Intersoft Solutions
A better web experience™

Intersoft WebUI Studio.NET 2006 R2 boost web application's usability and end user's experience for maximum productivity.

Case Study	: Migrating Intersoft Developer Network to WebUI Studio.NET 2006 R2
Application	: Intersoft Developer Network 2.0
Type	: Web-based Customer Portal (RIA) application.
Division	: IT Department, Intersoft Solutions Corp.

Technologies :

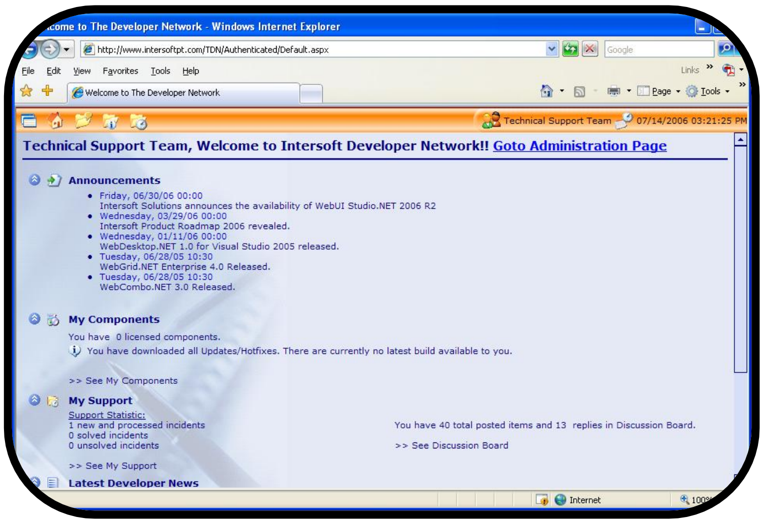
Microsoft ASP.NET 2.0, Microsoft Visual Studio 2005 Team Suite
Microsoft Internet Information Services 6.0, Microsoft SQL Server 2005
Intersoft WebUI.NET Framework 3.0
Intersoft WebUI Studio.NET 2006 R2

Objectives :

Improve overall user experience that simulates desktop application's user interfaces.
Rich user interfaces with windowing navigation system so that users can perform multi-tasking within the web application.
Sophisticated Outlook 2003® style layout, with ability to resize the panes, or collapse and expanding the panes for maximum readability and better usage of screen real estate.
The application's layout must be configurable by users. Some users loved to use modern Outlook 2003® reading model, however some users preferred traditional Outlook Express® reading model. So our application architecture must be flexible and extensible enough in that area.
The application's theme and style must be automatically applied to the entire web application globally so that all UI elements have consistent look and feel. The theme needs to be customizable by end users as well, without the needs to change application's codes or designs.
The application must be lightening fast with output size as small as possible.
Implement desktop-style UI design methodology such as using Tab for better information grouping, context menu in every Grid, and drag and drop capabilities.
Offer new kind of services and applications that boost user's productivity by allowing sophisticated user interactions that are very difficult to achieve in Web.

Table of Contents

A brief overview of Intersoft Developer Network 1.0



lcome to The Developer Network - Windows Internet Explorer
http://www.intersoftpt.com/TDN/Authenticated/Default.aspx
File Edit View Favorites Tools Help
Welcome to The Developer Network
Technical Support Team 07/14/2006 03:21:25 PM

Technical Support Team, Welcome to Intersoft Developer Network!! Goto Administration Page

Announcements

- Friday, 06/30/06 00:00
  Intersoft Solutions announces the availability of WebUI Studio.NET 2006 R2
- Wednesday, 03/29/06 00:00
  Intersoft Product Roadmap 2006 revealed.
- Wednesday, 01/11/06 00:00
  WebDesktop.NET 1.0 for Visual Studio 2005 released.
- Tuesday, 06/28/05 10:30
  WebGrid.NET Enterprise 4.0 Released.
- Tuesday, 06/28/05 10:30
  WebCombo.NET 3.0 Released.

My Components

You have 0 licensed components.
You have downloaded all Updates/Hotfixes. There are currently no latest build available to you.

>> See My Components

My Support

Support Statistic:
1 new and processed incidents
0 solved incidents
0 unsolved incidents

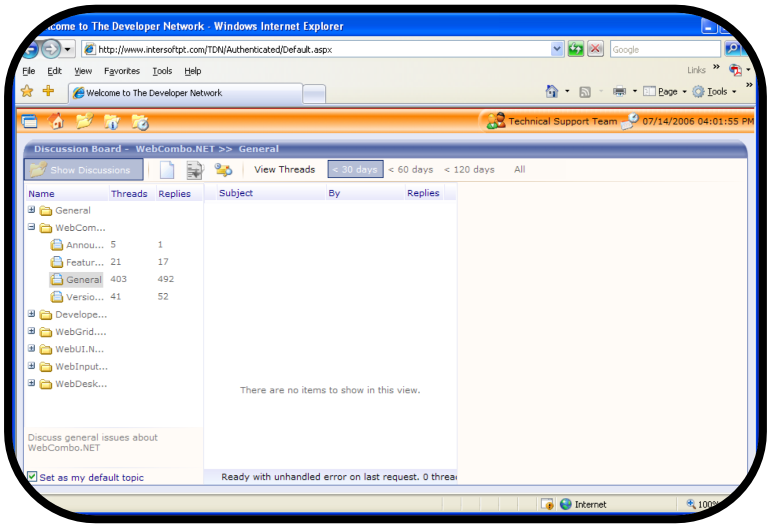
>> See My Support

You have 40 total posted items and 13 replies in Discussion Board.

>> See Discussion Board

Latest Developer News

Figure 1. The old fashioned, pre-WebDesktop era web application.



lcome to The Developer Network - Windows Internet Explorer
http://www.intersoftpt.com/TDN/Authenticated/Default.aspx
File Edit View Favorites Tools Help
Welcome to The Developer Network
Technical Support Team 07/14/2006 04:01:55 PM
Show Discussions
View Threads
< 30 days
< 60 days
< 120 days
All

Discussion Board - WebCombo.NET >> General

Name	Threads	Replies
General		
WebCom...		
Annou...	5	1
Featur...	21	17
General	403	492
Versio...	41	52
Develope...		
WebGrid....		
WebUI.N...		
WebInput...		
WebDesk...		

Subject	By	Replies

There are no items to show in this view.

Discuss general issues about WebCombo.NET

Set as my default topic

Ready with unhandled error on last request. 0 threa

Figure 2. Like most traditional web user interfaces, the user interactions are limited to single static page layout.

In the next several chapters, we will discuss in-depth on how to transform traditional web application model into exciting, rich desktop-style web application with pixel-perfect visual appearance and smooth user interactions through AJAX-powered components.

I. Improve overall web user experience that replicate the richness of desktop-style application.

Our first and the most important goal while prototyping *Intersoft Developer Network 2.0* (further called IDN2) is to significantly improve the overall web application's user experience so that users can interact with the application in better and productive manner. We were planning on a total revamp in the user interface bits, as well as architecture redesign which makes use of ASP.NET 2.0 features extensively.

Having fresh WebDesktop.NET Alpha bits from our component developer division at initial prototyping, we have had chance to review the overall functionalities provided by the new UI components. After several times of discussions, we are confident that building desktop-style model with rich user interaction is possible to be implemented in Developer Network 2.0.

The first effort is setting up User Interface guidelines and core architecture that will drive the entire User Interface actions such as shown in the following.

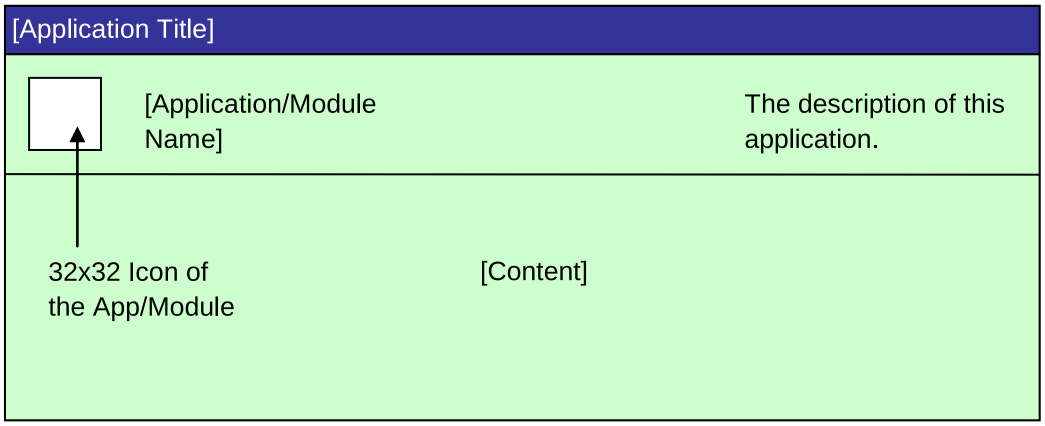

[Application Title]
[Application/Module Name]
The description of this application.
32x32 Icon of the App/Module
[Content]

Figure 3. The User Interface design for an application module which consists of Window element and partitioned layout.

We also setup the overall features for the User Interface guidelines, such as what features should be enabled on certain usage scenarios, the moving and resizing capabilities, drag and drop, window's size and start up position and more.

II. Rich User Interfaces with Windowing Navigation System.

After doing some mini prototyping on the first objective, the web application team has ensured that overall web user experience can be significantly improved through the use of UI components available in WebDesktop.NET.

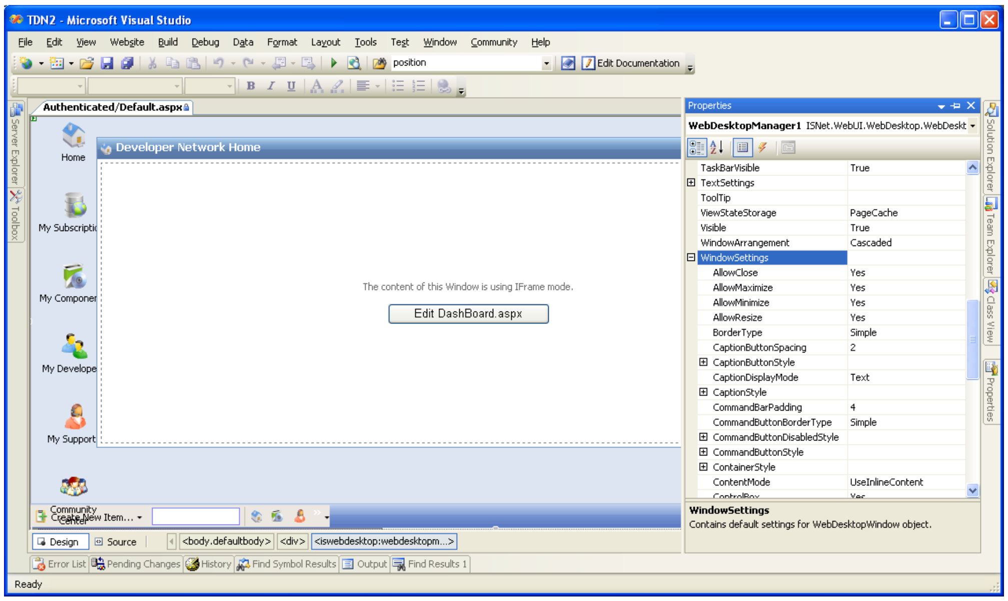
One of a goal that we would like to achieve since the past years is the capability to have windowing navigation system inside web application. This has been Intersoft's vision and mission to deliver better web user experiences.

With the desktop-style windowing navigation system, our end users enjoy benefits such as:

- Spacious and uncluttered working spaces.
- Multi tasking through multiple windows which accessible at TaskBar.
- Easy launch of applications through shortcut icons.
- Familiarity with desktop experience which increasing productivity, such as minimizing window, resizing or moving window or arranging all windows in cascading style or horizontally.

To achieve this objective, we picked *WebDesktopManager* component and utilize it to serve as universal shell that host the main User Interface and Navigation System for the web application.

In this chapter, we delve down into the development environment to configure *WebDesktopManager* settings.



Properties

WebDesktopManager1 ISNet.WebUI.WebDesktop.WebDeskt

TaskBarVisible	True
⊞ TextSettings	
ToolTip	
ViewStateStorage	PageCache
Visible	True
WindowArrangement	Cascaded
⊟ WindowSettings	
AllowClose	Yes
AllowMaximize	Yes
AllowMinimize	Yes
AllowResize	Yes
BorderType	Simple
CaptionButtonSpacing	2
⊞ CaptionButtonStyle	
CaptionDisplayMode	Text
⊞ CaptionStyle	
CommandBarPadding	4
CommandButtonBorderType	Simple
⊞ CommandButtonDisabledStyle	
⊞ CommandButtonStyle	
⊞ ContainerStyle	
ContentMode	UseInlineContent
ControlBox	Yes

WindowSettings
Contains default settings for WebDesktopWindow object.

Figure 4. Rich design-time experience helped us to easily configure Shortcut Icons and Windows.

Most configurations of the *WebDesktopManager* control can be easily configured through properties change in either Visual Studio's Property Window or WebDesktopManager's Designer.

The WebDesktopManager control allows us to quickly configure settings like:

The visibility of shortcut icons.
The window's animation.
The context menu, which applies to desktop element and window element.
Keyboard navigation. For instance, press Ctrl+Alt+C will close the currently active window.
The task bar visibility as well as numerous task bar related settings.
The window arrangement.
The window's features, such as the default window location, size, or the ability to minimize, maximize or close.
The window's rendering mode, which supports basic rectangular rendering and complex-images rendering.
The window's content mode supporting template or IFRAME.
More than 10 customizable styles through style object model.

It's also worthy to note that the *WebDesktopManager* control is capable to handle all runtime interactions automatically without additional configuration. For instance, when many windows are opened in the Desktop, the associated button in the taskbar will be automatically resized to fit the width of the screen. Otherwise, the associated button will have its default width (ie. 150px). Most of the behaviors in the control have been architected to follow the usability standard of desktop's behaviors.

After finishing the shortcut icons and windows configuration, we started to improve the usability of the taskbar. One of the favorite ideas is to put some commands in the taskbar area so that end users can have quick access to applications and common tasks. We also added a WebInput in the taskbar to allow users to search from the taskbar.

Thanks to the flexibility offered by the *WebDesktopManager* control, we can put custom content in the left and right side relative to the main window buttons area. The following screenshot shows the WebToolBar instance which contains commands.

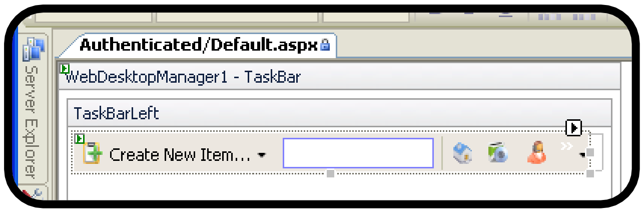


Figure 5. WebToolBar control inside WebDesktopManager's TaskBarLeft template.

III. Sophisticated Outlook 2003® style layout.

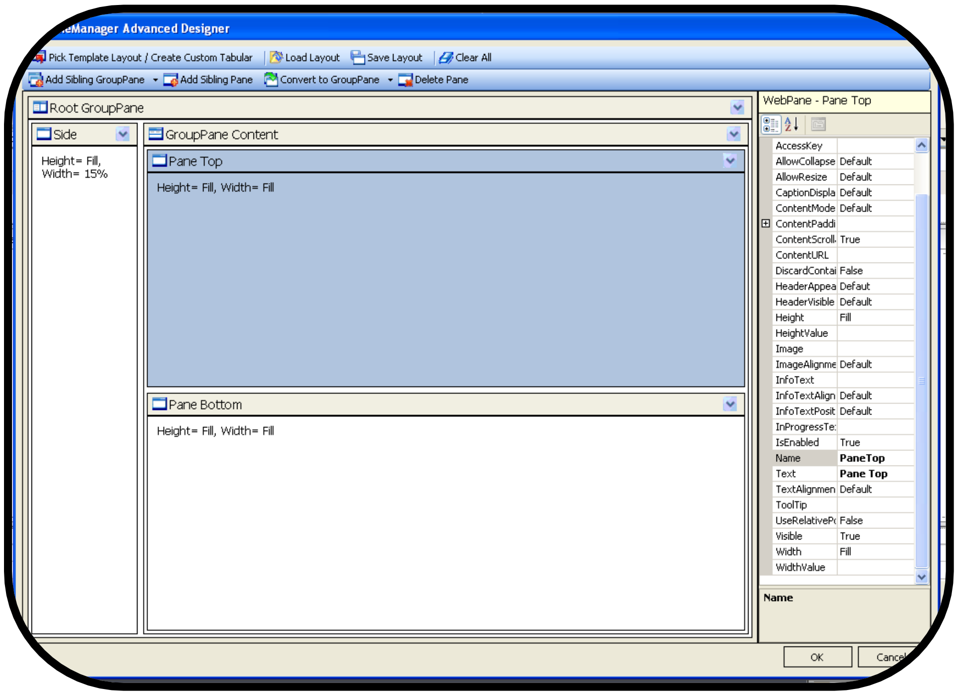
At this phase, we have settled most User Interface and Navigation System configurations which implemented through *WebDesktopManager* component.

In the third objective of our development list, we would like to have Outlook 2003® style Layout in the applications hosted inside IDN2. This objective is a “must achieved” because our end users have been demanding a Layout model which is resizable using drag and drop, and collapseable so they can minimize unused Panel to get maximized view of certain Panel. This enables them to work with the documents and information in more productive manner.

The flexible Layout model is one of the most difficult challenges in Web-based application development because designers have to carefully divide the tables into appropriate cells with correct spanning configuration. And not to mention the drag and drop stuff for resizing feature or future extensibility – which makes this feature almost impossible to be achieved.

With the availaibility of ***WebPaneManager*** in the WebDesktop.NET product family, the people at web development division are truly relieved. The *WebPaneManager component* has helped to ease the development and cutting required efforts and codes to 80 percent.

How is this possible? Let us show how you can create Outlook 2003® style layout in less than one minute and as simple as 1-2-3.



Height= Fill, Width= 15%

Height= Fill, Width= Fill

Height= Fill, Width= Fill

WebPane - Pane Top

AccessKey	
AllowCollapse	Default
AllowResize	Default
CaptionDispla	Default
ContentMode	Default
ContentPaddi	
ContentScroll	True
ContentURL	
DiscardContai	False
HeaderAppea	Defaut
HeaderVisible	Default
Height	Fill
HeightValue	
Image	
ImageAlignme	Default
InfoText	
InfoTextAlign	Default
InfoTextPosit	Default
InProgressTe:	
IsEnabled	True
Name	**PaneTop**
Text	**Pane Top**
TextAlignmen	Default
ToolTip	
UseRelativePc	False
Visible	True
Width	Fill
WidthValue	

Name

Figure 6. ***WebPaneManager Designer*** lets you easily design the Layout in WYSIWYG manner.

As seen in *Figure 6* above, we can freely design any model of Layout that we would like without has to deal with cumbersome HTML table structures. We even don't need to have deep understanding on the object model of the *WebPaneManager* component, since the designer itself already represents the logical view of the Layout in WYSIWYG manner.

Each individual *WebPane* object allows custom configuration of the resizing and collapsing feature. They are customizable through the *AllowCollapse* and *AllowResize* property which set to "*Default"* by default. The "*Default"* value allows all WebPane objects inside the *WebPaneManager* control to inherit global settings available in *WebPaneSettings*.

The third objective is achieved with the help of *WebPaneManager* component, which lets you easily create complex resizable and collapsible Layout.

IV. Developing User's Customizable Layout

The *WebPaneManager* component includes *advanced Designer* that helping you to configure the Layout at design time. That is obviously good. However, we have another complex requirement where the Layout needs to be customizable by end users. This requirement indicates that we can't configure the Layout at design-time because the Layout's mode needs to be retrieved from database at runtime according to the user's preference.

We could not be so fortunate if the *WebPaneManager* component was not architected with high-level of extensibility and flexibility as one of the design goals. As the matter of fact, the WebPaneManager's *advanced Designer* allows us to save the Layout into XML file. This can be easily done by pressing "Save Layout" command available in the toolbar.

After our designers created all three layouts required by the application, the layouts were saved as XML file and ready to be consumed by the *WebPaneManager* control at runtime.

The following technique shows how we can reuse the saved XML file at runtime.

```
string mode = GetUserReadingMode();
string xmlFile = "ReadingMode/Forum/Mode-" + mode + ".xml";
string src = page.Server.MapPath(xmlFile);

WebPaneManagerForum.LoadPanesStructureFromXml(src);
```

V. Consistent UI look and feel across entire application (application-wide theming)

One of the usability aspect that is lacking in today's web development tool is the ability to provide consistent visual styles and appearances that can be easily applied to the entire web application.

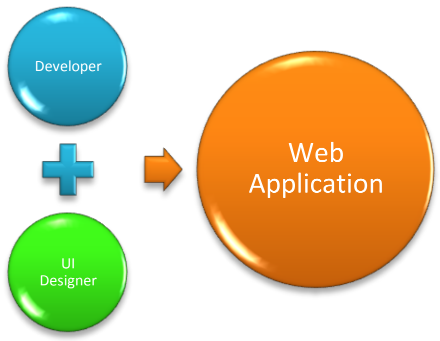
We are grateful to find out that Microsoft's ASP.NET 2.0 includes "theming" feature right out-of-the-box. This is a great feature that every web developers have been waiting. The theming feature allows developers to write the skins for the controls as well as custom external stylesheets. We are just going to delve down into the implementation and soon discover that ASP.NET 2.0's "theming" feature does not work for our scenario.

The main problem is, our application is developed by a team of web developers who don't have deep understanding in User Interface designs such as colors, spacing, or layouting. In our project development, we have divided application development into two roles:

1) *Application Developer Team*. Developer team includes software architects who develop middleware codes and web developers who develop the WebForm pages.
2) *User Interface Designer Team*. Designer team is responsible to define visual styles, create unique look and feel, develop necessary image files and choosing professional colors.

As in most enterprise company, our development methodology is based on agile development management which requires *Application Developer Team* to work robustly on functionalities and business requirements and thus leaving all controls' styles as default. Likewise, the *User Interface Designer Team* should be able to perform their tasks independently outside from the application project, as they need to experiment on visual styles and designs.

The following illustration demonstrate our development methodology:



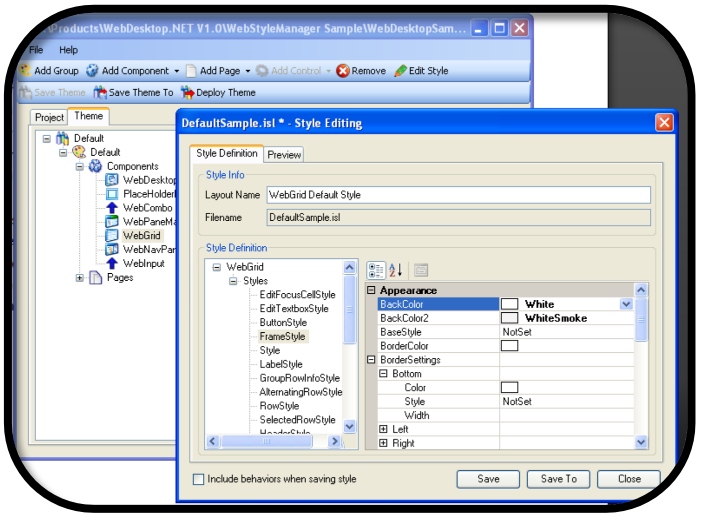
The bottom line is, our *User Interface Designer Team* requires a standalone tool that they can use to produce visual styles independently without the need to get the whole application project files.

This is where Intersoft's *WebUI Style Manager* comes to fill in the gaps.

By using WebUI Style Manager, our User Interface Designer Team can start to work with the visual styles, experimenting colors and images – without the needs to depend on application's project files or to wait Application Developer Team to complete the web pages.

Intersoft WebUI Style Manager introduced several benefits for our web development division:

Solved many limitations and issues related to global theming and roles separation.
Support for agile development methodology where clear roles separation between designer and developer can be enforced.
Saving lots of development time since developers can concentrate on business requirements and functionalities without the distraction for experimenting styles.
Easy to use for designers. Designers simply add the components that want to be themed, experiment with the styles in object model approach (no CSS coding required) and preview the theme in WYSIWYG output.
One click deployment. When the theme design has been finalized, designer simply clicks on "Deploy Theme" button. WebUI Style Manager will automatically copy necessary files and project file into the target web application.
No-touch theme compilation. The designed theme will be compiled on the first request of the web application by Intersoft WebUI.NET Framework. No code changes are required in either web form pages or code-behind.



Layout Name: WebGrid Default Style
Filename: DefaultSample.isl

Appearance	
BackColor	White
BackColor2	WhiteSmoke
BaseStyle	NotSet
BorderColor	
BorderSettings	
Bottom	
Color	
Style	NotSet
Width	
Left	
Right	

Figure 7. WebUI Style Manager lets designer to easily create visual styles.

VI. Page output optimization for high performance application

Along with the completion of first five objectives, we need to review further on the overall application's performance before we go further by implementing more advanced scenarios and richer desktop-style designs.

There are two factors that we are concerning most with rich web application design using server-side control approach:

1) The more controls in a web form, the larger the page output will be. This is due to the client side storage and cascading styles generated by each control.
2) The automatic-generated style's naming are too long specifically when used inside template. This could cause scalability issue since the page output size will be increased dramatically.

The traditional approach to these problems are by using manual stylesheet coding where the styles are placed in external location. The developers would then write the style name manually in each control's style property within all web form pages.

However due to our *agile development methodology*, that traditional approach simply doesn't work. It is because our *User Interface Designer Team* shouldn't depend on the physical web form pages. In simple words, the web form pages purely contain application-specific functions and business requirements – and should not contain visual styles for the future's extensibility, scalability and ease of theme customization.

The WebUI Style Manager once again comes in to solve the first problem. The Style Manager has innovative features named "Enable Output Partition" and "Enable CSS Optimization".

The following is the in-depth explanation on how these features helped in solving the above problems:

1) **Enable Output Partition** automatically takes out common styles from web form into separate external style sheet per control type. This is done without additional codes changes in the application project.

   Therefore by taking out the styles per control type into external style sheet, the HTML output of the web form is spectacularly decreased. This is possible because all controls of same type will refer to the same style name which is now located in the external style sheet.
   This feature alone helps our web application to run twice faster than before, by reducing up to 70 percent of page output size.

2) **Enable CSS Optimization** permits the lengthy generated style name available in the external style sheet to be optimized into 6-7 only characters. When this feature is enabled, the WebUI.NET Framework runtime will automatically recompile the theme and produce optimized style name into the external style sheet.

   With CSS Optimization feature, our second issue concerning the lengthy style name is completely solved.

In addition to two innovative features above, we are also aware of new feature available in our WebUI.NET Framework. The new feature called *Xml Compression* facilitates the capability to compress the *Xml-based Client-side Storage* that Intersoft's components used as foundation for rich object-oriented client-side framework.

The Xml Compression ratio is vary depending on the repetitive-level of the client-side storage content. In average, the Xml Compression feature can help to reduce around 40 percent of the overall page output size.

The following is the only step the we need to do in order to enable Xml Compression feature in the entire web application:

Add the following entry in *web.config* of the designated web application:

```
<add key="ISNet.WebUI.[Product].v[FullVersion].XmlCompressionEnabled" value="true"></add>
```

The maximum utilization of the powerful *WebUI Style Manager* and *Xml Compression* feature altogether enables web application to achieve better scalable web application which break the barrier and limitation in both reusability and performance.

VII. Desktop-style design methodology

Having the six critical objectives achieved is perhaps the most thrilling moment for our web development team division. The reason is very simple: without successful achievement of only one objective that we discussed above, we simply can't go further. It just does not make sense for us to have rich web user interfaces but having performance problem or vice versa. Or having great performance but without scalability support for our agile development methodology is not acceptable at all since it renders the entire team to be unproductive.

In the two last chapters, we will go further by implementing innovations that we had in mind since the first draft of the application's specification.

Before we delve down into the desktop designs topic, you may have raised some questions in your mind such as “Why should we develop web application the desktop’s way?” or “Why can’t we simply leave web application the traditional static page approach?”. It is very essential for every web developers to understand the concepts and usage scenarios of desktop-style web application in order to leverage the benefits and great potential of what it could bring and deliver to their end users.

The answer to these questions are quite straightforward: *users demanded it*. As the matter of fact, 95% of computer users today are familiar with desktop computing invented by Microsoft’s Windows® since early 1980s. With the introduction Dynamic HTML in earlier browsers, application is now feasible to be developed in Web-based that offer richer user interactions.

Considering the ease of deployment, world-wide connectivity and global accessibility through the Internet, web application has been the most chosen and preferred type for companies and enterprise in this decade. Since then, computer users and enterprise workers have been massively demanding rich desktop-experience web application that they can use to access information in anywhere and anytime.

Our web development projects extensively used *WebUI Studio.NET® Suite* along with *Visual Studio 2005®* to achieve sophisticated web application that designed specifically to deliver the richness of visual styles and ease-of-use offered by desktop application.

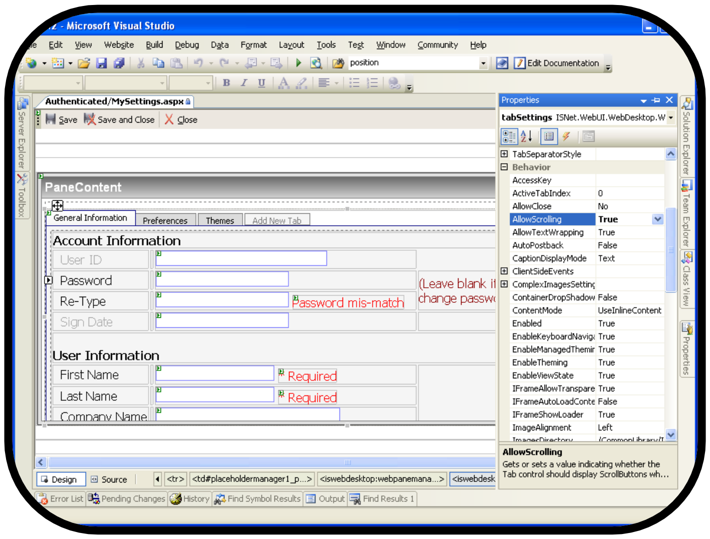
WebUI Studio.NET® used the following approaches to help achieving desktop-style web application:

Implementation of *AJAX* as integral part of core framework which expose high-level abstraction of functionalities to all data-bound components available in the Suite. Intersoft Solutions has implemented *AJAX* technique since year 2002 and currently has the most advanced *AJAX* implementation in the industry.
Extensive research and development of innovative User Interface functionalities through advanced browser capabilities such as superior DHTML techniques.
Transform basic User Interface functionalities into professional server-side components that can be easily consumed with high level of reusability.
Leverage latest Web trends and support for emerging technologies to deliver new level of innovations and user experiences.

So in our pursue to transform *Developer Network* into desktop-style web application, we discovered that *WebUI Studio.NET Suite* provides us with comprehensive tools and high-degree of functionalities that allow us to quickly achieve our design goals and objectives.

To finish this chapter, we will show you some of the tools offered by WebUI Studio.NET® and how we used them to achieve desktop-style user interface:

a) Using PlaceHolderManager, WebPaneManager, WebToolBar, WebTab, WebInput and WebCombo components to build a web user interface that our end users can access for customizing personal information such as email address or preferred theme.



Properties

tabSettings ISNet.WebUI.WebDesktop.W

⊞ TabSeparatorStyle	
⊟ Behavior	
AccessKey	
ActiveTabIndex	0
AllowClose	No
AllowScrolling	**True**
AllowTextWrapping	True
AutoPostback	False
CaptionDisplayMode	Text
⊞ ClientSideEvents	
⊞ ComplexImagesSettinç	
ContainerDropShadow	False
ContentMode	UseInlineContent
Enabled	True
EnableKeyboardNaviga	True
EnableManagedThemir	True
EnableTheming	True
EnableViewState	True
IFrameAllowTranspare	True
IFrameAutoLoadConte	False
IFrameShowLoader	True
ImageAlignment	Left

AllowScrolling
Gets or sets a value indicating whether the Tab control should display ScrollButtons wh...

Figure 8. User Interface can be arranged and configured as easy as the matter of mouse click. The rich design-time experience in *WebUI Studio.NET®* components significantly improve developer's productivity.

The *PlaceHolderManager* component is used to host dockable bars. The bar can be type of *WebToolBar* or *WebMenuBar*. When the bar is docked into PlaceHolderManager, the bar will inherit default settings from the PlaceHolderManager and the bar will be moveable by users at runtime into one of four screen edges (top, left, bottom, right).

The *WebTab* component is used as container and logical group of similar information. The *WebTab* component offers intuitive features such as capabilities to add new tab interactively and edit the tab's contents directly in the designer. The Click-and-Edit™ feature significantly increases the productivity of our team by being able to work with the controls visually and interactively without has to switch between template and normal mode.

The *WebInput* component is used extensively as main input control across our application. This enables all of our input user interface to have consistent styles and runtime interactivity. The WebInput component allows us to easily build masked input, intuitive DateTime input with calendar dropdown option, Access® style currency input with flexible formatting, and many more.

b) Using WebGrid.NET Enterprise to display *Workspace* data that arranged by group. The *preview row* feature is also enabled to let readers easily find information. We also used WebContextMenu component which is integrated into the Grid as context menu.

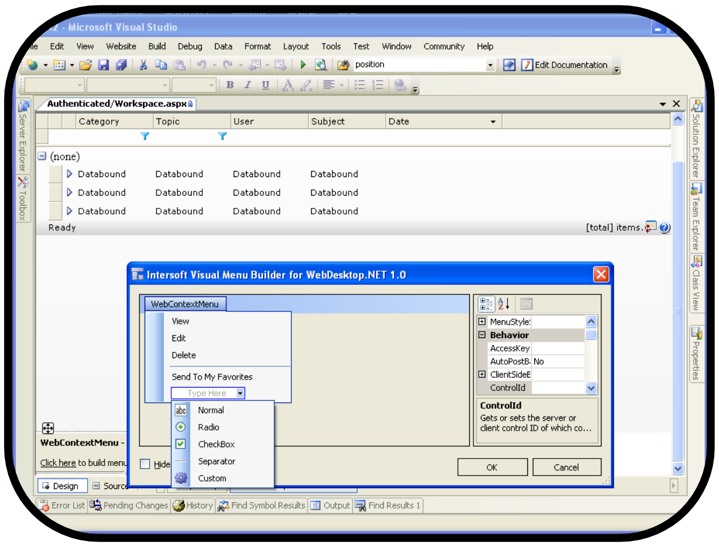


Figure 9. The advanced *Visual Menu Builder* lets you quickly configure complex, hierarchical menu by using Click-and-Type.

WebGrid.NET Enterprise is the core data presentation and manipulation, and reporting tool that we employed in most web forms of the application. The WebGrid component has dramatically helped us to achieve professional data presentation with advanced functionality and superior flexibility in order to deliver desktop-experience's web application.

Unlike other grids that we have tried, our team in web development division discovered that WebGrid.NET Enterprise is extremely easy to use and configure. With several properties set, we can provide rich user interactions for our users such as column moving, sorting, grouping and ungrouping, filtering, built-in context menu, multiple row selection, preview row and many more.

We are pleased to have known that the data-intensive operations such as grouping, sorting or data modification are automatically performed using AJAX technique. We even did not aware of it at first because we did not set any properties for that.

Furthermore, the WebGrid offers incredible flexibility in the area of "*features combination*". We found no significant difficulties in configuring hierarchical dataset that combined with load on demand feature. In the other hand, we employed the child table using self referencing feature combined with preview row and column set layout in grouped configuration which allow our end users to quickly analyze information.

VIII. New breed of web application made possible

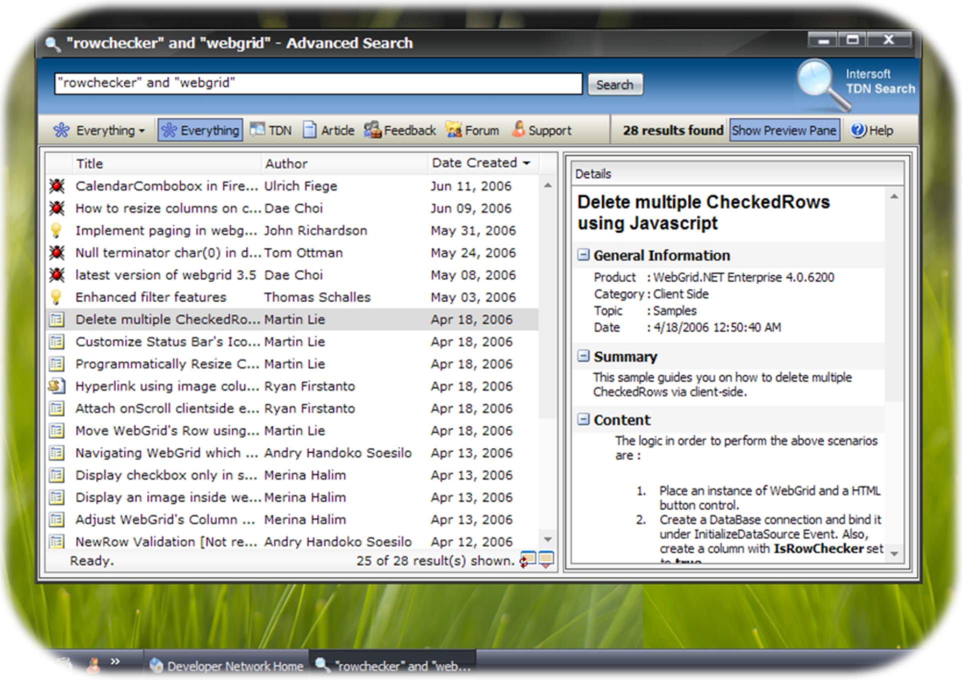
So far at this point, our entire web development members have understood about the desktop-style development concept and extensively used WebUI Studio.NET components to get the project done speedily.

One of the few wild ideas that we wish to get implemented in this version of Developer Network is the *Advanced Search*.

The Advanced Search borrows the concept of desktop search application like “Google Desktop Search” or “Windows Desktop Search” where users can simply type a keyword in the *Search textbox* located at taskbar area, and then a window appear to show the matched results.

The most exciting challenge in this objective is how we can bring the “ease of use” aspect of the desktop-search’s user interface into the Web. Needless to say, we need to use AJAX technique in most operations to ensure smooth and responsive user experience which fortunately can be done easily by using *WebGrid* and other AJAX-enabled components.

The *WebToolBar* is extensively utilized to allow users to toggle which location to search. The flexibility of the *WebToolBar* allows us to display complex menu items. Its strong object-model enables us to manipulate client side interfaces in professional fashion.



Title	Author	Date Created
CalendarCombobox in Fire...	Ulrich Fiege	Jun 11, 2006
How to resize columns on c...	Dae Choi	Jun 09, 2006
Implement paging in webg...	John Richardson	May 31, 2006
Null terminator char(0) in d...	Tom Ottman	May 24, 2006
latest version of webgrid 3.5	Dae Choi	May 08, 2006
Enhanced filter features	Thomas Schalles	May 03, 2006
Delete multiple CheckedRo...	Martin Lie	Apr 18, 2006
Customize Status Bar's Ico...	Martin Lie	Apr 18, 2006
Programmatically Resize C...	Martin Lie	Apr 18, 2006
Hyperlink using image colu...	Ryan Firstanto	Apr 18, 2006
Attach onScroll clientside e...	Ryan Firstanto	Apr 18, 2006
Move WebGrid's Row using...	Martin Lie	Apr 18, 2006
Navigating WebGrid which ...	Andry Handoko Soesilo	Apr 13, 2006
Display checkbox only in s...	Merina Halim	Apr 13, 2006
Display an image inside we...	Merina Halim	Apr 13, 2006
Adjust WebGrid's Column ...	Merina Halim	Apr 13, 2006
NewRow Validation [Not re...	Andry Handoko Soesilo	Apr 12, 2006

Ready. 25 of 28 result(s) shown.

Details

Delete multiple CheckedRows using Javascript

General Information

Product : WebGrid.NET Enterprise 4.0.6200
Category : Client Side
Topic : Samples
Date : 4/18/2006 12:50:40 AM

Summary

This sample guides you on how to delete multiple CheckedRows via client-side.

Content

The logic in order to perform the above scenarios are :

1. Place an instance of WebGrid and a HTML button control.
2. Create a DataBase connection and bind it under InitializeDataSource Event. Also, create a column with **IsRowChecker** set

Figure 10. Desktop-style search User Interface, made possible to the Web by using WebUI Studio.NET.

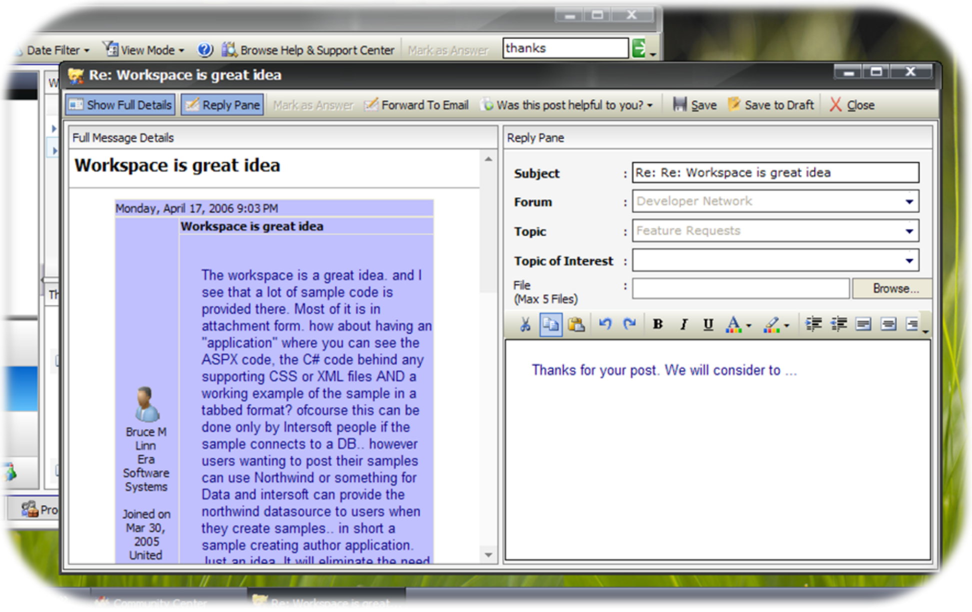
The "desktop search" idea above is just one of many applications that you can build using WebUI Studio.NET.

More sophisticated functionality that you can add to web application such as:

Real-time notifications, similar to those in Microsoft Outlook® or MSN Alert®
Live chat for customer service, similar to MSN Messenger®
Real-time geological or weather application using *WebFlyPostBackManager* and utilizing Goggle Map or Microsoft MapPoint web services.
Web "mash-ups" by aggregating information from various sources and using *WebDesktopManager* to host the services or gadgets.
And much more…

To complete this case study, permit us to share one more approach that we have implemented to solve usability issue while replying an email or a forum post. Many web users (including our customers) have experienced difficulty in replying email because most email programs included "previous replies" in the bottom. This caused usability issue because users have to constantly scroll back to bottom to read the point then scroll back to top to write the answer – and so on until several times for an email.

We solved the usability issue by implementing our own unique "Side-by-Side Reply Pane" where original email is shown at left pane and reply textbox is shown at the right pane. You should note that the left pane is resizable and collapsible so users can easily maximize the view for the reply pane.



thanks

Workspace is great idea

Monday, April 17, 2006 9:03 PM

Workspace is great idea

The workspace is a great idea. and I see that a lot of sample code is provided there. Most of it is in attachment form. how about having an "application" where you can see the ASPX code, the C# code behind any supporting CSS or XML files AND a working example of the sample in a tabbed format? ofcourse this can be done only by Intersoft people if the sample connects to a DB.. however users wanting to post their samples can use Northwind or something for Data and intersoft can provide the northwind datasource to users when they create samples.. in short a sample creating author application.

Bruce M Linn
Era Software Systems

Joined on Mar 30, 2005
United

Subject	: Re: Re: Workspace is great idea
Forum	: Developer Network
Topic	: Feature Requests
Topic of Interest	:
File (Max 5 Files)	:

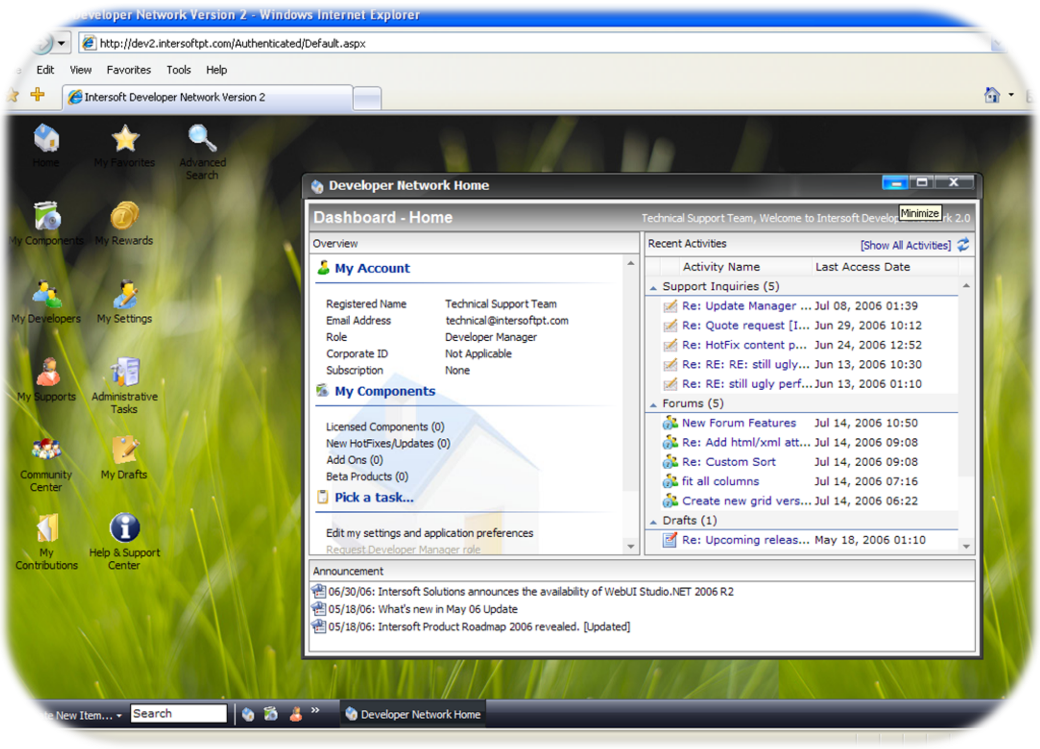
Thanks for your post. We will consider to ...

Figure 11. The "Side-by-Side Reply Pane" lets users to easily read the content while writing the reply without has to scroll back-and-forth.

IX. Summary

This case study is provided to you for a comprehensive reference for building rich desktop-style web application using Intersoft WebUI Studio.NET 2006 R2. We believed that the actual facts and the strength of our unique innovations described in this case study would help you design better web application and deliver better experience to your end users.

The following is the brief overview of the Intersoft Developer Network 2.0 after completely migrated to use WebUI Studio.NET components. Compare to traditional model shown in page 3.



Dashboard - Home

Technical Support Team, Welcome to Intersoft Developer Network 2.0

Overview

My Account

Registered Name	Technical Support Team
Email Address	technical@intersoftpt.com
Role	Developer Manager
Corporate ID	Not Applicable
Subscription	None

My Components

Licensed Components (0)
New HotFixes/Updates (0)
Add Ons (0)
Beta Products (0)

Pick a task...

Edit my settings and application preferences
Request Developer Manager role

Recent Activities [Show All Activities]

Activity Name	Last Access Date
Support Inquiries (5)	
Re: Update Manager ...	Jul 08, 2006 01:39
Re: Quote request [I...	Jun 29, 2006 10:12
Re: HotFix content p...	Jun 24, 2006 12:52
Re: RE: RE: still ugly...	Jun 13, 2006 10:30
Re: RE: still ugly perf...	Jun 13, 2006 01:10
Forums (5)	
New Forum Features	Jul 14, 2006 10:50
Re: Add html/xml att...	Jul 14, 2006 09:08
Re: Custom Sort	Jul 14, 2006 09:08
fit all columns	Jul 14, 2006 07:16
Create new grid vers...	Jul 14, 2006 06:22
Drafts (1)	
Re: Upcoming releas...	May 18, 2006 01:10

Announcement

06/30/06: Intersoft Solutions announces the availability of WebUI Studio.NET 2006 R2
05/18/06: What's new in May 06 Update
05/18/06: Intersoft Product Roadmap 2006 revealed. [Updated]

Figure 12. The new Intersoft Developer Network is built upon WebUI Studio.NET components.

The new Developer Network web application has helped our entire support department as well as end users to be more productive than ever before by making tasks easy and simple to be done in desktop-style environment.

For instance, end users can work with multiple documents and information at the same time such as writing new forum post in one window while reading another post in another window.

The user experience has been dramatically improved and since then our application's activity has been increased several times, which means our end users just loved it.

X. Frequently Asked Questions

Q: What is WebUI Studio.NET exactly?

A: WebUI Studio.NET is a suite of advanced ASP.NET server components designed to help you create professional web application in less codes and efforts. The Suite includes Grid, Combo, Input, Menu, ToolBar, Tab, Panes, Shell UI and many more.

Q: What do I need to run/use WebUI Studio.NET in my projects?

A: You only need a workstation that run Microsoft® .NET Framework® and Visual Studio development environment. The free 2005 Express edition is also supported.

Q: How can WebUI Studio.NET help to speed up our development?

A: WebUI Studio.NET components are built with rich design-time features and outstanding ease-of-use which help to reduce development time and hence increasing developer's productivity. In most common cases, you can simply use drag, drop and set properties for setting up user interface.

Q: How is the licensing scheme of WebUI Studio.NET? Does it have subscription option?

A: We take pride in offering affordable components with highest quality set of features. All components in WebUI Studio.NET is licensed per named developer and is royalty-free distribution. You can ship your products *to any number of customers, use it in any number of servers* – without the need for OEM license fees or whatsoever.

Regarding subscription, yes we do offer *WebUI Studio.NET Subscription* which includes 1 year maintenance for existing products and new releases, and also priority support.

Q: What type of applications are best developed with WebUI Studio.NET?

A: Although WebUI Studio.NET components can be used in any type of web development purpose, it is best suitable for *web application* type. Our customers used WebUI Studio.NET to develop these type of web applications:

ERP/CRM Application
Corporate Portal Application [aka Dashboard]
Financial Application
Business Solution Application
Procurement/B2B Application

Q: Does WebUI Studio.NET support AJAX technique to enhance user experience?

A: Definitely, Yes. All our data-bound components extensively implemented AJAX technique. In fact, we are among first vendors in the industry that adopt AJAX technique since year 2002 and so far has the most advanced AJAX implementation.[1]

Q: Our web application is used by wide range of users that used different browsers. Does WebUI Studio.NET provide cross-browser support?

A: Yes, WebUI Studio.NET components support Internet Explorer 5.5 SP2+ and Mozilla-derivative browsers. Our components extensively leverage Mozilla capabilities and push it to the maximum usage degree so that most functionalities are working properly and have similar behaviors as in Internet Explorer browser.

Q: What are the main differences of WebUI Studio.NET compared to other component suite?

A: The differences are obvious – our components offer supreme quality with advanced and innovative features designed to bring maximum business value. Take a simple example, our WebGrid component provides VirtualLoad™ and Hierarchical Load-on-Demand™ which have helped our customers to deliver great value to their end users since year 2003 – while our competitors are just offering similar features in 2006.

Another example, we introduced WebDesktop.NET with more than 10 patent-pending technologies in late 2005 – the industry's first kind of innovation which others may have not thought of its possibility before.

The bottom line is we continuously focus around Web technologies and strive to *create new kind of innovations* and *create higher standards* for the benefits of our customers. By subscribing to WebUI Studio.NET, customers have investment's peace of mind and ensure to receive new innovative products constantly.

Q: How can WebUI Studio.NET components benefit our end users?

A: There are lot of benefits, but we will cover only several top benefits here. For instance, all of our components are Section 508 compliant – that means your end users can easily interact with the UI elements of our components using keyboard.

In addition, our components are designed with very smooth and familiar user interface that end users would love to work with. The built-in sophisticated user interactions allow users to feel like working with desktop application which at the end improving their productivity and getting tasks done in less time.

[1] Refer to our *AJAX White Paper* for more information.

Q: We are a software company developing complex financial application. Does WebUI Studio.NET suitable for us?

A: Definitely, Yes. In fact our component suite provides advanced functionalities designed for complex enterprise application. For instance, **CITIGROUP** heavily relied on *WebUI Studio.NET* for developing high performance and very advanced financial application that require hierarchical drill-down reporting and more.

Q: I am newbie in ASP.NET web development. Does this comprehensive component suite suitable for me?

A: Yes, WebUI Studio.NET is suitable for both beginners and advanced developers. One of the good thing about our Suite is you don't have to be JavaScript savvy or DHTML expert to build rich and sophisticated web application. The rich designers included in the Suite enables you to achieve advanced functionalities using drag-drop and click.

Q: We have invested a large amount of dollars in other vendor's products but we are very interested to switch to WebUI Studio.NET. Do you offer competitive upgrade?

A: Yes, we have competitive upgrade offer. The only reason we provided this is to help developers around the world to enjoy better tool for their web development without the need to spend "double" costs.

Please contact sales@intersoftpt.com for competitive upgrade request. You usually only need to provide the official e-receipt of your previous purchase as the prove.

Q: What is the roadmap for WebUI Studio.NET? We need to know your future plan before deciding to invest on it.

A: We published our exact details of the roadmap for registered customers only for confidentiality reason. However in general, we can say that we have planned for even more exciting products for the next 2 years plan.

We are specialized in Web components and User Interface, our main focus and vision would be specifically around *Web technologies*. Therefore investing WebUI Studio.NET for your *Web application development* is definitely the right decision.

Q: Tell me more about WebUI Studio.NET – what are the included components, is it royalty free, what technologies that it employ?

A: You can get most of these answers in the following **Fast Facts**. To learn more, please see chapter **XII. References**.

XI. WebUI Studio.NET 2006 R2 Fast Facts

Suite Name	: WebUI Studio.NET®
Type	: ASP.NET Server Controls Suite
Number of Components	: 16 [per June 2006]
Included Components	:

WebGrid.NET Enterprise
WebCombo.NET
WebInput.NET
WebDesktop.NET, contains:

- WebDesktopManager
- WebPaneManager
- WebFlyPostBackManager
- PlaceHolderManager
- WebToolBar
- WebMenuBar
- WebContextMenu
- WebNavPane
- WebExplorerPane
- WebNotification
- WebDialogBox
- WebTab
- WebButton

Licensing	: Per Developer. Team-pack license is available.
Royalty-free Distribution	: Yes. Unlimited.
Server-side Technologies	:

Framework-based [WebUI.NET Framework]
Integrated AJAX through FlyPostBack™ architecture
AJAX WebForm Listener
Proprietary Advanced Theme Architecture
High reusability through strong object-oriented architecture leveraging interfaces, inheritances and encapsulations.
CSS Optimizer™
XML Compressor™
Object-based Style Architecture
Automatic Data Caching
ViewState Storage™

Client-side Technologies :

Client-side Framework
Strong object-oriented approach
AJAX Framework [including integrated and stand-alone AJAX Manager]
JavaScript and W3C DOM
Advanced DHTML, CSS and browser capabilities
XHTML compliant output[2]
IEMozBridge™ for Mozilla browsers
XML-based Client Storage

Designer Technologies :

Strong designer support for Visual Studio.NET 2003
Native designer support for Visual Studio 2005 [for .NET 2.0 compiled assemblies]
WYSIWYG Control Designers.
Advanced Editors.
Layout Manager.
Click-and-Edit™ interactive designer.
Click-and-Select™ interactive designer.
Click-and-Type™ visual menu designer.
XML-based Layout configuration.

.NET Framework Support : 1.x and 2.0 [Except WebDesktop.NET only supports 2.0]

Development Requirements:

Windows® based Workstation
.NET Framework
IIS 5.x / 6.0
Microsoft® Visual Studio.NET 2003 / 2005

Production Requirements :

Windows® based Server [2000 / 2003]
.NET Framework
IIS 6.0

Browsers Support :

Internet Explorer 5.5 SP2 / 6.0 / 7.0 for Windows®
Mozilla 1.8+ [Multi Platform]
Firefox 1.0 / 1.5 [Multi Platform]
Netscape 7.2+ [Multi Platform]
Camino for Mac®
Other Mozilla-derivative browsers

[2] Well-formed XML-based HTML output to be supported in upcoming 2006 R3 release. Read the limitations in XHTML White Paper.

XII. References

Intersoft Developer Network 2.0, http://dev2.intersoftpt.com

What's new in Developer Network 2.0, http://www.intersoftpt.com/Corporate/WhatsNewTDN2.pdf

Official WebDesktop.NET Press Release featured in Microsoft Virtual Press WebSite: http://download.microsoft.com/download/7/c/7/7c7ca062-9d42-4fd1-b675-3e80b3c1596d/intersoft.doc

Intersoft Solutions WebSite, http://www.intersoftpt.com

WebUI Studio.NET and AJAX White Paper, http://www.intersoftpt.com/WebUIStudio/AJAX.pdf

WebUI Studio.NET Home, http://www.intersoftpt.com/WebUIStudio

WebGrid.NET Enterprise Home, http://www.intersoftpt.com/WebGrid

WebCombo.NET Home, http://www.intersoftpt.com/WebCombo

WebInput.NET Home, http://www.intersoftpt.com/WebInput

WebDesktop.NET Home, http://www.intersoftpt.com/WebDesktop

What's new in WebDesktop.NET 1.5, http://www.intersoftpt.com/WebDesktop/WhatsNew

New technologies in WebDesktop.NET, http://www.intersoftpt.com/WebDesktop/Technologies

Components available in WebDesktop.NET, http://www.intersoftpt.com/WebDesktop/Components

Images Gallery of WebDesktop.NET, http://www.intersoftpt.com/WebDesktop/ImagesGallery

Experience WebUI Studio.NET Live Demo, http://www.intersoftpt.com/WebDesktop/Live

XIII. About Intersoft Solutions Corporation

Intersoft Solutions Corp. offers a wide range of eBusiness services in Web-development, software architecture, marketing solutions and consultancy. Building on its solid experience in Web-based application development, in 2002, the company started a new division focused entirely on the production of RAD components for the Microsoft .NET environment, specifically for ASP.NET Web development. Today, more than 50 "Fortune 500 Companies" have used the company flagship products; WebCombo.NET™ and WebGrid.NET™ Enterprise to enable richer and more effective information delivery which dramatically increases both developer and user productivity.