****

**PROYECTO WEB CON BLAZOR - REZOR**

**FORMADOR: JHON JAIRO OROZCO DÁVILA Ing.**

|  |  |
| --- | --- |
|  | **PROYECTO WEB - EJEMPLO** |
|  | ¿Qué es Blazor Wasm?  Blazor es un proyecto desarrollado por Microsoft creado para permitir crear **Single Page Applications** (SPAs) únicamente usando como lenguajes de programación C# y Razor Pages, haciendo nula la necesidad de programar en Javascript o frameworks derivados.  Es un nuevo framework open-source diseñado por Microsoft, que viene incluido de forma oficial en la [versión de .NET Core 3.0](https://docs.microsoft.com/es-es/aspnet/core/?view=aspnetcore-3.0)  Este framework se ejecuta mediante estándares web, por lo que no será necesario utilizar ningún plugin ni realizar ninguna transpilación para que funcione. Esto es así porque todo el código se ejecuta en el navegador mediante un . NET Runtime implementado en WebAssembly |
|  | Al finalizar, vemos las plantillas con prefijo Blazor que sirven para crear aplicaciones tanto del lado servidor (blazorserver), como lado cliente (blazorwasm). Más adelante explicaremos las diferencias entre cada una de ellas y escenarios de uso para decidir cual se adapta mejor a las necesidades de nuestro proyecto. |
|  | Instalación de las herramientas de Blazor  Antes de empezar, necesitamos en nuestro entorno de desarrollo la última versión de .NET Core 3.0 que podéis descargar aquí: [.NET Core 3.0 SDK](https://dotnet.microsoft.com/download/dotnet-core/3.0), y las plantillas Blazor que se instalan con el siguiente comando desde la CLI:  dotnet new -i Microsoft.AspNetCore.Blazor.Templates::3.0.0-preview9  Al finalizar, vemos las plantillas con prefijo Blazor que sirven para crear aplicaciones tanto del lado servidor (blazorserver), como lado cliente (blazorwasm). Más adelante explicaremos las diferencias entre cada una de ellas y escenarios de uso para decidir cual se adapta mejor a las necesidades de nuestro proyecto. |
|  |  |
|  | En la lista vemos las 2 plantillas disponibles para crear aplicaciones Razor:  1) Blazor Server App (blazorserver): Esta plantilla genera una aplicación Blazor que se comunica entre el navegador del usuario y el servidor .Net Core en back-end.  2) Blazor WebAssembly (blazorwasm): Esta plantilla crea una aplicación cliente Blazor, que funciona totalmente en el navegador del usuario mediante una tecnología llamada WebAssembly.  Para desarrollar con Blazor, podemos elegir Visual Studio Code que no requiere configuración adicional o Visual Studio 2019 que necesita añadir las extensiones desde el MarketPlace. |
|  |  |

|  |  |
| --- | --- |
|  | **TALLER APLICACIÓN – BLAZOR - RAZOR** |
|  | Abrir una ventana de comandos CMD como Administrador |
|  | Ubicarse en la raíz del disco duro C:\ |
|  | Crear la carpeta o directorio **proyectos\_ciclo3** |
|  | Acceder a la carpeta creada **c:\proyectos\_ciclo3** |
|  | Empezamos creando y ejecutando una aplicación Blazor basada en la plantilla *blazorwasm* para que funcione con WebAssembly y la voy a llamar **proyecto58**.  Desde la consola escribimos estos comandos de .NET Core: |
|  | **dotnet new blazorwasm -o proyecto58** |
|  |  |
|  | Por cada proyecto se crea automáticamente un nuevo directorio; ingresar al directorio donde se ha creado la solución **proyectos58** |
|  | Dentro de la solución, se crean automáticamente otros proyectos, unos fijos de configuración y otros que se modificarán para aplicarla a una solución en particular.  Ver árbol de carpetas y subcarpetas creadas de la solución con **tree** |
|  |  |
|  | Compilar la Solución **dotnet build**  Verificar que NO existan errores |
|  |  |
|  | Ejecutar la Solución: se crea una URL, para navegar desde un Navegador  **dotnet run** |
|  |  |
|  | En un navegador ejecutar [**https://localhost:5001/**](https://localhost:5001/)  **Quitar la advertencia de seguridad si la pide por primera vez** |
|  |  |
|  | Felicitaciones, ha creado su primera solución en C# con el FrameWork Blazor y Razor.  Esto nos indica que ya tenemos corriendo la aplicación Blazor, veremos el **Hello World!** que tendrá un aspecto como esta imagen. |
|  |  |
|  | Matar o abortar el proceso para realizar nuevos cambios. **Ctrl + C** |
|  |  |

|  |  |
| --- | --- |
|  | **Revisando el código fuente con Visual Studio Code** |
|  | Se utiliza **VSCode** para revisar el código fuente generado y la estructura de la aplicación Blazor que se acaba de crear. A simple vista, se trata de una típica aplicación .NET Core con algunos ajustes.  Abrir la solución en **Visual Studio Code** previamente Instalado, estando en el directorio del proyecto digitar code espacio punto  C:\proyectos\_ciclo3\proyecto58\code . |
|  | Revisar el panel izquierdo con los componentes de la Solución (ver gráfica) y compararla con la solución web obtenida. |
|  |  |
|  | En la clase Program.cs vemos el método Main, punto de entrada de la aplicación, donde se crea el HostBuilder usando la llamada a UseBlazorStartup. |
|  |  |
|  | En la clase **Startup.cs** encontramos los métodos para registrar y configurar nuestros servicios en .NET Core donde se añade el componente App de la aplicación de Razor. |
|  |  |
|  | En el archivo **proyecto58.csproj** tenemos las referencias a librerías Razor: |
|  |  |
|  | En la clase **App.razor** encontramos la definición para las rutas: |
|  |  |
|  | La clase **\_Imports.razor** contiene las referencias a librerías para importar los componentes y JSInterop: |
|  |  |
|  | En la carpeta **wwwroot**, donde van los archivos estáticos, encontramos el **index.html** con el código **HTML** inicial para cargar nuestra web cuando accedemos desde el navegador.  Bien en el link: **"\_framework/blazor.webassembly.js",** esto nos ofrecerá la interoperabilidad de JavaScript con WebAssebly. |
|  |  |
|  | En las carpetas **Pages** y **Shared** encontramos, los archivos con la nueva extensión **.razor** que son los componentes donde está el código HTML5, los layouts de las páginas, el sistema de rutas y código de **C#** que se debe ejecutar.  Los archivos de estas carpetas se corresponden con el menú de opciones de nuestra web utilizando el sistema de rutas que se define en la directiva **@page** para cada una de las páginas, y en la directiva **@code** esta el código C# que se ejecutará: |
|  |  |
|  | Si seleccionamos **Counter.razor**, vemos por una lado el código de la página (@page), por otro el código C# (@code) y que existe un botón con evento onclick, apuntando al método **IncrementCount**. Este botón es HTML y su función de onclick está invocando al método escrito en C#.  Qué cambiaría para iniciar el incremento en 100?, compilar de nuevo para ver cómo funciona.  Qué cambiaría por ejemplo para incrementar de 5 en 5? compilar de nuevo para ver cómo funciona. |

|  |  |
| --- | --- |
|  | **Revisando la ejecución del código en el navegador** |
|  | Al usar el código de forma correcta, de acuerdo a la especificación de HTML o CSS, nos aseguramos que nuestras páginas se vean correctamente y funcionen en el navegador de cualquier PC, Laptop o de un dispositivo portable. Quizás al probarlas en nuestro equipo pensamos que todo funciona bien, pero en ocasiones no es así. La gran mayoría de las páginas que están publicadas en diversos sitios, contienen errores en mayor o menor medida y no obstante funcionan. Al usar el código de forma correcta estamos indicando limpieza, confiabilidad, compatibilidad y apego a los estándares establecidos.  Para revisar realmente el funcionamiento de Blazor dentro del navegador con WebAssembly;   * Con la aplicación en marcha(**dotnet watch run**), en Vs Code * Al ejecutar la aplicación se ejecuta la clase **Program.cs** |
|  |  |
|  | Navegar la página con Hacer click en Home, para evaluar el archivo **index**.**razor**  En un navegador con **F12** activamos las herramientas de desarrollo en nuestro browser favorito y observamos el código fuente de la página index.html que nos muestra lo siguiente. |
|  |  |
|  |  |

|  |  |
| --- | --- |
|  | **Taller de Aplicación** |
|  | Realizar los cambios necesarios, para:   * La primera tabla a mostrar es la del 5 * Al presionar el botón SIGUIENTE, se cambia a la tabla siguiente 5,6,7,8…… |
|  |  |
|  | @\* File: Pages/Multiplication.razor \*@  @page "/tablas"  <h3>Multiplication</h3> |
|  | Mejorar el mismo formulario, para que desde un combo, se pueda seleccionar la tabla respectiva del 2 al 10 y se vean los resultados automáticamente |
|  |  |
|  | @\* File: Pages/Multiplication.razor \*@  @page "/tablas"  <h3>Multiplication</h3> |