**http://professorweb.ru/my/ASP\_NET/mvc/level1/**

**Контроллер –** (controller) представляет класс, с которого собственно и начинается работа приложения. Этот класс обеспечивает связь между моделью и представлением. Получая вводимые пользователем данные, контроллер исходя из внутренней логики при необходимости обращается к модели и генерирует соответствующее представление.

**Представление –** (view) - это собственно визуальная часть или пользовательский интерфейс приложения - например, html-страница, через которую пользователь, зашедший на сайт, взаимодействует с веб-приложением.

**Модель –** (model) представляет набор классов, описывающих логику используемых данных.

**Создание и визуализация представления**

После первого редактирования метод действия Index() возвращал строковое значение. Это означало, что инфраструктура MVC всего лишь передавала браузеру строковое значение в неизменном виде. Теперь же, когда метод Index() возвращает объект **ViewResult**, инфраструктура **MVC Framework** визуализирует представление и возвращает сгенерированную HTML-разметку.

Мы не сообщали инфраструктуре MVC, какое представление должно применяться, поэтому она воспользовалась соглашением об именовании для его автоматического поиска. Соглашение предполагает, что имя представления совпадает с именем метода действия и содержится в папке, названной в соответствие с именем контроллера: ~/Views/Home/Index.cshtml.

**Помимо строк и объектов ViewResult** из методов действий можно возвращать и другие результаты. Например, если мы возвращаем **RedirectResult**, то это ведет к перенаправлению браузера на другой URL. Если мы возвращаем **HttpUnauthorizedResult**, то тем самым вынуждаем пользователя войти в систему. Эти объекты в совокупности называются результатами действий, и все они являются производными от класса **ActionResult**. Система результатов действий позволяет инкапсулировать и повторно использовать часто встречающиеся ответы в действиях.

### Добавление динамического вывода

Один из способов передачи данных из контроллера в представление предусматривает использование объекта **ViewBag**, который является членом базового класса **ControllerBase**.

Объект **ViewBag** - это динамический объект, в котором можно устанавливать произвольные свойства, делая эти значения доступными в любом визуализируемом далее представлении. В примере ниже демонстрируется передача простых динамических данных в файле HomeController.cs указанным способом.

Данные для представления предоставляются во время присваивания свойству ViewBag.Greeting значения. Свойство Greeting не существует вплоть до момента, когда ему присваивается значение - это позволяет передавать данные из контроллера в представление в свободной и гибкой манере, без необходимости в предварительном определении классов. Чтобы получить значение данных, необходимо еще раз сослаться на свойство ViewBag.Greeting, но уже в представлении, как показано в примере ниже, содержащем соответствующее изменение, которое было внесено в файл Index.cshtml:

Выбор для свойства имени Greeting не диктуется никакими особыми соображениями. Его можно было бы заменить любым другим именем, и все работало бы точно так же при условии, что имя, применяемое в контроллере, совпадает с именем, используемым в представлении. Присваивая значения более чем одному свойству, можно передавать из контроллера в представление множество значений данных. После запуска проекта можно увидеть результат внесенных изменений: