ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет програмування та комп’ютерних

і телекомунікаційних систем

Кафедра інженерії програмного забезпечення

**Лабораторна робота №**7

з дисципліни «Програмування в інтернет»

на тему:

«Введення в мультимовні сайти. Ресурси. Логіка мультимовного сайту. Фільтр локалізації.»

**Виконав:**

студент 1 курсу, групи ІПЗс-20-1, варіант 12 Д.А.Ямборко

**Перевірив:**\_\_\_\_\_\_\_\_\_\_\_\_\_\_ О.М.Яшина

(підпис)

Хмельницький – 2020 р.

Мета. Отримати навики розробки веб-додатків з використанням мультимовних сайтів.

Завдання

Реалізувати логіку мультимовного сайту та фільтр локалізації для Web - додатка розробленого в завданні до лабораторної роботи №4.

Код Програми:

Моделі:

DTP.cs:

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.Linq;

using System.Web;

namespace Lr1.Models

{

public class DTP

{

public int Id { get; set; }

[Required(ErrorMessageResourceType = typeof(Resources.Resource),

ErrorMessageResourceName = "NameRequired")]

[Display(Name = "Name", ResourceType = typeof(Resources.Resource))]

public string Name { get; set; }

[Required(ErrorMessageResourceType = typeof(Resources.Resource),

ErrorMessageResourceName = "TypeRequired")]

[Display(Name = "Type", ResourceType = typeof(Resources.Resource))]

public string Type { get; set; }

[Required(ErrorMessageResourceType = typeof(Resources.Resource),

ErrorMessageResourceName = "PriceRequired")]

[Display(Name = "Price", ResourceType = typeof(Resources.Resource))]

public int Price { get; set; }

}

}

AccountViewModels.cs:

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

namespace Lr1.Models

{

public class ExternalLoginConfirmationViewModel

{

[Required]

[Display(Name = "Adrel", ResourceType = typeof(Resources.Resource))]

public string Email { get; set; }

}

public class ExternalLoginListViewModel

{

public string ReturnUrl { get; set; }

}

public class SendCodeViewModel

{

public string SelectedProvider { get; set; }

public ICollection<System.Web.Mvc.SelectListItem> Providers { get; set; }

public string ReturnUrl { get; set; }

public bool RememberMe { get; set; }

}

public class VerifyCodeViewModel

{

[Required]

public string Provider { get; set; }

[Required]

[Display(Name = "Code", ResourceType = typeof(Resources.Resource))]

public string Code { get; set; }

public string ReturnUrl { get; set; }

[Display(Name = "Rembrow", ResourceType = typeof(Resources.Resource))]

public bool RememberBrowser { get; set; }

public bool RememberMe { get; set; }

}

public class ForgotViewModel

{

[Required]

[Display(Name = "Adrel", ResourceType = typeof(Resources.Resource))]

public string Email { get; set; }

}

public class LoginViewModel

{

[Required]

[Display(Name = "Adrel", ResourceType = typeof(Resources.Resource))]

[EmailAddress]

public string Email { get; set; }

[Required]

[DataType(DataType.Password)]

[Display(Name = "Password", ResourceType = typeof(Resources.Resource))]

public string Password { get; set; }

[Display(Name = "Remme", ResourceType = typeof(Resources.Resource))]

public bool RememberMe { get; set; }

}

public class RegisterViewModel

{

[Required]

[EmailAddress]

[Display(Name = "Adrel", ResourceType = typeof(Resources.Resource))]

public string Email { get; set; }

[Required]

[Display(Name = "Namei", ResourceType = typeof(Resources.Resource))]

public string Name { get; set; }

[Required]

[Display(Name = "Sname", ResourceType = typeof(Resources.Resource))]

public string Surname { get; set; }

[Required]

[StringLength(100, ErrorMessage = "Значение {0} должно содержать не менее {2} символов.", MinimumLength = 6)]

[DataType(DataType.Password)]

[Display(Name = "Password", ResourceType = typeof(Resources.Resource))]

public string Password { get; set; }

[DataType(DataType.Password)]

[Display(Name = "Passconf", ResourceType = typeof(Resources.Resource))]

[Compare("Password", ErrorMessage = "Пароль и его подтверждение не совпадают.")]

public string ConfirmPassword { get; set; }

}

public class ResetPasswordViewModel

{

[Required]

[EmailAddress]

[Display(Name = "Adrel", ResourceType = typeof(Resources.Resource))]

public string Email { get; set; }

[Required]

[StringLength(100, ErrorMessage = "Значение {0} должно содержать не менее {2} символов.", MinimumLength = 6)]

[DataType(DataType.Password)]

[Display(Name = "Password", ResourceType = typeof(Resources.Resource))]

public string Password { get; set; }

[DataType(DataType.Password)]

[Display(Name = "Passconf", ResourceType = typeof(Resources.Resource))]

[Compare("Password", ErrorMessage = "Пароль и его подтверждение не совпадают.")]

public string ConfirmPassword { get; set; }

public string Code { get; set; }

}

public class ForgotPasswordViewModel

{

[Required]

[EmailAddress]

[Display(Name = "Adrel", ResourceType = typeof(Resources.Resource))]

public string Email { get; set; }

}

}

Контролери:

HomeController.cs

using Lr1.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using Lr1.Filters;

namespace Lr1.Controllers

{

[Culture]

public class HomeController : Controller

{

ApplicationDbContext context = new ApplicationDbContext();

public ActionResult Index()

{

IEnumerable<DTP> dtps = context.Dtps;

ViewBag.Dtps = dtps;

return View();

}

public ActionResult About()

{

ViewBag.Message = "Your application description page.";

return View();

}

public ActionResult Contact()

{

ViewBag.Message = "Your contact page.";

return View();

}

public ActionResult ChangeCulture(string lang)

{

string returnUrl = Request.UrlReferrer.AbsolutePath;

// Список культур

List<string> cultures = new List<string>() { "ua", "en"};

if (!cultures.Contains(lang))

{

lang = "ua";

}

// Сохраняем выбранную культуру в куки

HttpCookie cookie = Request.Cookies["lang"];

if (cookie != null)

cookie.Value = lang; // если куки уже установлено, то обновляем значение

else

{

cookie = new HttpCookie("lang");

cookie.HttpOnly = false;

cookie.Value = lang;

cookie.Expires = DateTime.Now.AddYears(1);

}

Response.Cookies.Add(cookie);

return Redirect(returnUrl);

}

}

}

Представлення:

\_Layout.cshtml

<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>@ViewBag.Title - приложение ASP.NET</title>

@Styles.Render("~/Content/css")

@Scripts.Render("~/bundles/modernizr")

</head>

<body>

<div class="navbar navbar-inverse navbar-fixed-top">

<div class="container">

<div class="navbar-header">

<button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse">

<span class="icon-bar"></span>

<span class="icon-bar"></span>

<span class="icon-bar"></span>

</button>

@Html.ActionLink(Resources.Resource.Dtp, "Index", "Home", new { area = "" }, new { @class = "navbar-brand" })

</div>

<div class="navbar-collapse collapse">

<ul class="nav navbar-nav">

<li>@Html.ActionLink(Resources.Resource.Home, "Index", "Home")</li>

@if (User.Identity.IsAuthenticated && User.IsInRole("admin"))

{

<li>@Html.ActionLink(Resources.Resource.About, "About", "Home")</li>

}

<li>@Html.ActionLink(Resources.Resource.Contact, "Contact", "Home")</li>

<li><a href="/AddDTP/Create/">@Resources.Resource.Add</a></li>

</ul>

@Html.Partial("\_LoginPartial")

</div>

</div>

</div>

<div class="container body-content">

<div>

@using (Html.BeginForm("ChangeCulture", "Home"))

{

<select name="lang">

<option disabled selected>@Resources.Resource.ChooseLang</option>

<option value="uk">Українська</option>

<option value="en">English</option>

</select>

<input type="submit" name="submit" value="@Resources.Resource.ChooseLang">

}

</div>

@RenderBody()

<hr />

<footer>

<p>&copy; @DateTime.Now.Year</p>

</footer>

</div>

@Scripts.Render("~/bundles/jquery")

@Scripts.Render("~/bundles/bootstrap")

@RenderSection("scripts", required: false)

</body>

</html>

\_LoginPartial.cshtml

@using Microsoft.AspNet.Identity

@if (Request.IsAuthenticated)

{

using (Html.BeginForm("LogOff", "Account", FormMethod.Post, new { id = "logoutForm", @class = "navbar-right" }))

{

@Html.AntiForgeryToken()

<ul class="nav navbar-nav navbar-right">

<li>

@Html.ActionLink(Resources.Resource.Hello + " " + User.Identity.GetUserName() + "!", "Index", "Manage", routeValues: null, htmlAttributes: new { title = "Manage" })

</li>

<li><a href="javascript:document.getElementById('logoutForm').submit()">@Resources.Resource.Logout</a></li>

</ul>

}

}

else

{

<ul class="nav navbar-nav navbar-right">

<li>@Html.ActionLink(Resources.Resource.Registration, "Register", "Account", routeValues: null, htmlAttributes: new { id = "registerLink" })</li>

<li>@Html.ActionLink(Resources.Resource.Login, "Login", "Account", routeValues: null, htmlAttributes: new { id = "loginLink" })</li>

</ul>

}

Home/Index.cshtml

@{

ViewBag.Title = "Home Page";

}

<!DOCTYPE html>

<html>

<head>

<meta name="viewport" content="width=device-width" />

<title>@Resources.Resource.DAI</title>

</head>

<body>

<div>

<h3>@Resources.Resource.Dtp</h3>

<table>

<tr>

<td><p>@Resources.Resource.Name</p></td>

<td><p>@Resources.Resource.Type</p></td>

<td><p>@Resources.Resource.Price</p></td>

<td></td>

</tr>

@foreach (var b in ViewBag.Dtps)

{

<tr>

<td><p> @b.Name </p></td>

<td><p> @b.Type </p></td>

<td><p> @b.Price </p></td>

<td><p> <a href="/Shtraf/Shtraf/@b.Id"> @Resources.Resource.Oform</a></p></td>

</tr>

}

</table>

</div>

</body>

</html>

AddDTP/Edit.cshtml

@{

ViewBag.Title = "EditDTP";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

@model Lr1.Models.DTP

<h2>@Resources.Resource.Detp № @Model.Id</h2>

@using (Html.BeginForm("EditDTP", "Home", FormMethod.Post))

{

<fieldset>

@Html.HiddenFor(m => m.Id)

<p>

@Html.LabelFor(m => m.Name, Resources.Resource.Detp)

<br />

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

</p>

<p>

@Html.LabelFor(m => m.Type, Resources.Resource.Type)

<br />

@Html.EditorFor(m => m.Type)

</p>

<p>

@Html.LabelFor(m => m.Price, Resources.Resource.Price)

<br />

@Html.EditorFor(m => m.Price)

</p>

<p><input type="submit" value="@Resources.Resource.Edit" /></p>

</fieldset>

}

AddDTP/Create

@model Lr1.Models.DTP

@{

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>@Resources.Resource.Ndtp</h2>

@using (Html.BeginForm())

{

@Html.LabelFor(model => model.Name, Resources.Resource.Name)

<br />

@Html.EditorFor(model => model.Name)

<br />

<br />

@Html.LabelFor(model => model.Type, Resources.Resource.Type)

<br />

@Html.EditorFor(model => model.Type)

<br />

<br />

@Html.LabelFor(model => model.Price, Resources.Resource.Price)

<br />

@Html.EditorFor(model => model.Price)

<br />

<br />

<input type="submit" value="@Resources.Resource.Add" />

}

<!DOCTYPE html>

<html>

<head>

<meta name="viewport" content="width=device-width" />

<title>@Resources.Resource.DAI</title>

</head>

<body>

<div>

<h3>@Resources.Resource.Dtp</h3>

<table>

<tr>

<td><p>@Resources.Resource.Name</p></td>

<td><p>@Resources.Resource.Type</p></td>

<td><p>@Resources.Resource.Price</p></td>

<td></td>

</tr>

@foreach (var b in ViewBag.Dtps)

{

<tr>

<td><p> @b.Name </p></td>

<td><p> @b.Type </p></td>

<td><p> @b.Price </p></td>

<td><p> <a href="/AddDTP/Delete/@b.Id"> @Resources.Resource.Delete </a></p></td>

<td><p> <a href="/AddDTP/EditDTP/@b.Id"> @Resources.Resource.Edit </a></p></td>

</tr>

}

</table>

</div>

</body>

</html>

Filters

CultureAttribute.cs

using System;

using System.Collections.Generic;

using System.Globalization;

using System.Threading;

using System.Web;

using System.Web.Mvc;

namespace Dovid.Filters

{

public class CultureAttribute : FilterAttribute, IActionFilter

{

public void OnActionExecuted(ActionExecutedContext filterContext)

{

string cultureName = null;

// Получаем куки из контекста, которые могут содержать установленную культуру

HttpCookie cultureCookie = filterContext.HttpContext.Request.Cookies["lang"];

if (cultureCookie != null)

cultureName = cultureCookie.Value;

else

cultureName = "uk";

// Список культур

List<string> cultures = new List<string>() { "uk", "en" };

if (!cultures.Contains(cultureName))

{

cultureName = "uk";

}

Thread.CurrentThread.CurrentCulture = CultureInfo.CreateSpecificCulture(cultureName);

Thread.CurrentThread.CurrentUICulture = CultureInfo.CreateSpecificCulture(cultureName);

}

public void OnActionExecuting(ActionExecutingContext filterContext)

{

//не реализован

}

}

}

Вигляд програми:

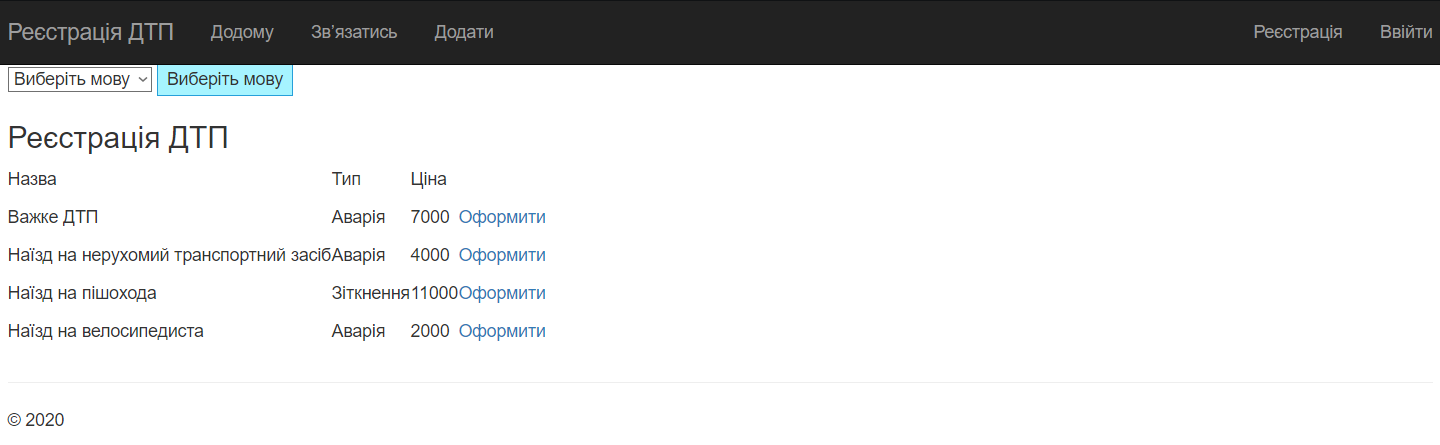


Рис 1.- Вигляд головної сторінки українською мовою

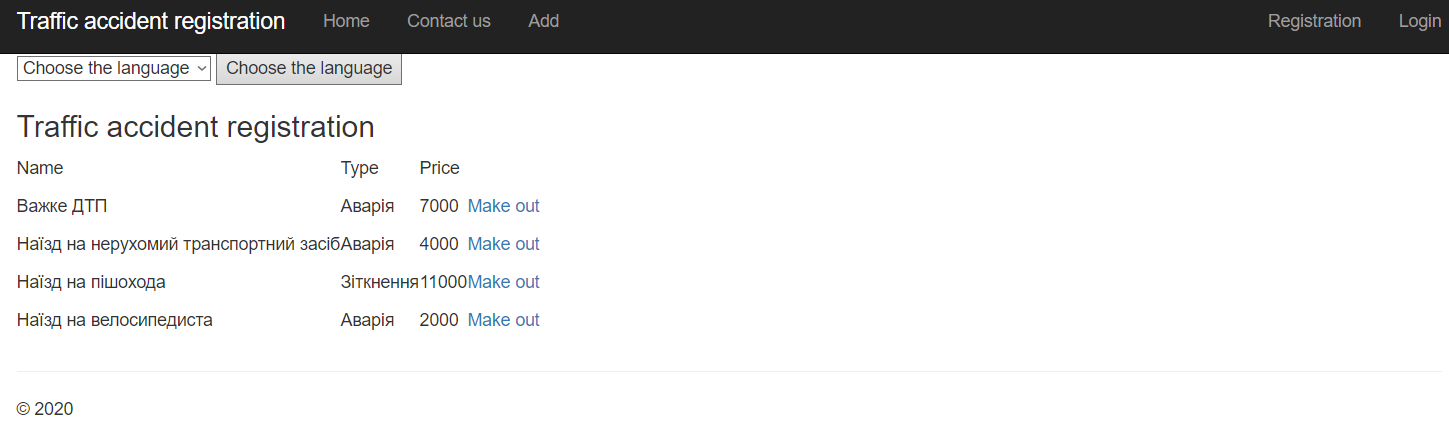


Рис 2.- Вигляд головної сторінки англійською мовою



Рис. 3 – Файл ресурсів українською мовою

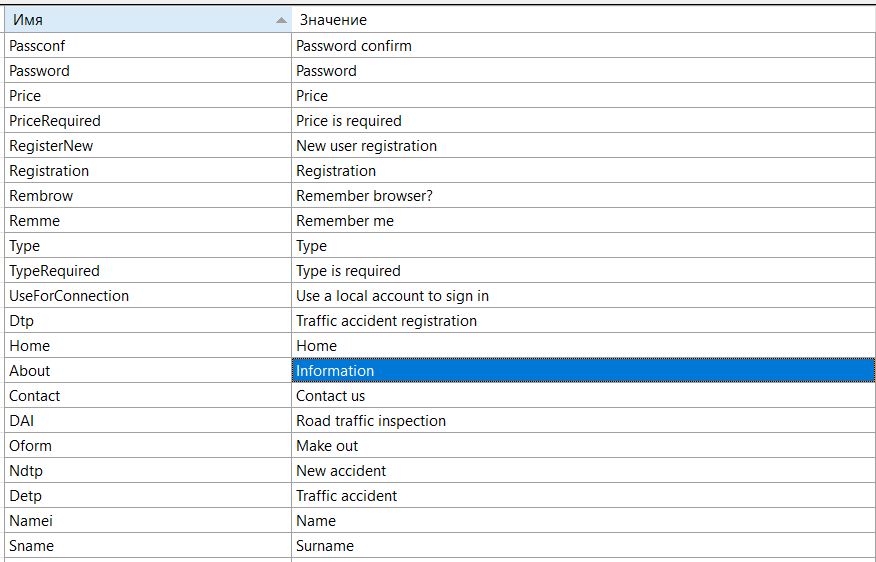


Рис. 4 – Файл ресурсів англійською мовою

GIT repository: https://github.com/alsowentafk/1\_semestr\_knu/tree/master/PI/labs/Lr7/Lr1

Контрольні питання:

1. Об’єкт HttpContext

Отримати дані про запит: про браузер, про ip-адресу користувача і ще ряд інших в ASP.NET MVC можна за допомогою HttpContext. Також в контролері ви можете звернутися до об'єкта ControllerContext, який має властивість HttpContext і по суті надає доступ до тієї ж функціональності. У той же час між ними є деякі відмінності. Об'єкт HttpContext описує дані конкретного http-запиту, який обробляється додатком.

2. Об’єкт ControllerContext

ControllerContext описує дані httpзапиту по відношенню до даного контролера

3. Властивість Request

Інформація про запит міститься у властивості Request. HttpContext.Request представляє об'єкт класу, наслідуваного від HttpRequestBase, і тому містить всі його властивості. Розглянемо деякі з них:

- HttpContext.Request.Browser - отримання браузера

користувача;

- HttpContext.Request.UserAgent - звернення до агента

користувача;

- HttpContext.Request.RawUrl - отримання url запиту;

- HttpContext.Request.UserHostAddress - отримання IP-адреси

користувача;

- Отримання реферера - HttpContext.Request.UrlReferrer == null?

"" - HttpContext.Request.UrlReferrer.AbsoluteUri

4. Властивість HttpContext.Request.Cookies

Щоб отримати куки, нам треба скористатися властивістю HttpContext.Request.Cookies:

string id =HttpContext.Request.Cookies["id"].Value;

В даному випадку, якщо у нас встановлена на стороні клієнта куки "id", то ми отримаємо її значення.

5. Властивість HttpContext.Response.Cookies

Однак перш ніж отримувати значення куки, їх треба встановити. Для цього нам служить інша властивість - властивість HttpContext.Response. Наприклад, встановимо в куки значення "id":

HttpContext.Response.Cookies["id"].Value = "ca-4353w";

6. Метод HttpContext.Response.Write

Ми також можемо управляти відповіддю за допомогою методів. Метод AddHeader дозволяє додати до відповіді додатковий заголовок. Крім того, нам необов'язково викликати метод View в дії контролера, щоб відправити клієнту відповідь запиту. Ми цілком можемо скористатися методом HttpContext.Response.Write:

public void ContextData()

{

HttpContext.Response.Write("<h1>Hello World</h1>");

}

Метод HttpContext.Response.Write тут додає в потік певний вміст, переданий в якості параметра.

7. Визначення користувача

Також об'єкт HttpContext дозволяє

визначити користувача:

bool IsAdmin = HttpContext.User.IsInRole("admin");

// Визначаємо, чи належить користувач до адміністраторів

bool IsAuth=HttpContext.User.Identity.IsAuthenticated;

// Аутентифікований користувач

string login = HttpContext.User.Identity.Name;

// Логін авторизованого користувача

8. Визначення типу ресурсу

[Required(ErrorMessageResourceType = typeof(Resources.Resource), ErrorMessageResourceName = "CountryRequired")]В даному випадку для спрощення прикладу використовуємо всього три поля. До двох застосовані атрибути Required і Display. Тільки в даному випадку атрибут Required вказує на тип ресурсу, який буде застосовуватися (ErrorMessageResourceType)

9. Створення ресурсів

Ресурси можуть містити не тільки повідомлення про помилки валідації, але й інші дані, наприклад, заголовки сторінок і т.д. Створимо в проекті папку Resources. У цю папку додамо файл ресурсів Resource.resx. Це буде файл ресурсів за замовчуванням, який буде використовуватися для української мови.

10. Файл ресурсів

Перед використанням ресурсів вкажемо деякі їх налаштування для кожного ресурсу у вікні Властивостей (Propertires) внизу під вікном проекту.

Тут нам треба встановити наступні параметри:

- Build Action - в якості типу побудови виберемо значення Embedded Resource;

- Custom Tool - як інструмент створення ресурсів введемо в поле PublicResXFileCodeGenerator;

- Custom Tool Namespace - в якості простору імен вкажемо Resources. У даному випадку важливо вказати саме той простір імен, яке ми збираємося використовувати. Використовуючи вище атрибути при визначенні моделі, ми вказали ErrorMessageResourceType = typeof (Resources.Resource), де в якості простору імен виступає Resources.

11. Заповнення ресурсів

Тепер перейдемо до заповнення самих ресурсів. Натиснемо на файл Resource.resx і у відкрилася таблицю внесемо назви ресурсів та їх значення.

Це і є весь набір використовуваних ресурсів, і ми можемо визначити безліч різних ресурсів. Подібний набір треба вказати і для інших ресурсів, тільки вказавши значення на відповідній мові.

12. Звернення до ресурсів в представленні

Крім того, ми можемо в коді представлень звернутися до ресурсів, наприклад: @Resources.Resource.NameRequired. Тут Resources - простір імен, Resource - тип ресурсу (утворюється від назви файлу ресурсів), а NameRequired - ім'я. У такому ж форматі можна використовувати й інші ресурси.

13. Умовності при створенні ресурсів

При локалізації мають значення в

найменуванні ресурсів суфікси. Так, ми можемо використовувати

нейтральні культури, наприклад: Resource.en.resx. Тут суфікс en

позначає англійську мову в цілому. Але ми можемо конкретизувати

культуру, наприклад, задати ресурс тільки для американської версії

англійської: Resource.en-US.resx. Суфікс культури має наступний

формат: languagecode-country / regioncode, де languagecode - код мови,

а country / regioncode - код субкультури.

14. Збереження культури в куки

Для збереження культури в куки потрібно в контролері реалізувати даний код:

cookie = new HttpCookie("lang"); //Створює куки з назвою “lang”, мова

cookie.HttpOnly = false; //Дає доступ до кука не тільки через Http

cookie.Value = lang; //Задає значення даного кука

cookie.Expires = DateTime.Now.AddYears(1); //Задає термін існування кука, в даному випадку на 1 рік.

15. Отримання куки з контексту

Для отримання куків з контексту потрібно реалізувати такий код:

HttpCookie cultureCookie = filterContext.HttpContext.Request.Cookies["lang"];

В даному випадку отримуються куки мови і даний код реалізується в фільтрі.