TMS TAdvOfficePager DEVELOPERS GUIDE

July 2019

Web: https://www.tmssoftware.com
Email: info@tmssoftware.com

Index

Availability 3
Online references 3
Purchase a license 3
Main features 4
Getting started 5
TAdvOfficePager styles 8
TAdvFormStyler, TAdvAppStyler 9
Built-in optional buttons on the pager 11
Tab appearance & behavior 13
TabPosition 13
TabSettings 13
Tab reordering 15
Tab undocking 15
Hotkeys 17
Events 18
Page properties 19
Persisting tab positions 20
Other page management methods 20
Tips and FAQ 22

Availability

TMS TAdvOfficePager is a VCL component for Win32 & Win64 application development and is available for Embarcadero Delphi 7, 2006, 2007, 2009, 2010, XE, XE2, XE3, XE4, XE5, XE6, XE7, XE8, XE9, 10.0, 10.1, 10.2, 10.3 & Embarcadero C++Builder 2006, 2007, 2009, 2010, XE, XE2, XE3, XE4, XE5, XE6, XE7, XE8, XE9, 10.0, 10.1, 10.2, 10.3.

Online references

TMS software website:
https://www.tmssoftware.com

TMS TAdvOfficePager page:
https://www.tmssoftware.com/site/aop.asp

Purchase a license

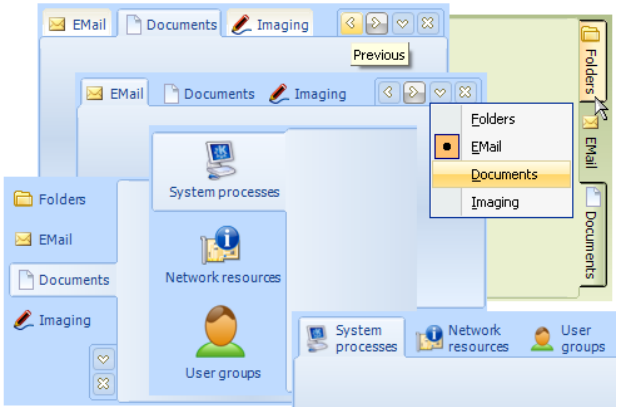
TMS TAdvOfficePager is available in the following bundles:

- TMS Panels Pack : http://www.tmssoftware.com/site/panels.asp
- TMS VCL UI Pack: http://www.tmssoftware.com/site/tmsvcluipack.asp
- TMS Component Studio: http://www.tmssoftware.com/site/studio.asp
- TMS VCL Subscription: http://www.tmssoftware.com/site/vdsub.asp

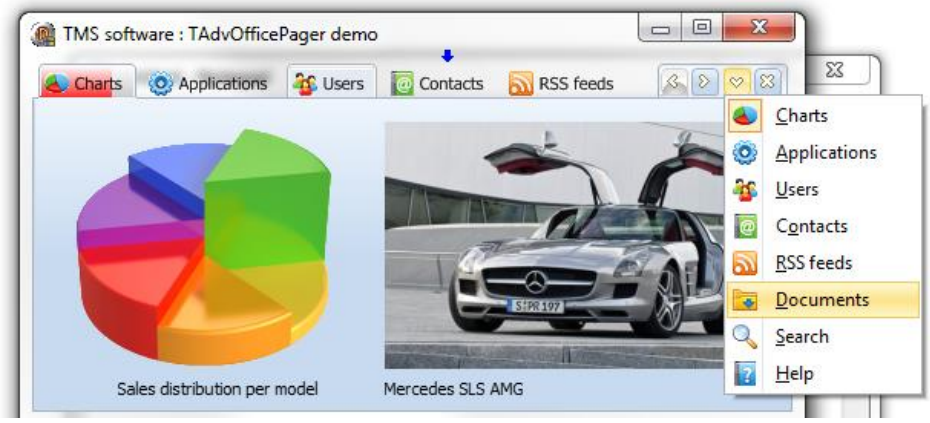
Main features

- Metro / Office 2013 / Office 2010 / Office 2007 / Office 2003 color styles
- Vertical or horizontal oriented tabs on left / right
- Page close button on active tab or fixed close button
- Page list button
- Tab scroll buttons
- Closed page list button
- Support to be used on glass frame
- Progress indicator on tabs
- Tab images via imagelist or PNG with anti aliasing & alpha opacity layer support
- Tab reordering with indicator showing new position
- Optional fixed tabs versus scrolling tabs
- Glow border attention indicator on tabs
- Support for Office 2007 / 2010 style hints per tab
- Can show non selected tabs as text or soft tab
- Smooth glow tab transitions
- Implements ITMSStyle interface for compatibility with TAdvFormStyler, TAdvAppStyler
- Implements ITMSTones interface for form-wide, app-wide Metro style config
- Persistence of tab/page order

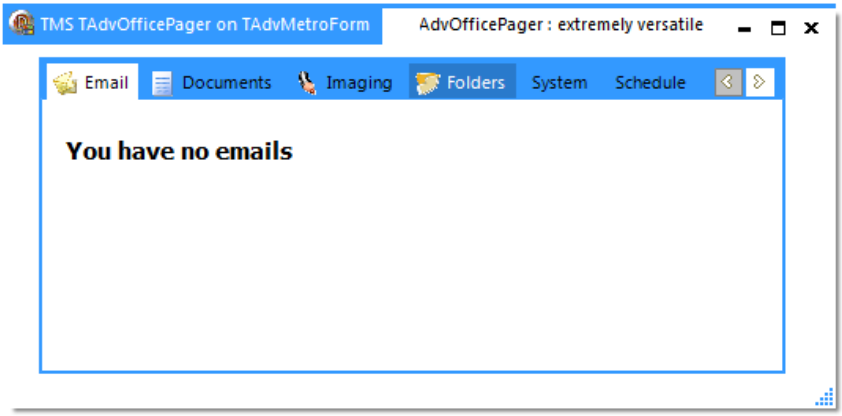
Sample Office 2007 / 2010 appearances:



Progress bar on tab, tab reordering, page list:



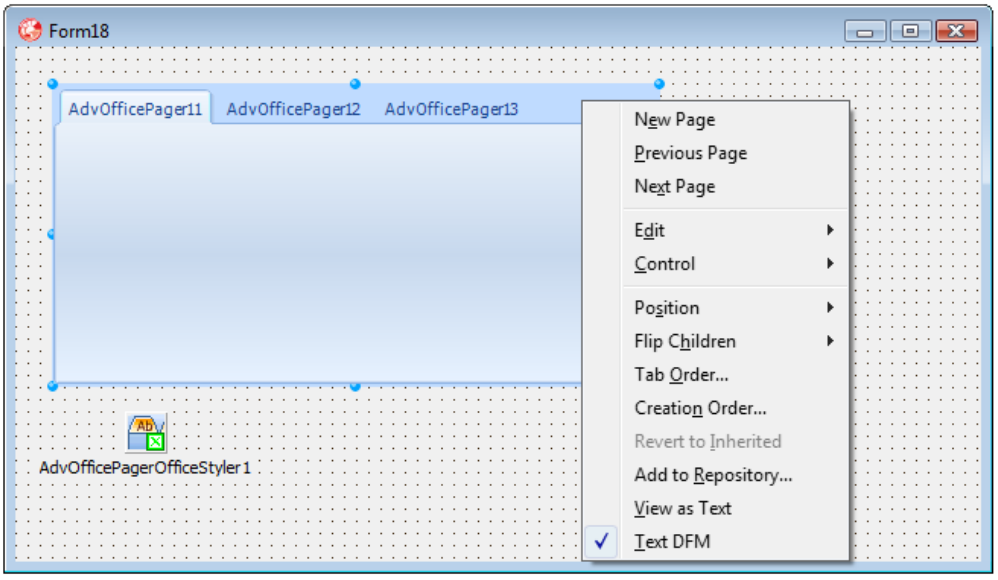
Metro style TAdvOfficePager on TAdvMetroForm:



Getting started

Drop a TAdvOfficePager on the form and right-click to add pages to the TAdvOfficePager via the content menu[1].

Add a TAdvOfficePagerOfficeStyler on the form and assign the component to TAdvOfficePager.AdvOfficePagerStyler. Select the style with TAdvOfficePagerOfficeStyler.Style.



The equivalent code to insert a page programmatically is:

[1] TIP: When using the TMS Component Pack and the context menu items to add pages does not appear, this means you have not installed the design time package. Make sure that the package TMSDExx.DPK is effectively installed and active in the IDE and this issue will be solved.

```
var
  aop: TAdvOfficePage;

begin
  aop := TAdvOfficePage.Create(AdvOfficePager1);
  aop.AdvOfficePager := AdvOfficePager1;
  aop.Caption := 'New page';
  AdvOfficePager1.ActivePage := aop;
end;
```

Following methods are available for basic manipulation of pages in the TAdvOfficePager:

TAdvOfficePager.AdvPages[Index: integer]: TAdvOfficePage;

This is an array, providing access to each page.

TAdvOfficePager.AdvPageCount: integer;

This property returns the total number of pages.

TAdvOfficePager.AdvPageVisibleCount: integer;

This property returns the number of pages with a visible tab index.

TAdvOfficePager.ActivePageIndex: integer;

This property gets or sets the active page by its index.

TAdvOfficePager.ActivePage: TAdvOfficePage;

This property gets or sets the active page as instance of TAdvOfficePage.

TAdvOfficePager.ClearAdvPages;

This method removes all pages of the TAdvOfficePager.

TAdvOfficePager.AddAdvPage(AdvPage: TAdvOfficePager): integer;

Adds an instance of a TAdvOfficePage to the pager. The return value is the new total number of pages in the AdvOfficePager.

TAdvOfficePager.AddAdvPage(ACaption: TCaption): integer;

Creates a new instance of a TAdvOfficePage with the caption set to ACaption. The return value is the new total number of pages in the AdvOfficePager.

TAdvOfficePager.RemoveAdvPage(AdvPage: TAdvOfficePager);

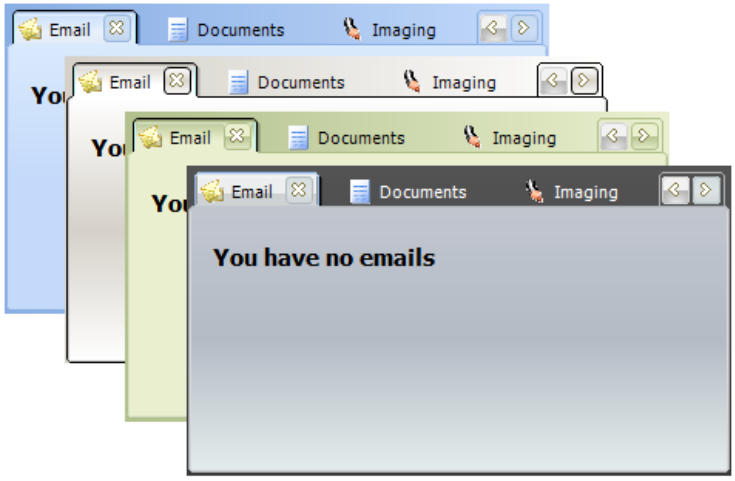
Removes the page passed as parameter from the TAdvOfficePager.

TAdvOfficePager.SelectNextPage(GoForward: Boolean);

Select the page after or before the current active page. When GoForward = true, this sets the page after the current active page as the new active page, otherwise, it sets the new active page to the page just before the current active page.

TAdvOfficePager styles

While the appearance of the TMS TAdvOfficePager can be fully customized, it is often desirable to make the application look and feel consistent with Microsoft™ Windows or Microsoft™ Office. To make it easier and faster, TAdvOfficePager has built-in presets for Office 2003, Office 2007 and Office 2010 colors and can also set the TAdvOfficePager in Metro style.



Current available Office styles in TAdvOfficePagerOfficeStyler are:

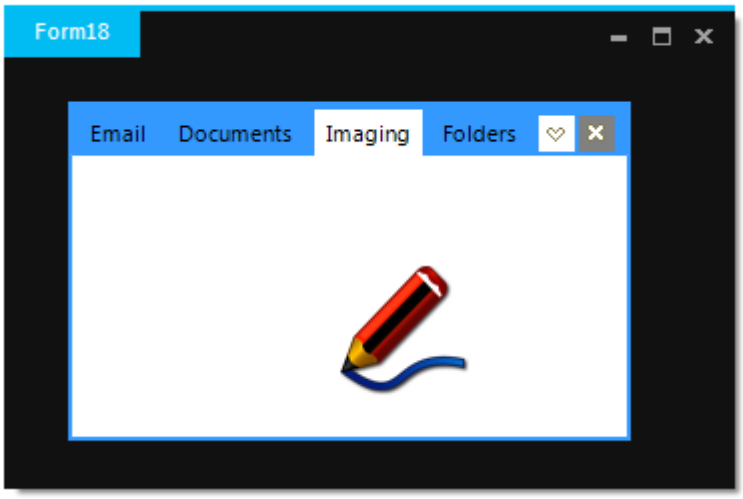
psOffice2003Blue	Office 2003 style on a blue XP theme
psOffice2003Silver	Office 2003 style on a silver XP theme
psOffice2003Olive	Office 2003 style on an olive XP theme
psOffice2003Classic	Office 2003 style on a non-themed XP or pre-XP operating system
psOffice2007Luna	Office 2007 Luna style
psOffice2007Silver	Office 2007 Silver style
psOffice2007Obsidian	Office 2007 Obsidian style
psOffice2010Blue	Office 2010 Blue style
psOffice2010Silver	Office 2010 Silver style
psOffice2010Black	Office 2010 Black style
psWindowsXP	Windows XP / Office XP style
psWindows7	Windows 7 style
psWindowsVista	Windows Vista Style
psCustom	Unforced style
psTerminal	Reduced color set for use with terminal servers
psWhidbey	Visual Studio 2005 style

To select a Metro style, following properties are available:

Metro: Boolean : when true, the Metro style is selected
MetroColor: TColor : sets the Metro accent color
MetroTextColor: TColor : sets the Metro text color
MetroStyle: TMetroStyle (msLight, msDark) : selects between a light (white background) or dark (black background) Metro style.

The Metro style can also be set in code. The TAdvOfficePager component implements the ITMSTones interface (see TMS Metro Controls guide for more background on this interface and Metro style in general). As such, the method AdvOfficePager.SetColorTones(ATones: TColorTones) can be called. To set the default Metro tones, add the unit AdvStyleIF and the code:

```
procedure TForm18.FormCreate(Sender: TObject);
begin
  AdvOfficePager1.SetColorTones(DefaultMetroTones);
end;
```



TAdvFormStyler, TAdvAppStyler

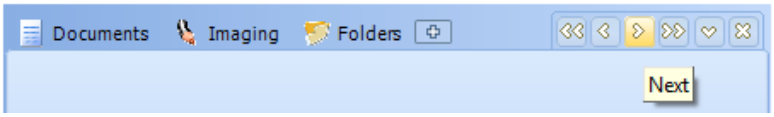
TAdvOfficePagerOfficeStyler is compatible with TAdvFormStyler, TAdvAppStyler.

A TAdvFormStyler is supposed to be dropped on a form and it will control the style of all TMS components on the form that implement the ITMSStyle interface. TAdvOfficePagerOfficeStyler implements this interface. A TAdvFormStyler will only affect components on the form itself. For application-wide appearance control, in addition to a TAdvFormStyler on a form, a TAdvAppStyler component can be dropped on a datamodule and is connected to the TAdvFormStyler components on the forms. By setting then a single property in TAdvAppStyler on the datamodule, the complete application appearance can change, both at design-time but also dynamically at run-time.

Please see the article: http://www.tmssoftware.com/site/atbdev3.asp that explains how you can use the TAdvFormStyler /TAdvAppStyler.

Built-in optional buttons on the pager

The property ButtonSettings bundles the settings for optional buttons that can be made available in the top right corner of the pager. Buttons to scroll tabs (prev/next), a button to close the active page, a button to show a dropdown with list of pages in the pagecontrol, a button to show a list of closed pages, …



ButtonSettings contains following subproperties:

ButtonSize: Integer;	Property to control the size of buttons, for example to make it bigger for touch screens
CloseButton: Boolean;	When true, a button to close the active page is shown in the top right corner
CloseButtonHint: String;	Sets the hint text for the close button
CloseButtonLook: TCloseButtonLook;	Sets the look of the close button in Chrome style or office style: cblChrome/cblOffice
CloseButtonPicture: TGDIPPicture;	Sets the image for the Close button
ClosedListButton: Boolean;	When true, a button that shows a list of previously closed pages is available in the top right corner. The list of closed pages is shown via a popup menu. The popup menu instance that will be used to show this list of closed pages is set via the property AdvOfficePager.ClosedListMenu.
ClosedListButtonHint: String;	Sets the hint text for the closed pages list button
ClosedListButtonPicture: TGDIPPicture;	Sets the image for the closed pages list button
FirstButton: Boolean;	When true, a button to scroll to the first tab is shown
InsertButtonHint: String;	Sets the hint text for the insert button
LastButton: Boolean;	When true, a button to scroll to the last tab is shown.
PageListButton: Boolean;	Property to show a list of non visible pages, when there are more page tabs than can fit. The list of available pages is shown via a popup menu. The popup menu instance that will be used to show this list of closed pages is set via the property AdvOfficePager.PageListMenu.
PageListButtonHint: String;	Sets the hint text for the PageList button
PageListButtonPicture: TGDIPPicture;	Sets the image for the PageList button
ScrollButtonFirstHint: String;	Sets the hint text for the first scroll button
ScrollButtonFirstPicture: TGDIPPicture;	Sets the image for the first scroll button

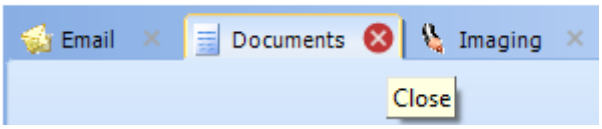
ScrollButtonLastHint: String;	Sets the hint text for the last scroll button
ScrollButtonLastPicture: TGDIPPicture;	Sets the image for the last scroll button
ScrollButtonNextHint: String;	Sets the hint text for the scroll next button
ScrollButtonNextPicture: TGDIPPicture;	Sets the image for the scroll next button
ScrollButtonPrevHint: String;	Sets the hint text for the scroll previous button
ScrollButtonPrevPicture: TGDIPPicture;	Sets the image for the scroll previous button
ScrollButtonsAlways: Boolean;	Property to show scroll buttons always irrespective of nr. of tabs
ShowInsertButton: Boolean;	When true a button to insert a new tab is shown

Besides de close button of the pager, you can also have a 'Chrome style' close button for each tab. To configure the TAdvOfficePager to have a close button on each tab, set the property AdvOfficePager.CloseOnTab = true.

Following code:

```
procedure TForm1.FormCreate(Sender: TObject);
begin
  AdvOfficePager1.ButtonSettings.CloseButton := True;
  AdvOfficePager1.ButtonSettings.CloseButtonLook := cblChrome;
  AdvOfficePager1.CloseOnTab := True;
  AdvOfficePager1.ShowCloseOnNonSelectedTabs := True;
end;
```

Results in:

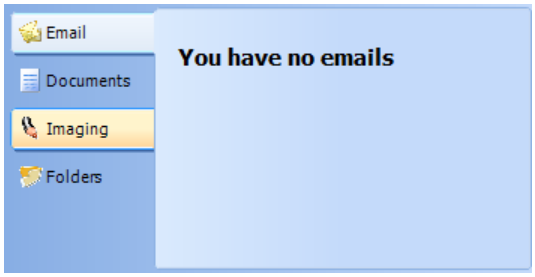


Tab appearance & behavior

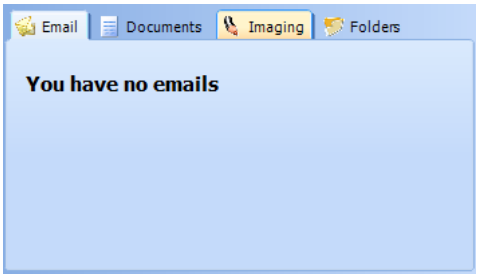
TabPosition

The property TabPosition offers the possibility to position the tabs at the top/bottom or on the left/right side of the pager. With the RotateTabLeftRight property you can configure whether the TAdvOfficePager will have 90 degrees rotated text when the tabs are shown either at the left or the right side of the pager.

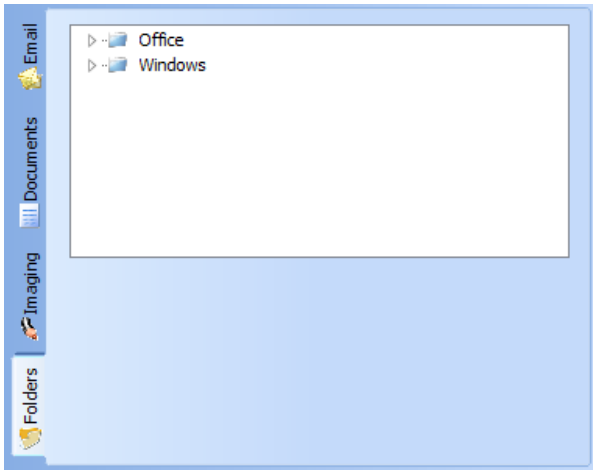
Examples:



TabPosition := tpLeft;
RotateTabLeftRight := false;



TabPosition := tpTop;

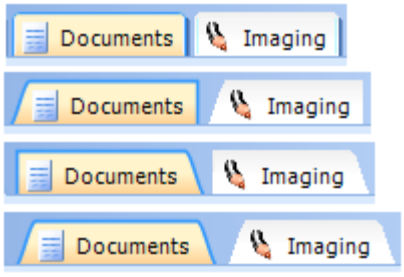


TabPosition := tpLeft;
RotateTabLeftRight := true;

TabSettings

The appearance of the tabs is further customizable via the TabSettings class property. You can choose the height, shape, image position, … of the tabs.

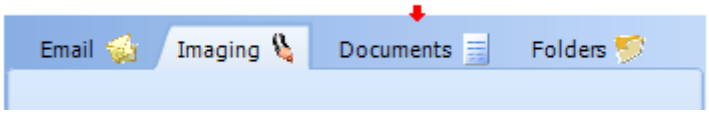
The TabSettings class property has following properties:

Alignment: TAlignment;	Property to set the alignment of tab caption: taRightJustify/ taLeftJustify/taCenter
EndMargin: Integer;	Sets the distance to keep from the right-edge of the pager for tabs (and buttons)
Glass: Boolean;	When true non selected tabs are shown transparent and will display properly on a glass frame.
Height: Integer;	Property to set the height of the tab
ImagePosition: TImagePosition;	Property to set the position of the image on the tab: ipLeft/ipRight/ipBottom/ipTop
LeftMargin: Integer;	Sets the distance to keep from the left side of the pager, i.e. where the first page tab starts.
RightMargin: Integer;	Sets the distance to keep between the rightmost page tab and the buttons in the top right corner of the pager.
Rounding: TTabRounding;	The value of the rounding of the tabs can be set between 0 and 8
Shape: TAdvTabShape;	Property to set the shape of the tab: tsLeftRamp, tsLeftRightRamp, tsRectagnle & tsRightRamp 
Spacing: Integer;	Property to set the space between tabs
StartMargin: Integer;	Property to set the startmargin of the tabs
Width: Integer;	Property to set the width of the tab. When the Width property is set to zero, the tab size is automatically adapted to fit the text in the tab.
WordWrap: Boolean;	When true, tab caption is wordwrapped

Tab reordering

When TabReorder = true, tabs can be dragged. Tab dragging starts after the mouse is moved after a tab is clicked for longer than 500ms. The drag position of the tab is indicated with an arrow. The colour of the arrow can be defined with the TabReorderIndicatorColor property:

```
procedure TForm1.FormCreate(Sender: TObject);
begin
  AdvOfficePager1.TabReorder := true;
  AdvOfficePager1.TabReorderIndicatorColor := clRed;
end;
```



Tab undocking

When AdvOfficePager.AllowTabUndock = true, tabs can be undocked from the pager, creating a floating form that holds the page that is undocked. The floating page can always at a later time be docked again. The list of floating pages can be accessed via the array AdvOfficePager.FloatingPages[index]: TAdvOfficePage. The total number of floating pages is returned by AdvOfficePager.FloatingPageCount;

Programmatically, a page can be made a floating page with the method AdvOfficePage.SetFloating(). A floating page can at a later time be docked again in the TAdvOfficePager by calling AdvOfficePager.AddAdvPage(Page);

Example:

This code snippet shows how the active page is made floating when the first button is clicked and docked again when the 2nd button is clicked:

```
procedure TForm1.Button1Click(Sender: TObject);
var
  X,Y: integer;
begin
  X := 200; // screen x-coord where the floating page will be shown
  Y := 100; // screen y-coord where the floating page will be shown

  AdvOfficePager1.ActivePage.SetFloating(X,Y);
```

```
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
  if (Advofficepager1.FloatingPageCount > 0) then
    AdvOfficePager1.AddAdvPage(Advofficepager1.FloatingPages[0]);
end;
```

By default, a TForm instance is used as the dock site that hosts a TAdvOfficePager with the undocked page. The border icons and border style for the floating dock site is set with TAdvOfficePager.FloatingBorderIcons and TAdvOfficePager.FLoatingBorderStyle. It is possible to specify a custom form as dock site via TAdvOfficePager.FloatingPagerWindowClass. This should be a class that descends from TCustomForm and implements the IFloatingPagerWindow interface. This interface has just two methods to set & get the TAdvOfficePager instance on the dock site.

This example code snippet shows how a TAdvMetroForm instance can be used as dock site for floating pages:

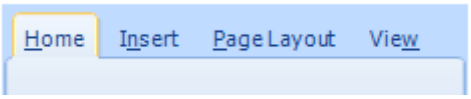
```
// class descending from TAdvMetroForm that implements
// IFloatingPagerWindow

uses
  AdvMetroForm;

type

  TAdvMetroPagerWindow = class(TAdvMetroForm, IFloatingPagerWindow)
  var
    FAdvPager: TAdvOfficePager;
  public
    function AdvPager: TAdvOfficePager;
    procedure SetAdvPager(APager: TAdvOfficePager);
  end;

implementation

// specify this class as dock site

procedure TForm1.FormCreate(Sender: TObject);
begin
  AdvOfficePager1.FloatingPagerWindowClass := TAdvMetroPagerWindow;
  AdvOfficePager1.FloatingBorderStyle := bsNone;
end;
```

Hotkeys

Hotkeys can be defined in the TAdvPage caption to navigate through the tabs.

Example:

Press Alt-H to navigate to Home, press Alt-W to navigate to View.




```
procedure TForm1.FormCreate(Sender: TObject);
begin
  AdvOfficePager1.AdvPages[0].Caption := '&Home';
  AdvOfficePager1.AdvPages[1].Caption := 'I&nsert';
  AdvOfficePager1.AdvPages[2].Caption := '&Page Layout';
  AdvOfficePager1.AdvPages[3].Caption := 'Vie&w';
end;
```

Events

Following additional events are available in the TAdvOfficePager:

OnClosedListClick: Triggered when the button “ClosedListButton” is clicked.

OnClosedPage: Triggered when the Close button for the active page is clicked after the page is effectively closed.

OnClosePage: Triggered when the Close button for the active page is clicked before the page is effectively closed. Setting the Allow parameter of the event to false will prevent the actual closing o the page.

OnFirstClick: Triggered when the “FirstButton” in the top right corner of the pager is clicked.

OnInsertPage: Triggered when the page insert button in the top right corner of the pager is clicked.

OnLastClick: Triggered when the “LastButton” in the top right corner of the pager is clicked.

OnNextClick: Triggered when the scroll button to navigate to the next page in the top right corner of the pager is clicked.

OnPageListClick: Triggered when the button “PageListButton” is clicked.

OnPrevClick: Triggered when the scroll button to navigate to the previous page in the top right corner of the pager is clicked.

OnTabCheckBoxClick: Triggered when the checkbox that can be displayed on a page tab is clicked.

OnTabClick: Triggered when a tab is clicked.

OnTabDblClick: Triggered when a tab is double-clicked.

OnTabDock: Triggered when a tab is docked back to the pager.

OnTabMoved: Triggered when a tab is moved to a different position by dragging (when property TabRearrange is set true).

OnTabRightClick: Triggered when a tab is right-clicked.

OnTabUnDock: Triggered when a tab is undocked from the pager.

OnPageListClick: Triggered when the button “PageListButton” is clicked.

Page properties

Per page, a number of properties is available to control individual settings of a page:

Caption: string : sets the text to show in the tab for a page

Checked: Boolean : when Page.ShowCheckBox = true, a checkbox is displayed in the tab and the property Checked will get & set the checked state of this checkbox.

DisabledPicture: TPicture : can set a picture that is shown on a tab when the page is disabled

Glow: Boolean : When true, the border around the page will glow between the regular border color and the color set with GlowColor to draw the attention on the tab.

GlowColor: TColor : sets the color between which the tab border color alternates to draw the attention.

ImageIndex: integer : sets the index of the imagelist image to show on the tab for the page when an imagelist is assigned to the TAdvOfficePager

Locked : Boolean : when true, the tab will not scroll along with other tabs when there are more tabs than can be visually shown on the TAdvOfficePager. Most typically this is used when the TAdvOfficePager contains a number of tabs from the left that must be always visible while other tabs right of these locked tabs can scroll.

OfficeHint: TAdvHintInfo : contains the settings for an Office hint shown when the mouse hovers over a tab. An Office hint contains a title text in bold, notes text, optionally a picture and a help icon. It is required that the TAdvOfficeHint component is put on the form to enable rendering these more complex Office hints.

PageAppearance: TVistaBackground : holds all settings for the background of the page. By default, the page background is equal for all pages and is controlled by the associated styler.PageAppearance. Via the PageAppearance property on TAdvOfficePage, it is possible to set a different page background per page. The settings in TAdvOfficePage.PageAppearance are used instead of the global styler properties when UsePageAppearance = true.

Picture: TPicture : can set a picture instead of an imagelist image for a page

Progress: TTabProgress : Holds the settings for an optional progressbar that can be shown as background of a tab. The progressbar is by default not shown but will show when AdvOfficePage.Progress.Visible is set to true.

ShowCheckBox: Boolean : When true, a checkbox is shown left from the tab caption. The state of the checkbox can be get or set with AdvOfficePage.Checked.

ShowClose: Boolean : When true, a close button is shown on the tab itself to allow to immediately close a page by clicking on the close button on the tab.

TabAppearance: TTabAppearance : : holds all settings for the background of the page tab. By default, the page tab background is equal for all pages and is controlled by the associated styler.TabAppearance. Via the TabAppearance property on TAdvOfficePage, it is possible to set a different page tab background per page. The settings in TAdvOfficePage.TabAppearance are used instead of the global styler properties when UseTabAppearance = true.

TabEnabled: Boolean : When true, the tab is enabled, allowing to select a page by clicking on the tab. When false, the tab will not react to a click and will show the tab in disabled style.

TabVisible: Boolean : When true, the page tab is now shown. The page is still available but the user cannot access it anymore or see its availability from the tab.

UsePageAppearance: Boolean : When True, the appearance of a page is controlled by TAdvOfficePage.PageAppearance instead of the global styler.PageAppearance setting.

UseTabAppearance: Boolean : When True, the appearance of a page tab is controlled by TAdvOfficePage.TabAppearance instead of the global styler.TabAppearance setting.

Persisting tab positions

As the user can reorder tabs, close tabs, undock tabs,… it is often desirable to persist the latest state of all tabs when the user quits the application and restore this state when the application restarts. The TAdvOfficePager makes it easy to manage this. It is important to note that the TAdvOfficePager manages this in relationship to the reference order of tabs as created at design time. At all times,the TAdvOfficePager can be restored to this default design time setting by calling TAdvOfficePager.ResetPages. When the pages are created programmatically, the reference tab ordering can be set at all times by calling AdvOfficePager.InitOrder;

To save or restore settings of a TAdvOfficePager tabs, a single & simple string property TAdvOfficePager.Settings: string is offered. This returns at all times the state of tabs encoded as string or can set the state of tabs. This makes it easy to persist the settings in an INI file value or a registry key. As an alternative, this can also immediately be saved or restored from file with the methods AdvOfficePager.SaveToFile(FileName:string) and AdvOfficePager.LoadFromFile(FileName: string);

Other page management methods

procedure OpenAllClosedPages : immediately reopens all previously closed pages of the TAdvOfficePager. (Unless AdvOfficePager.FreeOnClose was set to true, as in this case, a page is effectively destroyed when it is closed)

procedure OpenClosedPage(APage: TAdvOfficePage) ;

Reopens a specific previously closed page.

procedure MoveAdvPage(CurIndex, NewIndex: integer);

Moves a page with index CurIndex to a new position NewIndex in the TAdvOfficePager.

function FindNextPage(CurPage: TAdvOfficePage; GoForward, CheckTabVisible: Boolean): TAdvOfficePage;

Return the page immediately before or after the page CurPage with either a visible or not visible tab in the TAdvOfficePager.

function IndexOfPage(AdvPage: TAdvOfficePage): Integer;

Returns the index of a given page in the pages array AdvOfficePager.AdvPages[index]

Function IndexOfTabAt(X,Y: integer): integer;

Returns the index of a given page in the pages array AdvOfficePager.AdvPages[index] that has a tab at cursor coordinates X,Y

Tips and FAQ

Programmatically inserting a page

```
pc: TAdvOfficePage:
pc := TAdvOfficePage.Create(self);
pc.AdvOfficePager := AdvOfficePager1;
pc.Caption := 'New page';
```

Undock and dock a page

When setting AdvOfficePager.AllowTabUndock = true, pages can be undocked from the TAdvOfficePager. To dock the pages programmatically back, following code can be used:

```
begin
  if not AdvOfficePage.Locked then
    AdvOfficePager.AddAdvPage(AdvOfficePager);
end;
```