educación IT
CODE YOUR FUTURE

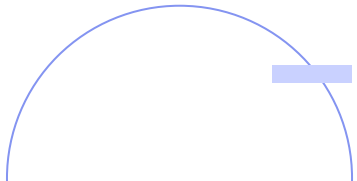
Programación Web .NET Core

Módulo 9

Registro de aplicaciones MVC

Registro de Excepciones

Las excepciones a las que te enfrentas durante el desarrollo pueden ser investigadas y depuradas utilizando las **herramientas de depuración de Visual Studio**. En **ASP.NET Core**, hay una robusta **biblioteca de registro integrada**. En lugar de trabajar con una forma específica de registro, utilizando la biblioteca de registro, puedes configurar **varios proveedores de registros, incluidos los integrados y varios de terceros compatibles**.



Procedimiento

Para agregar el registro, deberás agregar una llamada al método **`ConfigureLogging`** como parte de la canalización **`CreateDefaultBuilder`**.

El método `ConfigureLogging` expone dos parámetros, uno del tipo denominado **`WebHostBuilderContext`**, que se puede utilizar para recuperar el **`IHostingEnvironment`**, y un parámetro **`ILoggingBuilder`**, que se utiliza para configurar los proveedores y las configuraciones del registrador.

En el método `ConfigureLogging`, deberás agregar el proveedor de registro, que habilitará el registro. Puedes iniciar sesión en más de un proveedor al mismo tiempo.

ASP.NET Core tiene varios proveedores integrados, que incluyen:

- **`Console`**: Registra el mensaje en la ventana de la consola de la aplicación.
- **`Debug`**: Registra el mensaje en la ventana de depuración de la aplicación.
- **`EventSource`**: Utiliza la API de seguimiento de eventos para almacenar el evento. El comportamiento difiere entre los sistemas operativos, y puede que no funcione en todas las plataformas.
- **`EventLog`**: Registra el mensaje en el registro de eventos de Windows. Es exclusivo de Windows.
- **`TraceSource`**: Crea una fuente de seguimiento que se puede escuchar con una variedad de detectores de seguimiento.
- **`AzureAppServices`**: Crea registros que se integran con la plataforma Azure.

El siguiente código es un ejemplo del uso del método **`ConfigureLogging`**:

El método ConfigureLogging

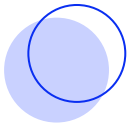
```
WebHost.CreateDefaultBuilder(args)
.ConfigureLogging((hostingContext, logging) =>
{
if (hostingContext.HostingEnvironment.IsDevelopment())
{
logging.AddConsole();
}
})
.UseStartup<Startup>();
```

Método de registro

Una vez configurado el registro, puedes agregar registros a componentes individuales inyectando un registrador en el constructor para el componente. El registrador que inyectes debe ser del tipo **`ILogger<* className *>`** . El nombre de la clase se usa para crear un identificador para el archivo desde el cual se llama al registrador. El identificador contiene el espacio de nombres completo para la clase y facilita que los desarrolladores lo averigüen. El código de la siguiente pantalla, es un ejemplo de un controlador que escribe en un archivo de registro.

Registro desde el controlador

```
publicclassHomeController :Controller
{
privateILogger _logger;
publicHomeController(ILogger<HomeController>logger)
{
_logger = logger;
}
publicIActionResultIndex()
{
_logger.LogInformation("Addinganentry to thelogger.");
return Content("Resultfromcontroller");
}
}
```



El siguiente código es un ejemplo de cómo usar el registro en el método **Configure**:

Registro desde el método de configuración

```
publicvoid Configure(IApplicationBuilder app, IHostingEnvironmentenv,
ILogger<Startup>logger)
{
logger.LogCritical("CriticalMessagefromlogger");
if (env.IsDevelopment())
{
app.UseDeveloperExceptionPage();
}
app.UseMvcWithDefaultRoute();
app.Run(async (context) =>
{
awaitcontext.Response.WriteAsync("HelloWorld!");
});
}
```

educación IT
CODE YOUR FUTURE

¡Sigamos trabajando!