



Manual de VFP y MySQL

Utilizando MySQL con C#

HERRAMIENTAS UTILIZADAS:

Herramienta	Donde se consigue
C#	<ul style="list-style-type: none">- Pueden usar el beta MSV .NET- Descargar el "Trial" de C# Builder
MySQL 4.0.17	<ul style="list-style-type: none">- www.mysql.com
MySQLDriverCS	<ul style="list-style-type: none">- www.sourceforge.net
MS .NET Framework 1.1	<ul style="list-style-type: none">- www.microsoft.com

Voy a suponer que se tienen los conocimientos necesarios de C# y que ya tienen instalado MySQLDriverCS.

ANTES DE EMPEZAR

Es necesario que instalen el MS .NET Framework 1.1 (o posterior) ya que el driver de MySQL no funciona, marca errores de I/O.



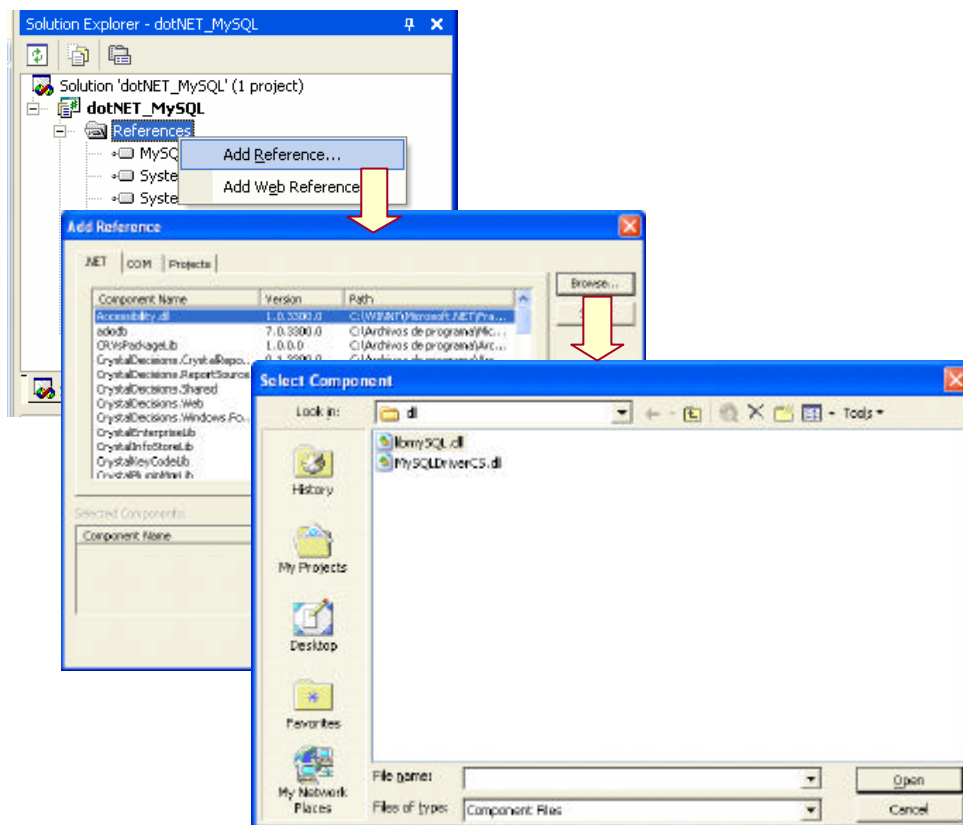
Manual de VFP y MySQL

Utilizando MySQL con C#

AGREGANDO LA REFERENCIA DEL DRIVER

Creen una nueva aplicación C#.

En la ventana "Solution Explorer" hagan clic con el botón derecho del ratón y seleccionen la opción "Agregar referencia"; presionen el botón "Browse" y seleccionen el archivo MySQLDriverCS.dll (como se muestra en la figura).



Agregen la sentencia using MySQLDriverCS en el código del formulario.



Manual de VFP y MySQL

Utilizando MySQL con C#

EL CÓDIGO

La conexión:

```
//Definimos las variables
MySQLConnection MyCnx;           //Objeto de conexión
MySQLConnectionString MyCnxStr;  //Objeto de la cadena de conexión

//Creamos la cadena de conexión, los parámetros son:
//1- El IP del servidor donde se encuentra la base de datos
//2- Nombre de la base de datos a utilizar
//3- Nombre de usuario
//4- Clave de acceso del usuario
MyCnxStr = new MySQLConnectionString(txtIP.Text.Trim(),
txtDB.Text.Trim(), txtUserName.Text.Trim(), txtPWD.Text.Trim());

//Creamos la conexión, el parámetro es la cadena de conexión como string
//por eso utilizamos la propiedad AsString del objeto
MyCnx = new MySQLConnection(MyCnxStr.AsString);
try
{
    //Intentamos conectarnos con el servidor
    MyCnx.Open();

    //Si se conecta habilitamos y/o deshabilitamos botones
    if (MyCnx.State == ConnectionState.Open)
    {ActivaControles(true);}
    else
    {MessageBox.Show(this, "No se logró la conexión con la base de
datos", "MyD Samples", MessageBoxButtons.OK, MessageBoxIcon.Error );}

}
catch (Exception MyEx)
{
    MessageBox.Show(this, "Ocurrió un error al intentar conectarse con
la base de datos " + MyEx.Message, "MyD Samples", MessageBoxButtons.OK,
MessageBoxIcon.Error );
}
```

Ejecutando sentencias en la base de datos

```
//Definimos las sentencias a ejecutar en la base de datos
//Para crear una base de datos:
String SQLCmd_DB = "CREATE DATABASE IF NOT EXISTS dotNET_MySQL";

//Para cambiar la base de datos en uso:
String SQLCmd_SelectDB = "USE dotNET_MySQL";

//Para crear la tabla:
String SQLCmd_Table = "CREATE TABLE IF NOT EXISTS Clientes (IDCliente INT NOT
NULL AUTO_INCREMENT PRIMARY KEY, Nombre VARCHAR(250) NOT NULL)";
```



Manual de VFP y MySQL

Utilizando MySQL con C#

```
//Para ejecutar una consulta
String SQLCmd_Query = "SELECT * FROM Clientes";

//Flag que indica que es lo que estamos haciendo
String EstadoCmd;
EstadoCmd = "";

//Creamos el objeto que ejecutará los commands en la base de datos
//Los parámetros son:
//1- Sentencia SQL (válida)
//2- Objeto conexión
MySQLCommand MyCmd = new MySQLCommand(SQLCmd_DB, MyCnx);
try
{
    /* Creamos la base de datos *
    //Establecemos el flag
    EstadoCmd = "Creando la base de datos";
    //Ejectamos el método ExecuteNonQuery porque no esperamos
    //que el servidor nos devuelva algún valor.
    MyCmd.ExecuteNonQuery();
    //Agregamos un item al listbos indicando lo que se ha hecho
    lbResult.Items.Add("Base de datos creada...");

    /* Seleccionamos la base de datos *
    //Nota: si no lo hacemos, la tabla se creará en la base de datos
    // que se está utilizando de acuerdo con el código de ejemplo
    // la tabla se crearía en la base de datos MySQL
    EstadoCmd = "Seleccionando base de datos";
    //Establecemos la nueva sentencia a ejecutar
    //en el primer caso no se establecio de esta forma ya que
    //se estableció cuando creamos el objeto
    MyCmd.CommandText = SQLCmd_SelectDB;
    //Ejecutamos la sentencia
    MyCmd.ExecuteNonQuery();
    lbResult.Items.Add("Base de datos seleccionada...");

    /* Creamos la tabla *
    EstadoCmd = "Creando la Tabla";
    MyCmd.CommandText = SQLCmd_Table;
    MyCmd.ExecuteNonQuery();
    lbResult.Items.Add("Tabla de Clientes creada...");

    /* Agregamos registros a la tabla *
    EstadoCmd = "Agregando registros";
    //Agregamos un registro a la tabla creada
    MyCmd.CommandText = "INSERT INTO Clientes (Nombre) VALUES ('Maricela')";
    MyCmd.ExecuteNonQuery();
    MyCmd.CommandText = "INSERT INTO Clientes (Nombre) VALUES ('Ana Fernanda')";
    MyCmd.ExecuteNonQuery();
    MyCmd.CommandText = "INSERT INTO Clientes (Nombre) VALUES ('Ana Sofia')";
    MyCmd.ExecuteNonQuery();
    lbResult.Items.Add("Registros agregados...");
```



Manual de VFP y MySQL

Utilizando MySQL con C#

```
/** Ejecutamos la sentencia *
EstadoCmd = "Ejecutando la consulta";
MyCmd.CommandText = SQLCmd_Query;

//(MySQLDataReader) se antepone a MyCmd.Execute(..) porque no se hace
//automáticamente la "conversion" a ese tipo de datos

//Como ahora si nos va a regresar un recordset (no se si llame así
//ahora en .NET) utilizamos el método ExecuteReader con el parámetro
//con el parámetro SingleResult

//Creamos el objeto DataReader (de mysql)
MySQLDataReader MyReader =
(MySQLDataReader)MyCmd.ExecuteReader(CommandBehavior.SingleResult);
lbResult.Items.Add("** Resultado **");

//Recorremos el resultado y agregamos los datos al listbox
while (MyReader.Read())
{
    lbResult.Items.Add(MyReader.GetString(0) + " - " +
MyReader.GetString(1));
}
//Cerramos el objeto Reader
MyReader.Close();
lbResult.Items.Add("** fin de registros **");
}
catch (Exception MyEx)
{
    MessageBox.Show(this, "Ocurrió un error durante: '" + EstadoCmd + "' -> " +
MyEx.Message, "MyD Samples", MessageBoxButtons.OK, MessageBoxIcon.Information);
}
}
```



Manual de VFP y MySQL

Utilizando MySQL con C#

RESULTADOS

Los resultados se presentan en la siguiente figura:

MySQL & dotNET - MyD Samples

Conección

IP: 127.0.0.1 Base de datos: MySQL

Nombre de usuario: root Conectar

Clave de acceso: Desconecta

Crear una base de datos

Nombre: Crear

Ejemplo

Este ejemplo crea una base de datos (dotNET_MySQL) una tabla de clientes (Clientes) agrega registros a la tabla y ejecuta una consulta sobre la tabla

Base de datos creada...
Base de datos seleccionada...
Tabla de Clientes creada...
Registros agregados...
*** Resultado ***
1 - Maricela
2 - Ana Fernanda

Ejecuta

Cerrar