WHITE PAPER

Duncan Kenzie
CTO, ExcelSystems Software Development, Inc.

Presto: A Staged Approach to Web Enablement of Green Screen Apps

Exclusive Worldwide Distribution and Marketing Rights:
Business Computer Design International, Inc.
950 York Road
Hinsdale IL, 60521 USA
Phone: 630-986-0800
E-Mail: sales@bcdsoftware.com
Web: www.bcdsoftware.com

Software Development and Technical Support:
ExcelSystems Software Development, Inc.
101-9724 Fourth Street
Sidney, BC Canada V8L 2Y7
Phone: 250-655-1766
E-Mail: excel@excelsystems.com
Web: www.bcdsoftware.com

BCD

Table of Contents

Making The Right Choice For Web-Enablement Of IBM i Applications

There are still a large number of 5250, text-based applications running on IBM i installations around the world. IT managers and developers alike are challenged with finding a cost-effective, yet reliable method of moving these applications from text terminals to browser-based platforms. There are many modernization options to choose from, some from IBM and some from tools vendors. Making the right choice is important, because any mistake can be costly, setting you back many man months or even years in effort, and possibly costing many thousands of dollars. This White Paper discusses reasons why you should consider modernizing your 5250 applications, and why Presto should be one of the tools in your arsenal to accomplish that goal.

Why Web-Enabling Your Applications Makes Sense

There are several compelling reasons for web-enabling your applications, some of which have become even more relevant recently with the advent of Web 2.0 technology. Some of these are:

- To provide end-users with access to more timely and understandable information.
- Make information more widely available, beyond the reach of current organizational boundaries to customers and business partners.
- To provide more productive, easier to learn and easier to use applications.
- Nowadays, employees new to a company or new to the workforce often have never used a green screen application, but are already familiar with a web browser interface.
- To demonstrate to upper management that the IBM i platform is relevant to upper management, and that there is no need to move critical business applications to other platforms.

The Challenges of Achieving Web Enablement

There are several barriers to modernizing legacy IBM i applications and making them web enabled that organizations encounter. These barriers have resulted in many IBM i shops delaying the transition to browser-based applications. Some of these are:

1. Confusion created by IBM's complex solutions, frequent renaming and rebranding of their tools, and frequent shifts in their guidance to customers of how to modernize.
2. The existence of a large legacy codebase that is complex and time-consuming to re-write.
3. Lack of web-development skills in existing personnel.
4. Lack of awareness of the productivity benefits of moving to web applications.

This White Paper discusses reasons why you should consider modernizing your 5250 applications and how to address the challenges of Web Enablement. It will explain why Presto should be one of the tools in your arsenal to accomplish that goal.

If you have been frustrated or confused by IBM's offerings, you can research solutions from IBM ISVs. For example, BCD provides highly productive, easy to learn solutions specifically designed for the IBM i platform. Our solutions are built specifically to meet the needs of existing IBM i shops and their personnel, with a strong emphasis on ease of use. In addition, we provide solutions that support the full range of options for modernization, from extensive reuse of existing code to complete rewrite as native web applications. For large, legacy codebases, you can web-enable with no underlying code changes required using Presto. Or, you can refactor some of that code into modules and reference it from true web apps or from Presto-enabled apps.

Comparing Modernization Strategies

There are several different strategies you can pursue to transform legacy applications to the web. Three of the strategies supported by BCD's products are outlined below.

1. Rapid re-engineering.
In this approach, you take your existing code base and separate the database and business logic from the presentation logic, and refactor the 'back-end' code into callable modules. Then, using a rapid application development tool such as WebSmart (ILE or PHP editions), you build a web interface and integrate the back-end business logic with the browser code. The result is a true web application, running with the IBM i as the application and database server.

This strategy is great for organizations who have the time and human resources to invest in training, planning, UI design and re-engineering work in order to build a web application. Using a rapid development tool such as WebSmart can dramatically cut down the amount of time required to succeed in such a project, by reducing both the learning requirements and the actual development time. But if you don't have the resources, here's another approach:

2. Instant Refacing.
This approach provides a web layer on your existing 5250 applications, without adding any additional server overhead. BCD's solution, Presto, intercepts the 5250 data stream and pipes it to the browser instead of a green-screen session, where it is transformed on the client into a web page.

This strategy is great because it requires little or no initial effort on the part of your IT staff, while providing a path for transforming features of individual pages on a 'play by play' basis: as you run an application, you can choose to enhance the UI, all non-invasively, without having to touch any of the underlying codebase.

3. Enhanced Re-engineering.
This is a combination of the two previous strategies, and it allows you to leverage the best of both worlds. You begin by instantly refacing your application- simply run it. With Presto you can then choose to apply global options, such as different 'skins' (color schemes, logos, appearance of function key buttons, etc.) to every screen in an application. Next, when running the application, as you encounter screens you need to enhance you can add functionality, including more sophisticated user interfaces and augmented server-side data access routines to supplement information needs. For example, using AJAX and SOA technology you can call WebSmart programs from within Presto-enhanced screens to incorporate new data into your application. A typical example might be where you provide a dropdown box of available shipping rates to a zip code by loading the information from a SOA interface to an external web server such as UPS. This approach is also non-invasive- you can still leave the underlying code base untouched. However, it has the added value of providing new, relevant information to users of your application.

How Presto Works

Presto intercepts the data stream that is normally sent to an interactive green screen session and pipes it to a standard web server application and browser interface. It takes advantage of native System i web server technology, such as the Apache Web Server, and uses standard web protocols such as HTTP and CGI programming. Presto provides 5250 screen enhancement technology, allowing developers to deploy 5250 applications on the web using non-invasive design techniques. This means you do not need to change any underlying RPG or DDS source code or recompile anything in order to enhance the look and functionality of your applications. The resulting applications use modern, native HTML, CSS and JavaScript to produce professional quality pages that give your users the advantages of powerful web applications. Presto comes with a PC-based application called Presto Designer that mitigates the need for your staff to learn web technologies such as HTML and CSS used by Presto.

Presto Runs Straight Out Of The Box

To start using Presto you complete a few simple installation steps. These include loading the Presto library, configuring an Apache web server instance, then starting that server. Next, open a browser window (Internet Explorer or FireFox, for example), and go to a specified URL for your server, based on how you configured the web server. You can literally be working with your existing green-screen applications in a browser within 15 minutes. As Presto captures the 5250 data stream it applies intelligent transformation rules to the data in order to present it in the browser with a web application look and feel.

You can also install the Presto Designer on your PC. With this you can set up global defaults for how your applications will look as web pages. These let you control the appearance of every page. For example, if you want your organization's logo on the page, you make a simple change to a single HTML file. Another option is to choose from one of several 'skins' that ship with Presto that determine the appearance of all pages. Or, initially, you can just accept the default configuration as shipped, so you are up and running as soon as possible.

Presto Eliminates 5250 OLTP Costs on V5R4 +

For i5/OS V5R4 or later, Presto only uses batch CPW when running. It uses no interactive CPW. Presto jobs use only slightly more overhead than completely native CGI applications to run, and significantly less than interactive jobs, so if your system uses interactive CPW you can actually see a performance improvement when using Presto instead of traditional green-screen sessions. If you run the WRKACTJOB command you will see jobs in your interactive subsystem (usually QINTER), but these are different than normal interactive jobs in that they are using no interactive processing CPWs.

Presto Requires No Additional Software or Middleware To Purchase

Presto uses the free HTTP Server powered by Apache that is available on every iSeries, System i or IBM i. All other required components, including the PC-based design tool, are included with Presto.

Presto Does Not Need WebSphere To Run

Because Presto uses the standard HTTP Server, it does not need WebSphere server to run. Compared to some solutions from IBM, this also reduces the complexity of installing, configuring and running Presto. In addition, it results in much fewer system resources being used than with Java-based solutions.

Presto Does Not Need DDS Source

Some of your legacy applications may come from third-party vendors, or some may have inaccurate or missing source code. Presto does not rely on source code in order to modernize your applications. It reads and interprets the 5250 data stream produced by your object code, dynamically, as you run your application. It also intelligently identifies the uniqueness of screens so that you can reliably apply customizations to those screens if you need to.

Presto Utilizes Web 2.0 Technology

Unlike other solutions, Presto is not a port of a PC Client/Server solution, but was built from scratch to take advantage of the latest browser technology. In addition, it uses the very latest software implementations, including partial page updates using AJAX and JavaScript libraries of code for complete cross-browser compliance.

'Web 2.0' means different things to different people. One aspect of web technology people often associate with Web 2.0 is social networking applications, such as Facebook and Twitter. Also, web-based productivity apps such as Google docs are often regarded as part of Web 2.0. All these applications rely heavily on JavaScript libraries for greater productivity in programming and cross-browser compliance. They also use AJAX technology extensively. The benefit of AJAX is that it provides partial page updates in immediate response to user actions. For example, you can immediately validate an input field as soon as the user exits it, and provide an error message next to it.

Presto was built from scratch to run in browsers and to take advantage of Web 2.0 concepts. For example, each web-enabled 5250 screen replaces a specific area in the main page, rather than regenerating a complete page each time. The result is pages render data from the server as fast as possible. In addition, the user experiences a smooth flow from one screen to the next, because the browser doesn't have to load

an entirely new page each time. Pop-up windows also appear with clean user interfaces, uncluttered by unnecessary borders or UI components. Also, you can use take advantage of AJAX to perform partial page updates easily, so you can actually enhance your original 5250 UI in the web browser, all without having to touch any of the original code. Another advantage of AJAX is that it makes heads-down data entry in web pages just as productive as in 5250 apps. This is because the response time of sending and receiving data to/from the server is much faster than traditional web page delivery, and comparable to 5250 interactive response times.

Presto also uses Web 2.0 JavaScript libraries, such as jQuery. jQuery provides easy ways to do complex tasks, such as using AJAX. Normally, an AJAX routine in JavaScript might be a 100 line long function. In jQuery you can accomplish the same thing with one line of code. In addition, jQuery provides cross-browser compliance, so Presto applications will run in all popular browsers such as Microsoft IE, FireFox and Safari.

Presto Produces True 'Liquid' Web Pages or Monospaced Pages

The resulting output from Presto's transformation of the 5250 data stream is true HTML, CSS and JavaScript, consistent with modern web programming techniques. For example, all HTML takes advantage of layout features of the markup language that make it easy for you to redesign page layouts en masse with few changes to the HTML. Pages are 'liquid' as opposed to 'fixed layout,' which means elements will stretch and move according to the size of the containing browser window, just like you would expect with any typical web site. Also, pages use HTML tables for presenting screens that contain subfiles, or lists of records in the underlying green-screen application. Tables are still recognized as the best way to present tabular data. Function keys can easily be transformed to appear like graphic buttons with 2D or 3D appearances. And, they can be grouped to appear on different places on the page. For example, you may choose to have them appear along the bottom, similar to a traditional green-screen app, or stacked vertically along the left side of the page.

If you prefer, though, you can use the mono-spaced font (Courier, Courier-new) skins provided with Presto. These have the advantage of having 5250 fields line up more consistently in their web representation. However, many modern apps tend to use proportional spaced fonts such as Arial or Verdana. You can easily switch back and forth between mono and proportional spaced fonts to see which looks better.

The Presto Designer Gives You Control Over Each Page Design

The Presto Designer lets you see both the green-screen version of your page and the web version. Figures 1 and 2 below show you an original green-screen followed by the web version. In this case, the web version was initially automatically created by Presto, based on the selected skin. Then, some transformations were added, such as dropdown boxes to replace subfile options. In addition, if you want to rearrange information, including text or data, the Presto Designer gives you complete text

editing capabilities for the HTML. You can add any HTML tags you choose, move information around, or even hide it from the user. Unlike some tools, which have their own proprietary language that you have to learn, Presto uses universal standards for web programming, such as HTML, CSS and JavaScript. Once you learn some of these skills, you can leverage them if you decide to build web applications using web development tools such as BCD's WebSmart.

Consider the illustrations below:

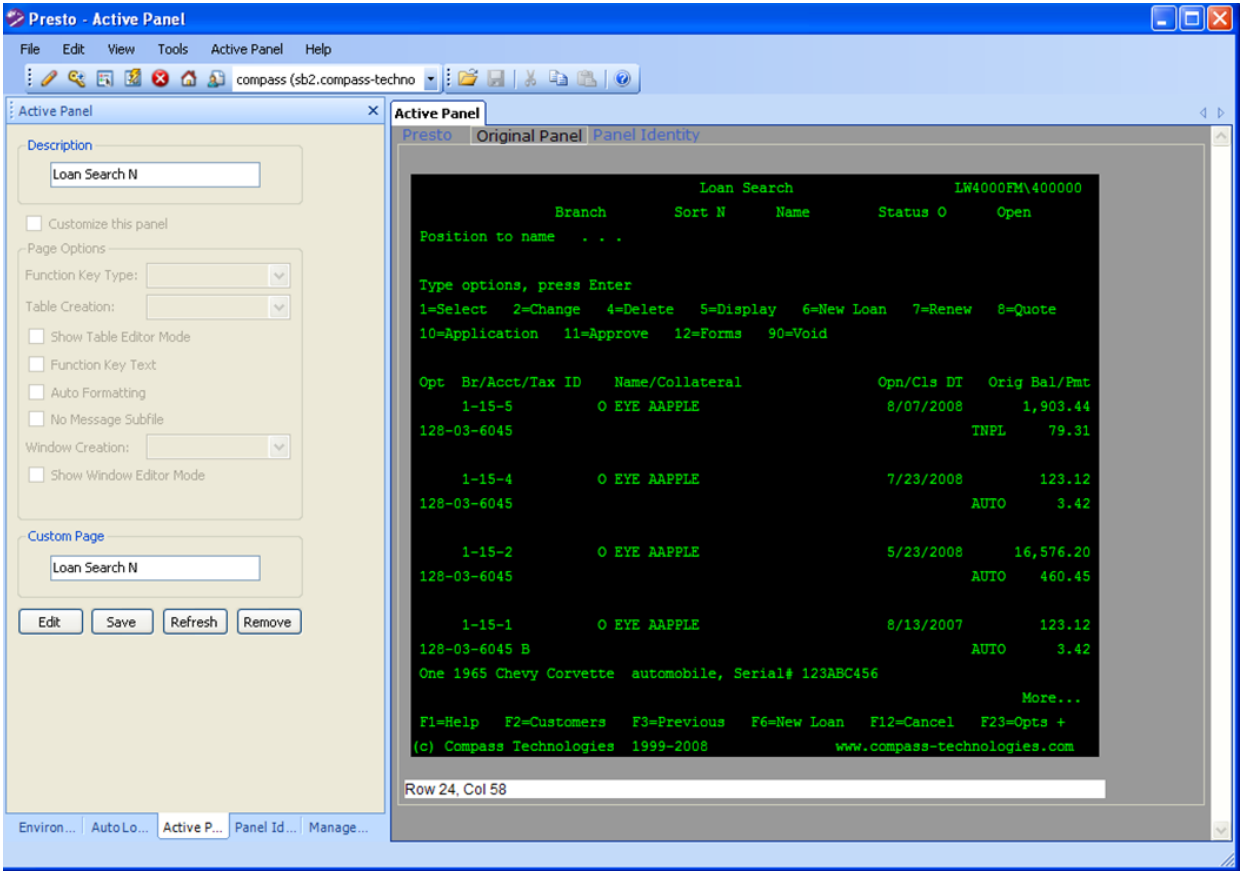

Figure 1. Original green-screen in the Presto Designer.

This is the same green-screen program, enhanced with a custom skin:

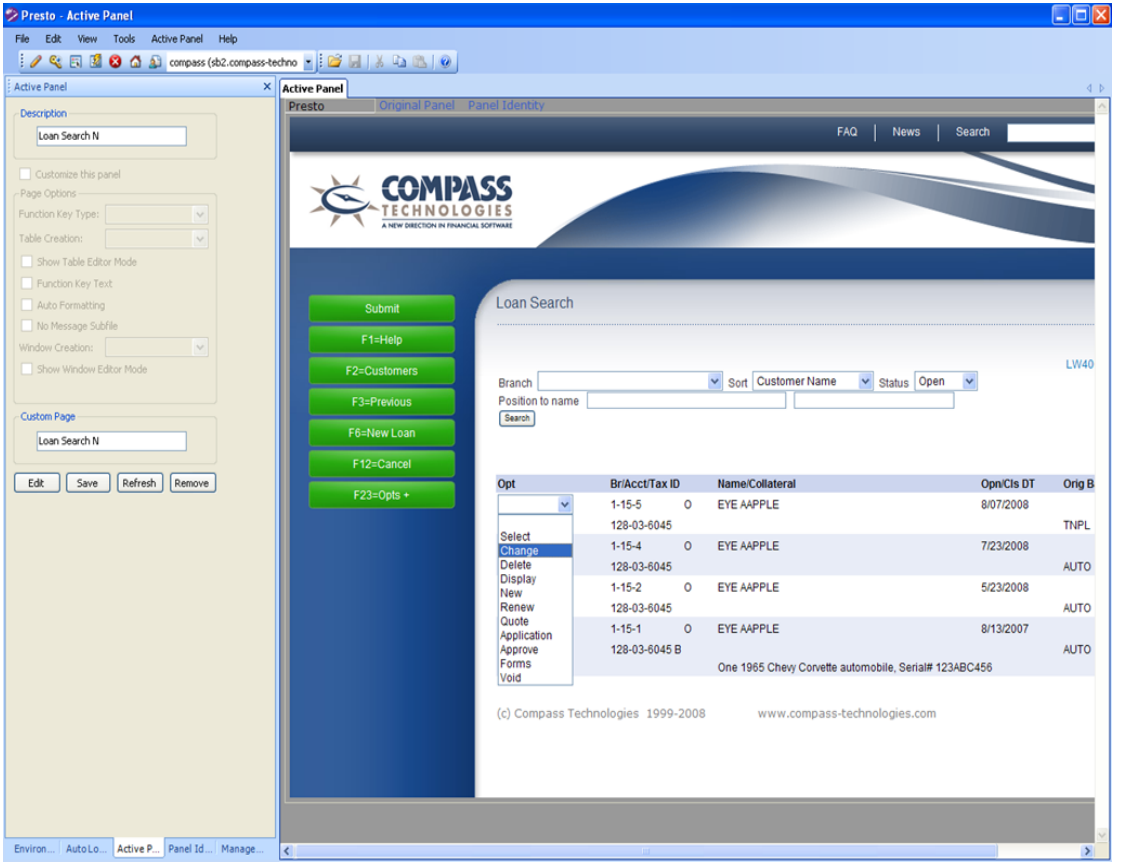

Figure 2. Same screen, transformed using a custom skin.

Notice that, in Figure 2, the function keys (in green) now look like buttons, are stacked vertically, and all have the same width. In this case, this was done by using a customized skin, so this design will automatically apply to every page in the application, without you having to apply it each time. The leftmost part of this screenshot is an area within the Presto Designer that is used to describe the customized screen.

Presto Can Auto Detect and Identify Most Screens

One of the configuration options of Presto lets you assign a specific part of each screen (by row and column) that will be used to identify that screen uniquely. This feature is important because it minimizes customization efforts. Instead of having to find unique elements on each screen as they appear, you can focus on defining the customizations. This works particularly well for applications where a screen id has been embedded in the original 5250 application, for example. However, if you need to, you can also define custom locations on any screen to uniquely identify it.

Enhance any Screen With The Presto Designer

The Presto Designer is a PC application that provides you with a suite of tools for customizing your applications, either at a global level or for individual screens.

For global settings you can determine what graphics and fonts to use, or the location of function or command keys on every page. You can also determine the mapping of the keyboard as it corresponds to typical 5250 keyboard functions such as Enter, Field Exit, Page Up and Page Down. You can run your application inside the Designer and edit the generated web source code for any screen at any time. Presto captures the underlying 5250 screen definition (screen title, screen constants, function key usage, field layouts and lengths, etc.) and presents you with an interface that lets you enhance the screen presentation.

Consider this illustration:

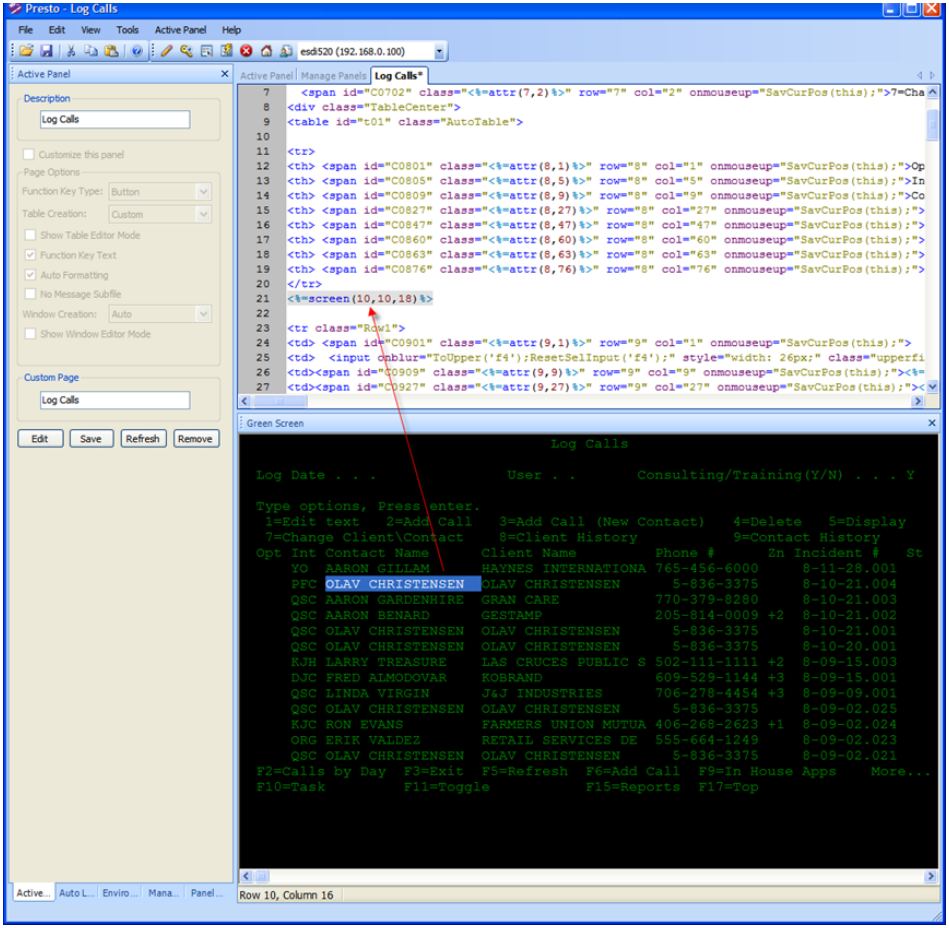

Figure 3. Moving dynamic content from the green screen display to the web version.

You can choose to see a vertical split-screen that shows the fixed-position green-screen image for the screen being enhanced, along with text-based HTML in the other part of the split. From this split screen you can highlight, by dragging the mouse, parts of the green-screen image, then drag those onto the HTML area (Figure 3). This provides a simple means of rearranging page content. For example, you may choose to move error messages from line 23 to some place near the top of your page. Simply highlight line 23, then drag and drop it where you want.

You can enhance the page presentation by adding any HTML or CSS to a page. Presto shows you the initial HTML it uses to render the screen and provides you with a full featured text editor for customizing the code. For example, you can easily add images, change fonts, add tables, date pickers, dropdown boxes, etc.

You can also augment the functionality of the page by adding any JavaScript code for client-side scripting (JavaScript routines). The next section explains this.

Using Client-Side Scripting For Flexible Presentations

One of the challenges of transforming 5250 screens into web pages is that each screen can show different fields based on conditional program logic. For example, in some circumstances a price override field on an order line might be available to a customer if that item is on special. Each time the screen is presented it may appear differently, even though it is coming from the same program. To address this, Presto supports using JavaScript to validate, format and transform input fields. For example, you can easily add a routine that checks if a field contains all numeric characters, or is a valid date format.

You can also augment the functionality of any page by adding server calls directly from the page to server-side code such as WebSmart programs. This functionality takes full advantage of AJAX technology, allowing almost infinite extensions to existing applications. For example, an AJAX call could be added to validate a field against a database file, and then return a message to display if it fails the validation. Or, you can easily add a calendar function to a date field, providing the ability to popup a calendar in another window.

The Presto Designer will save any customized screen enhancements you create, on the IBM i server, so you or other developers can modify them later. There is no need to compile anything- once your changes are saved, the screen will immediately be enhanced. The next and subsequent times you view the screen in a browser your enhancements will included.

An Example Transformation

To change UI components on a screen, you simply edit the page in the Designer and add lines of code to bottom of the page. These lines are called 'transformations' because they generally change something already on the screen, such as text or an input field, into an improved UI component. For example, 5250 applications commonly require users to type dates in input fields. When the user presses Enter,

the application validates the date. For example, it might check for valid month number, number of days, or the year being within a certain range. With a Presto transform statement you can change the input field into a 'date picker'. This provides the user with an icon next to the field to click on, so they can select the date from a calendar. This minimizes the need for the program to make a trip to the server to validate the date value. Here's some example code of how to change an input field on line 12, column 40 to a date picker:

```
{row: 12, col:12, to_object: "datepicker"}
```

To further simplify this, the next release of Presto contains a Visual design tool that lets you select the field visually and set its properties to a date picker or other UI elements.

Create Your Own Customizations

While we have provided a large number of common customizations with Presto, you may have unique needs, depending on the design of your green-screen apps, or on what you'd like for an enhancement. Presto has exit and entry points in the process where screens are written to the browser that let you inject your own customizations. These can be client-side, server-side (via AJAX calls), or a combination of both. You can embed them in the customized pages just as you would with the ones that ship with the product.

Project Management Provides Flexibility in Design

Presto supports a concept of 'Environments'. Environments provide a way for you to group a set of programs and/or screens with a common set of characteristics. You can associate a production and/or test library with an environment, so the same underlying 5250 program can have different versions of screen enhancements available, and so that you can test changes without affecting the production version. You can also associate a set of global defaults for any environment, such as which Presto skin to apply to each screen. For example, you might choose to have an Order Entry application use a different color scheme or graphics than a Human Resources application. You can also copy screens between environments, so if you decide a screen should have a different set of global values that affect its appearance it's easy to do so. Also, if you are an ISV and want to make a packaged application have a customized look for each client, simply create a new environment for each client and copy the screen enhancements there.

Seamlessly Integrates With Existing Web Pages or Applications

With Presto, you can specify a starting program and screen for your application, so that it is possible to invoke your application via a URL from another page or from the location bar of the browser and have it skip the initial standard 5250 signon

page. This feature helps make your applications work like true web apps, avoiding the appearance of a 5250 session.

Using embedded AJAX technology or HTML such as customized links or forms you can also integrate Presto applications directly with native web applications written with WebSmart or other web development tools.

Embed Presto Applications in Nexus Portal

Nexus Portal is BCD's web portal product, running on IBM i. Nexus Portal provides a secure and consistent web user interface for managing and deploying web applications. It also includes ECM (Enterprise Content Management) features. You can embed Presto applications in Nexus Portal by configuring portlets to run the applications. Again, you can configure these so that the initial 5250 signon screen is skipped, avoiding the need for the user to sign in twice. Instead, they log on to the portal using the web interface. The pages they see are controlled based on their user profile and the Nexus group(s) they belong to. Using the administration web interface in Nexus, the Nexus administrator can decide which Presto applications a user has access to, by granting the appropriate authority to the portlet. Presto comes with some skins that have an appearance compatible with Nexus skins, so Presto apps can have the same look as their container portlets and the Nexus portal in general.

Using WebSmart To Further Enhance Presto-Enabled 5250 Apps

WebSmart is BCD's Rapid Application Development Tool for building true web applications running on IBM i. WebSmart uses intelligent templates to create complete web applications with little or no programming required. You can use WebSmart in conjunction with Presto applications to extend the functionality of the apps beyond just UI formatting and incorporating web UI elements such as dropdown boxes and links. For example, WebSmart programs can perform additional database access or updates, or extend the existing application with new business logic. Using AJAX as the glue that holds Presto and WebSmart apps together, you can embed calls to WebSmart programs directly in Presto customized pages, without having to touch any of the original codebase. As the page appears, it first renders and translates the 5250 data stream. Then, if you have embedded calls in the custom pages, those calls are performed, and any additional output or business logic done by the called WebSmart programs are included in the final version of what the user sees.

You can write WebSmart programs using the ILE version, which generates RPG CGI objects, or with the PHP version.

What Skills Do You Need To Use Presto

The degree of web technology skills you and your team will need depends on how much customization you want to do. Here's a table of approaches and skills needed:

Approach	Web Technology Skills
Out of the box- just change skins and environment settings	None
Create your own skins	Some HTML and CSS (or hire a web designer)
Create custom pages of 5250 screens	HTML and possibly some CSS
Transform areas of screens to use web UI features	Minimal JavaScript (explained in User Guide extensively)
Include AJAX calls to WebSmart programs to include more UI elements or business logic	JavaScript function call to AJAX (explained in User Guide extensively), WebSmart programming

As you can see, you can produce great looking applications without having to know much web technology initially. Presto also encourages an incremental approach to translating 5250 screens. You can start running all your screens right out of the box, with no intervention on your part to customize those screens. Then, as you progress, or get more feedback from end-users, you can begin to customize screens on an as-needed basis. Eventually, you can move on to more sophisticated approaches where you actually start extending the application to include more functionality.

To customize pages, it is helpful to have basic HTML knowledge, along with CSS (Cascading Style Sheets). HTML is generally used to present content, while CSS is used to control page layout, but sometimes HTML is also used for layout. For example, coding HTML tables is still the best method for presenting tabular data like subfiles in a web page. Fortunately, when you initially edit a custom page that Presto identifies as being a list, or subfile screen, the initial HTML table markup is generated for you, making it easier for you to understand and customize.

Security and Presto

Because Presto runs as a web application, many of the same security considerations that apply to conventional web applications also apply to Presto-enabled applications. There are many possible ways to configure access to your System i, such as using Firewalls, proxy servers and VPN (Virtual Private Network) servers. A detailed discussion of these is beyond the scope of this document. These are the general areas of concern you should consider:

1. **Data transmission between client (browser) and server (System i/Presto server).** To secure data transmission, you should use SSL (Secure Socket Layers). SSL encrypts data prior to transmission, sends it encrypted , then decrypts it. This works both ways. In other words, if the user sends a response to a page via a browser, the data is encrypted in their browser prior to being sent back to the server. Likewise, when the web server sends data to the client, it encrypts it. All major commercial B2C sites use SSL, such as Expedia and Amazon. Note that this is more secure than Telnet, a common protocol for 5250 apps over TCP/IP, since Telnet does not encrypt transmitted data.
2. **Network configuration (firewalls, etc.).**
3. **Apache web server on IBM i.** You can configure Presto to run in an Apache web server instance that is shared by other applications (such as BCD's Nexus portal, or any WebSmart programs you might create). Or, you can configure a separate instance, just for Presto apps. The Apache configuration ensures that all users are restricted to only those areas (libraries and/or programs, IFS directories and/or files) that you explicitly open up for access. You can also impose authorization schemes on these resources, requiring users to type a valid user id and password before being allowed to proceed further.
4. **IBM i user profile security.** For most Presto apps, the first screen a user will see is the sign-on screen, requiring them to enter a user id and password, just like any 5250 session. You can configure Presto to bypass the signon screen. For example, you can embed Presto apps in Nexus. If you do this, Nexus can retrieve the user id and password from a temporary cookie to login to Presto. There is a slight security risk here. Even though the password is encrypted in Nexus, it must be briefly decrypted in the browser when it is passed to Presto. If someone got a hold of your PC and grabbed the cookie while you were logged in, they could possibly steal your password. However, the same risk exists with

5250 apps. If you leave them signed on, it's possible for someone to change your password and then login with the new one.

5. **IBM i object security.** All the same object security that applies to 5250 apps will be enforced by the Presto application.

If you want to allow outsiders to access your application in the web, you can take a couple of approaches. One is to create a unique user profile and password for every user who needs access. Another is to create a single user profile and password and write an API interface that maps those values to your own custom database of user ids. A third option is to use Nexus' user management features to interface to Presto.

Conclusion

Presto provides a low risk option for IT groups who do not have the time, resources or skill set to rewrite large, legacy 5250 applications as web applications. It is low risk because it provides you with a staged approach to web-enablement and because it does not require any changes to your existing source code. It also provides you with a future development path, in conjunction with rapid application development tools for the web, such as WebSmart ILE or WebSmart PHP. As your skills develop and resources become available, you can selectively re-engineer or rewrite parts of your application with WebSmart.

You and your staff need minimal time and minimal understanding of web technologies to produce professional-looking, functionally rich web versions of your 5250 applications. In addition, because Presto on V5R4 and higher uses no interactive resources, you can gain improved performance and lower system impact by running your apps using Presto. You can also use Presto to make applications that were previously shackled to inside users only available on the web to business partners such as customers, vendors or remote employees/contractors, thus further extending the life of your applications.

Your users will appreciate working in a new, attractive environment, with productivity and ease of use tools to make your applications exciting and friendly to use. Finally, management will believe in the relevance of the IBM i for the future and perceive it as a web server capable of delivering cutting-edge applications. All these benefits come with minimal investment in time and resources.

About the Author

Duncan Kenzie is the president and CTO of ExcelSystems Software Development Inc., the authors of Presto, WebSmart ILE & PHP, Nexus, Catapult, and many other System i productivity tools, exclusively marketed by BCD Software International.

Duncan began his career with IBM and has 30 years experience in the Midrange space. He is a frequent speaker and author of several technical articles for leading Midrange publications, and is an expert in web technologies on the System i.

For more information

For more information on Presto, and BCD Software International, visit our website: www.bcdsoftware.com

Copyright Notice

This White Paper is written and produced by ESDI using Adobe FrameMaker 7.0. Copyright © 2009. ESDI. All rights reserved. This document many not, in whole or part, be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine readable form without written consent from ESDI. Last updated April 23, 2009.

Conventions

In this guide the term 'iSeries' is used to refer to the System i, eServer i5, iSeries and AS/400 computer systems.

Acknowledgments

Throughout this white paper, reference is made to several trademarks: WebSmart is a registered trademarks of ESDI in the US and Canada, trademark elsewhere in the world Presto is a trademark of BCD. IBM, System i, iSeries and OS/400 are either registered trademarks or trademarks of International Business Machines in the United States and/or other countries. Windows® is a registered trademark of Microsoft Corporation in the United States and/or other countries. All other trademarks are acknowledged as the properties of their respective owners.