What's New in WebDesktop.NET version 2.5

In addition to new product added to WebDesktop.NET version 2.5, the existing components have also been significantly enhanced. The following lists the new features of each component.

WebDesktopManager

Shadow mode support for window moving and resizing

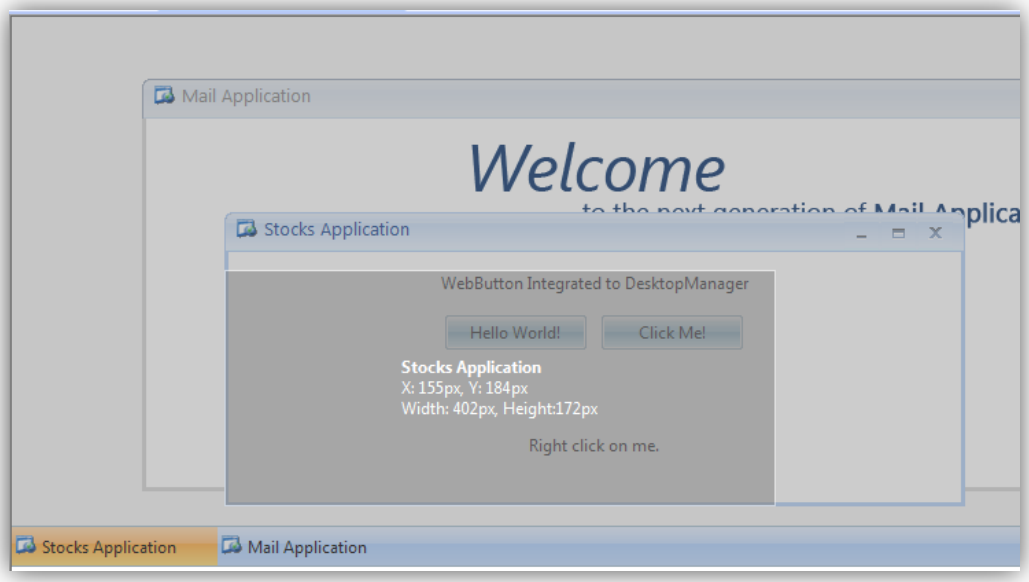
This new feature enables window moving and resizing operation to be performed in shadowing mode. That means the actual window will not be taken affect until the operation is done. During the interactive mode, only the shadow of the actual window will be displayed to indicate the current status of the interaction.

Unlike Physical interactivity mode which is the default value and the mode used in previous version, Shadow interactivity mode is significantly faster. It enables the window interaction such as moving and resizing to be performed in a very smooth and robust manner.

This new interactivity mode introduces several advantages:

- Significantly faster performance while moving and resizing window. This is made possible because the heavy contents are not re-rendered during the interaction.
- 100% smooth and robust interaction even with a lot of IFrame-kind of window content. This means, the window can still be smoothly moved and resized even though it is on the top of other IFrame windows. This was one limitation in previous "physical interaction" mode.
- Avoided "resize bogus" and improved resizing stability in Mozilla. Most components today extensively using *onresize* event to handle the layout and rendering when a container is being resized. The "physical interaction" is surely a performance killer because the *onresize* events that attached to the resized window will be called massively and continuously. This problem is nothing to worry with the new interactivity mode.

The following is a sample screenshot when the “Stocks Application” window is resized downward.



To change the interactivity mode to Shadow, simply set the *WindowInteractiveMode* to *Shadow* in the WebDesktopManager instance.

There are two new properties related to this feature:

1. **ShadowModeAnimation**. The default value is false. This property allows you to add nice fading effect when the moving and resizing is performed in Shadow mode. However, the animation effect is not suitable for complex Web application as it could slow down the screen rendering while performing interaction.
2. **ShadowModeTransparency**. By default, the desktop manager screen would be 100% transparent when the moving and resizing is performed in Shadow mode. When this property is set to true, the background of desktop manager will be dimmed to 50% transparent to give user better focus on the interactive operation. The above screenshot shows the ShadowModeTransparency enabled.

WebDialogBox

Direct Control Access in Template

This new enhancement enables developers to access the controls in the DialogBox's *ContentTemplate* directly in the code behind without have to use *FindControl* method. This enhancements increases code efficiency and reduce complexity. It also avoid the need to perform casting.

If you have developed Web form using the previous version of *WebDialogBox*, your Web form should be able to run properly without the need to perform upgrades. However, it is highly recommended that you take advantage of the benefits introduced with this new enhancement.

Previously, you will need the following codes to access a server control in ContentTemplate of a WebDialogBox:

```
DropDownList dropdown = (DropDownList)dlgBox.FindControl("DropDownList1");
dropdown.Items.Add(new ListItem("Item 1", "Item 1"));
```

Now, you only need a single line of code:

```
this.DropDownList1.Items.Add(new ListItem("Item 1", "Item 1"));
```

Keep Visible On Scroll

While WebDialogBox is best used in a Web page with no scrollbars, there are some scenarios where you will need to show dialog box in scrollable page. For instance, in a product information page or feature page.

With this new feature, you can keep WebDialogBox to stay visible when user is scrolling throughout the page. To enable this feature, simply set *KeepVisibleOnScroll* property to True. This property is set to False by default to preserve compatibility with existing applications.

Background Shadow Color and Opacity

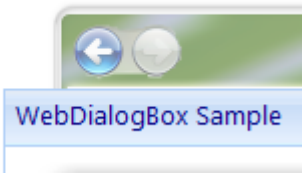
By default, when a WebDialogBox is shown, the background of the parent window will be blocked in the way that it does not accept mouse clicks, which simulates desktop application. The background is set to Transparent by default, and is not customizable in the previous version.

In this new version, you can now customize the background color of the shadow, as well as the transparency opacity. These new additions allow you to dim out the background screen to let end users easily focus on the dialog box.

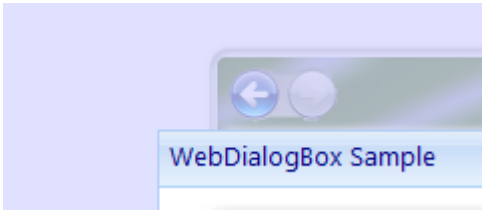
To set the background shadow color, set the *ShadowColor* property to a valid color. To change the transparency opacity, set the ShadowOpacity to a valid transparency level. The minimum is 0 which means opaque, and the maximum is 100 which means fully transparent. The default value for the opacity is 50.

The following illustration shows the comparison between fully transparent background and dimmed background.

Without Shadow Color (Fully Transparent)



With Shadow Color set to Gray, and Opacity set to 50%.



Show Dialog with Animation

The WebDialogBox now supports *ExpandShrink* type animation when showing a dialogbox from client codes. The animation is always a nice addition to give smooth and professional impression to your Web application.

The enhancement is made to the *ShowDialog* method of the WebDialogBox class. The method now introduces two new parameters: enableAnimation and fromElement.

Syntax:

[webDialogBoxInstance].ShowDialog(enableAnimation, fromElement);

enableAnimation is a boolean value indicating whether animation should be performed.

fromElement is a HTML element where the animation should start from.

Sample:

```
divEl = document.getElementById(“div1”);

webDialogBox1.ShowDialog(true, divEl);
```

Note: The *EnableAnimation* property of the WebDialogBox need to be set to True from the server side.

WebMenu

Complex Images for Menu Item Style

With this new feature, you can now build visually compelling design for your Web application's menu navigation. The complex images allow you to specify the images needed by the WebMenu to render the menu item. This enables you to build a pixel-identical menu design that similar to today's modern designs such as Vista Explorer or Office 2007.

The complex images concept consists of three parts:

- Left
- Center
- Right

All parts' image should have exactly the same dimension especially the height to deliver consistent and reliable design.

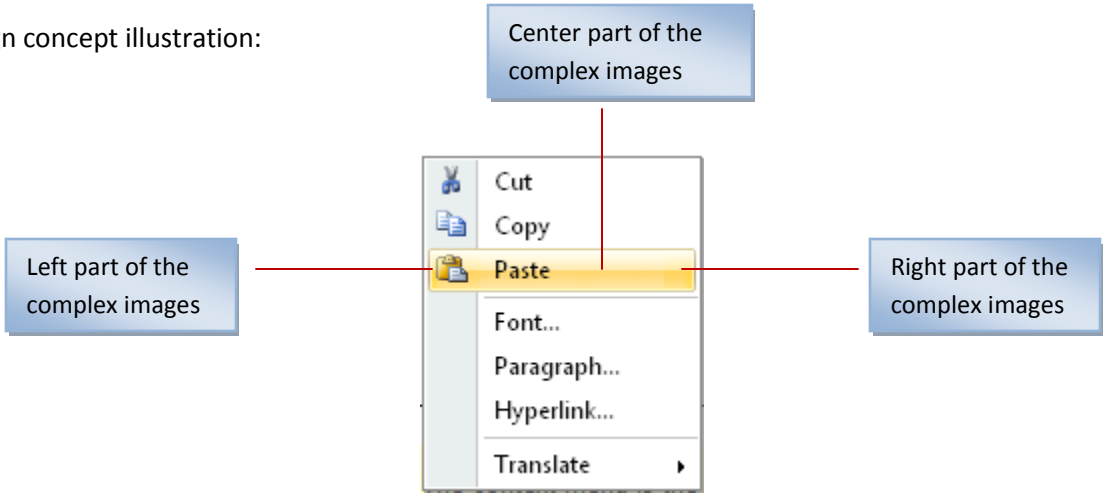
To enable complex images highlight, simply set *HighlightMode* property (inside MenuStyleSettings) to *UseComplexImages*.

The complex images can be specified in the *ImagesSettings* of the *MenuStyleSettings* object. *MenuStyleSettings* is available in all UI components that implement WebMenu interface, such as WebContextMenu, WebToolBar, WebMenuBar and WebNotification.

Sample ASPX declaration:

```
<ISWebDesktop:WebContextMenu ID="WebContextMenu1" runat="server" ControlId="Page">
        <MenuStyleSettings BackgroundStripColor2=""
              BackgroundStripImage="o7_menu_strip.gif"
              MenuAnimation="True" MenuDropShadow="True" MenuWindowType="Normal"
              BackgroundStripWidth="26"
              IconCellWidth="25" HighlightMode="UseComplexImages">
            ...<Other Styles Here>
            <ImagesSettings OverCenter="o7_menu_oc.gif" OverLeft="o7_menu_ol.gif"
              OverRight="o7_menu_or.gif"
              ActiveIconCell="o7_icon_active.gif" />
        </MenuStyleSettings>
        <Items>
            <ISWebDesktop:WebMenuItem Name="mnuCut" Text="Cut"
              ImageURL="images/cut.gif">
            </ISWebDesktop:WebMenuItem>
            ...
        </Items>
</ISWebDesktop:WebContextMenu>
```

Design concept illustration:



The layout, styling and rendering process is automatic based on the specified ImagesSettings. All you need is to provide the images needed by the complex images, and WebMenu will automatically arrange the layouting, margins, padding, repeating, edging, and other behaviors.

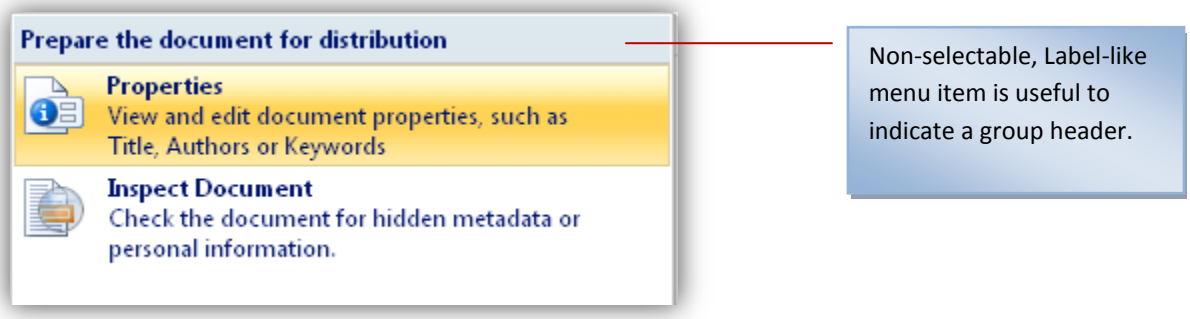
Walkthrough: Enable complex images in WebContextMenu.

[TBD]

Cosmetic Label Menu Item

This new feature provides ability for developers to insert cosmetic, non selectable menu item. This label-like menu item is very useful and suitable to be used as a group header, or to provide a simple explanation about the menu items beneath it.

In Office 2007, oftentimes we find this kind of cosmetic menu item, such as shown in the following image.

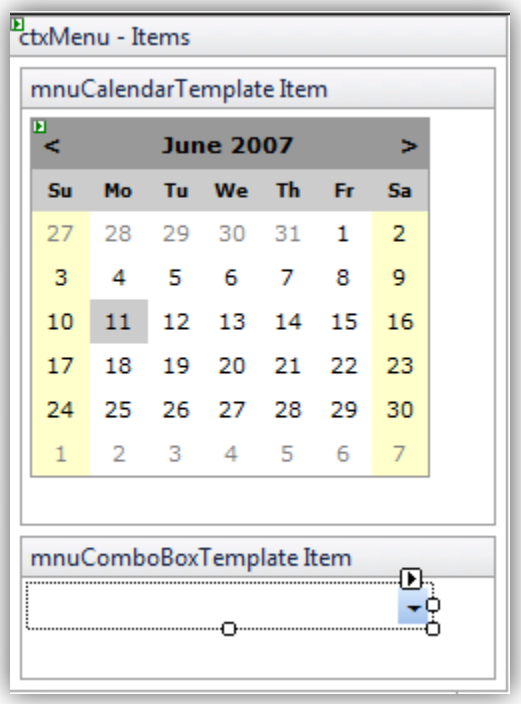


To use label menu item type, simply set the *Type* of the menu item to *Custom*. You can also customize the style and appearance of the custom menu item by accessing the CustomStyle in MenuSettings object of the control instance.

Template Support for Menu Item

The ability to use template inside Menu Item has been added in this new release of WebMenu. With templating, you can put any html or server-side controls to the menu item's cell content. This allows you to create efficient and richer navigation concept for your Web application.

To use Template, set the *Type* of the menu item to *Template*. Next, in the Visual Studio 2005 designer, right click on the control instance and choose Edit Templates > Items. Finally you will see the region's surface where you can perform editing. See the following screenshot for more details.

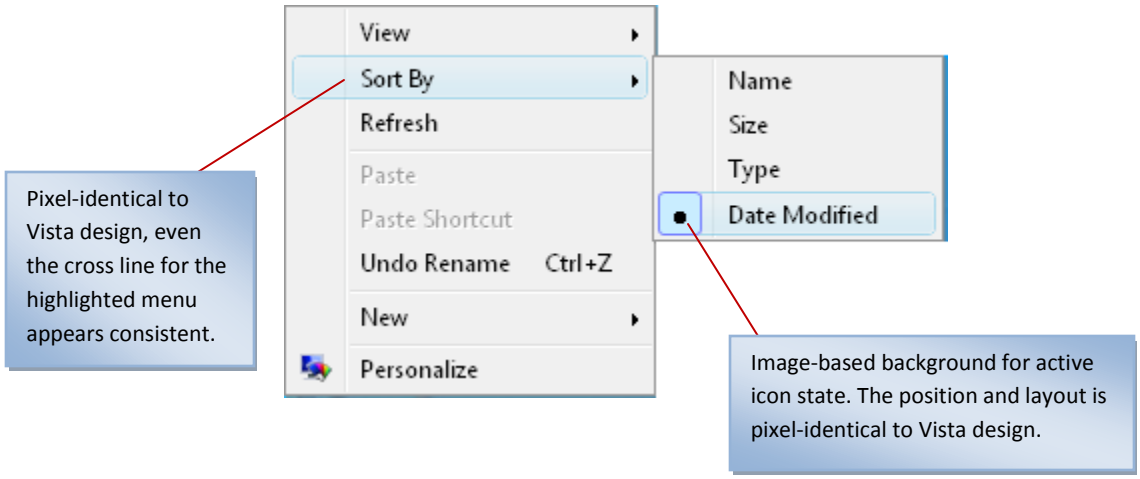


New Themes

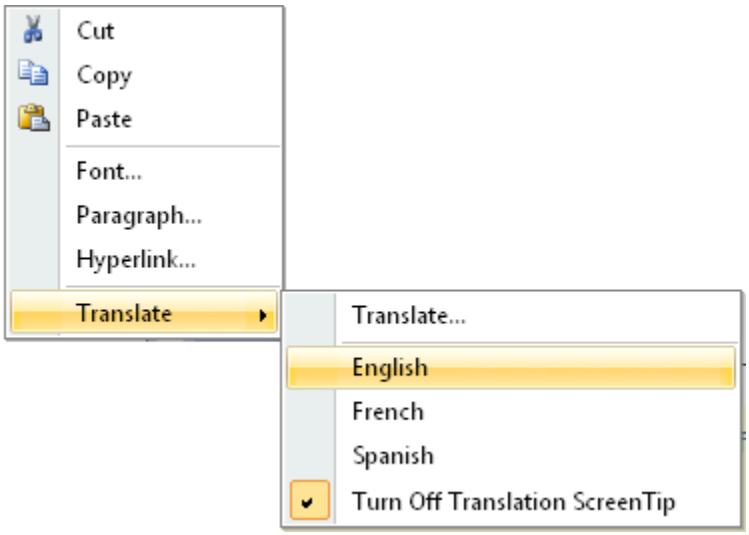
With the new features such as complex images, template support and new client side APIs added in this new version, you can now deliver fluid and pixel-identical styles and designs. Fluid means that the appearance should be consistent and looking good regardless of the size of the text or width of the control.

Two new designs are now possible to be achieved:

- Pixel-Perfect Fluid Vista Style



- Pixel-Perfect Fluid Office 2007 Style



WebButton

- Enhanced mouse state interactivity
- Complex Images
- DropDown Menu Support