OTK Custom db**Installer 2.0**

Documentation

Revision 1.3

December 2007

Table of Contents

1 Getting Started

1.1 What is Custom dbInstaller?

OTK Custom dbInstaller, also known as dbI in short, is an application database deployment tool that specifically designed for Independent Software Vendors (ISVs) and any technical personnel that leverage on Microsoft SQL Server database and Windows 32 bit platform. It was being developed by OTK Web Solutions, with an initial intention to provide full application deployment automation to any of their future products that utilizes SQL Server database. Due to the fact that such a product with affordable price is hard to find in the market, OTK decided to build their own database installer and released their first free version on 2nd January 2007.

After collecting some feedback and going through some dogfood experiences with their own software products, OTK finally decided to enhance its first version to a more feature rich, practical, and feasible database deployment tool.

1.2 Features and Benefits

OTK Custom dbInstaller 2.0 has the following features and benefits:

- **Provides Complete Automated Deployment Experience**
  - Easily to be included in developer's *.NET Setup/Web Setup Project* to provide full-fledge application and database deployment automation.
  - Automate database deployment tasks such as *backup, restore, custom SQL script execution, update application configuration file* and more.

- **Extremely Agile based on Real World Experience**
  - Capable of *showing/hiding any wizard dialog boxes* at runtime based on user knowledge level.
  - Developers can *provide schedule backup services* to their end users that do not have SQL Server Agent, *e.g. SQL Server Express Edition*. They can just configure dbI just to run the backup and add the command line to *Windows Task Scheduler*.

- **Cut Down Support Costs and Improve Customer Service**
  - *Provide fixes, upgrades,* and *data migrations* without travelling to user's site by writing your own *SQL script file* and distribute it together with dbI.
  - Making repetitive and proven database maintenance execution plan consistent by packaging it into a single distributable support tool.
  - Configure dbI just to fix your user's application *connectionString* when the SQL Server has been shifted, reconfigured, or due to any other circumstances.

- **Play Safe! Database is Crucial!**
  - Options to backup existing database before install/uninstall.
  - Options to *DETACH* database only instead of *DROPPING* it.

- *Get alerted* when you or your users have chosen certain UNSAFE combination of options.

1.3 System Requirements

OTK Custom dbInstaller is capable of supporting and running on the following environments:

- Require any one of the following operating systems (32bit only):
  - *Windows XP Service Pack 2*
  - *Windows Server 2003 Service Pack 2*
  - *Windows Vista*
- Require .NET Framework 2.0
- Require any one of the following Microsoft SQL Server editions:
  - *SQL Server 2000*
  - *SQL Server 2000 Desktop Engine (MSDE 2000)*
  - *SQL Server 2005*
  - *SQL Server 2005 Express Edition*
- Microsoft Visual Studio 2005 or above is required if you want to add dbI as a custom action to your setup project/Web setup project.

1.4 Downloading and Installing Custom dbInstaller 2.0

OTK Custom dbInstaller can be downloaded from the following location:

http://otkdownload.blogspot.com/

After downloading, run the **otkdbi_setup.msi** file and follow the instructions on the screen.

2 Understanding Custom dbInstaller 2.0

2.1 Technical Overview

Custom dbInstaller is a standalone executable program file running on Windows platform. It based on the application configuration defined by developers to carry out its missions. It also supports command line arguments in order to cater for the following needs:

- To simplify developer's task by providing only one single executable file that does everything, such as installation and uninstallation.

- Due to the reason that the users may change the target application path, which normally decides where the program configuration file stores. In order to allow dbI to update the configuration, dbI somehow need to allow the configuration path to be dynamically passed during runtime.

  "ⓘ ***NOTE:*** *Only when the database is completely installed, then only the program can decide how to get connected to the application database. Since dbI is responsible in the database installation, it can help the application to update its database connection string resided in the application's configuration file.* "

- To let dbI knows when to perform installation and when to perform uninstallation. The developers can just pass in the parameter to the same executable file when adding as custom action in their setup project.

  "ⓘ ***NOTE:*** *To enable dbI to be successfully launched while being added as custom uninstallation action, it is recommended that you use Otk.Dbi.CustomActions.dll as the InstallerClass.* "

- To provide greater flexibility and conveniences to the technical support to perform additional tasks, such as:
  - *On-site troubleshooting the application database by just running dbI without running the application installer.*
  - *Add scheduled tasks to Windows Task Scheduler to perform backup or other database maintenance jobs.*

Another thing is the use of **otkdbi.exe.config** to store the customization options defined by the developers. This will ease the following two major support scenarios:

- **Remote Support.** If in case dbI had been wrongly configured prior to the distribution of installer package, or there is an exceptional environment configuration at the user's end that makes the standard database installation plan fails, the use of XML configuration file makes the technical support possible to guide the end users in fixing up the dbI configuration via phone, e-mail, or online instant messaging (IM). This will definitely improve the support efficiency and reduce the support costs.

- **Onsite Support.** While having onsite support, the use of XML configuration file will give more conveniences to the technical support to investigate the cause of database

deployment failure rather than having everything hidden in the binary or encrypted text without knowing what was happening. These will definitely wasting the cost of the trip or slowing down the troubleshooting process.

2.2 How It Works?

2.2.1 How the Wizard Dialog Boxes work?

Before going into deeper on how each dialog box works, it is important for you to understand the overall concept of the Custom dbInstaller's wizard dialog box. The following diagram gives you the overall picture of how the wizard flows:

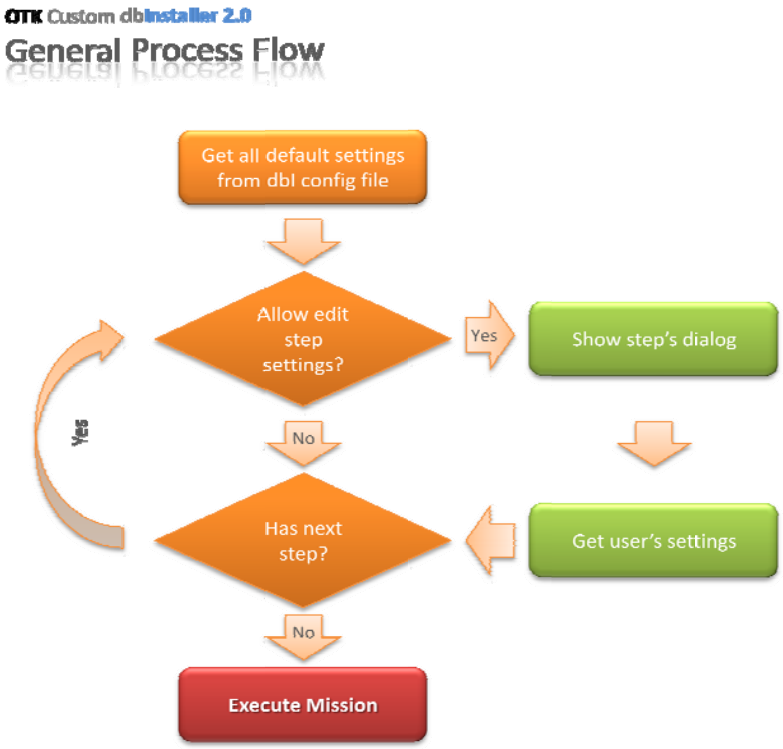

Figure 2-1 General Process Flow diagram giving you the concept of how the wizard flows

As you can see in the above figure, first of all, Custom dbInstaller reads all the default settings predefined by the developer from the **otkdbi.exe.config** file and stores them in the memory. Then, it checks whether the settings of the first dialog, that is the **SQL Server Connection** dialog box, is allowed to be edited by the user or not. If yes, it will show the **SQL Server Connection** dialog box. When the user clicks on the **Next** button, it will validate the settings and prompt the user for error. If the settings are valid, it will update the user-defined settings to the ones in the memory and proceed to the next step. This process will keep repeating for the following dialog boxes until it reaches the end of the wizard.

> ***NOTE:*** *If the **SQL Server Connection** dialog box is the only dialog to be allowed, the label of the **Next** button will change to **Install**. Likewise, if it is running on uninstallation mode, the label of the button will change to **Uninstall** instead of **Next**.*

Take a look at the following example showing the steps to accomplish the database installation process:

1. Edit SQL Server Connection
2. Edit Installation Options
3. Edit Database File Location

If the developer allows step 1 and 3 to be edited and not allowing step 2, when executing the installation mission, the process will based on the user's settings except step 2's. In other words, the settings for step 2 will be taken from config file's defaults. Likewise, if step 1 and 3 are disabled, leaving only the step 2 to be edited by the user, the mission will be executed based on the config file default settings except step 2's. The step 2's settings will be based on user's settings.

2.2.2 Understanding the SQL Server Connection Wizard Dialog

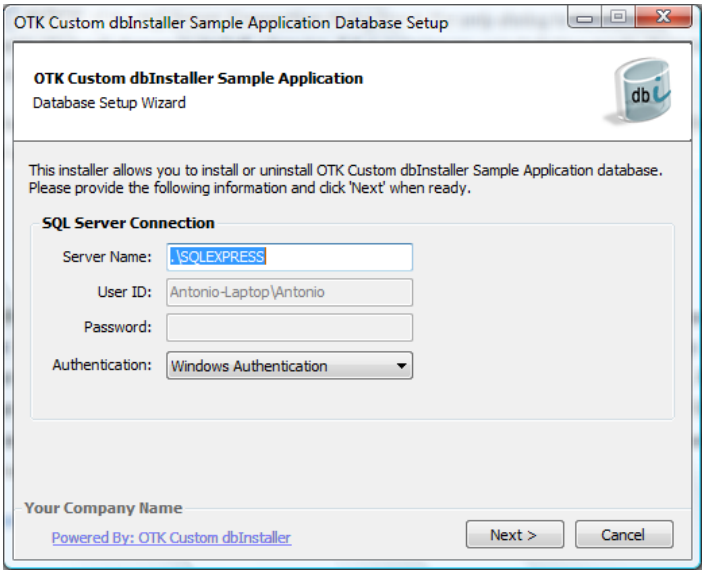

Figure 2-2 SQL Server Connection wizard dialog

The SQL Server Connection wizard dialog is the first dialog box to be shown if it is enabled, may it be the installation mode or uninstallation mode. It is the fundamental point to determine the success of the entire mission because you just can't do anything without getting the database connected.

In **Figure 2-2**, what you need to notice is the **Authentication** dropdown list. When **Windows Authentication** is selected, the **User ID** and **Password** text boxes will be disabled and the **User ID** text box will be filled with the current logged on user id. These text boxes will be enabled when you select **SQL Server Authentication**. This behavior is quite similar to the login screen that you found from **SQL Server Management Studio**. The difference is that, when selecting **SQL Server Authentication**, Custom dbInstaller will fill the text boxes with the default values from the **otkdbi.exe.config** file. Only until the credential is being corrected and the **Next** button is clicked, the

default values will change. If the user clicked the **Back** button, selected **Windows Authentication**, and chose back the **SQL Server Authentication** again, the **User ID** and **Password** text boxes will be filled by the recently entered and validated credential data. Once again, if the user changes the login information with an invalid one and clicks **Next**, *the most recent valid data is always preserved*.

For more information about the configuration of SQL Server Connection settings, see ***TABLE 2 – SQL SERVER CONNECTION SETTINGS*.**

2.2.3 Understanding the Installation Options Wizard Dialog

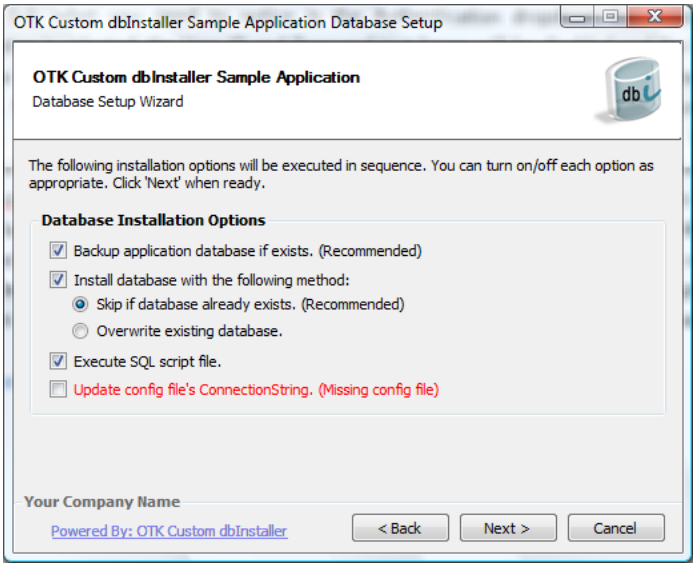

Figure 2-3 Database Installation Options wizard dialog

The Database Installation Options wizard dialog provides the following options for advance users to control the way they want Custom dbInstaller to be:

1. **Backup application database if exists. (Recommended)**
   Checking this check box will ask Custom dbInstaller to backup the existing database whose name is the same as what has been specified in the **otkdbi.exe.config** file. If the specific database does not exist, this task will be skipped. This is the recommended setting to prevent lost of current data on the existing database or for future recovery purposes. When the check box is unchecked, its caption will turn into red color text if:

   - *The overwriting existing database option is selected.*
   - *The executing SQL script file option is selected.*

   This is to alert the user that the current combination of options may not be safe to be implemented. This is just a warning. The user can still click the **Next** button to proceed. If you check the backup check box, you will see that it turns back to normal color.

2. **Install database with the following method:**
   a. ***Skip if database already exists. (Recommended)***
      This will ask Custom dbInstaller to skip performing database restoration if the database with the same name already exists on the specific server instance. This is

the recommended setting because it prevents the existing data from being overwritten with the new one. Unselecting this radio button will cause the caption of the unchecked backup check box turns into red color text.

b. ***Overwrite existing database.***
Selecting this option will ask Custom dbInstaller to overwrite the existing the database of the same name if exists. This option is not so risky if the backup option is selected. If the backup option is unselected, its caption will change to red color text, giving the user a warning. It will, however, still allowing the user to proceed as such circumstances may exist.

3. **Execute SQL script file.**
Choosing this option will allow Custom dbInstaller to run a T-SQL script file, whose file name is as specified in the **otkdbi.exe.config** file. If it is being enabled in the config file but Custom dbInstaller could not find the given file, the check box will be automatically unchecked and the caption will turn into red color text, with an additional message appended: *"(Missing script file)"*. It will look like something as shown in **Figure 2-3** for the **Update config file's ConnectionString** check box. When this happens, forcing it to check will have no effect although it looks like not being disabled. Custom dbInstaller choose not to disable it because disabling it will lose its color. This may give confusion to the end user as though it is not caused by missing file but could be due to being disabled in the config file. Therefore, the best way to communicate this situation in a more precise manner is to keep the red alert visible while prohibiting the user from checking the check box.

   Also, as mentioned in *option 1*, checking this option without checking the backup option will cause the caption of the backup check box turn into red color. This is because the SQL script might not be properly written or the developers may unintentionally delivered a wrong version of script file, which is not apply to the database that Custom dbInstaller is currently working on. If this is the case, backing up the database will make future recovery possible.

4. **Update config file's ConnectionString.**
The behavior of this option is quite similar to the third option, i.e. the **Execute SQL script file** check box, except that the file type is different – the application's config file. This could be the Windows Application's app.config file or the Web Application's Web.config file. If Custom dbInstaller could not find the specific config file, its caption will turn into red color, stating *"(Missing config file)" (see Figure 2-3)*. Another difference from *option 3* is that, checking this option will not cause the caption of the unchecked backup check box changing into red because this process will not involve the change of the existing database contents and/or structure.

For more information about the configuration of Database Installation Options, see ***TABLE 3 – INSTALLATION TASK SETTINGS***.

2.2.4 Understanding the Database File Location Wizard Dialog

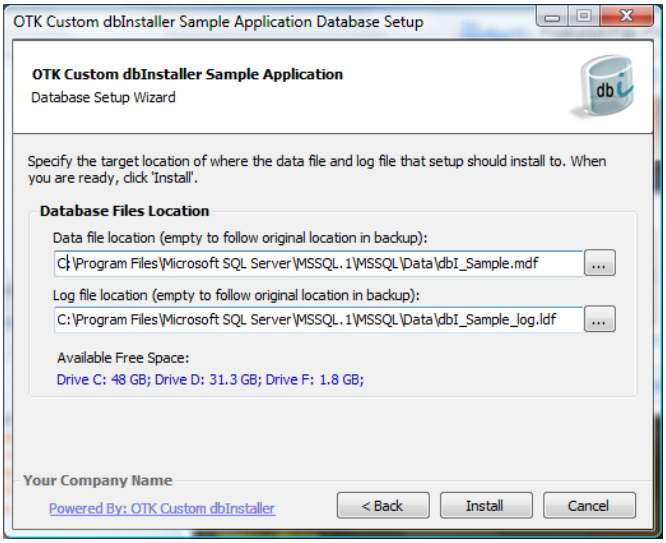

Figure 2-4 Database File Location wizard dialog

If you choose to restore the database and the database file location is allowed to be edited, clicking the **Next** button from the Database Installation Options dialog box will lead the user to this screen. This is particularly useful when the preferred target location of the database varies from the original location as of where the source database was backed up. If the original file location is preferred, just remove the default path and leave the text box empty. You can even choose to have the data file follows the original path in the backup file and configure another different path for the log file by just leaving the **Data file location** empty and customize your own path for the **Log file location**. To change to different file location, simply click on the browse button **("…")**.

Also notice that below the **Log file location** text box, there is information showing you the available free space for each hard drive or partition *(see Figure 2-4)*. It is always a good practice to choose the drive that has larger free space. Therefore, this information is useful for such decision making.

In a typical database installation mode, Database File Location dialog box is the last step's dialog, thus, you can see the **Install** button instead of **Next** button. If you choose not to allow the user to edit the file location, however, the last dialog box will be the previous screen and the **Install** button will appear in the previous screen. Custom dbInstaller is highly agile and flexible.

For more information about the configuration of Database Installation Options, see ***TABLE 3 – INSTALLATION TASK SETTINGS***.

2.2.5 Understanding the Uninstallation Options Wizard Dialog

In database uninstallation mode, the most that Custom dbInstaller 2.0 provides is only two dialogs, i.e. that SQL Server Connection and Database Uninstallation Options. The SQL Server Connection dialog box is exactly the same as what has been described in **2.2.2 – Understanding the SQL Server Connection Wizard Dialog**. Of course, if you can choose to hide the SQL Server Connection screen, it will also be hidden during the uninstallation mode. Again, you can also choose to hide this Database Uninstallation Options dialog from otkdbi.exe.config file. Keep in mind that you can always choose to

hide all the wizard dialog boxes and Custom dbInstaller is still working provided all the necessary configurations as defined in the otkdbi.exe.config are valid to the target environment.

Like the Database Installation Options dialog, the Uninstallation Options dialog box also has an option to backup the existing before uninstall *(see Figure 2-5)*. The alerting mechanism is also similar to Database Installation Options dialog whereby the caption of the backup check box will change to red color if you select the **Drop and delete the database** option without checking the backup check box. On the other hand, the user will not be alerted if the **Detach the database only** option is chosen because it is quite safe to just detaching the database as you can always attach it back for recovery.

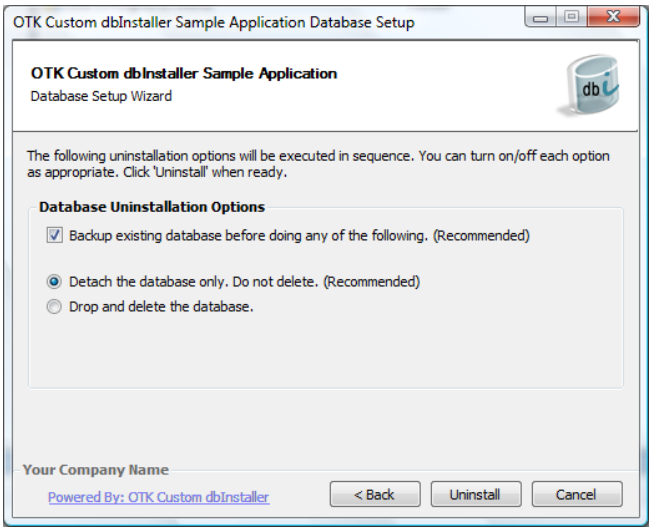

Figure 2-5 Database Uninstallation Options wizard dialog

2.3 Error Handling with Custom dbInstaller

There are always situation whereby errors will occur and a particular tool or software must provide a way for end users or technical staffs to manage the errors. The following describes how to handle the two common types of error when prompted by Custom dbInstaller:

1. **Error while connecting to SQL Server**
   This type of error usually occurred at the SQL Server Connection dialog box stage because the login credential is always being validated at this stage. However, such error may occur at the beginning of the last stage where the SQL Server connection will be validated again before executing the mission. When this happened, just click the Back button and try again.

   When Custom dbInstaller fails to connect to SQL Server, an error will be prompted as shown in **Figure 2-6**. After dismissing the message box, Custom dbInstaller allows you to correct the error, such as reentering the password, change to different authentication mode, or check if the SQL Server service is started. Once you have done, click the Next button to try again. If the problem persists, you can click the Cancel the button and try it some other time. To try fixing the error at the later stage, you can either run the entire application installer again (if Custom dbInstaller was being added as a custom action into the installer itself) or you can just run only the otkdbi.exe program with a -i switch. Note that even though it is packaged with the application installer, you can still able to run it independently provided you know

where the otkdbi.exe stores. However, the recommended way to end users is to run the entire application installer again or uninstall it and install again.

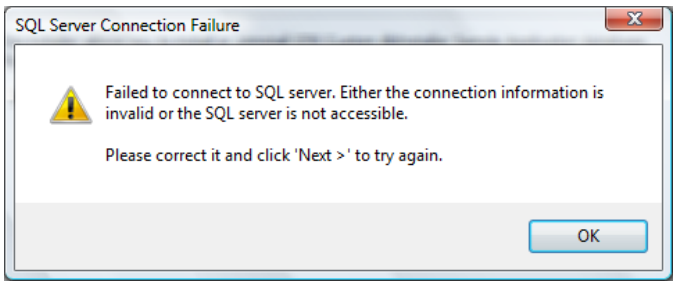

Figure 2-6 SQL Server connection error message

2. **Error during installation/uninstallation**

After passing the SQL Server Connection stage, you may encounter some other error like SQL script execution error while in the middle of the database installation process. When this happens, an error message will be displayed above the progress bar and the Install button will change to View Log button *(see Figure 2-7)*. Clicking the View Log button will open the setuplog.txt file in Notepad and change the button label to Retry. You can either click Retry to try again or click Back to go back to the previous screen to correct something. If you come back from the previous screen, the red color error message will no longer be displayed but the Retry button still preserved. Click the Retry button to run the setup again.

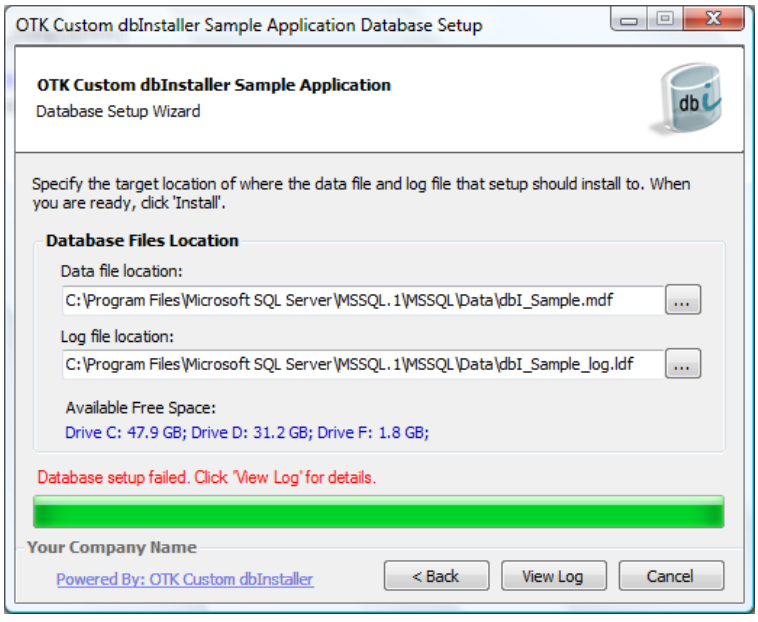

Figure 2-7 Database File Location wizard dialog showing the error message and the View Log button

2.4 Installing and Exploring the Sample Application

Before getting your hands dirty on OTK Custom dbInstaller 2.0, you are encouraged to install and try out the **Custom dbInstaller Sample Application**. This sample application has the following objectives:

- To give developers a quick look at how dbI 2.0 can do for them and their customers, and how it can perform.
- To allow developers to quickly get started with dbI 2.0 in getting their application installer delivered together with their database.

The following describes how the sample application helps to achieve these two objectives:

- After installing dbI 2.0, there will be a shortcut available at the **Start** menu pointing to **otkdbi_SampleSetup.msi**.
- The core purpose of **otkdbi_SampleSetup.msi** setup package to demonstrate to developers how the standard Windows Installer is going to call the dbI wizard dialog in helping this sample application to install its SQL Server database.
- After completing this installation, a new sub program group called **Sample Application** will be created under **OTK Custom dbInstaller** at the **Start** menu. This new sub program group contains the following shortcuts:

    - **OTK Custom dbInstaller Sample Application.** This will launch the sample application as shown in **Figure 2-8**, which allows you to test the connection from this sample application to its database, view the config file's connection string updated by dbI, view the sample SQL script, open the sample setup project folder and lots more.

    - **Open Sample Setup Project Folder.** This shortcut allows you to open the sample setup project folder, giving you an option to extract the project source code. This sample source code will be the most effective tool for you to quickly incorporate dbI 2.0 into your application setup project.

    - **Open otkdbi.exe.config.** This will let you quickly open and inspect how the **otkdbi.exe.config** was being configured for the sample application to run in the demonstrated behavior.

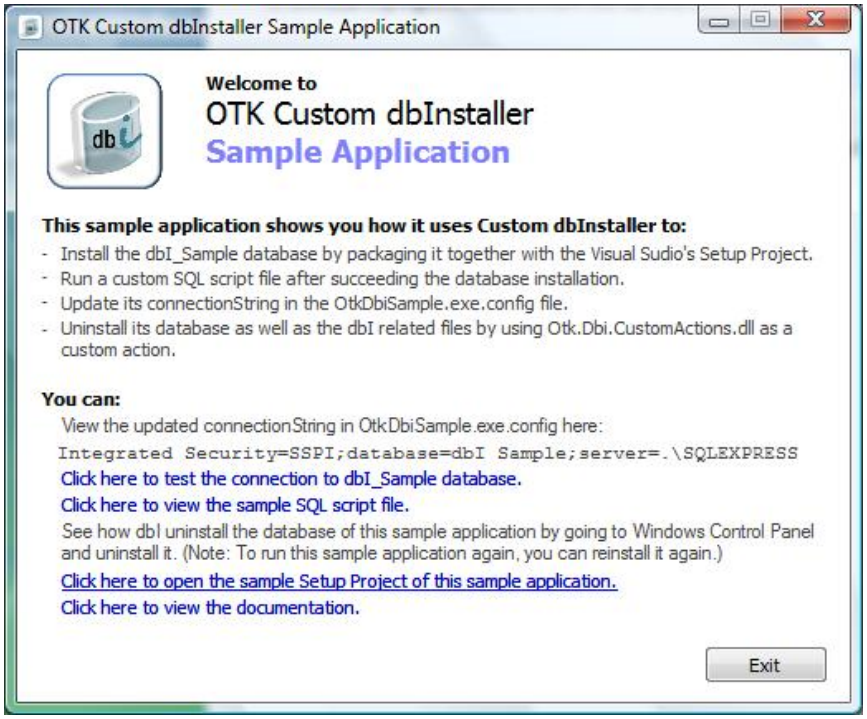

Figure 2-8 OTK Custom dbInstaller Sample Application

3 Configuring and Customizing Custom dbInstaller

3.1 Understanding the otkdbi.exe.config file

Custom dbInstaller comes with an XML configuration file called **otkdbi.exe.config**, which allows the developers to customize dbI to suit their needs. This configuration file is actually an app.config file that most.NET developers familiar with.

By default, dbI follows all the settings defined in the config file while executing its tasks unless one or more of them has been exposed to users for further changes during runtime *(see Figure 2-1)*. However, some settings like General Settings *(ApplicationName, LogoImageFileName etc)* and Wizard Dialog Customization settings *(AllowEditServerConnection, AllowEditInstallTasks etc)* will not be exposed to users for changes. Upon the success of the installation process, only the SQL Server connection and database file locations will be saved to the config file. This is for the purpose of future database uninstallation and it is particularly necessary when the `AllowEditServerConnection` is set to `False`. For more detailed references about the config file settings and its usage, see Appendix I – Configuration Reference.

3.2 Customizing the Wizard Dialog Boxes

3.2.1 Displaying your product or company logo on the Wizard Dialog Boxes

This version of Custom dbInstaller gives you the standard look-and-feel of a typical installer by allowing you to customize the logo image at the upper-right corner of the top banner on every wizard dialog box. To do this, just follow the easy steps below:

1. Prepare your logo image in 70 x 70 dimension in the format that supported by the **PictureBox** control, such as .BMP, .JPG, .GIF, or .PNG.
2. Place this image file together with **otkdbi.exe** on the same folder.
3. Open the **otkdbi.exe.config** file and assign the image file name to `LogoImageFileName`.
4. Save the config file and run **otkdbi.exe**.

3.2.2 Showing or hiding Wizard Dialog Boxes during runtime

This is one of the most powerful features of Custom dbInstaller brought to you by OTK Web Solutions. It provides *high agility and flexibility* in adapting to the following scenarios:

1. **Who will be the users? What is their knowledge level in terms of computing and SQL Server in particular?**
   a. If naïve, which screens can we hide in making our deployment process as simplest possible?
   b. If advance, which screens can we show to provide as much flexibility as possible?

2. **How would the target deployment environment like? How sure are we about their environment settings?**
   a. If we are sure, pre-configure it in config file and hide as many screens as possible to maximize simplicity.
   b. If we are not sure, pre-configure the recommended default and see which options we can expose to users to expect for the unexpected.
3. **What kind of database maintenance services that we are going to provide? Would it involve human intervention or unattended schedule task?**
   a. If involve human, which screens should we show?
   b. If not, let us pre-configure to fully tailored to their environment, hide all the screens and schedule it to run nightly.
4. **How is our software architecture like? Is it being designed to cater for such deployment options?**
   a. If our architecture is ready for that extent of flexibility, show the screens.
   b. If we are not ready for that, for the time being, we hide the screens.

Once you have identified the scenario and determined which wizard dialog boxes to hide/show, we can start configuring Custom dbInstaller to achieve this goal. In this case, we take scenario 1a. as described above as an example, where knowledge level of the user is low. Below is what we plan to do with Custom dbInstaller with some extended description for the scenario:

- **Show the SQL Server Connection dialog and allow SQL Server Authentication only**
  Our Web-based application can only support SQL Server authentication because we have not yet get it to work with Integrated Security. In addition, some of our users already have their SQL Server installed and we do not know what their SA password is.

- **Pre-configure the safest way to deploy our application database and hide the Installation Options dialog box**
  Since most of our users are naïve users and we do not need to know their SQL Server's configuration for the installation options, thus we can hide this dialog box.

- **Write our own SQL script to create tables, views, and stored procedures and hide the Database File Location dialog box**
  The users may not know where to store their database files. Also, we can't provide a properly done backup set as we do not know their default database file path. Furthermore, we can't assume every user has the same default SQL database path.

So, let's do the job:

1. Open the **otkdbi.exe.config** file in Visual Studio for better visualization. You can use Notepad if you want to.
2. Set the `AllowEditServerConnection` value to `True`.
3. Set the `Authentication` value to 1 (SQL Server Authentication).
4. Set the `AllowMixedModeAuthentication` value to `False`.

5. Set the value of the following to `False`:
   - `AllowEditInstallTasks`
   - `AllowEditDatabaseFileLocations`
   - `AllowEditUninstallTasks`

   This is to hide the Installation Options, Database File Location, and the Uninstallation Options dialog boxes.

   > ***NOTE:*** *If `EnableDatabaseRestoreTask` is set to `False`, the Database File Location dialog box will be automatically hidden even though `AllowEditDatabaseFileLocations` is set to True.*

6. Set the value of the following to `True`:
   - `EnablePreinstallBackupTask`
   - `EnableSQLScriptFileExecutionTask`
   - `EnableConfigFileUpdateTask`
7. Since we prefer to install the database solely from the SQL script file, then turn off the database restore by setting `EnableDatabaseRestoreTask` to `False`.
8. Assign a file name to your SQL script that you are going to create by setting the value of `SqlScriptFileName` to "`MyAppDbDeployment.sql`".
9. Set the `ConfigFileLocation` to "`/MyWebApp`" and the `ConnectionStringName` to "`MyAppDbConnection`".
10. Leave the `MdfFileLocation` and `LdfFileLocation` as empty string ("") as you have already turned off the database restore task.
11. Set of value of `EnablePreuninstallBackupTask` to `True`.
12. For safest uninstallation option, set the `DatabaseUninstallTaskOption` to 1, that is to detach the database only but not dropping.
13. Save the **otkdbi.exe.config** file and close Visual Studiio.
14. Create a SQL script file called **MyAppDbDeployment.sql** that will create your application's database tables, views, and stored procedures. This SQL script may have some `INSERT` statements to insert some default system data into the relevant tables.
15. Save your SQL script file and copy it to the same location as where you store your **otkdbi.exe** together with the config file that you previously modified.
16. Test run your new Custom dbInstaller configurations by issuing the following command at the command prompt:

    For installation:
    ```
    otkdbi.exe -i
    ```

    For uninstallation:
    ```
    otkdbi.exe -u
    ```

Now you can package these three files, i.e. otkdbi.exe, otkdbi.exe.config, and MyAppDbDeployment.sql, together with your Web Setup Project to deliver simple automated deployment experience to your end users.

4 Integrating with .NET Setup or Web Setup Project

OTK Custom dbInstaller allows you to integrate with your .NET Setup or Web Setup Project to enable you to deliver fully automated database-driven application deployment experience to your end users. You no longer need to write your own code to incorporate this feature into your application installer, of which the invested effort may only be catered for the scenario at the specific moment. With the flexibility of Custom dbInstaller, you just focus on the content that you want to deliver and spend only little effort to incorporate it into your existing setup package.

The following instructions guide you through how to add Custom dbInstaller as custom actions into your Setup Project:

1. To be well-organized, create a sub folder called **dbInstaller** under the main folder of your Setup Project.
2. Copy the following Custom dbInstaller files to the **dbInstaller** folder:
    - otkdbi.exe
    - otkdbi.exe.config
    - Otk.Dbi.CustomActions.dll
3. Create or copy one or more of the following files, whichever applicable, to the **dbInstaller** folder as well:
    - Your application database backup file, e.g. MyAppDb.bak.
    - Your custom SQL script file, e.g. MyAppDbDeployment.sql.
    - Your product logo image file, e.g. MyAppLogo.jpg
4. Open your Setup Project in Visual Studio and go to your **File System Editor**.
5. At the **File System Editor**, right-click on your **Application Folder** and create a sub folder called **dbInstaller**.
6. Add all the files as you previously stored in the physical **dbInstaller** folder to your **File System Editor's dbInstaller** folder.
7. Open your **Custom Action Editor**.
8. Right-click on the **Install** folder and select **Add Custom Action**.
9. At the **Select Item in Project** dialog box, double-click on **Application Folder** followed by **dbInstaller** folder.
10. Select **otkdbi.exe** and click **OK**. You will see that **otkdbi.exe** has been added to the **Install** folder of your **Custom Action Editor**.
11. Right-click on **otkdbi.exe** and select **Properties Window**.
12. Set the **InstallerClass** property to **False** and enter the following to the **Arguments** property:

    ```
    -i -cfg:"[TARGETDIR]MyApp.exe.config"
    ```

    This is for Custom dbInstaller to update your application config file.
13. Now go back to your **Custom Action Editor**.
14. Right-click on **Uninstall** folder and select **Add Custom Action**.
15. At the **Select Item in Project** dialog box, double-click on **Application Folder** followed by **dbInstaller** folder.

16. Select **Otk.Dbi.CustomActions.dll** and click **OK**. You will see that **Otk.Dbi.CustomActions.dll** has been added to the **Uninstall** folder of your **Custom Action Editor**.
17. If you have done the rest of your program files setup, build your setup project.
18. To test the installation, right-click on your setup project and select **Install**.
19. To test the uninstallation, right-click on your setup project and select **Uninstall**.
20. You are done.

> ***TIP:*** *For a clearer picture, you can run through the **Custom dbInstaller Sample Application** and explore the sample setup project. If you have not installed the sample application, you can go to the Windows **Start** menu, navigate to the **OTK Custom dbInstaller** program group and select **See Custom dbInstaller in action!**. This will launch the sample application installer, which will demonstrate to you the results of integrating Custom dbInstaller with your application installer. After that, you will have a new menu item at the **Start** menu to explore the sample setup project.*

5 How to Contact Us

You are encouraged to send us your feedback, technical questions, and bug report by submitting them as comments through the following blog:

http://otkfeedback.blogspot.com

For other inquiries, you may email us at:

otk.web@gmail.com

We will respond back to you as soon as possible, regardless whether you are a licensed user or just a demo user. Your feedback is always needed and appreciated.

6 How to Purchase and Enter License Key

To buy OTK Custom dbInstaller, you can visit our **OTK Retailer Center** Web site. There, you can choose your preferred retailer and follow the given instructions on the page. There are several ways for you to access our OTK Retailer Center Web site:

1. On any wizard dialog box, click the **Powered by: OTK Custom dbInstaller** link. When the About box popped up, click the **Buy Now** button.
2. On any wizard dialog box, click the **Powered by: OTK Custom dbInstaller** link. When the About box popped up, click the http://otkretail.blogspot.com link.
3. Open your Internet browser and enter the following at the address bar: http://otkretail.blogspot.com

After purchasing the product, our retailer will send us your purchase information. From your purchase information, we will generate a product key for you. This product key will be emailed to your mail box within 24 hours.

To enter the license key, run OTK Custom dbInstaller and use one of the following methods:

1. At the **Discover the magic of OTK Custom dbInstaller 2.0**, click the **Enter License Key** button and follow the instructions on the screen, or
2. On any wizard dialog box, click the **Powered by: OTK Custom dbInstaller** link. When the About box popped up, click the link below the **Licensed to** field and follow the instructions on the screen.

Thank you for using OTK Custom dbInstaller!

Appendix I – Configuration Reference

TABLE 1 - GENERAL SETTINGS

Key Name	Editable at User Interface	Save upon Success	Usage
`ApplicationName`	✘	-	Required. This is where you specify your product name that to be displayed on the wizard dialog boxes.
`ApplicationDatabaseName`	✘	-	Required. This is your application database name that being used to install/uninstall the database. It is also being used to construct your application's *connectionString*. Make sure that the supplied source database backup file contains the database with this name.
`LogoImageFileName`	✘	-	Optional. You can provide your product image logo in the format that accepted by .NET PictureBox control. The typical supported formats are .GIF, .JPG, .BMP, and.PNG. This image file will be displayed on the upper-right corner of each wizard dialog box. If you don't specify it or dbI could not find the given image file, it will display the Custom dbInstaller product logo as default.
`LicenseOwner`	✓	✓	Required. This is the license owner's company name as per the registration name during purchase. If you do not have a registered company name, you can specify your personal name here but make sure your purchase registration is based on this name and it is AFTER obtaining the license key. It can be entered through the About box by clicking on the Powered By: OTK Custom dbInstaller link on any wizard dialog. The default is always Your Company Name until you have entered the valid company name and license key. Any errors will cause the `LicenseOwner` to be changed back to Your Company Name.
`LicenseKey`	✓	✓	Required. This is where your license key stores. It can be entered through the Enter License Key dialog box. The Enter License Key dialog can be called from About box. To open the About box, click on the Powered By: OTK Custom dbInstaller link from any wizard dialog box. If you are in demo mode, the key is always appeared as (Demo). To obtain the license key, you must register and purchase Custom dbInstaller by clicking on the Buy Now button via the About box.

TABLE 2 – SQL SERVER CONNECTION SETTINGS

Key Name	Editable at User Interface	Save upon Success	Usage
`Authentication`	✓	✓	Required. 0 – Windows Authentication 1 – SQL Server Authentication This will store the selected authentication type, whether SQL Server Authentication or Windows Authentication. It also acts as a default selected authentication when the `AllowMixedModeAuthentication` option is set to `True`. Note that if Windows Authentication is selected, the login credential will be based on the current logon user and the User ID and Password text boxes will be disabled. These are needed by Custom dbInstaller to determine the construction of the *connectionString*.
`ServerName`	✓	✓	Required. This is the SQL Server instance name where your application database is going to be hosted on. It is also being used to construct your application's *connectionString*.
`SqlServerUserID`	✓	✓	Required if SQL Server Authentication is selected and **`AllowEditServerConnection`** is set to **`False`**. It is used by Custom dbInstaller to form the connectionString.
`SqlServerUserPassword`	✓	✓	Optional. If the target SQL Server allows the sa password to be blank or Windows Authentication is selected, then this can be optional. It is used by Custom dbInstaller to form the *connectionString*.
`AllowMixedModeAuthentication`	✗	-	Required. If you want to allow user to select between SQL Server and Windows Authentications, set it to `True`. If you set it to `False`, Custom dbInstaller will take the value from the `Authentication` key as the default selected authentication mode and the Authentication dropdown list will be disabled.

TABLE 3 – INSTALLATION TASK SETTINGS

Key Name	Editable at User Interface	Save upon Success	Usage
`EnablePreinstallBackupTask`	✓	✗	Required. `True` – This will backup the existing database prior to any other process and make the option check box visible to user. The database will be backed up to the same folder as where the otkdbi.exe stores. The backup file name will be in the following format: `<dbName>_db_<yyyyMMddHHmmss>.bak` If the user uncheck the check box and either the overwriting existing database option or the execute SQL script option is selected, the text color of this check box will turn to red, alerting the user that something unsafe may be happened. If no database with the same name (as specified in the `ApplicationDatabaseName`) exists, Custom dbInstaller will skip the backup process and no error will be prompted. `False` - The option check box will be hidden and no backup will be performed. The above behaviors apply to both installation and uninstallation.
`EnableDatabaseRestoreTask`	✓	✗	Required. `True` – Restore the database from the backup file as specified in the `DatabaseBackupFileName` key. This will also make the option check box visible to the user plus the other two associated radio buttons (see `DatabaseRestoreTaskOption` key). `False` - The option check box, together with the two associated radio buttons, will be hidden and no restoration will be performed.
`EnableSQLScriptFileExecutionTask`	✓	✗	Required. `True` – Execute SQL script file as specified in the `SqlScriptFileName` and make the option check box visible to user. `False` – Hide the option check box and do not execute SQL script file. If this option is set to `True` and the script file cannot be found, the text color of this check box will turn to red with an extended message stated (Missing script file). When this happens, the check box will be automatically unchecked regardless of the default value stated in the config file. Although it appears to be enabled, an attempt to check the check box will make no

			effect because Custom dbInstaller would like to let the user stay alerted at the same time prohibit he/she to enable it without the presence of the script file.
`EnableConfigFileUpdateTask`	✓	✗	Required. `True` – If you want Custom dbInstaller to update the *connectionString* of your application config file as specified in the `ConfigFileLocation` and `ConnectionStringName` key. `False` - The option check box will be hidden and no application configuration file will be updated. Setting this option to `True` will make the option check box visible to the user. If the given config file location cannot be found, the text color of this check box will turn to red with an extended message stated (Missing config file). When this happens, the check box will be automatically unchecked regardless of the default value stated in the otkdbi.exe.config file. Although it appears to be enabled, an attempt to check the check box will make no effect because Custom dbInstaller would like to let the user stay alert at the same time prohibit he/she to enable it without the presence of the application config file.
`DatabaseRestoreTaskOption`	✓	✗	Required. 0 = Skip restore if the database already exist; 1 = Overwrite the existing database if exist. These two options will be represented as radio buttons associated with the Install database with the following method check box. If this check box is unchecked, these radio buttons will be automatically disabled. Likewise, if this check box is hidden, these two radio buttons will also be hidden. If `1` is being set, namely option to overwrite the database if exist, unchecking the backup check box will turn the backup check box text color to red. This is to alert the user that such a combination of options could be unsafe.
`DatabaseBackupFileName`	✗	-	Required. This is required if you intend to turn on the `EnableDatabaseRestoreTask`. This should be the SQL database backup file name only but not the full path. You should place this file together with otkdbi.exe on the same folder.
`SqlScriptFileName`	✗	-	Required. This is required if you intend to turn on the `EnableSQLScriptFileExecutionTask`. This should be the SQL script file name only but not the full path. You should place this file together with otkdbi.exe on the same folder.
`ConfigFileLocation`	✗	-	Required. This is required if you intend to turn on the `EnableConfigFileUpdateTask`. If you

			intend to integrate Custom dbInstaller with your Setup Project, you will need to provide the config file location in the Arguments property *(see 4. Working with .NET Setup or Web Setup Project)*. The value of this key has the following syntax: E.g. for Web application "/<MyVirtualDirectoryname>" NOTE: A "/" is needed before the virtual directory name. E.g. for Windows Application "C:\Program Files\MyApp\MyApp.exe.config" .
`ConnectionStringName`	✘	-	Required. This is required if you intend to turn on the `EnableConfigFileUpdateTask`. The value should be the *connectionString's* name as you specify in your application config file. If the name is not found, a new *connectionString* with the given name will be added. .
`MdfFileLocation`	✓	✓	Optional. This must be a valid full data file path that ends with .MDF. You can specify this in otkdbi.exe.config file provided you are very sure that the specified path is also valid at your target computer. If you set `AllowEditDatabaseFileLocations` to `True` *(see TABLE 1.3)*, however, Custom dbInstaller will validate it and the user will be prompted if invalid. If you leave the value empty, the original data file location in the backup file will be taken, no error will be prompted by the Database File Location dialog box.
`LdfFileLocation`	✓	✓	Optional. This must be a valid full log file path that ends with .LDF. You can specify this in otkdbi.exe.config file provided you are very sure that the specified path is also valid at your target computer. If you set `AllowEditDatabaseFileLocations` to `True` *(see TABLE 1.3)*, however, Custom dbInstaller will validate it and the user will be prompted if invalid. If you leave the value empty, the original log file location in the backup file will be taken, no error will be prompted by the Database File Location dialog box.

TABLE 4 - WIZARD DIALOG CUSTOMIZATION SETTINGS

Key Name	Editable at User Interface	Save upon Success	Usage
`AllowEditServerConnection`	✘	-	Required. `True` – Show SQL Server Connection dialog box. `False` – Hide SQL Server Connection dialog box. ⓘ ***TIP:*** *If you choose to hide the SQL Server Connection dialog box, for greater reliability, default the* `Authentication` *to Windows Authentication, but make sure the application can accept Integrated Security connectionString (see TABLE 1.1 – SQL Server Connection Settings).*
`AllowEditInstallTasks`	✘	-	Required. `True` – Show Installation Options dialog box. `False` – Hide Installation Options dialog box. ⓘ ***TIP:*** *You can choose to allow only some of the option check boxes in this dialog box to be shown and editable (see TABLE 1.2 – Installation Task Settings).*
`AllowEditDatabaseFileLocations`	✘	-	Required. `True` – Show Database File Location dialog box. `False` – Hide Database File Location dialog box. ***CAUTION:*** *If you choose to hide the Database File Location dialog box, make sure that the target computer also has the directory path as specified in the MdfFileLocation and LdfFileLocation (see TABLE 1.2 – Installation Task Settings).*
`AllowEditUninstallTasks`	✘	-	Required. `True` – Show Uninstallation Options dialog box. `False` – Hide Uninstallation Options dialog box.

TABLE 5 – UNINSTALLATION TASK SETTINGS

Key Name	Editable at User Interface	Save upon Success	Usage
`EnablePreuninstallBackupTask`	✓	✗	Required. `True` – This will backup the existing database prior to any other process and make the option check box visible to user. The database will be backed up to the same folder as where the otkdbi.exe stores. The backup file name will be in the following format: `<dbName>_db_<yyyyMMddHHmmss>.bak` If the user uncheck the check box and dropping the existing database option is selected, the text color of this check box will turn to red, alerting the user that something unsafe might be happened. If no database with the same name (as specified in the `ApplicationDatabaseName`) exists, for uninstallation mode, Custom dbInstaller will prompt an error. This is because it assumes missing of application database is an abnormal case for an uninstallation task. Of course, dropping or detaching database task will be skipped since the database does not exist. `False` - The option check box will be hidden and no backup will be performed. The above behaviors apply to both installation and uninstallation.
`DatabaseUninstallTaskOption`	✓	✗	Required. 0 = Drop existing database; 1 = Detach existing database, do not delete. If `0` is being set, namely option to drop and delete the existing databas, unchecking the backup check box will turn the backup check box text color to red. This is to alert the user that such a combination of options could be unsafe.

Appendix II – Command-line Reference

OTK Custom dbInstaller 2.0 supports the following command-line arguments:

TABLE 6 – Command-line Arguments and Usage

Argument	Usage & Example
`-i`	Usage: To run OTK Custom dbInstaller 2.0 in installation mode. Please note that by default, Custom dbInstaller will run in installation mode even without the -i switch. Example: `otkdbi.exe -i`

`-cfg:"<path>"`	Usage:
	To provide the application configuration file location to Custom dbInstaller during installation mode.
	
	Example for Windows Application:
	`otkdbi.exe -i –cfg:"C:\Program Files\MyApp\MyApp.exe.config"`
	
	Example for Web Application:
	`otkdbi.exe -i –cfg:"/MyWebAppVirtualDirectory"`

`-u`	Usage: To run OTK Custom dbInstaller 2.0 in uninstallation mode. Example: `otkdbi.exe -u`