

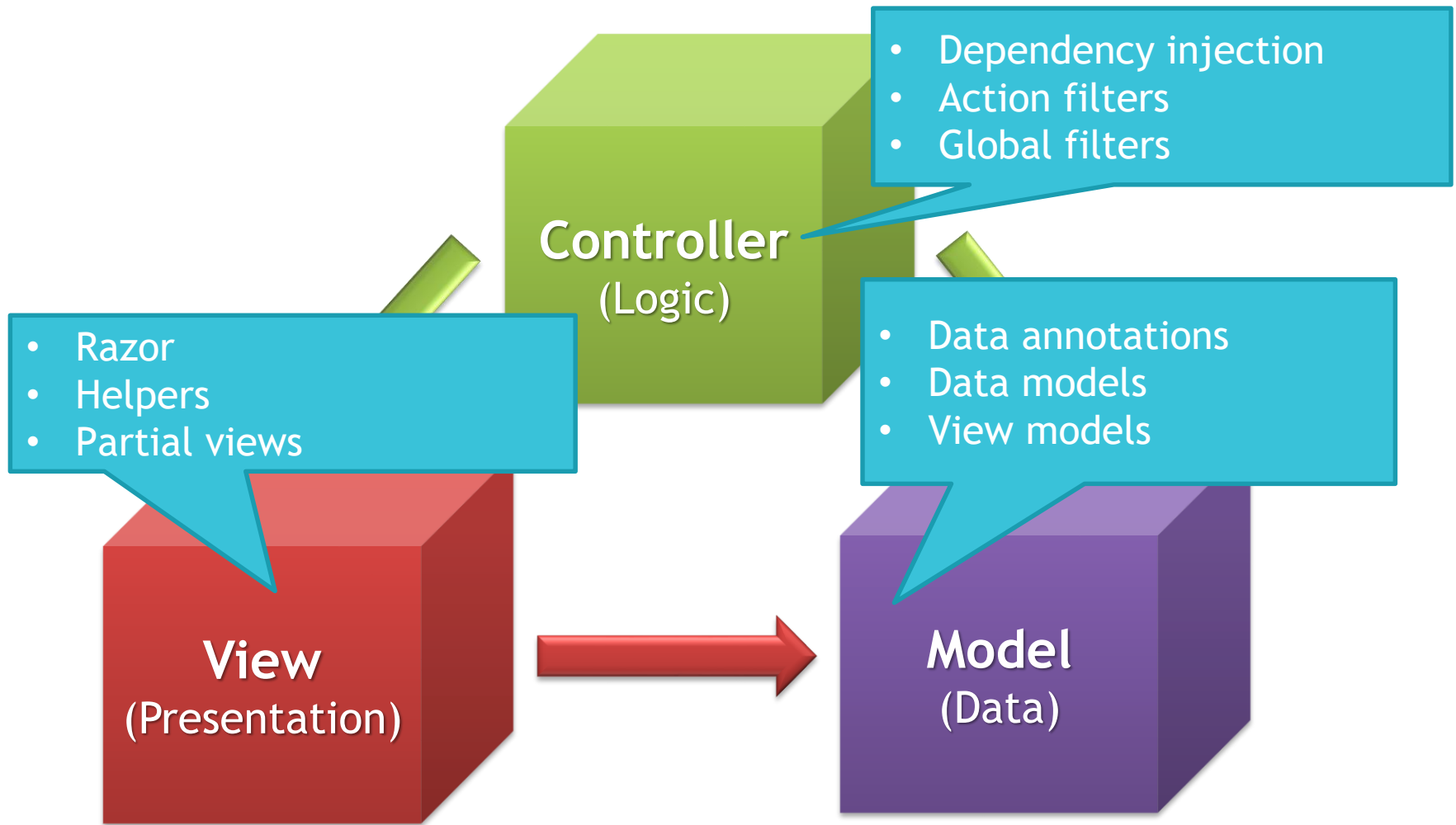


# ASP.NET MVC INTRODUCTION

**MODEL**

October, 2018

# ASP.NET MVC CORE COMPONENTS







**MODELS**

# MODELS

---

Моделями часто называют структуры данных, которые описывают основные элементы предметной области. В рамках такого подхода разделяют:

## Data Model:

- Работа с данными;
- сохранение и восстановление (DB).

## View Model:

- Передача данных во **view**;
- агрегирование моделей данных;
- целевое преобразование данных.

# MODELS

```
public class Person {
    public int Id { get; set; }
    public string FirstName { get; set; }
    public string LastName { get; set; }
    public int Age { get; set; }
}

public class People {
    public ICollection<Person> List { get; set; } = new List<Person> {
        new Person
        {
            FirstName = "Ivan", LastName = "Ivanov",
            Age = 23, Id = 0
        },
        new Person
        {
            FirstName = "Petr", LastName = "Petrov",
            Age = 32, Id = 1
        }
    };
}
```

# VIEWS WITH NON-TYPED MODEL

```
// ~/Controllers/HomeController.cs
```

```
public ActionResult PersonData()
{
    var people = new People();
    ViewBag.data = people.First();
    return View();
}
```

```
// или
```

```
public ActionResult PeopleData()
{
    var people = new People();
    return View(people.List);
}
```

# VIEWS WITH NON-TYPED MODEL

```
<!-- ~/Views/Home/PersonData.cshtml -->
```

```
<p>
```

```
Имя @ViewBag.data.FirstName <br />
```

```
Фамилия @ViewBag.data.LastName <br />
```

```
Возраст @ViewBag.data.Age
```

```
</p>
```

```
<!-- ~/Views/Home/PeopleData.cshtml -->
```

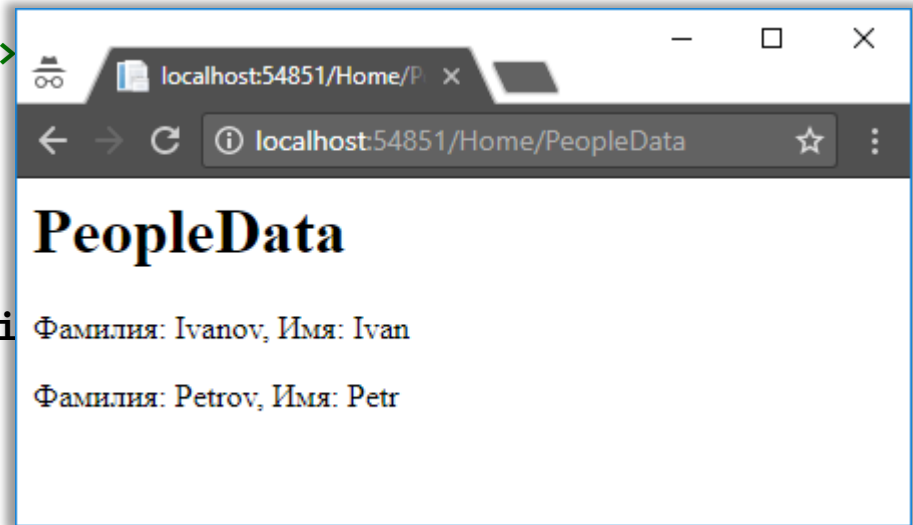
```
<h1>PeopleData</h1>
```

```
@foreach (var item in Model)
```

```
{
```

```
<p>Фамилия: @item.LastName, Имя: @item.FirstName
```

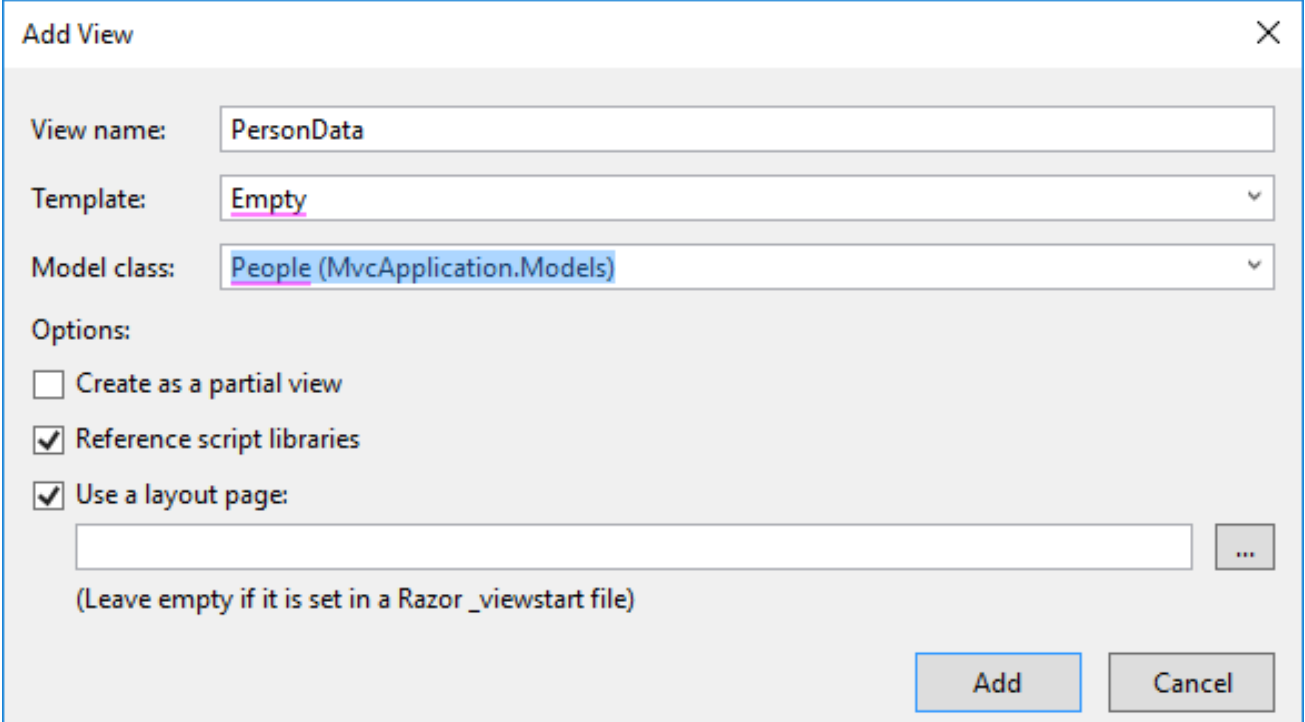
```
}
```





# VIEWS WITH TYPED MODEL

```
// ~/Controllers/HomeController.cs
public ActionResult PersonData() {
    var people = new People();
    return View(people);
}
```



Add View

View name:

Template:

Model class:

Options:

- ☐ Create as a partial view
- ☒ Reference script libraries
- ☒ Use a layout page:  
  
(Leave empty if it is set in a Razor \_viewstart file)



# VIEWS WITH TYPED MODEL

The image shows two overlapping screenshots of the Visual Studio IDE, specifically the code editor for `PersonData.cshtml`. The top screenshot shows the following code:

```
1 @model global::MvcApplication.Models.People
2
3 @{
4     ViewBag.Title = "Person Data";
5     Layout = "~/Shared/_Layout.cshtml";
6
7     var person = Model.List.First();
8 }
9 <p>
10     Имя: @person.First
11     Фамилия: @person.L
12     Возраст: @person.A
13 </p>
```

A red circle highlights the line `@model global::MvcApplication.Models.People`. A dropdown menu is open for the `@person.First` property, showing the `List` class and its properties:

Property	Type	Value
<code>Equals</code>	<code>bool</code>	
<code>GetHashCode</code>	<code>int</code>	
<code>GetType</code>	<code>Type</code>	
<code>ToString</code>	<code>string</code>	

The bottom screenshot shows the same code, but with the dropdown menu expanded to show the `Person` class and its properties:

Property	Type	Value
<code>FirstName</code>	<code>string</code>	
<code>Age</code>	<code>int</code>	
<code>Id</code>	<code>int</code>	
<code>LastName</code>	<code>string</code>	

# SCAFFOLDING

```
// ~/Controllers/HomeController.cs
public ActionResult PeopleData() {
    var people = new People();
    return View(people.List);
}
```

Add View

View name:

Template:

Model class:

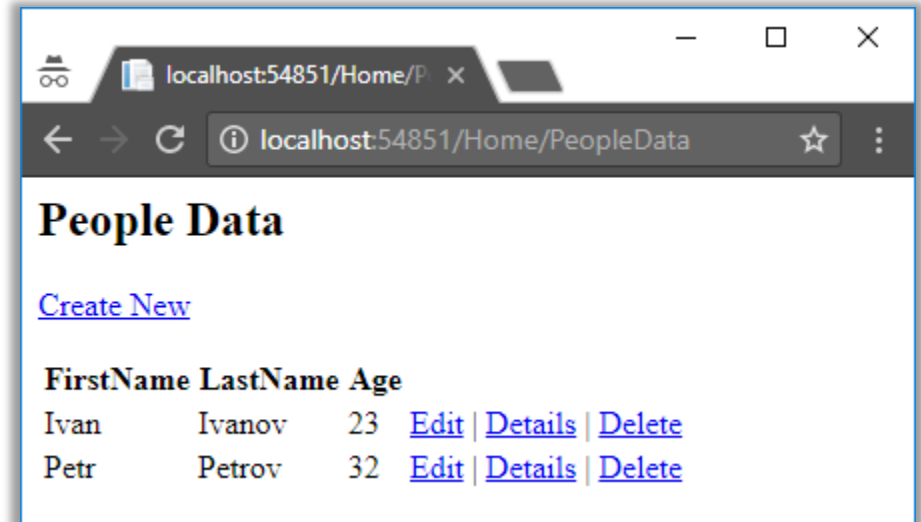
Options:

- ☐ Create as a partial view
- ☒ Reference scaffolding type
- ☒ Use a layout file

(Leave empty if it is set in a Razor \_viewstart file)

# SCAFFOLDING

```
HomeController.cs | PeopleData.cshtml | PersonData.cshtml
1  @model IEnumerable<global::MvcApplication.Models.Person>
2
3  @{
4      Layout = null;
5      ViewBag.Title = "People Data";
6  }
7
8  <h2>People Data</h2>
9
10 <p>
11     @Html.ActionLink("Create New", "Create")
12 </p>
13 <table class="table">
14     <tr>
15         <th>
16             @Html.DisplayNameFor(model => model.FirstName)
17         </th>
18         <th>
19             @Html.DisplayNameFor(model => model.LastName)
20         </th>
21         <th>
22             @Html.DisplayNameFor(model => model.Age)
23         </th>
24     </tr>
25
26     @foreach (var item in Model) {
27         <tr>
28             <td>
29                 @Html.DisplayFor(modelItem => item.FirstName)
30             </td>
31             <td>
32                 @Html.DisplayFor(modelItem => item.LastName)
33             </td>
34             <td>
35                 @Html.DisplayFor(modelItem => item.Age)
36             </td>
37             <td>
38                 @Html.ActionLink("Edit", "Edit", new { id = item.Id }) |
39                 @Html.ActionLink("Details", "Details", new { id = item.Id }) |
40                 @Html.ActionLink("Delete", "Delete", new { id = item.Id })
41             </td>
42         </tr>
43     }
44 </table>
45
46
```

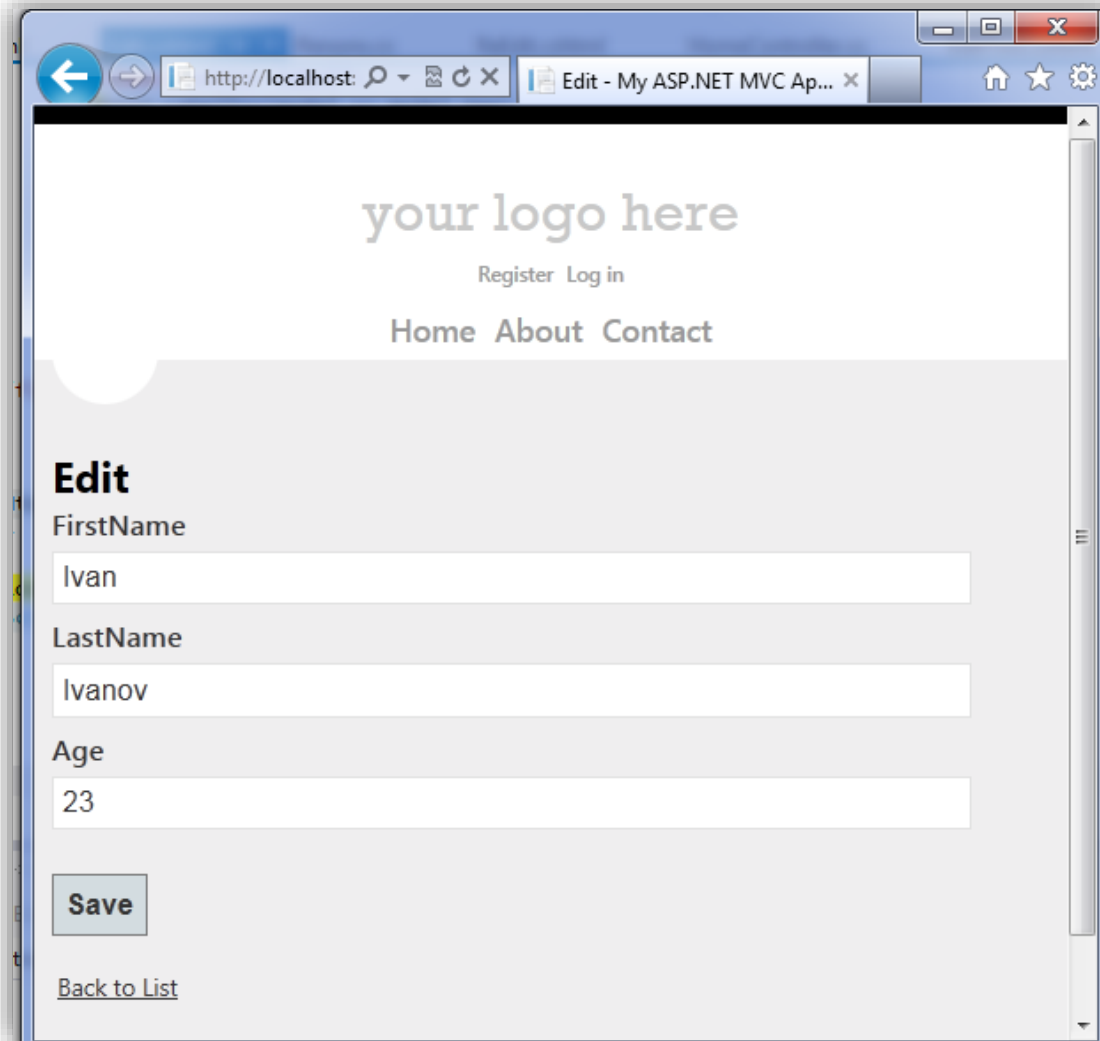


# STRONGLY TYPED HELPERS

```
@model MvcApp.Models.Person
```

```
@using(Html.BeginForm()) {  
    <div>@Html.LabelFor(m => m.FirstName)</div>  
    <div>@Html.EditorFor(m => m.FirstName)  
        @Html.ValidationMessageFor(m => m.FirstName)  
    </div>  
    <div>@Html.LabelFor(m => m.LastName)</div>  
    <div>@Html.EditorFor(m => m.LastName)  
        @Html.ValidationMessageFor(m => m.LastName)</div>  
  
    <div>@Html.LabelFor(m => m.Age)</div>  
    <div>@Html.EditorFor(m => m.Age)  
        @Html.ValidationMessageFor(m => m.Age)</div>  
  
    <p> <input type="submit" value="Save" /> </p>  
}  
<div>  
    @Html.ActionLink("Back to List", "Index")  
</div>
```

# STRONGLY TYPED HELPERS



The screenshot shows a web browser window with the address bar displaying 'http://localhost:...' and the tab title 'Edit - My ASP.NET MVC Ap...'. The page content includes a header with the text 'your logo here', links for 'Register' and 'Log in', and a navigation menu with 'Home', 'About', and 'Contact'. The main content area is titled 'Edit' and contains a form with three text input fields: 'FirstName' (containing 'Ivan'), 'LastName' (containing 'Ivanov'), and 'Age' (containing '23'). Below the form is a 'Save' button and a link labeled 'Back to List'.

your logo here

[Register](#) [Log in](#)

[Home](#) [About](#) [Contact](#)

## Edit

FirstName

LastName

Age

[Back to List](#)



# TEMPLATE HELPERS

**@Html.Display()** — Создает элемент разметки, который доступен только для чтения, для указанного свойства модели:

```
@Html.Display("Name")
```

**@Html.DisplayFor()** Строго типизированный аналог хелпера **@Html.Display()**:

```
@Html.DisplayFor(m => m.Name)
```

**@Html.Editor()** Создает элемент разметки, который доступен для редактирования, для указанного свойства модели:

```
@Html.Editor("Name")
```

**@Html.EditorFor()** Строго типизированный аналог хелпера **@Html.Editor()** (в зависимости от передаваемого типа данных формирует соответствующий элемент **<input>**, **<select>**, **<textarea>**):

```
@Html.EditorFor(m => m.Name)
```

**@Html.DisplayText()** Создает выражение для указанного свойства модели в виде простой строки:

```
@Html.DisplayText("Name")
```

**@Html.DisplayTextFor()** Строго типизированный аналог хелпера **@Html.DisplayText()**:

```
@Html.DisplayTextFor(m => m.Name)
```

# TEMPLATE HELPERS

**@Html.DisplayForModel()** — Создает поля для чтения для всех свойств модели:

**@Html.Html.DisplayForModel()**

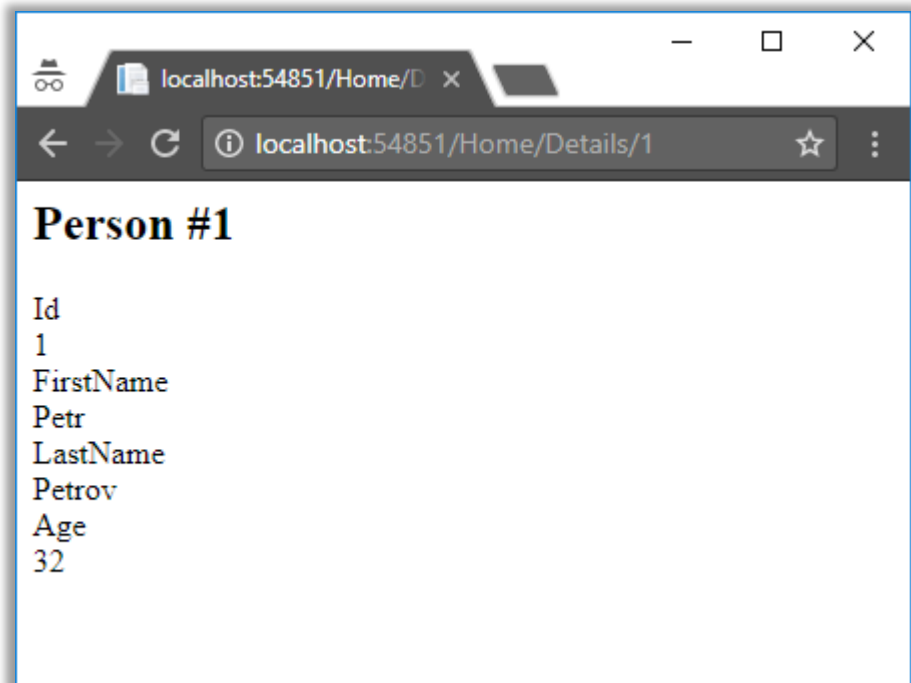
**@Html.EditorForModel()** — Создает поля для редактирования для всех свойств модели:

**@Html.Html.EditorForModel()**

```
@model MvcApp.Models.Person
```

```
<h2>Person #@Model.Id</h2>
```

```
@Html.DisplayForModel()
```



# TEMPLATE HELPERS

```
// ~/Controllers/HomeController.cs
```

```
public ActionResult Details(int id)
{
    var people = new People();

    var person = people.List.FirstOrDefault(p => p.Id == id);
    if (person == null)
    {
        return HttpNotFound();
    }

    return View(person);
}
```

# TEMPLATE HELPERS

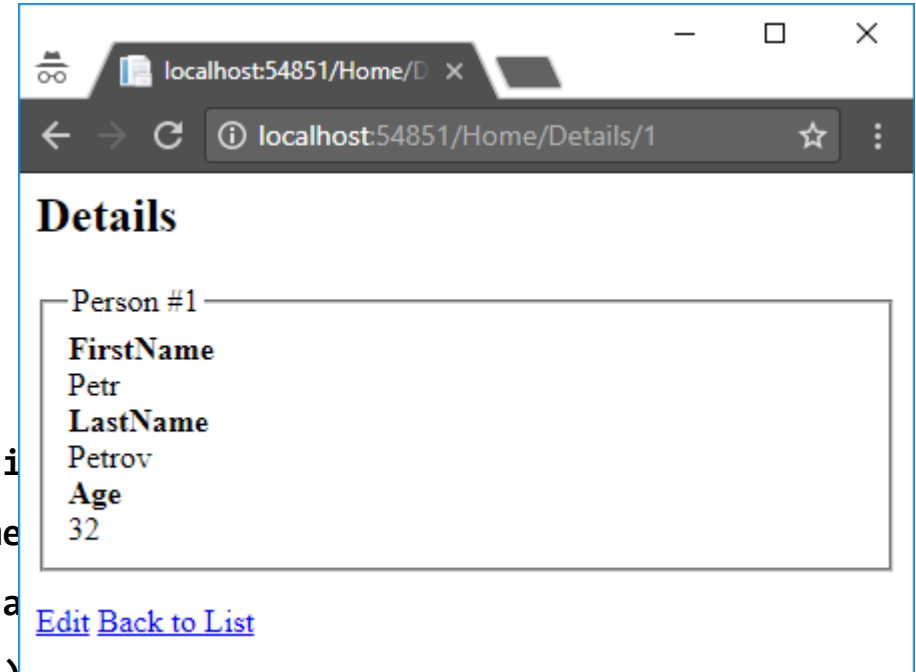
```
<!-- ~/Views/Home/Details.cshtml -->
@model MvcApp.Models.Person
@{
    ViewBag.Title = "Person Data";
    Layout = null;
}

<h2>Details</h2>

<fieldset>
    <legend>Person #@Model.Id</legend>

    <div><b>@Html.DisplayNameFor(m => m.FirstName)</b></div>
    <div>@Html.DisplayFor(m => m.FirstName)</div>
    <div><b>@Html.DisplayNameFor(m => m.LastName)</b></div>
    <div>@Html.DisplayFor(m => m.LastName)</div>
    <div><b>@Html.DisplayNameFor(m => m.Age)</b></div>
    <div>@Html.DisplayFor(m => m.Age)</div>
</fieldset>

<p>
    @Html.ActionLink("Edit", "Edit", new { id = Model.Id })
    @Html.ActionLink("Back to List", "Index")
</p>
```



# LIST HELPERS USAGE

```
// ~/Conrtollers/HomeController.cs

private readonly People _people = new People();
...
private IEnumerable<SelectListItem> GetPeople() {
    return _people.Select(
        p => new SelectListItem
        {
            Text = p.FirstName,
            Value = p.Id.ToString()
        });
}

[HttpGet]
public ActionResult DropDown() {
    ViewBag.List = GetPeople();
    return View();
}
```



# LIST HELPER USAGE

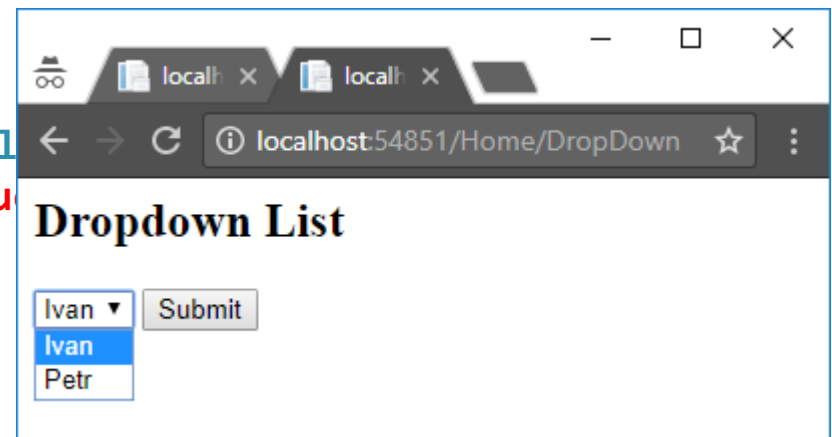
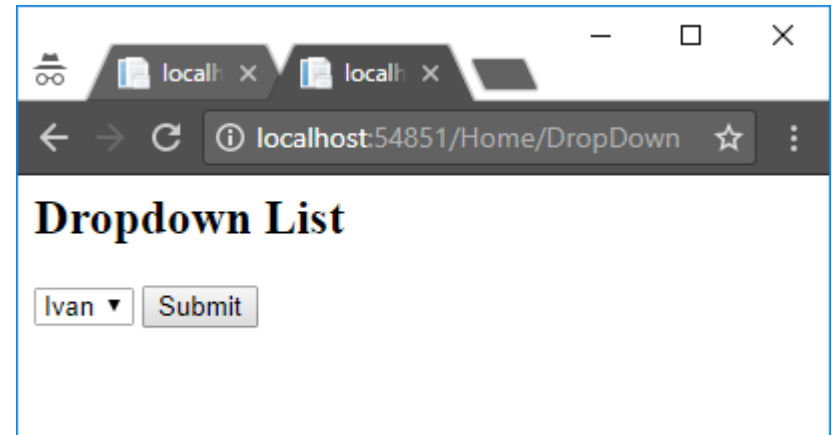
```
<!-- ~/Views/Home/DropDown.cshtml -->  
@model MvcApp.Models.Person  
@{  
    ViewBag.Title = "Person DropDown";  
    Layout = null;  
}
```

```
<h2>Dropdown List</h2>
```

```
@using (Html.BeginForm())  
{
```

```
    @Html.DropDownListFor(m => m.Id,  
        ViewBag.List as IEnumerable<Sel  
        <input type="submit" name="btn" valu
```

```
}
```



# DATA TRANSFER TO CONTROLLER

```
// ~/Conrtrollers/HomeController.cs
```

```
[HttpPost]
```

```
public ActionResult DropDown(int id) {  
    ViewBag.List = GetPeople();
```

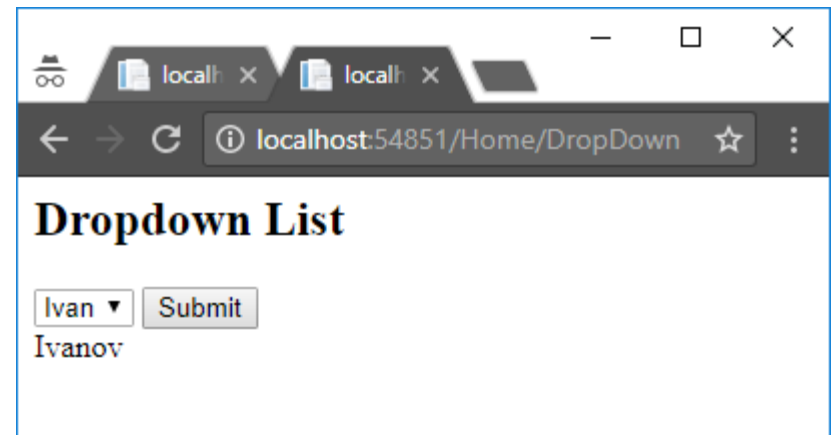
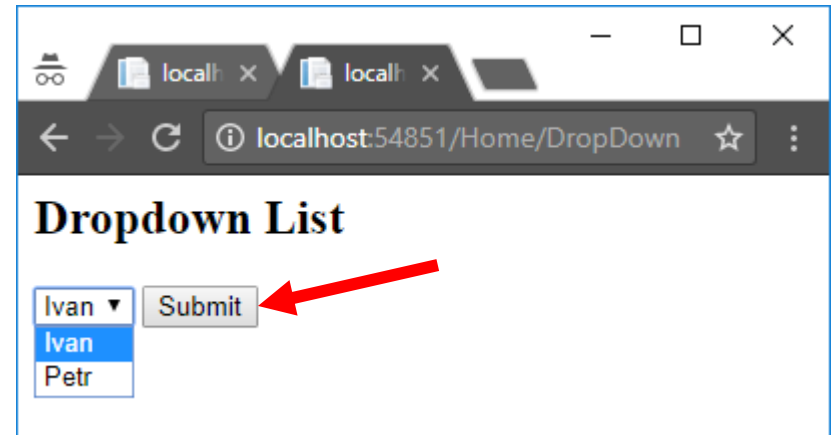
```
    ViewBag.PostBack = _people.List  
        .FirstOrDefault(p => p.Id == id)?.LastName ?? string.Empty;
```

```
    return View();
```

```
}
```

```
<!-- ~/Views/Home/DropDown.cshtml -->
```

```
@ViewBag.PostBack
```



**QUESTIONS?**

A stylized world map in a light blue color, centered on the Atlantic Ocean, serving as a background for the text.

**Thanks for attention!**