Beginners Guide to Visual Studio LightSwitch (Part - 1)

Visual Studio LightSwitch is a new tool for building data-driven Silverlight Application using Visual Studio IDE. It automatically generates the User Interface for a DataSource without writing any code. You can write a small amount of code also to meet your requirement.

Recently, I got some time to explore Visual Studio LightSwitch. I created a small DB application with proper data inserting UI within a small amount of time (without any XAML or C# code).

Here in this article, I will guide you to understand it with the help of a small application. There will be a series of articles on this topic regularly. Read the complete article to learn about creating a Silverlight data driven application with the help of Visual Studio LightSwitch.

Setting up LightSwitch Environment

Microsoft Visual Studio LightSwitch Beta 1 is a flexible, business application development tool that allows developers of all skill levels to quickly build and deploy professional-quality desktop and Web business applications. To start with LightSwitch application development, you need to install the Visual Studio LightSwitch in your development machine. To do so, follow the below steps:

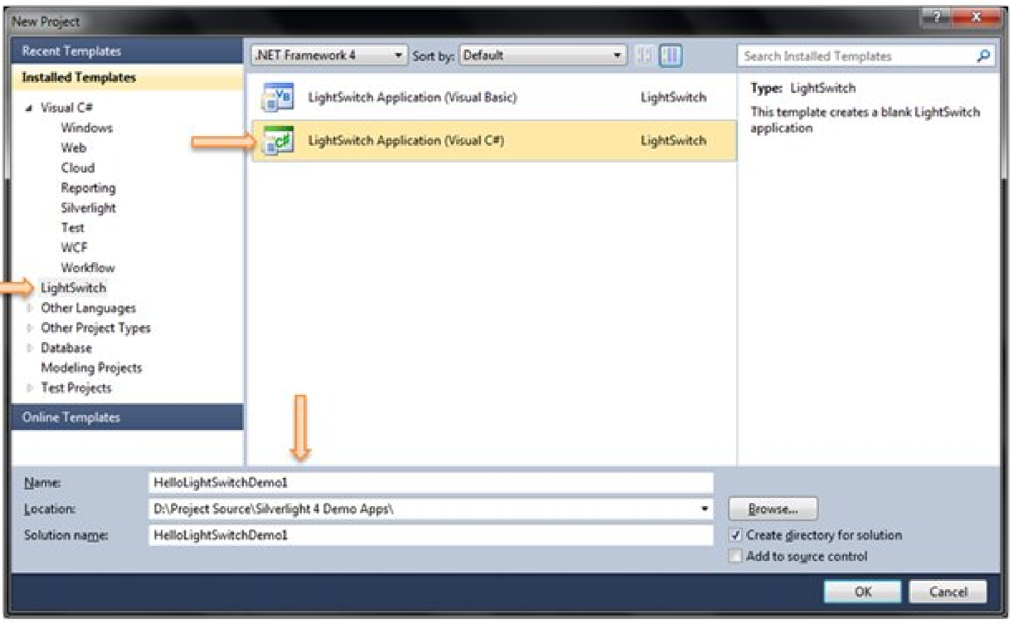
- Install Visual Studio 2010
- Install Visual Studio LightSwitch

The LightSwitch installation will install all other components to your PC one by one including SQL Express, Silverlight 4, LightSwitch Beta server etc.

You can download the LightSwitch Beta 1 from here: Microsoft Download Center (Visual Studio LightSwitch Beta 1)

Creating a LightSwitch Project

Once you installed Visual Studio LightSwitch, Run the Product to create a new project. Go to File –> New –> Project or press Ctrl + Shift + N to open the “New Project”. From the left panel, select “LightSwitch”. In the right pane, it will filter the LightSwitch project templates. This will include of type VB & C#. Select your respective type. Here I will use the C# version.

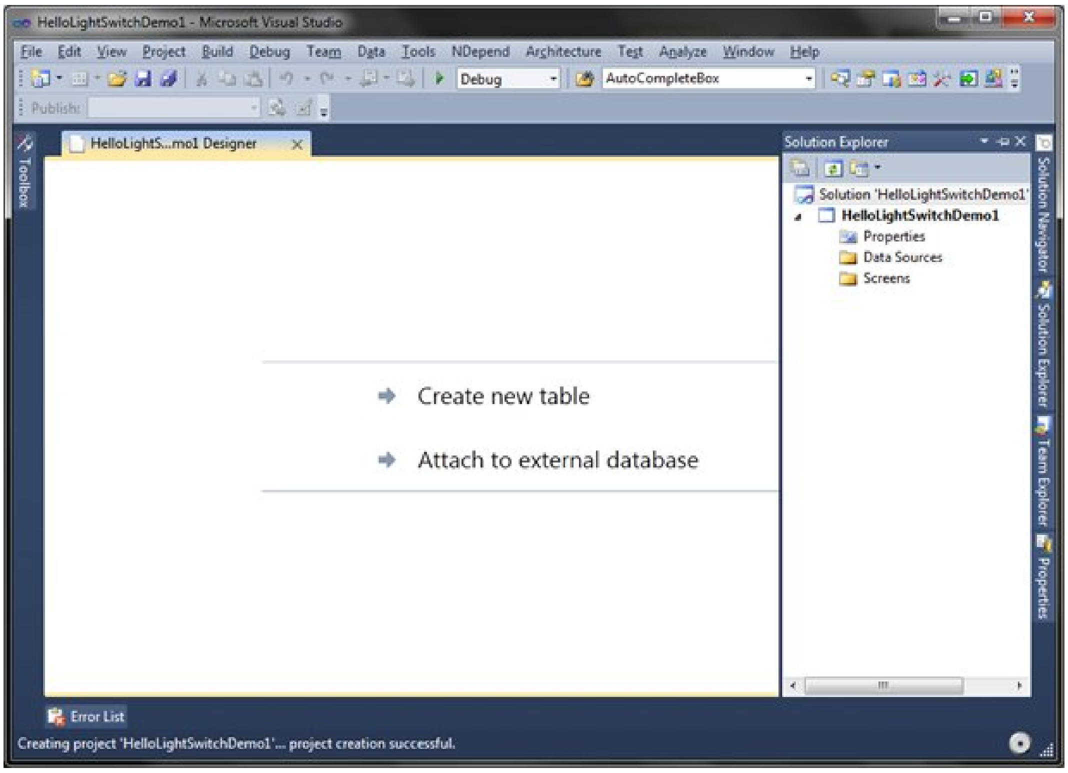


Name: HelloLightSwitchDemo1
Location: D:\Project Source\Silverlight 4 Demo Apps\
Solution name: HelloLightSwitchDemo1
✓ Create directory for solution
Add to source control

In the above dialog Window, enter the name of the project, select proper location for the project to create and hit "Ok". This will create the blank LightSwitch project for you by the Visual Studio 2010 IDE. It will take some time for the project creation. So, be patient.

Create a Database Table

After the project has been created by the IDE, it will open up the following screen:


HelloLightSwitchDemo1 - Microsoft Visual Studio
File Edit View Project Build Debug Team Data Tools NDepend Architecture Test Analyze Window Help
Debug
AutoCompleteBox
Publish:
Toolbox
HelloLightS...mo1 Designer
Create new table
Attach to external database
Solution Explorer
Solution 'HelloLightSwitchDemo1'
HelloLightSwitchDemo1
Properties
Data Sources
Screens
Solution Navigator
Solution Explorer
Team Explorer
Properties
Error List

Creating project 'HelloLightSwitchDemo1'... project creation successful.

You can see that, it has two options in the UI. You can create a new table for your application. Also, you can attach an external database. If you open the Solution Explorer, you will see that, the project is totally empty. It has only two folders named “Data Sources” and “Screens”.

“Data Sources” stores your application data i.e. Database Tables. On the other side, the “Screens” folder stores the UI screens created by you. I will describe them later in this tutorial.

Let’s create a new Table for our application. Click on the “Create new table” to continue. This will bring up the following screen in your desktop:

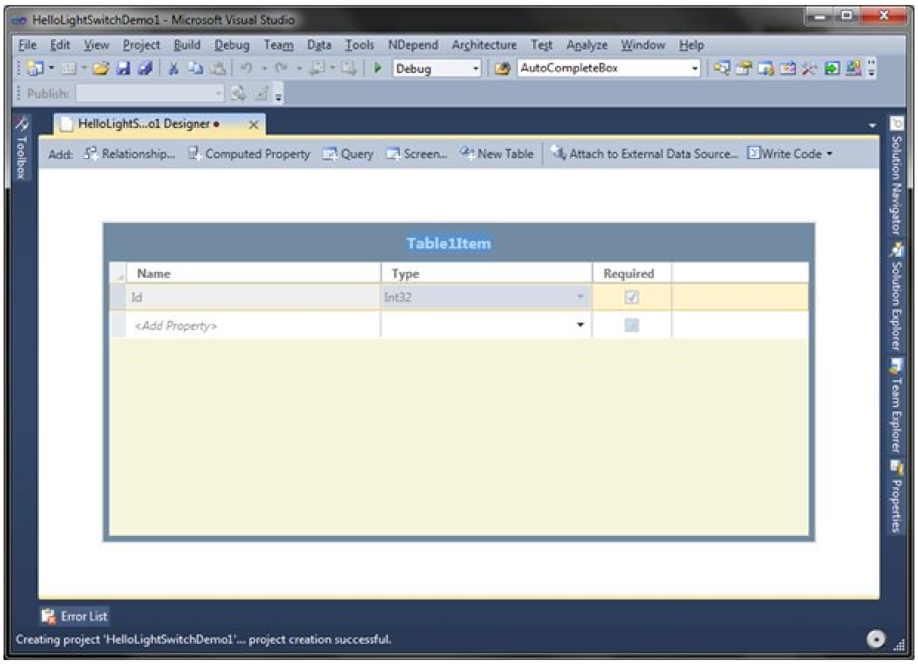

HelloLightSwitchDemo1 - Microsoft Visual Studio
File Edit View Project Build Debug Team Data Tools NDepend Architecture Test Analyze Window Help
Debug
AutoCompleteBox
Publish:
HelloLightS...o1 Designer
Add: Relationship... Computed Property Query Screen... New Table Attach to External Data Source... Write Code
Toolbox
Solution Navigator
Solution Explorer
Team Explorer
Properties
Error List

Table1Item

Name	Type	Required
Id	Int32	✓
<Add Property>		

Creating project 'HelloLightSwitchDemo1'... project creation successful.

In the above page, you can design your Table structure as you do in the SQL server while creating a new table. Each table will have a “Id” assigned to the table of type “Int32” & it is a Primary Key to the table. Add some additional columns to the table.

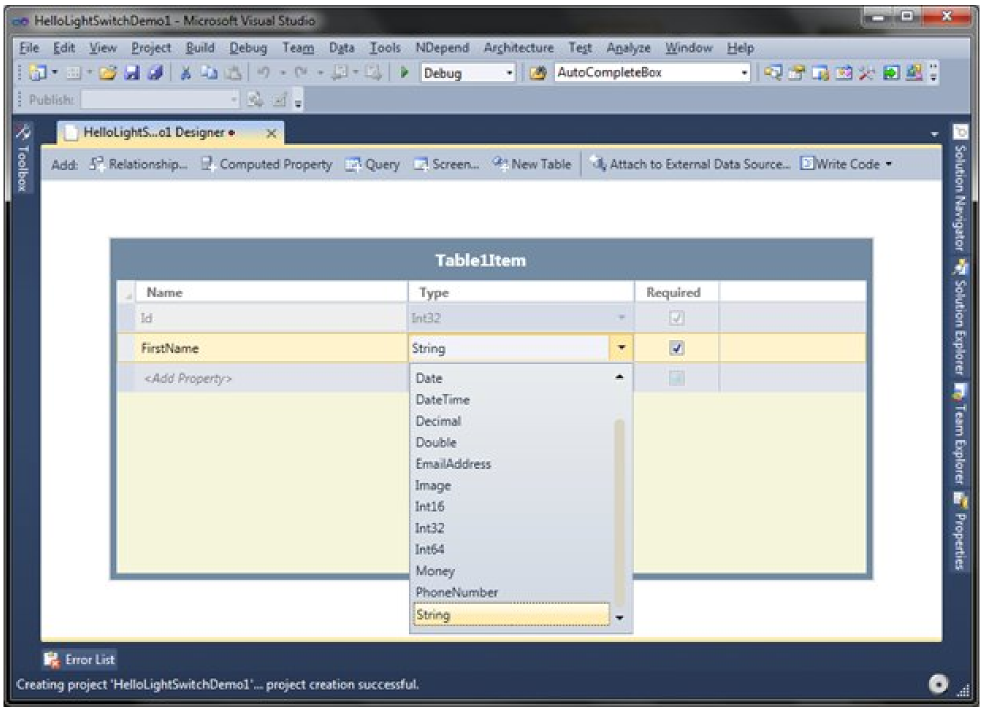
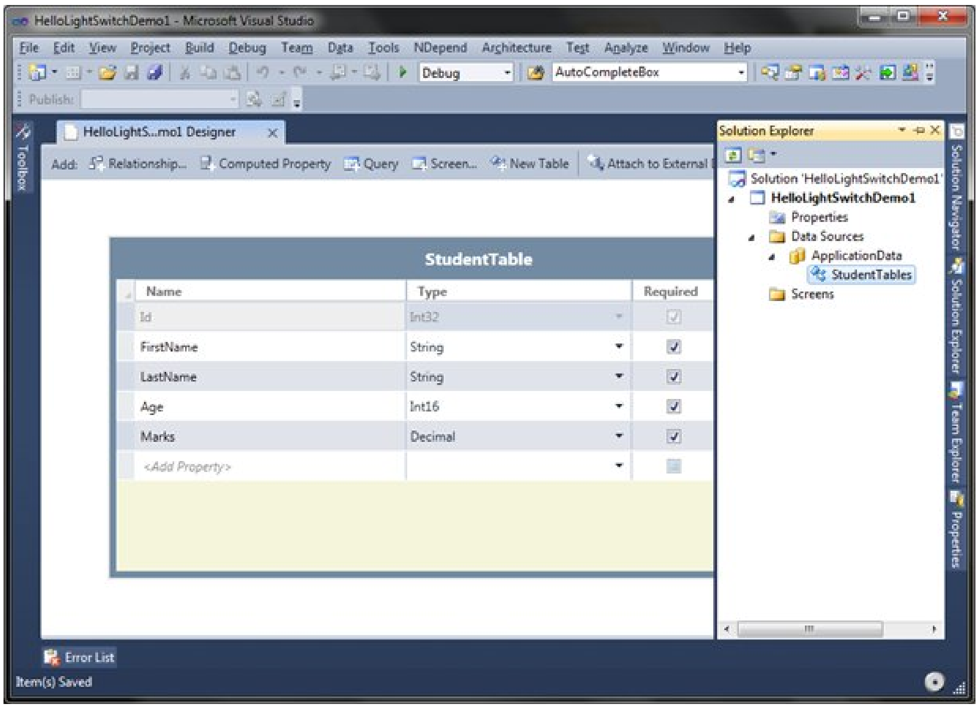

HelloLightSwitchDemo1 - Microsoft Visual Studio
File Edit View Project Build Debug Team Data Tools NDepend Architecture Test Analyze Window Help
Debug
AutoCompleteBox
Publish:
HelloLightS...o1 Designer
Add: Relationship... Computed Property Query Screen... New Table Attach to External Data Source... Write Code
Toolbox
Solution Navigator
Solution Explorer
Team Explorer
Properties
Date
DateTime
Decimal
Double
EmailAddress
Image
Int16
Int32
Int64
Money
PhoneNumber
String
Error List

Table1Item

Name	Type	Required
Id	Int32	✓
FirstName	String	✓
<Add Property>		

Creating project 'HelloLightSwitchDemo1'... project creation successful.

In the above snapshot, you can see that, there are several column types (data types) available in LightSwitch application. For our sample application, we will create 4 additional columns called “FirstName (String)”, “LastName (String)”, “Age (Int16)” and “Marks (Decimal)”. Change the title of the table from “Table1Item” to “StudentTable”. This name will be used while saving the table. Save the table now. If you want to change the name of the table later, just rename the table header and save it. This will automatically update the table name.


HelloLightSwitchDemo1 - Microsoft Visual Studio
File Edit View Project Build Debug Team Data Tools NDepend Architecture Test Analyze Window Help
Debug
AutoCompleteBox
Publish:
HelloLightS...mo1 Designer
Add: Relationship... Computed Property Query Screen... New Table Attach to External
Solution Explorer
Solution 'HelloLightSwitchDemo1'
HelloLightSwitchDemo1
Properties
Data Sources
ApplicationData
StudentTables
Screens
Toolbox
Solution Navigator
Solution Explorer
Team Explorer
Properties
Error List
Item(s) Saved

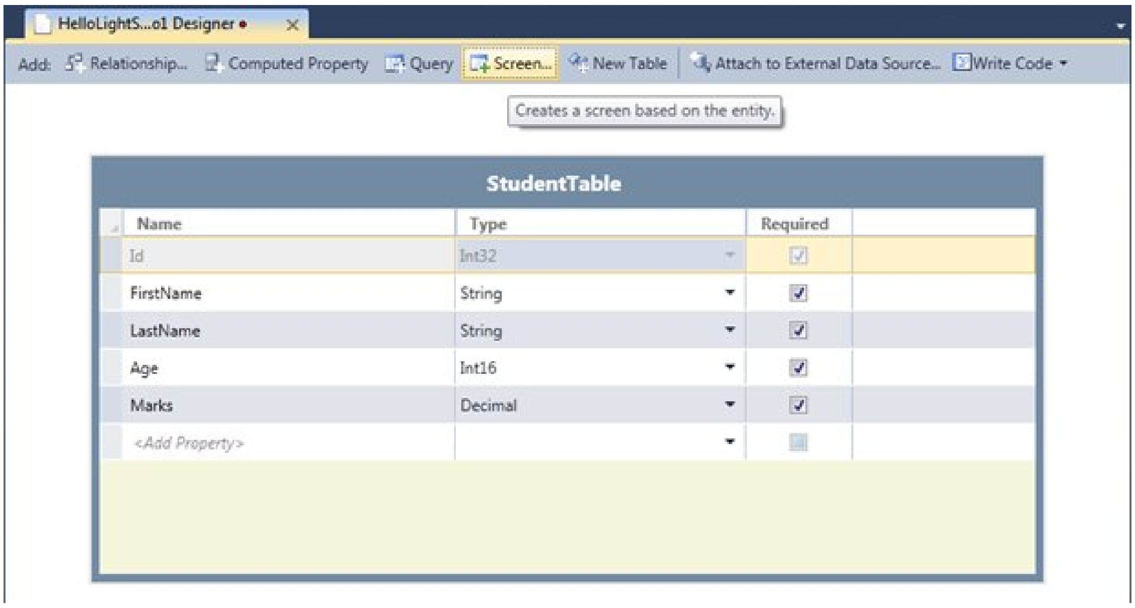
StudentTable

Name	Type	Required
Id	Int32	✓
FirstName	String	✓
LastName	String	✓
Age	Int16	✓
Marks	Decimal	✓
<Add Property>		

Be sure to set all the fields as “Required” field. This will be helpful validating the data. I will show it to you later.

Create a Data Entry Screen

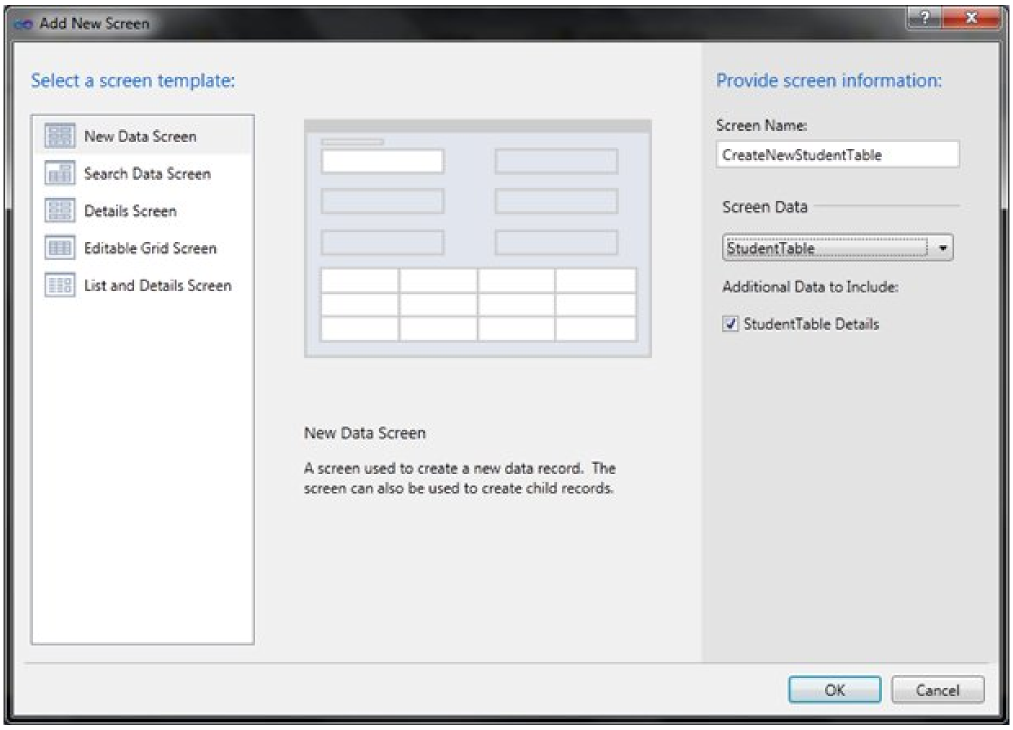
Once you are done with structuring your database table columns, you will need to create a UI screen for your application to insert records. Click on the “Screen…” button from the top panel as shown in the below snapshot:


HelloLightS...o1 Designer
Add: Relationship... Computed Property Query Screen... New Table Attach to External Data Source... Write Code
Creates a screen based on the entity.

StudentTable

Name	Type	Required
Id	Int32	✓
FirstName	String	✓
LastName	String	✓
Age	Int16	✓
Marks	Decimal	✓
<Add Property>		

This will open the “Add New Screen” dialog window to the screen. Select “New Data Screen” from the Screen Template, provide a Screen Name in the right panel & chose the database table as Screen Data from the dropdown.



Screen Name: CreateNewStudentTable

Screen Data: StudentTable

Additional Data to Include:

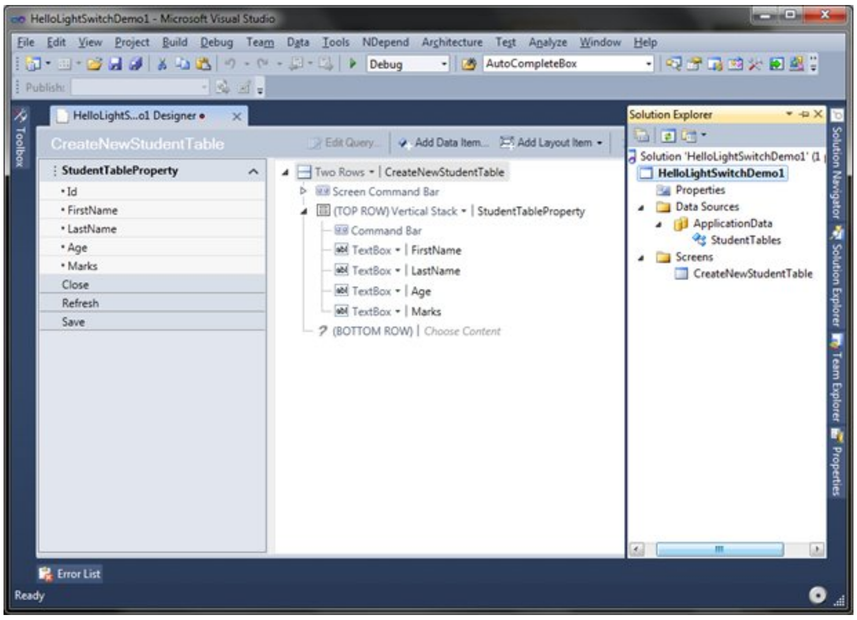
☑ StudentTable Details

New Data Screen

A screen used to create a new data record. The screen can also be used to create child records.

Click “Ok” to continue. This will create a new UI screen for your application to insert new data record by end user. Now, in the solution explorer you can see that, the “Data Sources” folder has one database named “ApplicationData” and it has a table called “StudentTables”. In the “Screens” folder you can find the recently created data entry screen named “CreateNewStudentTable”.

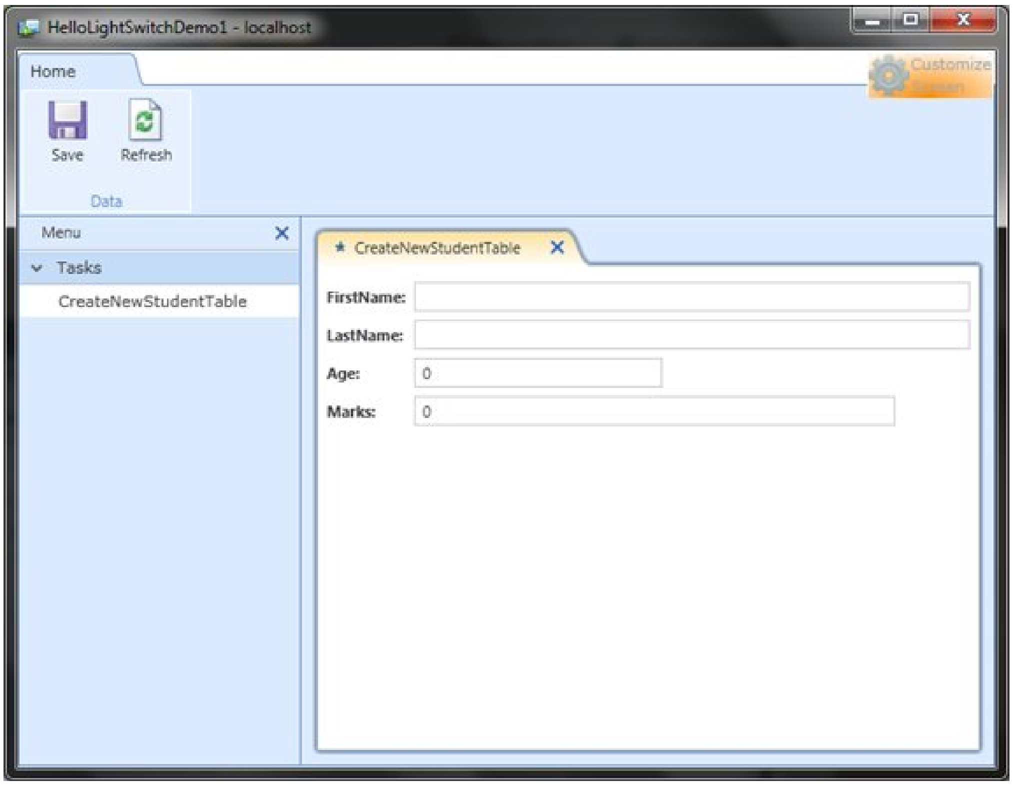
You can change the design of the UI from the below screen:



You can add or delete new field or controls. You can also rearrange the controls in the UI. For our first sample application, we will go with the default layout controls.

See the Application in Action

Woo!!! Our application is ready. We will able to insert new records in our database table from our application. No need to write a single line of code. What? You are not agreeing with me! Let's run the application by pressing F5. This will build your solution. It will take some time to compile the solution. Once it builds successfully, it will open the following UI in your desktop:



FirstName:
LastName:
Age: 0
Marks: 0

It is a Silverlight OOB Application. If you want to confirm, right click on the application & you will see the Silverlight context menu pops up in the screen.

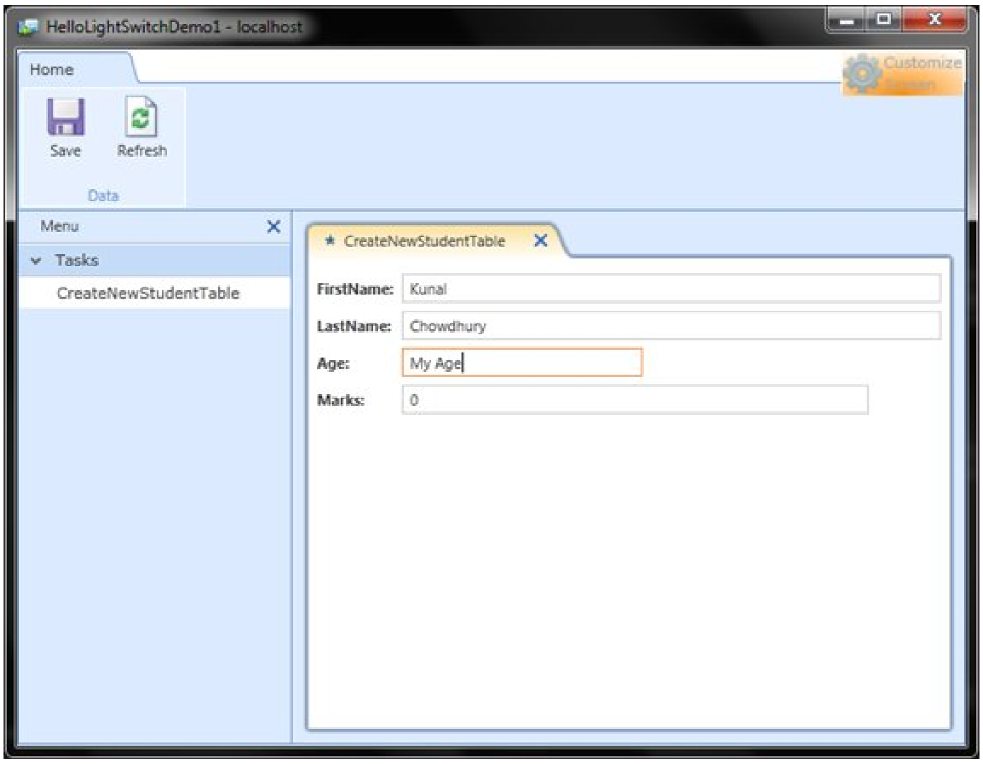
OMG!!! We didn't do anything to design the above UI! The Visual Studio LightSwitch automatically created the screen for you with a "Save" and "Refresh" button. You can see a collapsible "Menu" panel at the left of the screen. In the right side, you will see tabular panel containing some labels and TextBox which will be require to insert data in your application database.

In the top right corner of the screen, you will see a "Customize Screen" button. Once you click this, it will pop up another Child Window for you to customize the application screen at runtime. This will not be visible, once you deploy the application. We will cover them later in different article.

Validation of Fields

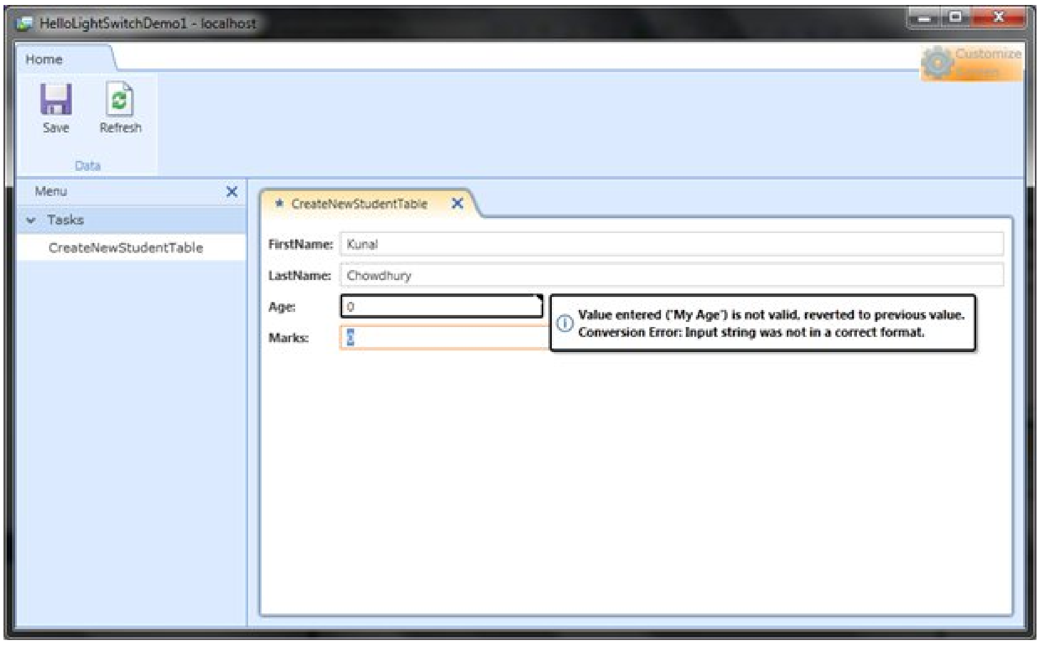
As I told you earlier, our application is ready. So, what to do for writing some simple validation for our application? Wait, you don't have to worry about that. LightSwitch internally does that for you. Let's have a demo on it.

In the form panel, insert some data. We will purposefully write invalid data in the "Age" field. Instead of entering age, we will enter some text in the age field. If you press tab or click elsewhere (on lost focus), it will do the validation & show the error on the screen.



FirstName: Kunal
LastName: Chowdhury
Age: My Age
Marks: 0

You can see the validation error popup in the screen as shown here:



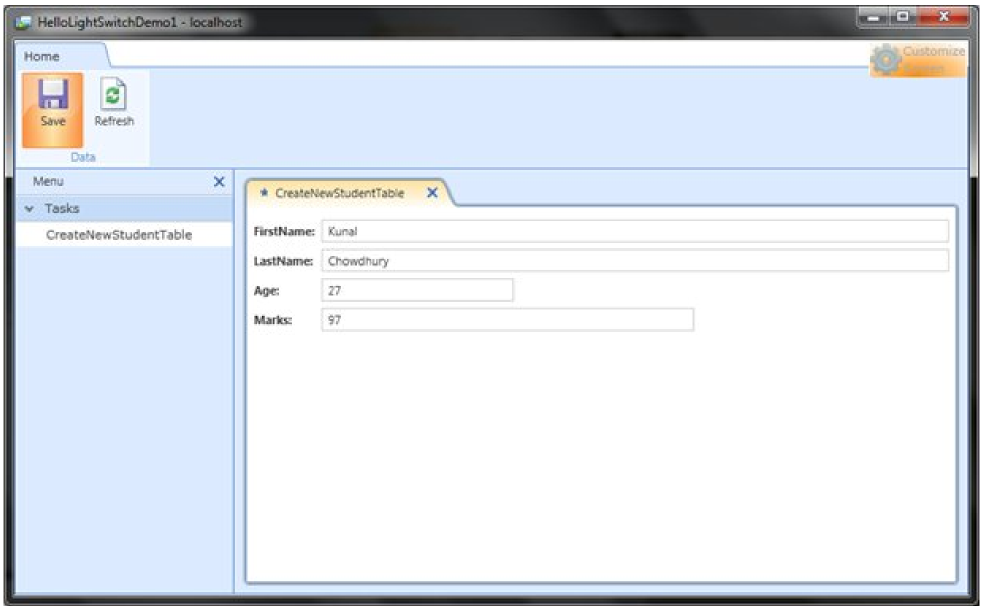
FirstName: Kunal
LastName: Chowdhury
Age: 0
Marks: 0

Value entered ('My Age') is not valid, reverted to previous value.
Conversion Error: Input string was not in a correct format.

Now enter proper values for all of the fields and click “Save”. This will save the records properly.

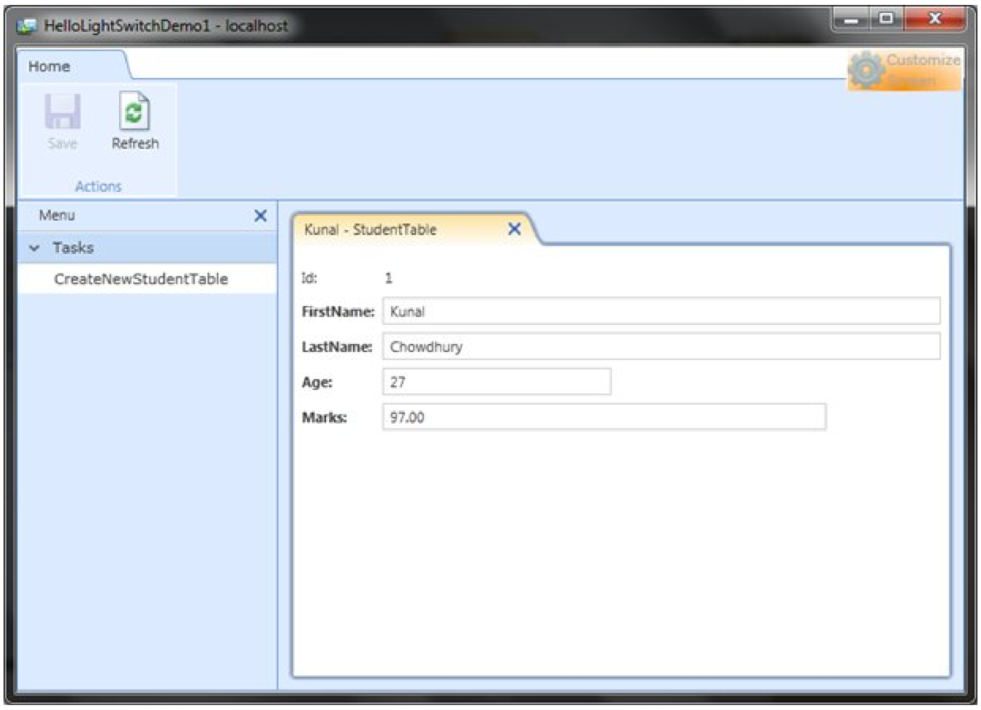
More on Save

Once you entered proper values in the input boxes, click “Save” to store the record.



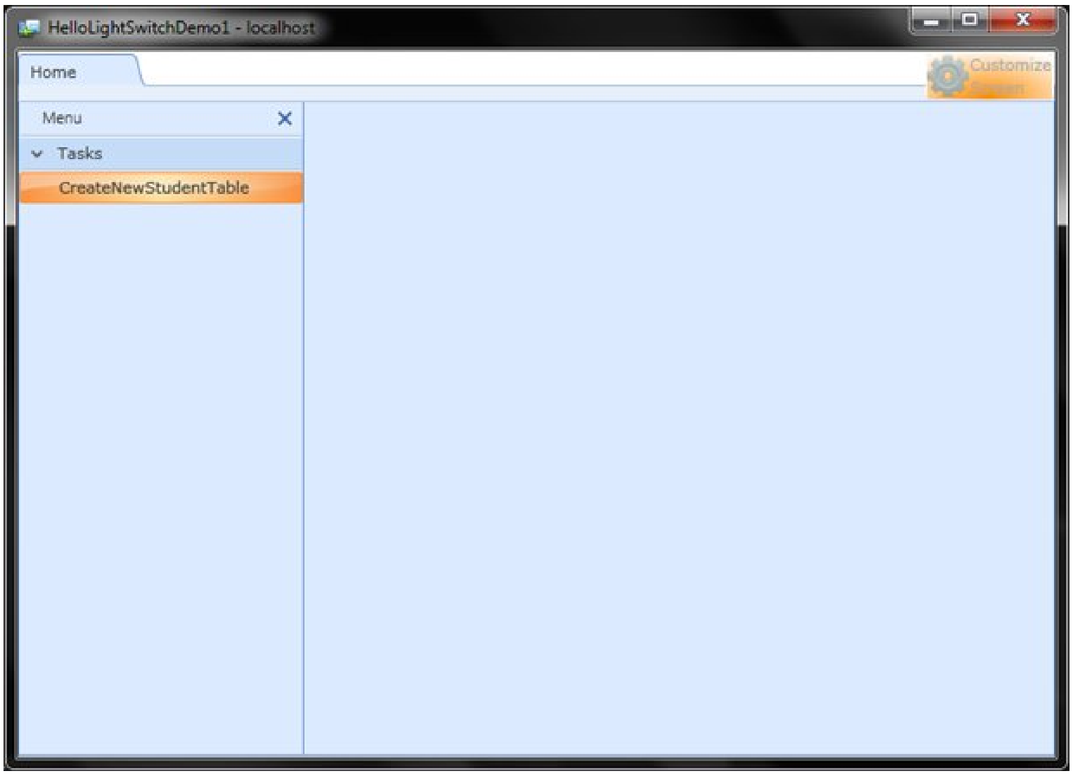
FirstName: Kunal
LastName: Chowdhury
Age: 27
Marks: 97

This will save the record and add another field called “Id” in the UI. The value for the Id will be 1 (one) as this was our first record & will increase by 1 as we add more records.

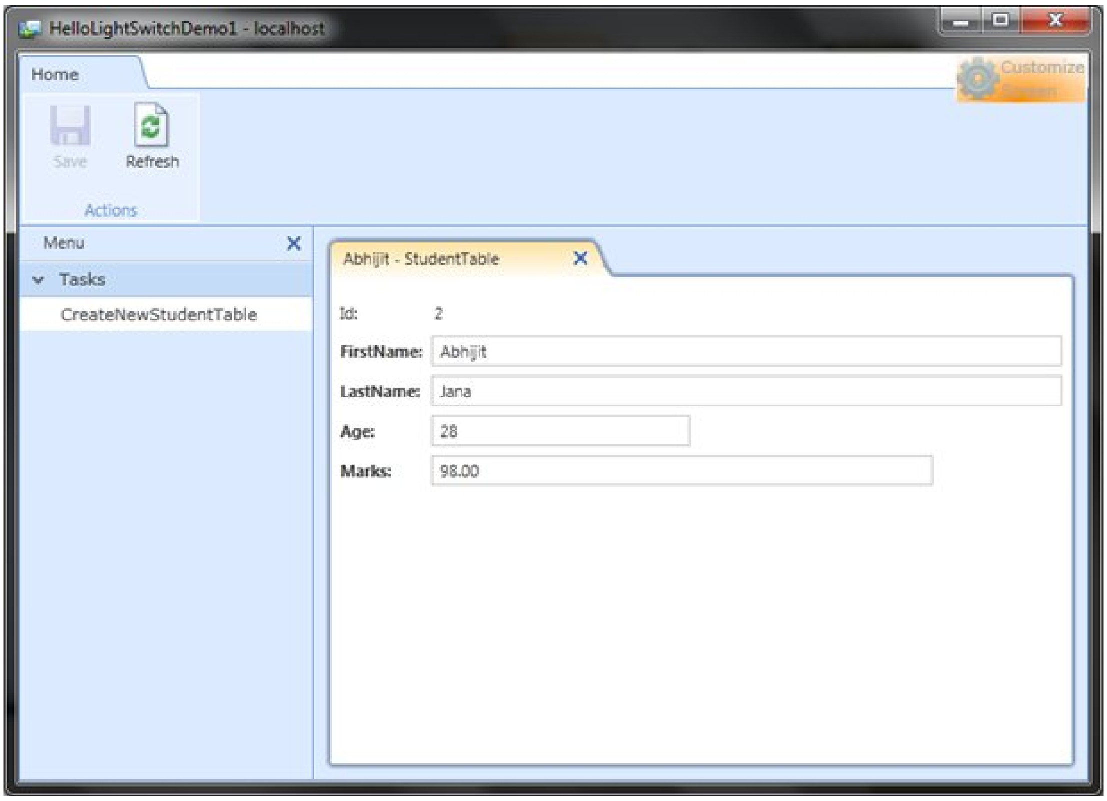


Id: 1
FirstName: Kunal
LastName: Chowdhury
Age: 27
Marks: 97.00

To add more records, close the current tab and click “CreateNewStudentTable” menu item from the left panel.

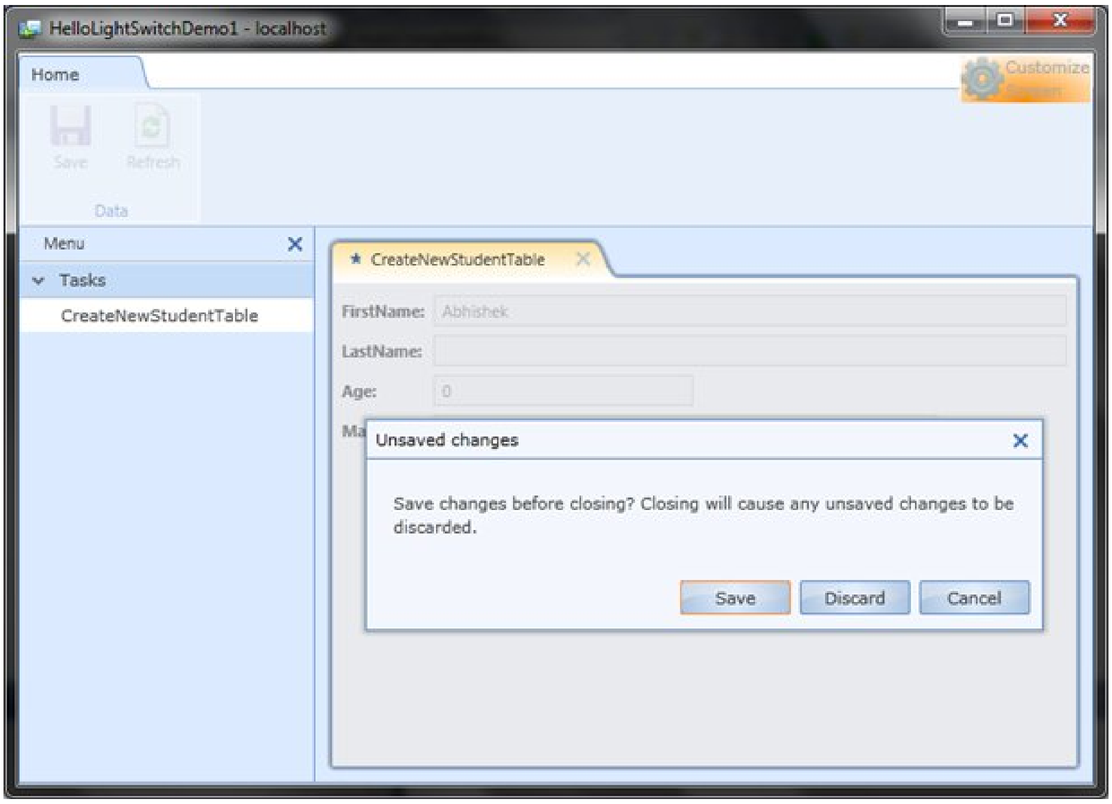


Now enter some records and click Save button. You will see it in action that, the Id is automatically increasing by 1 as and when you are saving a new record.



Id: 2
FirstName: Abhijit
LastName: Jana
Age: 28
Marks: 98.00

If you have any unsaved records and want to close the tab or want to add a new record, it will open a message dialog to the screen warning about unsaved changes. It will ask you to save the changes before proceeding.

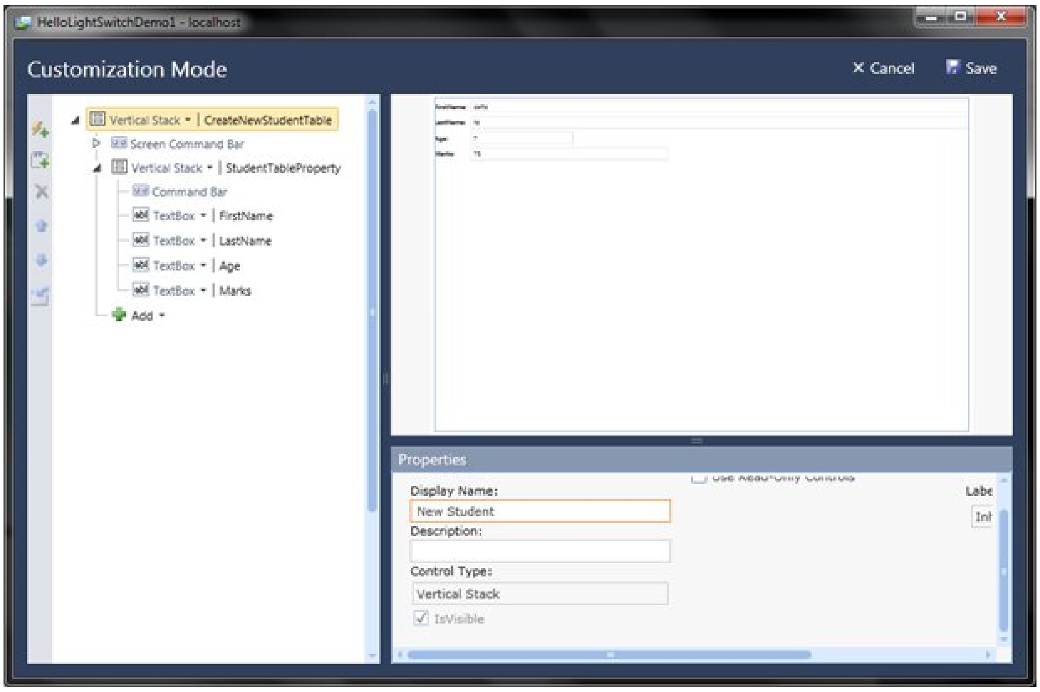


Save changes before closing? Closing will cause any unsaved changes to be discarded.

Click “Save” to save the unsaved record to database. Click “Discard” if you don’t want to save the record. Click “Cancel” to stay on the same page.

Customize the Screen

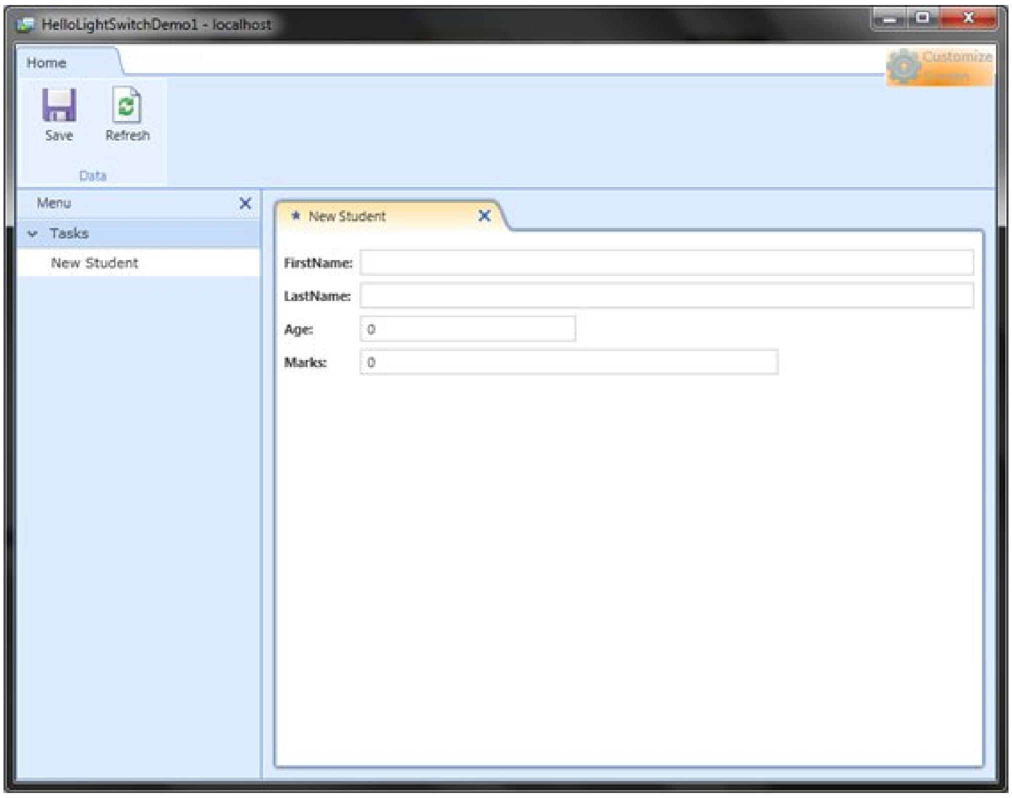
If you want to change the UI at runtime, click on the “Customize Screen” button available at the top right corner of the application window. Remember, this button will not visible once you deploy the application. Once you click the button, it will popup the Customization screen. Here you can change the layout of the screen, insert come new field or delete something which are already exist. For now, we will not go with changing the layout or controls. I will cover them in another article.



Display Name: New Student
Description:
Control Type: Vertical Stack
IsVisible

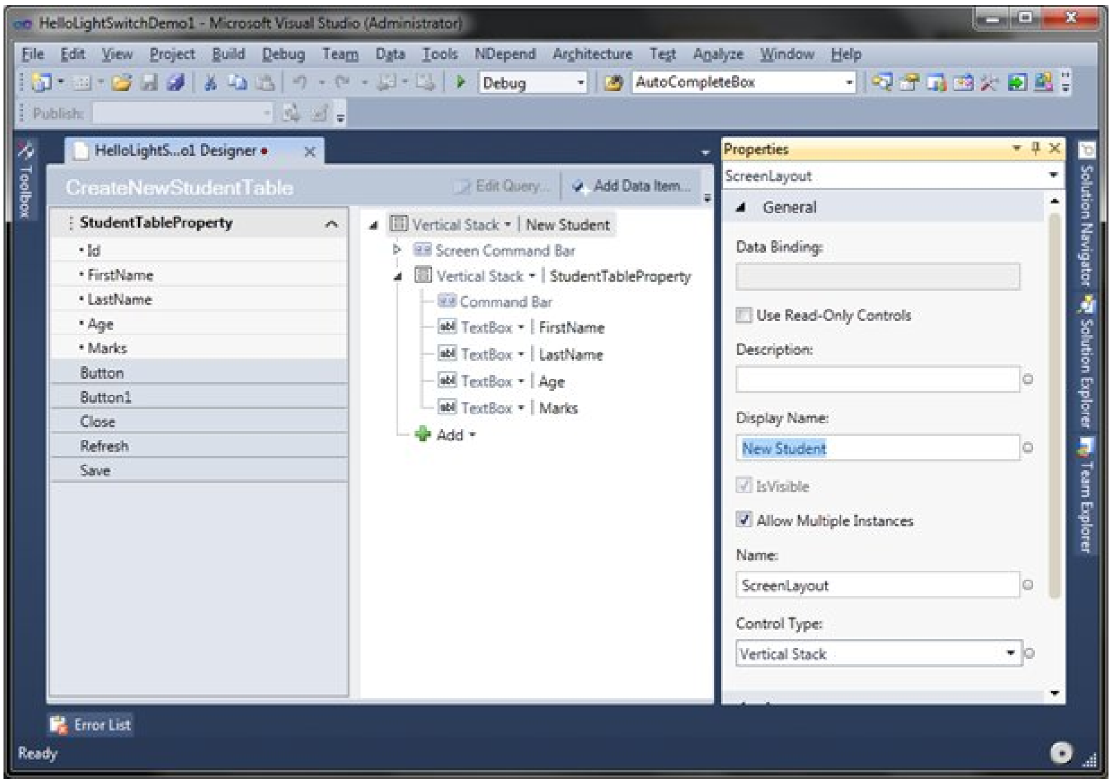
Here, we will change the display name of the new student menu item. Select the top level node of the screen items and in the properties panel change the “Display Name” from the default one to “New Student”. Now save the settings and restart the application for the changes to take effect.

Once you run you application now, you will see the display name of the menu item has been changed to the entered value i.e. “New Student”. This is more meaningful than the default text.



FirstName:
LastName:
Age: 0
Marks: 0

If you want to do the same thing inside the Visual Studio IDE, go to the properties pane as shown in the below figure. Find the “Display Name” property and change the display name to your own meaningful name. Run the application to see it in action.



You can change many other properties from the properties panel too.

End Note
Oh, it is too easy to create a data-driven Silverlight OOB application using Visual Studio LightSwitch. No need to write XAML or C# code to create such professional quality application in Silverlight.

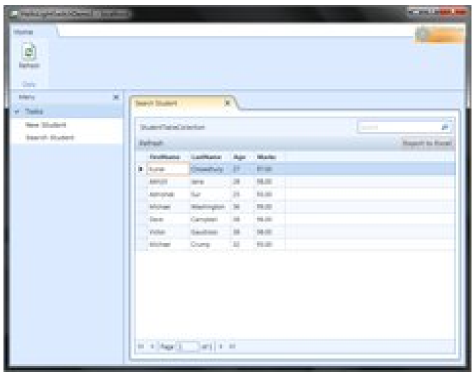
Hope, this helped you to understand Visual Studio LightSwitch to create a simple data entry application without writing any C# code or writing any XAML code. In the next few articles of the series, I will discuss more about the other screens.

I think, you enjoyed it a lot and hence please give your feedback about this article.

Beginners Guide to Visual Studio LightSwitch (Part - 2)

Introduction

Visual Studio LightSwitch is a new tool for building data-driven Silverlight Application using Visual Studio IDE. It automatically generates the User Interface for a DataSource without writing any code. You can write a small amount of code also to meet your requirement.



In my previous Chapter "**Beginners Guide to Visual Studio LightSwitch (Part–1)**" I described about LightSwitch and did demonstration from creating a new LightSwitch project to create a simple form to insert new Student record.

In this article, I will show you how to create a search record window & export the records to Excel Sheet using Visual Studio LightSwitch for Silverlight. No need to write a single piece of code, no need to write a single XAML code. Read through the complete article to learn more about it.

Background

If you are new to Visual Studio LightSwitch, I will first ask you to read the first chapter of this tutorial, where I demonstrated the following points:

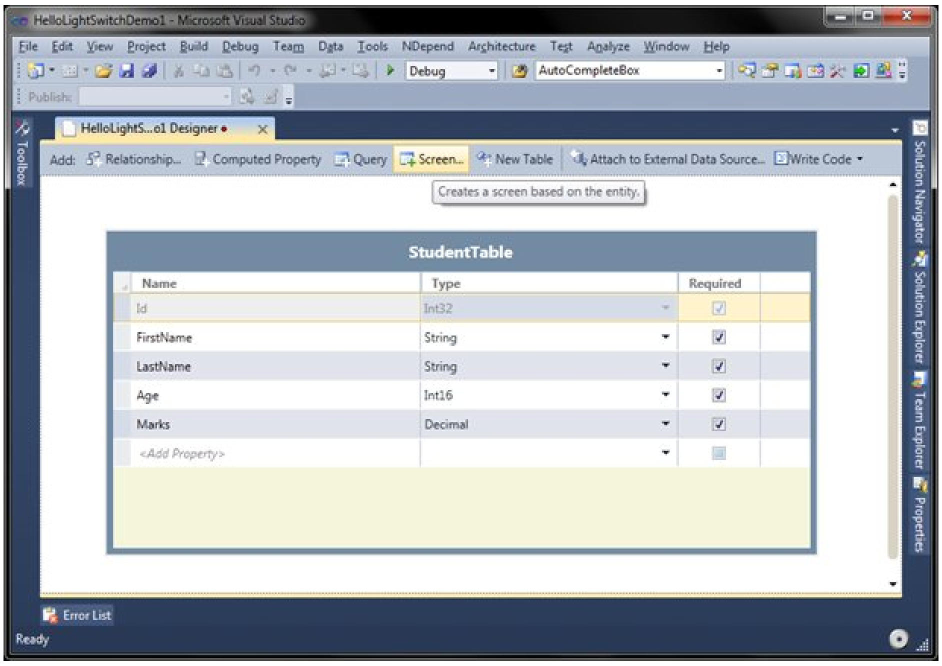
- Setting up LightSwitch Environment
- Creating a LightSwitch Project
- Create a Database Table
- Create a Data Entry Screen
- See the Application in Action
- Validation of Fields
- More on Save
- Customize the Screen

If you go through the above points, you will be familiar with it and you will be able to understand this Chapter. You can read the first Chapter here: Beginners Guide to Visual Studio LightSwitch (Part–1). Don't forget to vote for it.

In this article, we will talk about the Search Screen. We will create the search window, where we will be able to search for a particular record, sort the records and export the record to Excel Sheet. Yes, without writing any code.

Create the Search Screen

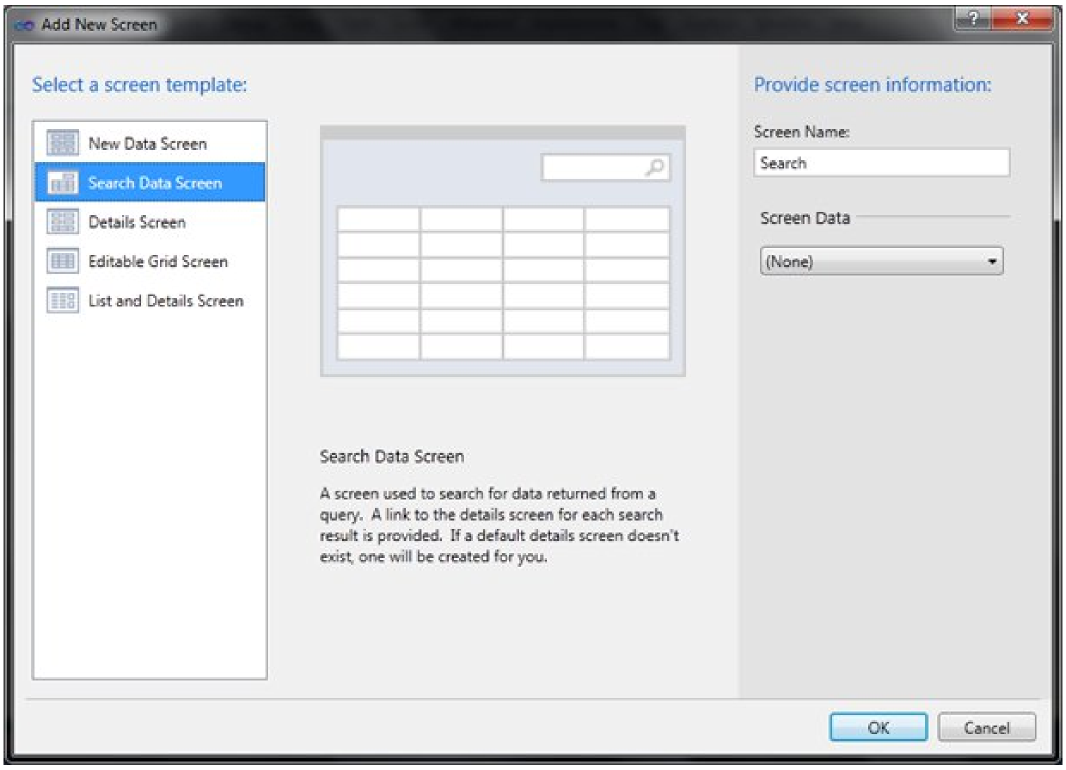
Open the solution that we created in the previous chapter. By default, the table designer will be visible on the screen. If not, open the “StudentTables” from the Solution Explorer. This will open the below screen in your Visual Studio IDE:


HelloLightSwitchDemo1 - Microsoft Visual Studio
File Edit View Project Build Debug Team Data Tools NDepend Architecture Test Analyze Window Help
Debug
AutoCompleteBox
Publish:
HelloLightS...o1 Designer
Add: Relationship... Computed Property Query Screen... New Table Attach to External Data Source... Write Code
Creates a screen based on the entity.
Toolbox
Solution Navigator
Solution Explorer
Team Explorer
Properties
Error List
Ready

StudentTable

Name	Type	Required
Id	Int32	✓
FirstName	String	✓
LastName	String	✓
Age	Int16	✓
Marks	Decimal	✓
<Add Property>		

Click on the “Screen” button as shown in the above figure. This will pop up the “Add New Screen” to the desktop. Here you can select a screen template. Default selection is “New Data Screen”. As we are going to create a Search Screen, hence choose the “Search Data Screen” template from the template panel situated at the left side of the dialog window. It will show you the preview of the screen in the middle panel. Here it is:



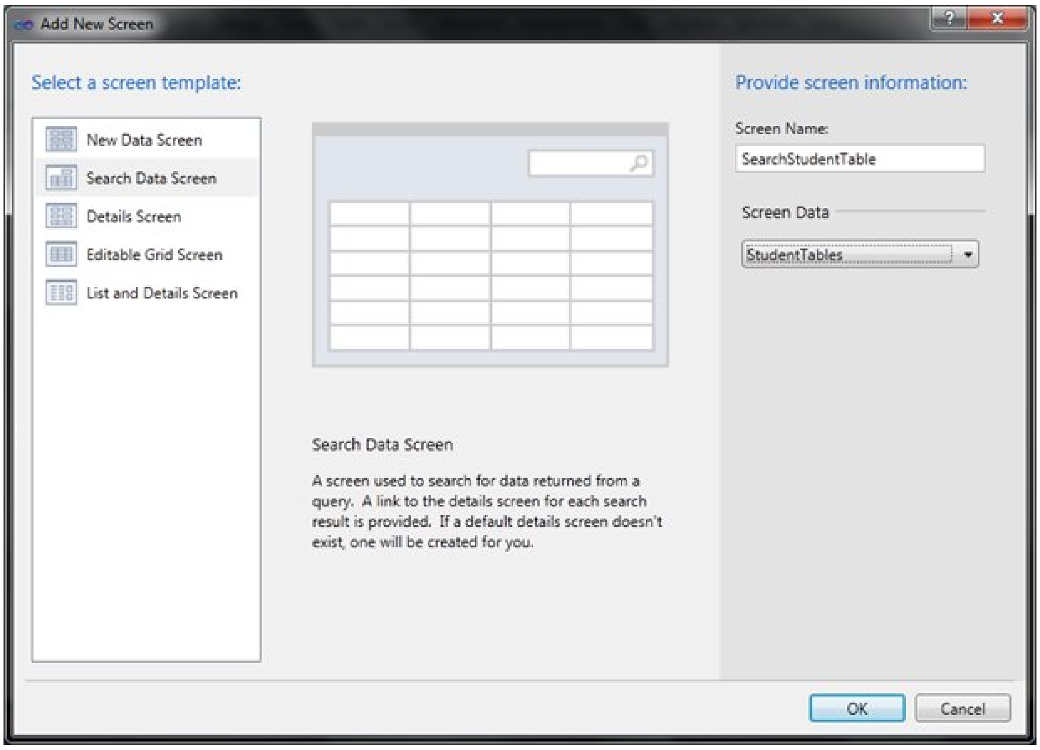
Screen Name: Search

Screen Data: (None)

Search Data Screen

A screen used to search for data returned from a query. A link to the details screen for each search result is provided. If a default details screen doesn't exist, one will be created for you.

In the right panel, select the appropriate “Screen Data” i.e. your table. In our case, it is the “StudentTables”. Select it from the dropdown. The “Screen Name” will be automatically updated by the IDE. If you want, you can change it here or can change it later from the Screen Designer window.



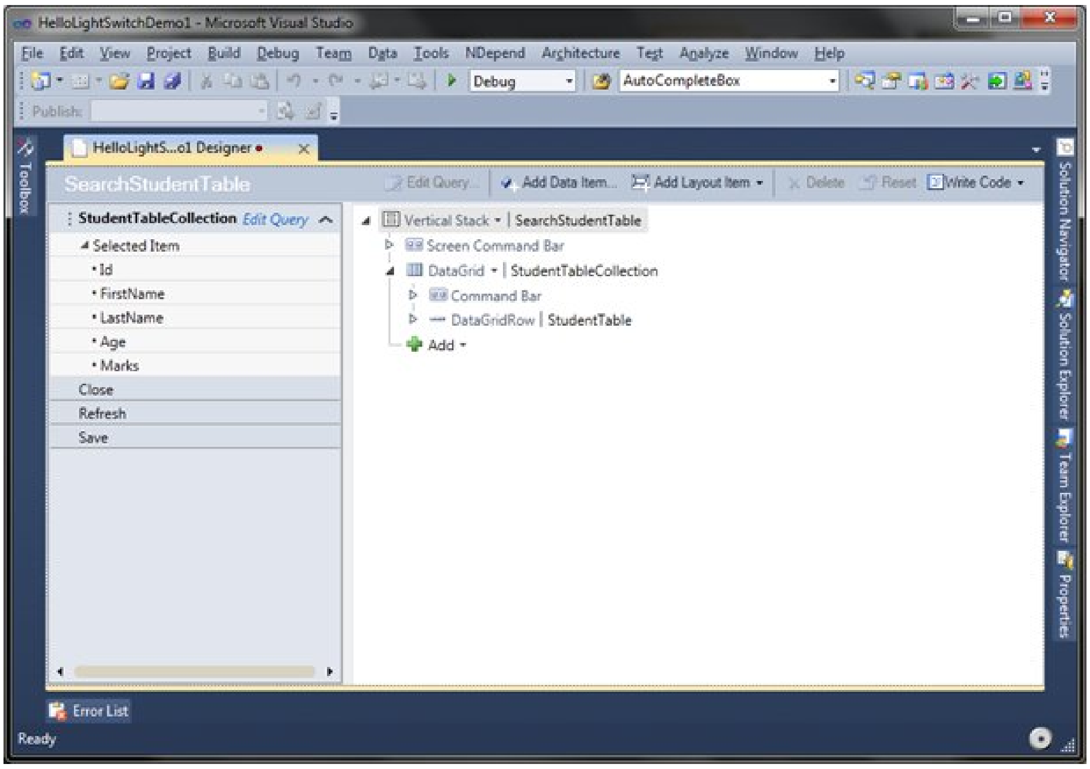
Screen Name: SearchStudentTable

Screen Data: StudentTables

Search Data Screen

A screen used to search for data returned from a query. A link to the details screen for each search result is provided. If a default details screen doesn't exist, one will be created for you.

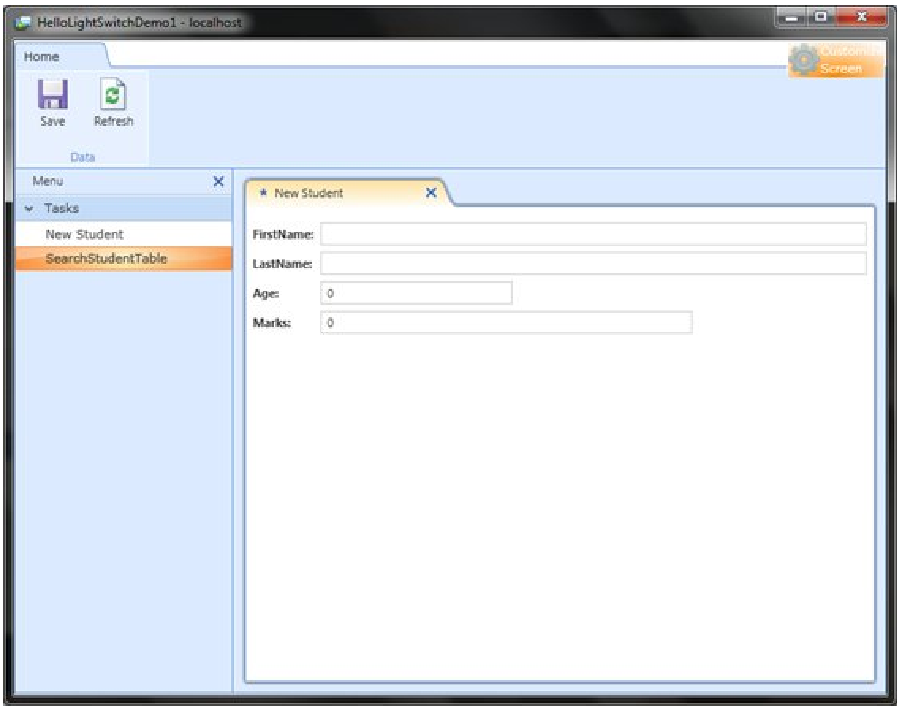
Click “Ok” to continue. This will generate the appropriate Search screen for your Student Table. Here it is in design view:


HelloLightSwitchDemo1 - Microsoft Visual Studio
File Edit View Project Build Debug Team Data Tools NDepend Architecture Test Analyze Window Help
Debug
AutoCompleteBox
Publish:
Toolbox
HelloLightS...o1 Designer
SearchStudentTable
Edit Query...
Add Data Item...
Add Layout Item
Delete
Reset
Write Code
StudentTableCollection Edit Query
Selected Item
Id
FirstName
LastName
Age
Marks
Close
Refresh
Save
Vertical Stack | SearchStudentTable
Screen Command Bar
DataGrid | StudentTableCollection
Command Bar
DataGridRow | StudentTable
Add
Solution Navigator
Solution Explorer
Team Explorer
Properties
Error List
Ready

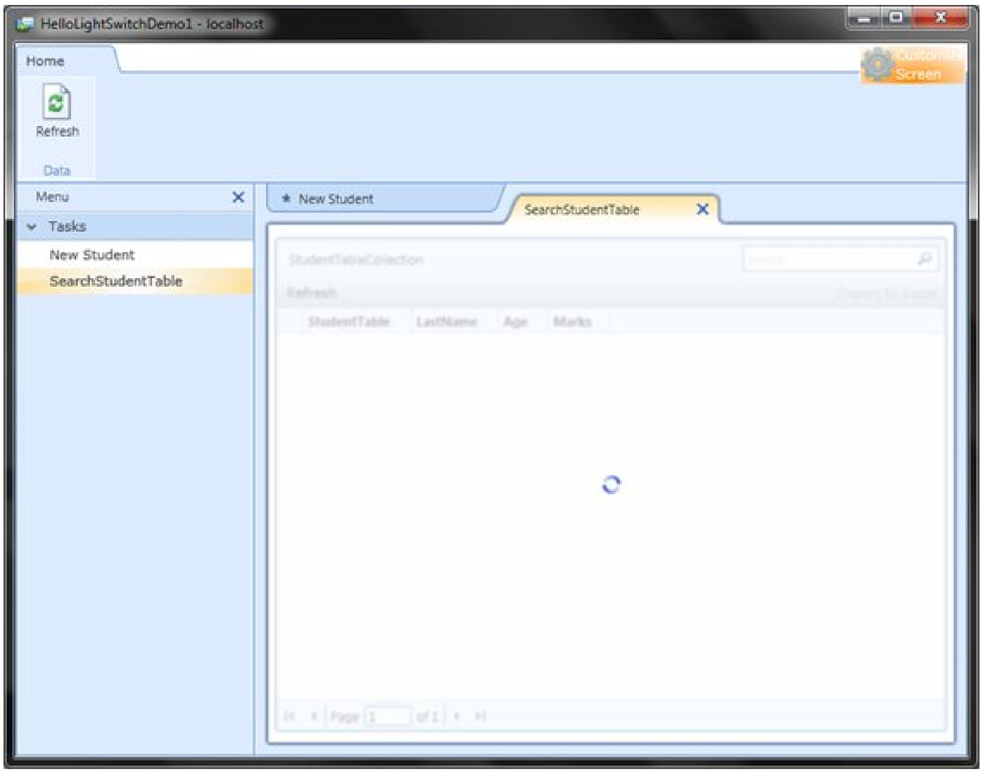
We will not modify anything there now. We will customize it at later point of time. As of now, our search screen is ready. We can now run the application and see the live demo.

See the Application in Action

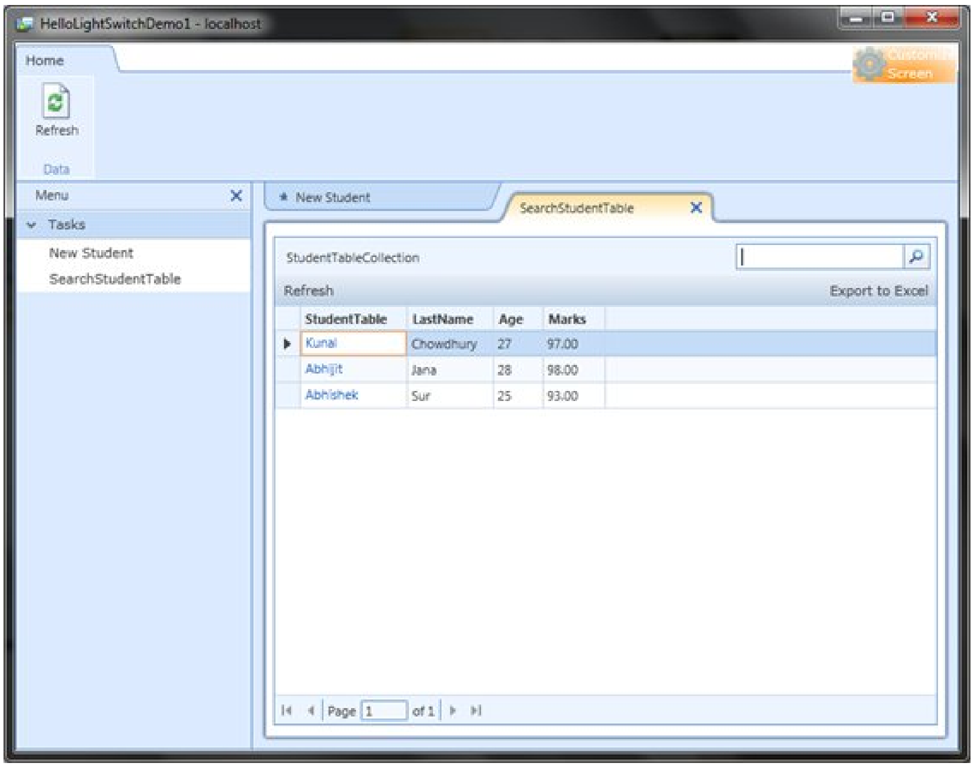
Once the screen is ready, press F5 to run the application. Once the application has been launched as the OOB Silverlight Window, you will see the below screen:



By default the first screen (i.e. the “New Student” Screen in our demo application) will be visible. From the left panel, select the menu item called “SearchStudentTable”. This will open the “SearchStudentTable” screen as a new Tab. You will also see a loader while fetching the table values.

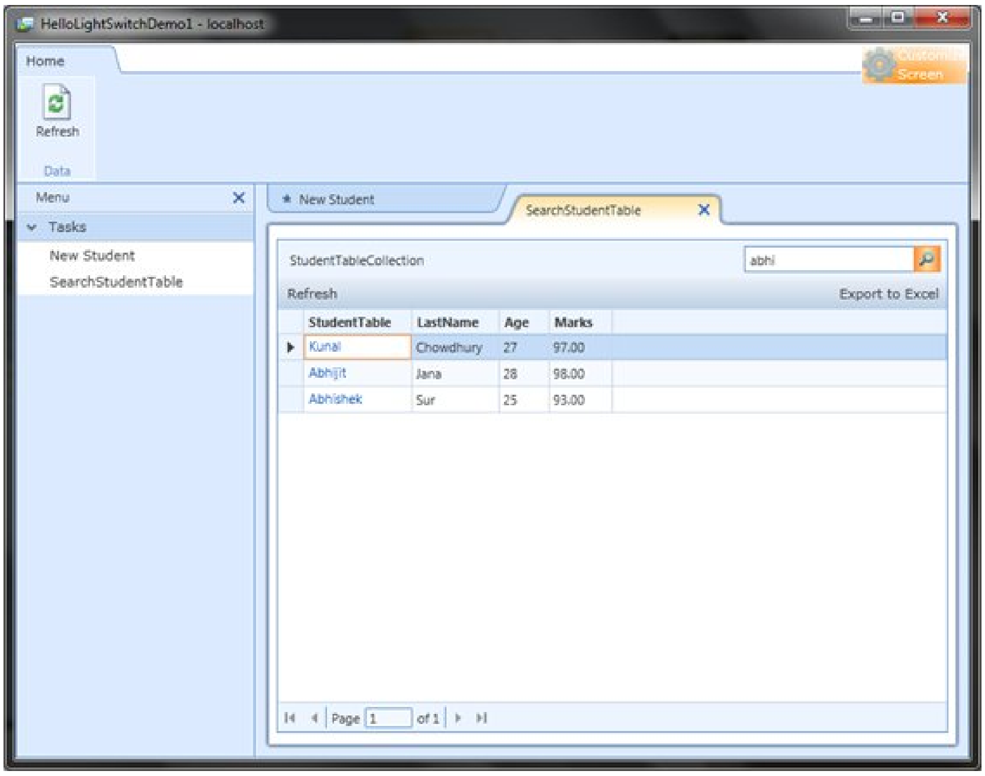


Once the fetch has been done, you will see the screen loaded with all the records in a Data Grid. In the same screen you will see a “Refresh” button, a “Search Panel” and a button called “Export to Excel”.



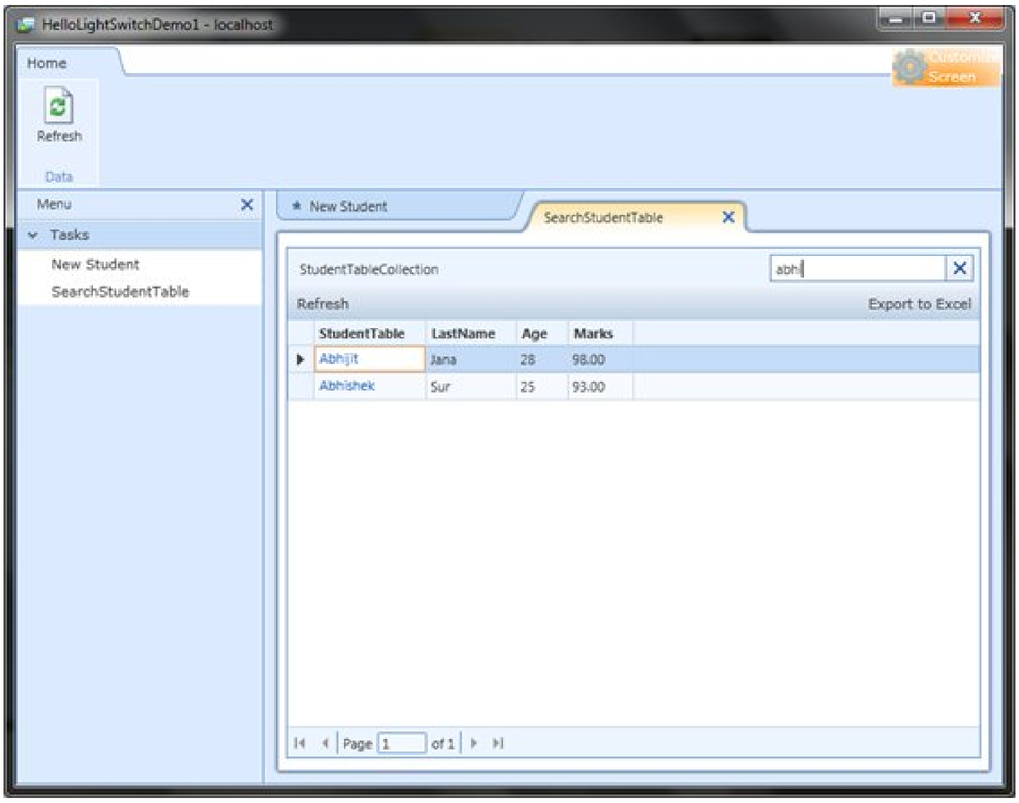
StudentTable	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00

Now, click on the Search box and start typing your search query. Once entered, click on the search button.



StudentTable	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00

This will filter out the DataGrid with the respective result based on the search term. You can see the same as live here:

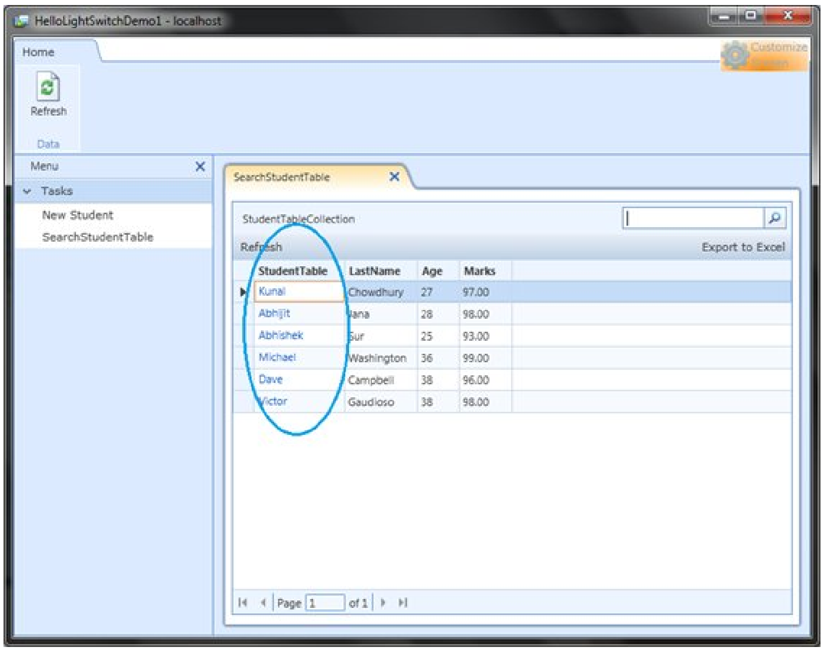


StudentTable	LastName	Age	Marks
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00

All the things demonstrated here is without writing a single piece of code. You can see that. It is really very easy to create the professional application using the LightSwitch within a small amount of time. No need to know more about Silverlight XAML.

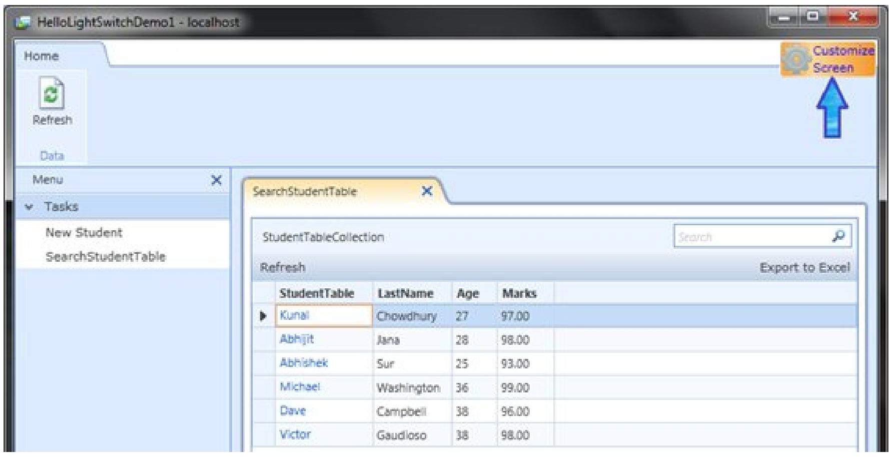
Customizing the Search Screen

If you look into the UI properly, you will notice that the table has a wrong column name. The first column name is "Student Table". This is nothing but the "FirstName" column of the Student Table. So, we need to modify the UI to show the proper column name. In this point, we will discuss about it. Add some more records into the database table.



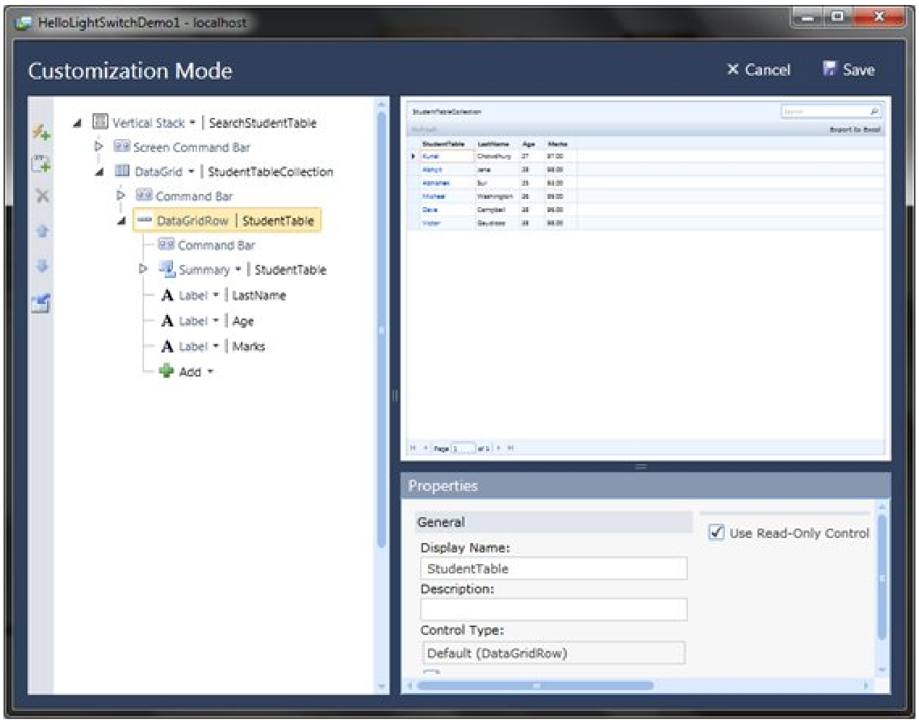
StudentTable	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00

At the top right corner, you will see a "Customize Screen" button. You can click it to start customizing the Search Screen. Be sure that, the search screen is open in the UI. Click the button to continue.

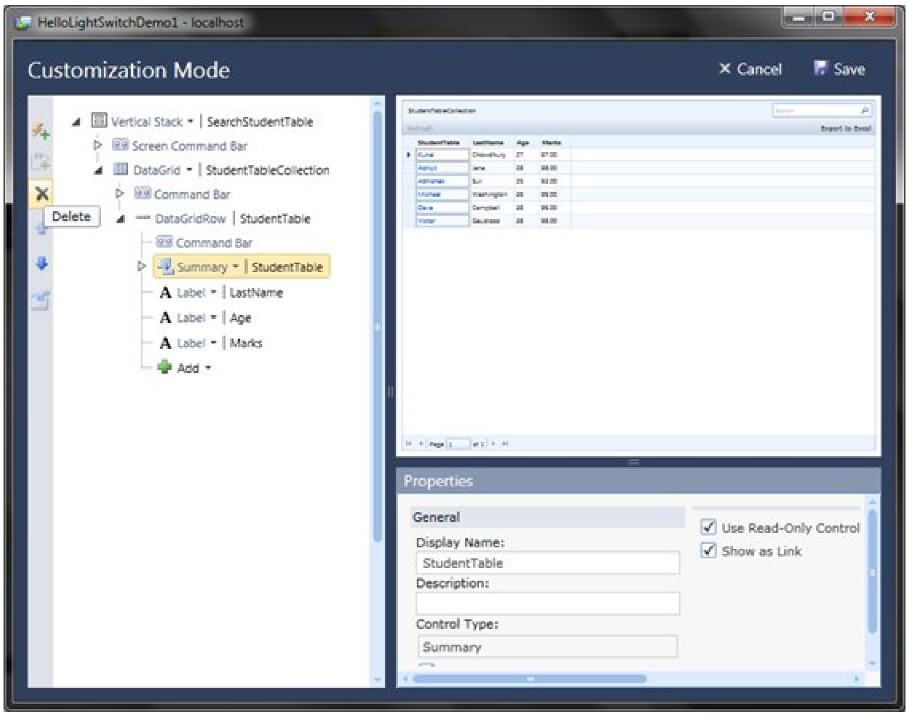


StudentTable	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00

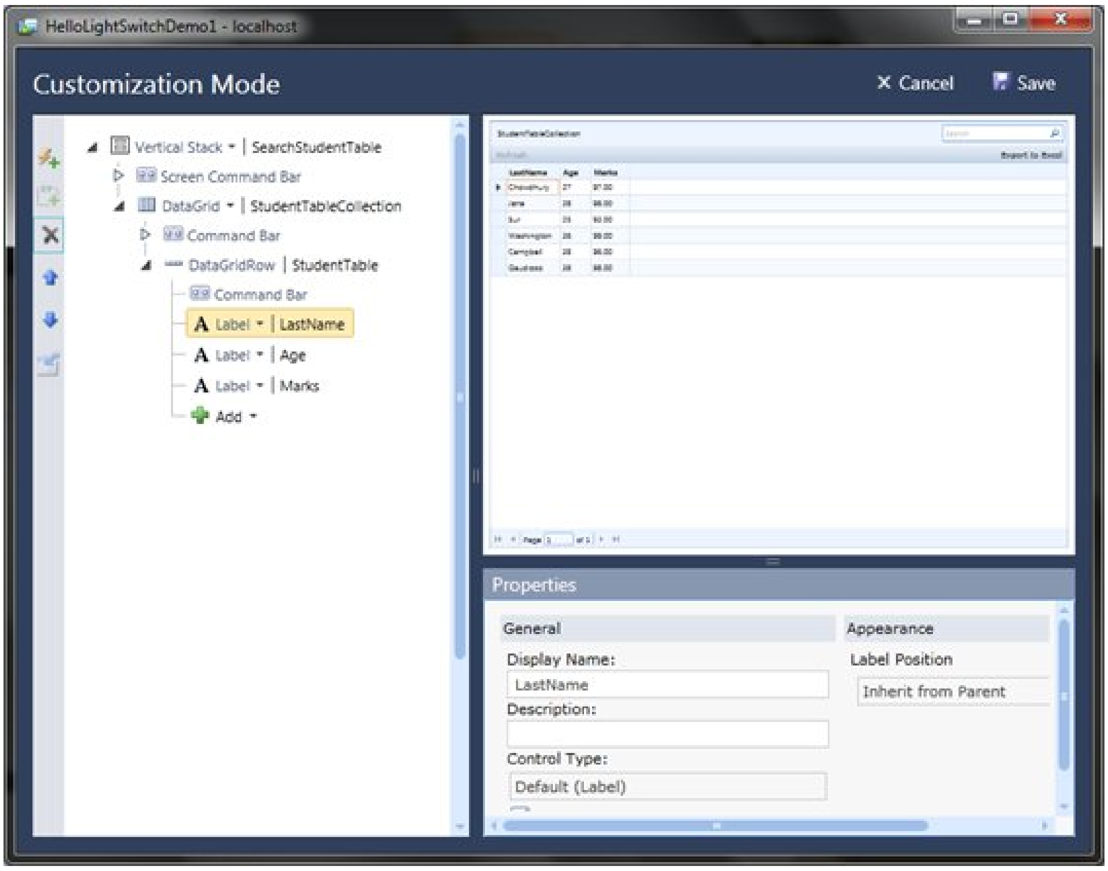
This will open the Search Screen in Customization Mode. We are familiar with this screen, right? We have already seen this screen in our first chapter. To modify the first column of the DataGrid, click the small arrow head near to the "DataGridRow | StudentTable" to expand it.



Once expanded, you will see a Tree item named “Summary” (as shown in the below screenshot). Click on it. Now click on the “Delete” button placed at the edge of the left panel.

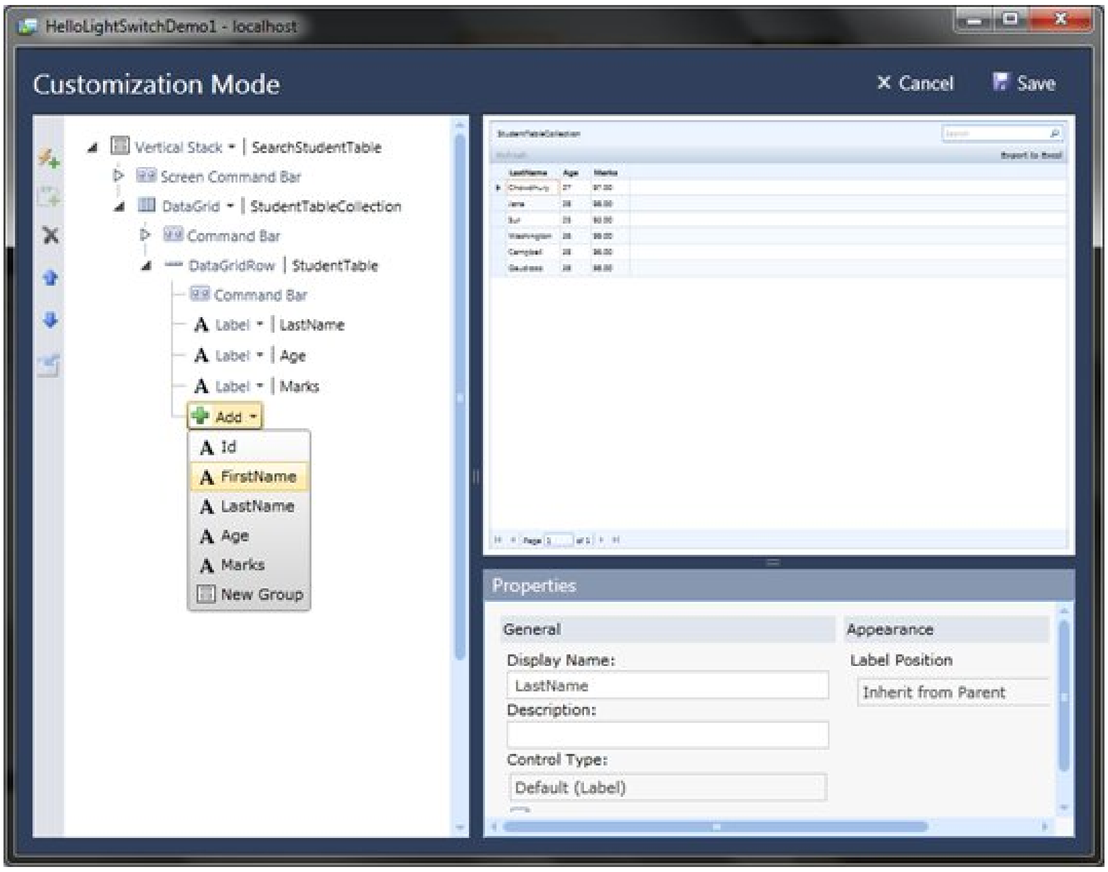


This will delete the summary column from the screen. You can see that in the right panel. Now select any of the label (as shown in the below screen) under the DataGridRow placed in left panel. You will see a dropdown menu added at the below screen.



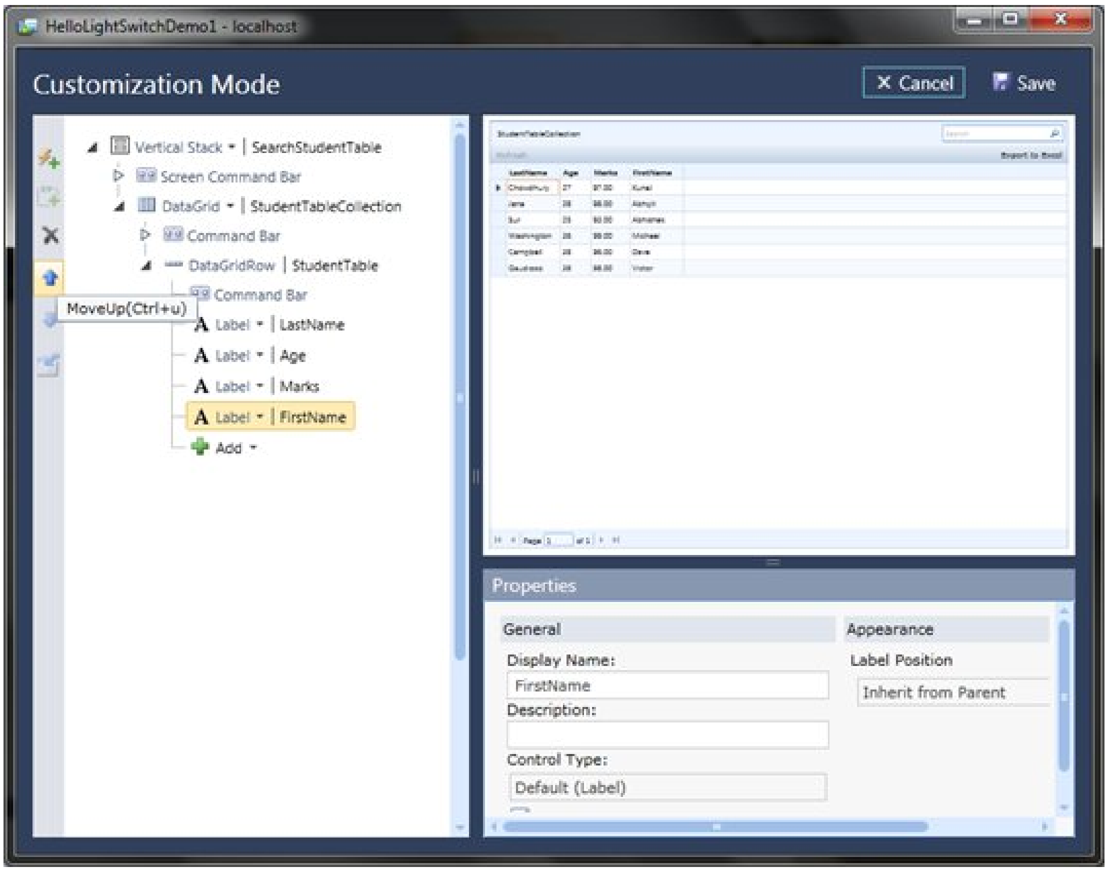
Display Name: LastName
Description:
Control Type: Default (Label)
Label Position: Inherit from Parent

Now click on the “Add” menu. This will pop up the full menu items in the screen. Select “FirstName” from the menu.



Display Name: LastName
Description:
Control Type: Default (Label)
Label Position: Inherit from Parent

Once selected, you will see it live in the right panel too. Click on MoveUp icon at the edge of the left panel or press Ctrl + u. Continue pressing the MoveUp button unless it moves up to the top.



- Vertical Stack | SearchStudentTable
  - Screen Command Bar
  - DataGrid | StudentTableCollection
    - Command Bar
    - DataGridRow | StudentTable
      - Command Bar
      - Label | LastName
      - Label | Age
      - Label | Marks
      - Label | FirstName
      - Add

StudentTableCollection

LastName	Age	Marks	FirstName
Chowdhury	27	97.00	Kunal
Jena	28	98.00	Abhijit
Sur	25	90.00	Abhishek
Washington	28	99.00	Michael
Campbell	28	96.00	Dave
Gaudreau	28	98.00	Victor

General

Display Name: FirstName

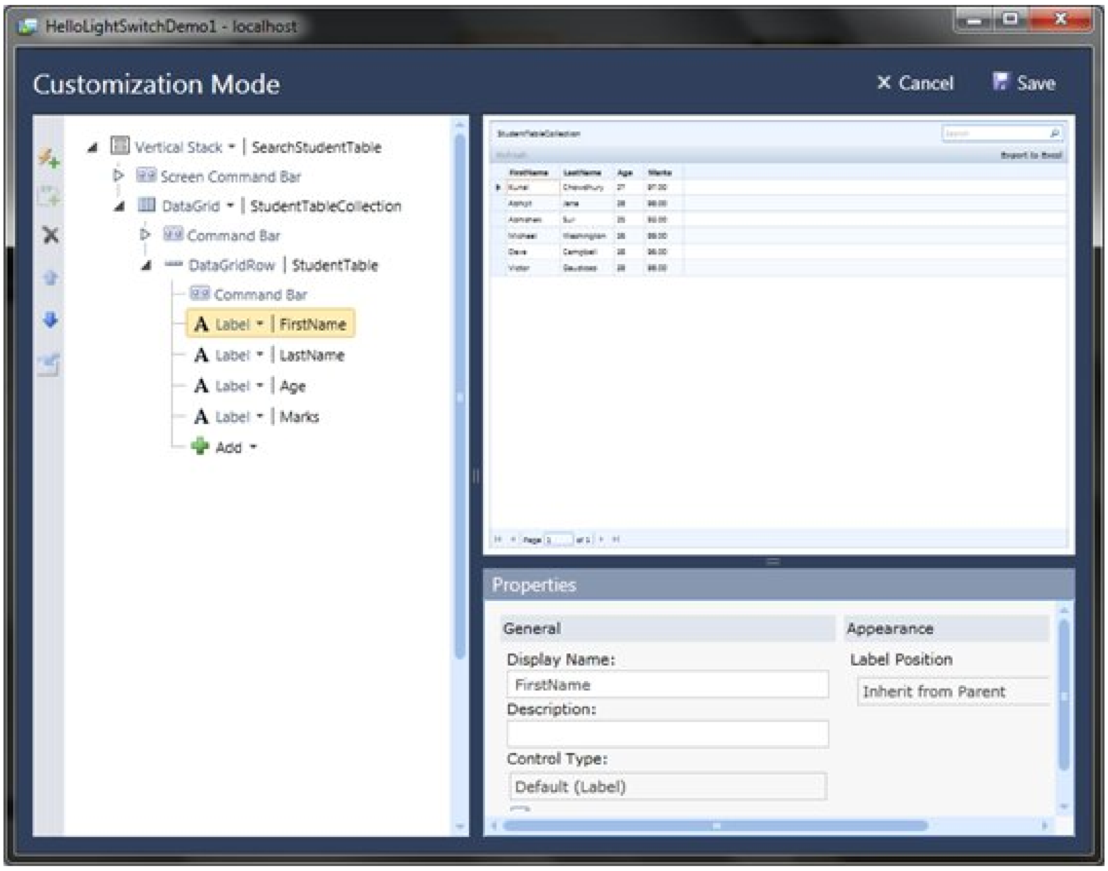
Description:

Control Type: Default (Label)

Appearance

Label Position: Inherit from Parent

Now you will see the DataGrid structured properly with FirstName, LastName, Age and Marks.



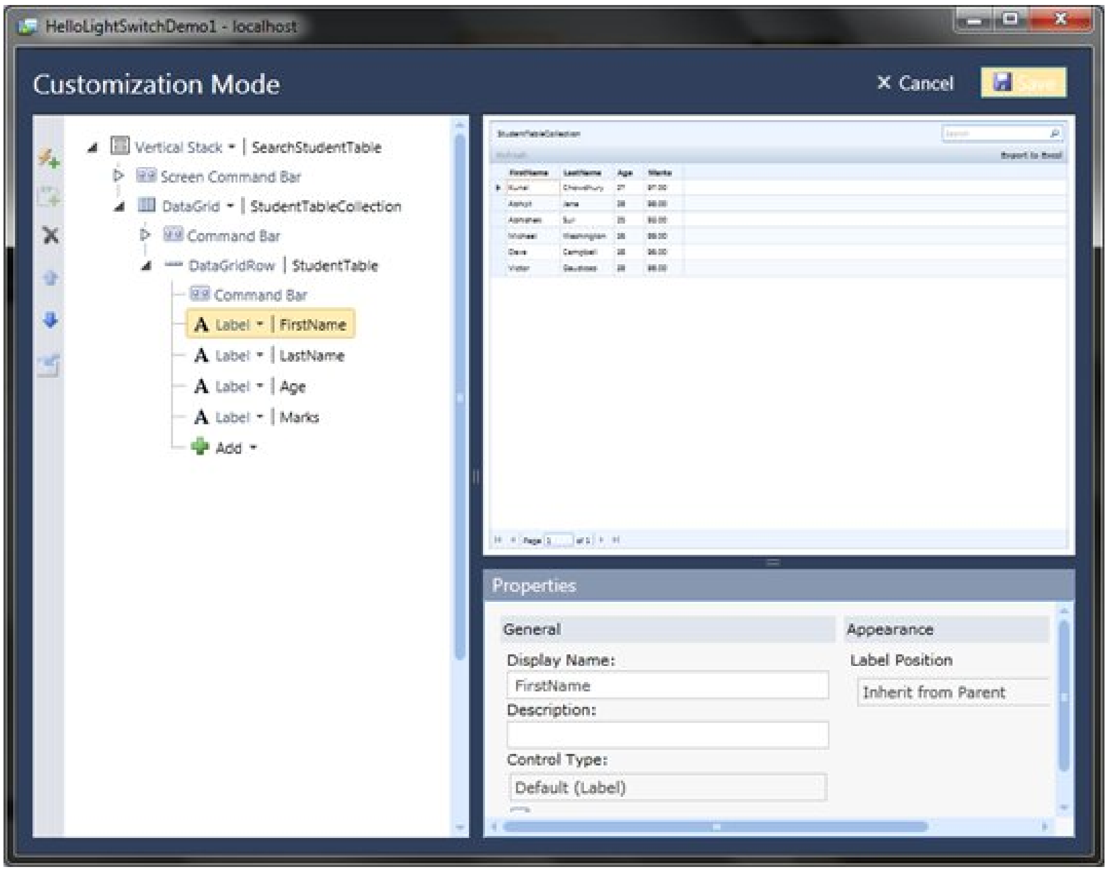
Display Name: FirstName

Description:

Control Type: Default (Label)

Label Position: Inherit from Parent

Click on “Save” button to save the changes.



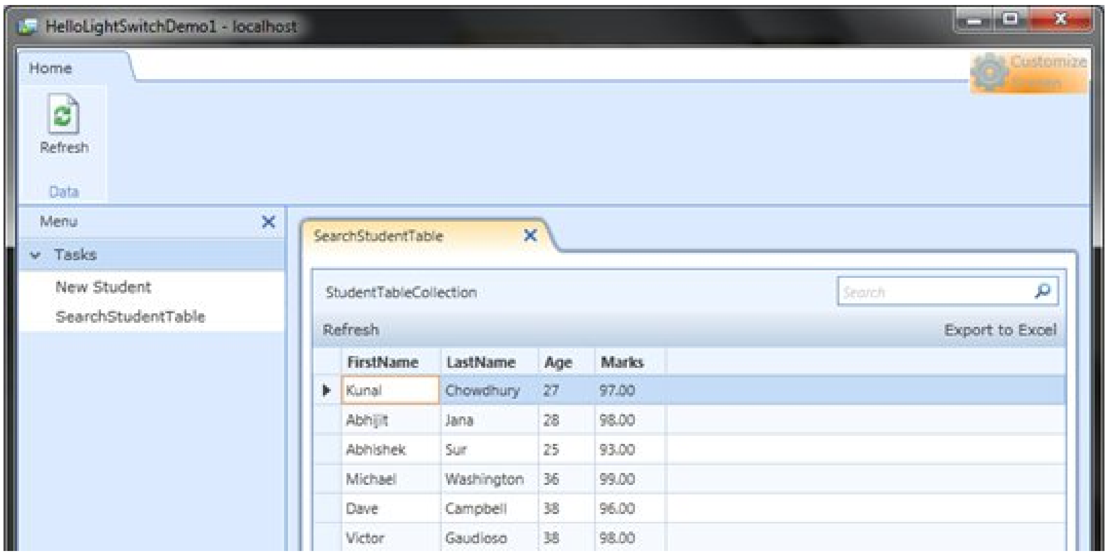
Display Name: FirstName

Description:

Control Type: Default (Label)

Label Position: Inherit from Parent

Now run your application to see the changes live. Here is the snapshot of the same:



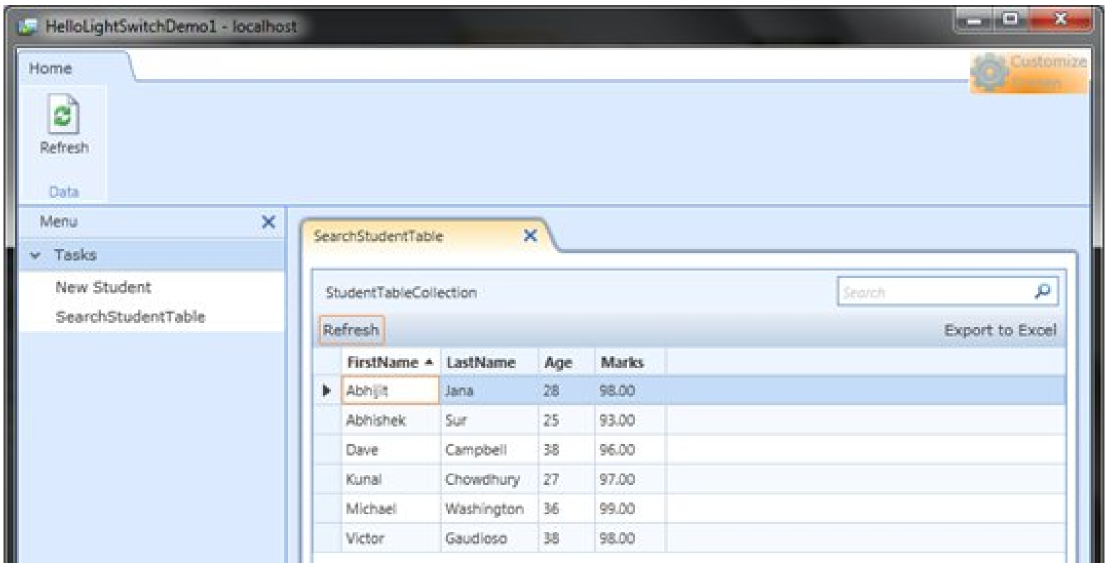
FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00

Here you will see the records shown as per the order of insertion in the table. They are based on the Record Id. If you want to show the Id in the UI, go to the design view and add the Id column into the Screen.

Sorting the Records

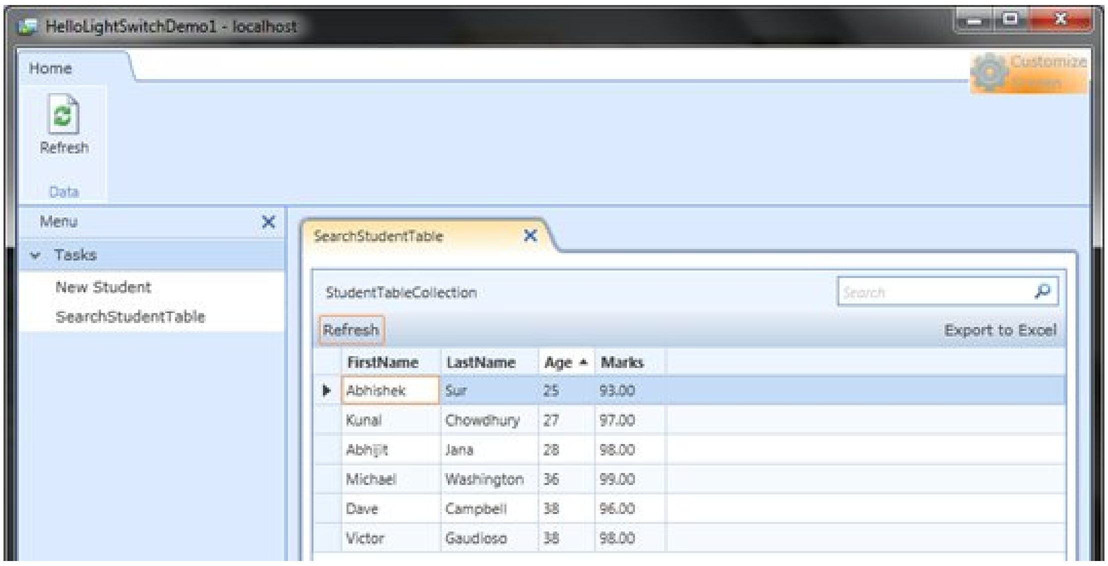
If you want to sort the records, you can easily do it. The DataGrid has the mechanism to do it automatically. If you want to sort the data based on a particular column values, click on that column to sort it.

Let us sort the records by the FirstName field. Hence, click on the FirstName column header. Woo, what did you see? The records has been automatically sorted on FirstName.



FirstName ▲	LastName	Age	Marks
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Dave	Campbell	38	96.00
Kunal	Chowdhury	27	97.00
Michael	Washington	36	99.00
Victor	Gaudioso	38	98.00

Let us do another sort based on the “Age” field. Click on the “Age” column header to do the Sort by the application.



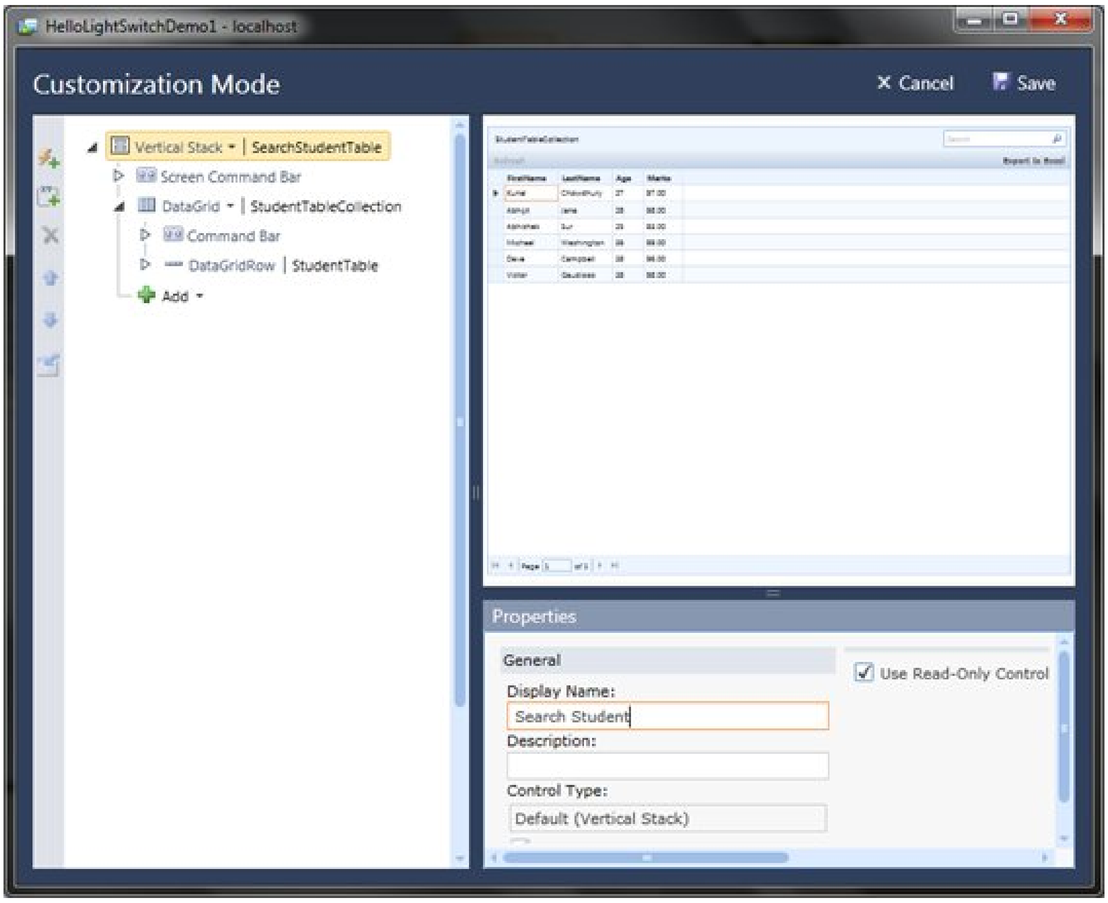
FirstName	LastName	Age	Marks
Abhishek	Sur	25	93.00
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00

If you want to sort the records for the other fields, click on the respective column header to sort.

Customize the Name of the Screen

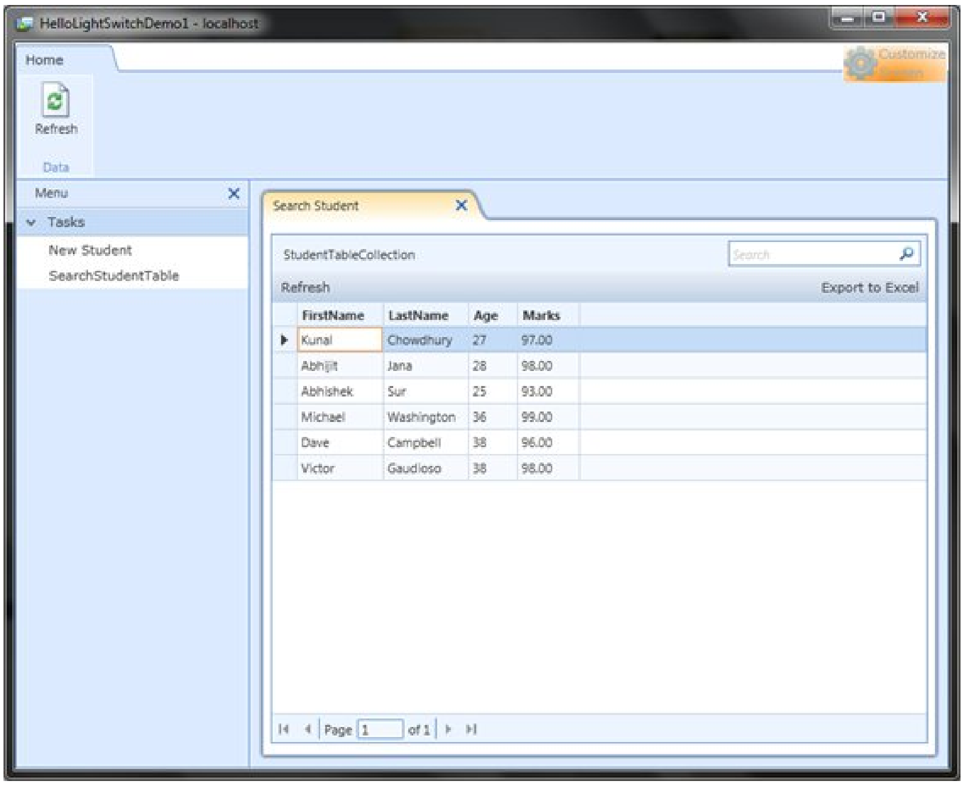
This point is already covered in the previous chapter. For sake of your understanding, I am again repeating this point. To change the title of the Search screen, click on the "Customize Screen" button to open up the screen in Customization Mode.

Select the first item from the left panel. In this example, it is: "Vertical Stack | SearchStudentTable".


HelloLightSwitchDemo1 - localhost
Customization Mode
Cancel
Save
Vertical Stack | SearchStudentTable
Screen Command Bar
DataGrid | StudentTableCollection
Command Bar
DataGridRow | StudentTable
Add
Properties
General

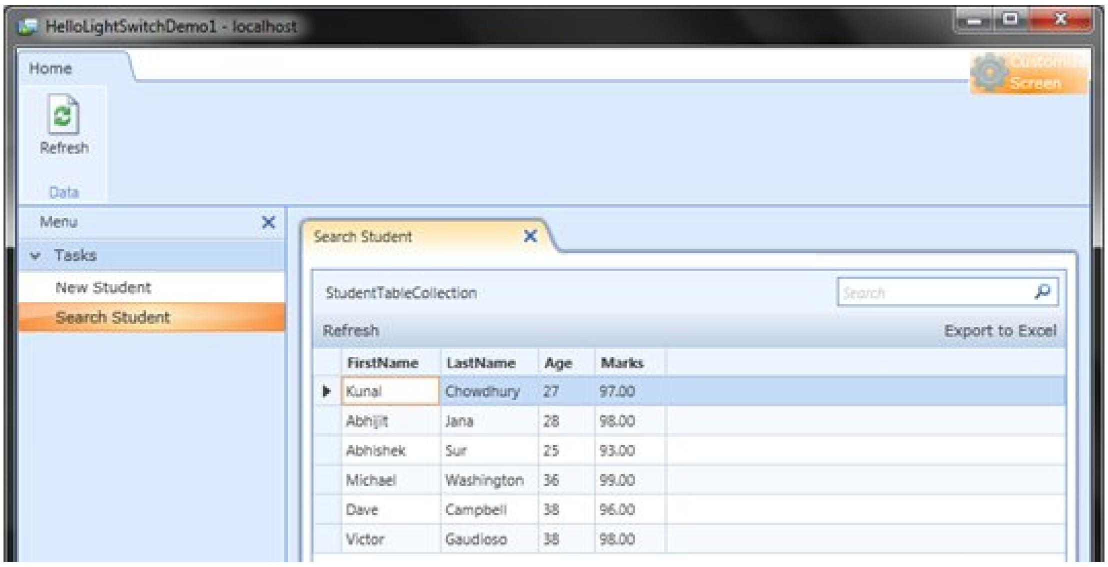
Display Name: Search Student
Description:
Control Type: Default (Vertical Stack)
☑ Use Read-Only Control

In the bottom right panel, find out the field named “Display Name”. Change the value for it from “SearchStudentTable” to “Search Student”. Save the settings.When you back to the UI screen, you will see that it has been updated in the tab. But in the left menu, the text is not yet updated.



	FirstName	LastName	Age	Marks
▶	Kunal	Chowdhury	27	97.00
	Abhijit	Jana	28	98.00
	Abhishek	Sur	25	93.00
	Michael	Washington	36	99.00
	Dave	Campbell	38	96.00
	Victor	Gaudioso	38	98.00

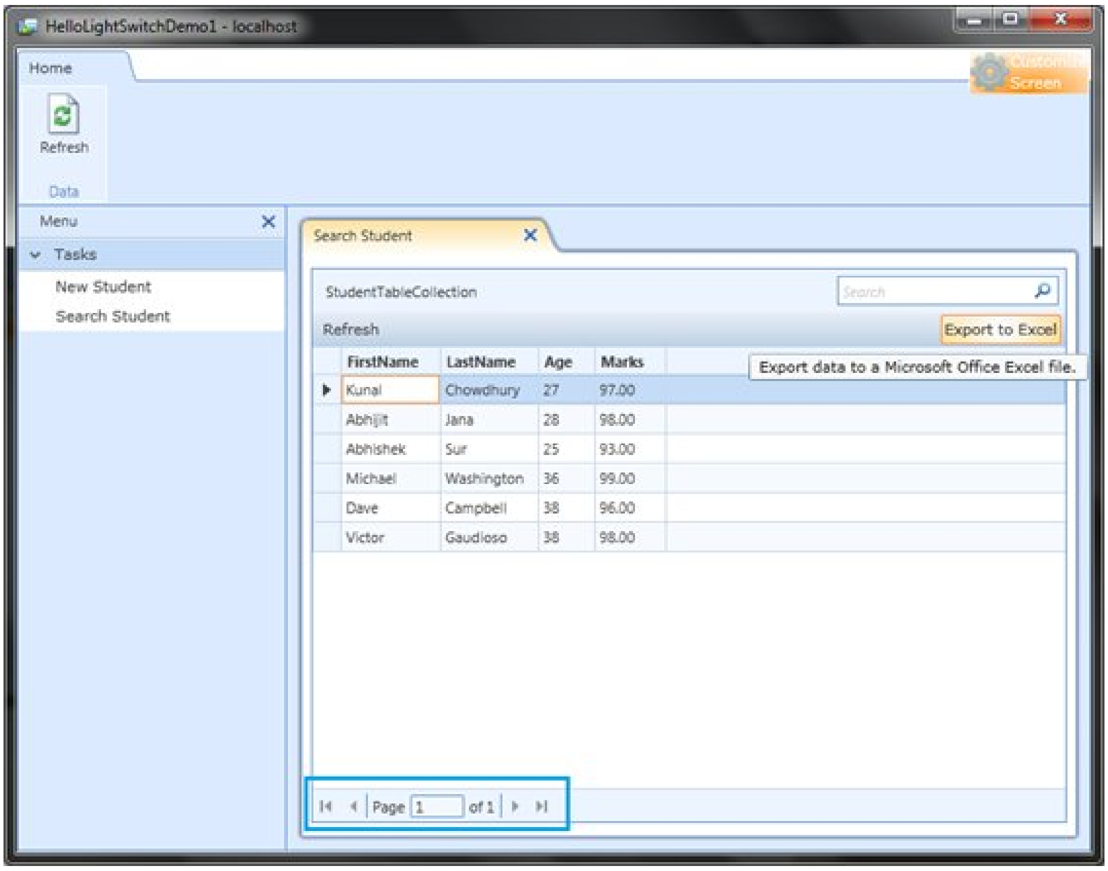
Restart the application to reflect all the changes. After restart, your application will look as below:



	FirstName	LastName	Age	Marks
▶	Kunal	Chowdhury	27	97.00
	Abhijit	Jana	28	98.00
	Abhishek	Sur	25	93.00
	Michael	Washington	36	99.00
	Dave	Campbell	38	96.00
	Victor	Gaudioso	38	98.00

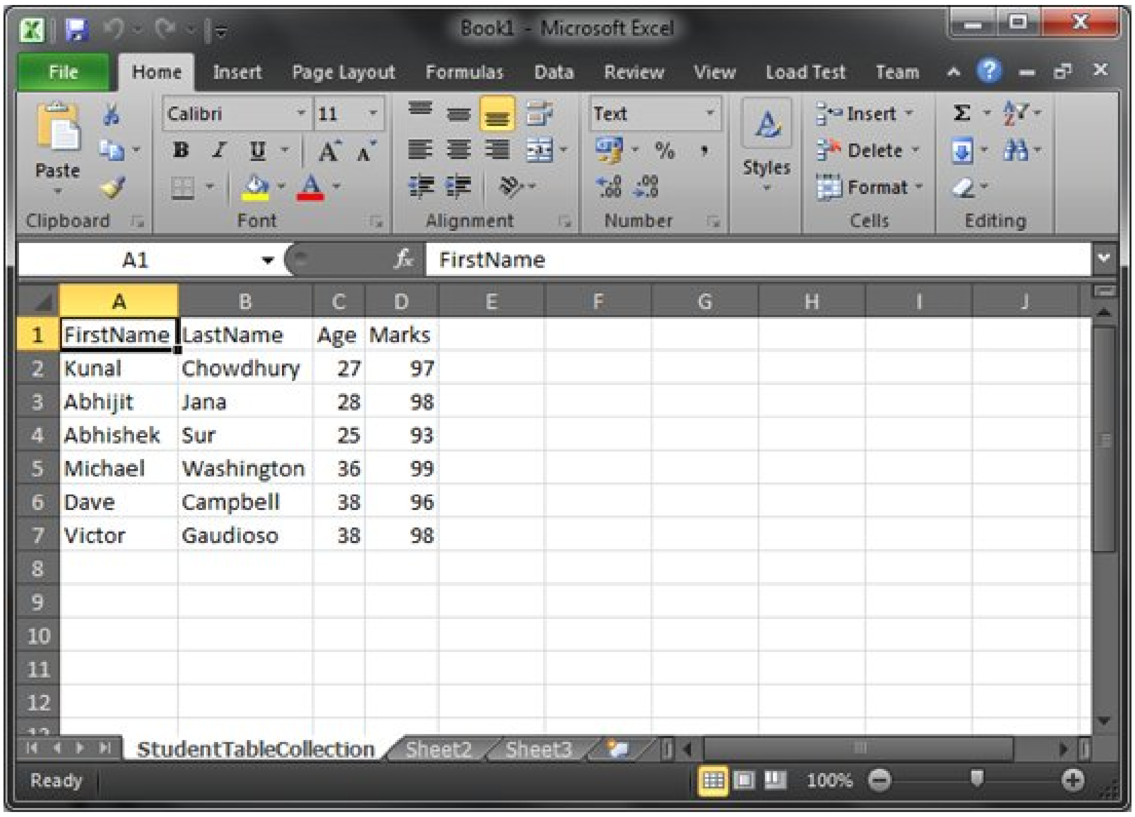
Navigation and Export to Excel

If you have a huge records in the table, you will see the navigation panel in action. See the below screenshot:



FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00

In the above screenshot you will see the “Export to Excel” button. This is nothing but exporting your records to the Excel Work Sheet automatically. Just click on the button. The whole content of the records will be exported to the Excel application and you will see it like this:



FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97
Abhijit	Jana	28	98
Abhishek	Sur	25	93
Michael	Washington	36	99
Dave	Campbell	38	96
Victor	Gaudioso	38	98

End Note

You can see that, for the whole sample application, I didn’t write a single piece of XAML or C# code. This is the power of Visual Studio LightSwitch for the Silverlight developers. If you haven’t yet started working on LightSwitch, try exploring it from now.

I am feeling it very useful, as I am exploring it now. I will post the next chapter soon for you, so that, you can learn about it in depth. Don’t forget to vote for it. Appreciate your feedbacks and/or suggestions.

Visual Studio LightSwitch is a new tool for building data-driven Silverlight Application using Visual Studio IDE. It automatically generates the User Interface for a DataSource without writing any code. You can write a small amount of code also to meet your requirement.

In my previous chapter “**Beginners Guide to Visual Studio LightSwitch (Part – 2)**”, I described you how to create a search record window & export the records to Excel Sheet using Visual Studio LightSwitch for Silverlight. I also demonstrated about sorting and navigating records without writing a single line of code.

In this chapter, I will guide you step-by-step to create a DataGrid of records. Here you will know, how to insert/modify/delete records. These all steps are without writing any code. We will use just the tool to improve our existing application.

Beginners Guide to Visual Studio LightSwitch (Part - 3)

Background

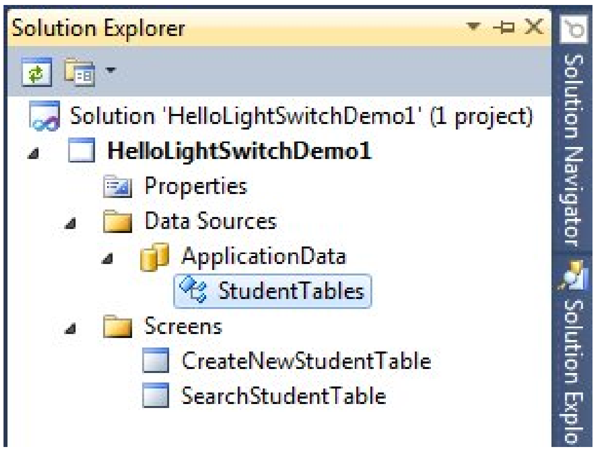
If you are new to Visual Studio LightSwitch, I will first ask you to read the first two chapter of this tutorial, where I demonstrated it in detail. In my 2nd chapter, I discussed the following topics:

- Create the Search Screen
- See the Application in Action
- Customizing the Search Screen
- Sorting the Records
- Customize the Name of the Screen
- Navigation and Export to Excel

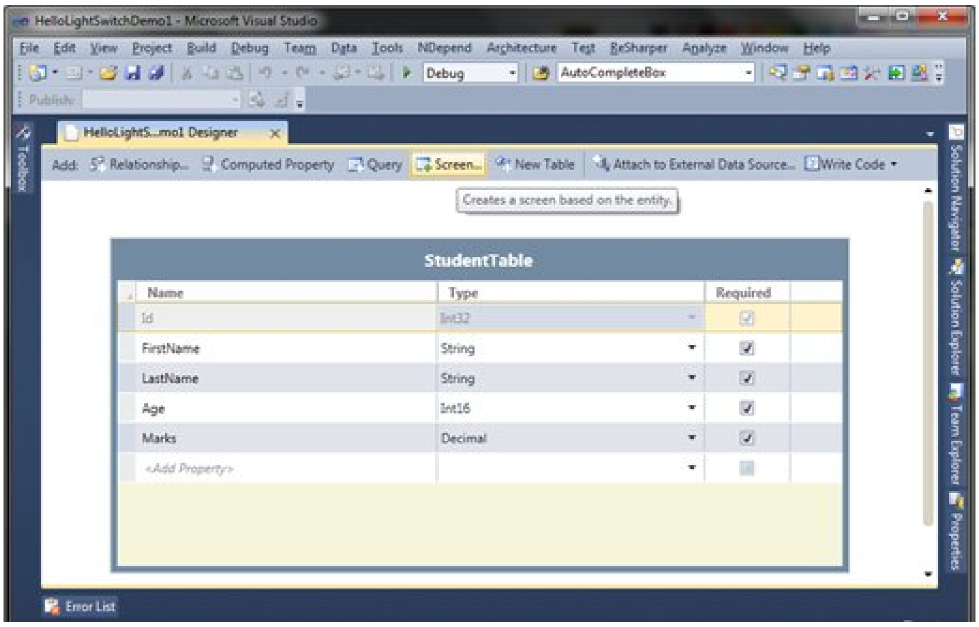
In this Chapter, we will discuss on the “Editable Data Grid” screen. Read it and learn more about this tool before it get release.

Create the Editable DataGrid Screen

Open the solution “HelloLightSwitchDemo1” that we created in our first chapter. From the Solution Explorer, double click to open the “StudentTables” as shown in the below figure:



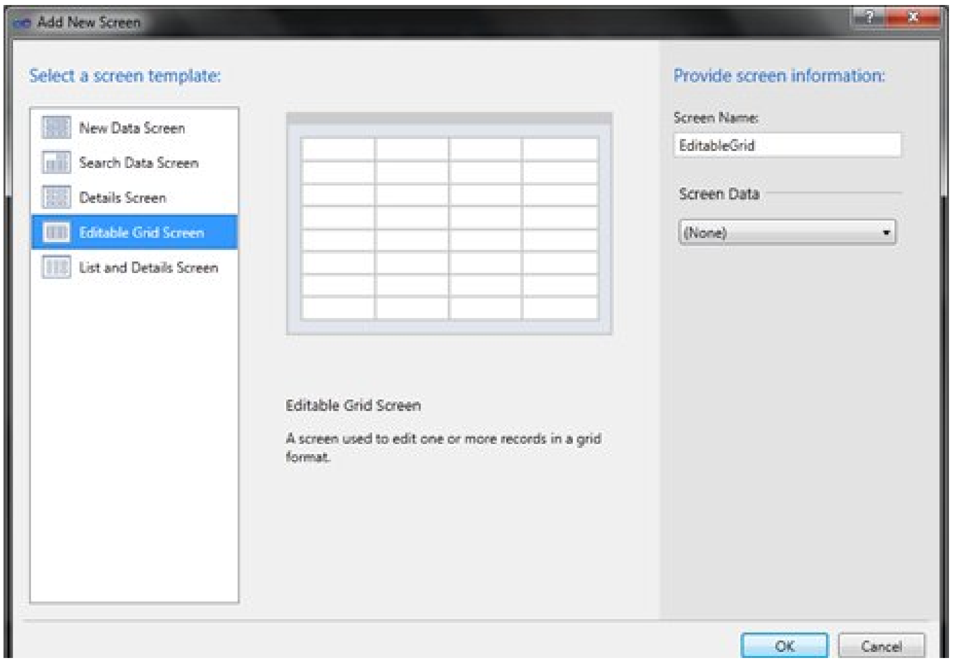
This will open the table in design view. Click on the button called “Screen” to create a screen based on the entity.


HelloLightSwitchDemo1 - Microsoft Visual Studio
File Edit View Project Build Debug Team Data Tools NDepend Architecture Test ReSharper Analyze Window Help
Debug
AutoCompleteBox
Publish:
HelloLightS...mo1 Designer
Add: Relationship... Computed Property Query Screen... New Table Attach to External Data Source... Write Code
Creates a screen based on the entity.
Toolbox
Solution Navigator
Solution Explorer
Team Explorer
Properties
Error List

StudentTable

Name	Type	Required
Id	Int32	✓
FirstName	String	✓
LastName	String	✓
Age	Int16	✓
Marks	Decimal	✓
<Add Property>		

This will open up the “Add New Screen” dialog in the screen. From the dialog shown below, select the “Editable Grid Screen” from the left panel. This will show the preview in the middle panel. The editable grid screen can be use to edit one or more records inside a datagrid.



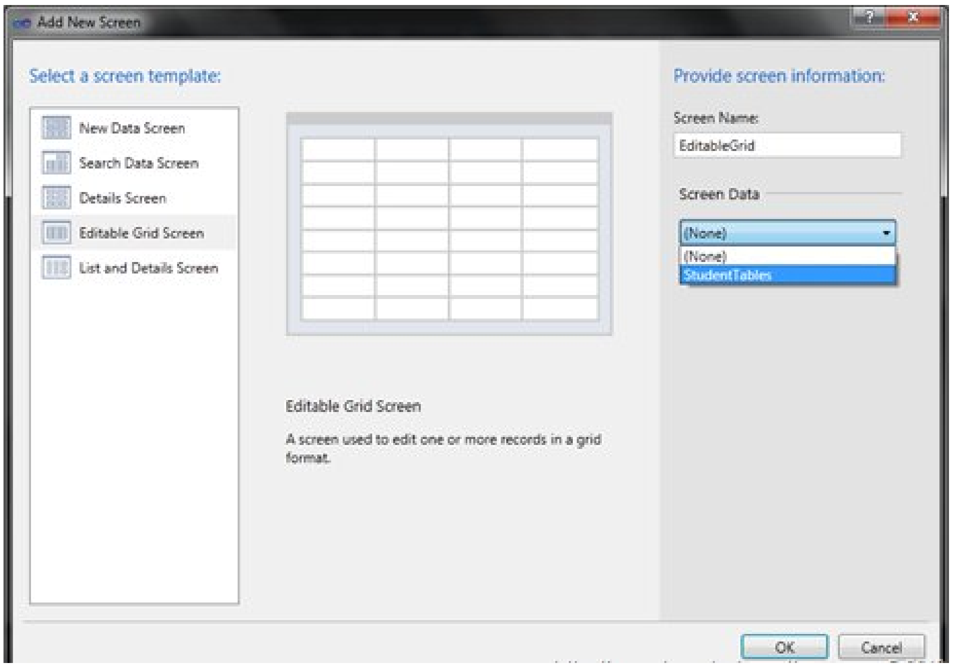
Editable Grid Screen

A screen used to edit one or more records in a grid format.

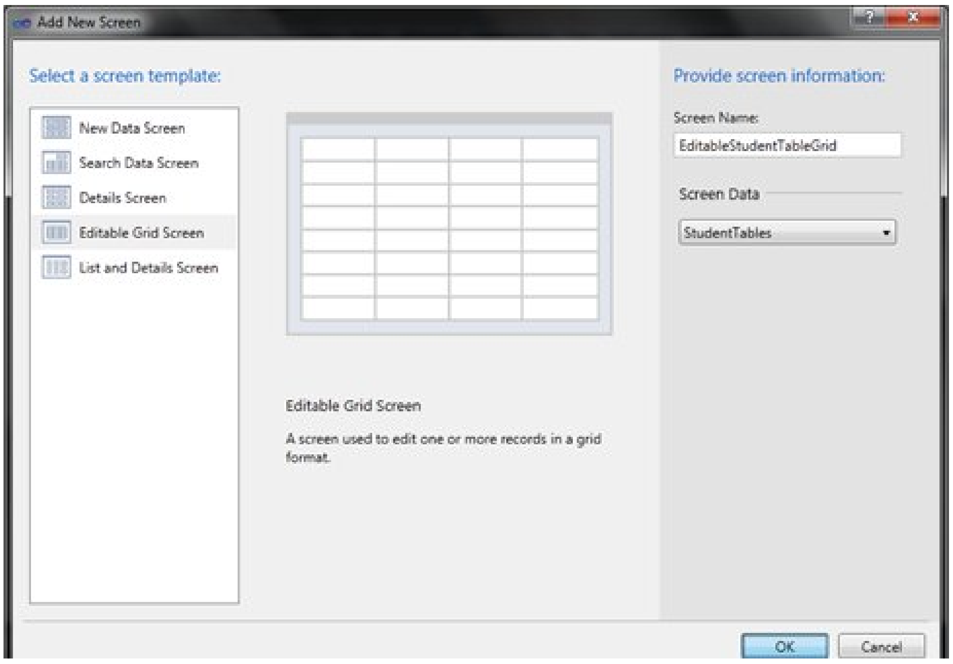
Screen Name: EditableGrid

Screen Data: (None)

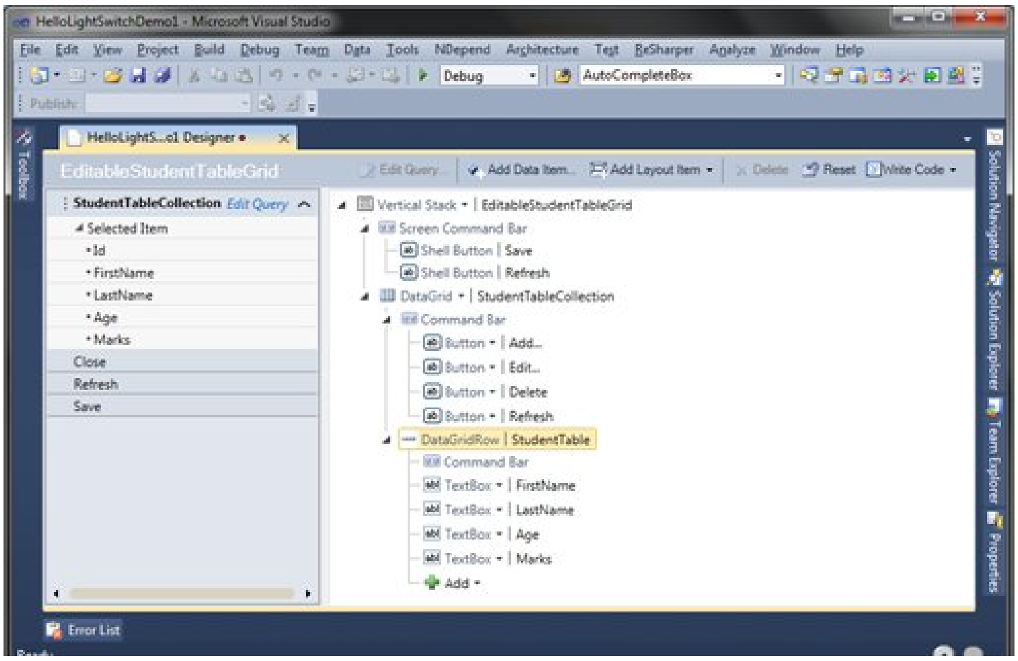
From the right panel, select the Screen Data i.e. the table you want to work with.



This will set the ScreenName in the same dialog. You can change it too here or can change it later as discussed in the earlier chapters. Currently, I am going with the default name. Click “OK” to continue creating the screen.



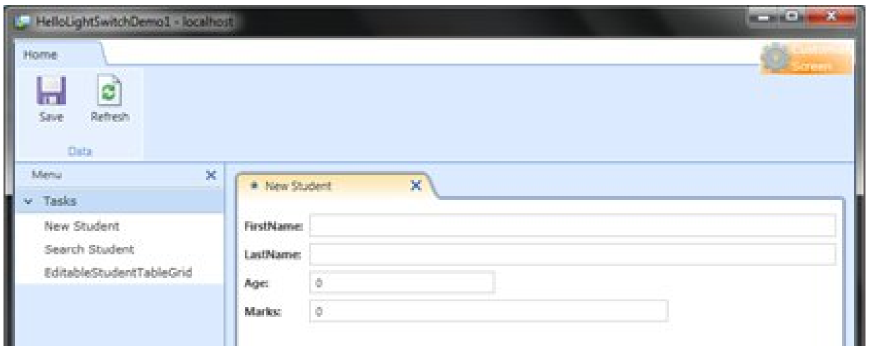
Once you click “OK”, it will close the dialog and open the designer window of the screen inside Visual Studio IDE. You can change something here. If you read the earlier chapters, you will familiar with this screen. Currently I am going with the default screen because I want you should understand it easily.



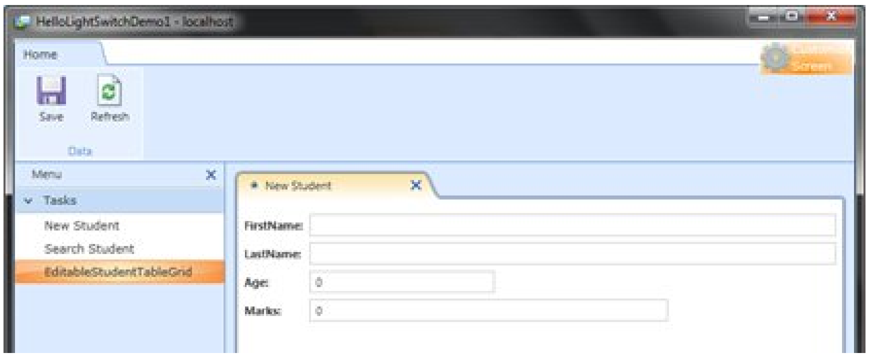
Once everything is done, run the application to see the screen in action.

See the Application in Action

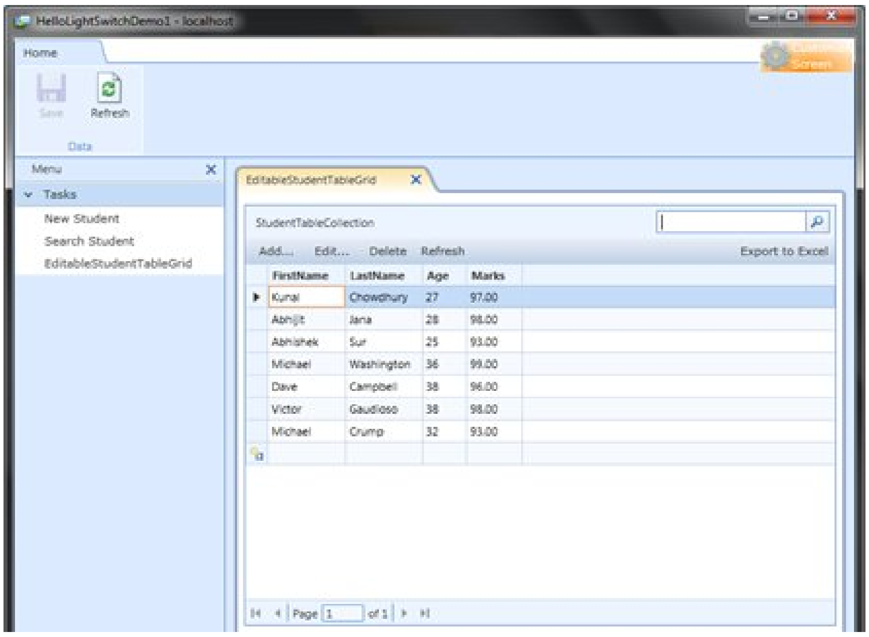
Now, it’s time to run the application to see the implemented screen. To do this, press F5. This will build the project and run the Silverlight application outside browser window (Out-of-Browser application). You will see the following UI once it run properly:



The default screen here will be the “New Student” tab. Close the tab and from the left panel click on the menu item called “EditableStudentTableGrid”.



It will open the tab ain to the screen s shown below:

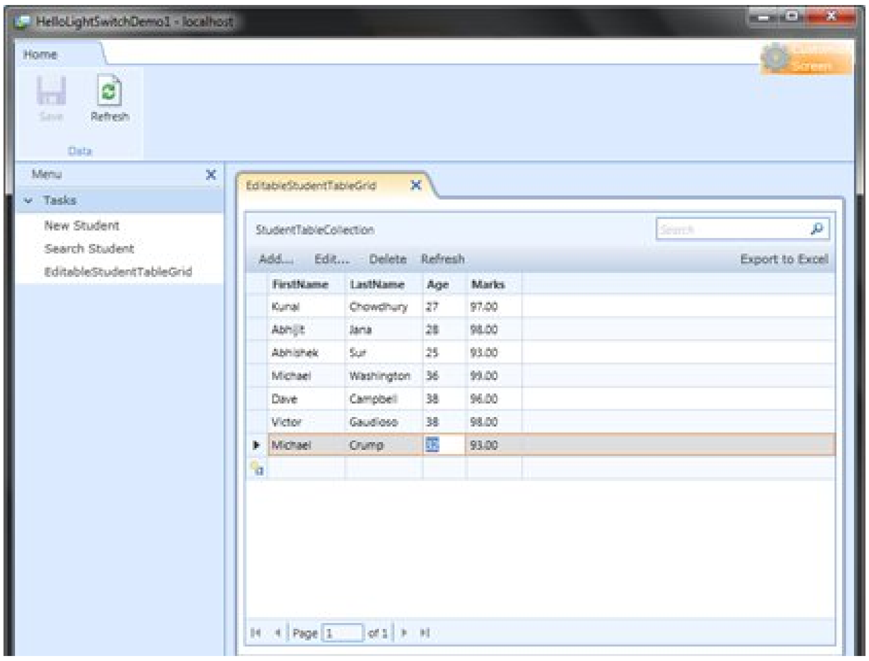


FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00
Michael	Crump	32	93.00

You can see that, the screen has one search textbox panel which actually filters the records. It has a command button panel to do Add, Edit, Delete and Refresh function. It has a button called “Export to Excel”. It exports the record to the excel application sheet. It has one navigation panel to paginate the records. Let’s discuss them one by one.

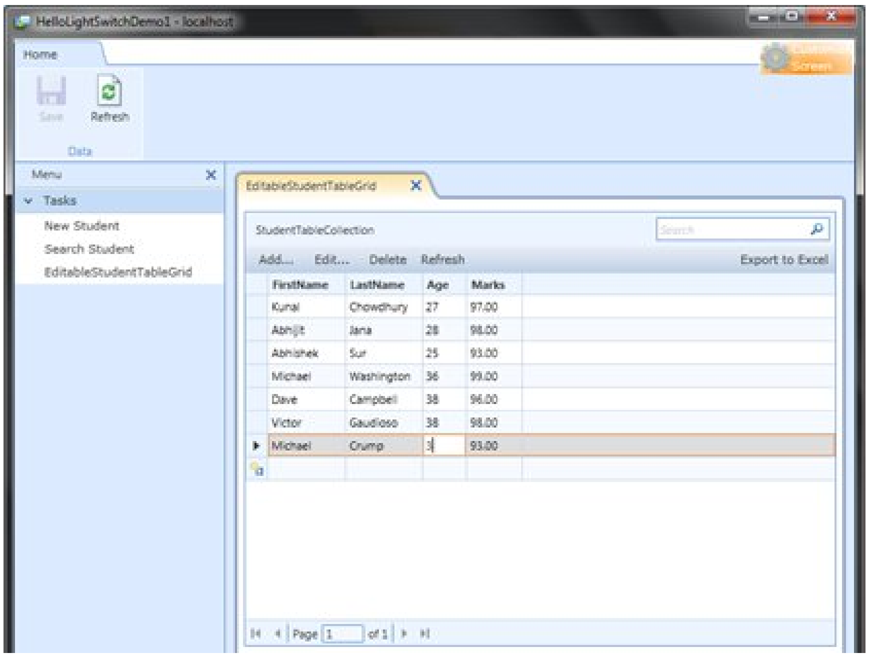
Edit a Record

Let us start discussing on how to edit a record first. I will show two different approach. Select any record and double click on the column that you want to modify. You will see that, the cell will become in editable mode and the value present in the cell will be selected automatically.



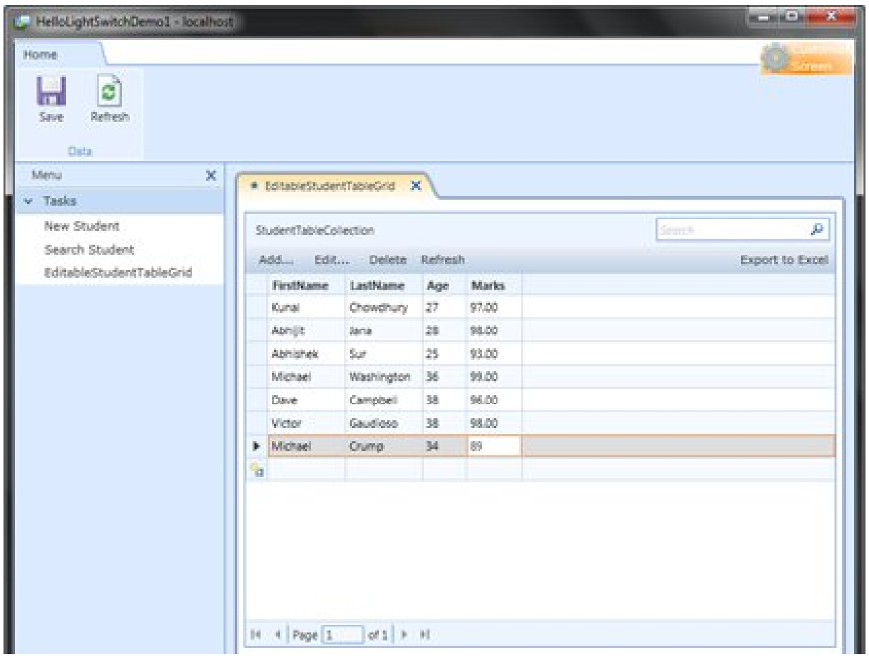
FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00
Michael	Crump	32	93.00

Now set a different value to the cell. Press TAB to modify another cell.



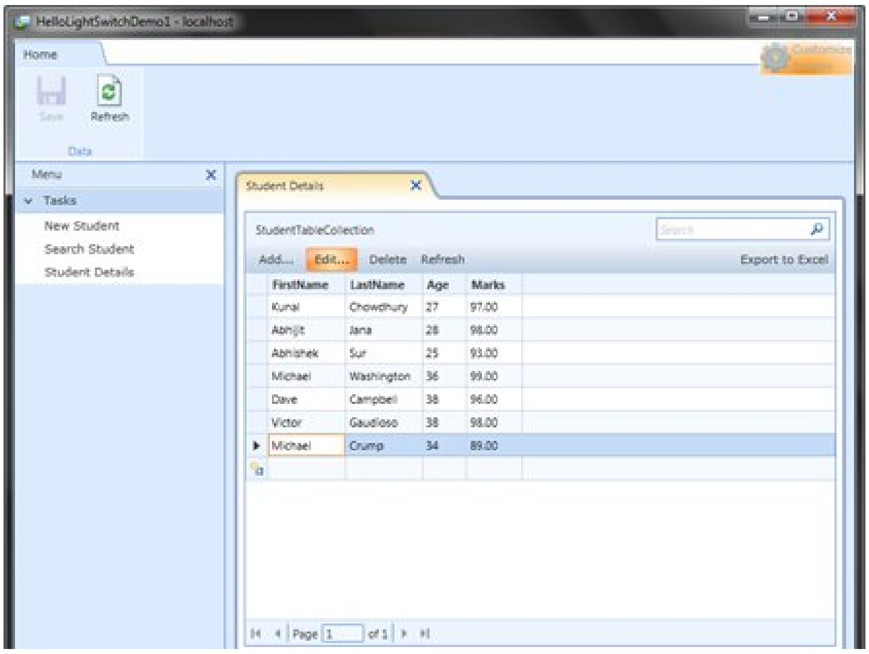
FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00
Michael	Crump	3	93.00

Once you done modifying your records, press the “Save” button placed at the top left corner of the screen. You can also press “CTRL + S” to save the modified records.



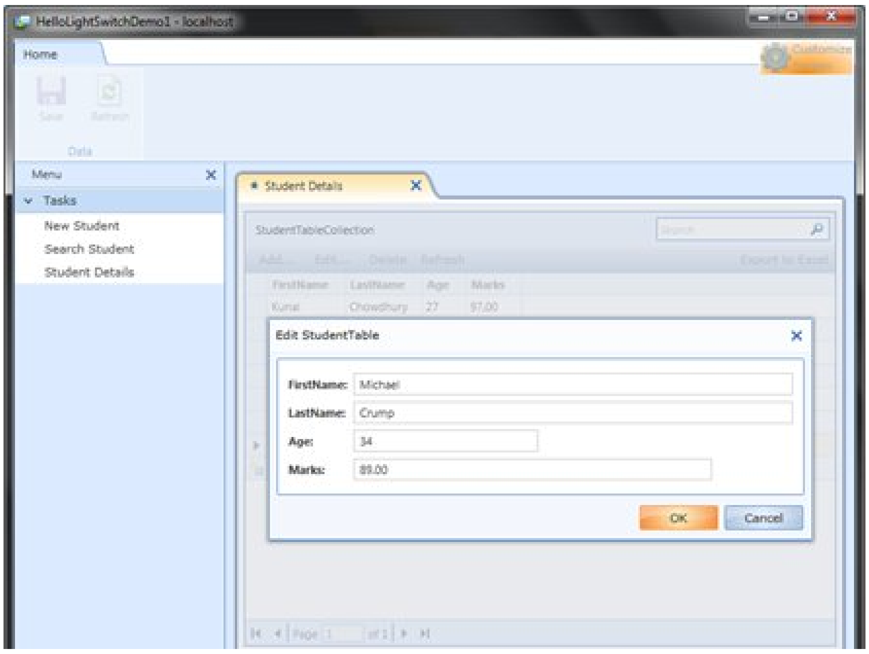
FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00
Michael	Crump	34	89

Now let’s see another approach to edit a record. Select the row that you want to modify and click on the button called “Edit…”.



FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00
Michael	Crump	34	89.00

This will bring a child window with all the fields related to the record. The input boxes will be auto field with the data records. Modify the fields that you want and click “OK”.

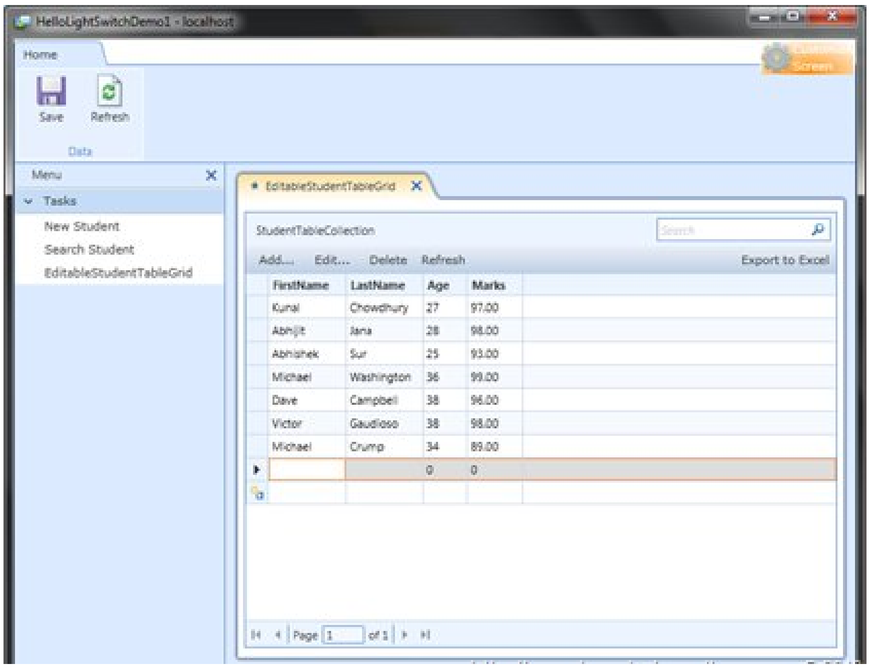


FirstName: Michael
LastName: Crump
Age: 34
Marks: 89.00

Now press “Save” to update the records in the database table.

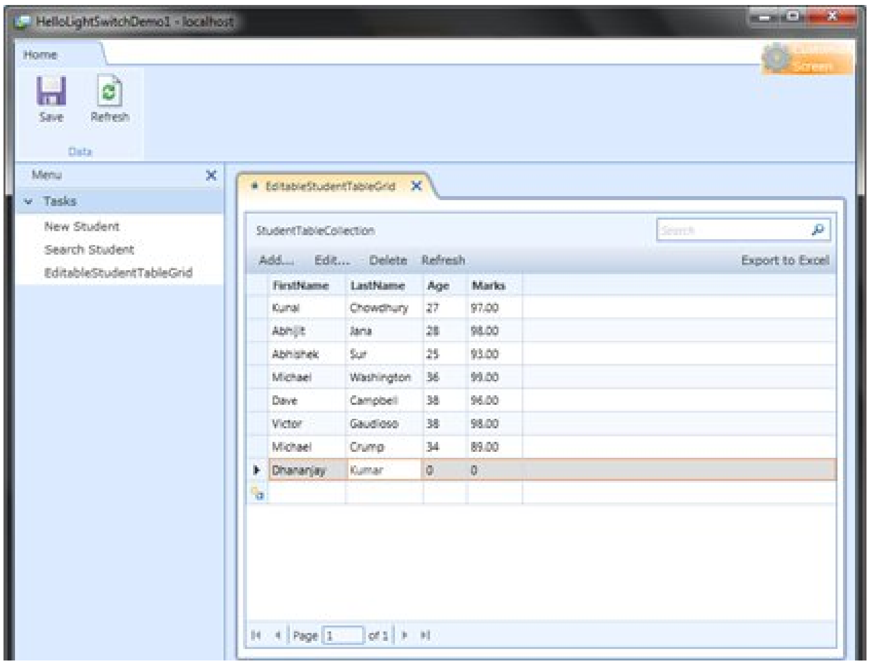
Create a New Record

We just completed editing the records. So, what next? Let’s create a new record from this screen. Here also, I will show you two different approach for inserting a new record.



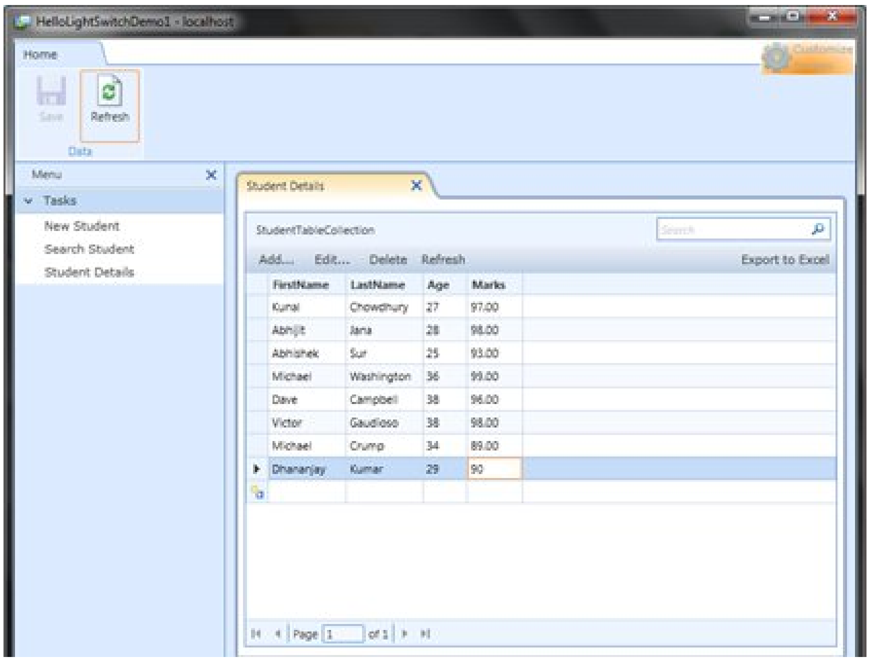
FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00
Michael	Crump	34	89.00
		0	0

In the first approach, click on a empty row and it will create a new row in the DataGrid as shown in the above UI. Start entering some records in the Grid.



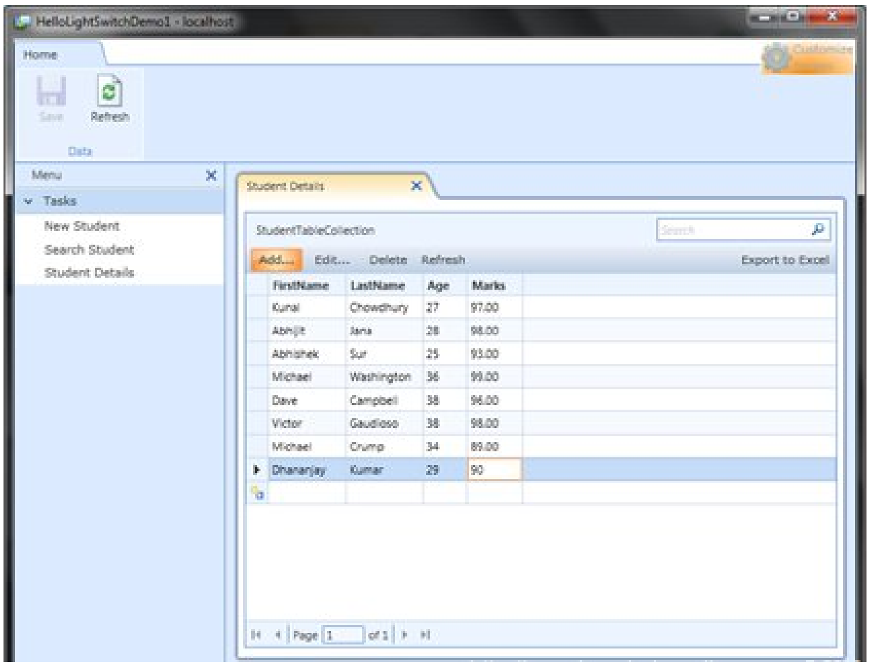
FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00
Michael	Crump	34	89.00
Dhananjay	Kumar	0	0

Fill up all the fields and press the Save button present at the top left corner of the screen. This will save the records to the table.



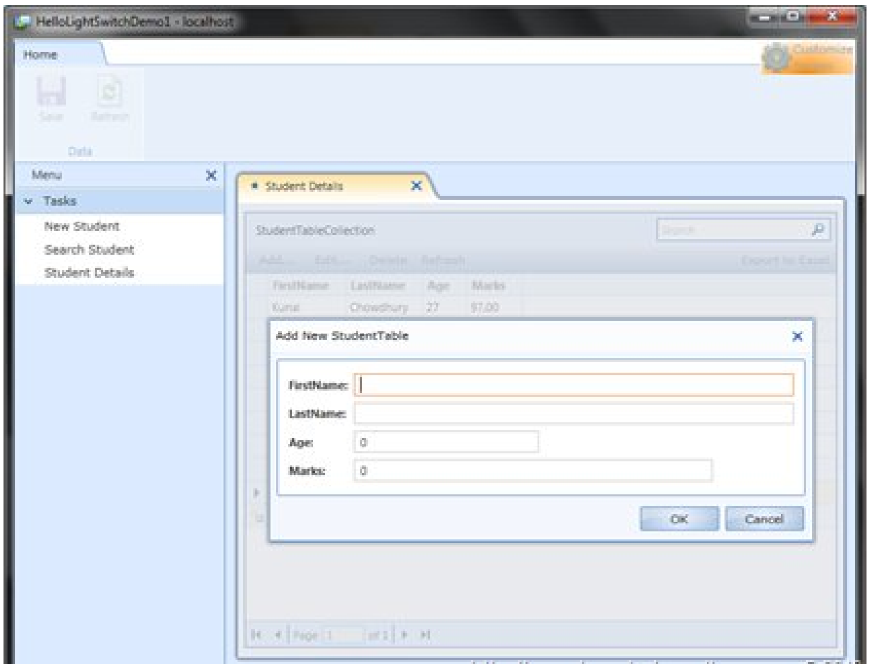
FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00
Michael	Crump	34	89.00
Dhananjay	Kumar	29	90

Now we will show the same step using the second approach. Click on the “Add…” button as shown in the below screen:



FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00
Michael	Crump	34	89.00
Dhananjay	Kumar	29	90

This will popup the Child Window on top of the screen. It will have all the fields present in the table. Enter all the records.

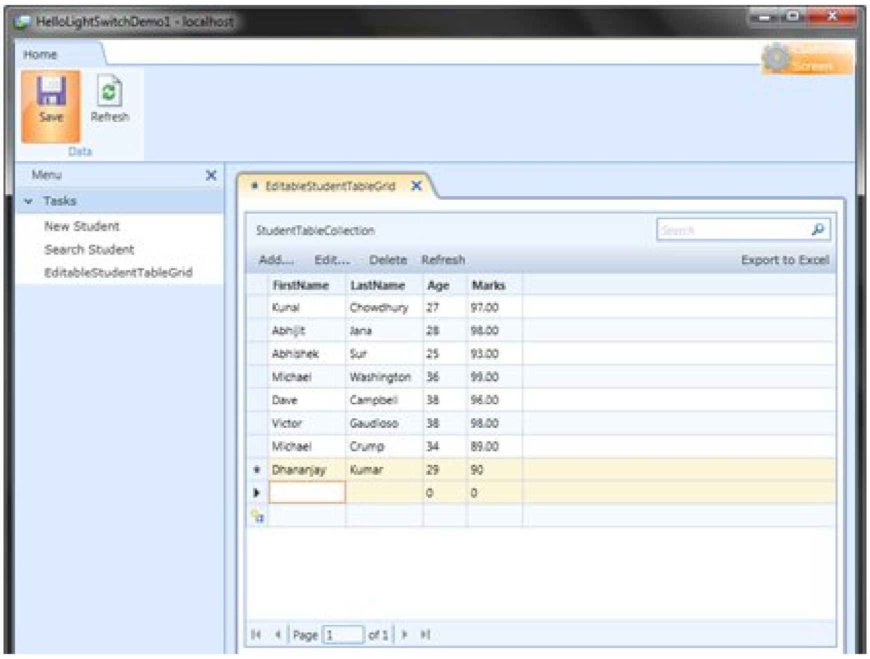


FirstName:	
LastName:	
Age:	0
Marks:	0

Click “OK” and in the parent screen, press “Save” to store the new record. You will see the new records in the DataGrid.

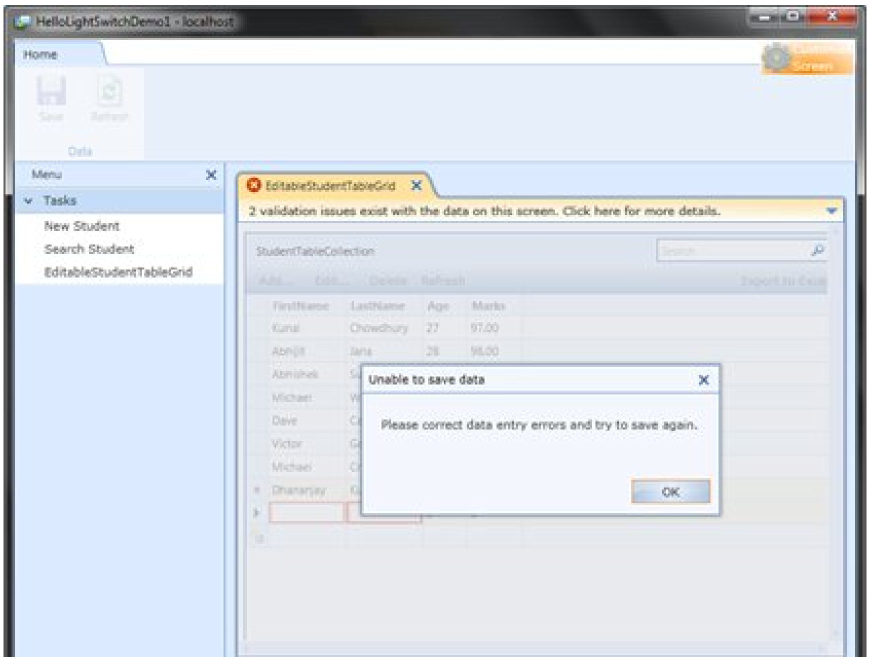
Validate the Record

Before doing anything, I will show you the validation in the same screen. We will enter empty record in the screen and press “Save” button now. See the following screenshots:



FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00
Michael	Crump	34	89.00
Dhananjay	Kumar	29	90
		0	0

You will see that, it actually validates the record automatically and show a message to the screen showing “Please correct data entry errors and try to save again.” Click “OK” to go back to the parent screen.

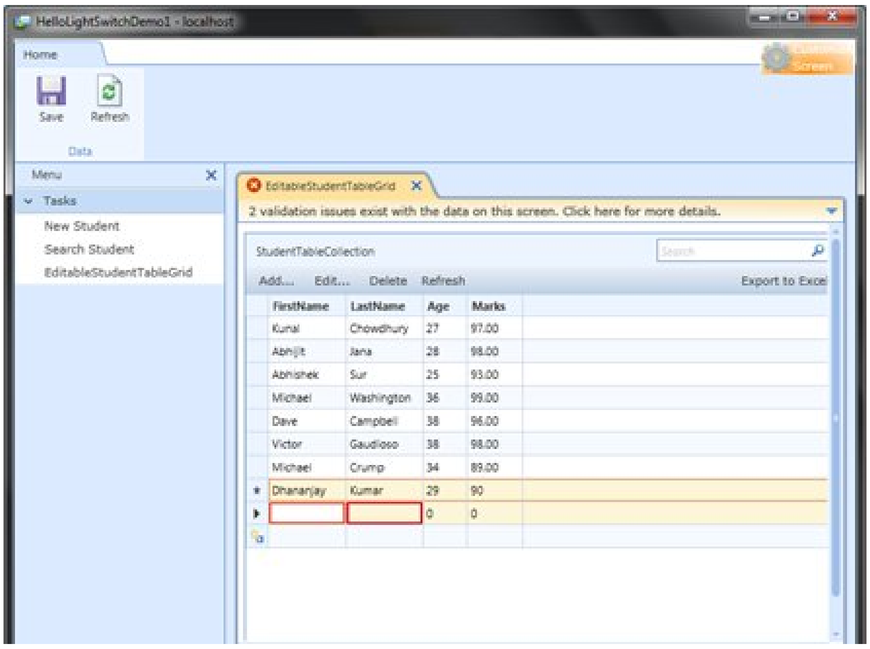


2 validation issues exist with the data on this screen. Click here for more details.

Unable to save data

Please correct data entry errors and try to save again.

You will notice that it actually mark those fields as "Red border".



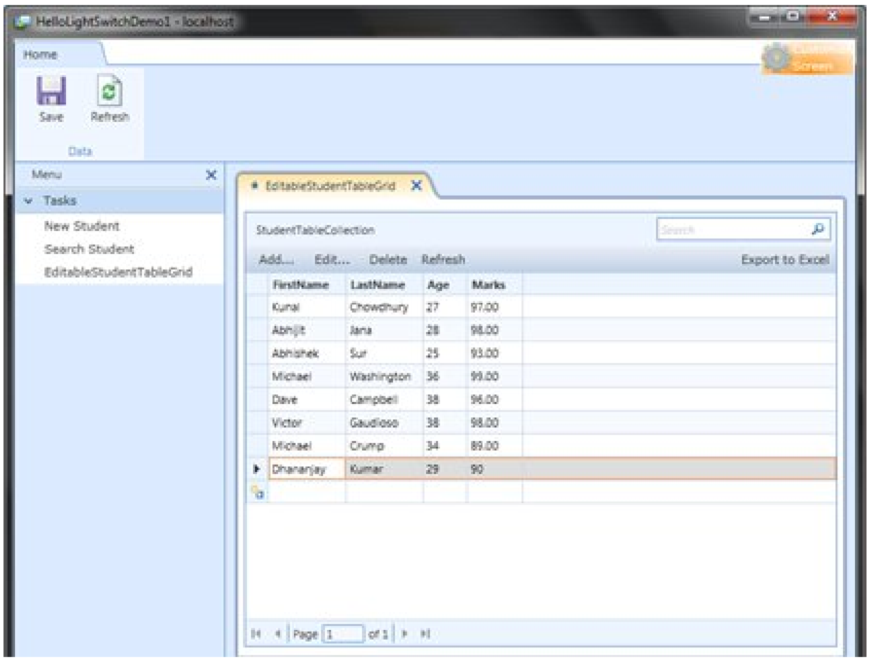
2 validation issues exist with the data on this screen. Click here for more details.

FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00
Michael	Crump	34	89.00
Dhananjay	Kumar	29	90
		0	0

Fill those mandatory fields and click "Save" once again. You will notice that, those validation error will go out from the screen and the new records will get saved in to the table.

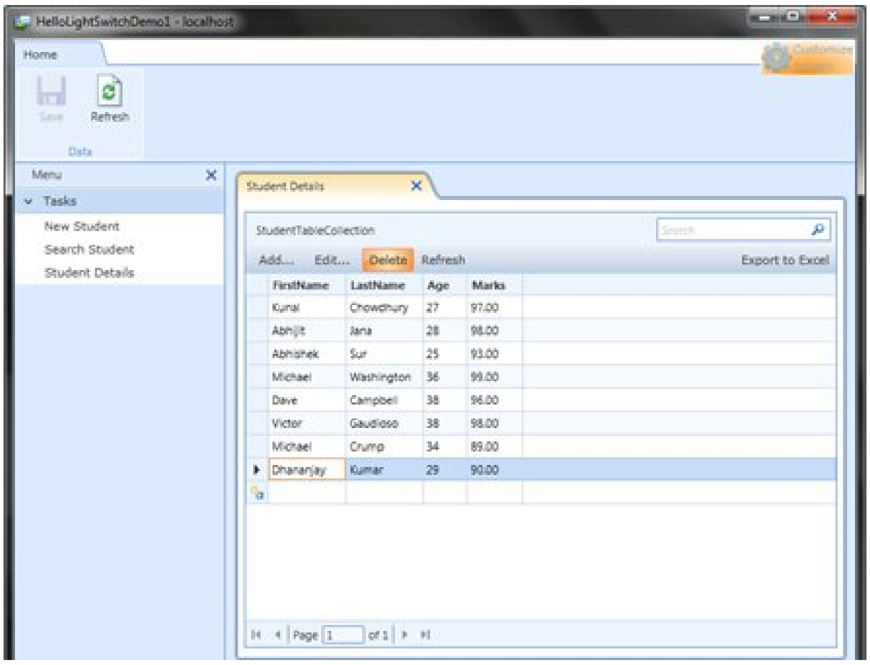
Delete a Record

We had gone to a lengthy discussion of inserting and modifying records. Now it's time to delete a record. Select the record(s) and press delete. You will notice that, the record has been deleted from the DataGrid.



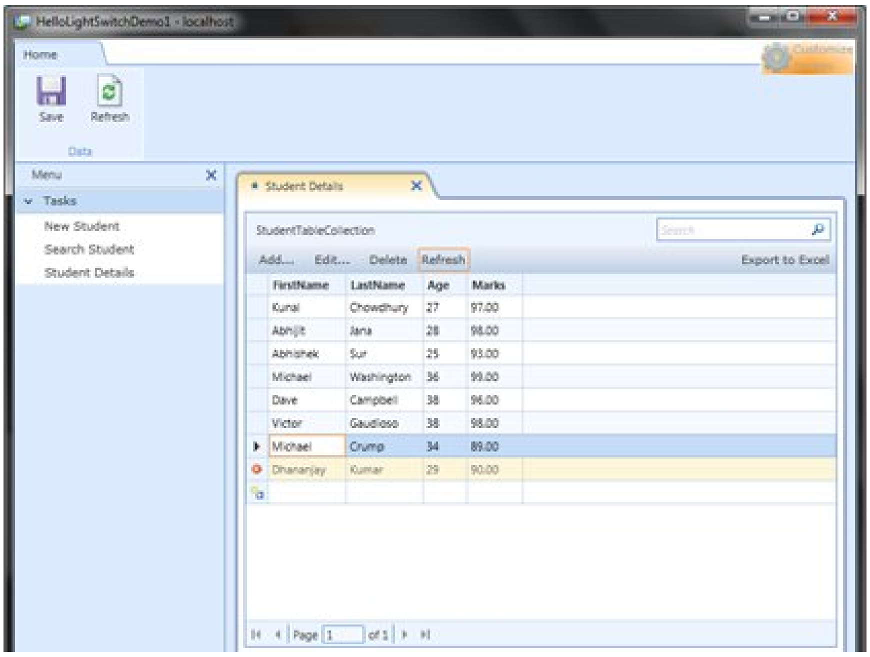
FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00
Michael	Crump	34	89.00
Dhananjay	Kumar	29	90

Now go with the second approach. It is the same thing that we saw in the first approach. Select the row(s) and click the “Delete…” button in the screen (as shown in the below screenshot).



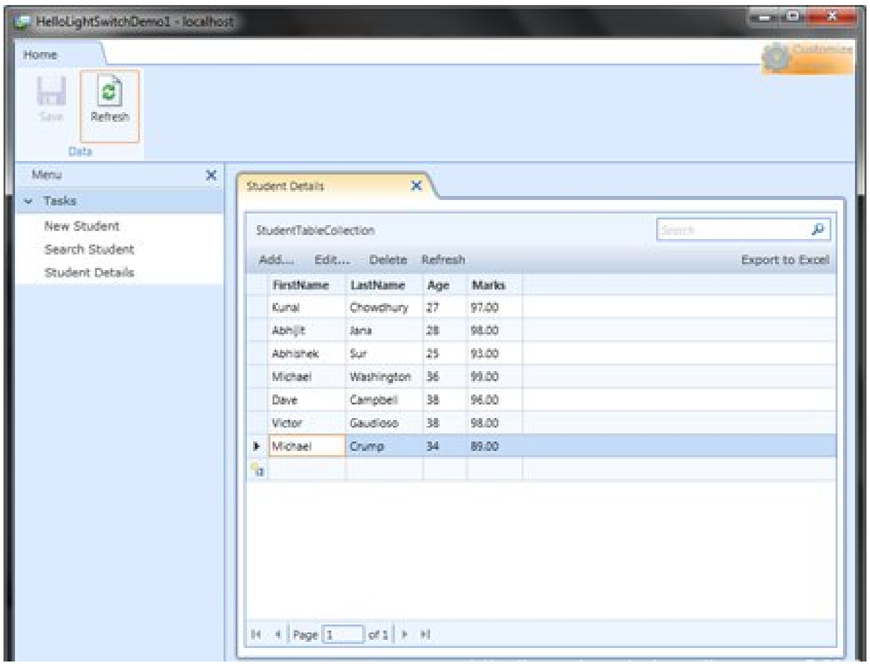
FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00
Michael	Crump	34	89.00
Dhananjay	Kumar	29	90.00

You will see that, the record has been deleted from the DataGrid and there is a red “x” mark placed at the side of the record. That means, the record has been deleted.



FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00
Michael	Crump	34	89.00
Dhananjay	Kumar	29	90.00

Press “Save”. You will see the record completely removed from the record table.

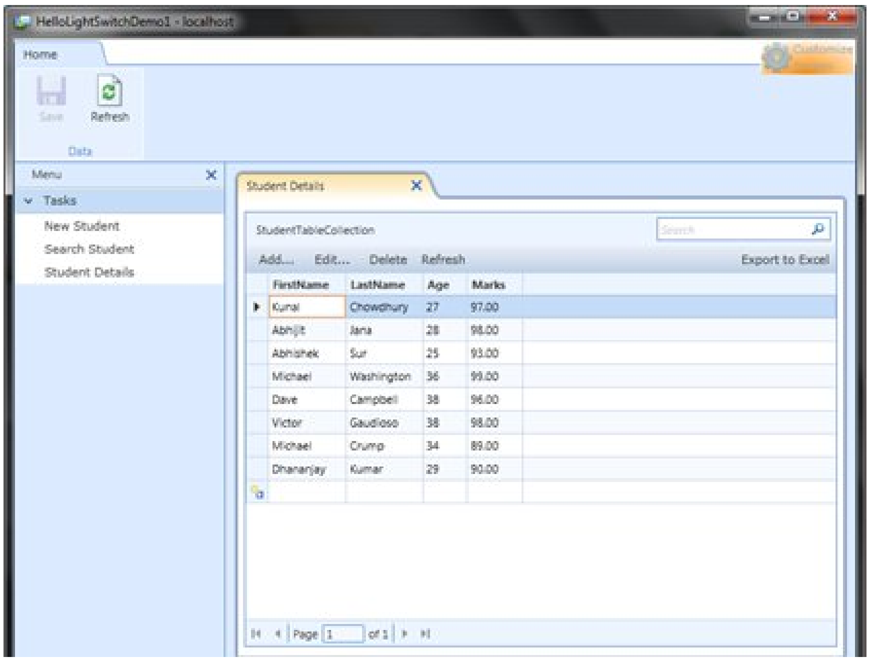


FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00
Michael	Crump	34	89.00

Woo, what a simple approach. No code, no XAML. The application is ready with add, edit and delete functionality. Let’s see some more features of it.

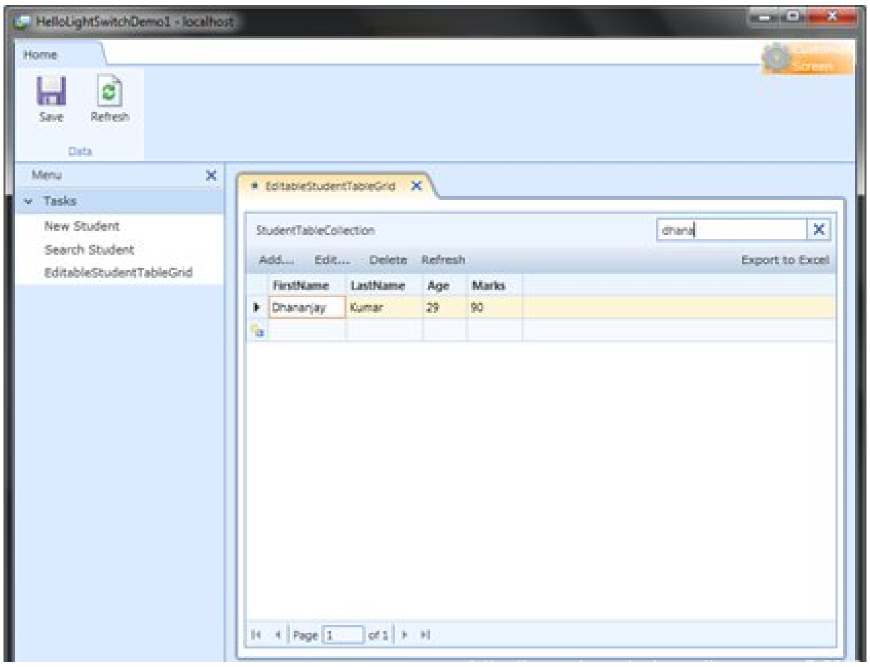
Filter & Export Records

This step we already discussed in the 2nd Chapter. But we will discuss it once again. In the same screen you can sort the records very easily. If you want to sort them by “Age”, click on the header of the column and you will see the records automatically sorted.



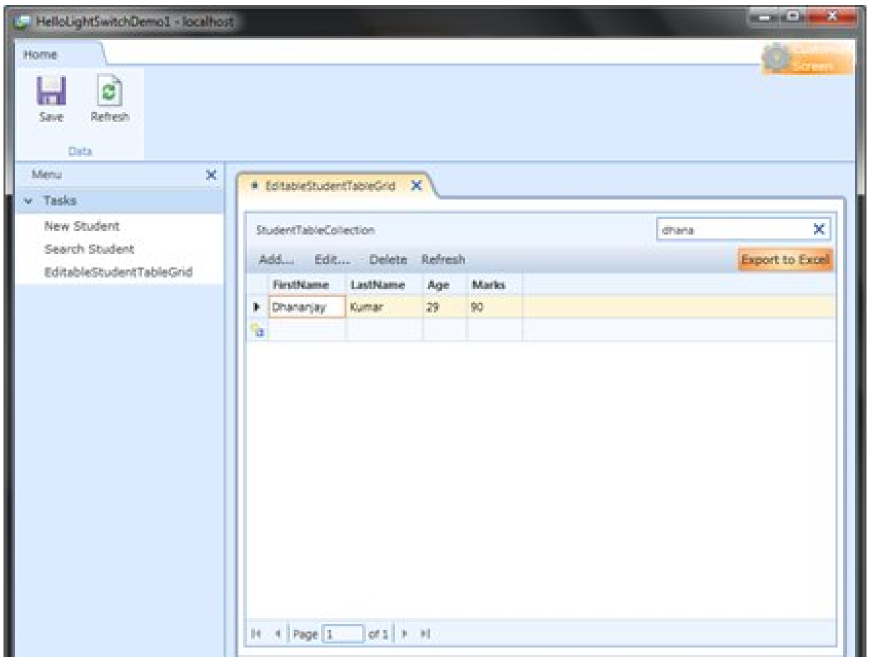
FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00
Michael	Crump	34	89.00
Dhananjay	Kumar	29	90.00

To filter out the records, start typing your query key in the search panel. You will notice that, the record has been filtered properly as per the search key. Have a look in to the following snapshot:



FirstName	LastName	Age	Marks
Dhananjay	Kumar	29	90

Oh yea, you can also export the records to an Excel worksheet by just pressing a single button. Click on “Export to Excel” button and you will watch the records exported to a Excel workbook. Yes, there’s a condition. You must have Excel installed in your PC. 😉

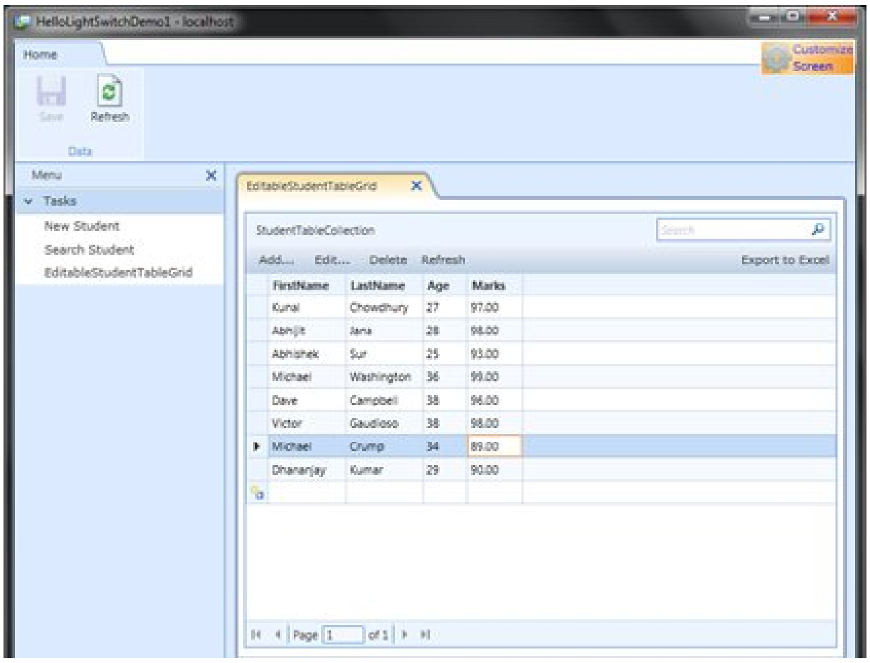


FirstName	LastName	Age	Marks
Dhananjay	Kumar	29	90

There’s also a Navigation panel to paginate all the records.

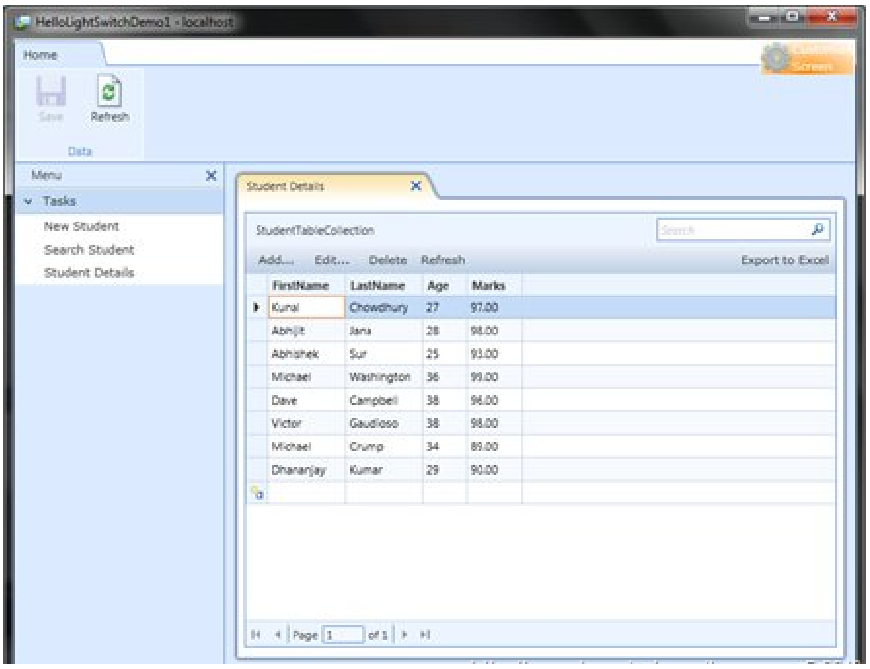
Customizing Screen

You can also customize the screen. You can change the title of the screen. I will not discuss more on this topic. Read the first two chapters. I discussed more about this point there. To change the screen name, click on the “Customize Screen” button placed at the right corner of the screen.



FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00
Michael	Crump	34	89.00
Dhananjay	Kumar	29	90.00

Once you update the display name there, save the settings and restart the application to see the updated settings in the screen.



FirstName	LastName	Age	Marks
Kunal	Chowdhury	27	97.00
Abhijit	Jana	28	98.00
Abhishek	Sur	25	93.00
Michael	Washington	36	99.00
Dave	Campbell	38	96.00
Victor	Gaudioso	38	98.00
Michael	Crump	34	89.00
Dhananjay	Kumar	29	90.00

That's all about it. You will now see that the new name has been assigned to the screen.

End Note

You can see that, throughout the whole application I never wrote a single line of code. I never did write a single line of XAML code to create the UI. It is presented by the tool template automatically. It has a huge feature to do automatically. From the UI design to add, update, delete and even sort, filter all are done automatically by the framework.

I hope, you enjoyed this chapter of the series a lot. Lots of figures I used here, so that, you can understand each steps very easily. If you liked this article, please don't forget to share your feedback here. Appreciate your feedback, comments, suggestion and vote.

I will soon post the next chapter. Till then enjoy reading my other articles published in my Blog and Silverlight Show.
Cheers

Beginners Guide to Visual Studio LightSwitch (Part - 4)

Visual Studio LightSwitch is a new tool for building data-driven Silverlight Application using Visual Studio IDE. It automatically generates the User Interface for a DataSource without writing any code. You can write a small amount of code also to meet your requirement.

In my previous chapter “**Beginners Guide to Visual Studio LightSwitch (Part – 3)**”, I guided you step-by-step to create a DataGrid of records. Here you understood, how to insert/modify/delete records.

In this chapter, I am going to demonstrate you how to create a List and Details screen using LightSwitch. This will show you how to integrate two or more tables into a single screen.

Background

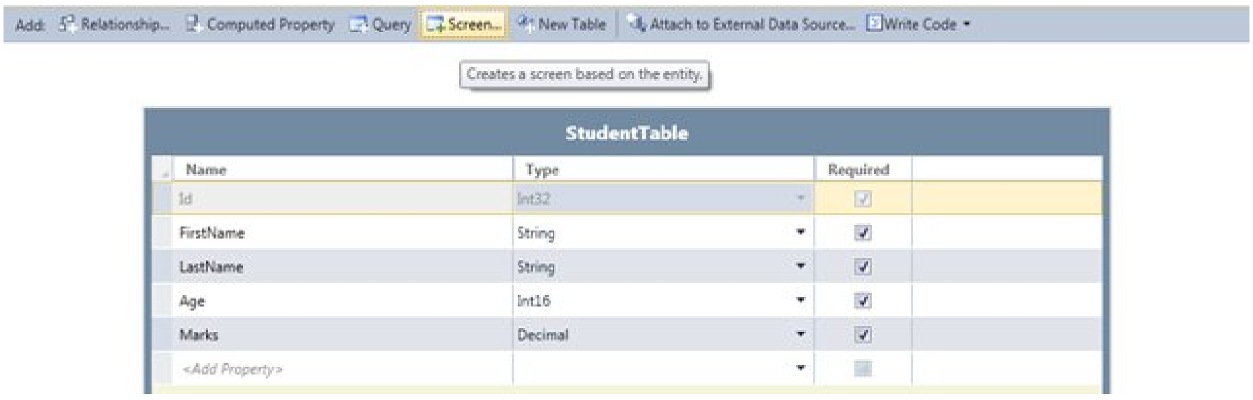
If you are new to Visual Studio LightSwitch, I will first ask you to read the previous three chapters of this tutorial, where I demonstrated it in detail. In my 3rd chapter, I discussed the following topics:

- Create the Editable DataGrid Screen
- See the Application in Action
- Edit a Record
- Create a New Record
- Delete a Record
- Filter & Export Records
- Customizing Screen

In this Chapter, we will discuss on the List and Details Screen where we can integrate two or more Tables inside a single Screen. Read the complete tutorial to learn it. Enough images are there for you to understand it very easily. Don’t forget to vote and write some feedbacks on this topic.

Creating the List and Details Screen

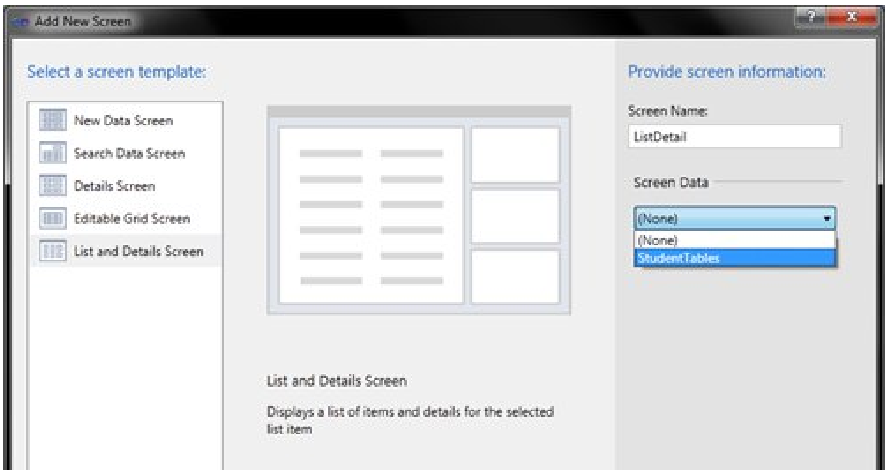
Like the other screens, it is very easy to create the List and Details screen. Hope you have the basic table available with you. If you don’t have, read my first chapter of this tutorial to create the one for you. Now open the StudentTable inside your Visual Studio 2010 IDE. As shown in the below screenshot, click on the “Screen…” button to open up the Screen Template selector.


Add:
Relationship...
Computed Property
Query
Screen...
New Table
Attach to External Data Source...
Write Code
Creates a screen based on the entity.

StudentTable

Name	Type	Required
Id	Int32	✓
FirstName	String	✓
LastName	String	✓
Age	Int16	✓
Marks	Decimal	✓
<Add Property>		

In the “Add New Screen” dialog that pops up (which is nothing but the Screen Template Selector), chose the “List and Details Screen” template. Check the below screenshot for details:

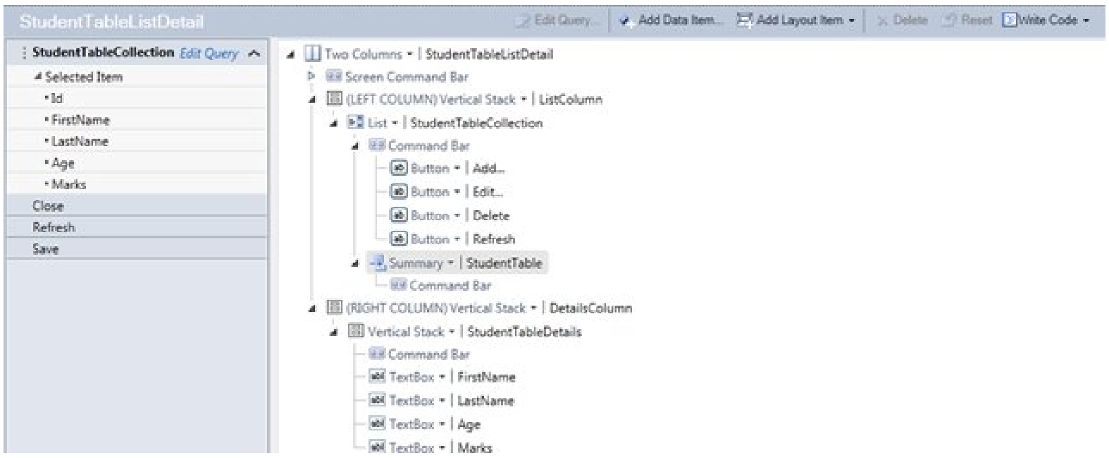


List and Details Screen

Displays a list of items and details for the selected list item

On the right side panel, chose the table that you want to use here. Select “StudentTables” from the Screen Data combo box and hit “Ok”. This will create the basic UI for you from the screen template with all the necessary fields.

You will be able to see the UI design inside the Visual Studio IDE itself. This will look just like this:

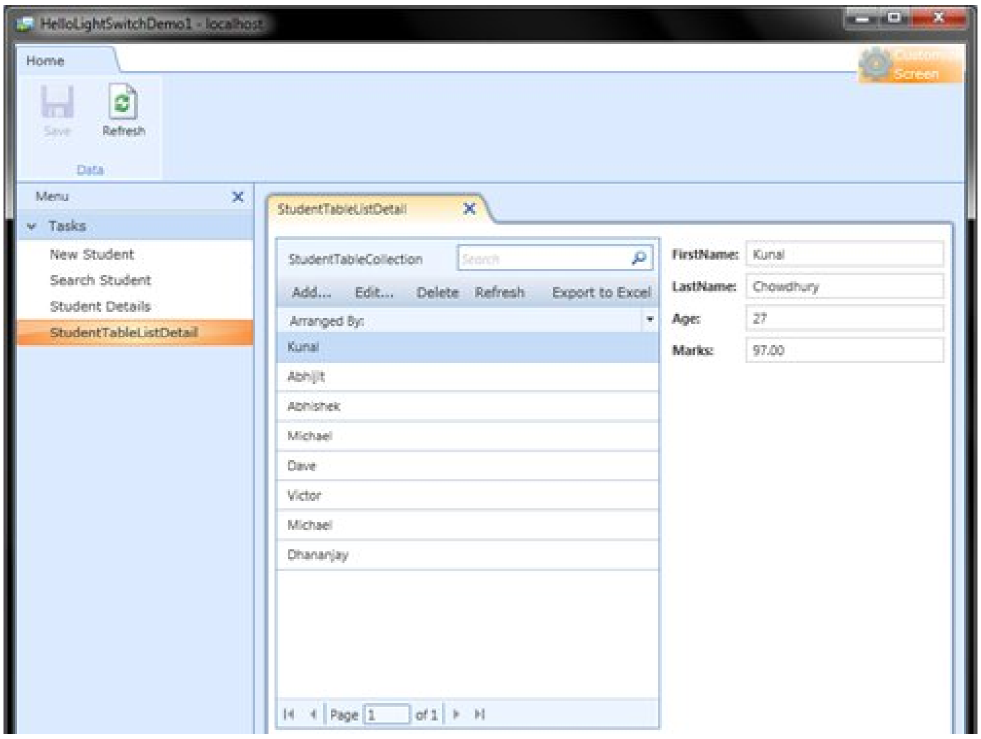


You will see that the screen has four TextBox named FirstName, LastName, Age and Marks. These are nothing but the table columns added to the screen.

Now run the application to see it in action. Once the application shows up in the display screen, click the “StudentTableListDetail” menu item from the left navigation panel. If you don’t know what is this navigation panel, read any one of the previous three chapters.

UI Screen Features

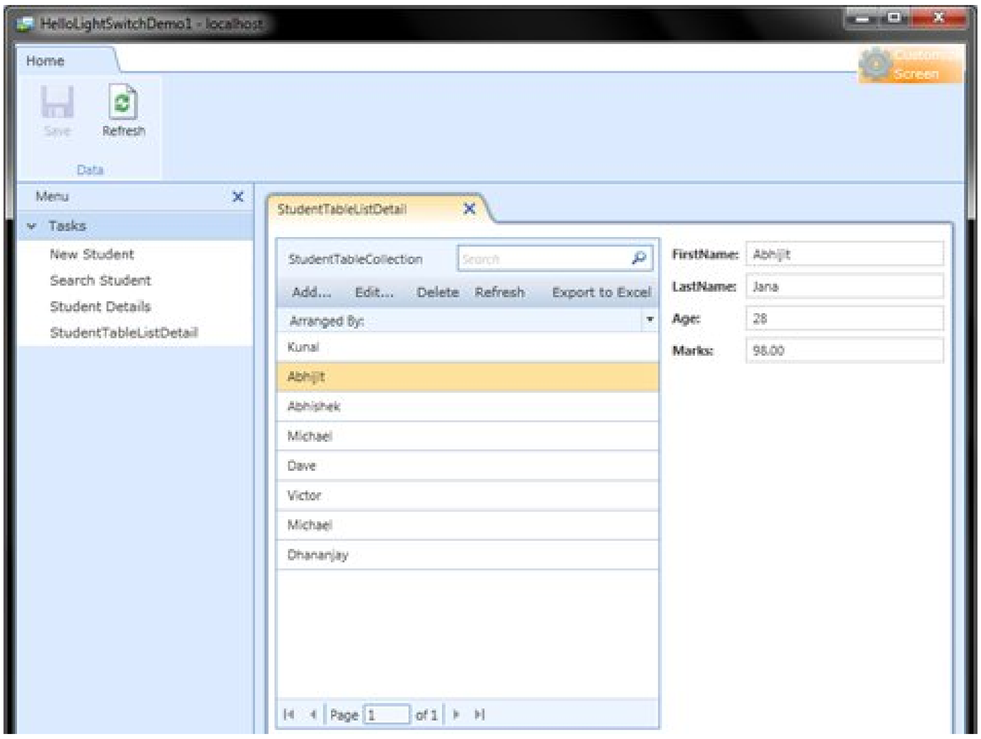
Here you will see that, there are two sections inside the main tab. One hosts only the FirstName from the records column and the right side panel has all the details for that specific record.



Kunal
Abhijit
Abhishek
Michael
Dave
Victor
Michael
Dhananjay

FirstName:	Kunal
LastName:	Chowdhury
Age:	27
Marks:	97.00

If you chose a different record from the left list, the right panel automatically reflect the record details. This screen calls as List and Details screen based on it's UI. The left panel is the list of records and right panel is the details of the selected record.

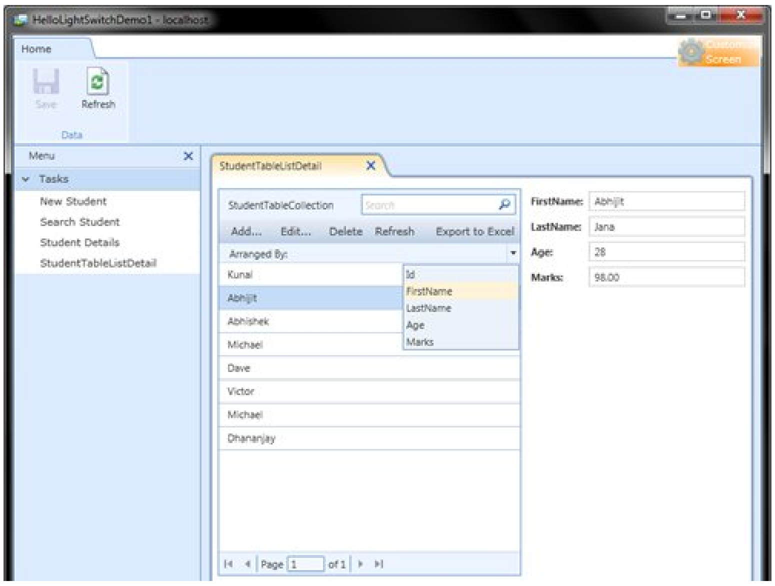


Kunal
Abhijit
Abhishek
Michael
Dave
Victor
Michael
Dhananjay

FirstName:	Abhijit
LastName:	Jana
Age:	28
Marks:	98.00

Here, you will see that, like other screens "Add…", "Edit…", "Delete", "Refresh" and "Export to Excel" buttons are available. The "Add" button will popup a new modal dialog with the form to enter new record. "Edit" button will popup a modal dialog with the selected record to modify it. "Delete" will simply delete the record from the table. It will not actually delete from the table unless you press “Save” after that. “Refresh” button will re-query the table to fetch the records. “Export to Excel” opens up the Excel application with the listed records.

You will also notice, there is a new Combo Box in the UI. It’s called as “Arrange By” (as shown in the below figure). If you click on the drop down you will see all the column names in the list.

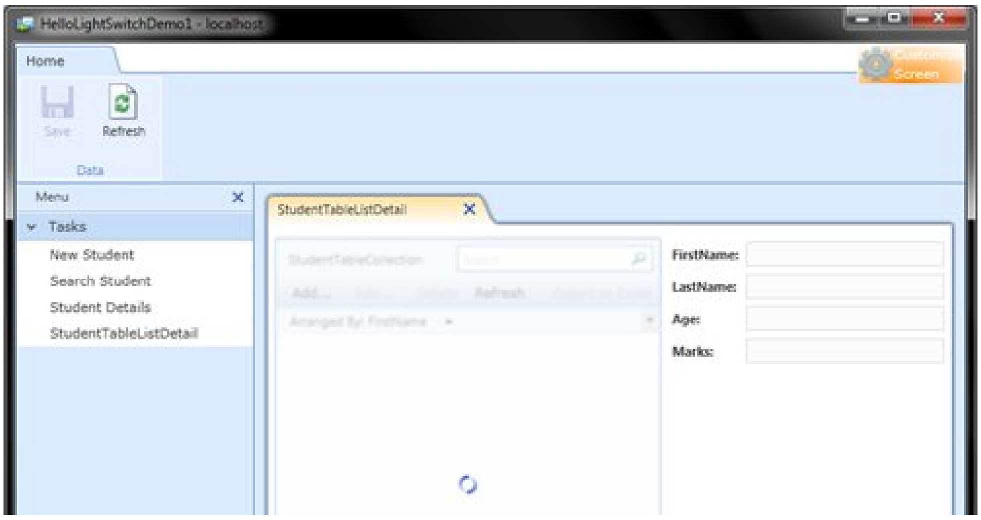


Kunal
Abhijit
Abhishek
Michael
Dave
Victor
Michael
Dhananjay

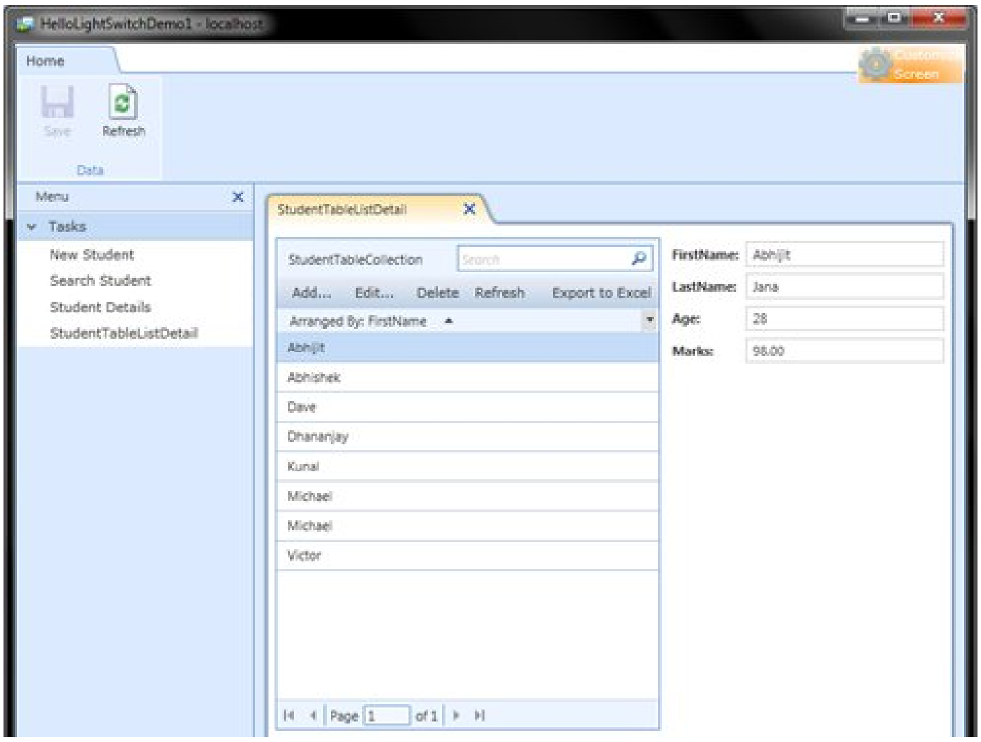
FirstName: Abhijit
LastName: Jana
Age: 28
Marks: 98.00

From there, you can sort the records by any field. Once you select a different field from the drop down, it will again load the records sorted by that field.

Like the below screenshot, you will see an indicator while sorting and loading the records. No code to be written for all these functionalities. They are all out-of-box features.



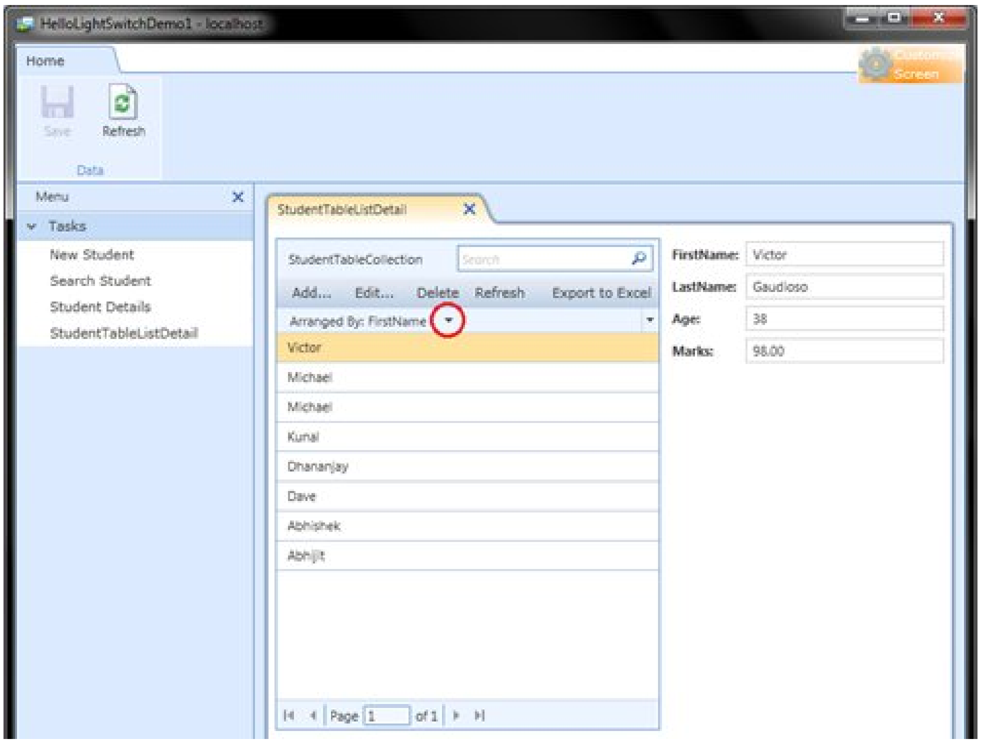
Here you will see how the records have been sorted by FirstName (in ascending order):



Abhijit
Abhishek
Dave
Dhananjay
Kunal
Michael
Michael
Victor

FirstName:	Abhijit
LastName:	Jana
Age:	28
Marks:	98.00

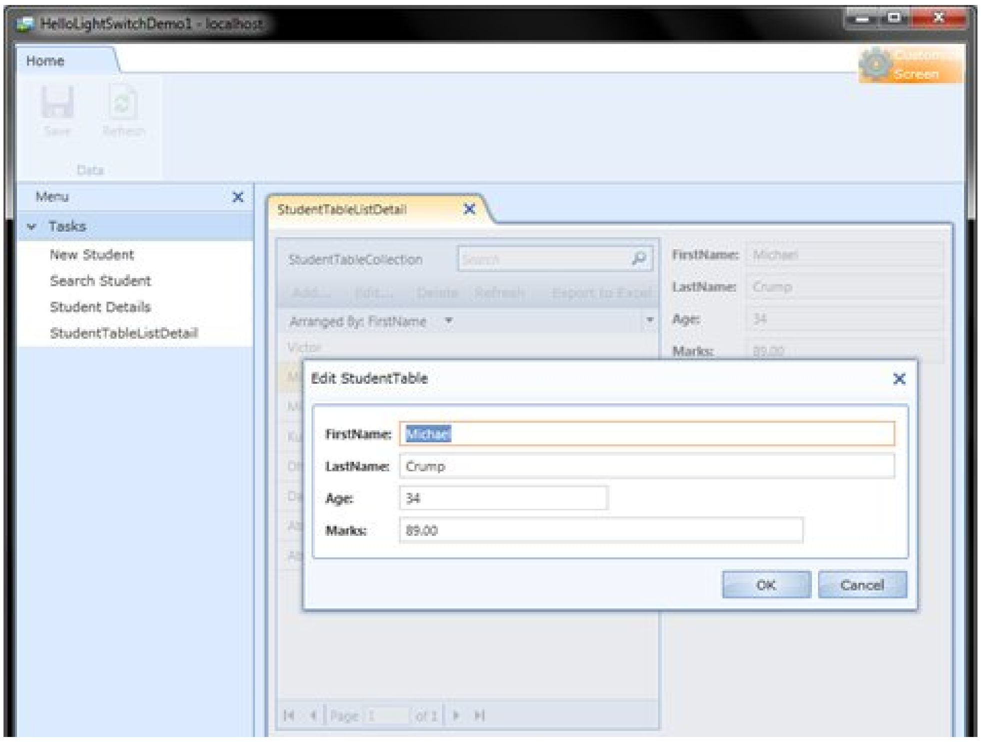
If you want to sort them on descending order, just click the small arrow head just right to the “Arrange By” text. Here is the screenshot of the same (marked with Red circle):



Victor
Michael
Michael
Kunal
Dhananjay
Dave
Abhishek
Abhijit

FirstName:	Victor
LastName:	Gaudioso
Age:	38
Marks:	98.00

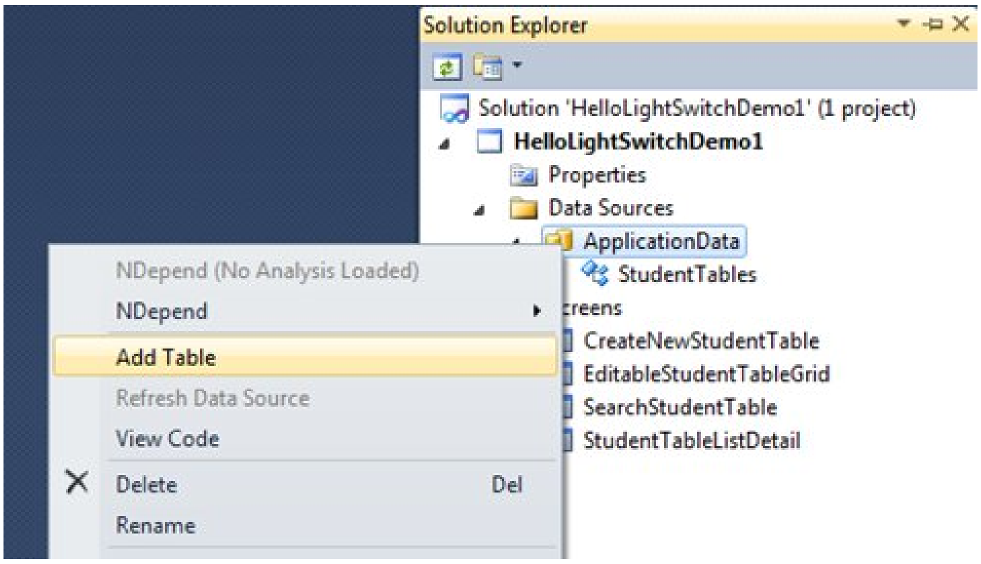
Add, Edit operations are similar to the earlier screens. Either press the respective buttons or double click the DataGrid. Double clicking the DataGrid cell will do the operation inline but pressing the button will pop up a new modal dialog on the screen.



You can see the above figure. Once you click "Edit" button, the modal window will pop up as shown in the figure. Update the records with desired values.

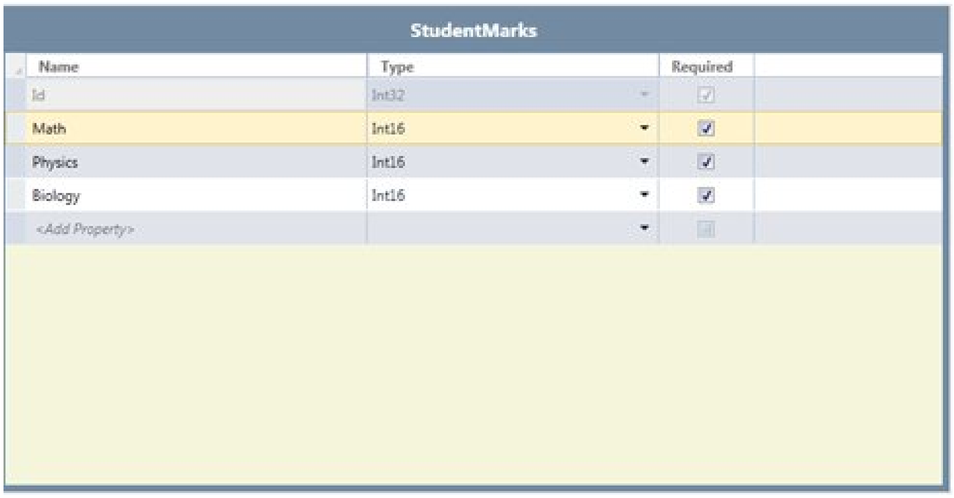
Adding a New Table

Let us create a new Table. Open your Solution Explorer and right click on the "Application Data". This will bring a context menu in the screen. As shown below, click on the "Add Table" menu item to open the Table designer.



In the table designer screen, enter all the columns with specific column types. For our example, we will create a StudentMarks table with three new columns named as "Math", "Physics" and "Biology". "Id" column is default primary key field.

All the columns are Int16 type in our case. The new table design will look like this:

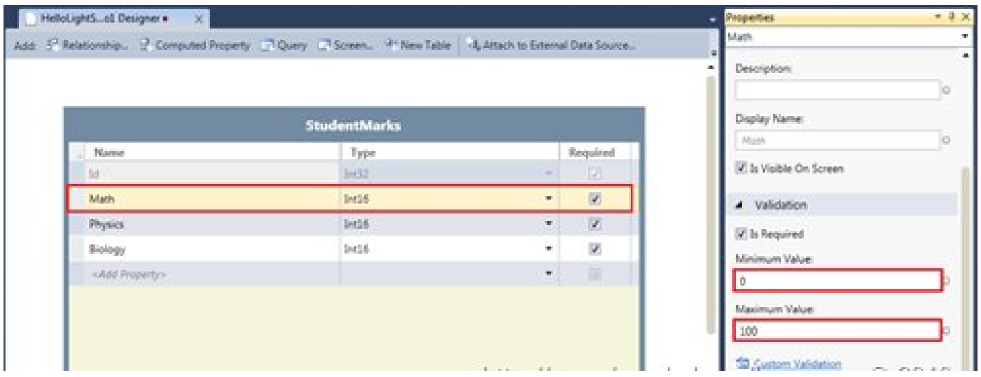


StudentMarks

Name	Type	Required
Id	Int32	✓
Math	Int16	✓
Physics	Int16	✓
Biology	Int16	✓
<Add Property>		

Creating the Validation Rules

In earlier chapters, we saw how the IDE creates the validation rule by default for null value. Here, we will jump a little bit depth and set some additional validation rules. Of-course, they will not be in code but setting some properties inside the property panel.


HelloLightS...ol Designer
Add: Relationship... Computed Property Query Screen... New Table Attach to External Data Source...
Properties
Custom Validation

StudentMarks

Name	Type	Required
Id	Int32	✓
Math	Int16	✓
Physics	Int16	✓
Biology	Int16	✓
<Add Property>		

Math

Description:

Display Name: Math

✓ Is Visible On Screen

Validation

✓ Is Required

Minimum Value: 0

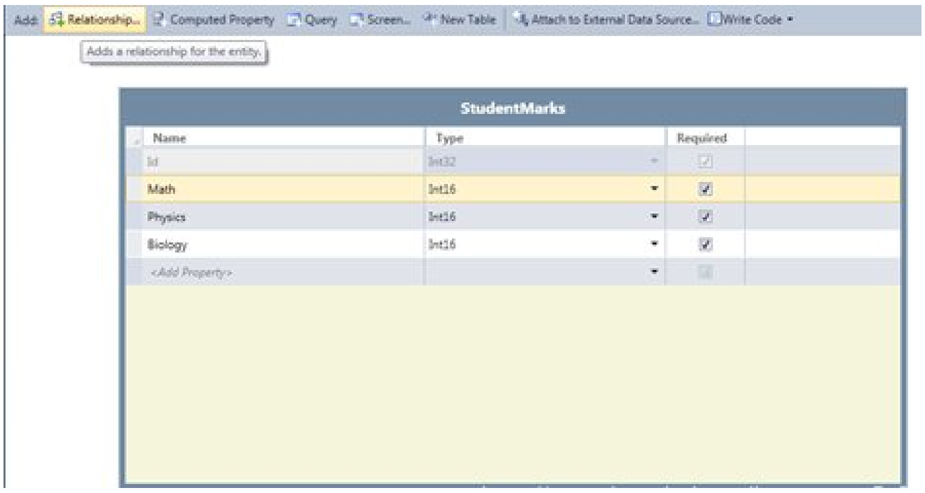
Maximum Value: 100

Chose each one of the subject columns one by one and set the Minimum and Maximum values for them. In our case, the minimum value will be 0 (zero) and maximum value will be 100. Repeat the same step mentioned in the above screenshot for all the other two fields. Now, those fields will take an integer value ranging between 0 and 100. You will not be able to enter any value less than 0 or greater than 100.

If you enter any other value outside the range, it will fail the validation and show appropriate message on the screen. We will see that in action later. But before that, we need to create the screen.

Adding Relationship between two tables

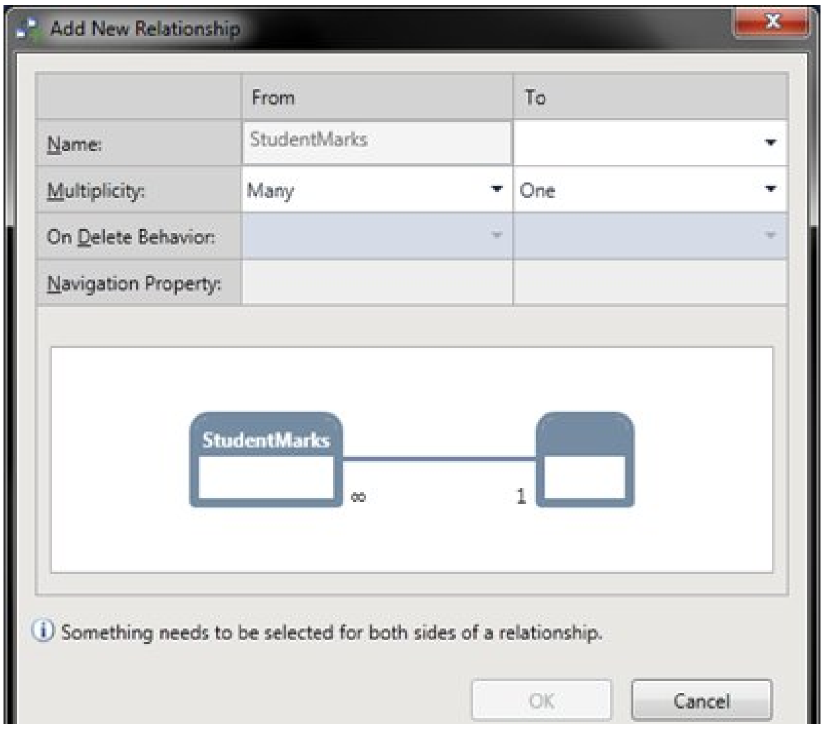
Yes, we can easily create a relationship between two tables. How? Create the “Relationship…” button placed at the top left corner of the table designer screen (as shown below);


Add: Relationship... Computed Property Query Screen... New Table Attach to External Data Source... Write Code
Adds a relationship for the entity.

StudentMarks

Name	Type	Required
Id	Int32	✓
Math	Int16	✓
Physics	Int16	✓
Biology	Int16	✓
<Add Property>		

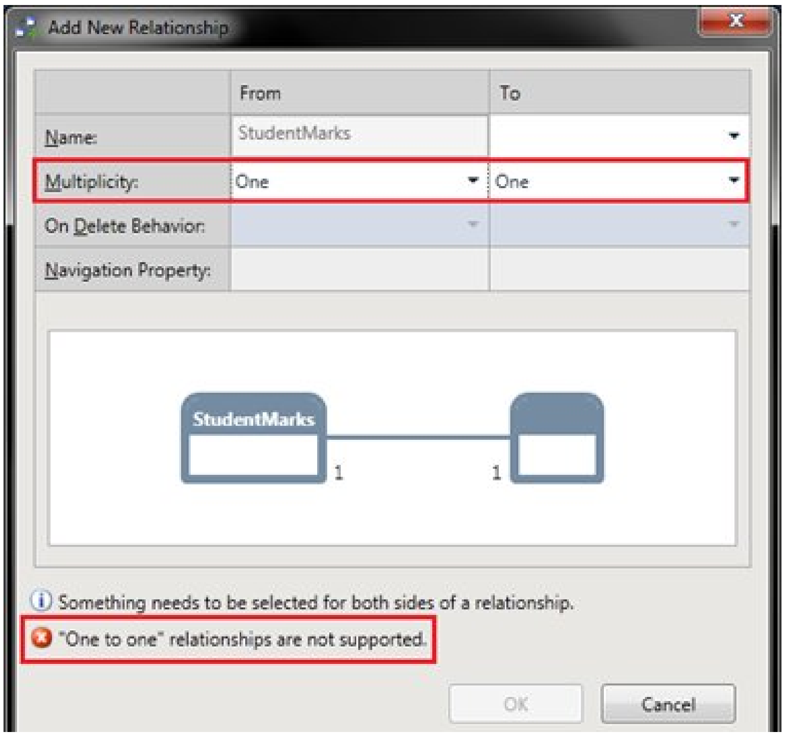
This will bring the “Add New Relationship” dialog in the screen. You will notice there that, it has two table field “from” and “to”. This means the relationship between two tables (one with the other). There is a field called “Multiplicity”. This is nothing but the relationship type (one to one, one to many, many to one etc.)



	From	To
Name:	StudentMarks	
Multiplicity:	Many	One
On Delete Behavior:		
Navigation Property:		

Something needs to be selected for both sides of a relationship.

One to one is not supported in this Beta 1 version. Can’t say whether it will be available in the next versions. If you select one to one relationship, the wizard will show the appropriate message saying that “One to one relationships are not supported”. Have a look here:

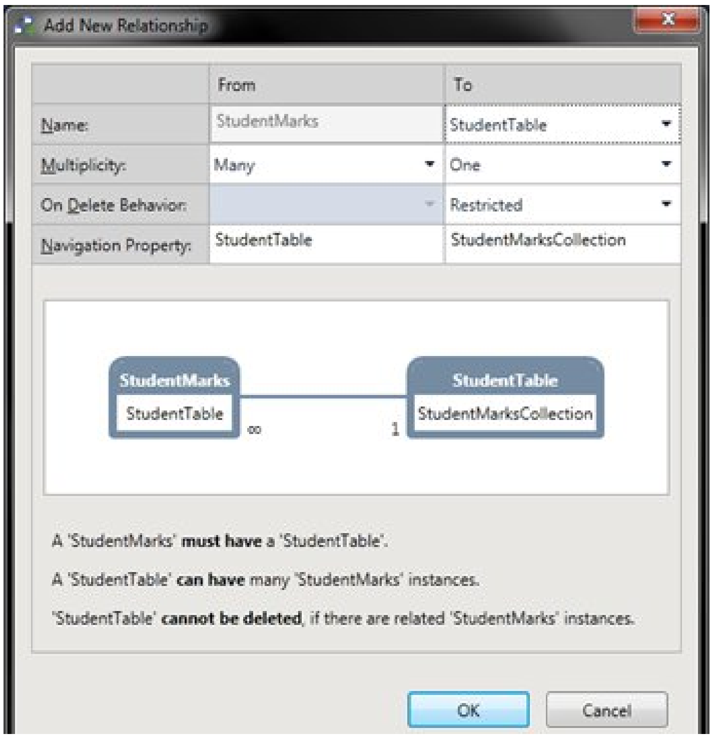

Add New Relationship

	From	To
Name:	StudentMarks	
Multiplicity:	One	One
On Delete Behavior:		
Navigation Property:		

Something needs to be selected for both sides of a relationship.

"One to one" relationships are not supported.

For our example, chose Many to One relationship between StudentMarks and StudentTable. You can select the delete behavior in the wizard, whether it will support restricted or cascaded delete operation.


Add New Relationship

	From	To
Name:	StudentMarks	StudentTable
Multiplicity:	Many	One
On Delete Behavior:		Restricted
Navigation Property:	StudentTable	StudentMarksCollection

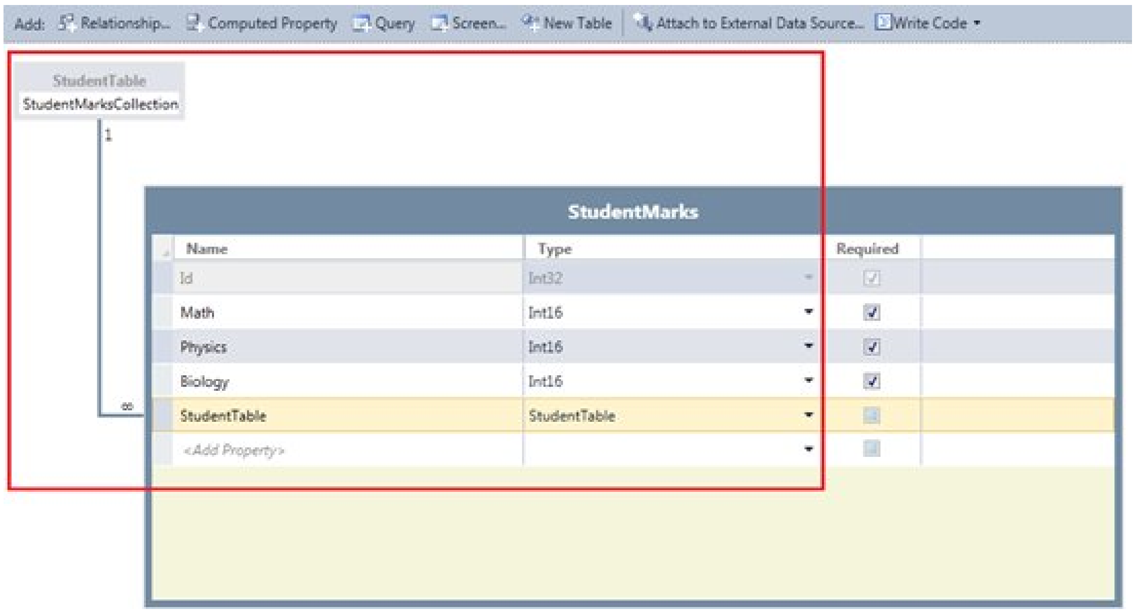
A 'StudentMarks' **must have** a 'StudentTable'.

A 'StudentTable' **can have** many 'StudentMarks' instances.

'StudentTable' **cannot be deleted**, if there are related 'StudentMarks' instances.

Once you setup the relationship between the two tables, click "Ok" to continue creating the relation. Now, Visual Studio 2010 IDE will show you how the relationship is binded between the two tables.

Here you will find the relationship as a diagram. See the box marked as Red:


Add: Relationship... Computed Property Query Screen... New Table Attach to External Data Source... Write Code
StudentTable
StudentMarksCollection
1
∞

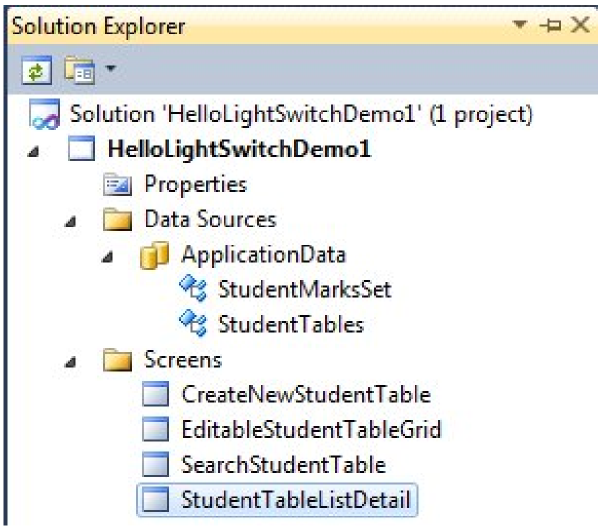
StudentMarks

Name	Type	Required
Id	Int32	✓
Math	Int16	✓
Physics	Int16	✓
Biology	Int16	✓
StudentTable	StudentTable	
<Add Property>		

You will see that, the StudentTable now has the one to many relationship with the StudentMarks table and it actually created a new field called StudenTable of type “StudentTable”.

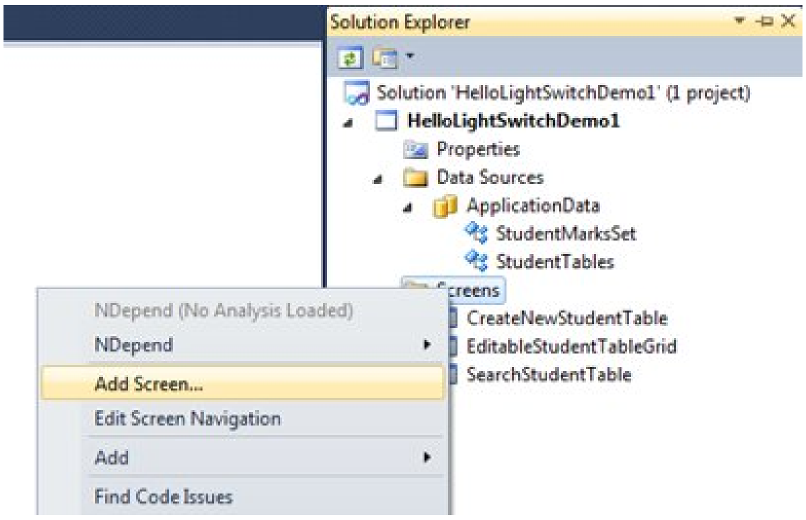
Creating the new List and Details Screen

Now as a beginner, we will first remove the previous ListDetails screen. To do this, go to Solution Explorer and remove the StudentTableListDetail screen from the Screens folder. Here is the snapshot of the same:

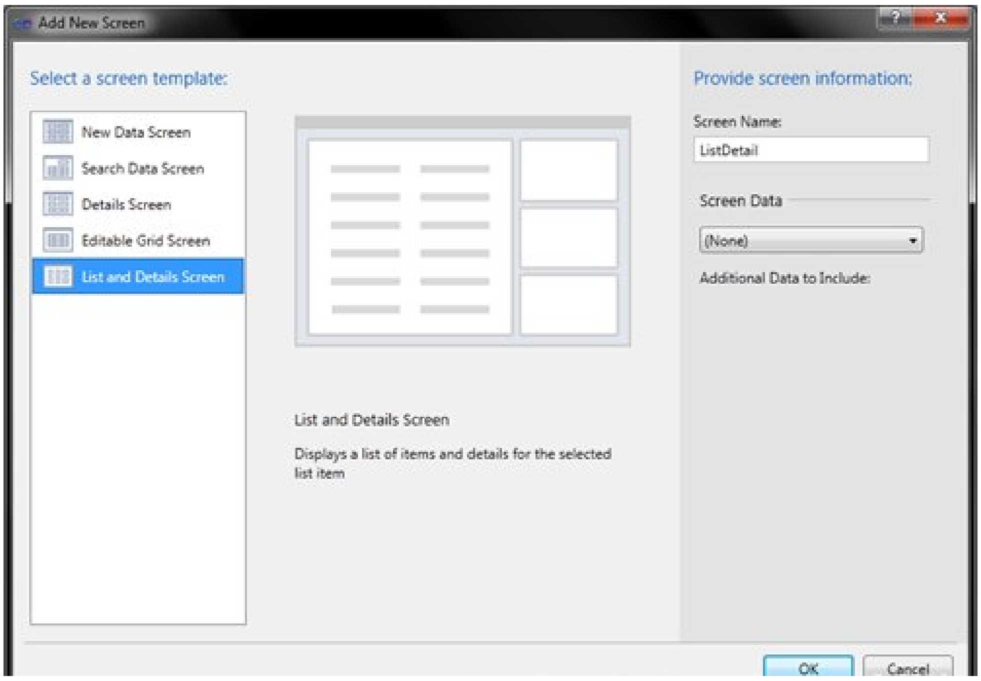


The above step is not require if you are familiar with the screen customization. As the tutorial is targeted for beginners, hence I am removing this screen for you to understand properly.

Let use create a new List & Detail screen for our example. Go to the solution explorer. Right click on the Screens folder (as shown in the below figure), from the context menu chose “Add Screen…”.



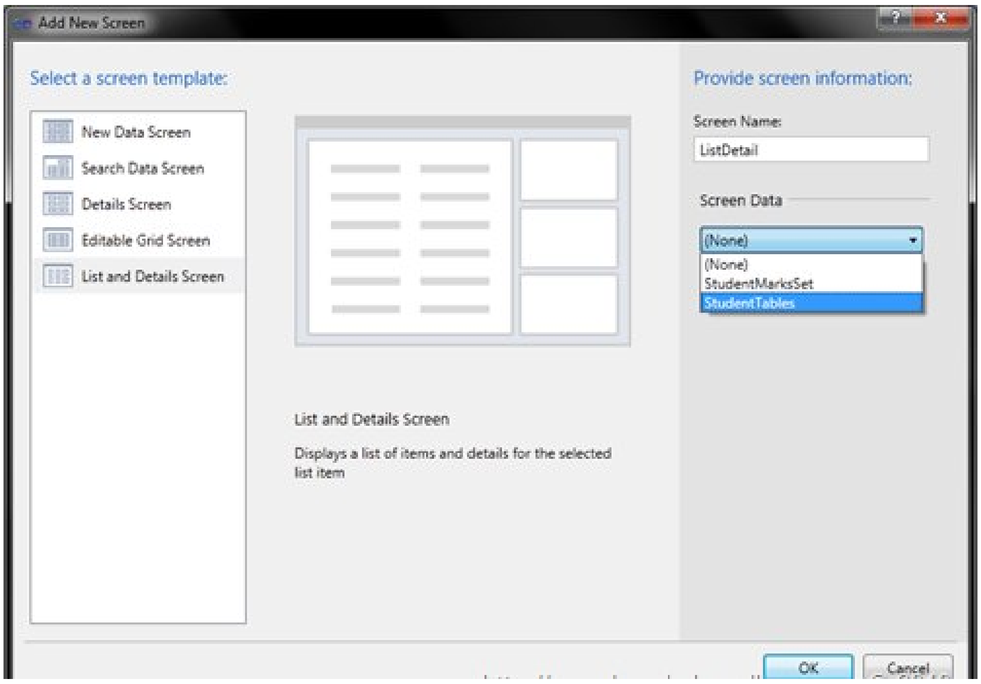
This will popup the “Add New Screen” dialog. You are very familiar with this page as we discussed it earlier for several times. In this page, select “List and Details Screen” from left template panel.



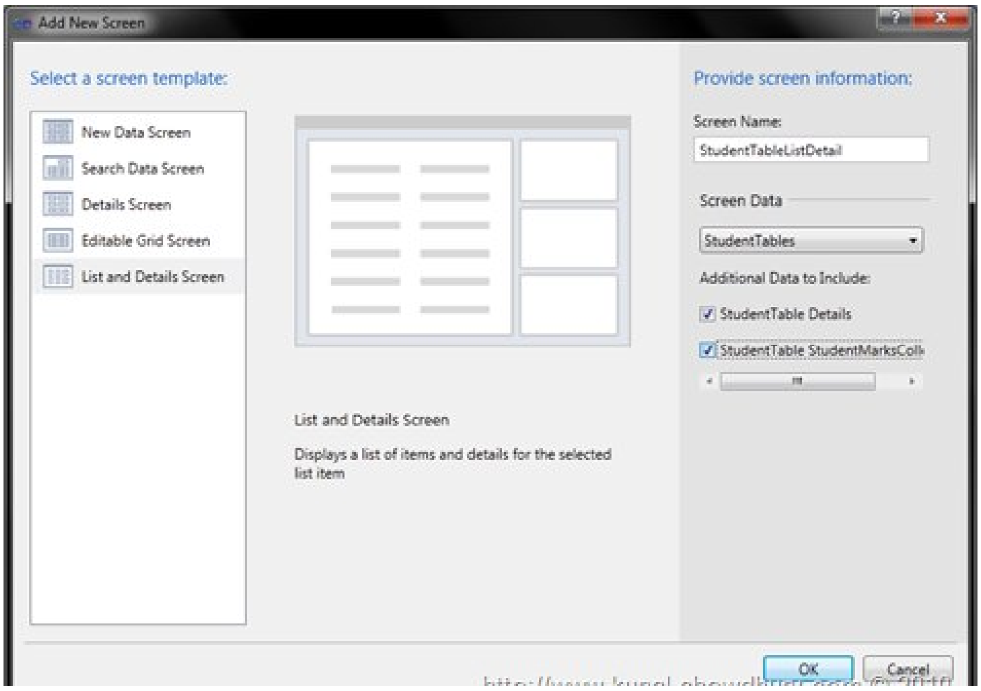
List and Details Screen

Displays a list of items and details for the selected list item

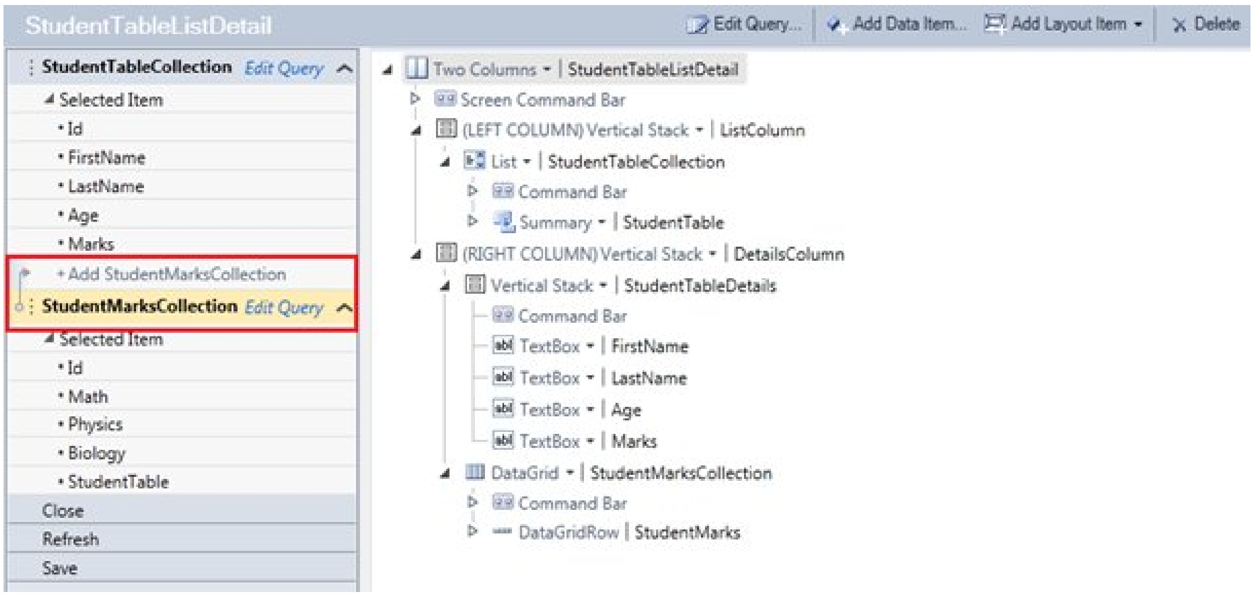
In the right panel, select the desire table as the data source. In our example, we are going to use the “StudentTables” as the data source. Select it.



You will see that, it has now one more extra table called “StudentMarks” which you can chose from there. Let us select all the additional tables from the screen and click “Ok”.



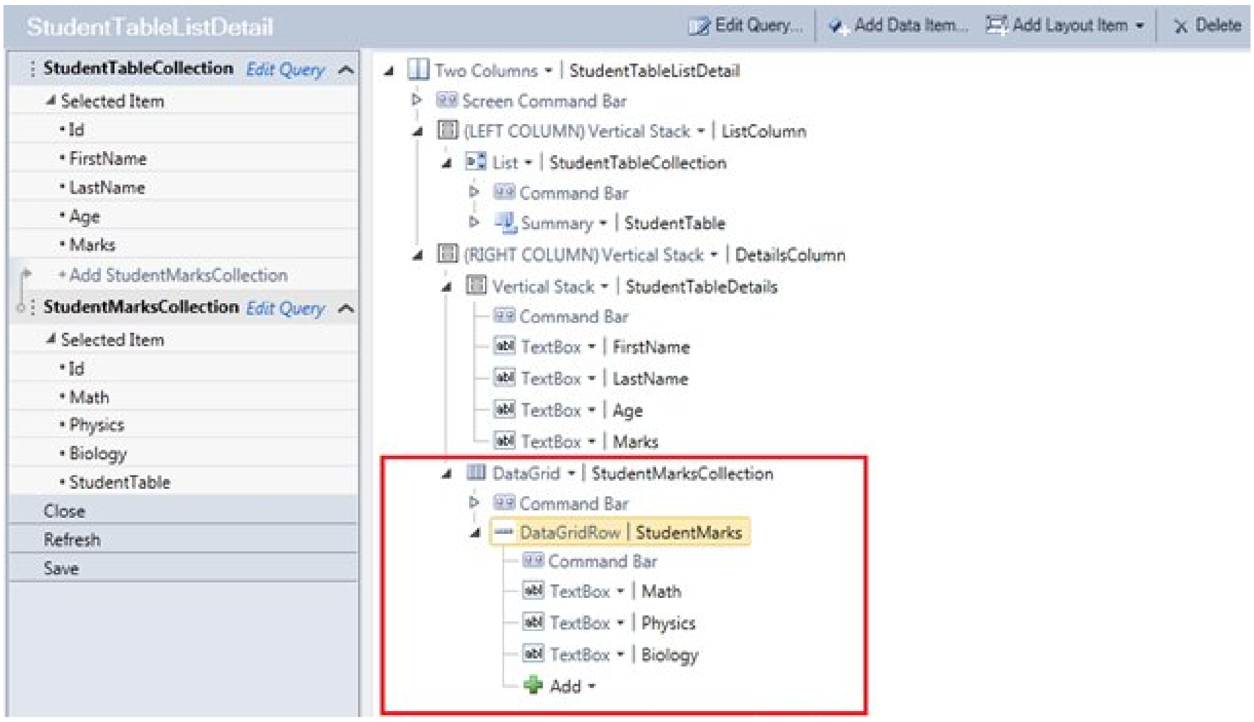
This will create the UI for you and open up the UI inside Visual Studio IDE in the design view. You will see that the StudentMarksCollection (marked as Red in the below screenshot) also has been added to the screen and has reference to the StudentTableCollection.



```
StudentTableCollection  Edit Query
  Selected Item
    Id
    FirstName
    LastName
    Age
    Marks
  + Add StudentMarksCollection
StudentMarksCollection  Edit Query
  Selected Item
    Id
    Math
    Physics
    Biology
    StudentTable
Close
Refresh
Save
```

```
Two Columns | StudentTableListDetail
  Screen Command Bar
  (LEFT COLUMN) Vertical Stack | ListColumn
    List | StudentTableCollection
      Command Bar
      Summary | StudentTable
  (RIGHT COLUMN) Vertical Stack | DetailsColumn
    Vertical Stack | StudentTableDetails
      Command Bar
      TextBox | FirstName
      TextBox | LastName
      TextBox | Age
      TextBox | Marks
    DataGrid | StudentMarksCollection
      Command Bar
      DataGridRow | StudentMarks
```

In the right side design view you will see that, the StudentMarksCollection has been added to the screen as a DataGrid, means all the related data related to the actual StudentTable will be populated here.



```
StudentTableCollection  Edit Query
  Selected Item
    Id
    FirstName
    LastName
    Age
    Marks
  + Add StudentMarksCollection
StudentMarksCollection  Edit Query
  Selected Item
    Id
    Math
    Physics
    Biology
    StudentTable
Close
Refresh
Save
```

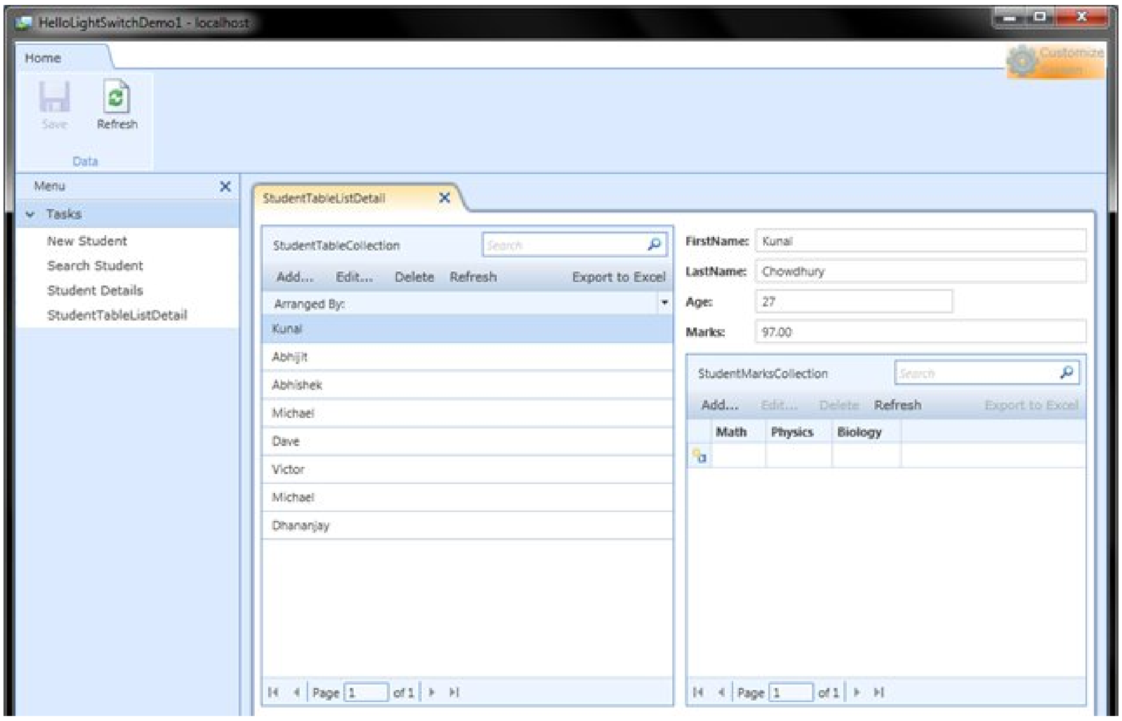
```
Two Columns | StudentTableListDetail
  Screen Command Bar
  (LEFT COLUMN) Vertical Stack | ListColumn
    List | StudentTableCollection
      Command Bar
      Summary | StudentTable
  (RIGHT COLUMN) Vertical Stack | DetailsColumn
    Vertical Stack | StudentTableDetails
      Command Bar
      TextBox | FirstName
      TextBox | LastName
      TextBox | Age
      TextBox | Marks
    DataGrid | StudentMarksCollection
      Command Bar
      DataGridRow | StudentMarks
        Command Bar
        TextBox | Math
        TextBox | Physics
        TextBox | Biology
        Add
```

You will be able to choose the fields from this screen. If you want, you can remove any field and/or select the control type for the respective column type.

Application in Action

This is all about the thing that we wanted to discuss here. Now it's time to see the demo of the same in action. This will exactly tell about the feature that we wanted to learn. We will now be able to tell whether we were able to achieve the goal of this chapter.

Let us build the solution and run it. Once run, it will open the application UI. from the left menu panel, click the menu item called “StudentTableListDetail”. It will open the ListDetail screen as a new tab into the screen.



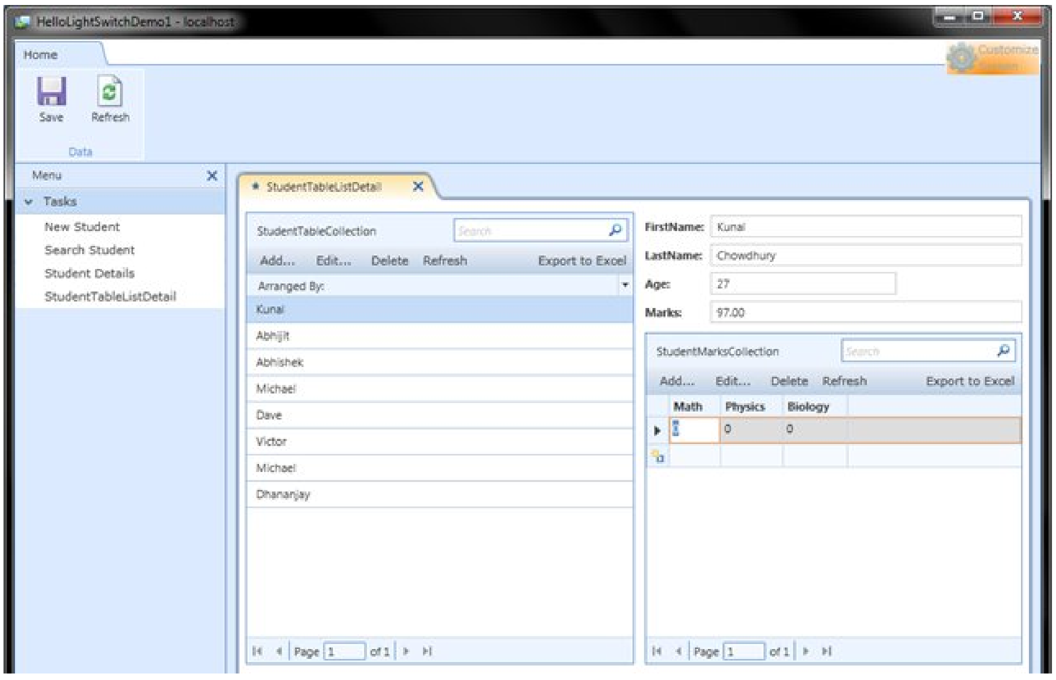
Kunal
Abhijit
Abhishek
Michael
Dave
Victor
Michael
Dhananjay

FirstName:	Kunal
LastName:	Chowdhury
Age:	27
Marks:	97.00

Math	Physics	Biology

You will see that all the records are displayed in the middle panel of the screen. It is only showing the FirstName from the table. You can chose which field to display here by customizing the UI.

In the right panel, you will see two sections. The top section is nothing but a Form where you can view and update the respective records of the selected one. Below to that, there is a DataGrid which actually shows the respective related records from the 2nd table.

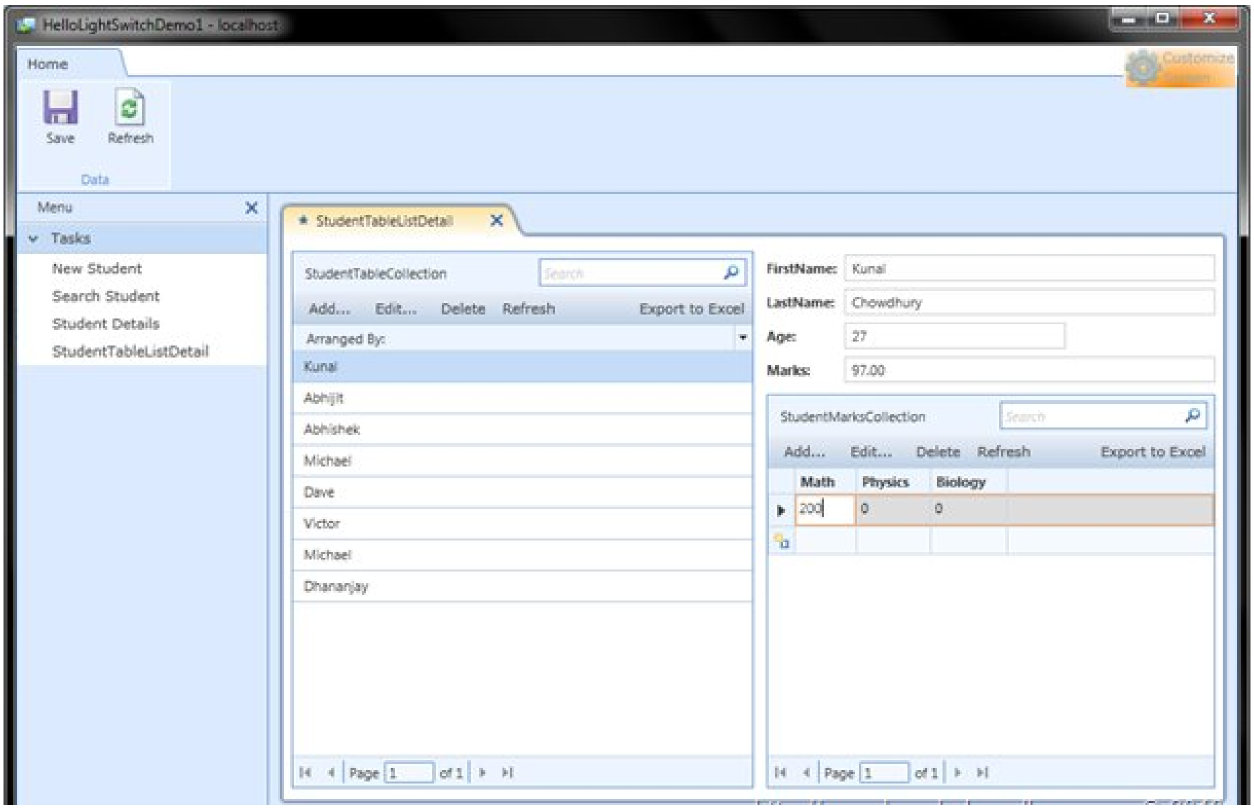


Kunal
Abhijit
Abhishek
Michael
Dave
Victor
Michael
Dhananjay

FirstName:	Kunal
LastName:	Chowdhury
Age:	27
Marks:	97.00

Math	Physics	Biology
	0	0

You can easily add, edit, delete and export records within this DataGrid. You all learnt about it in the previous chapter. Hence, I am not going to explain it more here.



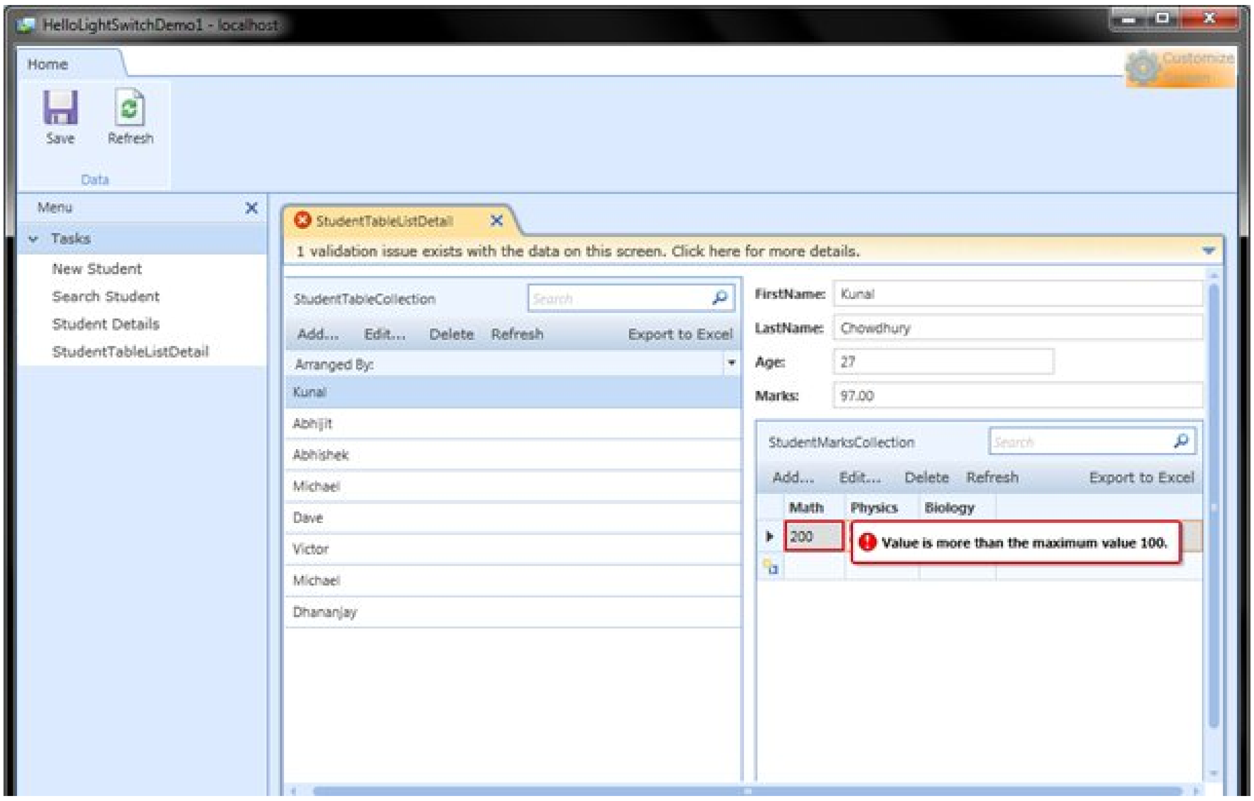
Kunal
Abhijit
Abhishek
Michael
Dave
Victor
Michael
Dhananjay

FirstName: Kunal
LastName: Chowdhury
Age: 27
Marks: 97.00

Math	Physics	Biology
200	0	0

Now comes the part to showcase the custom validation that we added for the fields. We entered the Maximum and Minimum value for the fields. Let us check whether these works actually. What do you think? Let’s do it.

Enter a negative value or a value higher than 100 (100 was our maximum value and 0 was the minimum value, if you can remember). You will see the proper validation message popup in the screen. If you entered a higher value, it will show “Value is more than the maximum value 100” and if you entered a lower value, it will show “Value is less than the minimum value 0”.

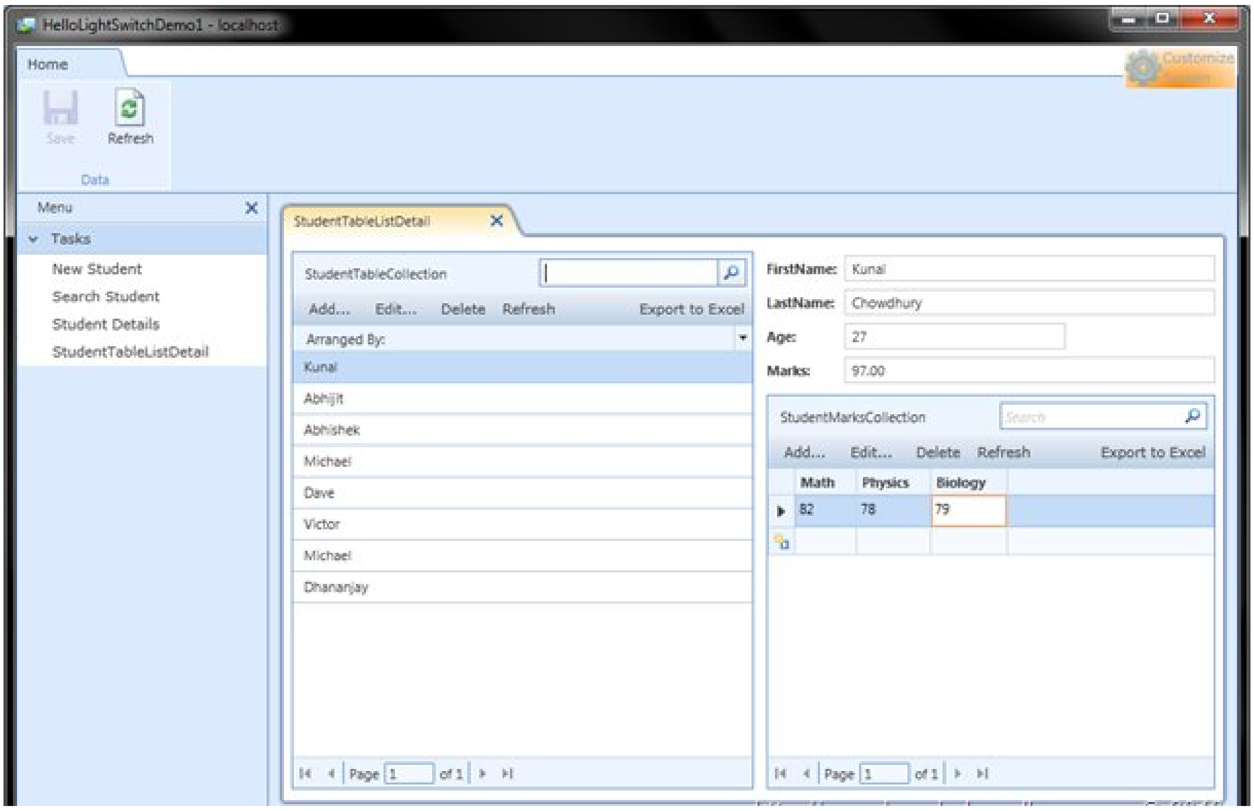


1 validation issue exists with the data on this screen. Click here for more details.

FirstName: Kunal
LastName: Chowdhury
Age: 27
Marks: 97.00

Value is more than the maximum value 100.

Once you corrrect those validation error, you will be now able to save the record.



FirstName: Kunal
LastName: Chowdhury
Age: 27
Marks: 97.00

Math	Physics	Biology
82	78	79

Hey, did you notice one thing? We didn’t enter the validation message there. These are coming from the Tool itself. Everything is inbuilt there. No code. Click, click and click, you professional quality application is ready for you.

End Note

You can see that, throughout the whole application (for all the 4 chapters) I never wrote a single line of code. I never did write a single line of XAML code to create the UI. It is presented by the tool template automatically. It has a huge feature to do automatically. From the UI design to add, update, delete and even sort, filter all are done automatically by the framework.

I hope, you enjoyed this chapter of the series a lot. Lots of figures I used here, so that, you can understand each steps very easily. If you liked this article, please don’t forget to share your feedback here. Appreciate your feedback, comments, suggestion and vote.

I will soon post the next chapter on Custom Validation soon. Till then enjoy reading my other articles published in my Blog and Silverlight Show. Cheers

Beginners Guide to Visual Studio LightSwitch (Part - 5)

Visual Studio LightSwitch is a new tool for building data-driven Silverlight Application using Visual Studio IDE. It automatically generates the User Interface for a DataSource without writing any code. You can write a small amount of code also to meet your requirement.

In my previous chapter “**Beginners Guide to Visual Studio LightSwitch (Part – 4)**” I guided you step-by-step process to create a List and Details screen. There I demonstrated you, how to integrate two or more tables inside a single screen.

In this chapter, I am going to demonstrate you of doing custom validation using Visual Studio LightSwitch. This time we will jump into writing some code to extend the functionalities.

Background

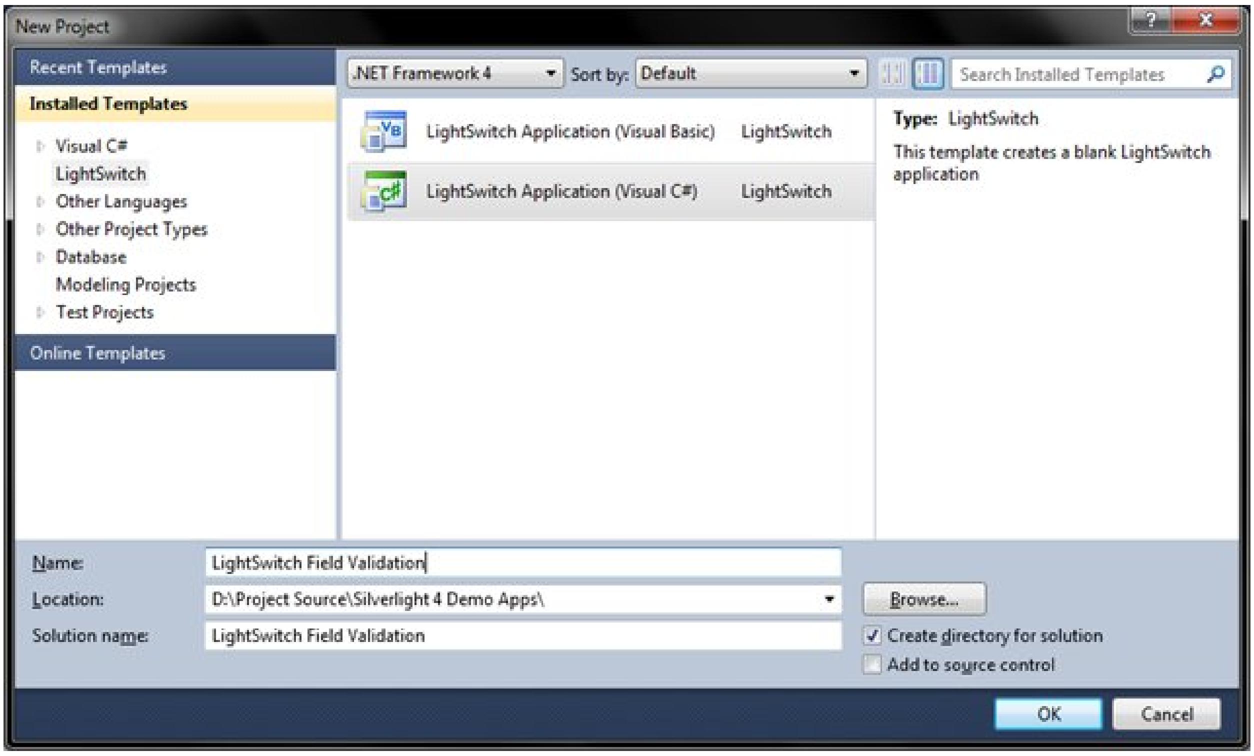
If you are new to Visual Studio LightSwitch, I will first ask you to read the previous three chapters of this tutorial, where I demonstrated it in detail. In my 3rd chapter, I discussed the following topics:

- Creating the List and Details Screen
- UI Screen Features
- Adding a New Table
- Creating the Validation Rules
- Adding Relationship between two tables
- Creating the new List and Details Screen
- Application in Action

In this chapter we learn how to write custom validation for our table fields. Read it and start implementing your own logic. A quick jump to code will be there (for the first time). Enjoy reading the tutorial.

Setting up the basic application

Hope, you already read my first four chapters mentioned above. If not, read them first. It will be easy for you to get the base. Let’s start creating our new LightSwitch project. You know how to do that. Select the “LightSwitch Application for C#” template from the new project dialog. Give a proper name & location for your project and click “OK” to start creating the basic project.



Type: LightSwitch

This template creates a blank LightSwitch application

Name: LightSwitch Field Validation

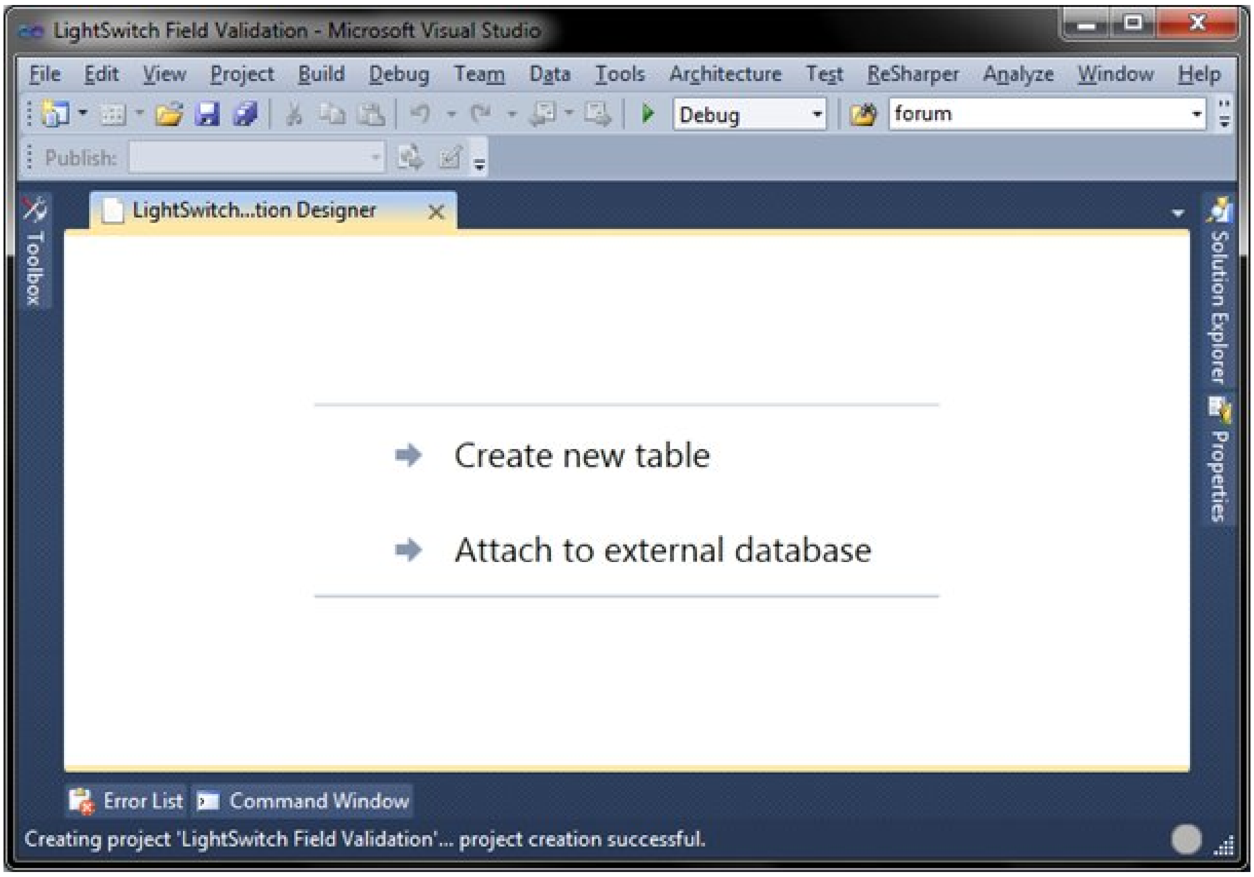
Location: D:\Project Source\Silverlight 4 Demo Apps\

Solution name: LightSwitch Field Validation

☑ Create directory for solution

☐ Add to source control

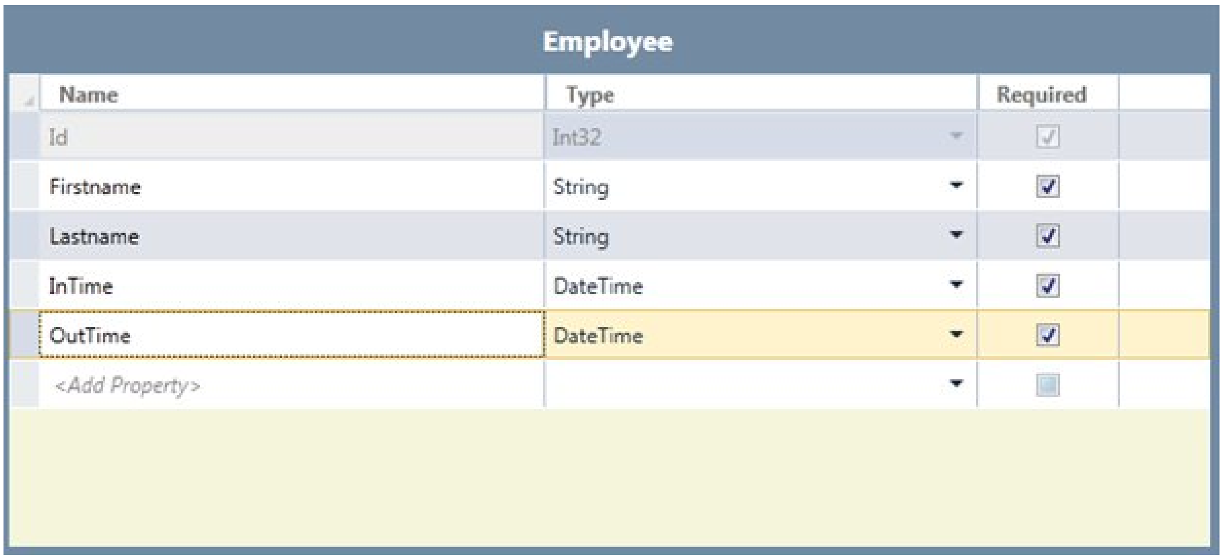
Give some time to Visual Studio 2010 IDE to create the project. Once the project has been created, you will see the below screen inside the Visual Studio IDE:


LightSwitch Field Validation - Microsoft Visual Studio
File Edit View Project Build Debug Team Data Tools Architecture Test ReSharper Analyze Window Help
Debug
forum
Publish:
Toolbox
LightSwitch...tion Designer
Solution Explorer
Properties
Create new table
Attach to external database
Error List
Command Window

Creating project 'LightSwitch Field Validation'... project creation successful.

Now you need to create a new table to start with the application. Create a simple basic table where we can add some custom validation logic. Chose your fields as per your requirement. For our sample, we will create a Employee table and there we will have some columns in it. Let us create 4 custom columns named “Firstname” of type(String), “Lastname” of type(String), “InTime” of type(DateTime) and “OutTime” of type(DateTime) to store the records of employees in and out time of particular day.

Here is the table structure for your reference:



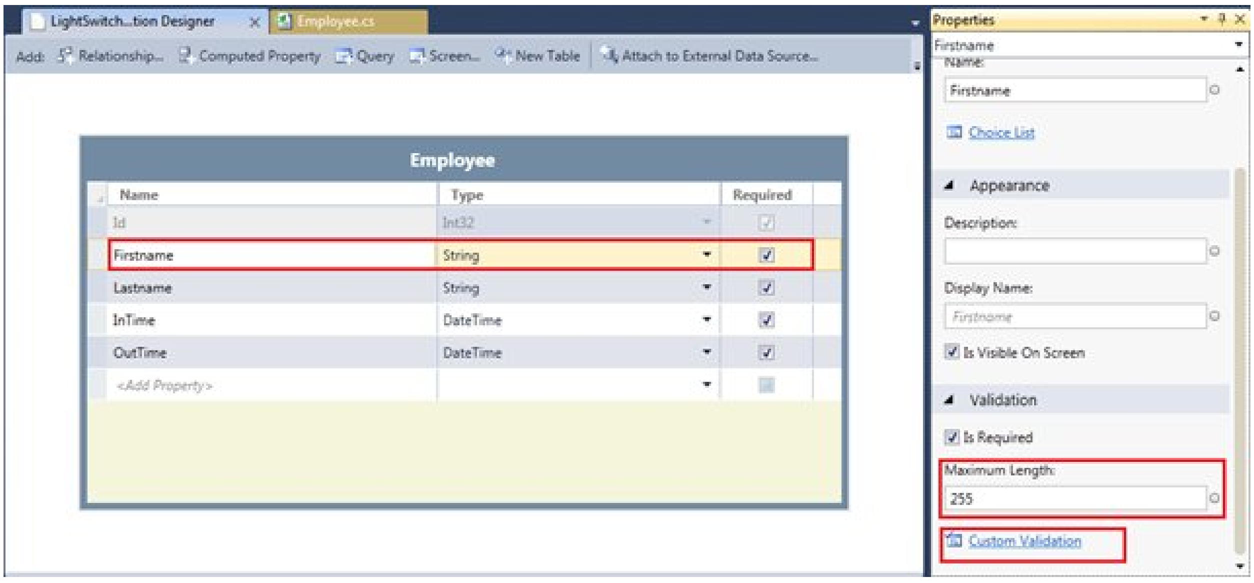
Employee

Name	Type	Required
Id	Int32	✓
Firstname	String	✓
Lastname	String	✓
InTime	DateTime	✓
OutTime	DateTime	✓
<Add Property>		

Check “Required” field to do the null validation automatically by the tool. Hence, we don’t have to do it explicitly. I already demonstrated the same in my earlier chapters. Read them to understand it.

Validating Fields

Now it’s time to start the validation process for each field. Let’s start with the “Firstname”. Click on the row as marked in the below screenshot. Now go to the properties panel and scroll down it to the end. There you will find a grouped item named “Validation”. This panel hosts all the information of validation related to the selected field. Have a look it here:


LightSwitch...tion Designer
Employee.cs
Add: Relationship... Computed Property Query Screen... New Table Attach to External Data Source...
Properties
Choice List
Appearance
Validation
Custom Validation

Employee

Name	Type	Required
Id	Int32	✓
Firstname	String	✓
Lastname	String	✓
InTime	DateTime	✓
OutTime	DateTime	✓
<Add Property>		

Firstname

Name: Firstname

Description:

Display Name: Firstname

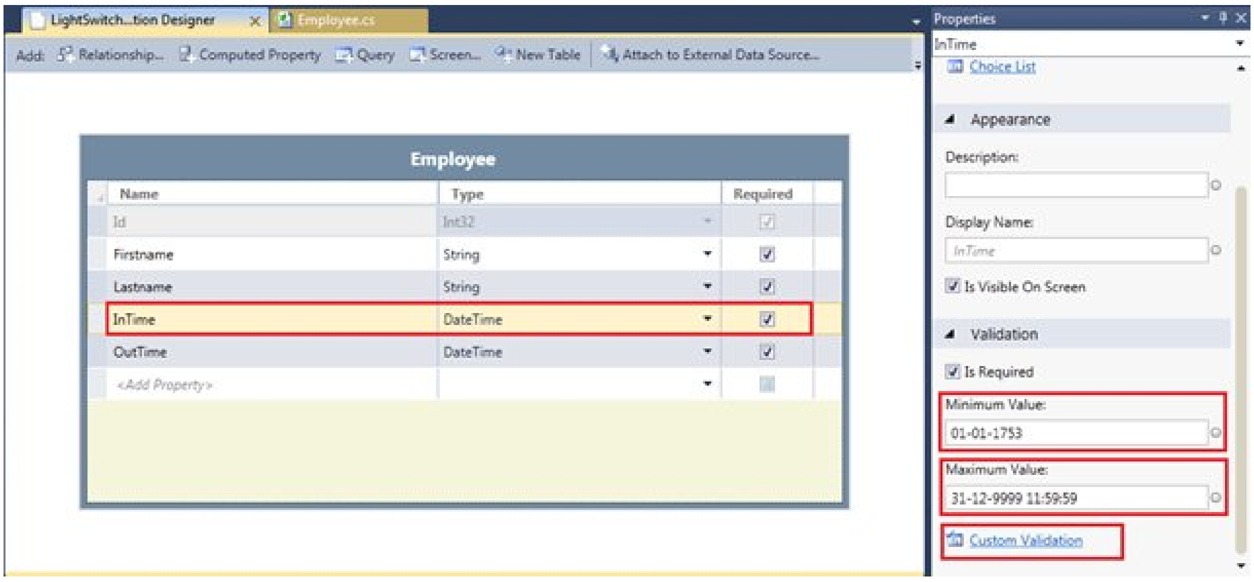
✓ Is Visible On Screen

✓ Is Required

Maximum Length: 255

As “Firstname” field is a String, you will see only “Maximum Length” as the default validation rule. If you want to change the maximum length of the string field, you can do that from here. If you want to write some custom validation rule, you can do it here by just clicking on the “Custom Validation” link from the properties panel. See the above screenshot for the details.

Let us check for the other field. This time we will do it for DateTime field. Select the InTime as shown below and check the properties panel. In the validation tab you will now see two fields “Minimum Value” and “Maximum Value”. Here you can specify your own custom value.


LightSwitch...tion Designer
Employee.cs
Add: Relationship... Computed Property Query Screen... New Table Attach to External Data Source...
Properties

Employee

Name	Type	Required
Id	Int32	✓
Firstname	String	✓
Lastname	String	✓
InTime	DateTime	✓
OutTime	DateTime	✓
<Add Property>		

InTime

Choice List

Appearance

Description:

Display Name: InTime

✓ Is Visible On Screen

Validation

✓ Is Required

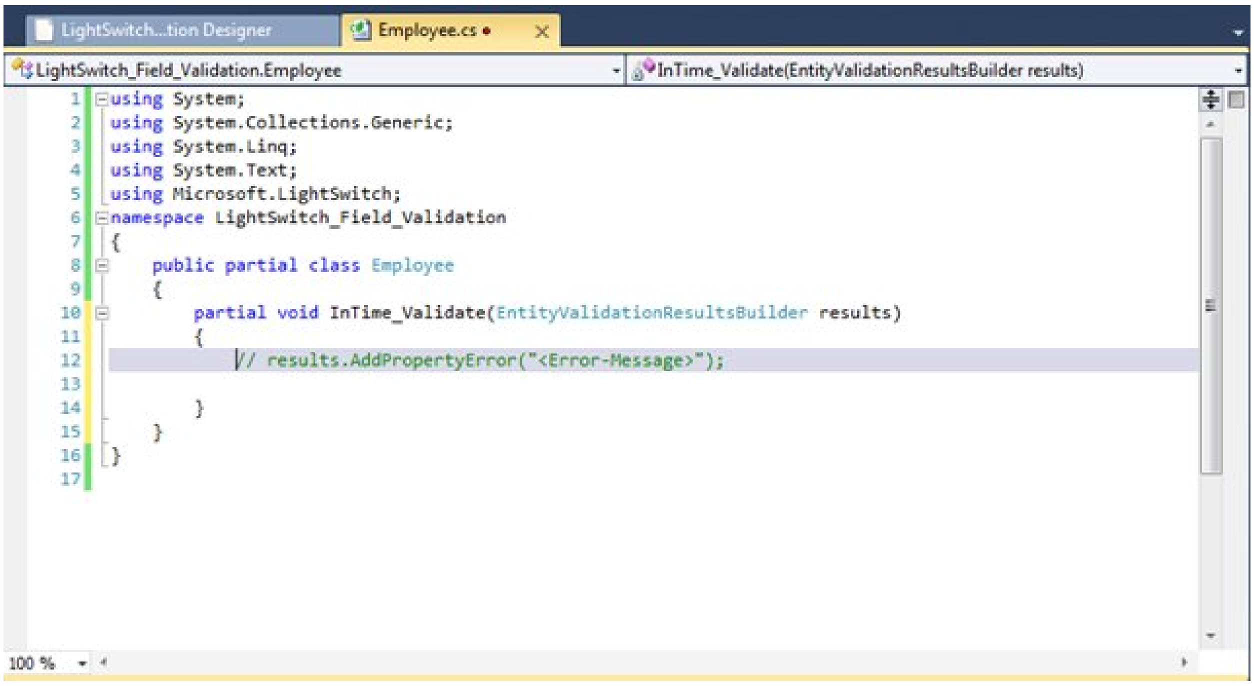
Minimum Value: 01-01-1753

Maximum Value: 31-12-9999 11:59:59

Custom Validation

As mentioned above, we also have a link called “Custom Validation” here. You can write your own logic for validation by clicking it. Let’s do it. This is the first time we are going to write some code for LightSwitch application.

Click the “Custom Validation” link in the properties panel. This will open the code editor window inside the IDE. This editor window is very familiar to you. In the editor window, you will see the class file opened for you named “Employee”. This is nothing but the name of the table. It has a single event there and it looks as below:

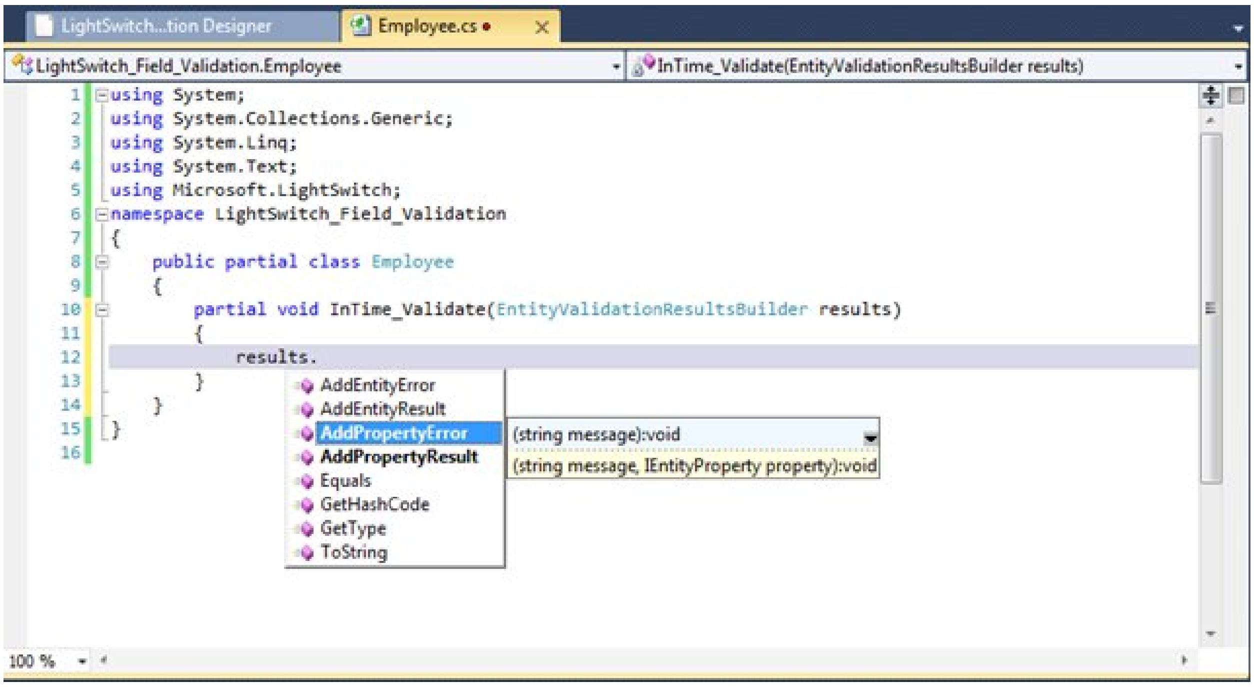

LightSwitch...tion Designer
Employee.cs
LightSwitch_Field_Validation.Employee
InTime_Validate(EntityValidationResultsBuilder results)
100 %

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Microsoft.LightSwitch;
namespace LightSwitch_Field_Validation
{
    public partial class Employee
    {
        partial void InTime_Validate(EntityValidationResultsBuilder results)
        {
            // results.AddPropertyError("<Error-Message>");

        }
    }
}
```

You can do whatever validation logic you want it here. Want to write some code? Oh yeah! Ok, before doing anything, let me show you the API to throw the message on validation error. The output parameter "results" of type "EntityValidationResultsBuilder" has some APIs. If you want to throw some property validation error, use the "AddPropertyError()" method to return back the error message to the end user.

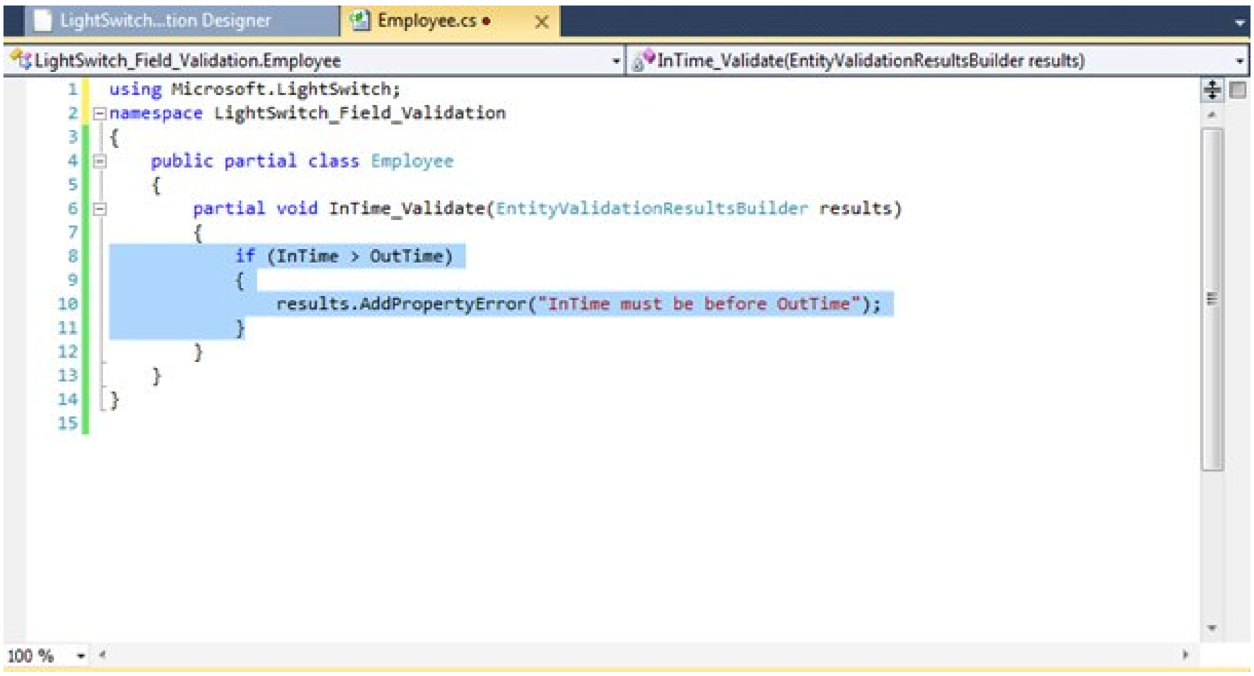
Have a look into the APIs mentioned in the below screenshot:


LightSwitch...tion Designer
Employee.cs
LightSwitch_Field_Validation.Employee
InTime_Validate(EntityValidationResultsBuilder results)
AddEntityError
AddEntityResult
AddPropertyError
AddPropertyResult
Equals
GetHashCode
GetType
ToString
(string message):void
(string message, IEntityProperty property):void
100 %

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Microsoft.LightSwitch;
namespace LightSwitch_Field_Validation
{
    public partial class Employee
    {
        partial void InTime_Validate(EntityValidationResultsBuilder results)
        {
            results.
        }
    }
}
```

Ok, as per our requirement of the application, the OutTime of the employee must be after the InTime. An employee can't go out if he/she is not in, am I right? So, let's write some logic for that. If InTime is greater than OutTime (by any mistake by the end user, of-course), we will throw the error message to the user. We will use AddPropertyError() which takes string as message.

Have a look into the implementation here:

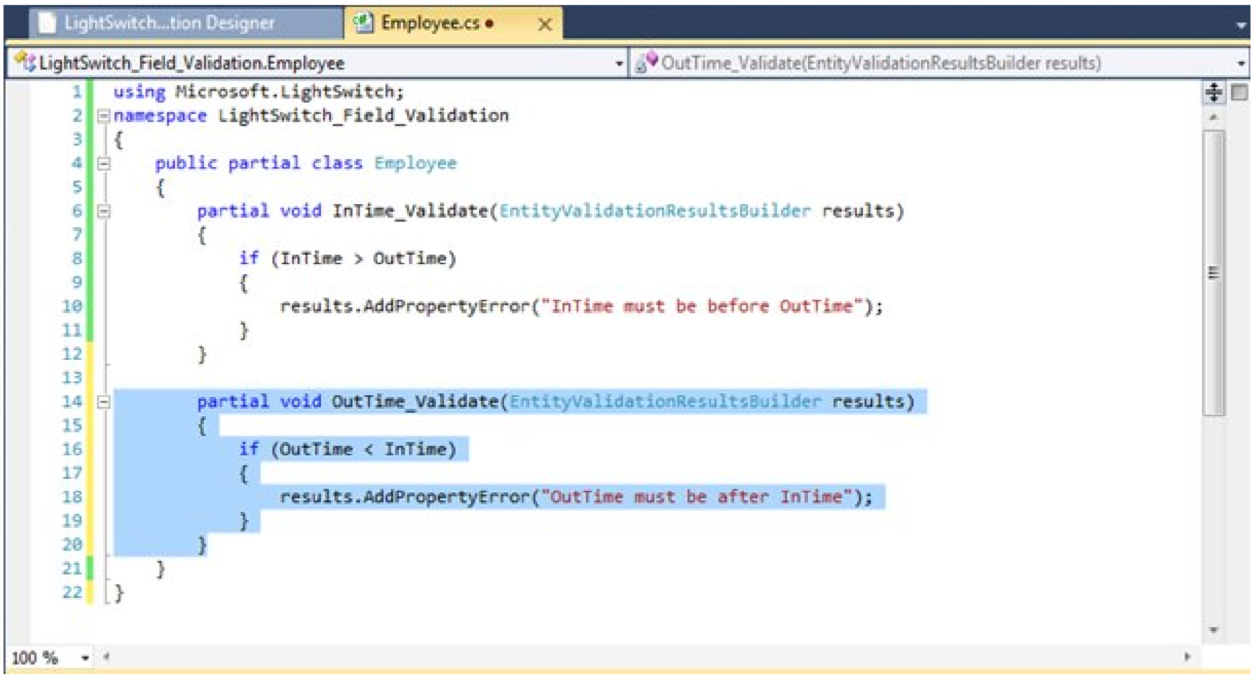

LightSwitch...tion Designer | Employee.cs •

LightSwitch_Field_Validation.Employee | InTime_Validate(EntityValidationResultsBuilder results)

```
using Microsoft.LightSwitch;
namespace LightSwitch_Field_Validation
{
    public partial class Employee
    {
        partial void InTime_Validate(EntityValidationResultsBuilder results)
        {
            if (InTime > OutTime)
            {
                results.AddPropertyError("InTime must be before OutTime");
            }
        }
    }
}
```

Hope, you got it. So, do it for the OutTime field too. Go to the table designer and select the row for “OutTime”. From the properties panel, click the “Custom Validation”, as we did it earlier. You will see the event generated for that in the code editor window.

Here, check whether OutTime is less than InTime. If so, throw the error as shown below:


LightSwitch...tion Designer | Employee.cs •

LightSwitch_Field_Validation.Employee | OutTime_Validate(EntityValidationResultsBuilder results)

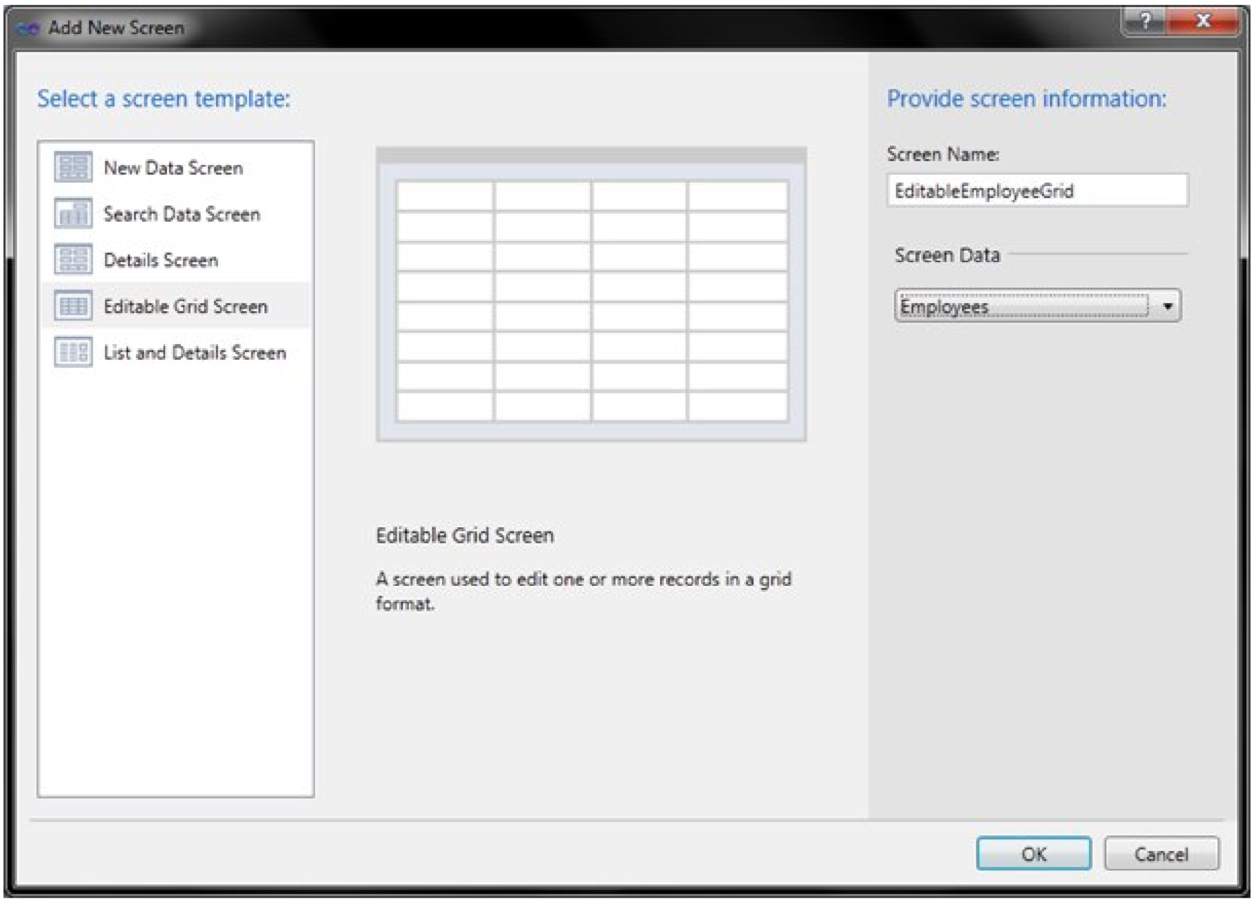
```
using Microsoft.LightSwitch;
namespace LightSwitch_Field_Validation
{
    public partial class Employee
    {
        partial void InTime_Validate(EntityValidationResultsBuilder results)
        {
            if (InTime > OutTime)
            {
                results.AddPropertyError("InTime must be before OutTime");
            }
        }

        partial void OutTime_Validate(EntityValidationResultsBuilder results)
        {
            if (OutTime < InTime)
            {
                results.AddPropertyError("OutTime must be after InTime");
            }
        }
    }
}
```

We are ready with the custom validation. If you want to check for the various default validation rule for different controls, try it out. The custom validation is same for all the cases and if any issue, let me know. I will try to answer you as soon as possible.

See it in Action

Before closing this tutorial, you must want to see the demo in action. You must want to check whether we achieve our requirement or not. Hence, create a screen to check it out. For our sample, we will create an Editable Grid Screen for simplicity. Add the screen and don't forget to chose the Employees table from the Screen Data dropdown.



Editable Grid Screen

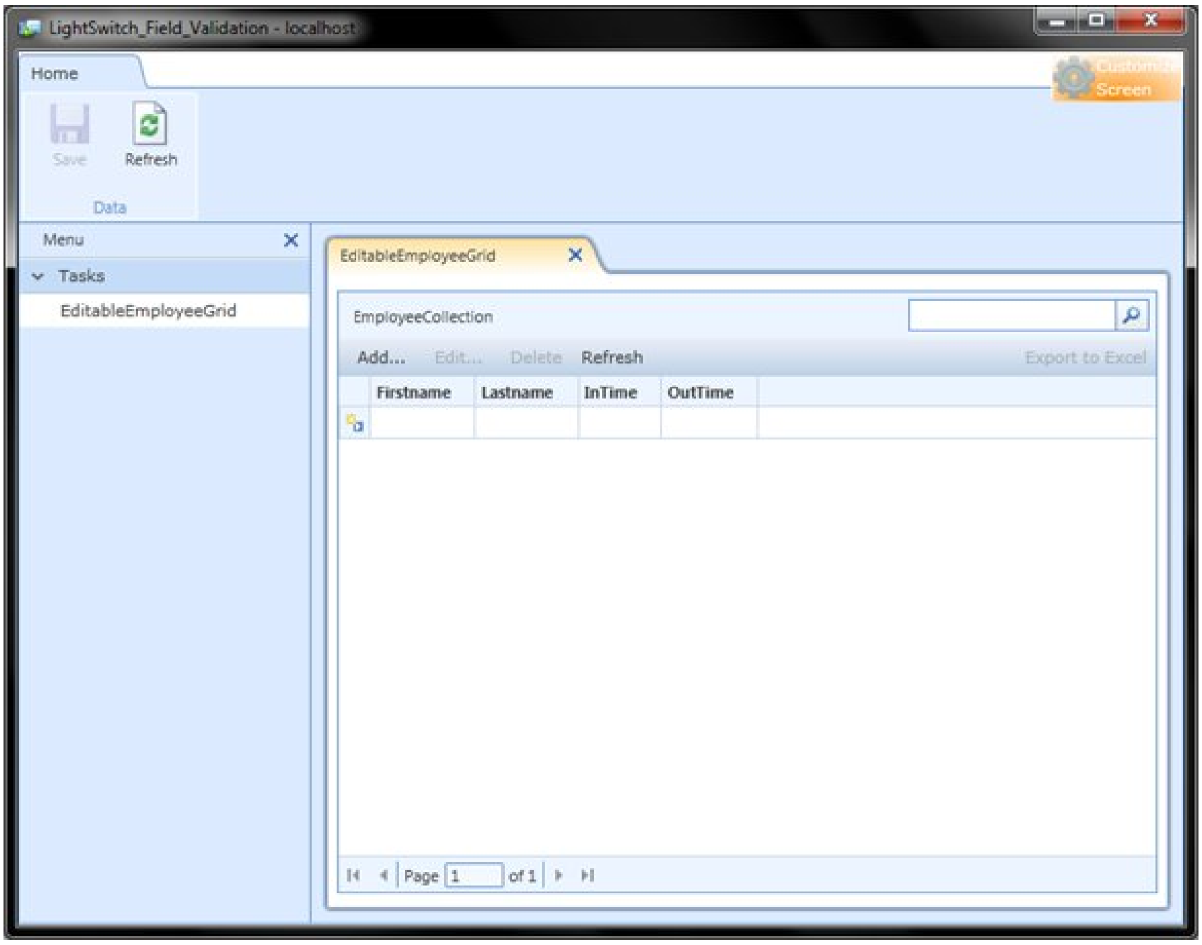
A screen used to edit one or more records in a grid format.

Provide screen information:

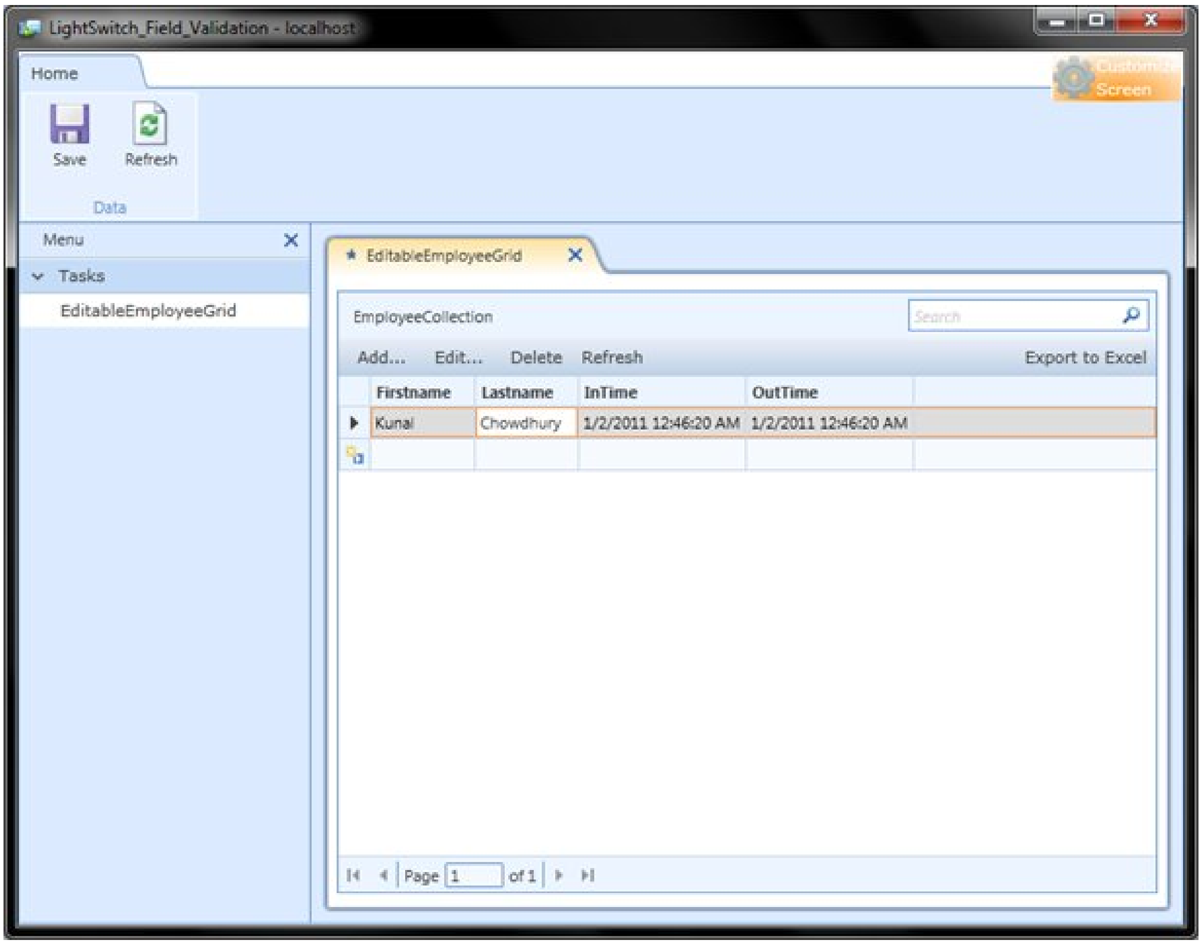
Screen Name: EditableEmployeeGrid

Screen Data: Employees

Click "OK" and the IDE will create the screen for you. To show this demo, we don't need to do any UI customization. Build the project. Hope, it will build successfully. If any issue, try to fix the error. Now run the application. You will see that, the application is up into the screen with the editable datagrid page opened in the main tab.

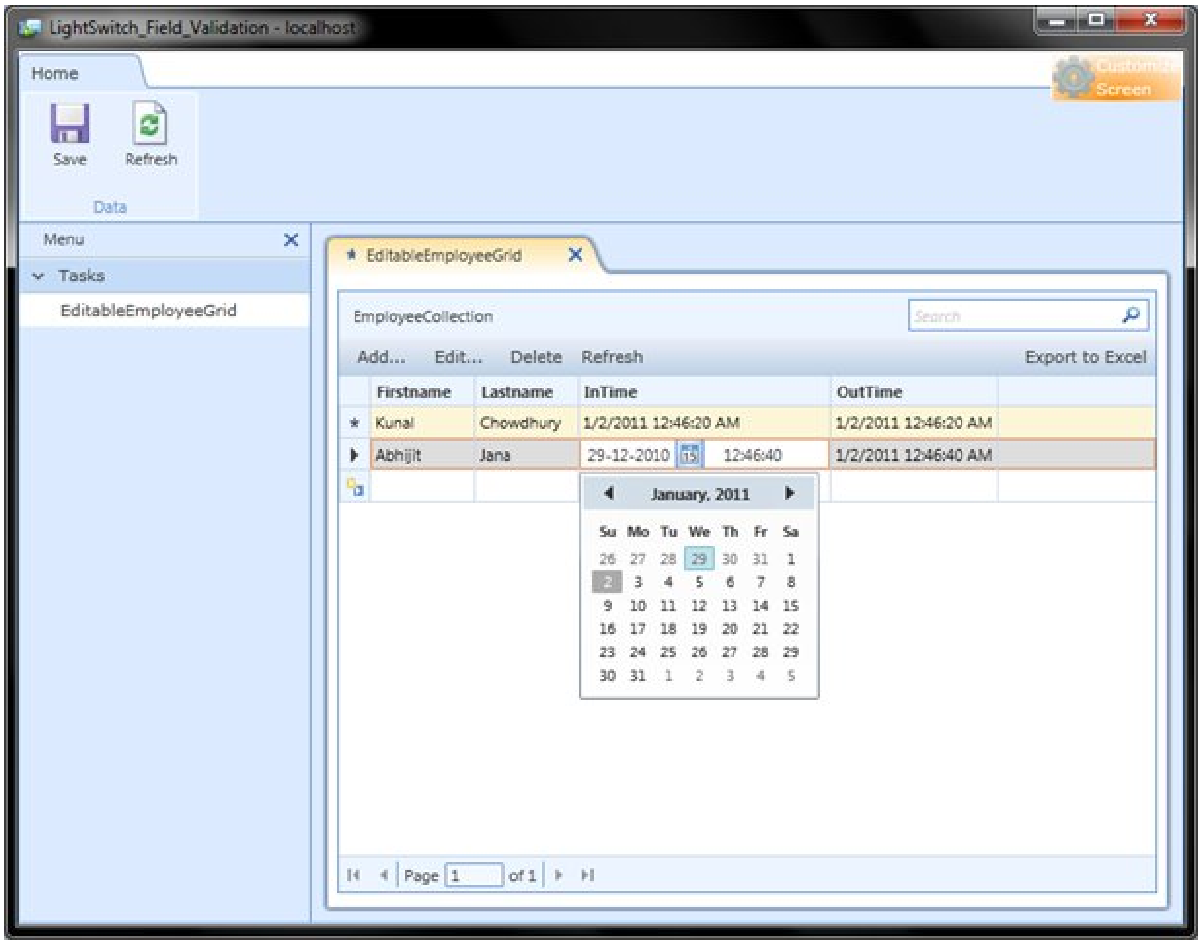


Try to add some records and you will notice that, the InTime and OutTime field has been automatically populated by the application. It will show you the current date time. Try to save the record. It will save properly without any error as the DateTime inserted is perfect.



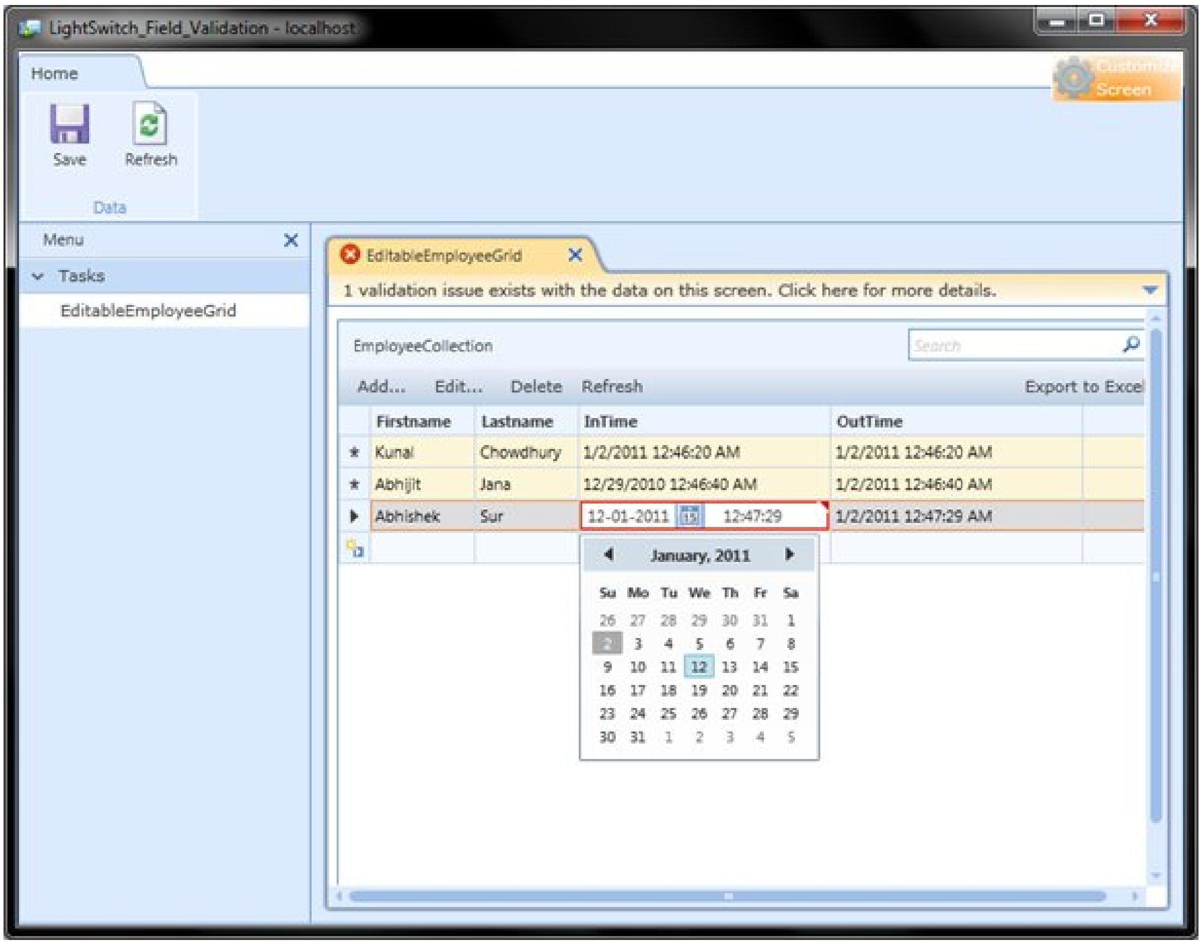
Firstname	Lastname	InTime	OutTime
Kunal	Chowdhury	1/2/2011 12:46:20 AM	1/2/2011 12:46:20 AM

Now, add a new record. Click on the InTime column. You will see that the calendar control. Yes, as you selected the column as DateTime type, the tool automatically added the calendar control for you. Now chose an earlier date from the calendar for the InTime field. Don’t change anything for the OutTime. Try to save the record. Woot, this time also the record saved successfully, because the validation rule passed the case.



Firstname	Lastname	InTime	OutTime
Kunal	Chowdhury	1/2/2011 12:46:20 AM	1/2/2011 12:46:20 AM
Abhijit	Jana	29-12-2010 12:46:40	1/2/2011 12:46:40 AM

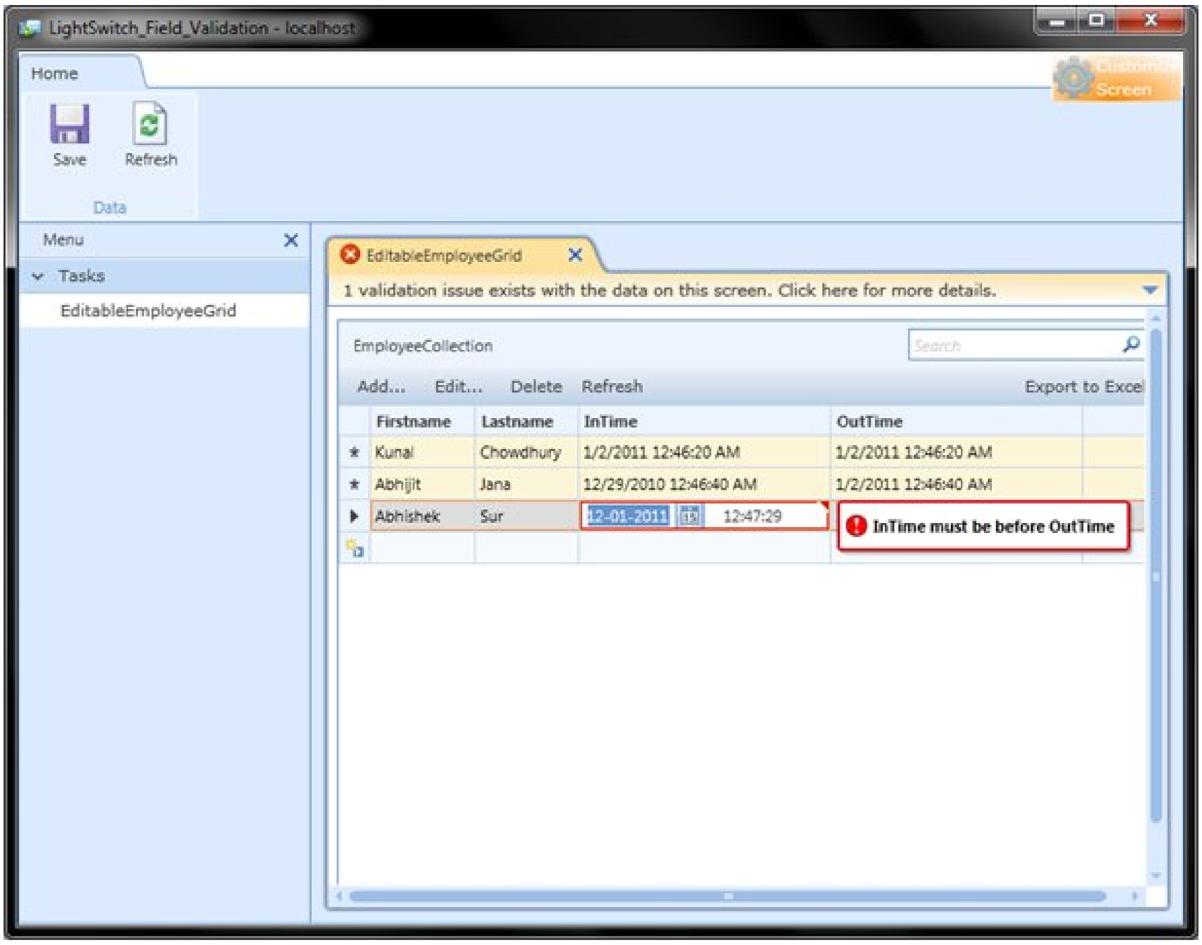
Let us do something wrong now. Try to select a future date for the InTime. You will see the validation rule fails and the error message shown in the screen with a Red border to the InTime field. Have a look it here:



1 validation issue exists with the data on this screen. Click here for more details.

	Firstname	Lastname	InTime	OutTime
*	Kunal	Chowdhury	1/2/2011 12:46:20 AM	1/2/2011 12:46:20 AM
*	Abhijit	Jana	12/29/2010 12:46:40 AM	1/2/2011 12:46:40 AM
▶	Abhishek	Sur	12-01-2011 12:47:29	1/2/2011 12:47:29 AM

If you place the cursor into the Date field now, you will see the same error message that we wrote in the code window for the validation error of the InTime field.

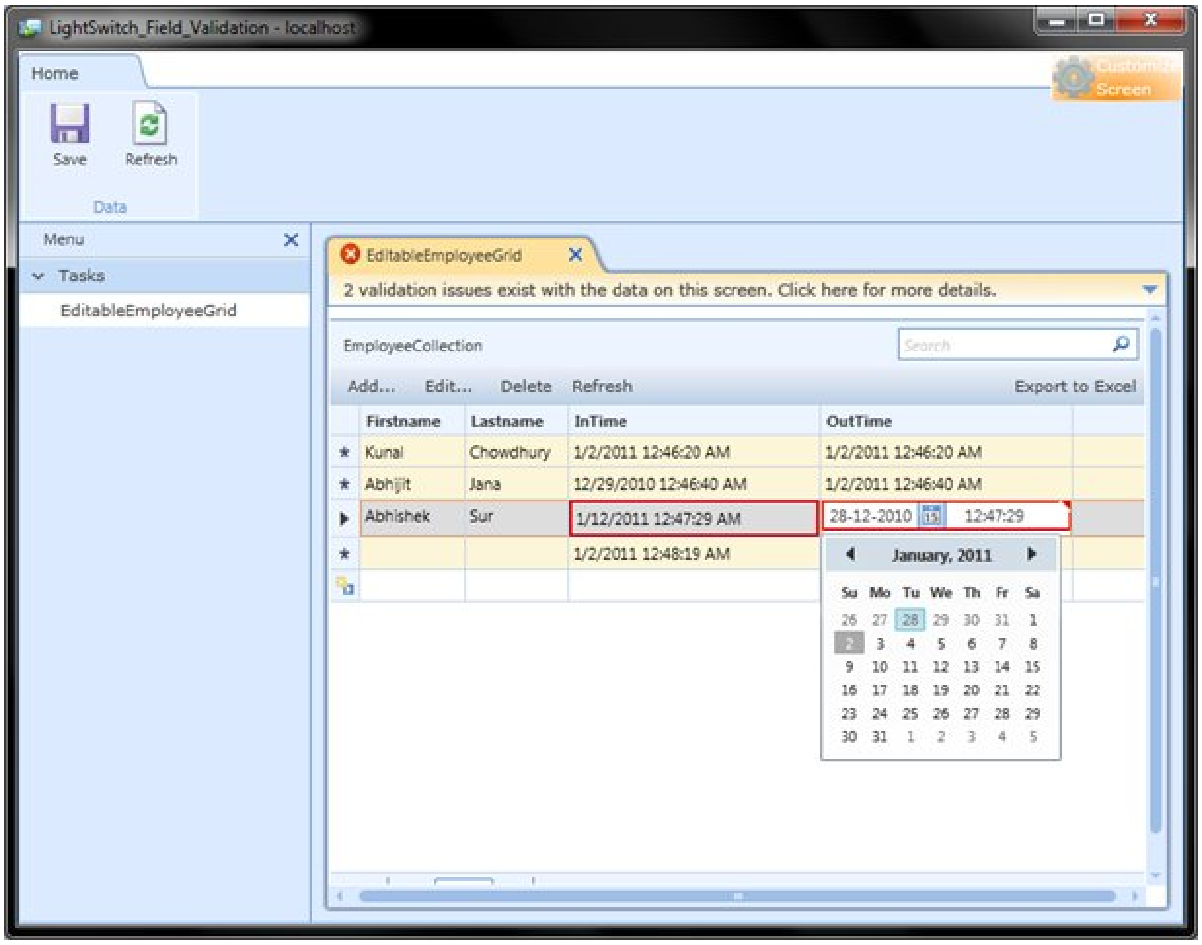


1 validation issue exists with the data on this screen. Click here for more details.

	Firstname	Lastname	InTime	OutTime
*	Kunal	Chowdhury	1/2/2011 12:46:20 AM	1/2/2011 12:46:20 AM
*	Abhijit	Jana	12/29/2010 12:46:40 AM	1/2/2011 12:46:40 AM
▶	Abhishek	Sur	12-01-2011 12:47:29	

InTime must be before OutTime

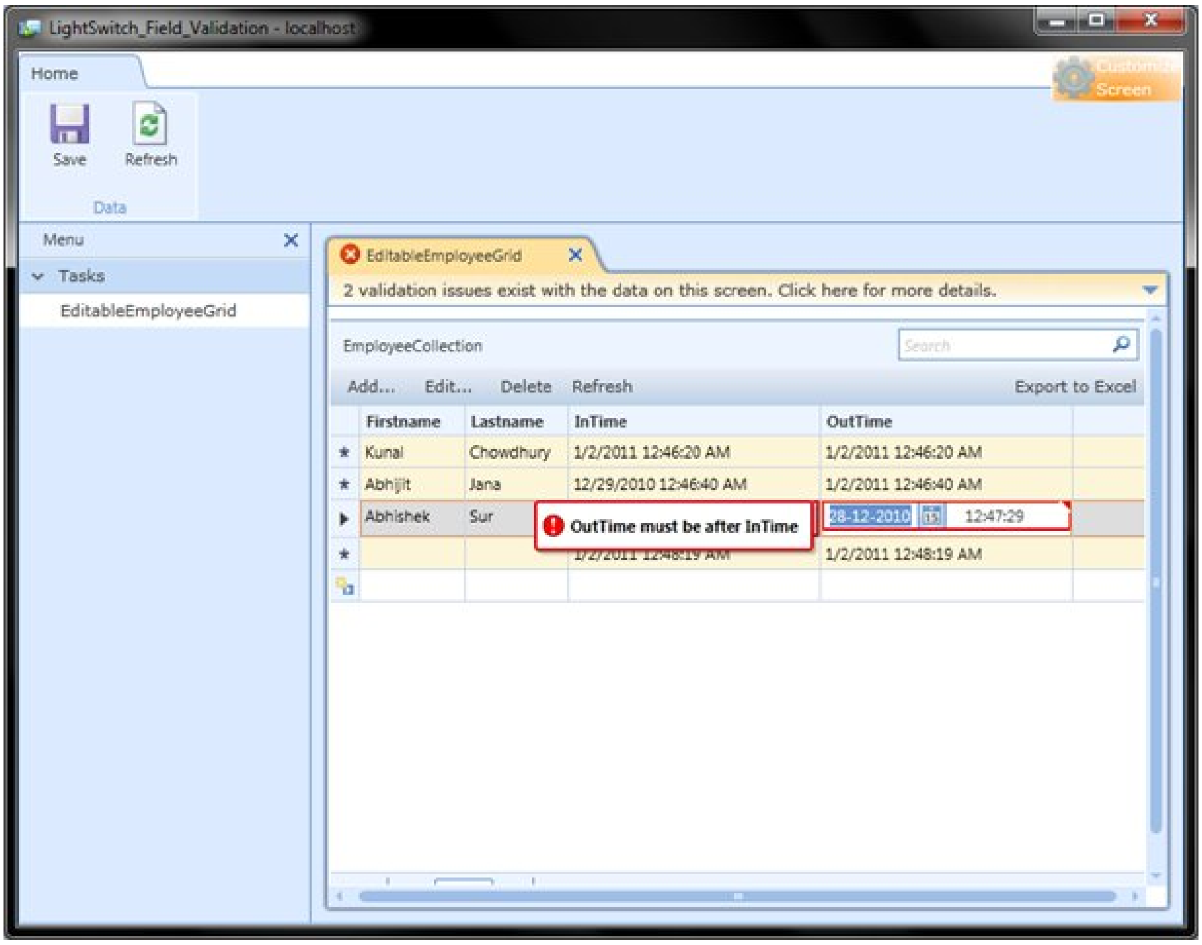
Try to change the OutTime to a previous date. Woo, you will see the validation rule failed here too and the field automatically marked with a Red border.



2 validation issues exist with the data on this screen. Click here for more details.

	Firstname	Lastname	InTime	OutTime
*	Kunal	Chowdhury	1/2/2011 12:46:20 AM	1/2/2011 12:46:20 AM
*	Abhijit	Jana	12/29/2010 12:46:40 AM	1/2/2011 12:46:40 AM
▶	Abhishek	Sur	1/12/2011 12:47:29 AM	28-12-2010 12:47:29
*			1/2/2011 12:48:19 AM	

If you place the cursor inside the field, you will see the error message shown into the screen. This is the same message we entered for the validation error of the OutTime field.

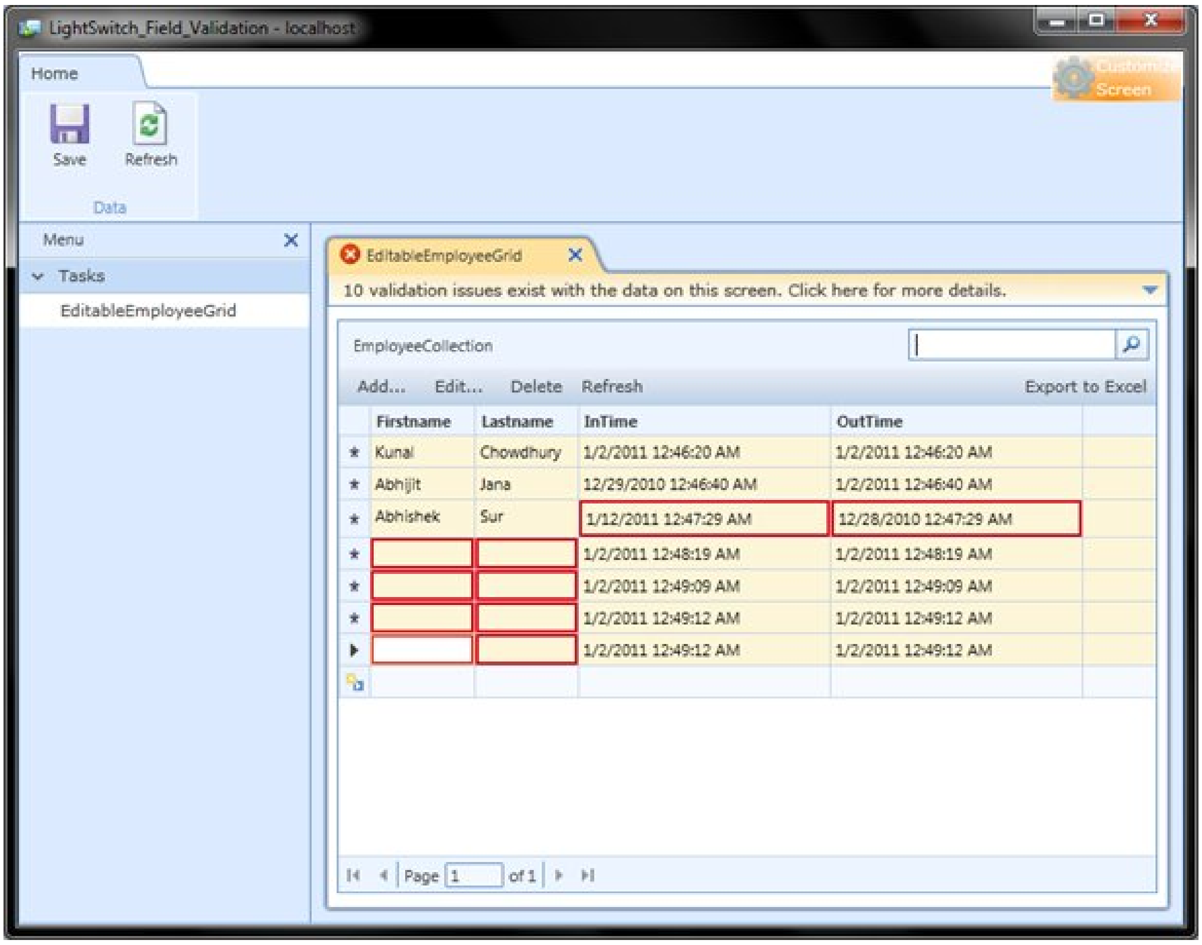


2 validation issues exist with the data on this screen. Click here for more details.

	Firstname	Lastname	InTime	OutTime
*	Kunal	Chowdhury	1/2/2011 12:46:20 AM	1/2/2011 12:46:20 AM
*	Abhijit	Jana	12/29/2010 12:46:40 AM	1/2/2011 12:46:40 AM
▶	Abhishek	Sur		28-12-2010 12:47:29
*			1/2/2011 12:48:19 AM	1/2/2011 12:48:19 AM

OutTime must be after InTime

Try to save the table data after adding some records with empty text for the Firstname and Lastname field. You will see that, here also the validation failed because we marked those field as Required field. Hence, null record will throw validation error.



10 validation issues exist with the data on this screen. Click here for more details.

Firstname	Lastname	InTime	OutTime
Kunal	Chowdhury	1/2/2011 12:46:20 AM	1/2/2011 12:46:20 AM
Abhijit	Jana	12/29/2010 12:46:40 AM	1/2/2011 12:46:40 AM
Abhishek	Sur	1/12/2011 12:47:29 AM	12/28/2010 12:47:29 AM
		1/2/2011 12:48:19 AM	1/2/2011 12:48:19 AM
		1/2/2011 12:49:09 AM	1/2/2011 12:49:09 AM
		1/2/2011 12:49:12 AM	1/2/2011 12:49:12 AM
		1/2/2011 12:49:12 AM	1/2/2011 12:49:12 AM

Correct the validation error and now you will be able to save the records without any issue.

End Note

You can see that, throughout the whole application (for all the previous 4 chapters) I never wrote a single line of code. I never did write a single line of XAML code to create the UI. It is presented by the tool template automatically. It has a huge feature to do automatically. From the UI design to add, update, delete and even sort, filter all are done automatically by the framework.

Only in this chapter, I introduced the code with you to extend the validation rule. I hope, you enjoyed this chapter of the series too. Huge nos. of figures I used here, so that, you can understand each steps very easily. If you liked this article, please don't forget to share your feedback here. Appreciate your feedback, comments, suggestion and vote.