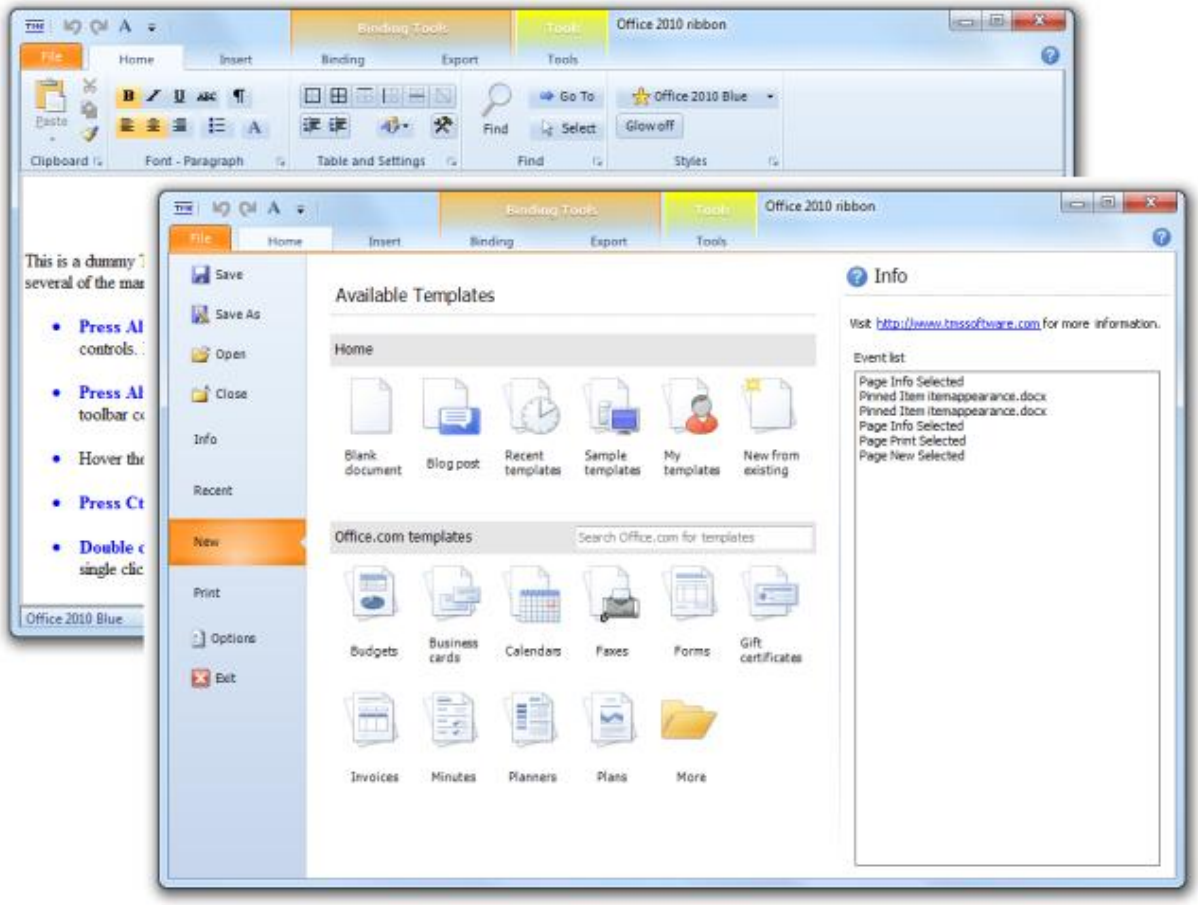
## Creating Office 2010 style VCL applications with TMS components

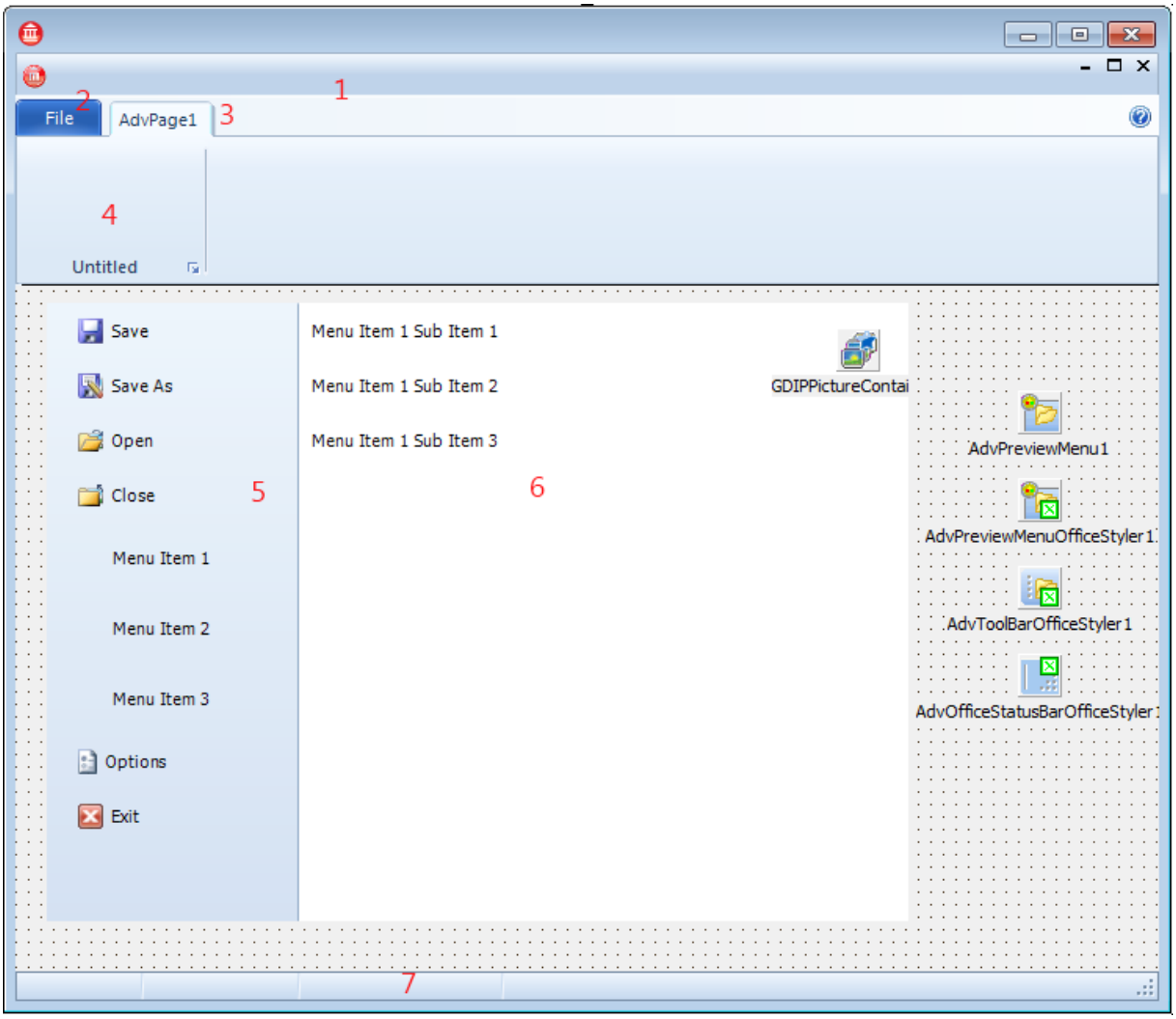
There are several reasons why a software developer would want to give his application the Microsoft Office look and feel. Most users are familiar with how Office works and as such, there will be less of a barrier when users start to use your application. Microsoft is also able to spend a lot of money on usability testing to see what works and what doesn't work in user interfaces. Inheriting the result of these studies is almost certain to also benefit the usability of your application. Finally, the perception that your application looks up-to-date and follows the latest trends in user interfaces will certainly also leave a good impression with potential new customers. This article not only briefly discusses what the TMS ribbon control offers but also how many more TMS components can be used to have a consistent Office 2010 style user interface. If you have not purchased the TMS Component Pack or TMS Advanced ToolBars & Menus, you can download the trial version of the TMS ribbon via: http://www.tmssoftware.com/site/advtoolbar.asp and explore what is described in the article.




### The ribbon

The first eye catcher when starting one of the applications in the Microsoft Office suite is the context sensitive ribbon control. The ribbon control comes in two flavors: the Office 2007 ribbon with the round application button and popup application menu and the Office 2010 more flat ribbon with rectangular application menu button and its full screen application menu. The TMS Advanced ToolBars & Menus offers both the Office 2010 and

Office 2007 look & feel. It even includes the Office 2003 style docking toolbars for applications that benefit from a more simple interface. This article will focus on Office 2010 though.
Starting a TMS ribbon application is as simple as choosing in the IDE: File, New, TMS Wizard, Office 2010 application. It creates the basic skeleton ribbon application with application menu button and application menu. This results in the IDE in:




1: TAdvOfficePager: main ribbon container
2: TAdvShapeButton: application menu button
3: TAdvOfficePage: one ribbon page / tab
4: TAdvToolBar: one toolbar on a ribbon page
5: TAdvPolyMenu: main left menu on application menu frame
6: TAdvPolyList: wedge item coupled poly list based sub menu
7: TAdvOfficeStatusBar: status bar in Office 2010 style

A TAdvOfficePager, which is the main ribbon container, is created with one page, the page has one toolbar. A frame is also created and inserted on the main form. This frame holds the full screen application menu. The sophisticated Office 2010 application menu uses the TMS PolyList component that provides you with an unlimited flexibility to organize menu items, menu settings etc... Finally, at the bottom of the form an Office 2010 status bar is added.

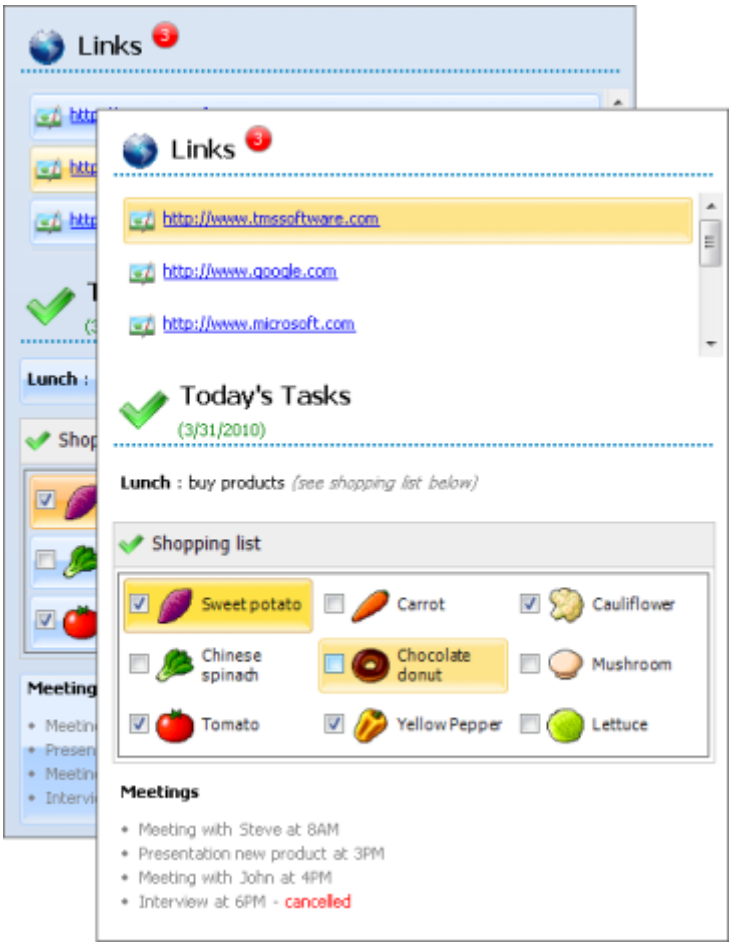
To edit the ribbon, click any element of the ribbon to set its properties in the Object Inspector. Right-click at design time and select to add a new page or a new toolbar from the context menu. Right-click the toolbar itself and select to insert any of the known toolbar controls such as toolbar button, toolbar dropdown button, and many Office style selectors such as color, font, table, … selectors. Other than this, you can drag & drop any component from the IDE tool palette on the toolbar. The application menu button (the File button on the screenshot TAdvShapeButton) is also automatically generated by the wizard. It is from a click on the application menu button that either a more simple Office application menu is shown (TAdvPreviewMenu) or a full screen application menu via the generated frame with TMS Polylist controls. When TAdvShapeButton.Frame is set to the frame on the form, a click on the application menu button will automatically show the frame full screen. When the frame property is not set but just the TAdvShapeButton.AdvPreviewMenu property, the menu as configured via the component TAdvPreviewMenu will be shown when the button is clicked.

## Application menus

The focus of the article is on the new Office 2010 application menu that now appears full screen when the application menu button is clicked. On Delphi application level, this is handled via a frame that hosts the application menu. This application menu is composed with TMS PolyList controls. There is only a left-aligned TAdvPolyMenu control and a client-aligned TAdvPolyList. The design time editor of both poly list controls is invoked by double-clicking it in the IDE. The TAdvPolyMenu is filled with TImageTextItem instances for the menu items that execute directly and with TWedgeItem instances for menu items that show more sub menu options in the TAdvPolyList. In the template application, there are 3 wedge items and from these items 3 different sub menu options appear that are organized in TAdvPolyList controls. From the TAdvPolyMenu.OnItemSelect event, a switch of the polylist is done:

```
procedure TTMSFrame4.AdvVerticalPolyList2ItemSelect(Sender: TObject;  Item:
TCustomItem; var Allow: Boolean);
begin
  case Item.Tag of
  1: AdvPolyList1.BringToFront;
  2: AdvPolyList2.BringToFront;
  3: AdvPolyList3.BringToFront;
  end;
end;
```

The client aligned TAdvPolyList can in turn also be configured by double-click of the component on the form or by opening the List property from the Object Inspector. The TAdvPolyList can host a wide variety of item types, going from items with a large option button, items with a dropdown list, items with an image, items with HTML formatted text, custom items, sections, control containers and much more…

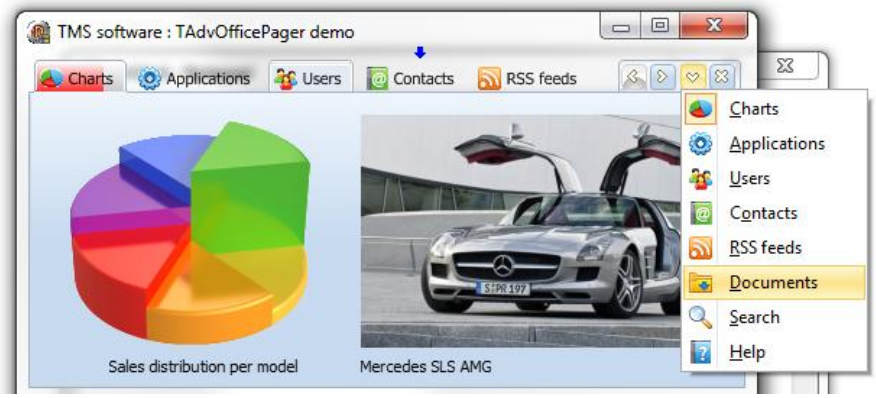



In fact, the TMS PolyList is so versatile, we'd like to invite you to read the developers guide http://www.tmssoftware.com/site/manuals/TMS%20Advanced%20Poly%20List.pdf as well as several online articles that exist for it:
http://www.tmssoftware.com/site/blog.asp?post=156
http://www.tmssoftware.com/site/blog.asp?post=173
http://www.tmssoftware.com/site/blog.asp?post=177

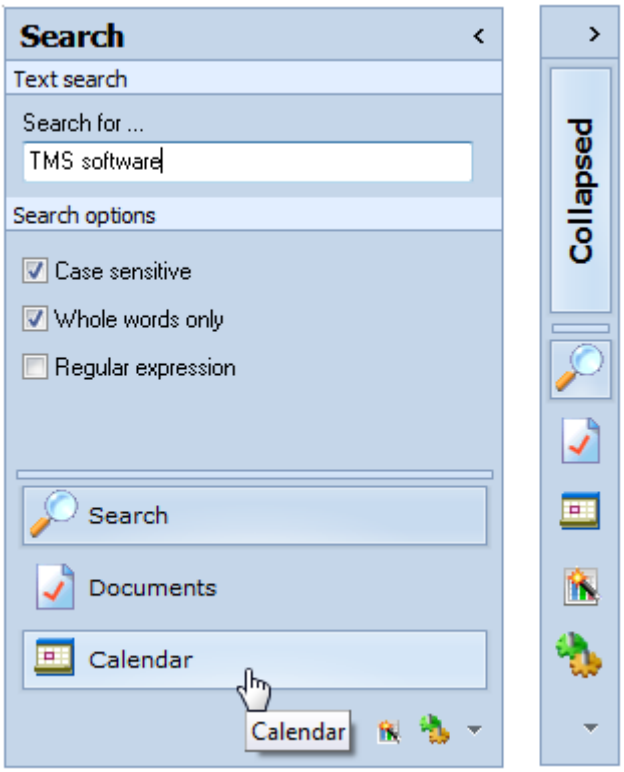
It will enable you to create any type of application menu you want.

## Outlook 2010 navigation bar and other controls

To create applications with an Office 2010 look & feel, there is of course more needed than just a ribbon. TMS Component Pack (http://www.tmssoftware.com/site/tmspack.asp) contains several other controls that are easily configured to have an appearance like in an Office 2010 application. There are several controls like panel, panelgroup, pagecontrol, tabset, calendar, datepicker, grid that are all designed to have the possibility to appear with colors consistent with Office 2010.




One of the more complex components is the Outlook 2010 style navigation bar. This control consists of several panels that can host any control and can be selected and collapsed. The TMS TAdvNavBar (http://www.tmssoftware.com/site/advnavbar.asp) that offers this functionality, will also mimic the Outlook 2010 appearance:



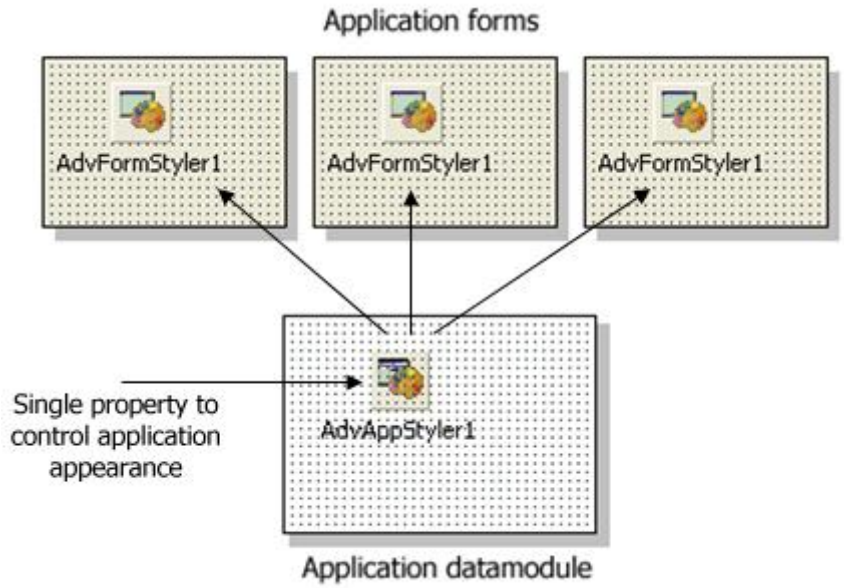

## Consistent style

Office 2010 offers the capability to select 3 color themes: Blue, Silver and Black. You might want to offer this same capability in your application or you might want to offer even more and allow your user to choose Office 2003 color style, a Windows 7 explorer color style, an Office 2007 Luna appearance or perhaps a very basic color style for use via a terminal server. When creating an application with an Office 2010 ribbon, you will without doubt use many more components. It is as such very important that it will be very easy and fast to change the overall look and feel of your application without the need to go diving into numerous color property settings for all components in your application. We strongly believe that not only for optimum performance, each control

should know how to paint itself in different styles but also for the sake of completeness and flexibility. Complex controls can have many different elements. A ribbon control has completely different elements from an agenda control. Ultimately, it is only the control itself that knows about the elements it has and how these elements should appear in a particular color style. Therefore, many TMS components have a configurable color style that can be set via consistent interface. We have spent quite some effort to simplify form-wide and application-wide appearance control for TMS components and possibly also your custom controls. To do this, two new components have been created:

TAdvFormStyler
TAdvAppStyler

A TAdvFormStyler can be dropped on a form. In a way, it will control just like TXPManifest the style of the TMS controls on the form. A TAdvFormStyler will only affect controls 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 runtime.




The component TAdvFormStyler has a property style. Setting this style property causes all components on the form that support the interface to set the style to change to the selected style. Similary, setting the style property for the TAdvAppStyler on a central datamodule invokes all TAdvFormStyler style changes and thus the style of all TMS controls on the form. The TAdvFormStyler will scan the entire form for all controls that implement the ITMSStyle interface and when found, call the interface SetComponentStyle method to update the style to the selected style. Many TMS controls already implement the ITMSStyle interface. As shown later, you can also extend your custom controls to be TAdvFormStyler aware.

Currently the TAdvFormStyler, TAdvAppStyler support following styles:

tsOffice2003Blue : Office 2003 style on a blue XP style
tsOffice2003Silver : Office 2003 style on a silver XP style
tsOffice2003Olive : Office 2003 style on an olive XP style
tsOffice2003Classic : Office 2003 style on a non-themed XP or pre XP operating system
tsOffice2007Luna : Office 2007 Luna style
tsOffice2007Obsidian : Office 2007 Obsidian style
tsOffice2007Silver: Office 2007 Silver style
tsWindowsXP : Windows XP / Office XP style
tsWhidbey : Visual Studio 2005 style
tsCustom : unforced custom style
tsWindowsVista: Windows Vista style
tsWindows7 : Windows 7 style
tsTerminal : reduced color set for use with terminal server
tsOffice2010Blue : Office 2010 Blue style
tsOffice2010Silver : Office 2010 Silver style
tsOffice2010Black : Office 2010 Black style

You can make your own controls also easily TAdvFormStyler, TAdvAppStyler aware so that your controls also automatically change their appearance when the application and/or form style changes. To do this, it is sufficient to add and implement the ITMSStyle interface to your control. This code snippet shows a sample custom control that was made TMS style aware:

```
interface
uses
  Classes, TAdvStyleIF;

type
  TMyCustomControl = class(TCustomControl, ITMSStyle)
  public
    procedure SetComponentStyle(AStyle: TTMSStyle);
  end;

{ TMyCustomControl }

procedure TMyCustomControl.SetComponentStyle(AStyle: TTMSStyle);
begin
  case AStyle of
  tsOffice2003Blue: // set properties correct here for the selected style
  tsOffice2003Silver:
  tsOffice2003Olive:
  tsOffice2007Luna:
  tsOffice2007Obsidian:
  ….
  end;
end;
```

## Conclusion

TMS software offers with TMS Advanced ToolBars & Menus components to create not only Office 2010 style ribbons but also Office 2007 ribbons or Office 2003 toolbars. Designing

applications with an Office 2010 look and feel is much more though than just the ribbon and application menu. Many more TMS UI Components of the larger collection of components that can be found in the TMS Component Pack are designed with built-in color styles for various Office and Windows color styles and via TAdvFormStyler, TAdvAppStyler, a mechanism is inplace to allow software developers to quickly select or change a style without touching a single color property both at design-time and run-time.