DataWindowHTTP　V1.00 产品说明书

http://datawindow.ltd/　http://powerbuilder.ltd/　http://mis2erp.com/

QQ：27-3939-617/Mob：137-9029-9411

Mail：chengang0769@21cn.com/273939617@qq.com

2021-10-06

目录

历史由来

PB 的优点就是连接数据库简单，设计方便。其 datawindow 控件集设计，展示，报表，存储，导入，导出于一体。本设计在 datawindow 的基础上做加强设计，将其连接数据库的功能，从传统局域网延展到互联网。

PB 连接数据库的技术很传统，很直接，我们习惯称之为： 直连数据库。直连数据库是许多开发工具连接数据库的模式，是最直接的 CS 模式没有 N 层。虽然要借助于数据库驱动，但本质仍然是一种 server/client 模式。我们通常只管连接，并不在意如何连接。是一种简单可靠的数据访问模式，常用于中小系统设计，正好契合企业管理系统的开发要求。这是 PB 的优点。

但 PB 写的程序，在局域网使用还算不错。不过也仅限于 PC 机+网线模式。如果是用笔记本电脑和 WIFI 接入，或者是使用外网的服务器和数据库时，会经常遇到网络中断：PPPOE 上网产生的中断，电脑睡眠产生的中断，WIFI 掉线产生的中断，移动笔记本带来的中断……从而导致程序无法提交或者崩溃退出。直接的后果是，我们无法向客户解释，它看起来像软件 BUG，而且是频繁发生的。这是 PB 的缺点。

2017 年开发一个广东肇庆的项目，不想直连数据库，因为：断线让程序显得很崩溃，操作员频繁报故障，设计重连也非常不完美，重连失败或者等待时间太长。突发奇想，想采用网络接口访问数据库。花了两到三个月时间，架构了 DataWindowHTTP（曾用名 DataWindowJSON）第一个版本。几年前有做过 PB 与通讯计费系统通过 JSON 数据交互方面的运用，积累了 PB 成功使用互联网接口方面的一点点经验。这是 DataWindowHTTP 的可行性。

在某些 PB 版本中，可以设计基于 webservice 的 dw，这个模式的优点是可以方便地调用 web 接口了。缺点是：它只是针对部分对外服务而设计的。并不能采用这个模式来设计所有 dw，因为那样需要无数个接口。而 DataWindowHTTP 是设计成类似“桥（brige）”或“路由（router）”的功能，只需要一个“公共接口”，就可以让所有 dw 访问互联网数据库。对于如今的管理系统而言，连接数据库变得复杂了，这个变化是因为互联网这样的异构造成的：服务器（数量增多，机房增多，局内局外都有），数据库（数据库类型变多，各种用途的库增多）和办公地点（省，市分公司模式，镇区分支，连锁零售点等），这些是分散的，他们的互通，都是通过互联网这个纽带。在整个公司范畴来看，数据是整体和统一的。所以：“PB 要以较小成本继续高效率开发的话，它需要 DataWindowHTTP。DataWindowHTTP 让 PB 轻松连接到基于互联网的服务器，而不需要在接口上增加一分钱成本。”。DataWindowHTTP 采用非持久连接，让它可以轻松连接到互联网数据库。DataWindowHTTP 就是基于这

样的设计目的。这就是 DataWindowHTTP 的技术思路。

以一个通讯业客户来举例：把上百台通讯机的客户资料话单结算信息（mysql 库）集中到一台服务器（mssql2005）来做统一管理，又通过这个管理服务器下发一些例如套餐变更，充值等信息到通讯机。业务变更，充值等接口基于 JSON，大量数据访问我采用了直接读 MYSQL 导出文件传输到 MSSQL 服务器再导入的方式，因为接口做不到这方面。所以这是一个相对复杂的系统。客户基本资料，电话资料，套餐，充值，计费，营收。就是客服人员频繁使用的界面，从多年的运行结果看，直连非常不好，但当时还没有诞生这个组件，还在忍受直连，以为问题会不那么严重。存在的主要问题：查询和录入都相当卡顿，而且频繁断线。真正数据库负担其实并不重，客户端才几十个到一百左右同时在线。但是各个分公司来的访问，网络条件各异，网络延迟，断线，访问慢，提交失败，访问很卡，问题层出不穷，每天都可能爆发问题，但都如出一辙。维护工作把分公司管理员和总公司管理员折腾得够呛。并不是管理员技术不够，主要是远在客户端的情况不是我们可控的。例如网络延迟太大，似乎 PB 在延迟较大时，优势就几乎没有了。卡到界面假死，甚至电脑死机。所以毛病集中在卡顿和断线。有少部分集中在互联互通问题，有少部分是 SQLNCLI 无法安装等等。对于这些恼人的问题，测试过 TEST 中的程序以后，您会瞬间感觉分发 PB 程序这个过程，已经轻松到不需要做任何事情了。就像“PB125-自解压单个 EXE”，完全可以放入一个“工作群组”里面就行了。而且您会明白，这个 TEST 随时打开都是可用的：如果网络断开了，它第一时间，一秒之内干干脆脆提示您网络无法连接。如果网络恢复，没有提交的单据可以继续提交，没有任何麻烦。它不需要任何发行，安装，配置，维护的成本。同时也证明了基于本组件开发的 PB 程序，它是稳定和易用的。这个案例是对 DataWindowHTTP 的真实需求。

产品定位

基于 Datawindow 的基本功能：设计，展示，增删改查等丰富特性，在其上嫁接互联网接口发送和接收功能。采用 DatawindowHTTP+JSON 封装数据+AES 动态密码+RSA 加密 AES 的动态密码+AES 对数据加密+服务器接口+PHP8.0+APACHE2.4+数据库的模式。这样 PB 开发的程序不再需要安装数据库接口。这个访问链主要分三段：

1) DatawindowHTTP+JSON 封装数据+AES 动态密码+RSA 加密 AES 的动态密码+AES 数据加密： 全部封装到一个控件中。开发者可以无视它们的复杂内部构造，它已经封装成组件了。
2) PHP 接口 + PHP8.0 + APACHE2.4：这是固定的接口，只需要简单安装配置，它等于一个“桥（brige）”或“路由(router)”，把 DW 的数据提交到数据库。
3) 数据库。目前经过测试的数据库有：

MSSQL2000，2005，2008，2012，2014，2017	（考虑多数人的最爱）
Oracle10g，11g，12c	（考虑专业的数据库）
Mysql5.7，8.0	（考虑到有人尝试开源，在香港见到 PB12.5+MYSQL 的销售系统）
MariaDB10.5	（考虑到有人尝试开源，不满 MYSQL 的可以研究这个）
PostgreSQL12.6	（考虑到有人尝试开源，这个值得研究，很多基于这个发行魔改的数据库）
Sybase16.0	（因时间关系，待测试）
Db2V11.1	（因时间关系，待测试）

产品特色

1) 轻松连接互联网云主机的数据库。
2) 做减法设计，为简化而设计。
3) 实现互联网与局域网有机统一。
4) 无需改变设计的习惯。设计阶段采用传统直连模式，而运行阶段采用互联网接口模式。
5) 程序发行时无需数据库驱动，无需管理员远程安装和配置。
6) 服务器带负载能力强，负载容易均衡，数据库压力小。
7) 同时连接多个局域网/互联网数据库时很方便。
8) 实现了程序的真正移动运用，用户可以随时随地在笔记本上面漫游使用，一笔数据即使延迟几天也可以成功提交。
9) 使用 AES 加密上传下载的数据，安全性有了保证。
10) 采用久经验证的经典配置：Apache+PHP 来驱动，对服务器的硬件配置要求低。

开发语言

组件代码采用 PB 代码实现，确保可以移植到多个版本。

适用版本

初始版本于 2017 年采用 PB12.5 开发，并基于它开发了 ERP。鉴于它的实用性，于 2021 年整理和简化接口，并成功移植到 PB9,10.0,10.5,11.0,12.5,2019。这个组件支持≥9.0 的所有版本。dwhttp.pbd 开发组件将以 PBD 作为发行组件，提供给开发者。dwhttp.pbd 将不会提供 PB 源代码。

授权方式

开发者在购买后是授权【公司名称+联系电话】或者【软件工作室名称+联系电话】。这个信息会在组件产生提示时显示出来。这个授权不限制开发者使用的电脑数量，最终开发的软件系统也不受限制（即不限制安装的服务器台数，也不限制连接的客户数量，也不限制最终产品可使用的时间）。

使用建议

本组件特别适用于基于互联网云主机的数据库管理系统。它彻底解决直连中产生的种种断线问题。提供了极具特色的驱动模式。

首先，它适合新的企业管理系统开发。

组件没有依赖其他框架，功能很单纯，您完全可以将其融合进界面框架，可以融合进业务架构，可以融合进报表系统，从而构建复杂的系统。

强调一：不强制要求所有地方都必须用本组件，您仍然可以使用 SQLCA，甚至定义新的 transaction 或者多个 transaction，本组件可以和 SQLCA 体系并存。只要你需要就择机使用，没有强制性。

强调二：您仍然可以使用 xml，json 等访问外部接口，比如医疗，第三方支付等等。本组件要解决的主要是：连接管理系统的核心数据库这个最基础问题。它和您要访问的其他“接口”不冲突，不相干。

强调三：你可以有很多数据库连接，但是如果某个数据库不是在咫尺之内，而是远在互联网的某个位置，它就特别适合本组件连接。所以采用本组件连接的，可以只是其中一个数据库。只要少量适用，你就可以使用。

强调四：您可以只在部分模组使用本组件。比如向远程数据库查询或者输送少量数据。比如很多连锁销售系统的汇总上传功能。

强调五：局域网能否运用本组件？局域网从网络的速度上面说，可以不使用。但是局域网现在的断线也越来越多。主要是 wifi 的断线和设备睡眠造成的断线。从某种意义上来说，PB 的直连模式早已经不适合现代的设备了。Wifi 强度总是在动态变化，并且毫无察觉地断线重连。PC 上面不必因为绿色节能而自动休眠，网卡可以一直是在线的。而现在多用笔记本电脑，绿色节能，电脑休眠，网卡会默认断开了。所以本组件其实挺适合局域网的。用它改造系统，能完全适应现在的网络以及设备。从这点上说，使用本组件倒成了必要了。

很多传统公司的 ERP 这些数据库其实还在局域网里面跑，因为数据安全性放在第一位，但也有部分公司数据库已经在托管机房或者用云主机了。也有折中的，比如内外各一个数据库。然后通过程序互通数据，麻烦但是安全。无论如何，放在互联网上面访问的数据库越来越多了。为何要开发访问互联网数据库的运用？这都是业务的实际需求或者是公司的布局造成的。有给上下游厂商自行录入资料的（比如供应链）；有方便上下游快速查询结果的（比如第三方检验）；有收集订单然后转入 ERP 系统（接单代工）；有分部集中统一使用总部系统的（分支驻点办公）；有为了实现了电子化的全流程的（有移动办公或者业务员对接公司系统）；有需要逐级落实完成的任务派发（比如客户投诉与故障单处理反馈回访系统）。过去人们曾寄希望 OA 系统能解决很多流程或填报单据的需求，但是到后来发现，OA 跑完的东西仍然要入具体的管理系统。把 ERP 系统/管理系

统做成任何地方都直接能访问则意义更大。这样的企业早已经把互联网当做“交换机”了，他们的数据早已经脱离了局域网的概念或者脱离了本地服务器概念。也可以说，这样的需求早从 2000+年就已经开始了，某些客户开设了驻镇区的办公室，特殊的行业决定了他们的需求会较早。甚至说，某些类型的客户他一开始就没有局域网系统的概念，他的系统从一开始就是要从外部进行访问的。这样的系统就非常适合采用本组件开发，而且系统整体都会受益的。

其次，它适合老项目的改造。

您可能觉得对于现有系统的整体改造很困难。但有折中的办法，可以把一些频繁使用的主要操作界面用本组件进行改造，极少使用的界面可以不用改造，不改造的界面仍然使用 SQLCA 连接，SQLCA 连接和本组件是可以并行存在的。我曾经统计了一些项目，发现虽然界面很多，但是极其频繁使用的界面非常少，通常不超过 5-10%。比如库存系统中，入库单出库单使用的频率极高；调整单和盘点单使用频率就极低。用这个组件改造，必然能起到立竿见影的效果。

改造一个简单界面的步骤：

第一步：将 dw_1,dw_2 等控件的祖先由 datawindow 改为 datawindow_http；（editsource 中搜索替换一次搞定）

第二步：将 settransobject(sqlca)改为 settranshttp(serverinfo)；（一般在 open/ue_open 事件/dw.constructor 中找到改写）

第三步：将 retrieve(parm1,parm2...)改为 retrievehttp(parm1,parm2...)；（一般在 ue_query/uf_query/查询按钮事件中找到改写）

第四步：将 update()改为 updatehttp()；不更新的页面则省略此步。（一般在 ue_update/uf_update/保存按钮事件中找到改写）

从哪些界面开始改动比较容易？

改造系统一个原则是先易后难，逐渐积累经验。而且要保证系统可用性。不要改出毛病。

首先，可以从报表开始。因为报表都是基于查询结果的。没有什么逻辑，不至于改用本组件而导致 bug。

其次，可以从使用频繁但主要是基于查询结果的界面开始，用 ERP 举例，料品基本资料，BOM 表，库存，历史报价，客户资料，供应商基本资料等都是基于“只读查询”的。而录入编辑部分可以暂缓。

再有，可以更改几个使用频率非常高的，非常容易导致断线故障的界面。

最后，复杂逻辑界面，短时间无法改造或者可能引入 BUG 的，可以不改造，仍然用 sqlca。

后续改进

后续版本，将完善“基于消息队列的分布式事务”和“基于 XA 协议的两阶段提交分布式事务”的例子。

- 目前虽然可以将多个 SQL 动态语句或者多个 dw 语句包在一个事务里面执行。但没法把中间结果放在一个事务中。
- “基于消息队列的分布式事务”是最实用的分布式事务。目前组件在技术上是支持的，只是缺少一个实际的例子程序。
- “基于 XA 协议的两阶段提交分布式事务”是 Tuxedo 提出，多数商业数据库实现了 XA，目前组件在技术上是支持的，只是缺少一个实际的例子程序。

如有需要可提供一个 Linux+Mysql+Apache2.4+PHP8.0 的安装文档。

编程实例

Code\：演示代码源码（为减少文件体积，压缩包只含有 12.5 版本，如需其他版本请联系）

Test\：已编译直接运行的测试程序（为减少文件体积，压缩包只含有 12.5 版本，如需其他版本请联系）

Server\apache-php-driver: apache，php，sqlsrv 等软件和数据库驱动。

Server\db：数据库

Server\interface：dwhttp 服务器接口 PHP 代码

在线演示

提供一台服务器用于在线演示。这样大家无需搭建环境而快速测试。请运行 Test 目录的程序。

有兴趣深度测试的，可以加 QQ 索取自己需要的 PB 版本的例子程序，按文档安装服务器，运行程序例子。

网络架构

客户的网络架构参差不齐，要使用本组件的前提条件就是：固定 IP 或域名+端口。只要满足这两个条件就行了。域名倒不是必须的。端口也不一定需要 80 端口，非 80 端口也可以，所以主机实名认证也不是必须的。

如果固定 IP 这个条件不具备，也有一些变通的做法。归纳如下：

1. 局域网内的自有服务器安装的数据库

1.1 专网上网，有固定 IP 地址

这种一般适合大企业，企业内部可能需要架设 web 和 mail 服务器，或者有重要的数据要由外部（比如：分公司，分部，网点）来连接。

适合本组件，直接使用。因为局域网共享一个外部固定 IP，所以需要映射端口到服务器。

端口映射：一般在接入的路由器上面设置，将某个外部不知名端口（出于安全考虑）映射到内部某个服务器的服务端口。在路由器中一般位于：高级配置》虚拟服务。具体配置请搜索“路由器虚拟服务”教程。

1.2 光纤或 ADSL（PPPOE 协议）拨号上网

1.2.1 获得动态 IP，IP 为公网 IP

比如 202.96.*.*，路由器一般在：接口信息》WAN 线路信息 显示。

因为 IP 不固定，可以使用动态域名解析系统（简称 DDNS：Dynamic Domain Name Server），即申请一个动态域名指向该局域网，局域网内某个服务器安装动态域名系统的客户端，这样外部可以通过域名访问内部系统。本组件必须通过动态域名使用。路由器也需要配置：高级配置》虚拟服务。

特点：不用申请专线，因为企业专线毕竟比较贵，一般企业也用不上。

说明： 现在的光纤网络比过去的铜线网络要稳定得多，快得多。偶尔的断线重新拨号会影响传统的 PB 直连方式，但是不会影响本组件的连接。因为本组件不是永久连接。就像偶尔的断网不影响你使用网页是一个道理。

1.2.2 动态 IP，IP 为私有 IP

现在公网 IP 稀有，所以运营商大量采用私有 IP，如：100.100.*。*，这样的 IP 就无法使用动态域名解析。但是可以使用 FRP（内网穿透）软件。将 FRP 安装在任意一台有公网 IP 的云主机上面（这台不是专门为这个目的而购买，只是“兼职”，就不占什么成本了），客户端安装在内网服务器上（安装方法可以搜“FRP 安装教程”）。如果的确没有购买任何云主机，可以考虑购买一台最低配置的“轻量云主机”，这种云主机最低配通常为 5M 带宽/每月 1000G 流量，费用很低。FRP 方案，我亲自用于项目中是成功的。

FRP 方法也适合 2.1 的情况，就可以不使用 DDNS。

DDNS 实际过程中有不稳定的因素。而且有延迟的问题。FRP 也存在网络中断引起短时不可连接。但是本组件很好的克服了短时掉线问题。

2. 云主机安装数据库

可使用本组件，标准模式。对于信赖云主机安全性的客户，推荐这种。

3. 云数据库（没独立主机）

不可以直接使用本组件，但是可以另外架设一台 web 服务器（或借用其他已经建立起来的 web 服务器）即可使用本组件。Web 服务器连接你购买的云数据库即可。但不推荐这种。

4. 托管主机

可使用本组件，标准模式。对于安全性要求高的客户，推荐这种。

5. 自建机房

可使用本组件。而且非常适合。因为这种方式 IP 资源丰富，灵活性很强，由专人管理。轻易满足条件。

接口说明

四个组件：

datastore_http，datawindow_http，datawindowchild_http，embeddedsql_http

1. datastore_http

用于不可视的 DataStore 的检索

1.1 public subroutine settranshttp (str_serverinfo serverinfo)

作用：

初始化接口信息

用例：

```
str_serverinfo gstr_serverinfo
embeddedsql_http gds_sql

//服务器，端口，目录，主页面
gstr_serverinfo.server = “***.***.***”                    //主页网址或者 IP
gstr_serverinfo.port = 8080                               //端口
gstr_serverinfo.path = "DataWindowHTTP”                   //子目录
gstr_serverinfo.page = "query.php"                        //接口主页面

//使用接口的用户名和密码
gstr_serverinfo.userid = "***"
gstr_serverinfo.pwd = "***"

//指定 rsa 公钥，两种方式：
//1. public.pem 以文件形式存在，指定：路径+文件名，方便定期更换公钥。
gstr_serverinfo.rsapubkey = curpath + "\public.pem"
//2.用字符串把 pubkey 写下来,这样可以不放到文件夹里面，也可以动态拼接，相对更安全，但是不好定期更换公钥。
gstr_serverinfo.rsapubkey = "-----BEGIN PUBLIC KEY-----~n"+&
"MIGfMA0GCSqESIb3EQECAQUAA4GNADCBiQKBgQDkfZ9VABDgVktkss3njg7sVdjY~n"+&
"8mP4mrUVUBAJlcIGtzFg6rjRcjaR2N/KXtQJ3kU319Nn1fl0SVQLS+0lCFa10gKu~n"+&
"mRMMGAg2WiOoYs4r9fzbkukbkWCpWZxoJNnG18ruVn7dc2tQx6gscfO5ZSh394qT~n"+&
"b9RmOfgmX7qgknj+SQIDAQAB~n"+&
"-----END PUBLIC KEY-----~n"

//server 的信息赋给控件并初始化，等同于 dw.settransobject()
gds_sql.settranshttp(gstr_serverinfo)
```

1.2 public subroutine retrievehttp ()

作用:

无参数检索

用例:

```
gds_fk.retrievehttp()
```

1.3 public function long retrievehttp (any pvalue1, ..., any pvalueN)

作用:

带参数检索

其中，参数个数固定为 1-30 个，35 个，40 个，45 个，50 个。

如果接口恰好需要 33 个，请采用 35 个参数调用。没有的参数送入字符串空值或者 0 皆可。

用例：

```
//检索，有参数
Long p1
String p2
```

//与 retrieve 接口一致。可以送入变量。或者常量，参数类型自动识别。
//但是参数顺序和设计时必须严格一致。
gds_fk.retrievehttp(p1,p2,"test")

1.4 public subroutine updatehttp (boolean ab_accepttext, boolean ab_resetupdate)
作用：
更新数据
参数：
ab_accepttext：执行 accepttext()
ab_resetupdate: 执行 resetupdate()

属性
public statictext ist_prompt
设置一个外部 statictext，可以实时显示数据检索进度，可以为空则不显示。
public int ret
执行检索后的返回值，判断检索成功以否的标志
public int nrows
执行检索后的行数

2. datawindow_http
用于可视的 DataWindow 的检索
2.1 public subroutine settranshttp (str_serverinfo newserver)
2.2 public subroutine updatehttp (boolean ab_accepttext, boolean ab_resetupdate)
2.3 public function long retrievehttp ()
2.4 public function long retrievehttp (any pvalue1,...any pvalueN)
同 datastore_http 的同名函数相同，此处略。
2.5 public subroutine bindparm (string pname, any pvalue)
绑定参数，一次绑定一个参数。
Pname：参数名
Pvalue：值
2.6 public subroutine turntopage (long al_page)
翻页
al_page：指定页码
2.7 public subroutine updatemerge (ref datawindow_http adw_mergedw1,...,ref datawindow_http adw_mergedwN,
boolean ab_accepttext, boolean ab_resetupdate)
多个 dw 合并 sql 语句，一次性更新（在一个事务里面，确保一致性）。最多支持 1-10 个 dw。
这个不同于传统的多个 dw 更新逐个提交模式。这个在提交上面是一次的，被包在一个 trans 中，效率更高。相对来说，事务锁定时间缩短。

属性：
public statictext ist_prompt
public int ret
public int nrows
同 datastore_http 的同名属性

public boolean setredraw = true
设置是否组件取得数据后刷新，有时需要检索后再过滤可以取消，避免不必要的刷新

public long pagerows = 0

分页时，每页的行数

public long pages = 0

分页时，根据检索 nRows/pagerows 得到的页数。

3. datawindowchild_http

用于下拉 dddw 的检索和 composite 报表的检索

3.1 public subroutine settranshttp (str_serverinfo serverinfo)

3.2 public function long retrievehttp ()

3.3 public function long retrievehttp (any pvalue1, ..., any pvalueN)

同 datastore_http 的同名函数

属性

public statictext ist_prompt

public int ret

public int nrows

同 datastore_http 的同名属性

4. embeddedsql_http

用于内嵌（动态）sql 的检索，包括内嵌 sql 语句和调用存储器。并可将动态语句生成的结果集放置于 ds 或者 dw 中。

4.1 public subroutine settranshttp (str_serverinfo serverinfo)

4.2 public subroutine bindparm (string pname, any pvalue)

同 datawindow_http 中同名函数

4.3 public subroutine selectinto (string ptype1)

4.4 public subroutine selectinto (string ptype1,..., string ptypeN)

模拟执行 select into 查询

ptype1：接收返回值的变量的类型,如”string”,”long”,”longlong”,”datetime”，和系统变量类型相同，也可以简写为”s”,”l”,”ll”,”dt”。

4.5 public subroutine selectintodw (ref datastore ds_ext)

public subroutine selectintodw (ref datawindow dw_ext)

将动态 sql 语句结果集放入 datastore/datawindow 中，比如存储器的结果。

ds_ext/dw_ext: datastore/datawindow 控件名

4.6 public function integer addsql (string sql)

添加动态执行的 sql 语句。可以多次添加，执行时按添加顺序执行。执行后清零。

4.7 public function integer addsql (string sql,boolean savesql)

添加动态执行的 sql 语句。可以多次添加，执行时按添加顺序执行。执行后保持语句。

如执行一个循环，可以写在循环外，只需要一次初始化赋值。

Savesql= QSAVESQL/QCLEARSQL

这个开关一旦打开，是针对所有 SQL 的，直到 executesql 后仍然保持。而且后续 addsql 也生效。

4.8 public subroutine executesql ()

执行动态语句，无参数。

4.9 public subroutine executesql (integer ai_intrans)

执行动态语句

ai_intrans: 是否将语句包括在一个事务(begin trans...end trans)中

ai_intrans = QNOTRANS/QTRANS

属性

public statictext ist_prompt

public int ret

public int nrows

同 Datastore_http 中同名属性

public str_dwparm intovar[]

执行 select into 模式的查询后，放置 into 变量取得的结果。

boolean ib_dontmsg = false

是否显示错误提示消息

有时不要组件的提示，直接根据 ret 返回值做静默处理