ЛАБОРАТОРНАЯ РАБОТА №5

АРХИТЕКТУРНЫЕ ПАТТЕРНЫ.

MVC

Заменить компонент взаимодействия с пользователем (Presentation Layer) из предыдущей лабораторной работы с WinForms на Asp.net MVC.

Для выполнения работы потребуется вспомнить элементы разметки html (div, p, a, table, form, input type=’submit’ и т.д.) из курса «web сайты».

Содержание работы:

1. На уровень Presentation Layer добавить проект типа asp.net MVC. Он так же, как и winform, взаимодействует с инфраструктурными слоями: добавить ссылки на компоненты бизнес-логики BLL , взаимодействия с данными DAL и на интерфейсы.
2. Изменить базовую начальную страницу в соответствии с предметной областью варианта задания. На начальной странице должны присутствовать ссылки на список объектов предметной области и отчеты.
3. Создать метод действия и представление для списка объектов.
4. Создать методы действий и представления для CRUD.
5. Создать методы действий и представления для отчетов.
6. Внедрить зависимости для asp.net mvc с помощью Ninject.

В результате получится вэб-приложение, работающее с компонентами BLL, DAL из предыдущей л.р. Продемонстрировать работу приложения.

Вопросы для защиты л.р.:

1. Объяснить на примере приложения механизм взаимодействия компонентов паттерна MVC.

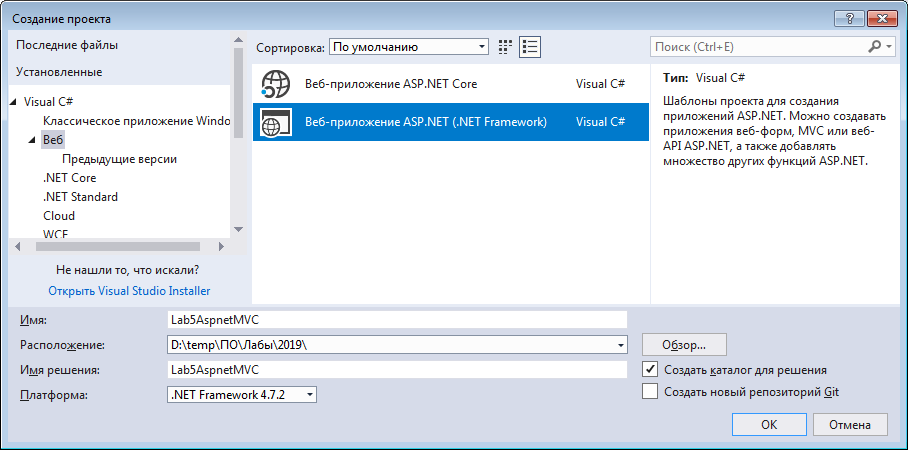
2. Использование компоновок в MVC ASP.net

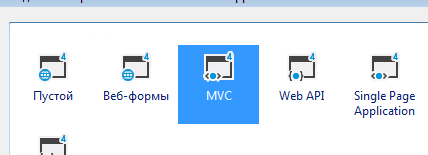
3. Почему метод действия контроллера может быть разбит на два (или более) метода?

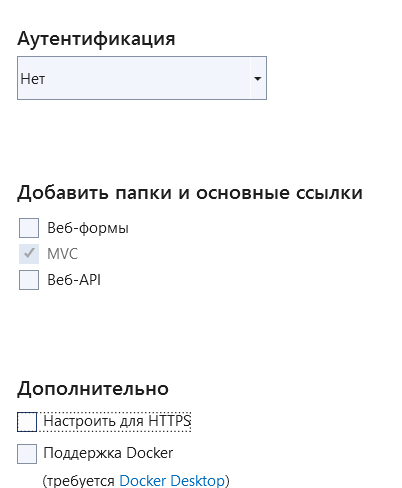
Основные этапы работы.

*Часть 1. Создание приложения Asp.net MVC.*

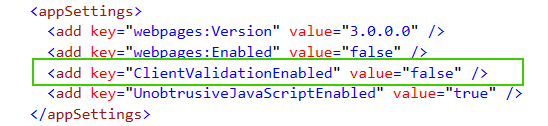
1. Для уровня представления будем использовать в решении каталог Presentation.
2. Создать приложение asp.net MVC или в новом решении или можно добавить проект MVC в существующем решении л.р.№4 .



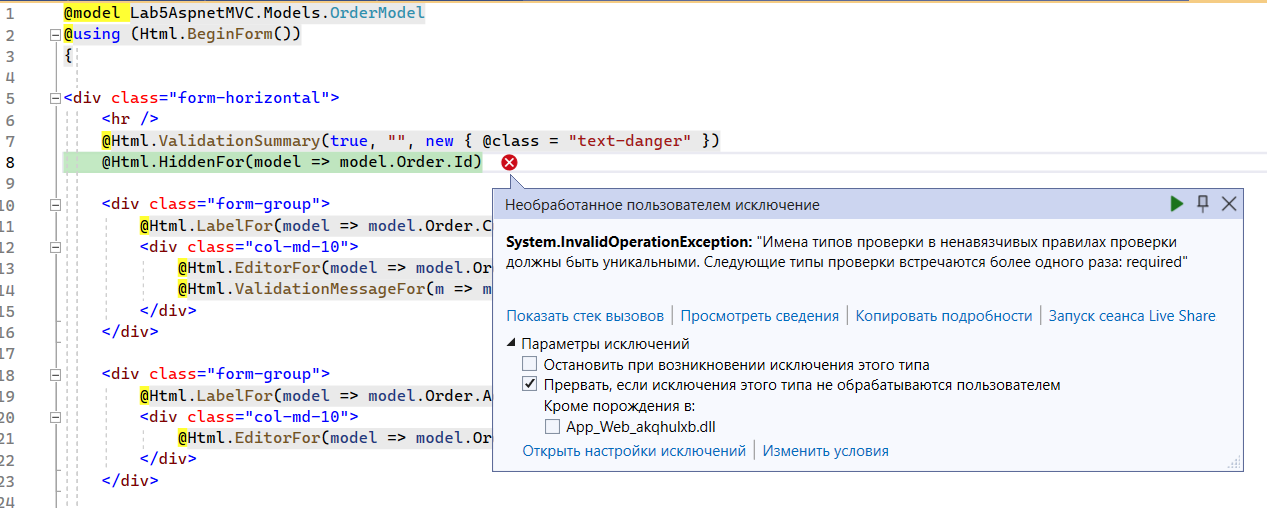




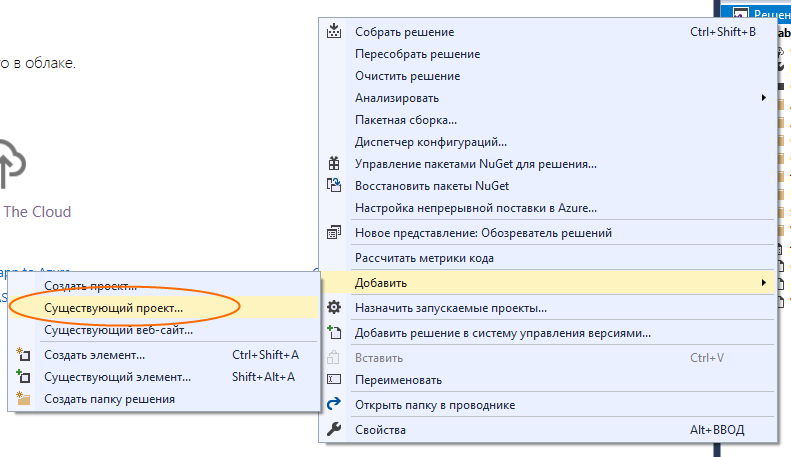
1. Отключить клиентскую валидацию в настройках web.config



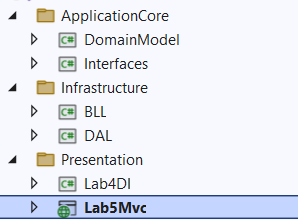
Этот параметр отключает проверки такого плана:



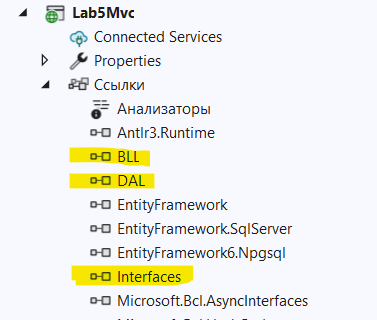
1. Если создано новое решение, то необходимо скопировать в каталог решения папки BLL , DAL , DomainModels, Interfaces, Lab4DI из л.р.№4 и включить их в проект.



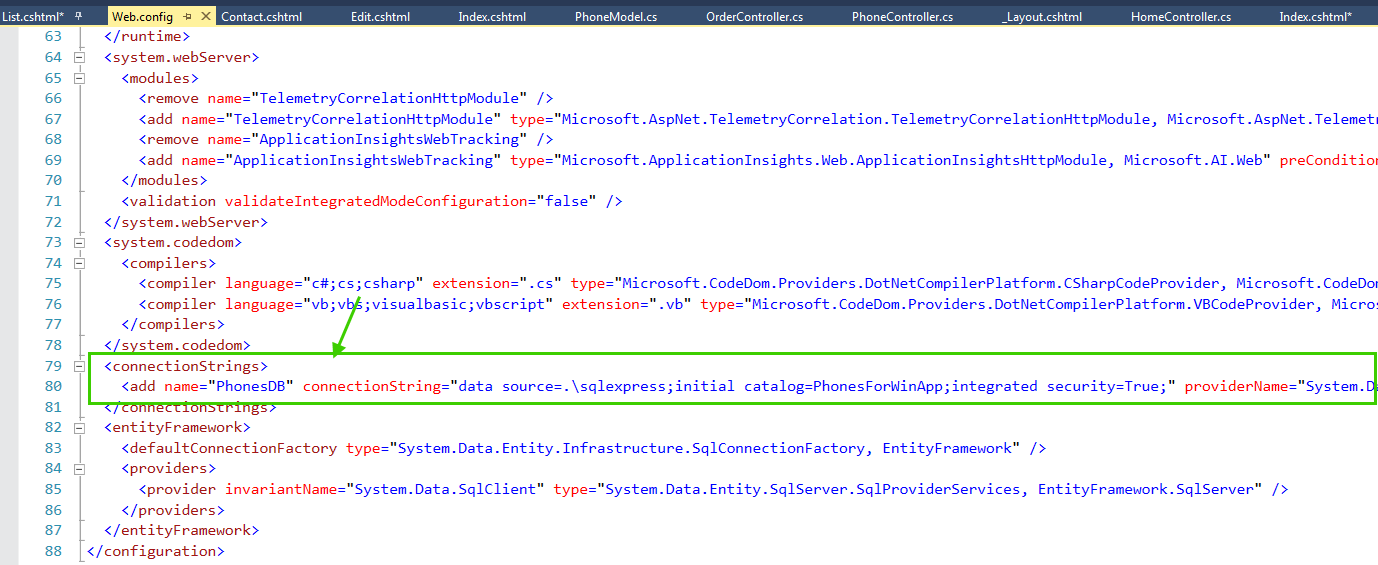
Итоговая структура решения: на уровне представления два проекта winForms (л.р.4) и aspNetMVC (созданное для л.р.5).



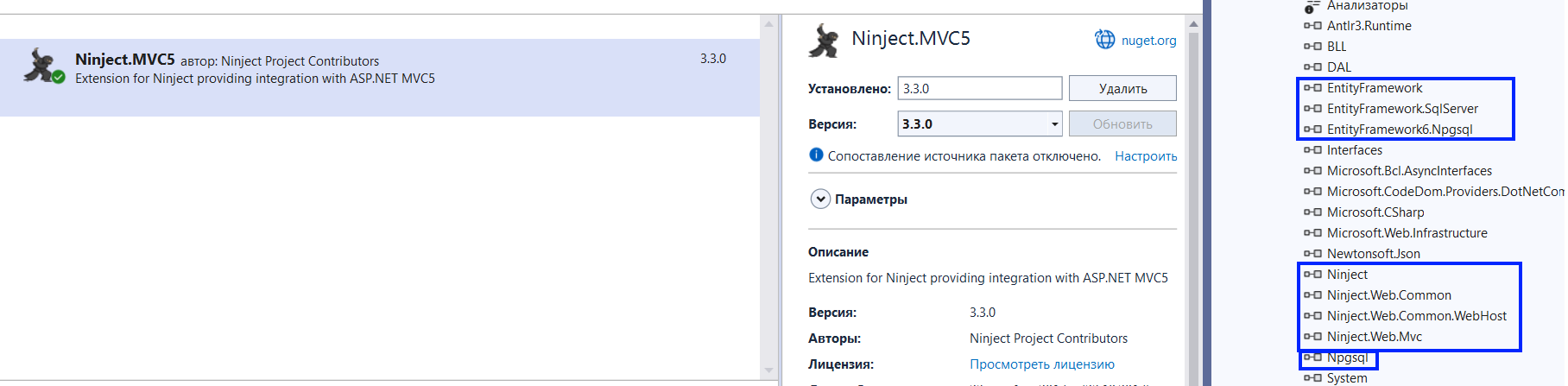
1. Добавить ссылки на BLL, DAL, Interfaces



1. Задать строку подключения в файле web.config (скопировать из л.р. №4 из app.config)



1. Добавить библиотеки EntityFramework (EntityFramework6.Npgsql), Ninject.MVC5.

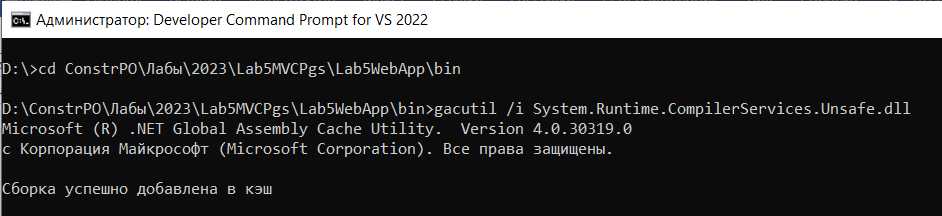


Если вы используете Postgres: Для правильной работы npgsql с mvc5 необходимо зарегистрировать дополнительно библиотеку System.Runtime.CompilerServices.Unsafe версии 4.0.4.1 в GAC

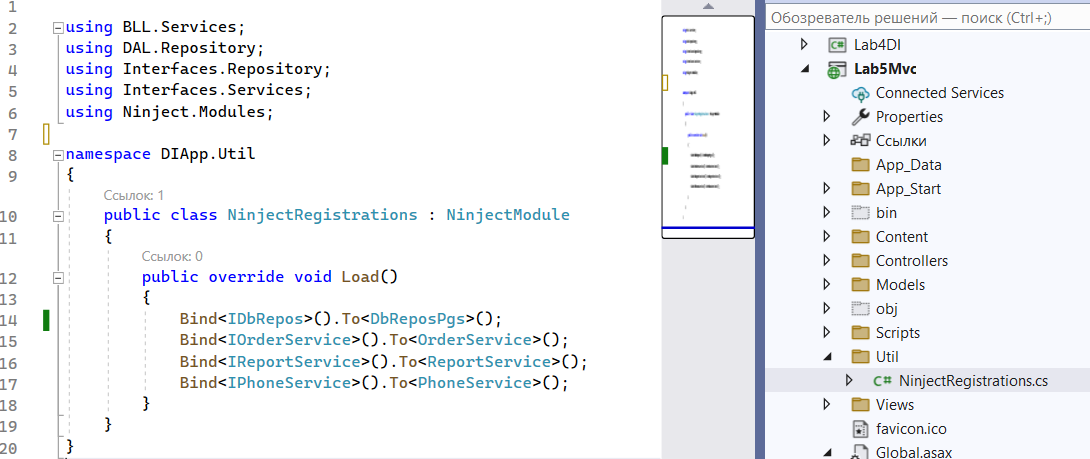
* **Запустите командную строку разработчика для VS2019** от имени **Администратора**
* введите:

cd xxxxx (путь к System.Runtime.CompilerServices.Unsafe 4.0.4.1, это в bin/debug вашего проекта)

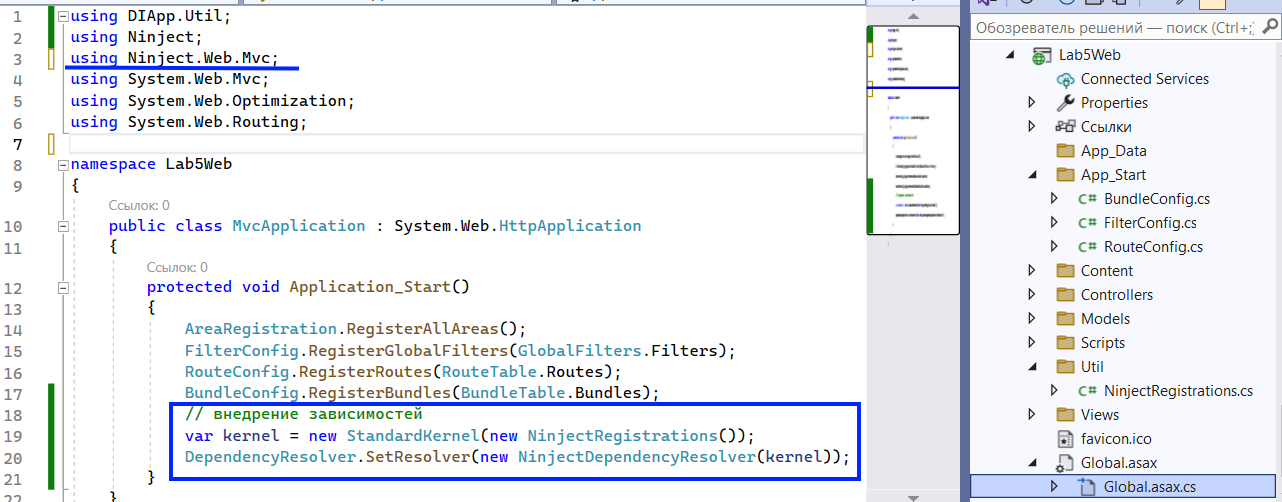
gacutil /i System.Runtime.CompilerServices.Unsafe.dll



1. Скопировать класс NinjectRegistrations из предыдущей л.р. в проект asp.net mvc.

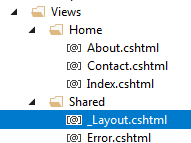


1. внедрить зависимости в файле global.asax (аналог program.cs WinForms)



*Часть 2. Создание шаблона страниц web-приложения..*

1. **Разработка собственного шаблона страниц вэб-сайта \_Layout.cshtml:**



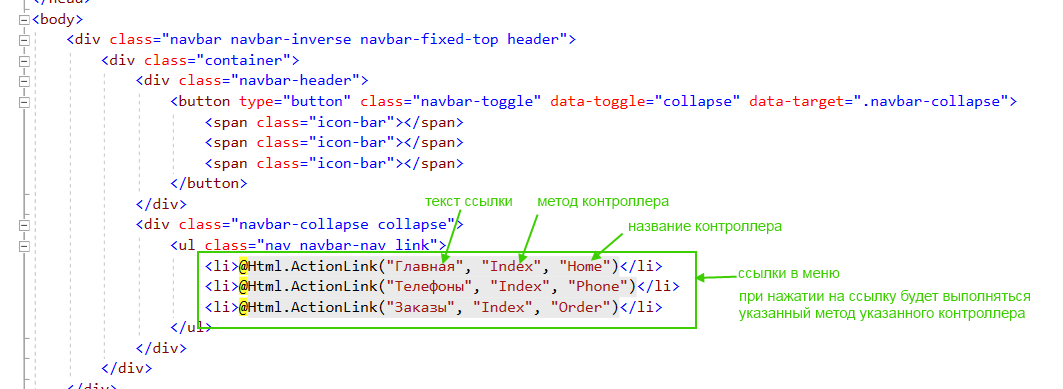
1. Удалить контент, созданный по умолчанию:



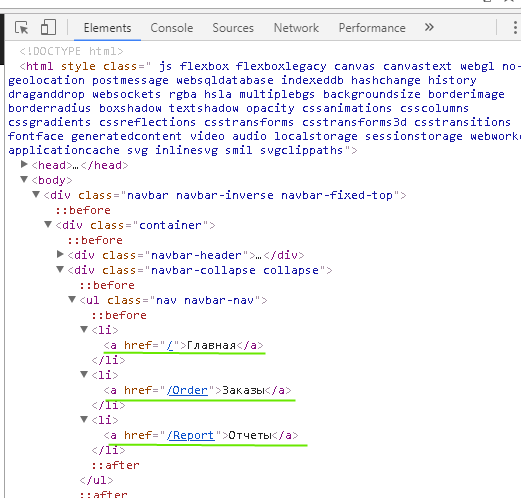
1. Создать свое меню: ссылки на

* главную страницу
* бизнес-логику
* список объектов
* Отчет1
* Отчет2

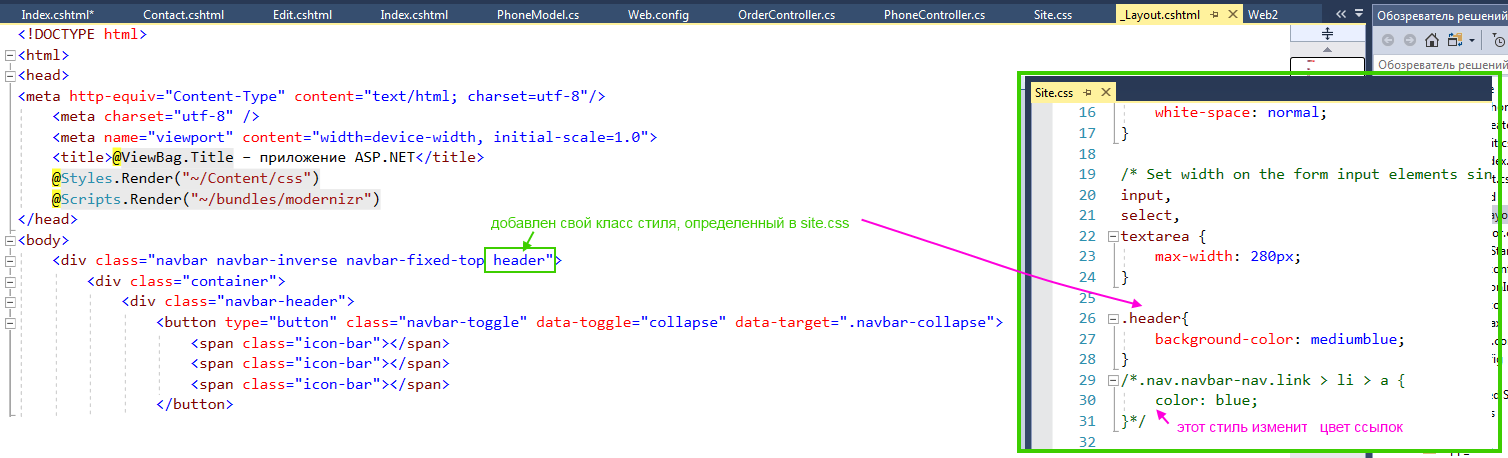
Для этого воспользуемся вспомогательным методом Html ActionLink (html-хэлпером). Он генерирует ссылку - элемент разметки html <a href>. В аргументах надо указать текст ссылки, имя контроллера, который будет обрабатывать запрос, и название метода действия в контроллере. Т.е. по клику на ссылку будет выполняться указанный метод указанного контроллера.



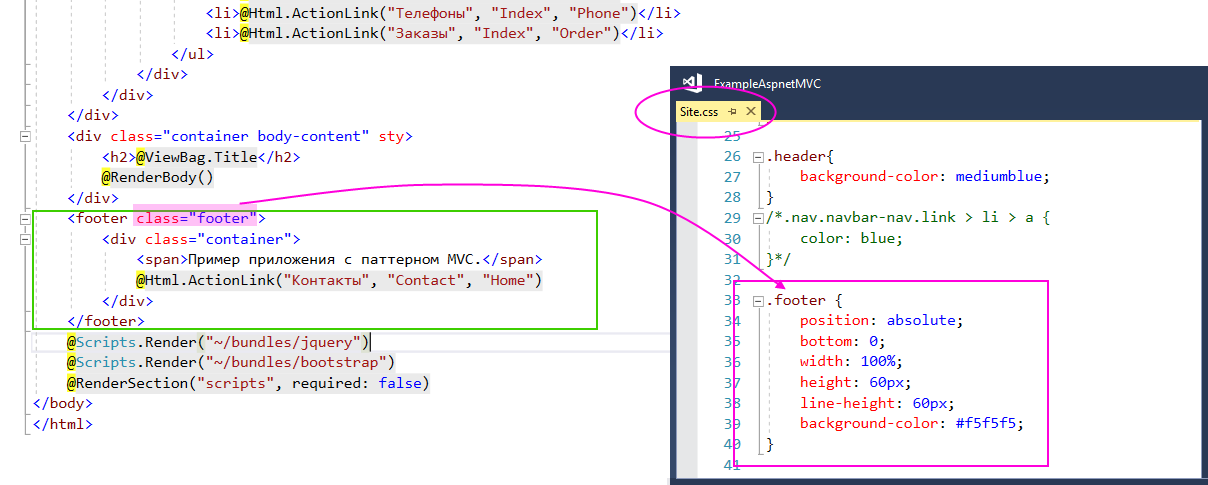
Для выше приведенного примера вместо кода C# будут создаваться три ссылки (можно посмотреть результат в отладчике браузера):



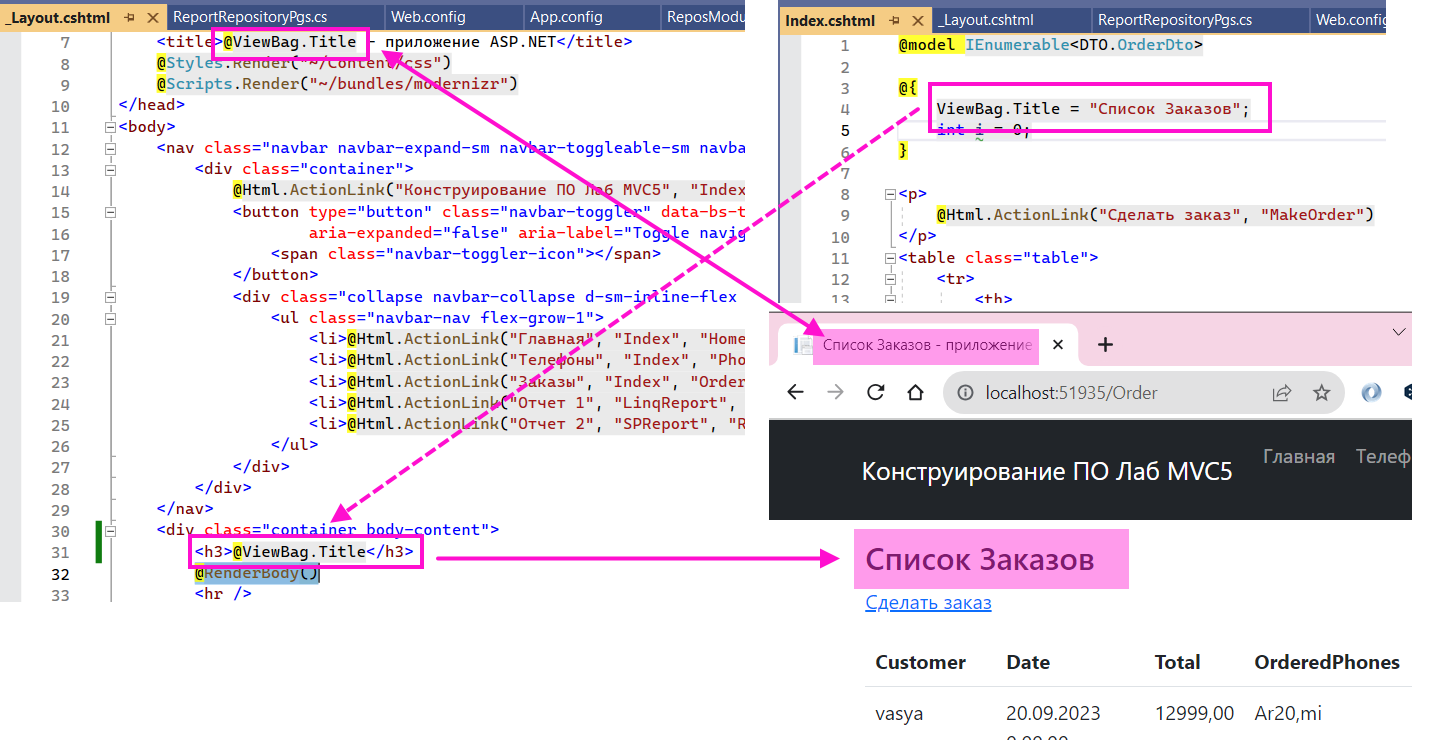
1. Изменить внешний вид макета страницы, добавив стиль (в файле Content/site.css или прямо в разметке с помощью атрибута style). Стили по умолчанию определены в файле Contents/site.css. Используем этот файл для добавления стилей . Например, заменим фон шапки на синий:



Изменение нижнего колонтитула (footer). Добавим стиль для элемента <footer>. В стиле “footer”заданы: притяжка к нижнему краю, высота 60px, серый фон. Придумайте и задайте СВОИ свойства. В приведенном ниже примере колонтитул содержит надпись «Пример приложения с паттерном MVC.» и ссылку на страницу «контакты». Придумайте СВОЁ содержимое нижнего колонтитула.



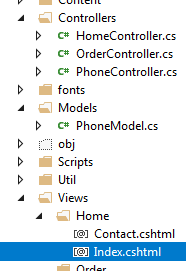
Перед блоком @RenderBody() в разметку добавлен <h2>@ViewBag.Title</h2>. Поэтому для каждой страницы будет отображаться текст переменной ViewBag.Title, так же, как и в названии вкладки. В примере лабораторной работы в каждом частичном представлении в @ViewBag задается название страницы, чтобы , переходя по вкладкам, был виден раздел, в котором находится пользователь. Например, для раздела «Список заказов»:

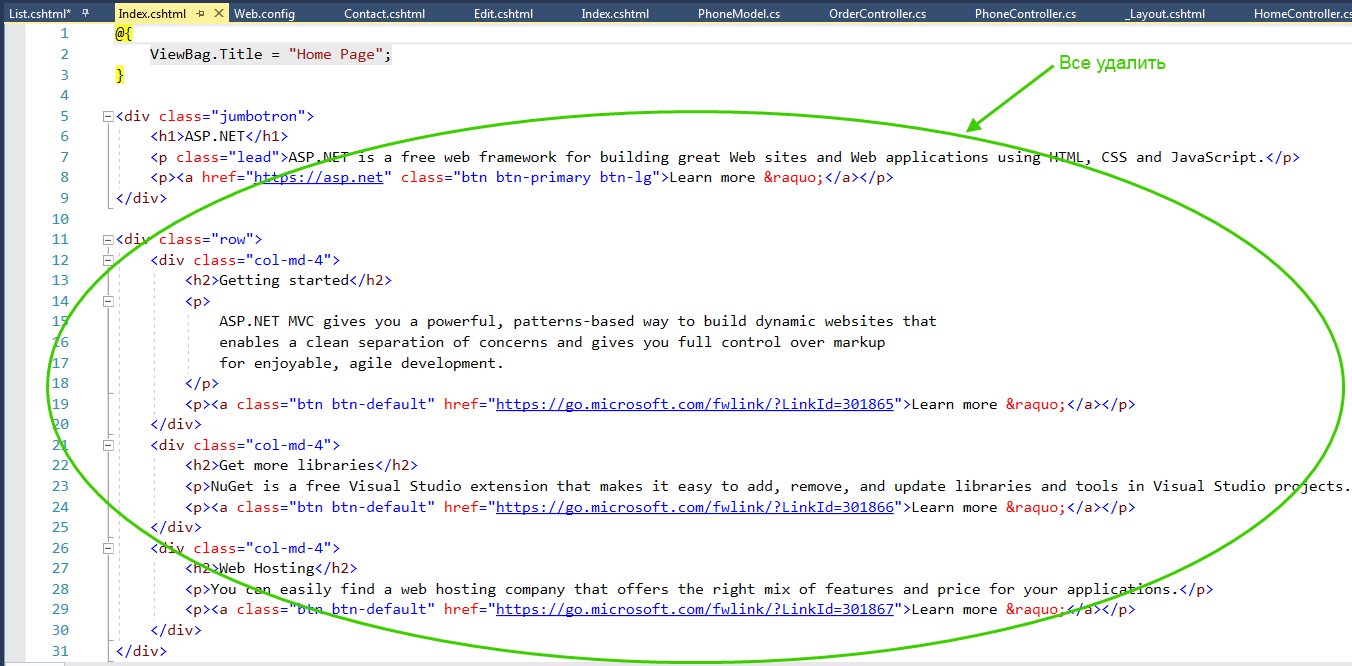


1. **Создать главную страницу приложения (представление).**

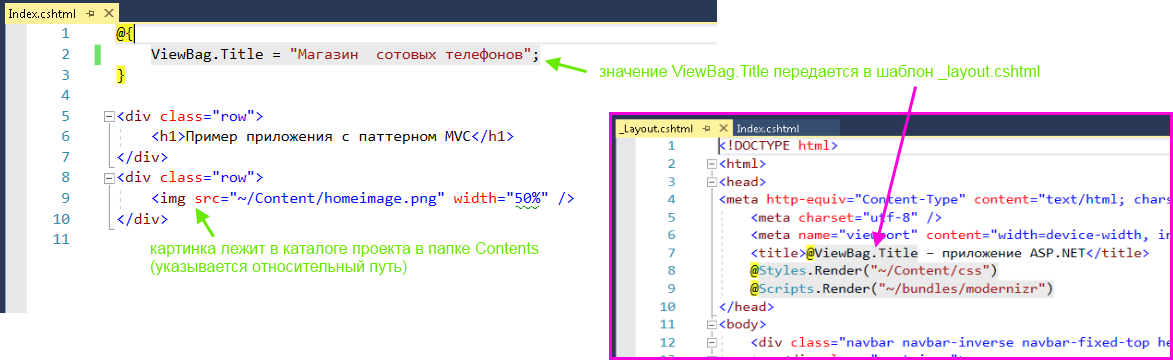
Она отображается при запуске. По умолчанию это представление Index контроллера HomeController.

Необходимо отредактировать представление Index.cshtml в папке Views/Home: удалить разметку html сгенерированную VisualStudio и написать СВОЮ.





Значение ViewBag.Title используется в шаблоне \_layout.cshtml для всего приложения. Оно используется для отображения названия страниц в браузере. Каждое представление может иметь свое название и передаёт его через переменную ViewBag.Title. Пример представления, которое содержит заголовок “ Пример приложения с паттерном MVC ” и картинку:

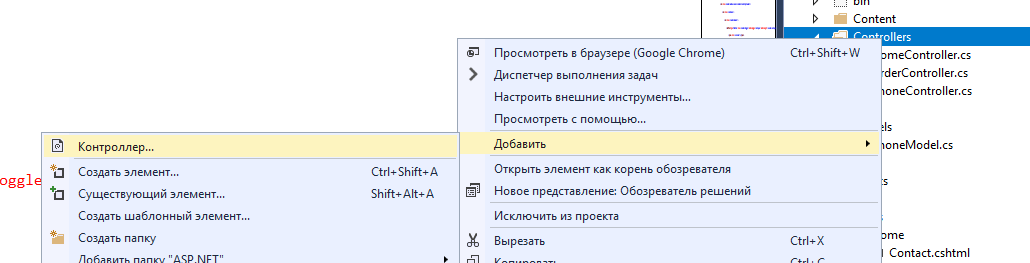


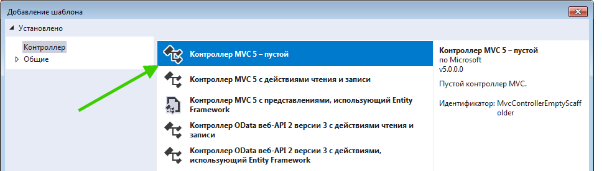
*Часть 3. Создание представлений Asp.net MVC (страниц).*

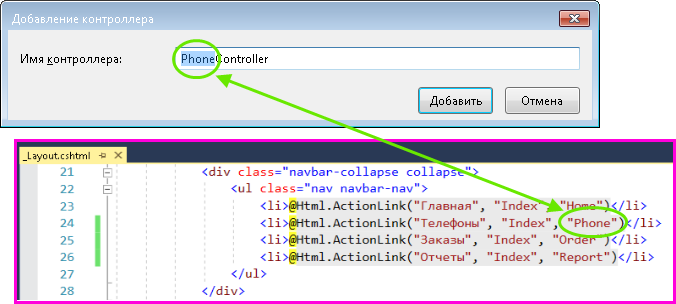
Необходимо создать несколько страниц, отражающих функциональность приложения WinForm из предыдущих л.р..

1. **Создание контроллера с методами действий для сущности предметной области**
2. Создать контроллер.

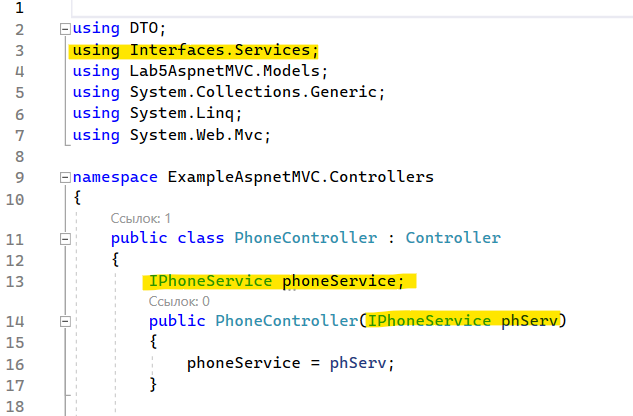
Добавить в папку Controllers новый контроллер с именем, соответствующим заданному в меню шаблона \_layout.schtml



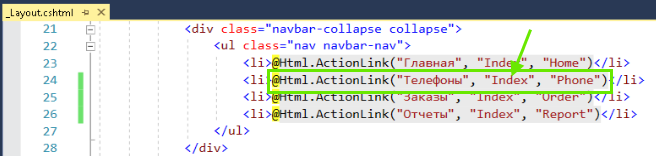


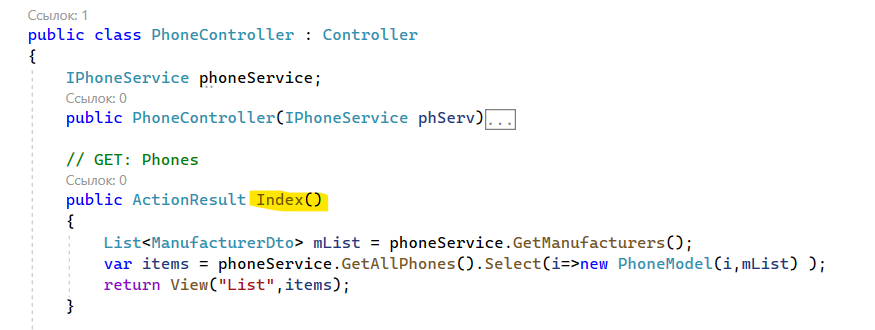


1. Создать конструктор контроллера, который принимает интерфейс класса бизнес логики. Через интерфейс сервиса будут выполняться запросы на получение данных.



1. Создать метод Index (название метода должно соответствовать указанному в меню в шаблоне). Этот метод будет выполнен контроллером при нажатии ссылки меню . Пример для страницы «Телефоны», на которой отображается список всех телефонов:





Метод Index получает список заказов и передает его в представление. Если не указано имя представления, то ищется представление по имени метода. Т.е. данный метод будет сопоставляться с файлом Views/Phones/Index.cshtml.

1. Создать методы для CRUD

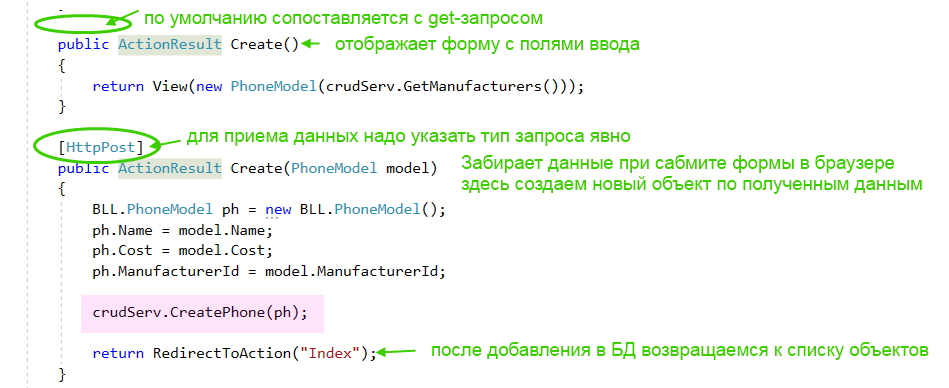
Запросы бывают разных типов: GET , POST. В общепринятой практике контроллер использует тип GET для отправки данных в браузер, а тип POST для приема входящих данных от браузера. Поэтому действия разбиваются на два метода: по одному для каждого типа запроса, когда вы хотите использовать метод с одинаковым названием и для запроса get, и для запроса post. При создании и редактировании объекта надо:

* Отобразить форму ввода данных. Используется get запрос.
* Обработать введенные данные. Используется post запрос. При нажатии кнопки с типом submit происходит отправка формы из браузера на сервер методом post.

Т.к. в данном примере лабораторной работы представление для объекта «Телефон» должно содержать данные по справочнику «Производители», используется не модель DTO.PhoneDto, а дополнительная модель Lab5AspnetMVC.Models.PhoneModel с дополнительным свойством «список производителей» List<ManufacturerDto> Manufs, которое и будет использоваться для хранения справочника производителей.

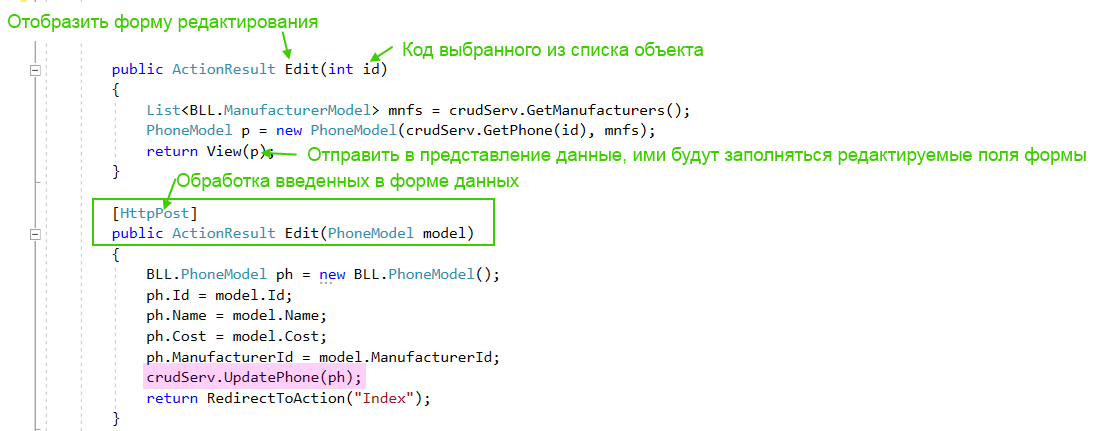
Создание объекта:

* отобразить форму ввода. Соответствующее представление создадим после.
* Принять данные, создать по ним объект и сохранить в БД. Вернуться к списку.



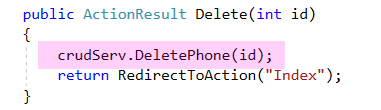
Редактирование:

* Найти данные объекта по выбранному идентификатору. отобразить форму ввода с заполненными полями ввода.
* Принять данные, создать по ним объект и обновить его в БД. Вернуться к списку.



Удаление:

* удалить из БД запись по выбранному Id и обновить список (используется функция «перейти к методу Index», т.к. он загружает весь список).

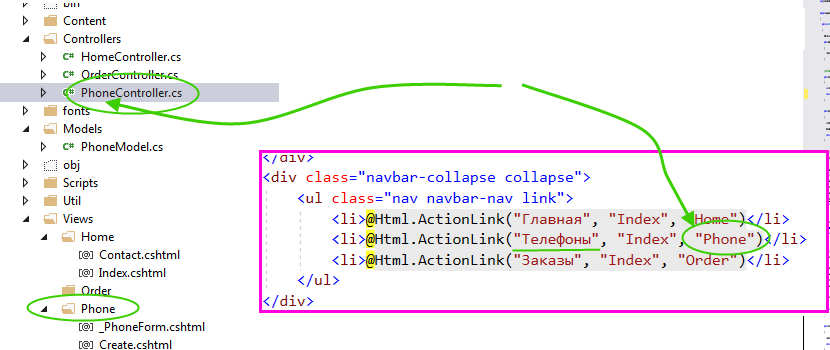


1. **Создание представлений для списка объектов и формы редактирования/создания**

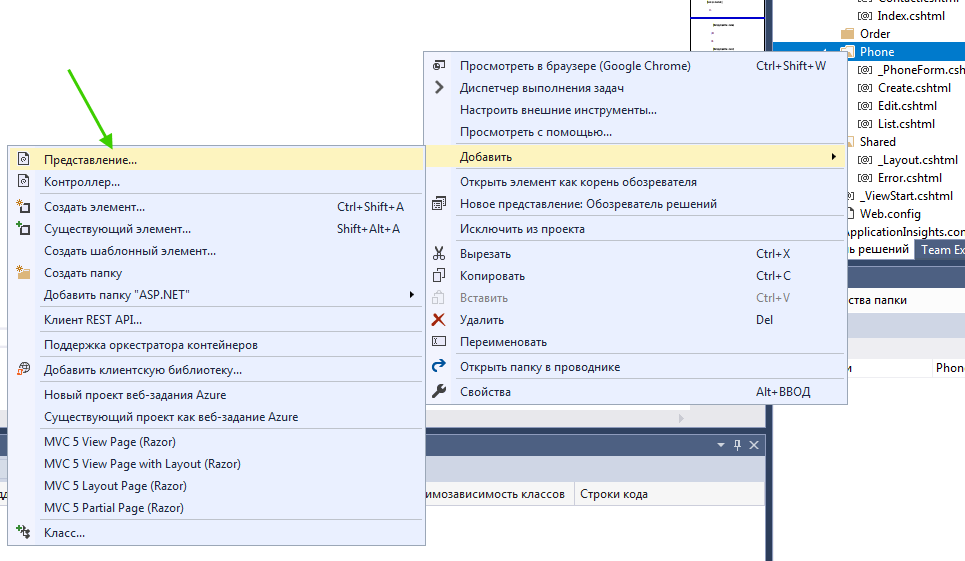
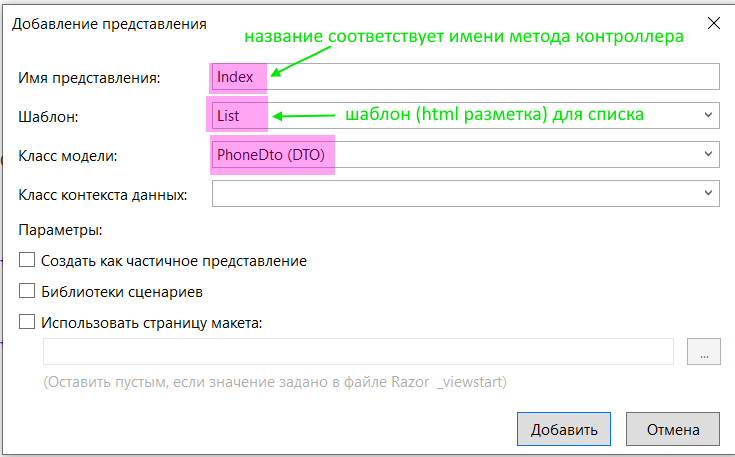
Представления - это файлы с расширением cshtml, которые содержат код пользовательского интерфейса в основном на языке html. Все добавляемые представления группируются по контроллерам в соответствующие папки в каталоге Views.

1. Создать представления для списка объектов

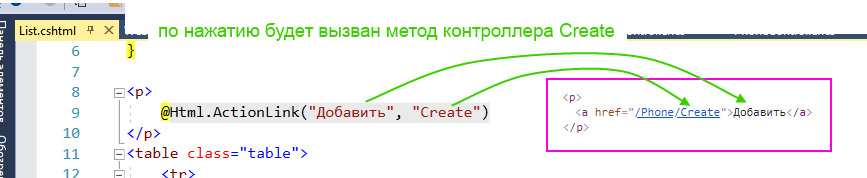
Создать в каталоге Views папку с именем, соответствующем имени контроллера.

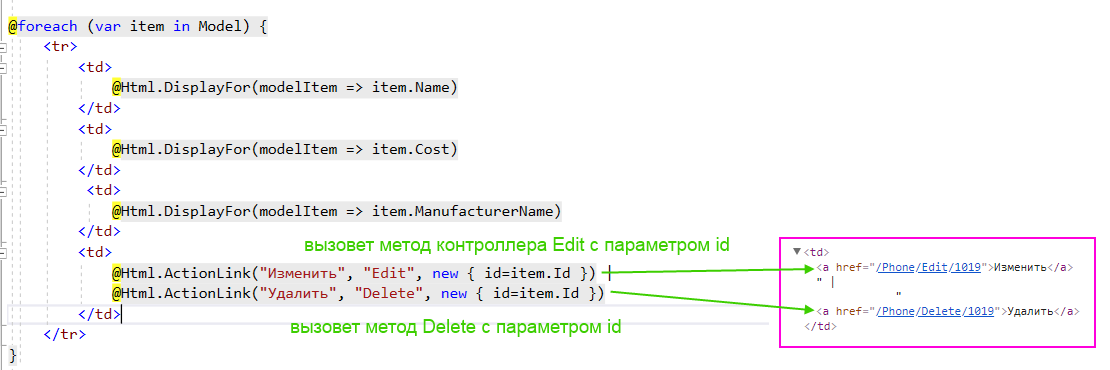


Добавить новый файл представления. Можно воспользоваться шаблоном list - для создания списка объектов и edit/create - для редактирования. Класс модели – это тип объекта, с которым будет работать представление (или классы dto или свои классы моделей приложения mvc).

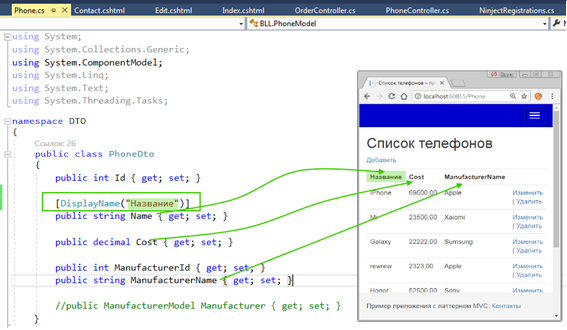
Представление списка объектов должно содержать ссылку/кнопку на форму добавления, а так же иметь возможность изменить и удалить элемент списка. Ссылки в представлениях должны указывать на соответствующие методы контроллера.

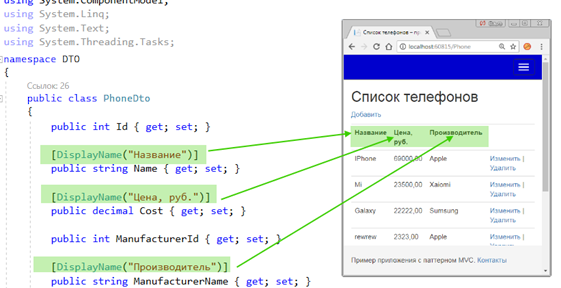




В качестве модели MVC можно использовать уже имеющуюся модель DTO уровня Interfaces . В демонстрационном примере создан дополнительный класс, т.к. модель редактирования содержит дополнительно справочник производителей.

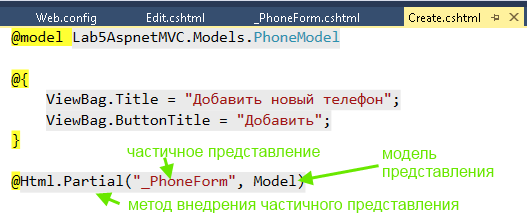
Для того, чтобы в списке и на форме редактирования отображались не имена свойств модели, а названия, понятные пользователю, можно воспользоваться механизмом аннотации данных для отображения свойств. Добавить атрибут DisplayName в модель сущности:

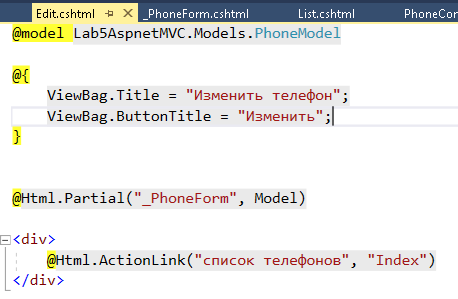


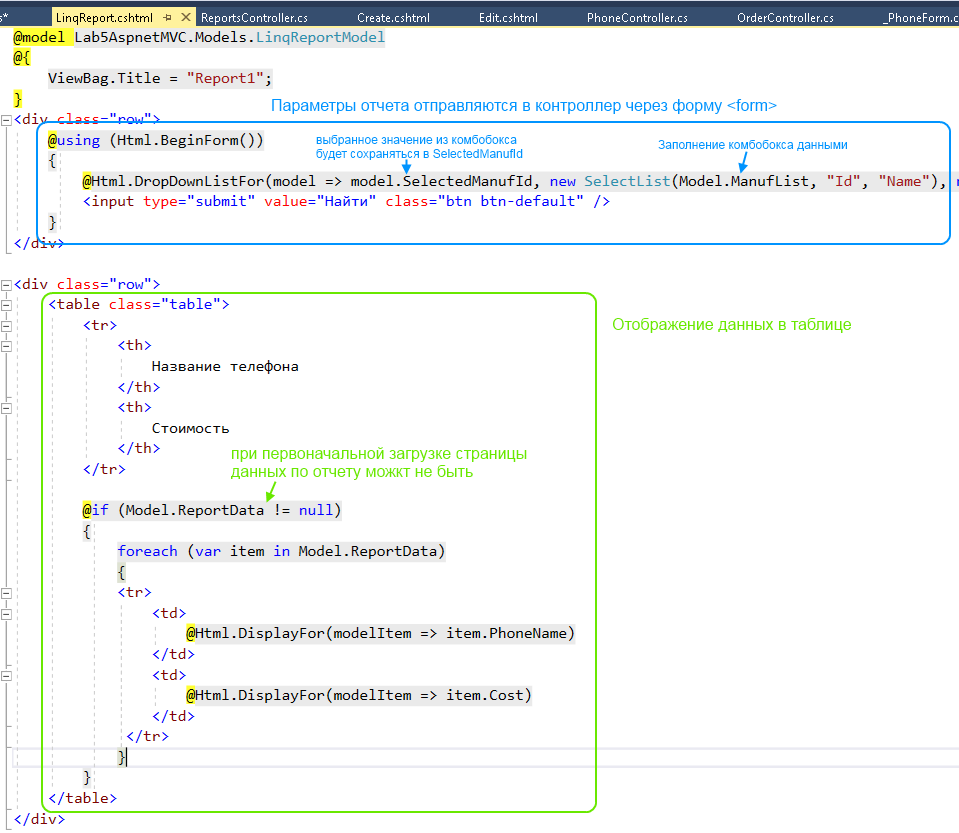


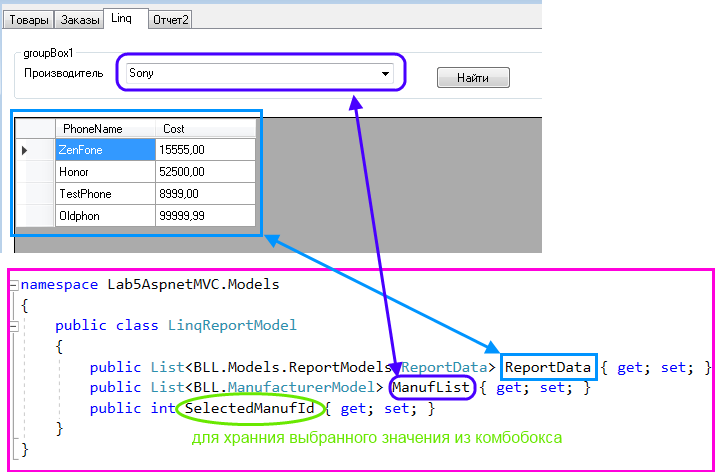
1. Создать представления для редактирования

Так как формы для создания и редактирования имеют абсолютно одинаковые поля ввода, то в таком случае рационально использовать частичное представление. В частичном представлении определяются общие элементы, используемые другими представлениями, чтобы избежать дублирования кода. Для создания нового объекта в представление передается пустая модель, для редактирования – модель с данными выбранного объекта. При этом, будем изменять название кнопки подтверждения редактирования: "Добавить" /"Изменить". Эта информацию можно передавать через объект ViewBag. Это объект типа dynamic, поэтому вы можете добавлять к нему новые свойства, не получая ошибок времени компиляции.





1. **Создание страницы для отчета**
2. Создать представления для отчета. 
3. Если для получения отчета требуются дополнительные данные (из БД, входные параметры для хранимой процедуры), необходимо создать модель представления, включающую дополнительные свойства.



1. Создать контроллер с методами действий, возвращающие формы ввода и результаты отчета.



Пример работы с датами:

