Desarrollador de Aplicaciones Web
Programación Web III

Departamento de Ingeniería e Investigaciones Tecnológicas

Partial Views – Layout - Validations

Ing. Mariano Juiz
Ing. Matias Paz Wasiuchnik
Ing. Pablo Nicolás Sanchez

Vistas

Razor procesa Vistas

Una vista es un archivo de texto que Razor reconoce que tiene trabajar en el

La extensión indica en que lenguaje Razor tiene que procesar el código que encuentra

Un Layout es una vista

Una Vista Parcial también es una Vista

Vistas

Si todo es una vista, ¿Cúal es la diferencia?

Layout: Es una vista usada como Layout

View: Es una vista que usa un Layout

Partial View: Es una vista que NO usa Layout

Layout Template

Puede contener:

- HTML
- Contenido Dinámico de Servidor

Utilizaremos el método “@RenderBody()” (helper) para especificar el contenido a cambiar dependiendo la URL especificada

La extensión también es “cshtml”

Sintáxis

Layout

```
<html>
        <head>
                <title>Title<title>
        </head>
        <body>
                @RenderSection("Menu")
                @RenderBody()
        </body>
</html>
```

Vista

```
@{
        Layout="~/Views/Shared/_Layout.cshtml";
}
@section Menu {
        <ul id="pageMenu">
                <li>Item 1</li>
                <li>Item 2</li>
        </ul>
}
```

Vistas

View Template

Puede contener:

- HTML
- Contenido Dinámico de Servidor

Utilizaremos la siguiente instrucción para especificar el Template a utilizar:

```
@{
        Layout = “NOMBRE_TEMPLATE.cshtml”;
   }
```

Vistas

Partial View

Renderizan una porción de página en otra Vista “padre”

Reutilizan porciones de una vista

Los Helpers

Tienen extensión “.cshtml” (C#) y “.vbhtml” (VB.NET)

Usan su propio “ViewDataDictionary” (como si fuera una copia local) aunque pueden usar el “ViewDataDictionary” de su página padre

Vistas

Partial View

@Html.RenderPartial() escribe directamente la vista como respuesta (output stream). No retorna nada (void). Es más rápido.

@Html.Partial() retorna el HTML de la vista como un STRING (MvcHtmlString). Se puede almacenar en una variable y ser manipulado (mediante Razor por ejemplo)

Vistas

Partial View

Helper Name	Description
@Html.RenderPartial(string partialViewName)	Renderiza el contenido del PartialView a la vista involucrada (como un HTML string codificado).
@Html.RenderPartial(string partialViewName,object model)	Renderiza el contenido del PartialView a la vista involucrada (como un HTML string codificado). Se envía el Modelo como parámetro a la PartialView
@Html.RenderPartial(string partialViewName, ViewDataDictionary viewData)	Renderiza el contenido del PartialView a la vista involucrada (como un HTML string codificado). Se envía el objeto ViewDataDictonary como parámetro a la PartialView
@Html.RenderPartial(string partialViewName,object model, ViewDataDictionary viewData)	Renderiza el contenido del PartialView a la vista involucrada (como un HTML string codificado). Se envían el Modelo y el objeto ViewDataDictonary como parámetro a la PartialView

Vistas

Partial View

Helper Name	Description
@Html.Partial(string partialViewName)	Renderiza el contenido del PartialView a la vista involucrada.
@Html.Partial(string partialViewName,object model)	Renderiza el contenido del PartialView a la vista involucrada. Se envía el Modelo como parámetro a la PartialView
@Html.Partial(string partialViewName, ViewDataDictionary viewData)	Renderiza el contenido del PartialView a la vista involucrada. Se envía el objeto ViewDataDictonary como parámetro a la PartialView
@Html.Partial(string partialViewName,object model, ViewDataDictionary viewData)	Renderiza el contenido del PartialView a la vista involucrada. Se envían el Modelo y el objeto ViewDataDictonary como parámetro a la PartialView

Data Annotations

Validación de Datos Ingresados

La manera más común de hacerlo es usando **DataAnnotations**. Simplemente debemos completar la propiedad que deseemos validar con:

- **Required**: Dato obligatorio
- **Range**: Se validará un Rango
- **StringLength**: Para limitar el número de caracteres de un campo texto. Puede tener además un tamaño mínimo
- **Compare**: Para que dos campos tengan el mismo valor
- **RegularExpression**: Para validar contra una expresión regular
- **CustomValidator**: Se utiliza un método custom para validar

Debemos incluir *System.ComponentModel.DataAnnotations*, *System.Web.Mvc y System.ComponentModel*

Además, podemos usar

Bind: Indica si un atributo se utilizará en un Bind de la Vista con su Modelo (a nivel clase)

Display: Indica un “alias” del atributo (se mostrará en la vista)

ScaffoldColumn: Indica si el atributo se generará ante la creación de una vista mediante Scaffolding

Data Annotations

Ejemplo:

```
public class DemoModel
   {
         [Required(ErrorMessage = "¡Aquí todo el mundo tiene un nombre!")]
         public string Nombre { get; set; }
         public int Edad { get; set; }
   }
```

En la Vista, para mostrar el mensaje de error, usamos el Helper **Html.ValidationMessageFor**

Ejemplo:

```
@Html.LabelFor(x=>x.Nombre) @Html.TextBoxFor(x=>x.Nombre) <br />
@Html.ValidationMessageFor(x => x.Nombre)
```

Data Annotations

Si los datos ingresados en el formularios no son correctos, el ModelState no es válido, por lo tanto se devuelve la vista que contiene el formulario

Los helpers mostrarán los errores (en color rojo con la CSS por defecto).

```
[HttpPost]
    public ActionResult GrabarDatos(DemoModel data)
    {
        if (!ModelState.IsValid)     {
                    return View(data);
         }
         else {
                    return View();
         }
     }
```

Data Annotations

Jquery Unobtrusive

Para usar las mismas validaciones del lado del cliente se realiza simplemente agregando:

- jquery.validate.min.js
- jquery.validate.unobtrusive.min.js

Ejemplo:

<script src="@Url.Content("~/Scripts/jquery.validate.min.js")" type="text/javascript"></script>

<script src="@Url.Content("~/Scripts/jquery.validate.unobtrusive.min.js")" type="text/javascript"></script>

Y agregar en el web.config

```
<configuration>
    <appSettings>
            <add key="UnobtrusiveJavaScriptEnabled" value=“true"/>
    </appSettings>
</configuration>
```

Desarrollador de Aplicaciones Web
Programación Web III

Departamento de Ingeniería e Investigaciones Tecnológicas

Muchas gracias

Ing. Mariano Juiz
Ing. Matias Paz Wasiuchnik
Ing. Pablo Nicolás Sanchez