

# CnPack 停靠组件帮助文档

作者：周益波(鲁小班) zhouyibo2000@sina.com 或 luxiaoban@sina.com。

刘啸 (liuxiao@cnpack.org)

部门：CnPack 开发组 不可视组件组 设计员、开发人员

类别：组件帮助文档

备注：该文档移植自原作者作品 DockPresident 的帮助文档

版本：V0.8.2.0

创建：2007.08.16

修改：2007.11.11

## 一、引言

### 1、 组件包概述

作为 Delphi 的忠实用户，我想大家对 Delphi 中的停靠窗体应该比较熟悉吧，是不是也希望自己编的程序也具有这样的功能？使她看起来更漂亮，更专业，更方便；CnPack 组件包中的停靠 (Dock) 系列组件正好能满足这种需求。不需要编写一行代码就可以做出具有停靠功能的窗体。

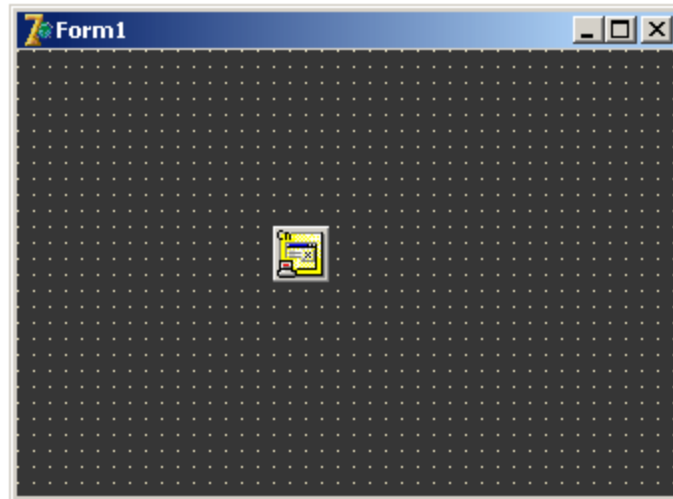
CnPack 停靠系列组件中有两个主要的组件，她们分别是 TCnDockServer 和 TCnDockClient。TCnDockServer 使其上面的窗体具有停靠服务器的功能，TCnDockClient 使其上面的窗体具有停靠客户的功能。这两个组件的具体功能请见本文的其他部分。另外还有一些设置停靠风格的类，用来改变 TCnDockServer 和 TCnDockClient 的停靠风格，而且用户可以制作自己的停靠风格。目前默认提供的是 TCnDelphiDockStyle、TCnVCDockStyle、TCnVIDDockStyle 和 TCnVSNETDockStyle。分别用来模仿 Delphi、Visual C++、Visual InterDev 和 Visual Studio.net 的停靠风格。

### 2、 使用入门

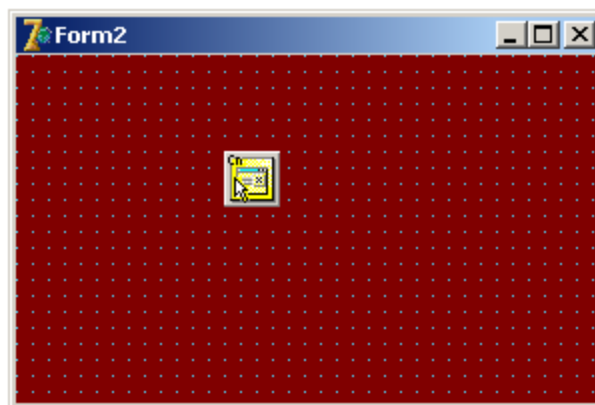
使用 CnPack 停靠系列组件很简单，下面用一个例子来说明。假设用户已经安装了 CnPack 组件包，在 Delphi 的组件面板会看见以下几个组件，如下图所示：



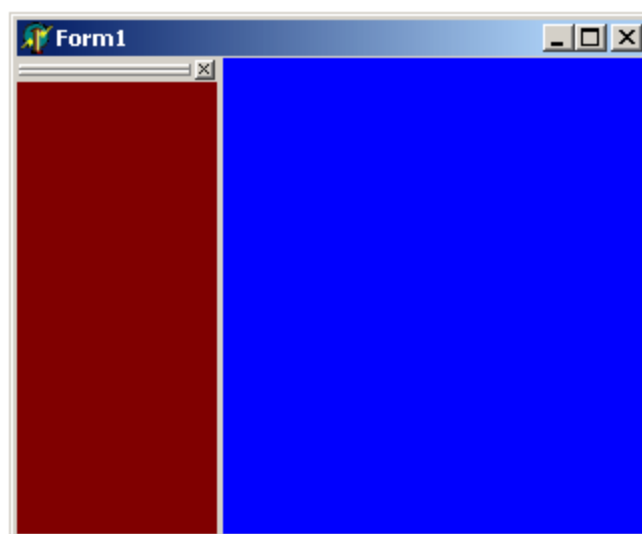
现在新建一个工程 Project1.dpr，主窗体叫做 Form1，然后用户可以在组件面板的 DockPresident 页面中选择第一个组件 TCnDockServer 放到 Form1 上面，名称为 CnDockServer1，如下图所示：



为了突出停靠的效果，我们这里特别把 Form1 的颜色设置成 `clBackground`。点击 Delphi 中工具栏中的 ‘New Form’ 按钮创建一个新的窗体，叫做 Form2，然后用户可以在组件面板中选择组件 `TCnDockClient` 放到 Form2 上面，名称为 `CnDockClient1`，如下图所示：



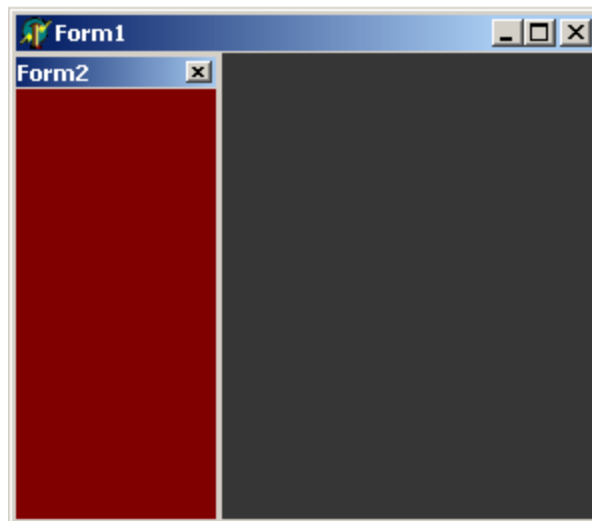
为了突出停靠的效果，我们特别把 Form2 的颜色设置成 `clMaroon`。最后不要忘记把 Form2 的 `Visible` 属性设置成 `True`。然后点击 Delphi 中工具栏上 ‘Run’ 按钮运行程序，我们会发现有二个窗体显示出来，然后把 Form2 拖动到 Form1 的边界附近，就可以完成停靠操作，如下图所示：



这是最简单的实现停靠功能的步骤。当然如果用户希望实现复杂一点的功能，也是很简单的，比如用户希望实现类似 Visual InterDev 的停靠风格，只要跟着下面的步骤操作就可以了：

选择 Delphi 的组件面板的组件 TCnVIDDockStyle，放到 Form1 窗体上面，名称为 CnVIDDockStyle1，然后选择 Form1 上的 CnDockServer1，在 Object Inspector（对象观察器）上面可以看到 CnDockServer1 有一个属性叫做 DockStyle，点击它的下拉按钮，会弹出一个选项，这个选项就是刚刚放到 Form1 上面的 CnVIDDockStyle1，选择它。然后打开 Form2，点击 Delphi 的 File 菜单中的 Use Unit 菜单项，在弹出的 Use Unit 对话框上面选择 Unit2，点击 Ok 确定（这一步是为下面的操作做准备的），再选中 Form2 上面的 CnDockClient1，在 Object Inspector（对象观察器）上面可以看到 CnDockClient1 也有一个属性叫做 DockStyle，点击它的下拉按钮，可以看到有一个叫做 Form1.CnVIDDockStyle1 的选项在里面，这个 Form1.CnVIDDockStyle1 的意思就是 Form1 上面的 CnVIDDockStyle1 组件（现在用户应该明白前面一步操作的作用了吧，它的作用是使 Form2 能访问到 Form1 上面的数据），也需要选中它。

现在用户就能够运行程序了，如下图所示：



## 二、TCnDockBaseControl 类

### 1、功能概述

在介绍 TCnDockServer 或者 TCnDockClient 组件之前，读者需要了解一下 TCnDockBaseControl 类，它是 TCnDockServer 和 TCnDockClient 共同的祖先，继承于 TCnComponent 类，是一个不可视组件。它里面定义了一些 TCnDockServer 或者 TCnDockClient 类中的共同的特性，比如 ParentForm, EnableDock 等。这个类主要和窗体打交道，在 TCnDockBaseControl 中有一个很重要的地方，请看如下代码：

```
constructor TCnDockBaseControl.Create(AOwner: TComponent);  
begin  
  if not (csDesigning in ComponentState) then  
    begin  
      { 保存老的窗口过程 }  
      FOldWindowProc := Parent.WindowProc;  
      { 重载新的窗口过程 }  
      Parent.WindowProc := WindowProc;  
    end;  
end;
```

这段代码把主窗体上面的窗口过程保存了下来，然后用新的窗口过程代替原来的窗口过程，这样用户就能预先捕获到父窗体上面的消息(包括 Windows 标准消息和 VCL 消息)，然后根据具体的消息进行处理，也可以过滤掉这个消息：

```
procedure TCnDockBaseControl.WindowProc(var Message: TMessage);  
begin  
  { 调用老的窗口过程 }  
  FOldWindowProc(Message);  
end;
```

并且这个函数被定义成虚拟的，也就是说从 TCnDockBaseControl 继承的类都有捕获父窗体的消息的能力，只要重载这个 WindowProc 函数。

由于 TCnDockBaseControl 主要是为了控制父窗体的行为，所以她也定义了一些父窗体的事件，代码如下：

```
begin  
  FOldFormShow := Parent.OnShow;  
  Parent.OnShow := FormShow;  
end;  
  
procedure TCnDockBaseControl.FormShow(Sender: TObject);
```

```
begin  
  if Assigned(FOldFormShow) then  
    FOldFormShow(Self);  
end;
```

这样，当系统调用 OnShow 的时候，就会预先调用 FormShow，然后根据条件判断是否执行默认的 OnShow(在这里是 FOldFormShow)事件。

TCnDockServer 和 TCnDockClient 都是从 TCnDockBaseControl 继承而来，并且重载了 TCnDockBaseControl 的 WindowProc 方法，所以 TCnDockServer 和 TCnDockClient 都可以控制她们的父窗体的消息，从而控制她们的行为，这就是本套组件的基本原理。

## 2、 所在文件

CnDockFormControl.pas，停靠服务和客户组件单元。

## 3、 属性说明

(见派生类中的说明)

## 4、 方法说明

(无)

## 5、 事件说明

(无)

# 三、 TCnDockServer 组件

## 1、 组件功能概述

TCnDockServer 的作用是使放了 TCnDockServer 组件的窗体具有停靠服务器的作用，也就是说可以让其他客户窗体(放了 TCnDockClient 组件的窗体)能够停靠进来，可以分别停靠在窗体的上下左右方向。为了实现这些功能，需要在窗体上创建四个 Panel，四个 Splitter。在程序窗体创建的时候会自动创建他们，Panel 的作用是作为客户窗体的 HostDockSite，Splitter 的作用是调整 Panel 的宽度或者高度，TCnDockServer 还定义了四个 TCnSplitterStyle 类，分别用来设置 Splitter 的风格。

## 2、 所在文件

CnDockFormControl.pas，停靠服务和客户组件单元。

### 3、 属性说明

**property** LeftDockPanel: TCnDockPanel;

左边的停靠服务组件，这个组件是继承自 TCustomPanel。

**property** RightDockPanel: TCnDockPanel;

右边的停靠服务组件。

**property** TopDockPanel: TCnDockPanel;

上边的停靠服务组件。

**property** BottomDockPanel: TCnDockPanel;

下边的停靠服务组件。

**property** LeftSplitter: TCnDockSplitter;

左边的分割条，这个组件继承自 TSplitter。

**property** RightSplitter: TCnDockSplitter;

右的分割条。

**property** TopSplitter: TCnDockSplitter;

上边的分割条。

**property** BottomSplitter: TCnDockSplitter;

下边的分割条。

**property** DockPanel[Index: Integer]: TCnDockPanel;

停靠服务组件数组，根据 Index 索引号来取得具体的 DockPanel，分别是 1:TopDockPanel;2:BottomDockPanel;3:LeftDockPanel;4:RightDockPanel。

**property** DockPanelWithAlign[Index: TAlign]: TCnDockPanel;

停靠服务组件数组，根据排列索引号来取得具体的 DockPanel，分别是 aLeft:LeftDockPanel; aRight:RightDockPanel; aTop:TopDockPanel; aBottom:BottomDockPanel。

**property** DockSplitter[Index: Integer]: TCnDockSplitter;

分割条数组，根据 Index 索引号来取得具体的 DockSplitter，类似于 DockPanel[Index: Integer]属性。

**property** DockSplitterWithAlign[Index: TAlign]: TCnDockSplitter;

分割条数组，根据排列索引号来取得具体的 DockSplitter，类似于 DockPanelWithAlign[Index: TAlign]属性。

**property** Version: Integer;

版本号。

**property** LeftSplitterStyle: TCnSplitterStyle;

左边分割条的一些基本属性，如 **Color**（颜色），**Cursor**（光标形状），**ParentColor**（是否和父组件的颜色一样），**ResizeStyle**（当分割条移动的时候是什么风格），**Size**（宽度或者高度），**MinSize**（和父组件的边界之间的最小的距离）。

**property** TopSplitterStyle: TCnSplitterStyle;

上边分割条的风格，见 **LeftSplitterStyle** 的描述。

**property** RightSplitterStyle: TCnSplitterStyle;

右边分割条的风格，见 **LeftSplitterStyle** 的描述。

**property** BottomSplitterStyle: TCnSplitterStyle;

下边分割条的风格，见 **LeftSplitterStyle** 的描述。

以下属性来源于基类 **TCnDockBaseControl**:

**property** EnableDock: Boolean;

能否停靠。

**property** LeftDock: Boolean;

左边能否停靠。

**property** TopDock: Boolean;

上边能否停靠。

**property** RightDock: Boolean;

右边能否停靠。

**property** BottomDock: Boolean;

下边能否停靠。

**property** DockStyle: TCnBasicDockStyle;

停靠风格。

## 4、 方法说明

（无）

## 5、 事件说明

**property** OnGetClientAlignSize: TGetClientAlignSizeEvent;

TGetClientAlignSizeEvent = **procedure**(Align: TAlign; **var** Value: Integer) **of** **object**;

当根据 **TCnDockServer** 的 **ParentForm** 中的组件的 **Align** 属性计算 **ParentForm** 客户区的

大小的时候调用，用户可以修改其中的引用参数。

参数：

1. Align: TAlign; 供修改的 Align 变量。
2. var Value: Integer; 供修改的 Value。

```
property OnFinishSetDockPanelSize: TFinishSetDockPanelSizeEvent;  
TFinishSetDockPanelSizeEvent = procedure(DockPanel: TCnDockPanel) of  
object;
```

当重新设置了 DockPanel 的宽度或者高度的时候调用。

参数：

1. DockPanel: TCnDockPanel; DockPanel 实例。

## 四、TCnDockClient 组件

### 1、 组件功能概述

TCnDockClient 的作用是使放了 TCnDockClient 组件的窗体具有停靠客户的作用，她们可以停靠进服务器中，如前所述，还有一点很重要的地方是，两个停靠客户之间可以相互停靠，只要把 TCnDockClient 的 EachotherDock 属性设成 True，就可以实现相互停靠的功能，相互停靠有两种风格，平铺和分页形式。

平铺方式：停靠客户相互之间是平等的关系，没有覆盖或者被覆盖的情况。如下图所示：



分页方式：停靠客户都在一个分页组件里面。如下图所示：



### 2、 所在文件

CnDockFormControl.pas，停靠服务和客户组件单元。

### 3、 属性说明

**property** DockState: Integer;

停靠客户的状态，是在浮动状态或者停靠状态，还是其他别的状态。

**property** LRDockWidth: Integer;

停靠时候客户的宽度。

**property** TBDockHeight: Integer;

停靠时候客户的高度。

**property** NCPopupMenu: TPopupMenu;

当鼠标右击停靠客户的标题栏，“把手”或者分页组件的 Tab 的时候，指定弹出哪一个 PopupMenu。

**property** DirectDrag: Boolean;

当鼠标点击停靠窗体的标题栏的时候，是否立刻进行停靠操作，如果 DirectDrag 属性是 True，就立刻处理停靠操作；否者就判断鼠标移动的距离是否超过了一定的距离，要是超过了就处理停靠操作，不然就不处理。

**property** ShowHint: Boolean;

是否显示提示信息。

以下属性来源于基类 TCnDockBaseControl:

**property** EnableDock: Boolean;

能否停靠。

**property** LeftDock: Boolean;

左边能否停靠。

**property** TopDock: Boolean;

上边能否停靠。

**property** RightDock: Boolean;

右边能否停靠。

**property** BottomDock: Boolean;

下边能否停靠。

**property** DockStyle: TCnBasicDockStyle;

停靠风格。

## 4、 方法说明

**procedure** ShowParentForm;

显示 ParentForm。

**procedure** HideParentForm;

隐藏 ParentForm。

**procedure** RestoreChild;

还原所有的停靠客户。

## 5、 事件说明

**property** OnFormShow: TNotifyEvent;

当 TCnDockClient 的 ParentForm 显示的时候调用。

**property** OnFormHide: TNotifyEvent;

当 TCnDockClient 的 ParentForm 隐藏的时候调用。

**property** OnNCButtonDown: TNCButtonDownEvent;

TNCButtonDownEvent = TNCButtonEvent;

TNCButtonEvent = **procedure** (DockClient: TCnDockClient; Button: TMouseButton;

X, Y: Smallint; HitTest: Longint; MouseStation: TMouseStation) **of object**;

当鼠标停靠客户的标题栏，‘把手’或者分页组件的 Tab 上点击的时候调用。

参数:

1. DockClient: TCnDockClient; DockClient 组件实例本身。

2. Button: TMouseButton; 鼠标按钮。

3. X, Y: Smallint; 发生事件时的鼠标坐标。

4. HitTest: Longint;

5. MouseStation: TMouseStation; 鼠标的位置;

TMouseStation = (msFloat, msConjoin, msTabPage);

分别代表“在浮动窗体上”，“在平铺服务器上”，“在分页服务器上”。

**property** OnNCButtonUp: TNCButtonUpEvent;

TNCButtonUpEvent = TNCButtonEvent;

当鼠标停靠客户的标题栏，‘把手’或者分页组件的 Tab 上释放的时候调用。

参数: (同 TNCButtonEvent)

**property** OnNCMouseMove: TNCMouseMoveEvent;

TNCMouseMoveEvent = **procedure** (DockClient: TCnDockClient;

X, Y: Smallint; HitTest: Longint; MouseStation: TMouseStation) **of object**;

当鼠标停靠客户的标题栏，‘把手’或者分页组件的 Tab 上移动的时候调用。

参数:

1. DockClient: TCnDockClient; DockClient 组件实例本身。

2. Button: TMouseButton; 鼠标按钮。
3. X, Y: Smallint; 发生事件时的鼠标坐标。
4. HitTest: Longint;
5. MouseStation: TMouseStation; 鼠标的位置;  
TMouseStation = (msFloat, msConjoin, msTabPage);  
分别代表“在浮动窗体上”,“在平铺服务器上”,“在分页服务器上”。

**property** OnNCButtonDblClk: TNCButtonDblClkEvent;  
TNCButtonDblClkEvent = TNCButtonEvent;

当鼠标停靠客户的标题栏,‘把手’或者分页组件的 Tab 上双击的时候调用。

参数:(同 TNCButtonEvent)

**property** OnPaintDockGrabber: TPaintDockGrabberEvent;  
TPaintDockGrabberEvent = TPaintDockEvent;  
TPaintDockEvent = **procedure** (Canvas: TCanvas;

Control: TControl; **const** ARect: TRect) **of object**;

当画‘把手’的时候调用。

参数:

1. Canvas: TCanvas; 供绘画的画布实例。
2. Control: TControl; 绘画组件本身。
3. **const** ARect: TRect; 绘画的矩形区域。

**property** OnPaintDockSplitter: TPaintDockSplitterEvent;  
TPaintDockSplitterEvent = TPaintDockEvent;

当画分割条的时候调用。

参数:(同 TPaintDockEvent)

**property** OnFormShowHint: TFormHintEvent;  
TFormHintEvent = **procedure** (HTFlag: Integer; **var** HintStr: **string**;  
**var** CanShow: Boolean) **of object**;

当显示提示信息的时候调用。

参数:

1. HTFlag: Integer; Hint 的标记。
2. **var** HintStr: **string**; 待提示的字符串。
3. **var** CanShow: Boolean; 可通过设置此变量以决定是否显示 Hint。

## 五、TCnBasicDockStyle 类

### 1、 功能概述

TCnBasicDockStyle 是所有停靠风格类的基类。可以继承 TCnBasicDockStyle 类来创建自己的停靠风格类,目前 CnPack 停靠组件系列中有四个停靠风格类,分别是 TCnDelphiDockStyle, TCnVCDockStyle, TCnVIDDockStyle 和 TCnVSNETDockStyle, 分别类似 Delphi、Visual

C++、Visual InterDev 和 Visual Studio.net 的停靠风格。

TCnBasicDockStyle 的子类主要用在 DockStyle 属性中。不光 TCnDockClient 组件有 DockStyle 属性，TCnDockServer 也有 DockStyle 属性。并且，TCnDockClient 组件和 TCnDockServer 的 DockStyle 属性必须相同，才能正确的显示不同的停靠效果。比如：我们要使用 Visual C++ 的停靠风格，就要在设计期设置 TCnDockServer 的 DockStyle 为 TCnVCDockStyle。

在 TCnBasicDockStyle 中定义了一些类的引用：

```
FCnDockPanelClass: TCnDockPanelClass;  
// TCnDockServer 中四个 TCnDockPanel 的类的引用。
```

```
FCnDockSplitterClass: TCnDockSplitterClass;  
// TCnDockServer 中四个 TSplitter 的类的引用。
```

```
FCnConjoinPanelClass: TCnConjoinPanelClass;  
// 平铺服务器的类的引用。
```

```
FCnTabDockClass: TCnTabDockClass;  
// 分页服务器的类的引用。
```

```
FCnDockPanelTreeClass: TCnDockTreeClass;  
// TCnDockPanel 中 IDockPresident 的类的引用。
```

```
FCnDockPanelZoneClass: TCnDockZoneClass;  
// TCnDockPanel 中 IDockPresident 的节点类。
```

这样用户只要自己定义一些类，然后把他们赋值给类的引用，用户就实现了用自己的类来代替原先的类的功能。

## 六、TCnDelphiDockStyle 组件

### 1、 组件功能概述

本组件是具有类似 Delphi 的停靠风格的组件。

### 2、 所在文件

CnDelphiDockStyle.pas，具有类似 Delphi 的停靠风格的单元。

### 3、 属性说明

**property** ConjoinServerOption: TCnBasicConjoinServerOption;  
配置平铺服务器的一些属性。

---

TCnBasicConjoinServerOption 类的属性:

**property** GrabbersSize: Integer;  
把手的大小。

**property** SplitterWidth: Integer;  
分割条的宽度。

**property** TabServerOption: TCnBasicTabServerOption;  
配置分页服务器的一些属性。

TCnBasicTabServerOption 的属性:

**property** HotTrack: Boolean;  
当鼠标移动到 Tab 上的时候, Tab 上的文字是否高亮显示。

**property** TabPosition: TTabPosition  
Tab 的位置, 上边, 下边, 左边或者右边。

## 七、 TCnVCDockStyle 组件

### 1、 组件功能概述

本组件是具有类似 Visual C++的停靠风格的组件。

### 2、 所在文件

CnVCDockStyle.pas, 具有类似 Visual C++的停靠风格的单元。

### 3、 属性说明

**property** ConjoinServerOption: TCnVCConjoinServerOption;  
配置平铺服务器的一些属性。

TCnVCConjoinServerOption 类的属性:

**property** GrabbersSize: Integer;  
把手的大小。

**property** BorderWidth: Integer;  
边框的宽度。

**property** SplitterWidth: Integer;  
分割条的宽度。

**property** TabServerOption: TCnVCTabServerOption;  
配置分页服务器的一些属性。

TCnVCTabServerOption 的属性:

**property** HotTrack: Boolean;  
当鼠标移动到 Tab 上的时候, Tab 上的文字是否高亮显示。

**property** TabPosition: TTabPosition  
Tab 的位置, 上边, 下边, 左边或者右边。

## 八、 TCnVIDDockStyle 组件

### 1、 组件功能概述

本组件是具有类似 Visual InterDev 的停靠风格的组件。

### 2、 所在文件

CnVIDDockStyle.pas, 具有类似 Visual InterDev 的停靠风格的单元。

### 3、 属性说明

**property** ConjoinServerOption: TCnVIDConjoinServerOption;  
配置平铺服务器的一些属性。

TCnVIDConjoinServerOption 类的属性:

**property** ActiveFont: TFont;  
停靠客户获得焦点的时候的字体。

**property** ActiveTitleEndColor: TColor;  
停靠窗体获得焦点的时候‘把手’的结尾颜色。

**property** ActiveTitleStartColor: TColor;  
停靠窗体获得焦点的时候‘把手’的开始颜色。

**property** GrabbersSize: Integer;  
把手的大小。

**property** InactiveFont: TFont;  
停靠客户失去焦点的时候的字体。

---

**property** InactiveTitleEndColor: TColor;

停靠窗体失去焦点的时候‘把手’的结尾颜色。

**property** InactiveTitleStartColor: TColor;

停靠窗体失去焦点的时候‘把手’的开始颜色。

**property** SplitterWidth: Integer;

分割条的宽度。

**property** SystemInfo: Boolean;

是否按照系统的设置来给 **ConjoinServerOption** 中其它的属性赋值（系统设置指桌面的外观，用户可以这样设置桌面的外观：在 **Windows** 的桌面上点击鼠标右键，弹出一个菜单，点击‘属性’菜单项，弹出一个对话框，里面是一个分页组件，点击第三个叫做‘外观’的标签，在那里就可以设置桌面的外观）。

**property** TextAlignment: TAlignment;

停靠客户的‘把手’上的文字的排列方向。

**property** TextEllipsis: Boolean;

如果显示文字的空间太小，停靠客户的‘把手’上的文字是否显示省略号。

**property** TabServerOption: TCnVIDTabServerOption;

配置分页服务器的一些属性。

**TCnVIDTabServerOption** 类的属性：

**property** ActiveFont: TFont;

当分页组件中的某一个标签获得焦点的时候，其上面的文字的字体。

**property** ActiveSheetColor: TColor;

获得焦点的标签的颜色。

**property** HotTrack: Boolean;

当鼠标移动到 **Tab** 上的时候，**Tab** 上的文字是否高亮显示。

**property** HotTrackColor: TColor;

热点的颜色。

**property** InactiveFont: TFont;

失去焦点的标签上的文字字体。

**property** InactiveSheetColor: TColor;

失去焦点的标签颜色。

---

**property** ShowTabImages: Boolean;  
是否显示标签上的图像。

**property** TabPosition: TTabPosition;  
Tab 的位置, 上边, 下边, 左边或者右边。

#### 4、 方法说明

(无)

#### 5、 事件说明

**property** SystemInfoChange: TSystemInfoChange;  
**type** TSystemInfoChange = **procedure** (Value: Boolean) **of object**;

当用户改变了 TCnVIDDockStyle 控件的 SystemInfo 属性, 或者改变了其他的属性, 就会触发 SystemInfoChange 事件。

参数:

1. Value: Boolean;表示当前的 SystemInfo 属性。

## 九、 TCnVSNETDockStyle 组件

### 1、 组件功能概述

本组件是具有类似 Visual Studio .NET 的停靠风格的组件。

### 2、 所在文件

CnVSNETDockStyle.pas, 具有类似 Visual Studio .NET 的停靠风格的单元。

### 3、 属性说明

(同 TCnVIDDockStyle 的属性)