aardio 范例: mysql.client 库 - 入门

```
//mysql.client 库 - 入门
import console;
import mysql.client;

/*
MySQL 快速入门
https://learnxinyminutes.com/docs/zh-cn/sql-cn/
https://quickref.me/mysql
*/
console.showLoading(" 正在连接测试数据库" )
var dbClient,err = mysql.client(
    server = "db4free.net"; //服务器，格式为"域名或IP:端口号"或"域名或IP"，省略则默认为 "localhost"。
    uid = "aardio_mysql";//用户名，省略则默认为"root"。
    pwd = "aardio.com";//密码。
    database = "aardio_mysql"; //可选指定数据库
);

if(!dbClient){
    console.log("如果是有人无聊修改了密码,请自行到 db4free.net 申请免费数据库")
    return console.logPause(err);
}

//选择数据库
//dbClient.selectDb("aardio_mysql")

//查询数据并返回记录集
var result = dbClient.stepQuery("select version()");
console.dump(result)

/*
dbClient.query(sql,...) 等所有需要格式化SQL语句的函数，
内部都是调用 dbClient.format(sql,...) 函数转换 SQL 语句。

如果SQL参数后面的格式化参数是一个数组，SQL中的占位符请使用?或??，
如果格式化参数是非数组的表参数，SQL中的占位符请使用@开头的命名参数。
如果格式化参数不是表，则调用  string.format(sql,...) 格式化。

dbClient.format()对于数值和布尔值不作转换，
字符串会进行安全转义处理（可避免SQL非法注入），buffer会转换为16进制编码。
如果参数值是数组则展开为列表,例如 'a','b',
如果是嵌套数组则展开为分组列表, 例如('a','b'),('c','d')
如果参数值是命名表对象（非数组），则格式化为SQL键值对（键调用dbMysql.escapeId 函数格式化 ）。
*/

//为避免有人乱改测试数据库表结构又没有删除，先删除旧的表
dbClient.query("DROP TABLE IF EXISTS ??",{library});
//上面的SQL会格式化为 "DROP TABLE IF EXISTS `library`"

//执行SQL语句,注意mysql表名、字段名可包含于反引号内(用键盘左上角ESC下方的按键内)
var ret,err = dbClient.query("CREATE TABLE IF NOT EXISTS `library` (
    `id` int NOT NULL AUTO_INCREMENT,
    `name` VARCHAR(100) NOT NULL,
    `auditing` TINYINT(1) DEFAULT '0',
    `bytes` BLOB,
    `time` DATETIME DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (`id`),
    UNIQUE KEY `id` (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;");
console.dump(ret,err)

var dbTable,err  = dbClient.listTables()
for tbl in dbTable.each(){
    console.log("发现数据表:",tbl);
}

var ret,err = dbClient.query("INSERT `library` VALUES(null,'测试',@num,@str,@time)",{
    num = 123;
    str = raw.buffer("测试:'这是字符串!'命名参数可以自动处理字符串转义");
    time = time();
} )

console.dump(ret)

//如果是占位符对应的格式化参数是嵌套数组则展开为分组列表, 例如('a','b'),('c','d')
var ret,err = dbClient.query("INSERT `library`(`name`,`auditing`,`bytes`,`time`) VALUES ?",{ {
    {'嵌套数组1',123,"测试xx",time()}
    {'嵌套数组2',123,raw.buffer("测试xxx2"),time()}
```

```
} } )
console.dump(ret,err);

var ret,err = dbClient.query("INSERT `library`(`name`,`auditing`,`bytes`,`time`) VALUES('测试2',@num,@str,@time)",{
    num = 123;
    str = "测试2";
    time = time().addday(-2);
} )
console.dump(ret,err);

//如果格式化参数值是数组则展开为列表,例如 'a','b', 示例如下:
dbClient.query("DELETE FROM `library` WHERE `name` IN (?)",{{'嵌套数组1','嵌套数组2'}})

//查询数据并返回记录集
var result = dbClient.query("SELECT * FROM `library` WHERE time >=@time",{
    time = tostring(time().addday(-3),"%Y/%m/%d")
});
for name,auditing,bytes in result.each(){
    console.log("输出", name,auditing,bytes ); //逐行输出所有记录
}

//查询数据并返回全部记录集到一个表
var result = dbClient.query("SELECT * FROM `library` WHERE time >= (
    SELECT DATE(`time`) FROM `library` ORDER BY `time` DESC LIMIT 0,1
)");
console.dumpJson(result.getTable())

//创建存储过程
var ret,err = dbClient.query("
CREATE PROCEDURE demo(IN n int)
BEGIN
    SET n=123;
    SELECT n;
END;
")

//调用存储过程
var ret,err = dbClient.getTable("CALL demo(@n)",{
    n=123 //传入命名参数
});

console.dump(ret);

//移除存储过程
console.dump( dbClient.query("DROP PROCEDURE IF EXISTS `demo`") );

//移除数据库
dbClient.query("DROP TABLE IF EXISTS `library`");
console.pause();
```

Markdown 格式