BASIS Advantage

Volume 15 • Autumn 2011

Document Management Solutions with UnForm®

Production › Delivery › Archiving › Scanning

UnForm is a powerful enterprise document management software solution that seamlessly integrates with any application. The UnForm suite includes laser form and electronic document production, document delivery via email and fax, document archiving and management, and document imaging/scanning. UnForm is a platform independent client server application for Windows®, Unix®, and Linux.

UnForm Laser Forms

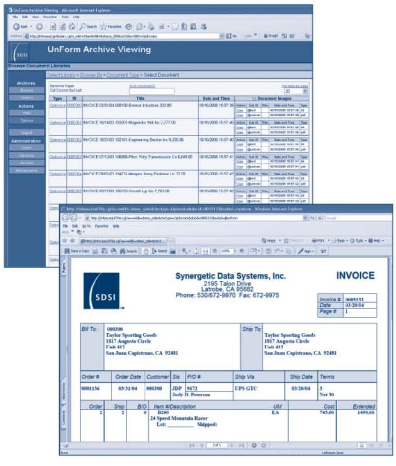
UnForm seamlessly integrates with any software application.

- Windows-based graphical design environment
- Eliminate pre-printed forms with laser printer output
- Produce presentation quality reports
- Create e-Documents in Adobe® Acrobat PDF format
- Email e-Documents automatically
- Print bar codes in most symbologies
- Create laser checks with MICR encoding
- Dynamic image conversion and scaling capability
- Database access via ODBC
- Microsoft® Fax Server support
- PCL 5 and Postscript® printer support
- Extensive programmability

Document Archiving and Management

The UnForm Document Archiving and Management component provides the ability to capture, store and retrieve paper-based and electronic documents.

- Rules-based document archiving
- Archive concurrent with document printing
- Store multiple versions of a document
- 10 levels of user defined category indexing
- Document linking control
- Fast web browser-based retrieval
- Client API for application-based retrieval
- Index oriented archive browsing
- Full feature search capability

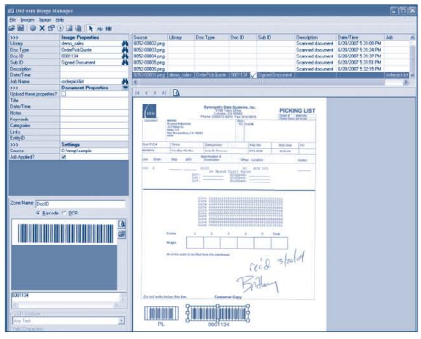


Universal web browser document retrieval

Document Imaging/Scanning

Windows client application that provides a document scanning and importing tool for capture of documents external to the UnForm processing environment.

- Windows client application
- Integrated work environment for image capture and upload
- TWAIN compliant scanning interface
- Multiple property assignment modes
- Barcode and OCR zone detection
- Automatically match or group images with related archive documents
- Extensibility via VB Script



Document Image Manager

Synergetic Data Systems, Inc.
2195 Talon Drive
Latrobe, CA 95682 USA

(800) 446-7374 or (530) 672-9970
Fax: (530) 672-9975
sales@synergetic-data.com
www.synergetic-data.com

Google, Cloud, Tablets, Smartphones – You are Future-Proof With BASIS

As we look to the future, there are ongoing challenges and opportunities that the ever-changing technology landscape presents to us such as the Cloud and mobile computing. Hence, it is a good time to pause, take stock, and strategize our next steps. It's what we do continually at BASIS, and I'd like to share our current vision with you – our partners and customers – the BBx® community.

BASIS has morphed over time from a company with the single focus of producing the best Business BASIC on a choice of operating system platforms, into a company that now delivers a much broader BBx eco-system. This environment of technology components serves as the platform for modern business application solutions, fully cloud-enabled and mobile capable, all from a single code base. Specifically, BASIS has evolved from a Business BASIC eXtended world to a Java-made-simple world with BBj®, the latest generation of BBx. BBj embraces the Java environment's cross-platform capabilities while augmenting and extending Java with a simple eco-system of System Administration & Deployment, Language & Interpreter, Database, and Development Tools components. BASIS takes the best that current and future Java versions have to offer and simplifies the complexities of the environment while retaining backward compatibility for all BASIS applications.

BASIS and Java share a commonly held goal for application developers to "write once, run anywhere." The great and valuable exception is that BASIS doesn't fall foul of the "write once, debug everywhere" criticism of Java since BASIS provides a layer of abstraction that overcomes the issue. In a truly revolutionary move, this layer of abstraction has allowed BASIS to further extend the "write once, run anywhere" term to include the browser environment. We have achieved this remarkable feat by interpreting BBj code into *either* code that executes in a JVM or into HTML5 and JavaScript that executes in a browser on either a desktop or a mobile device such as a tablet or a smartphone. This feat is an impossibility for a pure Java application. Our new Browser User Interface (BUI) is equally remarkable for the simple fact that most existing BBj graphical applications will run, unchanged, in this new environment. Naturally, developers need not stop there. With CSS (Cascading Style Sheet) support, they can graphically enhance the application in either very subtle or very dramatic ways.

So, what does this all mean?

Not only have we dramatically extended your existing investment in our technology for the new mobile world, but we have added an amazing number of new features and components to our technology stack. For ease of deployment of your new BUI-enabled applications, your Web Services, and your desktop applications on Web Start, we include the Jetty Web Server with every installation of BBj. BASIS also delivers, *for no extra fee*, application building blocks, utilities, and sample code for emailing, faxing, language translations, geolocation determination, signature capture, patch-deployment, integration with WYSIWYG-designed reports, Google Docs integration, a customizable installer, and data replication. The data dictionary-based RAD tool, Barista®, is continually being enhanced and now includes a notable new multi-table SQL-based Query Design feature.

The list doesn't stop here. The AddonSoftware® ERP modules are the customizable functional application building blocks for vertical market development. BASIS customers can jumpstart the process of creating and refining applications in their area of expertise. Building applications upon the foundation of proven accounting modules means that developers eliminate the need to 're-invent the wheel' and can invest their resources in coding what they know best. This edition of *The BASIS Advantage* includes articles on many of these features and functions.

How does this all fit into our strategy and yours? In a nutshell, we strive to simplify your lives while future-proofing your applications. Our strategy is to enable you, our partners and customers, to be more efficient in delivering your solutions to your end users. BASIS is adding more and more features and components to our eco-system so that you can focus on your specialized knowledge while knowing that whatever the future brings, BASIS will ensure that you will be technologically-enabled to meet the challenges and leverage the opportunities, be they on the server or client, desktop or mobile, terra firma or in the Cloud. ■

Nico Spence
Chairman & CEO

Editor in Chief *Nico Spence*
nspence@basis.com

Editor *Susan Darling*
sdarling@basis.com

Technical Editors *Dr. Kevin King, Nick Decker*
kking@basis.com, ndecker@basis.com

Copy Editor *Peggy Lewis*
plewis@basis.com

Art Director, Graphics *Patricia Catlett*
pcatlett@basis.com

Electronic Production *Amer Child*
achild@basis.com

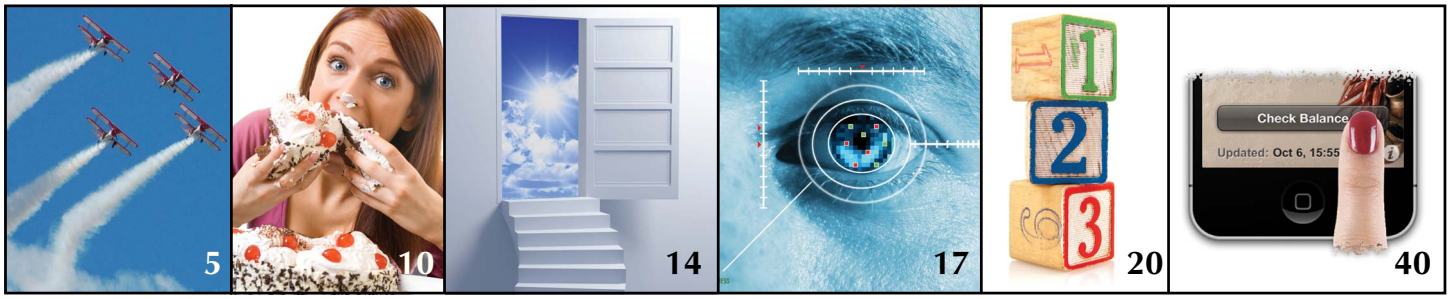
The *BASIS International Advantage* magazine is published and distributed by BASIS International Ltd.

BASIS does not endorse any products mentioned in the *BASIS International Advantage* other than those products licensed from BASIS International Ltd.

Subscribe at links.basis.com/subscribe

BASIS International Ltd.
5901 Jefferson Street NE
Albuquerque, NM 87109-3432

Phone +1.505.345.5232
US Sales 1.800.423.1394
International +1.505.338.4188
www.basis.com
info@basis.com



Columns

BUI Takes Flight

BBj®'s Browser User Interface (BUI) continues to generate excitement, and for good reason: BUI lets you bring your new or existing graphical BBj applications to the Web with *little or no additional development effort.*

BUI refers to BBj's integration with the Google Web Toolkit (GWT). The GWT is an open source development tool that cross-compiles Java-based code into highly optimized JavaScript to run on all major browsers. With BUI, developers can continue to build and maintain GUI applications in AppBuilder, the Barista® Application Framework, or directly in BBj, and then deploy them in a variety of browsers or other web-capable devices such as tablets and smartphones. And unlike the traditional thin client or Web Start architecture where the client must have a JVM installed on the desktop, BUI applications only require a JavaScript-enabled web browser, which means to say that it is available for most any desktop and most any mobile device.

The ERP (Enterprise Resource Planning) package, AddonSoftware®, is a great example of a complex graphical BBj application – developed entirely in Barista and, which now thanks to BUI, can also run in browsers *without having to write any additional code*! This article highlights the look and feel of a Barista application running in BUI and provides information for BUI configuration in Enterprise Manager.

Figure 1 shows the Barista MDI (multiple document interface) running AddonSoftware > >

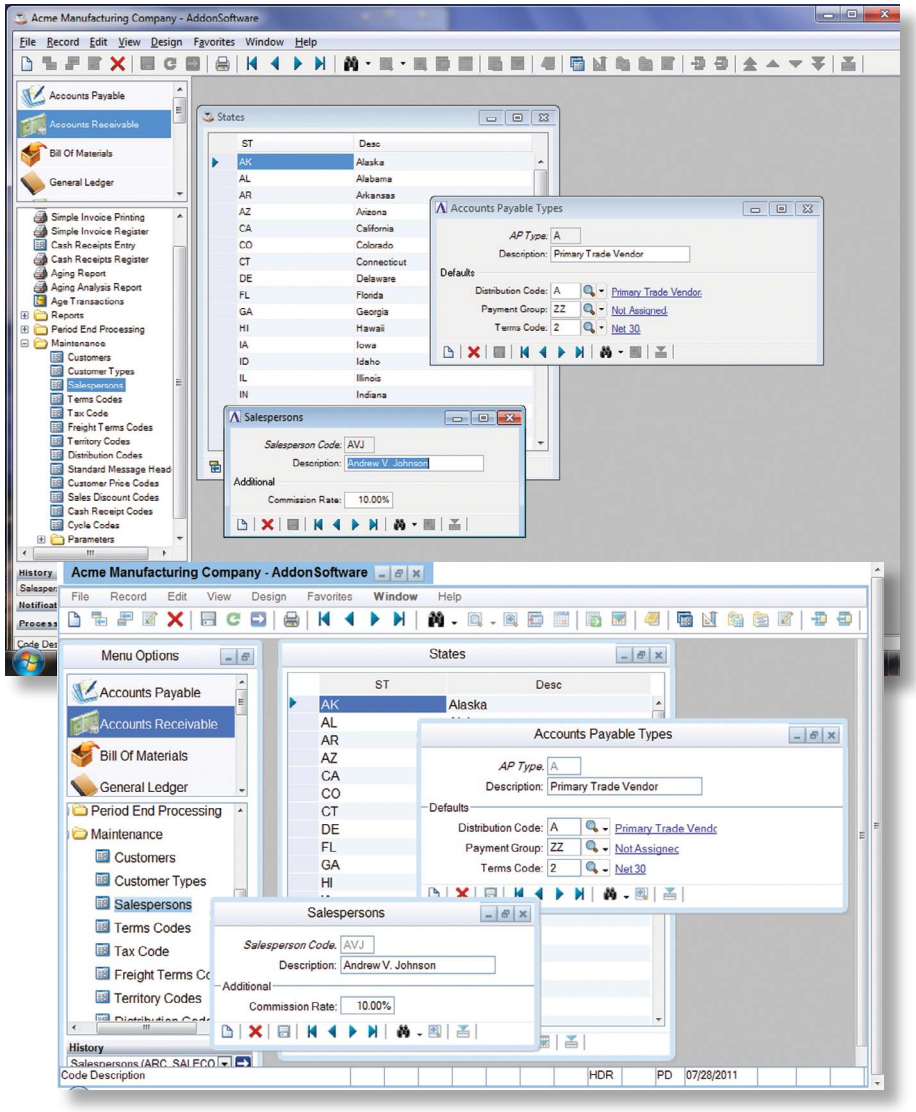


Figure 1. AddonSoftware running in GUI (top) and BUI with Google Chrome (bottom)

By Chris Hawkins
Software Developer

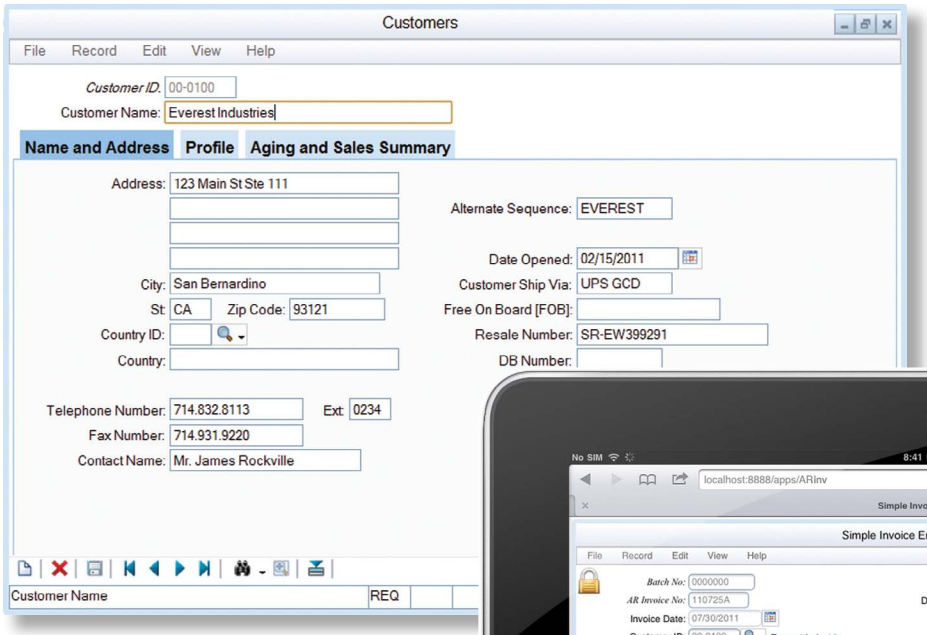


Figure 2. AddonSoftware Customer Master form in BUI using SDI

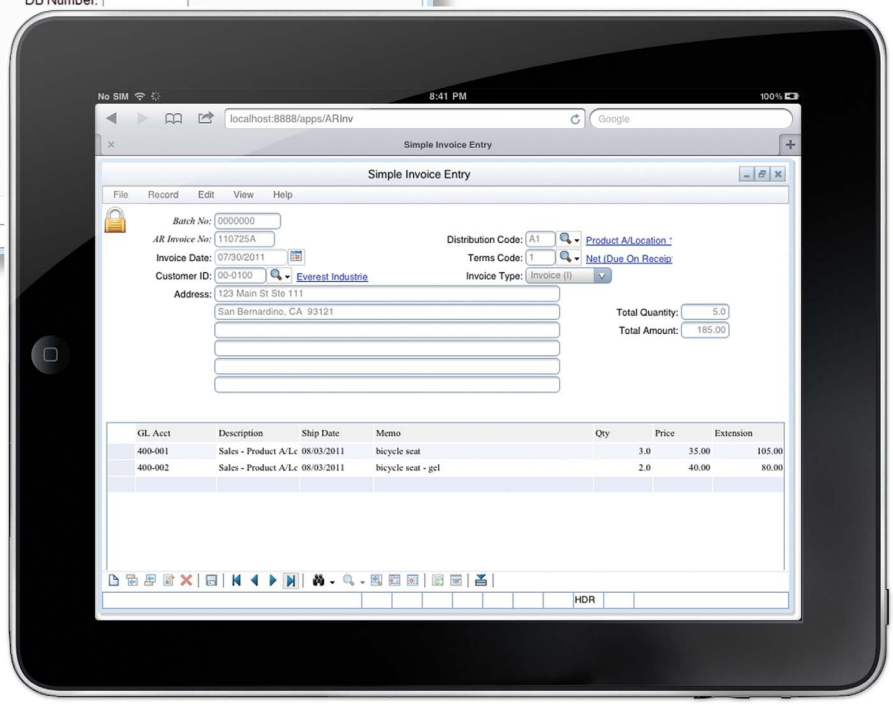


Figure 3. AddonSoftware Accounts Receivable Invoice Entry in BUI using SDI mode on a tablet

in GUI and BUI. The BUI functionality is very much the same as in GUI, but the look and feel in BUI corresponds to the browser. As with GUI, the MDI can contain multiple forms, the Barista menu and toolbuttons are context-sensitive, and hyperlinks provide one-click access to related forms.

Running individual tasks in BUI single document interface (SDI) mode, shown in **Figure 2**, is a great option when one does not need the entire MDI or is running on a smaller device such as a tablet or smartphone (**Figure 3**).

When it comes to reporting, BUI creates DocOut reports as PDFs, allowing local save or print options as shown in **Figure 4**.

And of course, sending documents as fax or email attachments, or saving them as Google Docs in the cloud, is as easy in BUI as it is in GUI using the DocOut output options.

Deploying the Barista application in BUI is simple, whether using the MDI or launching a single form in SDI mode. Enter the BUI Configuration information as shown in **Figure 5** to **> >**

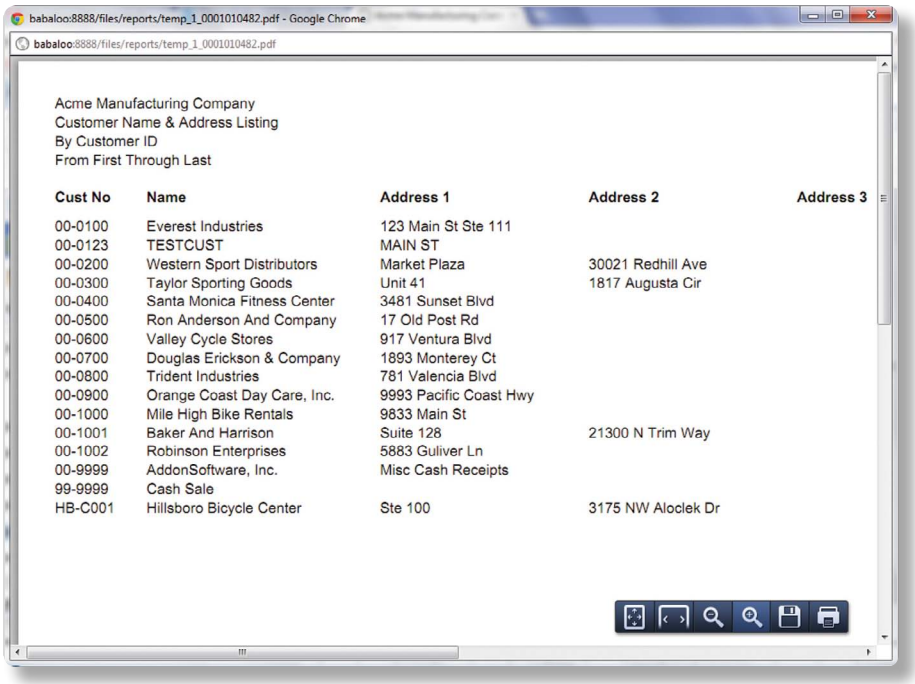


Figure 4. BUI generates DocOut documents in a PDF window

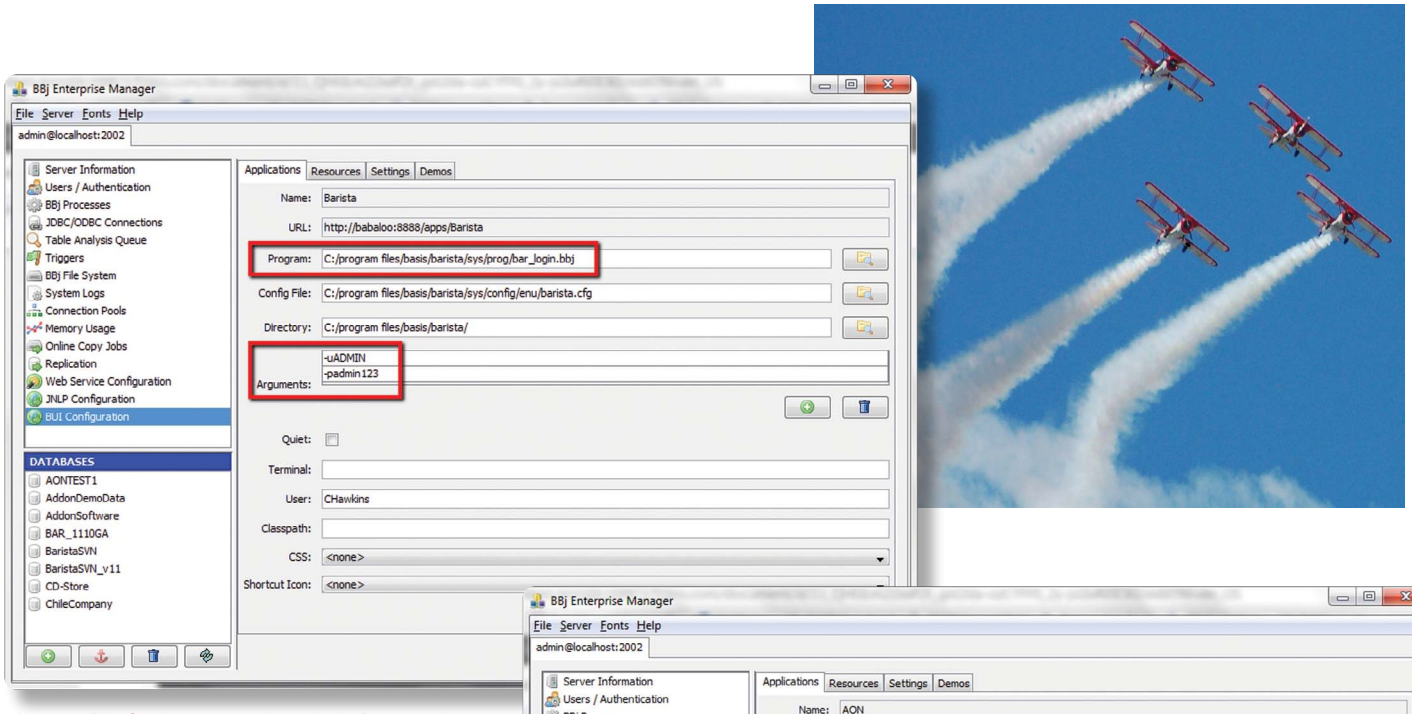


Figure 5. BUI Configuration to launch the Barista MDI

launch the Barista MDI for access to the entire application, or launch a single task using the setup in **Figure 6**. Figure 6 also shows how to use the special tokens [+BUI_USER] and [+BUI_PASSWORD], if desired, rather than entering the login and password directly in the configuration form. The tokens correspond to Barista Configuration records that carry the actual user credentials.

Alternatively, users can run a "quick start" in a BUI configuration to launch the Barista MDI directly from the Barista login form. Press [Settings], and select the checkbox to "Launch in a browser (BUI)" shown in **Figure 7**. This method creates a basic BUI configuration in Enterprise Manager if one doesn't already exist.

Figure 6. BUI Configuration to launch the AddonSoftware Customer Master form in SDI mode

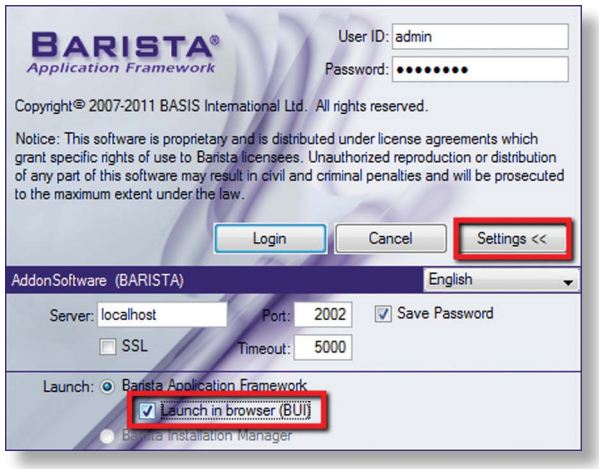


Figure 7. Launch Barista in BUI mode directly from the login form

Summary

BUI delivers graphical BBj applications to the Web – whether on a full screen, or other mobile device such as a tablet or smartphone – with just a few simple configuration steps and without having to code for different environments. Now develop your applications once, and deploy EVERYwhere in both GUI and BUI, saving time while bringing your application to the Web! BASIS brings new meaning to the phrase "zero deployment." ■

Rechanneling the Revenue Stream at MARK SYSTEMS

A A Revenue Stream Runs Through It spotlighted MARK SYSTEMS, Inc. (MS), a developer and distributor of a home builder management system, in the 2005 issue of *The BASIS International Advantage*. This article explained how MS spawned a flourishing revenue stream with a GUI release of their already successful software solution. In addition, their strategic move to a transaction-based pricing model introduced a low-risk point of entry for their prospects and consequently attracted many new customers.

At that time, the home building industry was strong and growing.

Life was good.

The Devastation

Then, in less than a year and without warning, the climate dramatically changed. Disturbing reports of the scorched home building market deluged the media (see **Figure 1**). Building was at a record low and devastated businesses spanned the nation.

The Resurgence

MARK SYSTEMS managed to survive this devastation and now, six years later, enjoy the biggest boom in their history.

> *How did they accomplish this feat amidst the devastating economic drought and destructive building market?*
>
> *How did they nurture a revenue stream that flourished in 2005 to one that has surged forward into a swollen stream during this brutal economic drought?*

Follow along with their key steps – **Survey the Ruins, Position for Survival, Remodel the Model** – to ultimately **Reap the Rewards**.

Survey the Ruins

As the building market began to crumble across the country, MARK SYSTEMS (MS) realized their survival depended on taking the initiative to help their customers through their struggles and in turn, secure their own future. While many builders literally closed their doors, others slashed the size of their staff in proportion to the reduced number of "house-starts" that dwindled from an average of 300+/yr to only 20-30. Similarly, those surviving home builders and MS alike took measures to change their processes to be more efficient and to "run a tighter ship."

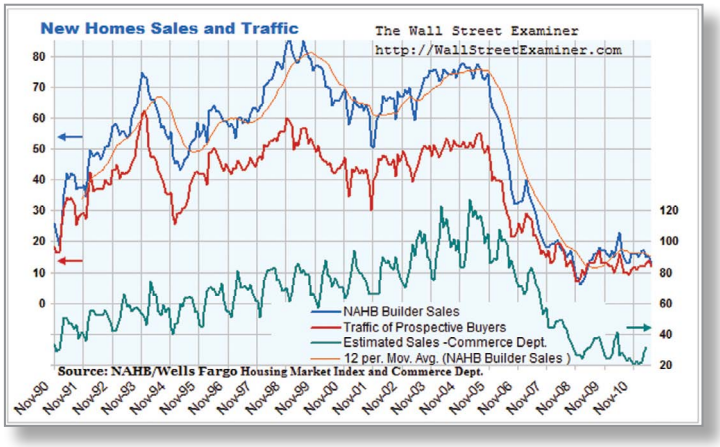


Figure 1. The scope of the building downturn

> How could MS continue to deliver their Integrated Homebuilder Management System (IHMS) as a viable business solution for the now greatly scaled-down builder while keeping the outlook bright for themselves?

Position for Survival

Without foreknowledge of the hard times to come but with a keen sense of market trends and business savvy, MS made some key decisions prior to this point in time that positioned them well for the downturn. >>

By Susan Darling
Technical/Marketing Writer

Licensing

Don Scattergood, principal of MS, recalls the logic of their move from the industry-common per-seat license structure, "*Our anticipation was that as companies grew, they would add more users. But what happened was, because the software was so efficient, customers were able to increase production capacity without adding people so they never ended up adding seats and we never made any more money. The other problem is that a seat-based licensing scheme front-loads a lot of revenue to us as a BASIS customer but also a lot of expense to the builder. We were trying to come up with a way of lowering that upfront cost, so we went with an annuity model where they pay for it by the month as they use it and license it for each house start. In other words, as they would buy door knobs and cabinets, they would purchase a software license for that house build.*" This licensing model was one key change that helped to buffer MS through the downturn.

Web Services

Also, years earlier just after BASIS released the BASIC Web Utility, MS developed and introduced the Internet ToolKit or ITK. This solution provided subcontractors and field personnel with Internet-based Job Starts, scheduling completion and Work Order authorization that update all back office information in real time. During the downturn, surviving builders relied even more heavily on such software tools for increased productivity and to take over where their dwindling staff left off. This was another key in buffering MS through the slump.

Remodel the Model

No one really knows how a 17th century quote morphed into today's adage "desperate times call for desperate measures," but this charge was certainly most applicable to the prevailing situation. While many builders in the throes of this dried up and diseased market clung onto this mode of survival, MS embraced the opportunity and, looking at their "glass half full," capitalized on what they had.

"*What we had was still the best single-source management solution in our industry, gaining major ground against our largest competitor that touted a 'best of brands' solution. This best of brands competitor bundled five different products to do what our IHMS' single and seamless product offered in one solution...one source, one point of contact, one database...one very well-oiled machine*," said Scattergood.

Beginning in 2008, when the market was still slithering towards its all-time low, MS concentrated their efforts in two key marketing areas; product and promotion.

Product - Move to BBj

First on the list was completing their planned move to the newest BBx® generation, BBj®, whose Java engine would help deliver new efficiencies, increase performance, drive significant cost reductions, and present a modern look and feel.

Thin Client

One of the biggest draws to BBj was thin client deployment. MS could now host IHMS on their own in-house server for their customers to access remotely. Delivering software as a service, or SaaS as it is commonly known, meant builders could run IHMSWeb on minimal hardware over the Internet while someone else maintained the server. That "someone else" was MS. Builders would no longer have to purchase multiple licenses and full software but could now pay for the IHMSWeb service on a monthly basis and only purchase a license for each house start.

Why is this such an attractive and viable option?

Existing customers upgrading from the Visual PRO/5® solution to the BBj version would likely have to upgrade their hardware to gain the full benefit of the new and faster functionality. As several MS customers have done, upgrading to the Web version of IHMS³ bought them time until they were able to purchase their own hardware and eventually install the application onsite. In some cases, however, customers have elected to stay on IHMSWeb as their permanent solution, forever leaving the hardware requirements and maintenance headaches in MS' hands.

For new builders starting fresh with IHMS³, Scott Duman, Vice President and CTO explains, "*Often they would say, 'We will have to delay the installation until after we are able to upgrade our hardware.' Now I can answer 'No problem, we can get you up and running tomorrow on our hosted version until you are ready to upgrade and install it onsite*.' " IHMSWeb helps reduce initial startup costs and the risk factor while delivering immediate benefits. Saving customers the overhead of hardware, software, and licenses creates a win-win situation for all and secured their current customers while drawing in many new ones. Builders could now more easily afford and justify this tool. IHMSWeb pay-for-what-you-use model for the building management software and BASIS license becomes a line item in the cost of building a home, just like the concrete package, framing, HVAC, licensing and permits, and so on.

Web Utility

The ITK was even more viable now than ever before for subcontractors and field personnel as the increase in productivity kept many builders lean. With the growing popularity of mobile devices and the rise in use of Apple products - iPad and iPhone - MS saw the niche and recognized the need to include full support for Apple products in IHMS³.

In August 2010, MS announced the release of full iPad integration for the IHMS Executive Vision strategic dashboard and the ITK Home Builder Field Portal. Superintendents in the field now have instant access to every schedule, job start, change order, purchase document, plan and drawing for every home in their communities. Equally important is the homebuyer's experience as they can now view >>

available option selections dynamically, including full images and specifications, and can select options and configurations anytime, from anywhere.

Forms

While the new Java-powered BASIS UI-engine brought a contemporary look and feel to IHMS[3], the printed forms still were reminders of the CUI past and were in great need of an overhaul to equal the slick look and feel of the new screen interface. Using their own custom extensions to the BASIS BBjForm object, MS redesigned all the reports to give them a fresh new look and feel. Users who may want to generate different reports can easily create them with iReports or other ODBC or JDBC compliant reporting tools to suit their exact needs. **Figure 2** shows both a sample report in legacy "CUI" look and feel and the redesigned beautified version.

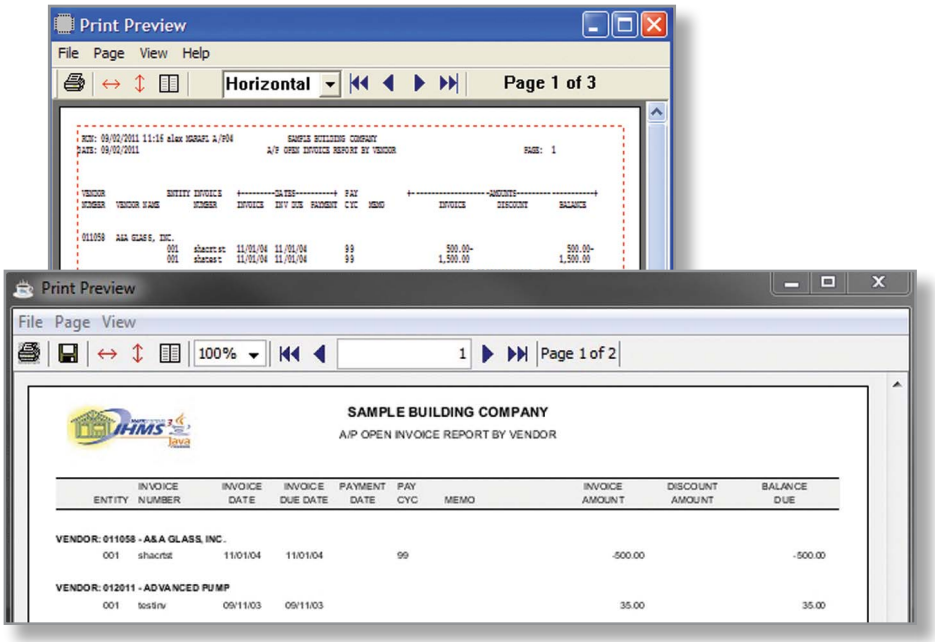


Figure 2. Before and after CUI/GUI views of a sample MARK SYSTEMS report

Product - Sell Efficiency and Productivity

Given the current times, MARK SYSTEMS knew the builders had to change the way they did business too. As builders dramatically trimmed their staff and resources, it became paramount for them to also trim the excess in their operations and processes. Efficiency and accuracy became key priorities to the builders, so tools like IHMS that could deliver that need and help keep the builder on track became a necessary next step in ensuring the builders' survival.

Another very powerful tool that MARK SYSTEMS distributes is the IHMS InternetToolKit Mobile Web Portal. Linked directly into a builder's existing website, the ToolKit's mobile access creates dynamic, personalized "home pages" for each field employee. From their page, superintendents can review Job Starts and Change Orders, request VPO's, review and update schedules - virtually every piece of construction-related information. With the introduction of mobile devices for such web apps, productivity now extends into the field.

Promotion - Make the Product Visible

Next, were some much-needed revisions in sales strategies and marketing efforts for MS to take a bigger leap into the market and make their website, and therefore their product, more visible and palpable for their customers and prospects, ensuring that all-important 'stickiness' that is the mark of great websites.

Redirect Marketing Efforts

The greatest challenge was to get the word out more effectively about this enhanced award-winning product. IHMS[3] and its IHMSWeb sister are now Java-powered, have a modern look and feel, are faster than ever before, and have a powerful Internet portal. In response to their tradition of "continual improvement based upon customer feedback," MS included in the new release "tighter controls on option selection and emails to the sales manager when prospects register from the Website, as well as updated features for job costing and purchasing, sales and warranty, and accounts payable and system administration, among others."

Prospects weren't exactly knocking at their door so they concentrated on changing their marketing efforts by redesigning their website as the Internet is now the most relied-upon tool for "shopping." It was time to look at virtual efforts rather than traditional methods of mailings, referrals, etc. Their primary goals were to increase their search engine visibility and to make it much more usable for their prospects and customers.

To increase the search engine visibility, Scattergood explains. "*We focused on content, keyword density and third party links. We started out by identifying the keywords we were interested in, particularly the "long tail" keywords - 'integrated home builder accounting software' instead of 'construction software', for example. Then we re-wrote every page of content to 1) include more content, and 2) include the keyword phrases as part of that content. We also added keyword phrases to alt-text on every image, and made the image and page names descriptive from a keywords perspective, For example, we changed: page2.html to home_builder_sales_software.html, .*"

In addition to these changes, MS also added images and videos to the website. This was rather "out of the industry norm" as competitors possibly feared exposing their trade secrets. But any risk was worth it to MS. Google likes images and videos in the content (such as the video at links.basis.com/ihms3) so by just adding these elements, MS moved up to 1st position after the paid ads in the resulting search on the long-tail keywords 'integrated home builder accounting software' (see **Figure 3**). This addition also resulted in MS appearing on page 1 instead of page 10 when searching on major keywords!

There are still many more enhancements MS would like to make to their website to take them to the next level but already by shifting heavier efforts to web-based marketing, they are experiencing builders coming to them, up to five in-bound requests a week! > >

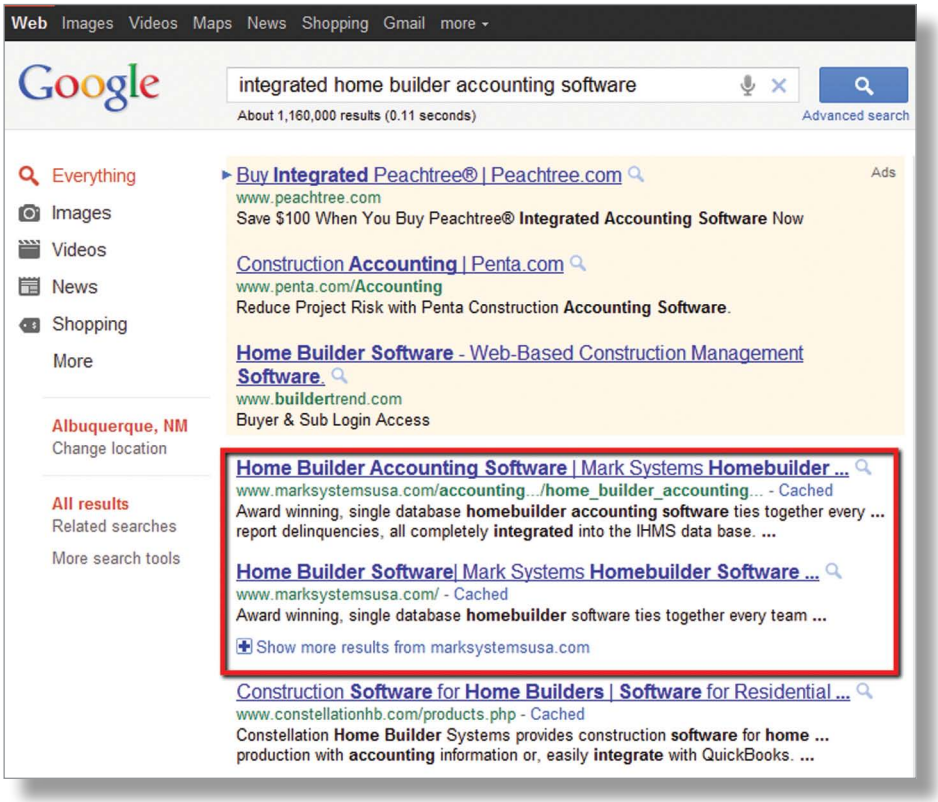


Figure 3. Long-tail keyword search lists MARK SYSTEMS first after ads

Figure 4. MARK SYSTEMS expanded Web page with press releases, videos, expanded search

Marketing - Add More Value

Provide more content

While adding content to their site helped optimize key word searches, it also delivered more useful information to both current customers and the prospects (see **Figure 4**). Today, people rely heavily on the Internet to search for answers to questions, research new business solutions, and learn about new updates and enhancements. MS chose to publicize their milestones with press releases to not only trumpet their accomplishments but to also provide a searchable and public archive of this valuable information.

Provide more video training

Getting the software sold and installed is only part of the process. Real success and productivity is directly dependent on "the person in the cockpit." MS conducts regular Web-based training but saw an opportunity to capture some of the training experience in online videos.

Reap the Rewards

As a result of their change in licensing from several years ago, the introduction of the ITK, their move to BBj, and introduction of IHMSWeb, to name a few, MS looks back on 2010 as the best year ever in unit sales and dollar sales volume. They saw a 172% increase in total units and 211% rise in total dollars over the combined sales of 2008 and 2009, eclipsing the previous sales record in 2006 by more than 19%. To top it off, they ended the year with the largest sale in their history to a customer who builds homes in 50 communities across five states.

Build References - Share the Success

Moving to IHMS gave one new five-state builder the opportunity to remodel their own business practices along with implementing the new software. With these two changes working hand-in-hand, supervisors in the field reported a three-hour a day increase in productivity! And if that wasn't exciting enough, those supervisors using iPads to access their data and run the ITK from the field realized increased productivity of up to five hours per day. Astonishing? Yes, but considering MS has spent years blending their expertise and power of BBj with field experience from their customers, their vertical application delivers built-in efficiencies that are refined and proven. This new customer realized the great wealth of business process intelligence they were inheriting so chose to invest in it >>

for their business. Streamlining their processes with a fully integrated single solution with a single database versus a loosely coupled best-of-breed approach with multiple databases paid off in exponential proportions for themselves and MS alike.

Prepare for the Future

What is downstream for MS? Their landscape is filled with significant enhancements. Leading the pack is a move already in progress to BASIS' BUI to open up seamless deployment of their application for the mobile device. **Figure 5** shows an example of their existing BWU application for ITK with an example of the BUI version embedded in the screenshot.

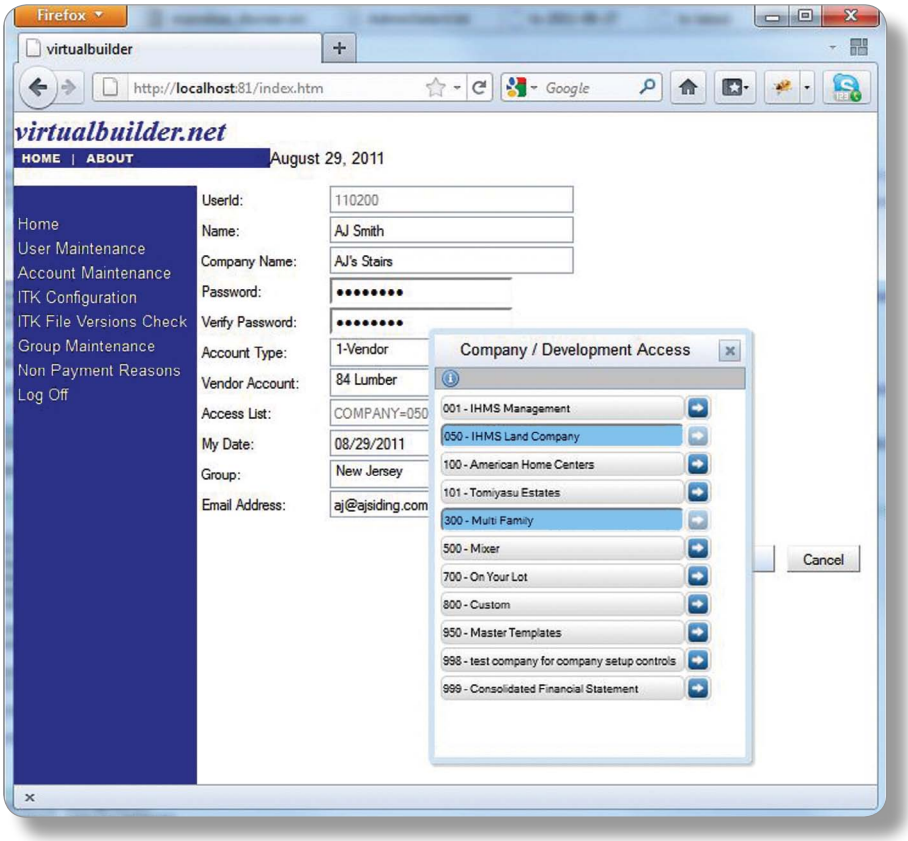


Figure 5. The a sneak peek at their move from BWU to BUI

Next is the integration with Google to provide support for their mail and calendar applications, and integrating geolocation functionality. With the success of hosting IHMS on their own servers, they look forward to hosting their application on the cloud. BBj opened the floodgates to the reality of easily adding these important capabilities to MS products and expect these extensions will likely blow their competition out of the water. ■

MARK SYSTEMS, Inc. develops and distributes integrated technology solutions for the home building industry. Over 30 years ago, MARK SYSTEMS began designing their Integrated Homebuilder Management System (IHMS) to adhere to National Association of Home Builders' best homebuilding practices. IHMS, the industry's only single database enterprise software package, offers full support for back office and field operations and provides state-of-the-art information processing tools for the homebuilder. Today's versions, IHMS^3 and its hosted version IHMSWeb, are the results of continual improvement based upon customer feedback, migration to the Java-powered BBj, and access via the Internet.

Once again, Construtech honored MARK SYSTEMS in this year's "50 Hottest" most influential construction technology providers (www.constructech.com/awards), adding to previous honors in 2009 and 2010, and multiple "Top Product" and "Vision" awards dating back to 2008.

MARKSYSTEMS Celebrating 30 Years 1981-2011

Visit MARK SYSTEMS at www.marksystemsusa.com

Editor's note: This is a prime real-life example of the results of employing the 4Ps and 4Rs. MARK SYSTEMS was spot on in their attack of "P"roduct, "P"rice, "P"lace (convenience), and "P"romotion, with specific refactoring of Product, Price and Promotion. But more importantly, with regards to a knowledge-based product such as software, paying attention to the 4Rs was essential. MARK SYSTEMS removed the "R"isk factor, improved the "R"ealization of performance, used "R"elationship marketing, and leveraged their strong base of "R"ecommendations and references. If you are not minding your Ps and Rs or would like more assistance in doing so, contact your BASIS account manager to set up a brainstorming session. Your success is essential for our success.

- Read the prelude to this story and details about their move from seat-based licensing at MARK SYSTEMS – A Revenue Stream Runs Through It
- Figure 1: Adler, Lee. The Wall Street Examiner | Be Prepared. Stay Ahead of the Herd. Web. wallstreetexaminer.com/2011/06/15/housing-market-tanks-again-say-builders

Parlez-BUI Français?

Localization is what we do to make our applications interact with users in their language and using currency, date, and other formats that are appropriate for their country and language. While there are some aspects of this process that can affect the back-end application such as currency conversions, localization is primarily user-focused. This article introduces the tools that BBj® and its browser user interface (BUI) provide to help build fully localized applications.

Language

The most obvious part of localization is language translation. The Barista® Application Framework login screen in **Figure 1** provides a convenient example of two languages.

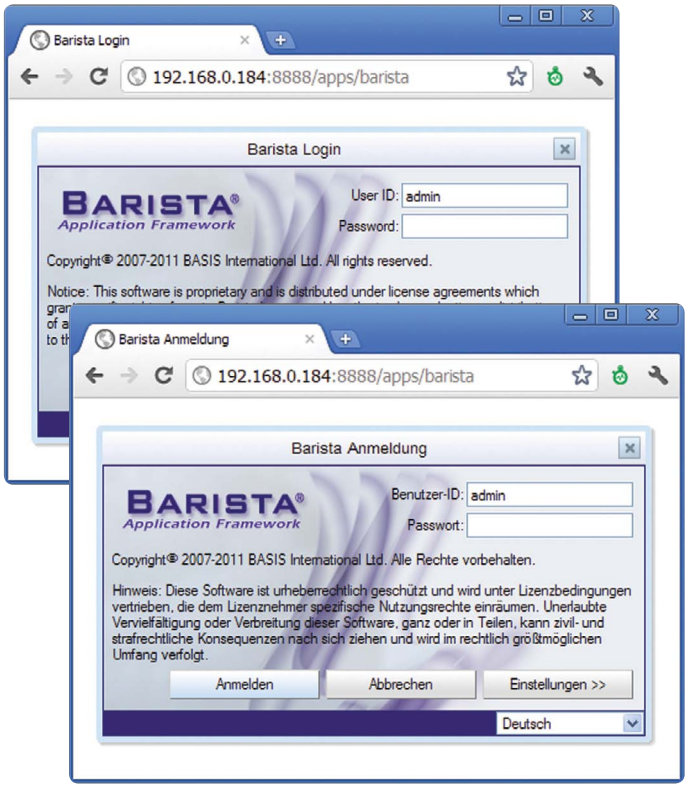


Figure 1. Barista login in English and German

Developers can implement language translation with a combination of approaches. For static screens stored in resource files, create multiple copies of each resource file, one for each supported language. If an application starts with a login screen, consider creating **`login_en.arc`**, **`login_de.arc`**, **`login_fr.arc`**, and so on. Dynamic text is stored in Java-style property resource bundles (refer to links.basis.com/fjizn). A property resource bundle is a set of text files containing key/value pairs for various locales. Files are named in the format **`login.properties`**, such as **`login_en.properties`**, **`login_de.properties`**, and so on. They contain key/value pairs like the samples shown in **Figure 2**. >>

Properties file	Property (key=value)
login.properties	**`userid=User ID:`** **`password=Password:`**
login_de.properties	**`userid=Benutzer-ID:`** **`password=Passwort:`**
login_fr.properties	**`userid=ID Utilisateur:`** **`password=Mot de passe:`**

Figure 2. Sample properties

By Jim Douglas
Software Developer

The BBTranslator utility can manipulate Java properties files. If those properties files exist in the current directory, retrieve the German text like this:

```
use ::bbtranslator.bbj::BBTranslator
Login! = BBTranslator.getInstance("login","de","",dsk("")+dir(""))
print login!.getTranslation("userid")
print login!.getTranslation("password")
```

```
>run
Benutzer-ID:
Passwort:
```

Non-English text will almost always contain text that cannot be expressed in simple ASCII. To ensure that properties files will work correctly on any system, use the native2ascii Java utility to convert text from a local character set to a cross-platform encoding format that will work with any character set.

Character Set

Applications that deal only with English, or run only on a single platform (like Microsoft Windows), often ignore character set issues, implicitly assuming that the character set is windows-1252. Applications might assume, for example, that the following code will always set euro$ equal to the euro (€) symbol:

```
euro$ = $80$
euro$ = "€"
```

The first line explicitly assumes the Windows-1252 character set, which encodes € as 80. The second line implicitly assumes the character set used when the program was created (probably Windows-1252 if the program was created on Windows). Both of these will fail when run on any other character set. For example, 80 maps to Ä in the Mac OS X Mac Roman character set. To ensure that an application will work correctly with any character set, one should rather code something like this:

```
euro$ = new String($20ac$,"UTF-16"); rem ' Unicode UTF-16 encoding
euro$ = new String($e282ac$,"UTF-8"); rem ' Unicode UTF-8 encoding
euro$ = new String($80$,"windows-1252"); rem ' Windows Codepage 1252
```

The windows-1252 version will probably work across all platforms; the Unicode versions are guaranteed to work.

It is often possible to use HTML encoding. For example, this will popup a msgbox() with the € symbol as the message:

```
print msgbox("<html>&euro;")
```

BASIS resource (`.arc`) files embed their character set as a specially formatted comment on the first line (e.g., `//#charset: windows-1252`), so they are inherently cross-platform-safe.

Server and Client Locales

Over time, BBx® has accumulated a range of features to allow for building localized applications. The oldest of these is SETOPTS byte 3, bit 20, combined with bytes 5 and 6, which can be used to localize numeric and currency formatting of the decimal point and the grouping separator. STBL("!DATE") can be used to customize date formats and text for the DATE() function. Those localization features were originally output-only; they were later extended to user input with the INPUTN and INPUTD GUI controls. As the BBj version of the BBx language has evolved, it has incorporated the Java Locale as STBL("!LOCALE"). A Locale is a one-, two- or three-part code used to identify a language, language+country, or language+country+variant. Language codes are lowercase two-letter ISO 639-1 codes. Country codes are uppercase two-letter ISO 3166-1 codes. Variant codes are added when >>

additional subdivisions beyond language+country are needed. When STBL("!LOCALE") is set, STBL("!DATE"), MSGBOX() button text, the print preview user interface, and the internationalized currency masking values are all updated to correspond with the specified locale (run the demo or try the sample code referenced at the end of this article). BBj supports these seven language codes:

Code	Language	Locales
de	Deutsch (German)	de_AT (Austria), de_CH (Switzerland), de_DE (Germany), de_LU (Luxembourg)
en	English	en_AU (Australia), en_CA (Canada), en_GB (Great Britain), en_IE (Ireland), en_IN (India), en_MT (Malta), en_NZ (New Zealand), en_PH (Philippines), en_SG (Singapore), en_US (United States), en_ZA (South Africa)
es	Español (Spanish)	es_AR (Argentina), es_BO (Bolivia), es_CL (Chile), es_CO (Colombia), es_CR (Costa Rica), es_DO (Dominican Republic), es_EC (Ecuador), es_ES (Spain), es_GT (Guatemala), es_HN (Honduras), es_MX (Mexico), es_NI (Nicaragua), es_PA (Panama), es_PE (Peru), es_PR (Puerto Rico), es_PY (Paraguay), es_SV (El Salvador), es_US (United States), es_UY (Uruguay), es_VE (Venezuela)
fr	Français (French)	fr_BE (Belgium), fr_CA (Canada), fr_CH (Switzerland), fr_FR (France), fr_LU (Luxembourg)
it	Italiano (Italian)	it_CH (Switzerland), it_IT (Italy)
nl	Nederlands (Dutch)	nl_BE (Belgium), nl_NL (Netherlands)
sv	Svenska (Swedish)	sv_SE (Sweden)

Figure 3. Supported languages and locales

BBj has always been a client/server system, with potentially different locales on the client and the server. In BBj 11.0 and above, the BBjThinClient::getClientLocale() function returns the client locale from the GUI or BUI client. The BUI client locale is always set to one of the BBj supported language codes listed in **Figure 3**. If the browser is configured for one of those values, it is used as the default, but the default can also be overridden by specifying a locale value in the URL (e.g.: `?locale=de_DE`). The client locale determines the language that will be used for assorted end-user messages, including the optional User Authentication login screen and text on the various chooser controls.

Example

The Localization.txt sample program, located at links.basis.com/11samples, demonstrates a way to implement user input that automatically adjusts to the client locale. It sets the BBj server-side STBL("!LOCALE") to match the client locale, and it generates BBj INPUTN numeric masks based on the standard currency format for that locale. **Figure 4** shows the program running with different locales, such as en_GB displaying United Kingdom.

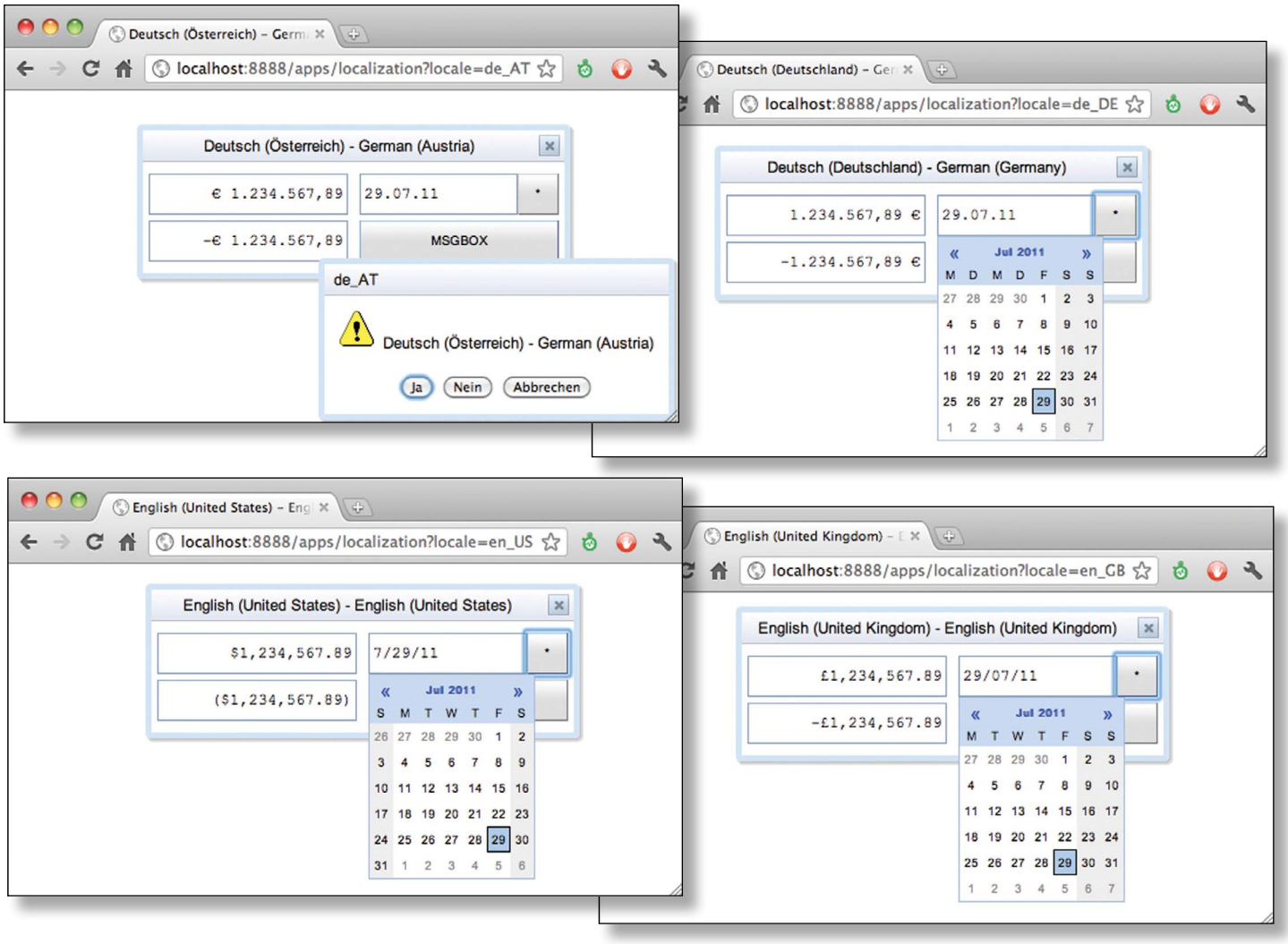


Figure 4. Localization demos in German (top) using Austrian and German locales and English (bottom) using Great Britain and United States locales

The Past Through Tomorrow

Our company name, BASIS International Ltd., has always been a recognition that software is global. With the continuing rapid growth of the Internet, particularly the mobile Internet, this is more true now than ever before. BASIS' BUI technology offers a seamless path to get your application up and running in web browsers all over the world. BBj and BUI make localization painless, with functionality and tools that simplify the process. Barista, BASIS' data dictionary-driven development framework and runtime engine, leads by example and runs in multiple languages. Get started on localizing your app today to expand your potential customer base all over the world! ■

- Download and launch the msgbox.txt sample program that runs the msgboxdemo
- Read Can Your App Speak to Your Customer? for other BASIS language translation utilities

New Barista Features Satiate Appetites

BASIS has been continually enhancing the Barista® Application Framework since its initial release in 2007. The version 11.x releases continue that track record, including more exciting features demonstrating BASIS' ongoing commitment to providing leading-edge cross-platform rapid application development tools.

Query Definition System

The new Query Definition System leverages enhancements made to the BASIS SQL engine and dramatically harnesses its power to remove single table limitations of the existing Barista query capabilities. In previous releases, creating a table definition automatically generated single table queries, supplying an immediate query with no additional work. The new query system shown in **Figure 1** provides a multitude of features for developers wanting more control over their data presentation.

One of the most important new features is the ability to join information from multiple tables into a single query. Developers can include columns from an unlimited number of tables related by primary or foreign keys. They can also sort data concurrently on one or more table-based columns in a specific case or case-neutral sequence.

By Sam Vulopas
Senior Software Developer

The query system also allows customized SQL SELECT statements. Barista creates the initial SQL statement based on the defined columns and allows developers to manually modify and fine-tune the query, including nested SQL select statements and stored procedure calls to generate a result set. A new option – specify custom filtering and data manipulation programs – allows users to produce finely detailed queries. >>

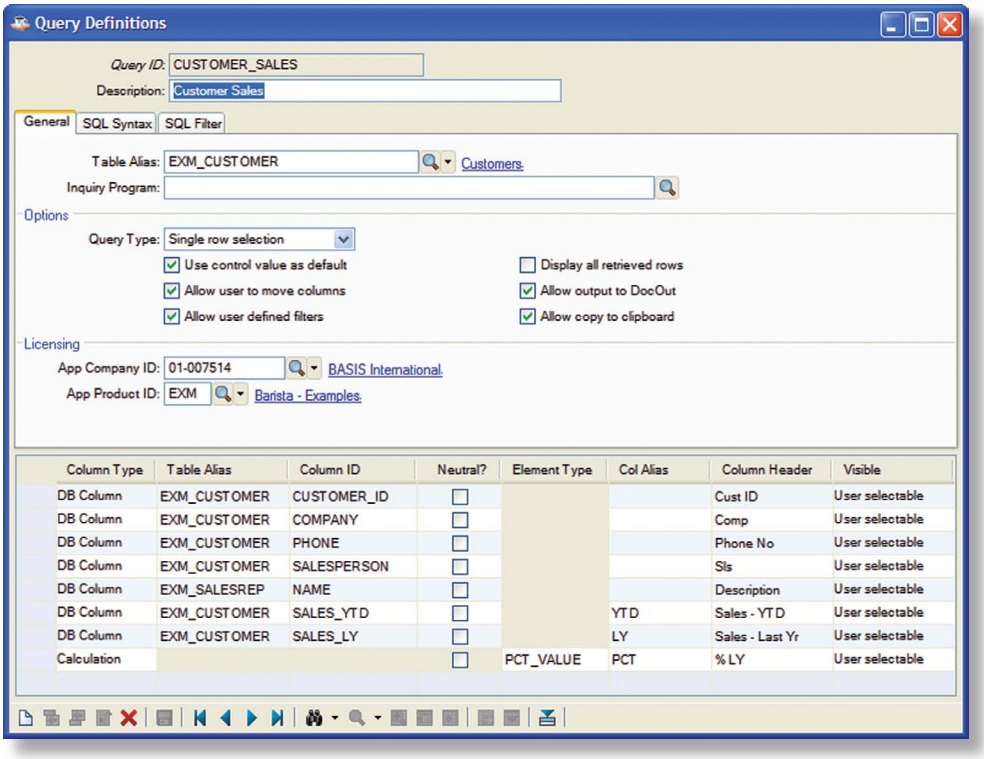


Figure 1. Query Definition Maintenance

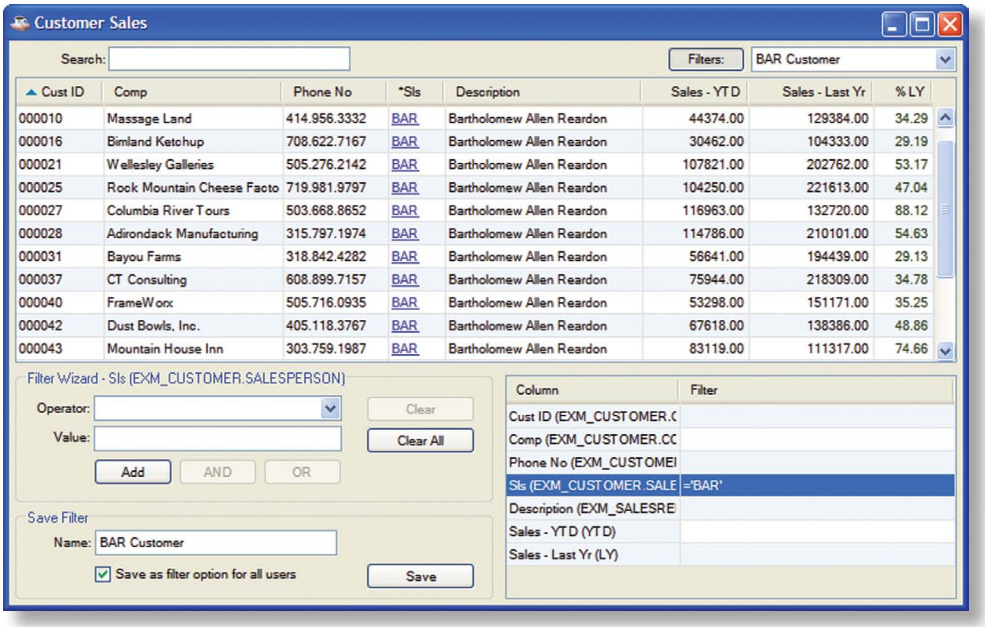


Figure 2. Defined query with inline filtering, hyperlinks, and a calculated %LY column

Version 11.10 also contains the ability to create new columns derived from calculations based on column data returned from the original query, and to total columns. With just a few clicks, this powerful feature creates displays containing detailed drilldown information with derived columns and totals.

The new query display maintains a consistent look and feel with the current Barista inquiry system, but contains enhancements that greatly improve filtering and data linking. Filtering is now accomplished "inline" via the Filter Wizard for beginning users, or for experienced users, through direct entry of filter criteria. The result set is immediately filtered and redisplayed, providing instant feedback. Columns linked to other tables via foreign keys are now hyperlinked to provide easy single-click access to master record data. See an example of these features in **Figure 2**.

Query Document Output

The second major set of enhancements also involves all Barista query systems and their ability to send a result set to the Barista Document Output System (DocOut). After refining the query via filtering and sorting, users can launch the "Additional Options" menu and

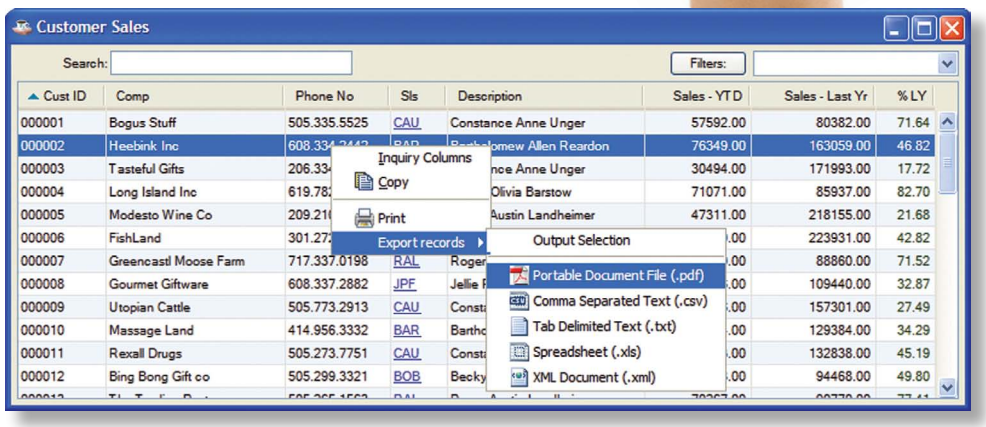


Figure 3. Query display with available "Export records" selections

select one of many options for exporting the data in the exact sequence as currently displayed. Options include the DocOut viewer and its various selections – printing, email, fax – and a number of supported file types (.pdf, .csv, .txt, .xml, and native .xls spreadsheet). In addition, five one-step export options immediately create and launch the requested file type in its associated application (see **Figure 3**).

Summary

Queries are an extremely important part of any graphical application. These recent product enhancements provide the tools and options to give developers greater control over the design of their applications. Perhaps most important are the huge time savings and productivity improvements these enhancements bring to the application developer. The never-ending and seemingly insatiable appetite that users have for seeing their data presented with different filters and in different formats can now be satisfied immediately by the developer or power user with little time and therefore extremely low cost, leading to satisfied customers and happy productive developers! ■

Read the white paper *Query Definition System*

The Beauty of Charts and Graphs in BUI

From GUI to BUI: BBjChart

BBj® implements a rich API for constructing graphical charts with the BBjChart control, which is built on top of the JFreeChart API. The BBjChart control presents a unique challenge for BUI – it exposes implementation details that depend on the assumption that the JFreeChart API is running in a Java Runtime Environment on the client. But BUI doesn't run in a client-side JRE; in fact, it runs on platforms (like iOS) that don't support Java at all.

To make BBjBarChart, BBjLineChart and BBjPieChart work in BUI, we moved the guts of the implementation from the client to the web server. We monitor all changes that you make to your chart, and every time we see a change or a batch of changes, followed by no changes for 250 milliseconds, we transparently generate an updated snapshot of the chart as an image and push that image over to the client. Using this strategy, most existing BBjChart programs automatically work in BUI as of BBj 11.10. **Figure 1** shows a BBjChart sample program running in GUI, and one in BUI on the deskop. Download and run this sample at links.basis.com/11samples.

In the BBjChart samples in shown **Figure 1**, you can see that only three of the four samples work in BUI. The fourth chart uses functionality that is not available in BUI. BBjGenericChart, setClientChart and getClientChart are not implemented in BUI because they depend on the infrastructure for Java client objects. But, with some minor adjustments, it is still possible to work with the full JFreeChart API in a BBj program that will run in BUI. The solution is to use the same basic strategy that BBj BUI itself uses – build the chart on the server, then push the rendered chart over to the client as an image. **Figure 2** shows this conversion process for a representative BBjGenericChart.

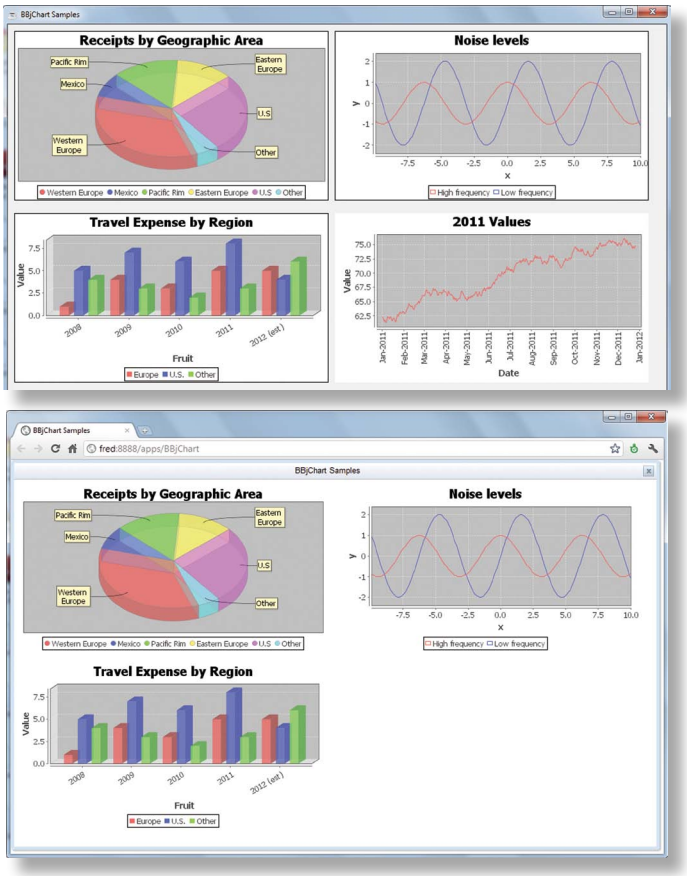


Figure 1. Four BBjCharts in GUI (top) and three charts in BUI desktop (bottom)

Figures 3 shows the results of the converted JFreeChart sample program. Note that the sample program uses randomly generated data each time it's invoked so the resulting charts are not identical. Download and run this sample located at links.basis.com/11samples. >>

By Jim Douglas
Software Developer

BBjGenericChart

```
series! = new TimeSeries@("Random Data")
current! = new org.jfree.data.time.Day@(1,1,2011)
value = 62
for i = 0 to 360
 delta = i/60
 value = value + Math.random() - .5
 series!.add(current!,new Double(value + delta))
 current! = current!.next()
next i
dataset! = new TimeSeriesCollection@(series!)

chart! = ChartFactory@.createTimeSeriesChart("2011
Values","Date","Value",dataset!,0,0,0)

list! = bbjapi().makeVector()
list!.add(new Float(10).floatValue())
list!.add(new Float(1).floatValue())
CAP_ROUND = BasicStroke.CAP_ROUND
JOIN_ROUND = BasicStroke.JOIN_ROUND
stroke! = new
BasicStroke@(1,CAP_ROUND,JOIN_ROUND,5,list!,0)

renderer! = new XYLineAndShapeRenderer@()
renderer!.setBaseShapesVisible(0)
renderer!.setSeriesStroke(0,stroke!)
plot! = chart!.getPlot()
plot!.setRenderer(renderer!)

format! = new SimpleDateFormat@("MMM-yyyy")
tickUnit! = new
DateTickUnit@(DateTickUnit.MONTH,1,format!)
axis! = plot!.getDomainAxis()
axis!.setTickUnit(tickUnit!)
axis!.setVerticalTickLabels(1)

genericChart! =
window!.addGenericChart(104,520,300,500,270)
genericChart!.setToolTipText("BBjGenericChart")
genericChart!.setClientChart(chart!)

genericChart!.getClientChartPanel()
.setMouseZoomable(1,0)

out! = new ByteArrayOutputStream()
ChartUtilities.writeChartAsPNG(out!,chart!,500,270)
bbjimage! =
imageManager!.loadImageFromBytes(out!.toByteArray())
chartControl! =
window!.addImageCtrl(104,510,290,500,270,bbjimage!)
chartControl!.setToolTipText("JFreeChart
TimeSeriesChart")
```

JFreeChart

```
series! = new TimeSeries("Random Data")
current! = new org.jfree.data.time.Day(1,1,2011)
value = 62
for i = 0 to 360
  delta = i/60
  value = value + Math.random() - .5
  series!.add(current!,new Double(value + delta))
  current! = current!.next()
next i
dataset! = new TimeSeriesCollection(series!)

chart! = ChartFactory.createTimeSeriesChart("2011
Values","Date","Value",dataset!,0,0,0)

list! = bbjapi().makeVector()
list!.add(new Float(10).floatValue())
list!.add(new Float(1).floatValue())
CAP_ROUND = BasicStroke.CAP_ROUND
JOIN_ROUND = BasicStroke.JOIN_ROUND
stroke! = new
BasicStroke(1,CAP_ROUND,JOIN_ROUND,5,list!,0)

renderer! = new XYLineAndShapeRenderer()
renderer!.setBaseShapesVisible(0)
renderer!.setSeriesStroke(0,stroke!)
plot! = chart!.getPlot()
plot!.setRenderer(renderer!)

format! = new SimpleDateFormat("MMM-yyyy")
tickUnit! = new
DateTickUnit(DateTickUnit.MONTH,1,format!)
axis! = plot!.getDomainAxis()
axis!.setTickUnit(tickUnit!)
axis!.setVerticalTickLabels(1)

out! = new ByteArrayOutputStream()
ChartUtilities.writeChartAsPNG(out!,chart!,500,270)
bbjimage! =
imageManager!.loadImageFromBytes(out!.toByteArray())
chartControl! =
window!.addImageCtrl(104,510,290,500,270,bbjimage!)
chartControl!.setToolTipText("JFreeChart
TimeSeriesChart")
```

Figure 2. Converting the fourth chart, BBjGenericChart, to a JFreeChart for BUI presentation

Figure 3. All four BBjCharts with JFreeChart in BUI desktop and BUI iPad

>>

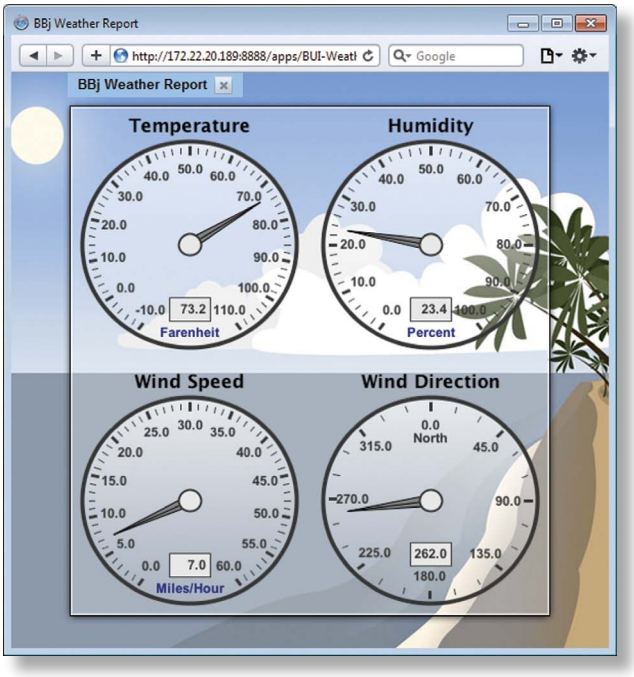


Figure 4. A BUI program using JFreeChart DialPlots

Applying those same kinds of changes to other JFreeChart-based programs works equally well. **Figure 4** shows the result of that modification to another program that uses JFreeChart DialPlots to show current weather conditions.

As BASIS' BUI technology matures, more and more of the traditional GUI functionality becomes available for use in the browser. Common controls such as buttons, listboxes, and input controls have been running well in BUI from the start, however, a few stragglers such as BBjCharts have not been available due to complexities with their underlying architecture. BASIS continues to chip away at the last few bits of GUI functionality not yet implemented in BUI; BBjCharts are the latest set of controls that BASIS recently modified to function in the BUI environment. Updates and enhancements are sure to come, so follow the ongoing progress in the "What's Implemented?" and "What's Not Yet Implemented?" lists in the *BBj BUI: Getting Started* document. ■

- For more information, refer to the online BBj*Chart documentation
- For features needed in the field that are not yet implemented in BUI, post a note on the BBj-Developer list at groups.google.com/group/bbj-developer/topics; subscribe at www.basis.com/discussion-forums

Partnership

AddonSoftware Steps Into the Cloud

This year's TechCon "tour," *Step into the Cloud with BASIS*, informed audiences around the world in Las Vegas, Nevada; Montreal, Canada; and Saarbrücken, Germany, of BASIS' own move to the cloud. Nearly all of BASIS' business functions – their custom version of the AddonSoftware® ERP solution, source code archives, build processes, test machines as well as company email and documentation – all now run in the cloud in data centers scattered across the United States and provide a local cloud frontage presence around the globe.

Introducing...

Using first-hand cloud experience as the template, BASIS introduces AddonSoftware Cloud Services, the software-as-a-service (SaaS) version of AddonSoftware by Barista®. This new service presents BASIS partners with a quicker and simpler way to deploy the AddonSoftware Enterprise Resource Planning (ERP) solution. BASIS prides itself on offering choices and flexibility to its partner community and is happy to extend this core value. Now AddonSoftware can run on-premise in the traditional way or online through an external host in a cloud environment...a single solution with deployment options to meet the business needs of developers and end users alike.

By Paul Yeomans
Vertical Market Account Manager

Strategically, the new cloud offering makes AddonSoftware more affordable to use, own, and deploy as illustrated in **Figure 1**. Affordability is achieved both through the low user-based monthly subscription model and the elimination of the typically large capital outlays for hardware, installation, and maintenance. Installations are quicker and there is no more guessing involved with purchasing a server that could be under- or over-sized. With the cloud, you can upgrade or downgrade at a moment's notice. Similarly, there is no concern that you'll be stuck running on last year's model; Moore's Law keeps you on the forefront of technology without the cost of continuous hardware investments. The immediate loss-of-value of purchasing a new car doesn't hold a candle to the immediate loss-of-value of purchasing computer hardware. The cloud is an infrastructure extension of the BASIS building blocks concept whereby software developers can remain focused on developing their vertical specialties while enjoying the benefits of a hosted solution. >>

	Cloud Services	On-Premise Client/Server
Pricing Model	Monthly subscription	License purchase or rental
Hardware	Included	Must purchase, maintain, and upgrade
OS Licenses	Included	Must purchase
Software Licenses	Included	Must purchase
Installation	No installation	Lengthy installation process
Annual Maintenance	Included	17% of AddonSoftware cost
Security, Patches, Backup Systems	Included	Customer responsibility

Figure 1. Comparison of AddonSoftware deployment in the cloud vs. on-premise

Including...

AddonSoftware Cloud Services includes all three ERP bundles; Accounting, Distribution, Manufacturing. The Cloud Service includes everything needed to get a company up and running quickly on AddonSoftware – a Cloud Server with data storage and nightly application and data backups, Amazon's world-class EC2 infrastructure, all AddonSoftware ERP bundles, BBj®; and to efficiently manage continued development, the Barista RAD tool.

Delivering...

Significant cost savings - The simple user-based subscription model provides a strong selling statement for upgrades or new sales. There are no large upfront costs - no hardware to purchase, install, or maintain. There is no need for complex hardware and software backup systems and their associated annual maintenance contracts.

Performance and reliability - The performance, reliability, and security are as good as, if not better than, a typical on-premise installation.

BASIS' experience, your benefit - BASIS' experience with Amazon Web Services (AWS) frees you to focus on application development while isolating developers from common hardware maintenance and failure scenarios. BASIS handles the creation of instance types, machine configuration, assigning operating systems, and the installation of AddonSoftware. Developers have unlimited access to the system to load databases, manage, configure, and install customized AddonSoftware.

Customization freedom - There are no restrictions to continuing your customization and solution development in the cloud.

Back-up service - Your data is secure. Nightly incremental backups (the same configuration BASIS itself uses) include data and application(s) on your entire persistent drive. Additionally, this service performs a full backup each week.

Replicate back-ups to other AWS Regions (Premium Service) - Duplicity is a good thing. Data and application(s) can be configured to restore to multiple geographic regions delivering an even higher level of recovery protection. These replicated backups are stored in Amazon's S3 (Simple Storage Service) ready to be deployed should they be required.

Summary

BASIS' experience moving their server functions, including their own customer version of AddonSoftware, to the cloud paved the way for the new AddonSoftware cloud offering. The cloud provides another deployment choice, either as an interim or final deployment step, wherein both end users and AddonSoftware partners benefit. And for the partner, this simple turnkey program with its affordable monthly costs, delivers further benefits to the end user while boosting the partner's value-added role in provisioning the complete business solution.

Opportunity is knocking so reach for the cloud! ■

Ready, Set, Go - Switch to IPv6

Many of you may have heard by now that the primary IPv4 (Internet Protocol version 4) address pool of the Internet was exhausted on February 3, 2011 and IPv6 (version 6) will succeed IPv4. BBj® and PRO/5® 11.0 are staying one step ahead with built-in support for IPv6, and as a result, most applications written in BBj and/or PRO/5 should be able to use IPv6 without any modifications. Read on for more details about what this all means.

What is IPv4?

An IPv4 address consists of four sections that contain a number in the range from 0 to 255. These numbers create an address that identifies a unique networked device, regardless of where in the world it is located. An example of an IPv4 address is **`107.20.236.91`**.

Since none of the four numbers can be any bigger than 255, there are 'only' about 4.3 billion possible combinations. While 4.3 billion IP addresses sounds like an enormous number, in reality it is not large enough for the ever-increasing number of global devices connected to the Internet. As a case in point, the Regional Internet Registry (RIR) that assigns IP addresses for the Asia Pacific region became the first region to run down to its last /8 CIDR block of IPv4 addresses and is now reserving its last set of IPv4 addresses for start ups.

What is IPv6?

The new IPv6 address is a 128-bit string divided into eight 16-bit segments, each segment consists of a hexadecimal number that ranges from 0 to ffff (6,535 in decimal). The number of possible combinations is 2^{128}, which is a staggeringly large 39-digit number. To put that into perspective, when BASIS received their allocation of IPv6 addresses, the range included more combinations than IPv4 addresses currently in use in the world! Wikipedia tries to put that large number in perspective a different way: "By comparison, this amounts to approximately $5x10^{28}$ addresses for each of the 6.8 billion people alive in 2010." An example of an IPv6 address is **`2001:4870:a1a5:1ff:643f:fd23:e8a5:3a05`**.

In addition to dramatically increasing the total number of IP addresses, IPv6 adds additional features beyond what IPv4 offers. Built-in network security, stateless address auto configuration, and network renumbering are a few of these enhancements that make IPv6 a significantly more advanced and capable Internet protocol.

By Dan Christman
Software Engineer

In the IPv6 address **`2406:da00:ff00::6b14:ec5b`**, the **`::`** is a shorthand notation for a center block that contains all zeros. The complete equivalent IPv6 address is **`2406:da00:ff00:0:0:0:6b14:ec5b`**.

Another interesting example is the IP address for localhost, a special address that always resolves to the local machine. In IPv4, **`127.0.0.1`** resolves to localhost and the IPv6 equivalent is simply **`::1`**.

Using IP Addresses in a BBx Program

BBx® programs may use IP addresses in a variety of circumstances to identify a remote machine with which to communicate. **Figure 1** shows one such use case - a program that makes a socket connection to www.basis.com and prints out its resolved IP address.

```
REM Assumes config.bbx contains an ALIAS N0 TCP
OPEN (1,MODE="host=www.basis.com,port=80")"N0"
PRINT KEY(1)
CLOSE (1)
END
```

Figure 1. Sample code opening a socket connection

This sample program opens a socket to www.basis.com and shows the IP address for the host machine. Since the URL www.basis.com is dual stacked having both an IPv4 and an IPv6 address, the result will depend on whether your Internet service provider and your Internet router supports IPv6. Running this program on a system that has an IPv6 connection to the Internet will cause something like **`2406:da00:ff00::6b14:ec5b`** to display, as both BBj and PRO/5 rev 11.0 will try to use IPv6 if available. Running it on a system that does not have an IPv6 connection to the Internet will result in something like **`107.20.236.91`** as both BBj and PRO/5 rev 11.0 will continue to use IPv4 when IPv6 is not available.

Another use case for IP addresses would be opening a file via a PRO/5 Data Server® and identifying the host machine via the IPv6 address.

For example, **`10 OPEN (1)"/<fd00::21,port=1100>myfile"`** opens a file called **`"myfile"`** on a data server located at **`fd00::21`**, which is an IPv6 address. It is written in shorthand notation for **`fd00:0:0:0:0:0:0:21`**. Any IPv6 address that begins with **`fd`** is a private IPv6 address similar to the IPv4 equivalent of **`192.168.x.x`**.

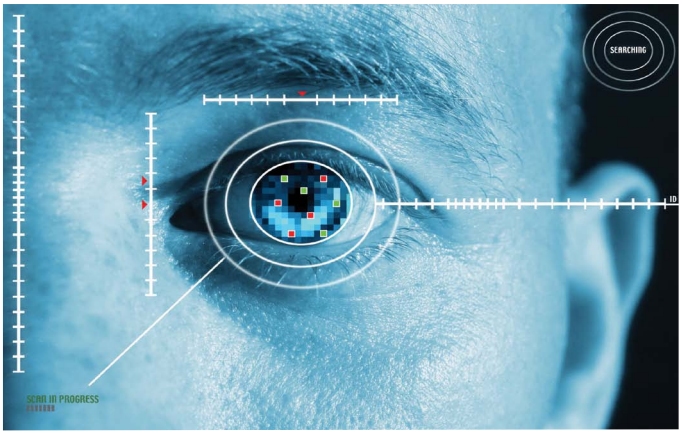
Summary

While it may take some time for everyone to switch over to IPv6, the writing is on the wall and the days of IPv4 are numbered. It won't be long before IPv6 will be required. BBx programmers should prepare for that day. By installing the latest versions of BBj and PRO/5, you and the business environments that you support can be assured of a smooth transition to the new technology. ■

DB Security That You Have Always Dreamed About

Security is, or should be, a major consideration for any database administrator. Depending on the requirements of each company, this could be as simple as requiring a user name and password to connect to the system, or as complex as different permissions for different types of SQL operations on various objects in the database for each individual user.

Standard permissions have been available in BBj® since version 1.0. These permissions give the administrator the ability to assign read-only or read/write access to an entire database. However, standard permissions did not permit assigning different permissions to different objects (tables, views, etc) in the database – it is all or nothing. While this is very easy to manage, it limits the control that the administrator has over access to sensitive data. This in turn could limit the ability for users to have easy desktop data query access to subsets of the company production database. Workarounds are possible but they require far more maintenance and structuring of the database by the database administrator.



BBj 11.0 now provides a complete feature set for managing user permissions in a powerful new feature called "Object Level Permissions." This new feature will, for many customers, unlock meaningful desktop data query access to corporate data for all the users within a company with appropriate restrictions that are easy to structure and maintain.

Object Level Permissions

The reference to "objects" in a database refers to tables, views, and stored procedures. Object Level Permissions gives the administrator the power to assign different permissions to different users or groups of users on varying objects in the database.

For example, a role or group of users called HUMAN_RESOURCES contains a list of all users in the HR department. This group of users might have access to information such as salaries, performance reviews, etc., that the company does not want everyone to access. The HR-specific tables would only grant permission to this role and any other individual users who might need the information. Anyone not explicitly granted permission or made a part of this role would be unable to access the information in these tables. This puts the security at the database level, instead of the application level, making it impossible to circumvent by accessing the database outside the application.

Standard Permissions vs. Object Level Permissions

Here is a quick look at the two different ways of managing permissions:

	Standard/Legacy Permissions	Object Level Permissions
User permission types	Read or read/write permissions	SELECT, UPDATE, INSERT, DELETE in any combination
Table, view, stored procedure permissions	The same permission for all	Different permissions for each
Permission setting locations	From the BBj Enterprise Manager or the Admin API Admin	From the BBj Enterprise Manager, API or using standard SQL GRANT/REVOKE statements
User group options	Cannot group users with similar permissions	Can group users using ROLES

>>

By Jeff Ash
Software Engineer

Using Object Level Permissions

The default setting for databases is Standard Permissions. With a click in a single checkbox, the administrator can switch to using Object Level Permissions or back to Standard at any time. To enable Object Level Permissions:

1. While logged into the Enterprise Manager as the "admin" user, click on the database to administer.
2. Select the Permissions tab.
3. Click in the checkbox labeled "Use Object Level Permissions" as shown in **Figure 1**.
4. Click the save button  to update the database configuration.

After enabling Object Level Permissions, developers can assign permissions at the database level as well as at the object level. The Permissions tab shows a list of the users and roles who have permissions assigned at the database level. These permissions include things such as CREATE TABLE, ALTER TABLE, CREATE PROCEDURE, ALTER PROCEDURE, etc.

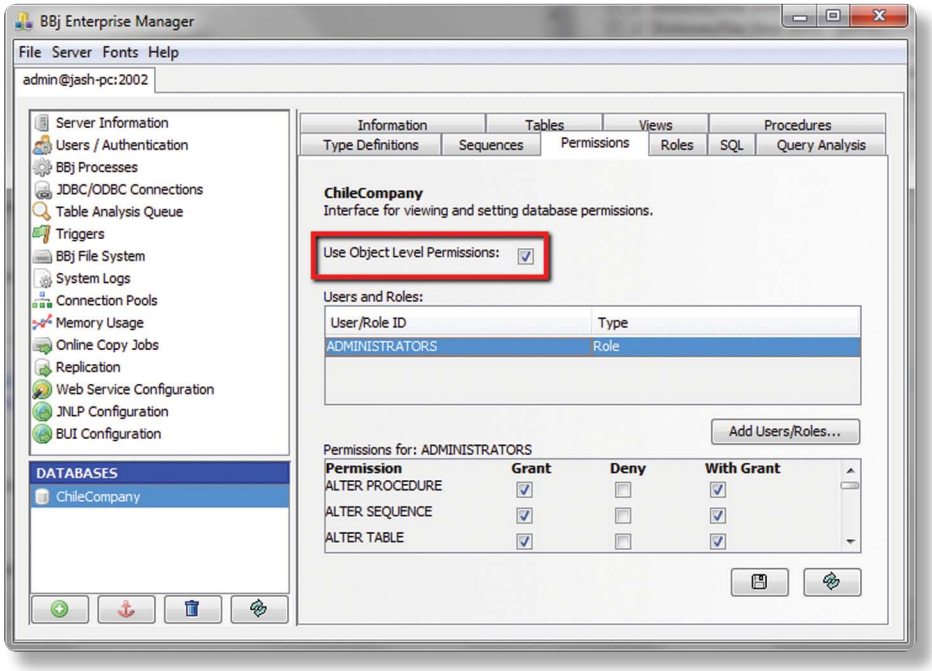


Figure 1. Database Permissions tab

Assigning Table, View, and SPROC Permissions

To assign table, view, and SPROC permissions, select the appropriate tab for the type of object to work with. Right-click on any table, view, or SPROC to show a popup menu with the "Permissions" option available. Then select multiple items at one time to assign permissions to all or some objects. **Figure 2** shows the permissions assignment dialog for tables.

Granting a permission gives the user the ability to perform that operation on the database. Adding the "With Grant" option gives the user the ability to grant that permission to other users on that object. Selecting "Deny" overrides any other permissions that may be set for the user if they should belong to a role that has that particular permission.

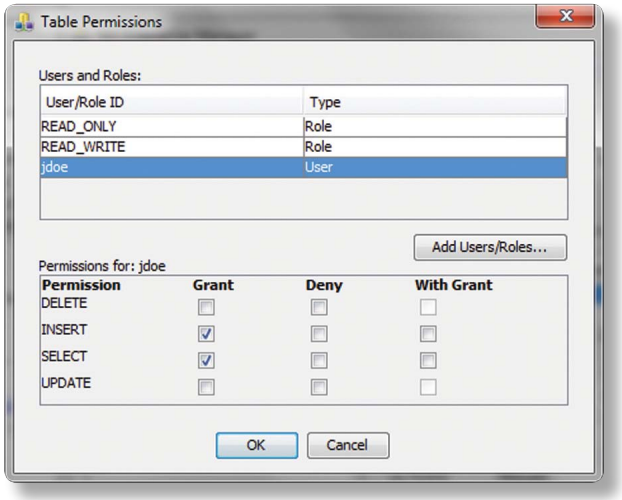


Figure 2. Table Permissions dialog

Using Roles

Roles are a powerful feature used to make the management of permissions much easier. A "role" is simply a list of users who will have the same permissions for certain objects in the database such as in the human resources example cited earlier in this article. Let's take a look at how to create roles and manage membership of those roles.

The Roles tab shows a list of roles currently defined for the database. These roles are specific to only the database for which they are defined. To view the members of a role, simply select the role from the list as shown in **Figure 3**.

After defining roles, developers can assign permissions to roles instead of individual users. This is the recommended way to assign permissions as it is highly unlikely that each user will need completely different permissions on database objects. The example in **Figure 3** shows a role for ADMINISTRATORS that would be used to assign database level permissions to various users who need to >>

perform administrative tasks such as CREATE and DROP TABLE. The READ_ONLY role will be used for users who need only READ_ONLY access, while READ_WRITE would be used for users who need to perform read/write operations from SQL.

Should it become necessary to change the type of access a user needs, simply add them to or remove them from a role, and those permissions instantly change for that user on any objects that have permissions defined for that role.

Using Other Ways to Manage Permissions

There are three ways to manage permissions: 1) using the BBj Enterprise Manager, 2) programmatically using the Admin API, or 3) using SQL GRANT/REVOKE.

SQL Statements Using GRANT/REVOKE

Use GRANT, REVOKE, CREATE ROLE, and DROP ROLE to manage permissions with standard SQL statements. Here are a few examples of how you would use these statements:

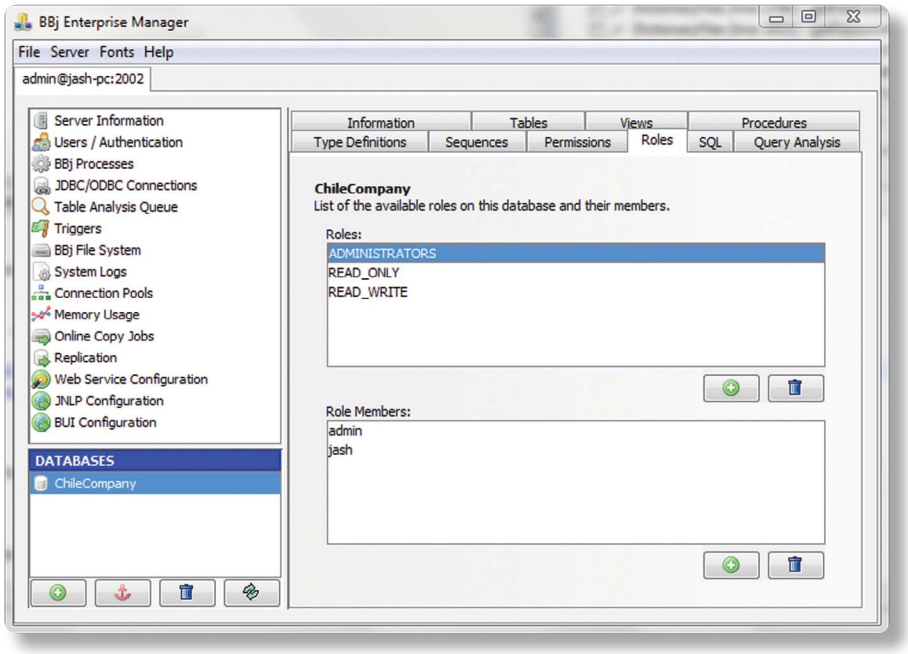


Figure 3. Database "Roles" management tab

Create a new role:

```
CREATE ROLE MY_NEW_ROLE
```

Add SELECT permission to a role on a table:

```
GRANT SELECT ON MY_TABLE TO MY_NEW_ROLE
```

Add a user to a role:

```
GRANT MY_NEW_ROLE TO 'jash'
```

Revoke UPDATE permission from two users on a table:

```
REVOKE UPDATE ON MY_TABLE FROM 'jash','jdoe'
```

Add UPDATE and INSERT permission to two users on a table and allow them to do the same for other users:

```
GRANT UPDATE,INSERT ON MY_TABLE TO 'jash','jdoe' WITH GRANT OPTION
```

BBj Admin API

Discussion of the use of the Admin API is beyond the scope of this article, however, for those interested, check out BBjAdminDatabase.grant(), BBjAdminDatabase.revoke(), BBjAdminDatabase.createRole(), and BBjAdminDatabase.dropRole() in the online BASIS Javadocs documentation for BBjAdminDatabase.

Summary

Security is a vital consideration in any organization and therefore, it is equally as important to BASIS who provides the framework and foundation for any application. BBj 11.0 provides another powerful security feature enabling administrators to better manage access to their precious data with Object Level Permissions. Using Object Level Permissions, administrators now have the power to control SQL access to their databases down to the individual table, view and stored procedure level with ease. Furthermore, the addition of roles allows administrators to group users together for even more flexibility, power, and efficiency. Allied with the ability to offload realtime queries to a replicated database (see Anatomy of a Replication Job on page 32), there is no longer any reason for database administrators to not open up access to live corporate data via a user's desktop data query tool. ■

See *Managing Database* and *Database Permissions* in the online documentation

Custom Apps Install – Easy as 1- 2- 3!

In the last issue of the *BASIS International Advantage*, Introducing the BASIS Custom Installer explained how BASIS had grown tired of using third party installers that were forever being discontinued or whose functionality and support did not meet BASIS' exacting standards. That article unveiled the exciting new smaller, faster installer, and hinted that it will facilitate the installation of non-BASIS applications in the future. Well, the future has arrived with the release of BBj® version 11.0!

In these three easy steps, the BASIS Customer Installer (BCI) lets you install your application software in tandem with the BASIS product suite:

1. Create a .jar file that contains all files necessary to run your software.
2. Create an instruction file.
3. Add the files created in Steps 1 and 2 to a BASIS installable jar.

Let's look more closely as just how easy this is.

Step 1 - Create a Jar

The Jar Utility installs with Java and is much like the familiar .zip or .tar archival tools. In this case it saves multiple files in compressed format into one file with a .jar extension. The jar file must contain all the files needed to make your software run such as BBj programs and resources, data files, configuration files, images, etc. These files are typically located in one directory so the jar command would be something like this:

```
jar cvf package_demos.jar demos
```

The syntax is **jar**, followed by "options," the name of the destination jar file, and then the source directory in which all the software resources exists. The **cvf** options in this example are "c"reate, "v"erbose, and "f"ile.

Step 2 - Create an Instruction File

The instruction file named **custominstall.xml** is simply an XML file that can be easily created, viewed, and modified using the BASIS IDE. It provides the installer with the detailed instructions for installing the software such as the directories in which to install specified files, on which ports (operating systems) to install particular files, and under what conditions to proceed with the installation. Other options include using installation variables like the **$InstallDir** variable that contains the installation directory selected by the user. See the example in **Figure 1**.

For a full-featured custom install, make use of some of the BCI's advanced features

- Create shortcuts on any OS to run your software
- Update files using Search/Keep and Replace nodes in the instruction file
- Create, modify, and remove registry entries on Windows
- Specify process execution, killing of processes, and invoking of custom Java code located in the instruction file to invoke at pre-install, post-install, pre-uninstall, or post-uninstall

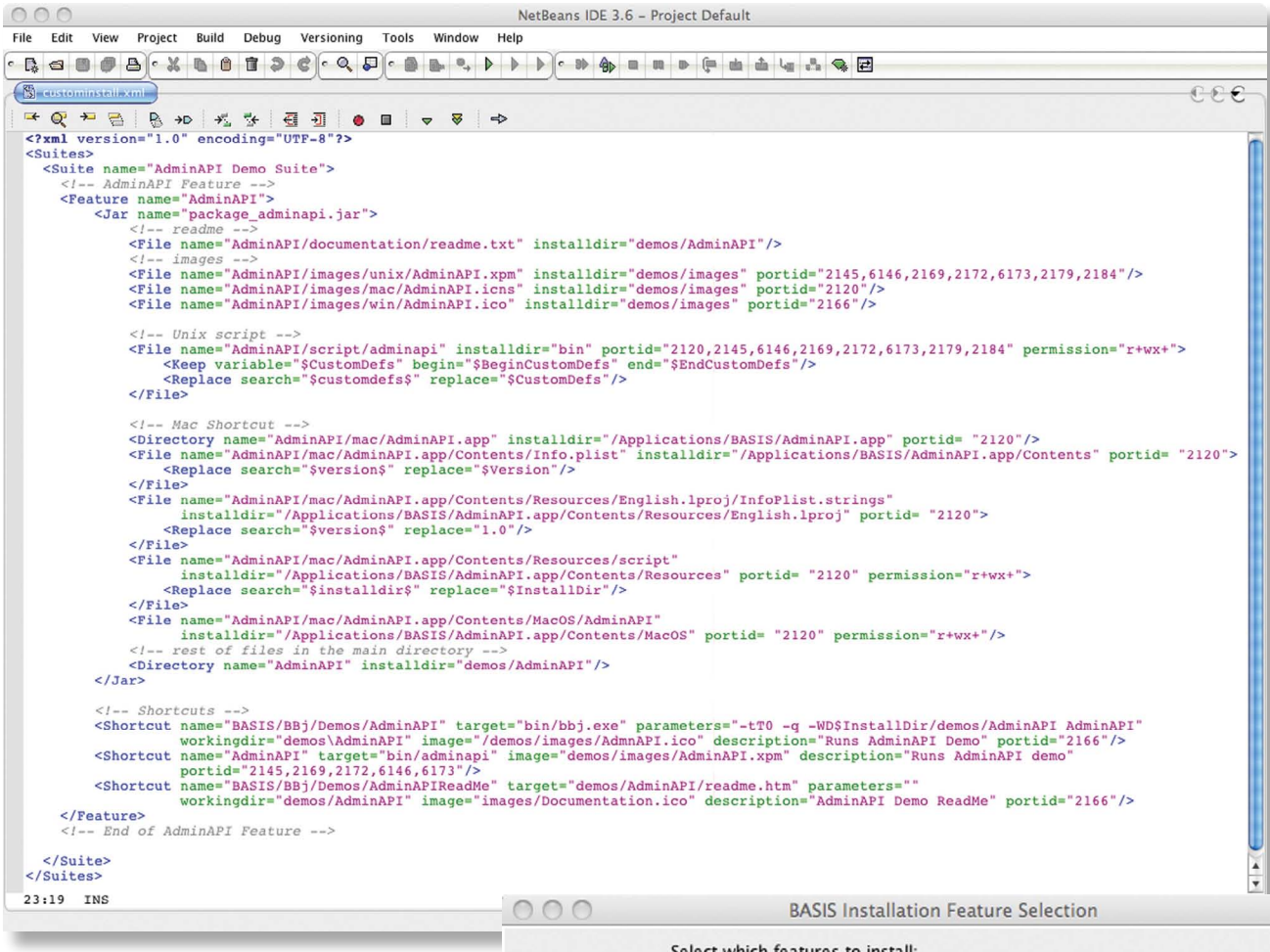
Step 3 - Add Files to the Installable Jar

Select the appropriate BBj installation bundle that fits your needs and simply add the jar that contains your software created in Step 1 and then add the instruction file that was created in Step 2. Once again, use the Jar Utility to accomplish this step as shown here:

```
jar uvf BBjBaristaIDE1110_08-02-2011_1413.jar package_demos.jar custominstall.xml
```

>>

By Brian Hipple
Quality Assurance Supervisor




```
<?xml version="1.0" encoding="UTF-8"?>
<Suites>
  <Suite name="AdminAPI Demo Suite">
    <!-- AdminAPI Feature -->
    <Feature name="AdminAPI">
        <Jar name="package_adminapi.jar">
            <!-- readme -->
            <File name="AdminAPI/documentation/readme.txt" installdir="demos/AdminAPI"/>
            <!-- images -->
            <File name="AdminAPI/images/unix/AdminAPI.xpm" installdir="demos/images" portid="2145,6146,2169,2172,6173,2179,2184"/>
            <File name="AdminAPI/images/mac/AdminAPI.icns" installdir="demos/images" portid="2120"/>
            <File name="AdminAPI/images/win/AdminAPI.ico" installdir="demos/images" portid="2166"/>

            <!-- Unix script -->
            <File name="AdminAPI/script/adminapi" installdir="bin" portid="2120,2145,6146,2169,2172,6173,2179,2184" permission="r+wx+">
                <Keep variable="$CustomDefs" begin="$BeginCustomDefs" end="$EndCustomDefs"/>
                <Replace search="$customdefs$" replace="$CustomDefs"/>
            </File>

            <!-- Mac Shortcut -->
            <Directory name="AdminAPI/mac/AdminAPI.app" installdir="/Applications/BASIS/AdminAPI.app" portid= "2120"/>
            <File name="AdminAPI/mac/AdminAPI.app/Contents/Info.plist" installdir="/Applications/BASIS/AdminAPI.app/Contents" portid= "2120">
                <Replace search="$version$" replace="$Version"/>
            </File>
            <File name="AdminAPI/mac/AdminAPI.app/Contents/Resources/English.lproj/InfoPlist.strings"
                  installdir="/Applications/BASIS/AdminAPI.app/Contents/Resources/English.lproj" portid= "2120">
                <Replace search="$version$" replace="1.0"/>
            </File>
            <File name="AdminAPI/mac/AdminAPI.app/Contents/Resources/script"
                  installdir="/Applications/BASIS/AdminAPI.app/Contents/Resources" portid= "2120" permission="r+wx+">
                <Replace search="$installdir$" replace="$InstallDir"/>
            </File>
            <File name="AdminAPI/mac/AdminAPI.app/Contents/MacOS/AdminAPI"
                  installdir="/Applications/BASIS/AdminAPI.app/Contents/MacOS" portid= "2120" permission="r+wx+"/>
            <!-- rest of files in the main directory -->
            <Directory name="AdminAPI" installdir="demos/AdminAPI"/>
        </Jar>

        <!-- Shortcuts -->
        <Shortcut name="BASIS/BBj/Demos/AdminAPI" target="bin/bbj.exe" parameters="-tT0 -q -WD$InstallDir/demos/AdminAPI AdminAPI"
                  workingdir="demos\AdminAPI" image="/demos/images/AdmnAPI.ico" description="Runs AdminAPI Demo" portid="2166"/>
        <Shortcut name="AdminAPI" target="bin/adminapi" image="demos/images/AdminAPI.xpm" description="Runs AdminAPI demo"
                  portid="2145,2169,2172,6146,6173"/>
        <Shortcut name="BASIS/BBj/Demos/AdminAPIReadMe" target="demos/AdminAPI/readme.htm" parameters=""
                  workingdir="demos/AdminAPI" image="images/Documentation.ico" description="AdminAPI Demo ReadMe" portid="2166"/>
    </Feature>
    <!-- End of AdminAPI Feature -->

  </Suite>
</Suites>
```

Figure 1. Sample instruction file

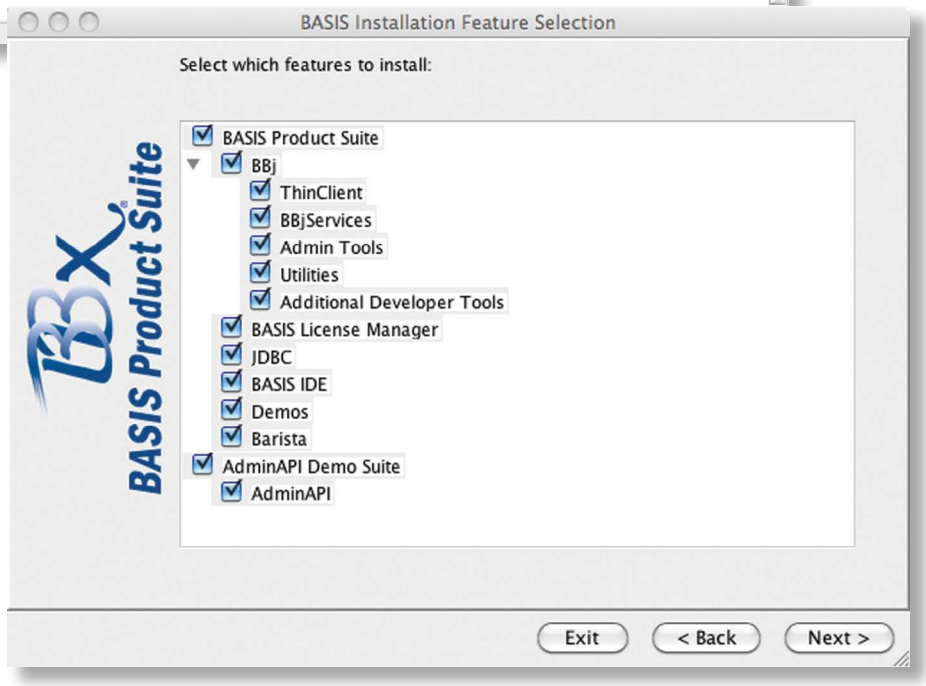


Figure 2. BASIS Installation Feature Selection

The syntax is similar to creating the jar; start with **`jar`**, followed by "options," the name of the installation bundle jar file, and then the jar and instruction file. The **`uvf`** options in this example are "u"pdate, "v"erbose, and "f"ile. The installation is now ready to run. Selecting a custom installation type shows your newly added suite as **Figure 2** illustrates.

Summary

Instead of your end users installing the BASIS products and then installing your software, wouldn't it be better to combine it all in one installation process? Using the powerful new BCI, bundling your software along with the BASIS products installation is as easy as 1-2-3! Remember too that you can create a response file and install silently, only prompting the user for user-specific information. Now you can benefit from BASIS' engineering efforts and reap the rewards both monetarily and functionally by discarding your third party installer and using the BCI. This is just another example of BASIS' building block strategy at work, saving you money and making your business more efficient.

Review the full description of a BCI custom app installation and all that is available in the instruction file in the online help topic *BCI Overview*

Let's See Your John Hancock

"Hello, Mr. Hancock. I have two Chile Lovers t-shirts and one Salsa Gift pack for you today. Would you please just sign here?"

Modern applications in today's paperless world often require a way to capture signatures on touchscreen-enabled devices. Whether for proof of delivery, service work orders, sales forms, or contracts, capturing handwritten text has become a necessity for today's businesses. **Figure 1** shows the Signature Capture demo running on an Android tablet and **Figure 2** shows the signature panel.

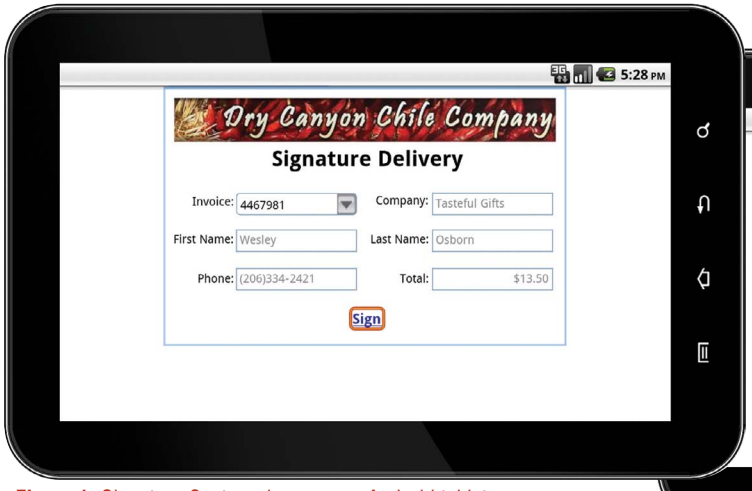


Figure 1. Signature Capture demo on an Android tablet

Figure 2. Draw panel for signature capture

John Hancock was president of Congress when the Declaration of Independence was adopted and signed. He is primarily remembered by Americans for his large, flamboyant signature on the Declaration, so much so that "John Hancock" became, in the United States, an informal synonym for *signature*.

The setScribble() method, new in BBj® 11.0, has an easy-to-implement mechanism of creating and storing handwritten text. Various supported platforms include laptops, smartphones, and tablets running on a variety of operating systems including Windows, iOS, Android, and Linux.

Available for all three BBj client flavors – GUI, BUI desktop, and BUI mobile/touch – getting and storing your John Hancock has never been so simple or secure.

The three primary methods to create and store a signature are setScribble(), getDrawPanelImage(), and getBytes().

- setScribble() is used on the window (draw panel) to intercept relevant mouse or touch events and draws using the current PENWIDTH, PENCOLOR, PATTERN, and DRAWMODE settings.

```
#SignatureWindow!.setScribble(1)
```

- getDrawPanelImage() is used on the window or child window object. This method returns the draw panel as a BBjImage object.

```
sigImage!=#SignatureWindow!.getDrawPanelImage()
```

By Robert Del Prete
Quality Assurance Engineer

- getBytes() is used on the BBjImage object to return raw image bytes that can then be written to disk as an image file. Image formats supported are .png, .gif, and .jpg. This file can be saved to the client, server, or stored in a database using a BLOB (Binary Large OBject) datatype.

```
sigImage$ = sigImage!.getBytes("png")
```

The code snippet in **Figure 3** demonstrates getting the image from the draw panel. The first section gets information from a list box to name the resultant .png file. We then create a new java.io.File for the image. The getDrawPanelImage() method is used to get the draw panel as a BBj image object and getBytes() allows the image to be written to disk.

```
rem Callback routine that is called when Save is pressed
method public void OnSavePressed(BBjButtonPushEvent p_event!)
    declare BBjNumber lbIndex!
    declare BBjString lbValue!
    declare BBjString sigFile$
    declare java.io.File sigFile!

    lbIndex! = #invoiceLB!.getSelectedIndex()
    lbValue! = #invoiceLB!.getItemAt(lbIndex!)

    sigFile$ = dsk("") + dir("") + lbValue! + "-signature.png"
    sigFile! = new java.io.File(sigFile$)
    sigImage! = #SignatureWindow!.getDrawPanelImage()
    sigImage$ = sigImage!.getBytes("png")

    image=unt
    open(image,mode="O_CREATE,O_TRUNK")sigFile$
    writerecord(image)sigImage$
    close(image)
    #SignatureWindow!.setScribble(0)
    #SignatureWindow!.setVisible(0)
    #this!.setDeliverButton()
methodend
```

Figure 3. Retrieving the image from the draw panel

Summary

While the need for the handwritten word has greatly changed since the time of John Hancock, there is still great need for capturing handwritten signatures electronically. Signature capture is available anywhere BBj runs, even on popular iOS and Android smart phones. All that is needed is the browser of any touch-enabled iOS or Android device to capture signatures in the field and to store the resultant image file safely on the server. Using setScribble() pays back big dividends in both time-savings and convenience. Digitally capturing signatures in the office, warehouse, or even on the go using mobile handheld devices is easy using the setScribble() method. ■

"Thank you, Mr. Hancock. Have a pleasant day!"

For more information, refer to setScribble(), getDrawPanelImage() and getBytes() in the online documentation at links.basis.com/basisdocs

New Browse Method Eases File Open Process

A common feature of web browsers is to allow the user to choose or specify a document they would like to see, and then open a corresponding application or viewer for that type of file. As websites continue to evolve towards becoming applications and vice-versa, more applications require the ability to display a document where the type of document the user will choose may not be known beforehand. This functionality, known as an application association, is already contained in popular browsers.

Wouldn't it be great if programmers could leverage the browser's functionality rather than having to roll their own? Java provides exactly this functionality through its Desktop::browse() method, and Javascript provides this functionality through Window.open(). Leveraging these two technologies, BBj® provides the BBjThinClient::browse() method in GUI and BUI.

BBjThinClient::browse() in Action

So, what exactly does this new method BBjThinClient::browse() do? Quite simply, the browse() method takes a URL as a parameter and opens that URL in a browser. URLs most commonly reference pages and files on the Web, but can also reference files locally via the `file://` protocol. Since BBj includes a Web Server, references to files on the server machine can be a URL.

To illustrate this feature, run either of the "*Browse - BUI" or the "Browse - GUI"* demos that are included with the BBj product when downloaded with the demo checkbox selected (BBj 11.10 or higher). The demo presents three options for opening a file. The first option is to specify a URL to the file, the second option is to open a file from the machine that is running BBj Services (server-side), and the third option is to open a file that resides on the same machine on which the demo is running (client-side). Once a file is specified and the "browse() file" button pressed, the file path is translated into a URL and the demo launches the default browser with the URL. The browser uses its file associations to determine the appropriate application for opening the file, and finally opens the file with that application. **Figure 1** shows the demo launching the native movie viewer application when a .mov file is passed to the browse() method.

By using the browser in this way, BBj programs can use the browse() method to open any type of file with the appropriate application without having to know which file will be specified ahead of time. For example, the default browser on a Windows system would likely open a new tab for an HTML file, launch Windows > >

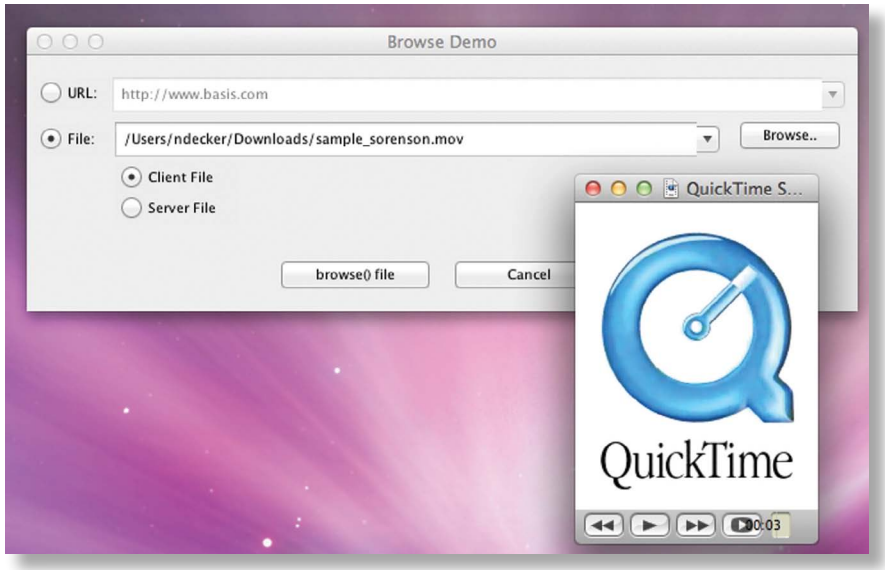


Figure 1. Browse Demo launching a native application to view the selected file

By Shaun Haney
Quality Assurance Engineer

Media Player for a .mp3 file, launch Microsoft Word for a .doc file, and launch Windows Photo Gallery for a .png file. What application opens what file, however, is ultimately in the hands of the user and how a file is opened is now outside of the realm of worry for the application developer. In many cases, the need for developers to integrate file viewers with their applications is gone.

Generally speaking, the default browser opens the specified URL and the **file://** protocol specifies URLs for local file locations. However, the browse() method treats the **file://** protocol differently from other protocols depending upon the environment in which it's used:

- The **file://** protocol is unsupported within BUI programs where the underlying Javascript mechanism does not have permissions to access the client's file system. Read on to learn how to overcome this limitation.
- Associations for URLs with the **file://** protocol may be handled by Java's URI handlers rather than being launched with a browser.

To illustrate the browse() method launching an application without the help of the default browser, enter **BBjAPI().getThinClient().browse("file:///C:/program%20files/basis/demos/adminapi/images/browse.png")** in SysConsole on a Windows Vista machine, which results in BBj opening **C:\Program Files\BASIS\demos\AdminAPI\images\browse.png** with Windows Photo Gallery. In this case, Java's URI handlers determined the file association independently of the browser on the user's system.

While the browse method delegates the task of opening files and greatly reduces or even eliminates the need for handling opening files of different types in code, a nagging question remains. Since every path passed to the browse() method must be in the form of a URL, how does one create URLs for client-side files and server-side files? The Browse demo comes in handy to answer this question since it contains code handling all of these cases.

Forming a URL for client-side files is straightforward in GUI. The URL's protocol is **file://**, followed by a beginning slash if the path does not already begin with a slash, followed by the path to the file using forward slashes as the path separator. Special characters, the most common of which is a space, require appropriate encoding for URLs. For example, if the client-side file resides at, **C:\Program Files\basis\demos\BBjThinClientBrowse\BUIBrowse.png** then the URL will be **file:///C:/Program%20Files/basis/demos/BBjThinClientBrowse/BUIBrowse.png**.

For server-side files, remember that every install of BBj comes with a Web Server (see *A Home (Page) in Every Port* at links.basis.com/11homepage) and that it runs as part of BBj Services on port 8888 by default on the server machine. Any file on the server machine placed in <BBj Install Directory>/htdocs will be accessible at **http://<server name>:8888/files/<file name>**. The Browse Demo copies the server-side file to the specified directory under the htdocs directory and then forms the URL as described above. **Figure 2** shows the result of the demo program copying the selected server file to a directory hosted by the web server and launching a new instance of the client's browser to display the file.

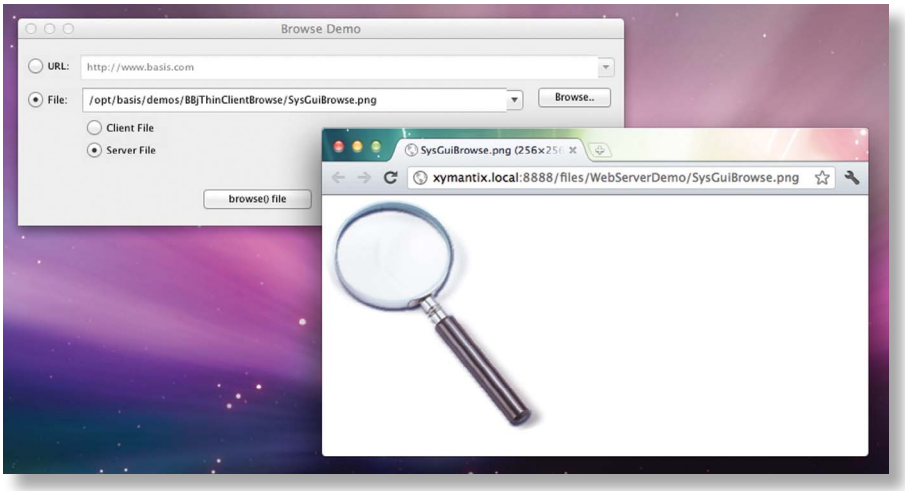


Figure 2. Browse Demo showing a server-side file in the client's browser

By way of example, if the file above resides at **C:\Program Files\basis\demos\BBjThinClientBrowse\BUIBrowse.png** on the server machine, the demo program copies it to **C:\Program Files\basis\htdocs\WebServerDemo\BUIBrowse.png**, and the corresponding URL becomes **http://<myserver>:8888/files/WebServerDemo/BUIBrowse.png**.

Opening a client-side file with the browse method in BUI is much like opening a server-side file except that the file starts out on the client-side. To view the file, it must be served up from the server-side web server. So, the first step is to create a BBjClientFile for the file that is to be browsed. Next, call the copyFromClient() method on the BBjClientFile object to copy the file to the server. The string returned by copyFromClient() reveals the location of the file on the server. Finally, copy the file over to the **htdocs** directory for the web browser to serve up.

Summary

Using the BBjThinClient::browse() method provides application developers the ability to present any BBj accessible file to the user, both client and server-side, without needing to determine ahead of time which application needs to be associated with the file for the user to view its contents. The user's machine configuration determines how the file will be opened and avoids the need for the developer to provide this knowledge from within the application. This BASIS-supplied functionality enables desktop and web application convergence, allowing developers to create applications that deliver content in a greater variety of formats regardless of the mechanism of their deployment. BASIS continues to deliver on its promise of "*Write once, run EVERYwhere*"! ■

For more information, refer to online documentation

- Desktop::browse
- BBjThinClient::browse

Adding *Style* to BBx® Web Apps With Custom CSS

Cascading Style Sheets, referred to as CSS, have been used since the mid-90's and over time have become the de facto way to define how a website should look. The primary goal of CSS has always been to separate the presentation (how the page looks) from the content (the actual text or article in the web page). There are many advantages to this as changes to one CSS file can propagate to an entire website, resulting in reduced development and testing time, greater consistency across the website, reduced ongoing maintenance, and so on.

CSS offers many benefits beyond that primary goal as CSS has continued to evolve, and add greater functionality, due in part to users and website designers demanding new features and capabilities so that web pages can compete with traditional desktop applications. Although CSS has conventionally been used to style web pages, BBx® developers can now reap the benefits of CSS when deploying their applications in BASIS' Browser User Interface (BUI).

By Nick Decker
Engineering Supervisor

So Why is CSS Exciting?

When a BBx application runs in BUI, it is executing on the end user's browser and making use of the standard HTML, JavaScript, and CSS capabilities that come with the platform. BASIS includes a default style sheet for use with BUI apps, but has designed BUI configuration to be open to overriding this CSS file with a custom sheet. Ultimately, this means that anyone now has the power to easily change a BBx application's look and feel using a popular, open, independent, and freely usable standard.

Before digging into the details, let us take a quick look at a BBj® BUI app that utilizes various aspects of CSS to create a rich UI. Because a picture is worth a thousand words, comparing the app visually with and without the custom CSS file says volumes about its capabilities and is the best illustration of how much CSS can affect the presentation. **Figure 1** compares the same BUI app running on an iPhone – one with custom CSS and one without.

The images in **Figure 1** on the next page, illustrate many of the formatting options that are available via CSS and drive home the point that we can manipulate the appearance of the content without modifying the content itself. In other words, it is the same BBx program running in both scenarios; both using a BBjStaticText control to display the "BUI Tip Calculator" title at the top of the screen. The CSS employed in the second example uses the entry in the stylesheet definition file and shown in **Figure 2** to format the static text and turn it into a fancier title. > >

```
/* Customized top bar for title */
.topGradientBar {
    background: -webkit-gradient(linear, left top, left bottom,
        color-stop(0.0, #8b98ba), color-stop(0.45, #4764a0),
        color-stop(0.46, #3c5899), color-stop(1.0, #314a82)) !important;
    font-family: Tahoma, Helvetica !important;
    font-size: 30px !important;
    font-weight: bold !important;
    line-height: 46px !important;
    text-align: center !important;
    color: #eceff5 !important;
    text-shadow: rgba(255,255,255, 0.75) 0px 1px 1px,
                 rgba(0,0,0,.9) 0px -1px 1px !important;
    -webkit-box-shadow: 0px 0px 4px rgba(0, 0, 0, .95) !important;
}
```

Figure 2. The CSS definition for the BBjStaticText control used for the title

Figure 1. A BUI app running without custom CSS (left) and with custom CSS (right)

Dissecting the CSS Example

The CSS entry starts off with a comment on line 1 that documents the function of that section of the file. Line 2 begins the actual definition for the CSS Selector, which is what CSS uses to define elements. The selector name is important as that is what is used to apply the new style to one or more BBj Controls. Selector names are unique and user-defined, but be aware that BASIS has reserved selectors and special selector names that correlate to control names. For example, if we named this selector **`.BBjButton`**, then the style would automatically apply to all BBjButton controls in the app. For more information on reserved selectors and BUI CSS in general, see the CSS API documentation. Because this is not a reserved selector, we can optionally apply it to any control with the **`addStyle()`** method.

Lines 3-14 in the example list a number of properties and their associated values that comprise the definition of the custom selector. CSS properties are generally well named, and **`font-size`** on line 7 serves as a good example that shows it should not be difficult to figure out what effect the property will have on the control. The BASIS IDE's built-in syntax coloring also comes in handy, making it easy to discern which properties are standard and which are proprietary. The properties in **Figure 2** are responsible for adding a background gradient to the control; setting the font, size, weight; vertical positioning, horizontal alignment, and color of the text. They even create an embossed effect for the title by applying a semi-transparent white shadow below the text and a semi-transparent black shadow above the text. This is just a taste of what CSS offers, as developers now have command over a multitude of presentation properties and can easily transform the look of their applications without making any changes to the program's source code.

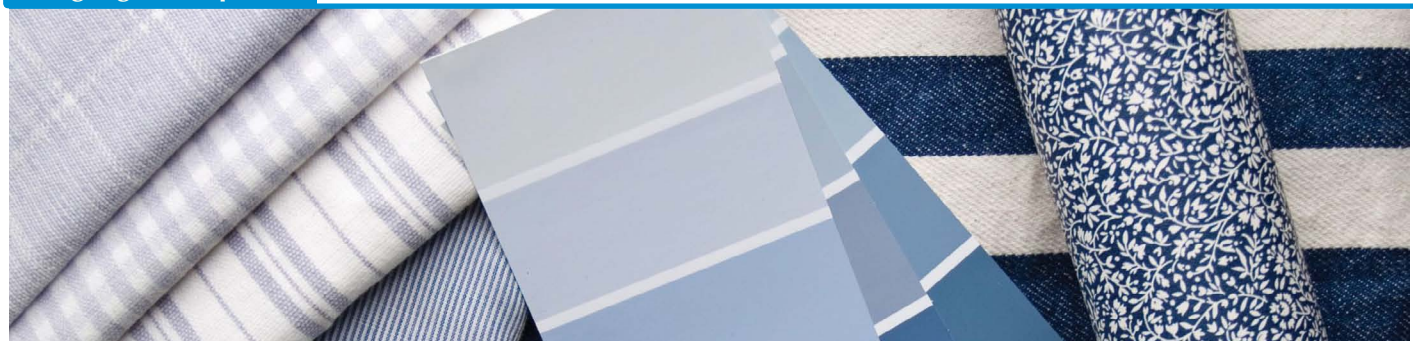
One Size Doesn't Have to Fit All

One of the selling points for CSS is that developers can easily change the look and feel of their BUI program without modifying the BBj source code. The previous example changed the look dramatically, but what about a special requirement or need such as an employee that is color blind or vision impaired? Or perhaps a company wants to brand their application with their corporate colors and logo? CSS offers solutions to all of these scenarios by allowing developers to create as many different themes for their applications as necessary. Enterprise Manager makes it easy to register a BBj program as a BUI app, and it is just as easy to register the same application multiple times with unique style sheets.

With CSS, simply deploy the same BUI app to different clients, but with each client having their own custom URL and look and feel for the app. Now adding a new color scheme for a color blind user is possible with very little effort. Additionally, one can improve legibility for those with eyesight limitations by increasing the font size, specifying higher-contrast color schemes, and clearly highlighting selected and active controls – all with just a few lines of CSS. Adding hover effects also enriches the user experience and gives immediate feedback for mouse movements, improving usability and adding an extra level of interactivity to your app.

Improving Upon the Defaults

Enhancing user interactivity is one area in which CSS excels as developers have so much influence on the appearance of the controls on a form. In addition to the CSS selectors mentioned previously, CSS has the concept of Pseudo-Classes/Elements that modify or add special effects to existing selectors. These pseudo-elements make it possible to further customize or differentiate controls in various states, such as the text control on a form that currently has keyboard focus, or the button that the user is clicking. Calling attention to these controls by changing their style enhances the user's sense of direct manipulation and is a great way to improve the user experience. >>



A common use case for customizing the look of a particular control's default state would be to enhance legibility for read-only or disabled text controls. Different operating systems have their own individual way of coloring these controls, but this sometimes results in making the application harder to use. Sometimes the differentiation between an editable and read-only control is so slight that users may not notice the difference and become frustrated when the application thwarts their attempts to edit the control. Other times the disabled version of a text control results in dark gray text on a medium gray background; sure, it is easy to tell that the application has disabled the control but the color scheme makes the text virtually impossible to read.

Using CSS, we can easily remedy these scenarios and even provide multiple solutions in the same application via CSS styles given the users' preference. We can do this on an application-wide level using two aforementioned BASIS reserved selectors, **`.bbj-disabled`** and **`.bbj-readonly`**. In addition, we could also specify multiple classes in the same selector to further filter which controls the selector applies to. For example, the selector

```
.bbj-disabled.myCustomButton { … }
```

will only affect controls that have previously had the "myCustomButton" style applied and are currently disabled. Multiple class selectors make it easy to isolate a specific set of controls and result in the selector overriding less specific definitions.

Other CSS Benefits

The BBjStaticText control in the BUI demo now looks a lot better, and one of the benefits of stylizing with CSS is that it scales extremely well. Unlike graphic images that lose quality when the user zooms in their browser view, CSS looks even better when zoomed. To illustrate this difference, **Figure 3** shows a side-by-side comparison of a zoomed-in BBjButton defined both as a graphic image, and by using CSS without images. Notice how the text and the button itself scale dramatically better on the right with CSS compared to using an image of the button, making the button sharper and easier to read.

Figure 3. A button defined with an image (left) compared with CSS (right)

This extra quality does not come with much of a cost, either. Although there are ways to optimize the use of graphic images in CSS, such as using sprites, image-free CSS is very lightweight. The client requires only the selector definition and then renders the object accordingly. This can reduce network overhead as compressed CSS text files have a small footprint and require much less bandwidth compared to what may amount to numerous large image files. With CSS, the client web browser is now responsible for the layout and rendering of many graphics, but that too may be optimized. Certain platforms, such as iPhone's iOS, offload some of the more tedious CSS rendering tasks to the dedicated GPU (graphics processing unit). This results in smooth animations such as fades, zooms, translations, scaling, etc., while reducing the load on the machine's CPU. The final result is higher quality renderings and reduced network traffic – a double whammy – making CSS a great way to increase the visual appeal of your apps while increasing performance at the same time.

Taking CSS Further - Custom Controls

With a little ingenuity and some fancy CSS, we can create a "custom control" from one or more of the traditional BBj controls. There are several reasons for doing this, not the least of which is "because we can!" Creating custom controls can add to the appeal of your web app and may even enhance usability and blend in better with the purpose of your app. As an example, the "BUI Tip Calculator" adds and removes CSS styles to regular BBjButtons to create both a star rating system and a graphical way to specify the number of guests at the restaurant. **Figure 4** shows a section of the app where the user has set the number of dinner guests to three.

Figure 4. Specifying the number of guests in the dinner party

Technically, we are just looking for a way to get a numerical value from the user and there are several controls that could accomplish that task: a BBjEditBox, BBjInputN, BBjSlider, to name a few. In this case though, a custom control makes a lot of sense, and is a better choice compared to the traditional controls. For starters, we designed the app to run on a smartphone, like the iPhone, a touch-enabled device with a small screen. Our custom control is perfect for this environment since the user can specify the number of guests via a single tap. In comparison, if we had > >

used an edit control, then the user would first tap on the control causing the phone's keyboard to slide up from the bottom of the screen. The user would then have to find the desired numerical key on the keyboard and tap it. A third and final tap would be required to dismiss the keyboard and return to the app. This process is inefficient, cumbersome, and requires more dexterity from the user which distracts from the 'flow' of the app and user experience in general. Lastly, a text control can be a bit boring, especially in the iPhone world where app developers focus on designing appealing applications with alluring user interfaces. Apple even provides a 150+ page document covering best practices for Mobile Human Interface Guidelines, which contains sections with titles like "Delight People with Stunning Graphics" to ensure a high level of quality.

Another example of a custom control is the loading indicator that displays when the "BUI Weather" portion of the same demo program refreshes its data. **Figure 5** shows how this custom control notifies the user that the app is busy loading new weather data.

Figure 5. Custom Busy Indicator

BBjStaticText controls, along with some CSS to define the appearance and provide animations, work together to create this custom control. This is another case where a traditional control, such as a BBjProgressBar, would be a reasonable candidate. However, since we want our app to look and feel more like a native iPhone app, we can mimic some of the more popular iOS elements and behaviors such as an alert message box or a HUD-style busy indicator. Our custom control does just that and relies on CSS to provide fade-in and fade-out animations, a translucent backdrop for the control with rounded corners, and an animated rotating busy indicator. The result is the same in that the user is still notified that the app is busy and will not accept any user input, but thanks to CSS, it is a pleasant experience that the user is accustomed to because the BUI app looks and acts like a native app.

Summary

We have now introduced you to the many facets of CSS and how it can single-handedly improve your app's look, functionality, and even performance. You gain complete control over each screen element's appearance including spacing, alignment, positioning, font, size, weight, color, opacity, shadows, and more. CSS also reduces your development time when you begin to share CSS files between apps, enforcing consistency and reducing maintenance.

With CSS, you can also brand applications on a per-customer basis, provide an updated look-and-feel without touching the BBj source code, and even provide multiple themes and color schemes within the same app to improve the user experience. CSS makes it possible to create custom control interfaces that mimic native controls on other platforms, increasing the ease-of-use and familiarity for users. Lastly, CSS is a free open standard that is pretty easy to learn – especially with the hundreds of books, web pages, online tutorials, and examples available on the topic.

So what are you waiting for? Add some *Style* to your BBx web apps today! ■

- For more information, review the online documentation for information on
  - BBjStaticText
  - BBjButton
  - CSS API
- Read about Pseudo Classes/Elements
- Check out Apple's "Mobile Human Interface Guidelines"

Easy Integration to Google Apps

Google Apps, or GApps for short, is a free service from Google providing independently customizable versions of several Google products under a custom domain name. It features several web applications with similar functionality to traditional office suites and includes Google docs, calendar, and sites. BASIS provides toolsets that allow developers to incorporate this powerful cloud functionality directly into their BBj® application, expanding the capability and value of their applications.

What GApps Provides

GApps allows for the sharing of information by using cloud technology for access and storage and provides security, a web browser-based user interface, APIs, and ever-changing improved functionality. Users who create or upload documents to Google can easily share them with any user across the Internet. As they create or upload the documents or "docs," cloud technology saves it around the world for redundancy, fast user access, simultaneous collaboration, and revision control. Gone are the days of emailing a document or saving it on a shared drive to distribute to others. Sharing a file is as easy as supplying the email addresses of the desired collaborators. Google then sends an email notification to the users with a link to the Google doc.

Users can edit shared Google docs in a web browser, just like in most any other word processor. Therefore, just like BASIS products, GApps is cross-platform and not tied to a particular operating system, nor does it require any additional software. If multiple people have the doc open, anyone who is viewing the document can see all the changes as well as their own, as they occur. In fact, collaborators can even chat among themselves inside the doc. Google provides various security options for defining users, groups, and domains as owners, editors, or simply viewers of the doc.

What the BASIS GApps Utility Provides

Google Apps are accessible in a BBj application since the Google API provides support for the Java language and BBj supports embedded Java code. The Google Java API is very powerful but has a steep learning curve. To simplify GApps integration, BASIS provides the GApps Utility that provides programmatic access and dialogs for Google Apps services including docs, calendar, and sites.

Docs

The Google Docs methods are quite extensive and allow for uploading, downloading, removing, and setting doc permissions. Dialogs for opening, saving, and selecting a folder for docs behave similarly to the standard open file dialogs that have been in the BASIS language for years (see **Figures 1** and **2**). These dialogs run in GUI and BUI, are fully localized, and will prompt the end user to log in to GApps, if necessary. UserProperties, discussed in detail in *Need Cookies? Get User Properties* at links.basis.com/11cookies, saves valid user credentials to avoid presenting a login dialog on subsequent accesses.

Calendar

The GApps Utility provides access to the Google Calendar Web Service and allows the addition of more functionality as the need arises.

By Brian Hipple
Quality Assurance Supervisor

See this in action in a fully functional calendar demo application that interacts with the Google time management Web Service as shown in **Figure 3**. This Google Calendar demo is a BUI application that retrieves a Google calendar from a gmail account and displays the calendar in a BBjHtmlView control. BUI is the mode of choice for this application since web browsers have the latest HTML, CSS, and JavaScript necessary to display the calendar. Since Java's HTML viewer control has limited HTML, CSS, and JavaScript capabilities, the calendar cannot display in SysGUI. Run the demo and see how easy it is to add your own calendar events with a custom name, date and time, and description. Additional options include guest invitations, event permissions, repeat criteria, attachments, and reminders.

Sites

The GApps Utility provides access to Google Sites which allows web page creation, maintenance, and sharing. The "BUI-GSites" program demonstrates how to modify a web page programmatically. First provide a Domain, Site Name, and Page Title. After successful authentication, this demo retrieves the web page and then displays it using a BBjHtmlView control. Now dynamically modify the web page by replacing string literals or by specifying the new content for the desired DIV. **Figure 4** shows this demo modifying the BASIS intranet Google site.

GApps at Work

As an example, see how BASIS puts GApps to work in two different applications.

- BASIS modifies their intranet web page on a weekly basis in batch mode to change the PIN numbers for the conference phone system.
- BASIS also uses the GApps utility for their Barista®-based AddonSoftware® accounting system. The GApps Utility saves all necessary documents, including Barista- and Jasper-generated reports, to the cloud in accordance with the BASIS Business Continuity and Disaster Recovery program. >>

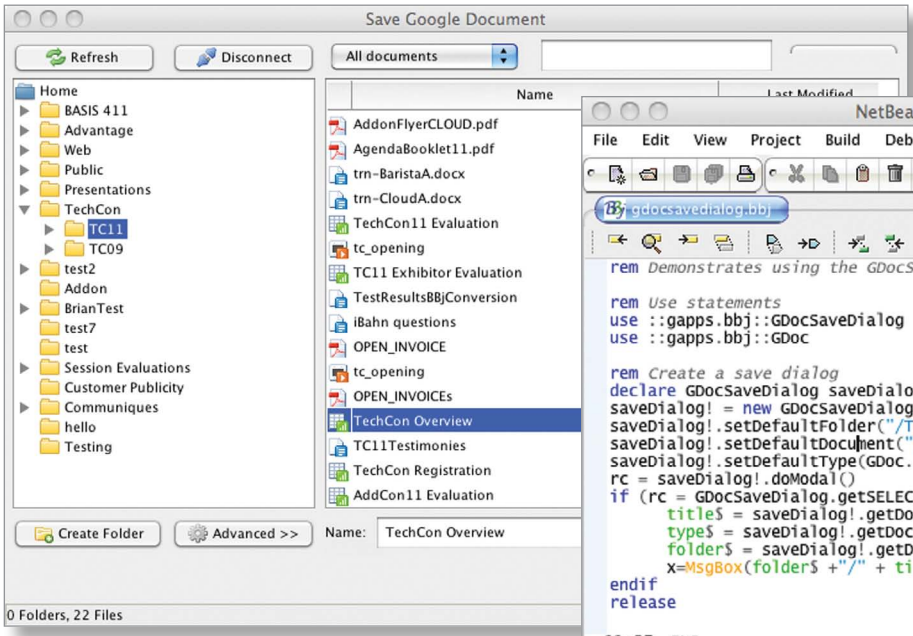


Figure 1. Google doc save dialog

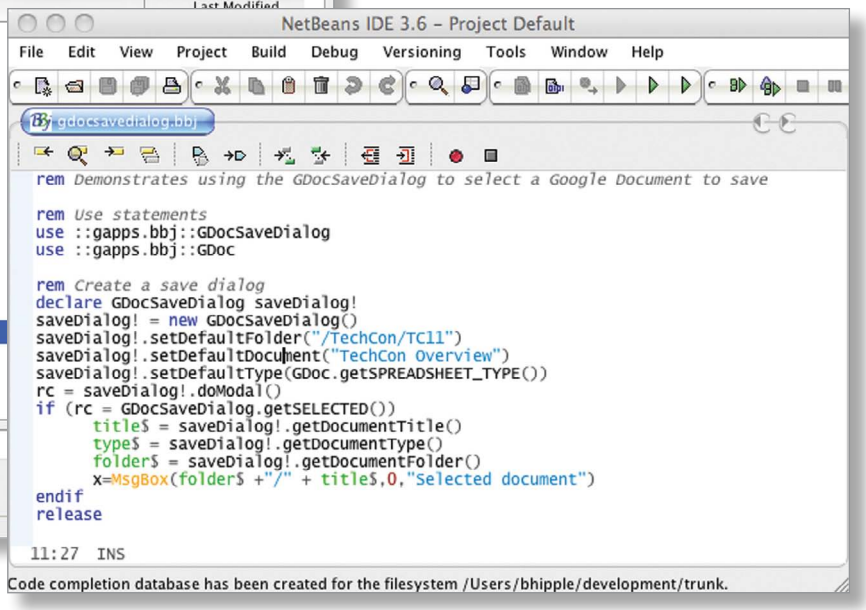


Figure 2. Code that creates the Save Google Document dialog

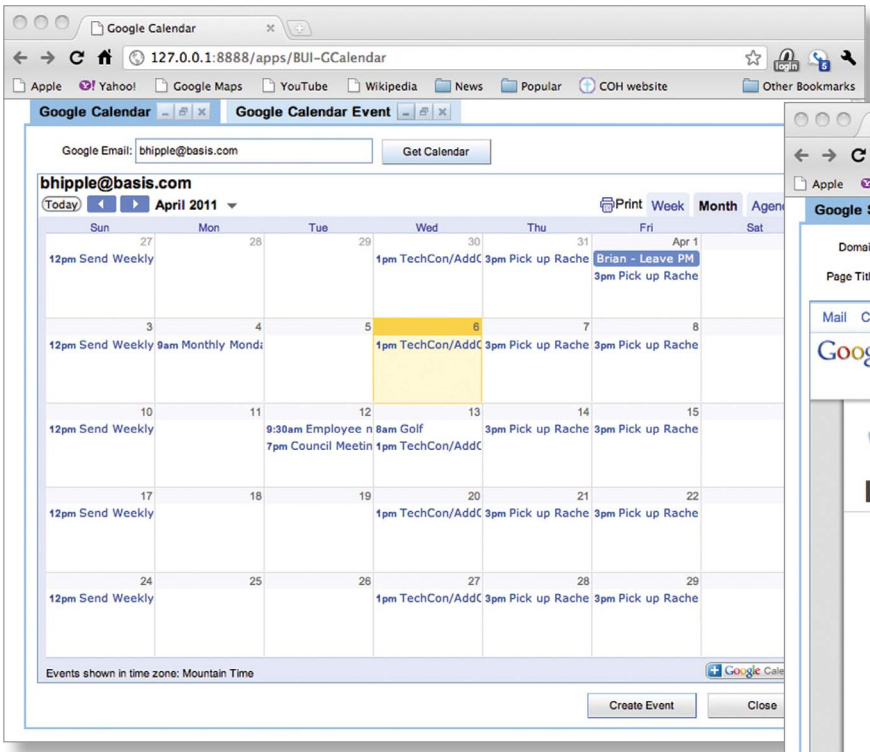


Figure 3. The BUI Google Calendar demo

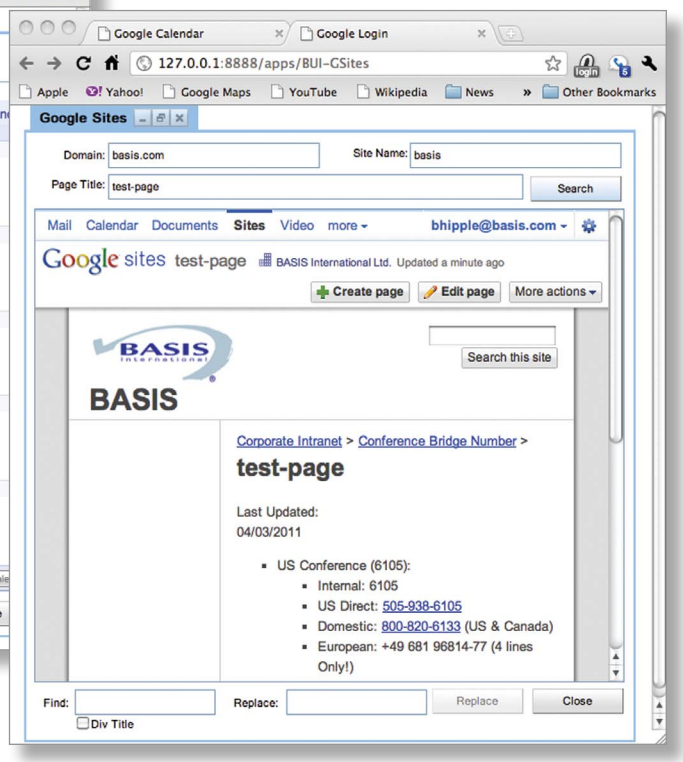


Figure 4. The BUI-GSites demo

Summary

BASIS is not the only company using GApps. The impressive three million businesses that use Google Apps include such well known companies as Berkeley Lab, BASF, Capgemini, Delta Hotels, US General Services Administration, Salesforce.com, KLM, Konica Minolta, Motorola Mobile Devices Division, and National Geographic, to name a few.

BASIS' GApps Utility building block will smooth the way for you and your clients to leverage the cloud today. What are you waiting for? ■

See who else uses Google Apps at www.google.com/apps/intl/en/customers

Anatomy of a Replication Job

BBj® replication allows users to have one or more up-to-date copies of their production data at all times. This feature is extremely useful for both backups and load balancing through read-only data access on target machines, and provides many benefits. Since BASIS built replication into BBj, developers do not have to update their applications to keep targets automatically current with changing source data. Replication overhead is extremely low, as BASIS designed it for minimal performance impact to writes on the source machine and no impact on read performance. It is also robust as it continues to replicate data flawlessly, regardless of how many times the source and/or target BBj Services are stopped and started. Administration is very flexible as users can maintain replication jobs easily in Enterprise Manager or in their own custom programs via the Admin API. Both methods allow replication job management on-premise and even in the cloud.

Replication Overview

The basic architecture of BBj replication involves a source BBj Services with one or more replication jobs. Each replication job describes a set of directories and files to replicate to a single target BBj Services. A user can create as many replication jobs as desired, each replicating any set of directories and/or files. In addition, users can either replicate the same data to many different targets or replicate different data to different targets.

Creating the Initial Copy

When a user first creates a replication job, the target requires an initial copy of all of the files that the job specifies. While the replication process is capable of making initial copies of missing or out-of-date files by itself, this process will keep the file open in the background for however long it takes to copy the file. The faster the network connection, the better; large files on a very slow network connection could potentially take several hours.

Users may access files and make changes while the initial replication copy is in process, but they cannot delete the files. If users are willing to shut down their server for a short period, BASIS recommends restarting BBj Services in the "admin only" mode and manually rsyncing the data just before creating the replication job. This allows the administrator to control the impact of the required initial copy and minimize the effect on users of their application.

Understanding the Replication Log

The replication job only needs to monitor changes to the source data, as read access does not modify the source data. Therefore, BBj Services tracks all modifications to the source files and records those changes to a replication log. The log then contains a continuous stream of file writes, removes, creations, and deletions for replicated files and directories without the replication job needing to actually open and read the source files or directories. Since this happens at the file level, the log reflects the data records exactly as they appear in the actual source file. >>

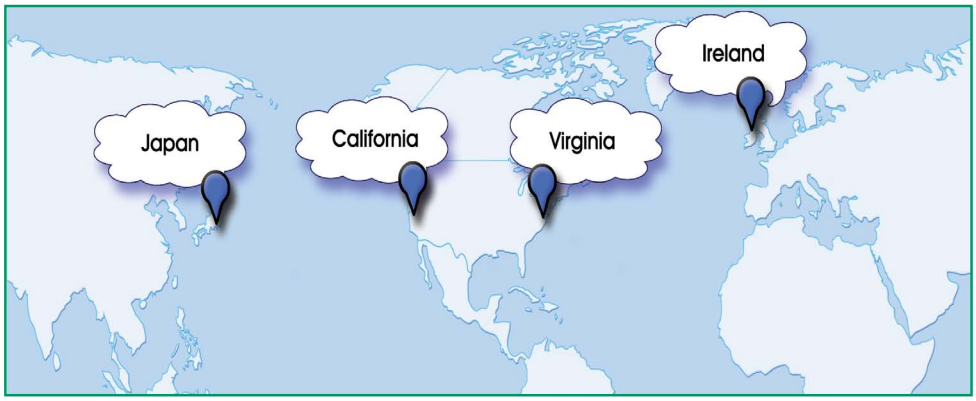


Figure 1. BASIS data replicated across the world

By Chris Hardekopf
Software Engineer

Each replication job runs in the background and reads from the replication log in order to propagate changes to the target BBj Services. Each target BBj Services independently remembers its current position in the replication log for restart capability. If the replication is synchronous, any user change operation such as file or directory creation, revision, deletion, etc. will block the user application, waiting for the background target to process the change. However, if the job is asynchronous, as BASIS recommends, the user application can continue without waiting for targets to apply the change. As the targets process the file/directory changes registered in the replication log, BBj Services removes those completed changes from the log to keep it as small as possible.

The replication log, therefore, allows for both the minimal performance impact of replication and safe and transparent handling of source and target BBj Services downtime. The performance impact is minimal since the most that enabling replication will do is append a change record to the replication log, no matter how many targets are configured or how slow the communication with the target. Downtime for the source BBj Services is easily handled since clearly there will be no new changes while it is down and all previous changes that have not yet been replicated to targets are preserved in the replication log. Downtime for the target BBj Services is also managed automatically, since any changes happening on the source BBj Services while the target is not available will be preserved in the replication log and sent to the target BBj Services as soon as it is running once again.

Seeing Replication in Action

BASIS is an example of a company who uses and relies on replication for continuous backups. They run their production BBj Services on the Amazon cloud region on the West coast of the United States. BASIS continuously runs replication jobs to three other Amazon cloud regions: the United States (East coast), Japan, and Ireland. This means that the replication jobs provide complete backups of all BASIS' production BBj data files in several places around the world. If necessary, BASIS could easily switch BBj Services production to a different region as shown in **Figure 1**.

Summary

Replication is such a powerful and easy-to-implement feature that every BASIS customer has reason to put it to work. It provides an invaluable safety net that anyone would pay dearly for if the situation ever required it. Files or directories, large or small...backups of any shape give extra security and a great deal of peace of mind. ■

Review these topics in the BASIS online help at

- *Replication Introduction*
- *Enterprise Manager - Replication Jobs*

Chef's Choice - the BUI Menu App

A common paradigm used for displaying options is the familiar menu – entrée choices at a restaurant, program choices to run from the Windows Start Menu, or a listing of tasks or modules to run from a typical Business BASIC program menu. The concept of a list of options is very effective, especially in the computer application world when programs are readily categorized and grouped by function. Users select a program to run from the menu program, which responds by replacing the menu screen with the desired application screen. When the user has finished with the selected program, control returns to the menu screen once again.

Are Menus Still the Best Choice?

A menu-based work flow is practical and ingrained in many of us, so there are good reasons to stick with it, even in a new model like web-based apps running on a mobile device. In contrast, in the traditional web page environment, we click on a link and expect it to open in a new browser window or tab. This is not a bad solution for tangentially-related content like web pages, but it gets cumbersome and confusing when dealing with a single application running in a browser. A smart phone exacerbates the problem as screen space is at a premium and navigating multiple web pages may be laborious and distracting. The goal is to streamline and simplify the user experience, especially when the program is a BUI web app running on a small-screen mobile device. So is it possible to retain the same effective menu paradigm in a BUI web app running on a smartphone? The answer is a resounding "Yes!"

BUI Web Apps are the Main Course

To illustrate the menu concept with a concrete BUI web app example, take a look at the "Phone BUI Menu App" demo. This app installs with the demos along with BBj® so anyone can view the underlying BBx® source code and CSS stylesheet. It is also published live on the Internet so anyone with a browser and Internet connection can run the web app without downloading or installing BBj. **Figure 1** is one example of the BUI Menu running on a smartphone. The code uses a fixed width for the main window and is somewhat flexible with the height based on the available screen real estate, requiring some tweaking for specific mobile devices. This accounts for the smartphone browser hiding or showing certain UI elements depending on how the user launched the app. For a more detailed discussion regarding screen layouts for specific mobile devices, see Let BUI Put Your App in Touch With Your Users on page 40.

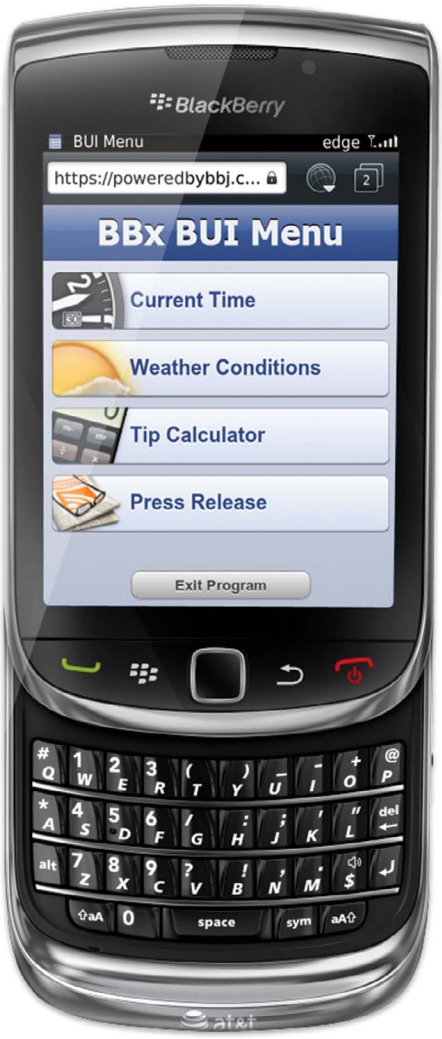


Figure 1. BBx BUI Menu running on a BlackBerry

Ordering From the Menu

To mimic the standard flow of the traditional BBx menu-driven application, the BUI app consists of a few utility programs >>

By Nick Decker
Engineering Supervisor

along with the main menu program. When the demo launches, the menu program displays the list of available utility programs in a series of vertically aligned graphical buttons.

In this simplified case, the user may launch only four programs from the menu – an HTML/CSS analog flip clock, a weather program, a restaurant bill tip calculator, and a BASIS press release viewer. The programs range in complexity from the rather simplistic clock with only a couple of controls, to the more full-featured tip calculator app that makes use of several different BBj controls. Although these are separate programs, the menu program launches each one via the SCALL function. The SCALL's parameter is of the form **`bbj <program_name>`**, meaning that the interpreter takes a shortcut and launches another instance of BBj instead of actually passing the command on to the operating system as it would for a true "System Call". This results in very fast execution of the chosen utility app and is very efficient because the SCALL consumes fewer resources compared to the similar BBjThinClient::clientExec method.

The Benefits of SCALL

Having the menu program SCALL the utility program makes a lot of sense since it happens quickly and has low overhead. But what really matters is that the chosen utility program uses the same display as the menu program. That is also desirable, as it helps to ensure that the transition between programs is as seamless as possible. In contrast, opening the chosen BUI app in another browser session results in the menu app page being pushed into the background while a new browser window opens to display the utility program. That in itself is not so disruptive, but exiting the utility app results in a nearly empty browser page showing a single link to reload the app. To get back to the menu, one would have to click on the browser's session icon on the bottom bar, tap to close the current page, slide the page carousel to the right to select the menu app page, then tap once more to go back to that web page. That whole routine is about as far from seamless as one can get, and requiring that many taps and gestures results in frustrated users and increases the potential for errors and mistakes. It turns out that there is a rather simple solution to this – maintain a single visible window regardless of which program is running.

Maintaining a Single Window

Because SCALLs are either synchronous or asynchronous, developers have a choice in how the menu program launches the utility. The concept of synchronicity – whether the main menu program waits for the SCALLed program to finish or not – has interesting ramifications for the BUI Menu demo. Because it was designed specifically to emulate a native smartphone app, the code does a few things that may be considered a bit unusual to accomplish this feat. In addition to creating the window without a title bar, the demo makes a concerted effort to ensure that only one window is visible at a time. When multiple windows are visible in a BUI program, BBj automatically displays a tabbed bar on top of the app to facilitate the act of switching windows. The tabbed title bar is very useful in normal circumstances, but does not accurately reflect a native smartphone application. Therefore, the demo must first hide the window associated with the menu program before SCALLing the selected utility. That way, the utility program can show its window normally and the tabbed window bar will not display since there is only one window visible. The trick, of course, is having the menu program show its window once again after the launched utility program has exited. If the menu program neglects to show its window, the user ends up staring at a blank screen.

Passing Control to the Utility

There are a couple of ways to ensure that only one window displays at any given time, based on which version of SCALL is employed. If the menu program uses a synchronous SCALL, execution of that program will halt until the launched utility program has exited. Therefore, the menu program hides its window, SCALLs the utility, and immediately shows its window once more. The synchronous nature of the program flow guarantees that only one window is visible at a time. However, if the menu program uses an asynchronous SCALL, it will have no way of knowing when the utility program has exited and therefore will not know when to show its window.

Although there are many ways of having the menu program and utility program communicate, such as via a socket, namespace variable, etc., it would require extra programming and would make it more difficult to add new utility programs to the menu. The extra overhead and complexity are certainly not desirable, but luckily a simpler alternative exists. The solution is to have the menu program hide its window, SCALL the utility program asynchronously, then RELEASE the BBj session. In order for this scheme to work, the utility programs must follow the same paradigm and hide their window, SCALL the main menu program, then RELEASE. This ensures that a program is running at all times and guarantees a single visible window. While the first solution is easier to understand and implement, choosing the best solution depends on the existing design and structure of your current menu system.

Summary

The BUI Menu demo is a good example of implementing a menu system for a mobile device, and illustrates a few of the special requirements and design decisions that are an important part of any mobile application development. This example is inspiring, as it shows what is possible when you take a traditional BBx menu-driven system and deploy it live on the Internet to browsers and smartphones EVERYwhere. ■

For more information
- Try the BUI Menu demo
- Read *Let BUI Put Your App in Touch With Your Users* on page 40

BASIS and paperPlace Have it in the (Paper) Bag

In Germany, stationery and office supplies is a tough market for mom-and-pop retailers today. Digital communication is replacing pen and paper more and more, both in offices and private households. And in this shrinking market, multi-national online retailers are gaining footholds, draining market share from local retailers. With a new concept called "paperPlace," BASIS teams up with industry association paperCompetence and their retailers to fight the behemoths with their own weapons.

paperCompetence is an industry association that commands the combined market power of 12 privately-held wholesalers in Germany and their 12,000 retailers, most of the latter being small- and medium-sized companies located all across the country. paperCompetence helps retailers stand their ground in a very competitive market environment, not only by supplying them with a large and diversified choice of goods, but also with innovative marketing concepts and with the means to support them in managing their businesses efficiently.

Step One: Simplifying Purchasing Processes

For instance, as early as 2002, paperCompetence had commissioned BASIS Europe Distribution to develop a sophisticated online store system that allows retailers to order stock materials directly on their office computer through the Web from their wholesaler, bringing the supply chain closer together, and saving both sides time and money. The first release of the system went live in 2003 and with ongoing development, has become an indispensable management tool for retailers. In addition, this tool is most valuable for processing sales and placing marketing campaigns for the wholesalers, promoting seasonal goods, publishing special offers, to name a few.

BASIS first designed the system on a PRO/5® platform. By operating on a common platform instead of isolated applications for each one of the wholesalers, they shared the investment cost and participated in a state-of-the-art system that gave them a competitive edge. Of course, the design of each store could be individualized to reflect the respective wholesaler's corporate identity. One of the technological challenges for BASIS was to allow for the real-time import of the different product ranges of 11 wholesalers – all of whom are equipped with different databases and ERP systems – while guaranteeing the 24/7 integrity and availability of the store system.

By Patrick Schnur
European Marketing and Public Relations

Step Two: Realizing Extra Sales Opportunities

While the first generation system made life easier for both retailers and wholesalers in their internal processes, it didn't do anything for the market success of the retailers. "*In 2005 or 2006, someone had the idea to extend the concept of the existing application – develop once, share it with many – to the retailers within the paperCompetence community*," recalls Stephan Wald, Director of Sales and Technical Service for BASIS Europe Distribution. "*As with any good idea, nobody really remembers who had it first*," he smiles. But the idea's appeal was obvious from the start. Based on the existing application, retailers could have their own individual web stores in a snap, "filled" with the product range of the wholesaler of their choice. This virtual store would mean more sales opportunities with up to 30,000 additional articles on sale, without any need for extra storage, and the possibility to use drop-ship delivery, even saving the sweat of packing parcels and printing out invoices.

Furthermore, the second generation application would not leave them powerless anymore against online retailers. These retailers kept scooping away customers – usually regional small and mid-sized companies, doctor's practices, stores and the like – who found it more comfortable to order from anonymous web stores. Thus, the idea for "paperPlace" was born. > >

Meeting Consumers' Demands

Certainly, in several aspects, a store application for end customers must meet tougher requirements than a B-to-B solution; on the Web, the competition is always just one click away. *"First, speed is a crucial issue in B-to-C. We knew from similar projects that we could deliver a full-text article search of 50,000 articles in less than one second,"* explains Wald. "*A second issue is to present the goods attractively with pictures and text, have an eye on usability, and to supply ample filter options so that users can easily find what they are looking for, like finding their favorite envelope from a choice of thousands. And thirdly, we needed to implement the payment options they are familiar with, such as credit card or PayPal payment*."

On the other hand, the store back end needed to be easy to handle for the retailers. "We developed a dashboard where our retailer can monitor all active orders, can check the status of dropship deliveries, key in their own articles if they so wish, and much more," Wald adds.

Technologically, the solution is a completely new development, developed in the OOP paradigm with BBj®, using Tomcat as an application server and the database model of AddonSoftware® by Barista®. By using this, BASIS profited from the pre-fabricated, building block modules that AddonSoftware provides, saving tremendous development effort, testing and other time-consuming detours. What about BUI? "*Each horse for its course,*" Wald replies. "*It was a paperCompetence requirement that we needed to deliver a solution that would run flawlessly even on very old legacy browsers. BUI requires JavaScript and HTML 5 that – while available on any modern browser for both mobile and desktop – was for now, a bridge too far for paperCompetence.*"

The front end was designed by an external agency specializing in web user interfaces. "*We believe that the look and feel of a web store contributes a lot to its acceptance with end consumers*," Wald explains (see **Figure 1**).

The Store is the First Building Block For an ERP System

In September 2010, BASIS Europe introduced the store at several industry trade fairs where it spurred a lot of interest with retailers. "*We hope to operate about 400 of these stores in the long run*," explains Klaus Danne, paperCompetence Managing Director. The solution is based on the SaaS (software as a service) model; the service is hosted in the cloud so that access times are equally fast from all regions of Germany, whether in the city or in the countryside. The list price starts at 49 Euros per month (~ $66 USD) for the basic module, which includes technical support from BASIS and regular product updates.

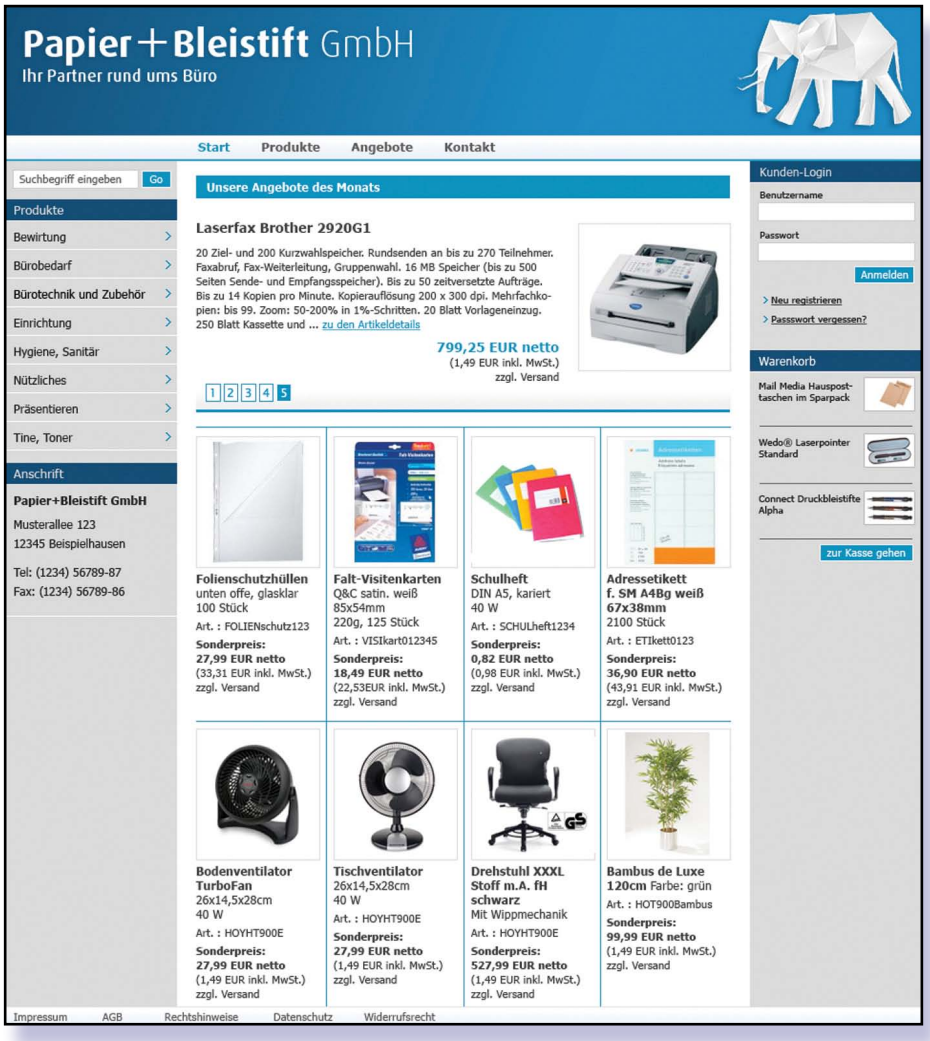


Figure 1. paperPlace web interface

The solution offers the retailers additional modules (Invoicing, Cashier, Stock and Store Management) which allows them to strategically extend their online sales activities beyond the standard product range of their paperCompetence wholesaler, and to extend the IT infrastructure up to a complete ERP system, all in the cloud and based on a monthly rental fee. It goes without saying that BASIS built these extra modules with building blocks from the AddonSoftware ERP package.

More Industries in Focus

Programming the second generation paper store in the object-oriented paradigm has the additional advantage that one can easily adopt the solution to the product ranges and specific challenges of other industries. There are already talks and ideas to extend the solution to a chemicals wholesale business and to a company in the packaging industry.

Summary

The paperPlace is an excellent example of leveraging the building blocks concept central to AddonSoftware by Barista. By using the BASIS toolset, which includes the OOP language BBj as well as AddonSoftware ERP building block modules, we could develop this solution in significantly less time than in many other languages such as .NET or PHP. ■

Perfection in the Cloud

Sequel to Our Salvation is in the Cloud

Software development is hard enough by itself, but testing and deploying software can be equally difficult. Most developers have discovered the power of virtualization through the use of VMware, Xen and other virtual OS vendors on their own back office servers. Cloud computing gives companies the power of this same virtualization, but with unlimited hardware capacity and flexibility. Most BASIS customers only have one production server and if they have a test server, it is seldom identical in size and speed to the production server. Differences between the test and production servers expose the company to insufficient and inadequate testing environments at a minimum, and to mismatched configuration nightmares at a maximum.

Dr. Kevin W. King
President & CIO

Identical Cloud Machines to the Rescue

The cloud architecture utilized by Amazon and many others, gives developers access to ~40,000 servers at each location or availability zone in each region. Since all instances of an image are Virtual machines and each instance of an AMI (Amazon Machine Image) is virtually identical, the test machine can be identical to the production machine. This solves the problem of testing in an environment that is almost like the production machine, but not exactly.

The next challenge is to test with the same database and software that is running in production. Through the magic of the cloud infrastructure, developers can create another drive that is exactly like the drive that is mounted to the production machine. Now, for the first time since the beginning of software development, even the companies with the smallest budgets can afford to test their software on identical OSs and hard drives before deploying their new software into production. Since most cloud providers only charge for each hour their machines or hard disks are in use, developers can perform comprehensive testing that all but eliminates the need to fail back to the old system. However, the cloud simplifies the fail-back process as well. For example, if you have two identical systems available in the cloud, while you are running your regression testing and doing an upgrade, if you do happen to find a problem with the new software, you can fail-back to the old system within a couple of minutes because you did not have to tear up the original during the upgrade.

Deployment Solutions and Benefits

BASIS uses large well-known cloud infrastructures like Amazon and Google to reduce the risks in cloud deployment. Through the use of content delivery networks (CDN), auto-scaling and load balancing, the dependence on a single machine is spread to a cluster of machines in numerous regions and multiple availability zones in each of those regions. Google's page speed service can allow BASIS' web sites to be accessed from a region nearest to the client browser. Amazon's Cloud Front CDN gives all of BASIS clients access to product, documentation, and videos from a cloud server farm nearest the client. >>

While this CDN architecture gives the BASIS customers the most pleasant surfing and downloading experience, it also adds to the robustness of the deployment architecture because it means that there are multiple copies of this data on multiple networks. This vast redundancy also provides speed and robustness never before imagined, while expenses are contained because the cost is calculated by the volume of the data stored and the frequency of the access, which is much better, easier, and cheaper than the traditional model of building and maintaining servers at multiple locations around the world.

BASIS' Drupal-powered websites and Bugzilla systems all depend on MySQL databases that are configured to run redundant databases in multiple regions. Products that depend on the BASIS language and RDBMS utilize the built-in database replication functionality to maintain real-time copies of the database on both the East and West Coasts as well as copies in Europe and Asia. Through the use of Amazon's Route 53 DNS, BASIS is able to pragmatically move access to their production servers all around the world with no more than a 5-minute delay in rerouting of the DNS traffic. By utilizing all of the modern day technology, BASIS has been able to eliminate many of the human limitations and flaws from their production facilities. Using as many automatic fail-over systems as possible makes it feasible for BASIS to maintain the 7/24 up-time systems that their customers expect.

Closing in on Perfection

Perfection is a strong word and may not ever be appropriate when used in conjunction with technology, but it is easy to see and say that the cloud infrastructure is the closest thing to perfection that this old technologist has ever seen or dreamed of experiencing. Having the ability to have identical hardware, software, OSs, and configuration for QA, testing, and deployment is the closest thing to technology nirvana one can hope to attain. When adding the CDNs and auto monitor and fail-over software to the mix, it is hard to imagine a sweeter system. Who knows what the next generation of technology pioneers that come along behind Steve Jobs, Bill Gates, and the Google Guys may figure out about how to make technology even more dependable and more robust. However, in 2011, the cloud is the closest thing to building, testing, and deploying perfection that BASIS has found.

Try the cloud out for yourselves; investigate the various options like BASIS did or consider taking advantage of our learning curve by contacting us to learn about our Cloud Hosted Offerings.

Read the prequel BASIS Advantage article *Our Salvation is in the Cloud*

Let BUI Put Your App in Touch With Your Users

There are a great number of reasons to write your next BUI web app for a smartphone, but probably the most compelling one is that you and your users can have immediate access to the web app anytime, anywhere. Today's smartphones have fast browsers with optimized JavaScript and hardware graphics rendering that are more than capable of running BUI apps with ease.

Creating a BUI version of your desktop app is a definite advantage as users can continue to work and be productive outside of the office – in line at the grocery store, walking on the treadmill at the gym – you name it! A full-blown BUI app is not limited to a simple static report, but is capable of being a fully immersive, and more importantly, interactive experience where the user can make payments, record receipts, adjust sales priorities, and even reroute shipments with just a few taps of a finger.

This article is a simple guide to help developers fearlessly take the plunge and put their apps in touch with their users!

Targeting the End User's Device

Now that we are sufficiently motivated <grin>, our first inclination would be to fire up the BASIS IDE and get our fingers dirty by starting to write some code. But before we open the SYSGUI device, we should stop for a minute and think about how the smartphone differs as a client compared to the traditional desktop computer. There are numerous differences, but the prominent ones are:

- Smaller screen
- Slower CPU
- Touch interface instead of a mouse
- Higher network latency and lower bandwidth
- Variable screen real estate and orientation
- Tiny simulated or physical keyboard
- No cursor
- No hover effects
- No right clicks

By Nick Decker
Engineering Supervisor

Reading through the list may have dampened your enthusiasm a bit, but none of these are deal breakers. Instead, they are simply challenges to keep in mind while designing our web app. Some of them may even lead to relatively simple code solutions with dramatic and satisfying results. More importantly, they underscore how the differences in the target device encourage us to thoughtfully plan and structure our app so that end users can navigate the app in a simple and effective manner.

Designing for Smaller Screens

Web designers have been dealing with many of these same issues over the years, so we should take a cue from them and write a web app that is well suited for a smartphone. To illustrate the point, take a look at the comparison of the Amazon website when viewed on a desktop with it viewed on an iPhone. **Figure 1** shows the difference between viewing the desktop site on the small screen versus the site that Amazon optimized for a mobile device.

Figure 1. Desktop (left) and mobile (right) versions of the Amazon website on iPhone

This comparison illustrates some noteworthy differences from which we can learn some things of value.

- **The mobile version of the website optimizes the limited screen space by telling mobile Safari to hide its Address Bar.** This is a smart move, and thankfully it is one that BBj® takes care of automatically for the BUI developer. BBj does this without any effort on the developer's part in order to maximize viewing area of the application.
- **The amount of information displayed on each page is strikingly different.** The traditional desktop page has dozens of tiny links that are impossible to accurately tap with your finger, regardless of how hard you may have concentrated and aimed. This highlights a couple of the most important guidelines:
  - Limit screen content to only that which is critical
  - Increase the size and spacing of controls

Because a smartphone's screen is both physically smaller and has fewer pixels, we should restrict screen content to information that is absolutely critical. Attempting to cram a full-size application onto a smartphone will almost assuredly lead to a frustrated user that spends more time trying to interface with the app instead of actually using it and getting things done.

Putting the User in Control

The good news is that we do not need to conduct usability studies and put our app in front of multiple focus groups to determine how best to construct an efficient user interface – all of that work has already been done for us. The takehome points are few and pretty easy to implement. In a nutshell, we should make tap-able elements such as buttons large enough and allow for enough room between >>

screen elements to avoid finger mis-taps. Fingers are fast, but interestingly enough, they are not usually as accurate as a mouse. So a user that has no problem clicking on the typical button that is 23 pixels high in a desktop application will have difficulty tapping on the same control on a small screen. Therefore, we should make buttons significantly larger for a smartphone app; 44 pixels high turns out to be a pretty good average. While that height is almost double the typical desktop button height as shown in **Figure 2**, it makes sense.

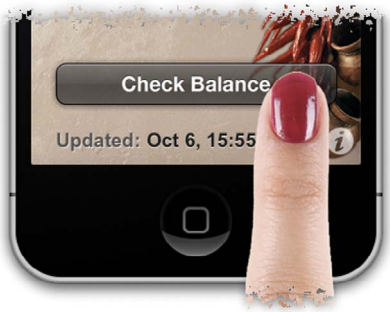


Figure 2. Big buttons provide easy targets

Many smartphone users operate their devices with one hand and oftentimes tap on buttons exclusively with their thumbs. Although tapping on a button may seem easy while waiting in line at the grocery store, accurately hitting the intended target may be quite another matter while jostling about on the metro. That fact leads nicely to the second point: arrange controls with enough space between them to make them easily touchable and to prevent the user from accidentally tapping a neighboring control.

The Tiny Keyboard

As fast as teenagers may be at texting their "BFFs", not all smartphone users enjoy the same level of keyboard competence and some are downright snails when it comes to how fast they input text. To combat this, the next goal is to allow users to accomplish as much as is feasible without ever having to bring up the keyboard. This means that traditional text-based controls are not necessarily the first choice when it comes to input. Instead, buttons, spinners, and graphical custom controls like the rating system and guest count controls employed by the "BUI Tip Calculator" (see **Figure 3** and accompanying article Adding Style to BBx Web Apps With Custom CSS on page 26) can dramatically reduce the number of taps, eliminate the popup keyboard, and result in a more enjoyable and productive user experience. Directly manipulating screen content via graphics is no longer considered extravagant, it is now the mark of a well-designed and efficacious app.

Figure 3. Specifying the number of guests in the dinner party with a custom graphical control

Single or Multiple Windows?

Typical desktop applications have a title bar and may show multiple windows at once. These windows may be modal, or in some cases the user is actively encouraged to interact with many windows in order to facilitate copying information from one screen to another. When designing our BUI app for the mobile Web, one of the more fundamental design decisions is whether to allow multiple windows or constrain the app to a single window to more accurately emulate a native smartphone application. The best choice, given the particulars of our app, may be to display multiple windows for increased flexibility. However, the tabbed window bar does consume more screen space and, if overused, could possibly confuse or frustrate mobile device users that are unaccustomed to it. When choosing the single-window paradigm, we need to be aware that the tabbed window bar will display automatically if more than one window is visible at once so we must be sure to write extra code to prevent this. **Figure 4** shows a comparison of the BUI Menu program running as a single-window app and as a multi-window app.

Figure 4. The BUI Menu demo program with and without the tabbed window bar

Screen Size and Orientation

Because screen space is at a premium on a smartphone, making the best use of every pixel is imperative. That means that we'll have to figure out what the target device's screen resolution is and decide whether to support both portrait and landscape orientations. BBj gives us all of the necessary information to gather the info and react accordingly via the following:

`INFO(3,6)` - Use to determine if the app is running in BUI

`INFO(3,8)` - Use to determine the web browser

`INFO(3,8)` - Retrieves the navigator.userAgent string from the browser when running in BUI

`FIN(sysgui_chan,IND=0)`- Use to determine the screen size

Armed with this information, it is possible to determine, for example, that our app is running in BUI on an iPhone in portrait orientation. It's important to realize that the available screen size can vary – not only between devices (such as comparing an Android device to an Apple iOS device), but also when the device orientation changes. The Android platform, for example, runs on a wide variety of hardware and supports at least four different screen resolutions. The screen size also differs depending on how the user launches the app. Mobile browsers typically reserve portions of the screen for common UI elements such as an upper URL address/search bar and a bottom button bar that handles forward/backward navigation, bookmarks, sharing, and more. In order to maximize space for a BUI app, BBj automatically directs Mobile Safari to hide the top URL bar resulting in an extra 320x60 pixel area that is now available for use by our app. The bottom button bar is still shown, as there is no way currently for BBj or any web page > >

to force Mobile Safari to hide it. However, there is a way for the user to accomplish this manually – simply save the BUI app's link to iPhone's home page, subsequent launches from that home screen icon will run the app in full screen.

Running the app in full screen results in a couple of benefits. First, the BUI app now has access to another 320x44 pixels of precious screen space as shown in **Figure 5**.

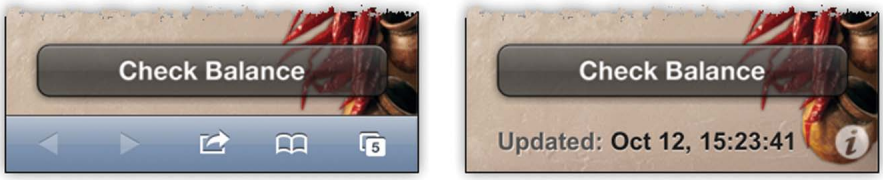


Figure 5. Maximizing screen space previously consumed by the browser's bottom button bar

The second benefit is that users now have an icon on their home screen that they can use to gain quick access to our app. Not only will it be easier to launch our app from the home screen, but our app also shows up in the phone's search results. By default, iOS will create an icon that is essentially a small version of a screenshot of the BUI app. This may suffice for some pages and apps, but the image is so small that it is usually quite difficult to see much of the detail. BBj helps out in this area as well, by allowing you to specify a custom icon when defining a BUI app in Enterprise Manager. **Figure 6** shows two copies each of the BUI Menu app and the ChileCustomer Balance app. The first icon for each app is the default screenshot version, and the second icon is a custom image that we associated with the BUI app in Enterprise Manager Customizing the icon for our app allows us to use an icon that has visual impact and is immediately recognizable, helping it to stand out amongst the other icons on the home screen.

Figure 6. Comparing default and custom BUI App icons on an iPhone

Handling Orientation Changes

Many smartphone apps are aware of the device's orientation and adjust their layout automatically. A BBj BUI program can do the very same thing by registering for the ON_SCREEN_RESIZE event that fires any time the screen resolution, or the browser's client area in the case of BUI, changes. On a desktop, this typically occurs when users resize their browser window. On a mobile device, this event typically occurs when the orientation changes, as the available screen size is different in portrait mode versus landscape. After registering a callback for that event, a BUI program can query the screen width and height via the aforementioned FIN() function to determine the size and orientation. Once the app determines that the orientation has changed, it can modify the control layout on the screen or, in the case of resources, load a new resource file that was specially built for the new orientation. Many end users have come to expect different layout options, and while supporting multiple layouts means added work and development time, it certainly is thrilling to see our app change its UI in response to rotating the phone!

Use Your Finger, Not Your Mouse

Regardless of whether a user launches a BUI app from their phone, tablet, or desktop browser, the app executes in much the same way. However, how users interact with the program can change dramatically depending on whether they use a mouse on their desktop or their finger on a touch screen. As mentioned earlier, a touch screen on a smartphone or tablet does not have the concept of a "hover" state and since you navigate without a mouse but with your fingers, the only interaction with the device occurs when you contact the screen.

Hovering your finger close to the screen doesn't register any events, at least not with today's hardware. It is something to look forward to in the future, though, as designers experiment with proximity sensors. In the meantime, carefully review any existing applications that you plan on deploying in BUI. During this review, look for areas to address that are incompatible with touchscreen devices such as concepts and events that are mouse-specific. In addition to hover events, BUI programs should not rely on mouse events such as the Mouse Move/Enter/Exit events. Right click events are not generated either, so if your programs rely on users selecting commands from popup menus launched from a right click event, you will have to make those commands accessible from the program menu or some other means.

Summary

Designing a BBj BUI application for a smartphone device brings with it several new concepts and challenges that are common to touchscreen devices. Although these may appear to be daunting at first, the challenges will gradually morph into a learning and ultimately rewarding experience. As a developer, there is nothing quite like seeing your code running in the palm of your hand. Putting that power and capability in the hands of your users is even more rewarding, and reaping the benefits from a "constantly connected" workforce ought to be motivating enough to "the powers that be" to get you started right away! ■

A Tab(let) a Day...Makes the Doctor's Day

Heimbas, a leader in the German nursing-home software market, services more than 700 customers. Development of the Heimbas solution began in 1988 followed by its first Windows release in 1995, based on Visual PRO/5®. The third generation, written in BBj® (see **Figure 1**), saw the light of day ten years later in 2005. Read on for the rest of their story.

Just What the Doctor Ordered

Founded in 1973, Heimbas was one of the first ISVs to offer integrated ERP software for nursing homes, including modules for administration, invoicing, accounting, costing, nursing documentation and duty roster management, all developed by the same development team. Today, they still hold firm to their original philosophy, "to offer a complete solution for the nursing business – integrated and fully-featured, yet particularly easy to use." The product, called "Heimbas" (*Heim*=home; *bas* from data'bas'e), was so successful that managing director Armin Kehler decided to name the whole company after it.

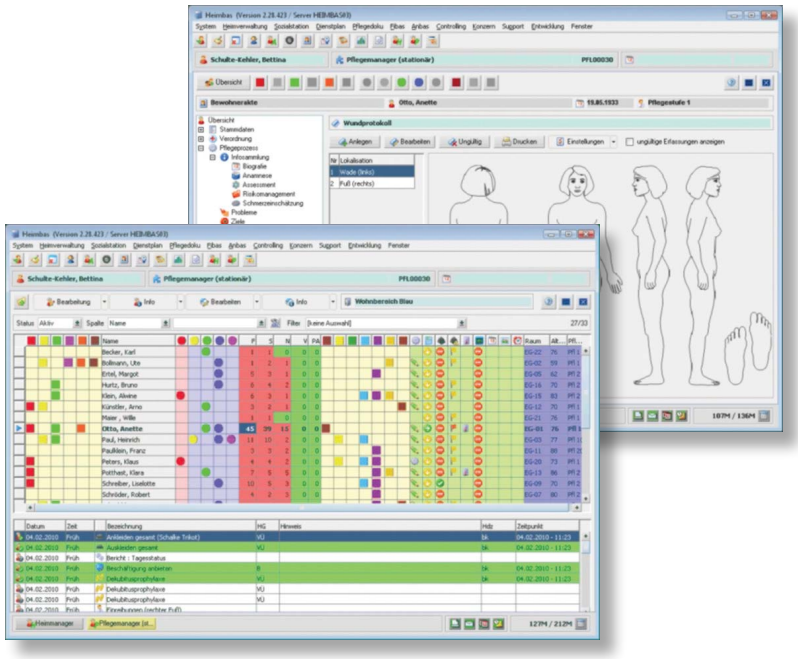


Figure 1. Third generation Heimbas Windows written in BBj

Electronic Patient Files

As everywhere else in the world, the German health industry is facing enormous cost problems. Heimbas understood at an early stage that mobile computing could offer nursing homes and home nursing services huge cost saving benefits, and acted accordingly.

Nursing homes are under pressure from two sides: a host of statutory regulations, and rising costs in the sector force all providers of health care services to become efficient without sacrificing quality of care. One way out from under this pressure is the concept of the electronic patient file, which stores all relevant data about a patient electronically, saving redundancies in data entry as well as read and transmission errors and allowing for easy data retrieval. As nursing homes are under close scrutiny from the public health insurance system, it is an important point that electronic patient files are more audit-proof than paperwork.

Touchscreen Nursing Monitors

A few years ago in a first attempt to complete the data chain, Heimbas implemented touchscreen monitors in nursing homes at the door of every patient's room for easy access to the nursing documentation module of Heimbas software. When nurses come in to a room, they simply activate the monitor and authenticate with a connector pin that they carry on a cord to see the patients' needs in an instant. As they leave the room, they can document the work with just a few clicks – no need to look for a pen, or specifically one that works; no need for legible handwriting, no need for forms that can be lost or damaged, no risk of forgetting to record the data or remembering it accurately when called upon to record it at a later date. When completing the entry, they simply pull out the connector pin to sign out automatically and conceal the patient data from unauthorized view.

Heimbas Anywhere

The monitors were a huge success for Heimbas because it saved nursing homes a lot of time and work while dramatically improving their care documentation. Several homes asked Heimbas to take the next step: to enable head nurses to access the patient data not only from their desktop, but from anywhere, to improve internal patient-centered communications. Application for such a solution is wide and varied throughout nursing homes and hospitals, alike. For example,

- **At the bedside**. On the ward rounds, all information about patients would be available at their bedside. The head nurse could retrieve the latest temperature charts entries for the doctor who could then decide any medication changes on the spot – all information/data would be stored in real-time in the patient file. Media discontinuity only occurs for legal reasons: The prescriptions have to be printed out, because the doctor needs to sign them.
- **In admissions.** When admitting acutely ill patients, the staff could collect the most important data on the spot – not just their name and address, but vital information such as known medication allergies or pre-existing conditions, and so on. This information would then be immediately available to all doctors and nurses. > >

By Patrick Schnur
European Marketing and Public Relations

Hardware

Hospitals and software companies had been experimenting with mobile patient documentation for quite some time but the best option had only been keyboard-equipped laptops, clumsily rolled from room to room on mobile cabinets, hardly a workable solution. The time was not yet ripe for mobile options...until now. Today, with modern touchscreen tablets, devices are smaller, lighter, and much more comfortable and easier to handle, the solution seemed to be at hand.

Software

The software side of technology has developed just as remarkably. Heimbas opted to use BASIS' revolutionary BUI technology for the new solution. "*All alternatives, such as native iPhone or Android apps, were out of the question,*" explains Armin Kehler. "*The reason is that not only did we know BBj inside out after having worked with BASIS tools for more than 20 years, but our customers are used to running our software on their hardware and OS, and we wanted to keep it that way. With BBj and BUI, we remain platform-independent and operating system-agnostic, without the need to re-develop the same solution several times over.*" With their BUI solution ported from the existing software package, it took Heimbas only 30 project days to complete the transformation, "*...just as we had planned!*" says Kehler.

One of Heimbas' main competitors uses .NET. "*We certainly have an edge over them. Using BBj,*" Kehler explains, "*we are operating system independent, and we don't need to worry about any release changes around the products we use."* And then there is easier deployment. *"We don't have to update at every single desktop terminal, but just once on the customer's server, and that's it.*"

HEIMBAS®
Softwarelösungen
für das Sozialwesen

Summary

The value of the flexibility of BASIS' interpretive language, which can execute code in a JVM or in a JavaScript/HTML 5 capable device without a JVM, such as an iPad, is the technological secret enabler behind the Heimbas success story. Existing GUI code can run on a mobile device without modification through the magic of the interpreter. BASIS, helping you deliver tomorrow's solutions today!

For source code and a sample of incorporating legacy business logic in a mobile application, see the BUI Mortgage Amortization application at links.basis.com/buidemos. ■

More Miles per SQL Gallon

BBj® 11.0 now supports SELECTs from nested SELECTs as well as SELECTs from stored procedures. These enhancements provide developers with additional leverage on those finely-tuned SQL statements and adds functionality to previously written stored procedures. Getting more out of those SQL statements and stored procedures is now more efficient and even easier than ever before, thereby unlocking access to all your production data to modern SQL-based report writing tools such as iReports. >>

By Robert Del Prete
Quality Assurance Engineer

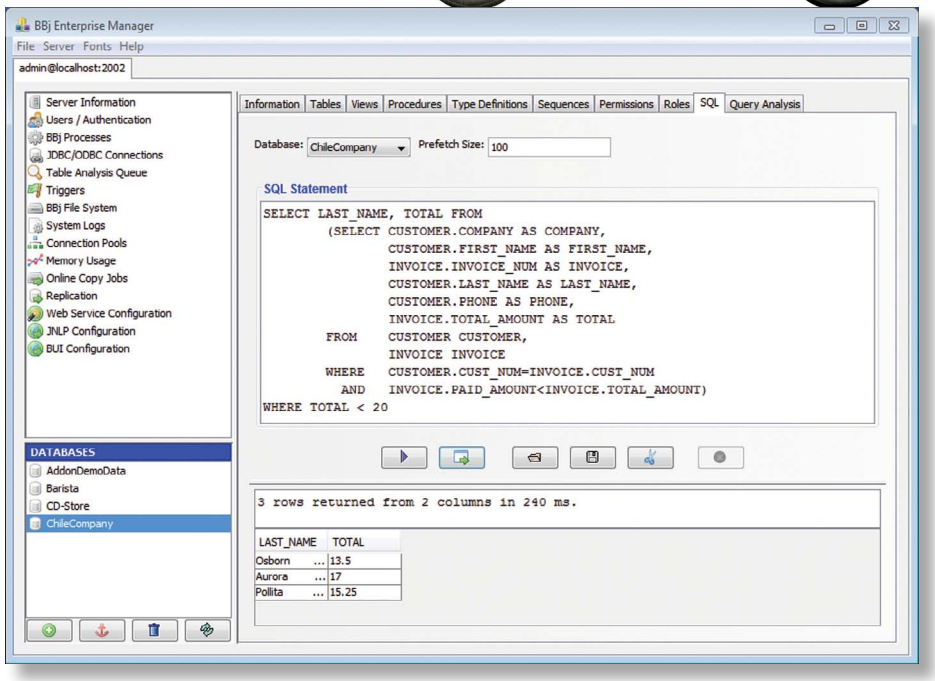


Figure 1. Using the result of a nested SELECT statement as though it were a table

SELECT FROM Nested SELECT

Using the result set of a SELECT statement, developers can now treat that result set the same as any other table in their SELECT statements. Think of it as treating the SELECT statement like a VIEW, although without the requirement to first create the view in the database. This can speed up your development cycle and applications as developers can execute complex queries on the fly without the need to manage and delete temporary views and tables. In addition, they can nest queries to any depth without any limit on the number of nested SELECTs used in any specific query. **Figure 1** shows an SQL query executed in Enterprise Manager that takes advantage of the nested SELECT syntax to extract a smaller result set from an existing query.

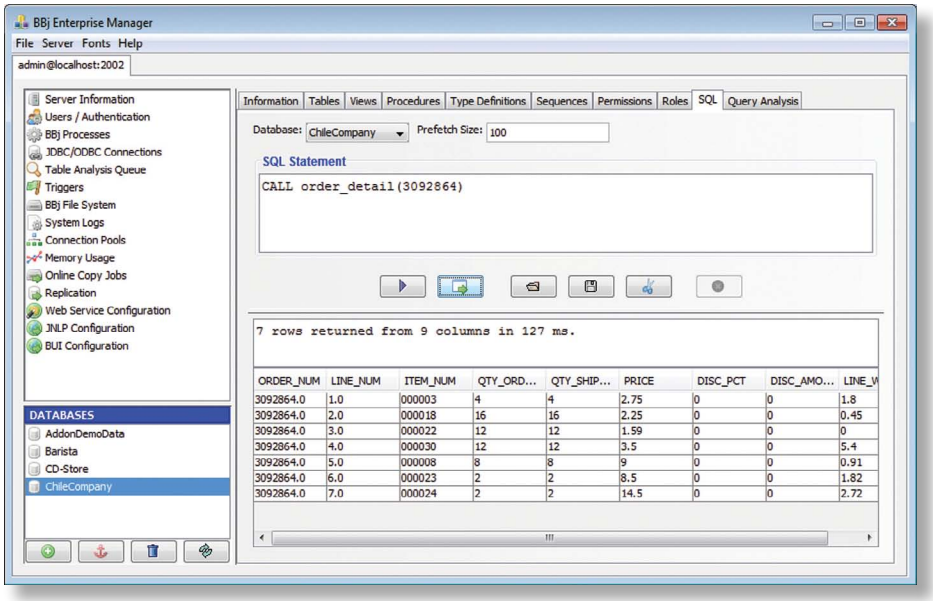


Figure 2. A stored procedure that generates current Order Detail information from MKeyed files

SELECT FROM SPROC

A stored procedure (SPROC) consists of a function or procedure written in BBj and embedded in the database. SPROCs can return result sets that are dynamically generated by the code within the procedure. SPROCs can generate results using READ RECORD, leveraging the original high-performing design of the data structures of the legacy production system, and return results usable by SQL. In BBj 11.0, BASIS introduced the ability to perform a SELECT <FIELDS> FROM <MY_CALL_STATEMENT>. This allows developers to treat the result of a SPROC call as a "table" in an SQL statement and provides the ability to further filter, sort, and group SPROC results as well as join those results with other tables in the database. **Figure 2** shows the result of a SPROC using the traditional SQL CALL syntax from the Enterprise Manager's SQL dialog.

A developer can treat this same SPROC's result set as just another table in the database and can use it as a part of a more advanced SQL query. **Figure 3** shows the result of joining the output from the Order Detail SPROC with the Item table as a convenient way to include item descriptions in the result set. This adds new flexibility to pre-existing SPROCs, as their output can be filtered and even augmented without having to change the underlying BBj code that defines the SPROC.

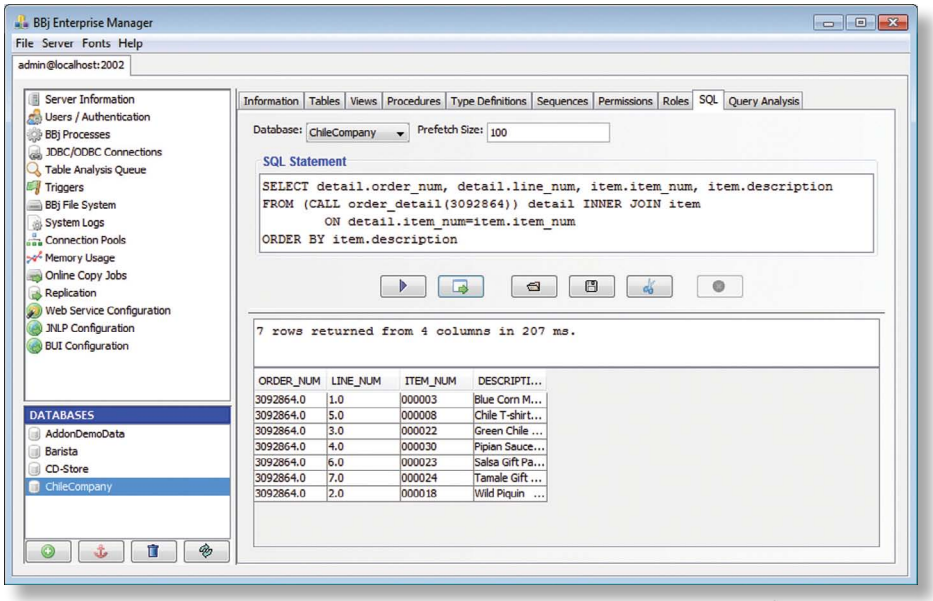
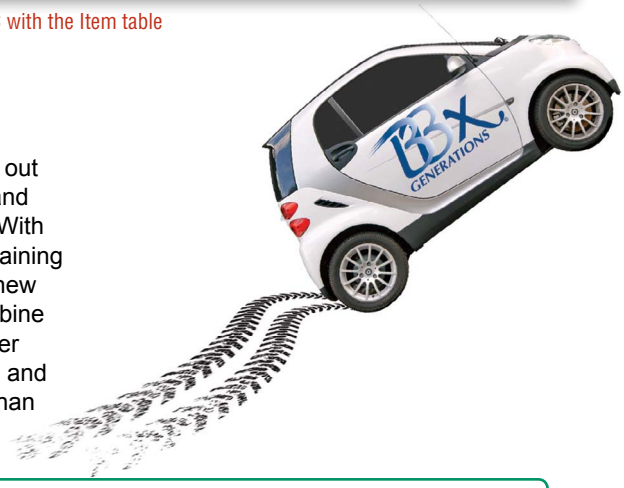


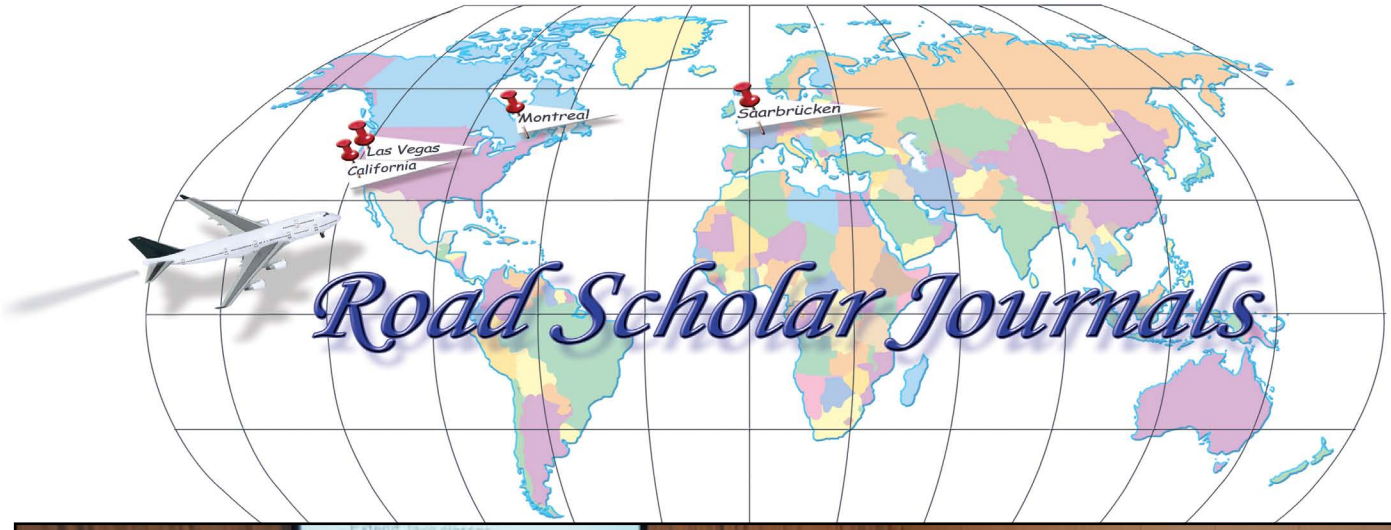
Figure 3. Joining the Order Detail SPROC with the Item table

Summary

You can now get additional miles out of your current SQL statements and stored procedures using BBj 11. With added support for SELECTs containing nested SELECTs along with the new functionality of being able to combine stored procedure results with other tables and SELECTs, developers and power end users can go further than ever before! ■



Refer to the BASIS Advantage article *Using Stored Procedures to Add Business Logic to the Database*



TechCon Europe 2011 – Learning from the people who make it happen

Saarbrücken

On June 7-10, BBj® developers from across Europe gathered in Saarbrücken, Germany to meet and see what's new and cool about BBj 11.0. The BASIS team of experts, from both BASIS US and the hosting team from BASIS Europe, shared how the many new functions and features in BBj and Barista® Application Framework make developers and implementers lives so much easier and more productive.

Cloud computing was a main focus of this year's conference. Attendees saw presentations that showed them that the cloud is much more than a buzzword or "pie in the sky," but a powerful way of collaborating smarter, both within a company and beyond. Since the cloud was the key focus of the TechCon, BASIS presented all demos "live" from the cloud. Bandwidth was not an issue; a normal household DSL connection was sufficient.

Among the takeaways:

- Integrating cloud capabilities into BBj, as BASIS did with version 11, helps customers cloud-enable apps without leaving their favorite development framework.
- Google Apps, a bundle of office suite-like functionalities, is now accessible from BBj. The GApps Utility, a BBj application that provides access to Google App Services in the cloud and an API for Google Docs, enables users to collaborate simultaneously and in real-time regardless of their physical location.
- Other Web-based concepts such as Web services or geolocation, are ready for integration into modern BBj as well as legacy applications.

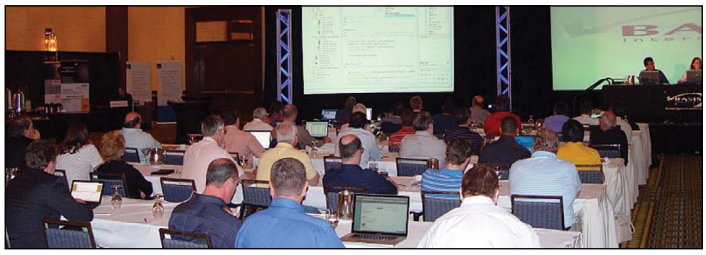
Of course, BUI and its ability to bring BBj/Barista applications to a web browser without any changes to the source code was another major highlight of the conference. Participants saw first-hand how to enhance the look of a BUI application using cascading style sheets. Database replication was another much-discussed feature of the new BBj release, giving BASIS customers more options for improved security in various business scenarios, including the ability to replicate their BASIS data in the cloud or elsewhere on *terra firma*.

TechCon Europe provided valuable experiences beyond the rich technical sessions. BASIS invited participants to an evening at the Saarbrücken Castle, offering the best of Europe: A Mediterranean barbecue, German beer, and French wine, complete with a post-dinner rain shower that gave everybody a typical German summer evening but without spoiling anyone's good time. On Thursday and Friday, BASIS held hands-on workshops for participants to learn more details about the technologies presented during the conference.

From the responses we received after the conference, BASIS customers were very satisfied with the TechCon and plan to come again next time. One attendee, new to BASIS' technology and a first-timer to TechCon Europe, was particularly enthusiastic in an email he sent a few days later, "*I thoroughly enjoyed all aspects of the conference and the city of Saarbrücken. The presentations were good and informative and the training was excellent. All the people at BASIS were great...you made me feel welcome in your city.*" Already he has implemented what he learned, "*[I] Have written my first GUI object. Planned others. Looked into the cloud and have investigated Barista and the AddonSoftware that comes with downloads. I can see opportunities there.*"

Looking for opportunities? Download the conference slides at links.basis.com/tceu11. ■

By Patrick Schnur
European Marketing and Public Relations



Las Vegas

Viva Las Vegas TechCon2011

There are numerous films about Las Vegas but few are more famous than the 1995 production *Leaving Las Vegas*. In contrast to that title, it was with great anticipation that BASIS was arriving, not leaving, Las Vegas, for TechCon2011 that began on Monday, May 9th. BASIS borrowed the name of another notable film from 1964 to present our 'Virtualization Is Value Added' theme - *Viva Las Vegas.*

TechCon2011 sessions focused on just how easy it is for our customers to "Step into the Cloud with BASIS"

By Laurence Guiney
Senior Account Manager

with BBx® generations applications – PRO/5®, Visual PRO/5® or BBj®. Over the course of TechCon, BASIS presenters showed attendees how they could use the BASIS toolset to create visually appealing applications, how to get their applications into the cloud, and tips on how to administer their cloud and terra firma-based applications. While these presentations were all well received, the three topics that generated the biggest buzz at the conference were how BASIS revolutionizes web apps with our Browser User Interface (BUI), the introduction of the BASIS file and database replication feature for PRO/5 and BBj applications, and the amazing multi-table nested SQL select capability of the new Query Definition Tool in Barista.

Of course it was not all work in Las Vegas as TechCon attendees were treated to some excellent entertainment that kicked off on Monday evening with magician Aaron Radatz. The laughter never seemed to cease on Tuesday evening with comedian David Crowe. It was apparent a good time was had by all as more than a few people were still talking about their favorite lines the next morning at breakfast.

Based on the feedback we received, attendees were very pleased with what they experienced at TechCon 2011. Hopefully what "happened in Las Vegas" will not stay in Las Vegas and attendees will take the knowledge they gained and begin using it to enhance their applications. If you are interested in the highlights from TechCon2011, read each of the feature articles in this edition of *The Advantage* and visit our Java Break library which contains some sessions in which we showcased highlighted topics presented at TechCon. ■

California

Post-TechCon Road Trip to California

Nico and I hit the road in May to visit several BASIS partners and end users in sunny California. The weather was especially pleasant as was our time with customers. We had a busy schedule and covered a lot of the southern California region from Santa Clarita to La Crescenta, then onto Laguna Hills and Lake Forest area, and finally to Chino.

Having the opportunity to spend face-to-face time with customers is invaluable. Every time we visit with our customers, we learn more about their business, needs, and challenges; and what they need from us. As a result, we are often able to share features and functions of BASIS technology that already meet the specific challenges of the customer. With each meeting, our goals were different as every partner and end user is at different stages of deploying BASIS technology.

By Gale Robledo
Account Manager

Our partner visits took us to see **BIS Computer Solutions, Inc.** and **Accu-Med Technology Solutions**, both successful BASIS partners who continue to support and grow their BASIS software business. Our focus was to make sure that they use our latest tools and technology that will give them the sharpest competitive edge in today's cutthroat market.

Our end user visits were to **Vacation Resorts International** and **Motivational Fulfillment and Logistic Services**. We recognize the special challenges that these self-programming end-user customers have – maintaining everyday business and user demands as well as planning for future needs of the company all while keeping up with the latest technology. We listened to their wish lists and Nico gave in-depth technology advice that will improve their business processes and strengthen their mission critical software.

Site visits always prove to be enlightening for both parties. The mutual challenge is to use what we all have learned and take action to improve our software and services. We had a productive trip with our customers discussing current business opportunities, talking about the latest technology enhancements, and getting their perspective on how we can better serve them. During our trip we also managed to fit in a quick drop-in visit with **Rennie, Lindsey and Associates**, **Western Computer Services**, and **PCS Group.**

Nico and I thank ALL of you for taking the time to visit with us. We hope to visit more customers as time permits, and remind you to take advantage of all of the resources available to you online. Java Breaks have proven to be a vastly successful resource to our developers, the Google Group forums are a great support option, the demos included with our downloads provide 'shareware' sample code for you to embed in your own solutions, and *The BASIS International Advantage* magazine articles are packed with valuable "how to" information. We hope to see you on the road soon! ■

Las Vegas

Partnering with OSAS - Las Vegas Style

Destination Viva Las Vegas was in our plans again! BASIS CEO Nico Spence, BASIS Engineer Jeff Ash, and I attended Open Systems Inc.'s *Partners for Profit* and *Customer Excellence* conferences for their resellers and end users respectively, of their ERP solution.

The conference began with a welcome reception where it was great to catch up with old friends and colleagues and make new acquaintances. With good conference attendance this year, Open Systems was ready to put on a show.

Dave Link, Vice President of OSAS Development, presented the recently released OSAS 7.6 with BBj® 11.0 that introduced 250+ new features! In addition to the many new application features, Version 7.6 took advantage of several BBj features such as PDF output, panel-style menus using MDI controls, BBj Forms, and additional user set-up options now provided in Enterprise Manager. Version 7.6 also took advantage of BBj's ability to incorporate third party Java objects into BBj code.

By Gale Robledo
Account Manager

Dave talked about the future of OSAS and their goals for the next release, OSAS 8.0. This will be a pure BBj release and no longer support Visual PRO/5® and PRO/5® interpreters. Moving to pure BBj gives them the opportunity to rejuvenate the product and take advantage of even more BBj features than ever before. Some of the new features in their sights are converting to VKeyed files, using iReport/JasperReports to replace many of their standard reports, and including stored procedures and Web Services, to name just a few. We are excited to hear about their upcoming technology release and stand ready to support the OSAS development team.

Nico and Jeff were busy showing off the newest BBj features during three days of breakout sessions. They demonstrated that you really can 'go mobile' with the Browser User Interface, and how PDF printing, XCALL, and Google apps integration allows you to be cloud-capable by leveraging the cloud benefits through your applications.

Database and file replication was another hot topic. Nico and Jeff demonstrated this new feature and how easy it is to use for backup and recovery, and data warehouse requirements without changing a line of code in one's application.

The *Customer Excellence* end user conference, following the *Partner in Profit* reseller conference, was a nice opportunity to meet users of the product. Our focus for this breakout session was how the technology can help the user in the areas of disaster recovery, data mining and warehousing, data security and mobile computing. Users are always looking for secure and better access to data and how to put a business continuity plan in place. We focused on the directory and file replication feature and how it provides all of these needs.

The sessions were well attended and received. Once again, "Thank you, Open Systems, for hosting these conferences and giving us the opportunity to partner with you to exhibit and present the latest BASIS technology features." Until next conference… ■

Nico Spence presents at the OSAS conference in Las Vegas with assistance from Jeff Ash

BASIS TechCon Canada by Descore

As the saying goes, "three times a charm," so for the third time BASIS presented TechCon2011 in the equally charming city of Montreal. Those who attended were also charmed by the best of the (BASIS) best and saw the most advanced demonstrations at this event hosted by Descore, Inc.

In past years, Descore hosted a "showcase" with some technical material but it was mostly a sales and marketing presentation. This year, Descore decided to host a technical conference at which BASIS would rerun the TechCon presentations. Audiences at the two previous US and Europe TechCons received the sessions with so much excitement, it was a very worthwhile decision for the Canadian audience.

With BASIS Chairman and CEO Nico Spence presiding, BASIS engineers demonstrated how to use the BASIS toolset, including the Barista Application Framework and the IDE to create visually appealing applications. Brian Hipple (QA Engineer Supervisor for BASIS International Ltd.) and Stephan Wald (Director of Sales and Technical Service for BASIS Europe Distribution) discussed how customers can move their applications into the cloud and manage them in the stratosphere. Attendees also learned web-based concepts such as how to use Web Services and geolocation to further enhance their applications.

Three topics that generated the most excitement at the conference were how BASIS revolutionizes Web Apps with the Browser User Interface (BUI), the introduction of BASIS database and file replication, and the capabilities of the new Query Definition System in Barista. BUI technology allows BASIS customers to bring their GUI BBj® applications to a web browser without any changes to the source code while enhancing the look of their new BUI application by using Cascading Style Sheets. The use of replication makes it quick and easy to replicate an entire database or set of files to another location on the same machine or on another across the globe, with just a few settings. The Query Definition System in Barista builds on the BASIS SQL engine to enhance the existing Barista query capabilities to allow developers more flexibility how they present their data set.

The attendee feedback was extremely encouraging.

> "*I came to TechCon Canada looking for answers to my customers' questions. I got the answers I needed and I can't wait to sit down with them and tell them what I found out. It was worth the trip*."
>
> Michael Rainbird
> Rainbird Programming and Consulting

> "*I am going to go back home to Vancouver and implement iReports and triggers. I am also very interested in what I learned about the cloud and using Web Services with BBj*."
>
> Rob Percival
> Pacific Rim Software

In summary, as Dave Foster of Descore put it, "*The conference presentations and demos opened minds to new concepts and trends, and the Q&A sessions lead to fruitful conversations between attendees and BASIS that will likely result in great technology improvements for end-users and resellers alike.*"

TechCon Canada was the first true technical conference hosted by Descore. The results were very positive. I know that I am in the majority when I say I look forward to future Canadian TechCons where BASIS will have the opportunity to again share our cutting-edge technology with an eager audience. ■

Nico Spence and Stephan Wald present at TechCon Canada in Montreal

By Laurence Guiney
Senior Account Manager

Tech Resource Zone

Enterprise Manager Admin Only "Safe" Mode

The BBj® Enterprise Manager (EM) has long been an essential tool for managing BBj deployments. From this one application it is possible to simultaneously connect to local and any number of remote servers, manage users and databases, change server ports, manage the built-in Jetty Web Server, deploy Web Start applications, and deploy BUI applications. The list is long and growing longer as BASIS adds new features to each release of BBj. You can do just about everything from EM.

Q. ***When my server is live with active users, and if I want to add a table to a database, change a trigger, start a replication job, how do I gain exclusive access so that I can make these changes safely?***

A. It really would be best to have the system to yourself for these types of changes. In the past, developers have taken drastic steps to ensure server configuration changes did not affect (or were not affected by) active users. They manually edited the BBj.properties file, they quiesced (en.wikipedia.org/wiki/Quiesce) the server by manually disabling certain ports, they watched BBj Processes module in EM for errant users from, for example, the Sales Department. ☺

With BBj 11.0, you can have the EM all to yourself with the new Admin Only Mode. Admin Only Mode allows you to restart BBj Services in such a way that only the Admin Server, port 2002, is running. This prevents users from running any programs or accessing any data files outside of EM or the Admin API. Once enabled, restarting BBj Services will start up in Admin Only Mode; a red background will be displayed in the Server Information window, reminding you that the server has been quiesced, as shown in **Figure 1**.

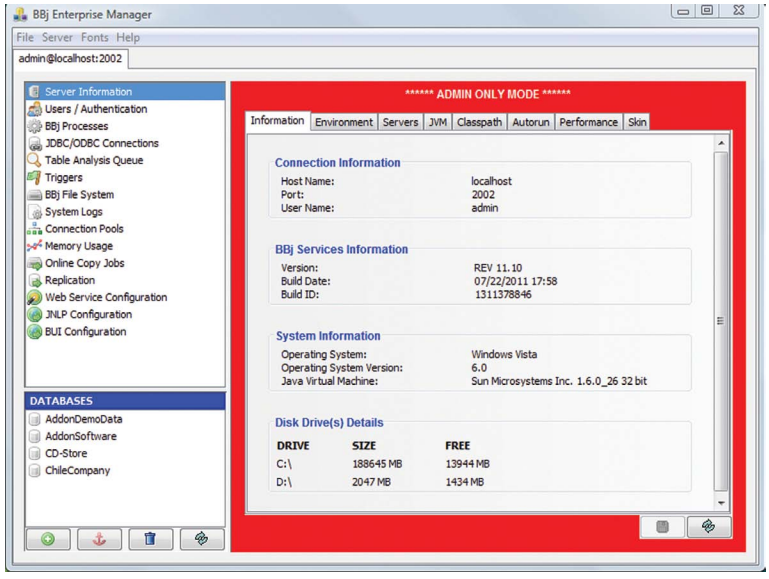


Figure 1. Enterprise Manager connected to a BBj Server running in Admin Mode

Now you can make your changes without worrying about complications that can arise in a live environment. Once you've committed your changes, simply restart BBj Services to bring it up normally with all of the servers running.

Q. ***How do I enable Admin Only Mode?***

A. This useful feature, tucked away in the BBj 11.0 EM, is available in three locations:

1. From the EM Server menu, select Shutdown to Admin Only Mode. Clicking this menu item will shutdown BBj Services with Admin Only Mode Enabled, as shown in **Figure 2**.

2. From the Server Information navigator in EM, click the Servers tab. The [Enable Admin Only Mode] button appears at the top of the window. Clicking this button merely enables Admin Only Mode; it does not shut down BBj Services, as shown in **Figure 3**. >>

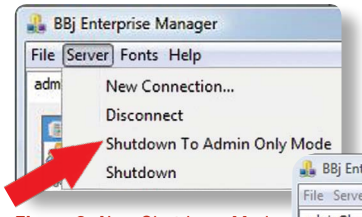


Figure 2. New Shutdown Mode

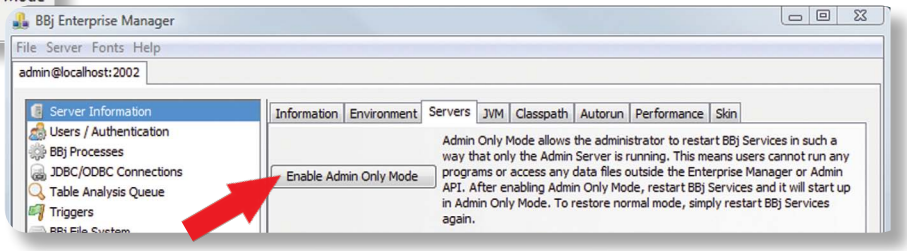


Figure 3. New Enable Admin Only Mode button

By Bruce Gardner
Technical Support Supervisor

3. The third place you'll find the Admin Only Mode is in EM's Replication Navigator. Once you start a replication job, you will have the opportunity to click the [Shutdown Server to Admin Only Mode] button, which is highly recommended for the initialization of replication jobs. As with the Server menu item, clicking this button will immediately shutdown BBj Services with Admin Only Mode enabled, as shown in **Figure 4**.

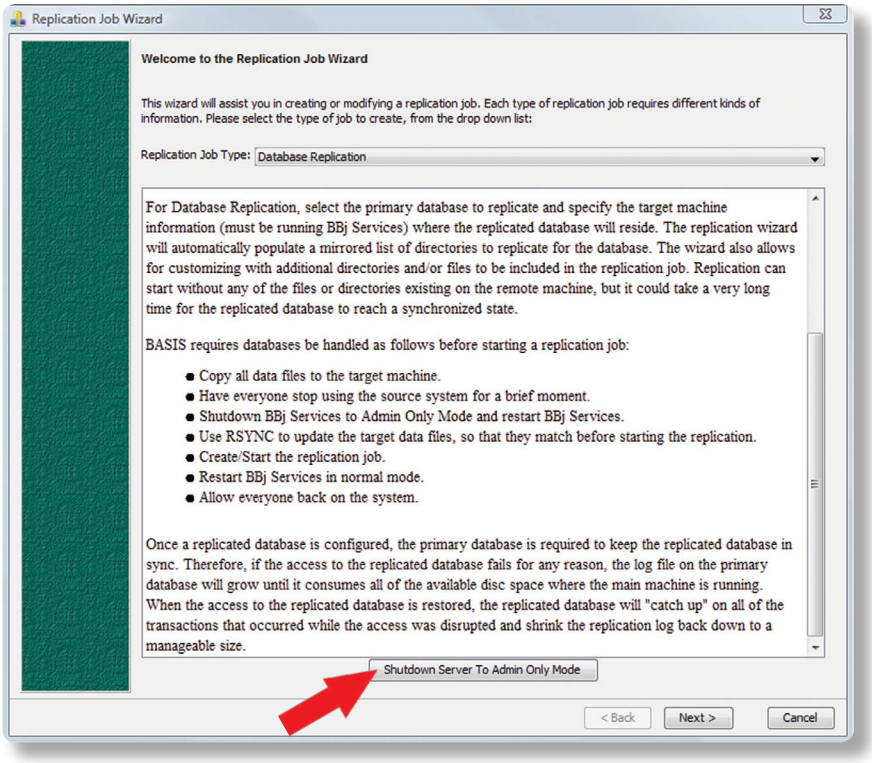


Figure 4. New [Shutdown Server To Admin Only Mode] button

Note: There's a fourth way to enable Admin Only Mode. If you want to be old-school about it, simply edit the BBj.properties file and add this line:

```
com.basis.server.BBjServices.adminOnlyMode=true
```

Once you've restarted BBj Services in Admin Only Mode, you will only be able to connect to the Admin port by running EM directly on the system console or by running EM on a remote machine and connecting to your quiesced server.

Now you're free to make the modifications you need, without worrying about some 'eager beaver' from the Sales Department sneaking in to do some work!

Summary

Admin Only Mode finally delivers the ability to use the intuitive interface of EM in a quiesced state without the worry of affecting live users or them affecting you. While this feature isn't one you would need often, when you need it, it's an extremely helpful maintenance tool. ■

Is "Commercial Open Source" for you?

Our innovative co-operative development concept rewards our partners for their efforts when they contribute to the core product. We call it "commercial open source." Our partners call it a fast track to high margins.

Element	Traditional OEM	Open Source	AddonSoftware's Commercial Open Source
Strong Company	✓	X	✓
Source Code	$$	✓	✓
Top to Bottom Support	?	X	✓
Development Tools	✓	✓	✓✓
Product Management	✓	X	✓
Revenue Opportunities			
Service Revenue	✓	✓	✓
Sales Revenue	✓	X	✓
Upgrades/Maintenance	✓	X	✓
Training/Customization	✓	✓	✓

Comparison table of ERP distribution models

Learn more about partner opportunities at info@addonsoftware.com

AddonSoftware by BASIS International Ltd. – the big little software company

AddonSoftware®

Trigger Functionality Extended with KEY Triggers

Triggers are a powerful feature of the BBj® filesystem and SQL engine, giving developers the ability to execute BBj code when certain file access operations occur such as a READ, WRITE, REMOVE, etc. BBj 11.0 introduced the ability to trigger events on calls to the KEY functions in the BBj language such as KEY(), KEYP(), KEYN(), KEYL() and KEYF().

Why is This Feature Important?

Let's take a look at a real-world example of triggers in action. The accounting group at BASIS wanted to upgrade their GL and AP modules from legacy AddonSoftware to the new 'AddonSoftware by Barista'. Though the file structures of the old and new systems were not the same, they did not want to change any code in the custom developed AR integration points to make the application read the alternate file structure. To solve this, BASIS implemented 'Instead Of' triggers to redirect operations made on the old files to the new files, specifically to redirect and read from the new files "instead" of the old files. These triggers took care of all of the application's READs and WRITEs to the data files, but calls to KEY(), KEYN(), KEYF(), KEYL(), and KEYP() functions still referenced the old data files and would not return the correct key values. Experiencing this new need, BASIS added triggers for key operations to the language and then defined the appropriate key triggers for the application's data files.

By Jeff Ash
Software Engineer

So "Why is this feature important?" The answer is simple, key triggers can handle necessary file operations and give users seamless interaction with the new files without changing the application.

Using KEY Triggers

Key triggers fire when any of the following BBj functions are called: KEY(), KEYP(), KEYN(), KEYF(), and KEYL(). Most key triggers will typically need access to two things: type of key call, and the value returned by the key call. In addition, the trigger can access things such as the key description, key number, key name, etc. For complete details, consult the BBj documentation on BBjTriggerData in the online documentation at links.basis.com/basisdocs.

The following example shows some of the calls that a typical key trigger might use:

```
declare BBjTriggerData td!
td! = BBjAPI().getFileSystem().getTriggerData()

REM Get the name of the key if it is a VKEYED file with a named key
keyName$ = td!.getKeyName()

REM Get the key number of the current key
keyNum = td!.getKeyNumber()

REM Get the key description
keyDesc$ = td!.getKeyDescription()

REM Get the key value read from disk in an after key trigger
REM NOTE: getKey() is NOT valid in a KEY trigger
keyValue$ = td!.getReadBuffer()

REM Get the type of key call that was made. Returns KEY, KEYF, KEYL,
REM KEYN, or KEYP
keyType$ = td!.getKeyCallType()

REM In an after key trigger, this will set the value returned
td!.setReadBuffer("Modified Key")
```

Figure 1. Sample Key Trigger Program

Summary

The new KEY triggers in BBj 11.0 round out the trigger functionality in the BBj filesystem. While they are not used by the SQL engine, they are common in BBj applications using READ RECORD and WRITE RECORD calls instead of SQL. With this new functionality, developers have a number of new options available when migrating users to a new version of their application. ■

Search the BASIS web site for additional trigger-related BASIS Advantage articles

Enterprise Manager Admin API Enhancements

Imagine using your own program code for setting user authentication, permissions, and maintenance; obtaining server information; administering processes and files; setting up BBj® applications to be autorun; and creating online copy jobs.

Sound interesting?

The Admin API now provides an interface for BBj applications to accomplish tasks that could previously only be performed in the Enterprise Manager (EM). Now developers can accomplish these tasks directly in their BBj or Java program. We'll take a closer look at these tasks by examining the capabilities of the Admin API demo.

User Functions

One of the most powerful features in the Admin API is the ability to authenticate, validate privileges of, and maintain users.

To validate a user, simply try to obtain an Admin object from the appropriate BBj Services using the user's credentials. If the Admin object was successfully obtained, then the user is valid, otherwise the method throws an error. After obtaining the Admin object, test the user for various permissions. For example, a program could test that a user has the ability to create a database via the getSingleUserPermissions method before actually offering to create a database for the user. If a user has the set permission attribute, then any permission can be granted/revoked for other users. In addition, an Admin object a user obtained with the appropriate permissions can add and remove users, just like the Admin API demo does in its User Management screen (see **Figure 1**).

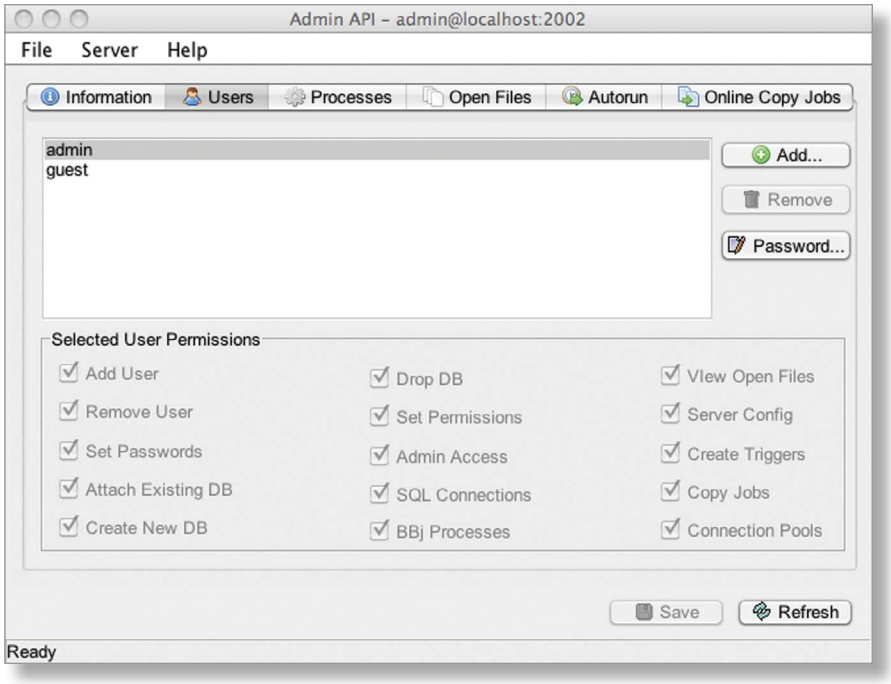


Figure 1. Admin API demo's User Management screen

Server and System Information

Server information such as connection, BBj Services, and system are available via the Admin API. Connection information includes the currently connected client user, host, and connected port. The BBj Services information contains the revision, build date, and build ID. The system information has the OS type and version as well as providing the Java vendor, bit level, and version, as the Admin API demo System Information screen shows in **Figure 2**. > >

By Brian Hipple
Quality Assurance Supervisor

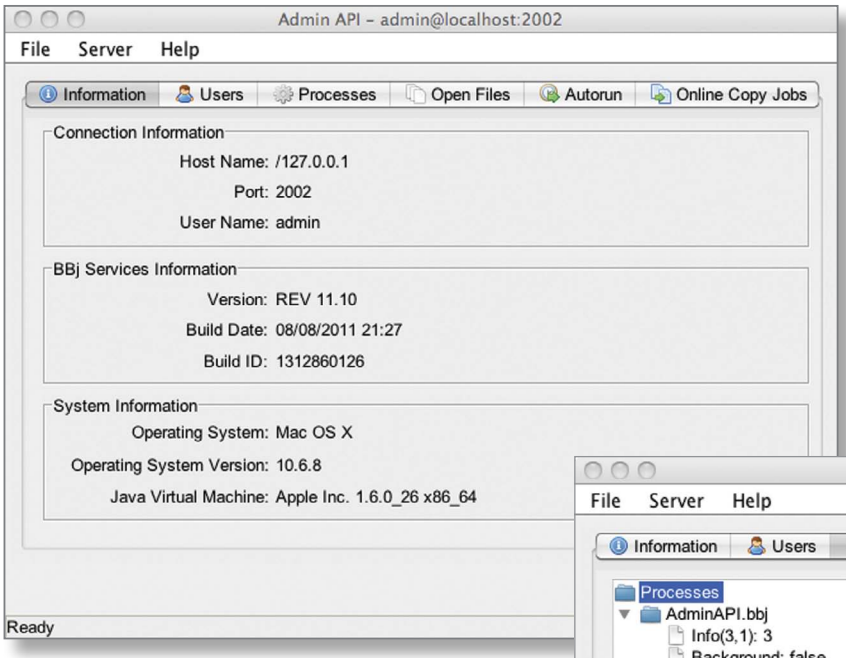


Figure 2. Admin API demo's Information screen

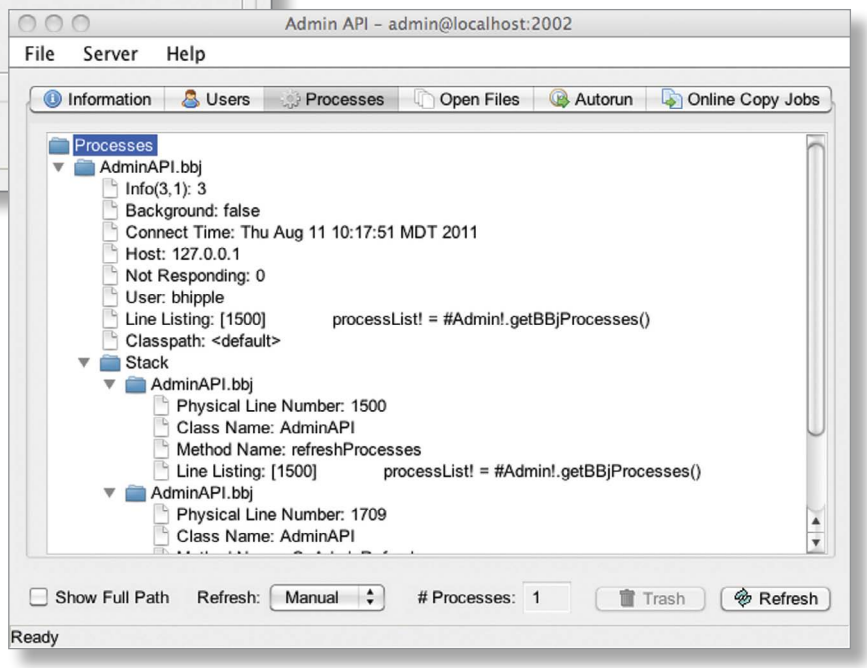


Figure 3. Admin API demo's Process screen

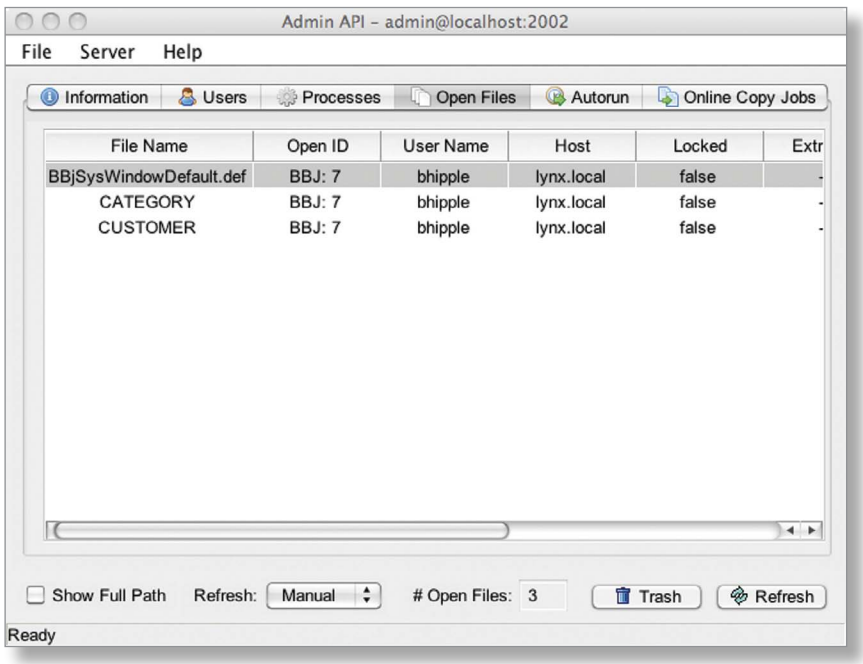


Figure 4. Admin API demo's Open files screen

Processes

One of the mostly widely used features in EM is the ability to view and sometimes terminate running BBj processes. This feature is now available to BBj applications, providing process information such as program name, user, current line listing, start time, and much more. Developers can obtain the call stack, which is invaluable when debugging an application. Processes shown in **Figure 3** are terminated by calling the BBjAdmin BBjProcess::terminate method.

Files

The Admin API (see **Figure 4**.) provides a list of currently open files as well as an abundance of additional information such as the name, user, host, process id, extracts, advisory locks, etc. To close the files, use the BBjAdminOpenFile::forceClose method.

Autorun

Autorun starts BBj programs when BBj Services starts and provides a way to invoke long-running BBj applications such as a service or daemon. Default settings for autorun programs include the config file, working directory, and user. In addition, autorun programs can have custom parameters or use the default >>

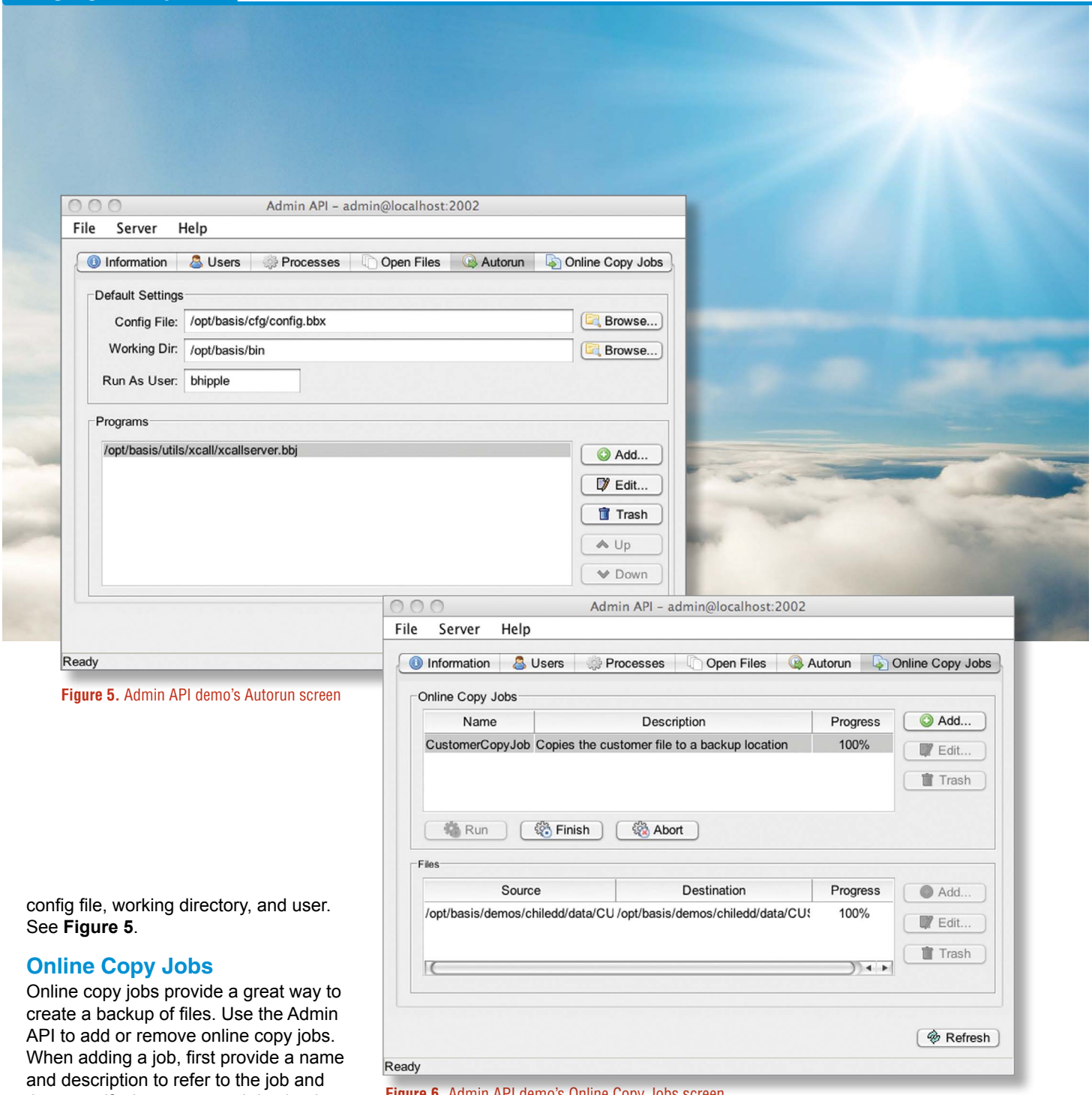


Figure 5. Admin API demo's Autorun screen

Figure 6. Admin API demo's Online Copy Jobs screen

config file, working directory, and user. See **Figure 5**.

Online Copy Jobs

Online copy jobs provide a great way to create a backup of files. Use the Admin API to add or remove online copy jobs. When adding a job, first provide a name and description to refer to the job and then specify the source and destination files. Next, simply run the job and obtain a status. A running job keeps the source and destination files in sync and can be aborted or finished at any time. When the status reaches 100% as shown in **Figure 6**, the source and destination files are in sync. Then, complete the online job by invoking the Admin API.

Summary

The Admin API gives you the power to do virtually all that the Enterprise Manager does inside your program. With BASIS' introduction of BUI, developers can now perform these powerful functions in a web browser without even having Java or BBj installed. This means that developers can perform BBj system administration from virtually any machine or mobile device in the world! To get started harvest the code from the Admin API demo and infuse it into your own business application today. ■

- For further information, refer to the Admin User Credential
- Install the demos and review the full source code of the BBj Admin API demo program at <bbj install>/demos/AdminAPI/AdminAPI.bbj

Centralizing App Security Using Admin User Credentials

In many cases, developers offer a suite of BBj® business applications and include utility programs for the application administrators that setup and create a database connection or call BBjAPI().getAdmin() for system administrative functions such as looking at open files and registering BUI applications. Developers would also commonly recommend that their customers' administrators follow good security practices by changing their BBj Services password after installation, making it necessary for the utility programs to obtain the customer's usernames and passwords whenever needed.

You are likely one of these developers, so how do you go about making the administration program obtain this password and keep it secure, so that the program does not have to ask for username and password every time it is needed?

Today's Solutions

There are several ways to address this situation. One solution would be to hard code the client's password into the program and then compile the source into a protected program before distributing it to the client. However, this solution would cause some maintenance overhead for each client (either on the developer's part or the client's), not to mention the unfortunate side effect that anyone who can access the client would also be able to access the client's BBj Services.

A somewhat more secure alternative would be to create a properties file on each client's machine with the username and password stored in encrypted form. While possibly inconvenient, this solution requires that someone get access to the client's machine before accessing BBj Services.

Tomorrow's Solution, Today!

However, what if there was a way to keep persistent client credentials for accessing BBj Services, like a server-side cookie? Suppose then, that before a program attempts to make a call that requires user credentials, if it could not find a valid cookie for the client, it would present a login dialog, get the username and password, and any other configuration needed to access BBj Services. Upon a successful connection, the program would then optionally store the user credentials in a server-side cookie.

Sound great? Well, in the spirit of easing the developers burden and providing application building block tools, BASIS now provides this functionality in the Admin Utility!

Admin Utility Overview

The Admin Utility is located at <install_path>/utilities/admin.bbj and contains two BBj objects of relevance – Admin and UserCredentials. Admin is the object that asks for user credentials and creates the cookies; UserCredentials is obtained from the Admin object and contains the information necessary to connect to BBj Services. Look at how these objects handle user credentials:

1. Instantiate the Admin object.
2. Configure the Admin object (optional).
3. Before calling a section of code that requires credentials, obtain a UserCredentials object by calling Admin::getUserCredentials().
   a. The Admin utility will look for the user's cookie.
   b. If the Admin utility cannot find a cookie for the user, a login dialog box appears.
4. Use the fields in the UserCredentials object to connect to JDBC or BBj Services before performing the system administration tasks.

Run the simple program in **Figure 1**. If this is the first time you have ever run a program using the Admin utility and your BBj Services password is other than the default, the login dialog shown in **Figure 2** will display. Upon completing the fields and clicking [Login], the Utility verifies that you are able to connect to BBj Services. Your credentials are then saved to a cookie. Run this sample a second time and notice that since your credentials are saved, the dialog box does not display. >>

By Shaun Haney
Quality Assurance Engineer

```
rem Admin User Credentials Example: Obtain User Credentials to get a list of open files.

use ::admin.bbj::Admin
use ::admin.bbj::UserCredentials

declare Admin adminUtilityObject!
declare UserCredentials userCredentials!
declare BBjAdmin admin!
declare BBjVector openFileInfos!
declare BBjOpenFileInfo fileInfo!

REM We need to make a call to BBjAPI().getAdmin(), to obtain credentials
adminUtilityObject!=new Admin()
adminUtilityObject!.setRequiredPermissions(adminUtilityObject!.getDELETE_DB())
userCredentials!=adminUtilityObject!.getUserCredentials()

if userCredentials!=null() then
   print "Sorry, could not log in.  Press Enter.  ",
   read a$
   end
endif

userName$=userCredentials!.getUser()
password$=userCredentials!.getPassword()
admin!=BBjAPI().getAdmin(userName$,password$)
openFileInfos! = admin!.getOpenFileInfos()

numFiles=openFileInfos!.size()
if (numFiles<>0) then
   print "Currently open files:"
   for flCtr=0 to numFiles-1
      fileInfo!=cast(BBjOpenFileInfo,openFileInfos!.get(flCtr))
      print fileInfo!.getFilename()
   next flCtr
else
   print "No open files to report"
fi
```

Figure 1. Using the Admin Utility to get credentials

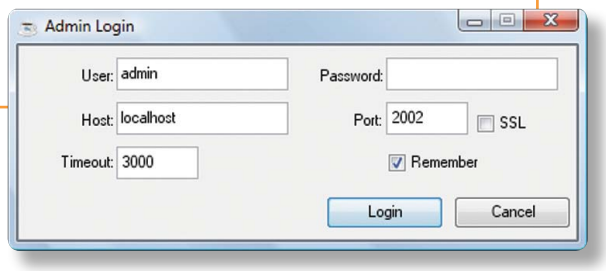


Figure 2. The resulting dialog from Figure 1 when no credentials are available

Admin Utility Options

Even though the Admin Utility has a small interface, its functionality is robust. Typically, username and password are the only user credentials developers consider. In addition to the required username and password, BBj Services also offers several configuration options for establishing the connection, and each database or administrative task requires privileges that the user may or may not have. Therefore, the Admin Utility demonstrates other user credentials to control both the connection to BBj Services and additional user privileges.

Pre-configure

The first feature is the ability to preconfigure BBj Services connection parameters and specify whether the user can modify them at login. Parameters that are configurable at the login prompt for BBj Services are the hostname, port, username, password, whether to use SSL, the timeout for obtaining a connection in seconds, and whether to remember these parameters on the next login. One can configure each of these parameters before showing the dialog via a simple "set" method. Additionally, one can configure them to be a "required" parameter or to set the field to the default value and prevent editing.

```
REM We need to make a call to BBjAPI().getAdmin(), to obtain credentials
adminUtilityObject!=new Admin()
adminUtilityObject!.setHostRequiredValue(1)
adminUtilityObject!.setPortRequiredValue(1)
adminUtilityObject!.setSSLRequiredValue(1)

userCredentials! = adminUtilityObject!.getUserCredentials()
```

Figure 3. Code from Figure 1 modified to make Host, Port, and SSL Uneditable

If a company wanted its employees to connect to a specific BBj Services, then it could pre-configure the host and port to point to its server according to the sample in **Figure 3**. When the login dialog pops up, the host and port would display in uneditable fields. The user's only choice would be to connect to the specified BBj Services. This feature is >>

useful in the common situation where users are only expected to enter a username and password, and should not change the other parameters.

Privileges

The next feature is the ability to specify which privileges the connecting user must have for a successful login. If the application has configured the Admin object to require certain user privileges, the user may specify the correct username and password, but still not be able to log in due to the user lacking one or more of the required privileges.

Imagine an administration utility that includes a tab for database configuration. Only users with the appropriate user credentials must be allowed to select the tab to configure the database. Mary, who uses this application, has a normal user account with the username "mary" for performing everyday tasks. She also has a second account with the username "mary_admin", for performing maintenance tasks that require special permissions.

Mary selects this tab and specifies her "mary" account in the user credentials. The "mary" account has permission to make a JDBC Connection for executing queries, but does not have permission to delete databases, which is one of the many permissions required to have access to the features in the tab. Even after Mary correctly enters her username and password, her login fails and she receives a message that she does not have the permissions required to switch to the tab.

After seeing the error message, Mary selects the tab again and now enters credentials for "mary_admin". Her administrative account has the privilege to maintain databases so she is now able to switch to the tab successfully and perform database administration.

```
REM We need to make a call to BBjAPI().getAdmin(), to obtain credentials
adminUtilityObject!=new Admin()
adminUtilityObject!.setRequiredPermissions(adminUtilityObject!.getDELETE_DB())
userCredentials!=adminUtilityObject!.getUserCredentials()
```

Figure 4. Privileges for login specified with the Admin::setRequiredPermissions()

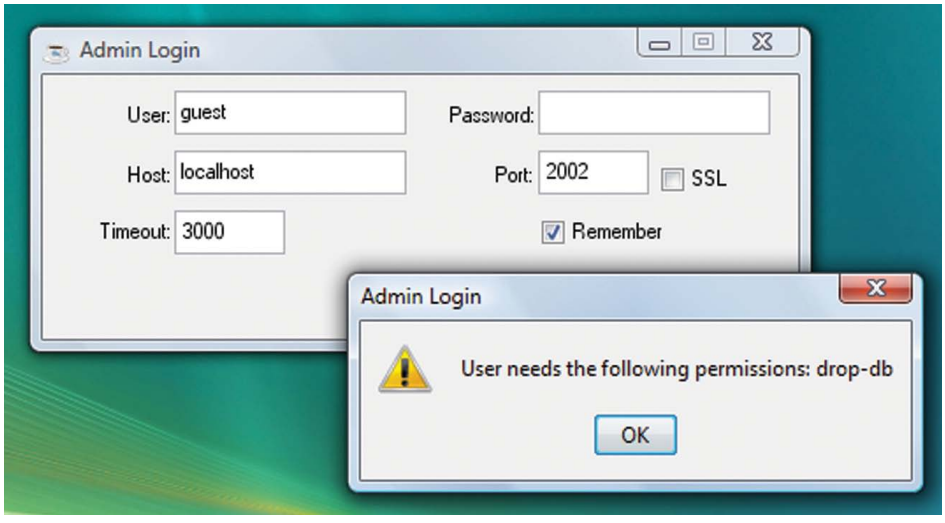


Figure 5. Log in attempt without privileges

Necessary privileges for login can be specified with the Admin::setRequiredPermissions() method (see **Figure 4**). **Figure 5** shows the resulting error message when "guest" attempts to log in without privileges. Each privilege is represented as a constant of BBjAdmin Object, not to be confused with the "Admin" Object from the Admin Utility. An example of one such BBjAdmin Object constant is BBjAdmin.DELETE_DB. Permissions are numeric constants and can be added together to set multiple permissions.

Summary

While requiring user credentials to access databases and administrative functions keeps clients secure, managing those credentials over multiple system administration programs can be inconvenient and potentially cause configuration and code maintenance nightmares if the solution is not carefully centralized. With the Admin Utility application building block, BASIS now centralizes that solution, along with all the configuration details needed to connect to BBj Services. Developers are now free to focus on developing applications without worry and unnecessary effort to secure their application. ■

Review the Admin Utility UserCredentials methods in the online documentation.

A Home (Page) in Every Port

Beginning in BBj® version 9.0, BASIS bundled BBj with a built-in Jetty Web Server. Therefore, every installation of your BBj server application allows for the creation of a default home page that provides countless opportunities for improving interaction with the users of the system.

If you're a developer, this custom home page can help your customers know what features and functions your product is capable of, and what new supplementary modules might be available, thereby improving customer satisfaction and increasing sales. The default page can also help BBj integrate more tightly with a corporate intranet or provide a starting point for creating one. Another great option is the ability to include tutorials and instructions for applications online, right in BBj itself. Custom error pages can provide a similar role, getting users the help they need when they don't find the applications they're looking for. Read on for more details of valuable ways to put the default home page to work for you.

By Jason Foutz
Software Programmer

Product Page

The BBj installation includes a custom home page that presents the features and benefits of BBj. Developers can create custom pages for their own product, detailing and describing what features make their product great. This can build interest in the product and show what new features are available. In addition, this default page can include links to samples or demonstrations, or even a call to action with a link to the developer's website or an email request for more information. Web pages are all about providing information to users. It is important to keep in mind that information should include some sales support so developers can help their customers find the products and features they need to succeed.

Intranet Page

Another great use of the default web page is integration with a corporate intranet site. Organizations often have an employee start page with contact information, calendars of upcoming events, company policies, and tools for working at the company. The design of the default page can match the company's internal style to help make BBj and developer's applications feel like the company standard. Getting users looking at BBj's built-in default will help educate and inform users about how to use the tools BBj and custom applications provide.

Some organizations may not have an internal home page. BBj's default page is a great option for getting started with one. Collecting key information like upcoming meetings and after hours contact information and publishing it in a central location for all corporate users to see will make any organization more effective. Smaller organizations may have avoided doing this because of the hassle of installing and configuring a web server, but since they have already done the work of installing BBj, they can leverage that effort and just use the built-in web server.

How-To Page

User education can be challenging as different users need to know different functional areas of their applications to get their jobs done. The default home page can provide many varying categories of information about the installed application – from getting started tutorials to in depth articles about specific processes. Having a central location for documentation is a great way to provide the information users need to keep working and to learn how to work more effectively.

Documentation appears in many different structures and forms. Some users want to follow a step-by-step example, some prefer to pour over reference documentation, while others want to view a video. It is important to make all of these forms available to your users, and the default home page is a great way to do that.

Error Page

When something goes wrong, users need to know what to do to correct the problem. The default error page is a good place to present that kind >>

Figure1. BBj default page

of documentation. Perhaps the most important thing to provide from an error page is a link to restart the application, either BUI or JNLP, so users can get back on track. It might be worthwhile to include simple tips such as clearing caches or whatever is appropriate for the application. Contact information that includes phone numbers, email addresses, and links to developer websites are great to publish on an error page, as well as the default home page. If something does go wrong, there is no wasted time searching around for who to call; users have what they need to contact the support team directly.

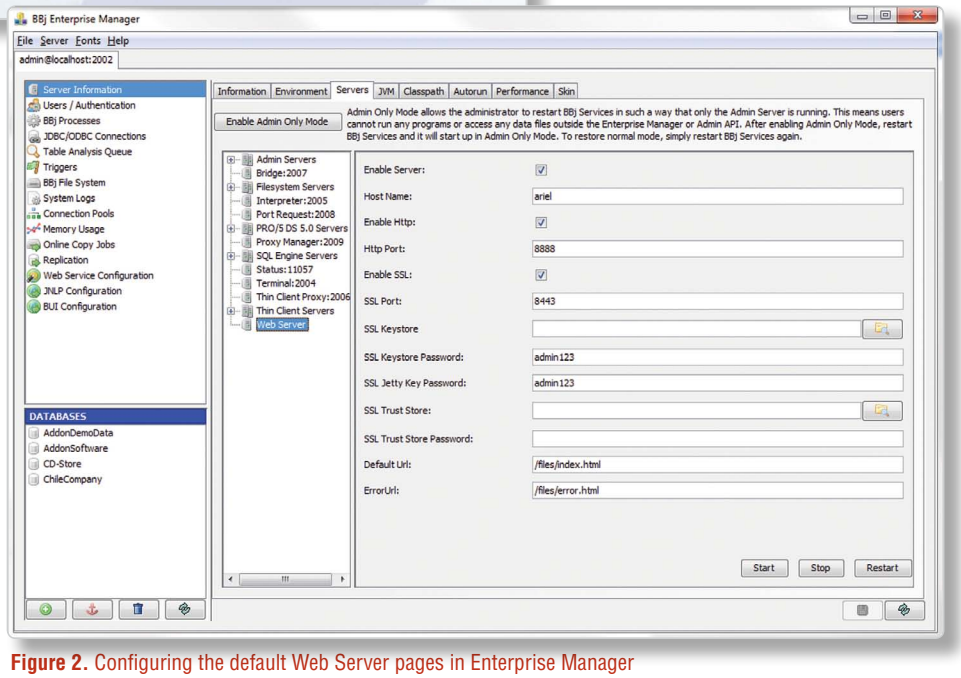


Figure 2. Configuring the default Web Server pages in Enterprise Manager

BBj's Default Page

BBj provides a default page with several helpful links, including the BUI Showcase, accessed by clicking the smartphone images as shown in **Figure 1.**

The default page also provides a link to launch Enterprise Manager so that users can quickly configure their new installation. BASIS customers can easily contact us via the Facebook, Twitter, and Linkedin links, and view demos of new functionality on the BASIS YouTube Channel. The default page helps developers connect with BASIS and receive the latest BBx updates and BASIS news.

Enterprise Manager provides the configuration for the default URL and the error URL, each specified as relative paths. For example, to replace the default index with myindex.html, simply supply the relative path /files/myindex.html to redirect the user to the new location. The actual file myindex.html should be saved in the htdocs directory in BBj's home directory. See **Figure 2**.

Summary

The default home page provides many opportunities for both developers and corporate IT departments alike to stay in contact with their users. Providing information about how to use the product as well as contact information when users run into trouble can help users succeed. The default home page also can help BBj integrate with a corporate intranet, or even serve as one if necessary. Most importantly, it gives developers a way to stay in contact with customers about the products they have purchased. Customizing the default home page is a valuable way to provide you and your customers a "port in a storm" and a "portal" to the organization. ■

Need Cookies? Get User Properties Cookies

What may come to mind when you read about cookies are those small, round, flat, sweet, crispy treats loaded with carbs. But as you probably know, a cookie in the computer arena is information stored on the client in key and value pairs. Cookies carry over from one visit to the next, generally via a web browser and server. Since the browser is the main component in the new BBx® generations implementation of BUI, it behooves BASIS to provide this functionality in the BBj® API for applications, both for the browser and for the desktop. And BASIS has provided just that!

Typical Uses

Typical uses for cookies include shopping cart contents, user preferences, identification, and authentication. User preferences can include the customization of a web page or screen's display of fonts, colors, layout, etc. Identification can include items like name, address, email address, etc. Authentication values, such as user names and passwords, can also be stored in cookies. Many e-commerce web sites offer shopping carts in which users can place items to purchase and cookies allow users to store the contents of the shopping cart. End users can even exit the site and return later to find their items still in the cart.

By Brian Hipple
Quality Assurance Supervisor

Before Cookies

Before BASIS implemented cookies, the program had to store this information on the server on behalf of the client. This often cluttered the server with data, which the client may or may not be able to remove. If there are many clients or if there is a lot of information to save, this data can take up quite of bit of storage space. It is also very hard to uniquely identify clients on the web without requiring some type of login procedure each time the client visits the site, making it difficult to know when to purge data that is no longer needed.

Security

Using encryption ensures that secure information such as user names and passwords are not stored in plain text or transferred to the server in clear text. Another commonly used security tactic is to store a unique look-up value in the cookie and look up this value on the server in a server-side table. This approach does not pass anything over the wire that is at risk for compromise. Shortening the lifespan of the cookie can further secure the information, and the client can always clear cookies, if desired. If the cookie is a look-up value, then the developer can change the value and/or subsequent action on the server.

Implementation

Although cookies are historically a browser concept, BASIS understands that enabling a developer to set and save client information is valuable for both the browser (BUI) and the desktop (SYSGUI). Cookies are implemented in the BBj API with two methods: BBjThinClient::setUserProperty and BBjThinClient::getUserProperty. **Figure 1** shows an excerpt of the "Cookies (Get/Set User Properties)" demo program employing the BBjThinClient::setUserProperty method.

```
method protected void setCookie(BBjString key$,BBjString value$)
   if (#TC! <> null())
      #TC!.setUserProperty(key$,value$)
   endif
methodend
```

Figure 1. The Cookies demo program saving a user property

In BUI, cookies are stored on the client using the standard JavaScript cookie API. The properties are set with the domain that is serving the BUI application and are accessible by other BUI applications that are served below the URL for the application. If the BUI application URL is secure (HTTPS) when the property is set, the program is only allowed access to the cookie over other secure connections. In SysGUI, the information is stored using Java Preferences API under the **`com.basis.bbj.client.$default`** node. In the future, cookies set in SysGUI will have similar scope as in BUI and will offer the ability to specify an expiration date like their BUI counterparts. >>

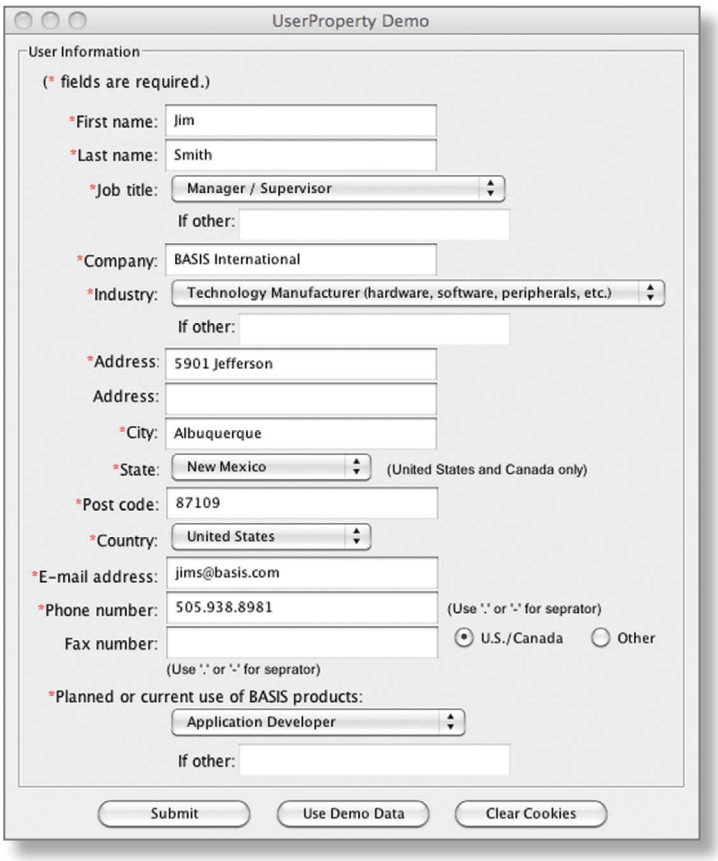


Figure 2. Cookies in SysGUI

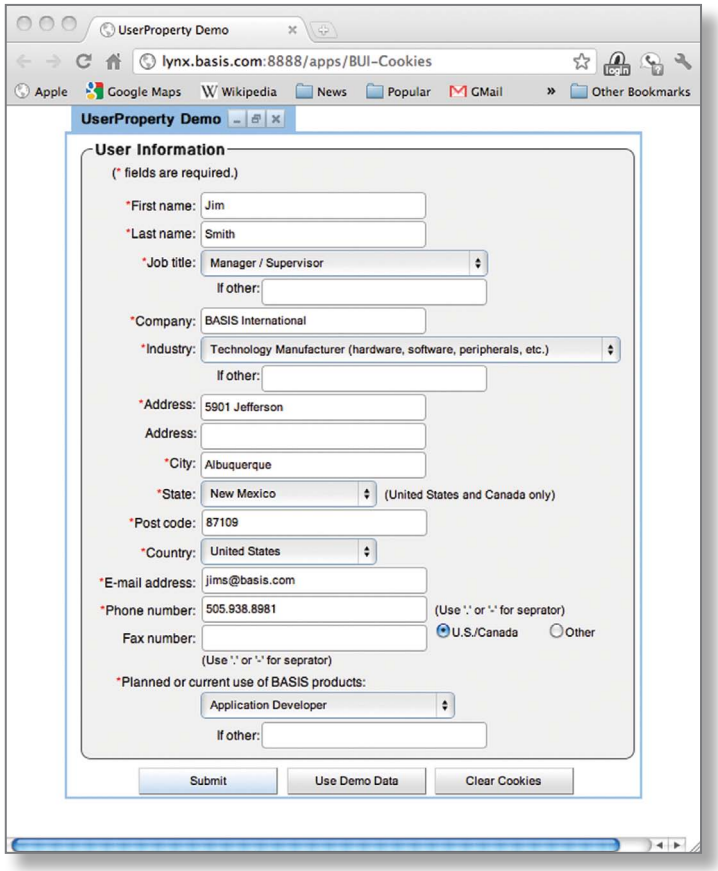


Figure 3. Cookies in BUI

In Action

A very familiar and well-used BASIS form is the user information required before downloading a product from the BASIS website. Our TechCon demo of the new cookie functionality uses a BBj application to mimic the download form, obtaining the same user information and subsequently storing the field information in cookies when the user presses the [Submit] button. When the program runs again in either SysGUI (**Figure 2**) or BUI (**Figure 3**), it populates the form with the information from the cookies. In the demo, as well as the real product download page, cookies save the user both time and aggravation. Without the benefit of cookies, users would have to enter their information each time they wanted to download a new release of a BASIS product.

Access the Cookies demo by selecting demos when installing BBj and then run the Cookies demo from the Demos LaunchDock. Be on the lookout for our product download page changing from a hard-to-maintain Java script program to an easy-to-create/maintain BUI application that uses cookies.

Summary

Cookies are a wonderful way of saving client information where it belongs – on the client. Unlike the outcome of the popular children's book "*If You Give a Mouse a Cookie*" in which the boy is busy all day trying to satisfy a hungry mouse, giving computer coders "cookies" translates into time savings and great satisfaction for users of the resultant application. Give cookies a try and take a bite today! ■

- Read more about cookies at www.whatarecookies.com
- Numeroff, Laura Joffe., and Felicia Bond. *If You Give a Mouse a Cookie*. New York: Laura Geringer, 2007. Print.

Copy With Ease, Paste With Confidence:

The New SysConsole Paradigm

Copying and pasting in the BBj ThinClient SysConsole has never been elegant. Designed to work just like copying and pasting in the Visual PRO/5® SysWindow, it implements an older character-based terminal style of placing text into a copy buffer and re-inserting it at the cursor location. Reliable and simple, but more than a little strange nowadays, this old method is not very convenient.

New in BBj® 11.0 is the option to use the much more familiar CTRL+C/CTRL+V copy/paste technique in the ThinClient SysConsole. It is a nice touch that will make users happy and remove the need to purchase Terminal Emulation software to get that functionality – all deployed in an instant – while saving you money and time.

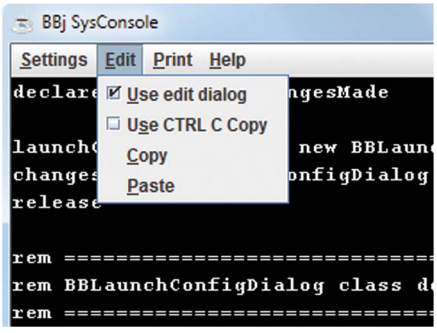


Figure 1. The new Edit menu

The Old Way – What's Wrong With It

Looking at **Figure 1**, you will see a new check box menu item in the Edit menu. If you are a Windows or Linux user, the menu displays **Use CTRL C Copy** while the Mac displays **Use CMD C Copy**. The default setting leaves this check box unchecked, meaning the SysConsole uses the traditional Visual PRO/5 SysWindow copy/paste mechanism.

To copy text from the SysConsole to the clipboard the old way:

1. Choose **Copy** from the **Edit** menu. There is no keyboard shortcut for this; you have to move the mouse to the **Edit** menu, open it and click on **Copy**. The mouse cursor changes to a cross-hair.
2. Select the text
   a. Move the cursor to one corner of the area containing the text to be copied
   b. Press and hold the left mouse button while dragging the cursor to the opposite corner. A dashed rectangle outlines the current text selection area, as shown in **Figure 2**.

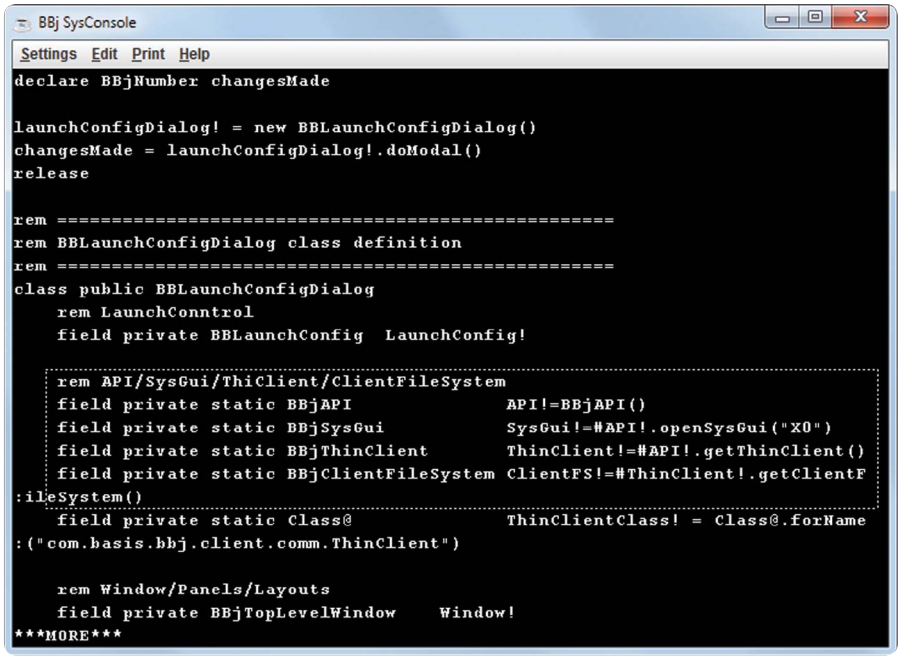


Figure 2. Selecting text to be copied

3. After completely enclosing the text to copy in the rectangle, release the left mouse button to put it into the copy clipboard, adding line feeds between lines.

That was easy, wasn't it? Sure, except for the fact that there are no keyboard shortcuts, and that you must carefully place the cross-hair cursor *between the lines of text* instead of on the lines of text, and that you can only select what fits inside a rectangle, which may be more text or less text than you wanted, depending on its layout.

Pasting is "just as easy." Move the mouse to the **Edit** menu once again and click on **Paste**. If your copy selection included a little extra text at the beginning or end because >>

By Mike Phelps
Software Engineer

of its rectangular shape, you'll have to manually edit the results, but who really cares about a few seconds of extra work, eh?

The New Way – What's Right With It

To change the copy/paste paradigm used in the BBj SysConsole, check the box next to **Use CTRL C Copy**. (Incidentally, the **Use CTRL C Copy** setting is preserved for you automatically in the SysConsole's .def configuration file so you won't have to continually change it with each new BBj session.) As **Figure 3** shows, you're in a whole new ball game now:

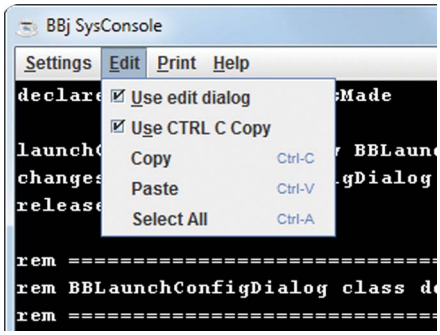


Figure 3. Using the new copy/paste paradigm

The **Edit** menu now has **Copy** and **Paste** menu items with keyboard shortcuts, and there's even a new **Select All** choice. You can pretend you are inside a word processor window instead of a character terminal window and copy text like this:

1. Position the mouse cursor on the first character of the text you would like to copy, then hold down the left mouse button and drag the cursor to the end of the text.

2. Release the left mouse button and the selected text is highlighted with reverse video. If your text is on more than one line, notice that you selected an irregular shape (not a rectangle) and got only the precise text that you wanted (see **Figure 4**).

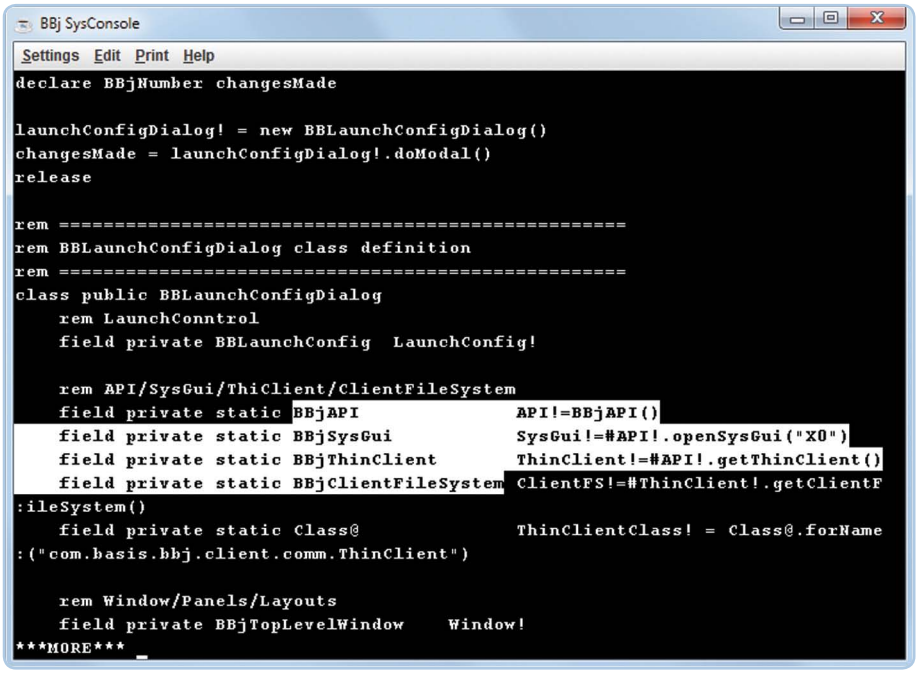


Figure 4. Selecting only desired text in the new copy/paste paradigm

3. Copy the highlighted text to the clipboard by pressing CTRL+C (Command-C on Mac OSX) or by choosing Copy from the Edit menu or by pressing the right mouse button to open a popup menu and choosing Copy.

   Pasting is more convenient as well: Press CTRL+V (Command-V on Mac OS X) or choose Paste from the Edit menu or press the right mouse button to open a popup menu and choose Paste. No manual editing required after the fact (see **Figure 5**)!

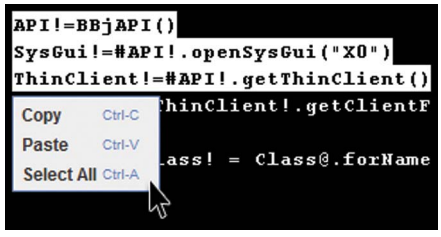


Figure 5. Copy, Paste, and Select All options from the right click popup menu

Programmatic Control

You can also control which method of copy/paste is active in the SysConsole by using the 'CTRLCCOPY' mnemonic in your program code: `print'CTRLCCOPY'("on")` enables the new mechanism, while `print'CTRLCCOPY'("off")` returns control to the old mechanism.

So What's the Catch?

The new CTRL+C/CTRL+V copy/paste mechanism is definitely more practical and convenient than the traditional Visual PRO/5 method, but know that it also imposes two serious caveats that may cause problems in certain situations:

- The CTRL+C/Command-C key combination is no longer available as a program interrupt; it can't function as a copy trigger and an interrupt at the same time. For the same reason, the CTRL+A/Command-A and CTRL+V/Command-V key combinations are no longer available to CUI applications when this new copy/paste mechanism is active.
- Mouse hotspots (created with the SysWindow 'MOUSE' and 'AMOUSE' mnemonics) are ignored. All mouse clicks and drags performed in the area of the BBj SysConsole window apply to text selection, making it impossible to activate a mouse hotspot.

Summary

The new copy/paste mechanism helps to freshen up your legacy CUI applications and improve the user experience. It is a must-have for character-based data entry applications that involve entering text into many different input fields or that call for transferring text between BBj and non-BBj applications. If you don't use mouse hotspots or rely on CTRL+C to interrupt execution, click the Use CTRL C Copy checkbox right away and get back to work.

If missing this feature is the last reason that you've held back from using the BBj ThinClient, perhaps its time for you to stop paying for and configuring your Terminal Emulator software. Just upgrade to Enterprise Edition and use the built-in Jetty Web server to install and launch your CUI app via Web Start. Then reduce your user counts and free up your time by moving to zero installation, and never again install software on client machines! ■

BUI CGI – Gateway to the Internet

The Common Gateway Interface (CGI) has been around for years, and anyone who has ever purchased a product off of the Internet has dealt with CGI first hand. In a nutshell, CGI refers to how a web server delegates the generation of dynamic HTML pages to a server-side program.

Our general purchasing experience on the Internet is a prime example of CGI, as the checkout process involves the customer adding items to a shopping cart, filling out a purchase form, and submitting the form. The form generally contains customer information like name and address, as well as purchasing information such as a credit card number and expiration date. When the customer fills in the form and clicks on the submit button, the web server executes the server-side CGI program referenced in the HTML page's ACTION attribute of the form. The CGI program then accesses the form's data and proceeds accordingly, eventually sending the customer a confirmation page. The gateway portion in this example is the interesting part - it is the connection between the data as it exists in the customer's web browser and the program on the web server that will eventually access and process the data.

Why use BUI Instead of Traditional CGI?

So what does all of this have to do with BBx® and BUI technology in particular? Because a BUI application is able to access CGI environment data, just like any other CGI application, it is possible to leverage a BBj® BUI program as a reasonable alternative to the typical CGI Script/HTML page combo. This is exciting for all sorts of reasons, including:

- BBj is a familiar, straightforward language that produces rapid results
- BBj program development and debugging is much easier compared to most other CGI solutions
- BBj offers advanced controls such as masked input controls and popup calendars that simplify development

But one of the most tempting reasons to use a BUI program is that one can update existing portions or add new capabilities to an existing web system in an à la carte fashion – there is absolutely no need to completely rewrite the whole system. Because a BUI program has access to the CGI environment, it's possible to replace a single aspect of an existing system by simply changing the existing HTML page's form ACTION attribute to reference a BUI program instead of the legacy CGI script.

A Sample Use Case

Let's say your company has an existing web-based solution that utilizes a CGI language like Perl to gather form-based user information for customer feedback. The existing system is in need of an overhaul to add more features and flexibility to the survey capabilities. The original HTML page that gathers customer feedback still suffices, but the back-end CGI script > >

By Nick Decker
Engineering Supervisor

needs modifications to further process the customer information and propagate it into an updated database structure. One solution would be to swap out the outdated Perl CGI program and slip a BBj BUI application in its place. Development and testing time would be decreased dramatically as the BUI program could access the existing business logic in your company's internal application. This means that a single code base is now responsible for processing customer information, instead of having separate systems written in different languages. The only trick is getting the customer information from the web-based form to the existing BBj processing routines.

The BUI CGI Demo Shows How It Is Done

The "BUI CGI" demo installed with BBj is a simplified version of our sample use case. When run, it first presents the user with an HTML page in their browser that uses a FORM element to gather customer feedback, as shown in **Figure 1**.

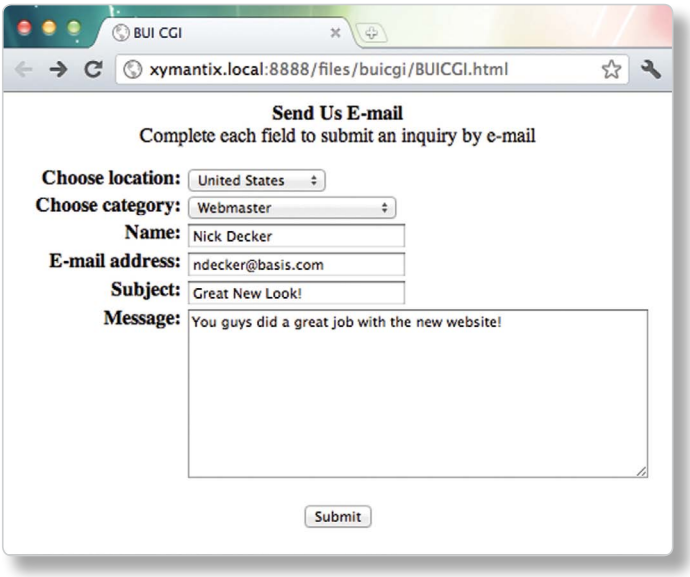


Figure 1. The BUI CGI's customer feedback HTML page

The HTML page is a straightforward web page served by the Jetty Web Server, included with the BBj product suite. Taking a look under the covers at the page's source code reveals the key FORM element, shown in **Figure 2**.

```
<form action="/apps/BUICGI" method="post">
```

Figure 2. The FORM element for the customer feedback HTML page

HTML FORM elements have an ACTION attribute that specify where to send the form's data when the user submits the form. Instead of referring to a Perl CGI script, this demo references a BBj BUI app instead. We've previously registered the BUI CGI program on our Web Server, and Enterprise Manager provides a quick way to copy the BUI app's URL into your clipboard, so it is a cinch to paste it into the HTML source code. The result of this change is that the Web Server launches our BUI program following the form submission.

The BUI program makes use of the CLIENTENV() function to retrieve the HTML FORM's data and store it into string variables for later use, as shown in **Figure 3**.

```
REM Get the information from the HTML form via clientenv
location$ = clientenv("location")
category$ = clientenv("category")
name$ = clientenv("name")
email$ = clientenv("email")
subject$ = clientenv("subject")
message$ = clientenv("message")
```

Figure 3. Using the CLIENTENV() function to retrieve FORM data

Now that the BUI program has access to all the FORM's data, it can CALL existing routines to process the customer feedback. The demo is a simplified version, though, so it creates a window with several controls to mimic the original HTML page and loads all the appropriate controls with the customer feedback. **Figure 4** shows the end result which is a "BUI-ized" version of the original HTML survey form.

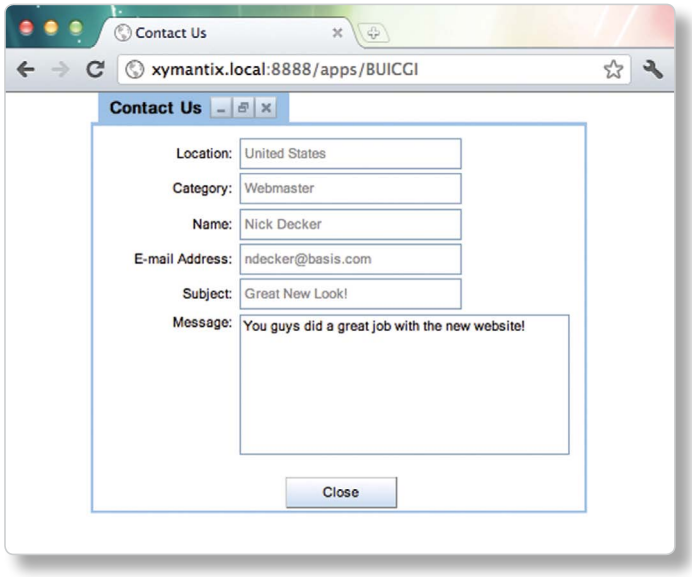
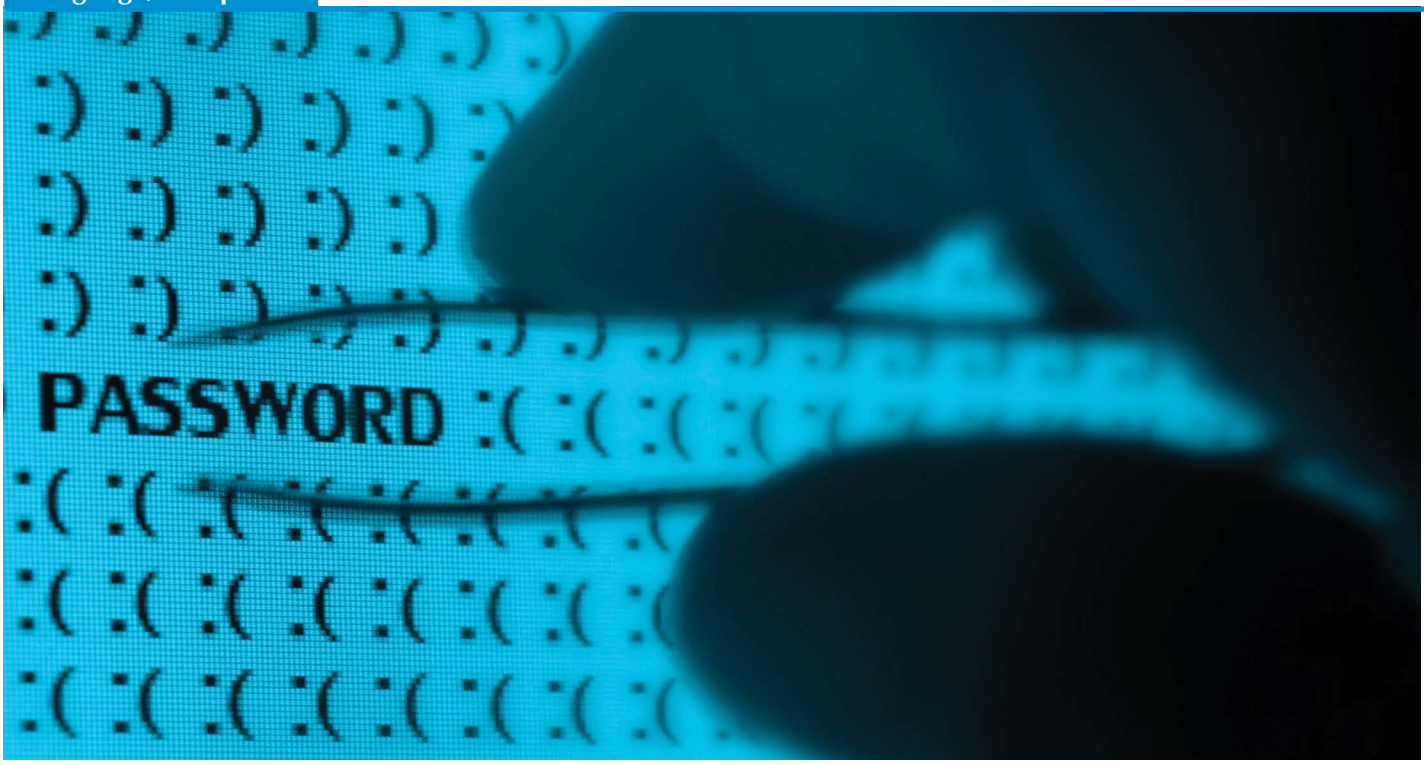


Figure 4. The BUI version of the feedback form loaded with the user's data

This example shows how easy it is to have an existing HTML page send all of its data to a BUI program. Our BUI demo resulted in a GUI program displayed on the end user's browser, but it could have just as easily redirected the browser to an existing submission confirmation page after processing the data.

Summary

Launching a BUI program is as easy as clicking on a link or pasting in a URL. Now you can use that URL in a brand new way – by specifying a BBj BUI program as the resultant ACTION for HTML form submissions. A BBj program is ready for the task, as it has complete access to all the HTML form's data, supports both GET and POST methods, and is a fast, robust programming language with excellent database integration. All of these features should make BUI your first choice for new Web development in the future. ■



PDF Security in Your Hands

BASIS customers asked and BASIS delivered – the ability for Sysprint devices to create PDFs. The next natural progression was to add security options. BASIS customers asked and BASIS once again delivered – password and security options for sysprint-created PDFs. This article covers the whole gamut, from hyper-security to default security to custom passwords and finally, to no security at all.

While the default security level meets most general business needs, users may require more control over documents. Sysprint provides new options that allow developers to deliver the precise control that security-minded users demand.

Red Security Alert

A new OPENPDFPASS mode, which is available in the forthcoming release of BBj 12.0, provides the highest level of security. The document will not even be viewable without the password specified in OPENPDFPASS. Further modes can be invoked with PDFPASS as described below.

Orange Security Alert

PDFs in BBj® are secure by default; the document is locked and cannot be edited with standard tools, making sysprint documents safe from accidental modification. The functionality is available in Print Preview without having to change code in any way. Users can now choose PDF format in the preview's [Save as] menu option. Alternatively, adjusting the mode line to indicate that you want to create a PDF and a specific file name creates a locked PDF for your users. These new options create PDFs that cannot be edited with common office tools, eliminating the risk of accidental modification.

Yellow Security Alert

Some users need to review documents created in BBj and make modifications. The PDFPASS mode line option makes this possible. By setting PDFPASS equal to a password of your choice, you can selectively control who can edit documents. While anyone could view the document, editing it requires the password. Since there are many cases when documents might need modification by select users, PDFPASS easily provides that control.

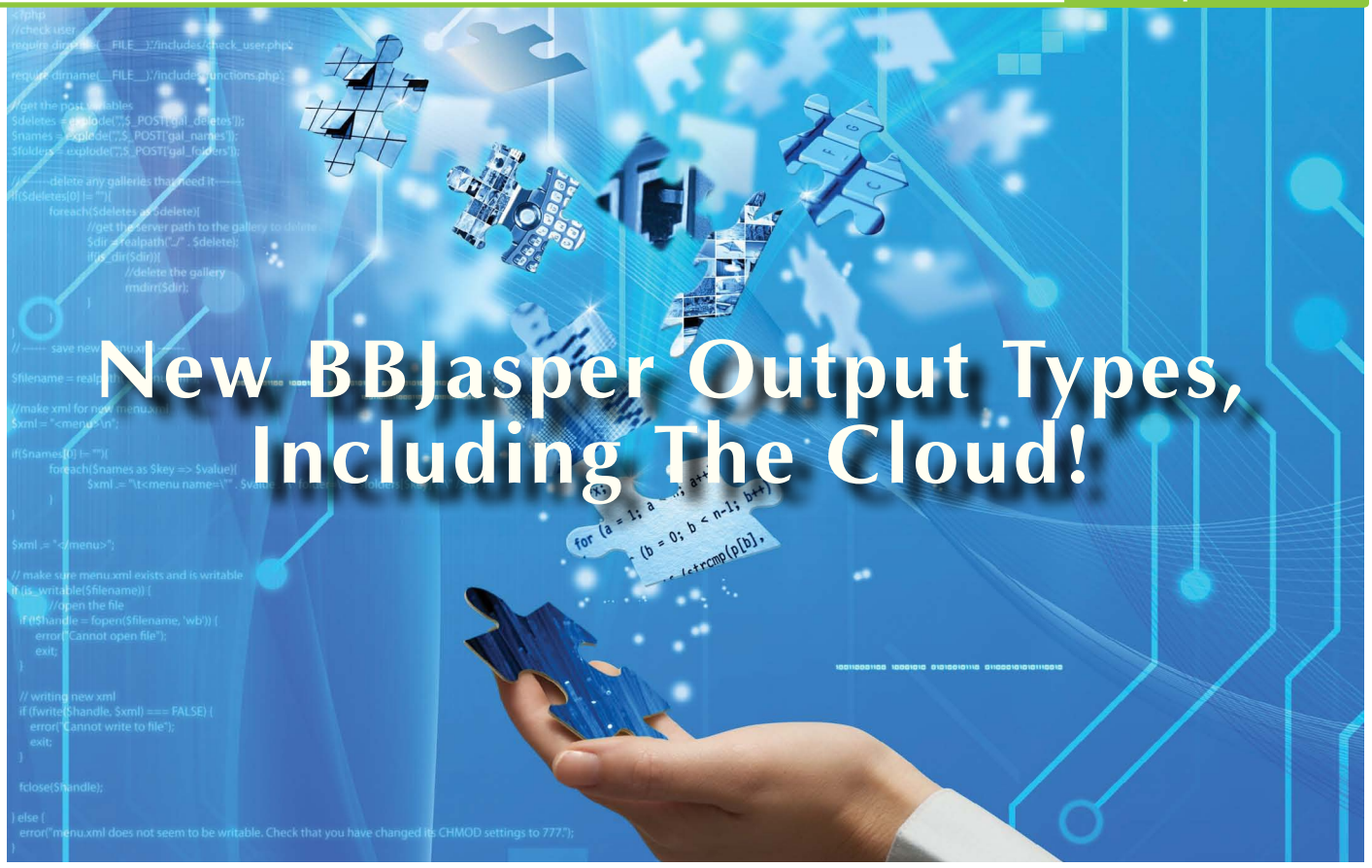
Green Security Alert

Finally, some PDFs require no security at all. BBj uses the PDFPASS option with no argument, or a blank password such as **PDFPASS=""**, to create documents that anyone can freely edit. While not appropriate for invoices, this is a great option for providing information to your users. Eliminating the security clearly avoids the overhead of the security mechanisms. Sophisticated users like to edit PDFs directly because it simplifies creation of brochures, or annual reports, or any other media where BBj-created PDFs might be modified for inclusion in a different environment.

Summary

BBj provides a range options for securing your PDFs. By default, no one can modify SYSPRINT documents. Providing a password to users gives more control over who can edit documents. By eliminating security completely, anyone is free to modify their PDFs as they see fit. BASIS' delivery of password control to your application-generated PDFs meets the requirements of today's security conscious world while providing the practical flexibility demanded by our users. ■

By Jason Foutz
Software Programmer



New BBJasper Output Types, Including The Cloud!

In BBj® version 11.0, the BBJasper Utility is compatible with JasperReports created in iReport version 4 and supports additional industry standard save formats. In conjunction with the GApps Utility, developers and end users alike now have the ability to save generated reports to the cloud. Viewing a Jasper report via the BBJasperViewer is no longer a hands-off experience as the Utility now saves viewer defaults and can issue receive notifications to end users' save actions. Let's take a closer look into these enhancements.

Compatibility with iReport

New BBj releases include, as applicable, updated JasperReports jars to ensure compatibility with the latest version of iReport – the WYSIWYG JasperReports design tool. **Figure 1** shows release compatibility between BBj and iReport, both of which rely upon the underlying JasperReports library.

New Save Formats

In previous BBj versions, one could save a JasperReport in the following formats:

- jprint - Jasper Print Format
- pdf - Portable Document Format
- html - Hypertext Markup Language

Now, BBj 11.0 adds these formats to that list:

- xls - Excel Spreadsheet (single or multiple sheets)
- csv - Comma Separated Values
- rtf - Rich Text Format
- odt - Open Document Text (OpenOffice)
- xml - Extensible Markup Language >>

BBj 9.0	iReport 3.5.0
BBj 9.01	iReport 3.5.0
BBj 9.10	iReport 3.5.1
BBj 9.12	iReport 3.6.0
BBj 10.0	iReport 3.6.2
BBj 10.01	iReport 3.6.2
BBj 10.02	iReport 3.7.4
BBj 10.03	iReport 3.7.4
BBj 10.04	iReport 3.7.4
BBj 11.0	iReport 4.02
BBj 11.11	iReport 4.1.1

Figure 1. BBj and iReport version compatibility

By Brian Hipple
Quality Assurance Supervisor

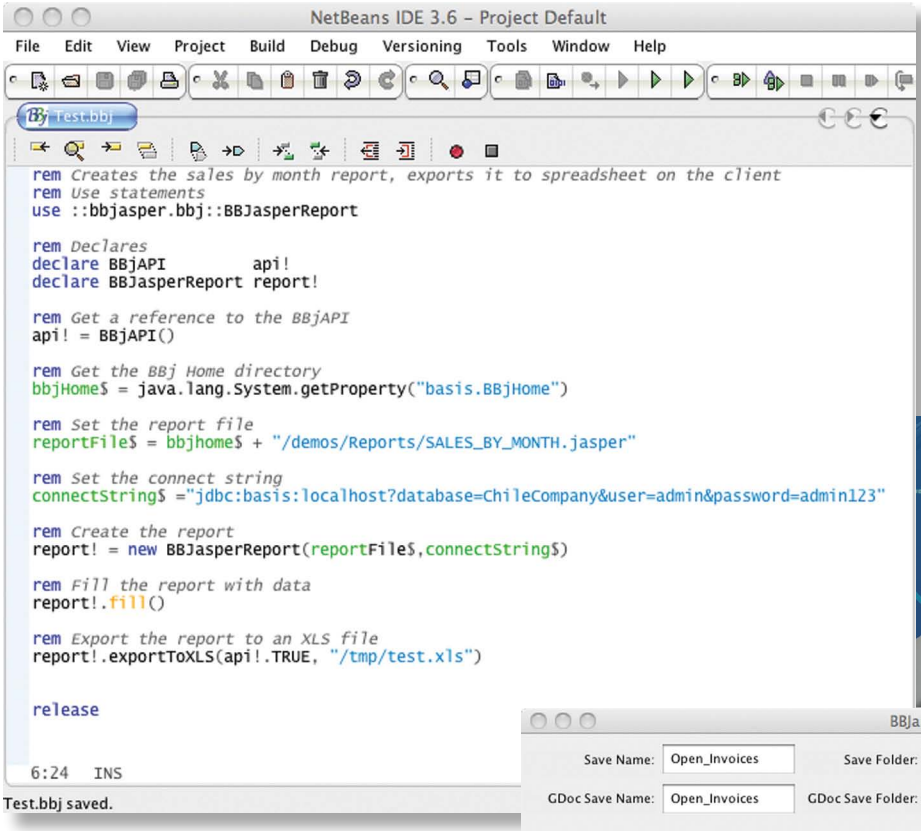


Figure 2. Sample code that exports a report to Excel format

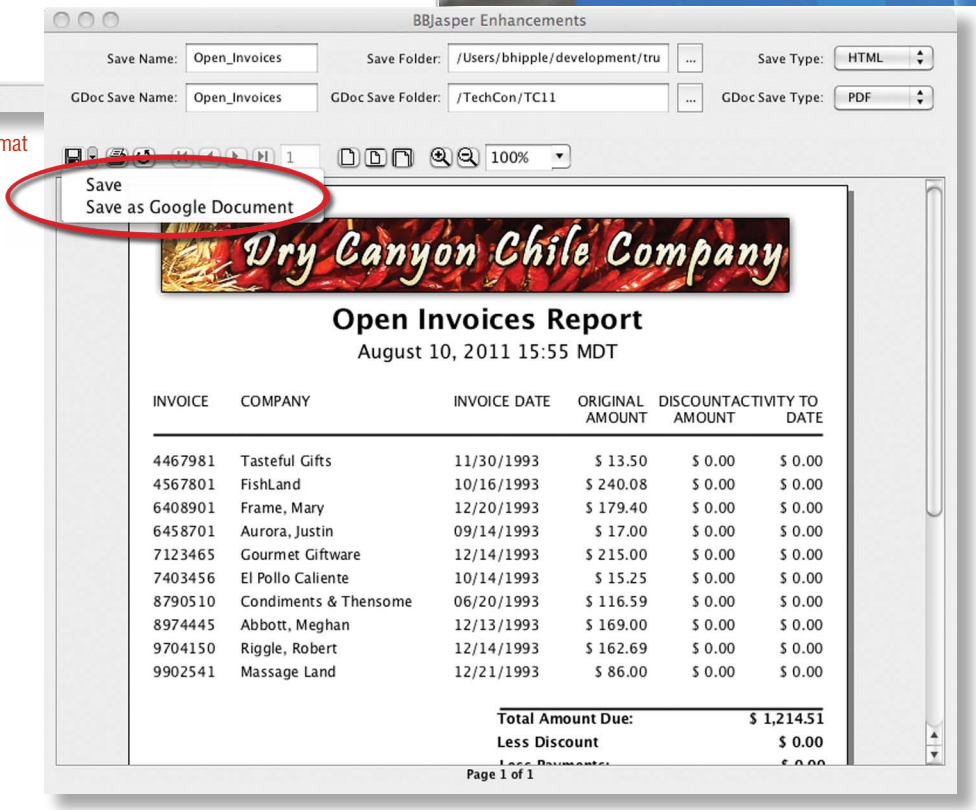


Figure 3. BBJasperViewer new Save options

An exportTo<format> method, shown in **Figure 2**, is available in the BBJasper Utility for each type listed above.

Saving to the Cloud

The GApps Utility provides an easy way to upload documents to the cloud. Calling the GDocsService method uploadDocument uploads a generated JasperReport for authorized users to share and view. A new menu button in BBJasperViewer (**Figure 3**) allows the end user to save the displayed report in the cloud to Google docs. After the user provides authentication to the Google docs via a login dialog, a save dialog appears in which the user selects the folder, file name, and type of document to save.

Setting Viewer Defaults and Getting Events

Previously at run time, the BBj program created the report, filled it in with data, and subsequently handed it off to the BBJasperViewer for the user to view. The developer had no control over the default folder, file name, and file type that the save dialog presented to the user. There was also no way for the BBj application to know when the user saved the report, with BBj 11.0 all of that was changed.

BBj 11.0 introduces the following methods to set the default folder, file name, and file type for the both standard file saves and for saving to the cloud:

- GSelectionDialog (links.basis.com/pljmr) - setDefaultFolder, setDefaultDocument, setDefaultType
- BBJasperViewerControl (links.basis.com/dhcpt) - setGoogleDefaultFolder, setGoogleDefaultDocument, and setGoogleDefaultType >>

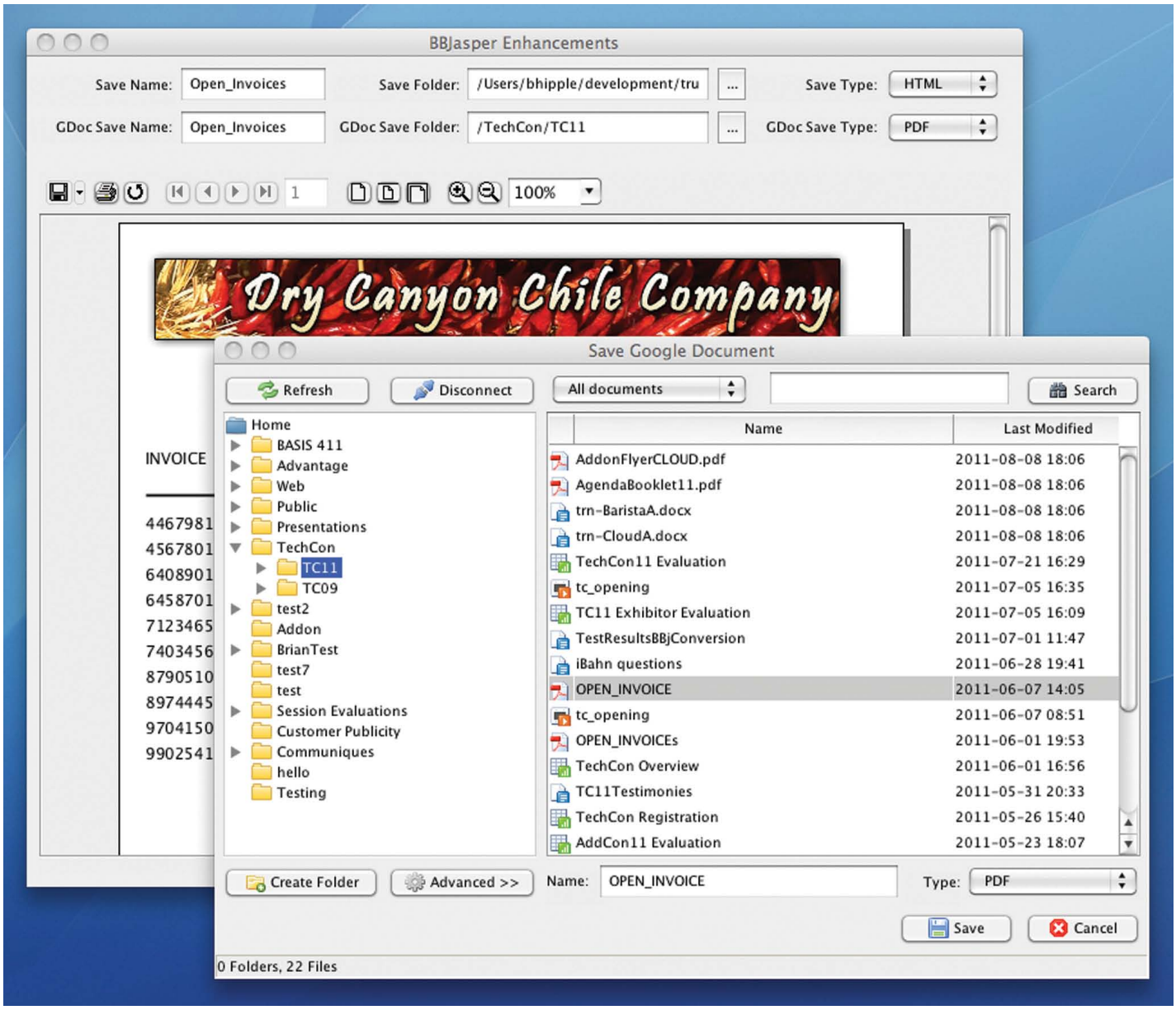


Figure 4. The Save dialog

The Save dialog shown in **Figure 4** allows the user to change these file specifications as desired.

Two additional methods, setFileSaveCallback and setGoogleDocSaveCallback, register for events when the user saves the report to disk or to the cloud. These events contain the folder, file name, and file type of the saved report, a very useful capability for archiving purposes that Barista's Docout sybsystem levereges.

Summary

Reporting is a very important part of almost every computer system. BASIS provides the BBJasper utility to facilitate easy integration of your BBj application code with the iReport open source Java-based report design tool for the JasperReports standard. BASIS now extends the functionality of BBJasper to include five additional save formats. And, in keeping with the widespread adoption of the cloud, the BBJasper utility now also sports integration to the Google Cloud offerings. Another powerful building block utility for your world-class applications. ■

Read Easy Integration To Google Apps

Our New Frontier: BASIS Web in the Cloud

In the last issue of the BASIS Advantage, "Our Salvation is in the Cloud," we detailed our efforts to move all of our business processes to the cloud and introduced the services we are using to do so. Amazon cloud machines, Amazon S3 data storage buckets, Amazon hosted RDBMS, and Amazon CouldFront are just a few of the services that have made this effort possible and successful. One part of this corporate effort included moving the two BASIS maintained websites, www.basis.com and www.addonsoftware.com to the cloud, this article shares the lessons learned.

Move to the Cloud

To move our websites to the cloud, we chose Amazon's EC2 and RDBMS service to provide BASIS with scalability and redundancy. With EC2, we utilize a load balancer on the front end that allows us to easily upgrade to a faster machine or add additional machines as traffic to our websites increase. The beauty of utilizing the load balancer is that with a just few clicks of the mouse, we can have 2 or 200 machines serving our content in just a matter of minutes. Amazon's RDBMS provides us valuable centralized data storage and scalability with the ability to backup data on-the-fly.

As a final piece of the migration, we are taking full advantage of the Amazon's S3 buckets. BASIS uses these buckets to serve our static Web content and store data backups. Since S3 buckets are available in various regions throughout the world, we have the ability to access all content from any other region should one particular region go down.

First Move

For a smaller and more manageable test case, we first moved addonsoftware.com to Amazon's RDBMS system to take advantage of their ready-built centralized database access that also offers easy back ups. A valuable time saver in our deployment process was to mount an S3 bucket to the file system so that if and when we choose to upgrade the system in the cloud, we would not have to sync files between machines. If we start a new test system, we can simply create a 'snapshot' of the machine and within seconds have the new system running without any downtime.

Second Move

Once we completed moving addonsoftware.com to the cloud, we began looking at moving our largest site to the cloud infrastructure. Because basis.com was so large, we also took the opportunity to cull out a lot of irrelevant and outdated data to provide our community with the accurate and clean content they require.

Manage the Content

Moving our sites to the cloud also gave us the opportunity to address another major need; to make our websites more dynamic and to empower all BASIS employees with the ability to manage and update the content in real-time. Our solution was Drupal, a free open-source content management system with over 600,000 people in 228 countries constantly improving it. As they state at drupal.org, Drupal is "...powering millions of websites and applications... built, used, and supported by an active and diverse community of people around the world."

Since addonsoftware.com was our first move to the cloud, we took the opportunity to learn and use Drupal to serve up the content. This gave us the experience needed to tackle the basis.com site, which contained a great deal more content to manage. Now we were equipped to share our skills with the entire BASIS staff and train them on how to edit and create new pages. >>

By Amer Child
Digital Communications/ Web Developer

Go Google

While applying cosmetic improvements with Drupal and its template design, BASIS integrated some of Googles free features to enhance the user experience.

Search

Google's Custom Search Engine enables Google to crawl a website directly for keywords. By adding this module (links.basis.com/cse) to both of our sites, users are presented with additional options or filters to refine the results when performing a keyword search. Choices includes BASIS Docs, Knowledge Base, Tutorials, and Advantage articles, as shown in **Figure 1.**

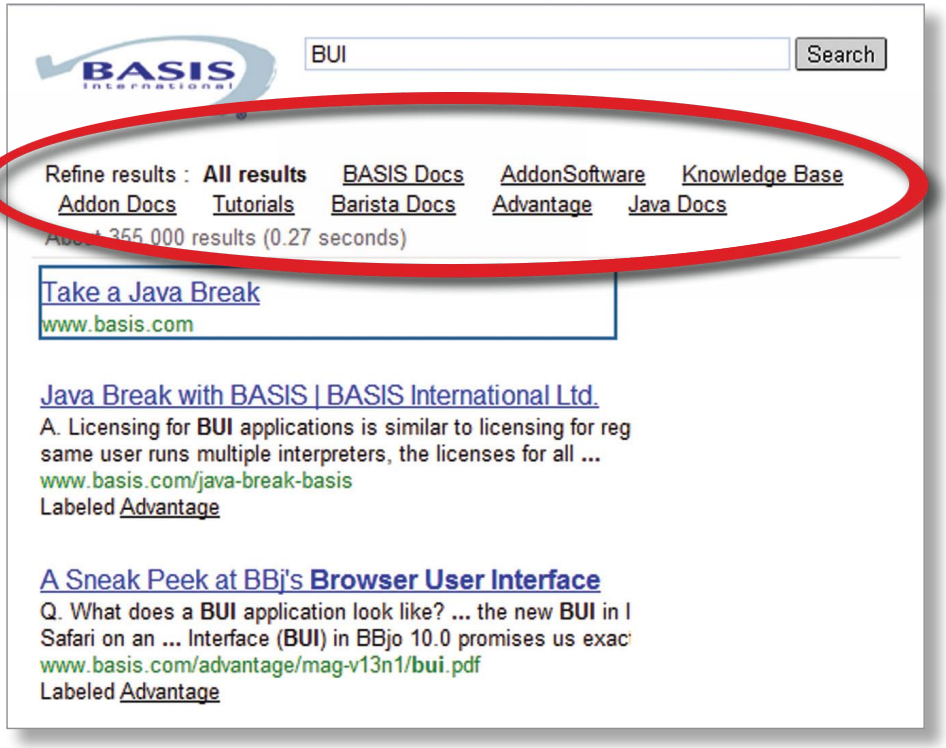


Figure 1. Options to refine keyword results when performing a search

Performance

Utilizing Google Mod_pagespeed, Mod_deflate, and Drupal Internal Caching markedly increases the speed of the site. This has improved our page loads by 30%. Google offers some great resources like PageSpeed Online to quickly determine how to increase site performance on the Internet. Mod_deflate is an Apache module that is configured on the server so that specific types of content can be cached on the server side.

As a final step to our move to the cloud, we utilized a CDN (Content Delivery Network) to give us the ability to serve content from the geographic location closest to the user accessing our page(s). We chose to use Amazon's CloudFront, which serves the content using edge servers that are housed in specific locations across the world. For example, when you download our product, you are placed on Amazon's Edge server closest to your geographic location so that content is available at a moments notice. Since we also serve all our own video content, the use of CloudFront provides faster streaming of our Java Breaks and product videos on-the-fly.

Summary

BASIS continues to blaze the trail to the cloud, reaping benefits and gaining valuable experience that we pass along to our customers and partners. As more and more of our products and services are hosted in the cloud, we gain speed, availability, and a previously unheard of level of robustness backed by solid guarantees.

BASIS' two main websites are now fully deployed in the cloud. They take advantage of several performance boosting techniques and are propagated automatically all over the world to optimize your browsing experience, regardless of your location. Our new CMS systems are the icing on the cake, dramatically increasing our freedom and improving our capabilities to update the site content at a moment's notice with a few clicks of the mouse. Our voyage to the cloud has been exhilarating, and we invite you to join us! ■

Read the *BASIS Advantage* article Our Salvation is in the Cloud and the sequel Perfection in the Cloud in this issue