## Ejercicio A.15: Testing Avanzado – Gestión de elementos de UI

Cuando ejecutamos Tests, estos tienen que poder ejecutarse sin ninguna intervención del usuario. Para ello, tenemos que capturar todos los elementos de interfaz de usuario que aparezcan a lo largo de la ejecución del código.

En ejercicios previos vimos como capturar un ERROR, pero hay más elementos de UI que pueden aparecer y que debemos gestionar:

- Message
- Confirm
- StrMenu
- Page
- Report
- Etc.

Si nos aparece un Confirm, por ejemplo, el usuario podrá contestar Sí o No, y el código se tendrá que comportar de forma distinta en función de esa respuesta. Para hacer un test completo, deberíamos probar ambos casos.

## Datos de conexión

Para este desarrollo, hay que utilizar la siguiente base de datos:

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

## Indicaciones para el ejercicio

Realiza las siguientes acciones:

1. Añade un mensaje de confirmación al trigger OnValidate del campo “Default Ship-To Address”
2. Modifica los tests escritos anteriormente para gestionar este elemento de UI

## Indicaciones paso a paso

Sigue los siguientes pasos en Visual Studio Code:

1. Añade un mensaje de confirmación al trigger OnValidate del campo "Default Ship-To Address"

```
trigger OnValidate();
var
    ConfirmTxt: TextConst ENU = 'Do you want to update the Ship-To Code on Sales Documents?';
begin
    if Confirm(ConfirmTxt,true) then
    begin
        UpdateShipToCodeOnSalesDocuments();
    end;
end;
```

2. Crea la función UpdateShipToCodeOnSalesDocuments

```
local procedure UpdateShipToCodeOnSalesDocuments()
var
    SalesHeader : Record "Sales Header";
begin
    SalesHeader.SetRange("Sell-to Customer No.","No.");
    SalesHeader.SetRange(Status,SalesHeader.Status::Open);
    if SalesHeader.FindSet then
    repeat
        SalesHeader.Validate("Ship-to Code","CLIP Default Ship-To Address");
        SalesHeader.Modify();
    until SalesHeader.Next = 0;
end;
```

3. Crea una función de test con el siguiente escenario: `Comprobar que al responder que si al confirm del validate del campo, los documentos de venta se actualizan adecuadamente`

```
[Test]
procedure TestFunctionalityOnConfirmValidate()
var
    Customer : Record Customer;
    ShipToAddress : Record "Ship-to Address";
    SalesHeader1 : Record "Sales Header";
    SalesHeader2 : Record "Sales Header";
begin
    // [Scenario] Comprobar que al responder que si al confirm del validate del campo, los documentos de
venta se actualizan adecuadamente

    // [Given] Un nuevo cliente con 2 documentos de venta: uno lanzado y uno abierto
    LibrarySales.CreateCustomer(Customer);
    LibrarySales.CreateShipToAddress(ShipToAddress,Customer."No.");
```

```
    LibrarySales.CreateSalesHeader(SalesHeader1,SalesHeader1."Document Type"::Order,Customer."No.");
    SalesHeader1.Status := SalesHeader1.Status::Open;
    SalesHeader1.Modify();
    LibrarySales.CreateSalesHeader(SalesHeader2,SalesHeader2."Document Type"::Order,Customer."No.");
    SalesHeader2.Status := SalesHeader2.Status::Released;
    SalesHeader2.Modify();

    // [When] Se valida el campo CLIP Default Ship-To Address
    Customer.Validate("CLIP Default Ship-To Address", ShipToAddress.Code);

    // [Then] Los documentos lanzados no tienen que tener información en Ship-To Code
    //        Los documentos abiertos si tienen que tener información en Ship-To Code
    SalesHeader1.Get(SalesHeader1."Document Type",SalesHeader1."No.");
    SalesHeader2.Get(SalesHeader2."Document Type",SalesHeader2."No.");

    Assert.AreEqual('',SalesHeader2."Ship-to Code",'Ship-To Code no debería tener información');
    Assert.AreEqual(Customer."CLIP Default Ship-To Address",SalesHeader1."Ship-to Code",'Ship-To Code debería
tener la información de la dirección de envío predeterminada');
end;
```

4. Ejecuta el test y comprueba que éste falla por no haber gestionado el Confirm

5. Añade la siguiente línea al procedimiento de test que estamos desarrollando

```
[Test]
[HandlerFunctions('HandleConfirmCLIPDefaultShipToAddresValidate_Yes')]
procedure TestFunctionalityOnConfirmValidate()
```

6. Crea la función `HandleConfirmCLIPDefaultShipToAddresValidate_Yes`

```
[ConfirmHandler]
procedure HandleConfirmCLIPDefaultShipToAddresValidate_Yes(Question: Text[1024]; VAR Reply: Boolean)
begin
    Assert.AreEqual('Do you want to update the Ship-To Code on Sales Documents?', Question, 'El
CONFIRM es incorrecto');
    Reply := true;
end;
```