

Html.DropDownList And Html.DropDownListFor Example In ASP.NET MVC

In this chapter, you will learn:

1. What is **Html.DropDownList** and **Html.DropDownListFor** Methods?
2. What is the difference between **Html.DropDownList** and **Html.DropDownListFor** method?
3. **Html.DropDownList** Example
4. **Html.DropDownListFor** Example
5. How Bind **Html.DropDownListFor** with a Model
6. How to display List item in **DropDownList**
7. How to send **DropDownList** Value to Controller
8. **Html** Attributes of **Html.DropDownList**

DropDownList is a visual control with down arrow which displays a list of items and forces you to choose only one item at a time. **DropDownList** is not editable, it means the user cannot enter own value and limited to choose an item from the selection.

Html.Helper class provides two extension method for working with **DropDownList**.

- a. **Html.DropDownList**
- b. **Html.DropDownListFor**

@HTML.DROPDOWNLIST: DEFINITION AND EXAMPLE

Html.DropDownList is a loosely type that means it is not strongly bound to any list items and model properties.

Define **Html.DropDownList**

Html.DropDownList(string name, IEnumerable<SelectListitem> selectList, string optionLabel, object htmlAttributes)

- a. **String** name is the name of **DropDownList**
- b. **IEnumerable<SelectListitem>** selectList is the list of items
- c. string optionLabel is the optional informative string like please choose item, select item, select value etc.
- d. object **HtmlAttributes** adds extra features to dropdownlist.

Model: UserModel.cs

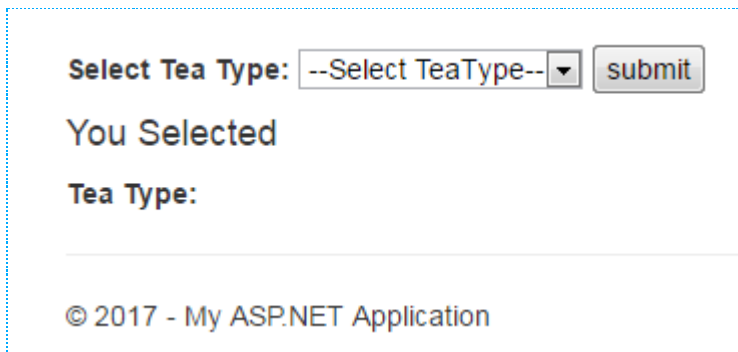
```
1 namespace HtmlHelperDemo.Models
2 {
3     public class UserModel
4     {
5         public TeaType SelectTeaType { get; set; }
6     }
7
8     public enum TeaType
9     {
10         Tea, Coffee, GreenTea, BlackTea
11     }
12 }
```

View: Index.cshtml

```
1 @using HtmlHelperDemo.Models
2 @model UserModel
3 <br /><br />
4 @using (Html.BeginForm("Index", "Home", FormMethod.Post))
5 {
6     <b>Select Tea Type: </b>
7
8     @Html.DropDownList("SelectTeaType", new SelectList(Enum.GetVa
9     <input type="submit" value="submit" />
10 }
11
12 <h4>You Selected</h4>
13 <b>Tea Type: </b>@ViewBag.TeaType
```

Controller: HomeController.cs

```
1  using System.Web.Mvc;
2  using HtmlHelperDemo.Models;
3
4  namespace HtmlHelperDemo.Controllers
5  {
6      public class HomeController : Controller
7      {
8          public ActionResult Index()
9          {
10             return View();
11         }
12
13         [HttpPost]
14         public ActionResult Index(UserModel model)
15         {
16             var selectedValue = model.SelectTeaType;
17             ViewBag.TeaType = selectedValue.ToString();
18             return View();
19         }
20     }
21 }
```



Select Tea Type: --Select TeaType-- submit

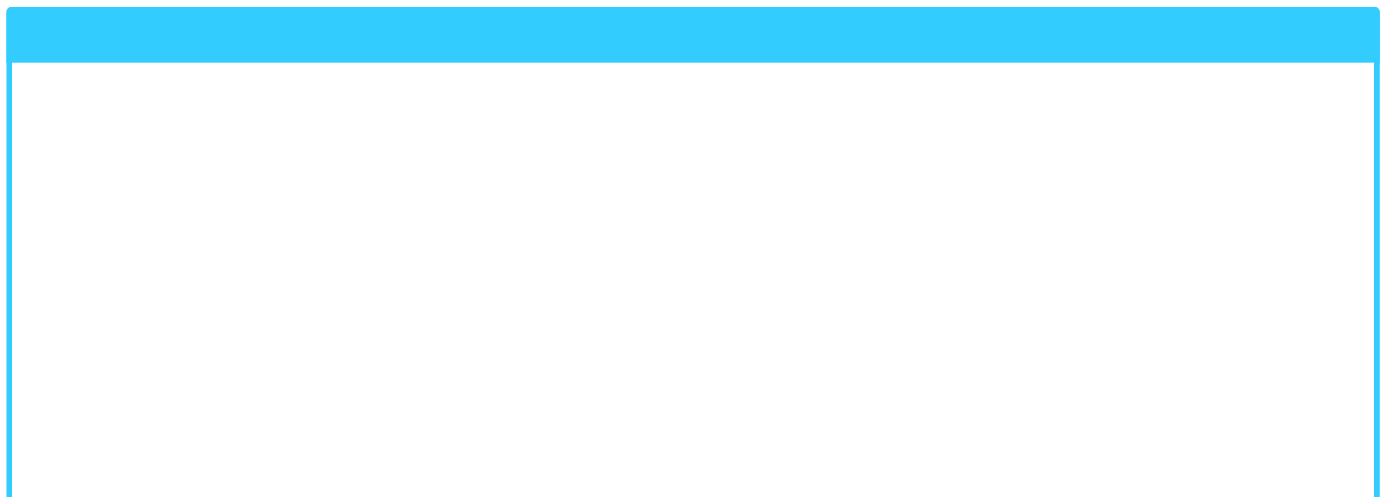
You Selected

Tea Type:

© 2017 - My ASP.NET Application

(https://www.completecsharp tutorial.com/wp-content/uploads/2018/02/1_dropdownlist.png)

Html Output:

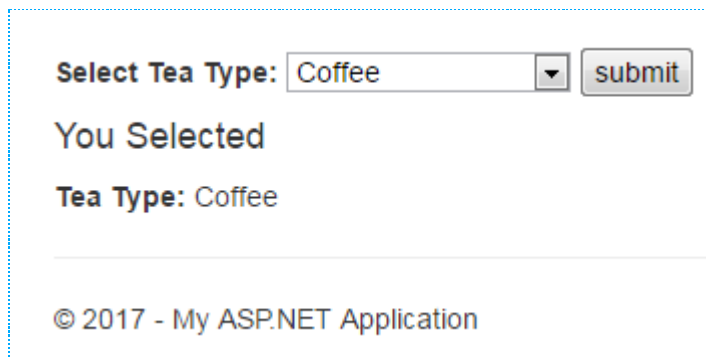


```
<select data-val="true" data-val-required="The SelectTeaType field is
  <option value="">Select TeaType</option>
  <option>Tea</option>
  <option selected="selected">Coffee</option>
  <option>GreenTea</option>
  <option>BlackTea</option>
</select>
```

You have noticed that 2 extra attributes are added in the html output: **data-val** and **data-val-required**.

Data-val = "true" and **data-val-required = "The SelectTeaType field is required."** are coming because By default MVC adds [Required] attribute to non-nullable value types. To fix it, add the following line into Application_Start method in Global.asax file.

```
DataAnnotationsModelValidatorProvider.AddImplicitRequiredAttributeForValueTypes
= false;
```



The screenshot shows a web form with a label "Select Tea Type:" followed by a dropdown menu currently displaying "Coffee". To the right of the dropdown is a "submit" button. Below the form, a message states "You Selected Tea Type: Coffee". At the bottom of the page, there is a copyright notice: "© 2017 - My ASP.NET Application".

(https://www.completecsharp tutorial.com/wp-content/uploads/2018/02/2_dropdownlist.png)

@HTML.DROPDOWNLISTFOR: DEFINITION AND EXAMPLE

Html.DropDownListFor is strongly bounded with model properties and checks for all errors at compile time. Before using this method, you must bind views with a model. Add model name in the top of the view as follows:

```
@using HtmlHelperDemo.Models
@model UserModel
```

Define:

You can define **Html.DropDownListFor** as follows:

```
Html.DropDownListFor(model => model.property, IEnumerable<SelectListItem>  
selectList, string optionLabel, object htmlAttributes)
```

a. model => model.property: It binds DropDownListFor with specific model property.

b. IEnumerable<SelectListItem> selectList: It is the list of items.

c. string optionLabel is the optional informative string like please choose item, select item, select value etc.

d. object HtmlAttributes adds extra features to dropdownlist.

Example:

Model: UserModel.cs

```
1 namespace HtmlHelperDemo.Models
2 {
3     public class UserModel
4     {
5         public TeaType SelectTeaType { get; set; }
6     }
7
8     public enum TeaType
9     {
10         Tea, Coffee, GreenTea, BlackTea
11     }
12 }
```

Views: Index.cshtml

```
1  @using HtmlHelperDemo.Models
2  @model UserModel
3  <br /><br />
4  @using (Html.BeginForm("Index", "Home", FormMethod.Post))
5  {
6      <b>Select Tea Type: </b>
7      @Html.DropDownListFor(m => m.SelectTeaType, new SelectList(En
8      <input type="submit" value="submit" />
9  }
10
11 <h4>You Selected</h4>
12 <b>Tea Type: </b>@ViewBag.TeaType
```

Controller: HomeController.cs

```
1  using System.Web.Mvc;
2  using HtmlHelperDemo.Models;
3
4  namespace HtmlHelperDemo.Controllers
5  {
6      public class HomeController : Controller
7      {
8          public ActionResult Index()
9          {
10             return View();
11         }
12
13         [HttpPost]
14         public ActionResult Index(UserModel model)
15         {
16             var selectedValue = model.SelectTeaType;
17             ViewBag.TeaType = selectedValue.ToString();
18             return View();
19         }
20     }
21 }
```

```
using System.Web;  
using System.Web.Mvc;  
using System.Web.Optimization;  
using System.Web.Routing;  
  
namespace HtmlHelperDemo  
{  
    public class MvcApplication : System.Web.HttpApplication  
    {  
        protected void Application_Start()  
        {  
            AreaRegistration.RegisterAllAreas();  
            FilterConfig.RegisterGlobalFilters(GlobalFilters.Filters);  
            RouteConfig.RegisterRoutes(RouteTable.Routes);  
            BundleConfig.RegisterBundles(BundleTable.Bundles);  
            DataAnnotationsModelValidatorProvider.  
                AddImplicitRequiredAttributeForValueTypes = false;  
        }  
    }  
}
```

(https://www.completecsharp tutorial.com/wp-content/uploads/2018/02/3_dropdownlist.png)

Html output

Coffee ▼

SUMMARY

In this chapter, you learned basic **Html.DropDownList** operation with complete programming example. Here I am adding some more programming examples on DropDownList, you must check and learn them for better understanding. In the next chapter, you will learn Html.ListBox.

MORE ARTICLES

Previous Article

(<https://www.completecsharp tutorial.com/asp-net-mvc5/html-radiobutton-and-html-radiobuttonfor-example-in-asp-net-mvc5/>)

mvc.php) (<https://www.completecsharptutorial.com/asp-net-mvc5/html-listboxfor-and-html-listboxforfor-example-in-asp-net-mvc.php>)

[Next Article](#)

SHARE YOUR THOUGHT



Copyright © 2011-2022 Complete C# Tutorial (<https://www.completecsharptutorial.com>) | Copyright (<https://www.completecsharptutorial.com/copyright.php>) | Cookie and Privacy (<https://www.completecsharptutorial.com/privacy.php>)