Default Manager

NextAge Consulting
Pete Halsted
110 East Center St. #1035
Madison, SD 57042
pete@thenextage.com
www.thenextage.com
www.thenextage.com/wordpress

Table of Contents

NextAge Consulting - Pete Halsted

Pete Halsted has been developing custom business management applications for small to medium-sized companies, since 1987. His focus is on client/server, distributed and cloud based development utilizing WinDev, WebDev, and PostgreSQL. Pete is a Clarion Certified Developer with 25 years in the industry, has spoken at several Developers conferences, and provided Developer training and mentoring on a one on one basis. He has served companies both large and small as Project Manager, Lead Architect, Lead Developer and Chief Technology Officer. Pete tours the country full-time by motor home with his wife and dog, enjoying the freedom provided by cloud based technologies. Pete is available for Project Management, Custom Design, Development, Training, and Speaking assignments. For more information please visit www.thenextage.com or follow his blog at www.thenextage.com/wordpress

General Information

The Default Manager Class was created to provide a uniform method of storing and retrieving default settings. There are four levels of default storage; Registry, User Global, Window Global and User Global. With version 1.01 this includes the ability to store and retrieve binary data.

Saving a Default Value (SaveDefault Method)

There is a code brick to assist with saving a value. There are three parameters:

1. DefaultType
    a. The type of storage for the default. One of the four equate can be used.
2. DefaultName
    a. This is the name of the default value, think of it as similar to a variable name.
3. Value
    a. The value to store as the default

Examples

- DM.SaveDefault(DefaultManager.Registry,"test","x")
    - Stores the value “x” for the default “test” in the registry
- DM.SaveDefault(DefaultManager.UserGlobal,"test","x")
    - Stores the value “x” for the default “test” in the DefaultManager file for the current user
- DM.SaveDefault(DefaultManager.WindowGlobal,"test","x")
    - Stores the value “x” for the default “test” in the DefaultManager file for the current window. This will be the default for all users.
- DM.SaveDefault(DefaultManager.UserWindow,"test","x")
    - Stores the value “x” for the default “test” in the DefaultManager file for the current window and user. This will be the default for this user only.

Retrieving a Default Value (GetDefault Method)

There is a code brick to assist with getting a value. There are three parameters:

1. DefaultType
    a. The type of storage for the default. One of the four equate can be used.
2. DefaultName
    a. This is the name of the default value, think of it as similar to a variable name.
3. DefaultValue
    a. This is an optional parameter. If specified, and a Default Value is not found, a default will be saved with the value specified and the value will be returned
    b. When storing and retrieving Binary Data, the Default Value must be supplied as the default value, this is needed so the binary data can be properly deserialized.

Examples

- ResultValue = DM.GetDefault(DefaultManager.Registry,"test","x")
    - Retrieves the value for the default “test” from the registry, if not found “x” will be stored and returned.
- ResultValue = DM.GetDefault(DefaultManager.UserGlobal,"test","x")
    - Retrieves the value for the default “test” from the DefaultManger file that is for the current user, if not found “x” will be stored and returned.
- ResultValue = DM.GetDefault(DefaultManager.WindowGlobal,"test","x")
    - Retrieves the value for the default “test” from the DefaultManger file that is for the current window, if not found “x” will be stored and returned.
- ResultValue = DM.GetDefault(DefaultManager.UserWindow,"test","x")
    - Retrieves the value for the default “test” from the DefaultManger file that is for the current window and user, if not found “x” will be stored and returned.
- StructureVariable = DM.GetDefault(DefaultManager.UserWindow, ”StructureVariable”, StructureVariable)
    - Retrieves the binary data for the default “StructuredVariable” and deserializes it into StructuredVariable
    - Note: The Structured Variable must be passed as the default value parameter as well, so the deserialization can be performed.

Database Table

Many of the defaults are stored in a database table. Therefore you must have a table configured for this purpose. The class has been written in a generic fashion that allows the file name and field names to be set via properties if desired. The class also uses Hxxxx commands for all file access, meaning that the class should work with any database that WX supports. The suggested definition for this file follows. If you do not use this definition you will have to use the associated properties to point to the correct values (*See the Special Properties*).

DefaultManager

- `DefaultManagerId`
  - `Automatic ID`
- `DefaultType`
  - `Text (20)`
- `TypeDetail`
  - `Text (50)`
- `Variable`
  - `Text (256)`
- `Value`
  - `Text (256)`
- `BinaryValue`
  - `Binary`

Project Code

The class should be initialized in your project initialization code. There is a code brick to assist with this (DMSetup)

1. `DM is DefaultManager`
    - Instantiates an instance of the class global for the entire project.
2. DM.TheSystemLogClass = SysLog
    - Pointer to the System Log class for logging errors. (*See Notes under Properties*)
3. `DM.SilentErrors = False`
    - If set to True Error Messages in the class will not be displayed. The default value is False. Note: If using the System Log class the error will still be logged even if an error is not displayed.
4. `DM.RegistryPath = "HKEY_LOCAL_MACHINE\SOFTWARE\theTaxCure\Defaults"`
    - The root node you want to use in the registry. It will be created if it doesn't exists.
5. `DM.UserIdVariable = "Glo.UserId"`
    - The project variable that stores the User Id

Properties

Note: Private Properties are not documented.

RegistryPath
The registry key that will be used as the “Root” key for all defaults stored in the registry. The key will automatically be created.

SilentErrors
If set to True Error Messages in the class will not be displayed. The default value is False. Note: If using the System Log class the error will still be logged even if an error is not displayed.

TheSystemLogClass
Pointer to the System Log class for logging errors.

- The System Log class is another Open Source class available from NextAge. The Default Manager class will perform without it. However if you don’t not want to include the System Log class in your project you will have to comment out the related lines of the class. Unfortunately WX does not provide the ability to optionally compile code based on the existence of another class. As long as the class is include in your project you will not have compile errors and the class calls will be skipped, however if you remove the System Log class from your project, you will have compile errors in the Default Manager class.

UserIdVarialbe
The project variable that stores the User Id

Methods

Note: Private Methods are not documented.

GetDefault
Retrieves a default value, based on one of the four methods available. If a default value is specified and a default entry is not found one will be created and the value specified will be returned.

SaveDefault
Saves a default value, based on one of the four methods available. There is a Code Brick to assist.

Equates

Equates are provided to assist with the calls to the Save and Get methods.

Registry
: Setting are stored in the Windows Registry. This option is good for storing defaults that need to be retrieved even without a database connection. This settings will by their nature machine dependent and will only be available on the computer they are stored on.

UserGlobal
: These settings are stored in a database table and are keyed to the logged in users id. This is a great method of storing “User Preference” type settings.

WindowGlobal
: These settings are stored in a database table, but are not keyed to a user id, meaning all users of the system will share the default. Handy for storing “Company Preference” type settings.

UserWindow
: These settings are stored in a database table and are keyed on both the userid and the current window. This is useful for storing default parameters for Report prompt windows, etc.

Code Bricks

DMSetup - Default Manager Setup
Used to populate the initialization code of the template in the project initialization process.

DMSave - Default Manager Save
Used to make a call to the SaveDefault Method

DMGet - Default Manager Get
Used to make a call to the GetDefault Method

Special Properties

The below properties can be used to change the default file name and field names used.

DefaultManagerFile
The name of the Default Manager file. Default = “DefaultManager”

DefaultTypeField
The name of the DefaultType field. Default = “DefaultType”

TypeDetailField
The name of the TypeDetail field. Default = “TypeDetail”

VariableField
The name of the Variable field. Default = “Variable”

ValueField
The name of the Value field. Default = “Value”

BinaryValueField
The name of the BinaryValue field. Default = “BinaryValue”

Change Log

1.0 - December 25, 2012
Initial Release

1.01 - Not Released

- Rename Local Instance of SystemLogClass to TheSystemLogClass to avoid intellisense issues.
- Ability to store Binary Data - Thanks to suggestions from Pragma Tix.
- Corrected issue with the Assignment of Window Name not being correct if called from within a control's code process.
- Fix for initial creation of Registry Key