**DeWeb开发指南**

# DeWeb介绍

DeWeb是一个可以直接将Delphi程序快速转换为网页应用的工具！

使用DeWeb, 开发者不需要学习HTML/CSS、JavaScript、Java、PHP、ASP、C#等新知识，用Delphi搞定一切。

DeWeb开发的网页支持所有客户端，包括手机、平板等。

网站: http://www.delphibbs.com

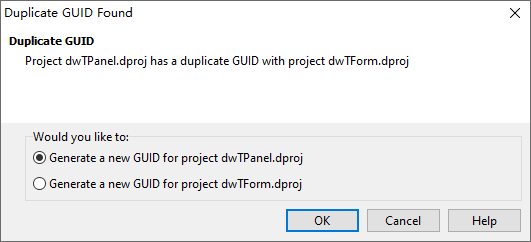
# 支持控件

* 支持所有数据连接组件(ADOConnection/FDConnection/UniConnection)等；
* 支持所有数据表组件（Query/Table）;
* 支持所有三层中间件；
* 支持BitBtn(用于选择拟上传的文件/图像/视频)
* 支持Button
* 支持Calendar
* 支持Chart
* 支持CheckBox
* 支持ComboBox
* 支持DateTimePicker
* 支持Edit
* 支持Image
* 支持Label
* 支持ListBox
* 支持MainMenu
* 支持Memo
* 支持Panel
* 支持PageControl
* 支持RadioButton
* 支持SpinEdit
* 支持Shape(用于扫描二维码)
* 支持SpeedButton
* 支持StringGrid
* 支持Timer
* 支持TreeView

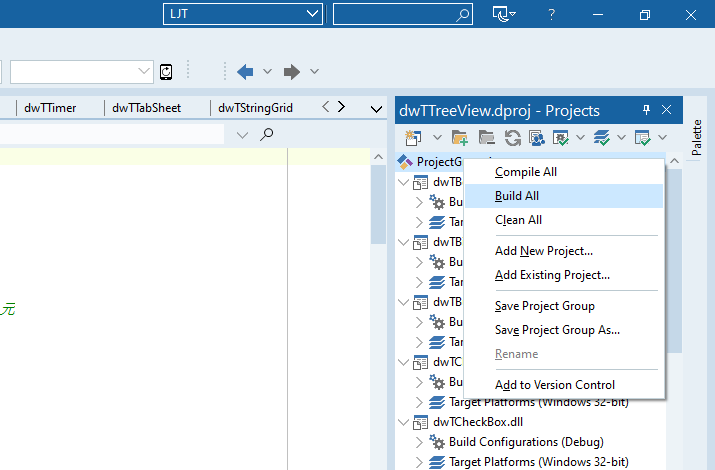
# DeWeb入门

* 开发版本
* 目前支持Delphi 10.3/10.4/11.3。
* DeWeb默认使用Delphi 10.4.2 编译，推荐使用此版本开发！
* 第一次编译前准备（**重要！** 许多新用户忽略这一步，导致出现空白界面）
* 打开Source\dwVcls\中所有dpr

如果遇到以下情况，选择“OK”按钮



* 然后Build all；



* 如果有Source\dwThirdVcls\，则打开Source\dwThirdVcls\ 中所有的dpr，然后Build all。这一部分可能需要提前安装相应第三方控件；
* 复制对应的服务器程序(仅Delphi版本非10.4.x的需要)。

先删除DeWebServer.exe，

如果是D11.3， 则重命名DeWebServer113.exe为DeWebServer.exe；

如果是D10.3.x，则重命名DeWebServer103.exe为DeWebServer.exe；

* 入门教程hello,world!
* 新建一个DLL，命名为hello1.dpr
* 新增一个Form.(File->New->VCL Form - Delphi)。**注**：不要更改单元名称和Form名称，即分别为unit1.pas和Form1
* 在Form1中添加一个Label,设置其Caption为"Hello,DeWeb"
* 将hello1.dpr源文件代码更改为

library hello1;

uses

ShareMem,

SysUtils,

Forms,

Messages,

StdCtrls,

Variants,

Windows,

Classes,

Data.Win.ADODB,

unit1 in 'unit1.pas' {Form1};

{$R \*.res}

var

DLLApp : TApplication;

DLLScreen : TScreen;

function dwLoad(AParams:String;AConnection:TADOConnection;AApp:TApplication;AScreen:TScreen):TForm;stdcall;

begin

//

Application := AApp;

Screen := AScreen;

//

Form1 := TForm1.Create(nil);

//Form1.ADOQuery1.Connection := AConnection;

Result := Form1;

end;

procedure DLLUnloadProc(dwReason: DWORD);

begin

if dwReason = DLL\_PROCESS\_DETACH then begin

Application := DLLApp; //恢复

Screen := DLLScreen;

end;

end;

exports

dwLoad;

begin

DLLApp := Application; //保存 DLL 中初始的 Application 对象

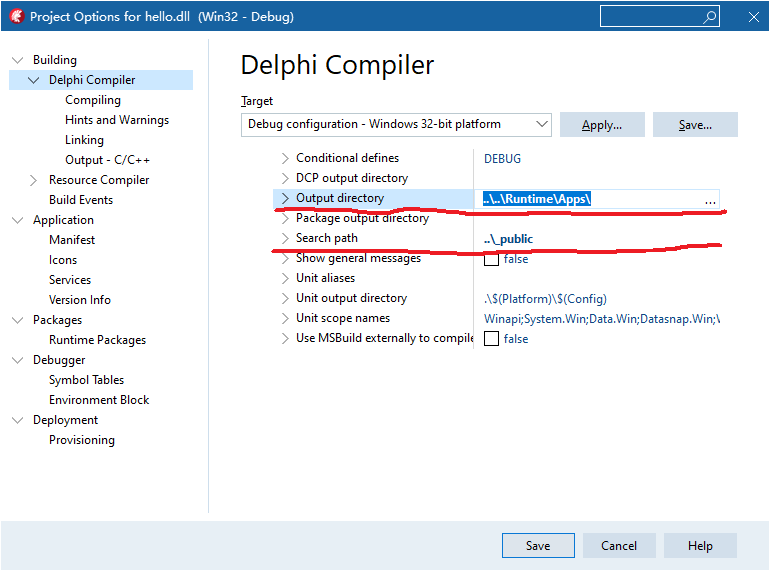
DLLScreen := Screen;

DLLProc := @DLLUnloadProc;//保证DLL卸载时恢复原来的Application

DLLUnloadProc(DLL\_PROCESS\_DETACH);

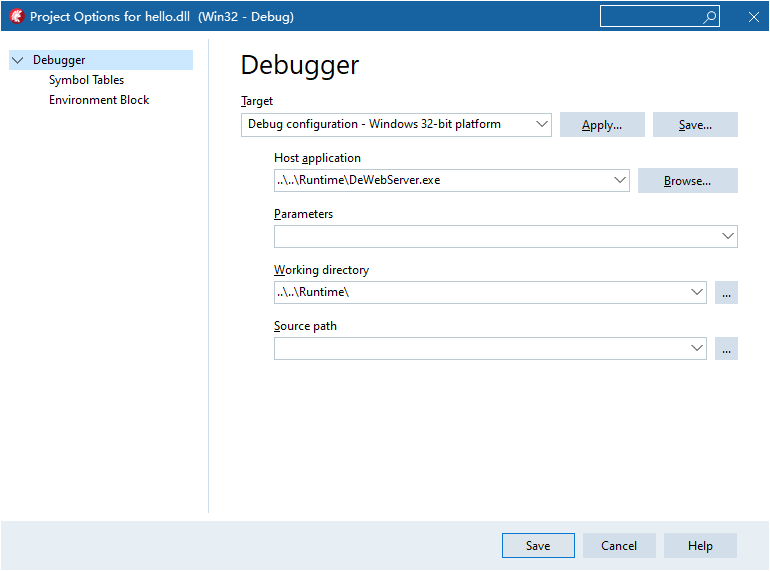
end.

* 保存到source\demos\hello1目录中
* 更改Project设置为

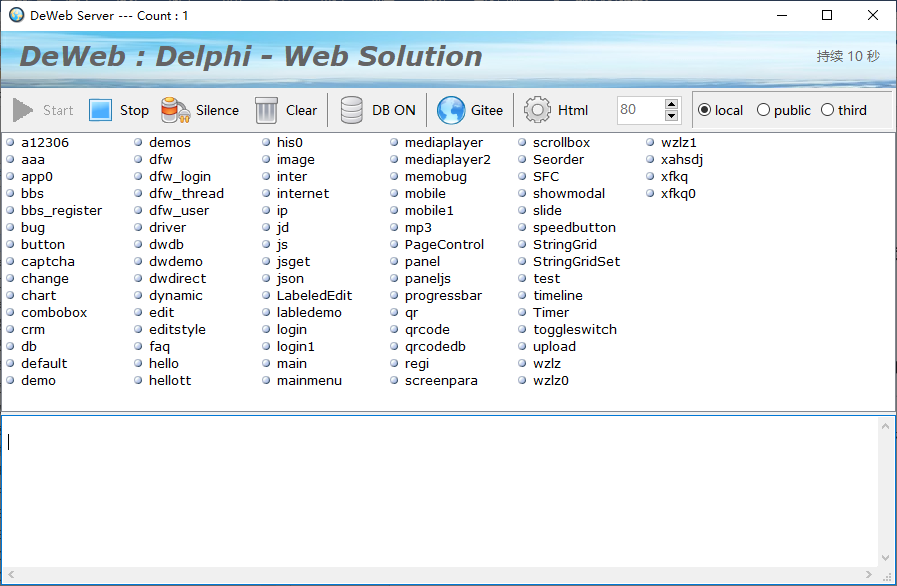


* 设置运行参数

菜单:Run – Parameters



* 编译运行，即可以直接打开服务器程序DeWebserver.exe



* 双击hello1,即可打开当前应用

# 服务程序

* 服务程序为DeWebServer.exe

双击直接运行

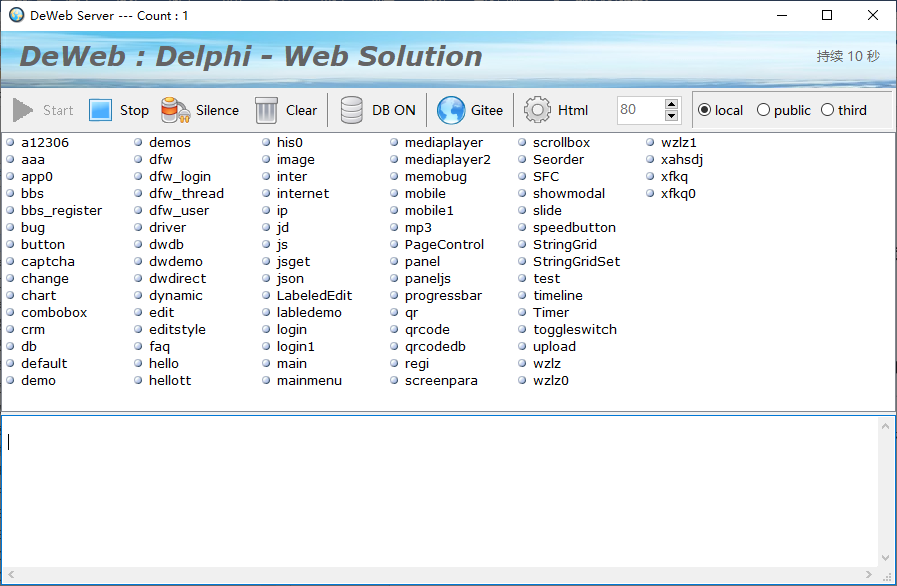
* 带参数命令行运行

如果1个参数，则为运行端口号，如：DeWebServer 8081

如果3个参数，则分别为端口号、窗体Left、窗体Top

如果5个参数，则分别为端口号、窗体Left、窗体Top、窗体Width、窗体Height

* 如果点击start按钮后，start按钮仍处于可用状态，则表明当前端口已被占用，请更换一个端口号或关闭使用该端口的程序后重新start。



其中，

* 按钮Start用于启动DeWeb服务；
* 按钮Stop用于停止DeWeb服务；
* 按钮Slience用于系统调试，点击切换为Debug，可以查看客户端与服务器交互信息
* 按钮Clear用于清空交互信息
* 按钮DB ON/DB OFF用于显示当前公用数据库连接状态。公用数据库连接字符串在配置文件中
* 按钮Gitee用于直接打开DeWeb在码云（https://www.gitee.com）的链接
* 按钮Html用于生成可离线使用的同名Html文件（需要在工程中设置Form1的AlphaBlend=True）。
* local/public/third用于设置系统采用的js/css源，分别为本地、公用和第三方。一般建议为本地，如果部署在互联网，建议采用CDN加速；如果采用第三方，请先在配置文件中进行设置。

# 配置文件

DeWebServer的配置文件为同目录下的config.json文件，基本如下

{

"port": 80,

"resource": 0,

"third": "http://cdn.delphibbs.com",

"redir": [

["mm","hello.dw"],

["gg","login.dw"]

],

"upload\_dir":"upload",

"max\_upload\_size":60,

"errorhtml": "<html><body><br/><p style='text-align:center'>DeWeb web not exist !</p></body></html>",

"connectionstring": "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=DeWeb.mdb",

"disconnect": "网络已断线，请刷新！"

}

其中：

**port** : 整型，为默认启动时的端口；

**resource** : 整型，为启动时资源文件的位置。0表示本地源，1表示公用源，2表示第三方源；

**third** : 字符串型，和resource为2时配合使用，为第三方源网址；

**redir** : 字符串二维数组，为自动地址切换表，每组由2个字符串组成。如：["mm","hello.dw"]；

当访问http://127.0.0.1/mm的时候，DeWebServer会自动打开http://127.0.0.1/hello.dw；

**upload\_dir**:为上传文件存放目录；

**max\_upload\_size**为上传文件最大size，单位为MB

**errorhtml** : 字符串型，为应用不存在时的提示（HTML格式）。

**connectionstring** : 字符串型，为公用数据库ADOConnection的连接字符串。

**disconnect** : 字符串型，为网络断线时的提示字符串。

# 部署

部署时需要runtime中的大部分文件

需要的目录有：

* apps，仅包括需要部署的dll即可
* apps目录应有hot子目录（非必须）。把生成后的应用（dll）复制到hot子目录，可以实现在不重启服务程序的前提下，实现应用更新
* dist，必须。请勿改动
* media，非必须，一般媒体文件存在于此目录
* upload，必须，接收上传文件存放目录
* vcls，必须，包括支持的控件，可以删除未用于的控件
* zxing\_files，非必须，仅当使用扫码识别时用到

需要的文件有：

* DeWebServer.exe或DeWebServer103.exe、DeWebServer104.exe、DeWebServer11.exe等，必须！
* borlndmm.dll，必须！
* config.json，必须！配置文件，可以手动修改
* libcrypto-1\_1.dll，必须！
* libeay32.dll，必须！
* libssl-1\_1.dll，必须！
* sqlite3.dll，非必须！仅使用sqlite数据库时使用
* ssleay32.dll，必须！

# 应用基本设计

* 为了尽量和网络显示一致，建议在Form默认字体为"微软雅黑，11号"，BorderStyle = bsNone；
* 大部分控件的Hint用来存放额外的信息，请勿使用；
* 不能使用ShowMessage/MessageDlg/InputQuery等所有模态窗体；

可以使用DeWeb自带的类似函数dwShowMessage, dwMessage, dwMessageDlg, dwInputQuery等；

* 所有窗体不能Show/ShowModal，

可以使用dwShowModalPro函数来弹出窗体；

# 数据库连接

* 数据库连接可以在Form中创建
* 也可以在DeWebServer中的配置文件中设置数据库连接，然后使用函数dwLoad中的参数AConnection，即在dwLoad函数中加入类似代码：

AForm.ADOQuery1.Connection := AConnection;

# 纯数据交互（post/get）

DeWeb支持通过post/get进行纯数据交互

* 使用方法，类似：

https://delphibbs.com/dwdirect.dll?name=westwind&age=18

* 例程dwdirect位于source\demos\dwdirect

关键函数如下：

**function** dwDirectInteraction(AData:PWideChar):PWideChar;stdcall;

**function** \_Get(AParaName:String):String;

**var**

iPos : Integer; //位置,类似:name=

**begin**

Result := AData;

//删除以前的

iPos := Pos(AParaName+'=',Result);

Delete(Result,1,iPos+Length(AParaName));

//删除&后面的内容

iPos := Pos('&',Result);

if iPos>0 then begin

Result := Copy(Result,1,iPos-1);

end;

**end**;

**begin**

Result :=PWideChar(Format(' Name is %s Age is %s',[\_Get('name'),\_Get('age')]));

**end**;

* **注**：此功能支持其他语言混合编程

# 特殊函数/用法

* 打开一个新页面请使用

dwOpenUrl(self,'/discuss.dw','\_self'); //当前页面打开

dwOpenUrl(self,'/plan.dw','\_blank'); //新建页面打开

* **ShowMessage**采用dwShowMessage代替

procedure dwShowMessage(AMsg:String;AForm:TForm);

其中:

* AMsg：消息内容
* AForm：当前窗体，一般可以使用self

例：dwShowMessage('Welecome to DeWeb!',Self);

**注意**：需要提前把AMsg的字符串的HTML特殊字符替换掉，

如：“\”替换为“\\”等

如果需要自定义标题和按钮标题，则可以用 dwShowMsg函数

procedure dwShowMsg(AMsg,ACaption,AButtonCaption:String;AForm:TForm);

其中:

* AMsg：消息内容
* ACaption：消息框标题
* AButtonCaption：按钮标题
* AForm：当前窗体，一般可以使用self

例：dwShowMessage('Welecome to DeWeb!','DeWeb Demo','确定',Self);

* **dwMessage**函数，用于显示消息提示

procedure dwMessage(AMessage,AType:String;AForm:TForm);

其中:

* AMessage：消息内容
* AType：为消息类型:info/success/warning/error
* AForm：当前窗体，一般可以使用self

例：dwMessage('Welecome to DeWeb!','warning',Self);

**注意**：需要提前把AMsg的字符串的HTML特殊字符替换掉，

如：“\”替换为“\\”等

* 确定取消的**MessageDlg**采用dwMessageDlg代替

procedure dwMessageDlg(AMsg,ACaption,confirmButtonCaption,cancelButtonCaption,AMethedName:String;AForm:TForm);

其中:

* AMsg：消息内容
* ACaption：消息框标题
* confirmButtonCaption：确定按钮标题
* cancelButtonCaption：取消按钮标题
* AMethedName：返回结果后触发Form的OnStartDock中的事件标志
* AForm：当前窗体，一般可以使用self

dwMessageDlg没返回值，所以不能在当前函数/过程中处理，需要等结果返回后会自动激活Form的OnStartDock函数，

所以需要在Form的OnStartDock函数中添加事件，在事件中通过以下取得前期拟触发的事件标志

dwGetProp(Self,'interactionmethod');

通过类似以下得到返回的结果，为'1'时表示为"确定"，否则为"取消"

dwGetProp(Self,'interactionvalue');

* **InputQuery**采用dwInputQuery代替

procedure dwInputQuery(AMsg,ACaption,ADefault,confirmButtonCaption,cancelButtonCaption,AMethedName:String;AForm:TForm);

其中:

* AMsg：消息内容
* ACaption：消息框标题
* ADefault：输入框的默认值
* confirmButtonCaption：确定按钮标题
* cancelButtonCaption：取消按钮标题
* AMethedName：返回结果后触发Form的OnStartDock中的事件标志
* AForm：当前窗体，一般可以使用self

dwInputQuery没返回值，所以不能在当前函数/过程中处理，需要等结果返回后会自动激活Form的OnStartDock函数，

所以需要在Form的OnStartDock函数中添加事件，在事件中通过以下取得前期拟触发的事件标志

dwGetProp(Self,'interactionmethod');

通过类似以下得到返回的结果，为输入的内容。

dwGetProp(Self,'interactionvalue');

* **dwOpenUrl**打开网页

procedure dwOpenUrl(AForm:TForm;AUrl:String;AParams:String);

AForm:TForm 当前窗体, 一般用self

AUrl: String， 网址

AParams: String; '\_blank'是新窗口打开，'\_self'是当前窗口打开

* **dwSetCookie**设置Cookie

function dwSetCookie(AForm:TForm;AName,AValue:String;AExpireHours:Double):Integer

AForm:TForm 当前窗体, 一般用self

AName: String， Cookie名称

AValue: String; Cookie值

AExpireHours:Double 有效期（小时）

* **dwSetCookiePro**设置域名通用Cookie

function dwSetCookiePro(AForm:TForm;AName,AValue,APath,ADomain:String; AExpireHours : Integer ) : Integer;

AForm:TForm 当前窗体, 一般用self

AName: String， Cookie名称

AValue: String; Cookie值

APath:String;

ADomain:String;

AExpireHours:Double 有效期（小时）

类似：

dwSetCookiePro(self,sCookie,sValue,'/','.delphibbs.com',30\*24);

* **dwGetCookie**读取Cookie值

function dwGetCookie(AForm:TForm;AName:String):String;

AForm:TForm 当前窗体, 一般用self

AName: String， Cookie名称

* **dwRunJS** 运行javascript代码

例：alert('hello,world!');

原生的弹框函数

例：this.Button1\_\_cap='btn1';this.Label1\_\_lef='20px';

设置按钮Button1的Caption为“btn1”,设置Lable1的Left为20

* **用Javascript向Form发送消息（传递数据）**

可以在javascript中执行以下代码：

let me=this;axios.get('{"m":"interaction","i":'+this.clientid+',"t":"**varname**","v":"**varvalue**"}').then(resp =>{me.procResp(resp.data)});

以上代码中加下划线的部分可以根据需要自行设置（其他部分尽量不要修改）

varname一般用作拟传递变量的名称；

varvalue一般用作拟传递变量的值。

执行以上代码后，会自动激活Form的OnStartDock函数，

所以需要在Form的OnStartDock函数中添加事件，在事件中通过以下取得前期拟触发变量的名称

dwGetProp(Self,'interactionmethod');

通过类似以下得到拟传递变量的值。

dwGetProp(Self,'interactionvalue');

* **dwGetMobileAvailHeight**取得移动端实际可用高度函数

可以通过类似以下代码得到

Height := dwGetMobileAvailHeight(self);

* **dwSetMobileMode**设置当前屏幕显示模式为移动应用模式

function dwSetMobileMode(AForm:TForm;ADefaultWidth,ADefaultHeight:Integer):Integer;

其中:

ADefaultWidth,ADefaultHeight为电脑浏览时的默认大小。如果有一项为0，则为当前屏幕大小（只设置宽度，不设置高度）如：

//设置当前屏幕显示模式为移动应用模式.如果电脑访问，则按414x736（iPhone6/7/8 plus）显示

dwSetMobileMode(self,414,736);

* **dwUpload**文件上传函数

function dwUpload(AForm:TForm;AAccept,ADestDir:String):Integer;

参数

AForm : 窗体本身，一般用self

AAccept : 文件上传类型控制，类似"image/gif, image/jpeg"

ADestDir : 上传目录，为空时上传到服务器upload目录，有值时上传到指定目录，支持子目录，类似：'media/doc/dms/'

事件:

上传开始时，自动触发窗体的OnStartDock事件。注意：此事件激活时

dwGetProp(Self,'interactionmethod')的值为\_\_upload\_start

上传完成后，自动触发窗体的OnEndDock事件

可以通过dwGetProp(self,'\_\_upload')得到上传成功的真实文件名（如果同名文件存在，则会自动重命名），不带路径

如果在弹出式窗体中，则使用类似以下的代码

dwGetProp(TForm(self.Parent),'\_\_upload')

可以通过dwGetProp(self,'\_\_uploadsource')得到上传时原始文件名，不带路径

* **dwUploadPro**文件上传函数

function dwUpload(AForm:TForm;AAccept,AdestDir,ACapture:String):Integer;

参数:

AForm : 窗体本身，一般用self

AAccept : 文件上传类型控制，类似"image/gif, image/jpeg"

ADestDir : 上传目录，为空时上传到服务器upload目录，有值时上传到指定目录，支持子目录，类似：'media/doc/dms/'

Acapture：这个 html 的媒体属性允许开发者调用一些设备的媒体功能，主要常用值有：

* user : 前置摄像头
* camera : 相机
* environment ：后置摄像头
* 空值 ：调用麦克风和摄像

事件:

上传开始时，自动触发窗体的OnStartDock事件。注意：此事件激活时

dwGetProp(Self,'interactionmethod')的值为\_\_upload\_start

上传完成后，自动触发窗体的OnEndDock事件

可以通过dwGetProp(self,'\_\_upload')得到上传成功的真实文件名（如果同名文件存在，则会自动重命名），不带路径

如果在弹出式窗体中，则使用类似以下的代码

dwGetProp(TForm(self.Parent),'\_\_upload')

可以通过dwGetProp(self,'\_\_uploadsource')得到上传时原始文件名，不带路径

* **dwPrint**打印函数

function dwPrint(ACtrl:TControl):Integer;

参数

ACtrl为拟打印的控件，可以是DBGrid1或Button\_Edit等，全界面打印用self

* **全局刷新**

如果动态创建或删除了控件，可以采用以下代码进行全局刷新

DockSite := True;

* **指定控件强制刷新**

如果非浏览器的其他渠道更新了控件状态，可以采用以下代码对指定控件刷新

Memo1.ParentCustomHint := False; //假设该控件为Memo1

# 开发DeWeb控件

开发控件非常简单，只需要开发一个适合需要的DLL即可。

注意几点：

1. 命名：工程名必须以"dw"+"控件类名"，如：dwTButton.dpr, dwTEdit.dpr等；
2. 源码存放目录：Source\dwVcls
3. 工程输出目录：Source\RunTime\Vcls\
4. uses的第1个单元必须是sharemem;
5. 包含以下函数，分别为：

function dwGetExtra(ACtrl:TComponent):String;stdCall;

function dwGetEvent(ACtrl:TComponent;AData:String):String;StdCall;

function dwGetHead(ACtrl:TComponent):String;StdCall;

function dwGetTail(ACtrl:TComponent):String;StdCall;

function dwGetData(ACtrl:TComponent):String;StdCall;

function dwGetMethod(ACtrl:TComponent):String;StdCall;

1. dwGetExtra用于在html中增加对第三方网页控件的支持。

目前DeWeb中默认采用element-ui进行处理。

如果使用element-ui中的控件，则代码如下（返回一个空的JSON字符串数组）：

Result := '[]';

如果element-ui不存在某控件，比如TChart，就需要引入第三方UI库。

在dwGetExtra中可以增加以下代码。

Result := '["<script src=\"dist/charts/echarts.min.js\"></script>",'

+'"<script src=\"dist/charts/lib/index.min.js\"></script>",'

+'"<link rel=\"stylesheet\" href=\"dist/charts/lib/style.min.css\">"]';

1. dwGetEvent用于处理控件事件

有2个参数

ACtrl:TComponent，控件自身

AData:String，控件信息，为JSON字符串，一般应先转换为JSON对象，比如joData。其中joData.e为事件名称，如“onchange”或“onclick”等（大小写敏感）；joData.v为值，如Edit的Value,或CheckBox的Checked等.

在这个函数中，需要根据这2个参数，运行相应的事件，如：

function dwGetEvent(ACtrl:TComponent;AData:String):String;StdCall;

var

joData : Variant;

begin

//

joData := \_Json(AData);

if joData.e = 'onclick' then begin

if Assigned(TButton(ACtrl).OnClick) then begin

TButton(ACtrl).OnClick(TButton(ACtrl));

end;

end else if joData.e = 'onenter' then begin

if Assigned(TButton(ACtrl).OnEnter) then begin

TButton(ACtrl).OnEnter(TButton(ACtrl));

end;

end else if joData.e = 'onexit' then begin

if Assigned(TButton(ACtrl).OnExit) then begin

TButton(ACtrl).OnExit(TButton(ACtrl));

end;

end;

end;

1. dwGetHead用于生成当前控件改造时的头部信息，需要根据控件对应的VUE控件来组成字符串。
2. dwGetTail用于生成当前控件改造时的尾部信息，一般与dwGetHead中的HTML要素对应即可。
3. dwGetData用于当前控件的各项关键控制参数值，比如控件的Left,Top,Text等。
4. dwGetMethod用于当前控件的事件控制值，

# Form

* 目前主窗体Name应为Form1，其他窗体Name应以"form"开头，

如Form2,Form3,Form\_VIP,...

* 有2种模式：

1. 如果TransparentColor=False，则为全色模式，如图：



1. 如果TransparentColor=True，则为双色模式，如图：



此时，TransparentColorValue为网页底色，Color为主体部分颜色

* 属性VertScrollBar.Visible决定当窗体高度不够时是否自动显示滚动条
* OnCreate代码有效
* OnShow代码有效

在OnShow中可以通过dwGetProp(Self,'params')得到URL中的参数

* 如果把工程名改为default.dpr，则可以URL中不输入http://127.0.0.1:8080/default, 直接通过<http://127.0.0.1:8080>即可访问
* ip可以通过dwGetProp(Self,'ip')得到（目前好像不准确）；
* AlphaBlend=True时可以通过DeWebserver.exe的Html按钮生成纯html文件，可用于非DeWeb环境；
* 在代码中设置

self.DockSite :=True;

可以在动态创建和删除控件时刷新整个窗体；

**注**：不要用Form1.DockSite:=True;

* 多平台适配代码在OnMouseUp中。

X,Y分别表示当前设备的Width/Height；

Button : mbLeft : 屏幕纵向, mbRight:屏幕横向；

Shift：ssShift, ssAlt, ssCtrl,ssLeft, ssRight,

分别对应0:未知/1:PC/2:Android/3:iPhone/4:Tablet

另外，浏览窗体的

screenWidth可以通过dwGetProp(Self,'screenwidth')得到；

screenHeight可以通过dwGetProp(Self,'screenheight')得到；

innerWidth可以通过dwGetProp(Self,'innerwidth')得到；

innerHeight可以通过dwGetProp(Self,'innerheight')得到；

clientWidth可以通过dwGetProp(Self,'clientwidth')得到；

clientHeight可以通过dwGetProp(Self,'clientheight')得到；

物理分辨率宽度可以通过dwGetProp(Self,'truewidth')得到；

物理分辨率高度可以通过dwGetProp(Self,'trueheight')得到；

设置像素比可以通过dwGetProp(Self, 'devicepixelratio')得到；

其中：

screenWidth/screenHeight为屏幕分辨率。 注意：移动端为虚拟屏幕分辨率

innerWidth/innerHeight为可视区的宽高，包括了滚动条的宽度

clientWidth/clientHeight为可视区的宽高，不包括了滚动条的宽度

* 页面滚动事件代码在OnHelp中。

Data 表示当前滚屏的顶部，例如：设置

Label1.Top := Data；

可以将当前Label1一直在顶部

如果想一直在底部, 需要取得浏览器高度, 需要根据设备是移动端还是PC端分别进行处理。

如果是PC端， 则需要用到clientwidth/clientheight

如果是移动端， 则需要用到screenwidth/screenheight

* 支持通过修改hint属性引入第三方js/css.下面的例子中当前Form对应的应用会引入2个第三方js/css

例如：

{

"head": [

"<script src=\"http: //aaa.com/a.js\" type=\"text/javascrip\" > < /script>",

"<link rel=\"stylesheet\" type=\"text/css\" href=\"dist/b.css\" /> "

]

}

* 关于多窗体，如何ShowModal，需要先把Form2变量定义Form1的public中，然后在dpr中中手动创建

Form1.Form2 := TForm2.Create(Form1);

Form1.Form2.Parent := Form1;

此窗体内的按钮关闭ShowModal的窗体可以用类似以下代码

dwRunJS('this.'+self.Name+'\_\_vis=false;',self);

* 如果需要多个FORM显示在首页的PageControl的多个TabSheet中，可以如下设置：

将Form1,Form2,Form2,...定义到Form1的public中

在dpr中类似写

function dwLoad(AParams:String;AConnection:TADOConnection;AApp:TApplication;AScreen:TScreen):TForm;stdcall;

var

sDir : string;

begin

//

Application := AApp;

Screen := AScreen;

//

//

Form1 := TForm1.Create(nil);

Form1.gsMainDir := ExtractFilePath(Application.ExeName);

//

Form1.Form2 := TForm2.Create(Form1);

Form1.Form2.Parent := Form1.TabSheet2;

Form1.Form2.HelpKeyword := 'embed';

//

Form1.Form3 := TForm3.Create(Form1);

Form1.Form3.Parent := Form1.TabSheet3;

Form1.Form3.HelpKeyword := 'embed';

//

Form1.Form4 := TForm4.Create(Form1);

Form1.Form4.Parent := Form1.TabSheet4;

Form1.Form4.HelpKeyword := 'embed';

//Form1.ADOQuery1.Connection := AConnection;

Result := Form1;

end;

**注：**

* 不要使用全局变量
* 多窗体系统需要把所有非主窗体作为Form1的子对象，放到private或public中；其他窗体以Form1为Owner创建,也就是类似      
   Form1.Form2   := TForm2.Create(Form1);//括号内应为主窗体的名称
* 在程序中不能直接使用窗体变量(如：Form1,Form2,...)进行操作，必须采用self来操作(除以上的创建外)；
* 窗体操作操作本单元的代码用self代替；
* 如果Form2需要操作Form3, 可以用如

TForm(self.Owner).Form3.Label3.Caption := 'new';

* 公共函数库需要用来窗体，建议把窗体作为一个输入参数

# BitBtn(用于选择拟上传的文件/图像/视频)

* 响应OnEnter事件
* 响应OnExit事件
* 设计阶段通过更改Hint来改变颜色(默认为空或无type)

分别为

{"type":"primary"}

{"type":"success"}

{"type":"info"}

{"type":"warning"}

{"type":"danger"}

{"type":"text"}

* 设计阶段通过更改Hint来改变样式(默认为空)

分别为

{"style":"plain"}

{"style":"round"}

{"style":"circle"}

* 设计阶段通过更改Hint来改变图标(默认为无)

例：

{"icon":" el-icon-star-off"}

各图标名称见附件或<https://element.eleme.cn/#/zh-CN/component/icon>

* 如果无accept，则默认为上传文件

{"accept":"image/\*"}表示上传图片；

{"accept":"video/\*"}表示上传视频；

可以同时多个如

{"accept":"image/gif, image/jpeg"}

也可以直接用扩展名

如{"accept":".doc,.docx,.xls,.xlsx,.pdf"}

常用选项

accept="application/msword"  
accept="application/pdf"  
accept="application/poscript"  
accept="application/rtf"  
accept="application/x-zip-compressed"  
accept="audio/basic"  
accept="audio/x-aiff"  
accept="audio/x-mpeg"  
accept="audio/x-pn/realaudio"  
accept="audio/x-waw"  
accept="image/gif"  
accept="image/jpeg"  
accept="image/tiff"  
accept="image/x-ms-bmp"  
accept="image/x-photo-cd"  
accept="image/x-png"

* 如果无capture，则默认为使用文件管理器、相册或相机

{"capture":"camera"}表示使用相机

* 在Hint中设置{"auto":1}，则可以选择后自动上传
* 在Hint中采用类似设置"dir":"mydir/images"，则可以将文件上传到服务器的mydir\images目录(**注意：在Hint中设置时多级目录需要用/，而不是\**)；
* 注意图标和颜色可以同时配合使用，例：{"type":"success","icon":" el-icon-star-off"}
* 开始上传自动激活当前控件的OnStartDock事件  
  上传完成自动激活当前控件的OnEndDock事件
* 可以通过dwGetProp(self,'\_\_upload')得到上传成功的真实文件名（如果同名文件存在，则会自动重命名）

如果在弹出式窗体中，则使用类似以下的代码

dwGetProp(TForm(self.Parent),'\_\_upload')

* 可以通过dwGetProp(self,'\_\_uploadsource')得到上传时原始文件名
* 实际上传时可以通过

dwRunJS('this.dwInputSubmit('+IntToStr(Self.Handle)+',"BitBtn1");',self);

如果在弹出式窗体中，则使用类似以下的代码

dwRunJS('this.dwInputSubmit('+IntToStr(Self.Handle)+',"'+Self.Name+'\_\_BitBtn1");',self);

* 注意：除双引号内的部分，不要使用全角符号！，请特别注意引号和逗号

# Button

* 响应OnClick事件
* 响应OnEnter事件
* 响应OnExit事件
* 设置ElevationRequired为True时，点击一次时会自动禁用该按钮，以防止多次点击。可以在按钮事件加入类似代码以解除禁用

Button1.ParentCustomHint := False;

或者以下

dwRunJS('this.'+dwPrefix(Button1)+Button1.Name+'\_\_dis=false;',self);

以上2种代码使用一种即可，其中第2种方法更高效一些。

* 设计阶段通过更改Hint来进行配置
* 改变按钮样式(默认为空或无type)

分别为

{"type":"primary"}

{"type":"success"}

{"type":"info"}

{"type":"warning"}

{"type":"danger"}

{"type":"text"}

* 自定义颜色

{"backgroundcolor":"rgb(51,123,208)" }

其中：backgroundcolor为按钮背景色

* 设计阶段通过更改Hint来改变样式(默认为空)

分别为

{"style":"plain"}

{"style":"round"}

{"style":"circle"}

* 设计阶段通过更改Hint来改变图标(默认为无)

例：

{"icon":" el-icon-star-off"}

右侧显示图标用（如果此项创建时没有设置，后面不能再设置了）

{"righticon":" el-icon-star-off"}

各图标名称见附件或<https://element.eleme.cn/#/zh-CN/component/icon>

* 通过更改Hint来改变圆角半径

{"radius":"5px"}

{"radius":"5px 5px 0px 0px"}

数值有三种表示方法：px、%、em，一般推荐在用px；

如果你在 border-radius 属性中只指定一个值，那么将生成 4 个 圆角。

但是，如果你要在四个角上一一指定，可以使用以下规则：

* 四个值: 第一个值为左上角，第二个值为右上角，第三个值为右下角，第四个值为左下角。
* 三个值: 第一个值为左上角, 第二个值为右上角和左下角，第三个值为右下角
* 两个值: 第一个值为左上角与右下角，第二个值为右上角与左下角
* 一个值： 四个圆角值相同



* 注：按钮仅有图标，没有文字标题时，可以通过设置Cancal为True, 这样可以使图标居中。缺点是不能动态设置文字标题。
* 常用的左侧有圆角，右侧没有，则可以

{"radius":"10px 0 0 10px"}

如图



* 左侧无圆角，右侧有，则可以类似设置hint

{"radius":"0 10px 10px 0"}

如图



* 注意图标和颜色可以同时配合使用，例：{"type":"success","icon":" el-icon-star-off"}
* **注意**：除双引号内的部分，不要使用全角符号！，请特别注意引号和逗号
* 本地执行javascript代码（**注：本功能需要熟练掌握javascript知识，请谨慎使用！**）

可以在Hint中添加可在本地执行的javascript代码，执行时机分别o为OnEnter,OnExit,OnClick

类似：

{"onenter":"this.Button1\_\_cap='enter';","onexit":"this.Button1\_\_ cap ='exit';"}

表示当鼠标进入时，置Button1的caption为enter；

当鼠标退出时，置Button1的caption为exit；

或者

{"onclick":"this. Button1\_\_cap='click';"}

表示当鼠标单击时，置Button1的caption为click；

在Hint中如下设置，可以在点击时启动进度条

{"onclick":"this.dwloading=true;"}

可以控制的变量有：

Button1\_\_top : 上边距

Button1\_\_lef : 左边距

Button1\_\_wid : 宽

Button1\_\_hei : 高

Button1\_\_vis : 可见性

Button1\_\_cap : 标题

其中Button1需要根据当前控件名相应更改，支持在控件A中控制另一控件B的属性。

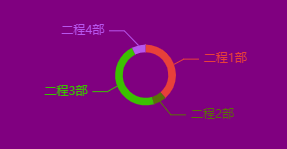


# Calendar

* 支持OnClick事件
* 不支持OnChange事件

# Chart

* 支持TLineSeries/TBarSeries/THorizBarSeries/TAreaSeries/TFastLineSeries/TPieSeries
* 每个Chart只支持一种Series，即所有Series按第一个Series的类型处理；
* 同一Chart的所有Series需要按同一X值添加数据，且数据个数相同；
* ShowHint用来指示是否显示提示框；
* BackWall.Pen.Color用来指示背景边框颜色，clNone为透明；
* Legend.Visilbe用来指示是否显示Legend;
* Legend.Alignment指示Legend位置
* Legend.Invert用来指示是横向（False）和纵向排列（True）;
* Legend.ColorWidth用来指示当Legend在左边或右边时所占的宽度；
* Legend.Pen.Color用来指示Legend的文本颜色；
* Color用来配置背景色，clNone为透明；
* 当为饼图Pie时,设置series的tag为1027，即可显示为环图；



# CheckBox

# ComboBox

* 高度可以通过设置Hint为{"height":35}来实现；
* 高度也可以通过改变字体来实现。但实际显示效果中字体还是默认的，仅高度改变
* 指定ItemIndex时，需要手动指定Text，类似：

ComboBox1.ItemIndex := 1;

ComboBox1.Text := ComboBox1.Items[ComboBox1.ItemIndex];

* 取得ItemIndex时，需要手动取得，类似：

procedure TForm1.Button\_GetRowClick(Sender: TObject);

var

I : Integer;

iRes : Integer;

begin

//

iRes := -1;

for I:=0 to ComboBox1.Items.Count-1 do begin

if ComboBox1.Items[I] = ComboBox1.Text then begin

iRes := I;

break;

end;

end;

//

ComboBox1.ItemIndex := iRes;

//

dwShowMessage('ItemIndex : '+IntToStr(ComboBox1.ItemIndex),self);

end;

# DateTimePicker

# DBGrid

* 只能显示，如果需要编辑，可以用Edit/ComboBox等结合代码完成；
* 行高为HelpContext值
* 如果数据库中日期时间型不能区分日期时间型/日期型/时间型，可以设置该字段的tag来区分。其中默认为日期时间型，格式YYYY-MM-DD hh:mm:ss

当Tag为1时，为日期型，格式YYYY-MM-DD；

当Tag为2时，为时间型，格式hh:mm:ss；

# DBGrid（西风增强版）

* **注**：用作验证码时需要设置其HelpKeyword='ww'
* 字段列可以采用普通的fieldname来定义列；
* Ctl3D为True表示为正常表格；为False时显示为单列显示模式，一般用于手机显示；
* Options中dgEditing为真时显示编辑框；
* ParentBidiMode为False时自动按比例缩放，否则不缩放；
* 字段也可以采用通过Title.Caption设置成JSON格式字符串来进行增强设置
  + 默认字段类型为string，等同于{"type": "string"}

如果有list项，则编辑时可以选择

{"list": ["BeiJing","NewYork","Rio","Paris"]}

* + 设置为多选类型，值为{"type": "check"}

其中：

type为当前列的显示类型

* + 设置为序号列，值为{"type": "index","caption":"序号"}

其中：

type为显示类型

caption为标题

**注**：该序号为行号，非数据库id号

* + 设置为图片列，值为{"type":"image","caption":"相片","fieldname":"photo", "format":"media/images/mm/%s.jpg","dwstyle":"border:solid 1px #ddd;border-radius:8px;width:32px;height:32px;fill:fill;"}

其中：

type为显示类型

caption为标题

fieldname为对应字段名

format为实现显示图片存放位置与字段的对应的关系。上例中，如果photo值aaa,则图片存放位置为media/images/mm/aaa.jpg

dwstyle为图片的显示样式，上例中为：显示1像素颜色为#ddd的边框；圆角半径为8像素；宽32像素；高32像素；拉伸缩放显示。更多样式请参考css相关知识

* + 设置为个性化布尔型，值为{"caption": "性别","type": "boolean",

"list": ["汉子","美女"],"fieldname": "sex"}

其中：

type为显示类型

caption为标题

fieldname为对应字段名

list为当值为True/False显示的文本，例子中如字段sex为True时显示“汉子”，否则显示“美女”

* + 设置为日期时间型，值为{"type":"date","caption":"出生日期", "fieldname":"birthday", "format":"YYYY-MM-DD"}

其中：

type为显示类型

caption为标题

fieldname为对应字段名

format为当值的显示样式，具体用FormatDateTime函数实现

* + 设置为浮点型，值为{"type":"float","format":"%n"}

其中：

type为显示类型

format为当值的显示样式，具体用Format函数实现

* + 设置为进度条型，值为{"type":"progress","caption":"工作进度","fieldname":"progress","total":100}

其中：

type为显示类型

caption为标题

fieldname为对应字段名

minvalue为最小值，默认为0

maxvalue为最大值，默认为100

total为当前数据的最大值，应该显示进度为数据库字段的值/total\*100%

* + 设置为操作按钮型，值为{"type":"button","caption":"操作","list":[["审核", "primary"],["查看","success"],["删除","danger"]]}

其中：

type为显示类型

caption为标题

list为按钮组信息，其中每项(如["审核", "primary"])中第一个值为按钮标题，第二个值为按钮样式，有：primary/success/info/warning/danger

* **多表头设置**

多表头采用TDBGrid控件的Hint中的merge二维数组来设置。其中merge中的一个元素表示一个合并设置。如

{"merge":[

[3,5,8,"基本信息"],

[2,7,8,"通信地址"]

]}

以上表示有2个合并项。

第1个元素[3,5,8,"基本信息"]表示在3楼合并，从第5列合并到第8列（从0开始），合并后标题为"基本信息"；

第2个元素[2,7,8,"通信地址"]表示在2楼合并，从第7列合并到第8列（从0开始），合并后标题为"通信地址"

最终效果如下



* **标题换行**

可以在标题中增加<br/>以换行

如“出生日期”，可以写成“出生<br/>日期”，即可显示为2行

* **颜色设置**
* hover颜色 ，默认为#f5f5f5，可通过TDBGrid控件的Hint中的hover属性设置
* 记录位置颜色，默认为#ffe，可通过TDBGrid控件的Hint中的record属性设置
* 标题栏背景色，默认为FixedColor

例如：{"hover":"#800","recode":"#008"}

* **汇总设置**

通过TDBGrid控件的Hint中的summary属性设置

summary属性为数组类型，如：

{"summary":["汇总",[6,"sum","合计：%.0f"],[6,"avg","平均：%.0f"],[9,"max","最大：%.0f%%"],[9,"min","最小：%.0f%%"]]}

按格式展开显示如下：

{

"summary": [

"汇总",

[6,"sum","合计：%.0f"],

[6,"avg","平均：%.0f"],

[9,"max","最大：%.0f%%"],

[9,"min","最小：%.0f%%"]

]

}

* **高度设置**
* 数据行高，默认为35，可通过TDBGrid控件的Hint中的rowheight属性设置
* 表头行高，默认为35，可通过TDBGrid控件的Hint中的headerheight属性设置

例如：{"rowheight":50,"headerheight":40}

* **升序排序**

需要在字段Title.Caption的JSON对象中增加"sort":1，例如：

{"type":"progress","caption":"工作进度","fieldname":"progress","total":100,"sort":1}

操作时自动激活TDBGrid控件OnEndDock事件，其中参数

X固定为1

Y为列序号，从0开始

* **逆序排序事件**

需要在字段Title.Caption的JSON对象中增加"sort":1，同上。

操作时自动激活TDBGrid控件OnEndDock事件，其中参数

X固定为2

Y为列序号，从0开始

* **CheckBox选中事件**

需要在字段Title.Caption的JSON对象中设置为多选类型。

操作时自动激活TDBGrid控件OnEndDock事件，其中参数

X固定为3

Y为选中情况，-1表示批量选中，0表示批量清除，其他值为选中行记录（从1开始）。

具体所有的选中记录情况，保存在Hint中的\_\_selection中（注意前缀是2个下划线）。

默认保存数据表第一值字段的值。也可以通过在Hint中增加primarykey

改变选中是保存的字段的值，注意该字段值应没有重复项。

保存的\_\_selection如：

{"\_\_selection":["1","5","6","17","25"]}

* **保存前事件（点击保存按钮执行保存操作前）**

操作时自动激活TDBGrid控件OnEndDock事件，其中参数

X固定为4

Y为当前记录号（对应TDataSet的RecNo）。

* **保存后事件（点击保存按钮执行保存操作完成后）**

操作时自动激活TDBGrid控件OnEndDock事件，其中参数

X固定为5

Y为当前记录号（对应TDataSet的RecNo）。

**注意**： 一次保存事件会激活2次OnEndDock事件，激活时机为保存（post）前和保存（post）后

* **取消事件（进入编辑状态后不保存）**

操作时自动激活TDBGrid控件OnEndDock事件，其中参数

X固定为6

Y固定为0。

* **操作按钮事件**

需要在字段Title.Caption的JSON对象中设置为操作按钮型。

操作时自动激活TDBGrid控件OnEndDock事件，其中参数

X固定为100+列序号（从0开始），比如103表示第3列（从0开始）有按钮事件

Y为按钮序号，从0开始

* **编辑状态时编辑框的编辑事件**

操作时自动激活TDBGrid控件OnEndDock事件，其中参数

X固定为7

Y为列序号（从0开始）

选中的数值为当前字段的AsString(未保存)，注意：日期型为YYYY-MM-DD

* **筛选事件（预留）**

需要在字段Title.Caption的JSON对象中增加"filter":1，且增加选项列表，如：

{"caption":"省份","fieldname":"province","filter":1,"list":["北京","上海","湖北","陕西","山东"]}

操作时自动激活TDBGrid控件OnEndDock事件，其中参数

X固定为300+列序号（从0开始），比如303表示第3列（从0开始）有筛选事件

Y为选中情况，用其对应二进制表示筛选选项的选中情况。

例如：可筛选的选项有5项，["北京","上海","广州","深圳","西安"]。

如Y值为7，其对应的二进制为111，则表示采用["北京","上海","广州"]进行筛选；

如Y值为25，其对应的二进制为11001，则表示采用["北京","上海","西安"]筛选

# Edit

* 支持OnChange事件
* 支持OnEnter事件
* 支持OnExit事件
* 支持OnMouseEnter事件
* 支持OnMouseLeave事件
* **注意**：
* Edit的Text值中有“或<或>时会影响解析，需要用\"或\<或\>来代替；同时在代码中注意：处理时多一个\，而在前端显示时无此\
* Edit的Text值前后的空格会被忽略
* 可以通过设置Hint来前置图标，如{"prefix-icon":"el-icon-s-home"}
* 可以通过设置Hint来后置图标，如{"suffix-icon":"el-icon-s-home"}
* 可以通过设置Hint来设置输入提示，如{"placeholder":"请输入用户名"}
* 可以组合，如：{"placeholder":"请输入用户名","prefix-icon":"el-icon-s-home","suffix-icon":"el-icon-search"}
* 为了保持输入样式的一致性，在Delphi的IDE中修改Edit的Font.Size不能更新web显示时字体大小。如果特别需要，可以通过采用在Hint中设置类似{"dwstyle":"font-size:24px;"}的写法，设置字体大小等
* **注意**：设置Hint时除双引号内的部分，不要使用全角符号！，请特别注意引号和逗号
* **注意：**OnChange事件中必须有代码，可以是一个空的注释

# Edit（用作Rate评分控件）



* **注**：用作验证码时需要设置其HelpKeyword='rate'
* 支持OnChange事件
* 取值 ：

直接取类似 Edit1.Text

* 设置：

直接赋值类似 Edit1.Text

* 注意：Text应为0-5之间的带1位小数的字符串，如：1.6,3.5等

# Image

* 可以采用设置Hint为类似{"radius":"50%"}或{"radius":"5px"}来设置圆角
* 采用设置Hint中的href来设置新窗口跳转网址，例如：

{"href":"/hello"}，

hrefself来设置当前窗口跳转网址，例如：

{"hrefself":"/hello"}

* 采用Hint中的src来设置图片源，如：{"src":"media/images/mn/1.jpg"}
* 设置IncrementalDisplay为True时，点击图片可进行放大/旋转预览；
* 如果在程序中动态指定src，由于浏览器缓存的原因可能显示不更新。可以通过清除缓存，也可以通过类似

Hint := '{"src":"media/a.jpg?time='+IntToStr((DateTimeToUnix(Now)-86060)\*1000)+';"}'；

来强制刷新显示。

# Label

* Delphi界面效果和网页效果会有一些区别，宽度和高度最好都留一些余量
* 支持OnClick事件
* 支持OnMouseEnter事件
* 支持OnMouseLeave事件
* 如果需要强制换行，可以在Caption中增加<br>,

如：delphi是一个很棒的开发工具！我喜欢！

可以写成：delphi是一个很棒的开发工具！<br>我喜欢！

# Label（用作艺术字）

* **注**：用作验证码时需要设置其HelpKeyword为'cool'
* 用HelpContext区分多种效果

0：空心文字

1：空心+阴影

2：空心+叠加

3：立体文字

4：发光文字

5：彩色渐变文定

6：动态变色文字

7：阴影文字

8：霓虹灯效果

9：实心文字

# Label（用作验证码）

* **注**：用作验证码时需要设置其HelpKeyword为'captcha'
* 注：把AutoSize 设置为false
* 直接读Caption得到验证码
* 点击可直接刷新，无需要代码
* 用类似以下代码主动刷新验证码(uses dwBase)

dwRunJS('document.getElementById("'+dwPrefix(Label1)+Label1.Name+'").click();',self);

# ListBox

* 支持Enabled/Visible/Left/Top/Width/Height
* 支持OnClick事件;
* 支持单选/多选;
* OnShow后动态改变Items需要执行DockSite:=True;
* Hint中设置{"dwstyle":"overflow-y:hidden"}可以去掉垂直滚动条; {"dwstyle":"overflow-y:auto"} 可以根据当前元素自动显示或不显示滚动条

# MainMenu

* 采用dwSetCompLTWH函数设置Left/Top/Width/Height （默认：0，0，600，50）
* 采用dwSetMenuDefault函数设置默认选中的菜单项，

如：dwSetMenuDefault(MainMenu,'1-0-2');

注：序号从0开始，每层之间用-隔开

* 默认Left:0;Top:0;Width:600;Height:38
* AutoHotKeys 必须设置为 maManual
* AutoMerge为是否折叠菜单
* 事件必须写在菜单项的OnClick中
* ParentBiDiMode为True时为水平显示,为True时垂直显示
* 不要使用属性Tag/DesignInfo
* 在子菜单项的Hint中设置类似{"dwstyle":"padding-left:30px;"}可以调节菜单左边距；
* 第1个菜单项的Hint用于菜单配置信息
* 其中前景色，字体色等信息；

{"background-color":"#FF0000","text-color":"#0000FF","active-text-color":"#FFFFFF"}

* 如果有dwloading项，如Hint为{"dwloading":"1"}，则点击当前MainMenu中所有菜单项时先显示”loading...“标识。关闭该标识可以在菜单项执行事件中加入以下代码即可在事件执行完成后关闭”loading...“标识。

dwRunJS('this.dwloading=false',self);

* ImageIndex见dwVars中的dwIcons常量和ElementUI中图标的定义，网址：<https://element.eleme.cn/#/zh-CN/component/icon>





# MediaPlayer

本控件有2种模式：

* 普通模式（HelpKeyword <> 'm3u8'）,支持的视频格式有：
* MP4 = MPEG 4文件使dao用 H264 视频zhuan编解码器和AAC音频shu编解码器
* WebM = WebM 文件使用 VP8 视频编解码器和 Vorbis 音频编解码器
* Ogg = Ogg 文件使用 Theora 视频编解码器和 Vorbis音频编解码器
* 视频流模式（在控件中设置HelpKeyword = 'm3u8'），支持m3u8等视频流格式，可以播放在线电视。

具体的属性有

* AutoRewind表示是否循环播放；
* Enabled表示是否自动播放（一些浏览器不支持）,**注意**不是AutoOpen；
* FileName表示当前视频位置，注意需要采用网络位置，如./ocean.mp4
* 播放，采用MediaPlayer1.EnabledButtons := [btPlay];
* 暂停，采用MediaPlayer1.EnabledButtons := [btPause];
* 设置时刻，采用类似MediaPlayer1.HelpContext := 6; //表示播放位置为6-1秒处

# MediaPlayer（EasyPlayer）

功能说明

* 支持 MP4 播放
* 支持 m3u8/HLS 播放;
* 支持 HTTP-FLV/WS-FLV 播放;
* 支持直播和点播播放;
* 支持播放器快照截图;
* 支持点播多清晰度播放;
* 支持全屏或比例显示;
* 自带的 flash 支持极速和流畅模式;
* 自带的 flash 支持 HTTP-FLV 播放;
* 支持重连播放；

**注**：用作验证码时需要设置其HelpKeyword='easyplayer'

支持的Delphi属性有：

* AutoRewind表示是否循环播放；
* AutoOpen表示是否自动播放（一些浏览器不支持）；
* FileName表示当前视频位置，注意需要采用网络位置，如

https://ablive.easyvaas.com/live/633dc2c85fc2050afb407a9e7d22f65b.flv

支持的Hint属性有：

| **参数** | **说明** | **类型** | **默认值** |
| --- | --- | --- | --- |
| video-title | 视频右上角显示的标题 | String | - |
| snap-url | 视频封面图片 | String | - |
| live | 是否直播, 标识要不要显示进度条 | Boolean | true |
| speed | 是否显示倍速播放按钮。注意：当live为true时，此属性不生效 | Boolean | true |
| alt | 视频流地址没有指定情况下, 视频所在区域显示的文字 | String | 无信号 |
| muted | 是否静音 | Boolean | false |
| aspect | 视频显示区域的宽高比 | String | 16:9 |
| isaspect | 视频显示区域是否强制宽高比 | Boolean | true |
| loading | 指示加载状态, 支持 sync 修饰符 | String | - |
| fluent | 流畅模式 | Boolean | true |
| stretch | 是否不同分辨率强制铺满窗口 | Boolean | false |
| show-custom-button | 是否在工具栏显示自定义按钮(极速/流畅, 拉伸/标准) | Boolean | true |
| isresolution | 是否在播放 m3u8 时显示多清晰度选择 | Boolean | false |
| isresolution | 供选择的清晰度 "yh,fhd,hd,sd", yh:原始分辨率 | fhd:超清，hd:高清，sd:标清 | - |
| resolutiondefault | 默认播放的清晰度 | String | hd |

参考Hint值：

{"muted":false,"alt":"no valid singal"}

# Memo

* 支持OnChange事件
* 支持OnEnter事件
* 支持OnExit事件
* 支持OnMouseEnter事件
* 支持OnMouseLeave事件

# Memo（用于直接显示HTML代码）

* **注**：用作验证码时需要设置其HelpKeyword='html'
* Memo的Text为拟显示的HTML代码。

**注**：请谨慎使用！ 注意HTML代码的正确性！

# Memo（用于显示ECharts图表）

* **注**：用作验证码时需要设置其HelpKeyword='echarts'
* **注：**请设置ScrollBars为ssBoth，以避免超长字符串自动换行造成错误！
* Memo的Text为拟显示的echarts对应图表中代码配置的option部分。

# Memo（用于额外引用js/css）

* **注**：用作验证码时需要设置其HelpKeyword='uses'
* **注：**请设置ScrollBars为ssBoth，以避免超长字符串自动换行造成错误！
* Memo的Text为拟额外引用的js/css文件，参考格式如下

<script src="dist/axios.min.js" type="text/javascript"></script>

<link rel="stylesheet" type="text/css" href="dist/theme-chalk/index.css" />

<script src="dist/\_easyplayer/EasyPlayer-element.min.js"></script>

# Memo（用作MarkDown编辑器）

* **注**：用作MarkDown编辑器时需要设置其HelpKeyword='markdown’；

属性对应：

* ReadOnly有效；
* ShowHint表示是否显示工具栏，为True隐藏实时预览；
* ParentBiDiMode表示是否显示编辑器，为True时显示编辑器，否则为仅显示；

Hint扩展属性

* 自定义工具栏：参考

{"toolbaricons":["undo", "redo", "|", "bold", "hr", "|", "preview", "watch", "|", "fullscreen", "info", "testIcon", "testIcon2", "file", "faicon", "||", "watch", "fullscreen", "preview", "testIcon"]}

所有可选项有：

"undo","redo","|","bold","del","italic","quote","ucwords" ,"uppercase","lowercase","|","h1","h2","h3","h4","h5","h6","|","list-ul","list-ol","hr","|","link","reference-link", "image", "code","preformatted-text","code-block","table","datetime","emoji","html-entities","pagebreak","|","goto-line","watch","preview","fullscreen","clear","search","|", "help","info","releaseIcon", "index"

# PageControl

* ParentBiDiMode和ParentShowHint联合配置是否显示边框
* ParentBiDiMode=True且ParentShowHint=True时，则显示全边框，如



* ParentBiDiMode=True且ParentShowHint=False时，则显示顶部边框，如



* ParentBiDiMode=False时，不论ParentShowHint值，都不显示边框，如



* 设置Hint为{"dwattr":"editable"}，可以在TabSheet的标题处显示关闭按钮和新增按钮，当按下按钮时，激活OnEndDock事件，类似：

procedure TForm1.PageControl1EndDock(Sender, Target: TObject; X, Y: Integer);

其中参数：

* X = 1 为新增Tab ;
* X = 0 为删除Tab,待删除Tab的序号为Y（从0开始）
* 设置Hint为{"dwattr":":stretch=true"}

这样所有页面宽度会平均分布

* 设置Hint为{"dwattr":"closable"}，可以在TabSheet的标题处显示关闭按钮，当按下关闭按钮时，激活OnEndDock事件，参数和以上相同。
* 可以设置特定页面显示关闭按钮，即设置TabSheet的Hint为{"dwattr":"closable"}即可。

注意：此处不要设置PageControl的Hint为{"dwattr":"closable"}或{"dwattr":"editable"}

# PageControl（用作时间线）

* **注**：PageControl用作时间线时需要设置其HelpKeyword='timeline’，如下图



* 每个TabSheet为一个时间节点
* Tag为时间值
* Caption为时间节点标题（黑体）
* 可以在Hint中通过类似{"color":"#FF0000"}来设置节点颜色
* 每个TabSheet中可以有若干Label或Memo，按创建顺序显示为标题下面的文本
* TabSheet中不能有其他控件，否则容易造成混乱！！！
* 各时间节点的顺序，根据TabSheet激活的顺序，也就是后激活的TabSheet（时间节点）排在后面。

如果需要正序，可以先执行类似代码  
    for var I := 0 to PageControl1.PageCount-1 do begin  
        PageControl1.ActivePageIndex    := I;  
    end;  
如果需要逆序，可以先执行类似代码  
    for var I := PageControl1.PageCount-1 downto 0 do begin  
        PageControl1.ActivePageIndex    := I;  
    end;

# PageControl（用作轮播组件）

* **注**：PageControl用作轮播组件时需要设置其HelpKeyword='carousel’；
* 每个TabSheet为一个页面

# Panel

* BorderStyle为bsSingle，显示边框，且为圆角；否则无框直角
* Caption属性无效，实现显示时不显示Caption；
* HelpContext用来设置透明度，0：不透明，2~9：半透明，10：完全透明；
* 通过更改Hint来改变圆角半径

{"radius":"5px"}

{"radius":"5px 5px 0px 0px"}

数值有三种表示方法：px、%、em，一般推荐用px；

如果你在 border-radius 属性中只指定一个值，那么将生成 4 个 圆角。

但是，如果你要在四个角上一一指定，可以使用以下规则：

* 四个值: 第一个值为左上角，第二个值为右上角，第三个值为右下角，第四个值为左下角。
* 三个值: 第一个值为左上角, 第二个值为右上角和左下角，第三个值为右下角
* 两个值: 第一个值为左上角与右下角，第二个值为右上角与左下角
* 一个值： 四个圆角值相同
* 可以通过设置其Hint属性dwstyle完成设置Panel单独边框，类似

{"dwstyle":"border-bottom:solid 2px #aaa;border-top:solid 2px #aaa;"}

本例完成实现边框： 显示下边框，样式为solid, 宽度为2，颜色 为#aaa

{"dwstyle":"border:solid 1px #DCDFE6;"}

本例完成实现边框： 显示所有边框（上下左右），样式为solid, 宽度为2，颜色 为#aaa

{"dwstyle":"border-bottom:solid 2px #800;border-top:dashed 1px #008;"}

本例完成实现边框： 显示上边框和下边框，其中下边框样式为solid, 宽度为2，颜色 为#800，

上边框样式为dashed虚线, 宽度为1，颜色 为#008

其实，dwstyle属性还可以实现许多其他的功能，html中div 元素中的style 属性可以实现的功能都可以通过dwstyle来设置实现

* 常用的左侧有圆角，右侧没有，则可以

{"radius":"5px 0 0 5px"}

如图



* 左侧无圆角，右侧有，则可以类似

{"radius":"0 5px 5px 0"}

如图



* **注意** : Panel的Enabled不能导致子控件的Enabled
* 当Color为clNone时，显示为透明色；
* 可以采用dwattr来设置渐变色，如：

从上到下渐变

{"dwstyle":"background-image: linear-gradient(#7A88FF, #7AFFAF);"}



从左到右渐变  
{"dwstyle":"background-image: linear-gradient(to right , #7A88FF, #7AFFAF);"}



* 本地执行javascript代码（**注：本功能需要熟练掌握javascript知识，请谨慎使用！**）

可以在Hint中添加可在本地执行的javascript代码，执行时机分别为OnEnter,OnExit,OnClick

类似：

{"onenter":"Panel1\_\_col='#ccc';","onexit":"Panel1\_\_col='#eee';"}

表示当鼠标进入时，置Panel1的color为#ccc；

当鼠标退出时，置Panel1的color为#eee；

或者

{"onclick":"Panel4\_\_col='#bbb';"}

表示当鼠标单击时，置Panel1的color为#bbb；

可以控制的变量有：

Panel1\_\_top : 上边距

Panel1\_\_lef : 左边距

Panel1\_\_wid : 宽

Panel1\_\_hei : 高

Panel1\_\_vis : 可见性

Panel1\_\_col : 颜色

其中Panel1需要根据当前控件名相应更改，支持在控件A中控制另一控件B的属性。

# Panel（用作iframe，可用来显示pdf）

* BorderStyle为bsSingle时，显示边框；否则无边框;
* Hint中采用src来控制iframe的src，如{"src":"/a.html"}
* 显示pdf时采用类似{"src":"/dist/pdf/web/viewer.html?file=c.pdf"}，其中c.pdf位于runtime/dist/pdf/web目录

# Panel（用作徽章Badge）

* **注**：需要设置其HelpKeyword为badge
* Caption为右上角显示数值，一般可以为32，99+，new等
* 如果仅显示圆点，可以在设置Hint为{"dwattr":"is-dot"}
* 可以通过设置type来设置颜色，类似Hint为{"type":"primary"},其它还有

{"type":"primary"}

{"type":"success"}

{"type":"info"}

{"type":"warning"}

{"type":"danger"}

{"type":"text"}

# Panel（用作SpeedButton）

* **注**：需要设置其HelpKeyword为button
* 采用设置Hint中的src来设置图片源，例如：

{"src":"media/images/kehu.png"}

* 采用设置Hint中的href来设置新窗口跳转网址，例如：

{"href":"/hello"}，

hrefself来设置当前窗口跳转网址，例如：

{"hrefself":"/hello"}

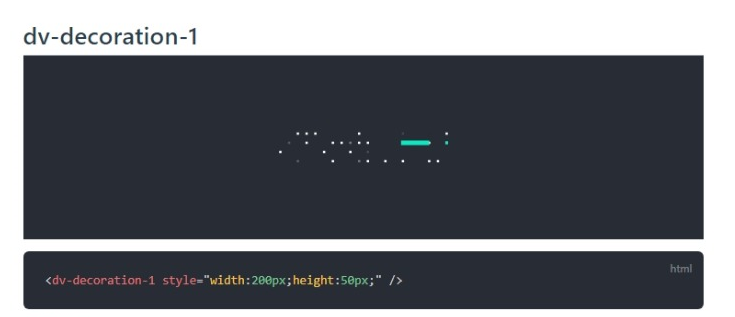
# Panel（用作大屏显示的边框）

* **注**：用作验证码时需要设置其HelpKeyword为dvbox
* **注**： 共有11种效果，通过helpcontext区分，其中默认为效果1，其余对应helpcontext数字对应的效果（其中可以4，5，8可以设置为负值，显示水平反向效果）。效果见

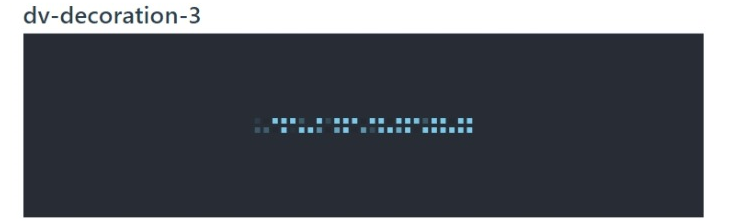
http://datav.jiaminghi.com/guide/borderBox.html#dv-border-box-1

# Panel（用作大屏显示的装饰）

* **注**：用作验证码时需要设置其HelpKeyword为'dvdeco'
* 有两个颜色设置，分别为Color 和 Font.Color，请注意区分；
* 如果有文本显示，颜色默认采用Font.Color，也可以通过设置Hint为类似{"fontcolor":"#0f0"}修改
* 共有12种效果，通过helpcontext区分，其中默认为效果1，其余对应helpcontext数字对应的效果（其中可以2，4，8可以设置为负值，显示水平反向效果）。效果图分别如下：









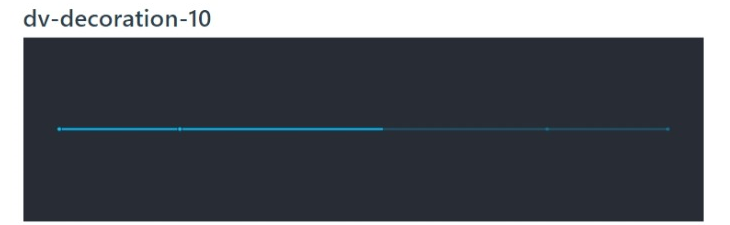




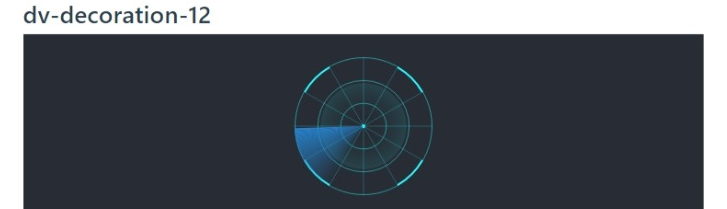












* 点击可直接刷新，无需要代码
* 用类似以下代码主动刷新验证码(uses dwBase)

dwRunJS('document.getElementById("'+dwPrefix(Label1)+Label1.Name+'").click();',self);

# ProgressBar

# RadioButton

# RadioGroup

* Color为选中颜色，建议$00FF9E40

# ScrollBox

* 其内**必须有且仅有一个Panel**，该Panel的align为alTop，其他控件请放到该Panel中；
* 滚动条滚动时自动激活OnEndDock事件，其中X为滚动值，Y为方向，1：向下，-1向上
* 滚动条滚动停止时（200毫秒内无变化）自动激活OnMouseUp事件，其中X为滚动值，
* HelpContext可以用来防止滚动条事件频繁激活，滚动值大于HelpContext时才激活OnEndDock事件。例如：HelpContext为100，也就是当滚动值（相对上次事件时的位置）大于100时才激活 OnEndDock事件
* 主要控制滚动位置，可以使用类似以下代码：

dwRunJS('this.$refs['''+dwFullName(ScrollBox1)+'''].wrap.scrollTop =200;',self);

# Shape(用于扫描二维码)

* Pen.Width为扫描框边框宽度
* Pen.Color为扫描框边框颜色
* 可以通过Hint设置视频窗口的显示情况

如{"dwstyle":"object-fit: contain;"}，其中object-fit 属性可接受如下值：

fill - 默认值。调整替换后的内容大小，以填充元素的内容框。如有必要，将拉伸或挤压物体以适应该对象。

contain - 缩放替换后的内容以保持其纵横比，同时将其放入元素的内容框。

cover - 调整替换内容的大小，以在填充元素的整个内容框时保持其长宽比。该对象将被裁剪以适应。

none - 不对替换的内容调整大小。

scale-down - 调整内容大小就像没有指定内容或包含内容一样（将导致较小的具体对象尺寸）

* 开始扫描采用函数dwSetZXing(AShape,AValue);

输入参数：

AShape类型为TShape,为当前TShape对象；

AValue类型为整型，备用

* 当扫描识别完成后自动激活TShape对象的OnEndDock事件，并通过类似的代码取得识别的值：

dwISO8859ToChinese(dwUnEscape(Shape1.HelpKeyword));

在此代码后，应该加上以下代码以用于二次扫描

Shape1.HelpKeyword := ‘’;

* 可以用类似代码关闭摄像头

dwStopZXing(Shape1,0);

**注**：使用扫描识别二维码/条形码需要服务器采用https

# SpeedButton

用于生成图标居上的按钮

* 采用设置Hint中的src来设置图片源，例如：

{"src":"media/images/kehu.png"}

* 采用设置Hint中的href来设置新窗口跳转网址，例如：

{"href":"/hello"}，

hrefself来设置当前窗口跳转网址，例如：

{"hrefself":"/hello"}

* Flat为False为有外框，为True时为无外框；
* 用Layout来控制类型
* blGlyphLeft 为普通类型



* blGlyphRight 右上显示三角形，颜色为groupindex

(常用红色为255，蓝色为16711680)



* blGlyphTop 右上显示显示Badge标记，字体色为白色，数量为Spacing, 最大限制为Margin，底色通过hint中的类似Button的{"type":"primary"}等控制。



* blGlyphLeft 为右上红点提示



# SpinEdit

* Ctrl3D为False时增减按钮在左右侧；
* 如果需要显示外框可以设置Hint为

{"dwstyle":"border:solid 1px #eee;border-radius:2px"}

# StaticText

TStaticText用于链接，其Hint属性中有3项

* href用来标志链接地址，如：http://www.163.com
* type用于标志链接类型（颜色），选项有：primary,success,warning,danger,info。默认为primay
* target用来设置是否新标签页打开，"\_blank"表示在新窗口打开，"\_self"表示在当前窗口打开，如：{"href":" http://www.163.com ","target":"\_blank"}
* ParentFont=True时为系统配置的字体（字体字号颜色样式），否则为当前设置
* 设置边框，设置Hint为{"dwstyle":"border: 1px solid #DCDFE6;"}

# StringGrid

* Height应为RowCount \* DefaultRowHeight+1，以保持最佳显示效果
* BorderStyle=bsNone时为无边框
* DefaultDrawing=False时如果单元格内容超出长度，则会自动换行。

**注意**：如果内容字数太多，行高可能会被撑开

* 支持OnClick事件, 在点击时激活
* 隐藏列：可将相应列的宽度设为-1即可以隐藏，设为正值即可显示；
* 标题栏（第一行）可以通过在列标题中添加[\*left\*]或[\*center\*]或[\*right\*]设置当前列的对齐属性。列默认为左对齐（仅OnShow之前有效）。
* 标题栏可以通过设置JSON格式的字符串设置复杂列模式，如：

Cells[1,0] := '{"caption":"名称","sort":true,"filter":["净水管道","暖气片","电视机","空调","热水器","LG笔记本"]}';

该行代码表示第1列（从0开始）的标题为：名称；

显示排序按钮；

显示筛选窗口，选项有"净水管道","暖气片","电视机","空调","热水器","LG笔记本"等6个选项，如图



当设置了显示**排序**按钮时，当点击排序按钮时，StringGrid会自动激活OnGetEditMask事件。其中参数：

ACol : Integer ; 为所在列序号（从0开始）；

ARow : Integer; 为排序方向，1为升序，0为降序；

Value : string; 为标识，固定为字符串'sort'

当设置了显示**筛选**项时，当点击筛选或重置按钮时，StringGrid会自动激活OnGetEditMask事件。其中参数：

ACol : Integer ; 为所在列序号（从0开始）；

ARow : Integer; 无意义，预留

Value : string; 为标识，为类似"filter":["电视机","空调"]。前面的"filter":为固定字符串，用于事件标识；方括号内的字符串列表为当前拟筛选的选项列表。当Value为"filter":{表示当前点击了”重置“按钮

* Hint中可以通过设置标题行颜色，如：

{"headerbackground":"#eee"}

* 可以通过类似设置将首列设为CheckBox

StringGrid1.Cells[0,0] := '{"caption":"日期","type":"selection"}';

设置某行的选中状态可用dwSGSetRowChecked 函数，类似

dwSGSetRowChecked(StringGrid1,2,True);

取某行的选中状态可用dwSGGetRowChecked函数，类似

dwSGGetRowChecked(StringGrid1,2)

* TStringGrid的Hint提供一种新的简洁高效(简单粗暴)的方式，即dwstyle  
  DeWeb会把dwstyle的字符直接封装到TStringGrid对应的el-table中，快速实现一些el-table已有的功能  
  如{"dwattr":"border"}表示带边框和竖线  
  {"dwattr":"stripe"}表示带斑马纹  
  {"dwattr":"show-summary"}表示带合计
* 还可以组合，比如：  
  {"dwattr":"border stripe"}表示带边框和竖线,带斑马纹
* 设置单元格样式

function dwSGSetCellStyle(

ASG : TStringGrid;

ARow,ACol : Integer;

BackGroundColor : TColor;

FontName : String;

FontSize : Integer;

FontBold : Boolean;

FontItalic : Boolean;

FontColor : TColor

):Integer;

输入值：

* + ASG : TStringGrid; 为控件名称
  + ARow,ACol : Integer;为行、列ID
  + BackGroundColor : TColor;为背景色
  + FontName : String;为字体名称
  + FontSize : Integer;为字体大小
  + FontBold : Boolean;为字体是否粗休
  + FontItalic : Boolean;为字体是否斜体
  + FontColor : TColor为字体颜色

参考：

dwSGSetCellStyle(StringGrid1,0,0,clGreen,'宋体',20,True,False,clWhite);

* 设置列样式

function dwSGSetCol(

ASG:TStringGrid;

ARow:Integer;

BackGroundColor:TColor;

FontColor:TColor

):Integer;

输入值：

* + ASG:TStringGrid;为控件名称
  + ACol:Integer;为列ID
  + BackGroundColor:TColor;为背景色
  + FontColor:TColor为字体色
* 设置行样式

function dwSGSetRow(

ASG:TStringGrid;

ARow:Integer;

BackGroundColor:TColor;

FontColor:TColor

):Integer;

输入值：

* + ASG:TStringGrid;为控件名称
  + ARow:Integer;为行ID
  + BackGroundColor:TColor;为背景色
  + FontColor:TColor为字体色

# StringGrid(用作大转盘抽奖)

* 需要设置HelpKeyword 为 'luck'
* RowCount为红包数量 \* DefaultRowHeight+1，以保持最佳显示效果
* 列数至少为7列

//第一列,红包标题

SG.Cells[0,I-1] := (I\*10).ToString+'元红包';

//第二列,红包区颜色

if I mod 2 = 0 then begin

SG.Cells[1,I-1] := '#f9e3bb';

end else begin

SG.Cells[1,I-1] := '#f8d384';

end;

//第三列, 红包显示文本

SG.Cells[2,I-1] := (I\*10).ToString+'元红包';

//文本顶部距离,建议10%

SG.Cells[3,I-1] := '10%';

//红包图片

SG.Cells[4,I-1] := 'media/images/luck/'+(I-1).ToString+'.png';

//红包图片宽度, 建议30%

SG.Cells[5,I-1] := '30%';

//红包图片顶部距离 , 建议45%

SG.Cells[6,I-1] := '45%';

* 抽奖完成后自动激活OnEndDock事件,其中参数X为当前奖项所在行序号(从0开始)

# StringGrid(用作股市K线图)

* 需要设置HelpKeyword 为 'vecandle'
* ColCount必须为6
* 第0行为参数名称，建议分别为：‘日期’，‘open’，‘close’，‘lowest’，‘highest’，‘vol’
* 从第1行开始为K线数据，分别为相应值

# Timer

* **！！！注！！！**通过设置DesignInfo来控制是否运行，其中:

1为Enabled=True；0为Enabled=False;不要通过Timer自身的Enabled来控制

# ToggleSwitch



* State为当前状态；
* ThumbColor为选中时的颜色
* SwitchWidth为选择框宽度
* StateCaptions-CaptionOff/CaptionOn为未选中/选中的标题

# TrackBar

# TrackBar（用作Pagination分页）

* **注**：TrackBar用作轮播组件时需要设置其HelpKeyword='page’；
* Max为总数量（页数\*每页数据量）；
* Min建议为0；
* PageSize为每页显示的数据量；
* Position为当前页序号（1为首页）；
* 点击页码（包括上一页，下一页）后会触发OnChange事件
* 可以通过设计Hint的dwattr为设置显示的组件

如{"dwattr":"layout=\"prev, pager, next\""}

表示仅显示“下一页”，“页码”和“下一页”，如图



layout的选项有：sizes, prev, pager, next, jumper, ->, total, slot

* 可以通过设计Hint的dwattr为设置显示背景

{"dwattr":"background"}

如图



* 更多设置请参考

https://element.eleme.cn/#/zh-CN/component/pagination

# TrackBar（西风版Pagination分页组件）

* **注**：TrackBar用作轮播组件时需要设置其HelpKeyword='wwpage’；
* Max为总数量（页数\*每页数据量）；
* Min建议为0；
* PageSize为每页显示的数据量；
* Position为当前页序号（1为首页）；
* 点击页码（包括上一页，下一页）后会触发OnChange事件
* 在Hint中设置类似{"goto":0}可以隐藏跳转部分
* 在Hint中设置类似{"pagesize":0}可以隐藏每页条数部分
* 在Hint中设置类似{"total":0}可以隐藏总条数部分

# TreeView

* 树形控件
* 不要使用节点的StateIndex

# 附件1: ElementUI 图标

网址：<https://element.eleme.cn/#/zh-CN/component/icon>





# 附件2：ElementUI图标编号（ImageIndex）

1. el-icon-platform-eleme
2. el-icon-eleme
3. el-icon-delete-solid
4. el-icon-delete
5. el-icon-s-tools
6. el-icon-setting
7. el-icon-user-solid
8. el-icon-user
9. el-icon-phone
10. el-icon-phone-outline
11. el-icon-more
12. el-icon-more-outline
13. el-icon-star-on
14. el-icon-star-off
15. el-icon-s-goods
16. el-icon-goods
17. el-icon-warning
18. el-icon-warning-outline
19. el-icon-question
20. el-icon-info
21. el-icon-remove
22. el-icon-circle-plus
23. el-icon-success
24. el-icon-error
25. el-icon-zoom-in
26. el-icon-zoom-out
27. el-icon-remove-outline
28. el-icon-circle-plus-outline
29. el-icon-circle-check
30. el-icon-circle-close
31. el-icon-s-help
32. el-icon-help
33. el-icon-minus
34. el-icon-plus
35. el-icon-check
36. el-icon-close
37. el-icon-picture
38. el-icon-picture-outline
39. el-icon-picture-outline-round
40. el-icon-upload
41. el-icon-upload2
42. el-icon-download
43. el-icon-camera-solid
44. el-icon-camera
45. el-icon-video-camera-solid
46. el-icon-video-camera
47. el-icon-message-solid
48. el-icon-bell
49. el-icon-s-cooperation
50. el-icon-s-order
51. el-icon-s-platform
52. el-icon-s-fold
53. el-icon-s-unfold
54. el-icon-s-operation
55. el-icon-s-promotion
56. el-icon-s-home
57. el-icon-s-release
58. el-icon-s-ticket
59. el-icon-s-management
60. el-icon-s-open
61. el-icon-s-shop
62. el-icon-s-marketing
63. el-icon-s-flag
64. el-icon-s-comment
65. el-icon-s-finance
66. el-icon-s-claim
67. el-icon-s-custom
68. el-icon-s-opportunity
69. el-icon-s-data
70. el-icon-s-check
71. el-icon-s-grid
72. el-icon-menu
73. el-icon-share
74. el-icon-d-caret
75. el-icon-caret-left
76. el-icon-caret-right
77. el-icon-caret-bottom
78. el-icon-caret-top
79. el-icon-bottom-left
80. el-icon-bottom-right
81. el-icon-back
82. el-icon-right
83. el-icon-bottom
84. el-icon-top
85. el-icon-top-left
86. el-icon-top-right
87. el-icon-arrow-left
88. el-icon-arrow-right
89. el-icon-arrow-down
90. el-icon-arrow-up
91. el-icon-d-arrow-left
92. el-icon-d-arrow-right
93. el-icon-video-pause
94. el-icon-video-play
95. el-icon-refresh
96. el-icon-refresh-right
97. el-icon-refresh-left
98. el-icon-finished
99. el-icon-sort
100. el-icon-sort-up
101. el-icon-sort-down
102. el-icon-rank
103. el-icon-loading
104. el-icon-view
105. el-icon-c-scale-to-original
106. el-icon-date
107. el-icon-edit
108. el-icon-edit-outline
109. el-icon-folder
110. el-icon-folder-opened
111. el-icon-folder-add
112. el-icon-folder-remove
113. el-icon-folder-delete
114. el-icon-folder-checked
115. el-icon-tickets
116. el-icon-document-remove
117. el-icon-document-delete
118. el-icon-document-copy
119. el-icon-document-checked
120. el-icon-document
121. el-icon-document-add
122. el-icon-printer
123. el-icon-paperclip
124. el-icon-takeaway-box
125. el-icon-search
126. el-icon-monitor
127. el-icon-attract
128. el-icon-mobile
129. el-icon-scissors
130. el-icon-umbrella
131. el-icon-headset
132. el-icon-brush
133. el-icon-mouse
134. el-icon-coordinate
135. el-icon-magic-stick
136. el-icon-reading
137. el-icon-data-line
138. el-icon-data-board
139. el-icon-pie-chart
140. el-icon-data-analysis
141. el-icon-collection-tag
142. el-icon-film
143. el-icon-suitcase
144. el-icon-suitcase-1
145. el-icon-receiving
146. el-icon-collection
147. el-icon-files
148. el-icon-notebook-1
149. el-icon-notebook-2
150. el-icon-toilet-paper
151. el-icon-office-building
152. el-icon-school
153. el-icon-table-lamp
154. el-icon-house
155. el-icon-no-smoking
156. el-icon-smoking
157. el-icon-shopping-cart-full
158. el-icon-shopping-cart-1
159. el-icon-shopping-cart-2
160. el-icon-shopping-bag-1
161. el-icon-shopping-bag-2
162. el-icon-sold-out
163. el-icon-sell
164. el-icon-present
165. el-icon-box
166. el-icon-bank-card
167. el-icon-money
168. el-icon-coin
169. el-icon-wallet
170. el-icon-discount
171. el-icon-price-tag
172. el-icon-news
173. el-icon-guide
174. el-icon-male
175. el-icon-female
176. el-icon-thumb
177. el-icon-cpu
178. el-icon-link
179. el-icon-connection
180. el-icon-open
181. el-icon-turn-off
182. el-icon-set-up
183. el-icon-chat-round
184. el-icon-chat-line-round
185. el-icon-chat-square
186. el-icon-chat-dot-round
187. el-icon-chat-dot-square
188. el-icon-chat-line-square
189. el-icon-message
190. el-icon-postcard
191. el-icon-position
192. el-icon-turn-off-microphone
193. el-icon-microphone
194. el-icon-close-notification
195. el-icon-bangzhu
196. el-icon-time
197. el-icon-odometer
198. el-icon-crop
199. el-icon-aim
200. el-icon-switch-button
201. el-icon-full-screen
202. el-icon-copy-document
203. el-icon-mic
204. el-icon-stopwatch
205. el-icon-medal-1
206. el-icon-medal
207. el-icon-trophy
208. el-icon-trophy-1
209. el-icon-first-aid-kit
210. el-icon-discover
211. el-icon-place
212. el-icon-location
213. el-icon-location-outline
214. el-icon-location-information
215. el-icon-add-location
216. el-icon-delete-location
217. el-icon-map-location
218. el-icon-alarm-clock
219. el-icon-timer
220. el-icon-watch-1
221. el-icon-watch
222. el-icon-lock
223. el-icon-unlock
224. el-icon-key
225. el-icon-service
226. el-icon-mobile-phone
227. el-icon-bicycle
228. el-icon-truck
229. el-icon-ship
230. el-icon-basketball
231. el-icon-football
232. el-icon-soccer
233. el-icon-baseball
234. el-icon-wind-power
235. el-icon-light-rain
236. el-icon-lightning
237. el-icon-heavy-rain
238. el-icon-sunrise
239. el-icon-sunrise-1
240. el-icon-sunset
241. el-icon-sunny
242. el-icon-cloudy
243. el-icon-partly-cloudy
244. el-icon-cloudy-and-sunny
245. el-icon-moon
246. el-icon-moon-night
247. el-icon-dish
248. el-icon-dish-1
249. el-icon-food
250. el-icon-chicken
251. el-icon-fork-spoon
252. el-icon-knife-fork
253. el-icon-burger
254. el-icon-tableware
255. el-icon-sugar
256. el-icon-dessert
257. el-icon-ice-cream
258. el-icon-hot-water
259. el-icon-water-cup
260. el-icon-coffee-cup
261. el-icon-cold-drink
262. el-icon-goblet
263. el-icon-goblet-full
264. el-icon-goblet-square
265. el-icon-goblet-square-full
266. el-icon-refrigerator
267. el-icon-grape
268. el-icon-watermelon
269. el-icon-cherry
270. el-icon-apple
271. el-icon-pear
272. el-icon-orange
273. el-icon-coffee
274. el-icon-ice-tea
275. el-icon-ice-drink
276. el-icon-milk-tea
277. el-icon-potato-strips
278. el-icon-lollipop
279. el-icon-ice-cream-square
280. el-icon-ice-cream-round