Ejercicio A.3: Interoperabilidad .NET en entornos on-premise

En desarrollos para instalaciones on-premise, se puede usar variables .NET y llamar a sus métodos, miembros y constructores.

Datos de conexión

- Entorno de desarrollo:
  - Servidor: localhost\NAVDEMO
  - BBDD: BCCU0_AppDevelopment
  - Autenticación: Windows

Indicaciones para el ejercicio

En vscode, vamos a crear una extensión para una instalación on-premise, en la que vamos a usar variables .NET.

Documentación:
https://docs.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/devenv-get-started-call-dotnet-from-al

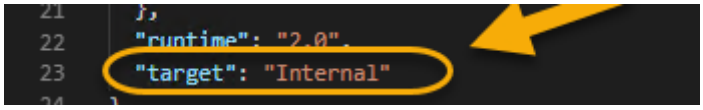
Indicaciones paso a paso

Sigue los siguientes pasos en el entorno de desarrollo C/SIDE:

1. En vscode, crea un nuevo proyecto y descarga los símbolos.
   1.1. Pulsa F1 para abrir la paleta de comandos.
   1.2. Ejecuta el comando AL: Go!
   Nombre del proyecto: DotNetOnPremise
   1.3. Modifica el fichero launch.json

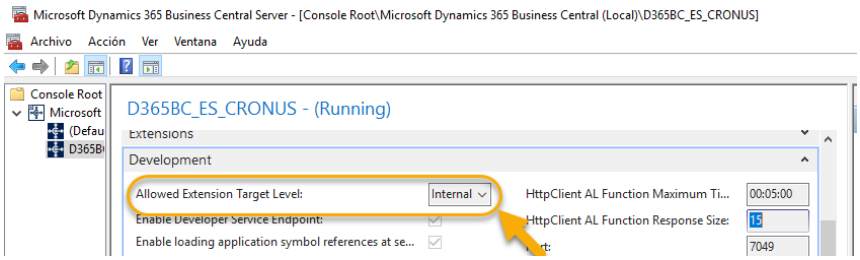
```
"serverInstance": " BCCU0_AppDevelopment ",
"authentication": "Windows",
```

   1.4. Modifica el fichero app.json, para indicar que el target es Internal.



   1.5. Ejecuta el comando AL: Download symbols.

2. Abre el servicio. Asegúrate que el Allowed Extension Target Level, está en internal.
   Si no es así, modifica el valor y reinicia el servicio.



3. Abre el fichero HelloWorld.al
   Al inicio del fichero, declara el paquete dotnet que vas a utilizar.

HelloWorld.al

```
dotnet
{
    assembly(mscorlib)
    {
        type(System.DateTime; MyDateTime) { }
    }
}
```

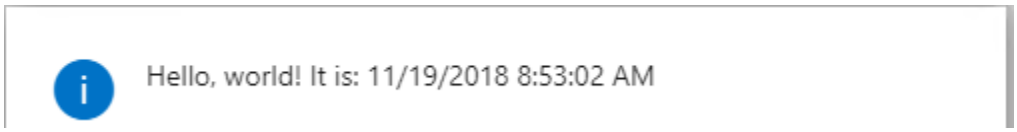
4. En el trigger OnOpenPage, crea una variable local de tipo DotNet.

```
trigger OnOpenPage();
var
    now: DotNet MyDateTime;
```

5. Modifica el Message, para mostrar la hora del mensaje

```
pageextension 50100 CustomerListExt extends "Customer List"
{
    trigger OnOpenPage();
    var
        now: DotNet MyDateTime;
    begin
        now := now.UtcNow();
        Message('Hello, world! It is: ' + now.ToString());
    end;
}
```

6. En el cliente web, abre la lista de clientes. Debe aparecer el siguiente mensaje:



Por defecto, el compilador sólo conoce la ubicación de la librería mscorlib. Para poder utilizar otras librerías, debemos indicarle al compilador la ruta en la que se encuentran, utilizando el parámetro de configuración "al.assemblyProbingPaths"

1. Añade una referencia a la librería System en el archivo dotnet.al

```
assembly(System)
{
}
```

2. El sistema nos dará el siguiente error:

```
.NET Assembly System
An assembly named 'System, PublicKeyToken=null' could not be found in the assembly probing paths 'c:\ScaleUp\Extensiones\DotNetOnPremise\./.netpackages' AL(AL0451)
```

3. Abre la configuración del espacio de trabajo, el archivo settings.json

4. Introduce la siguiente configuración

settings.json ●

Unsaved changes (cannot determine recent change or authors)

```
{
    "al.assemblyProbingPaths": [
        "./.netpackages",
        "C:/Windows/assembly/",
        "C:/Program Files/Microsoft Dynamics 365 Business Central/130/Service/Add-ins/"
    ]
}
```

Ahora podemos utilizar cualquiera de las librerías que se encuentran en estas rutas.

1. Declara el tipo System.Timers.Timer

```
assembly(System)
{
    type(System.Timers.Timer; MyTimer) { }
}
```

2. En el archivo HelloWorld.al, declara la siguiente variable global

```
var
    [WithEvents]
    0 references
    timer: DotNet MyTimer;
```

3. En el trigger OnOpenPage, inicializa el Timer

```
trigger OnOpenPage();
begin
    now := now.UtcNow();


    timer := timer.Timer(2000);
    timer.AutoReset := true;
    timer.Enabled := true;
    timer.Start();
```

4. Crea el trigger timer::Elapsed

```
trigger timer::Elapsed(sender: Variant; e: DotNet "System.Timers.ElapsedEventArgs")
var
    myInt: Integer;
begin

end;
```

5. Mueve la definición del tipo System.Timers.ElapsedEventArgs al archivo dotnet.al y utiliza el alias en el trigger timer::Elapsed

6. Escribe el siguiente código

```
trigger timer::Elapsed(sender: Variant; e: DotNet MyElapsedEventArgs)
begin
    Message('OnOpenPage: ' + now.ToString() +
            '\Timer:      ' + e.SignalTime().ToString());
    timer.Stop();
end;
```

7. Publica la extension y abre la lista de clientes para probar el desarrollo.

Aunque es posible utilizar Interoperabilidad .Net en entornos on-premise, los MVPs y expertos de la comunidad no lo recomiendan.

Es su lugar se recomienda:

- Usar codeunits en C/SIDE que actúen como interficie
- Usar Azure functions o WebServices
- Usar Javascript
  https://community.dynamics.com/nav/b/conceptsindailynavisionwork/archive/2018/11/23/javascript-and-al-the-regular-expressions-example