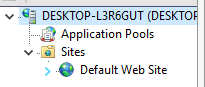
При установке автоматически создает каталог C:\inetpub\wwwroot, который представляет ваш веб-сайт. Можно добавить доп. Страницы, их надо скопировать в этот каталог и обращаться к ним через <http://localhost/testfile.html>. Можно создавать вложенные папки.



Слева – имя используемого сервера.

Центр – представление сервера. Набор значков, которые помогают конфигурировать параметры сервера.

Справа – список доступных действий. Можно запускать, останавливать и перезапускать сервер.



Если развернуть элемент сервера, мы увидим Sites, содержащий единственную запись Default Web Sites. Сайт – набор файлов и каталогов, образующих веб-сайт.

На одном сервере IIS может поддерживать несколько сайтов, как правило, на различных портах (по умолчанию 80).

Сочетание имени сервера и порта сайта образуют первую часть url-адреса.

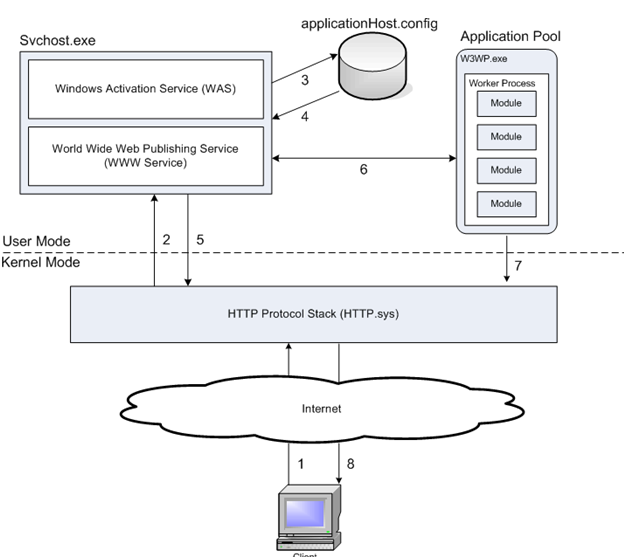
Порт можно поменять в “Изменить привязки” (Bindings)

Можно развернуть веб-сайт с помощью копирования файлов, на сервер.

Развертывание с помощью Visual Studio

VS предлагает несколько возможностей для интеграции с IIS:

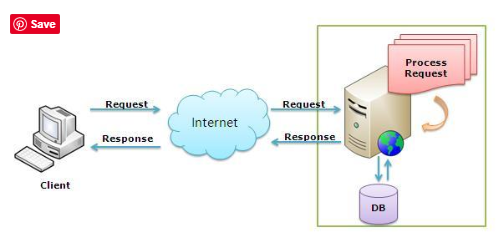
* Можно создавать виртуальный каталог при создании нового проекта
* Можно использовать функцию copy web site для передачи существующего веб-сайта в виртуальный каталог
* Можно использовать функцию publish web site чтобы скомпилировать свой сайт и передать его по новому адресу



1. Браузер обращается к веб-серверу по определённому URL, на стороне сервера запрос перехватывает драйвер HTTP.SYS(Http protocol stack)
2. HTTP.SYS стучится к WAS(Windows Activation Service) для получения информации из хранилища конфигурации
3. Служба WAS запрашивает конфигурацию из хранилиза – из файла в папки IIS applicationHost.config
4. Поскольку данный запрос получен по протоколу HTTP конфигурационную информацию получает служба W3 SVC (она же WWW Service), эта информация содержит в себе данные о пуле приложений и прочих параметрах сайта
5. Служба W3 SVC использует эту информаию для конфигурации HTTP.SYS
6. Служба WAS запускает процесс W3WP.exe для пула приложений, если он ещё не был запущен
7. В процессе W3WP.exe работает приложение веб-сайта, которое, собственно, формирует и возвращает ответ драйверу HTTP.SYS
8. HTTP.SYS отправляет ответ браузеру

Веб-сервер отвечает за обработку всех запросов, их обработку и предоставление ответов.

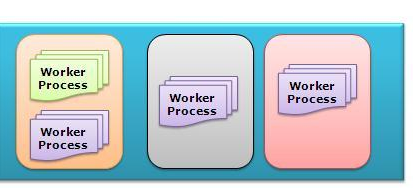
IIS – сервер от Microsoft, который используется для размещения вашего веб-приложения.

IIS имеет собственный ASP.NET Process Engine для обработки запроса ASP.NET. Таким образом, когда запрос поступает от клиента к серверу, IIS принимает этот запрос, обрабатывает его и отправляет ответ клиентам. 

Два понятия: 1) рабочий процесс, 2) пул приложений

Рабочий процесс w3wp.exe – запускает приложение asp.net в iis. Отвечает за управление всеми запросами и ответами. Все функциональные возможности asp.net выполняются в рамках рабочего процесса. Когда на сервер поступает запрос, рабочий процесс отвечает за генерацию запроса и ответа.

Пул приложений – контейнер рабочего процесса. Пулы приложений используются для разделения наборов рабочих процессов, с одинаковой конфигурацией. Когда один рабочий процесс сталкивается с проблемой или перезагружается, другие не затрагиваются. Приложение не влияет на другое, т.к. они настроены в разных пулах.

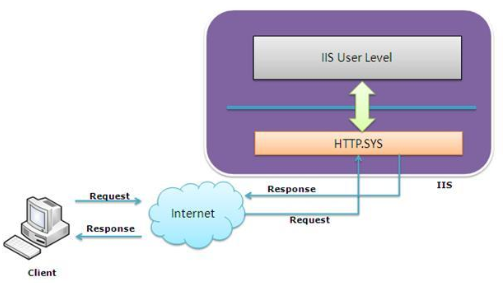


Пул приложений с несколькими рабочими процессам называется Web-garden

Обработка нового запроса

Два уровня: 1) режим ядра, 2) режим пользователя

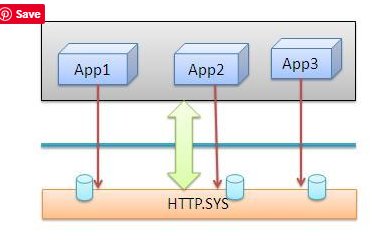
Режим ядра содержит HTTP.SYS. Всякий раз, когда запрос приходит от клиента к серверу, он сначала обращается к HTTP.SYS



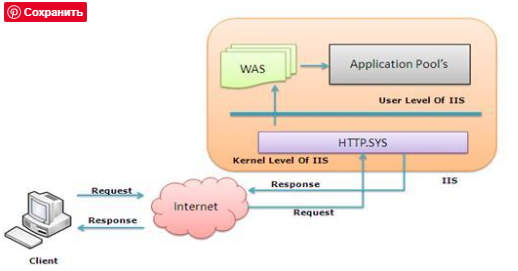
Теперь HTTP.SYS отвечает за передачу запроса в определённый пул приложений

Как узнаёт, куда отправить запрос?

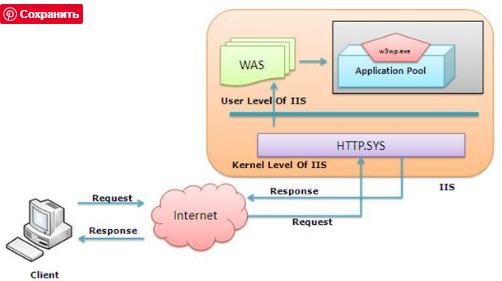
Всякий раз когда мы создаём новый пул приложений, генерируется идентификатор пула приложений, и он регистрируется в HTTP.SYS. Поэтому всякий раз, когда HTTP.SYS получал запрос от любого веб-приложения, он проверял пул приложений и на основе пула приложений отправлял запрос



На уровне пользователя IIS есть WAS(web admin services), который принимает запрос от HTTP.SYS и передаёт его в соответствующий пул приложений



Когда пул приложений получает запрос, он просто передаёт запрос рабочему процессу w3wp.exe. Рабочий процесс ищет url-адрес запроса для зарузки правильного расширения ISAPI. Расширение ISAPI – способ IIS для обработки запросов на разные ресурсы. После установки ASP.NET он устанавливает собственное расширение ISAPI(aspnet\_isapi.dll) и добавляет сопоставление в IIS.



Когда рабочий процесс загружает aspnet\_isapi.dll, он запукает HTTPRuntime, который являет точкой входа приложения. HTTPRuntime – класс, который вызывет метод ProcessRequest для запуска обработки.

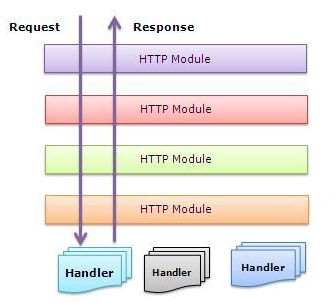
Когда вызывается этот метод, создаётся новый экземпляр HTTPContext. Который доступен через HttpContext.Current. Этот объект остаётся живым в течение всего времени запроса объекта. Используя этот объекта HttpContext.Current – можно получить доступ к некоторым другим объектам(request, response, session)

Каждый запрос должен проходить через сответствующий модуль HTTPModule для доступа к HttpHandler, тот список модуля настраивается приложением HttpApplication.

Теперь эта концепция называется HTTPPipelone. Она называется конвейером, потому что содержит набор HttpModule, которые перехватывают запрос на пути к httphandler.

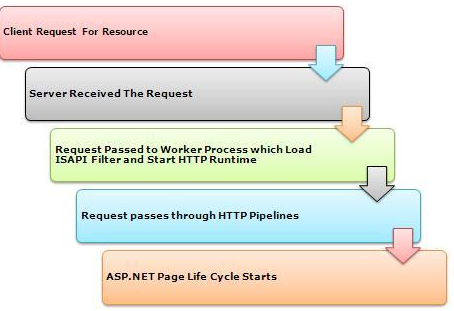
HttpModules – классы, которые имеют доступ к входящему запросу.

Можем создать свой модуль, если нужно что-то обработать во время предстоящего запроса и ответа.



Http-обработчики являются конечными точками в конвейере http. Все запросы проходящие через httpmodule, должны доходить до httphandler. http-обработчик генерирует вывод для запрошенного ресура. Т.о., когда мы запрашивает любые веб-страницы, он возвращает соответствующий вывод html.

Теперь весь запрос переходит из httpModule в соотв. Httphandler. Затем запускается метод и жизненный цикл страницы asp.net.



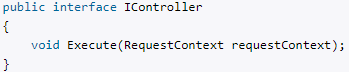
Когда клиент запрашивает некоторую информацию с веб-сервера, запрос сначала достигает HTTP.SYS IIS. Затем HTTP.SYS отправляет запрос в определённый пул приложений. Затем пул приложений пересылает запрос рабочему процессу для загрузки расширения ISAPI, которое создаст объект HTTPRuntime для обработки запроса через HTTPModule и обработчик HTTP. После этого запускаются события ASP.NET page LifeCycle.

Контроллер

Получает ввод пользователя, обрабатывает его и посылает обратно результат обработки. Например, в виде представления.

Контроллер – класс, наследуется от System.Web.Mvc.Controller. Controller реализует абстрактный класс ControllerBase, а через него и интерфейс IController.

Интерфейс IController содержит единственный метод Execute(отвечает за обработку контекста запроса)



При обращению к любому контроллеру система передаёт в него контекст запроса. В этот контекст запроса включается: куки, данные формы, строки запроса, данные пользователя и др.

Методы действий и их параметры

Акции – такие методы контроллера, который обрабатывают запросы по определённому url.

Акции всегда имеют модификатор public, private – точно нет.

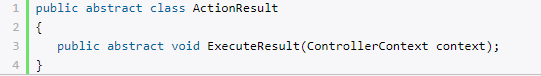
В контроллере могут быть и обычные методы, которые могут использоваться в вспомогательных целях.

Название параметров метода, должно совпадать с названием параметров в строке запроса.

Можем получить данные с контекста запроса, доступны следующие объекты: Request, Response, RoutedData, HttpContext и Server.

Результаты действий

ActionResult – абстрактный класс, в котором определён один метод ExecuteResult, переопределяемый в классах наследниках.



Производные от ActionResult

ContentResult -> return Content(string)

EmptyResult – отправляет пустой результат

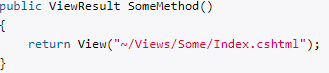
HttpStatusCodeResult – высылает статусный код ошибки.

ViewResult – производит рендеринг представления в веб-страницу и возвращает её в виде ответа клиенту. Return View([“name view”]);

/Views/Name\_Controller/Name\_View.cshtml

Если имя представление не указано явно, то будет использоваться то, которое совпадает с названием акции контроллера.

Можем полностью преопределить путь, по которому система будет искать представление



Если такого пути не будет, ошибка!

Передача данных из контроллера в представление

ViewData – словарь из пар ключ-значение

Производный от ViewDataDictionary – является типом словаря

Жизнь во время текущего запроса. Значения будут очищены если произойдёт перенаправление

Значение преобразовать перед использованием.

Внутренне, ViewBag вставляет данные в словарь ViewData, поэтому ключ ViewData и свойство ViewBag не должны совпадать!

ViewBag – позволяет установить любое свойство и присвоить ему любое значение.

Передаёт данные только с контроллера.

Жизнь во время текущего запроса.

Значение будет нулевым если происходит перенаправление!

Обёртка вокруг ViewData, если ключ ViewData и ViewBag совпадает – ошибка.

Свойства определяются динамически.

Нельзя передавать динамические значения из ViewBag в методы расширения(хелперы) в представлениях. Например, мы не можем написать так @Html.TextBox(“Name”, ViewBag.Name), так как компилятор должен знать тип каждого параметра во время компиляции, чтобы выбрать нужный метод расширения. В этом случае либо использовать ViewData[“Name”], либо преобразовать (string)ViewBag.Name

Почитать про во время компиляции и во время выполнения приложения!

Если определить с одинаковым ключам и ViewBag, но с разными значениями, то будет 2 одинаковых значения и приоритет будет у того, кто объявлен последним!

TempData – словарь, как и ViewData, но отличается. TempData позволяет сохранять переданное значение в течении всего текущего запроса

Внутренне использует Сессию.

Используется для хранения временных данных, которые можно использовать в последующем запросе. Будет очищен после последующего запроса.

TempData.Keep(); - для сохранение данные в последующем запросе!

Производный от – TempDataDictionary

Полезен, когда хотим перенести нечувствительные данные из одной акции, в другую этого же или другого контроллера, а также данные будут сохранены во время перенаправления.

Перед использованием преобразовать в соответствующий тип.

Используется для однократных сообщений, таких как сообщениях об ошибках, проверки.

Переадресация – временная и постоянная

ЧИТАТЬ ЧТО ЭТО ТАКОЕ

301 – постоянная

302 – временная

RedirectResult – объект для переадресации

Redirect, RedirectToAction – метод временной переадресации

RedirectPermanent, RedirectToActionPermanent – метод постоянной переадресации

Есть ещё один класс RedirectToRouteResult

RedirectToAction – метод позволяющий перейти к акции определённого контроллера



RedirectToRoute – метод позволяющий перейти к определённому маршруту внутри домена



FileResult – предназначен для отправки файлов клиенту.

Абстрактный класс, дело будем иметь с наследниками.

FileContentResult – отправляет клиенту массив байтов, считанный из файла

FilePathResult – напрямую отправляет файл с сервера

FileStreamResult – класс, создаёт поток Stream, с помощью которого считывает и отправляет файл клиенту.

Во всех трёх случаях используется для отправки метод File, который возвращает объект FileResult.

ЧИТАТЬ СЕССИЯ И КУКИ

АСИНХРОННЫЕ МЕТОДЫ

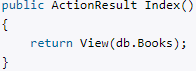
Представления – файл с расширением cshtml

При компиляции приложения на основе требуемого представления сначала генерируется класс на языке C#, зачем этот класс копмилируется.

Все добавляемы представления, как правило, группируются по контроллерам в соответствующие папки в каталоге Views.

ЧТО ТАКОЕ RAZOR, движок представлений

Строго типизированные представления – позволяют передавать данные напрямую в представление через параметр метода View.



Чтобы связать представление с передаваемым параметром, надо добавить в представление директиву @model с указанием типа передаваемых параметров

Объект Model представляет тип модели, указанной в директиве @model, и будет хранить переданные из контроллера данные.

Мастер-страницы используются для создания единообразного вида сайта

Мастер-страницы – это те же самые представления, но позволяющие включать в себя другие представления.

Можно определять общие для всех остальных представлений элементы, можно подключить общие стили и скрипты.

Views/Shared, там находится по умолчанию \_Layout.cshtml

@RenderBody – на его место будут подставляться другие представления.

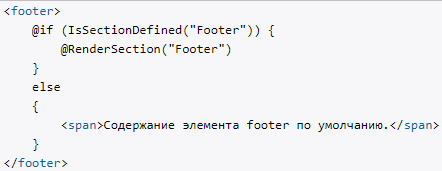
Можно создавать много мастер-страниц

Мастер-страница может иметь несколько секций, куда представления могут поместить своё содержимое.

Представление должно передавать содержание для каждый секции мастер-страницы, иначе ошибка!

2 одинаковые секции создать нельзя!

Но если у нас много представлений и они используют мастер-страницу, где много секций, каждому представлению тогда надо определять секцию, обход:

1. @RenderSection(“Footer”, false) – значит, что не обязательно определять
2. Можно задать содержимое секции по умолчанию, если данная секция не определена в представлении
3. 

ViewStart – указывает на мастер-страницу, которая будет использоваться по умолчанию

\_ViewStart.cshtml – код этой страницы выполняется до когда любого из представлений, расположенных в том же каталоге.

Частичные представления

Кроме обычных представлений, акции могут возвращать частичные представления.

Особенность, можно встраивать в другие обычные представления

Объект PartialViewResult – отвечает за рендеринг частичных представлений.

Возвращается методом PartialView

Не использует мастер-страницу

Можем использовать строго-типизированные частичные представления

Если встраиваем частичное в обычное, и хотим передать данные в частичное, допустим ViewBag – тогда следует ViewBag определить в обычном представлении, тогда эти данные будут доступны как для обычного, так и для частичного представления

Помимо хелпера Html.Partial(“name\_partial”), чтостичное представление можно встроить с помощью другого хелпера, Html.RenderPartial. – этот хелпер также принимает имя представления, только используется не в строчных выражениях кода Razor, а в блоке кода – то есть обрамляется фигурными скобками.

@ { Html.RenderPartial(“name\_partial”) }

RenderPartial – напрямую пишет в выходной поток, работает чуть быстрее

Html хелперы

Есть строчные хелперы, похожи на обычные методы С#, только начинаются с тега @helper

Можем определить в любом месте представления и использовать также в любом.

Удобно использовать, если необходимо создать один метод, которые предполагается использовать в представлении многократно.

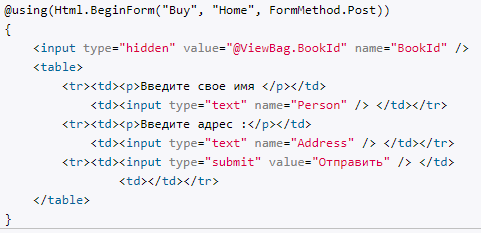
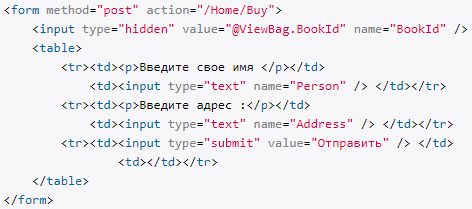
Чтобы не захламлять разметку объёмными хелперами, выносить в отдельные файлы

App\_Core – зарезервированное место для хранения хелперов.

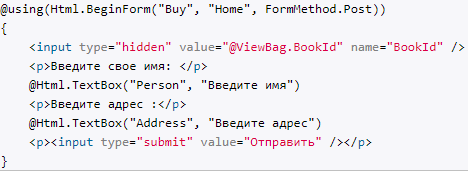
Встроенные хелперы

Прочитать про формы. Пример с блокнотом

@Html.BeginForm



@Html.TextBox



@Html.Hidden, Html.TextArea и др.

Строго-типизированные хелперы

Принимает в качестве параметра лямбда-выражение, в котором указывается свойство модели, к которому должен быть привязан данный хелпер

Могут использоваться только в строго-типизированных представлениях

Добавляется суфикс For



Модели

Модели – классы, описывают логику данных

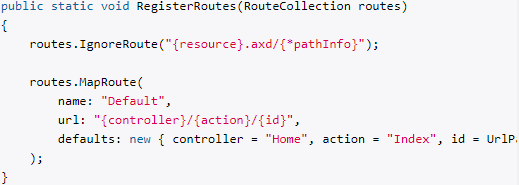
Манипуляция с данными и бизнес-логкига – сфера контроллера

Может иметь конструктор, методы. Но главное не перегружать класс модели.

Маршрутизация

Все определения маршрутов находятся в RouteConfig.cs, каталог App\_Start

Статический метод RegisterRoutes занимается установкой маршрутов



В файле Global.asax в методе Application\_Start – Вызывается метод RegisterRoutes

Первая строка – отключает обработку запросов, например с расширением .axd

Далее идёт определение маршрута по умолчанию

Routes.MapRoute(name, url, defaults) – выполняет сапоставление маршрута запроса.

Можем перегрузить метод и добавить доп. Параметры

При получении запроса, механизм маршрутизации парсит стоку url и помещает значения маршрута в словарь, объект RouteValueDictionary, доступный через контекст приложения RequestContext.