

## **Exercise Solution: User Defined Object**

User and Data Interface API

**PUBLIC**

## INTRODUCTION

In this exercise, you will perform the following tasks:

1. Create Use Defined Table
2. Create User Defined Fields
3. Create User Defined Object
4. Insert records to the User Defined Object
5. Bind data on the form
6. Test your Add-On

## PREREQUISITE:

- This document is using the **C Sharp** (C#) language
- This document is using the Microsoft Visual Studio 2015
- Continue to work with the project finalized in previous exercise.
- This document is using the SAP Business One Studio for Microsoft Visual Studio 2015
- Use the demo database for SAP Business One, version for SAP HANA or SAP Business One
- Credentials: User code: **manager**

## GUIDELINES:

The screenshots provided here are for your reference only and may differ from the actual screenshots in your system.

## 1. TASK – CREATE USER DEFINED TABLE

1.1. Create new class (name it UDO) in your Visual Studio Project.

1.2. Create a new **CreateUDT** function in Class UDO, which will handle the User Defined Table creation.

```
public static void CreateUDT(string MyTableName, string MyTableDescription,
SAPbobsCOM.BoUTBTableType MyTableType)
{
    try
    {
        SAPbobsCOM.UserTablesMD oUDT;

        oUDT =
(SAPbobsCOM.UserTablesMD)Program.di Company.GetBusinessObject(SAPbobsCOM.BoObjectTypes.oUserTables);

        if (oUDT.GetByKey(MyTableName) == false)
        {
            oUDT.TableName = MyTableName;
            oUDT.TableDescription = MyTableDescription;
            oUDT.TableType = MyTableType;
            int ret = oUDT.Add();

            if (ret == 0)
            {
                Application.SBO_Application.MessageBox("Add Table: " +
oUDT.TableName + " successful!");
                System.Runtime.InteropServices.Marshal.ReleaseComObject(oUDT);
                GC.Collect();
            }
            else
                Application.SBO_Application.MessageBox("Add Table error: " +
Program.di Company.GetLastErrorDescription());
        }
        else
            Application.SBO_Application.MessageBox("Table: " + MyTableName + "
already exists");
    }
    catch (Exception ex)
    {
        Application.SBO_Application.MessageBox(ex.Message);
    }
}
```

1.3. Add a User-Defined Table by executing the function **CreateUDT**.

Table name: TB1\_CAR  
Table description: Car Master Data  
Table type: Master Data

```
UDO.CreateUDT("TB1_CAR", "Car Master Data",
SAPbobsCOM.BoUTBTableType.bott_MasterData);
```

1.4. Add a User-Defined Table by executing the function **CreateUDT**.

Table name: TB1\_CAR\_D  
Table description: Car Details

Table type: Master Data Rows

```
UDO.CreateUDT("TB1_CAR_D", "Car Details",
SAPbobsCOM.BoUTBTableType.bott_MasterDataLines);
```

1.5. Add a User-Defined Table by executing the function **CreateUDT**.

Table name: TB1\_CAR\_D  
Table description: Car Details  
Table type: Master Data Rows

```
UDO.CreateUDF("TB1_CAR_D", "BODY", "Body Type", SAPbobsCOM.BoFieldTypes.db_Alpha,
30);
```

## 2. TASK - CREATE USER DEFINED FIELDS

2.1. Create a new **CreateUDF** function in Class UDO, which will handle the User Defined Field creation.

```
public static void CreateUDF(string MyTableName, string MyFieldName, string
MyFieldDescription, SAPbobsCOM.BoFieldTypes MyFieldType, int MyFieldSize)
{
    try
    {
        SAPbobsCOM.UserFieldsMD oUDF;
        oUDF =
(SAPbobsCOM.UserFieldsMD)Program.diCompany.GetBusinessObject(SAPbobsCOM.BoObjectTypes.oUserF
ields);

        oUDF.TableName = MyTableName;
        oUDF.Name = MyFieldName;
        oUDF.Description = MyFieldDescription;
        oUDF.Type = MyFieldType;
        oUDF.EditSize = MyFieldSize;
        int ret = oUDF.Add();

        System.Runtime.InteropServices.Marshal.ReleaseComObject(oUDF);
        GC.Collect();
    }
    catch (Exception ex)
    {
        Application.SBO_Application.MessageBox("Exception: " + ex.Message);
    }
}
```

2.2. Add a User-Defined Field by executing the function **CreateUDF**.

Table name: TB1\_CAR\_D  
Field name: MODEL  
Field description: Car Model  
Field type: Alphanumeric  
Field size: 30

```
UDO.CreateUDF("TB1_CAR_D", "MODEL", "Car Model", SAPbobsCOM.BoFi el dTypes.db_Al pha, 30);
```

2.3. Add a User-Defined Field by executing the function **CreateUDF**.

Table name: TB1\_CAR\_D  
Field name: FUEL  
Field description: Fuel Type  
Field type: Alphanumeric  
Field size: 30

```
UDO.CreateUDF("TB1_CAR_D", "FUEL", "Fuel Type", SAPbobsCOM.BoFi el dTypes.db_Al pha, 30);
```

2.4. Add a User-Defined Field by executing the function **CreateUDF**.

Table name: TB1\_CAR\_D  
Field name: BODY  
Field description: Body Type  
Field type: Alphanumeric  
Field size: 30

```
UDO.CreateUDF("TB1_CAR_D", "BODY", "Body Type", SAPbobsCOM.BoFi el dTypes.db_Al pha, 30);
```

2.5. Add a User-Defined Field by executing the function **CreateUDF**.

Table name: TB1\_CAR\_D  
Field name: POWER  
Field description: Horse Power  
Field type: Alphanumeric  
Field size: 30

```
UDO.CreateUDF("TB1_CAR_D", "POWER", "Horse Power", SAPbobsCOM.BoFi el dTypes.db_Al pha, 30);
```

### 3. TASK – CREATE USER DEFINED OBJECT

3.1. Create a new **CreateUDO** function in Class UDO, which will handle the User Defined Object creation.

```
public static void CreateUDO()  
{  
    try  
    {  
        SAPbobsCOM.UserObj ectsMD oUserObj ectMD;  
  
        oUserObj ectMD =  
(SAPbobsCOM.UserObj ectsMD)Program.di Company.GetBusi nessObj ect(SAPbobsCOM.BoObj ectTypes.oUser  
Obj ectsMD);  
  
        oUserObj ectMD.Code = "TB1_CAR";
```

```

oUserObjectMD.Name = oUserObjectMD.Code;
oUserObjectMD.ObjectType = SAPbobsCOM.BoUD00bjType.boud_MasterData;
oUserObjectMD.TableName = "TB1_CAR";

oUserObjectMD.CanDelete = SAPbobsCOM.BoYesNoEnum.tYES;
oUserObjectMD.CanFind = SAPbobsCOM.BoYesNoEnum.tYES;

oUserObjectMD.FindColumns.ColumnAlias = "Code";
oUserObjectMD.FindColumns.Add();
oUserObjectMD.FindColumns.ColumnAlias = "Name";
oUserObjectMD.FindColumns.Add();

oUserObjectMD.ChildTables.TableName = "TB1_CAR_D";

int ret = oUserObjectMD.Add();
if (ret != 0)
    Application.SBO_Application.MessageBox("error: " +
Program.diCompany.GetLastErrorDescription());

System.Runtime.InteropServices.Marshal.ReleaseComObject(oUserObjectMD);
GC.Collect();
}
catch (Exception ex)
{
    Application.SBO_Application.MessageBox(ex.Message);
}
}

```

#### 4. TASK - INSERT RECORDS TO THE USER DEFINED OBJECT

- 4.1. Create a new **InsertToUDO** function in Class UDO, which will insert data into the User Defined Object.

```

public static void InsertToUDO(string MyCode, string MyName, string MyModel,
string MyFuel, string MyBody, string MyPower)
{
    SAPbobsCOM.GeneralService oGeneralService = null;
    SAPbobsCOM.GeneralData oGeneralData = null;
    SAPbobsCOM.GeneralData oChild = null;
    SAPbobsCOM.GeneralDataCollection oChildren = null;
    SAPbobsCOM.GeneralDataParams oGeneralParams = null;
    SAPbobsCOM.CompanyService oCompanyService = null;
    try
    {
        oCompanyService = Program.diCompany.GetCompanyService();
        oGeneralService = oCompanyService.GetGeneralService("TB1_CAR");
        oGeneralData =
((SAPbobsCOM.GeneralData)(oGeneralService.GetDataInterface(SAPbobsCOM.GeneralServiceDataInterfaces.gsGeneralData)));
        oGeneralData.SetProperty("Code", MyCode);
        oGeneralData.SetProperty("Name", MyName);
        oChildren = oGeneralData.Child("TB1_CAR_D");
        oChild = oChildren.Add();
    }
}

```

```

oChild.SetProperty("U_MODEL", MyModel);
oChild.SetProperty("U_FUEL", MyFuel);
oChild.SetProperty("U_BODY", MyBody);
oChild.SetProperty("U_POWER", MyPower);
oGeneralParams = oGeneralService.Add(oGeneralData);

}
catch (Exception ex)
{
    //Application.SBO_Application.MessageBox(ex.Message);
}
}

```

#### 4.2. Add a User-Defined Object record by executing the function **InsertToUDO**.

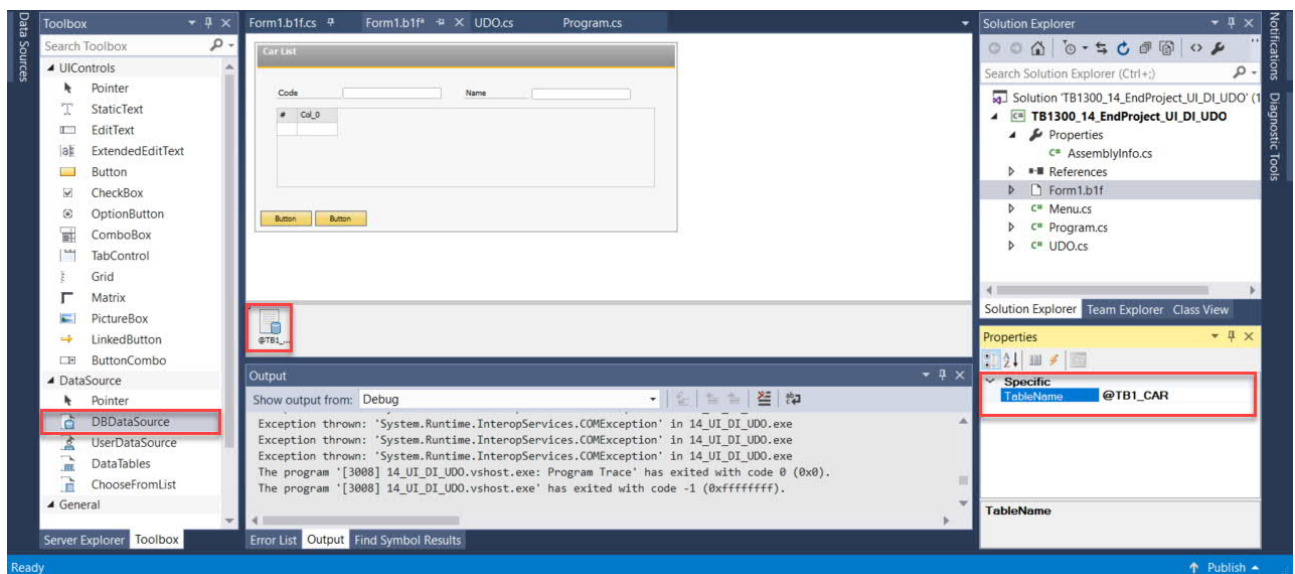
```

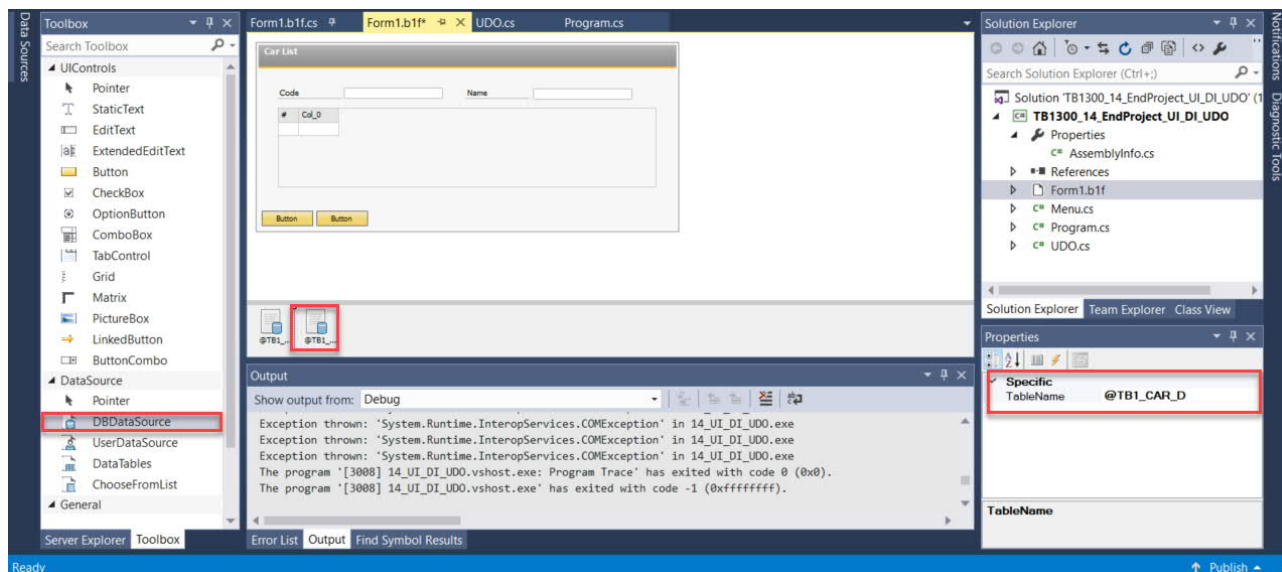
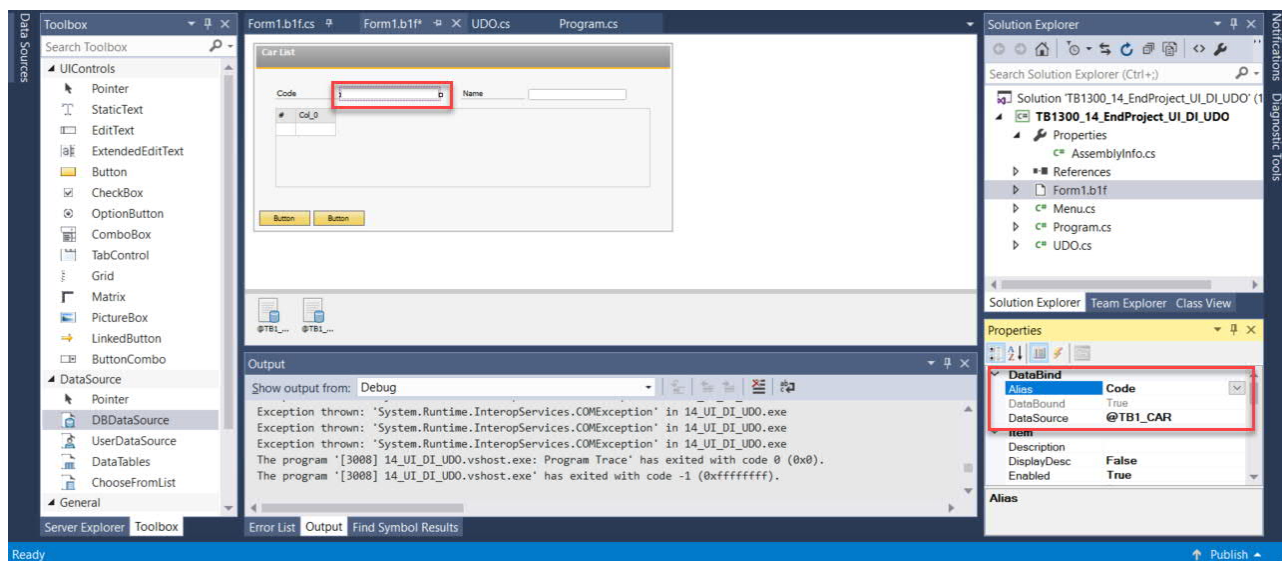
UDO.InsertToUDO("01", "BMW", "320i", "Petrol", "Sedan", "110");
UDO.InsertToUDO("02", "Ford", "Focus", "Diesel", "Hatchback", "120");
UDO.InsertToUDO("03", "Kia", "Rio", "Petrol", "Tourer", "130");
UDO.InsertToUDO("04", "Mercedes", "SLS", "Diesel", "Coupe", "140");
UDO.InsertToUDO("05", "Skoda", "Octavia", "Petrol", "Sedan", "150");
UDO.InsertToUDO("06", "Alfa Romeo", "Giulia", "Hybrid", "SUV", "160");
UDO.InsertToUDO("07", "Volkswagen", "Golf", "Petrol", "Coupe", "170");
UDO.InsertToUDO("08", "Peugeot", "Partner", "Diesel", "Van", "180");
UDO.InsertToUDO("09", "Lexus", "LS300", "Hybrid", "Sedan", "190");
UDO.InsertToUDO("10", "Toyota", "Yaris", "Petrol", "Hatchback", "200");

```

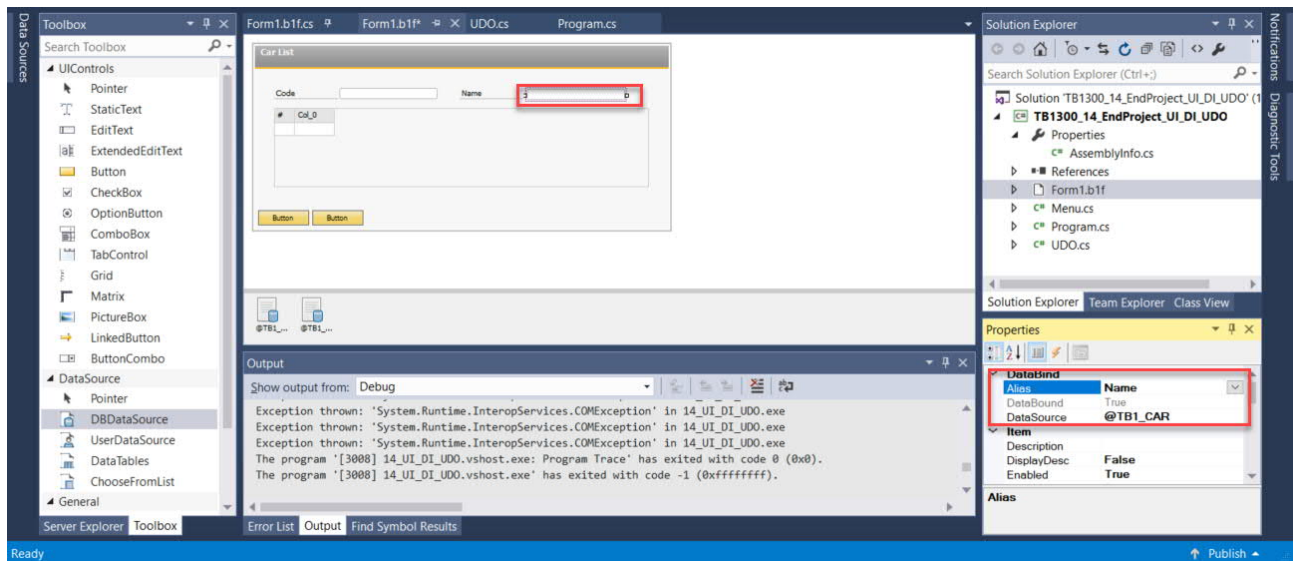
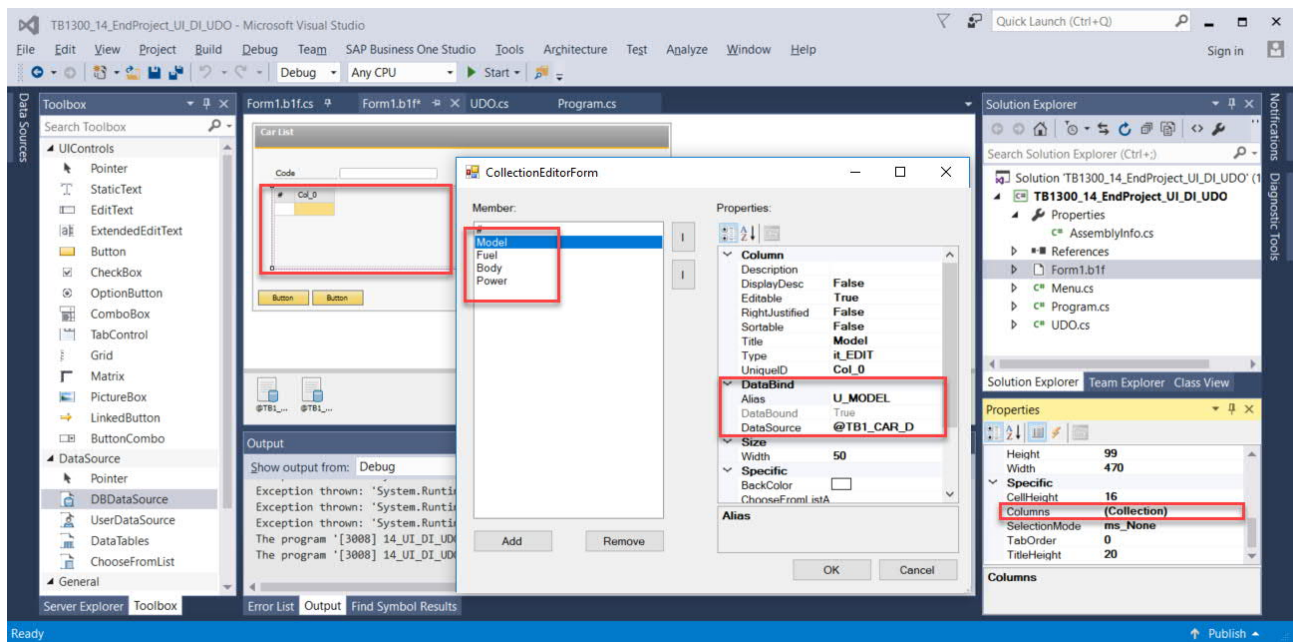
## 5. TASK - BIND DATA ON THE FORM

### 5.1. Add DB Data Source to the form for table **TB1\_CAR**.

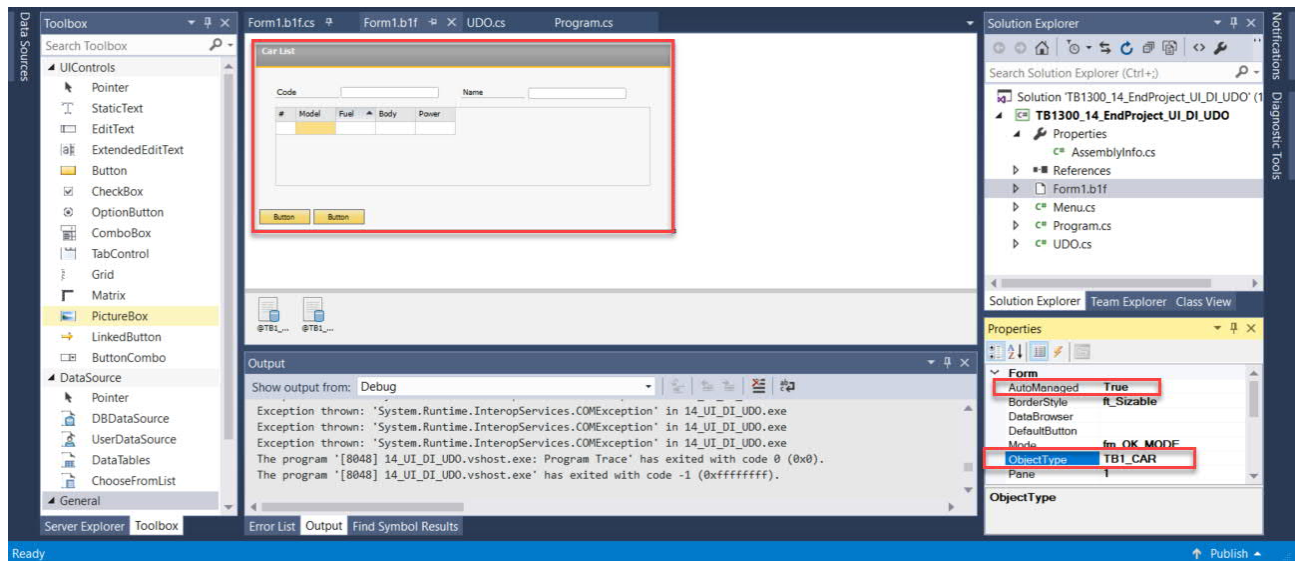


5.2. Add DB Data Source to the form for table **TB1\_CAR\_D**.5.3. Defined *Data Binding* properties for the *EditTex* object displaying the Code from table **TB1\_CAR**.



5.4. Defined *Data Binding* properties for the *EditTex* object displaying the Name from table **TB1\_CAR**.5.5. Defined *Data Binding* properties for the *Matrix* object displaying the value from table **TB1\_CAR\_D**.

- 5.6. Set the *AutoManaged* property value to **true** and the *ObjectType* to **TB1\_CAR** for the Form object.



## 6. TASK – TEST YOU ADD-ON SOLUTION

- 6.1. Search for an existing User Defined Object record.

Start your Add-On → switch the form to Find mode → enter the UDO Code into the corresponding EditText object → press the Find button

#	Model	Fuel	Body	Power
320i	Petrol	Sedan	110	



© 2018 SAP SE or an SAP affiliate company. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. Please see

<http://www.sap.com/corporate-en/legal/copyright/index.epx#trademark> for additional trademark information and notices. Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors.

National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP SE or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP SE or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

SAP SE or its affiliated companies have no obligation to pursue any course of business outlined in this document or any related presentation, or to develop or release any functionality mentioned therein. This document, or any related presentation, and SAP SE's or its affiliated companies' strategy and possible future developments, products, and/or platform directions and functionality are all subject to change and may be changed by SAP SE or its affiliated companies at any time for any reason without notice. The information in this document is not a commitment, promise, or legal obligation to deliver any material, code, or functionality. All forward-looking statements are subject to various risks and uncertainties that could cause actual results to differ materially from expectations. Readers are cautioned not to place undue reliance on these forward-looking statements, which speak only as of their dates, and they should not be relied upon in making purchasing decisions.