Реализация динамического выпадающего списка

По мотивам статьи «ASP.NET MVC 3: реализация динамического выпадающего списка» для ASP.NET 5.

# ASP.NET MVC 3: реализация динамического выпадающего списка

Опубликовано: 29.07.2011 23:26

Обновлено: 27.08.2011 14:47

Источник: [Implementing Dynamic DropDownList in ASP.NET MVC 3 Framework](http://www.highoncoding.com/Articles/770_Implementing_Dynamic_DropDownList_in_ASP_NET_MVC_3_Framework.aspx)

Перевод: [ASP.NET MVC 3: реализация динамического выпадающего списка | Addicted to .Net](http://blog.vkuznetsov.ru/posts/2011/07/29/aspnet-mvc-3-realizaciya-dinamicheskogo-vypadayushhego-spiska)

## Аннотация

Динамические выпадающие списки (DropDownList) позволяют пользователю выбрать элемент из первого выпадающего списка, который заполняет второй выпадающий список на основе выбора в первом. В этой статье мы намерены продемонстрировать, как реализовать динамические выпадающие списки с использованием фреймворка ASP.NET MVC 3.

## Создание модели представления

Первым шагом является создание модели представления (ViewModel), которая представляет элементы для представления. Наша модель представления "вращается" вокруг отношений между категориями и продуктами. Пользователь будет выбирать категорию из выпадающего списка, который заполнит все соответствующие продукты в другом выпадающем списке. Реализация ниже показывает модель ArticleEditViewModel:

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.Linq;

using System.Web;

using System.Web.Mvc;

namespace MvcApplication1.ViewModels

{

public class ArticleEditViewModel

{

private List<SelectListItem> \_categories = new List<SelectListItem>();

private List<SelectListItem> \_products = new List<SelectListItem>();

[Required(ErrorMessage = "Please select a category")]

public string SelectedCategory { get; set; }

[Required(ErrorMessage = "Please select a product")]

public string SelectedProduct { get; set; }

public List<SelectListItem> Products

{

get { return \_products; }

}

public List<SelectListItem> Categories

{

get

{

\_categories.Add(new SelectListItem(){ Text = "Ruby", Value = "1"});

\_categories.Add(new SelectListItem(){ Text = "JavaScript",Value="2"});

return \_categories;

}

}

}

}

Как вы можете видеть, чтобы сохранить наш пример простым, мы жестко закодировали категории. В следующем разделе мы покажем, как заполнить выпадающий список категорий используя модель ArticleEditViewModel.

## Заполнение выпадающего списка категорий:

Действие (action) Index контроллера HomeController возвращает новую модель ArticleEditViewModel для представления. Представление использует свойство Categories модели ArticleEditViewModel для заполнения выпадающего списка. Действие Index приведено ниже:

public ActionResult Index()

{

return View(new ArticleEditViewModel());

}

Выпадающий список заполняется кодом, представленным ниже:

<text>Select a category: </text> @Html.DropDownListFor(x => x.SelectedCategory, new SelectList(Model.Categories, "Value", "Text"), "Please Select a category")

@Html.ValidationMessageFor(x => x.SelectedCategory)

Выпадающий список привязан к свойству SelectedCategory (выбранная катерория) модели представления. Свойство SelectedCategory также декорировано атрибутом [Required] (обязательное), который гарантирует, что SelectedCategory не останется пустым или со значением по по умолчанию. Скриншот ниже показывает результат проверки, которая срабатывает, когда выпадающий список категорий остается со значением по умолчанию при завершении заполнения формы.