

WHAT'S NEW IN PETER'S DATA ENTRY SUITE v5



Click on any of these topics to jump to them:

- ◆ [Breaking changes when upgrading from ANY version](#) **IMPORTANT**
- ◆ [Upgrading from v4.0](#)
- ◆ [Upgrading from pre 4.0 products: Changes to Installation](#)
- ◆ [Professional Validation and More Users](#)
 - [Breaking Changes](#)
 - [General Features](#)
 - [Peter's Professional Validation and Peter's More Validators modules](#) (was VAM: Essential Validators)
 - [Peter's TextBox module](#) (was VAM: Data Entry Controls)
 - [Peter's Interactive Pages module](#) (was VAM: Client-side Toolkit)
 - [Peter's Input Security module](#) (was VAM: Visual Input Security™)
 - [New module: Peter's Date and Time](#)
 - [Design Mode](#)
- ◆ [Changes made converting Peter's Date Package v2 to Peter's Date and Time](#)
 - [Breaking Changes](#)
 - [General Changes](#)
 - [DateTextBox, AnniversaryTextBox, and MonthYearTextBox](#)
 - [TimeOfDayTextBox and DurationTextBox](#)
 - [Calendar and MultiSelectionCalendar](#)
 - [PopupCalendar](#)
 - [MonthYearPicker and PopupMonthYearPicker](#)
 - [TimePicker and PopupTimePicker](#)
 - [SpecialDates](#)
 - [QuickDateMenu](#)
 - [ContextMenu](#)
 - [Validators](#)
- ◆ [Table of Contents](#)

Breaking changes when upgrading from ANY version

Due to the impact of the major breaking changes shown below, users have the option of using Version 4 instead of Version 5. Version 4 will continue to be supported, but code changes will be limited to bug fixes. Version 5 should be used if you need BLD or feature updates.

Your Version 5 serial number and license files work with Version 4. You can upgrade from Version 4 to Version 5 when you are ready. If you want to retrieve Version 4, click [here](#) and login with your version 5 serial number.

Major breaking changes

- ASP.NET 1.0 and 1.1 platforms are no longer supported. DES Dynamic Data users no longer have support for ASP.NET 3.5. They must use the replacement, Peter's Business Driven Logic UI, with ASP.NET 4 or higher.
- The code base has been refactored. The basic idea is that in the past, a web control object held most of the code, and now it uses a multiple tiered approach. There are usually 3 classes for each control: the web control top layer, the HTML output layer, and code that is UI neutral.

Most users **will not be impacted** by this. The upgrade process renames changed classes and obsolete properties still work, only they are marked with the `ObsoleteAttribute` to advise you to change your code. Expect a lot of these.

Here is where changes will impact you: If you have subclassed any of the DES, VAM, or Peter's Date Package classes, they are likely to break except those that override **ONLY** the constructor. Most significant is to any `Condition`, `DESTypeConverter`, and `ErrorFormatter` class that you have developed, however your web control subclasses may also break. **Expect compiler errors (not warnings) in this case.**

Please let Peter help you convert the classes by emailing the code to these classes, including any javascript support files. Contact Peter at support@peterblum.com. Peter may do the actual work. If so, he may charge for this work, depending on its extent. Costs will be discussed prior to doing the work.

- Several Date and Time controls have been given a fresh appearance. This impacts the style sheet files and images used for buttons. The upgrade process establishes the pre-DES 5 appearance by using modified versions of your style sheet files that were in the `\PetersDatePackage\Appearance` folder or `\DES\Appearance\Date and Time` folder. As a result, you should not see much difference. However, you should review the appearance of these controls to ensure you are satisfied, and adjust the style sheets accordingly. Additionally, the header and command areas of the Calendar (where buttons appear) have a new HTML structure, abandoning tables in favor of using `<div>` tags. If you have customized properties that determine the layout of these buttons, you may have to make adjustments to the new properties that determine their appearance. Tech support can help you migrate your files.

Easy to fix breaking changes

- Style sheet classes have been modified to reflect new features. The Web Application Updater program edits the default files and class names they contain to add, edit, or remove items. All changes are documented with comments in the file, including removal, which preserves the old styles within the comments.

Here are a few style sheet classes from the **Calendar.css** file to show how the edits look:

```

/* Property: HeaderRowsCssClass
Rows within the Header. Each is actually a small table.
For changing the font.
*/
.DES_CalHeaderRows
{
  /* font-size:8pt; */ /* deleted in DES 5.0. handled in DES_CalControl */
  /* font-family:Arial; */ /* deleted in DES 5.0. handled in DES_CalControl */
  /* cursor:default; */ /* deleted in DES 5.0. handled in DES_CalControl */
  vertical-align:middle; /* added in DES 5.0 */
  height:16px; /* added in DES 5.0 */
  text-align:center; /* added in DES 5.0 */
}

/* deleted in DES 5.0 Tables are no longer used in the header
.DES_CalHeaderRows TR
{
  background-color:transparent;
}
*/

/* Based on DES_CalHeaderRows for the first row on the left */
.DES_CalHeaderRowsLeft1 /* added in DES 5.0 */
{
  margin-left:3px; /* added in DES 5.0 */
}

```

- The LocalizableLabel control introduces a new property, **AutoHide**, which is active by default. Its job is to set the control's **Visible** property to `false` when the control specified in the **AssociatedControlID** property is already **Visible** = `false`. Look at your code for any LocalizableLabel using **AssociatedControlID** to determine if the other control has its visibility changed programmatically. Determine if you want the label to hide in that case. If not, specify **AutoHide=false** in the LocalizableLabel.

```

<des:MultiSegmentDataEntry id="id" runat="server"
  AssociateControlID="textbox1" />

<asp:TextBox id="textbox1" runat="server" />

```

And in code:

```
textbox1.Visible = false
```

Consider making this change:

```

<des:MultiSegmentDataEntry id="id" runat="server"
  AssociateControlID="textbox1" AutoHide="false" />

```

CONTINUED ON THE NEXT PAGE

- The Context menu built into various Date and Time controls could be prepared programmatically. The technique to populate it has changed. You must use the new **ContextMenuUpdated** event handler to run your code that modifies the Context menu.

Example

See the “Defining the ContextMenuUpdated event handler” of the **Date and Time User’s Guide** for details.

[C#]

```
using PeterBlum.DES.Web;
using PeterBlum.DES.Web.WebControls;
...
private void Page_Load(object sender, System.EventArgs e)
{
    DateTextBox1.ContextMenu.ContextMenuUpdated += new System.EventHandler
    <ContextMenuUpdatedArgs<System.Web.UI.Control>>(DateTextBox1_ContextMenuUpdated);
}

private void DateTextBox1_ContextMenuUpdated(
    object sender, <ContextMenuUpdatedArgs<System.Web.UI.Control>> e)
{
    e.ContextMenu.Items.AddHint(101, "Date Format is MM/DD/YYYY");
}
}
```

[VB]

```
Imports PeterBlum.DES.Web
Imports PeterBlum.DES.Web.WebControls
...
Private Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
    AddHandler DateTextBox1.ContextMenu.ContextMenuUpdated, _
        AddressOf DateTextBox1_ContextMenuUpdated
End Sub

Private Sub DateTextBox1_Menu_ContextMenuUpdated(ByVal sender As Object,
    ByVal e As ContextMenuUpdatedArgs Of System.Web.UI.Control)
    e.ContextMenu.Items.AddHint(101, "Date Format is MM/DD/YYYY")
End Sub
```

- The **AutoPostBackValidates** property on the MultiSegmentDataEntry control switched types from Boolean to PeterBlum.DES.AutoPostBackValidates. To correct this, replace false with AutoPostBackValidates.No. Replace true with AutoPostBackValidates.Control.

```
<des:MultiSegmentDataEntry id="id" runat="server"
    AutoPostBackValidates="true" AutoPostBackValidates="Control" />
```

Upgrading from v4.0

Version 5.0 has had few feature changes since version 4. Most of the work is due to the introduction of the Peter's Business Logic Driven UI module and a major refactoring of the code base. There are several breaking changes to be aware of. See "Breaking changes when upgrading from ANY version".

Here are notable changes:

- Introduction of the **Peter's Business Logic Driven UI** module ("BLD"). (Formerly called **DES Dynamic Data** in its prerelease form.) This module provides a different and very smart way to build applications. It is a number of classes that allow you to separate your business logic from the user interface, and controls that are smart enough to follow your business logic. The result is both better application design and rapid development of a rich UI. See the **BLD User's Guide** or [click here](#) for details.

ALERT: *BLD is intended to be used as you create the basic infrastructure of an application. So start with it as you start a new app, but don't try to use it on existing pages, unless you are prepared to rewrite those pages.*

Existing DES Dynamic Data applications will be converted to use new control and class names. There are a number of breaking changes. There are also a multitude of improvements made, especially when you take advantage of BLD DataAccessObjects to keep the CRUD code separate from the UI. Doing so allows you to have a rich set of filtering tools, both for inside your business logic and as controls within the UI.

- Major refactoring of the code base occurred resulting in the following:
 - Many properties were renamed or relocated. The code base still includes the previous properties in most cases. They are marked OBSOLETE and usually still work. When you compile, you will notice compiler warnings for these changes. The warnings provide direction to correct the issue, yet in most cases you can leave things unchanged.
 - Many controls in the Peter's Date and Time module had a multitude of properties to describe their commands embedded into the control. Now there are properties for each command that host the properties needed to setup the command. For example, the Today command had **TodayCommandName**, **TodayCommandKeys**, and **TodayImageUrl**. Now there is a property called **TodayCommand**, with child properties of **CommandName**, **MenuKeys**, and **ImageUrl**. This reorganization won't break most code since the old properties are depreciated but still work. Each Command class offers new formatting options.
 - There are similar rearrangements of properties into the Calendar.Footer and MultiSelectionCalendar.MessageCenter properties.
 - If you have subclassed from any of DES's classes, there may be *breaking changes*. See the **Breaking Changes** section, below.
- Includes assembly for ASP.NET 4.0 and higher. (Still includes an assembly for ASP.NET 2 but no longer supports ASP.NET 1.x.)
- AJAX setup simplified when using ASP.NET 4 along with Microsoft's ScriptManager and UpdatePanel controls. You no longer have to do the 3 preregistration steps. The software will load scripts, style sheets, and even the popup controls when needed during a callback, not initially. This not only simplifies setup, but also reduces the overhead on the first page request.

```
<des:PageManager id="PageManager1" runat="server"
  AJAXFramework="MicrosoftAJAX"
  PreLoadForAJAX_Validators="True" PreLoadForAJAX_SubmitControls="True" />
<des:DateTextBox id="dummyDateTextBox" runat="server" />
dummyDateTextBox.PreLoadForAJAX()
```

- New control: **DropDownMenu** – A button with built-in ContextMenu, providing an easy way to launch a context menu at a specific location on the page. See the **Interactive Pages User's Guide**.
- New control: **HtmlList** – Similar to the [System.Web.UI.WebControls.BulletedList](#). In fact, it originated because the BulletedList could not handle a few cases. It draws a list of items that postback on click and preserves the selected item. The selected item is drawn with a different style sheet and without the ability to postback on click. Thus it works like a RadioButtonList with a postback upon selection. Use style sheets to drive appearance, such as making the selectable items look like hyperlinks. See the **General Features Guide**.

- Calendar and MultiSelectionCalendar control has much more flexible layout within the content above and below the actual month calendar. These areas host buttons like Next and Previous month, Today, and Clear commands, along with the Current Date Label. Your existing calendar controls should still work without change.
- Calendar and MultiSelectionCalendar control now place an emphasis on style sheet classes where some properties used to adjust their look. For example, instead of using **SideGaps**, **TopGap**, and **BottomGap** properties, edit the .DESCalendar class to have padding styles. Conversion code adjusts existing style sheets and old properties are depreciated but still work.
- The Calendar, MultiSelectionCalendar, MonthYearPicker, and TimePicker have a fresh appearance which does not appear after conversion. To review it, look here: <http://www.peterblum.com/des/dateandtime.aspx>.

To switch to this new appearance, go to the <appSettings> section of the web.config file and remove this line:

```
<add value="DES_DES4Appearance" value="" />
```

- The UnselectableTimesValidator now supports a times that are unselectable. In the past, you could show unselectable times using a TimePickerTimeFiller, but that value was not evaluated by the validator. Now use the new TimePickerUnselectableTimeValue class. Specify the **TimeValue** as the start of a range of unselectable times. Use the **UnselectableSeconds** property to define the number of seconds of the range. If you want to hide the time from the TimePicker, but still validate, set the **Hidden** property to true.
- Validators now can evaluate other sources of data than controls. In the past, you always assigned data through the **ControlIDToEvaluate** or **ControlToEvaluate** properties. Now you can evaluate values from other sources. For example, your business logic may want to compare two integer variables. That would be a case for the CompareTwoFieldsValidator, if only you could avoid using controls to deliver those two integers.

If the Validator has a **ControlIDToEvaluate** property, it also has a **ValueToEvaluate** property. When **ValueToEvaluate** is assigned the data, you no longer assign **ControlIDToEvaluate**. Similarly, the **SecondControlIDToEvaluate** property can be ignored if you assign values to **SecondValueToEvaluate**.

Similarly, Condition objects offer **ValueToEvaluate**, and **SecondValueToEvaluate** properties. They also can be used in a very similar way, as shown below.

See the “Using Validator controls to evaluate data not from Controls” topic in the **Validation User’s Guide**.

- AlertImageErrorFormatter, ToolTipImageErrorFormatter, and PopupErrorFormatter join the other ErrorFormatters in offering the **HTMLBefore** and **HTMLAfter** properties, to allow additional HTML to enclose the image they show.
- Most controls offer the **ViewStateMgr** property. The Button, ImageButtons, and LinkButtons controls now join the list of those that do. Additionally, you can use the **ViewStateMgr** to track a list of properties by assigning a pipe delimited list of property names to the **PropertiesToTrack** property. This avoids writing code to call `ViewStateMgr.TrackProperties()`.
- CalculationController offers the **RunOnlyOnDemand** property which indicates that this calculation cannot run on the client-side unless you call the `DES_CalcOnDemand()` javascript function. You generally setup the onchange event to textboxes that will invoke this function.
- The LocalizableLabel control introduces a new property, **AutoHide**. It sets the label’s **Visible** property to false when the control specified in the **AssociatedControlID** property is already **Visible = false**. See **Breaking Changes** below for additional information.
- CountTrueConditionsValidator and CountTrueConditions classes can generate the child Condition objects for each row of a ListView, GridView, Repeater, or DataList control, helping you avoid writing code. Just assign the databound control to the **ListControlID** property and add one Condition object to the **Conditions** collection.
- CombinedErrorMessages control now offers **HiliteFields**, **Label**, and **SecondLabel** properties, which already exist on Validator controls.
- The ContextMenu control has a new property, **AutoScroll**, which can be used to add scrollbars to large menus.
- CalculationController offers new CalcItem classes: **TotalingCalcItem** and **CheckStateCalcItem**. Use TotalingCalcItem to create a total of a column within a ListView, GridView, DataGrid, or Repeater. Use CheckStateCalcItem to add one of two constant values based on the state of a checkbox or radiobutton.

- The **AutoPostBackValidates** property on the MultiSegmentDataEntry control now allows validation based on a validation group, much like other controls with the **AutoPostBackValidates** property. Set it to Group and set the validation group name in the **ValidationGroup** property. This is also a **breaking change** as the property switched types from Boolean to PeterBlum.DES.AutoPostBackValidates. See **Breaking Changes** below for additional information.
- The PopupErrorFormatter class has these new properties:
 - **PopupOnMouseExit**, which determines if **PopupOnMouseOver** will pop down the PopupView once the mouse moves off the error formatter.
 - **TargetIsValidator**, which changes the target control for the PopupView from the data entry control to the validator's errorformatter position.

Upgrading from pre 4.0 products: Changes to Installation

Most of the manual installation process has been replaced by the **Web Application Updater** program. See the **Installation Guide**. This program will:

- Perform a first time installation
- Install a service release
- Convert native buttons and textboxes to their DES equivalents – Replaces the **Convert Page to VAM.exe** utility
- Upgrade from Professional Validation And More
- Migrate from Peter's Date Package

Professional Validation and More Users

This section describes changes made since Professional Validation and More v3. *If you want a document with changes made since v2, please email support@peterblum.com.*

Click on any of these topics to jump to them:

- ◆ [Breaking Changes](#)
- ◆ [New module: Peter's Date and Time](#)
- ◆ [Design Mode General Features](#)
- ◆ [Peter's Professional Validation and Peter's More Validators modules](#)
(was VAM: Essential Validators and VAM: Specialized Validators)
- ◆ [Peter's TextBox module](#) (was VAM:Data Entry Controls)
- ◆ [Peter's Interactive Pages module](#) (was VAM: Client-Side Toolkit)
- ◆ [Peter's Input Security module](#) (was VAM: Visual Input Security™)

Breaking Changes

The **Installation Guide** section “Upgrading from Professional Validation and More” provides extensive details on conversion issues in the FAQ section.

Impact of refactoring the DES v4 code base

See “[Breaking changes when upgrading from ANY version](#)”. **This is very important.** If you encounter issues converting your custom classes based on VAM controls, you can elect to upgrade to DES v4 which doesn't have these issues.

Obsolete Properties

The following properties are obsolete and no longer supported. (There are many obsolete properties that are hidden but still work to allow for conversion.)

- On any textbox, the properties **IncrementButtonUrl**, **DecrementButtonUrl**, **AutoRepeatSpeed1** and **AutoRepeatSpeed2** have been removed. They are relocated to the page-level property **PeterBlum.DES.Globals.Page.SpinnerManager**. This allows a consistent setup for all spinners on the page.
If you have used any of these properties, you will need to assign their values to the **SpinnerManager** property, which is set in **PeterBlum.DES.Globals.Page** in `Page_Load()` and on the PageManager control. Alternatively setup them up in the “SpinnerManager” topic of the **Global Settings Editor**.
- On any textbox, the **SupportClientSideLookupByID** property is ignored if used. It will be automatically setup.

Style sheets

VAM kept its style sheets in a single file, **VAMStyleSheets.css**. DES has split up this file so categories of controls have their own style sheet files. DES is also using different names for style sheet classes. The term “VAM” has been replaced by “DESVAL” in the names (example: `VAMFieldWithError` → `DESVALFieldWithError`).

To avoid major hassles in conversion, DES still supports the old **VAMStyleSheet.css** file and the old names. Please see “FAQs – Upgrading from Professional Validation And More” question 2 in the **Installation Guide** for directions.

Compatibility with your existing scripts

The names of every JavaScript function declared in **Professional Validation And More** have changed. In some cases, the term “VAM” is replaced by “DES” such as `VAM_GetById()` is now `DES_GetById()`. In other cases, a more complex renaming occurred. There are some functions that take different parameters.

Fortunately, DES includes a script file that maps the old names to the new ones. It is added automatically when you upgrade. The **Web Application Updater** program adds this line to `<appSettings>` of **web.config** to provide this service. If you don't need the converted scripts, you can remove it:

```
<add key="DES_VAMCompatibleScriptFile" value="" />
```

DES provides the source script file **VAMCompatible.js** in the **[DES product folder]\Upgrading** folder. Use it if you are getting JavaScript errors to see if a function you are using is still supported. If not, please allow Tech Support assist you (support@peterblum.com).

New module: Peter's Date and Time

A reworking of the **Peter's Date Package** product, to take advantage of the **Professional Validation And More** code base and more tightly integrate the two. This product brings in many new controls, including a DateTextBox, Calendar, and TimeOfDayTextBox.

See the **Date And Time User's Guide**. If you already use **Peter's Date Package**, see "[Changes made converting Peter's Date Package v2](#)".

Despite its presence, you do not have to convert any existing **Peter's Date Package** controls to DES's equivalents unless you want to use a feature introduced in DES.

Design Mode

- Extensive use of SmartTags (Ⓛ). Most controls provide the most common properties for quick setup.
- Expanded Properties Editor. This tool extends the features of the existing Properties Editor.

The most powerful feature of this Properties Editor is the “Best Order” button immediately above the property names. Click it to have the properties and categories listed in the order recommended by PeterBlum.com for setting up the control.

Once the Best Order button is selected, the toolbar looks like this:



All properties have been assigned one of these states: Required, Recommended, Sometimes used, or Rarely used. Click the **Recommended Properties** button to show only the Required and Recommended properties. This is a very good way to setup a newly added control. If you want to customize which states are shown, use the **Properties Assistant** button.

General Features

- Features shared amongst the modules are now documented in a single place: the **General Features Guide**. This includes AJAX, the String Lookup System, and much more.
- Client-side scripts are no longer taken from stand-alone files. They are read from resources in the assembly.
- A single `<script>` tag is generated, which combines all requested script files. This reduces the number of transactions made between browser and server.
- Style sheets are compressed, removing comments (which are extensively used) and optionally whitespace.
- A single `<link>` tag is generated, which combines all requested style sheet files. This reduces the number of transactions made between browser and server.
- **ViewStateMgr** property added to most of the controls. Provides an efficient way to use the ViewState by explicitly identifying properties to track. See “The ViewState and Preserving Properties forPostBack” in the **General Features Guide**.
- **ViewStateMgr** now automatically saves the two most popular properties, Visible and Enabled. You no longer have to track them explicitly. (It uses a single byte to represent the state of these properties.)
- The **Global Settings Editor** has been rewritten with a new user interface. Instead of a Wizard, it is a treeview of topics. See the **General Features Guide**.
- String Lookup System previously had a few “string groups” to separate different types of strings. Now there are many more strings groups to better track different types of data. For example: confirm messages, hints, validation errors, date and time module strings, etc. See the **General Features Guide**.
- When using Microsoft ASP.NET AJAX, DES is smarter about outputting scripts. You no longer need to pass the ScriptManager or UpdatePanel control to the `AJAXManager.UsingMicrosoftAJAX()` manager method (although those parameters are still supported). In fact, you don't have to call that function if you are using the new PageManager control, because you can set its `AJAXFramework` property to `MicrosoftAJAX`. See the “Using these controls with AJAX” section of the **General Features Guide**.

By default, DES will set the **InAJAXUpdate** properties when its controls are found in UpdatePanels, RadAJAXPanels, MagicAJAX's AJAXPanel, and RadAJAXManager (although that requires setup). It also sets **InAJAXUpdate** on DES controls outside of these controls in many cases, although you will have to set it in some situations.

While its much simpler to setup, you still need to use the “preregister for AJAX” features when a type of control is not on the page until a callback creates it.

- The **VAM** folder has been replaced by the **IDES** folder. The **IDES** folder serves the same purpose as before. Conversion will retain the **VAM** folder but it is not used by DES. If you are not using VAM and DES side-by-side in the app, you can delete it.
- The **custom.vam.config** file has been replaced by **custom.des.config**. It serves the same purpose as before. Conversion copies your **custom.vam.config** file to the **IDES** folder and names it **custom.des.config**.
- The namespace `PeterBlum.VAM` has been replaced by the namespace `PeterBlum.DES`.
- Support for Telerik RadControls “Prometheus”. RadEditor, RadComboBox, RadMaskedTextBox, RadTextBox, RadNumericTextBox, RadDateInput, and RadDatePicker work with validators. RadMenu and RadGrid can provide client-side validation before they submit the page. See the **Using Third Party Controls** guide.
- New debugging reports that output runtime information for licenses, ajax setup, global settings, page-level settings, validation and style sheets. Once setup, you add a querystring parameter to the page's URL that invokes the reports. See the “Exploring the Current Settings” section of the **General Features Guide**.

PageManager Control

The PageManager control lets design mode and ASP.NET declarative syntax users set properties on **PeterBlum.DES.Globals.Page** so you don't have to write any code. For design mode users, it makes sense to add this control to each web form using DES controls early on, so it's ready for you when you need it.

Each of its features relates to an aspect of DES that is covered elsewhere. The PageManager control effectively groups together page-level settings in one place.

You can set these features with the PageManager control:

- Make DES aware of AJAX on the web form
- Validation page-level properties
- Culture used for localizing the page
- HintManager – rules used by the Interactive Hints feature
- SpinnerManager – rules used by spinners on textboxes
- ChangeMonitor – rules used by the ChangeMonitor
- An assortment of other properties from **PeterBlum.DES.Globals.Page**.

The SmartTag  for the PageManager has many useful features:

- Quickly setup AJAX
- Access to the most popular validation properties
- Run the **Global Settings Editor** (also in the controls' context menu)
- Open any of the User's Guides (also in the controls' context menu)

NativeControlExtender Control

The NativeControlExtender lets you add some DES features to the native controls. It also extends controls for which DES has no equivalent.

- Apply the Interactive Hints system to almost any control. For controls that allow focus, show a Hint on a PopupView or Label. For most, switch from the standard tooltip to a PopupView (shown to the right).
- Extend native Buttons, LinkButtons, and ImageButtons, with these DES features:
 - DES Validation (client-side only)
 - Disable On Submit
 - ConfirmMessage
- Extend these controls which can submit the page to offer client-side DES validation and confirmation messages:
 - BulletedList when **DisplayMode** = LinkButton.
 - Menu for selected menu items
 - TreeView for TreeNodes with a **SelectAction** of Select or SelectExpand. *ConfirmMessages are not offered on this control.*
- ChangeMonitor monitors edits on controls that support client-side onchange and onclick events.
- Provide DES validation to controls using AutoPostBack.
- Intercept the ENTER key and use it to click a button.
- Extend TextBoxes with DES's SmartChange feature that fires the client-side onchange event in cases where it would be expected but doesn't happen: after using the AutoComplete menu and after a programmatic edit.

PeterBlum.DES.Globals.Page

- **PageIsLoading** property lets you change the default text shown in an alert that says "Page is loading. Please wait".

Peter's Professional Validation and Peter's More Validators modules

Previously named VAM: Essential Validators and VAM: Specialized Validators

Click on any of these topics to jump to them:

- ◆ [New Validators](#)
- ◆ [All Validators](#)
- ◆ [DataTypeCheckValidator](#)
- ◆ [RangeValidator](#)
- ◆ [CompareToValueValidator](#)
- ◆ [EmailAddressValidator](#)
- ◆ [CreditCardNumberValidator](#)
- ◆ [ValidationSummary](#)
- ◆ [DES Buttons](#)

New Validators

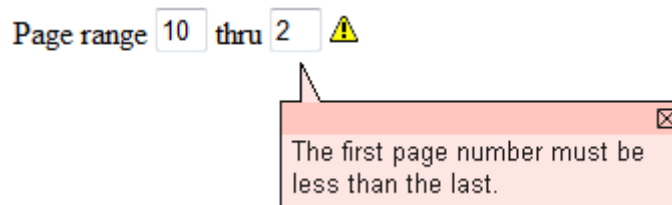
See the **Validation User's Guide** for all of these features.

- **MultipleRequiredControlsValidator** – The **MultipleRequiredControlsValidator** evaluates two or more data entry controls to evaluate if they are blank or have data. This is an extension of the idea behind the **RequiredTextValidator** and **RequiredListValidator** by looking at the state of several controls and determines if the right combination has data or not. You can demand the following: All, All or None, Only one, At least one, or a range.
- **ListSizeValidator** - The **ListSizeValidator** evaluates a listbox or dropdownlist to determine if has enough elements listed. It is used when the UI allows adding and removing elements to a list. Typically users want to report an error when the list is empty, however, this allows a range (Min/Max) to provide more flexibility.
- **RequiredSelectionValidator** – When a selection is required on controls that are not textual or lists. Includes **ListView**, **DataList**, DES's **Calendar**, **MultiSelectionCalendar**, **MonthYearPicker**, and **TimePicker**.

All Validators

See the **Validation User's Guide** for all of these features.

- **New ErrorMessageFormatter: PopupErrorMessageFormatter**. Using the new **PopupView** feature, the error message appears as a floating box next to the control with the error, either when focus is in that field or the mouse is over the image where the validator resides. This is like the **ValidationCallout** control in the ASP.NET AJAX Control Toolkit, except it's a lot fancier.



You have extensive control over the presentation of the **PopupView**, using the **Global Settings Editor**. You can supply a second value in the **ErrorMessageHelp** property that compliments the original **ErrorMessage** by offering a **Help** button which can show more text, invoke a hyperlink URL, or run a script.

- **TextErrorMessageFormatter** and **HyperLinkErrorMessageFormatter** now support the {DEFAULT} token in their **ImageUrl** property. Use the **Global Settings Editor** to define a global default for your graphic image in **DefaultImageErrorMessageFormatterImageUrl**.
- In VAM, the following **PeterBlum.VAM.Globals.Page** properties were initially unassigned: **ControlErrorCssClass**, **ListErrorCssClass**, **CheckBoxErrorCssClass**, **TextHiliteFieldCssClass**, and **NonTextHiliteFieldCssClass**.

Because of this, the user had extra work to set them up: they had to assign the style sheet and update the style sheet class definition.

Now these are preassigned and their style sheet classes all have some styles defined.

New properties

Because they are preassigned, the user needs a way to turn on and off the feature. They use these new properties:

- **ChangeStyleOnControlsWithError** – When `true`, the **ControlErrorCssClass**, **ListErrorCssClass**, and **CheckBoxErrorCssClass** properties are active. The global setting is from the **DefaultChangeStyleOnControlsWithError** property in the **Global Settings Editor**.
- **HiliteFieldsNearbyError** – When `true`, the **TextHiliteFieldCssClass** and **NonTextHiliteFieldCssClass** properties are active. The global setting is from **DefaultHiliteFieldsNearbyError** property in the **Global Settings Editor**.

Conversion from VAM

The **Web Application Updater** will predefine the above two properties as `true` so your existing setup continues to work.

The **DES\Appearance\Validation\Validation.css** file has modified the style sheets used by **TextHiliteFieldCssClass** and **NonTextHiliteFieldCssClass** to provide styles.

- **ReportErrorsAfter** property - Any validator that can evaluate two or more controls can use the **ReportErrorsAfter** property to delay the onchange validation until the user moves out of those controls. It allows them to edit all controls first before seeing an error.

Suppose you have 3 textboxes and use a validator to require that all have text. In the past, as you edited each textbox, it would evaluate the validator. So when the first textbox was finished, the validator error would appear. With **ReportErrorsAfter** set to `AllEdits`, the focus must leave all 3 textboxes before the validator is evaluated.

Several of the validators set **ReportErrorsAfter** to `AllEdits` by default.

- **OtherStyles** property - Determines the source for style sheets associated with two features that are not controlled by the **ErrorFormatter** property.
 - The **PeterBlum.DES.Globals.Page.ChangeStyleOnControlsWithError** property assigns a style sheet to the control to evaluate when there is an error. (See above.)
 - The ValidationSummary Control shows the validator's error messages using a specific style sheet defined in its **ErrorMessageCssClass** property.

Use the **OtherStyles** property to use a second set of style sheets for those features to differentiate them from other controls. For example, differentiating required field errors from other errors.

- The **Data Type** property supports several new types:
 - Percent – Percentage numbers. Requires Peter's TextBoxes module.
 - Percent with symbol – Percentage numbers that includes the % symbol. Requires Peter's TextBoxes module.
 - Time – Time of day. Requires Peter's Date and Time module
 - Duration – Time duration. Requires Peter's Date and Time module
 - DateTime – Date and time combined. Requires Peter's Date and Time module
- **PeterBlum.DES.Globals.Page.AutoDisableValidators** property - Validators normally evaluate controls that are hidden or disabled, unless you use their **Enabler** property with the `VisibleCondition` and/or `EnabledCondition`. This is excessive work for something that is pretty common: not validating hidden/disabled controls.

AutoDisableValidators makes validators detect if any of the controls they evaluate are hidden or disabled and turns the validator off. This property is available in `Page_Load()` using **PeterBlum.DES.Globals.Page** and in the `PageManager` control.

While this feature is on by default, when converting from VAM, it is off by default because VAM didn't have it and conversion shouldn't change the current behavior of a site. You can use the global setting **DefaultAutoDisableValidators** to adjust it in the **Global Settings Editor**.

- **PeterBlum.DES.Globals.Page.DelayAlertOnSubmitCount** property – When using **PeterBlum.DES.Globals.Page.ShowAlertOnSubmit**, you can delay the alert from appearing until the user has attempted to submit several times. Use it when you normally wouldn't show an alert because it helps the user when they click the submit button several times without noticing the errors that are preventing the page from being submitted.

When 0, the alert shows each time. When 1, it shows on the second attempt to submit. When 2, it shows on the third attempt, etc.

- **PeterBlum.DES.Globals.Page.WarningConfirmOnSubmit** property – Validators can be used as warnings instead of errors when you set their **EventsThatValidate** property to **OnChange**. A warning looks like a validator error, but it does not prevent submitting or saving the page. For example, you may prefer the user to enter a birth date that is at least 18 years old and if they don't tell them they need a parent to come to the appointment with them.

To remind the user of these warnings, you can show a confirmation message that lists the warnings on the page. When **PeterBlum.DES.Globals.Page.WarningConfirmOnSubmit** is **true**, the confirmation appears when the user attempts to submit the page. The user can click OK to submit the page or Cancel to prevent submission. The warnings will be on the page at that time for their review.

- **PeterBlum.DES.Globals.Page.AutoHideRequiredFieldMarkers** property – When a validator's **ShowRequiredFieldMarker** property is **true** and it has an Enabler, set **AutoHideRequiredFieldMarkers** to **true** to hide the RequiredFieldMarker any time the validator is disabled.

If you feel that the RequiredFieldMarker does not belong when the validator is disabled, use this feature.

- **PeterBlum.DES.Globals.Page.DefaultErrorFormatterSkinID** property – Provides a page-level value for each Validator's **ErrorFormatterSkinID** property. It is used when **ErrorFormatterSkinID** is set to "{DEFAULT}", which is the default case. This property makes it easy to set the skinID of all validators at once.
- **PeterBlum.DES.Globals.Page.BeforeValidation** event – This event handler is fired first when the **DESPage.Validate()** method is called. Use it to do any pre-validation preparation.
- The **ErrorFormatterSkinID** property on each validator has been extended to allow a property list that modifies the skin's value. The syntax is **ErrorFormatterSkinID="skinID {PropertyName='value'}"**. Multiple properties are allowed in a space delimited list. This is an excellent way to modify something like the **Display** property, which defaults to **Static** but you may want it as **Dynamic** without creating a new skin.

```
ErrorFormatterSkinID="skinID {Display='Dynamic'}"
```

This feature is also supported on the new **DefaultErrorFormatterSkinID** property.

DataTypeCheckValidator

See the **Validation User's Guide** for all of these features.

- New method **ControlToNative()** returns the native data type found on the control being evaluated. For example, when evaluating with a **DataType** of **Date**, it returns a **System.DateTime** object. Use it to get the value you will save in your database. It is very useful when the validator is evaluating a textbox because you don't have to parse the **TextBox.Text** property while carefully applying all of the parsing rules used by DES. Instead, you let DES do the parsing.
- The **DataTypeCheckCondition** (but not the validator) offers **Minimum** and **Maximum** properties. When set, they are the range used by this condition to report an error. Use them in cases where you feel a range is considered a data type error instead of a range error, requiring a separate error message.
- When used with any of DES's numeric, date or time textboxes, it can respect their minimum and maximum values (**MinValue** for numeric, **MinDate** for date, and **MinTime** for time) when the control's **DataTypeCheckReportsRangeErrors** property is **true**.

RangeValidator

See the **Validation User's Guide** for all of these features.

- The **Minimum** and **Maximum** properties are strings that must represent values of a native data type in a certain format, such as a date in **CultureInfo.DateTimeFormat.ShortDatePattern**. It is easier if you can assign the native data type (like a DateTime or integer) and let DES convert them to a string. Use **MinimumAsNative** and **MaximumAsNative** properties to do this.

CompareToValueValidator

See the **Validation User's Guide** for all of these features.

- The **ValueToCompare** property is a string that must represent the value of a native data type in a certain format, such as a date in **CultureInfo.DateTimeFormat.ShortDatePattern**. It's easier if you can assign the native data type (like a DateTime or integer) and let DES convert them to a string. Use **CompareToValueAsNative** property to do this.

EmailAddressValidator

See the **Validation User's Guide** for all of these features.

- If you want to allow multiple email addresses in the same textbox, set **MultipleAddressesAllowed** to `true`. You can define the delimiter character in **DelimiterCharacter**, which defaults to a semicolon. You can allow spaces after the delimiter when **DelimiterAllowsSpaces** is `true`.

CreditCardNumberValidator

See the **Validation User's Guide** for all of these features.

- The **AllowSpaces** property lets the user enter spaces into the credit card number. Those spaces are ignored by the validator. Your server side code will probably use the text without spaces, which is available from the **CleanedUpNumber** property.

ValidationSummary

See the **Validation User's Guide** for all of these features.

- When using the **AutoUpdate** property, you can now specify when the ValidationSummary first appears, using the **AutoUpdateFirstShows** property. While it defaults to showing when the user attempts to submit the page, its values let you show it when anywhere from 1-5 errors is shown on the page. Use **AutoUpdateFirstShows=OneError** to show it on the first error.
- Using the Related Control feature, you can show a message when all errors are resolved or anytime the ValidationSummary is hidden by using the **RelatedControlDisplayMode** property.
- The **HeaderText** and **FooterText** properties support two tokens: `{COUNT}` and `{COUNT:singular:plural}`. They state the number of errors shown. For example, "Found: `{COUNT}` `{COUNT:error:errors}`" will show "Found: 1 error" and "Found: 5 errors".

DES Buttons

See the **Validation User's Guide** for all of these features.

- **SkipPostBackEventsWhenInvalid** property - After running server side validation, the DES Buttons (Button, LinkButton, and ImageButton) can test **PeterBlum.DES.Globals.Page.IsValid** and automatically skip the **Click** and **Command** event handlers as determined by this property.

The feature is off by default. It uses the global setting **ButtonsSkipPostBackEventsWhenInvalid** in the **Global Settings Editor** when **SkipPostBackEventsWhenInvalid** = Default.

- **ButtonValidationState** property – A readonly property that tells how the button handled validation. It returns these values:
 - No – It was not the button that fired postback. If called on or before `Page_Load()`, it will always be No even if the button fired postback.
 - ValidationOff – Postback event handlers were run but **CausesValidation** is false.
 - Valid – It ran validation and **PeterBlum.DES.Globals.Page.IsValid** is true.
 - Invalid – It ran validation and **PeterBlum.DES.Globals.Page.IsValid** is false.

Peter's TextBox module

Previously named VAM: Data Entry Controls

Click on any of these topics to jump to them:

- ◆ [New Controls](#)
- ◆ [All TextBoxes](#)
- ◆ [All Numeric TextBoxes: Integer, Decimal, Currency, Percent](#)
- ◆ [IntegerTextBox](#)
- ◆ [DecimalTextBox](#)
- ◆ [CurrencyTextBox](#)
- ◆ [MultiSegmentDataEntry](#)
- ◆ [Validation Support](#)

New Controls

See the **TextBoxes User's Guide** for all of these features.

- **PercentTextBox** control - The **PercentTextBox** is a **TextBox** designed for percent data entry. It knows how to convert a `System.Double` and `System.Integer` types into text and back. It has properties to determine the number of decimal places or even restrict to integer entry, if it supports negative numbers and shows the percent symbol. You can add a spinner control for the user to increment the value.

All TextBoxes

See the **TextBoxes User's Guide** for all of these features.

- **ConvertCase** property – Forces letters to be upper or lower case.
- **TabByArrowKeys** property – Advance the focus to the next or previous control (identified by **NextControlID** and **PreviousControlID**) as the user types an arrow key that hits the start or end of the textbox.
- **TabOnBackspace** property – Move the focus to the previous control when the user types a backspace and the textbox is empty.
- **ValueWhenBlankCssClass** property now supports the “+” notation allowing its style to merge with the existing **CssClass** property instead of override it. This preserves styles from **CssClass** and only overrides the styles that are introduced in **ValueWhenBlankCssClass**.

All Numeric TextBoxes: Integer, Decimal, Currency, Percent

See the **TextBoxes User's Guide** for all of these features.

- Support up and down arrow keys to increment and decrement the current value.
- Spinners have been reworked in several ways:
 - The properties **IncrementButtonUrl**, **DecrementButtonUrl**, **AutoRepeatSpeed1** and **AutoRepeatSpeed2** have been removed from the textbox and relocated to the page-level property **PeterBlum.DES.Globals.Page.SpinnerManager**. This allows a consistent setup for all spinners on the page. These properties have defaults setup in the **Global Settings Editor** so you can have consistent setup throughout the site.
 - Mouseover effects now available. Previously you had two images: mouse up and mouse down. Now there is a mouse over image. If you are not using the default images, you can create a mouseover image using the same name as the mouse up graphic + “MouseOver”: `UpArrow1.gif` (mouseup), `UpArrowMouseOver.gif`, and `UpArrowMousePressed.gif`.
- The **SpinnerMinValue** and **SpinnerMaxValue** properties have been replaced by **MinValue** and **MaxValue**. They support more than just spinners now.

- RangeValidator automatically establishes its limits from these properties if its own **Minimum** and **Maximum** are not setup.
- Up and down arrows on the keyboard stop when they hit these limits.

While **MinValue** and **MaxValue** are string types, you can assign integer and decimal types to their equivalent properties: **MinValueAsNative** and **MaxValueAsNative**. This avoids you doing the native to string conversion incorrectly.

- **DataTypeCheckReportsRangeErrors** property – When `true`, the `DataTypeCheckValidator` reports an error when the value is out of range as determined by the textbox's **MinValue** and **MaxValue** properties.
- **AutoContainer** property replaced by **ContainerMode** property (old property still works). **ContainerMode** introduces several additional ways to create a container when the textbox has spinners.
- **ReadOnlyAllowsEdits** property – When the textbox is **ReadOnly**, the spinners and up/down arrow keys will still work if this is `true`.
- **AcceptPeriodAsDecimalSeparator** property – Lets cultures whose decimal separator is not a period to accept a period character in addition to their decimal separator. Great for numeric keypad entry of numbers.

IntegerTextBox

See the **TextBoxes User's Guide** for all of these features.

- **IntegerBindable** property is a better choice for databinding than its **IntegerValue** counterpart. It is typeless, accepting a variety of types including various size integers and a string containing an integer value.
- **IntegerNullable** property allows assignment of an integer type or `null`. When set `null`, **TextBox.Text** = "". When get and **TextBox.Text** = "", returns `null`.

DecimalTextBox

See the **TextBoxes User's Guide** for all of these features.

- **DoubleBindable** property is a better choice for databinding than its **DoubleValue** counterpart. It is typeless, accepting a variety of types including double, decimal, single, various size integers and a string containing a floating point value.
- **DoubleNullable** property allows assignment of a double type or `null`. When set `null`, **TextBox.Text** = "". When get and **TextBox.Text** = "", returns `null`.
- **DecimalValue** property allows assigning a Decimal type

CurrencyTextBox

See the **TextBoxes User's Guide** for all of these features.

- **DoubleBindable** property is a better choice for databinding than its **DoubleValue** counterpart. It is typeless, accepting a variety of types including double, decimal, single, various size integers and a string containing a floating point value.
- **DoubleNullable** property allows assignment of a double type or `null`. When set `null`, **TextBox.Text** = "". When get and **TextBox.Text** = "", returns `null`.
- **DecimalValue** property allows assigning a Decimal type
- **WholeNumberOnly** property - Prevent entry of decimal part of currency while still formatting for currency.

MultiSegmentDataEntry

See the **TextBoxes User's Guide** for all of these features.

- **TabByArrowKeys** property – Advance the focus to the next or previous segment as the user types an arrow key that hits the start or end of the textbox.
- **TabOnBackspace** property – Move the focus to the previous segment when the user types a backspace and the textbox is empty.

Validation Support

See the **TextBoxes User's Guide** for all of these features.

- Since **Peter's TextBoxes** module can be purchased as a stand-alone module, they need to support native ASP.NET validation framework. DES includes the assembly **PeterBlum.DES.NativeValidators.dll** with equivalents to the CompareValidator and RangeValidator for use with the numeric textboxes. It introduces a DifferenceValidator for the numeric textboxes and MultiSegmentDataEntryValidator for the MultiSegmentDataEntry control.

Peter's Interactive Pages module

Previously named VAM: Client-Side Toolkit.

Click on any of these topics to jump to them:

- ◆ [New Controls and Tools](#)
- ◆ [FieldStateControllers](#)
- ◆ [CalculationController](#)
- ◆ [Interactive Hints](#)
- ◆ [Enhanced Buttons](#)

New Controls and Tools

See the **Interactive Pages User's Guide** for all of these features.

- **Enhanced ToolTips** - The browser provides the tooltip to describe almost any field as the mouse passes over it. That tooltip is very limited. For most browsers, it cannot be multiline. It has one style (yellow). It cannot support HTML.

Using the same PopupView feature found in Interactive Hints and the DES Validator's PopupErrorFormatter, DES gives you a better tooltip. You control its appearance and supply it with HTML to convey the information better.
- **ChangeMonitor** - The ChangeMonitor watches for edits in the form and changes the appearance of buttons and other fields upon the first detected edit.

The classic case is to have a disabled OK button that gets enabled as you start typing. Another case is to show a message like "This form has changed" in a label. Both of these cases are handled.

DES's enhanced buttons are already capable of showing a confirmation message. With the ChangeMonitor in use, that message can be shown based on whether or not the user has edited the form.
- **TextCounter Control** - Displays the number of characters or words within a textbox. It assists users when there are limits to the size of text they can enter. It compliments, but does not replace the TextLengthValidator/WordCountValidator, because it does not impose a limit. It merely communicates the count and if a limit is exceeded.

The user interface of the TextCounter can act like a dynamically updated Label control. It also can display itself in the Interactive Hints feature of DES TextBoxes.
- **ContextMenu Control** – Originally found in Peter's Date Package, its now part of DES in the Peter's Interactive Pages module. It provides a context menu (the right click menu) where you define commands and their associated JavaScript to invoke.

FieldStateControllers

See the **Interactive Pages User's Guide** for all of these features.

- **RevalidateOnly** property - The FieldStateController can fire validators attached to the control it's changing when you use these properties: **ValidationChangedControls** and **UseValidationGroups**. Often users prefer that it only shows or hides validators that the user previously triggered. For example, you have 4 textboxes each with a RequiredFieldValidator. You use a FSC and have it validate those textboxes. If the user didn't edit those textboxes, they will still report "Required".

With the **RevalidateOnly** property set to `true`, it will only validate those that the user previously triggered. The rest will not display themselves.

RevalidateOnly requires the **ValidationChangedControls** or **UseValidationGroups** property be `true`.
- **UpdateWhileEditing** property – Determines if the FieldStateController is triggered as the user types into a textbox that it uses to evaluate its condition. By default, it does not and only triggers when focus leaves the textbox. Set this to `true` to evaluate the FieldStateController with each keystroke.

CalculationController

See the **Interactive Pages User's Guide** for all of these features.

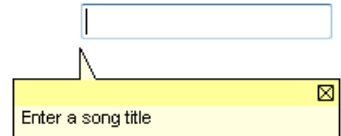
- **InvalidLabelCssClass** property will merge its style with the existing style of the label when the plus (+) character is the first character in the property name. This allows preserving the overall look of the label from the original style sheet while adding a few additional styles from the **InvalidLabelCssClass**.

Interactive Hints

See the **Interactive Pages User's Guide** for all of these features.

- The **PopupView** feature is now available for hints. A **PopupView** floats near the control with the hint and has extensive formatting capabilities. It has a close box, is draggable, and uses opacity to make it easier to work with when it covers other objects.

When the **HintDisplayMode** is setup to **Popup**, the hint uses a **PopupView** whose name is defined in the **PopupViewName** property. *These properties are found on the new **HintFormatter** object, as described below.*



You define **PopupView** definitions in the **Global Settings Editor**. Controls that support hints now offer the **HintHelp** property, which expands the **PopupView** with a Help button that can show additional text, hyperlink to a URL, or run a script.

- Page-level settings moved from **PeterBlum.VAM.Globals.Page** to **PeterBlum.DES.Globals.Page.HintManager**. The **HintManager** is also available using the **PageManager** control.

Conversion note: The old properties are still there, but hidden. Your code will work without modification.

- Hints are now described using the **PeterBlum.DES.HintFormatter** class. It encapsulates formatting rules, such as **HintDisplayMode**, **HintInStatusBar**, and **HintControlID**. These properties have been renamed within the **HintFormatter**.
- Define a list of shared **HintFormatters**, each with a unique name, in the **HintManager.SharedHintFormatters** property. This lets you just pick a name on the data entry control instead of setting up a **HintFormatter** each time.
- **TextBoxes** have only 4 hint properties: **Hint**, **HintHelp** (new), **SharedHintFormatterName**, and **LocalHintFormatter**. **MultiSegmentDataEntry** also has these properties, with **Hint** and **HintHelp** located on individual segments. The **SharedHintFormatterName** property uses a **HintFormatter** from **HintManager.SharedHintFormatters**. **LocalHintFormatter** lets you define a unique **HintFormatter** on the **textbox** or **MultiSegmentDataEntry** control.

Conversion note: The old properties for setting up hints are still there, but hidden. Your code will work without modification.

- Use the new **HintFormatter.TextFunctionName** property to run a function that modifies the text of the hint prior to showing it.
- The **Hint** property supports a new token, "{NEWLINE}". It is replaced by the appropriate text for a newline, such as `
` in HTML and a space in the status bar.

Enhanced Buttons

See the **Interactive Pages User's Guide** for all of these features.

- Like DES's Buttons, the EditCommandColumn, ButtonColumn, CommandField, and ButtonField controls have been extended to support **DisableOnSubmit** and **MayMoveOnClick** properties.
- LinkButton hides its JavaScript from the status bar. Normally browsers show the contents of the <a> tag's href= attribute in the status bar. LinkButton hides it with limitations: browsers like IE 7 and FireFox 2 do not allowed window.status to be changed without a user setting.
- ImageButton can use separate graphics for mouse pressed and mouseover effects. See the **MultipleImages** property.
- ImageButton can appear visually dimmed by using the browser's opacity feature. The FieldStateController, ChangeMonitor, and DisableOnSubmit feature all impose this appearance when they cause the button to disable.
- Buttons support an alternative style sheet class when disabled with the **WhenDisabledCssClass** property. ImageButton also supports an alternative image when disabled with the **WhenDisabledImageUrl** property.

Peter's Input Security module

Previously named VAM: Visual Input Security™.

No product changes.

Changes made converting Peter's Date Package v2 to Peter's Date and Time

Changes made since v2. If you want a document with changes made since v1.1, click [here](#).

Click on any of these topics to jump to them:

- ◆ [How DES features affected the PDP conversion](#)
- ◆ [How PDP features enhanced DES](#)
- ◆ [Breaking Changes](#)
- ◆ [DateTextBox, AnniversaryTextBox, and MonthYearTextBox](#)
- ◆ [TimeOfDayTextBox and DurationTextBox](#)
- ◆ [Calendar and MultiSelectionCalendar](#)
- ◆ [PopupCalendar](#)
- ◆ [MonthYearPicker](#)
- ◆ [TimePicker](#)
- ◆ [PopupTimePicker](#)
- ◆ [SpecialDates](#)
- ◆ [QuickDateMenu](#)
- ◆ [ContextMenu](#)
- ◆ [Validators](#)

Peter's Date and Time module (PDT) of Peter's Data Entry Suite (DES) is a reworking of the Peter's Date Package (PDP) code base to conform with DES. DES brings many enhancements to PDP. PDP also brings enhancements to DES.

How DES features affected the PDP conversion

Here are some of the ways DES features were applied to the PDP code base.

- PDP's textbox controls now inherit from the `PeterBlum.DES.TextBox`. As a result, they inherit its extensive properties including the often requested `ValueWhenBlank` property.
- PDP had some validation-like features on its textbox controls such as setting focus and changing the color when there is an error. These capabilities are found in the DES Validation Framework. As a result, they are obsolete on the individual controls.
- Properties have been renamed to omit the leading "x" character and trailing "B" character (example: `xIsValidB` → `IsValid`). Most of the old properties are hidden but still work. In addition, the **Web Application Updater** renames most of them for you.
- PDP established `CultureInfo` settings on individual controls in their `xDateTimeFormatInfo` property. DES uses a single `CultureInfo` object on the `PeterBlum.DES.Globals.Page.CultureInfo` property. With a single property, setup is done in one place. So PDT makes the `xDateTimeFormatInfo` property obsolete and gets its formatting from the `PeterBlum.DES.Globals.Page.CultureInfo` property.
- PDT controls now support the String Lookup System, allowing for a very flexible way to localize controls.
- DES's `AJAXManager` class was written based on PDP's, and included many design improvements that are now being used by PDT.
- These controls shut down client-side code support when DES detects the browser isn't running JavaScript.

How PDP features enhanced DES

- PDP has an extensive toolkit for popups. Now that its part of DES, DES can start using popups. In this release, the PopupErrorFormatter, Popup Hint, and Popup ToolTip take advantage of it.
- PDP v2 had a much better way to manage multiple style sheets, where there are separate style sheets for different controls. This allowed loading smaller style sheet files based on what is added to the page. DES has split its style sheet file into several, based on the control type.
- PDP v2 introduced the Expanded Properties Editor. This tool is now available throughout DES controls in design mode.
- PDP v2 uses the token “{APPEARANCE}” in any URL to identify the path to the Appearance folder. DES has applied this to its properties.
- The ContextMenu control is now available as part of DES in the **Peter’s Interactive Pages** module.

ALERT: *Peter’s Date and Time controls do not provide a context menu or help button if no license covers Peter’s Interactive Pages.*

Breaking Changes

As a result of these changes, the DES implementation of the PDP controls often differs enough that there are breaking changes. The **Installation Guide** provides a migration path for existing PDP controls that covers these breaking changes. Yet, it's important to know that you do not need to convert a PDP control to its DES equivalent unless you plan to take advantage of a feature used by DES. DES and PDP can work side-by-side, whether the assemblies, scripts, or the controls on a page.

Click on any of these topics to jump to them:

- ◆ [Impact of refactoring the DES v4 code base](#)
- ◆ [Property and Class name changes](#)
- ◆ [Style sheets](#)
- ◆ [Compatibility with your existing scripts](#)

Impact of refactoring the DES v4 code base

See "[Breaking changes when upgrading from ANY version](#)". **This is very important.** If you encounter issues converting your custom classes based on PDP controls, you can elect to upgrade to DES v4 which doesn't have these issues.

Property and Class name changes

Nearly every property has been renamed to remove the leading "x" character and the trailing "B" character (on Booleans). Many property names were changed in other ways, especially to conform with both DES and ASP.NET standards. There is several class names changed too:

CS_Calendar → Calendar

CS_Menu → ContextMenu

Fortunately, the migration process helps you quickly deal with these changes. The **Web Application Updater** renames most of the names and flags those as obsolete with no remaining support. The actual controls still have most of the properties in their original name, although it's hidden so you don't risk using it in the future. Yet, it will still compile without being renamed.

See the "Migrating from Peter's Date Package" section of the **Installation Guide** for details.

Obsolete or redesigned features

- ContextMenus are now part of the **Peter's Interactive Pages** module. As a result, these controls do not setup their own ContextMenus or Help Buttons if you do not have a license covering Peter's Interactive Pages. With such a license present, these features are active. Whether or not you have a license, any properties that support the ContextMenu and Help Button will still accept values, to limit migration errors.

Exception: The QuickDateMenu internally uses a ContextMenu. Because it's the central feature of this control, the QuickDateMenu works if you don't have a license covering Peter's Interactive Pages.

- Textboxes: the AutoToolTip feature has been redesigned. If you customized it previously, you will need use new properties: **AutoToolTipEntryPattern**, **AutoToolTipCommandPattern**, and **AutoToolTipCommandHeader**.

It now uses a PopupView by default instead of a tooltip. You can edit this as it's built upon the Interactive Hints system which allows you to switch back to using a tooltip, or have it show up on the page in a Label. To restore it to a hint:

- **SharedHintFormatterName** = ""
- **LocalHintFormatter.DisplayMode** = None
- **LocalHintFormatter.InToolTip** = true
- Textboxes no longer provide these Error Handling properties:
 - **xInvalidDateMsg** and **xInvalidTimeMsg** – Use a DataTypeCheckValidator (or PeterBlum.DES.NativeValidators.CompareValidator when using the native validation framework). In addition, the

validator no longer gets its error message from **xInvalidDateMsg**. So be sure to assign the error message directly to the validator.

- **xOutOfRangeDateMsg** and **xOutOfRangeTimeMsg** – Use a RangeValidator (or PeterBlum.DES.NativeValidators.RangeValidator when using the native validation framework). In addition, the validator no longer gets its error message from **xOutOfRangeDateMsg**. So be sure to assign the error message directly to the validator.
- **xErrorAlertOnChangeB** – When using the DES Validation Framework, set the property **PeterBlum.DES.Globals.Page.ShowAlertOnChange** to true in Page_Load() or **ShowAlertOnChange** in the PageManager control. There is no equivalent in the Native Validation Framework.
- **xFocusOnErrorB** – When using the DES Validation Framework, set the property **PeterBlum.DES.Globals.Page.FocusOnChange** to true in Page_Load() or **FocusOnChange** in the PageManager control. In the Native Validation Framework, use the **SetFocusOnError** property found on the CompareValidator.
- **xErrorForeColor** and **xErrorBackColor** – When using the DES Validation Framework, change the style sheet DESVALFieldWithError in the **DES\Appearance\Validation\Validation.css** style sheet file to provide the equivalent appearance. There is no equivalent in the Native Validation Framework.
- **xShowErrorOnPopupB** – No longer used. The Popup Calendar/MonthYearPicker/TimePicker will always appear if there is an error.
- **xDateTimeFormatInfo** property has been deprecated. The page now uses a single CultureInfo object to describe the date and time format. It is in **PeterBlum.DES.Globals.Page**. If you have code that assigns **xDateTimeFormatInfo**, you may want to rework it to assign to **PeterBlum.DES.Globals.Page.CultureInfo.DateTimeFormat**. If it is not changed, your code will be updating that global.
- On the Calendar, Style Sheet management of Day Cells has been completely redesigned. It now uses the class “merging” feature found elsewhere in DES.

DayCssClass, **OtherMonthDayCssClass** and the style sheet class from the SpecialDates control serve as the “base” class. They should reflect all style sheet attributes that appear on a date.

The properties **TodayCssClass** (for Today’s date), **SpecialCssClass** (for the SpecialDate), and **SelectedCssClass** (for the selected date) will merge with the base. Setup their style sheets with only the differences introduced, such as a new background color. If you have created alternative style sheet classes for any of these properties, add a lead + character to the value of the property, such as **SelectedCssClass** = “+SelectedDate”.

The property **xSelectedTodayCssClass** is no longer used because of the merging feature.

- MultiSelectionCalendar’s highlight feature has been redesigned. Style sheets are used instead of the color properties to give far more flexibility on what the highlight should look like. The **HiliteRangeSelectColor** and **HiliteRangeUnselectColor** properties are obsolete. Edit the following style sheet classes in **DES\Appearance\Date and Time\MultiSelectionCalendar.css** to apply the colors from **HiliteRangeSelectColor** and **HiliteRangeUnselectColor**: DES_MSCHiliteToSelect and DES_MSCHiliteToUnselect.
- ContextMenu uses style sheets instead of properties that define color. Edit the style sheet classes in **DES\Appearance\Date and Time\ContextMenu.css** to apply the colors you previously assigned to properties.
- QuickDateMenu uses style sheets instead of properties that define color. Edit the style sheet classes in **DES\Appearance\Date and Time\ContextMenu.css** to apply the colors you previously assigned to properties.
- The QuickDateMenu supports the SharedGroup popup feature, allowing multiple instances of the QuickDateMenu to share a common popup ContextMenu. Since this feature is on by default, if you have multiple QuickDateMenus that have a different list of items, change the value of the **SharedGroup** property to a unique name for each list.

Style sheets

There are numerous differences between the **Peter's Date Package** and DES style sheet files. Here are some examples:

- The Calendar, MultiSelectionCalendar, MonthYearPicker, and TimePicker have a fresh appearance which does not appear after conversion. To review it, look here: <http://www.peterblum.com/des/dateandtime.aspx>.

To switch to this new appearance, go to the <appSettings> section of the web.config file and remove this line:

```
<add value="DES_DES4Appearance" value="" />
```

- Most properties that specified colors have been replaced by style sheet classes.
- Buttons now offer mouse over effects which require new style sheet classes. The style sheet name must have a specific syntax: use the name of the style sheet class for the normal appearance + "MouseOver".
- Properties that specified the style sheet class used when the mouse is pressed are no longer supported. Instead, the style sheet name must have a specific syntax: use the name of the style sheet class for the normal appearance + "Pressed".
- Borders on the Calendar's day cells get their color from in style sheets. Previously they were handled programmatically. You **must** introduce the border colors (ex: border-left-color: gray) to the style sheets to have borders. Code still generates the presence of a line by defining the border line style and width.
- Mouseovers on the Calendar day cells, the MonthYearPicker month and year cells, and TimePicker time cells are implementing in style sheets.
- Previously the Calendar created the appearance of Date cells by using unique style sheet classes for various situations: normal, selected, today, and specialdate. DES still has separate style sheet classes, but they are designed to be merged. DES will always get the normal style then add the selected or today style as needed. So the style definitions for selected, today, and specialdate have fewer attributes.
- Several colors that were specified by name are now specified by their RGB color to accommodate browsers that lacked support for the color name.

The Web Application Updater program will migrate your Peter's Date Package style sheet files into the **[web app]\DES\Appearance\Date and Time** folder. Migration renames the files and the classes inside them. It also adds, edits, and removes styles and classes to DES 5 levels.

Compatibility with your existing scripts

The names of every JavaScript function declared in **Peter's Date Package** have changed. In some cases, the term "PDP" is replaced by "DES" such as `PDP_GetById()` is now `DES_GetById()`. In other cases, a more complex renaming occurred. There are some functions that take different parameters.

Fortunately, DES includes a script file that maps the old names to the new ones. It is added automatically when you upgrade. The **Web Application Updater** program adds this line to <appSettings> of **web.config** to provide this service. If you don't need the converted scripts, you can remove it:

```
<add key="DES_PDPCCompatibleScriptFile" value="" />
```


DES provides the source script file **PDPCCompatible.js** in the **[DES product folder]\Upgrading** folder. Use it if you are getting JavaScript errors to see if a function you are using is still supported. If not, please allow Tech Support assist you (support@peterblum.com).

General Changes

- String Lookup System is supported. New properties for each string-type property support the LookupID. Most refer to the String Group "DateTime". When using resources, enter your strings in the file **DESDateTime.resx**. Those for hints and tooltips refer to the String Group "Hints" (**DESHints.resx** resource file).
- When using Microsoft ASP.NET AJAX, DES is smarter about outputting scripts. You no longer need to pass the ScriptManager or UpdatePanel control to the `AJAXManager.UsingMicrosoftAJAX()` manager method (although those parameters are still supported). In fact, you don't have to call that function if you are using the new PageManager control, because you can set its `AJAXFramework` property to `MicrosoftAJAX`. See the "Using these controls with AJAX" section of the **General Features Guide**.

By default, DES will set the **InAJAXUpdate** properties when its controls are found in UpdatePanels, RadAJAXPanels, MagicAJAX's AJAXPanel, and RadAJAXManager (although that requires setup). It also sets **InAJAXUpdate** on DES controls outside of these controls in many cases, although you will have to set it in some situations.

While its much simpler to setup, you still need to use the "preregister for AJAX" features when a type of control is not on the page until a callback creates it.

- Most buttons now provide mouseover effects. Supported by both images and style sheets. Previously you had images for normal and pressed states. Now it adds a mouse over effect. New images and style sheet classes predefine this state.
- The URL to images and PDT style sheets is now **\DES\Appearance\Date and Time** except for help, close, and arrows which are shared amongst modules. They are in **\DES\Appearance\Shared**.
- Does not generate client-side code support when DES detects the browser isn't running JavaScript. Gracefully scales down to a server-side only control.
- Extensive support of the SmartTag () , a feature of Visual Studio 2005 and Visual Web Developer's design mode. Controls provide the most common properties for quick setup.
- Where a control references another through a property that takes an ID, there is a second property that takes a reference to the control. This allows the other control to be in another naming container.
- New debugging reports that output runtime information for licenses, ajax setup, global settings, page-level settings, validation and style sheets. Once setup, you add a querystring parameter to the page's URL that invokes the reports. See the "Exploring the Current Settings" section of the **General Features Guide**.

DateTextBox, AnniversaryTextBox, and MonthYearTextBox

ALERT: Peter's Date and Time controls do not provide a context menu or help button if no license covers Peter's Interactive Pages.

- See also in General Changes.
- On the Calendar and MonthYearPicker, the **ClientSideCreatesHTML** property lets you determine if you want most of the HTML to be created through JavaScript or on the server. This will reduce the size of the page dramatically but take more time running the page initialization code. When used as a popup, you can delay creating the HTML until the first popup is requested.
- **EnableKeystrokeCommands** property lets you turn off the keystroke commands. When `false`, the context menu and help button do not show the keystroke commands either.
- The Automatic ToolTip feature has been redesigned.
 - Uses the DES Interactive Hints system. By default, it will show a PopupView. You can edit the **LocalHintFormatter** or **SharedHintFormatterName** to use a label on the page or just use a tooltip like it was in the past. When using **LocalHintFormatter**, set **SharedHintFormatterName** to "".
 - The message is assembled from three properties instead of two. **AutoHintEntryPattern** shows the date format. **AutoHintCommandPattern** is used to show an individual keystroke command. **AutoHintCommandHeader** separates the date format from the commands. See "[Obsolete or redesigned features](#)".
- **AutoValueWhenBlank** property fills in the **ValueWhenBlank** property with a pattern that assists the user in entry. It's a similar pattern to the **AutoHint**. Short date will always use DD, MM, and YYYY. Long date will always be capitalized.

Other Property Name Changes

Conversion note: The Web Application Updater changes most of these names for you. The control also supports the original property name, internally mapping its value to the new property name.

- `xShowPopupB` → `ShowPopupCalendar` or `ShowPopupMonthYearPicker`
- `xAutoToolTipB` → `AutoHint`
- `xAutoTransferToCalendarID` → `ConnectToCalendarControlID`
- `xAutoTransferToMonthYearPickerID` → `ConnectToMonthYearPickerControlID`

TimeOfDayTextBox and DurationTextBox

ALERT: Peter's Date and Time controls do not provide a context menu or help button if no license covers Peter's Interactive Pages.

- See also in General Changes.
- On the TimePicker, the **ClientSideCreatesHTML** property lets you determine if you want most of the HTML to be created through JavaScript or on the server. This will reduce the size of the page dramatically but take more time running the page initialization code. When used as a popup, you can delay creating the HTML until the first popup is requested.
- The Automatic ToolTip feature has been redesigned.
 - Uses the DES Interactive Hints system. By default, it will show a PopupView. You can edit the **LocalHintFormatter** or **SharedHintFormatterName** to use a label on the page or just use a tooltip like it was in the past. When using **LocalHintFormatter**, set **SharedHintFormatterName** to "".
 - The message is assembled from three properties instead of two. **AutoHintEntryPattern** shows the time format. **AutoHintCommandPattern** is used to show an individual keystroke command. **AutoHintCommandHeader** separates the date format from the commands. See "[Obsolete or redesigned features](#)".

Other Property Name Changes

Conversion note: The Web Application Updater changes most of these names for you. The control also supports the original property name, internally mapping its value to the new property name.

- xShowTimePickerB → ShowPopupTimePicker
- xShowCommandButtons → ShowSpinners
- xAutoToolTipB → AutoHint

Calendar and MultiSelectionCalendar

ALERT: Peter's Date and Time controls do not provide a context menu or help button if no license covers Peter's Interactive Pages.

- See also in General Changes.
- The **ClientSideCreatesHTML** property lets you determine if you want most of the HTML to be created through JavaScript or on the server. This will reduce the size of the page dramatically but take more time running the page initialization code. When used as a popup, you can delay creating the HTML until the first popup is requested.
- Style Sheet management of Date Cells has been completely redesigned. It merges classes together instead of forcing you to define complete styles for each case.

DayCssClass, **OtherMonthDayCssClass** and the style sheet class from the SpecialDates control serve as the "base" style sheet class. Their classes should reflect all style sheet attributes that normally appear on a date.

They are merged with the style sheet classes for Special date, Today, and selected, when those apply to the Date Cell.

Merging goes in this order: [Base] [Special] [Today] [Selected]

Weekend coloring will still use the original rules where you specify a color in the **WeekendBackground** property.

- MultiSelectionCalendar's highlight feature has been redesigned. Style sheets are used instead of color properties to give far more flexibility on what the highlight should look like.
- Supports keyboard movement of the selected date and changing months on FireFox 1.5. (Previously existed on IE and when in a PopupCalendar, in FireFox too)
- MultiSelectionCalendar by default has always supported setting a range when the shift key is down. Set **ShiftKeyForRange** to `false` to ignore the shift key.

Other Property Name Changes

Conversion note: The Web Application Updater changes most of these names for you. The control also supports the original property name, internally mapping its value to the new property name.

- `xFooterCssClass` → `CommandAreaCssClass`
- `xFooterButtonCellWidth` → `CommandButtonCellWidth`
- `xFooterButtonCssClass` → `CommandButtonCssClass`
- `xFooterButtonInset` → `CommandButtonInset`
- `xFooterRowHeight` → `CommandAreaRowHeight`
- `xFooterVerticalAlign` → `CommandButtonVerticalAlign`
- `xControlToUpdateID` → `ShowSelectedDateControlID`
- `xControlToUpdateDateFormat` → `ShowSelectedDateControlDateFormat`
- `OnDateChanged` → `SelectedDateChanged`
- `OnMonthViewChanged` → `MonthViewChanged`

PopupCalendar

- No additional changes beyond those listed in [General Changes](#).

MonthYearPicker

- See also in [General Changes](#).
- The **ClientSideCreatesHTML** property lets you determine if you want most of the HTML to be created through javascript or on the server. This will reduce the size of the page dramatically but take more time running the page initialization code. When used as a popup, you can delay creating the HTML until the first popup is requested.
- **HideRowsOutOfRange** property hides rows that have entirely out-of-range dates.
- **FooterCssClass** property provides a style sheet class for the footer area on the Few Years format. It defaults to "DES_FYPPFooter".

Other Property Name Changes

Conversion note: The Web Application Updater changes most of these names for you. The control also supports the original property name, internally mapping its value to the new property name.

- xDateTime → MonthYearAsDateTime
- xDateNullable → MonthYearAsDateTimeNullable
- xDateTimeBindable → MonthYearAsDateTimeBindable
- xDateTimeBindableMode → MonthYearBindableMode
- OnDateChanged (event) → SelectedDateChanged

PopupMonthYearPicker

- No additional changes beyond those listed in [General Changes](#).

TimePicker

- See also in [General Changes](#).
- The **ClientSideCreatesHTML** property lets you determine if you want most of the HTML to be created through javascript or on the server. This will reduce the size of the page dramatically but take more time running the page initialization code. When used as a popup, you can delay creating the HTML until the first popup is requested.

Other Property Name Changes

Conversion note: The Web Application Updater changes most of these names for you. The control also supports the original property name, internally mapping its value to the new property name.

- TimeSpanCssClass -> TimeValueCssClass
- SelectedTimeSpanCssClass -> SelectedTimeValueCssClass

PopupTimePicker

- No additional changes beyond those listed in [General Changes](#).

SpecialDates

- See also in General Changes.
- The **DayOfWeekConflictRule** property determines what happens when both a SpecialDayOfWeek and any other SpecialDate item share the same date. One of the two must be selected based on this rule.

QuickDateMenu

- See also in General Changes.
- Style sheets have replaced many properties that set the font and background colors.
- Supports the SharedGroup popup feature, allowing multiple instances of the QuickDateMenu to share a common popup ContextMenu.

Warning: Since this feature is on by default, if you have multiple QuickDateMenus that have a different list of items, change the value of **SharedGroup** property to a unique name for each list.

ContextMenu

ALERT: Peter's Date and Time controls do not provide a context menu or help button if no license covers Peter's Interactive Pages.

The stand-alone ContextMenu has been moved from the **Peter's Date and Time** module to the **Peter's Interactive Pages** module.

- The **ClientSideCreatesHTML** property lets you determine if you want most of the HTML to be created through javascript or on the server. This will reduce the size of the page dramatically but take more time running the page initialization code. When used as a popup, you can delay creating the HTML until the first popup is requested.
- Style sheets have replaced many properties that set the font and background colors.
- When using the stand-alone ContextMenu, it now supports real keyboard equivalents. You can define the desired keystroke and the menu command will be invoked if the user types that keystroke.
- MenuCommand objects are much more powerful. They have a number of built in script actions so you have less need to write scripts. Here are the new properties:
 - **CausesValidation** - When true (defaults to false), it validates before running any scripts. Does not invoke scripts if validation fails.
 - **ValidationGroup** - Validation group name used when validating.
 - **PostBack** - When true, it posts back. A new postback event handler **MenuSelected** is called for these menu commands.
 - **NavigateUrl** - When defined, goes to this URL.
 - **PostBackTracksFocus** - ASP.NET 2 only. When true and PostBack is used, it uses the TrackFocus feature of ASP.NET 2.
- When the **Height** property is assigned, scrollbars will appear if needed. This will let your menu contents be longer than what is shown.

Validators

- When using the Native Validation Framework (as opposed to the DES Validation Framework), use the validators from the **PeterBlum.DES.NativeValidators.dll** assembly. It includes versions of the CompareValidator and RangeValidator that support the date and time formats of PDT. It includes a DifferenceValidator, UnselectableDatesValidator and UnselectableTimeValidator.
- When using the DES Validation Framework, the standard DES Validators, such as DataTypeCheckValidator and CompareToValueValidator, work automatically with these controls.
- The Calendar control supports validators with the **DataType** property. This is often used with the CompareTwoFieldsValidator to set up a start and end date range using calendars.

Table Of Contents

BREAKING CHANGES WHEN UPGRADING FROM ANY VERSION.....	2
Major breaking changes.....	2
Easy to fix breaking changes.....	3
UPGRADING FROM V4.0.....	5
UPGRADING FROM PRE 4.0 PRODUCTS: CHANGES TO INSTALLATION	8
PROFESSIONAL VALIDATION AND MORE USERS	9
Breaking Changes.....	10
Impact of refactoring the DES v4 code base.....	10
Obsolete Properties	10
Style sheets	10
Compatibility with your existing scripts.....	10
New module: Peter's Date and Time.....	11
Design Mode.....	12
General Features.....	13
PageManager Control	14
NativeControlExtender Control.....	15
PeterBlum.DES.Globals.Page.....	15
Peter's Professional Validation and Peter's More Validators modules	16
New Validators	16
All Validators.....	16
DataTypeCheckValidator	18
RangeValidator	19
CompareToValueValidator.....	19
EmailAddressValidator.....	19
CreditCardNumberValidator	19
ValidationSummary	19
DES Buttons	20
Peter's TextBox module	21
New Controls.....	21
All TextBoxes.....	21
All Numeric TextBoxes: Integer, Decimal, Currency, Percent.....	21
IntegerTextBox	22
DecimalTextBox	22
CurrencyTextBox	22
MultiSegmentDataEntry	23
Validation Support.....	23
Peter's Interactive Pages module	24
New Controls and Tools	24
FieldStateControllers	24
CalculationController	25
Interactive Hints.....	25
Enhanced Buttons	26

Peter's Input Security module 27

CHANGES MADE CONVERTING PETER'S DATE PACKAGE V2 TO PETER'S DATE AND TIME 28

How DES features affected the PDP conversion 28

How PDP features enhanced DES 29

Breaking Changes 30

- Impact of refactoring the DES v4 code base 30
- Property and Class name changes 30
- Obsolete or redesigned features 30
- Style sheets 32
- Compatibility with your existing scripts 32

General Changes 33

- DateTextBox, AnniversaryTextBox, and MonthYearTextBox 34
- TimeOfDayTextBox and DurationTextBox 35
- Calendar and MultiSelectionCalendar 36
- PopupCalendar 36
- MonthYearPicker 37
- PopupMonthYearPicker 37
- TimePicker 38
- PopupTimePicker 38
- SpecialDates 39
- QuickDateMenu 39
- ContextMenu 39
- Validators 40