Autor Carles Aubia
Fecha 12/08/2019
Versión 0.1

WDO
Web Database Objects

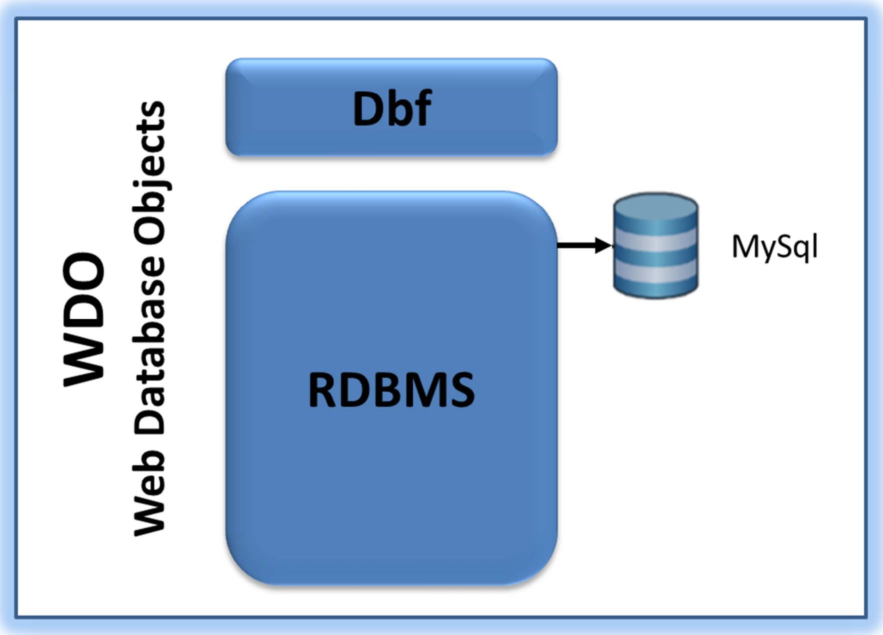
Harbour for Web

Autor Carles Aubia
Fecha 12/08/2019
Versión 0.1

Preamble

Web Database Objects (WDO) is a library that will allow you to connect to databases from mod_harbour in an easy and powerful way. Basically we can connect to any RDBMS and use the data tables in the same way regardless of the data engine.

There will be a particular way to use DBF and another to use popular DBs such as MySql, Postgre, SQLite, Oracle ... In this first version we already have available the connection to Dbf and MySql



The basic steps to run and use this plug-in for Harbour for Web are described below.

Loading library

In order to use the library, we will first have to download it from the repository https://github.com/carles9000/wdo . Imagine we have the wdo_lib. hrb module in \lib\wdo folder, the first thing we will have to do from our program is to load the hrb and we will do it with the LoadHRB() function.

```
//	{% LoadHRB( '/lib/wdo/wdo_lib.hrb' ) %}
```

We note that at the beginning of our prg the comment bar // and then enclosed with the macrosubstitution symbol {% ...%} we load the library with LoadHrb().

At the operating level the line is commented and will not be executed like any comment in the program, but mod_harbour before executing the code replaces and executes what is between {% ...%} as if it were a pre-processed. This gives us a great power to our system being able to execute before our prg different functions that will help us to easily configure our environment.

An also important case and that we will see later is for example in the case of MySql, if we want to specify where we have the dll installed, and we can do it as follows:

```
//	{% HB_SetEnv( 'WDO_PATH_MYSQL', "c:/xampp/htdocs/" ) %}			//	Users Xampp
```

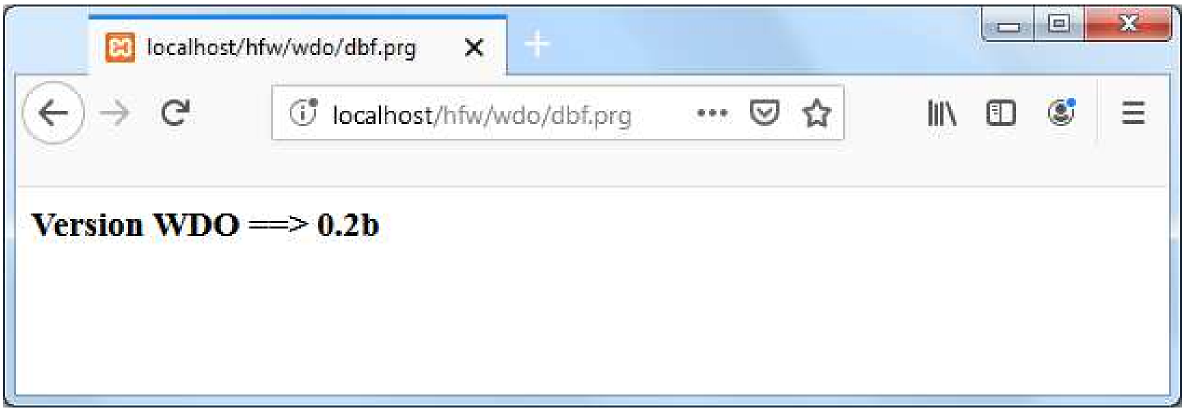
We establish an environment variable that we will use later from our prg.

Hello WDO

The first example is to check if we have the library loaded. It will try to load the library and show the version of this

```
//     --------------------------------------------------------------
//     Title......: WDO Web Database Objects
//     Description: Test WDO
//     Date.......: 28/07/2019
//
//     {% LoadHRB( '/lib/wdo/wdo_lib.hrb' ) %}//     Loading WDO lib
//     --------------------------------------------------------------

FUNCTION Main()

       ?? '<h3>Version WDO ==> ' , WDO():Version()

RETU NIL
```

If the system is ready and prepared, the library version should appear on the screen



Version WDO ==> 0.2b

Autor Carles Aubia
Fecha 12/08/2019
Versión 0.1

DBF

To load an object for the use of Dbf's we will do it as follows

```
o := WDO():Dbf( <cTable>, <cIndex>,  [<lOpen>] )
```

By default the RDD used is DBFCDX
By default <lOpen> is .T. and what it does is open in the same action the indicated table / index

The system will open the dbf and index if it is indicated and will return an instance of the class to the variable.

```
oDb1 := WDO():Dbf( 'customer.dbf', 'customer.cdx' )
```

We can indicate to the WDO a series of data that will be used by default every time we open a table:

```
::cDefaultPath
::cDefaultRdd
```

The way to use it is to create a WDO instance and assign values

```
//      Config Sistema...

        o := WDO():Dbf()
            o:cDefaultPath  := hb_getenv( 'PRGPATH' ) + '/data'
            o:cDefaultRdd   := 'DBFCDX'
```

From this moment, every time we create a WDO, it will search in the indicated path and will use the Rdd by default.

```
oDb1 := WDO():Dbf( 'customer.dbf', 'customer.cdx' )
oDb2 := WDO():Dbf( 'states.dbf', 'states.cdx' )
```

If there is any table outside the default values, we can indicate in the 3 parameter .F. (do not open the table) and change the values. Later we will have to execute the method :: Open ()

```
oDb1 := WDO():Dbf( 'customer.dbf', 'customer.cdx' )
oDb2 := WDO():Dbf( 'states.dbf', 'states.cdx' )

oDb3 := WDO():Dbf( 'vendors.dbf', 'vendors.cdx', .F. )
             oDb3:cPath      := hb_getenv( 'PRGPATH' ) + '/data/sales'
             oDb3:cRdd       := 'DBFNTX'
             oDb3:Open()

oDb4 := WDO():Dbf( 'users.dbf', 'users.cdx' )
```

Once the table is open we can execute all the methods defined in the class and handle the DBF as if it were a typical Tdatabase object

See source code RDBMS_DBF for all available methods.

A basic example of use could be the following

```
//     -------------------------------------------------------------------------------
//     Title......: WDO Web Database Objects
//     Description: Test WDO
//     Date.......: 28/07/2019
//
//     {% LoadHRB( '/lib/wdo/wdo_lib.hrb' ) %}       //     Loading WDO lib
//     -------------------------------------------------------------------------------

FUNCTION Main()

       LOCAL o
       LOCAL oDb1, oDb2, oDb3, oDb4

       ?? 'Init ' + time() + '<hr>'


       //     Config Sistema...

             o := WDO():Dbf()
                    o:cDefaultPath      := hb_getenv( 'PRGPATH' ) + '/data'

                    o:cDefaultRdd             := 'DBFCDX'

       //     Open Tables

             oDb1 := WDO():Dbf( 'customer.dbf', 'customer.cdx' )

             oDb2 := WDO():Dbf( 'states.dbf', 'states.cdx' )

             oDb3 := WDO():Dbf( 'vendors.dbf', 'vendors.cdx', .F. )
                          oDb3:cPath    := hb_getenv( 'PRGPATH' ) + '/data/sales'

                          oDb3:Open()
             oDb4 := WDO():Dbf( 'users.dbf', 'users.ntx', .F. )   //     3 param. Open
                          oDb4:cRdd    := 'DBFNTX'
                          oDb4:Open()

             ? '<b>==> Open tables...</b>'


      //     Seek States...

             ? '<br><b>==> Seek States...</b>'

             oDb2:Focus( 'CODE' )

             IF oDb2:Seek( 'MA' )

                    ? oDb2:FieldGet( 'code' ), oDb2:FieldGet( 'name' )

             ENDIF

      //     List Vendors...

             ? '<br><b>==> List Vendors...</b>'

             oDb3:First()

             WHILE !oDb3:Eof()

                    ? oDb3:FieldGet( 'id' ), oDb3:FieldGet( 'name' ),  oDb3:FieldGet( 'phone'
)
                    oDb3:next()
             END

      //     Info Customer...

             ? '<br><b>==> Info Customer...</b>'

             ? oDb1:cDbf, oDb1:cRdd

             ? 'lOpen', oDb1:lOpen
             ? 'First()', oDb1:first()
             ? 'Fieldget(1)', oDb1:FieldGet( 1 )
             ? 'Fieldget( "first" )', oDb1:FieldGet( 'first' )
             ? 'Count()', oDb1:Count()
             ? 'Fieldname(1)', oDb1:FieldName( 1 )
             ? 'Next(5)', oDb1:next( 5 ), oDb1:Recno(), oDb1:FieldGet( 1 )
             ? 'Goto(7)', oDb1:goto(7), oDb1:Recno(), oDb1:FieldGet( 1 )
             ? 'Prev(2)', oDb1:prev( 2 ), oDb1:Recno(), oDb1:FieldGet( 1 )
             ? 'Last()', oDb1:last(), oDb1:FieldGet( 1 )
             ? 'First()', oDb1:first(),oDb1:Recno()

             IF oDb1:Rlock()
                oDb1:FieldPut( 'street', time() )
             ENDIF

             ? 'Street: ' , oDb1:FieldGet( 'street' ), oDb1:Recno()
             ? '<hr>'


RETU NIL
```

RDBMS – MySql

WDO (): Rdbms () will help you easily connect with the different database managers. MySql is already operational and its use is described below

MySql needs a dll that comes with the mod_harbour library: libmysql.dll and libmysql64.dll for windows versions

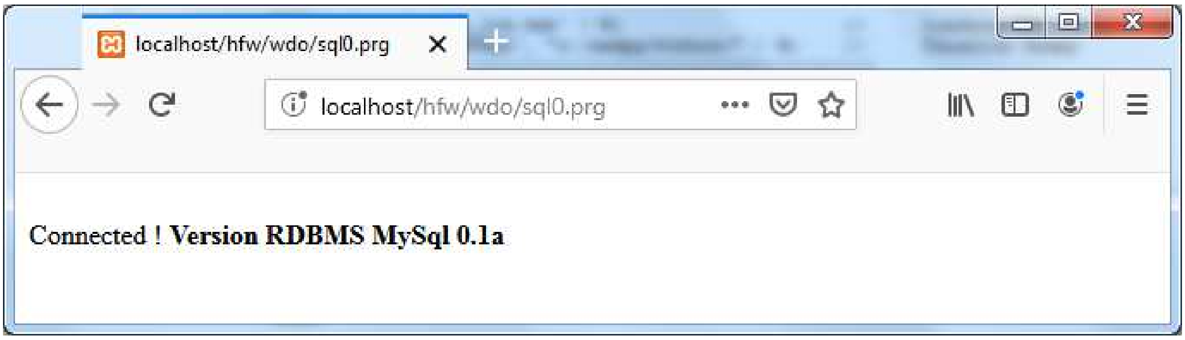
If we have this dll installed in a special directory, for example xampp users in windows, we can indicate its location at the beginning of the program:

//   {% HB_SetEnv('WDO_PATH_MYSQL', "c:/xampp/htdocs/") %}   //   Usuarios Xampp

The basic example to check if the library has been loaded and if you have connected with mysql would be the following:

```
//      --------------------------------------------------------------
//      Title......: WDO Web Database Objects
//      Description: Test WDO
//      Date.......: 28/07/2019
//
//      {% LoadHRB( '/lib/wdo/wdo_lib.hrb' ) %}                     //      Loading WDO lib
//      {% HB_SetEnv( 'WDO_PATH_MYSQL', "c:/xampp/htdocs/" ) %}     //      Usuarios Xampp
//      --------------------------------------------------------------

FUNCTION Main()

        LOCAL o

                o := WDO():Rdbms( 'MYSQL', "localhost", "harbour", "password", "dbHarbour", 3306 )

                IF o:lConnect

                        ? 'Connected !', '<b>Version RDBMS MySql', o:Version()

                ELSE

                        ? o:cError

                ENDIF

RETU NIL
```

If the system is well parameterized, a screen similar to this should appear:



Connected ! **Version RDBMS MySql 0.1a**

Once connected the class offers the following methods

METHOD Query(cSql)
METHOD Count(hRes)
METHOD FCount(hRes)
METHOD LoadStruct()
METHOD DbStruct()
METHOD Fetch(hRes)
METHOD Fetch_Assoc(hRes)
METHOD FetchAll(hRes, lAssociative)
METHOD Free_Result(hRes)

A basic example of use could be the following:

```
//      --------------------------------------------------------------
//      Title......: WDO Web Database Objects
//      Description: Test WDO
//      Date.......: 28/07/2019
//
//      {% LoadHRB( '/lib/wdo/wdo_lib.hrb' ) %}                   //      Loading WDO lib
//      {% HB_SetEnv( 'WDO_PATH_MYSQL', "c:/xampp/htdocs/" ) %}   //      Usuarios Xampp
//      --------------------------------------------------------------

FUNCTION Main()

        LOCAL o, oRs, a

                o := WDO():Rdbms( 'MYSQL', "localhost", "harbour", "password", "dbHarbour", 3306 )


                IF o:lConnect

                        ?? 'Version WDO', o:ClassName(), o:Version()

                        IF !empty( hRes := o:Query( 'select * from users' ) )

                                ? 'Count(): ', o:Count( hRes )
                                ? 'Fields: ',  o:FCount( hRes )


                                ? '<br><b>Fields</b>'
                                for n := 1 to len( o:aFields )
                                        ? o:aFields[n][1], o:aFields[n][2]
                                next

                                ? '<br><b>Data</b>'
                                //while ( !empty( a := o:Fetch( hRes ) ) )
                                //      ? valtochar( a )
                                //end

                                //      Associative array
                                while ( !empty( a := o:Fetch_Assoc( hRes ) ) )
                                        ? valtochar( a )
                                end

                        ELSE

                                ? 'Error: ' , o:cError

                        ENDIF

                ELSE

                        ? 'Error: ' , o:cError

                ENDIF

RETU NIL
```