All Topics, MFC / C++ >> Miscellaneous Controls >> Tooltips
http://www.codeproject.com/miscctrl/pptooltip.asp

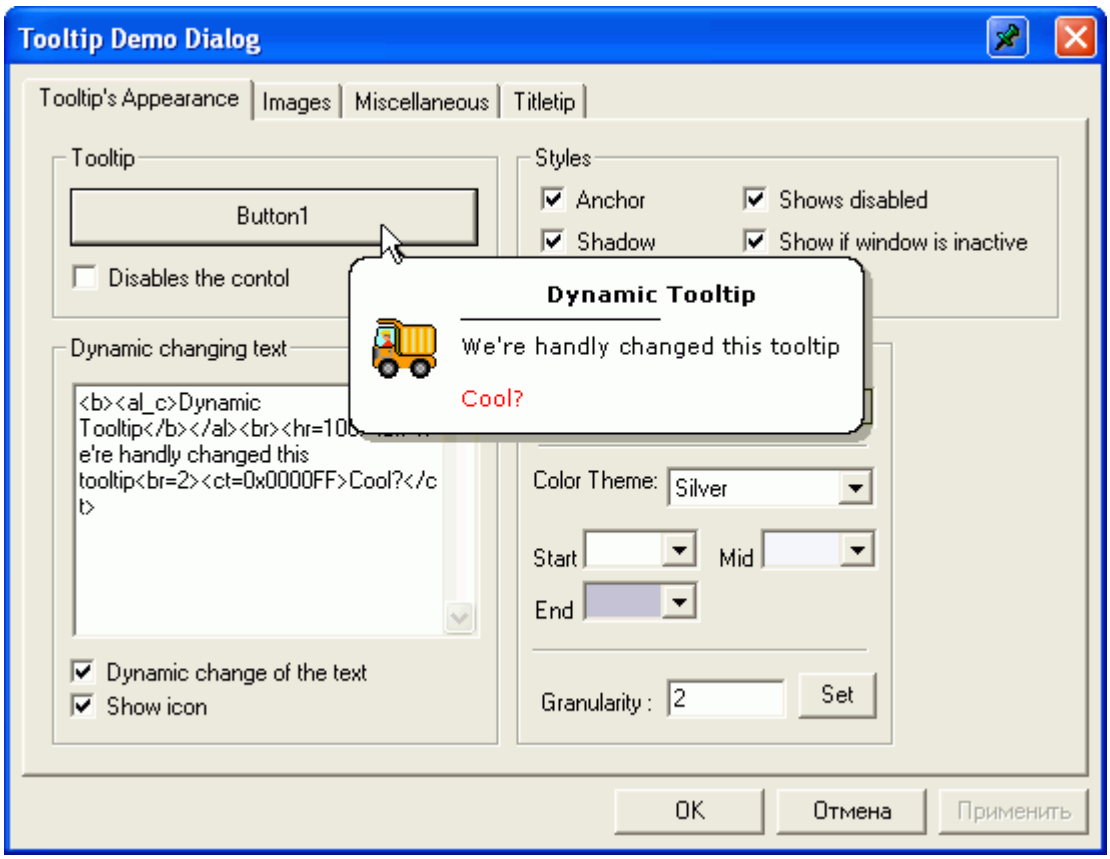
VC7, VC6, XP, W2K, Win9X, MFC
Posted 13 Feb 2003
Updated 9 May 2003
48,156 views

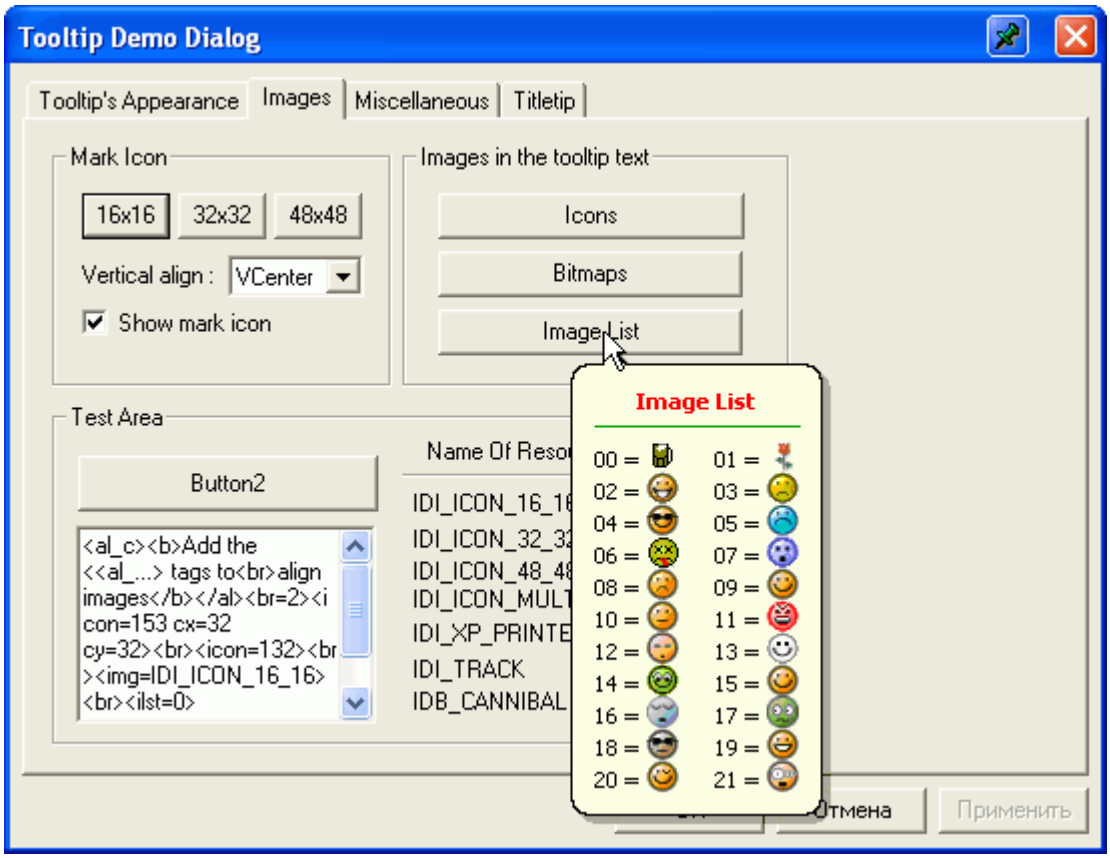
CPPTooltip v1.4

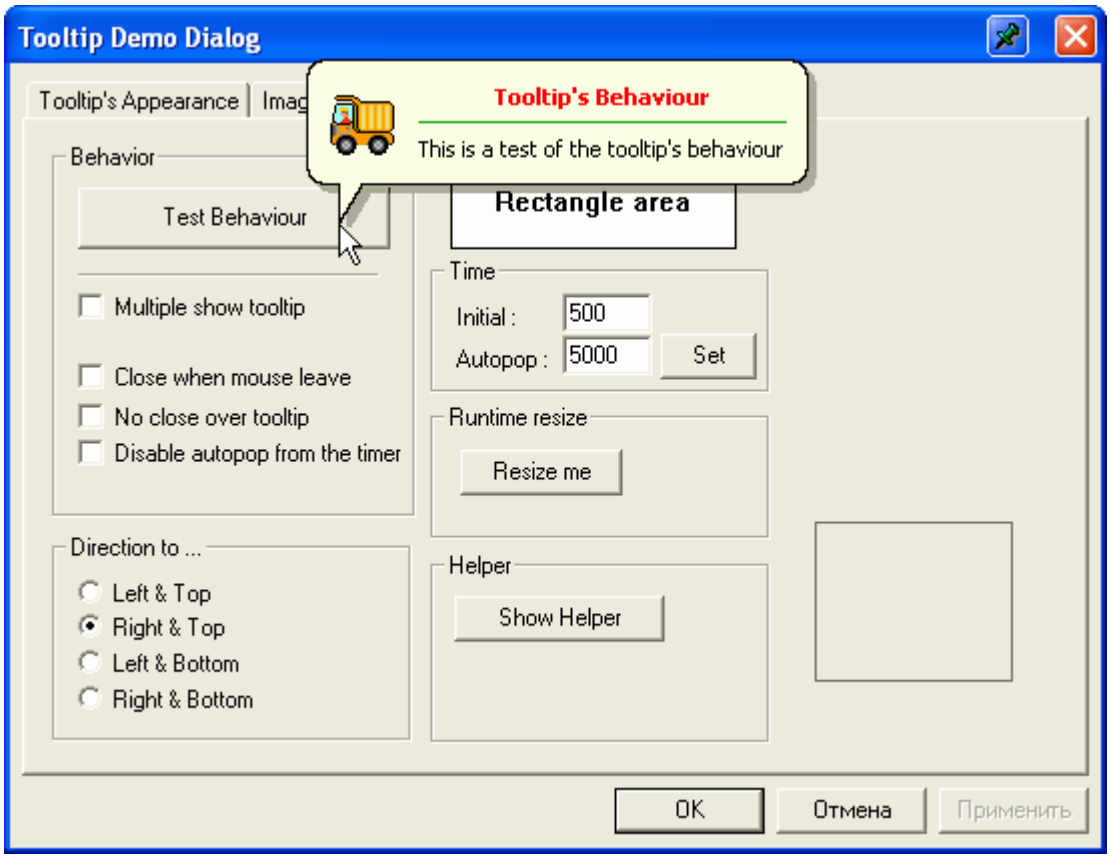
By Eugene Pustovoyt

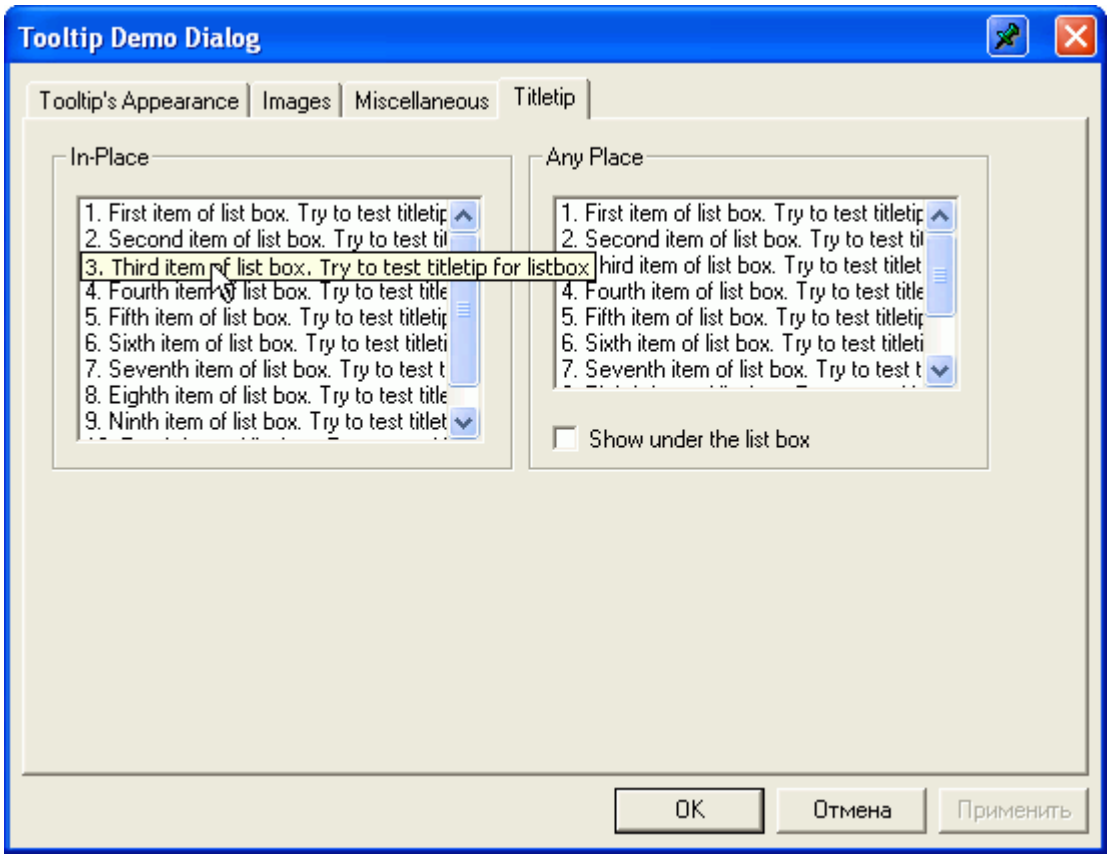
A class that allows you to display your data for a control as tooltip

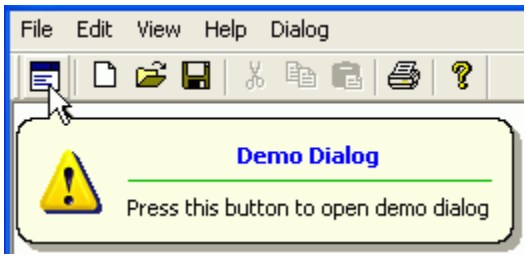
Download source files - 28 Kb
Download demo project - 260 Kb




Tooltip Demo Dialog
Tooltip's Appearance
Images
Miscellaneous
Titletip
Mark Icon
16x16
32x32
48x48
Vertical align :
VCenter
Show mark icon
Images in the tooltip text
Icons
Bitmaps
Image List
Test Area
Button2
Image List
00 =
01 =
02 =
03 =
04 =
05 =
06 =
07 =
08 =
09 =
10 =
11 =
12 =
13 =
14 =
15 =
16 =
17 =
18 =
19 =
20 =
21 =


Tooltip Demo Dialog
Tooltip's Behaviour
This is a test of the tooltip's behaviour
Behavior
Test Behaviour
Rectangle area
Multiple show tooltip
Close when mouse leave
No close over tooltip
Disable autopop from the timer
Time
Initial :
500
Autopop :
5000
Set
Runtime resize
Resize me
Direction to ...
Left & Top
Right & Top
Left & Bottom
Right & Bottom
Helper
Show Helper
OK
Отмена
Применить





Contents

How to integrate CPPToolTip in your application

To integrate PPToolTip control in your application you should add next files to your project:

- **PPToolTip.h**
- **PPToolTip.cpp**
- **CeXDib.h, CeXDib.cpp thanks to Davide Pizzolato and Davide Calabro. This class use for extend background's effect.**

and you must include afxtempl.h file to your StdAfx.h file:

```
#include <afxtempl.h>
```

Extend background effects by Davide Pizzolato and Davide Calabro become available if defined PPTOOLTIP_USE_SHADE**:**

In PPToolTip.h

```
#define PPTOOLTIP_USE_SHADE
```

Create a CPPToolTip object

Include PPToolTip.h in the header file where you want to use the CPPToolTip window and create a member variable for the window:

```
CPPToolTip m_tooltip;
```

Now create the window. For dialog-based applications, in your OnInitDialog:

```
    // Call the base-class method
    CDialog::OnInitDialog();

    // Create the CPPToolTip object
    m_tooltip.Create(this);
```

Now call AddTool **function to register a tool with the tool tip control, so that the information stored in the tool tip is displayed when the cursor is on the tool:**

```
m_tooltip.AddTool(GetDlgItem(IDC_BUTTON1),
                  _T("Tooltip to the control IDC_BUTTON1"));
```

or for rectangle area

```
m_tooltip.AddTool(this, _T("Tooltip for rectangle area"),
                  IDI_TRACK, CRect (100, 100, 200, 200));
```

Now you must add RelayEvent function call to pass a mouse message to a tooltip control for processing.

```
BOOL ... ::PreTranslateMessage(MSG* pMsg)
{
  m_tooltip.RelayEvent(pMsg);
}
```

Feature to use the tooltip with toolbar

1. Create CPPToolTip object in the header CMainFrame

```
CPPToolTip m_tooltip;
```

2. Disables to support the standard tooltip for toolbar. Remove CBRS_TOOLTIPS style from `Create` method of the toolbar.

```
m_wndToolBar.CreateEx(this, TBSTYLE_FLAT, WS_CHILD | WS_VISIBLE | CBRS_TOP
  | CBRS_GRIPPER | /*CBRS_TOOLTIPS |*/ CBRS_FLYBY | CBRS_SIZE_DYNAMIC);
```

3. Create tooltip window in `CMainFrame::OnCreate()` and add the rectangles of the toolbar's buttons to tooltip.

```
int CMainFrame::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
  ...
  m_tooltip.Create(this);

  //Adds tooltip for toolbar
  CRect rcHot;
  int nIndex = m_wndToolBar.CommandToIndex(ID_FILE_OPEN);
  m_wndToolBar.GetItemRect(nIndex, rcHot);
  m_Tooltip.AddTool(&m_wndToolBar, _T("Tooltip for File Open button")
                  , IDI_PP_ATTENTION, rcHot);

  return 0;
}
```

4. Add `RelayEvent` function call to pass a mouse message to a tooltip control for processing.

```
BOOL CMainFrame::PreTranslateMessage(MSG* pMsg)
{
  m_tooltip.RelayEvent(pMsg);
}
```

!!! Attention !!! This version of tooltip isn't correct work if toolbar changed size (multiline, vertical etc.)

Class Members

Construction	
CPPToolTip	**Constructs a `CPPToolTip` object.**
Create	**Creates a tooltip control and attaches it to a `CPPToolTip` object.**

Styles	
SetStyles	**Sets the styles**
ModifyStyles	**Changes the style of a control**
SetDefaultStyles	**Sets the styles in default value**
GetStyles	**Retrieves information about the control styles**

Colors

SetDefaultColors	Sets the control colors to the default colors.
SetColor	Specifies the color to use for the control.
GetColor	Retrieves the color for the control.
SetGradientColors	Specifies the gradient colors to use for the control.
GetGradientColors	Retrieves the gradient colors for the control.

Tools

AddTool	Adds the info of the tool.
GetTool	Retrives the info of the tool.
FindTool	Searches the tool.
IsExistTool	Returns whether the exist tool.
RemoveTool	Removes specified tool.
RemoveAllTools	Removes all tools.
SetAtTool	Sets the info to specified tool
ShowHelpTooltip	Show the tooltip as help window

CImageList

SetImageList	Sets the image list to tooltip.
GetImageList	Gets the image list from tooltip.

Mask of the tool

SetMaskTool	Sets the control colors to the default colors.
ModifyMaskTool	Specifies the color to use for the control.
GetMaskTool	Retrieves the color for the control.

Background's effect

SetEffectBk	Sets the control colors to the default colors.
GetEffectBk	Specifies the color to use for the control.

Notification

SetNotify	Enable notification about changing the data or the view.
GetNotify	Is enabled notification about changing the data or the view..

Font

SetDefaultFont	Sets the control font to the default value (Courier, 8pt.).
SetFont	Specifies the font for the control.
GetFont	Specifies the font for the control.

Delays

SetDelayTime	Sets first visible address in the control

GetDelayTime	**Gets current visible address from the control**

Sizes

SetDefaultSizes	**Sets the control font to the default value (Courier, 8pt.).**
SetSize	**Specifies the font for the control.**
GetSize	**Specifies the font for the control.**

Direction

SetDirection	**Sets first visible address in the control**
GetDirection	**Gets current visible address from the control**

Behaviour

SetBehaviour	**Sets first visible address in the control**
GetBehaviour	**Gets current visible address from the control**

Names of the resources

AddNameOfResource	**Associates name of resource with his ID.**
FindNameOfResource	**Search index of array by with specified name or ID resource.**
RemoveNameOfResource	**Removes specified index of array.**
RemoveAllNamesOfResource	**Removes all elements of array.**

Operations

RelayEvent	**Sets the range of the selected block**

Version

GetVersionI	**Get CPPToolTip version**
GetVersionC	**Get CPPToolTip version**

CPPToolTip::CPPToolTip

```
CPPToolTip ();
```

Remarks
Constructs a CPPToolTip object. You must call Create after that.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::Create

```
BOOL Create (CWnd* pParentWnd, BOOL bBalloonSize = TRUE);
```

Parameters

- `pParentWnd` - **Points to the parent window of the tooltip control, usually a CDialog. It must not be NULL.**
- `bBalloonSize` - **If TRUE CPPToolTip object will be sets to balloon size, otherwise to standard size.**

Return Value

Nonzero if the CPPToolTip object is successfully created; otherwise 0.

Remarks
You construct a CPPToolTip in two steps. First call the constructor to construct the CPPToolTip object; then call Create to create the tool tip control and attach it to the CPPToolTip object.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::SetStyles

```
void SetStyles (DWORD nStyles, int nIndexTool = -1);
```

Parameters

- `nStyle` - **A value containing a combination styles which will be add to a tool or an object of PPToolTip.**
- `nIndexTool` - **The index of the tool array for which tool changes styles. If less 0 styles changes for `PPToolTip` object.**

Remarks

Call this member function to set the styles for a tooltip. This function automatically sets the PPTOOLTIP_MASK_STYLES

CPPToolTip Overview | Class Members | Styles

CPPToolTip::ModifyStyles

```
void ModifyStyles (DWORD nAddStyles, DWORD nRemoveStyles, int nIndexTool = -1)
```

Parameters

- `nAddStyles` - **A value containing a combination styles which will be adds to a tool or an .**
- `nRemoveStyles` - **A value containing a combination styles which will be removes from a tool or an `PPToolTip` object.**
- `nIndexTool` - **The index of the tool array for which tool changes styles. If less 0 styles changes for `PPToolTip` object.**

Remarks

The function modifies styles of the tooltip. This function automatically sets the PPTOOLTIP_MASK_STYLES

CPPToolTip Overview | Class Members | Styles

CPPToolTip::SetDefaultStyles

```
void SetDefaultStyles (int nIndexTool = -1)
```

Parameters

- `nIndexTool` - **The index of the tool array for which tool changes styles. If less 0 styles changes for** `PPToolTip` **object.**

Remarks

Sets all styles to a default value. The function set following styles to tooltip:

PPTOOLTIP_BALLOON

This function automatically sets the PPTOOLTIP_MASK_STYLES

CPPToolTip Overview | Class Members | Styles

CPPToolTip::GetStyles

```
DWORD GetStyles(int nIndexTool = -1)
```

Return value

The tooltip's style.

Parameters

- `nIndexTool` - **The index of the tool array for which tool retrives styles. If less 0 styles retrives from** `PPToolTip` **object.**

Remarks

Retrieves the tooltip's styles.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::SetDefaultColors

```
void SetDefaultColors()
```

Remarks

This member function will set the default COLORREF values for the tooltip.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::SetColor

```
void SetColor(int nIndex, COLORREF crColor)
```

Parameters

- `nIndex` - **Index of the color.**
- `crColor` - **New COLORREF value for the object**

Remarks

This member function will set the color value for the object.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::GetColor

```
COLORREF GetColor(int nIndex)
```

Return value

A COLORREF value that contains the RGB information for the color selected.

Parameters

- `nIndex` - **Index of the color.**

Remarks

Call this function to retrieve the information about the color.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::SetGradientColors

```
void SetGradientColors(COLORREF crBegin, COLORREF crMid, COLORREF crEnd, int
nIndexTool = -1)
```

Parameters

- `crBegin` - **A COLORREF value that contains RGB information for the first color.**
- `crMid` - **A COLORREF value that contains RGB information for the middle color. Used with PPTOOLTIP_EFFECT_3HGRADIENT, PPTOOLTIP_EFFECT_3VGRADIENT, PPTOOLTIP_EFFECT_NOISE, PPTOOLTIP_EFFECT_DIAGSHADE, PPTOOLTIP_EFFECT_HSHADE, PPTOOLTIP_EFFECT_VSHADE, PPTOOLTIP_EFFECT_HBUMP, PPTOOLTIP_EFFECT_VBUMP, PPTOOLTIP_EFFECT_SOFTBUMP, PPTOOLTIP_EFFECT_HARDBUMP, PPTOOLTIP_EFFECT_METAL effects.**
- `crEnd` - **A COLORREF value that contains RGB information for the end color.**
- `nIndexTool` - **The index of the tool array for which tool will be draws with gradient color. If less 0 gradient colors sets for `PPToolTip` object.**

Remarks

Function sets the colors for gradient filling. This function automatically sets the PPTOOLTIP_MASK_COLORS & PPTOOLTIP_MASK_EFFECT

CPPToolTip Overview | Class Members | Styles

CPPToolTip::GetGradientColors

```
void GetGradientColors(COLORREF & crBegin, COLORREF & crMid, COLORREF & crEnd,
int nIndexTool = -1)
```

Return values

- `crBegin` **- A COLORREF value that receives RGB information for the first color.**
- `crMid` **- A COLORREF value that receives RGB information for the middle color.**
- `crEnd` **- A COLORREF value that receives RGB information for the end color.**

Parameters

- `nIndexTool` **- The index of the tool array for which tool retrives gradient colors. If less 0 colors retrives from** `PPToolTip` **object.**

Remarks

Function retrives the colors for gradient filling.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::AddTool

```
void AddTool (CWnd * pWnd, UINT nIdText, HICON hIcon = NULL, LPCRECT lpRectTool
= NULL, UINT nIDTool = 0)
void AddTool (CWnd * pWnd, UINT nIdText, UINT nIdIcon, LPCRECT lpRectTool =
NULL, UINT nIDTool = 0)
void AddTool (CWnd * pWnd, CString sTooltipText, HICON hIcon = NULL, LPCRECT
lpRectTool = NULL, UINT nIDTool = 0)
void AddTool (CWnd * pWnd, CString sTooltipText, UINT nIdIcon, LPCRECT
lpRectTool = NULL, UINT nIDTool = 0)
void AddTool (PPTOOLTIP_INFO & ti)
```

Parameters

- `pWnd` **- Pointer to the window that contains the tool. A value cannot be NULL.**
- `nIdText` **- ID of the string resource that contains the text for the tool.**
- `sToolTipText` **- String of the text for the tool. String can contents tags.**
- `hIcon` **- Identifies the handle of the icon to be drawn. This parameter can be NULL**
- `nIdIcon` **- ID of the icon resource**
- `lpRectTool` **- Pointer to a RECT structure containing coordinates of the tool's bounding rectangle. The coordinates are relative to the upper-left corner of the client area of the**

window identified by pWnd.
- `nIDTool` **- ID of the tool.**
- `ti` **- A pointer to a PPTOOLTIP_INFO structure that specifies the information to set**

Remarks

A tooltip control can be associated with more than one tool. Call this function to register a tool with the tooltip control, so that the information stored in the tooltip is displayed when the cursor is on the tool.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::GetTool

```
BOOL GetTool(int nIndexTool, PPTOOLTIP_INFO & ti)
```

Return Value

Nonzero if successful; otherwise 0.

Parameters

- `nIndexTool` **- The index of the tool in array.**
- `ti` **- the PPTOOLTIP_INFO structure that is filled with information about the tool**

Remarks

Call this function to retrieve the information that a tooltip control associate with tool. If that tool has been registered with the tooltip control through a previous call to AddTool, the PPTOOLTIP_INFO structure is filled with information about the tool.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::FindTool

```
void FindTool (CPoint & pt)
void FindTool (CWnd * pWnd, LPCRECT lpRect = NULL)
void FindTool (UINT nIDTool)
```

Parameters

- `pt` **- The client coordinates of the point**
- `pWnd` **- Pointer to the window that contains the tool. A value cannot be NULL.**
- `lpRectTool` **- Pointer to a RECT structure containing coordinates of the tool's bounding rectangle. The coordinates are relative to the upper-left corner of the client area of the window identified by pWnd.**
- `nIDTool` **- ID of the tool.**

Remarks

This function searches the tool from tool's array. First searches any tool under the point. Second searches tool for pWnd and specified rectangle. Third searches first tool with nIDTool.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::IsExistTool

```
BOOL IsExistTool (int nIndexTool)
```

Return value

TRUE - if tool with specified index is exist

Parameters

- `nIndexTool` **- The testing index of the tool from array.**

CPPToolTip Overview | Class Members | Styles

CPPToolTip::RemoveTool

```
void RemoveTool(int nIndexTool)
```

Parameters

- `nIndexTool` **- The index of the tool from array.**

Remarks

Call this function to remove the tool specified by index.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::RemoveAllTools

```
void RemoveAllTools()
```

Remarks

Call this function to remove the full collection of tools aggregated in a tooltip control.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::SetAtTool

```
void SetAtTool(int nIndexTool, PPTOOLTIP_INFO & ti)
```

Parameters

- `nIndexTool` - **The index of the tool in array.**
- `ti` - **the PPTOOLTIP_INFO structure sets to specified index of the tool**

Remarks

Call this function to set the information associate with tool.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::SetMaskTool

```
void SetMaskTool(int nIndexTool, UINT nMask = 0)
```

Parameters

- `nIndexTool` - **The index of the tool from array.**
- `nMask` - **A value containing a combination of masks which will be add to a tool.**

Remarks

Call this member function to set the masks for a tooltip.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::ShowHelpTooltip

```
void ShowHelpTooltip (CPoint & pt, UINT nIdText, HICON hIcon = NULL)
void ShowHelpTooltip (CPoint & pt, UINT nIdText, UINT nIdIcon)
void ShowHelpTooltip (CPoint & pt, CString sTooltipText, HICON hIcon = NULL)
void ShowHelpTooltip (CPoint & pt, CString sTooltipText, UINT nIdIcon)
void ShowHelpTooltip (CPoint & pt, PPTOOLTIP_INFO & ti)
```

Parameters

- `pt` - **The point of the tooltip's anchor in the client coordinates.**
- `nIdText` - **ID of the string resource that contains the text for the tool.**
- `sToolTipText` - **String of the text for the tool. String can contents tags.**
- `hIcon` - **Identifies the handle of the icon to be drawn. This parameter can be NULL**
- `nIdIcon` - **ID of the icon resource**
- `ti` - **A pointer to a PPTOOLTIP_INFO structure that specifies the information about help tooltip**

Remarks

Call this function to show the tooltip as help window.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::ModifyMaskTool

```
void ModifyMaskTool(int nIndexTool, UINT nAddMask, UINT nRemoveMask)
```

Parameters

- `nIndexTool` - **The index of the tool from array.**
- `nAddMask` - **A value containing a combination of masks which will be added to a tool.**
- `nRemoveMask` - **A value containing a combination of masks which will be removed from a tool.**

Remarks

The function modifies masks of the tooltip.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::GetMaskTool

```
UINT GetMaskTool(int nIndexTool)
```

Return value

A value containing a combination of masks for specified tool.

Remarks

Retrieves the tooltip's masks.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::SetEffectBk

```
void SetEffectBk(UINT nEffect, BYTE nGranularity = 2, int nIndexTool = -1)
```

Parameters

- `nEffect` - **This value sets the effect of the background bitmap. Can be one of the following values:**

```
PPTOOLTIP_EFFECT_SOLID
PPTOOLTIP_EFFECT_HGRADIENT
PPTOOLTIP_EFFECT_VGRADIENT
PPTOOLTIP_EFFECT_HCGRADIENT
PPTOOLTIP_EFFECT_VCGRADIENT
PPTOOLTIP_EFFECT_3HGRADIENT
PPTOOLTIP_EFFECT_3VGRADIENT
PPTOOLTIP_EFFECT_NOISE
PPTOOLTIP_EFFECT_DIAGSHADE
PPTOOLTIP_EFFECT_HSHADE
PPTOOLTIP_EFFECT_VSHADE
PPTOOLTIP_EFFECT_HBUMP
PPTOOLTIP_EFFECT_VBUMP
PPTOOLTIP_EFFECT_SOFTBUMP
PPTOOLTIP_EFFECT_HARDBUMP
```

```
PPTOOLTIP_EFFECT_METAL
```

- nGranularity - this parameter add an uniform noise to the background bitmap. A good value is from 5 to 20; 0 to disable the effect. The noise has a positive effect because it hides the palette steps. This parameter has no effect if the nEffect value is from PPTOOLTIP_EFFECT_SOLID to PPTOOLTIP_EFFECT_3VGRADIENT.
- nIndexTool - The index of the tool from array for which tool will be drawn. If less 0 an effect sets for PPToolTip object.

Remarks

Sets an effect for the background bitmap. Note that effect value from PPTOOLTIP_EFFECT_NOISE to PPTOOLTIP_EFFECT_METAL is available when

```
#define PPTOOLTIP_USE_SHADE 0x1
```

only. This function automatically sets the PPTOOLTIP_MASK_EFFECT.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::GetEffectBk

```
UINT GetEffectBk(int nIndexTool = -1)
UINT GetEffectBk(BYTE & nGranularity, int nIndexTool = -1)
```

Return value

A value containing the effects of the background bitmap. See SetEffectBk for the list of available effects.

Parameters

- nGranularity - this variable retrives the value of the bitmap's noise.
- nIndexTool - The index of the tool from array for which tool will be drawn. If less 0 an effect sets for PPToolTip object.

Remarks

Retrives the effects of the background bitmap for specified tool.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::SetNotify

```
void SetNotify(BOOL bParentNotify = TRUE)
void SetNotify(HWND hWnd)
```

Parameters

- `bParentNotify` - **If TRUE the control will send the notification to parent window. Otherwise the notification will not send.**
- `hWnd` - **If non-NULL the control will send the notification to specified window. Otherwise the notification will not send.**

Remarks

This function enables or disables the ability to send notification messages to destination window from the control.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::GetNotify

```
BOOL GetNotify()
```

Return value

TRUE if the control notified the specified window.

Remarks

This function determines if notification messages are passed to destination window from the control or not.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::SetDefaultFont

```
void SetDefaultFont(BOOL bRedraw /* = TRUE */)
```

Parameters

- `bRedraw` - **Specifies whether the control is to be redrawn. A nonzero value redraws the control. A 0 value does not redraw the control. The control is redrawn by default.**

Remarks

Sets the font to the default value (Courier, 8pt)

CPPToolTip Overview | Class Members | Styles

CPPToolTip::SetFont

```
BOOL SetFont(CFont & font)
BOOL SetFont(LPLOGFONT lf)
BOOL SetFont(LPCTSTR lpszFaceName, int nSizePoints /* = 8 */, BOOL bUnderline /*
= FALSE */, BOOL bBold /* = FALSE */,
             BOOL bStrikeOut /* = FALSE */, BOOL bItalic /* = FALSE */)
```

Parameters

- `font` - A CFont object allowing full control over the appearance of the font.
- `lf` - A LOGFONT structure allowing full control over the appearance of the font also.
- `lpszFaceName` - The font face name.
- `nSizePoints` - The font size in points. The default value is 8.
- `bUnderline` - Specifies whether the font is to be underlined. A nonzero value sets the font to be underlined. A 0 value sets the font not to be underlined. The font is not underlined by default.
- `bBold` - Specifies whether the font weight is to be bold or not. A nonzero value sets the font to be bold. A 0 value sets the font weight to be normal. The font weight is normal by default.
- `bStrikeOut` - Specifies whether the font is to be struck out or not. A nonzero value sets the font to be struck out. A 0 value sets the font not to be struck out. The font is not struck out by default.
- `bItalic` - Specifies whether the font style is to be italic or not. A nonzero value sets the font style to be italic. A 0 value sets the font style to be normal. The font style is normal by default.

Remarks

Sets the font for all elements of the control

Attention! You must set the font with fixed width (for example: Courier)

CPPToolTip Overview | Class Members | Styles

CPPToolTip::GetFont

```
void GetFont(CFont & font)
void GetFont(LPLOGFONT lf)
```

Parameters

- `font` - A CFont object allowing full control over the appearance of the font.
- `lf` - A LOGFONT structure allowing full control over the appearance of the font also.
- `lpszFaceName` - The font face name.
- `nSizePoints` - The font size in points. The default value is 8.
- `bUnderline` - Specifies whether the font is to be underlined. A nonzero value sets the font to be underlined. A 0 value sets the font not to be underlined. The font is not underlined by default.
- `bBold` - Specifies whether the font weight is to be bold or not. A nonzero value sets the font to be bold. A 0 value sets the font weight to be normal. The font weight is normal by default.
- `bStrikeOut` - Specifies whether the font is to be struck out or not. A nonzero value sets the font to be struck out. A 0 value sets the font not to be struck out. The font is not struck out by default.
- `bItalic` - Specifies whether the font style is to be italic or not. A nonzero value sets the font style to be italic. A 0 value sets the font style to be normal. The font style is normal by default.

Remarks

Sets the font for all elements of the control

Attention! You must set the font with fixed width (for example: Courier)

CPPToolTip Overview | Class Members | Styles

CPPToolTip::SetDelayTime

```
void SetDelayTime(DWORD dwDuration, UINT nTime)
```

Parameters

- `dwDuration` - **Flag that specifies which duration value will be set. This parameter can be one of the following values:**

  TTDT_AUTOPOP - Sets the length of time the tool tip window remains visible if the pointer is stationary within a tool's bounding rectangle.
  TTDT_INITIAL - Sets the length of time the pointer must remain stationary within a tool's bounding rectangle before the tool tip window appears. Recomended don't use this value less 100.
- `nTime` - **The specified delay time, in milliseconds.**

Remarks

Call this function to set the delay time for a tool tip control. The delay time is the length of time the cursor must remain on a tool before the tool tip window appears.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::GetDelayTime

```
UINT GetDelayTime(DWORD dwDuration) const
```

Return Value

The specified delay time, in milliseconds

Parameters

- `dwDuration` - **Flag that specifies which duration value will be retrieved. This parameter can be one of the following values:**

  TTDT_AUTOPOP - Retrieve the length of time the tool tip window remains visible if the pointer is stationary within a tool's bounding rectangle.
  TTDT_INITIAL - Retrieve the length of time the pointer must remain stationary within a tool's bounding rectangle before the tool tip window appears.

Remarks

Call this function to get the delay time for a tool tip control.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::SetDefaultSizes

```
void SetDefaultSizes(BOOL bBalloonSize = TRUE)
```

Parameter

- `bBalloonSize` - **If TRUE CPPToolTip object will be sets to balloon size, otherwise to standard size.**

Remarks

This function sets all sizes as default.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::SetSize

```
void SetSize(int nSizeIndex, UINT nValue)
```

Parameters

- `nSizeIndex` - **Index of the size.**
- `nValue` - **The size in pixels.**

Remarks

Call this member function to set the tooltip's sizes.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::GetSize

```
UINT GetSize(int nSizeIndex)
```

Return value

The size of the specified value.

Parameters

- `nSizeIndex` - **Index of the size.**

Remarks

This method returns the size of the specified value.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::SetDirection

```
void SetDirection(UINT nDirection = PPTOOLTIP_RIGHT_BOTTOM, int nIndexTool = -1)
```

Parameters

- `nDirection` - **mouse-pointer relative placement direction of the tooltip. This parameter can be one of the following values:**
  - **PPTOOLTIP_LEFT_TOP**
  - **PPTOOLTIP_RIGHT_TOP**
  - **PPTOOLTIP_LEFT_BOTTOM**
  - **PPTOOLTIP_RIGHT_BOTTOM**
- `nIndexTool` - **The index of the tool from array for which tool placement will be applied. If less 0 the direction sets for** `PPToolTip` **object.**

Remarks

This function sets the default direction of the tooltip from mouse pointer. This function automatically sets the PPTOOLTIP_MASK_DIRECTION.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::GetDirection

```
UINT GetDirection(int nIndexTool = -1)
```

Return value

The direction of the tooltip's placement from mouse-pointer. See SetDirection for the list of available directions.

Parameters

- `nIndexTool` - **The index of the tool from array for which tool placement will be applied. If less 0 the direction sets for** `PPToolTip` **object.**

Remarks

This function retrives the direction for the specified tool or tooltip.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::SetBehaviour

```
void SetBehaviour(UINT nBehaviour = 0, int nIndexTool = -1)
```

Parameters

- `nBehaviour` - **The behaviour of the tooltip. This parameter can be one of the following values:**
  - **PPTOOLTIP_MULTIPLE_SHOW - This option sets multiple show of the tooltip for specified tool or all tools of tooltip object. With this option the tooltip will be shown for each stop of the mouse pointer.**
  - **PPTOOLTIP_CLOSE_LEAVEWND - If set this flag then tooltip will be hide when**

mouse pointer to leave the control.
- PPTOOLTIP_NOCLOSE_OVER - If set this flag then tooltip will not hide if the mouse pointer over the tooltip
- PPTOOLTIP_DISABLE_AUTOPOP - a tooltip with this flag will not hide from the autopop timer.

- `nIndexTool` - The index of the tool from array for which tool's behaviour will be set. If less 0 the behaviour sets for `PPToolTip` object.

Remarks

Calls the function to set the tooltip's behaviour.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::GetBehaviour

```
UINT GetBehaviour(int nIndexTool = -1)
```

Return value

The behavior of the tooltip. See the SetBehaviour function for the list of the available flags

Parameters

- `nIndexTool` - The index of the tool from array for which tool behaviour will be get. If less 0 the behaviour gets from `PPToolTip` object.

Remarks

Retrieves the behavior of the tooltip.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::RelayEvent

```
void RelayEvent(MSG* pMsg)
```

Parameters

- `pMsg` - Pointer to a MSG structure that contains the message to relay.

Remarks

Call this function to pass a mouse message to a tooltip control for processing. A tooltip control processes only the following messages, which are sent to it by `RelayEvent`:

WM_LBUTTONDOWN	WM_MOUSEMOVE
WM_LBUTTONUP	WM_RBUTTONDOWN
WM_MBUTTONDOWN	WM_RBUTTONUP
WM_MBUTTONUP	

CPPToolTip Overview | Class Members | Styles

CPPToolTip::SetImageList

```
void SetImageList(UINT nIdBitmap, int cx, int cy, int nCount, COLORREF crMask /*
= RGB(255, 0, 255) */)
void SetImageList(HBITMAP hBitmap, int cx, int cy, int nCount, COLORREF
crMask /* = RGB(255, 0, 255) */)
```

Parameters

- `nIdBitmap` - **Resource IDs of the bitmap to be associated with the image list**
- `hBitmap` - **Handle of the bitmap to be associated with the image list**
- `cx` - **Dimensions of each image, in pixels.**
- `cy` - **Dimensions of each image, in pixels.**
- `nCount` - **Number of images that the image list initially contains.**
- `crMask` - **Color used to generate a mask. Each pixel of this color in the specified bitmap is changed to black, and the corresponding bit in the mask is set to one.**

Remarks

This function sets the image list to tooltip. After this function in body of the tooltip string you can to use the <ilst> tag which will be draw the image from image list in specified place.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::GetImageList

```
CImageList * GetImageList(CSize & sz)
```

Return values

The pointer to the image list and the dimensions of the single image

Remarks

This function return the pointer to the image list.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::AddNameOfResource

```
void AddNameOfResource(CString sName, UINT nID, BYTE nTypeRes /* =
TYPE_RES_TRAN_BITMAP */, COLORREF crMask /* = RGB(255, 0, 255) */)
```

Parameters

- `sName` - **A name of a resource in format string of the tooltip**
- `nID` - **A resource ID associated with name.**
- `nTypeRes` - **A type of a resource:**

    - **TYPE_RES_ICON - an icon**
    - **TYPE_RES_BITMAP - a bitmap**
    - **TYPE_RES_MASK_BITMAP - a masked bitmap**
- `crMask` **- Color used to generate a mask. Each pixel of this color in the specified bitmap is changed to black, and the corresponding bit in the mask is set to one.**

Remarks

This function associates name of the resource with ID. After this function in body of the tooltip string you can to use the `<img>` tag which will be draw the specified image in specified place.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::FindNameOfResource

```
int FindNameOfResource(CString sName)
int FindNameOfResource(UINT nID)
```

Return value

An index of the element of array which to contents specified name or ID of resource

Parameters

- `sName` **- A name of a resource in format string of the tooltip**
- `nID` **- A resource ID associated with name.**

Remarks

This function search index of the element of array to contents name or ID resource.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::RemoveNameOfResource(int nIndex)

```
void RemoveNameOfResource(int nIndex)
```

Parameters

- `nIndex` **- An index of the element of array to removes.**

Remarks

This function removes specified element from array.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::RemoveAllNameOfResource()

```
void RemoveAllNameOfResource()
```

Remarks

This function removes all elements from array.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::GetVersionI()

```
short GetVersionI()
```

Return Value

Number version of CPPToolTip.

Remarks

Returns the CPPToolTip version as a short number. Divide by 10 to get actual version.

CPPToolTip Overview | Class Members | Styles

CPPToolTip::GetVersionC()

```
LPCTSTR GetVersionC()
```

Return Value

Number version of CPPToolTip as string.

Remarks

Returns the CPPToolTip version as a string.

CPPToolTip Overview | Class Members | Styles

Constants

Styles:

PPTOOLTIP_ANCHOR	**Draws an tooltip's anchor**
PPTOOLTIP_SHADOW	**Draws a tooltip's shadow**
PPTOOLTIP_ROUNDED	**Draws a rounded rectangle of the tooltip**
PPTOOLTIP_BALLOON	**Combination of the PPTOOLTIP_ANCHOR, PPTOOLTIP_SHADOW and PPTOOLTIP_ROUNDED styles**
PPTOOLTIP_SHOW_INACTIVE	**The tooltip will shown even the window is inactive**

PPTOOLTIP_SHOW_DISABLED	**The tooltip will shown even the control disabled**
PPTOOLTIP_ALIGN_VCENTER	**The tooltip's text align to vertical center**
PPTOOLTIP_ALIGN_BOTTOM	**The tooltip's text align to bottom**
PPTOOLTIP_ICON_ALIGN_VCENTER	**The tooltip's icon align to vertical center**
PPTOOLTIP_ICON_ALIGN_BOTTOM	**The tooltip's icon align to bottom**

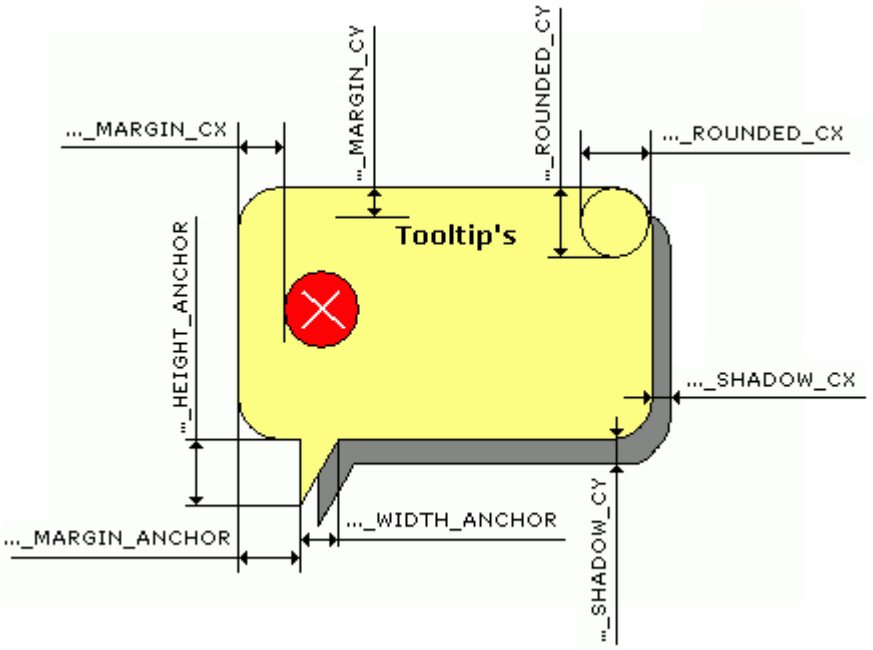
Masks:

PPTOOLTIP_MASK_STYLES	**Uses the tool's styles, else uses the tooltip's style**
PPTOOLTIP_MASK_EFFECT	**Uses the tool's effect, else uses the tooltip's effect**
PPTOOLTIP_MASK_COLORS	**Uses the tool's colors, else uses the tooltip's colors**
PPTOOLTIP_MASK_DIRECTION	**Uses the tool's direction, else uses the tooltip's direction**
PPTOOLTIP_MASK_BEHAVIOUR	**Uses the tool's behaviour, else uses the tooltip's behaviour**

Sizes:

PPTTSZ_ROUNDED_CX, PPTTSZ_ROUNDED_CY	**Uses the tool's styles, else uses the tooltip's style**
PPTTSZ_MARGIN_CX, PPTTSZ_MARGIN_CY	**Uses the tool's effect, else uses the tooltip's effect**
PPTTSZ_SHADOW_CX, PPTTSZ_SHADOW_CY	**Uses the tool's colors, else uses the tooltip's colors**
PPTTSZ_WIDTH_ANCHOR	**Uses the tool's direction, else uses the tooltip's direction**
PPTTSZ_HEIGHT_ANCHOR	**Uses the tool's behaviour, else uses the tooltip's behaviour**
PPTTSZ_MARGIN_ANCHOR	**Uses the tool's behaviour, else uses the tooltip's behaviour**
PPTTSZ_BORDER_CX, PPTTSZ_BORDER_CY	**Uses the tool's behaviour, else uses the tooltip's behaviour**
PPTTSZ_MAX_SIZES	**Max quantities of the sizes**

For more information see following picture:



Colors:

PPTOOLTIP_COLOR_0	**Color with index 0**
PPTOOLTIP_COLOR_1	**Color with index 1**
PPTOOLTIP_COLOR_2	**Color with index 2**
PPTOOLTIP_COLOR_3	**Color with index 3**
PPTOOLTIP_COLOR_4	**Color with index 4**
PPTOOLTIP_COLOR_5	**Color with index 5**
PPTOOLTIP_COLOR_6	**Color with index 6**
PPTOOLTIP_COLOR_7	**Color with index 7**
PPTOOLTIP_COLOR_8	**Color with index 8**
PPTOOLTIP_COLOR_9	**Color with index 9**
PPTOOLTIP_COLOR_10	**Color with index 10**
PPTOOLTIP_COLOR_11	**Color with index 11**
PPTOOLTIP_COLOR_12	**Color with index 12**
PPTOOLTIP_COLOR_13	**Color with index 13**
PPTOOLTIP_COLOR_14	**Color with index 14**
PPTOOLTIP_COLOR_15	**Color with index 15**
PPTOOLTIP_COLOR_FG	**Color of the text**
PPTOOLTIP_COLOR_BK_BEGIN	**General color of the solid background or first color of the effect background**
PPTOOLTIP_COLOR_BK_MID	**Second color of the effect background**
PPTOOLTIP_COLOR_BK_END	**Third color of the effect background**
PPTOOLTIP_COLOR_LINK	**Hyperlink's color (***reserved***, don't use)**
PPTOOLTIP_COLOR_VISITED	**Visited hyperlink's color (***reserved***, don't use)**
PPTOOLTIP_COLOR_HOVER	**Hot area's color (***reserved***, don't use)**
PPTOOLTIP_COLOR_SHADOW	**Shadow's color**
PPTOOLTIP_COLOR_BORDER	**Border's color**

PPTOOLTIP_MAX_COLORS	Ìàõ **quantities of the colors**

Directions:

PPTOOLTIP_LEFT_TOP	**The tooltip draws to left and top from mouse pointer**
PPTOOLTIP_RIGHT_TOP	**The tooltip draws to right and top from mouse pointer**
PPTOOLTIP_LEFT_BOTTOM	**The tooltip draws to left and bottom from mouse pointer**
PPTOOLTIP_RIGHT_BOTTOM	**The tooltip draws to right and bottom from mouse pointer**
PPTOOLTIP_MAX_DIRECTIONS	**Max quantities of the directions**

Background effects:

PPTOOLTIP_EFFECT_SOLID	**Solid background (PPTOOLTIP_COLOR_BK_BEGIN)**
PPTOOLTIP_EFFECT_HGRADIENT	**Horizontal gradient background (from PPTOOLTIP_COLOR_BK_BEGIN to PPTOOLTIP_COLOR_BK_END)**
PPTOOLTIP_EFFECT_VGRADIENT	**Vertical gradient background (from PPTOOLTIP_COLOR_BK_BEGIN to PPTOOLTIP_COLOR_BK_END)**
PPTOOLTIP_EFFECT_HCGRADIENT	**Centered horizontal gradient background (from PPTOOLTIP_COLOR_BK_BEGIN to PPTOOLTIP_COLOR_BK_END to PPTOOLTIP_COLOR_BK_BEGIN)**
PPTOOLTIP_EFFECT_VCGRADIENT	**Centered vertical gradient background (from PPTOOLTIP_COLOR_BK_BEGIN to PPTOOLTIP_COLOR_BK_END to PPTOOLTIP_COLOR_BK_BEGIN)**
PPTOOLTIP_EFFECT_3HGRADIENT	**3 Color Horizontal background (from PPTOOLTIP_COLOR_BK_BEGIN to PPTOOLTIP_COLOR_BK_MID to PPTOOLTIP_COLOR_BK_END)**
PPTOOLTIP_EFFECT_3VGRADIENT	**3 Color Vertical background (from PPTOOLTIP_COLOR_BK_BEGIN to PPTOOLTIP_COLOR_BK_MID to PPTOOLTIP_COLOR_BK_END)**
PPTOOLTIP_EFFECT_NOISE	**Special effects are available with CCeXDib class from Davide Pizzolato and Davide Calabro**
PPTOOLTIP_EFFECT_DIAGSHADE	
PPTOOLTIP_EFFECT_HSHADE	
PPTOOLTIP_EFFECT_VSHADE	
PPTOOLTIP_EFFECT_HBUMP	
PPTOOLTIP_EFFECT_VBUMP	
PPTOOLTIP_EFFECT_SOFTBUMP	
PPTOOLTIP_EFFECT_HARDBUMP	
PPTOOLTIP_EFFECT_METAL	
PPTOOLTIP_MAX_EFFECTS	**Maximum quantity effects**

Structures

PPTOOLTIP_INFO

The PPTOOLTIP_INFO structure contains information about a tool in a PPToolTip control.

```
typedef struct tagPPTOOLTIP_INFO
{
  HWND hWnd;               // Window handle of the control
  UINT nIDTool;            // ID of tool
  CRect rectBounds;        // Bounding rect for toolinfo to be displayed.
  HICON hIcon;
  CString sTooltip;
  UINT nMask;
  UINT nStyles;
  UINT nDirection;
  UINT nEffect;
  UINT nBehaviour;
  BYTE nGranularity;
  COLORREF crBegin;
  COLORREF crMid;
  COLORREF crEnd;
} PPTOOLTIP_INFO;
```

Members

- `hWnd` - **Window handle of the control**
- `nIDTool` - **ID of tool**
- `rectBounds` - **Bounding rect for toolinfo to be displayed. The coordinates are relative to the upper-left corner of the client area of the window identified by hWnd.**
- `hIcon` - **The icon of the tooltip**
- `sTooltip` - **The string of the tooltip**
- `nMask` - **The mask**
- `nStyles` - **The tooltip's styles**
- `nDirection` - **Direction display the tooltip relate cursor point**
- `nEffect` - **The color's type or effects**
- `nBehaviour` - **The tooltip's behaviour**
- `nGranularity` - **The effect's granularity**
- `crBegin` - **Begin Color**
- `crMid` - **Mid Color**
- `crEnd` - **End Color**

Messages

If user enabled notify message passing from tooltip to owner class then tooltip object sends the UDM_TOOLTIP_DISPLAY notification before displaying. In a handler of this notify owner class can change any parameters of tooltip displaying. With UDM_TOOLTIP_DISPLAY notification NM_PPTOOLTIP_DISPLAY structure is transferred, contained all necessary information.

Structure NM_PPTOOLTIP_DISPLAY looks as follows:

NM_PPTOOLTIP_DISPLAY

The NM_PPTOOLTIP_DISPLAY structure contains information how to display the tooltip.

```
typedef struct tagNM_PPTOOLTIP_DISPLAY
{
   NMHDR hdr;
   //The current screen coordinates of the tooltip's anchor
   CPoint * pt;
   //The pointer to the PPTOOLTIP_INFO structure which contents
   //parameters for displaying tooltip
   PPTOOLTIP_INFO * ti;

} NM_PPTOOLTIP_DISPLAY;
```

Owner object should process notification messages to have the ability to change tooltip appearance and placement by changing appropprite members of this structure.

Here example:

Add next lines to header and implementation files accordingly.

```
afx_msg void NotifyDisplayTooltip(NMHDR * pNMHDR, LRESULT * result);


ON_NOTIFY (UDM_TOOLTIP_DISPLAY, NULL, NotifyDisplayTooltip)
```

and

```
void CParentDlg::NotifyDisplayTooltip(NMHDR * pNMHDR, LRESULT * result)
{
         *result = 0;
         //Gets pointers to structure NM_PPTOOLTIP_DISPLAY
         NM_PPTOOLTIP_DISPLAY * pNotify = (NM_PPTOOLTIP_DISPLAY*)pNMHDR

         switch (CWnd::FromHandle(pNotify->ti->hWnd)->GetDlgCtrlID())
         {
         case IDC_BUTTON1:
         //Changes the tooltip's text
            pNotify->ti->sTooltip = _T("Dynamically changed text for BUTTON1");
           break;
         case IDC_BUTTON2:
         //Changes background color of the tooltip as RED
            pNotify->ti->crBegin = RGB (255, 0, 0);
            break;
         }
}
```

Text format

I have develop a simple text formatting language, similar to HTML. It consists of a small number of tags which allows to operate with text in tooltip.

Here is a list of these tags with short description:

Tags	Description
text	**"text" will be drawn as bold**
text	**"text" will be drawn as *italic***
<u>**text**</u>	**"text" will be drawn as underline**

<s>text</s>	"text" will be drawn as ~~strikeout~~
<ct=0x0000FF>	Text's color after this tag will be Red (RGB(0xFF, 0x00, 0x00)). You can change RGB value to set other color
<cti=5>	Text's color after this tag corresponds to the PPTOOLTIP_COLOR_5 from colors' table. You can change this index to select other color from the table
<cb=0xFFFF00>	Background color after this tag will be Cyan (RGB(0x00, 0xFF, 0xFF)). You can change RGB value to set other color
<cbi=7>	Background color after this tag corresponds to the PPTOOLTIP_COLOR_7 from colors' table. You can change this index to select other color from the table
<al> or <al_l>	Text will be aligned to the left
<al_c>	Text will be centered
<al_r>	Text will be aligned to the right
<hr=100%>	Draws horizontal line with length of whole text area in the tooltip
<hr=50>	Horizontal line with length of 50 point.
<t> or <t=1> or \t	1 tab
<t=5>	5 tabs
  or <br=1> or \n	1 new line
<br=3>	3 new lines

Images

<img=IDB_BITMAP1>	In this place will be drawn the image specified by parameter (in example: Draws the image associate with IDB_BITMAP1 string). Associating name and ID resource makes with **AddNameOfResource** method *Format:* <img=name of resource cx=width cy=height> - name of resource is a resource's name specified in **AddNameOfResource** method. If specified name of resource is not exist than image is not displayed. - Width of the image. 0 or absence for the original size. If the specified width does not coincide with the original size the image will be scaled by horizontal - Height of the image. 0 or absence for the original size. If the specified width does not coincide with the original size the image will be scaled by vertical *For example:* *<img=MultiIcon>* - Draws the MultiIcon image with original sizes. *<img=MultiIcon cx=72>* - Draws the MultiIcon with scaling by horizontal *<img=MultiIcon cy=72>* - Draws the MultiIcon with scaling by vertical *<img=MultiIcon cx=48 cy=48>* - Draws the MultiIcon with scaling. If MultiIcon is a multiple icon than the icon with the specified size without scaling is drawn
<ilst=5>	In this place will be drawn the image No.5 from image list if it is exist.

	Image list sets by SetImageList method. ***Format:*** `<ilst=`**image's index of image list**`>` **- If specified image's index is not exist in an image list than image is not displayed.** ***For example:*** ***<ilst=10>*** **- Draws the 10th image from image list specified by SetImageList method.**
`<icon=138>`	**In this place will be drawn the icon with ID = #138.** ***Format:*** `<icon=`**iconID** `cx=`**width** `cy=`**height**`>` **- iconID is an identificator of icon resource. If specified resource is not available than image is not displayed.** **- Width of the icon. 0 or absence for the original size. If the specified width does not coincide with the original size the icon will be scaled by horizontal.** **- Height of the icon. 0 or absence for the original size. If the specified width does not coincide with the original size the icon will be scaled by vertical.** **If iconID is a multiple icon than the icon with the specified size without scaling is drawn.** ***For example:*** ***<icon=138>*** **- Draws the icon with ID=138 with original sizes.** ***<icon=138 cx=72>*** **- Draws the icon with scaling by horizontal** ***<icon=138 cy=72>*** **- Draws the icon with scaling by vertical** ***<icon=138 cx=48 cy=48>*** **- Draws the icon with scaling. If icon is a multiple icon than the icon with the specified size without scaling is drawn**
`<bmp=136 mask=0xFF00FF>`	**In this place will be drawn the bitmap with ID = #136.** ***Format:*** `<bmp=`**bitmap's ID** `mask=`**mask color** `cx=`**width** `cy=`**height**`>` **- bitmap's ID is an identificator of bitmap resource. If specified resource is not available than image is not displayed.** **- mask color is color of a bitmap's mask. If this value is not specified than default color of a mask = RGB (255, 0, 255).** **- Width of the image. 0 or absence for the original size. If the specified width does not coincide with the original size the image will be scaled by horizontal** **- Height of the image. 0 or absence for the original size. If the specified width does not coincide with the original size the image will be scaled by vertical** ***For example:*** ***<bmp=136>*** **- Display the bitmap with ID=136 without color of a mask.** ***<bmp=136 mask>*** **- Display the bitmap with ID=136 with default color of a mask as RGB(255, 0, 255)** ***<bmp=136 mask=0xFF0000>*** **- Display the bitmap with ID=136 with specified color of a mask as RGB(0, 0, 255)** ***<bmp=136 cx=40 cy=40>*** **- Draws the icon with scaling but without color mask**

For example, string `_T("<al_c><b>Example</b><br><hr=100%><br=2><al><t><ct=0xFF0000>Here placed the tooltip<br>text.")` **will be drawn as follow:**

Example

```
Here placed the tooltip
text
```

History

14 Feb 2003	First release
17 Feb 2003	Released version 1.1 - Added new styles PPTOOLTIP_SHOW_DISABLED and PPTOOLTIP_SHOW_INACTIVE - Added new feature to Create() and SetDefaultSizes() methods for set the type of the tooltip's size. - Fixed bug with 0 TTDT_AUTOPOP value. - Fixed bug shows the tooltip after switch application on Alt+Tab
19 Feb 2003	Released version 1.2 - Added support the tooltip for rectangles - Added new FindTool, IsExistTool and SetAtTool methods. - Many methods was updated. - New tool's collection basis on CArray instead CMap - Added three members to PPTOOLTIP_INFO structures - Removed the pointer to the window from NM_PPTOOLTIP_DISPLAY structure - Added new demo for implementation the tooltip with toolbars
02 Apr 2003	Released version 1.3 - Added support the tooltip as help window (see ShowHelpTooltip) - Added the vertical aligns for the icon (see Styles) - Added support the icons with difference sizes (16x16, 32x32, 48x48 etc.) - Added new behaviours - Now the tooltip is not show if the PPTOOLTIP_INFO members as sTooltip is empty and hIcon is NULL. - Added two tags <img>, <ilst> to support draw a bitmaps and an icons inside a tooltip's text - A few minor update
13 Apr 2003	Released version 1.4 - Added two tags <bmp> and <icon> to support draw a bitmaps and an icons inside a tooltip's text - Scales images which drawn with <img>, <bmp> and <icon> tags. - Added new behaviour (PPTOOLTIP_DISABLE_AUTOPOP) - A few minor update

Thanks to ...

- Yaroslav Petrikevich for your help with writing this file.
- Davide Calabro for his class CButtonST in which decisions of some questions have been found.
- Chris Maunder for his articles devoted subclassing of the control.
- Tomasz Sowinski for his help with Tooltip.
- Michael Ushakov his help and advices on work with rectangles and toolbars.
- Many people assisting to me the answers on CodeProject's forum.

Known Problems

- I understand, that use of new AddNameOfResource method is inconvenient and that the developer is obliged to not forget to specify all images used in a tooltip. But this unique

way to realize a legible string of a tooltip now. If you will find a way to read resource on his name let me know and I shall correct it in the following version.

- **There is a problem with display of a tooltip for disabled controls, static controls and rectangle areas edged in groupbox. To display a tooltip for such controls it is necessary that TabOrder groupbox were more than TabOrder specified controls, contained in him.**
- **If you can help me, please let me know so that I can incorporate them into the next release.**

Planned Enhancements

- **If you have any other suggested improvements, please let me know so that I can incorporate them into the next release.**

Contacting the Author

Please send any comments or bug reports to me via email.

Eugene Pustovoyt

Click here to view Eugene Pustovoyt's online profile.

Discussions and Feedback

185 comments have been posted for this article. Visit http://www.codeproject.com/miscctrl/pptooltip.asp to post and view comments on this article.

All Topics, MFC / C++ >> Miscellaneous Controls >> Tooltips
Updated: 9 May 2003