**MySQL  Connector/Arduino**

Dr.  Charles  Bell  January  2016

v1.1.1a

Traducción: @fenixbinario

Alguna vez has querido para conectar el proyecto Arduino a una base de datos para almacenar cualquiera de los datos que ha recogido o recuperar los datos guardados para desencadenar eventos en su dibujo? Pues bien, ahora se puede conectar el proyecto Arduino directamente a un servidor MySQL sin necesidad de utilizar un ordenador intermedio o un web-- o nube - servicio basado. Tener acceso directo a un servidor de base significa que puede almacenar datos adquiridos de su proyecto, así como valores de comprobación almacenada en tablas en el servidor.

Esto también significa que usted puede configurar su propia, servidor MySQL local para almacenar los datos de la eliminación de la necesidad de que la conexión a Internet. Si eso no es un problema, todavía se puede conectar a y almacenar datos en un servidor MySQL a través de su red, Internet, o incluso en la nube!

El MySQL Connector / Arduino es una biblioteca que le permite hacer exactamente eso y más! Este documento está destinado a ayudar a aprender acerca de la biblioteca MySQL Connector / Arduino y descubrir sus capacidades de gran alcance. También podrá descubrir información relevante sobre cómo utilizar la biblioteca, así como consejos de solución de problemas vitales para superar los problemas. Lea sobre como descubrir las capacidades y vemos ejemplos de cómo funciona esta biblioteca.

**Nota**     *Esta versión del conector es una revisión importante de la rama 1.0. Si usted ha estado utilizando la versión 1.0.x, es posible que desee leer la sección titulada, "Cambios de versiones previas" para ver qué ha cambiado.*

**Empezando**

Si ha utilizado algunos de los otros métodos de almacenamiento de datos desde un Arduino tales como la escritura de datos en la memoria flash (por ejemplo, una tarjeta digital segura) o un dispositivo de memoria EEPROM, es probable que tuviera que escribir código adicional para leer los datos para su uso posterior. Si desea utilizar esos datos en un dispositivo que no sea el Arduino, es probable que tuviera que copiar manualmente los datos con el fin de usarlo. El uso de una base de datos para almacenar los datos puede eliminar el método de copia de datos y la extracción manual de conjunto. Del mismo modo, si el proyecto es tal que no puede o no desea conectarse a Internet para guardar los datos, la capacidad de escribir a un servidor de base de datos local resuelve ese problema también.

Guardar los datos en una base de datos no sólo preservará los datos para su análisis en un momento posterior, sino que también significa que su proyecto puede alimentar datos a las aplicaciones más complejas que hacen uso de los datos. Mejor aún, si usted tiene proyectos que utilizan grandes valores de datos para los cálculos o las búsquedas, puede almacenar los datos en el servidor y recuperar sólo los datos que necesita para el cálculo o el funcionamiento de todo sin ocupar grandes bloques de memoria en la placa Arduino. Claramente, esto abre una nueva vía conjunto de proyectos de Arduino!

La tecnología se denomina conector / Arduino (para su uso en este documento simplemente, el conector). El conector gestiona el protocolo de comunicación del cliente de MySQL en una biblioteca construida para la plataforma Arduino. De hecho, todos los mecanismos para comunicarse con el servidor MySQL están ocultos por lo que no necesita aprender las minucias del protocolo.

**Nota**       *De aquí en adelante nos referiremos a Conector / Arduino cuando se habla de conceptos generales y funciones y se refieren al código fuente real usando el término de la biblioteca / Arduino conector o simplemente la biblioteca.*

Sketches (programas) escrita para utilizar el permiso de la biblioteca que le permite codificar las sentencias SQL para insertar datos

y ejecutar consultas pequeñas para devolver datos de la base de datos (por ejemplo, usando una tabla de consulta).

Usted puede preguntarse cómo un microcontrolador limitada de memoria y procesamiento posiblemente puede soportar código para insertar datos en un servidor MySQL. Podemos hacer esto porque el protocolo para la comunicación con un servidor MySQL no sólo está bien conocida y documentada, sino también intencionalmente diseñado para ser ligero. Que es uno de los pequeños detalles que hacen de MySQL atractivo para los desarrolladores de sistemas embebidos.

Con el fin de comunicarse con MySQL, el Arduino se debe conectar con el servidor de MySQL a través de una red. Para ello, el Arduino debe utilizar un protector de Ethernet o WiFi y estar conectado a la misma red Ethernet como servidor de base de datos. De hecho, la biblioteca es compatible con la mayoría de los nuevos escudos Arduino Ethernet y clones compatibles que apoyan la biblioteca estándar de Ethernet.

**Requisitos Hardware**

El conector requiere un clon de Arduino o Arduino con al menos 32k de memoria. Si está utilizando un Arduino mayor como el Duemilanove, asegúrese de que tiene la versión que utiliza el procesador ATmega328P.

Si el Sketch es más que una pocas líneas de largo, que está utilizando una gran cantidad de bibliotecas, tienen una gran cantidad de sensores conectado, o desea consultas complejas, se debe considerar el uso de una de las más grandes (como en la memoria, no el tamaño físico) Tableros tales como el Mega o Debido. Veremos qué esto es así en una sección posterior.

El conector también requiere el escudo Arduino Ethernet o equivalente. Esto es porque la biblioteca de referencia a la biblioteca Ethernet escrito para el escudo Ethernet. Si tienes alguna otra forma de escudo Ethernet, Ethernet o el escudo actual utiliza una biblioteca diferente, tendrá que hacer una ligera modificación a la biblioteca para utilizarlo.

Por último, el conector está escrito específicamente para el Arduino Ethernet y escudos o módulos que son compatibles con la clase Ethernet que viene con el IDE Arduino WiFi. Si tiene otro escudo o módulo que requiere una biblioteca adicional, lo más probable es que no funcionará con este conector. Sólo esos escudos que utilizan el Arduino - suministran clase Ethernet funcionará.

**Precaución***La compatibilidad en este sentido significa que se puede utilizar cualquier escudo o módulo que implementa la misma firma de clase (métodos) como el Arduino - suministrado clase Ethernet. Si desea utilizar el conector con otra biblioteca, tendrá que escribir una clase intermedia para traducir la biblioteca que desea utilizar para la firma de clase de cliente Ethernet.*

**Una Nota acerca de la Memoria**

El conector se implementa como una biblioteca de Arduino. Mientras que el protocolo es ligero, la biblioteca hace consumir algo de memoria. De hecho, la biblioteca requiere aproximadamente 20K de memoria flash para cargar. Por lo tanto, se requiere de los ATmega328 o procesador similar con 32k de memoria flash.

Esto puede parecer que no hay mucho espacio para la programación de su nodo sensor, pero como resulta que realmente no necesita mucho para la mayoría de los sensores. Si lo hace, siempre se puede intensificar a un nuevo Arduino con más memoria. Por ejemplo, la última Arduino, la causa, tiene 512k de memoria para el código del programa. En base a eso, una mera 20k es una cantidad insignificante de los gastos generales.

Sin embargo, las limitaciones de memoria se puede llegar fácilmente al utilizar bibliotecas adicionales. Cada biblioteca se carga consumirá memoria lo que se reduce la memoria disponible para las variables dinámicas. El conector debe asignar memoria para almacenar la consulta de ser enviado (como una cadena estática), así como los resultados devueltos (memoria dinámica). Por lo tanto, si usted tiene varias consultas que desee enviar, cada uno de

los que requerirán espacio y si devuelve filas de una consulta, cada fila requiere espacio. Una combinación

de éstos a lo largo puede causar un boceto de complejidad moderada en una Uno se quede sin espacio.

Se puede hacer mucho para mitigar este problema. Puede utilizar una tarjeta con más memoria, reducir el número de variables, reducir el tamaño de las filas devueltas (especificando una lista de columnas en lugar de SELECT \*), y limitar el uso de las bibliotecas y código innecesario. Pero la tarea más importante que puede hacer es comprobar que su boceto para pérdidas de memoria.

Las pérdidas de memoria hará que su dibujo se bloquee cuando se queda sin memoria. Vea la sección de preguntas frecuentes a continuación para más detalles. Por regla general, sugiero dejando al menos 800 bytes de memoria disponibles para las variables dinámicas. Esto se puede ver cuando se compila el boceto como se muestra a continuación.

Sketch uses 20,654 bytes (64%) of program storage space. Maximum is 32,256 bytes.

Global variables use 1,186 bytes (57%) of dynamic memory, leaving 862 bytes for local variables. Maximum is 2,048 bytes.

Aquí vemos el dibujo deja sólo 862 bytes para las variables locales. Esto debería ser suficiente para la mayoría

pequeños bocetos que simplemente se escriben datos en la base de datos.

Sin embargo, tenga en cuenta lo que sucedería si quisiera recuperar una fila de la base de datos que fue

400 bytes de longitud. El conector tendría que asignar memoria para esa fila y, por tanto, dejar alrededor de 450 bytes a la izquierda. Teniendo en cuenta el Arduino utiliza esta memoria (por ejemplo la pila), esto no es suficiente memoria para permitir el boceto para funcionar correctamente. El resultado final es el Arduino es probable que cuelgue.

Tenga en cuenta también el caso en que usted no puede liberar la memoria dinámica asignada. En este caso, considere un caso donde queremos recuperar una muy pequeña cantidad de datos - por ejemplo alrededor de 40 bytes. El croquis probable que se ejecuta bien durante algún tiempo, dependiendo de la frecuencia con la que recuperar los datos, pero cada vez que lo hace, de 40 bytes más será asignado y no devuelto por tanto, al cabo de unos 10 o así las consultas, el Arduino se quedará sin memoria y congelar.

**Networking - Hardware en la Red**

El hardware de red deben ser los dispositivos habituales y normales que se encuentran típicamente en un hogar o pequeña oficina. Es decir, usted debe tener algún tipo de router o punto de acceso que le permite conectar su red Ethernet o WiFi escudo a la red.

Por ejemplo, un puerto de acceso inalámbrico típico o módem de cable tendrán puertos Ethernet adicionales que se pueden utilizar para conectar un escudo Ethernet usando un cable de Ethernet normales (similar CAT5 o). No utilice un cable de conexión a menos que sepa lo que es y cómo usarlo.

Del mismo modo, si se utiliza un escudo Wi-Fi, el router WiFi debe permitir conexiones con los protocolos de seguridad compatibles (<https://www.arduino.cc/en/Main/ArduinoWiFiShield> para más detalles).

Por otra parte, el servidor MySQL y Arduino deben residir en segmentos de red que son alcanzables. Idealmente, deben estar en la misma subred, pero eso no es un requisito duro.

Si no está seguro de la configuración de red o que está tratando de construir una solución en un laboratorio en un soporte universidad, colegio o en el trabajo, usted debe buscar local de TI para ayudarle a configurar su hardware en la red.

**MySQL Servidor**

Los requisitos para el servidor MySQL para su uso con el Arduino son simples. Primero y ante todo, debe asegurarse de que la configuración del servidor MySQL para permitir las conexiones de red. Consulte el manual de referencia de MySQL en línea para más detalles acerca de la plataforma - instalación específica y la configuración (<http://dev.mysql.com/doc/>).

**Nota**       *El conector está diseñado para trabajar con MySQL 5.0 y posterior utilizando los últimos protocolos de cliente. Si desea utilizar el conector con una versión más nueva, más segura del servidor MySQL, debe asegurarse de que está utilizando los protocolos de autenticación de la era 5.X. El conector no funciona con un protocolo de autenticación personalizado.*

Más específicamente, debe asegurarse de que su servidor MySQL no está configurado para unirse en una dirección de red (comentando bind\_address en my.cfg) y que no hay servidores de seguridad o bloqueo de puertos de software para prohibir el acceso al servidor. Por ejemplo, no es raro para anti agresiva - virus y firewall para bloquear el acceso al puerto 3306 (el puerto de escucha predeterminado para MySQL).

Por último, el conector está diseñado para trabajar con el servidor MySQL. No funciona con otra

servidores de bases de datos. Por lo tanto, no se puede utilizar con otros sistemas de bases de datos.

**Cuentas de Usuario**

Usted necesitará una cuenta de usuario y una contraseña para utilizar en su boceto. Si bien no estamos necesariamente preocupados por estrictos protocolos de seguridad (pero no hay nada de malo en ello) como el usuario y la contraseña se han codificado en el dibujo (al menos, en los siguientes ejemplos - usted puede utilizar sus propios métodos, más seguros si prefieres).

Este es quizás el primer error usuarios hacen. O bien utilizar la cuenta de root sin contraseña (no recomendable) o que crean un usuario que no está permitido para conectarse a la base de datos. Más precisamente, MySQL usa una combinación de usuario y huésped para formar un inicio de sesión. Examine las siguientes declaraciones. ¿Son los mismos usuarios o por distintos usuarios?

CREATE USER [bob@localhos HYPERLINK "mailto:bob@localhost"t HYPERLINK "mailto:bob@localhost" HYPERLINK "mailto:bob@localhost"IDENTIFIE HYPERLINK "mailto:bob@localhost"D HYPERLINK "mailto:bob@localhost" HYPERLINK "mailto:bob@localhost"B HYPERLINK "mailto:bob@localhost"Y HYPERLINK "mailto:bob@localhost" HYPERLINK "mailto:bob@localhost"' HYPERLINK "mailto:bob@localhost"secret HYPERLINK "mailto:bob@localhost"' HYPERLINK "mailto:bob@localhost"; HYPERLINK "mailto:bob@localhost" HYPERLINK "mailto:bob@localhost"CREAT HYPERLINK "mailto:bob@localhost"E HYPERLINK "mailto:bob@localhost" HYPERLINK "mailto:bob@localhost"USE HYPERLINK "mailto:bob@localhost"R HYPERLINK "mailto:bob@localhost" HYPERLINK "mailto:bob@localhost"bob@ HYPERLINK "mailto:bob@localhost"' HYPERLINK "mailto:bob@localhost"192.168.0.5 HYPERLINK "mailto:bob@localhost"' HYPERLINK "mailto:bob@localhost" HYPERLINK "mailto:bob@localhost"IDENTIFIE HYPERLINK "mailto:bob@localhost"D HYPERLINK "mailto:bob@localhost" HYPERLINK "mailto:bob@localhost"B HYPERLINK "mailto:bob@localhost"Y HYPERLINK "mailto:bob@localhost" HYPERLINK "mailto:bob@localhost"' HYPERLINK "mailto:bob@localhost"secret HYPERLINK "mailto:bob@localhost"' HYPERLINK "mailto:bob@localhost";](mailto:bob@localhost)

La respuesta puede sorprenderte. Son dos usuarios diferentes a pesar de que el nombre de usuario y la contraseña son los mismos! Uno tiene permitido conectarse sólo a través de la máquina host local. Es decir, el usuario debe estar conectado a la misma máquina que el servidor. El otro se le permite conectarse al servidor si y sólo si el usuario se encuentra en una máquina con una dirección IP de 192.168.0.5.

Ahora considere la placa Arduino se conecta a través de una red Ethernet (o escudo WiFi) - ver más abajo. Esto significa que su Arduino recibirá su propia dirección IP y cualquier usuario que se conecta al servidor debe ser validado a través del nombre de usuario y host (IP). Por ejemplo, si su Arduino se asigna la dirección de

192.168.0.11, que no puede conectarse a través de cualquiera de las cuentas de usuario creadas por encima! Por otra parte, si su Arduino utiliza DHCP para obtener una dirección IP, puede que no sepa lo que se da la dirección IP. Así que, ¿cómo superar esto? Utilizar el enmascaramiento.

La forma más sencilla de crear el usuario es mediante el uso de un comodín para el nombre de host de la siguiente manera.

CREATE USER bob@’%’ IDENTIFIED BY 'secret';

Por supuesto, esto no es muy seguro ya que el usuario puede conectarse desde cualquier host, pero va a llegar a donde necesita estar y es suficiente para la mayoría de los proyectos de Arduino. Sin embargo, si quieres una cuenta de usuario más seguro, usted podría limitar los anfitriones a una subred de la siguiente manera.

CREATE USER bob@’192.168.0.%’ IDENTIFIED BY 'secret';

**Privilegios**

También tendrá que dar al usuario el acceso a cualquier base de datos (s) que desea acceder. No voy a entrar en todos los detalles aquí, pero basta con decir que necesita para conceder permisos a cada usuario en función de lo que quiere hacer. Por ejemplo, si sólo desea leer los datos, un simple permiso SELECT es todo lo que se necesita. En el siguiente ejemplo, voy al otro extremo y concedo todos los permisos para el usuario. Esto está bien ya que estoy tanto impedir al usuario a un host específico y limitar el acceso a una base de datos única. Observar.

GRANT ALL ON test\_arduino.\* TO bob@’192.168.0.11’;

Aquí tengo que dar al usuario acceso y todos los permisos para cualquier objeto en la base de datos test\_arduino. Tenga en cuenta que, por defecto, en ausencia de cualquier otro comando GRANT, el usuario no tiene acceso a otras bases de datos en el sistema. Esta es la razón por usar el usuario root es una mala idea. Si comete un error en su boceto y de actualizar o suprimir la fila equivocada o peor elimina la base de datos, que va a perder permanentemente los datos. Siempre use una cuenta de usuario cree de nuevo con permisos mínimos para sus proyectos de Arduino.

**Acceso de Prueba**

Una vez que su cuenta de usuario se configura y se han concedido los permisos correctos, usted debe comprobar la conexión. Esta es quizás la única cosa que la mayoría de los nuevos usuarios se saltan y asumen todo funcionará. Por otra parte, cometen el error de poner a prueba la cuenta de usuario desde la misma máquina que el servidor. Esto enmascarar una serie de peligros potenciales y no es la mejor prueba.

La mejor manera de comprobar si su cuenta de usuario y permisos es el uso de otro equipo para iniciar sesión en el servidor MySQL. Sólo tiene que conectar otro equipo de la misma red y abrir el cliente de mysql

e intente conectarse. Si no tiene otro equipo a utilizar, puede obligar al cliente de MySQL para utilizar la red para conectar (en oposición a la toma por defecto, conexiones) especificando el host y el puerto como se muestra a continuación. El anfitrión en este caso es el nombre de host o dirección IP del servidor MySQL, no el Arduino.

mysql -ubob -psecret –h192.168.0.2 --port=3306

Una vez que se haya conectado, intente acceder a la base de datos y se tratará de realizar las operaciones que desee incluir en su dibujo. Ahora sería un buen momento para poner a prueba las sentencias SQL que planea incluir en su dibujo. Simplemente escriba en ellos y ejecutarlos. Por ejemplo, trate de insertar algunos datos ficticios, la creación de objetos, la selección de filas - lo que planea hacer en su boceto. Esto no sólo verificar su cuenta de usuario tiene los permisos correctos; verificará, además, las sentencias SQL están correctamente escritas y así evitar errores extraños cuando se ejecuta la secuencia de comandos.

Una vez que puede conectarse correctamente y haya verificado que puede acceder a los objetos de base de datos correctos,

restablecer los datos y transferir la información de acceso a su boceto o anótelo para su posterior consulta.

**Cómo obtener MySQL Connector / Arduino**

La forma más fácil de empezar a usar el conector es utilizar el Administrador de biblioteca para descargar e instalar

el conector. Basta con abrir el Administrador de bibliotecas en el IDE de Arduino de la *Sketch-­‐>Include  Library-­‐*

*>Manage  Libraries*  menu.  Esto abre el gestor de bibliotecas. En el filtro de su cuadro de búsqueda, escriba "MySQL" a continuación, elija el conector y haga clic en Instalar. En cuestión de segundos, la nueva biblioteca está instalado y listo para su uso. También puede visitar el Administrador de biblioteca y actualizar la biblioteca de conector cada vez que se libera una nueva versión. Por lo tanto, se puede mantener más fácilmente las bibliotecas de arriba - a - fecha!

Si no desea utilizar el Administrador de biblioteca (o no puede porque está utilizando un IDE o editor diferente), también se puede descargar desde GitHub, descomprimirlo, y lo coloca en la carpeta de Arduino / Bibliotecas. Puede descargar Conector / Arduino desde GitHub (<https://github.com/ChuckBell/MySQL_Connector_Arduino>). La biblioteca es de código abierto, con licencia como GPLv2, y es propiedad de Oracle Corporation. Por lo tanto, cualquier modificación a la biblioteca que tiene la intención de compartir deben cumplir con la licencia GPLv2.

Una vez que haya descargado la biblioteca, tiene que copiar o mover a la carpeta de Arduino / Bibliotecas. Colocar la carpeta del archivo .zip MySQL\_Connector\_Arduino llamado a sus bocetos carpeta de biblioteca.

Puede localizar el punto donde esto es mediante el examen de las preferencias para el entorno Arduino como se muestra en la Figura 1. Por ejemplo, mis bocetos carpeta en mi Mac es  /Users/cbell/Documents/Arduino.  Por lo tanto, he copiado a la carpeta /Users/cbell/Documents/Arduino/Libraries/.

**Nota**   *Sólo necesita una copia del conector en la carpeta de bibliotecas. No coloque el mysql\_ \* archivos en la carpeta de croquis o colocar una segunda copia en otro lugar. Si lo hace, dará lugar a errores de compilación como el IDE no sabrá qué archivos de la biblioteca de su uso.*



*Figure  1:  Arduino  Preferences  Dialog*

**Tip**         *Si copia una biblioteca para las bibliotecas de la carpeta durante la aplicación Arduino está en ejecución,*

*debe reiniciarlo para detectar la nueva biblioteca.*

**Writing Sketches using Connector/Arduino**

Ok,  now  that  you’ve  downloaded  the  connector,  what  do  you  do  with  it?  This  section  will  explain  the  steps  needed  to  write  your  first,  simple  sketch.  We  will  also  see  some  examples  of  more  advanced  sketches  to  give  you  an  idea  of  what  is  possible.  But  first,  let’s  discuss  some  requirements  from  the  MySQL  side  of  things  then  move  on  to  the  physical  and  network  connection.  Paying  attention  here  will  save  you  tons  of  time  troubleshooting  later!

I  begin  with  the  trivial  sketch  –  simply  connecting  to  the  database  server.  I  then  present  examples  on  how  to  write  sketches  to  do  the  most  common  options.  Keep  in  mind  these  are  examples  and  that  your  specific  needs  may  require  additional  changes.

**Cómo realizar la conexión Ethernet Uso del Escudo**

Lo primero que hay que hacer cuando la creación de un nuevo croquis de utilizar el conector es incluir las bibliotecas y las variables correctas. Por lo general, colocamos las llamadas iniciales al conector para la puesta en marcha en el método de configuración (). Esto incluye no sólo el código de inicio, sino también la llamada para conectar con el servidor. Presento todas ellas en el paso - sabia orden de salida con un nuevo dibujo. Si quieres seguir a lo largo, abrir el IDE de Arduino y crear un nuevo dibujo, en blanco.

Si bien este y los siguientes ejemplos demuestran cómo utilizar la biblioteca, tenga en cuenta que hay varias variaciones de estilo e incluso la elección del flujo de codificación que también son ejemplos válidos. Recomiendo probar estos como por escrito antes de adaptarlos a su propio estilo.

En primer lugar, es necesario añadir el archivos de inclusión. Estos incluirán todas las bibliotecas necesarias para compilar y ejecutar un boceto usando la biblioteca. Note que hemos incluido la librería Ethernet, que es un built - in biblioteca que no es necesario descargar. A continuación se incluyen la biblioteca MySQL\_Connector en forma de

incluyendo el archivo de cabecera. Recuerdan los archivos de cabecera y fuente son parte del archivo .zip que ha descargado

que incluye el conector.

#include "Ethernet.h"

#include "MySQL\_Connector.h"

A continuación, hay un par de estados necesarios para inicializar y trabajar con la clase de Ethernet. Estos se muestran a continuación. Estos incluyen la dirección de control de acceso al medio (CAM1) del Arduino y la dirección IP del servidor MySQL. Esta **no es la dirección IP** de la Arduino! La dirección MAC puede ser cualquier 6 válida, - posición de la dirección hexadecimal que no figure en la red. Por lo tanto, se puede usar el uno en este ejemplo, pero para los proyectos con múltiples nodos Arduino Ethernet tendrá que hacer cada dirección MAC única.

byte mac\_addr[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED }; IPAddress server\_ip(10, 0, 1, 35);

La siguiente sección define una instancia de la clase del cliente Ethernet y la clase de conexión para el conector. Aquí hay que definir el cliente Ethernet principio, ya que se pasa a la clase conector MySQL como un parámetro requerido.

EthernetClient client;

MySQL\_Connection conn((Client \*)&client);

**Tip**       *Este es uno de ellos muchas mejoras en la nueva versión. Ahora, siempre y cuando la clase es compatible con la clase de cliente Ethernet, se puede utilizar cualquier clase para iniciar el conector. Lo que significa que puede utilizar otro,  non-­‐ Arduino  library  too  –  al igual que siempre, ya que tiene el cliente de Ethernet como su ancestro.*

La siguiente sección incluye las variables que vamos a utilizar para suministrar las credenciales de usuario para la conexión. En este caso, necesitamos una variable para crear una instancia del conector, un nombre de usuario y una contraseña. Asegúrese de utilizar la cuenta de usuario y la contraseña que previamente probado.

char user[] = "root"; // MySQL user login username char password[] = "secret"; // MySQL user login password

Ahora estamos listos para iniciar la clase de Ethernet y hacer la conexión con el servidor de base de datos. El siguiente contiene el código completo para hacer esto en el setup()  method.  Explico cada línea siguiendo el ejemplo.

void setup() { Serial.begin(115200);

while (!Serial); // wait for serial port to connect

Ethernet.begin(mac\_addr); Serial.println("Connecting...");

if (conn.connect(server\_ip, 3306, user, password)) {

delay(1000);

// You would add your code here to run a query once on startup.

}

else

Serial.println("Connection failed.");

conn.close();

1  <https://en.wikipedia.org/wiki/MAC_address>

}

La primera línea inicia la clase serial. A continuación, iniciamos la clase Ethernet. Aquí se pasa en la dirección MAC que hemos especificado anteriormente. A continuación, se realiza un comando de impresión que indica que estaremos tratando de conectar. Tenga en cuenta que el uso de declaraciones de impresión - sin embargo la vieja escuela - es una forma válida para seguir el progreso de su dibujo si algo sale mal. Tenga en cuenta que estas cadenas se van a imprimir utilizando la memoria así que asegúrese de no abusar de ellos, sobre todo en pequeñas placas Arduino.

El siguiente constructo es una sentencia condicional cuando llamamos al método para conectarse al servidor. En este caso, el método es conectar () y toma los siguientes parámetros; la dirección IP del servidor (o el nombre de host), el puerto del servidor, nombre de usuario y contraseña. Si la conexión se realiza correctamente, el método devolverá un valor que se evalúa como "true" y se imprimirá el mensaje de éxito. En caso de que la conexión falla, el método devolverá un valor que se evalúa como "false" y el mensaje no se imprimirá.

A continuación se muestra un ejemplo de los estados producidos en el monitor mientras se ejecuta esta serie

bosquejo. Si estás siguiendo junto con su propia placa Arduino, debería ver algo similar.

Connecting...

Connected to server version 5.7.9-log

Query Success!

Tenga en cuenta que vemos los mensajes desde el boceto que indican una conexión exitosa. También vemos una respuesta de la biblioteca que imprime la versión del servidor al que se conecta. Esto puede ser útil en el diagnóstico de consultas erróneas más adelante. El Listado 1 muestra el dibujo terminado para su referencia. Siéntase libre de copiarlo sustituyendo sus datos específicos (dirección del servidor, etc.).

*Listado 1: Ejemplo de conexión de prueba - Ethernet*

/\*

MySQL Connector/Arduino Example : connect

\*/

#include <Ethernet.h>

#include <MySQL\_Connection.h>

byte mac\_addr[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

IPAddress server\_addr(10,0,1,35); // IP of the MySQL \*server\* here char user[] = "root"; // MySQL user login username

char password[] = "secret"; // MySQL user login password

EthernetClient client;

MySQL\_Connection conn((Client \*)&client);

void setup() { Serial.begin(115200);

while (!Serial); // wait for serial port to connect Ethernet.begin(mac\_addr); Serial.println("Connecting...");

if (conn.connect(server\_addr, 3306, user, password)) {

delay(1000);

// You would add your code here to run a query once on startup.

}

else

Serial.println("Connection failed.");

conn.close();

}

void loop() {

}

Observe la última línea de código en nuestro dibujo. El método close () se utiliza para desconectar del servidor y liberar cualquier memoria utilizada. Siempre es una buena idea llamar a este método para desconectarse del servidor de una manera limpia. Si va a dejar dormir a su boceto para un largo período de tiempo, se pueden utilizar los métodos connect () y close () dentro del bucle () para conectarse sólo en la medida que necesita para llevar a cabo operaciones de datos y luego se desconectan.

**Cómo realizar la conexión Wi-Fi Uso del Escudo**

Este ejemplo muestra la misma operación - Conexión sencilla - pero esta vez utilizando una shield2 WiFi. Para usar el escudo WiFi, sólo tiene que cambiar una cosa pequeña en su dibujo. Sólo es necesario incluir el archivo de cabecera para la biblioteca WiFi y una instancia de una clase para el cliente WiFi. Así es - usted ya no tiene que realizar cambios en los archivos de la biblioteca para utilizar el escudo WiFi. ¡Yupi!

#include <WiFi.h> // Use this for WiFi instead of Ethernet.h

...

WiFiClient client; // Use this for WiFi instead of EthernetClient

MySQL\_Connection conn((Client \*)&client);

Ahora tenemos que hacer algunos cambios en nuestro dibujo. Tenemos que especificar dos variables más; el SSID y la contraseña de la siguiente manera. Esto debe coincidir con la configuración de su puerto de acceso inalámbrico o router inalámbrico.

// WiFi card example

char ssid[] = "my\_lonely\_ssid";

char pass[] = "horse\_no\_name";

A continuación, tenemos que el código de configuración para detectar que el escudo Wi-Fi está activado y conectado. La mayoría de las veces esto no es un problema, pero si a su vez en el router y luego el fuego de la placa Arduino, el escudo WiFi puede no tener tiempo para inicializar correctamente. También es posible usar un retraso como lo he hecho. Aviso que no usamos el método Ethernet.begin() .

Serial.begin(115200);

while (!Serial); // wait for serial port to connect. Needed for Leonardo only

// Begin WiFi section

int status = WiFi.begin(ssid, pass);

if ( status != WL\_CONNECTED) { Serial.println("Couldn't get a wifi connection");

while(true);

}

// print out info about the connection:

else {

Serial.println("Connected to network"); IPAddress ip = WiFi.localIP();

2 [https://www.arduino.cc/en/Guide/ArduinoWiFiShield](http://www.arduino.cc/en/Guide/ArduinoWiFiShield)

Serial.print("My IP address is: "); Serial.println(ip);

}

// End WiFi section

Serial.println("Connecting...");

if (conn.connect(server\_addr, 3306, user, password)) {

delay(1000);

}

else

Serial.println("Connection failed.");

conn.close();

Aquí vemos el código para comprobar para ver si el WiFi está listo y si lo es, que recuperar la dirección IP asignada. Esto puede ser muy útil para determinar si hay un problema con subredes entre la placa Arduino y el servidor MySQL. A raíz de este código, conectamos como de costumbre. Un bosquejo completo se muestra en el Listado 2 a continuación.

*Listado 2: Prueba de conexión de la muestra - WiFi*

/\*

MySQL Connector/Arduino Example : connect by wifi

\*/

#include <WiFi.h> // Use this for WiFi instead of Ethernet.h

#include <MySQL\_Connection.h>

#include <MySQL\_Cursor.h>

byte mac\_addr[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

IPAddress server\_addr(10,0,1,35); // IP of the MySQL \*server\* here char user[] = "root"; // MySQL user login username

char password[] = "secret"; // MySQL user login password

// WiFi card example

char ssid[] = "horse\_pen"; // your SSID

char pass[] = "noname"; // your SSID Password

WiFiClient client; // Use this for WiFi instead of EthernetClient

MySQL\_Connection conn((Client \*)&client);

void setup() { Serial.begin(115200);

while (!Serial); // wait for serial port to connect. Needed for Leonardo only

// Begin WiFi section

int status = WiFi.begin(ssid, pass);

if ( status != WL\_CONNECTED) { Serial.println("Couldn't get a wifi connection");

while(true);

}

// print out info about the connection:

else {

Serial.println("Connected to network"); IPAddress ip = WiFi.localIP(); Serial.print("My IP address is: ");

Serial.println(ip);

}

// End WiFi section

Serial.println("Connecting...");

if (conn.connect(server\_addr, 3306, user, password)) {

delay(1000);

}

else

Serial.println("Connection failed.");

conn.close();

}

void loop() {

}

Si tiene problemas para obtener su escudo WiFi para trabajar, revise su nombre SSID y la contraseña para asegurarse de que está utilizando los valores correctos. Pruebe estos valores en otro equipo para probarlos. También debe consultar la documentación de su escudo WiFi ya que algunos escudos WiFi compatible con Arduino requieren un poco diferente código de inicio.

Ahora vamos a ver cómo lo hacemos una recopilación de datos sencilla mediante la adición de una consulta INSERT.

**Insertar Básico**

Este ejemplo demuestra cómo emitir una consulta a la base de datos. En este caso, se trata de un INSERT simple que registra la conexión con sólo insertar una fila en una tabla. Puede utilizar el ejemplo anterior como plantilla. Pero primero, tenemos que crear una base de datos de prueba y mesa. Emita los siguientes comandos en el servidor MySQL.

CREATE DATABASE test\_arduino;

CREATE TABLE test\_arduino.hello\_arduino ( num integer primary key auto\_increment, message char(40),

recorded timestamp

);

Estos comandos crear la base de datos y un test\_arduino hello\_arduino llamado sencilla tabla que tiene una columna de incremento automático, una cadena de texto, y una marca de tiempo. Desde las primeras y últimas columnas se generan de forma automática, es necesario suministrar solamente una cadena de texto.

Para ello, tenemos que utilizar una consulta SQL como la siguiente instrucción INSERT.

INSERT INTO test\_arduino.hello\_arduino (message) VALUES ('Hello, Arduino!');

Vaya por delante y abrir un cliente MySQL, conectar y poner a prueba esa consulta. A continuación, emita una consulta SELECT y ver los resultados. Deben ser similar a la siguiente. Si ejecuta el comando varias veces, verá varias filas del conjunto de resultados.

mysql> SELECT \* FROM test\_arduino.hello\_arduino;

+-----+-----------------+---------------------+

| num | message | recorded |

+-----+-----------------+---------------------+

| 1 | Hello, Arduino! | 2015-07-27 14:39:13 |

+-----+-----------------+---------------------+

1 row in set (0.00 sec)

Como se puede ver, cada vez que insertamos estos datos vamos a obtener una nueva fila en la tabla completa con una clave única (autogenerado) y una marca de tiempo en que se inserta la fila. ¡Guay! Ahora, vamos a añadir esto a nuestro dibujo.

Para ello, se añade una nueva variable de cadena para contener la consulta a continuación, utilizar la clase MySQL\_Cursor para ejecutar la consulta. Para utilizar el cursor, añadimos otra directiva include para incluir el archivo de cabecera del cursor, a continuación, asignar dinámicamente el objeto con una nueva operación, realizar la consulta, a continuación, utilizar una operación de eliminación para liberar el objeto y toda su memoria. A continuación se muestran los pasos en orden.

#include <MySQL\_Cursor.h>

...

MySQL\_Cursor \*cur\_mem = new MySQL\_Cursor(&conn);

cur\_mem->execute(INSERT\_SQL);

delete cur\_mem;

Observe llamar al método execute () para ejecutar la consulta. El Listado 3 muestra el dibujo terminado. Nótese que sólo sumó tres líneas de código y cambiamos el estado de una copia de impresión para aclarar el flujo (en negrita).

*Listado 3: Simple datos Insertar croquis*

/\*

MySQL Connector/Arduino Example : basic insert

\*/

#include <Ethernet.h>

#include <MySQL\_Connection.h>

#include <MySQL\_Cursor.h>

byte mac\_addr[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

IPAddress server\_addr(10,0,1,35); // IP of the MySQL \*server\* here char user[] = "root"; // MySQL user login username

char password[] = "secret"; // MySQL user login password

// Sample query

**char INSERT\_SQL[] = "INSERT INTO test\_arduino.hello\_arduino (message) VALUES ('Hello, Arduino!')";**

EthernetClient client;

MySQL\_Connection conn((Client \*)&client);

void setup() { Serial.begin(115200);

while (!Serial); // wait for serial port to connect Ethernet.begin(mac\_addr); Serial.println("Connecting...");

if (conn.connect(server\_addr, 3306, user, password)) {

delay(1000);

}

else

Serial.println("Connection failed.");

}

void loop() {

delay(2000);

Serial.println("Recording data.");

**// Initiate the query class instance**

**MySQL\_Cursor \*cur\_mem = new MySQL\_Cursor(&conn);**

**// Execute the query**

**cur\_mem->execute(INSERT\_SQL);**

**// Note: since there are no results, we do not need to read any data**

**// Deleting the cursor also frees up memory used delete cur\_mem;**

}

Nótese también que poner el código para ejecutar la consulta en el método de bucle (), lo que significa que va a ejecutar varias veces hasta que apague la placa Arduino. Esto se debe a que grandes bosquejos y más significativas que se insertan los datos periódicamente el código de registro de datos se ponen en el método () bucle.

Vaya por delante y ejecutar esta operación varias veces y luego emitir la consulta SELECT de nuevo. Ahora debería ver una fila

por cada vez que corrió el boceto (además de cómo cada vez muchos ensayos que hizo anteriormente). Ahora vamos a ver un inserto de datos más complejos con variables.

**Inserto Complejo**

El uso más frecuente de que el conector está grabando los datos recogidos por el Arduino. Esto podría ser un sensor (o varios), tales como la temperatura, la ocurrencia de enganche (puerta abierta / cerrada) botón, presiona, etc. Como tal, sólo tenemos que registrar los datos y seguir adelante. Sin embargo, es probable que sea algo leer o generado en lugar de una cadena estática de los datos en este caso.

Para insertar los datos que se genera (o lectura), hay que construir la cadena de consulta antes de emitir la consulta. Hacemos esto mediante el método de sprintf (). El siguiente ejemplo simula la lectura de un sensor. La consulta está todavía dentro del método de configuración (), ya que sólo queremos hacer esto una vez que una prueba.

Antes de ser, vamos a crear una nueva tabla que almacenará los resultados de un valor entero y el flotador leer. También vamos a mantener la cadena de texto para etiquetar la observación - en este caso un nodo sensor simulado. Dado que la mayoría de los sensores producen flotante - números de punto, incluyo un campo para demostrar cómo convertir flotante - números de punto.

CREATE TABLE test\_arduino.hello\_sensor ( num integer primary key auto\_increment, message char(40),

sensor\_num integer, value float, recorded timestamp

);

El siguiente se llama una cadena de formato utilizado por el método sprintf () para formar la cadena. Esto funciona mediante la sustitución de los valores de las variables para los caracteres especiales en la propia cadena de formato. Como puede suponer, estaremos construyendo una nueva cadena y por lo tanto vamos a asignar más memoria para esto. Como ya he eludido anteriormente, la más de estas cadenas especiales que debes construir, más memoria, es probable que consumen y por lo tanto si se utiliza una placa Arduino más pequeño debe ser mezquino con sus variables. La siguiente es la cadena de formato para este ejemplo.

INSERT INTO test\_arduino.hello\_sensor (message, sensor\_num, value) VALUES ('%s',%d,%s)

Note que tenemos tres variables aquí. El primero, un mensaje, es sólo una cadena que pasamos. El segundo es el

número de sensor (y entero). El último es un flotante - número de punto. Mientras que utilizamos un% s para significar una

cadena y un% d para indicar el número entero de sustitución, que tienen otra cadena% s para el alisado - valor en puntos. Esto se debe a que la biblioteca de Arduino no soporta actualmente la conversión flotantes - números de punto en sprintf (). Por lo tanto, debemos usar el dtostrf () 3 Método como se ilustra en el siguiente fragmento de código.

dtostrf(50.125, 1, 1, temperature);

sprintf(query, INSERT\_DATA, "test sensor", 24, temperature);

conn.execute(query);

Aquí estamos convirtiendo el valor de punto flotante de 50.125 en una cadena y almacenarla una temperatura variable llamada, que utilizamos más adelante en el sprintf () llamamos junto con nuestro mensaje (sensor de prueba) y el número de sensores (24). Por lo tanto, tenga en cuenta que la flotación - números de punto son un poco desordenado de tratar. La buena noticia es que este código funciona muy bien. Usted debe terminar con un resultado similar a la salida de abajo.

mysql> select \* from test\_arduino.hello\_sensor;

+-----+-------------+------------+-------+---------------------+

| num | message | sensor\_num | value | recorded |

+-----+-------------+------------+-------+---------------------+

| 1 | test sensor | 24 | 50.1 | 2015-07-27 15:12:38 |

+-----+-------------+------------+-------+---------------------+

1 row in set (0.00 sec)

Las variables que necesitamos para este bosquejo incluyen uno para un buffer para almacenar la consulta con formato, el formato de cadena de consulta, y un tampón para la temperatura. El Listado 4 muestra el esquema completo con las nuevas declaraciones en negrita.

*Listado 4: Complejo Insertar croquis*

/\*

MySQL Connector/Arduino Example : complex insert

\*/

#include <Ethernet.h>

#include <MySQL\_Connection.h>

#include <MySQL\_Cursor.h>

byte mac\_addr[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

IPAddress server\_addr(10,0,1,35); // IP of the MySQL \*server\* here char user[] = "root"; // MySQL user login username

3 [HYPERLINK "http://www.atmel.com/webdoc/AVRLibcReferenceManual/index.html"  HYPERLINK "http://www.atmel.com/webdoc/AVRLibcReferenceManual/index.html"http://www.atmel.com/webdoc/AVRLibcReferenceManual/index.htm](http://www.atmel.com/webdoc/AVRLibcReferenceManual/index.html)l (Search  for  dtostrf)

char password[] = "secret"; // MySQL user login password

// Sample query

**char INSERT\_DATA[] = "INSERT INTO test\_arduino.hello\_sensor (message, sensor\_num, value) VALUES ('%s',%d,%s)";**

**char query[128];**

**char temperature[10];**

EthernetClient client;

MySQL\_Connection conn((Client \*)&client);

void setup() { Serial.begin(115200);

while (!Serial); // wait for serial port to connect Ethernet.begin(mac\_addr); Serial.println("Connecting...");

if (conn.connect(server\_addr, 3306, user, password)) {

delay(1000);

**// Initiate the query class instance**

**MySQL\_Cursor \*cur\_mem = new MySQL\_Cursor(&conn);**

**// Save**

**dtostrf(50.125, 1, 1, temperature);**

**sprintf(query, INSERT\_DATA, "test sensor", 24, temperature);**

**// Execute the query cur\_mem->execute(query);**

**// Note: since there are no results, we do not need to read any data**

**// Deleting the cursor also frees up memory used delete cur\_mem;**

Serial.println("Data recorded.");

}

else

Serial.println("Connection failed.");

conn.close();

}

void loop() {

}

Usted podría hacer los amortiguadores dinámicos - y que probablemente sería una buena idea - sólo asegúrese de que siempre suelta la memoria después de que haya terminado de lo contrario se quede sin memoria rápidamente. También, asegúrese de que las variables o memoria asignada es lo suficientemente grande como para almacenar las cadenas de formato. El método sprintf () no se producirá un error y en su lugar se desbordará la memoria que puede causar todo tipo de dolor, así que asegúrese de comprobar su asignación de memoria (estática o dinámica)!

Como se puede ver, la parte más dura de la recogida de datos es la gestión de las memorias intermedias necesarias para las consultas y asegurarse de que suelte a la memoria asignada. Usted puede pensar que la selección de los datos sería un poco más fácil, y lo es, pero requiere un poco más de trabajo para hacer uso de los datos devueltos.

**Basic Select**

A veces es necesario para recuperar datos de su base de datos para su uso en el boceto. Tanto si

leyendo de una tabla de valores o la lectura de los resultados de otro proyecto Arduino, los datos necesarios

es probable que se utiliza en alguna forma de cálculo.

Este ejemplo muestra una consulta SELECT simple que recupera una fila de la base de datos y los almacena en una variable para su uso en el boceto. Al igual que los otros ejemplos, he hecho esto tan simple como sea posible colocando el código en el método setup ().

Antes de empezar, vamos a considerar lo que está sucediendo aquí. En primer lugar, nos estamos emitiendo una instrucción SELECT a la base de datos, que devolverá una o más filas (en función de la consulta). Pero antes de eso, la base de datos devolverá una lista de columnas y después de que una fila a la vez hasta que no haya más filas se dejan. Por lo tanto, hay que leer primero las columnas a continuación, una fila a la vez. Vamos a ver cómo hacer esto en el siguiente ejemplo a partir de la consulta.

**Nota**        *Esta consulta se emite contra la base de datos de ejemplo mundo. Puede descargar esta base de datos desde el siguiente enlace (*[*http://dev.mysql.com/doc/index*](http://dev.mysql.com/doc/index)*- other.html). Para ejecutar este esquema, tendrá que descargar el archivo, descomprimirlo, y siga las instrucciones del sitio web para instalarlo. Una vez que instale la base de datos mundial, puede ejecutar la consulta en un cliente MySQL.*

mysql> SELECT population FROM world.city WHERE name = 'New York';

+------------+

| population |

+------------+

| 8008278 |

+------------+

1 row in set (0.01 sec)

Note que hay una fila devuelta. El siguiente es el código que necesitamos para leer este valor.

MySQL\_Cursor \*cur\_mem = new MySQL\_Cursor(&conn);

// Execute the query cur\_mem->

execute(query);

// Obtener las columnas (requerido) pero no usamosthem. column\_names \*columns = cur\_mem->get\_columns();

// Leer la fila (estamos esperando sólo el uno)

do {

row = cur\_mem->get\_next\_row();

if (row != NULL) {

head\_count = atol(row->values[0]);

}

} while (row != NULL);

// Eliminar el cursor también libera la memoria

used delete cur\_mem;

// Show the result Serial.print(" NYC pop = "); Serial.println(head\_count);

Note que primero ejecutar la consulta a continuación, lea las columnas. Esta es una función especial en la biblioteca. Si desea leer los nombres de las columnas, se puede, pero que es una operación rara vez se utiliza. Vamos a ver cómo se hace esto en el siguiente ejemplo.

A continuación, leemos las filas de uno en uno. Ya que sabemos que la consulta devuelve una sola fila, puede tener la tentación de codificar sólo la llamada de un get\_next\_row (), pero no lo hace. Mientras que sólo vemos la fila en el resultado, hay un reconocimiento o un paquete de salida después de la última fila de leer y por lo tanto es preciso identificar el bucle incluso si sólo hay una fila devuelta.

Una vez que se lee la fila, se utiliza el método de atol () para guardar el valor leído de la fila de la primera columna (comienza a contar a 0). Se puede utilizar la variable de fila para hacer referencia a cualquier columna que necesita si la fila devuelve más de una sola columna. Pero hay que tener cuidado porque los más columnas devueltos, se consumirá más memoria. Eso es por lo que especificamos una columna en la consulta - para ahorrar espacio y solicitud únicamente los datos necesarios y nada más. Usted debe adoptar esta práctica la miseria al escribir bocetos.

Por último, imprimir los resultados nos leen desde la fila. El Listado 5 muestra el dibujo terminado. Inténtelo usted mismo para asegurarse de que obtiene el mismo valor de la base de datos. Una vez más, las nuevas líneas de código se destacan en negrita.

*El Listado 5: Bosquejo simple Select*

/\*

MySQL Connector/Arduino Example : basic select

\*/

#include <Ethernet.h>

#include <MySQL\_Connection.h>

#include <MySQL\_Cursor.h>

byte mac\_addr[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

IPAddress server\_addr(10,0,1,35); // IP of the MySQL \*server\* here char user[] = "root"; // MySQL user login username

char password[] = "secret"; // MySQL user login password

// Ejemplo de consulta

**char query[] = "SELECT population FROM world.city WHERE name = 'New York'";**

EthernetClient client;

MySQL\_Connection conn((Client \*)&client);

// Crear una instancia del cursor que pasa en la conexión

MySQL\_Cursor cur = MySQL\_Cursor(&conn);

void setup() { Serial.begin(115200);

while (!Serial); // wait for serial port to connect Ethernet.begin(mac\_addr); Serial.println("Connecting...");

if (conn.connect(server\_addr, 3306, user, password)) {

delay(1000);

}

else

Serial.println("Connection failed.");

}

void loop() {

**row\_values \*row = NULL;**

**long head\_count = 0;**

delay(1000);

Serial.println("1) Demonstrating using a cursor dynamically allocated.");

**// Iniciar la instancia de clase de consulta**

**MySQL\_Cursor \*cur\_mem = new MySQL\_Cursor(&conn);**

**// Execute the query cur\_mem->execute(query);**

**// Fetch the columns (required) but we don't use them.**

**column\_names \*columns = cur\_mem->get\_columns();**

**// Leer la fila (estamos esperando sólo el uno)**

**do {**

**row = cur\_mem->get\_next\_row();**

**if (row != NULL) {**

**head\_count = atol(row->values[0]);**

**}**

**} while (row != NULL);**

**// Deleting the cursor also frees up memory used delete cur\_mem;**

**// Show the result Serial.print(" NYC pop = "); Serial.println(head\_count);**

}

El siguiente ejemplo combina la necesidad de pasar de las variables a la consulta SELECT para recuperar datos basados

en la información dinámica.

**Seleccionar Complejo**

Este ejemplo muestra cómo utilizar una consulta SELECT con una cláusula WHERE formada a partir de un cálculo. En este caso, se simula el cálculo con el uso de un número arbitrario. Sin embargo, sólo tiene que sustituir a la lógica con la lectura de la entrada del usuario, un sensor, otro Arduino, cálculos en su dibujo, etc.

Seguimos usando la base de datos mundial, pero en este caso queremos seleccionar aquellos países con una específica

población (es decir, mayor que un valor específico proporcionado). La consulta que queremos utilizar es el siguiente.

SELECT name, population FROM world.city

WHERE population > 9000000

ORDER BY population DESC;

Hay mucho que hacer aquí! Aviso no solo especificamos la población, sino que también ordenar los resultados por la población. Por lo tanto, vamos a ver cómo navegar resultado múltiples conjunto de filas, así como ver cómo imprimir la columna de nombres devueltos.

Observe el tamaño de la variable que desea establecer para la cláusula WHERE. Aquí vamos a utilizar otra llamada sprintf () para dar formato a la cadena. En este caso, necesitamos un entero largo así utilizamos% lu (unsigned long).

El Listado 6 muestra el código necesario para leer las columnas, imprimirlos, a continuación, leer las filas y pantalla

los valores con el código pertinente en negrita.

*Listado 6: Complejo Seleccione Boceto*

/\*

MySQL Connector/Arduino Example : complex select

\*/

#include <Ethernet.h>

#include <MySQL\_Connection.h>

#include <MySQL\_Cursor.h>

byte mac\_addr[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

IPAddress server\_addr(10,0,1,35); // IP of the MySQL \*server\* here char user[] = "root"; // MySQL user login username

char password[] = "secret"; // MySQL user login password

**// Ejemplo de consulta**

**//**

**// Note la "% lu" - que es un marcador de posición para el parámetro que se**

**// Suministro. Ver sprintf () documentación para obtener más especificador de formato**

**// opciones**

**const char QUERY\_POP[] = "SELECT name, population FROM world.city WHERE**

**population > %lu ORDER BY population DESC;";**

**char query[128];**

EthernetClient client;

MySQL\_Connection conn((Client \*)&client);

void setup() { Serial.begin(115200);

while (!Serial); // wait for serial port to connect Ethernet.begin(mac\_addr); Serial.println("Connecting...");

if (conn.connect(server\_addr, 3306, user, password)) {

delay(1000);

}

else

Serial.println("Connection failed.");

}

void loop() {

delay(1000);

Serial.println("> Running SELECT with dynamically supplied parameter");

**// Iniciar la instancia de clase de consulta**

**MySQL\_Cursor \*cur\_mem = new MySQL\_Cursor(&conn);**

**// Proporcionar el parámetro para la consulta**

**// Aquí usamos la QUERY\_POP como la cadena de formato y de consulta como el**

**// Destino. Esto utiliza el doble de memoria así que otra opción sería**

**// Asignar una memoria intermedia para todas las consultas a formato o asignar el**

**// Memoria según sea necesario (sólo asegúrese de asignar suficiente memoria y**

**// Liberarla cuando haya terminado!).**

**sprintf(query, QUERY\_POP, 9000000);**

**// Ejecutar la consulta**

**cur\_mem->execute(query);**

**// Obtener las columnas e imprimirlas**

**column\_names \*cols = cur\_mem->get\_columns();**

**for (int f = 0; f < cols->num\_fields; f++) { Serial.print(cols->fields[f]->name);**

**if (f < cols->num\_fields-1) {**

**Serial.print(',');**

**}**

**}**

**Serial.println();**

**// Lea las filas y de impresión**

**them row\_values \*row = NULL;**

**do {**

**row = cur\_mem->get\_next\_row();**

**if (row != NULL) {**

**for (int f = 0; f < cols->num\_fields; f++) { Serial.print(row->values[f]);**

**if (f < cols->num\_fields-1) {**

**Serial.print(',');**

**}**

**}**

**Serial.println();**

**}**

}

**} while (row != NULL);**

**// Eliminar el cursor también libera la memoria**

**used delete cur\_mem;**

Observe cómo el código está escrito para recorrer las columnas primera a continuación, los valores para cada fila. Ahora que hemos visto algunos ejemplos de bocetos y usos comunes de la biblioteca, la siguiente sección se describen algunos consejos y técnicas para escribir sus bocetos para interactuar con MySQL.

**Consejos para escribir bosquejos con el conector**

Esta sección contiene una lista de sugerencias para tomar mejores bocetos con el conector. En algunos casos esto es un consejo y en otros casos se sugiere código o técnicas. Si el boceto incluirá consultas más complejas que las que se muestra arriba, usted debe leer esta sección para la incorporación en sus propios bocetos.

**Utilice los ejemplos**

Hay muchos bocetos ejemplo que se incluye con el conector. Debe ejecutar uno o más de estos para así saber cómo funciona el conector antes de escribir su propio dibujo. Yo recomiendo empezar con los ejemplos de conexión, basic\_insert, y basic\_select primero. Conoce a estos y ponerlos a prueba para asegurar su servidor MySQL está configurado correctamente y la placa Arduino puede conectarse a él. Si tiene problemas con estos ejemplos, no culpes al conector (al menos no inicialmente). Lea la sección de solución de problemas para resolver uno o más de los problemas comunes y probar su ejemplo de nuevo. No se olvide de cambiar la dirección IP, nombre de usuario y contraseña!

**Mantenlo simple**

Éste me siento es un hecho para escribir código para los microprocesadores, pero puede ser sorprendido por el número de peticiones que he tenido para ayudar a resolver los problemas. La causa raíz o el factor importante para gran parte de los problemas de los usuarios tallos en torno a hacer el boceto mucho más compleja de lo que debe ser.

Esto es especialmente cierto para aquellos que escribir la totalidad de su solución antes de probarlo. Es decir, que escriben cientos de líneas de código, que se compile (a veces no tanto) y luego tratar de ejecutarlo. En este caso, el usuario ha podido darse cuenta de todos los aspectos de su solución debe ser probado en un paso - de manera sabia.

Por ejemplo, escribir el boceto para hacer los pasos necesarios para demostrar minimalistas (prueba) cada parte. Para trabajar con la base de datos MySQL, comenzar con una prueba de conexión sencilla a continuación, proceder a las pruebas de todos y cada consulta utilizando datos ficticios o valores simulados.

Del mismo modo, se trabaja con sensores u otros dispositivos se debe hacer de manera aislada para que pueda eliminar

grandes porciones del boceto para la investigación si algo sale mal.

Si usted adopta esta filosofía, sus bocetos serán más fáciles de escribir y que tendrán mucho más éxito

que el "código de una vez y rezar funciona" filosofía.

**Connectar/Cerrar**

La mayoría de los bocetos se escriben para conectar una vez en el inicio. Sin embargo, para soluciones complejas que recogen o interactúan con la base de datos, la conexión es crítica para proyectos de uso más larga. A menudo es el caso que las redes pueden no ser fiable. De hecho, no hay nada en la especificación de los protocolos o equipos de redes que sugiera que está siempre sin pérdidas. De hecho, la red está diseñada para ser "mayoría" fiable con alguna pérdida aceptable.

Cuando se produce la pérdida, a veces puede causar errores en el conector cuando se lee de la base de datos o puede hacer que el escudo de Ethernet a abandonar su conexión. En casos extremos, puede causar el boceto para colgar o bucle fuera de control (dependiendo de cómo las sentencias condicionales se escriben).

Para combatir esto, se puede utilizar una técnica mediante la cual nos conectamos y cerca de cada paso a través del bucle. Esto funciona, pero no es una solución más elegante que le permite volver a conectar cada vez que se interrumpe la conexión. Lo siguiente demuestra este concepto.

void loop() {

delay(1000);

if (conn.connected()) {

// do something

} else { conn.close(); Serial.println("Connecting...");

if (conn.connect(server\_addr, 3306, user, password)) {

delay(500);

Serial.println("Successful reconnect!");

} else {

Serial.println("Cannot reconnect! Drat.");

}

}

}

Nótese aquí, comprobamos el estado del conector y si no está conectado, volvemos a conectar. Esta voluntad

salvarte de los casos en que se interrumpe la conexión a la red de bases de datos o errores.

**Reinicio (Fix)**

En estrecha relación con la técnica de conexión / cierre es una técnica para reiniciar el Arduino debería ocurrir algo malo. Esto puede ser muy útil si usted tiene un proyecto que debe trabajar, pero está bien si hay lagunas de datos cortos. Por ejemplo, si está supervisando algo y realizar cálculos es posible su hardware podría tener problemas de periódicos, así como los errores de lógica o fallos de redes simples.

Para superar estas situaciones, se puede programar el Arduino para reiniciar usando el siguiente código. Tenga en cuenta que esto demuestra que esta técnica se utiliza con la opción de conexión / cierre, ya que son de cortesía. Después de todo, si no se puede conectar después intenta N, un reinicio puede no doler y en la mayoría de los casos en los que es un problema con la memoria o el escudo de Ethernet o relacionado, funciona.

void soft\_reset() {

asm volatile("jmp 0");

}

void loop() {

delay(1000);

if (conn.connected()) {

// do something num\_fails = 0;

} else { conn.close(); Serial.println("Connecting...");

if (conn.connect(server\_addr, 3306, user, password)) {

delay(500);

Serial.println("Successful reconnect!");

num\_fails++;

if (num\_fails == MAX\_FAILED\_CONNECTS) {

Serial.println("Ok, that's it. I'm outta here. Rebooting...");

delay(2000);

soft\_reset();

}

}

}

}

Nótese aquí usamos una llamada ensamblador para saltar a la posición 0. Esto se reinicia de manera efectiva el microcódigo Arduino. Fresco, eh? Y usted pensó que tendría que sudar tinta a la granja de cerdos y pulse el botón de reinicio wee poco.

**Comprobador de memoria**

Otra técnica útil está monitoreando o diagnosticar problemas de memoria mediante el cálculo de la cantidad de

la memoria se queda. Hacemos esto con el siguiente método.

int get\_free\_memory()

{

extern char bss\_end; extern char \* brkval; int free\_memory; if((int) brkval == 0)

free\_memory = ((int)&free\_memory) - ((int)& bss\_end);

else

free\_memory = ((int)&free\_memory) - ((int) brkval);

return free\_memory;

}

Puede utilizar este método en cualquier parte del código. Me gusta usarlo con una sentencia de impresión para imprimir el

el valor calculado de la siguiente manera.

Serial.print(" RAM: "); Serial.println(get\_free\_memory());

La colocación de este código estratégicamente en el boceto y ver los resultados en el monitor serie puede ayudar

pérdidas de memoria le mancha y situaciones en las que se quede sin memoria.

**¡Haz tu tarea! (Do Your Homework!)**

Es en este punto que me gustaría aclarar una cosa sobre el uso de las bibliotecas, como el conector. Este es un consejo para todos los que están aprendiendo a programar el Arduino. Asegúrese de hacer su tarea y su propia investigación antes de hacer preguntas. Tantas veces me quedan preguntas sobre el más básico

cosas (bueno, básicos a la experiencia) que no tienen nada que ver con el conector. Por ejemplo,

trabajar con la memoria, las variables y las cadenas parecen ser piedras de tropiezo para los nuevos usuarios.

Al final, usted recibirá ayuda mucho más útil a partir de la biblioteca y otros autores experimentados Arduinistas

si se toma el tiempo de leer un libro, página web, o escuchar un podcast antes de contactar con el autor para solicitar ayuda o quejarse de un error de compilación. Una pequeña cantidad de aprendizaje de su parte va a cosechar los dividendos cuando se puede hacer una pregunta específica o buscar ayuda para un problema complejo.

Un ejemplo de ello es este documento. Desde mi experiencia, este documento es mucho más detallada que cualquier otra biblioteca disponible para el Arduino (con notables excepciones). Parte de la motivación para escribir este documento fue consolidar la información sobre el conector y para garantizar los nuevos en el uso del conector tenían un tutorial suficientemente detallada. En la siguiente sección se completa el cuerpo de información acerca del conector mediante la presentación de las preguntas más comunes de los usuarios.

**Solución de problemas**

En esta sección se presenta una breve pero demostrado la práctica para solucionar bocetos que usan el conector. En caso de tener una situación en la que el boceto falla o no funciona cuando modificado o trasladado a otra red, desplegado, etc., siguiendo este proceso puede ayudar a aislar el problema.

1. Verificar la red. Intente conectar otro ordenador en lugar de la Arduino para asegurarse de que puede conectarse a la red y la base de datos de Sever. Corregir cualquier problema de red antes de continuar.

2. Verificar su cuenta de usuario. Con el mismo equipo, intente iniciar sesión en la base de datos utilizando las credenciales en su dibujo. Corregir cualquier problema con los permisos y cuentas de usuario antes de continuar.

3. Compruebe los permisos. Si reinicia el Arduino y todavía no se puede conectar, volver atrás y comprobar

sus permisos de nuevo.

4. Compruebe el hardware de red. Asegúrese de que no hay servidores de seguridad, escáneres de puertos, etc., que son

bloqueando el acceso al servidor de base de datos.

5. Aislar el código. Una vez que se resuelven todos los problemas de conexión, compruebe el código. Lo más probable es

puedes comentar o eliminar la mayor parte del código para comprobar solamente las partes mínimo. YO

recomendar romper el código en secciones y las pruebas de cada hasta que se encuentra con la sección

con el problema.

6. Comprobar el hardware. Cuando todo lo demás falla, pruebe con otro Arduino. He visto casos en los que un Arduino se rompe o tiene un corto o alguna otra falla en los que puede arrancar y ejecutar simples bocetos, pero nada más que eso falla.

**Preguntas Frecuentes**

La siguiente es una lista de preguntas que se han planteado en numerosas ocasiones en los foros. Se dirigen a un montón de errores comunes y explican algunas nuevas técnicas no tratados anteriormente. Asegúrese de escanear esta lista antes de hacer nuevas investigaciones en los foros. La siguiente se enumeran en ningún orden en particular.

***¿Puedo usar el conector para conectarse a otros servidores de bases de datos?***

No. El conector sólo funciona con el servidor MySQL.

***¿Puedo utilizar el conector con la no - módulos Ethernet compatibles Arduino?***

El conector sólo funciona con escudos Ethernet y módulos que soportan Ethernet Arduino

clase. Si el módulo requiere una nueva clase de Ethernet, que no funcionará con el conector.

***Mi dibujo es de bloqueo hacia arriba. ¿Qué debo hacer?***

El problema puede ser una de varias cosas, pero la causa más probable es que se quede sin memoria o caídas de la red. Comprobar el uso de memoria para asegurarse de que tiene suficiente memoria. Se puede cambiar a una gran Arduino si su boceto crece más su tablero. Para los problemas de red, puede utilizar las técnicas de conexión / cerrar o reiniciar el sistema anterior.

**Estoy recibiendo "múltiples definición de` Conector :: check\_ok\_packet () ' "y los errores de compilación similares. *¿Qué ocurre?***

*Si usted está viendo los errores de compilación sobre las funciones duplicadas y similares, es porque tiene el código del conector en más de un lugar. Es decir, que ha duplicado el código en sus bibliotecas o carpeta de croquis. En máquinas Windows, esto es posible si se copia el archivo a una carpeta temporal o descomprimirlo en varias ubicaciones. Existen estar seguro de una sola copia del mysql. \* archivos en la carpeta Bibliotecas.*

***Me estoy poniendo "error: '' COLUMN\_NAMES no estaba declarado en este ámbito" y los errores de compilación similares. ¿Qué ocurre?***

Debe habilitar WITH\_SELECT en mysql.h para permitir a los métodos para el procesamiento de conjuntos de resultados de consultas de selección ().

***Estoy recibiendo errores de compilación en las bibliotecas SHA1. ¿Qué ocurre?***

Si utilizó una versión más antigua del conector y actualizado recientemente, es posible que su carpeta es SHA1

fuera de plazo. Asegúrese de copiar la carpeta última SHA1 del archivo .zip y reiniciar el IDE.

Recibo mensajes de error del compilador cuando se trata de hacer una consulta con variables.

Asegúrese de sprintf () y dtostrf () para dar formato a la consulta con variables. El código no hace

apoyar la sustitución de variables.

***¿Cuál es el 3306 en el código de ejemplo?***

Es el puerto en el que escucha el servidor MySQL. Puede especificar otro puerto, pero su MySQL

servidor debe ser configurado para escuchar en el puerto. 3306 es el valor predeterminado.

Mis consultas no están funcionando!

Debe probar las consultas utilizando el cliente de MySQL antes de intentar ejecutar el boceto. Muchas veces

hay pequeños errores de sintaxis que debe corregir antes de la consulta va a funcionar.

***¿Por qué no se seleccione consultas habilitadas por defecto?***

I inhabilitó a propósito del código para procesar conjuntos de resultados para ahorrar unos pocos bytes. Es decir, si su dibujo (como

la mayoría) son sólo la inserción de datos, que no necesita el código extra ocupando la memoria valiosa.

Me pone "Error de conexión".

Si usted está recibiendo un mensaje fallido de conexión (como está escrito en el boceto), lo más probable es que su Arduino no está conectado a la red correctamente o su cuenta de usuario y la contraseña no es correcta o si el usuario no tiene permisos para conectarse. Use un segundo equipo y las credenciales de su boceto para comprobar que puede conectarse. Resolver cualquier problema y vuelva a intentar el boceto.

***Me sale el error, "Conector no nombra a un tipo". ¿Qué ocurre?***

El escenario más probable es que no se ha colocado el conector en su carpeta de Arduino bibliotecas o haber cambiado el nombre o se lo colocó en otra carpeta. Asegúrese de que esté instalado correctamente y reiniciar el IDE.

***¿Puedo asignar una dirección IP a la placa Arduino?***

Sí, utilice uno de los conjunto alternativo de parámetros para la clase de Ethernet para configurar la IP manualmente. Ver

<https://www.arduino.cc/en/Reference/EthernetBegin>.

***¿Puedo utilizar un nombre de host en lugar de una dirección IP para el servidor?***

Sí, pero requiere el uso de la biblioteca DNS de la siguiente manera.

# include <Dns.h>

...

Char nombre de host [] = "[www.google.com](http://www.google.com/)"; // el cambio de nombre de host de su servidor / URL

...

ip\_servidor DirecciónIP; dns DNSClient; dns.begin (Ethernet.dnsServerIP ()); dns.getHostByName (nombre de host, ip\_servidor); Serial.println (ip\_servidor) Serial.println ( "Conectando ...");

si (conn.connect (ip\_servidor, 3306, usuario, contraseña)) {

...

***¿Puedo utilizar más de una consulta?***

Sí, sólo asegúrese de que tiene suficiente memoria para las cadenas.

***Consigo PACKET\_ERROR. ¿Que es eso?***

Este error se produce cuando el conector recibe la cabecera del paquete equivocado o una respuesta inesperada del servidor. Ocurre con más frecuencia cuando se utilizan las consultas de selección en los que hay más filas que no se leen. Ver los ejemplos anteriores para asegurarse de que está procesando el conjunto de resultados. También puede utilizar una cláusula WHERE o LIMIT para ayudar a restringir el número de filas devueltas.

Veo caracteres extraños en el monitor serie.

Asegúrese de que la velocidad de transmisión del monitor serie coincide con su dibujo. Cambiar una o la otra

para que coincida y debería ver los caracteres válidos.

***Consigo conexión fallida. ¿Qué podría estar mal?***

Tiene uno o más de los siguientes incorrecto:

• dirección del servidor

• uso de IP estática (prueba de DHCP)

• la conexión de red no es viable o detrás de un interruptor

• las credenciales de usuario no funcionan

Su mejor diagnóstico es el uso de un segundo equipo en la misma línea Ethernet con las mismas credenciales (dirección del servidor, usuario, contraseña) e intente conectarse. Si puede, entonces puede que tenga un problema con su hardware.

***Todavía no puedo conseguir la conexión con el trabajo, lo que más puedo intentar?***

Debe utilizar uno de los ejemplos que vienen con el IDE de Arduino como el boceto de cliente Web. Prueba esto y si funciona, usted sabe que su escudo de Ethernet está funcionando. Puede hacer lo mismo con el escudo WiFi. Una vez que verifique las obras de escudo, volver atrás y comprobar su servidor MySQL y prueba de conectarse a ella desde otro equipo hasta que las credenciales y permisos son correctos.

Estoy utilizando un segundo equipo, pero todavía no puedo acceder a la base de datos.

Las causas principales son:

• la dirección IP del servidor ha cambiado

• hay un bloqueo de conexiones entrantes en el servidor de seguridad 3306

• el / router / switch puerto de red no funciona

• los permisos de usuario y host no son correctas (No se puede iniciar sesión)

***¿Cómo puedo encontrar mi dirección IP del servidor MySQL?***

Hay muchas maneras. Si está ejecutando Linux, Unix o Mac OS X, utilice la siguiente:

ifconfig

Para Windows usar esto:

ipconfig

Va a encontrar la dirección IP en la salida de estos comandos. También puede hacer esto en un cliente mysql:

mostrar variables como el 'nombre de host';

A continuación, utilice ping (desde un terminal) para hacer ping al nombre de host se muestra. La salida se mostrará la dirección IP.

Funciona el conector con los módulos GPRS? No. Sólo el Arduino Ethernet o WiFi escudos. ¿Cómo puedo grabar la fecha y hora de mi evento?

Utilice una columna de marca de tiempo en su mesa. Esto se actualiza con la hora y la fecha cuando la corriente

se inserta la fila.

***¿Cómo uso PROGMEM para almacenar cadenas?***

Incluir el encabezado espacio de memoria de programa declarará la cadena con la palabra clave tal como se muestra. Recuerde usar el segundo parámetro opcional en el método execute () al pasar en estas cadenas para las consultas.

#include <avr / pgmspace.h>

...

const consulta PROGMEM char [] = "SELECT nombre, población de la world.city";

...

conn.Execute (consulta, true);

***¿Puedo utilizar el nuevo WiFi Escudo 101?***

Sí. Hay un ejemplo de cómo usar la nueva WiFi Escudo 101. Véase el Archivo - Ejemplos> -> menú de MySQL Connector Arduino.

***Cam utilizo la Ethernet Shield 2?***

Si y no. Sí, el conector trabajará con el nuevo escudo, pero tendrá que hacer un menor de edad

cambio en el archivo MySQL\_Packet.h. Abra el archivo MySQL\_Packet.h y el cambio:

# include <Ethernet.h>

    a:

# include <Ethernet2.h>

Y no es porque no se puede utilizar el nuevo escudo (en la actualidad) con el IDE de Arduino de arduino.cc. Debe descargar el software arduino.org, no el software de arduino.cc. Sí, hay una diferencia. No voy a entrar en eso aquí, pero basta con decir que hay diferencias. Para descargar el IDE, vaya a <http://www.arduino.org/software>. Se puede ejecutar junto a otra versión, sólo asegúrese de instalarlo en otra ubicación. Una vez instalado, puede compilar su boceto pero primero cambiar las directivas de inclusión a la lista a continuación.

#include <SPI.h> // <---- Add this include

#include <Ethernet2.h> // <---- Change to use the new library :)

**Limitaciones**

Dada la plataforma de destino - un pequeño microcontrolador con memoria limitada - hay algunas limitaciones en el uso de una biblioteca compleja en la plataforma Arduino. La primera cosa que usted debe saber sobre el conector es que no es una pequeña biblioteca y puede consumir una gran cantidad de memoria. Mientras que la biblioteca utiliza la memoria dinámica para mantener el uso de memoria al mínimo, la cantidad de memoria utilizada depende de cómo se utilice el conector.

Más específicamente, tendrá que limitar la cantidad de constantes de cadena que crea. Si emite órdenes de inserción de datos simples (INSERT INTO), una forma sencilla de calcular esto es el conector utiliza un poco más que el máximo del tamaño de la cadena de consulta más larga, además de la suma de todas sus cadenas. Si se está consultando el servidor de datos, el conector utiliza un poco más que el tamaño acumulado de una fila de datos devueltos.

Hay otras limitaciones a tener en cuenta, pero la más notable es el uso de memoria. Si está utilizando la última Arduino Debido esto puede no ser un problema. Pero hay otras consideraciones. A continuación se enumeran las limitaciones conocidas del conector / Arduino.

• Las cadenas de consulta (SQL) los estados deben caber en la memoria. Esto es porque la clase utiliza un buffer interno para la construcción de paquetes de datos para enviar al servidor. Se sugiere cadenas largas pueden almacenar en la memoria de programa usando PROGMEM.

• Los conjuntos de resultados se leen una fila - en - un - tiempo y un campo - en - un - tiempo.

• La longitud combinada de una fila en un conjunto de resultados debe caber en la memoria. El conector lee un paquete - a - a - tiempo y desde el Arduino tiene un tamaño limitado de datos, la longitud combinada de todos los campos debe ser menor que la memoria disponible.

• respuestas de error del servidor se procesan inmediatamente con el código de error y el texto escrito a través

Serial.print.

**Cambios con respecto a versiones anteriores**

Esta sección describe los cambios de una versión a otra que tendrán los desarrolladores saber con el fin de convertir el código existente para utilizar la nueva versión. Si bien se introducen por lo general no hay grandes cambios durante un ciclo de lanzamiento de la versión mayor.menor, es se harán cambios probables cuando se incrementa la versión de mayor o menor.

**Versión 1.0.4 -> 1.1.X**

El tema para la versión 1.1.X era para dar un salto adelante para hacer que el conector más fácil de usar y para adaptarse a las nuevas directrices para la escritura de las bibliotecas para el Arduino. Como tal, muchos de los nombres de los métodos cambiado, así como se han añadido nuevas clases para ayudar a mejorar la usabilidad. A continuación se enumeran los desarrolladores principales cambios necesitan saber con el fin de adaptar la nueva versión.•

**Nueva clase de conexión**: Se añadió una nueva clase **MySQL\_Connection**. Esta clase hereda de una clase **MySQL\_Packet**, que contiene todo el código de manejo de paquetes. Por lo tanto, la nueva clase de conexión es más pequeño con sólo unos pocos métodos por lo que es más fácil de usar. Esta clase requiere una instancia de una clase de cliente compatible con el Arduino bibliotecas Wi-Fi o Ethernet. Como beneficio adicional, el conector puede ahora ser usado con cualquier clase que implementa la misma

métodos como el método original de cliente Ethernet. Por ejemplo, si usted compró un escudo de Ethernet más reciente que utiliza un nuevo conjunto de chips (como el de Seeedstudio), se puede utilizar con el conector, ya que la clase base para la nueva biblioteca Ethernet2 es la misma que la biblioteca cliente Ethernet. Basta con incluir la nueva clase e inicializar el conector con una nueva instancia del cliente.

• **Nueva Clase Cursor**: Se añadió una nueva clase **MySQL\_Cursor**. Esta clase permite a los usuarios

ejecutar consultas. Se hizo una clase separada principalmente para eliminar la compilación condicional, sino también

para simplificar la gestión de la memoria.

• **Nuevos Ejemplos**: Los ejemplos de código originales se han reescrito para que se corresponda con los nuevos ejemplos de documentación. También hay una serie de nuevos ejemplos para ayudar a los usuarios a familiarizarse con mayor rapidez.

• **Manejo simplificado de la memoria**: El código original requería que la persona que llama para gestionar la memoria asignada por el conector. Con la nueva versión, los usuarios no tienen que incluir los métodos de memoria libres, que ahora se manejan internamente por la clase de conector y el cursor.

• **Los métodos han cambiado de nombre**: Con el fin de ajustarse a las bibliotecas de conectores MySQL más tradicionales, se cambió el nombre de varios métodos. A continuación se resumen los nuevos nombres. Algunas funciones de menor importancia es ligeramente diferente, como se muestra.

|  |
| --- |
| Old  Method   Class   New  Method |
| mysql\_connect() MySQL\_Connection connect() |
| disconnect() MySQL\_Connection close() |
| is\_connected() MySQL\_Connection connected() |
| cmd\_query() MySQL\_Cursor execute()  ex: execute(query); |
| cmd\_query\_P() MySQL\_Cursor execute()  ex: execute(query, true); |
| free\_\*() MySQL\_Cursor close() |

**Para más información**

Hay una configuración foro para responder a preguntas sobre el conector, que incluye preguntas sobre el uso

y los problemas que lo usan (<http://forums.mysql.com/list.php?175>).

También puede responder a mis blogs (<http://drcharlesbell.blogspot.com/>), pero tenga en

mente algunas de estas entradas están recibiendo bastante largo y muchos repiten las mismas preguntas una y otra vez.

Así que antes de hacer su pregunta, asegúrese de que ha leído este documento en su totalidad (en especial el FAQ)

antes de presentar una nueva pregunta. Es probable que, otros han visto su problema y una solución

ya existe.

Voy a aceptar peticiones especiales enviadas por correo electrónico a mí directamente en drcharlesbell@gmail.com o chuck.bell@oracle.com, pero se reserva el derecho de retrasar mi respuesta hasta que el tiempo lo permite. Por lo tanto, no se espera una respuesta inmediata (pero a veces voy a responder dentro de 24 horas).