**NCPN Version Control Modules**

**Project Name:** NCPN Version Control Modules

**Modifications:** Initial version.

**Last Update:** 7/1/2015

**Prepared By**

|  |  |
| --- | --- |
| **Document Owner(s)** | **Project/Organization Role** |
| Bonnie Campbell | Asst. Data Manager (design/develop/test) |
|  |  |

**Document Version**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Change Description** |
| 1.0 | 6/30/2015 | Bonnie Campbell | Document created |
| 1.1 | 7/1/2015 | Bonnie Campbell | Added comments for naming conventions and module updates. |
|  |  |  |  |
|  |  |  |  |

**TABLE OF CONTENTS**

[1 PURPOSE 3](#_Toc423504378)

[1.1 Background 3](#_Toc423504379)

[2 DESIGN 3](#_Toc423504380)

[2.1 Version Control Modules 3](#_Toc423504381)

[2.1.1 VCS\_DataMacro 4](#_Toc423504382)

[2.1.2 VCS\_Dir 4](#_Toc423504383)

[2.1.3 VCS\_File 5](#_Toc423504384)

[2.1.4 VCS\_IE\_Functions 6](#_Toc423504385)

[2.1.5 VCS\_ImportExport 7](#_Toc423504386)

[2.1.6 VCS\_Loader 8](#_Toc423504387)

[2.1.7 VCS\_Reference 9](#_Toc423504388)

[2.1.8 VCS\_String 9](#_Toc423504389)

[2.1.9 VCS\_Table 10](#_Toc423504390)

[3 User Actions Required 12](#_Toc423504391)

# PURPOSE

Prepare an initial set of NCPN development modules which can be utilized for a variety of database development projects.

## Background

In addition to the NCPN database framework’s set of common database modules and forms and the NCPN development modules, this set of modules handles version control for access databases. Like the NCPN database framework and development modules, this common set focuses on code modularization and reuse.

# DESIGN

Like the development modules, this set of version control modules provides a method of clean module integration into framework projects. By using standard conventions, the modules can clearly be identified as version control modules and readily be added or removed from a project. These modules are critical to the overall workflow for Access Git Versioning[[1]](#footnote-1).

## Version Control Modules

These modules are utilized for version control purposes only. While there is some overlap between these modules and development modules, primary version control functionality is found within the version control’s version control system (VCS) modules.

These modules originate from Tim Abell’s GitHub repository[[2]](#footnote-2) for Access version control.

|  |
| --- |
| **Module** |
| ***Version Control*** |
| *VCS\_DataMacro* |
| *VCS\_Dir* |
| *VCS\_File* |
| *VCS\_IE\_Functions* |
| *VCS\_ImportExport* |
| *VCS\_Loader* |
| *VCS\_Reference* |
| *VCS\_String* |
| *VCS\_Table* |

All module names begin with *VCS\_* to identify them as version control modules.

|  |  |
| --- | --- |
| Note | Not all version control modules may be in use during a project, however it is recommended that all be included in development releases to reduce replication of previously created subroutines and functions. |

### VCS\_DataMacro

VCS\_DataMacro handles macro export and imports.

|  |  |
| --- | --- |
| ***Subroutines*** | |
| *ExportDataMacros* | *Export macros.as an XML file.*  ***Inputs:***  *TableName – name of table (string)*  *directory – file path (string)* |
| *ImportDataMacros* | *Import data macro into Access from XML file*  ***Inputs:***  *TableName – name of table (string)*  *directory – file path (string)* |
| *FormatDataMacro* | *Splits exported data macro XML file into multiple lines and allows Git to find changes within lines using diff.*  ***Inputs:***  *filePath – full file path of XML file (string)* |

### VCS\_Dir

VCS\_Dir handles directory related subroutines and functions.

|  |  |
| --- | --- |
| ***Subroutines*** | |
| *MkDirIfNotExist* | *Create directory if it does not exist.*  ***Inputs:***  *Path – directory path (string)* |
| *DelIfExist* | *Delete file if it exists.*  ***Inputs:***  *Path – directory path (string)* |
| *ClearTextFilesFromDir* | *Erase all files with the input extension.*  ***Inputs:***  *Path – directory path (string)*  *Ext – file extension (string)* |

|  |  |
| --- | --- |
| ***Functions*** | |
| *ProjectPath* | *Return the path for the current project.*  ***Inputs:*** *-* |
| *DirExists* | *Determines if directory exists.*  ***Inputs:***  *strPath – directory path (string)*  ***Returns:*** *True if directory exists, false if not (Boolean)* |

### VCS\_File

VCS\_File handles binary file reading and writing.

|  |  |
| --- | --- |
| ***Types*** | |
| *BinFile* | *Binary file structure for tracking buffered reading and writing.*  ***Inputs:***  *file\_num – number of file (integer)*  *file\_len – length of file (long)*  *file\_pos – position in file (long)*  *buffer – content of buffer (string)*  *buffer\_len – length of buffer (integer)*  *buffer\_pos – position in buffer (integer)*  *at\_eof – whether position is at end of file (Boolean)*  *mode – read (r) or write (w) (string)* |

|  |  |
| --- | --- |
| ***Subroutines*** | |
| *BinWrite* | *Open binary file for reading or writing.*  ***Inputs:***  *f – binary file (BinFile)*  *B – byte to write (integer)* |
| *BinClose* | *Close binary file.*  ***Inputs:***  *f – binary file (BinFile)* |
| *ConvertUcs2Utf8* | *Convert a UCS2 little endian encoded file to UTF-8.*  ***Inputs:***  *source – source file (string)*  *dest – destination file (string)* |
| *ConvertUtf8Ucs2* | *Convert a UTF-8 encoded file to UCS2 little endian.*  ***Inputs:***  *source – source file (string)*  *dest – destination file (string)* |

|  |  |
| --- | --- |
| ***Functions*** | |
| *getTempPath* | *API call to GetTempPathA*  ***Inputs:***  *nBufferLength – length of buffer (long)*  *lpBuffer - buffer content (string)*  ***Returns:*** *temporary file path (long)* |
| *getTempFileName* | *API call to GetTempFileNameA*  ***Inputs:***  *lpszPath – file path (string)*  *lpPrefixString - file prefix (string)*  *wUnique – unique number (long)*  *lpTempFileName – temp file name (string)*  ***Returns:*** *temporary file name (long)* |

|  |  |
| --- | --- |
| *BinOpen* | *Open binary file for reading or writing.*  ***Inputs:***  *File\_path – full path to file (string)*  *mode - read (r) or write (w) (string)*  ***Returns:*** *Binary file (BinFile)* |
| *BinRead* | *Read binary file byte by byte.*  ***Inputs:***  *f – binary file (BinFile)*  ***Returns:*** *bytes from file (integer)* |
| *UsingUcs2* | *Determine if database imports/exports code as UCS-2-LE.*  ***Inputs:*** *-*  ***Returns:*** *True if database imports/exports as UCS-2-LE, false if not (Boolean)* |
| *TempFile* | *Generate a random/unique temporary file name.*  ***Inputs:***  *Optional*  *sPrefix – file prefix, default is “VBA” (string)*  ***Returns:***  *Temporary file name (string)* |

### VCS\_IE\_Functions

VCS\_IE\_Functions handles binary file reading and writing.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Constants*** | | | | |
| *AgressiveSanitize* | *Private* | *Boolean* | *True* |  |
| *StripPublishOption* | *Private* | *Boolean* | *True* |  |
| *ArchiveMyself* | *Private* | *Boolean* | *True* |  |
| ***Constants for Scripting.FileSystemObject API*** | | | | |
| *ForReading* | *-* | *Integer* | *1* |  |
| *ForWriting* | *-* | *Integer* | *2* |  |
| *ForAppending* | *-* | *Integer* | *8* |  |
| *TristateTrue* | *-* | *Integer* | *-1* |  |
| *TristateFalse* | *-* | *Integer* | *0* |  |
| *TristateUseDefault* | *-* | *Integer* | *-2* |  |

|  |  |
| --- | --- |
| ***Subroutines*** | |
| *ExportObject* | *Export a database object with optional UTF8 to UCS2 conversion.*  ***Inputs:***  *obj\_type\_num – number representing object type (integer)*  *obj\_name - name of object (string)*  *File\_path – full path of file (string)*  *Ucs2Convert – whether to convert the file to UCS2, default is false (Boolean)* |
| *ImportObject* | *Import a database object with optional UTF8 to UCS2 conversion.*  ***Inputs:***  *obj\_type\_num – number representing object type (integer)*  *obj\_name - name of object (string)*  *File\_path – full path of file (string)*  *Ucs2Convert – whether to convert the file to UCS2, default is false (Boolean)* |
| *SanitizeTextFiles* | *For each \*.txt in path, find and remove problematic VB code lines inserted automatically by Access.*  ***Inputs:***  *Path – full file path (string)*  *Ext – file extension (string)* |

### VCS\_ImportExport

VCS\_ImportExport handles import and export of text files used for version control.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***Constants*** | | | | | | | |
| *INCLUDE\_TABLES* | *Private* | *String* | | *Varies by application* | | *List of lookup tables that are part of the program and should be exported with code. Separated by commas.* | |
| *DebugOutput* | *Private* | *Boolean* | | *False* | | *Whether to debug output or not.* | |
| *ArchiveMyself* | *Private* | *Boolean* | | *False* | | *Whether VCS\_code modules should be archived.* | |
| ***Constants for Scripting.FileSystemObject API*** | | | | | | | |
| *ForReading* | *-* | | *Integer* | | *1* | |  |
| *ForWriting* | *-* | | *Integer* | | *2* | |  |
| *ForAppending* | *-* | | *Integer* | | *8* | |  |
| *TristateTrue* | *-* | | *Integer* | | *-1* | |  |
| *TristateFalse* | *-* | | *Integer* | | *0* | |  |
| *TristateUseDefault* | *-* | | *Integer* | | *-2* | |  |

While the module contains several subroutines, the *ExportAllSource* is called most frequently from the MS Access immediate window in preparation for Git versioning.

|  |  |
| --- | --- |
| ***Subroutines*** | |
| *ExportAllSource* | *Export all database objects (forms, reports, queries, macros, modules, lookup tables) to source folder in the same folder as the database.*  ***Inputs:*** *-* |
| *ImportAllSource* | *Import all database objects (forms, reports, queries, macros, modules, lookup tables) from the source folder in the same folder as the database.*  ***Inputs:*** *-* |
| *ImportProject* | *Drop all forms, reports, queries, macros & modules and import these from the source folder.*  ***Inputs:*** *-* |

|  |  |
| --- | --- |
| ***Functions*** | |
| *IsNotVCS* | *Determine if a module is part of VCS code.*  ***Inputs:***  *name – module name (string)*  ***Returns:*** *True if module is part of VCS, false if not (Boolean)* |
| *CloseFormsReports* | *Closes all open forms and reports.*  ***Inputs:*** *-*  ***Returns:*** *-* |

### VCS\_Loader

VCS\_Loader loads files from a source directory containing text based files.

|  |  |
| --- | --- |
| ***Subroutines*** | |
| *LoadVCS* | *Load source files from source directory.*  ***Inputs:***  *Optional*  *SourceDirectory – full path of ‘source’ folder (string)* |

### VCS\_Reference

VCS\_IE\_Functions handles binary file reading and writing.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Constants*** | | | | |
| ***Constants for Scripting.FileSystemObject API*** | | | | |
| *ForReading* | *-* | *Integer* | *1* |  |
| *ForWriting* | *-* | *Integer* | *2* |  |
| *ForAppending* | *-* | *Integer* | *8* |  |
| *TristateTrue* | *-* | *Integer* | *-1* |  |
| *TristateFalse* | *-* | *Integer* | *0* |  |
| *TristateUseDefault* | *-* | *Integer* | *-2* |  |

|  |  |
| --- | --- |
| ***Functions*** | |
| *ImportReferences* | *Import project references from a CSV file.*  ***Inputs:***  *obj\_path – CSV file path (string)*  ***Returns:*** *True if successful, false if not (Boolean)* |
| *ExportReferences* | *Export project references as a CSV file.*  ***Inputs:***  *obj\_path – CSV file path (string)*  ***Returns:*** *True if successful, false if not (Boolean)* |

### VCS\_String

VCS\_IE\_Functions handles binary file reading and writing.

|  |  |
| --- | --- |
| ***Functions*** | |
| *Sb\_Init* | *Initialize string builder.*  ***Inputs:*** *-* |
| *Sb\_Clear* | *Clear string builder.*  ***Inputs:***  *sb() – string builder (string)* |
| *Sb\_Append* | *Append to string builder*  ***Inputs:***  *sb() – string builder (string)*  *Value – content of string (string)* |
| *Sb\_Get* | *Get string builder value*  ***Inputs:***  *sb() – string builder (string)* |

|  |  |
| --- | --- |
| *PadRight* | *Pad the string on the right to make it count characters long.*  ***Inputs:***  *Value – content of string (string)*  *count – length string should be (integer)*  ***Returns:*** *The padded string (string)* |
| *SubString* | *Return part of a string between two positions. Internal brackets are skipped.*  ***Inputs:***  *p – number of characters to return (integer)*  *s – string to return part from (string)*  *startsWith – beginning of string subset (string)*  *endsWith – end of string subset (string)*  ***Returns:*** *Portion of the string (string)* |

### VCS\_Table

VCS\_IE\_Functions handles binary file reading and writing.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Constants*** | | | | |
| ***Constants for Scripting.FileSystemObject API*** | | | | |
| *ForReading* | *-* | *Integer* | *1* |  |
| *ForWriting* | *-* | *Integer* | *2* |  |
| *ForAppending* | *-* | *Integer* | *8* |  |
| *TristateTrue* | *-* | *Integer* | *-1* |  |
| *TristateFalse* | *-* | *Integer* | *0* |  |
| *TristateUseDefault* | *-* | *Integer* | *-2* |  |

|  |  |  |  |
| --- | --- | --- | --- |
| ***Types*** | | | |
| *structEnforce* | | | *Structure to keep track of “on Update” and “on Delete” clauses.*  ***Inputs:***  *foreignTable – name of foreign table (string)*  *foreignFields – name of foreign fields (string)*  *table – table name (string)*  *refFields – array of reference fields (string)*  *isUpdate – whether or not clause is an update (Boolean)* | |
| *ExportLinkedTable* | | *Export linked table to text file.*  ***Inputs:***  *tbl\_Name – name of linked table (string)*  *obj\_path – full path name for file (string)* | |

|  |  |
| --- | --- |
| ***Subroutines*** | |
| *ExportTableDef* | *Export linked table definition to text file.*  ***Inputs:***  *db - databse (database)*  *td – table definition (TableDef object)*  *TableName – name of table (string)*  *directory – full path name for database (string)* |
| *ExportTableData* | *Export desired table to ‘\source\tables’ folder.*  ***Inputs:***  *tbl\_name – table to export (string)*  *obj\_path – full path to export location (string)* |
| *KillTable* | *Drop the specific table from the database.*  ***Inputs:***  *tblName – name of table to drop (string)*  *db – database where the table is located (object)* |
| *ImportLinkedTable* | *Import a specific table into the database.*  ***Inputs:***  *tblName – name of table to import (string)*  *obj\_path – location of table definition file (string)* |
| *ImportTableDef* | *Import a specific table definition into the database.*  ***Inputs:***  *tblName – name of table whose definition is being imported (string)*  *directory – location of table definition file (string)* |
| *ImportTableData* | *Import a specific table data into the database.*  ***Inputs:***  *tblName – name of table whose data is being imported (string)*  *obj\_path – location of table data file (string)* |

|  |  |
| --- | --- |
| ***Functions*** | |
| *formatReferences* | *Format database references.*  ***Inputs:***  *db - databse (database)*  *ff – table definition (TableDef object)*  *TableName – name of table (string)*  ***Returns:*** *SQL reference statement for reference tables (string)* |
| *formatConstraint* | *Format SQL constraint statement.*  ***Inputs:***  *keyw – name of key (string)*  *idx – SQL index (index)*  ***Returns:*** *SQL statement for the index constraint (string)* |

|  |  |
| --- | --- |
| *strName* | *Retrieve text as a bracketed expression which can be used in SQL statements. For example “species” is returned as “[species]”.*  ***Inputs:***  *s – text (string)*  ***Returns:*** *Bracketed string (string)* |
| *strType* | *Determine the object type based on the integer passed in.*  *Returns “VARCHAR” if integer isn’t matched.*  ***Inputs:***  *i – number representing the object (integer)*  ***Returns:*** *Name of the object type (string).* |
| *FieldsIdentical* | *Determin if two fields are identical.*  ***Inputs:***  *ff – field (object)*  *gg – field (object)*  ***Returns:*** *True if fields match, false if not (Boolean)* |
| *FieldInFields* | *Determine if a field is found in a fields collection.*  ***Inputs:***  *fi – field (field object)*  *ff – fields (fields collection)*  ***Returns:*** *True if field is found in fields, false if not (Boolean)* |
| *TableExists* | *Determine if a specific table exists.*  ***Inputs:***  *TName – table name (string)*  ***Returns:*** *True if table is found, false if not (Boolean)* |
| *TableExportSql* | *Retrieve the SQL statement for a table. Sorting is by first to last field.*  ***Inputs:***  *tbl\_name – table name (string)*  ***Returns:*** *SQL statement for generating the table and it’s data (string)* |

# User Actions Required

This is the first version of the NCPN version control modules. No specific actions are necessary aside from those indicated in implementing the associated protocol database and database back end files.

|  |  |
| --- | --- |
| Note | These modules should be removed from protocol databases prior to their release, however if they remain their existence should not impact database functionality. |

1. *See the NCPN\_Git\_Versioning documentation.* [↑](#footnote-ref-1)
2. Modules may be updated from this repository if new functionality or bug fixes warrant doing so. [↑](#footnote-ref-2)