**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、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

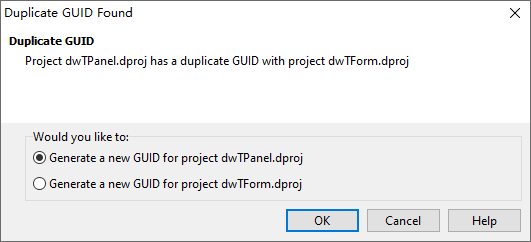
# 支持控件

* 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

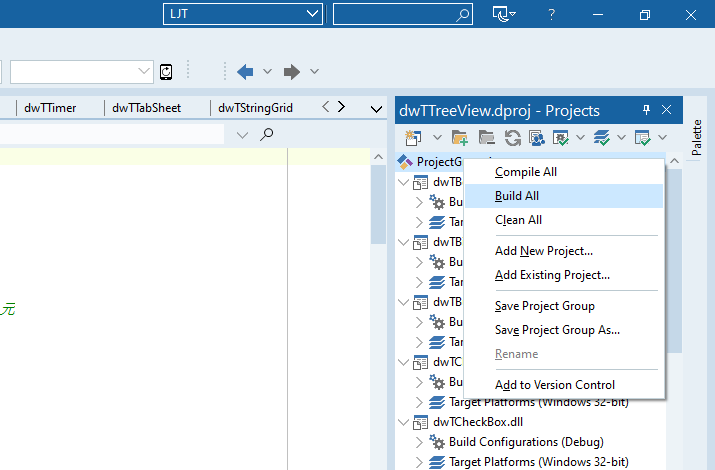
# deweb入门

* 开发版本
* 目前支持Delphi 10.1/10.2/10.3/10.4。
* 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
* 打开Source\dwVcls\ AllVcls.groupproj

如果遇到以下情况，选择“OK”按钮



* 然后Build all；



* 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D10.1.x，则重命名dewebServer101.exe为dewebServer.exe；

如果是D10.2.x，则重命名dewebServer102.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

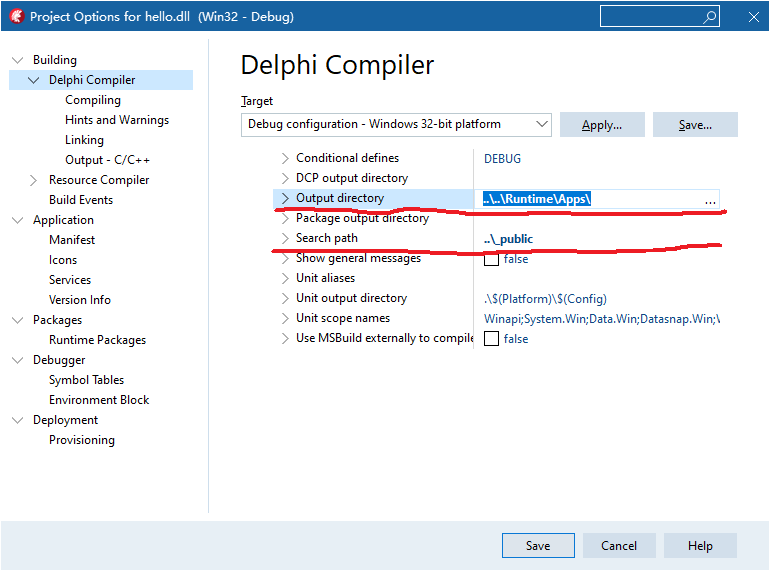
DLLScreen := Screen;

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

DLLUnloadProc(DLL\_PROCESS\_DETACH);

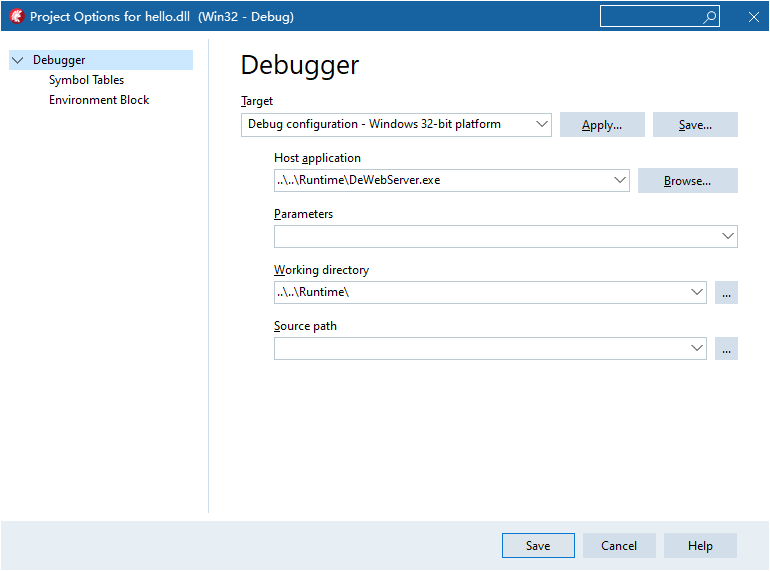
end.

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

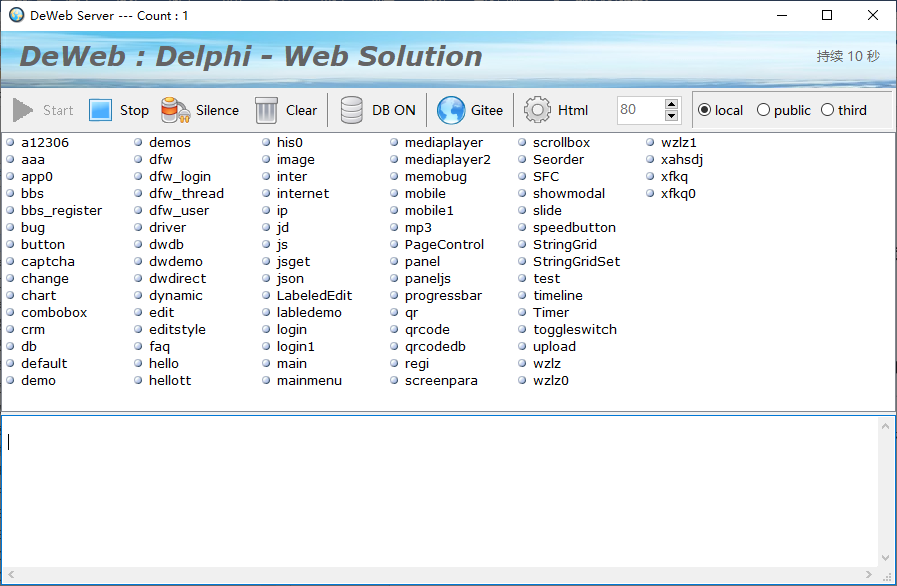


* 设置运行参数

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

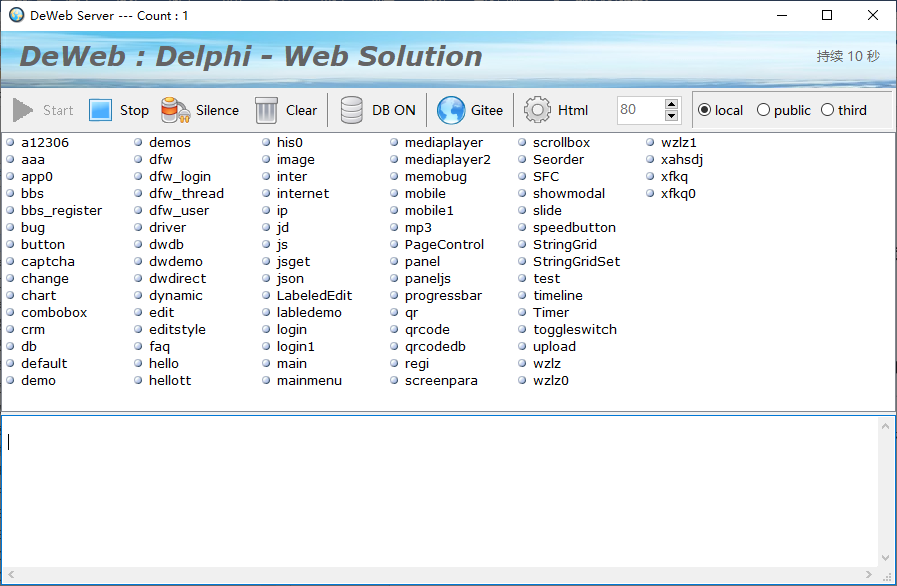
* 带参数命令行运行

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，

* Start用于启动deweb服务；
* Stop用于停止deweb服务；
* 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
* Clear用于清空交互信息
* 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
* Gitee用于直接打开deweb在码云（https://www.gitee.com）的链接
* 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,

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

}

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

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

# 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
* 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dewebServer\_1042.exe等，必须！
* borlndmm.dll，必须！
* deweb.ini，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；
* 所有窗体不能Show/ShowModal;

# 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

* 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);

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);

* 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）

* **dwPreGetCookie**预读取Cookie

读取Cookie时必须先预读取！执行预读取函数后一段时间（一般3秒内）才有可能采用dwGetCookie函数读取Cookie值

function dwPreGetCookie(AForm:TForm;AName,ANull:String):Integer;

AForm:TForm 当前窗体, 一般用self

AName: String， Cookie名称

ANull: String; 没读到Cookie时的空值

* **dwGetCookie**读取Cookie值

读取Cookie时必须先预读取dwPreGetCookie！执行预读取函数后一段时间（一般1秒内）才有可能采用dwGetCookie函数读取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);

# 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

* 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

* 如果网页高度比屏幕高度高，可以**在OnShow中**采用类似dwSetHeight(self,2000)来设置高度。

**注意：**需要将窗体的VertScrollBox的Visible设置为False

* 如果窗体Name = TDefault，则可以URL中不输入http://127.0.0.1:8080/default.dw, 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）；
* URL中的参数可以通过dwGetProp(Self,'params')得到；
* 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；
* 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\" /> "

]

}

# 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；

* 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

* 注意图标和颜色可以同时配合使用，例：{"type":"success","icon":" el-icon-star-off"}
* 上传成功后，自动激活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，则可能重命名）

* **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)","color":"rgb(200,200,200)"}

其中：backgroundcolor为按钮背景色，color为按钮字体颜色

* 自定义字体大小

{"fontsize":"16px"}

其中：fontsize为按钮字体大小

* 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>

* 通过更改Hint来改变圆角半径

{"borderradius":"5px"}

{"borderradius":"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



* 常用的左侧有圆角，右侧没有，则可以

{"borderradius":"10px 0 0 10px"}

如图



* 左侧无圆角，右侧有，则可以类似设置hint

{"borderradius":"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":"Button1\_\_cap='enter';","onexit":"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":"Panel4\_\_cap='click';"}

表示当鼠标单击时，置Button1的caption为click；

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。



# 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用来指示边框颜色；
* 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；

# 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

# Edit

* 支持OnChange事件（必须有代码，可以是注释）
* 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"}
* **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

# Image

* 可以采用设置Hint为类似{"radius":"50%"}或{"radius":"5px"}来设置圆角
* 可以采用设置Hint中的src来设置图片源。如果不设置，当直接显示当前显示的图片（如果窗体较多，请尽可能将TImage的名称和其他TImage的名称不一样）。
* 设置Hint中的src来设置图片源的源文件可能是所有网络支持类型（\*.jpg,\*.png,\*.gif等），如果不设置，系统将自动保存当前显示的图片为jpg格式，并显示。

# Image(用作验证码)

* **注**：用作验证码时需要设置其HelpKeyword='captcha'
* 生成的图片默认保存在目录runtime/media/images，请定时清理
* 通过dwGetProp(Image1,'captcha')得到验证码的字符串值（uses dwBase;）
* 用类似以下代码主动刷新验证码

dwRunJS('document.getElementById("'+Image1.Name+'").click();',self);

# 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
* 如果需要在运行过程中动态改变Label的字体、字号、颜色、样式、对齐等，可以通过设置HelpKeyword为rich为支持该功能

# ListBox

* 支持Enabled/Visible/Left/Top/Width/Height
* 支持OnClick事件
* 支持单选/多选
* 暂不支持在OnCreate后动态改变Items

# 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
* 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
* 第1个菜单项的Hint用于保存前景色，字体色等信息；

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

* 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

# 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

# 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



# 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！！！

# 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

{"borderradius":"5px"}

{"borderradius":"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 2px #aaa;"}

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

{"borderradius":"10px 0 0 10px"}

如图



* 左侧无圆角，右侧有，则可以类似

{"borderradius":"0 10px 10px 0"}

如图



* **注意** : Panel的Enabled不能导致子控件的Enabled
* 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":"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。

# ProgressBar

# RadioButton

# ScrollBox

* 其内**必须有且仅有一个Panel**，其他控件请放到该Panel中
* 滚动条事件自动激活OnEndDock事件，其中X为滚动值，Y为方向，1：向下，-1向上

# 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
* 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))

# 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"}

* 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

# 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），否则为当前设置的的

# StringGrid

* Height应为RowCount \* DefaultRowHeight+1，以保持最佳显示效果
* BorderStyle=bsNone时为无边框
* 支持OnClick事件, 在点击时激活
* 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。
* 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{"dwstyle":"border"}表示带边框和竖线  
  {"dwstyle":"stripe"}表示带斑马纹  
  {"dwstyle":"show-summary"}表示带合计  
    
  还可以组合，比如：  
  {"dwstyle":"border stripe"}表示带边框和竖线,带斑马纹

# 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
* ThumbWidth为选择框宽度
* StateCaptions-CaptionOff/CaptionOn为未选中/选中的标题

# TrackBar

# 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