PROGRAM.CS

using Microsoft.AspNetCore.Localization;

using Microsoft.Extensions.Options;

using System.Globalization;

using System.Linq;

using System.Net.Http;

var builder = WebApplication.CreateBuilder(args);

// ? ???????? ?? ???????? ?? ????????????

builder.Services.AddLocalization(options => options.ResourcesPath = "Resources");

// ? ????????? ???????

var supportedCultures = new[] { "mk", "en", "sr" };

var defaultCulture = "en"; // ??????? ?? ??????

// ? ????????????? ?? RequestLocalizationOptions

builder.Services.Configure<RequestLocalizationOptions>(options =>

{

options.DefaultRequestCulture = new RequestCulture(defaultCulture);

options.SupportedCultures = supportedCultures.Select(c => new CultureInfo(c)).ToList();

options.SupportedUICultures = supportedCultures.Select(c => new CultureInfo(c)).ToList();

// ????????? ?????? ?? ?????????? ?? RequestCultureProviders

options.RequestCultureProviders.Clear(); // ????????? ?? ???????????

options.RequestCultureProviders.Add(new CookieRequestCultureProvider()); // ????????? CookieRequestCultureProvider

});

builder.Services.AddRazorPages();

builder.Services.AddServerSideBlazor();

builder.Services.AddScoped<HttpClient>(sp =>

new HttpClient { BaseAddress = new Uri("http://localhost:5098") });

var app = builder.Build();

// ? Middleware ?? ????????????

var localizationOptions = app.Services.GetRequiredService<IOptions<RequestLocalizationOptions>>().Value;

app.UseRequestLocalization(localizationOptions);

// ? ???????? ??????? ? ????????

app.UseStaticFiles();

app.UseRouting();

// ? ???????? ?? ?????????? (??? ???? API ?????????? ??? ??????? ?? ????)

app.MapControllers();

// ? ???????? ?? Blazor Hub

app.MapBlazorHub();

// ? ?????? ?? Blazor ??????????

app.MapFallbackToPage("/\_Host");

// ? ?????????? ?? ????????????

app.Run();

PAGINATEDDATA.RAZOR

@page "/taxiridesapp"

@using BlazorAppPaginationClient.Models

@using System.Text

@inject IJSRuntime JSRuntime

@inject HttpClient Http

@inject IStringLocalizer<SharedResource> L

@inject NavigationManager NavigationManager

@using BlazorAppPaginationClient.Resources

@using Microsoft.Extensions.Localization

<h3>@L["TaxiRides"]</h3>

<div class="controls-container">

<div class="language-selector">

<label>@L["Language"]</label>

<select @onchange="OnLanguageChanged">

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

<option value="mk">Македонски</option>

<option value="sr">Српски</option>

</select>

</div>

<div class="search-box">

<label>@L["SearchLabel"]</label>

<input type="text" @bind="SearchText" @bind:event="oninput" placeholder="@L["SearchPlaceholder"]" class="search-input" />

</div>

<div class="page-size-selector">

<label>@L["PageSize"]</label>

<select @bind="PageSize" @bind:event="onchange" class="dropdown">

<option value="10">10</option>

<option value="20">20</option>

<option value="50">50</option>

<option value="100">100</option>

</select>

</div>

<div class="total-records">

<strong>@L["TotalRecords"] @FilteredRecords</strong>

</div>

</div>

@if (TaxiRecords == null)

{

<p>@L["Loading"]...</p>

}

else

{

<table class="table">

<thead>

<tr>

<th>#</th>

@foreach (var property in typeof(TaxiRecord).GetProperties())

{

<th @onclick="() => SortByColumn(property.Name)" class="sortable">

@property.Name @GetSortIndicator(property.Name)

</th>

}

</tr>

</thead>

<tbody>

@foreach (var (ride, index) in PaginatedRides.Select((r, i) => (r, i + 1 + ((\_currentPage - 1) \* PageSize))))

{

<tr>

<td>@index</td>

@foreach (var property in typeof(TaxiRecord).GetProperties())

{

<td>@property.GetValue(ride)</td>

}

</tr>

}

</tbody>

</table>

<div class="pagination-container">

<button @onclick="PreviousPage" disabled="@(\_currentPage == 1)" class="pagination-button">@L["Previous"]</button>

<span>@L["Page"] @\_currentPage @L["Of"] @TotalPages</span>

<button @onclick="NextPage" disabled="@(\_currentPage >= TotalPages)" class="pagination-button">@L["Next"]</button>

</div>

<div class="page-input-container">

<label>@L["EnterPage"]</label>

<input type="number" @bind="EnteredPage" class="page-input" />

<button @onclick="GoToPage" class="pagination-button">@L["Go"]</button>

</div>

<div class="export-buttons">

<button class="btn btn-primary" @onclick="ExportToCsv">@L["ExportCsv"]</button>

<button class="btn btn-success" @onclick="ExportToExcel">@L["ExportExcel"]</button>

<button class="btn btn-danger" @onclick="ExportToPdf">@L["ExportPdf"]</button>

</div>

}

<style>

.controls-container {

display: flex;

flex-wrap: wrap;

justify-content: space-between;

align-items: center;

margin-bottom: 15px;

gap: 10px;

}

.search-input, .dropdown, .page-input {

padding: 5px;

border-radius: 5px;

border: 1px solid #ccc;

}

.table {

width: 100%;

border-collapse: collapse;

margin-top: 10px;

}

.table th, .table td {

border: 1px solid #ddd;

padding: 8px;

text-align: left;

}

.sortable:hover {

cursor: pointer;

background-color: #f1f1f1;

}

.pagination-container, .page-input-container {

display: flex;

justify-content: center;

align-items: center;

margin-top: 10px;

gap: 10px;

}

.pagination-button {

padding: 5px 10px;

border: none;

background-color: #007bff;

color: white;

border-radius: 5px;

cursor: pointer;

}

.pagination-button:disabled {

background-color: #ccc;

cursor: not-allowed;

}

.export-buttons {

margin-top: 15px;

text-align: center;

}

</style>

@code {

private List<TaxiRecord> TaxiRecords = new();

private string SearchText = "";

private int \_currentPage = 1;

private int PageSize = 10;

private int EnteredPage;

private int TotalRecords = 0;

private int FilteredRecords => FilteredRides.Count;

private int TotalPages => (int)Math.Ceiling((double)FilteredRecords / PageSize);

private string SortColumn = nameof(TaxiRecord.pickupTime);

private bool SortAscending = true;

private List<TaxiRecord> FilteredRides =>

string.IsNullOrWhiteSpace(SearchText)

? TaxiRecords

: TaxiRecords.Where(r => typeof(TaxiRecord).GetProperties()

.Any(p => p.GetValue(r)?.ToString()?.Contains(SearchText, StringComparison.OrdinalIgnoreCase) == true)).ToList();

private List<TaxiRecord> PaginatedRides =>

FilteredRides

.OrderBy(r => SortAscending ? GetPropertyValue(r, SortColumn) : null)

.ThenByDescending(r => SortAscending ? null : GetPropertyValue(r, SortColumn))

.Skip((\_currentPage - 1) \* PageSize)

.Take(PageSize)

.ToList();

protected override async Task OnInitializedAsync()

{

try

{

var uri = new Uri(NavigationManager.Uri);

var queryParams = System.Web.HttpUtility.ParseQueryString(uri.Query);

var culture = queryParams["culture"] ?? "en"; // Default to English if no culture parameter is found

var cultureInfo = new System.Globalization.CultureInfo(culture);

System.Threading.Thread.CurrentThread.CurrentCulture = cultureInfo;

System.Threading.Thread.CurrentThread.CurrentUICulture = cultureInfo;

TaxiRecords = await Http.GetFromJsonAsync<List<TaxiRecord>>("http://localhost:5098/api/TaxiRides/GetTaxiRides2");

TotalRecords = TaxiRecords.Count;

}

catch (Exception ex)

{

Console.WriteLine($"Error: {ex.Message}");

}

}

private void NextPage()

{

if (\_currentPage < TotalPages) \_currentPage++;

}

private void PreviousPage()

{

if (\_currentPage > 1) \_currentPage--;

}

private void GoToPage()

{

if (EnteredPage >= 1 && EnteredPage <= TotalPages)

{

\_currentPage = EnteredPage;

}

}

private void SortByColumn(string columnName)

{

if (SortColumn == columnName)

{

SortAscending = !SortAscending;

}

else

{

SortColumn = columnName;

SortAscending = true;

}

}

private string GetSortIndicator(string columnName)

{

if (SortColumn == columnName)

{

return SortAscending ? "▲" : "▼";

}

return "";

}

private object GetPropertyValue(TaxiRecord ride, string propertyName)

{

var property = typeof(TaxiRecord).GetProperty(propertyName);

return property?.GetValue(ride, null);

}

private async Task ExportToCsv()

{

var csvContent = GenerateCsvContent(PaginatedRides);

await JSRuntime.InvokeVoidAsync("saveAsFile", "taxi\_rides.csv", csvContent);

}

private async Task ExportToExcel()

{

var excelContent = GenerateExcelContent(PaginatedRides);

await JSRuntime.InvokeVoidAsync("saveExcel", "taxi\_rides.xlsx", excelContent);

}

private async Task ExportToPdf()

{

var pdfContent = GeneratePdfContent(PaginatedRides);

await JSRuntime.InvokeVoidAsync("savePdf", "taxi\_rides.pdf", pdfContent);

}

private string GenerateCsvContent(List<TaxiRecord> data)

{

var sb = new StringBuilder();

sb.AppendLine("Medallion,Hash License,Pickup Time,Drop-off Time,Duration,Distance,Pickup Longitude,Pickup Latitude,Drop-off Longitude,Drop-off Latitude,Payment Type,Fare Amount,Surcharge,Tax,Tip Amount,Tolls Amount,Total Amount");

foreach (var item in data)

{

sb.AppendLine($"{item.medallion},{item.hashLicense},{item.pickupTime},{item.dropOffTime},{item.duration},{item.distance},{item.pLongitude},{item.pLatitude},{item.dLongitude},{item.dLatitude},{item.paymentType},{item.fareAmount},{item.surcharge},{item.tax},{item.tipAmount},{item.tollsAmount},{item.totalAmount}");

}

return sb.ToString();

}

private object GenerateExcelContent(List<TaxiRecord> data)

{

var sheet = new List<List<object>> {

new List<object> { "Medallion", "Hash License", "Pickup Time", "Drop-off Time", "Duration", "Distance", "Pickup Longitude", "Pickup Latitude", "Drop-off Longitude", "Drop-off Latitude", "Payment Type", "Fare Amount", "Surcharge", "Tax", "Tip Amount", "Tolls Amount", "Total Amount" }

};

foreach (var item in data)

{

sheet.Add(new List<object> { item.medallion, item.hashLicense, item.pickupTime, item.dropOffTime, item.duration, item.distance, item.pLongitude, item.pLatitude, item.dLongitude, item.dLatitude, item.paymentType, item.fareAmount, item.surcharge, item.tax, item.tipAmount, item.tollsAmount, item.totalAmount });

}

return sheet;

}

private string GeneratePdfContent(List<TaxiRecord> data)

{

var content = "<h1>Taxi Rides</h1><table><thead><tr><th>Medallion</th><th>Hash License</th><th>Pickup Time</th><th>Drop-off Time</th><th>Duration</th><th>Distance</th><th>Pickup Longitude</th><th>Pickup Latitude</th><th>Drop-off Longitude</th><th>Drop-off Latitude</th><th>Payment Type</th><th>Fare Amount</th><th>Surcharge</th><th>Tax</th><th>Tip Amount</th><th>Tolls Amount</th><th>Total Amount</th></tr></thead><tbody>";

foreach (var item in data)

{

content += $"<tr><td>{item.medallion}</td><td>{item.hashLicense}</td><td>{item.pickupTime}</td><td>{item.dropOffTime}</td><td>{item.duration}</td><td>{item.distance}</td><td>{item.pLongitude}</td><td>{item.pLatitude}</td><td>{item.dLongitude}</td><td>{item.dLatitude}</td><td>{item.paymentType}</td><td>{item.fareAmount}</td><td>{item.surcharge}</td><td>{item.tax}</td><td>{item.tipAmount}</td><td>{item.tollsAmount}</td><td>{item.totalAmount}</td></tr>";

}

content += "</tbody></table>";

return content;

}

private async Task OnLanguageChanged(ChangeEventArgs e)

{

var culture = e.Value.ToString();

var uri = NavigationManager.ToAbsoluteUri(NavigationManager.Uri);

var queryParams = System.Web.HttpUtility.ParseQueryString(uri.Query);

queryParams["culture"] = culture;

NavigationManager.NavigateTo(uri.GetLeftPart(UriPartial.Path) + "?" + queryParams.ToString());

}

}

SHAREDRESOURCE.CS

namespace BlazorAppPaginationClient.Resources

{

public class SharedResource

{

}

}

CULTURECONTROLLER.CS

using Microsoft.AspNetCore.Localization;

using Microsoft.AspNetCore.Mvc;

using System;

[Route("culture")]

public class CultureController : Controller

{

[HttpGet("setculture")]

public IActionResult SetCulture(string culture, string redirectUri)

{

if (!string.IsNullOrEmpty(culture))

{

// Поставување на културата во колачињата

Response.Cookies.Append(

CookieRequestCultureProvider.DefaultCookieName,

CookieRequestCultureProvider.MakeCookieValue(new RequestCulture(culture)),

new CookieOptions { Expires = DateTimeOffset.UtcNow.AddYears(1) }

);

}

// Пренасочување назад на страницата

return LocalRedirect(redirectUri ?? "/");

}

}

LANGUAGESELECTOR.RAZOR

@inject NavigationManager NavigationManager

<select @onchange="OnLanguageChanged" class="form-select">

<option value="mk" selected="@IsSelected("mk")">Македонски</option>

<option value="en" selected="@IsSelected("en")">English</option>

<option value="sr" selected="@IsSelected("sr")">Srpski</option>

</select>

@code {

private async Task OnLanguageChanged(ChangeEventArgs e)

{

var culture = e.Value?.ToString() ?? "en"; // Поставување на култура по избор

var uri = NavigationManager.Uri;

// Креирање на новото URL за пренасочување

var newUrl = $"/culture/setculture?culture={culture}&redirectUri={Uri.EscapeDataString(uri)}";

NavigationManager.NavigateTo(newUrl, forceLoad: true); // Пренасочување

}

private string IsSelected(string culture)

{

var currentCulture = System.Globalization.CultureInfo.CurrentUICulture.Name;

return currentCulture.StartsWith(culture) ? "selected" : null; // Ако е избрана култура, врати "selected"

}

}

NAVMENU.RAZOR

@using BlazorAppPaginationClient.Resources

@using Microsoft.AspNetCore.Components

@using Microsoft.Extensions.Localization

@inject IStringLocalizer<SharedResource> localizer

@inject NavigationManager Navigation

<div class="top-row ps-3 navbar navbar-dark">

<div class="container-fluid">

<a class="navbar-brand" href="">@localizer["Taxi Rides"]</a>

<button title="Navigation menu" @onclick="ToggleNavMenu">

<span class="navbar-toggler-icon"></span>

</button>

</div>

</div>

<div class="@NavMenuCssClass" @onclick="ToggleNavMenu">

<nav class="flex-column">

<div class="nav-item px-3">

<**NavLink** class="nav-link" href="taxiridesapp">

<span class="oi oi-list-rich" aria-hidden="true"></span>

@localizer["TaxiRides"]

</**NavLink**>

</div>

</nav>

</div>

@code {

private bool collapseNavMenu = true;

private string? NavMenuCssClass => collapseNavMenu ? "collapse" : null;

private void ToggleNavMenu()

{

collapseNavMenu = !collapseNavMenu;

}

}

MAINLAYOUT.RAZOR

@inherits LayoutComponentBase

@using BlazorAppPaginationClient.Shared

<**PageTitle**>BlazorAppPaginationClient</**PageTitle**>

<div class="page">

<div class="sidebar">

<**NavMenu** />

</div>

<main>

<div class="top-row px-4" style="display: flex; justify-content: space-between; align-items: center;">

<a href="https://docs.microsoft.com/aspnet/" target="\_blank">About</a>

<LanguageSelector />

</div>

<article class="content px-4">

@Body

</article>

</main>

</div>

SHAREDRESOURCE.EN.RESX

<?xml version="1.0" encoding="utf-8"?>

<root>

<!--

Microsoft ResX Schema

Version 2.0

The primary goals of this format is to allow a simple XML format

that is mostly human readable. The generation and parsing of the

various data types are done through the TypeConverter classes

associated with the data types.

Example:

... ado.net/XML headers & schema ...

<resheader name="resmimetype">text/microsoft-resx</resheader>

<resheader name="version">2.0</resheader>

<resheader name="reader">System.Resources.ResXResourceReader, System.Windows.Forms, ...</resheader>

<resheader name="writer">System.Resources.ResXResourceWriter, System.Windows.Forms, ...</resheader>

<data name="Name1"><value>this is my long string</value><comment>this is a comment</comment></data>

<data name="Color1" type="System.Drawing.Color, System.Drawing">Blue</data>

<data name="Bitmap1" mimetype="application/x-microsoft.net.object.binary.base64">

<value>[base64 mime encoded serialized .NET Framework object]</value>

</data>

<data name="Icon1" type="System.Drawing.Icon, System.Drawing" mimetype="application/x-microsoft.net.object.bytearray.base64">

<value>[base64 mime encoded string representing a byte array form of the .NET Framework object]</value>

<comment>This is a comment</comment>

</data>

There are any number of "resheader" rows that contain simple

name/value pairs.

Each data row contains a name, and value. The row also contains a

type or mimetype. Type corresponds to a .NET class that support

text/value conversion through the TypeConverter architecture.

Classes that don't support this are serialized and stored with the

mimetype set.

The mimetype is used for serialized objects, and tells the

ResXResourceReader how to depersist the object. This is currently not

extensible. For a given mimetype the value must be set accordingly:

Note - application/x-microsoft.net.object.binary.base64 is the format

that the ResXResourceWriter will generate, however the reader can

read any of the formats listed below.

mimetype: application/x-microsoft.net.object.binary.base64

value : The object must be serialized with

: System.Runtime.Serialization.Formatters.Binary.BinaryFormatter

: and then encoded with base64 encoding.

mimetype: application/x-microsoft.net.object.soap.base64

value : The object must be serialized with

: System.Runtime.Serialization.Formatters.Soap.SoapFormatter

: and then encoded with base64 encoding.

mimetype: application/x-microsoft.net.object.bytearray.base64

value : The object must be serialized into a byte array

: using a System.ComponentModel.TypeConverter

: and then encoded with base64 encoding.

-->

<xsd:schema id="root" xmlns="" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">

<xsd:import namespace="http://www.w3.org/XML/1998/namespace" />

<xsd:element name="root" msdata:IsDataSet="true">

<xsd:complexType>

<xsd:choice maxOccurs="unbounded">

<xsd:element name="metadata">

<xsd:complexType>

<xsd:sequence>

<xsd:element name="value" type="xsd:string" minOccurs="0" />

</xsd:sequence>

<xsd:attribute name="name" use="required" type="xsd:string" />

<xsd:attribute name="type" type="xsd:string" />

<xsd:attribute name="mimetype" type="xsd:string" />

<xsd:attribute ref="xml:space" />

</xsd:complexType>

</xsd:element>

<xsd:element name="assembly">

<xsd:complexType>

<xsd:attribute name="alias" type="xsd:string" />

<xsd:attribute name="name" type="xsd:string" />

</xsd:complexType>

</xsd:element>

<xsd:element name="data">

<xsd:complexType>

<xsd:sequence>

<xsd:element name="value" type="xsd:string" minOccurs="0" msdata:Ordinal="1" />

<xsd:element name="comment" type="xsd:string" minOccurs="0" msdata:Ordinal="2" />

</xsd:sequence>

<xsd:attribute name="name" type="xsd:string" use="required" msdata:Ordinal="1" />

<xsd:attribute name="type" type="xsd:string" msdata:Ordinal="3" />

<xsd:attribute name="mimetype" type="xsd:string" msdata:Ordinal="4" />

<xsd:attribute ref="xml:space" />

</xsd:complexType>

</xsd:element>

<xsd:element name="resheader">

<xsd:complexType>

<xsd:sequence>

<xsd:element name="value" type="xsd:string" minOccurs="0" msdata:Ordinal="1" />

</xsd:sequence>

<xsd:attribute name="name" type="xsd:string" use="required" />

</xsd:complexType>

</xsd:element>

</xsd:choice>

</xsd:complexType>

</xsd:element>

</xsd:schema>

<resheader name="resmimetype">

<value>text/microsoft-resx</value>

</resheader>

<resheader name="version">

<value>2.0</value>

</resheader>

<resheader name="reader">

<value>System.Resources.ResXResourceReader, System.Windows.Forms, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>

</resheader>

<resheader name="writer">

<value>System.Resources.ResXResourceWriter, System.Windows.Forms, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>

</resheader>

<data name="Taxi Rides" xml:space="preserve">

<value>Taxi Rides</value>

</data>

<data name="Search" xml:space="preserve">

<value>Search</value>

</data>

<data name="Page Size" xml:space="preserve">

<value>Page Size</value>

</data>

<data name="Total Records" xml:space="preserve">

<value>Total Records</value>

</data>

<data name="Loading..." xml:space="preserve">

<value>Loading...</value>

</data>

<data name="Previous" xml:space="preserve">

<value>Previous</value>

</data>

<data name="Next" xml:space="preserve">

<value>Next</value>

</data>

<data name="Page" xml:space="preserve">

<value>Page</value>

</data>

<data name="of" xml:space="preserve">

<value>of</value>

</data>

<data name="Enter Page" xml:space="preserve">

<value>Enter Page</value>

</data>

<data name="Go" xml:space="preserve">

<value>Go</value>

</data>

<data name="Export to CSV" xml:space="preserve">

<value>Export to CSV</value>

</data>

<data name="Export to Excel" xml:space="preserve">

<value>Export to Excel</value>

</data>

<data name="Export to PDF" xml:space="preserve">

<value>Export to PDF</value>

</data>

<data name="medallion" xml:space="preserve">

<value>Medallion</value>

</data>

<data name="hashLicense" xml:space="preserve">

<value>Hash License</value>

</data>

<data name="pickupTime" xml:space="preserve">

<value>Pickup Time</value>

</data>

<data name="dropOffTime" xml:space="preserve">

<value>Drop-off Time</value>

</data>

<data name="duration" xml:space="preserve">

<value>Duration</value>

</data>

<data name="pLongitude" xml:space="preserve">

<value>Pickup Longitude</value>

</data>

<data name="pLatitude" xml:space="preserve">

<value>Pickup Latitude</value>

</data>

<data name="dLongitude" xml:space="preserve">

<value>Drop-off Longitude</value>

</data>

<data name="dLatitude" xml:space="preserve">

<value>Drop-off Latitude</value>

</data>

<data name="paymentType" xml:space="preserve">

<value>Payment Type</value>

</data>

<data name="fareAmount" xml:space="preserve">

<value>Fare Amount</value>

</data>

<data name="surcharge" xml:space="preserve">

<value>Surcharge</value>

</data>

<data name="tax" xml:space="preserve">

<value>Tax</value>

</data>

<data name="tipAmount" xml:space="preserve">

<value>Tip Amount</value>

</data>

<data name="tollsAmount" xml:space="preserve">

<value>Tolls Amount</value>

</data>

<data name="totalAmount" xml:space="preserve">

<value>Total Amount</value>

</data>

<data name="AppTitle" xml:space="preserve">

<value>Taxi Ride Viewer</value>

</data>

<data name="distance" xml:space="preserve">

<value>Distance</value>

</data>

</root>

SHAREDRESOUCE.MK.RESX

<?xml version="1.0" encoding="utf-8"?>

<root>

<!--

Microsoft ResX Schema

Version 2.0

The primary goals of this format is to allow a simple XML format

that is mostly human readable. The generation and parsing of the

various data types are done through the TypeConverter classes

associated with the data types.

Example:

... ado.net/XML headers & schema ...

<resheader name="resmimetype">text/microsoft-resx</resheader>

<resheader name="version">2.0</resheader>

<resheader name="reader">System.Resources.ResXResourceReader, System.Windows.Forms, ...</resheader>

<resheader name="writer">System.Resources.ResXResourceWriter, System.Windows.Forms, ...</resheader>

<data name="Name1"><value>this is my long string</value><comment>this is a comment</comment></data>

<data name="Color1" type="System.Drawing.Color, System.Drawing">Blue</data>

<data name="Bitmap1" mimetype="application/x-microsoft.net.object.binary.base64">

<value>[base64 mime encoded serialized .NET Framework object]</value>

</data>

<data name="Icon1" type="System.Drawing.Icon, System.Drawing" mimetype="application/x-microsoft.net.object.bytearray.base64">

<value>[base64 mime encoded string representing a byte array form of the .NET Framework object]</value>

<comment>This is a comment</comment>

</data>

There are any number of "resheader" rows that contain simple

name/value pairs.

Each data row contains a name, and value. The row also contains a

type or mimetype. Type corresponds to a .NET class that support

text/value conversion through the TypeConverter architecture.

Classes that don't support this are serialized and stored with the

mimetype set.

The mimetype is used for serialized objects, and tells the

ResXResourceReader how to depersist the object. This is currently not

extensible. For a given mimetype the value must be set accordingly:

Note - application/x-microsoft.net.object.binary.base64 is the format

that the ResXResourceWriter will generate, however the reader can

read any of the formats listed below.

mimetype: application/x-microsoft.net.object.binary.base64

value : The object must be serialized with

: System.Runtime.Serialization.Formatters.Binary.BinaryFormatter

: and then encoded with base64 encoding.

mimetype: application/x-microsoft.net.object.soap.base64

value : The object must be serialized with

: System.Runtime.Serialization.Formatters.Soap.SoapFormatter

: and then encoded with base64 encoding.

mimetype: application/x-microsoft.net.object.bytearray.base64

value : The object must be serialized into a byte array

: using a System.ComponentModel.TypeConverter

: and then encoded with base64 encoding.

-->

<xsd:schema id="root" xmlns="" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">

<xsd:import namespace="http://www.w3.org/XML/1998/namespace" />

<xsd:element name="root" msdata:IsDataSet="true">

<xsd:complexType>

<xsd:choice maxOccurs="unbounded">

<xsd:element name="metadata">

<xsd:complexType>

<xsd:sequence>

<xsd:element name="value" type="xsd:string" minOccurs="0" />

</xsd:sequence>

<xsd:attribute name="name" use="required" type="xsd:string" />

<xsd:attribute name="type" type="xsd:string" />

<xsd:attribute name="mimetype" type="xsd:string" />

<xsd:attribute ref="xml:space" />

</xsd:complexType>

</xsd:element>

<xsd:element name="assembly">

<xsd:complexType>

<xsd:attribute name="alias" type="xsd:string" />

<xsd:attribute name="name" type="xsd:string" />

</xsd:complexType>

</xsd:element>

<xsd:element name="data">

<xsd:complexType>

<xsd:sequence>

<xsd:element name="value" type="xsd:string" minOccurs="0" msdata:Ordinal="1" />

<xsd:element name="comment" type="xsd:string" minOccurs="0" msdata:Ordinal="2" />

</xsd:sequence>

<xsd:attribute name="name" type="xsd:string" use="required" msdata:Ordinal="1" />

<xsd:attribute name="type" type="xsd:string" msdata:Ordinal="3" />

<xsd:attribute name="mimetype" type="xsd:string" msdata:Ordinal="4" />

<xsd:attribute ref="xml:space" />

</xsd:complexType>

</xsd:element>

<xsd:element name="resheader">

<xsd:complexType>

<xsd:sequence>

<xsd:element name="value" type="xsd:string" minOccurs="0" msdata:Ordinal="1" />

</xsd:sequence>

<xsd:attribute name="name" type="xsd:string" use="required" />

</xsd:complexType>

</xsd:element>

</xsd:choice>

</xsd:complexType>

</xsd:element>

</xsd:schema>

<resheader name="resmimetype">

<value>text/microsoft-resx</value>

</resheader>

<resheader name="version">

<value>2.0</value>

</resheader>

<resheader name="reader">

<value>System.Resources.ResXResourceReader, System.Windows.Forms, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>

</resheader>

<resheader name="writer">

<value>System.Resources.ResXResourceWriter, System.Windows.Forms, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>

</resheader>

<data name="Taxi Rides" xml:space="preserve">

<value>Такси Возења</value>

</data>

<data name="Search" xml:space="preserve">

<value>Пребарување</value>

</data>

<data name="Page Size" xml:space="preserve">

<value>Големина на страница</value>

</data>

<data name="Total Records" xml:space="preserve">

<value>Вкупно записи</value>

</data>

<data name="Loading..." xml:space="preserve">

<value>Се вчитува...</value>

</data>

<data name="Previous" xml:space="preserve">

<value>Претходна</value>

</data>

<data name="Next" xml:space="preserve">

<value>Следна</value>

</data>

<data name="Page" xml:space="preserve">

<value>Страница</value>

</data>

<data name="of" xml:space="preserve">

<value>од</value>

</data>

<data name="Enter Page" xml:space="preserve">

<value>Внеси страница</value>

</data>

<data name="Go" xml:space="preserve">

<value>Оди</value>

</data>

<data name="Export to CSV" xml:space="preserve">

<value>Извези во CSV</value>

</data>

<data name="Export to Excel" xml:space="preserve">

<value>Извези во Excel</value>

</data>

<data name="Export to PDF" xml:space="preserve">

<value>Извези во PDF</value>

</data>

<data name="medallion" xml:space="preserve">

<value>Медалјон</value>

</data>

<data name="hashLicense" xml:space="preserve">

<value>Хеш лиценца</value>

</data>

<data name="pickupTime" xml:space="preserve">

<value>Време на подигнување</value>

</data>

<data name="dropOffTime" xml:space="preserve">

<value>Време на симнување</value>

</data>

<data name="duration" xml:space="preserve">

<value>Траење</value>

</data>

<data name="pLongitude" xml:space="preserve">

<value>Географска должина (поаѓање)</value>

</data>

<data name="pLatitude" xml:space="preserve">

<value>Географска ширина (поаѓање)</value>

</data>

<data name="dLongitude" xml:space="preserve">

<value>Географска должина (дестинација)</value>

</data>

<data name="dLatitude" xml:space="preserve">

<value>Географска ширина (дестинација)</value>

</data>

<data name="paymentType" xml:space="preserve">

<value>Тип на плаќање</value>

</data>

<data name="fareAmount" xml:space="preserve">

<value>Цена на возење</value>

</data>

<data name="surcharge" xml:space="preserve">

<value>Доплата</value>

</data>

<data name="tax" xml:space="preserve">

<value>Данок</value>

</data>

<data name="tipAmount" xml:space="preserve">

<value>Бакшиш</value>

</data>

<data name="tollsAmount" xml:space="preserve">

<value>Патарини</value>

</data>

<data name="totalAmount" xml:space="preserve">

<value>Вкупен износ</value>

</data>

<data name="AppTitle" xml:space="preserve">

<value>Преглед на такси возења</value>

</data>

<data name="distance" xml:space="preserve">

<value>Растојание</value>

</data>

</root>

SHAREDRESOURCE.RESX

<?xml version="1.0" encoding="utf-8"?>

<root>

<!--

Microsoft ResX Schema

Version 2.0

The primary goals of this format is to allow a simple XML format

that is mostly human readable. The generation and parsing of the

various data types are done through the TypeConverter classes

associated with the data types.

Example:

... ado.net/XML headers & schema ...

<resheader name="resmimetype">text/microsoft-resx</resheader>

<resheader name="version">2.0</resheader>

<resheader name="reader">System.Resources.ResXResourceReader, System.Windows.Forms, ...</resheader>

<resheader name="writer">System.Resources.ResXResourceWriter, System.Windows.Forms, ...</resheader>

<data name="Name1"><value>this is my long string</value><comment>this is a comment</comment></data>

<data name="Color1" type="System.Drawing.Color, System.Drawing">Blue</data>

<data name="Bitmap1" mimetype="application/x-microsoft.net.object.binary.base64">

<value>[base64 mime encoded serialized .NET Framework object]</value>

</data>

<data name="Icon1" type="System.Drawing.Icon, System.Drawing" mimetype="application/x-microsoft.net.object.bytearray.base64">

<value>[base64 mime encoded string representing a byte array form of the .NET Framework object]</value>

<comment>This is a comment</comment>

</data>

There are any number of "resheader" rows that contain simple

name/value pairs.

Each data row contains a name, and value. The row also contains a

type or mimetype. Type corresponds to a .NET class that support

text/value conversion through the TypeConverter architecture.

Classes that don't support this are serialized and stored with the

mimetype set.

The mimetype is used for serialized objects, and tells the

ResXResourceReader how to depersist the object. This is currently not

extensible. For a given mimetype the value must be set accordingly:

Note - application/x-microsoft.net.object.binary.base64 is the format

that the ResXResourceWriter will generate, however the reader can

read any of the formats listed below.

mimetype: application/x-microsoft.net.object.binary.base64

value : The object must be serialized with

: System.Runtime.Serialization.Formatters.Binary.BinaryFormatter

: and then encoded with base64 encoding.

mimetype: application/x-microsoft.net.object.soap.base64

value : The object must be serialized with

: System.Runtime.Serialization.Formatters.Soap.SoapFormatter

: and then encoded with base64 encoding.

mimetype: application/x-microsoft.net.object.bytearray.base64

value : The object must be serialized into a byte array

: using a System.ComponentModel.TypeConverter

: and then encoded with base64 encoding.

-->

<xsd:schema id="root" xmlns="" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">

<xsd:import namespace="http://www.w3.org/XML/1998/namespace" />

<xsd:element name="root" msdata:IsDataSet="true">

<xsd:complexType>

<xsd:choice maxOccurs="unbounded">

<xsd:element name="metadata">

<xsd:complexType>

<xsd:sequence>

<xsd:element name="value" type="xsd:string" minOccurs="0" />

</xsd:sequence>

<xsd:attribute name="name" use="required" type="xsd:string" />

<xsd:attribute name="type" type="xsd:string" />

<xsd:attribute name="mimetype" type="xsd:string" />

<xsd:attribute ref="xml:space" />

</xsd:complexType>

</xsd:element>

<xsd:element name="assembly">

<xsd:complexType>

<xsd:attribute name="alias" type="xsd:string" />

<xsd:attribute name="name" type="xsd:string" />

</xsd:complexType>

</xsd:element>

<xsd:element name="data">

<xsd:complexType>

<xsd:sequence>

<xsd:element name="value" type="xsd:string" minOccurs="0" msdata:Ordinal="1" />

<xsd:element name="comment" type="xsd:string" minOccurs="0" msdata:Ordinal="2" />

</xsd:sequence>

<xsd:attribute name="name" type="xsd:string" use="required" msdata:Ordinal="1" />

<xsd:attribute name="type" type="xsd:string" msdata:Ordinal="3" />

<xsd:attribute name="mimetype" type="xsd:string" msdata:Ordinal="4" />

<xsd:attribute ref="xml:space" />

</xsd:complexType>

</xsd:element>

<xsd:element name="resheader">

<xsd:complexType>

<xsd:sequence>

<xsd:element name="value" type="xsd:string" minOccurs="0" msdata:Ordinal="1" />

</xsd:sequence>

<xsd:attribute name="name" type="xsd:string" use="required" />

</xsd:complexType>

</xsd:element>

</xsd:choice>

</xsd:complexType>

</xsd:element>

</xsd:schema>

<resheader name="resmimetype">

<value>text/microsoft-resx</value>

</resheader>

<resheader name="version">

<value>2.0</value>

</resheader>

<resheader name="reader">

<value>System.Resources.ResXResourceReader, System.Windows.Forms, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>

</resheader>

<resheader name="writer">

<value>System.Resources.ResXResourceWriter, System.Windows.Forms, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>

</resheader>

<data name="Taxi Rides" xml:space="preserve">

<value>Taxi Rides</value>

</data>

<data name="Search" xml:space="preserve">

<value>Search</value>

</data>

<data name="Page Size" xml:space="preserve">

<value>Page Size</value>

</data>

<data name="Total Records" xml:space="preserve">

<value>Total Records</value>

</data>

<data name="Loading..." xml:space="preserve">

<value>Loading...</value>

</data>

<data name="Previous" xml:space="preserve">

<value>Previous</value>

</data>

<data name="Next" xml:space="preserve">

<value>Next</value>

</data>

<data name="Page" xml:space="preserve">

<value>Page</value>

</data>

<data name="of" xml:space="preserve">

<value>of</value>

</data>

<data name="Enter Page" xml:space="preserve">

<value>Enter Page</value>

</data>

<data name="Go" xml:space="preserve">

<value>Go</value>

</data>

<data name="Export to CSV" xml:space="preserve">

<value>Export to CSV</value>

</data>

<data name="Export to Excel" xml:space="preserve">

<value>Export to Excel</value>

</data>

<data name="Export to PDF" xml:space="preserve">

<value>Export to PDF</value>

</data>

<data name="medallion" xml:space="preserve">

<value>Medallion</value>

</data>

<data name="hashLicense" xml:space="preserve">

<value>Hash License</value>

</data>

<data name="pickupTime" xml:space="preserve">

<value>Pickup Time</value>

</data>

<data name="dropOffTime" xml:space="preserve">

<value>Drop-off Time</value>

</data>

<data name="duration" xml:space="preserve">

<value>Duration</value>

</data>

<data name="pLongitude" xml:space="preserve">

<value>Pickup Longitude</value>

</data>

<data name="pLatitude" xml:space="preserve">

<value>Pickup Latitude</value>

</data>

<data name="dLongitude" xml:space="preserve">

<value>Drop-off Longitude</value>

</data>

<data name="dLatitude" xml:space="preserve">

<value>Drop-off Latitude</value>

</data>

<data name="paymentType" xml:space="preserve">

<value>Payment Type</value>

</data>

<data name="fareAmount" xml:space="preserve">

<value>Fare Amount</value>

</data>

<data name="surcharge" xml:space="preserve">

<value>Surcharge</value>

</data>

<data name="tax" xml:space="preserve">

<value>Tax</value>

</data>

<data name="tipAmount" xml:space="preserve">

<value>Tip Amount</value>

</data>

<data name="tollsAmount" xml:space="preserve">

<value>Tolls Amount</value>

</data>

<data name="totalAmount" xml:space="preserve">

<value>Total Amount</value>

</data>

<data name="AppTitle" xml:space="preserve">

<value>Taxi Ride Viewer</value>

</data>

<data name="distance" xml:space="preserve">

<value>Distance</value>

</data>

</root>

SHAREDRESOURCE.SR.RESX

<?xml version="1.0" encoding="utf-8"?>

<root>

<!--

Microsoft ResX Schema

Version 2.0

The primary goals of this format is to allow a simple XML format

that is mostly human readable. The generation and parsing of the

various data types are done through the TypeConverter classes

associated with the data types.

Example:

... ado.net/XML headers & schema ...

<resheader name="resmimetype">text/microsoft-resx</resheader>

<resheader name="version">2.0</resheader>

<resheader name="reader">System.Resources.ResXResourceReader, System.Windows.Forms, ...</resheader>

<resheader name="writer">System.Resources.ResXResourceWriter, System.Windows.Forms, ...</resheader>

<data name="Name1"><value>this is my long string</value><comment>this is a comment</comment></data>

<data name="Color1" type="System.Drawing.Color, System.Drawing">Blue</data>

<data name="Bitmap1" mimetype="application/x-microsoft.net.object.binary.base64">

<value>[base64 mime encoded serialized .NET Framework object]</value>

</data>

<data name="Icon1" type="System.Drawing.Icon, System.Drawing" mimetype="application/x-microsoft.net.object.bytearray.base64">

<value>[base64 mime encoded string representing a byte array form of the .NET Framework object]</value>

<comment>This is a comment</comment>

</data>

There are any number of "resheader" rows that contain simple

name/value pairs.

Each data row contains a name, and value. The row also contains a

type or mimetype. Type corresponds to a .NET class that support

text/value conversion through the TypeConverter architecture.

Classes that don't support this are serialized and stored with the

mimetype set.

The mimetype is used for serialized objects, and tells the

ResXResourceReader how to depersist the object. This is currently not

extensible. For a given mimetype the value must be set accordingly:

Note - application/x-microsoft.net.object.binary.base64 is the format

that the ResXResourceWriter will generate, however the reader can

read any of the formats listed below.

mimetype: application/x-microsoft.net.object.binary.base64

value : The object must be serialized with

: System.Runtime.Serialization.Formatters.Binary.BinaryFormatter

: and then encoded with base64 encoding.

mimetype: application/x-microsoft.net.object.soap.base64

value : The object must be serialized with

: System.Runtime.Serialization.Formatters.Soap.SoapFormatter

: and then encoded with base64 encoding.

mimetype: application/x-microsoft.net.object.bytearray.base64

value : The object must be serialized into a byte array

: using a System.ComponentModel.TypeConverter

: and then encoded with base64 encoding.

-->

<xsd:schema id="root" xmlns="" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">

<xsd:import namespace="http://www.w3.org/XML/1998/namespace" />

<xsd:element name="root" msdata:IsDataSet="true">

<xsd:complexType>

<xsd:choice maxOccurs="unbounded">

<xsd:element name="metadata">

<xsd:complexType>

<xsd:sequence>

<xsd:element name="value" type="xsd:string" minOccurs="0" />

</xsd:sequence>

<xsd:attribute name="name" use="required" type="xsd:string" />

<xsd:attribute name="type" type="xsd:string" />

<xsd:attribute name="mimetype" type="xsd:string" />

<xsd:attribute ref="xml:space" />

</xsd:complexType>

</xsd:element>

<xsd:element name="assembly">

<xsd:complexType>

<xsd:attribute name="alias" type="xsd:string" />

<xsd:attribute name="name" type="xsd:string" />

</xsd:complexType>

</xsd:element>

<xsd:element name="data">

<xsd:complexType>

<xsd:sequence>

<xsd:element name="value" type="xsd:string" minOccurs="0" msdata:Ordinal="1" />

<xsd:element name="comment" type="xsd:string" minOccurs="0" msdata:Ordinal="2" />

</xsd:sequence>

<xsd:attribute name="name" type="xsd:string" use="required" msdata:Ordinal="1" />

<xsd:attribute name="type" type="xsd:string" msdata:Ordinal="3" />

<xsd:attribute name="mimetype" type="xsd:string" msdata:Ordinal="4" />

<xsd:attribute ref="xml:space" />

</xsd:complexType>

</xsd:element>

<xsd:element name="resheader">

<xsd:complexType>

<xsd:sequence>

<xsd:element name="value" type="xsd:string" minOccurs="0" msdata:Ordinal="1" />

</xsd:sequence>

<xsd:attribute name="name" type="xsd:string" use="required" />

</xsd:complexType>

</xsd:element>

</xsd:choice>

</xsd:complexType>

</xsd:element>

</xsd:schema>

<resheader name="resmimetype">

<value>text/microsoft-resx</value>

</resheader>

<resheader name="version">

<value>2.0</value>

</resheader>

<resheader name="reader">

<value>System.Resources.ResXResourceReader, System.Windows.Forms, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>

</resheader>

<resheader name="writer">

<value>System.Resources.ResXResourceWriter, System.Windows.Forms, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>

</resheader>

<data name="Taxi Rides" xml:space="preserve">

<value>Такси Вожње</value>

</data>

<data name="Search" xml:space="preserve">

<value>Претрага</value>

</data>

<data name="Page Size" xml:space="preserve">

<value>Величина стране</value>

</data>

<data name="Total Records" xml:space="preserve">

<value>Укупно записа</value>

</data>

<data name="Loading..." xml:space="preserve">

<value>Учитавање...</value>

</data>

<data name="Previous" xml:space="preserve">

<value>Претходна</value>

</data>

<data name="Next" xml:space="preserve">

<value>Следећа</value>

</data>

<data name="Page" xml:space="preserve">

<value>Страна</value>

</data>

<data name="of" xml:space="preserve">

<value>од</value>

</data>

<data name="Enter Page" xml:space="preserve">

<value>Унеси страну</value>

</data>

<data name="Go" xml:space="preserve">

<value>Иди</value>

</data>

<data name="Export to CSV" xml:space="preserve">

<value>Извези у CSV</value>

</data>

<data name="Export to Excel" xml:space="preserve">

<value>Извези у Excel</value>

</data>

<data name="Export to PDF" xml:space="preserve">

<value>Извези у PDF</value>

</data>

<data name="medallion" xml:space="preserve">

<value>Медаљон</value>

</data>

<data name="hashLicense" xml:space="preserve">

<value>Хеш лиценца</value>

</data>

<data name="pickupTime" xml:space="preserve">

<value>Време преузимања</value>

</data>

<data name="dropOffTime" xml:space="preserve">

<value>Време остављања</value>

</data>

<data name="duration" xml:space="preserve">

<value>Трајање</value>

</data>

<data name="pLongitude" xml:space="preserve">

<value>Дужина (полазак)</value>

</data>

<data name="pLatitude" xml:space="preserve">

<value>Ширина (полазак)</value>

</data>

<data name="dLongitude" xml:space="preserve">

<value>Дужина (одредиште)</value>

</data>

<data name="dLatitude" xml:space="preserve">

<value>Ширина (одредиште)</value>

</data>

<data name="paymentType" xml:space="preserve">

<value>Начин плаћања</value>

</data>

<data name="fareAmount" xml:space="preserve">

<value>Износ вожње</value>

</data>

<data name="surcharge" xml:space="preserve">

<value>Доплата</value>

</data>

<data name="tax" xml:space="preserve">

<value>Порез</value>

</data>

<data name="tipAmount" xml:space="preserve">

<value>Напојница</value>

</data>

<data name="tollsAmount" xml:space="preserve">

<value>Путарина</value>

</data>

<data name="totalAmount" xml:space="preserve">

<value>Укупан износ</value>

</data>

<data name="AppTitle" xml:space="preserve">

<value>Преглед такси вожњи</value>

</data>

<data name="distance" xml:space="preserve">

<value>Растојање</value>

</data>

</root>