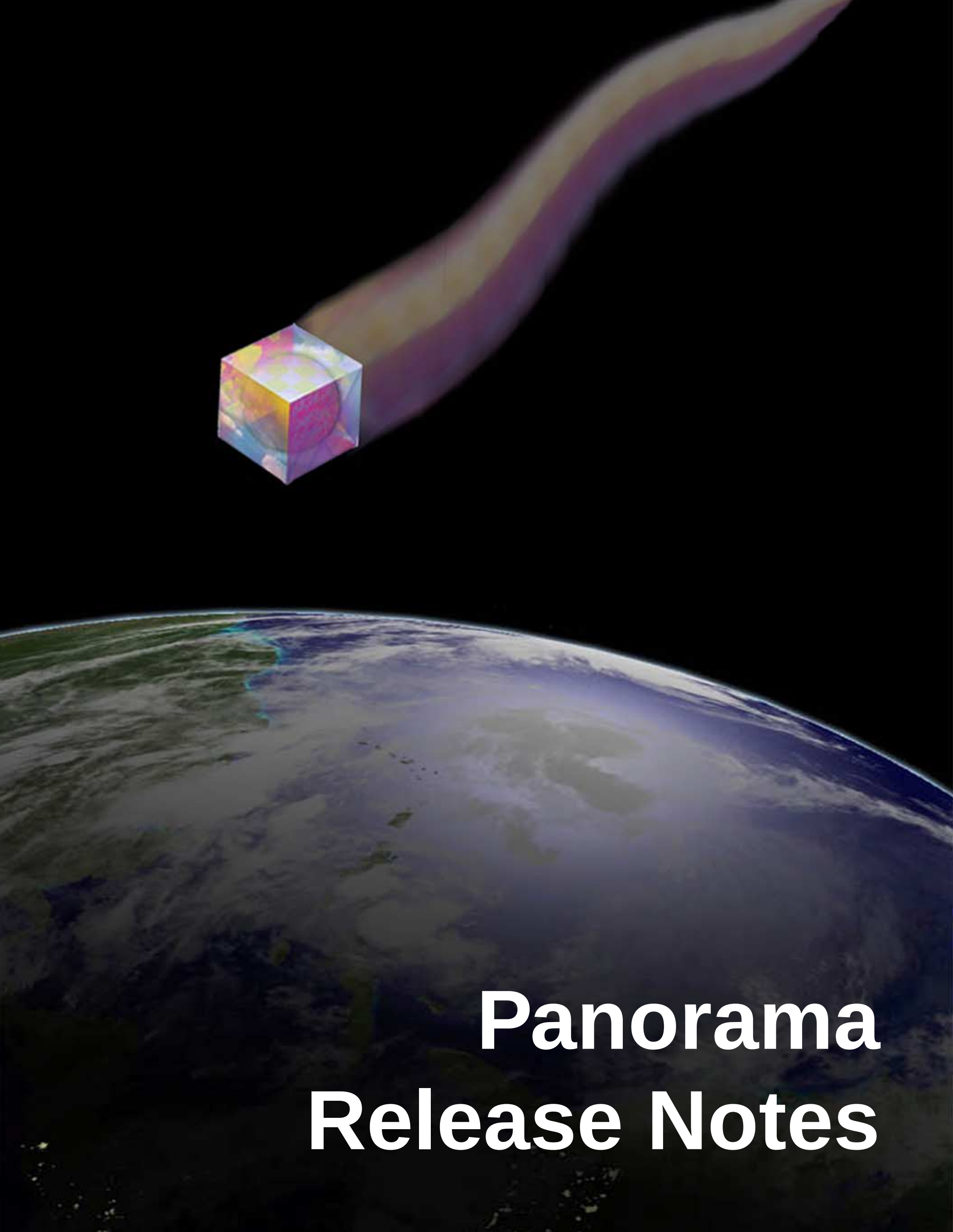


Panorama Release Notes

Panorama Release Notes (Version 6.0)

ProVUE Development
18685-A Main Street PMB 356
Huntington Beach, CA 92648
USA

www.provue.com

Table of Contents

– Click on any entry to jump to the page —

History of Panorama

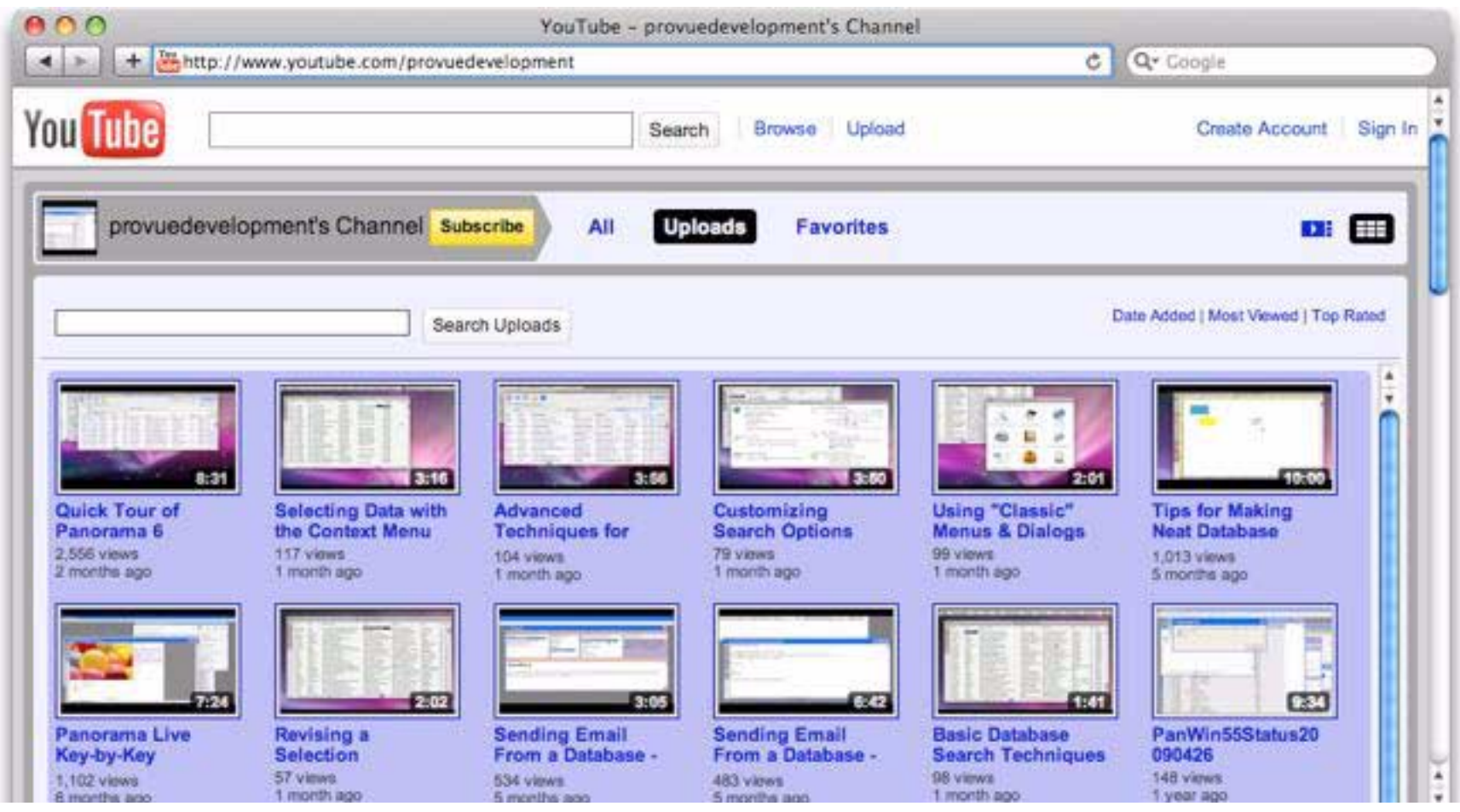
Software is always a work in progress. The first line of code for Panorama was written in 1986. Today Panorama contains over 400,000 lines of code, and we keep making progress (check **www.provue.com** for information about the latest updates. If you are already familiar with a previous version of Panorama you can find out what has changed by examining the listings below. We've also included a general corporate history of ProVUE and its products over the years (see "General Corporate History" on page 180).

Version 6.0.0

This version was released in June 2010 for Mac OS X 10.4 and above, and Windows XP and above. This is the first version of Panorama to be native on Intel based Macintoshes (it is actually universal, so it also runs on PowerPC computers).

Panorama Training Videos on YouTube

This isn't strictly a new Panorama feature, but ProVUE Development now has a video channel on YouTube (**http://www.youtube.com/provuedevelopment**). We're adding new videos on a regular basis, so be sure to check in frequently to see what's new.



Intel Native on Mac OS X

Panorama 6.0 is fully native on Intel processors (and also supports PowerPC, see below). Benefits of the move to Intel native code include:

- Runs about 5x faster (sorts, selects, procedure code, etc.)
- Launches up to 10x faster
- Uses about 1/3 as much overall memory, so other programs run faster
- Does not require Rosetta
- More stable (avoids Rosetta bugs)

Snow Leopard Support

A happy byproduct of the move to Intel native code is that Panorama icons now work with Snow Leopard. If you were experiencing icon problems with Panorama 5.x and Snow Leopard these will be resolved when you install Panorama 6.0. (Technical note — we think Apple's bug in the Snow Leopard Finder relates to the use of PEF vs. Mach-O packaging for the application.)

PowerPC Support

Panorama 6.0 is Universal, so it runs on both Intel and PowerPC machines (Mac OS X 10.4 or later). Note: While Panorama 6.0 is much faster than Panorama 5.5 on Intel machines, this is not true on PowerPC computers.

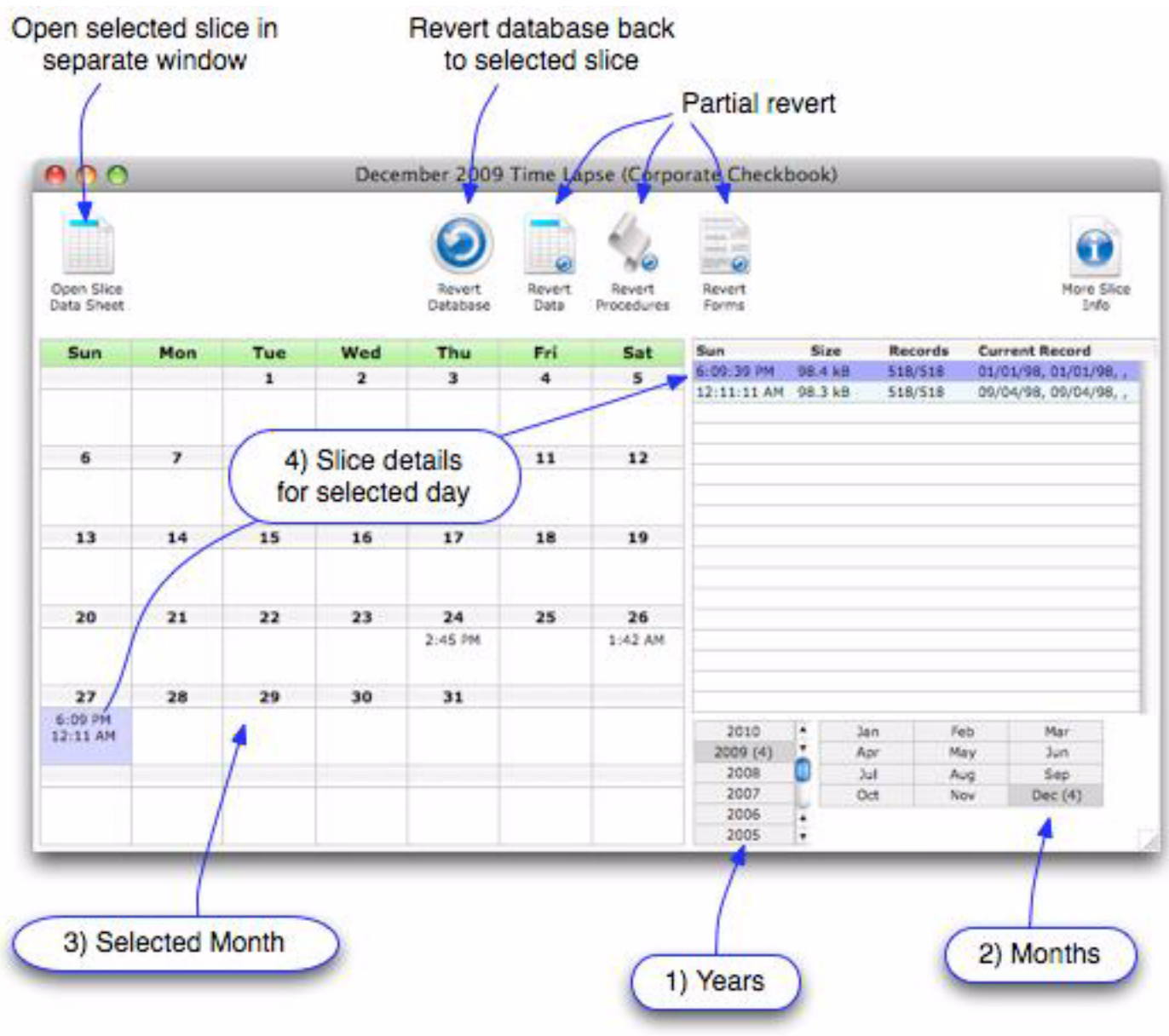
Total Recall

Total Recall allows Panorama to fully recover after a crash -- whether due to a power outage, hardware problem, system crash, or Panorama bug (never!). When Panorama is relaunched after a crash it asks you if you want to start over or resume where you left off.

If you press **Resume Previous Session** Panorama automatically restores everything just as you left it -- all open files, open windows, etc. Only the last few seconds of work will be lost. It's almost as if nothing has happened at all — you can just continue with your work as if nothing had happened. To learn more see “Total Recall (Auto-Save / Crash Recovery)” on page 66 of the *Panorama Handbook*.

Time Lapse

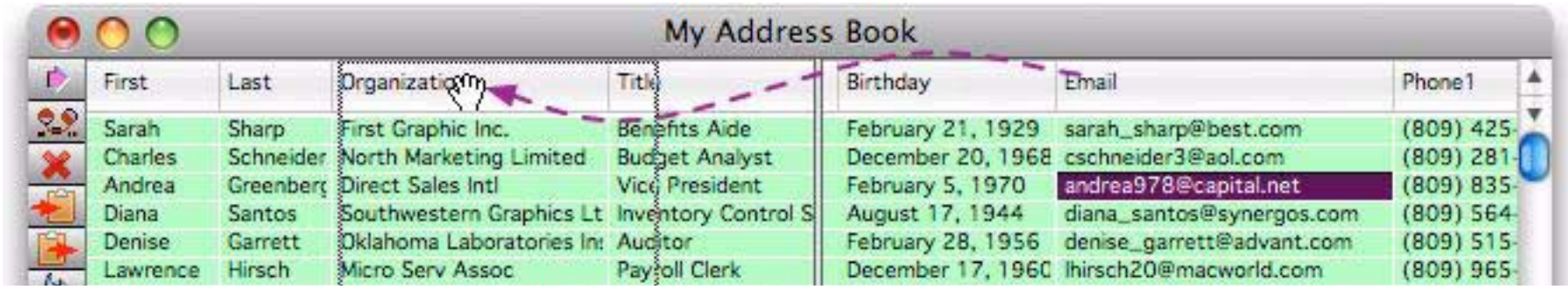
Panorama's new Time Lapse feature is like **Revert to Saved** on steroids. It allows you to roll back to previous versions of your database made earlier today or at various times in the past. In addition to doing a full revert, you can also selectively revert only the data, only the procedures, or only the forms. You can also open previous versions of the database (called "slices") in separate windows so that you can compare them with the current database or even copy and paste data, code or form objects between versions. The main Time Lapse window shows the recent history of the current database in a monthly calendar format.



For more information see "Time Lapse" on page 67 of the *Panorama Handbook*.

Dragging to Re-Arrange Data Sheet Fields

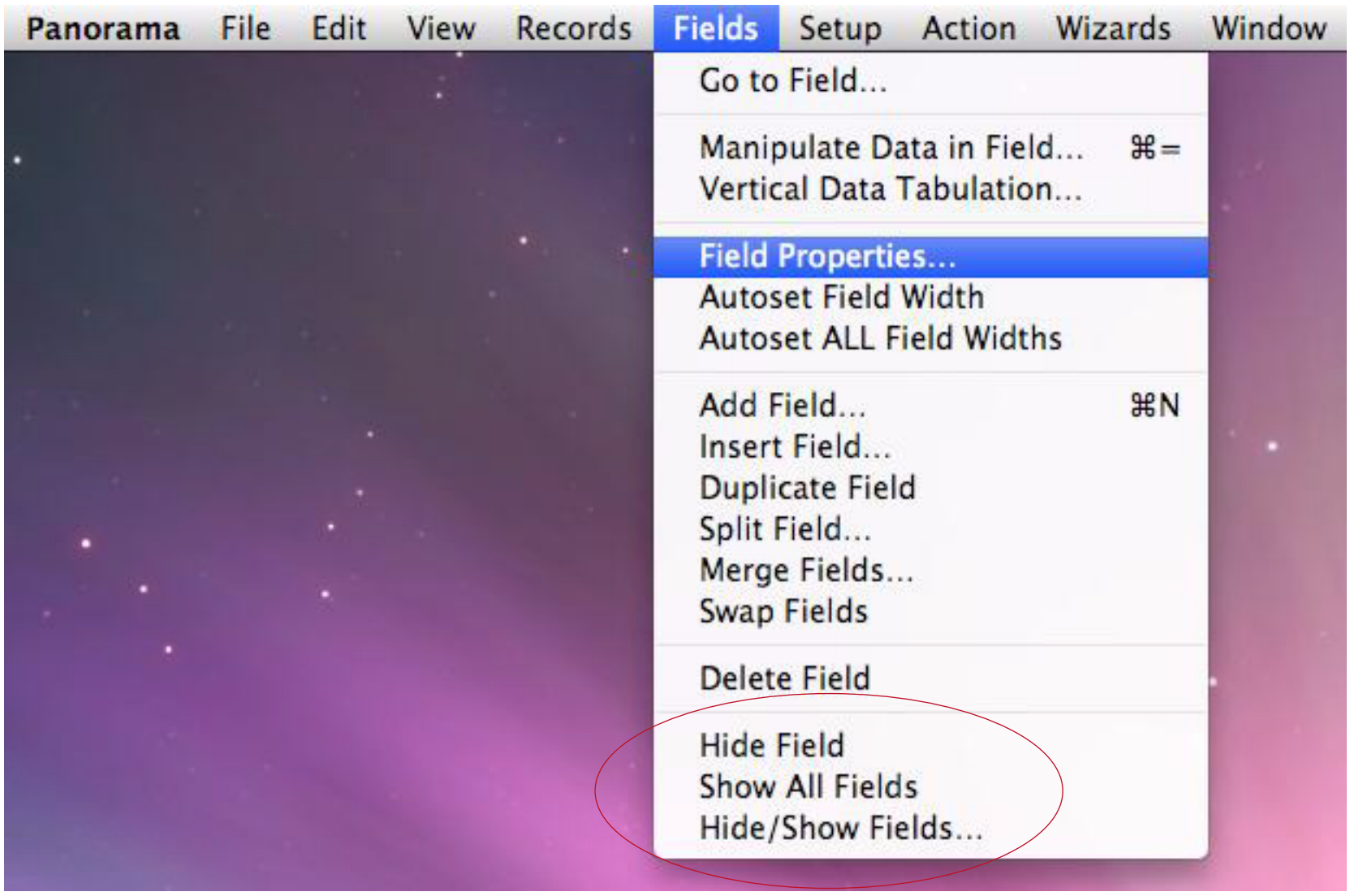
Panorama 6 allows you to re-arrange fields in the data sheet simply by dragging the column header.



To learn more "Re Arranging the Field Order" on page 200 of the *Panorama Handbook*.

Temporarily Hiding Data Sheet Columns

If you don't want to see one or more fields, Panorama 6 now allows you to temporarily hide them in the data sheet. This is great for de-cluttering the data sheet and also eliminates the need to create a custom form for many simple reports. The commands for hiding and showing fields are at the bottom of the Fields menu. (In addition to the menu bar, these commands also appear when you right-click on a data sheet column title.)



To hide and / or show a bunch of fields at once, open the **Hide/Show Fields** dialog.

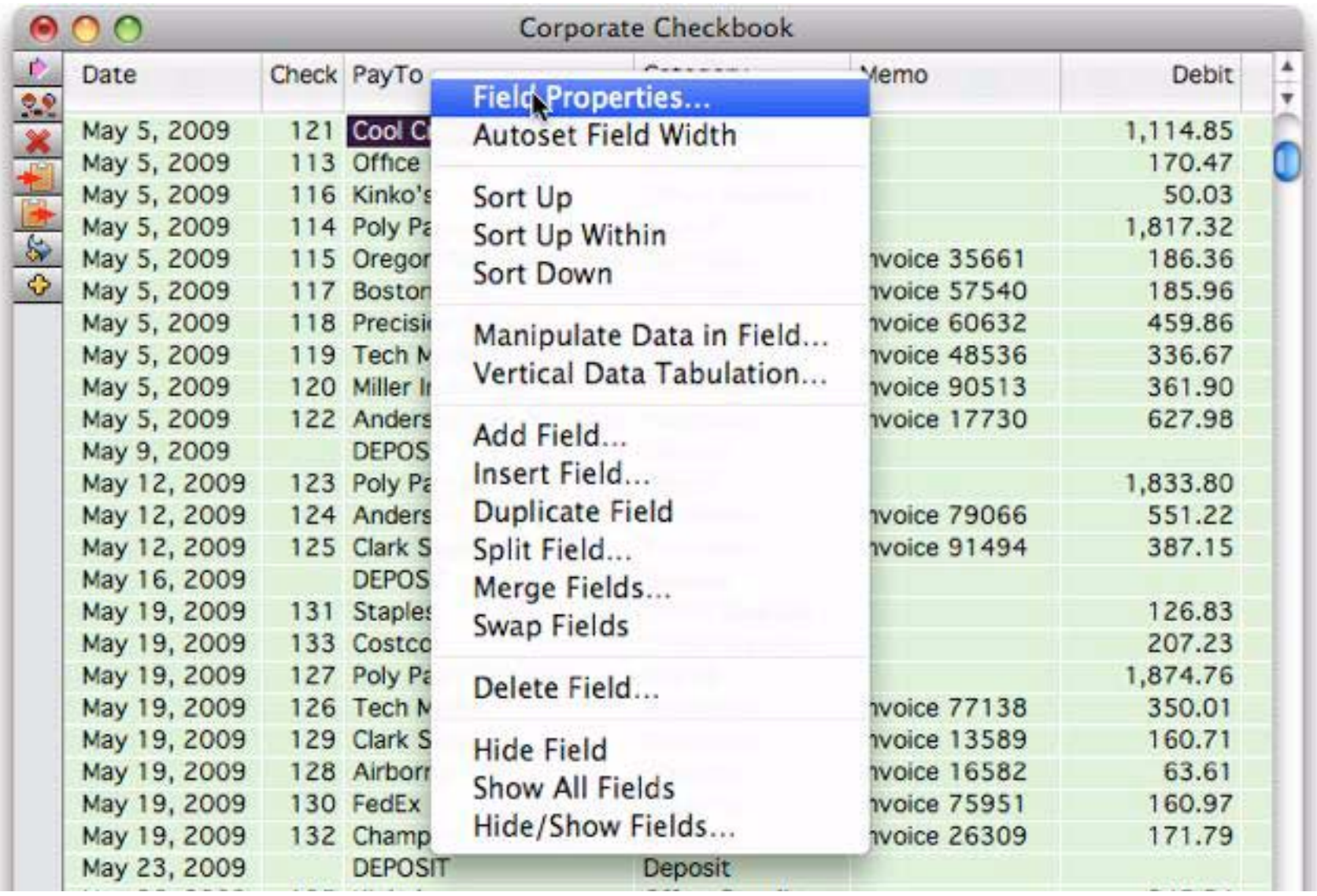
Learn more about "Hiding and Showing Fields" on page 209 of the *Panorama Handbook*.

Data Sheet Context Menus (Right-Click)

Right clicking anywhere on the data sheet will access a menu of commands (if your mouse doesn't have two buttons, you can also hold down the Control key and click). The contents of the menu will depend on where you click.

Right Clicking on a Column Header

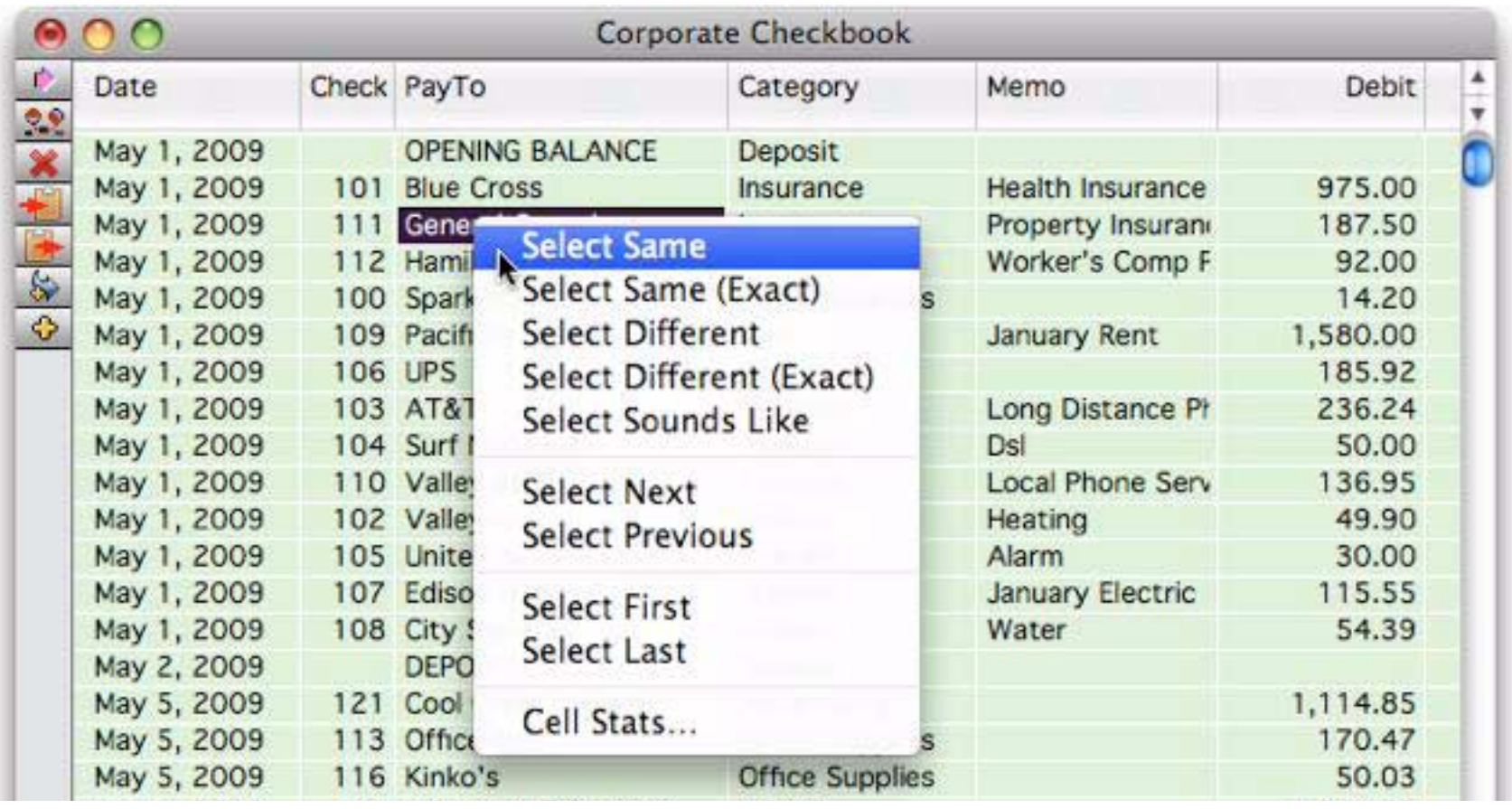
Right-click in a column header and you'll see a list of commands from the **Sort** and **Fields** menus.



These commands are described in chapters 5, 8 and 12 of the Panorama Handbook.

Right Clicking on a Text Cell

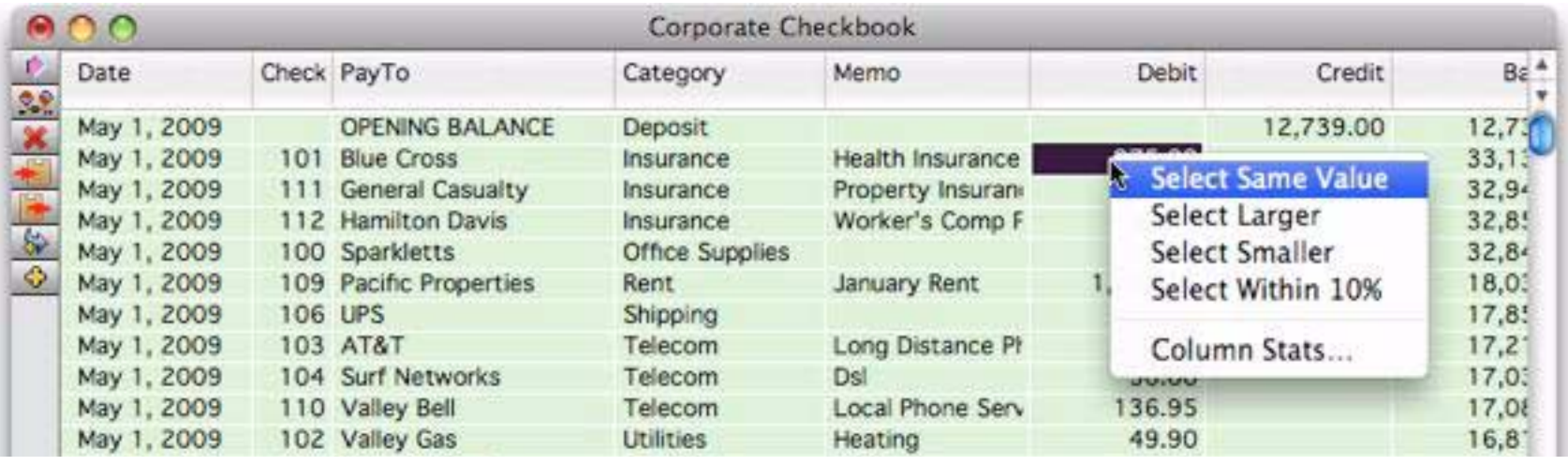
Right-clicking on a text cell produces this menu (see "Selecting with the Context Menu" on page 333 of the *Panorama Handbook*).



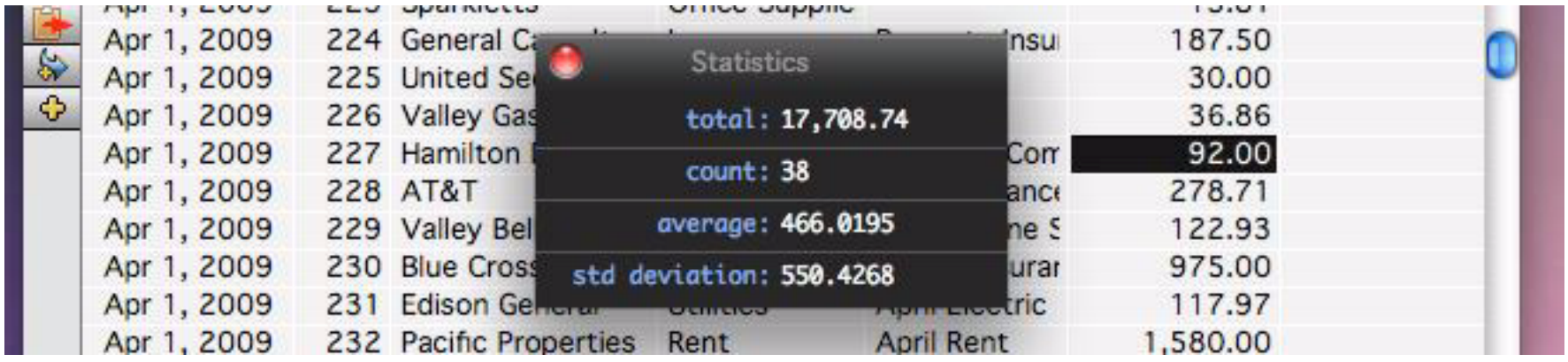
The first five commands select data that is identical to or very similar to the currently selected cell. The next four commands allow you to "walk" thru each value in a field - it's especially handy for fields with a handful of values (for example a category field). The **Cell Stats...** command opens a small modal window that displays information about the current cell.

Right-clicking on a Numeric Cell

Right-clicking on a numeric cell produces this menu (see "Selecting with the Context Menu" on page 333 of the *Panorama Handbook*).



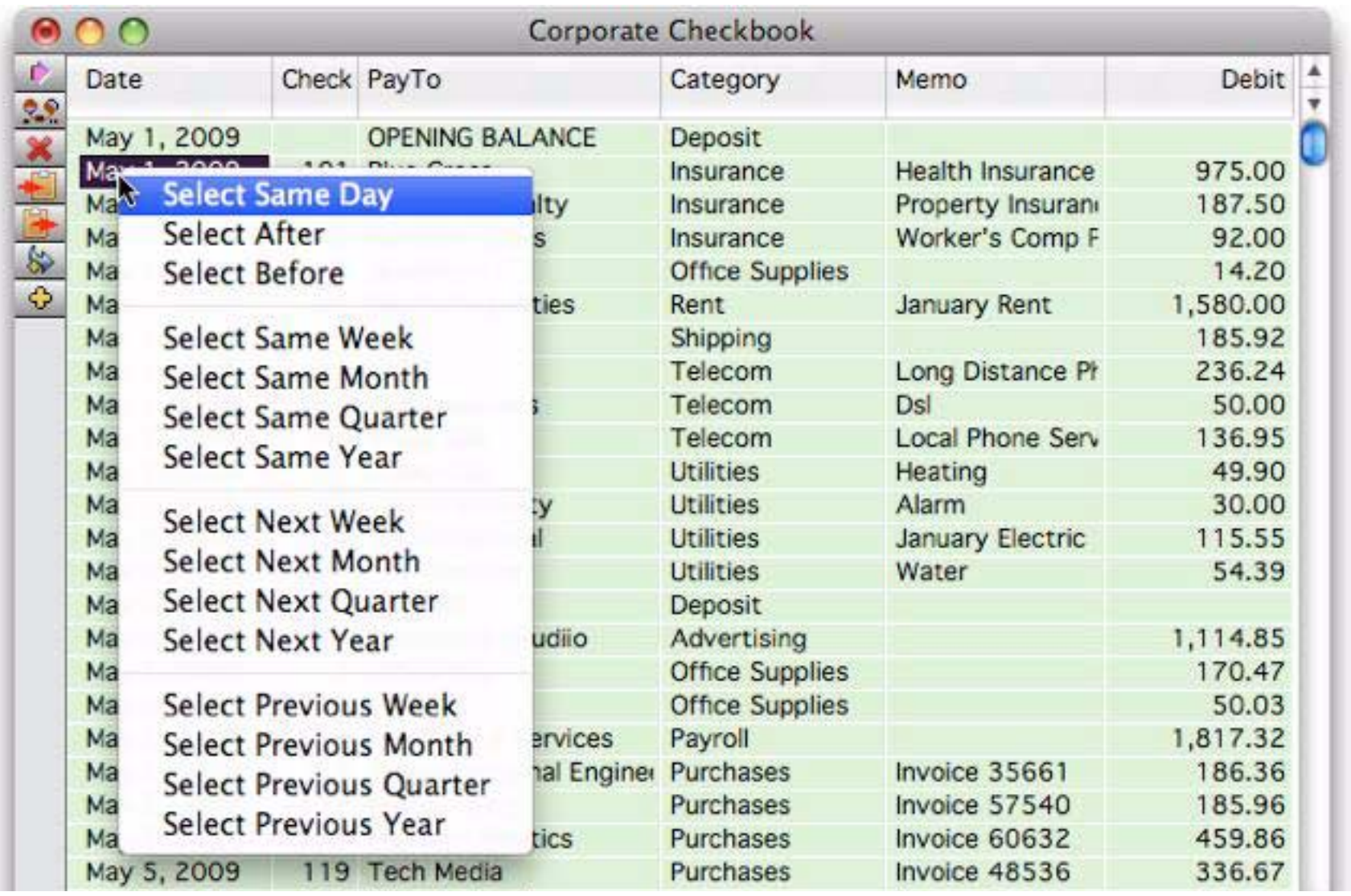
The first four commands select data related to the currently selected cell. The **Column Stats...** command opens a small modal window that displays information about the current column.



The statistics are calculated for selected, non-empty cells in the column (see "Quick Subtotals" on page 335 of the *Panorama Handbook*).

Right Clicking on a Date Cell

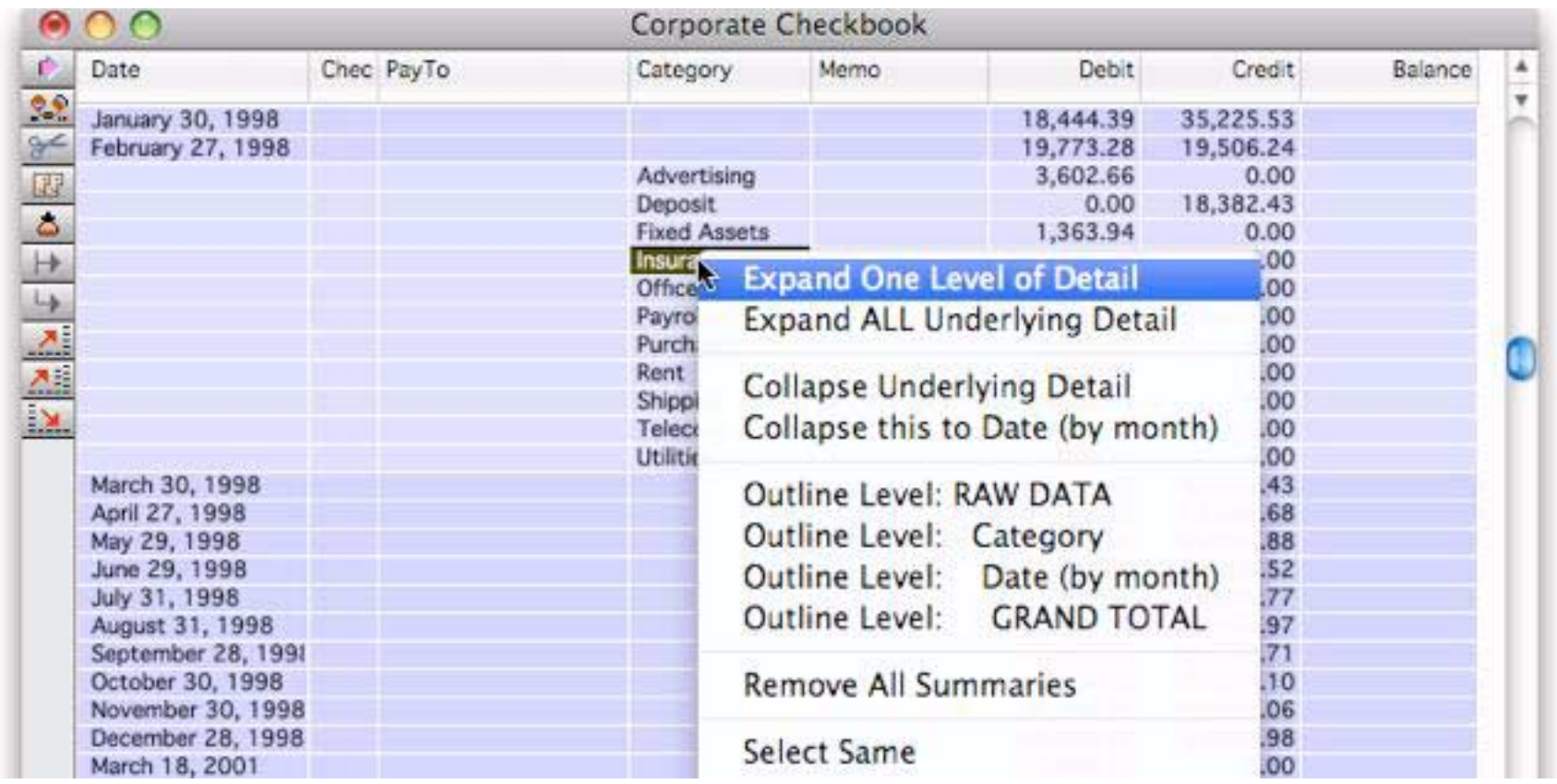
Right-clicking on a date cell produces this menu (see "Selecting with the Context Menu" on page 333 of the *Panorama Handbook*).



These commands allow you to select dates related to the current date, and to quickly "walk" thru the data by week, month, quarter or year.

Right Clicking for Outline Management

Whenever summary records are present in the database the context menu will contain additional commands for manipulating the outline (see "Expanding and Collapsing the Summary Outline" on page 376 of the *Panorama Handbook*):



The **Expand One Level of Detail** command expands the next level of detail associated with the current summary record. (This is the same as the **Expand** tool in the tool bar.)

The **Expand ALL Underlying Detail** command expands all detail associated with the current summary record, right down to the raw data. (This is the same as the **Expand All** tool in the tool bar.)

The **Collapse Underlying Detail** command collapses all detail associated with the current summary record. (This is the same as the **Collapse** tool in the tool bar.)

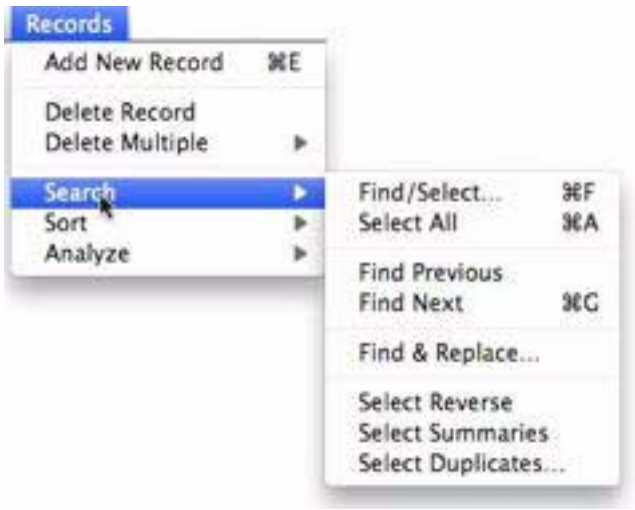
The **Collapse this ...** command collapses to the next higher level of detail. The current record and all of it's "siblings" will be collapsed, along with any associated detail. (There is no tool bar item corresponding to this command.)

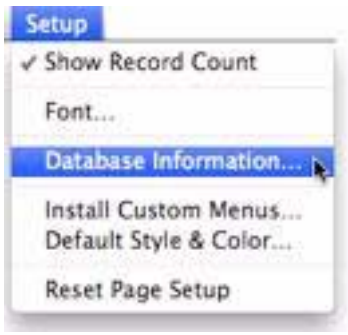
The **Outline Level** commands allow you to quickly expand and collapse the entire outline. (This is the same as using the Outline Level dialog, but is much faster.)

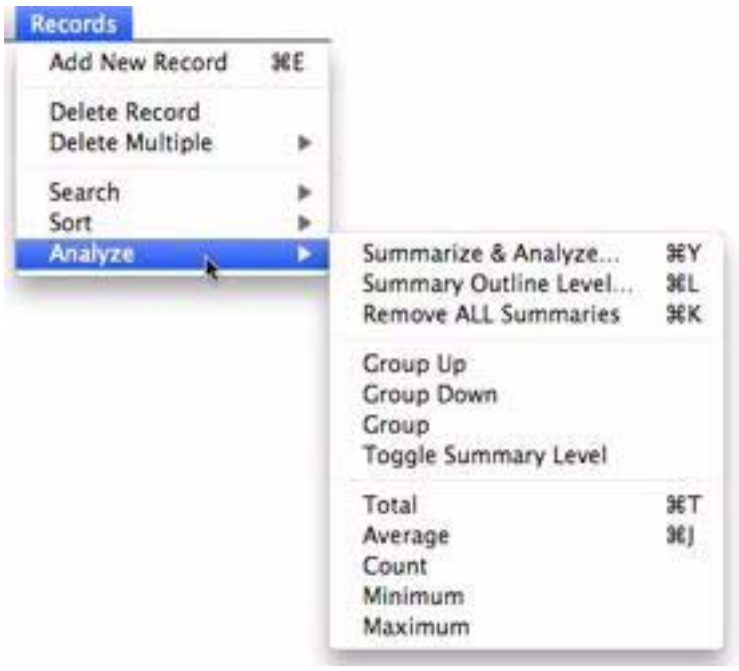
When you're done with an outline you can use the **Remove All Summaries** command in the context menu (or the **Analyze** menu) to get back to the raw data (however, if you are going to do another analysis with the **Analyze Data** dialog this isn't necessary).

New Menu Organization

If you've used previous versions of Panorama, you'll immediately notice that many of the menus and dialogs have been re-arranged to make them more logical and easier to learn, especially in the data sheet. In particular, you'll notice that the **Search**, **Sort** and **Math** menus have now been replaced by the **Records** and **Fields** menus.







"Classic" Menus & Dialogs

If you've been using Panorama for decades, it might take some time to get used to the new menu arrangement. To make the transition easier, you can temporarily, or even permanently, switch back to the old arrangement. To do this, start by opening the Preferences window, then open General Preferences (see "General Preferences" on page 9 of *Wizards & Demos*).

Now check the Use "Classic" menus option. This preference takes place immediately, you don't have to even close the Preference window. Just click on a data sheet window and you'll see the old menu arrangement is now active. When the "classic" mode is active, the dialogs in these menu also revert back to the old style. You can flip back and forth between the modern and classic arrangements at any time.

Including "Built-In" Wizards

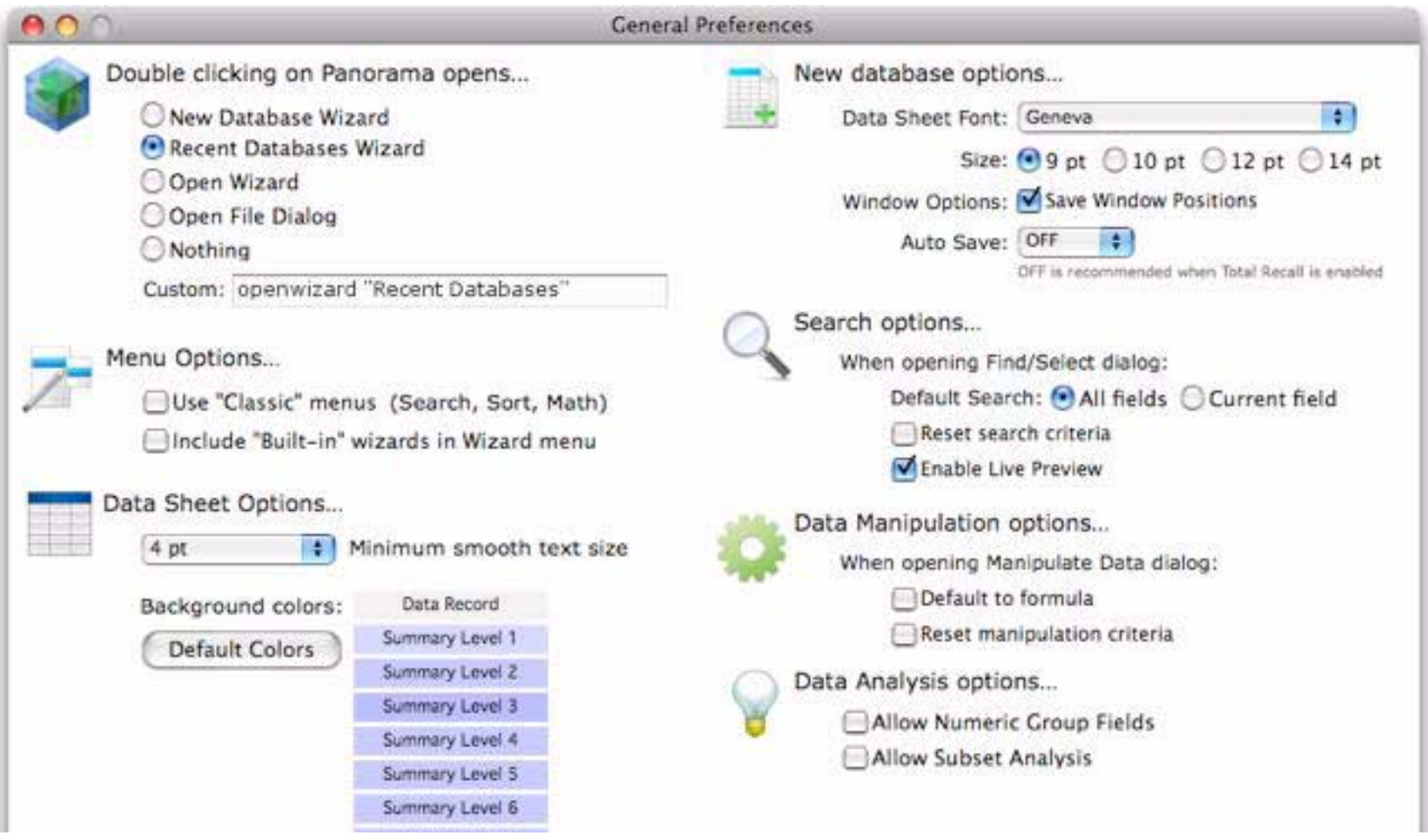
Many commands that started out as wizards are now available in Panorama's regular menus -- examples include Help & Documentation, New Database, Open Recent, Import Text, Export Text, and many others. Starting with Panorama 6, these "built-in" wizards are no longer listed in the Wizard menu. If you're used to accessing these commands from the Wizard menu, simply set the preference option to enable them. You'll then find all of the wizards are listed, even if they are also included in Panorama's regular menus.

Preferences

Panorama preferences are now accesible from the Panorama menu (see "Preferences" on page 8 of *Wizards & Demos*). Choosing **Panorama**-> **Preferences** opens the main preferences window (the exact configuration may vary somewhat):

General Preferences

The first option is always **General Preferences**.



See "General Preferences" on page 9 of *Wizards & Demos* for detailed explanations of each of these options.

New *Find/Select* Dialog

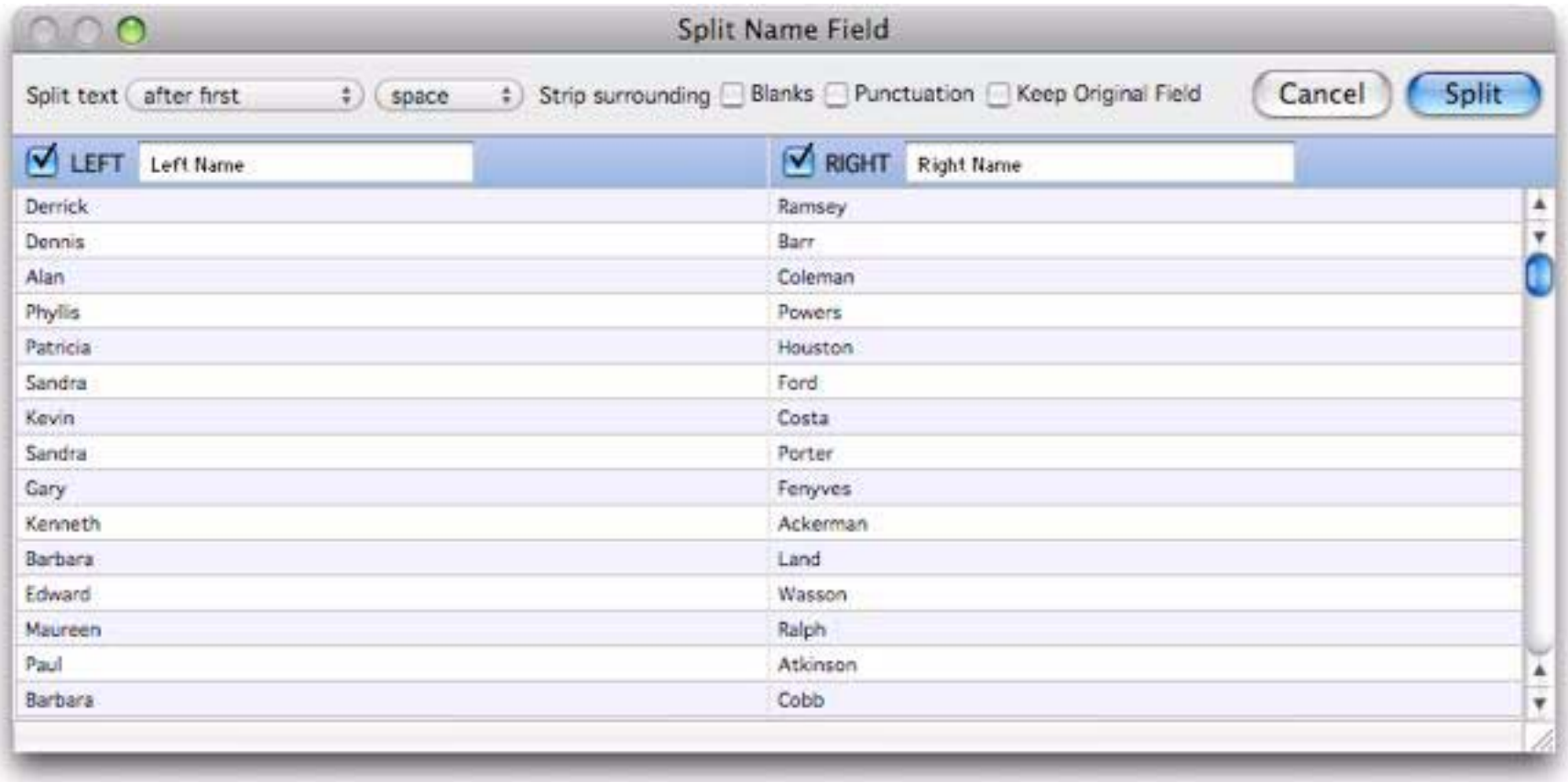
The Panorama 5.5 Search menu had six different commands for selecting data. Panorama 6 simplifies these into a single new dialog that is more powerful and more approachable for non-expert users.

First	Last	Organization	Title	Address	City	St	Zip	Birthday
Marcus	Phillips	Little Architects Corp.	Quality Assurance	549 W Grove Ct	Hadley	MA	01035	July 23
Mary	Atkinson	Stamford Industrial Services	Engineering Tech	697 W. Whitney Cir.	Northampton	MA	01063	Septem
Linda	Michael	Clark Digital Intl	Assistant Engineer	4 E. Raymond Dr.	Shutesbury	MA	01072	Decemb
Anna	Hammond	Dartmouth Engineering International	Community	2978 Berkshire Rt	West Springfield	MA	01089	Octobe
Donald	Simons	Middlesex Energy Inc	Assistant Engineer	39411 S. Tremont Place	Ft Devens	MA	01433	March 2
Walter	Lund	Northern Gas Corp	Engineering	284 E Dover Rt	Worcester	MA	01602	July 17
Jason	Herron	Johnson Electronics International	Engineering Tech	45887 S.W Humboldt Rd.	Worcester	MA	01608	April 16
Cynthia	Doyle	Little Engineering College	Budget & Research	7826 S Lee Place	Hopkinton	MA	01748	Octobe
Nicole	Gilmore	Venture Union Assoc	Mechanical	464 N.E. Arizona Cir	Sudbury	MA	01776	January
Carl	Jenkins	Chicago Clinic Systems	Engineering Aide	5301 W Brown St.	North Billerica	MA	01862	May 4,
William	Morse	Coast Communication Corp.	Engineering	15418 S Pacific St	Canton	MA	02021	March 7
William	Gorman	Iowa Art Intl	Engineering Aide	43029 N Stanford Rd.	Dover	MA	02030	Februar

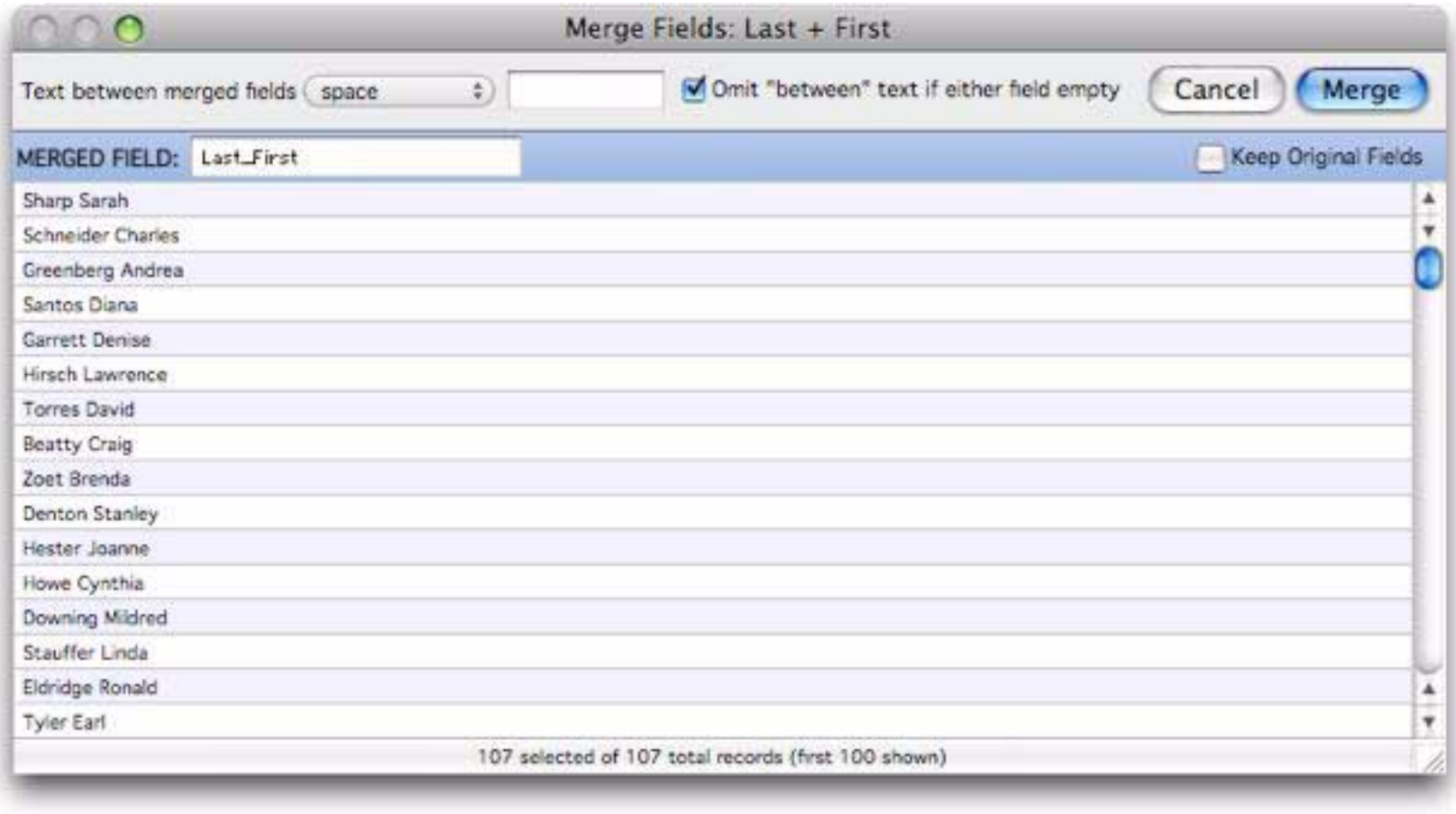
This dialog combines the features of the Find / Select dialog, Formula Find / Select and Live Clairvoyance wizard into one easy-to-use dialog. See “The Find / Select Dialog” on page 336 of the *Panorama Handbook*.

Splitting and Merging Fields

Splitting and merging fields are commonly needed operations that are difficult and slow to do in most databases. Panorama has always been able to do these operations quickly if you knew the secret incantations (via **Formula Fill**), but now Panorama 6 makes these operations super easy and fast for anyone, including non-technical users. To split an existing field in two use the new **Split Field** dialog in the **Fields** menu. The dialog allows you to specify how the data in the field will be split, and what the names of the new fields are (see "Splitting a Field" on page 201 of the *Panorama Handbook*).

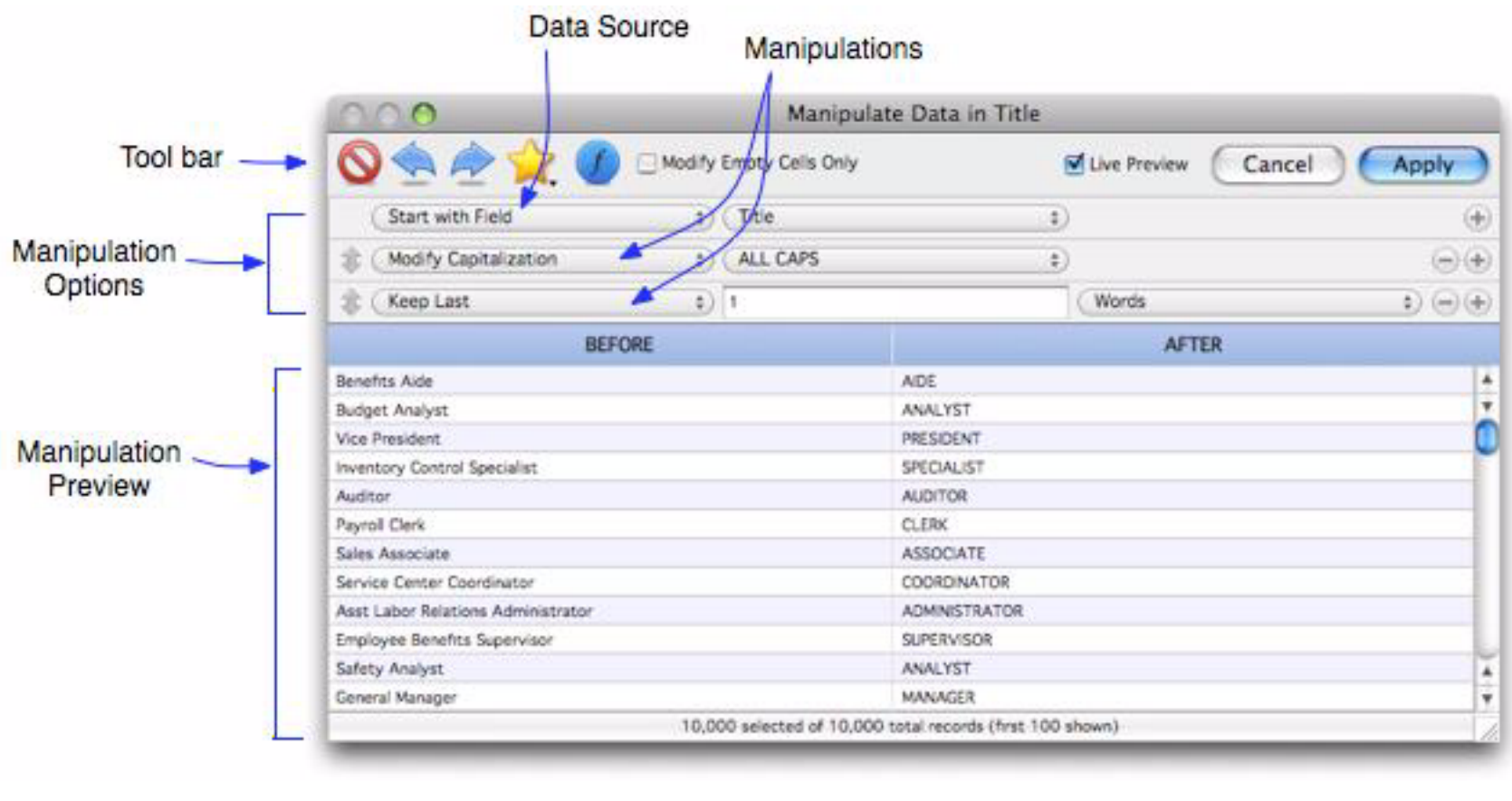


To merge two adjacent fields use the **Merge Fields** dialog, also in the **Fields** menu. The dialog allows you to specify "joiner" text to be inserted between the merged fields (see "Merging Adjacent Fields" on page 207 of the *Panorama Handbook*).



New *Manipulate Data in Field* Dialog

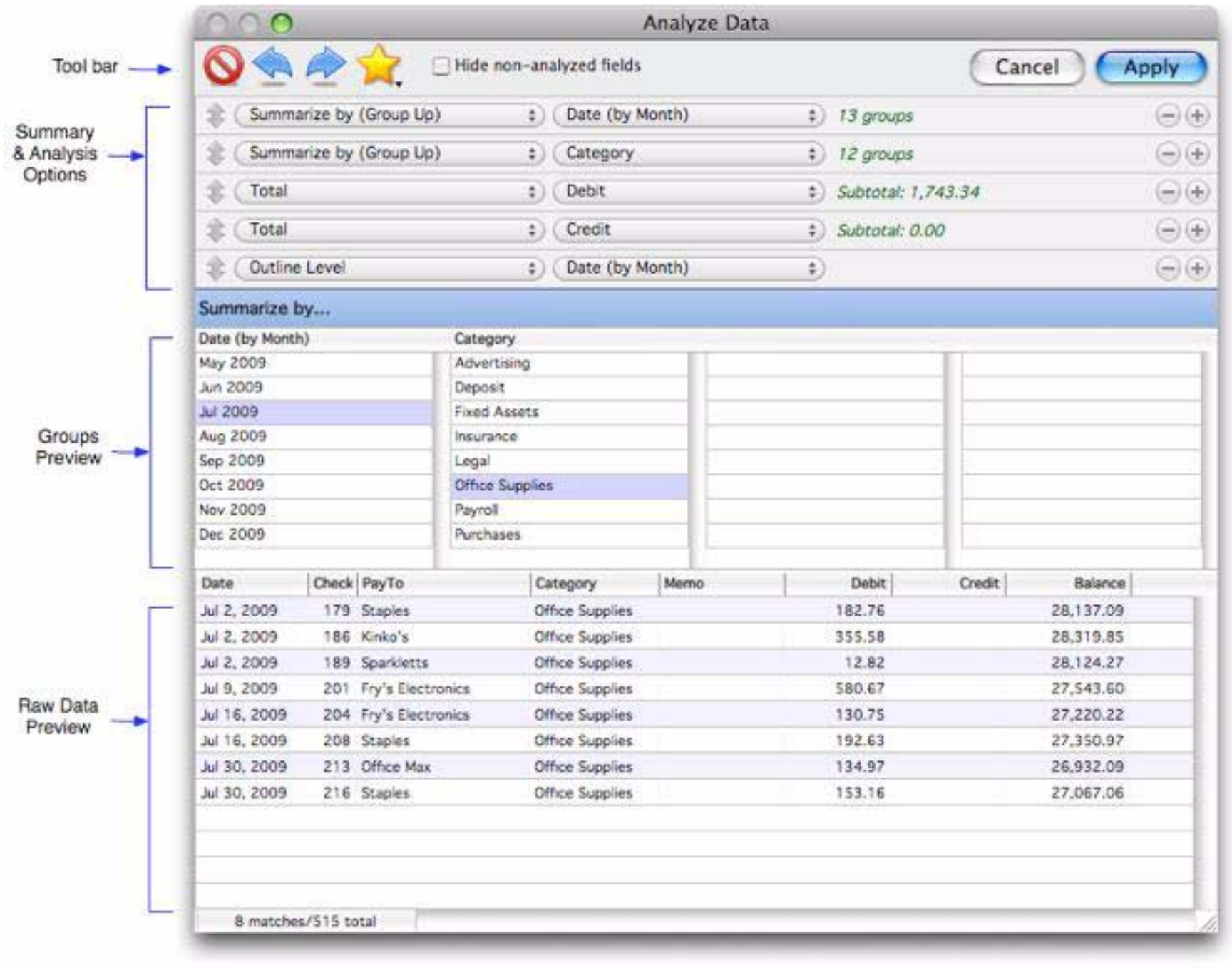
The new Panorama 6 **Manipulate Data in Field** dialog replaces the **Fill**, **Formula Fill**, **Empty Fill** and **Sequence** commands. Like these commands you start by clicking on the field you want to manipulate (and possibly also selecting only the records you want to manipulate), then you open the dialog. This new dialog is designed to make the power of Panorama's data manipulation capabilities more accessible for non-power users. Operations that in the past would have required concocting an arcane formula can now easily be performed with a few menu selections.



To learn the details of this dialog see "The Manipulate Data Dialog" on page 434 of the *Panorama Handbook*.

New *Analyze Data* Dialog

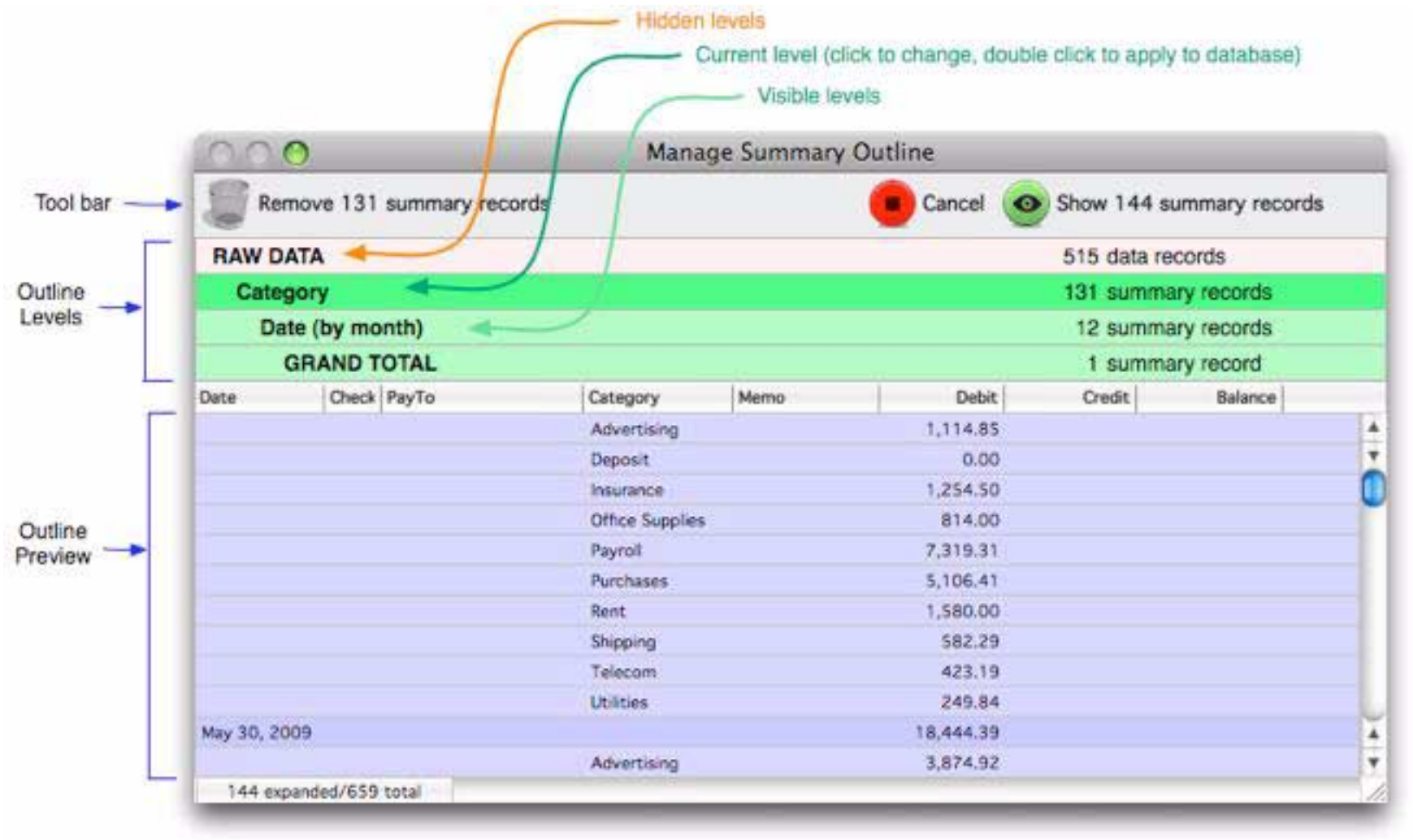
This new dialog is a one-stop alternative to using the Group / Math / Outline commands. Instead of building an outline incrementally one menu command at a time, you can now set up the outline by choosing from simple pop-up menus (you can still do it the old way if you like, but we think you'll like the new dialog much better after you've tried it once or twice).



Much of Chapter 10, Summaries, has been rewritten to reflect this new dialog. See "The Summarize & Analyze Dialog" on page 371 of the *Panorama Handbook*.

New *Manage Outline* Dialog

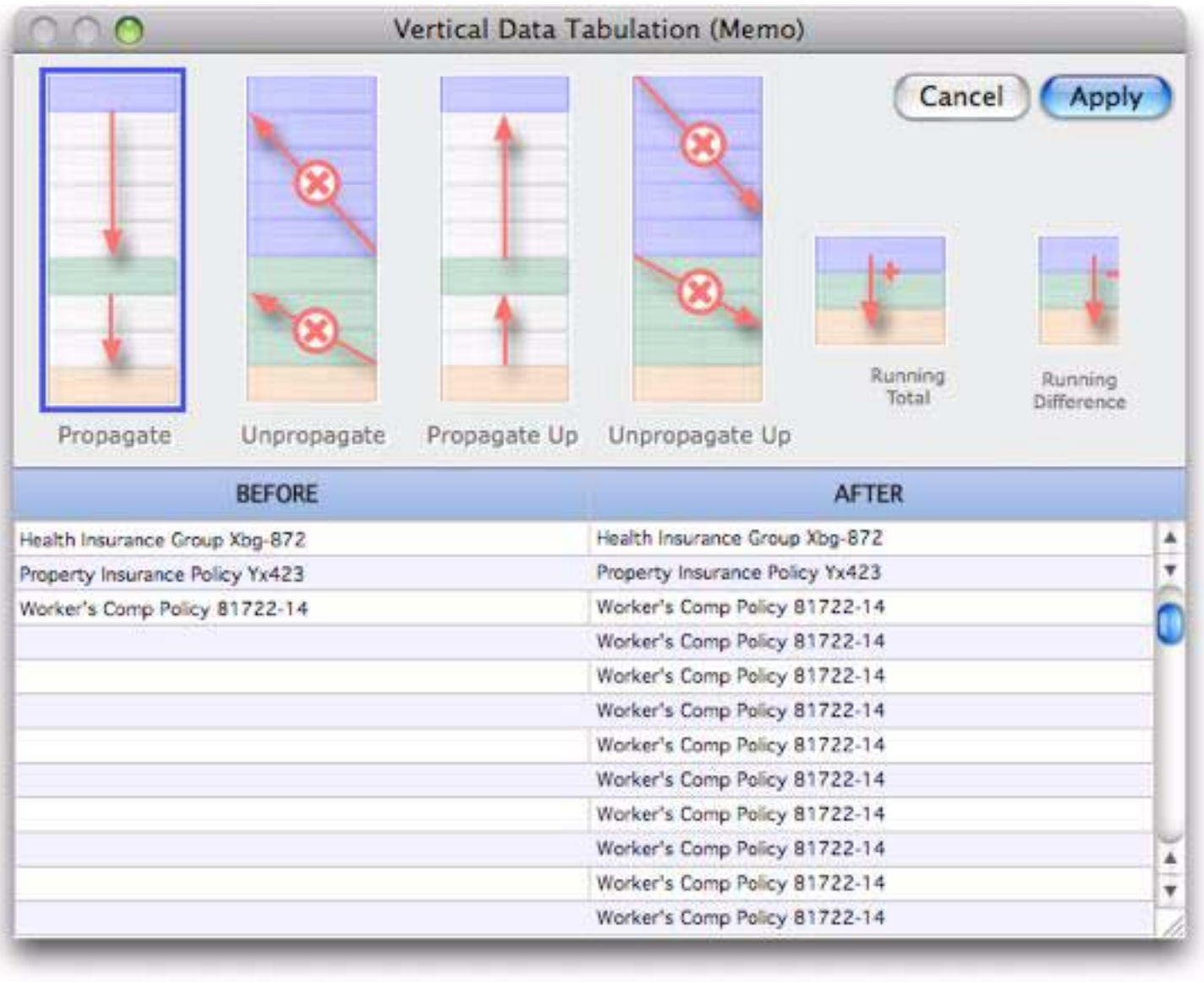
This dialog replaces three dialogs in previous versions of Panorama — **Outline Level**, **Remove Summaries**, and **Remove Detail**. (Note: In many cases you'll probably want to skip this dialog completly and use the outline management tools in the Data Sheet context menu, see "Expanding and Collapsing the Summary Outline" on page 376 of the *Panorama Handbook*.)



To learn more about this dialog see "Expanding and Collapsing the Overall Summary Outline" on page 380 of the *Panorama Handbook*.

New *Vertical Data Tabulation* Dialog

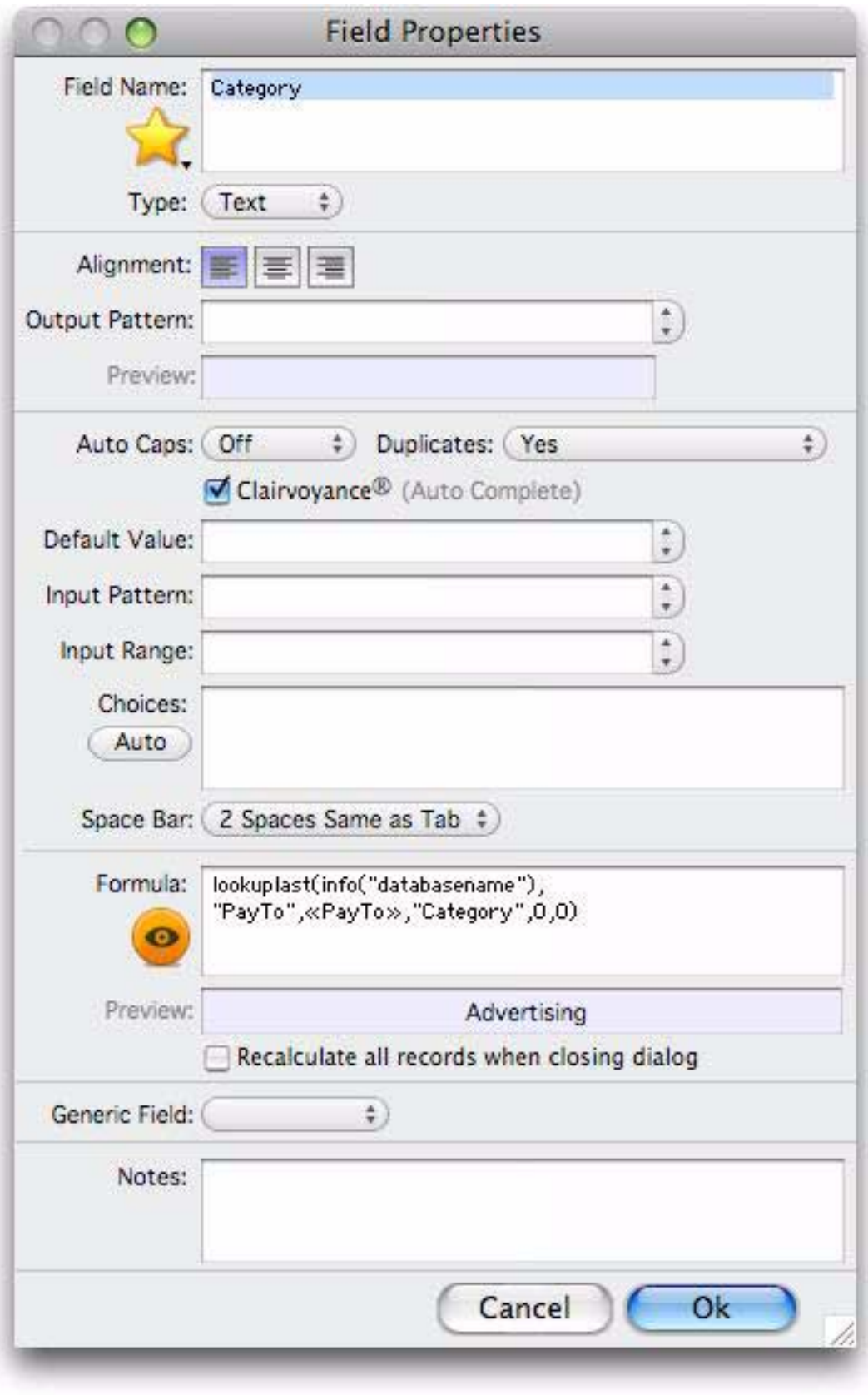
The **Manipulate Data** dialog manipulates each record independently. The **Vertical Data Tabulation** dialog, on the other hand, manipulates data vertically across multiple records.



This dialog replaces six separate menu commands: **Propagate**, **Unpropagate**, **Propagate Up**, **Unpropagate Up**, **Running Total** and **Running Difference**. See "Vertical Data Tabulation" on page 465 of the *Panorama Handbook*.

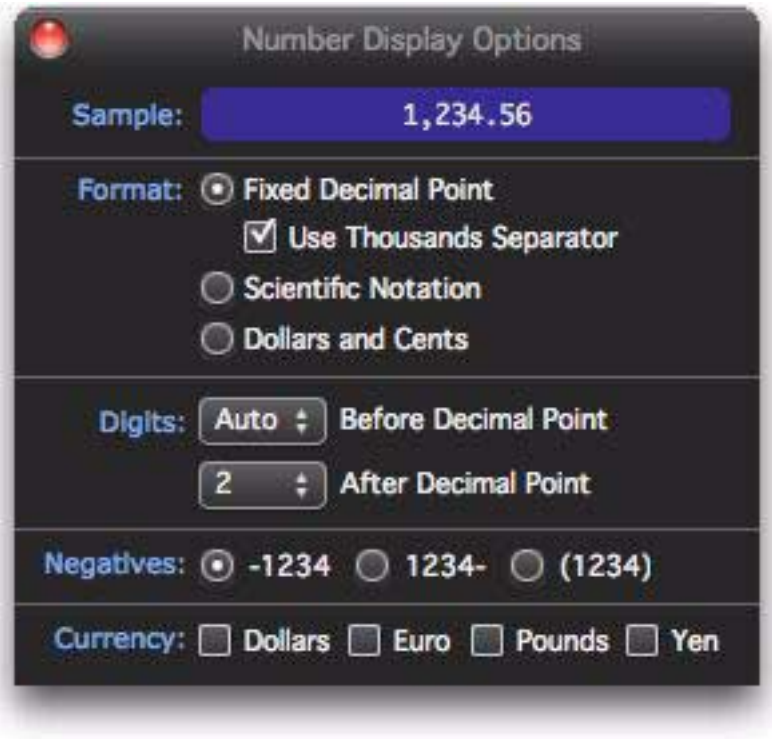
Field Properties Dialog

This dialog has been made both more complete and more streamlined, see "Modifying the Properties of an Existing Field" on page 195 of the *Panorama Handbook*. (Note: This same dialog is used by the **Add Field** and **Insert Field** commands.)

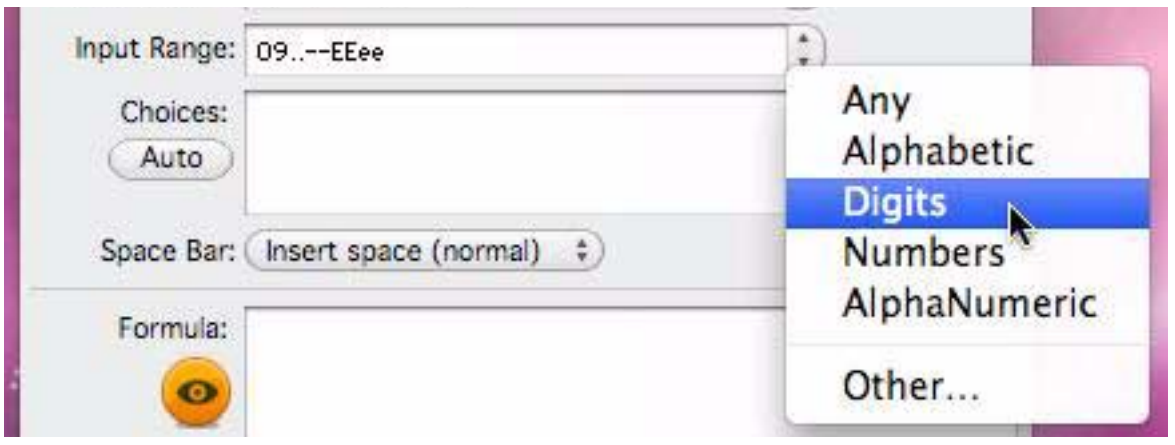


The Field Name, Type, Digits and Alignment are the same as they were before. More on that yellow star icon in a moment.

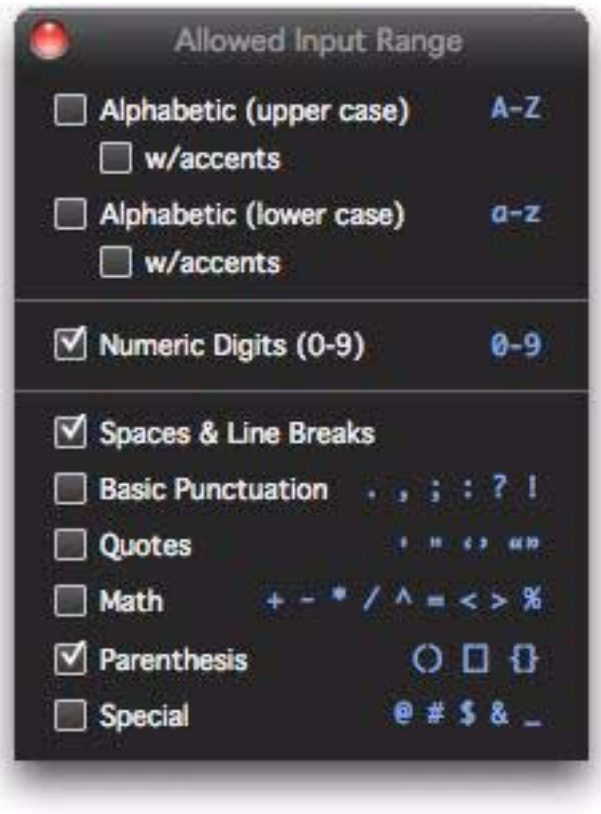
The output pattern for numeric or date fields can be typed in directly if you know it. The dialog shows a preview of what the pattern will look like in use. Click on the pop-up menu to choose from the most commonly used patterns (this menu adjusts depending on whether this is a number or date field). For number fields you can also choose Other… to pick from more advanced options (see “Numeric Output Patterns” on page 250 and “Date Output Patterns” on page 255 of the *Panorama Handbook*). Click the red close button when you've set up the pattern you want (you can also drag this window by clicking on the top area).



The next section of the dialog has data entry options. These all work pretty much the same as they did in earlier versions of Panorama. The Default Value, Input Pattern and Input Range can be typed in directly, and you can also pick from a pop-up menu with common values for these options (see “Input Patterns” on page 291 of the *Panorama Handbook*).



The Input Range pop-up menu also includes an Other… option that brings up a separate window with preset options for common input ranges.



The Formula option allows you to edit the formula associated with this field. The preview area shows what the value of this field will be for the current field. You can type in the formula manually, and you can also use menu items to automatically type in field and function names. Another option is to click on the orange Build Formula icon. This opens a new window that allows you to build a formula step-by-step. The operation of this dialog is identical to the Manipulate Data Dialog. See “Spreadsheet Mode Calculations” on page 303 of the *Panorama Handbook*.

However you set up the formula, changes you make to the formula usually apply only to new data entered into the database. However, if you check the Recalculate all records when closing dialog option Panorama will recalculate all existing cells in this field when the Ok button is pressed (in other words, it will automatically do a formula fill for you.)

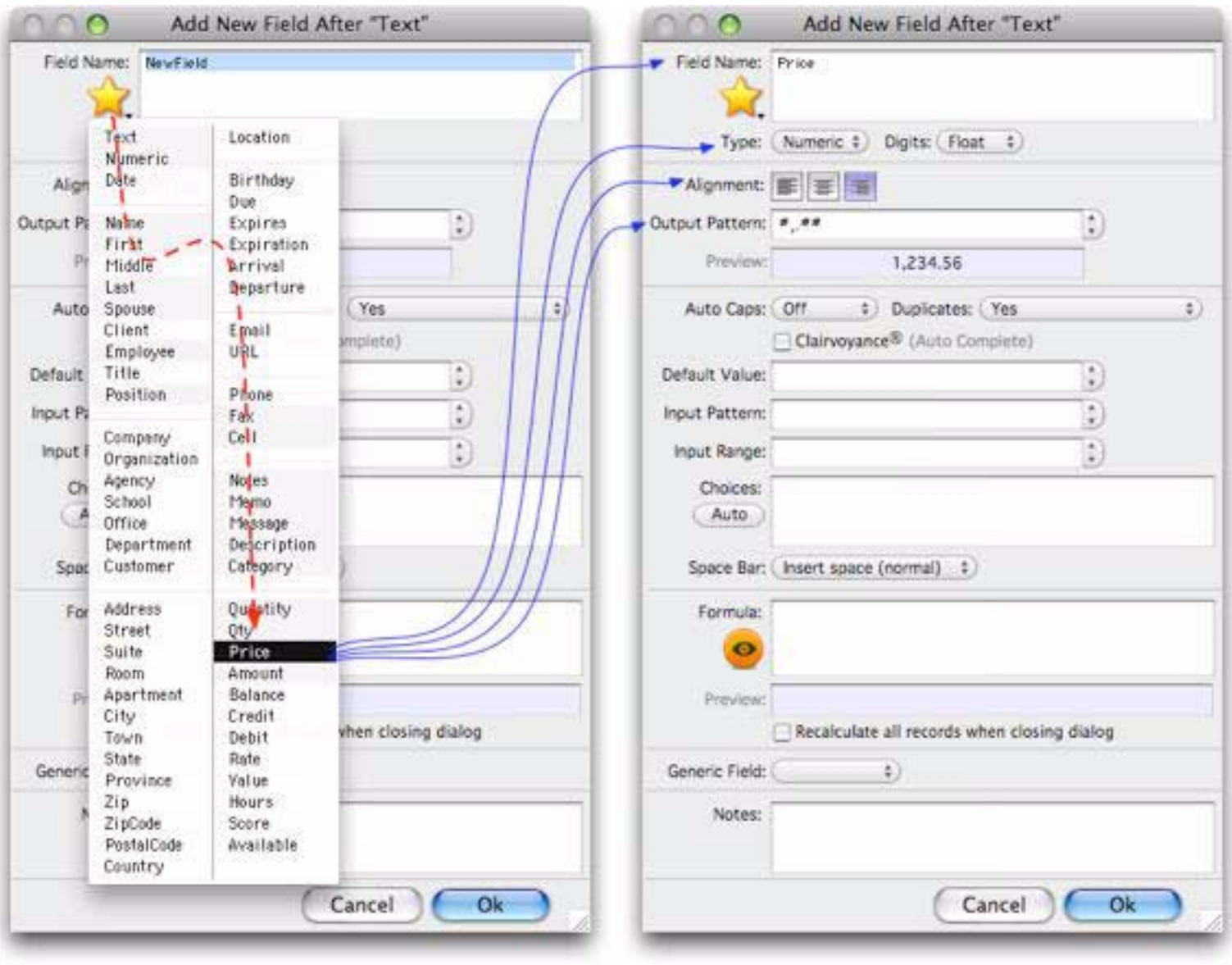
Note: The formula preview, build dialog, and recalculate all records features only work if the database is in spreadsheet mode — they are disabled if the database is in the older procedure mode.

The Generic Field pop-up menu allows you to associate this field with a generic field (for example for the VCard wizard). In previous versions this could only be done with the Generic Field wizard, now it can be done directly from the Field Properties dialog. See “‘Generic” Fields” on page 230 of the *Panorama Handbook*.

The Notes option is just that — a place for your notes about this field.

The Quick Fields menu

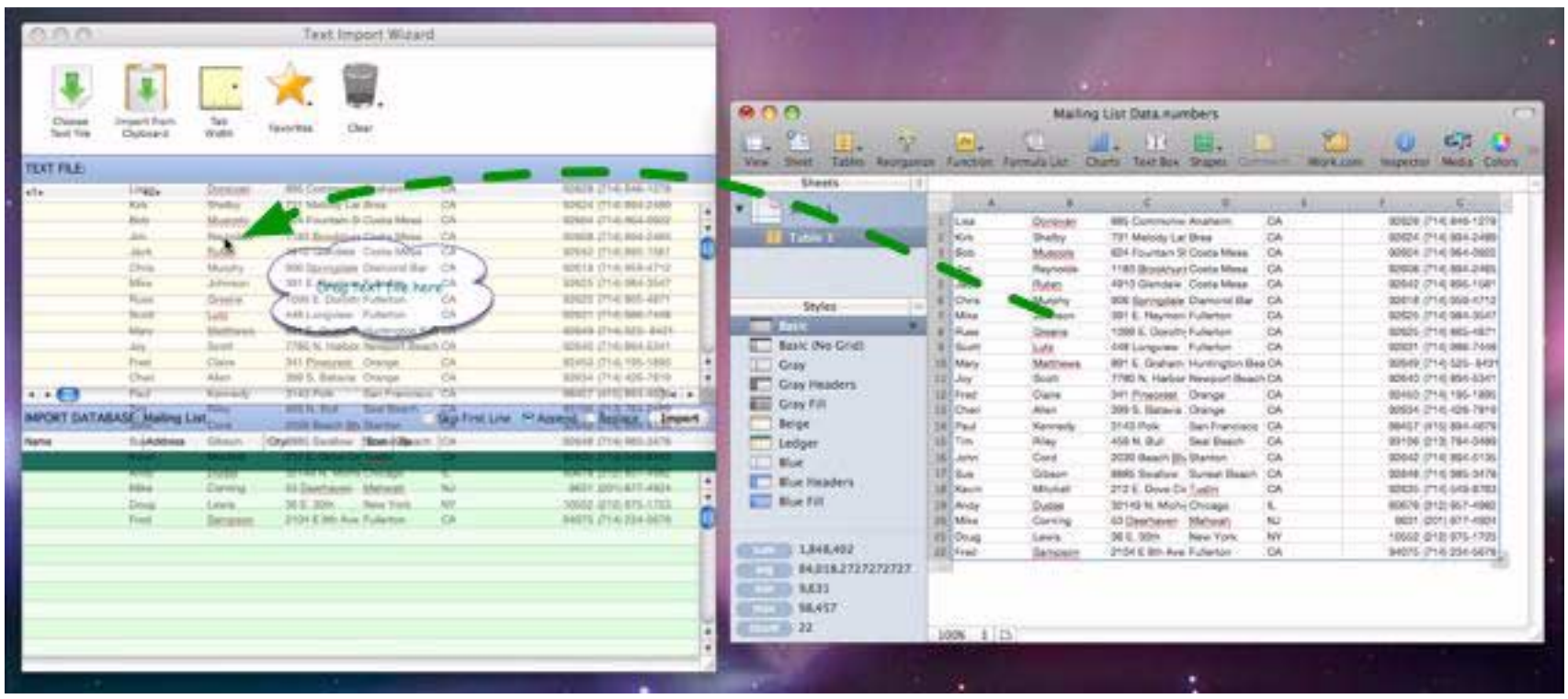
Click on the yellow star at the top of the dialog to see the Quick Fields menu. This menu lists several dozen common field names, grouped into categories.



When you choose an item from this menu Panorama fills in all of the attributes for the field, including the name, type, patterns, formulas, generic options, etc. It's all done for you with a single mouse click. This is especially handy when creating new fields. You can use the new field as-is, or customize it for your own purposes. See “Adding New Fields” on page 198 of the *Panorama Handbook*.

Text Import Wizard

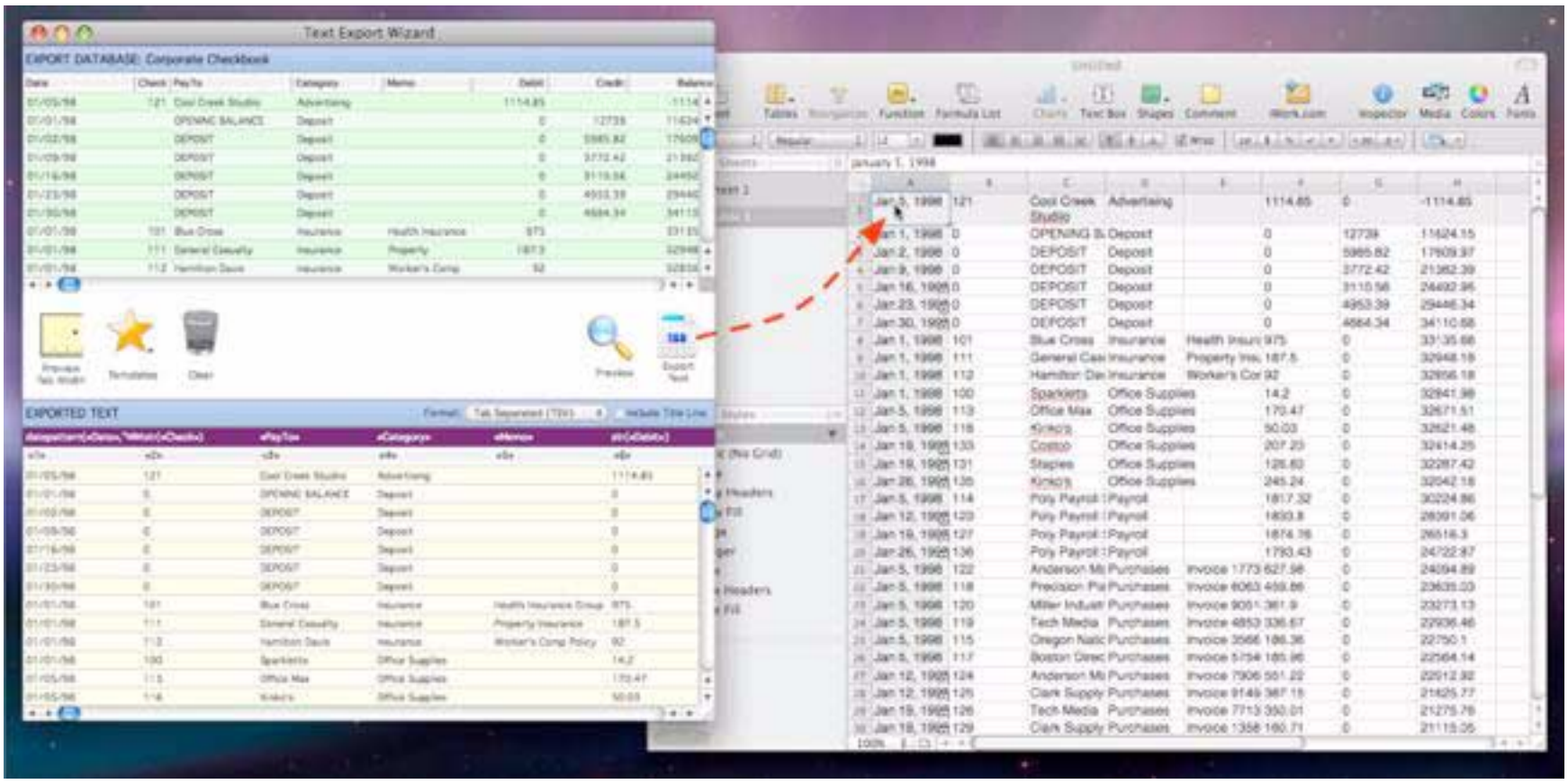
The user interface of the text import wizard has been reworked to make it much more visual and intuitive. It also allows you drag data directly from a spreadsheet, word processor or text editor into Panorama (you can drag onto the Text Import Wizard or even drag data directly onto the data sheet).



To learn more about this revised wizard see "Using the Text Import Wizard" on page 94 of the *Panorama Handbook*.

Text Export Wizard

Like the import wizard, this wizard has been redesigned to make it more visual and intuitive. It also allows you to drag data directly to a spreadsheet, word processor or text editor.



See "Exporting with the Text Export Wizard" on page 109 of the *Panorama Handbook* to learn more.

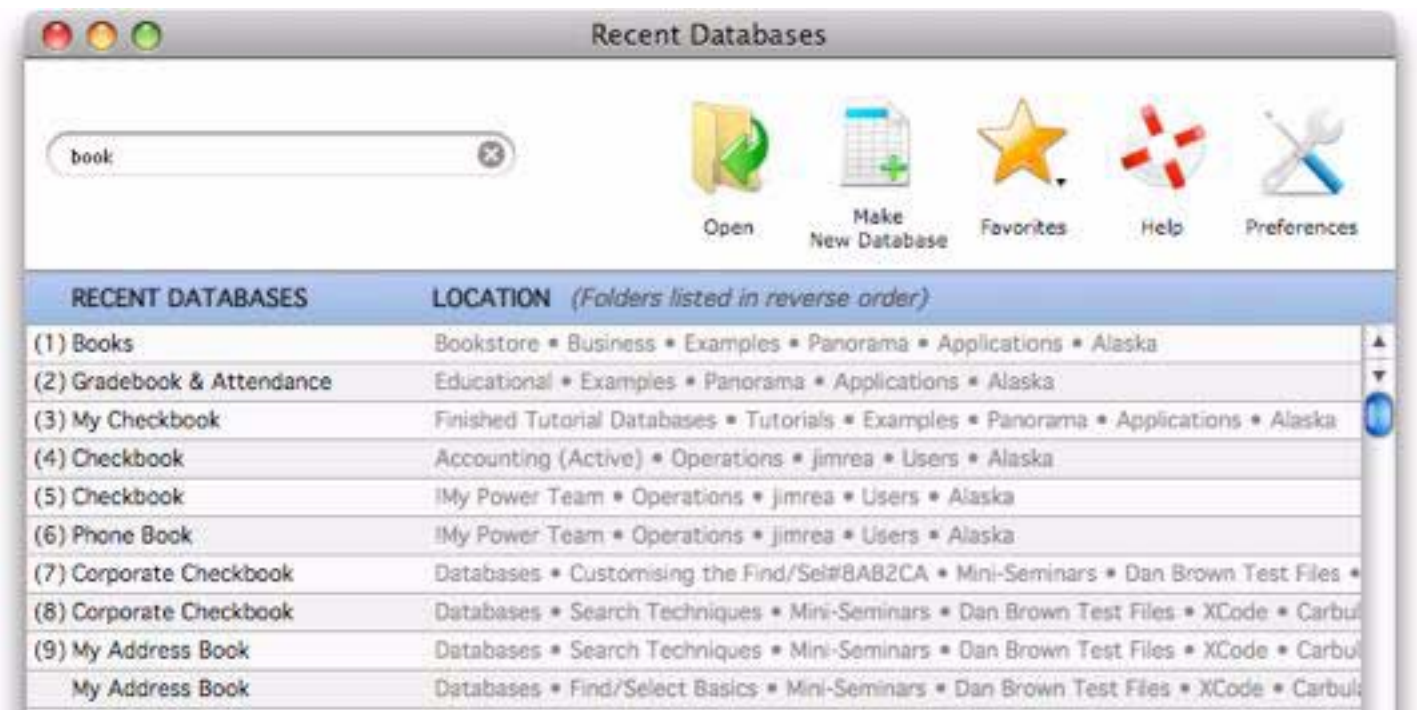
New Database Wizard

The New Database wizard is where most new users get their first impression of Panorama, so we decided it needed a complete makeover.

For complete details on this new wizard see "Using the New Database Wizard" on page 50 of the *Panorama Handbook*.

Recent Databases Wizard

This wizard also got a significant makeover (see "The Recent Databases Wizard" on page 44 of the *Panorama Handbook*).



Favorite Databases Wizard

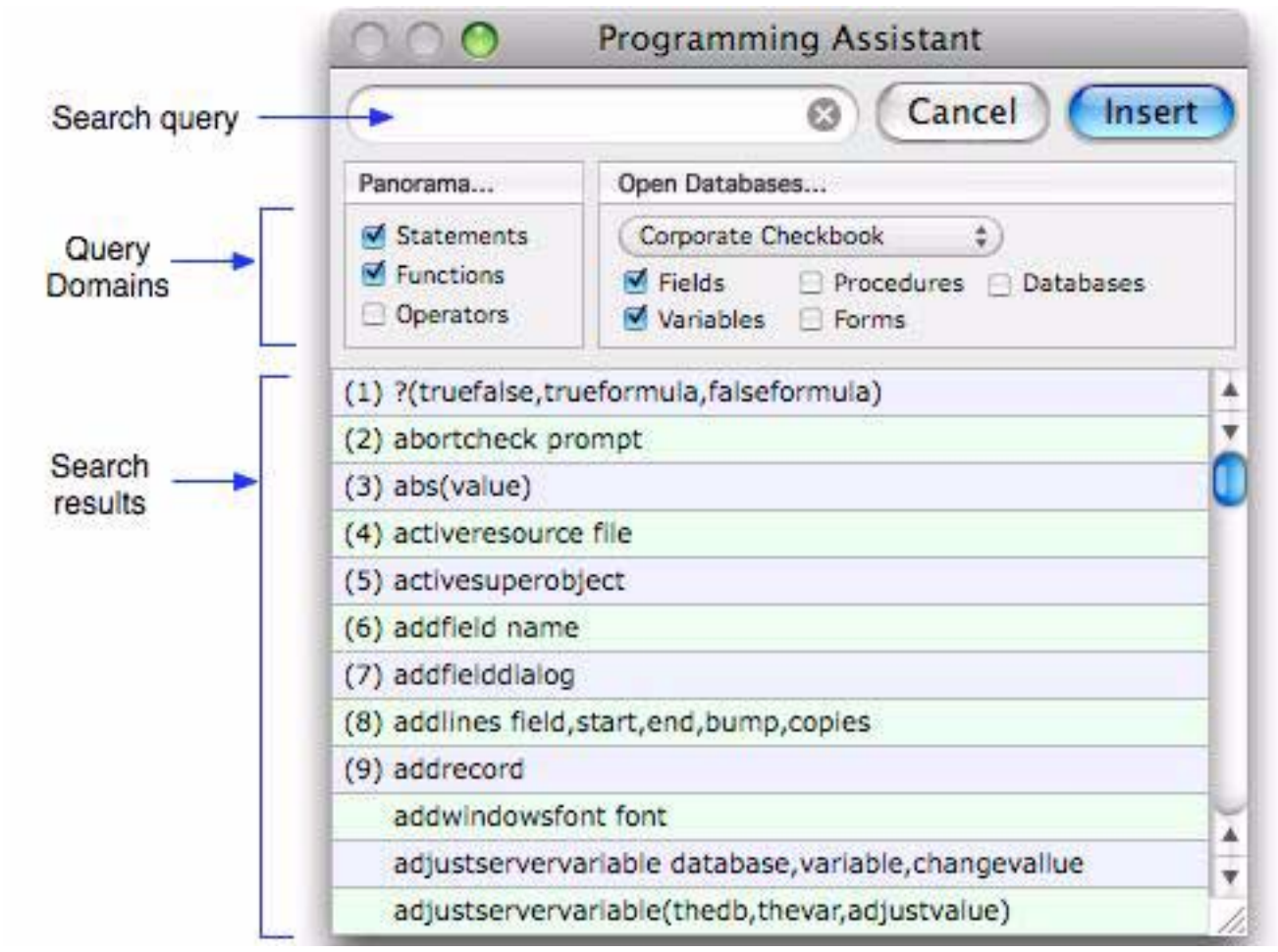
This wizard has been removed, and a favorites feature added to the Recent Databases wizard. See "The Favorites Menu" on page 45 of the *Panorama Handbook*.

Programming Helpers

There are two big improvements to Panorama's procedure editor: the **Programming Assistant** dialog and **Programming Context Menu** (right click).

Programming Assistant Dialog

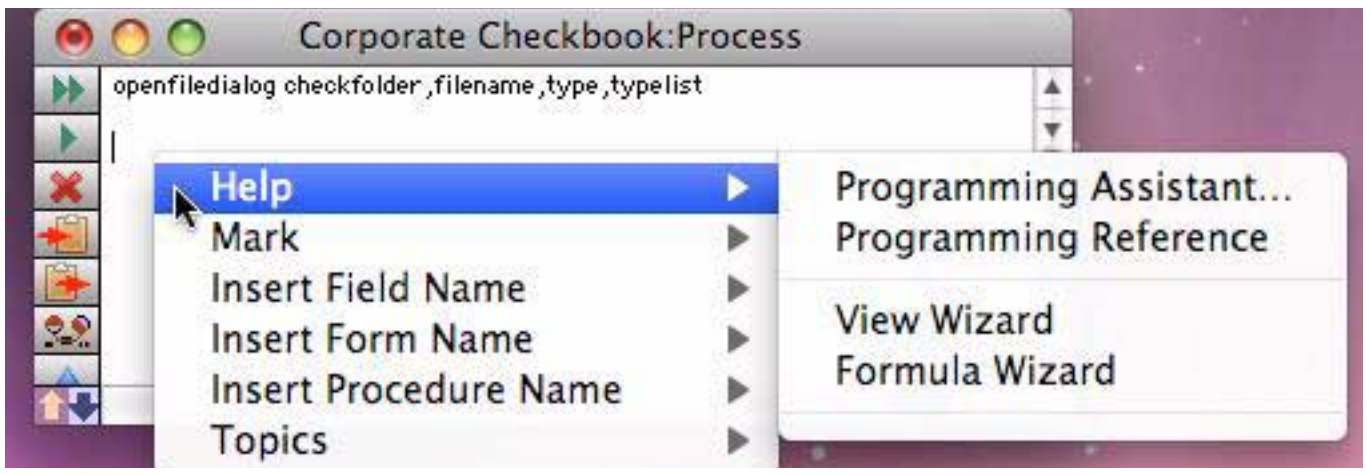
The **Programming Assistant** dialog makes it easy to type in any statement, function, field, variable or other programming element. You only need to type in the first few characters and the assistant will fill in the rest. Within a procedure you can open the assistant from the **Edit** menu, or by pressing **Command-Quote**, or by right clicking on the procedure and choosing **Programming Assistant** from the **Help** submenu. No matter how you open it, the dialog opens and displays a list of items that can be inserted into the procedure.



To learn more about this dialog see "The Programming Assistant Dialog" on page 225 of *Formulas & Programming*.

Programming Context Menu (Right Click)

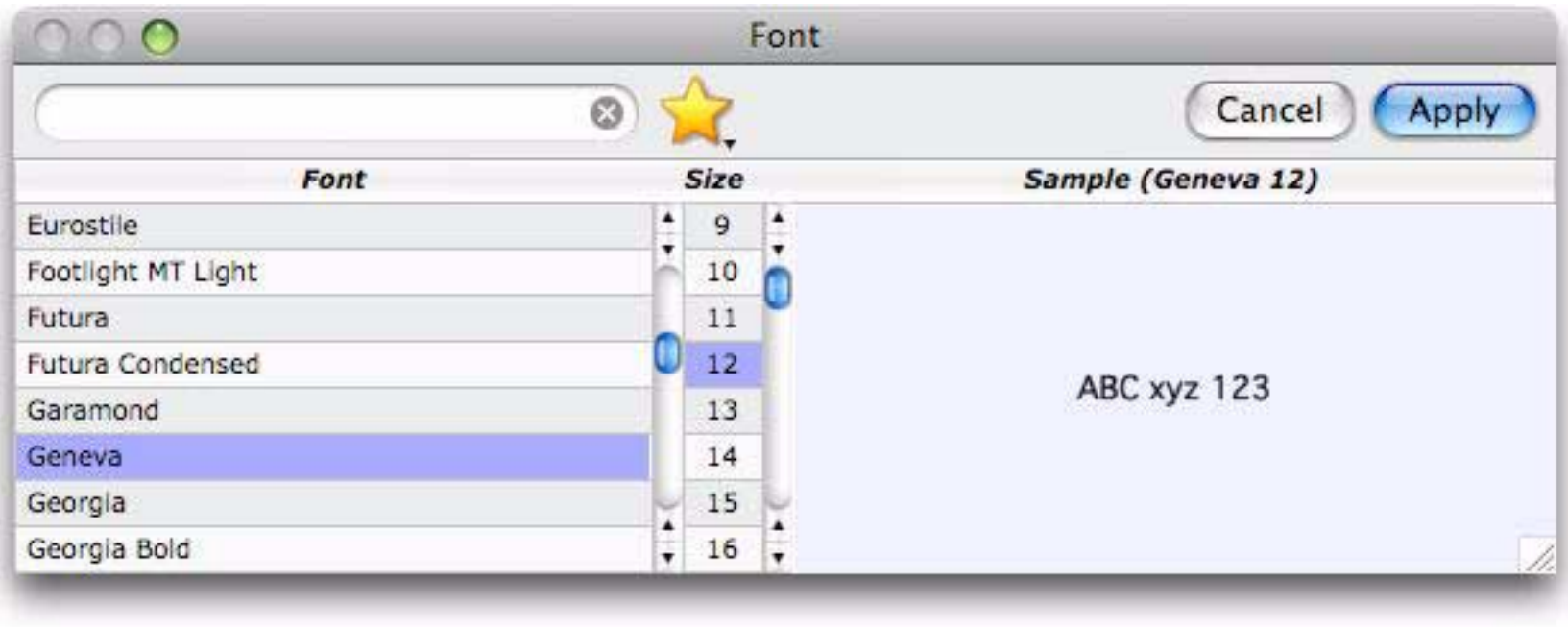
When you're editing a procedure you can right click in the editing window to activate a special menu that is chock full of programming goodies. The exact contents of the menu will vary depending on the database and the current selection, but it will generally look something like this:



This menu has six sub-menus, to learn all about them see "The Programming Context Menu" on page 230 of *Formulas & Programming*.

Data Sheet Font Dialog

The **Font** and **Size** submenus, which used to be in a separate **Text** menu, have been removed. Now there is a **Font...** command in the **Setup** menu. You can search for fonts, and save your favorite fonts for easy recall.



Data Sheet Goto Field Dialog

The previous **Fields** menu, which used to allow you to jump to a field, has now been changed into a dialog. Choose **Goto Field** from the **Fields** menu (see “Moving from Field to Field” on page 267 of the *Panorama Handbook*).

Automatic Field Width

Panorama allows you to adjust column widths in the data sheet simply by dragging left or right on the column heading. Panorama 6 adds new commands for automatically calculating the field width based on the data in the field. This is especially useful after you've imported new data or made major changes to a field with a formula or other manipulation. See “Automatically Setting the Field Width” on page 199 of the *Panorama Handbook* for details on this feature.

Version 5.5.0

This version was released in May 2007 for Mac OS X 10.3 and above. This is the first version of Panorama to support the Panorama Enterprise Server for database sharing and web publishing. Here are some of the highlights of this release.

Panorama Enterprise Server (Database Sharing and Web Publishing)

Panorama 5.5 is the first version of Panorama that works with the Panorama Enterprise Server. This server can be used to share databases between multiple users across a network and to publish Panorama databases on the web. To learn more see the separate **Panorama Enterprise** PDF documentation.

New and Upgraded Wizards

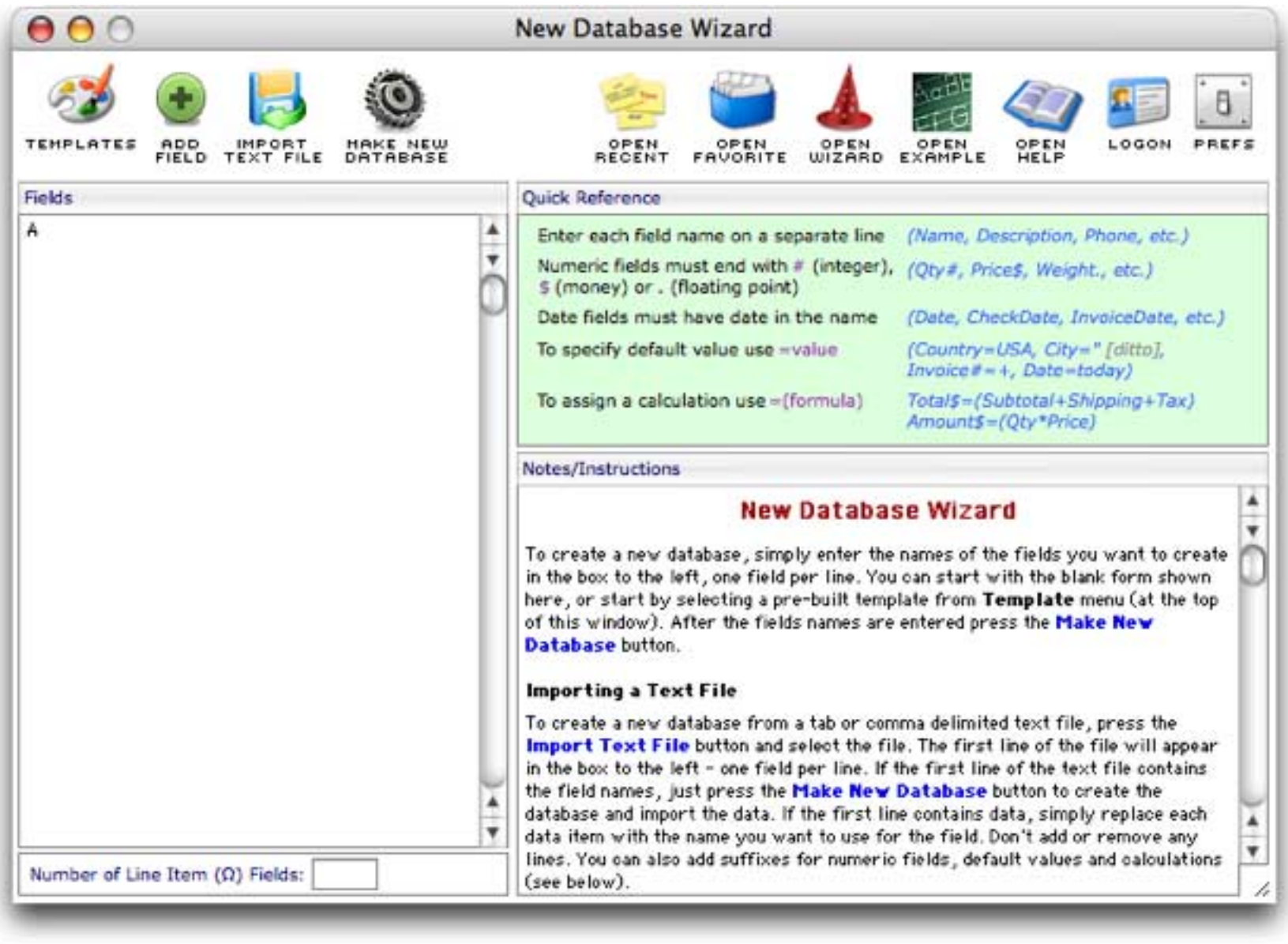
Panorama 5.5 contains almost twenty new and revised wizards.

Primary Wizards

These wizards appear in the main **Wizard** menu.

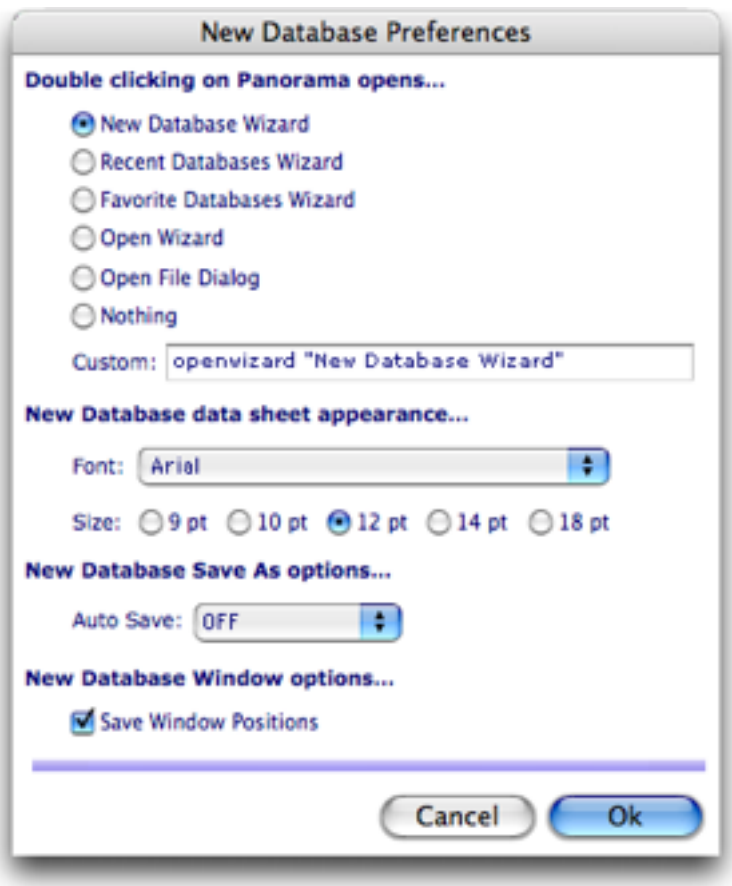
New Database Wizard

The **New Database Wizard** has been enhanced and now opens by default when you double click on Panorama (though this can be changed with the **Preferences** dialog). It can now be opened by choosing **New File** from the **File** menu in addition to being listed in the **Wizard** menu.



The icons in the top left are used to build new databases (see also for details). Most of the icons on the right serve as shortcuts for opening commonly used wizards. The **Logon** icon duplicates the same item in the **File** menu (see the optional **Panorama Security Handbook**).

The final icon is **Prefs**. Clicking it opens this dialog.



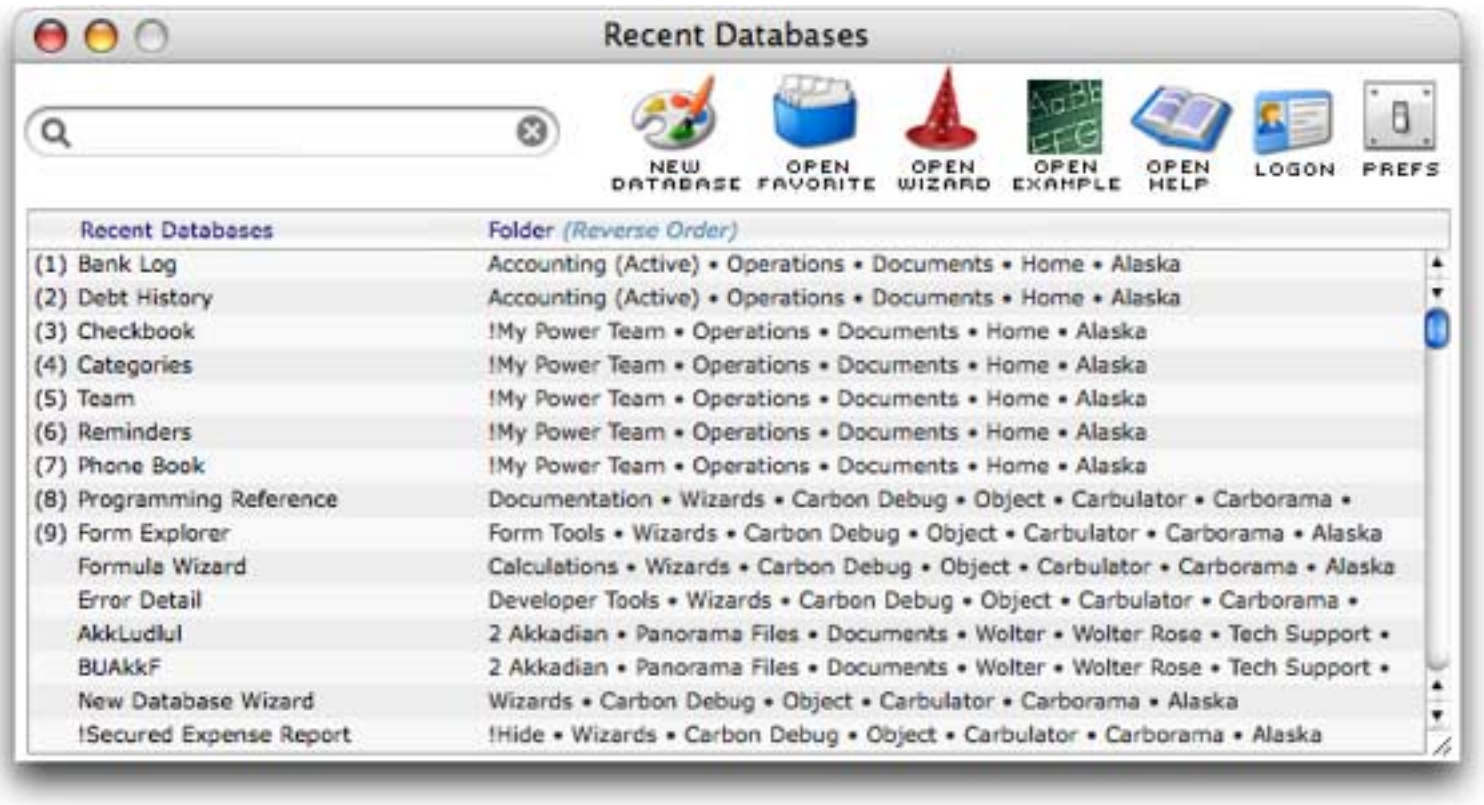
The top section of this dialog specifies what action Panorama should take when you double click on the Panorama icon (these defaults do not apply when you double click on a database file to open Panorama.

Choice	Notes
New Database Wizard	Open named wizard
Recent Databases Wizard	
Favorite Databases Wizard	
Open Wizard	
Open File Dialog	This was the default in earlier versions of Panorama
Nothing	File and Wizard menus will appear
Custom	Type in any procedure you want

The rest of the options in this dialog specify various default options for the new databases you create. For a complete description of the **New Database Wizard** see “Using the New Database Wizard” on page 50 of the *Panorama Handbook*.

Recent Databases Wizard

This wizard makes it easy to re-open recently opened databases. It can be opened by choosing **Recent Databases** from the **File** menu in addition to being listed in the **Wizard** menu. The wizard lists the databases that have been opened recently.



To re-open a database simply double click its name on the list. You can also open the first nine items simply by pressing the 1 thru 9 keys on your keyboard. There's no need to press **Return**, **Enter**, or anything else, just press the number and the database will open.

To search for a particular database simply type into the search box at the top of the wizard.

At any time you can press the 1 thru 9 keys to re-open a database. For example type **boo1** to open the **Checkbook** database, **boo2** to launch the **Phone Book** database, etc. You can re-open any previously opened database with just a few keystrokes.

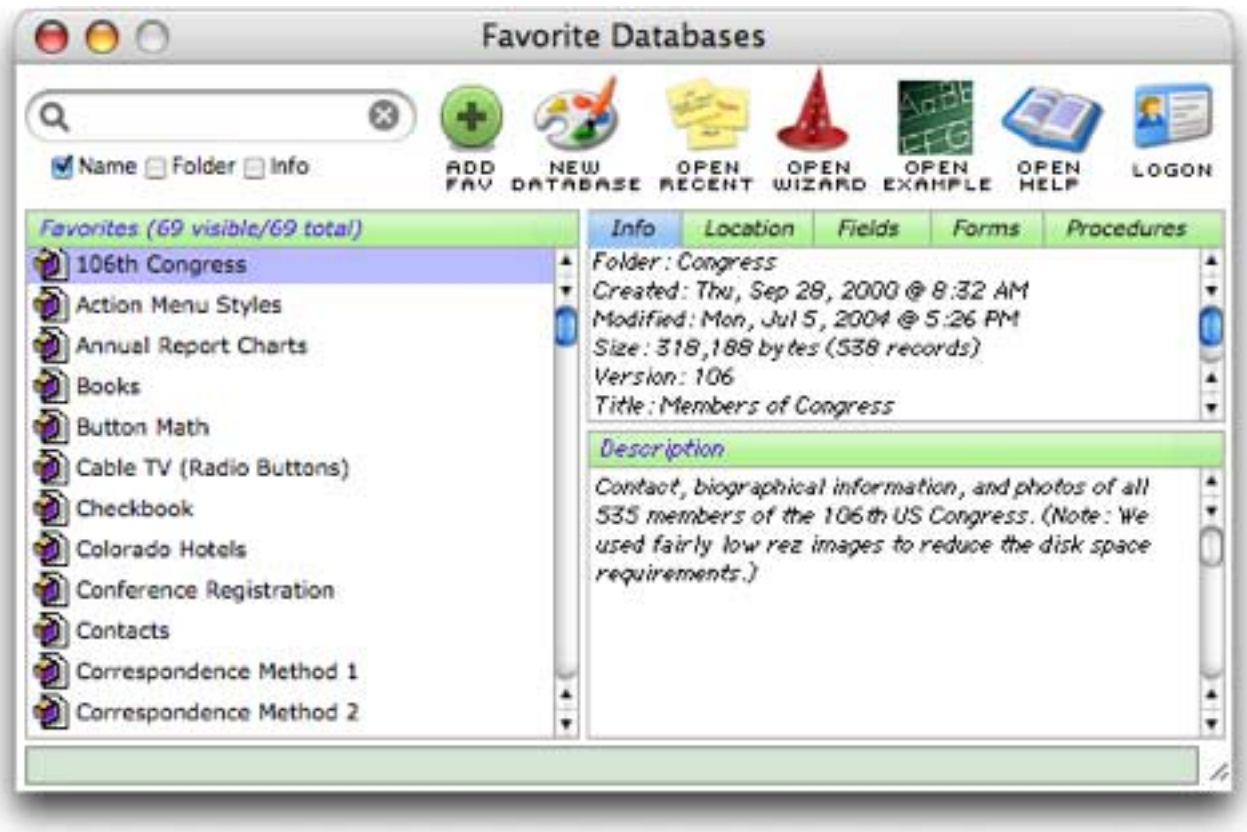
The Prefs dialog allows you to exclude specified folders and files from the list of recent databases. By default wizards are excluded.

If you want to include other folders or files simply list them in the dialog, one per line. You can use the ∂ symbol (Option-D) to specify the current Panorama folder.

With the setting shown above any databases in the Accounting folder will be excluded. For additional information on this wizard see "The Recent Databases Wizard" on page 44 of the *Panorama Handbook*.

Favorite Databases Wizard

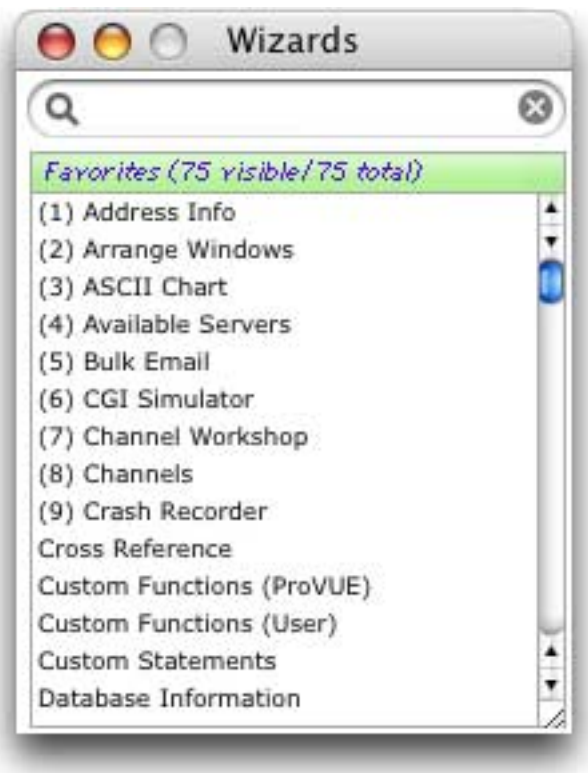
The user interface for this wizard has been slightly updated. You can now use this wizard as the default when you double click on Panorama (see "New Database Wizard" on page 36 of the *Panorama Handbook*).



For a complete description of this wizard see "Advanced Database Opening Techniques" on page 47 of the *Panorama Handbook*.

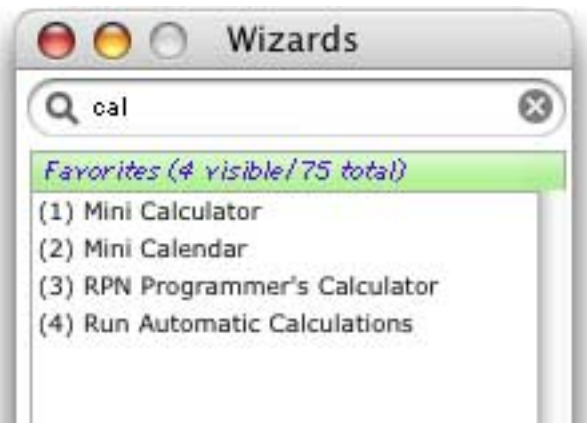
Open Wizard

Panorama now includes 75 wizards spread across a dozen submenus of the **Wizard** menu. The **Open Wizard** wizard provides an alternate method to quickly open any of these 75 wizards. When you open this wizard it initially displays an alphabetical list of all available wizards.



To open a wizard simply double click its name on the list. You can also open the first nine items simply by pressing the 1 thru 9 keys on your keyboard. There's no need to press **Return**, **Enter**, or anything else, just press the number and the wizard will launch.

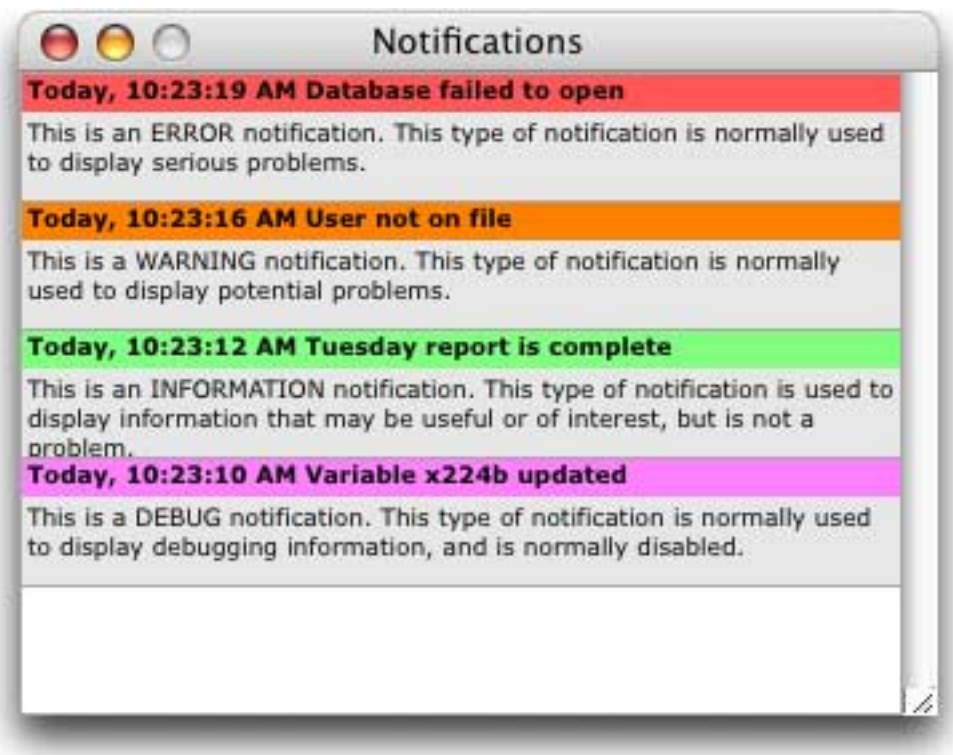
To search for a particular wizard simply type into the search box at the top of the wizard.



At any time you can press the 1 thru 9 keys to launch a wizard. For example type **cal1** to launch the **Mini Calculator**, **cal2** to launch the **Mini Calendar**, etc. You can launch any wizard with just a few keystrokes.

Notification Wizard

The **Notification Wizard** allows Panorama to inform you about a problem or potential problem without interrupting your work. Unlike an alert, the **Notification Wizard** doesn't interrupt your work by requiring you to dismiss the message before continuing — the wizard simply opens in the background and allows you to continue (or you can configure the wizard to use the open source **Growl** notification system, which displays the message and then fades away). You can also review problems that occurred earlier in the session.



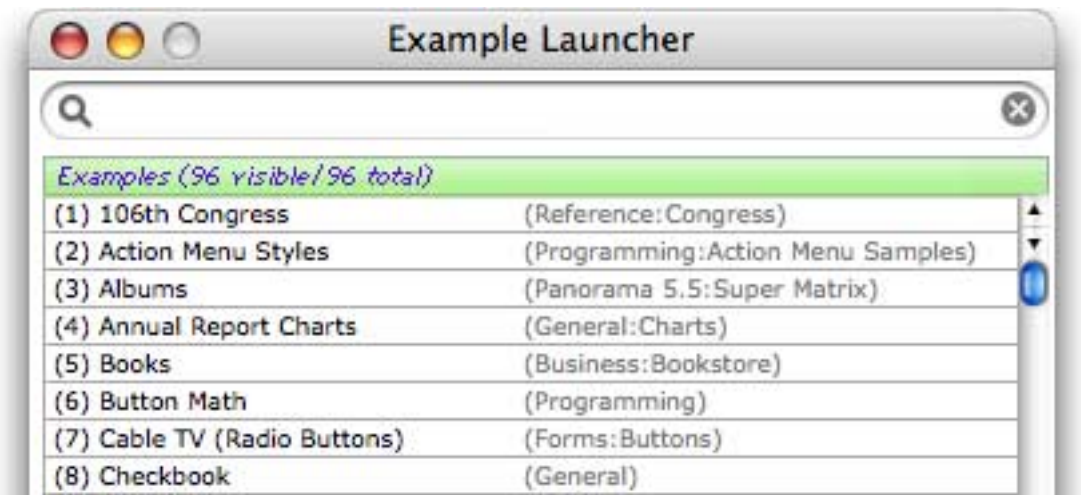
To learn more about the **Notification Wizard** see "Notifications Wizard" on page 14 of *Wizards & Demos*.

Demos Submenu

The wizards in this submenu help you to access the examples and demos included with Panorama.

Example Launcher

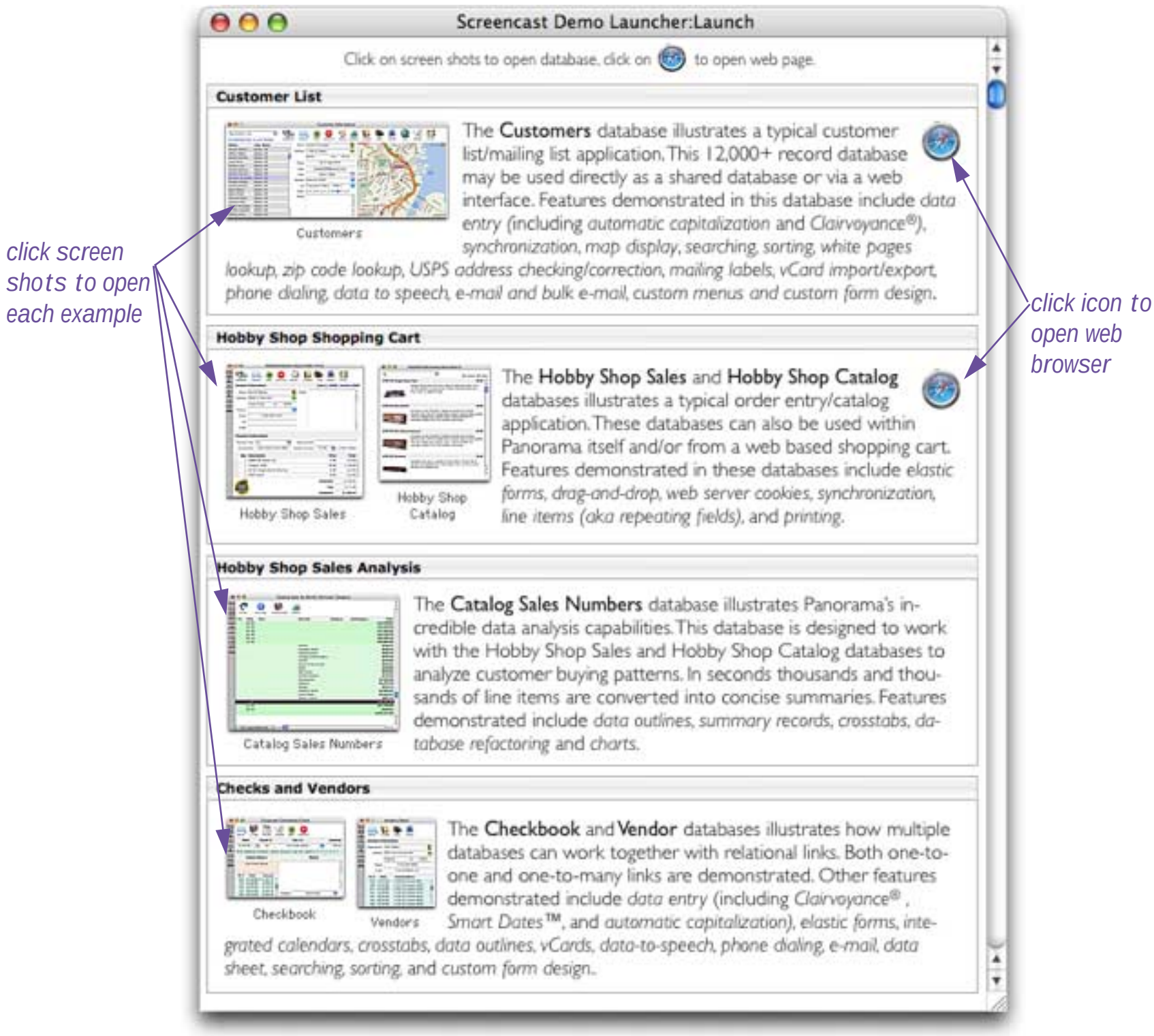
This database lists all of the demos included in Panorama's Example folder.



To open one of these example databases simply double click its name on the list. You can also open the first nine items simply by pressing the 1 thru 9 keys on your keyboard. There's no need to press **Return**, **Enter**, or anything else, just press the number and the database will open. You can also use the search box at the top of the wizard to help locate a particular example. For more information about this wizard see "Example Launcher" on page 25 of *Wizards & Demos*.

Screencast Demos

After watching the Panorama screencasts on the ProVUE web site you may want to play with and examine the databases used in these screencasts. This wizard assists you installing and running these databases.



For more information see "Screencast Demos" on page 26 of *Wizards & Demos*.

Developer Tools Submenu

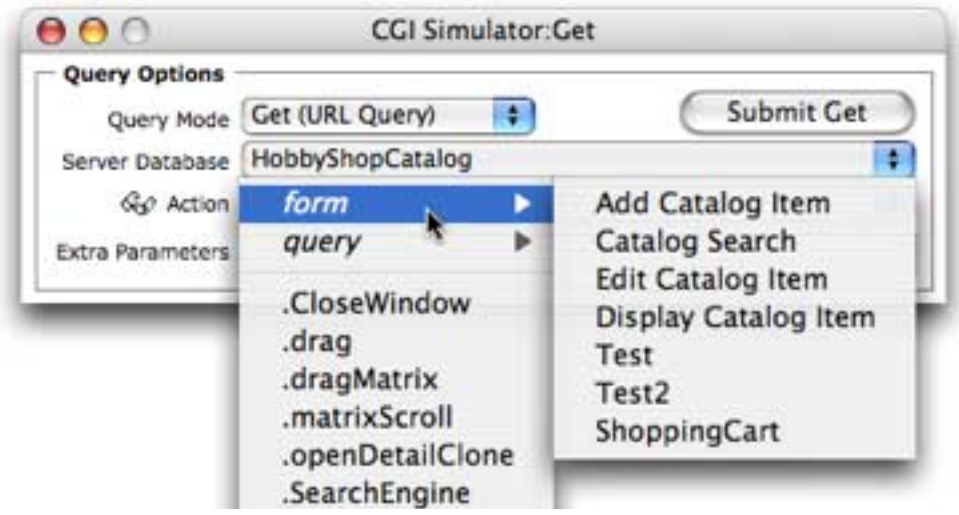
There are five new or revised Developer Tool wizards: **Crash Recorder**, **CGI Simulator**, **Error Detail**, **TTY** and **Web Form Converter**.

Crash Recorder

The **Crash Recorder** wizard can be used to help ProVUE engineers track down problems in Panorama itself. To learn more about this wizard see "Crash Recorder" on page 29 of *Wizards & Demos*.

CGI Simulator

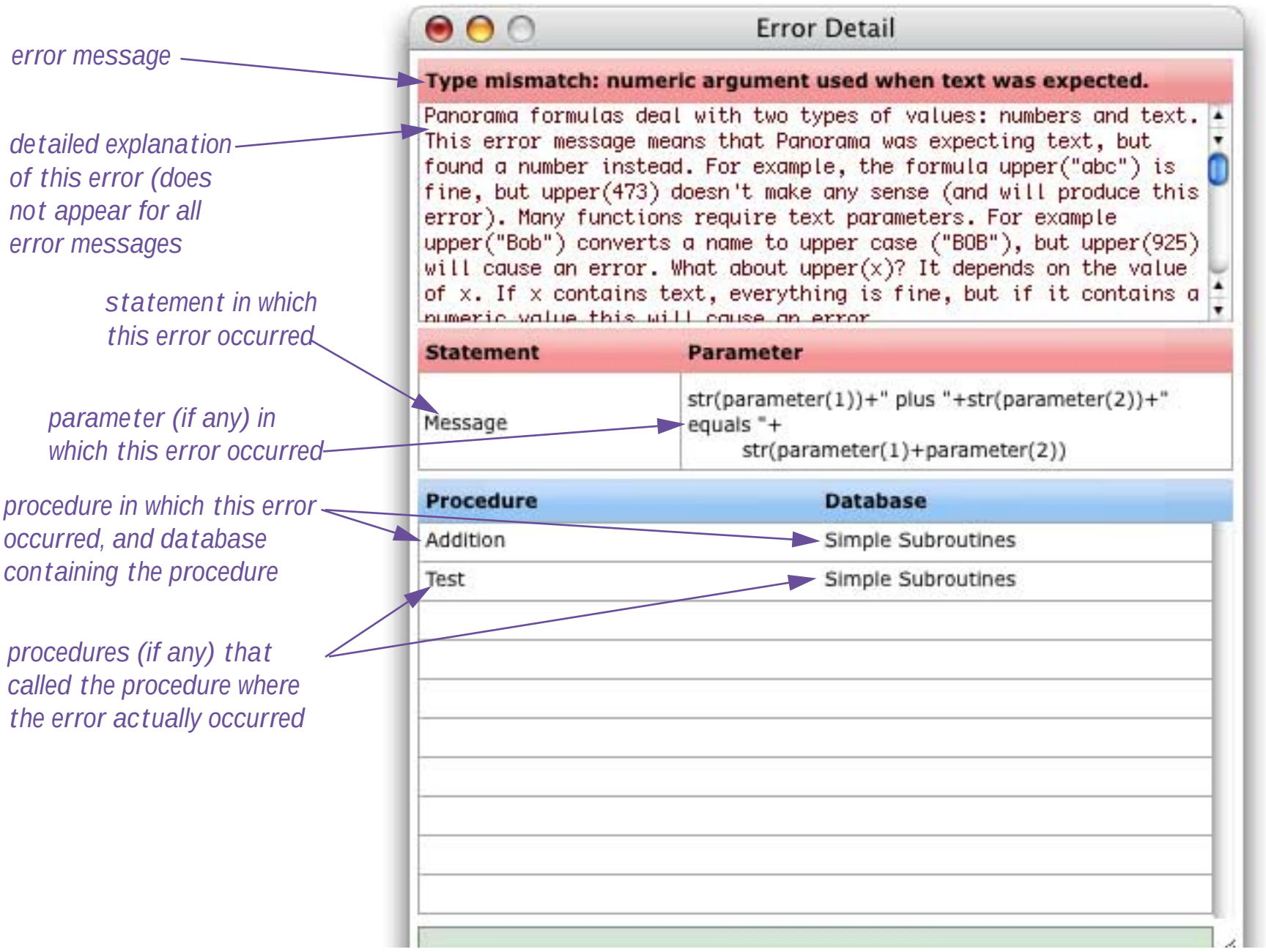
The **CGI Simulator** wizard is designed to help debug web procedures for use with the Panorama Enterprise server. This wizard simulates the operation of your web browser and server, allowing you to test procedures in Panorama's normal environment.



For more information about this wizard see Chapter 8 of the **Panorama Enterprise Handbook**.

Error Detail Wizard

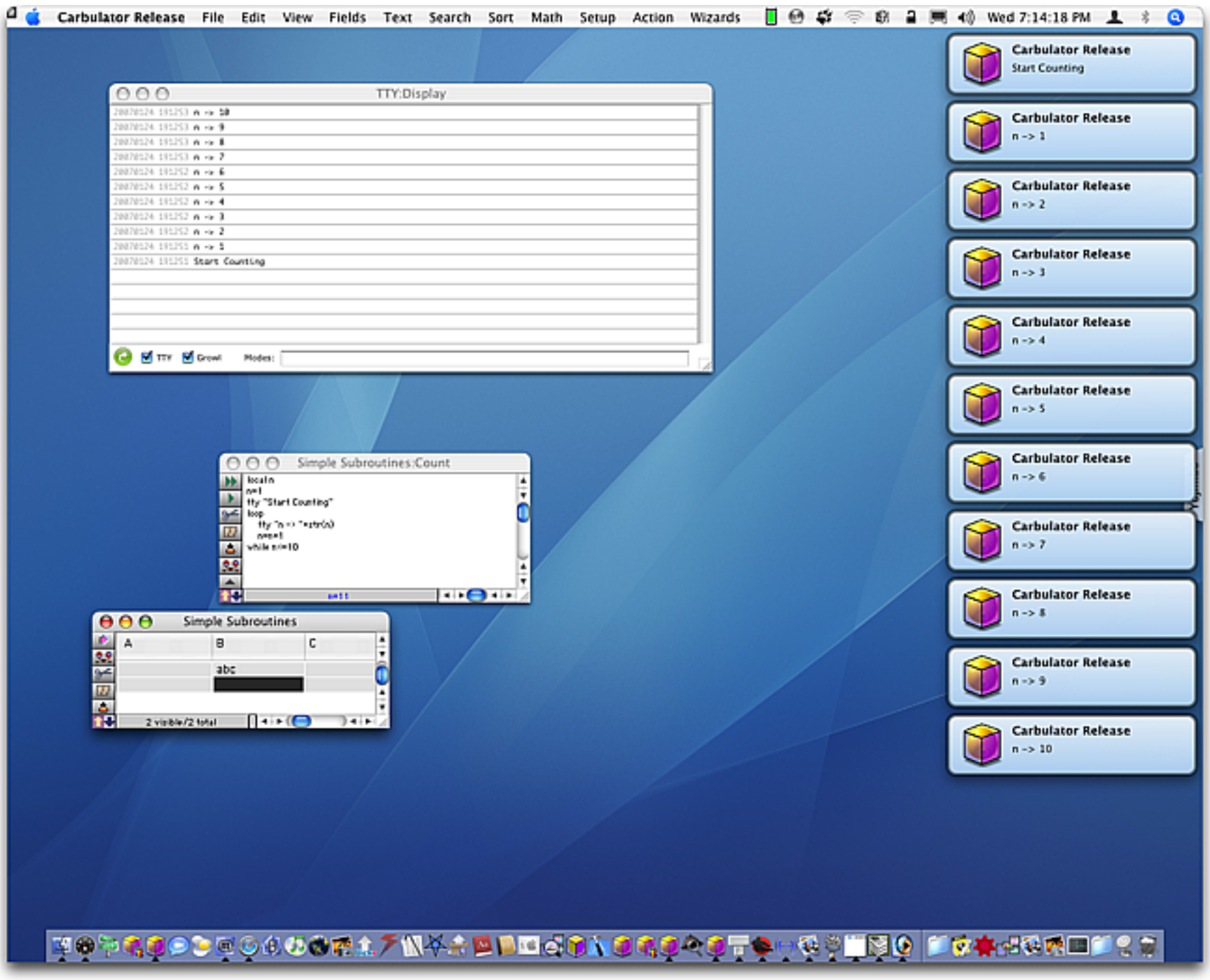
The **Error Detail** wizard can help track down the source of an error in a procedure or formula. Once it is enabled this wizard will display extensive information when a programming error occurs.



For more information see "Error Detail Wizard" on page 325 of *Formulas & Programming*.

TTY Wizard (Virtual Teletype)

Back in the dark ages of computer history (before 1980) computers generally didn't have fancy debugging systems, and the most common method for finding bugs was inserting "print" statements in the code to type messages on the teletype printer attached to the computer. By looking at the output of the print statements the programmer could monitor the operation of the program in question. Though we now have many other options for debugging, sometimes simply "printing" can still be the most effective way to monitor program operation. Of course most of us no longer have actual teletypes connected to our computers any more, so Panorama now includes a virtual teletype — the **TTY** wizard. (Back in the day *TTY* was frequently used as an abbreviation for *teletype*.)



To use this wizard you need to insert one or more `tty` statements in your code. The `tty` statement is kind of like the `message` statement, but instead of displaying an alert it sends the message to the TTY wizard. (The message can also optionally be displayed in a floating **Growl** alert as shown above.) To learn more about this wizard see "Debugging with the TTY (Virtual Teletype) Wizard" on page 330 of *Formulas & Programming*.

Web Form Converter

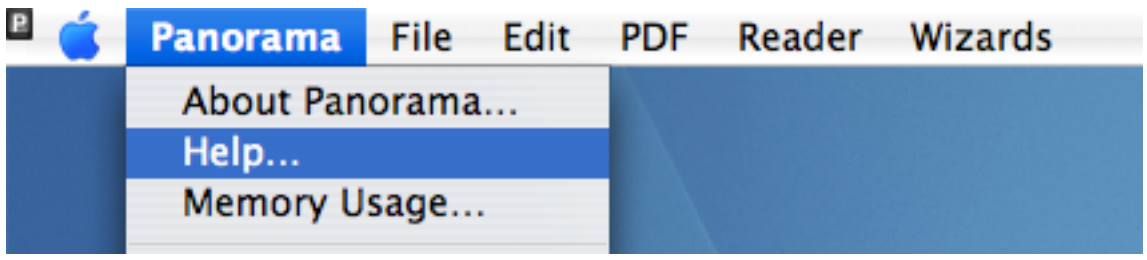
Whenever you render a Panorama form to HTML Panorama will automatically open the **Web Form Converter** wizard. To learn more about this wizard and about converting Panorama forms to HTML see Chapter 6 of the *Panorama Enterprise Handbook*.

Documentation Submenu

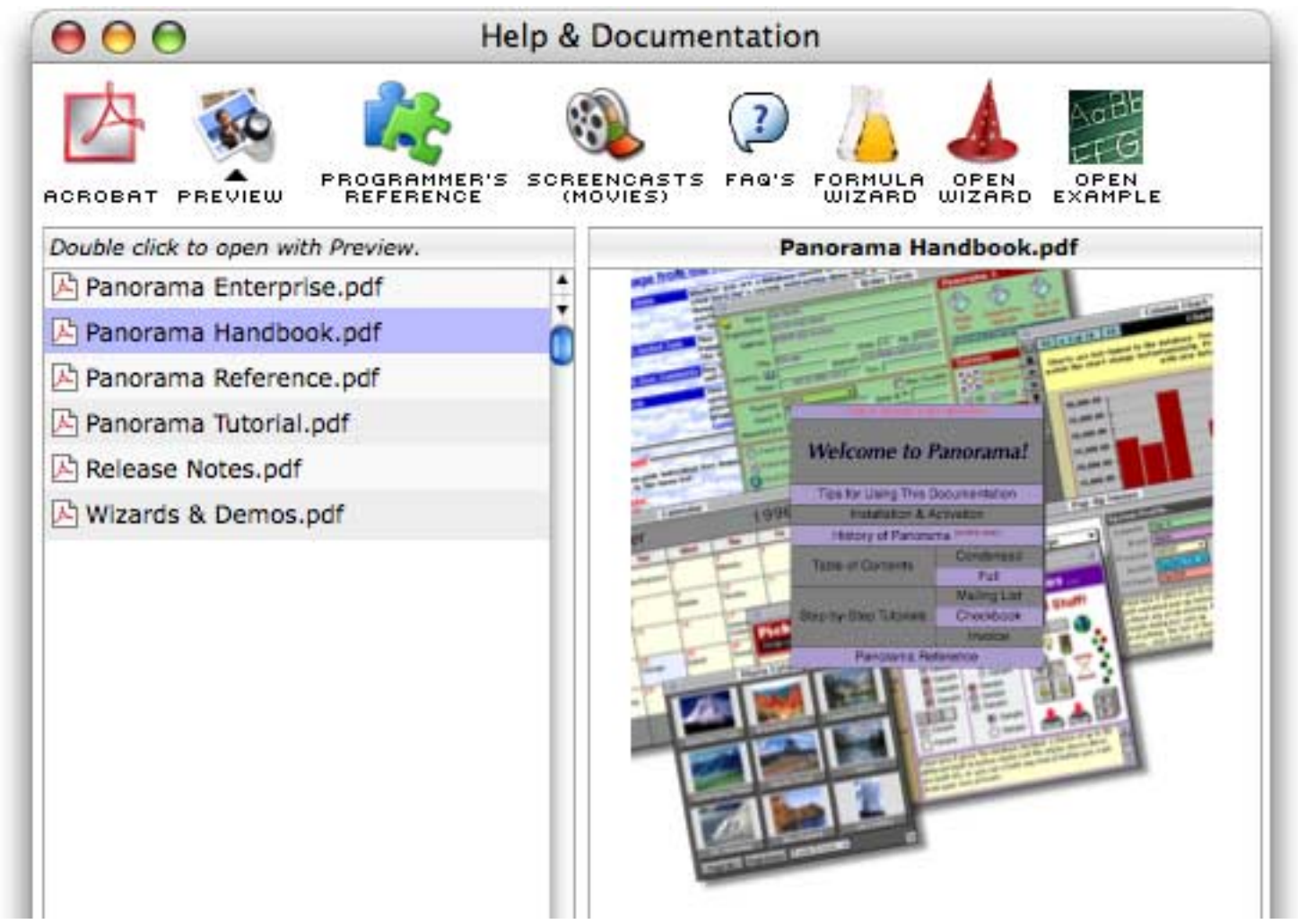
In this submenu the **Panorama Movies** wizard has been replaced by the **Screencasts** wizard, and the **PDF Documentation** wizard has been replaced by the **Help & Documentation** wizard. The **Help & Documentation** wizard is the new central command post for all Panorama documentation. This wizard is now opened when you choose Help from the Panorama menu (finally replacing the old Panorama II era help file).

Help & Documentation

This wizard makes it easy to access all of the documentation that comes with Panorama. You can open this from the **Documentation** submenu of the **Wizard** menu or simply by choosing **Help** from the Panorama menu.



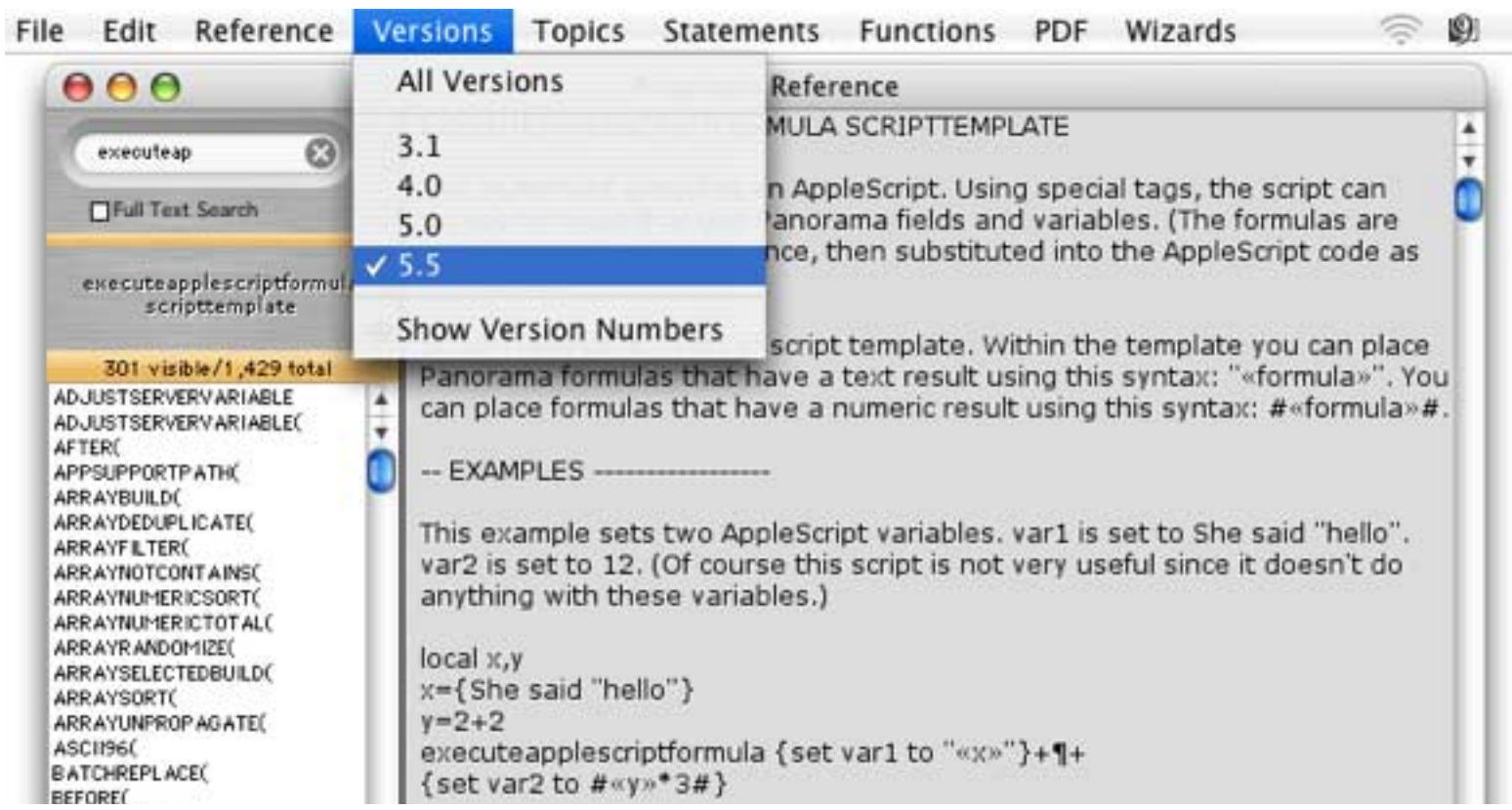
This wizard gives you instant access to all of the PDF documentation available for Panorama (it will even download PDF files for you if they are not already installed on your system), to the Programmer's Reference wizard, to screencasts, FAQ's — all the documentation available for Panorama at your fingertips. See "Help & Documentation" on page 45 of *Wizards & Demos* for complete information about this wizard.



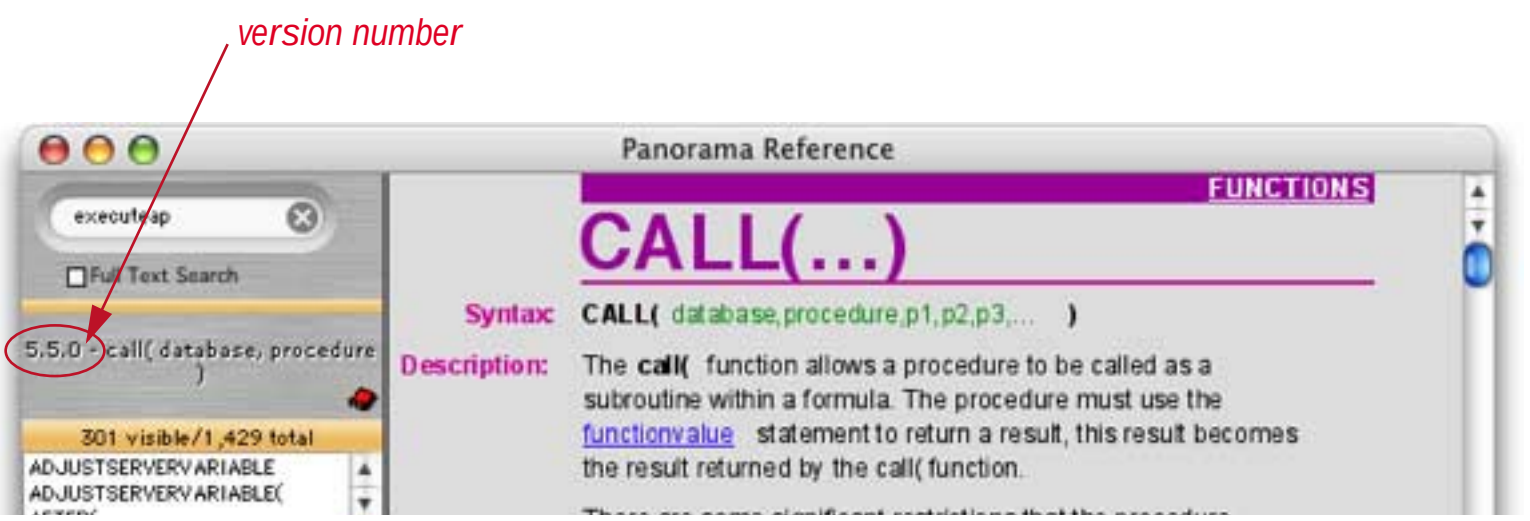
To open a PDF file simply double click on the name on the left or click once on the cover page on the right. The PDF files normally open with Apple's **Preview** program. If you'd like them to open with **Adobe Acrobat Reader** instead simply click on the Acrobat icon above the list. You can switch back and forth between Acrobat and Preview at any time.

Programming Reference Wizard

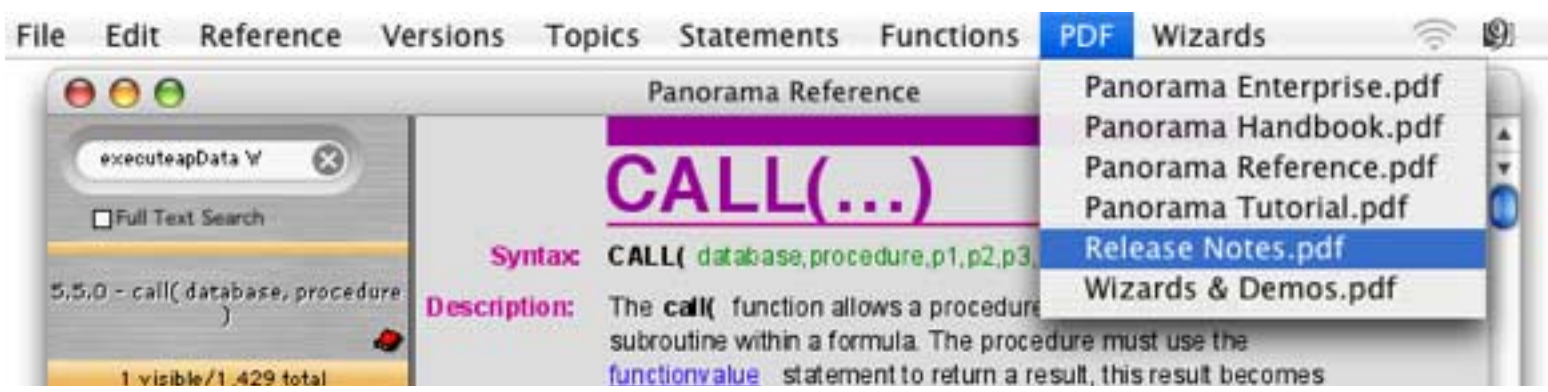
This wizard is the ultimate reference for information on statements and functions used in Panorama programming. This wizard has been updated with several hundred new statements and functions added in Panorama 5.5. To see a list of these new statements and functions choose **5.5** from the **Versions** menu.



A new feature in this wizard is the ability to display the version number when a particular statement or function was added to Panorama. Simply choose **Show Version Numbers** from the **Version** menu.



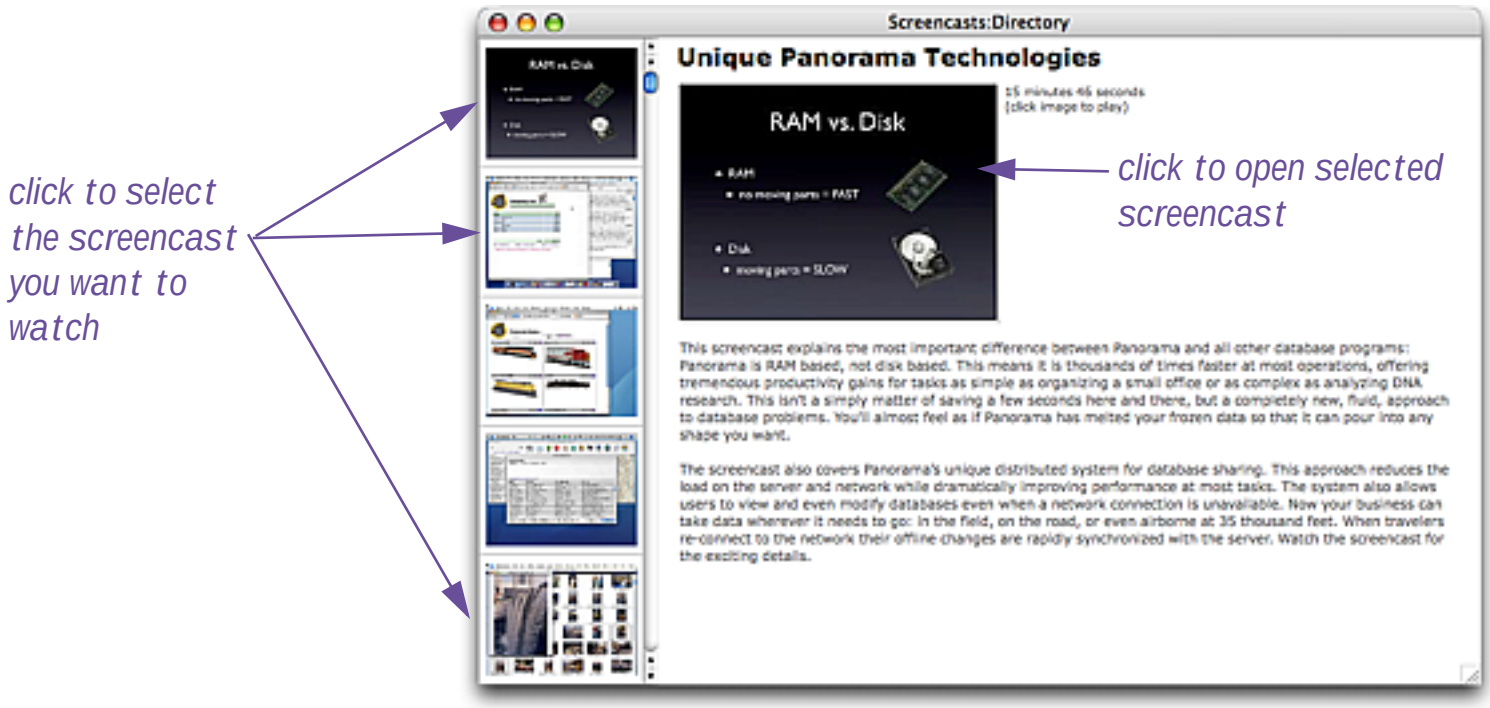
A new **PDF** menu has been added to allow you to quickly access Panorama's PDF documentation from this wizard.



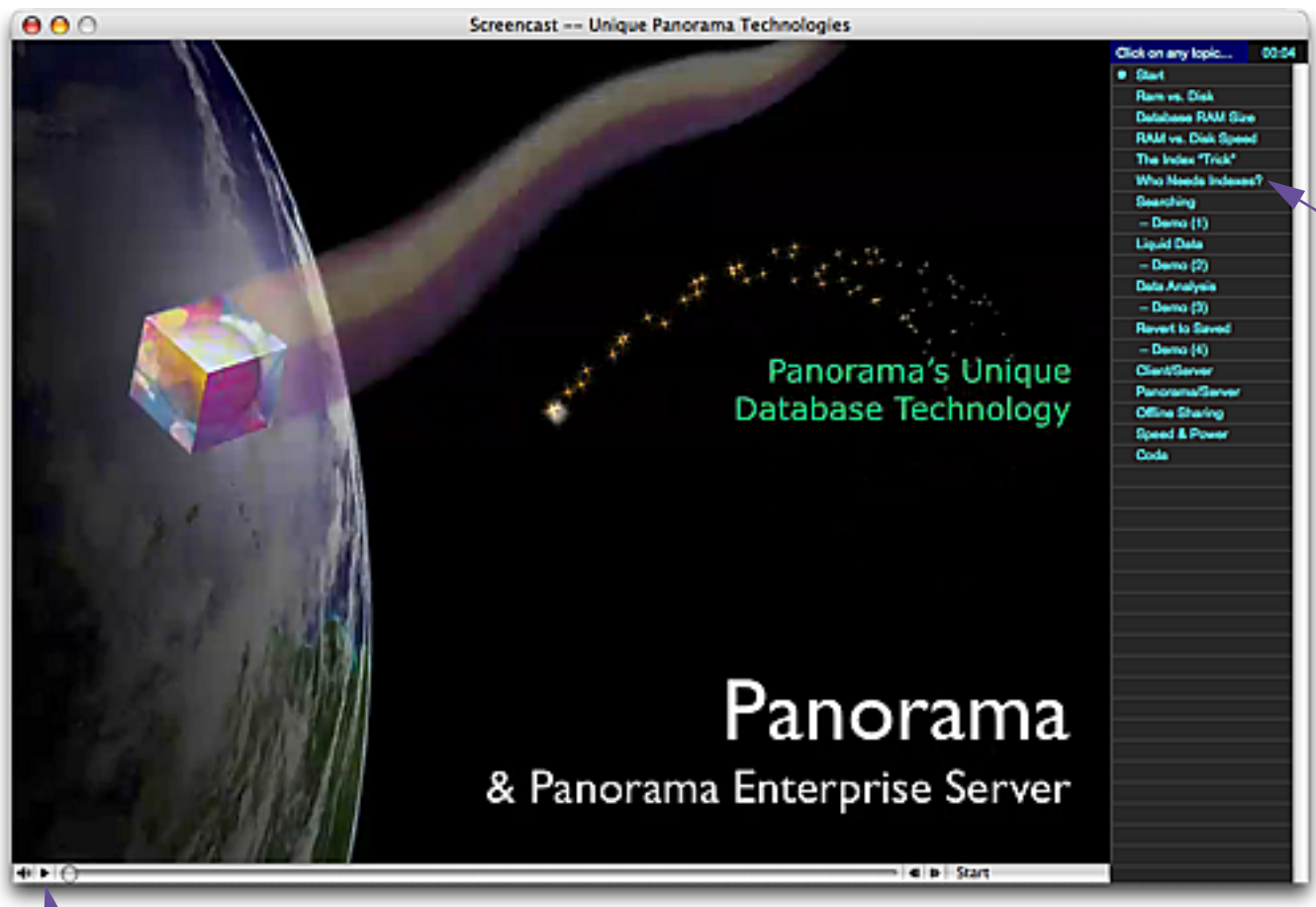
For more information on this wizard see "Programming Reference Wizard" on page 237 of *Formulas & Programming*.

Screencasts

If you've installed Panorama from a CD you had the opportunity to install video screencasts when you install Panorama. (You can also watch these screencasts directly from the ProVUE web site.) If you have installed these screencasts you can use the **Screencast** wizard to view them.



Click on the left to select the screencast you want to watch, then click on the larger version of the screencast to open it.



You can use the chapter list on the right to jump directly to different topics in the movie.

Guided Tour Submenu

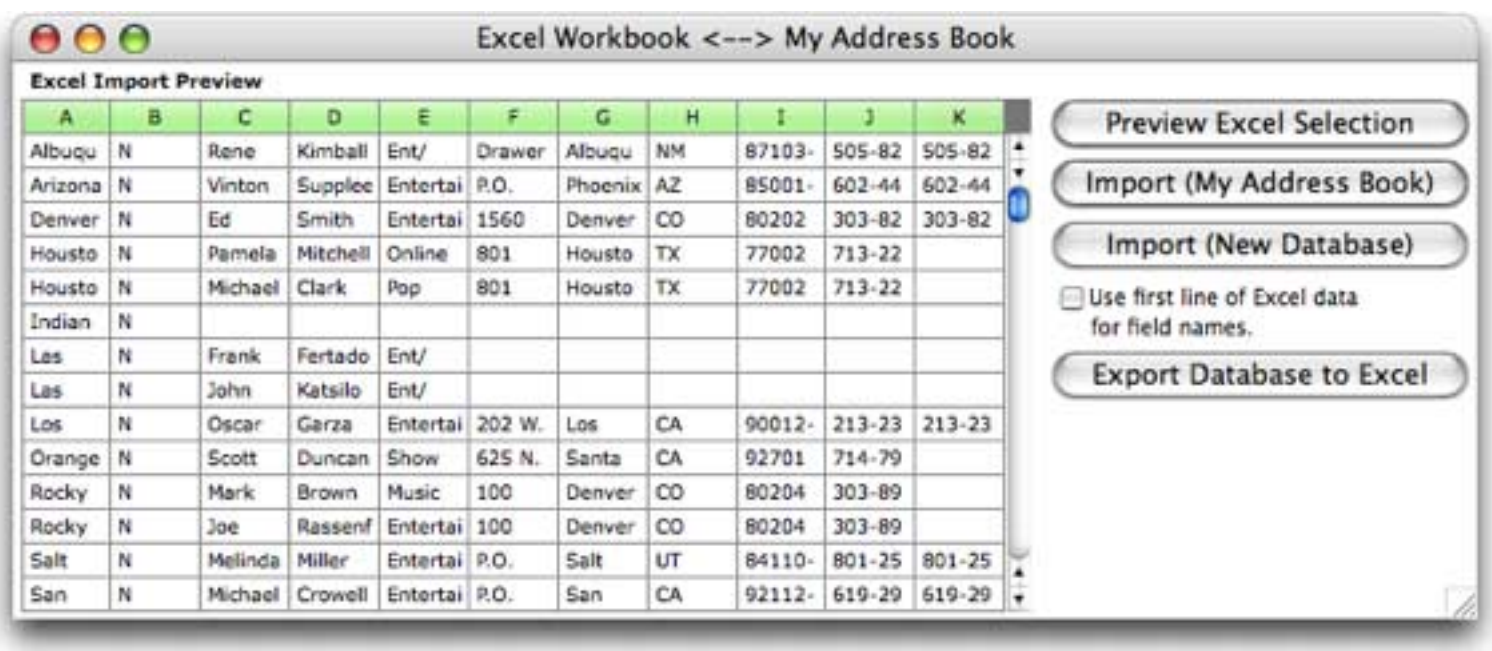
This submenu has been removed from Panorama 5.5.

Import-Export Submenu

This submenu is new in Panorama 5.5. Three older wizards have been moved into this submenu (**Text Export Wizard**, **Text Import Wizard**, and **VCard wizard**) and two new wizards have been added (**Excel Wizard** and **Financial Data Wizard**).

Excel Wizard

This new wizard allows data to be transferred directly from Microsoft Excel to Panorama and from Panorama to Excel.

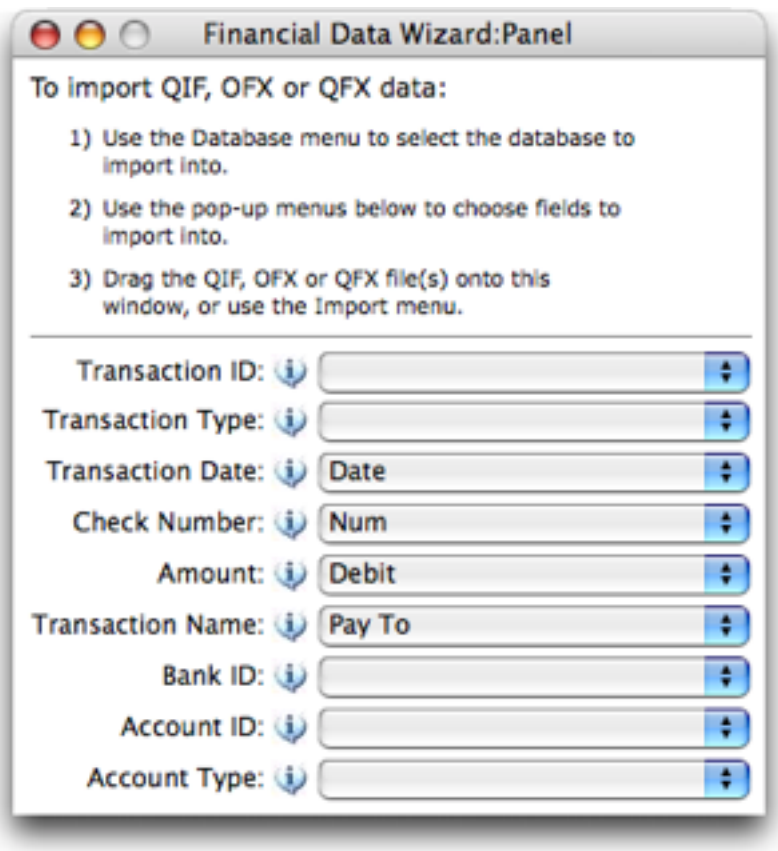


A	B	C	D	E	F	G	H	I	J	K
Albuqu	N	Rene	Kimball	Ent/	Drawer	Albuqu	NM	87103-	505-82	505-82
Arizona	N	Vinton	Supplee	Entertai	P.O.	Phoenix	AZ	85001-	602-44	602-44
Denver	N	Ed	Smith	Entertai	1560	Denver	CO	80202	303-82	303-82
Housto	N	Pamela	Mitchell	Online	801	Housto	TX	77002	713-22	
Housto	N	Michael	Clark	Pop	801	Housto	TX	77002	713-22	
Indian	N									
Las	N	Frank	Fertado	Ent/						
Las	N	John	Katsilo	Ent/						
Los	N	Oscar	Garza	Entertai	202 W.	Los	CA	90012-	213-23	213-23
Orange	N	Scott	Duncan	Show	625 N.	Santa	CA	92701	714-79	
Rocky	N	Mark	Brown	Music	100	Denver	CO	80204	303-89	
Rocky	N	Joe	Rassenf	Entertai	100	Denver	CO	80204	303-89	
Salt	N	Melinda	Miller	Entertai	P.O.	Salt	UT	84110-	801-25	801-25
San	N	Michael	Crowell	Entertai	P.O.	San	CA	92112-	619-29	619-29

For more information on this wizard see “Excel Wizard” on page 52 of *Wizards & Demos*.

Financial Data Wizard

This new wizard allows financial information in QIF, OFX and QFX format to be imported into Panorama databases. Sources of this type of financial data include Intuit Quicken products and bank web sites.



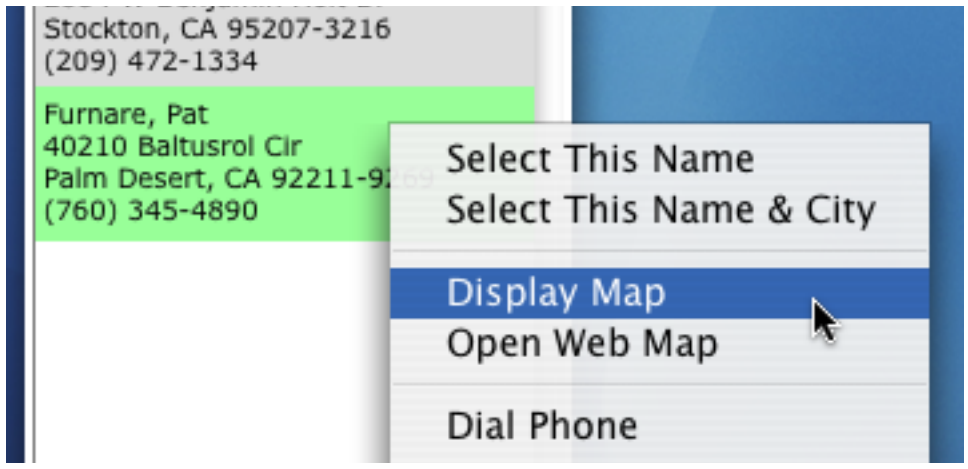
For more information on importing financial information see “Financial Data Wizard” on page 60 of *Wizards & Demos*.

Internet Submenu

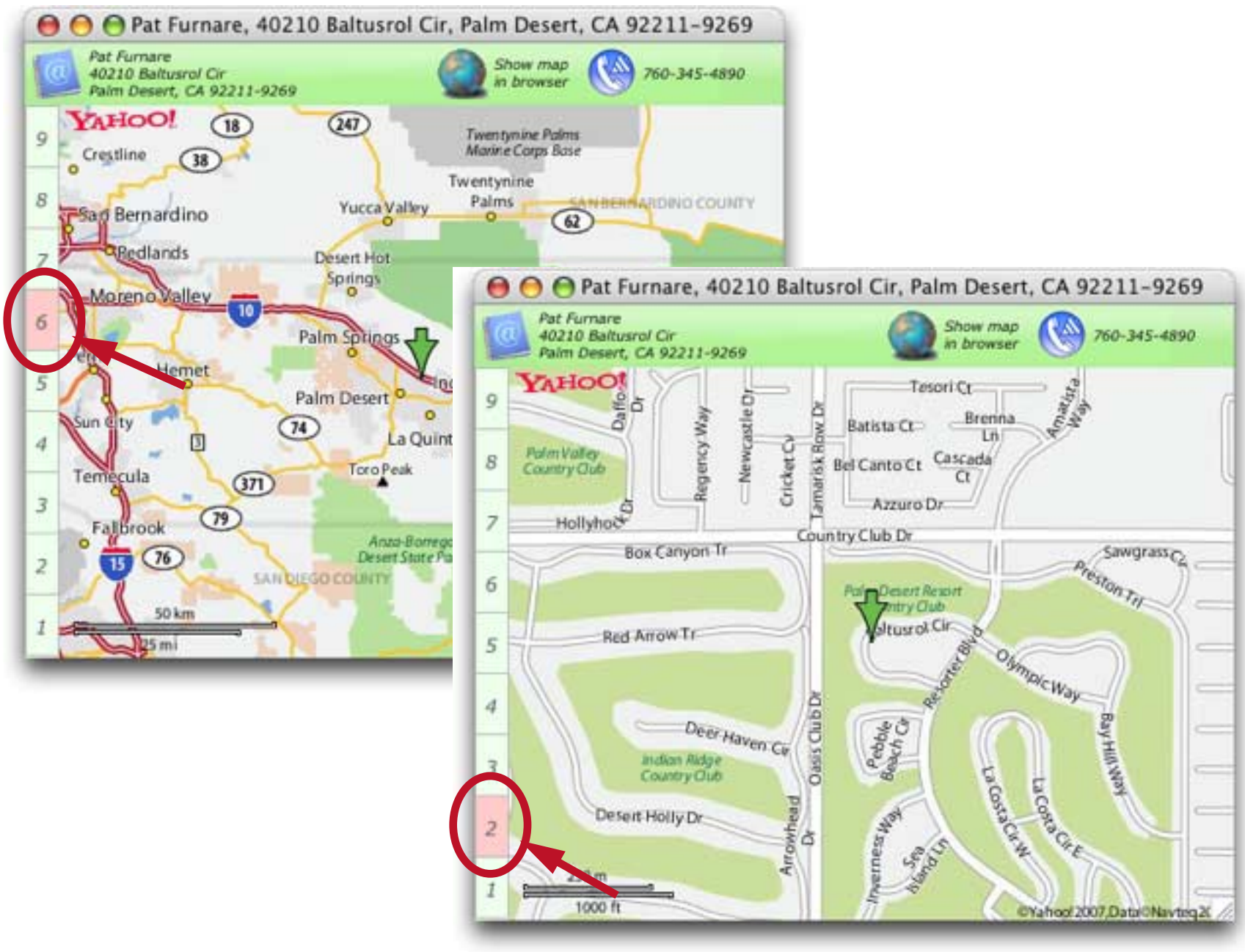
The map feature of the **White Pages** wizard has been enhanced.

White Pages

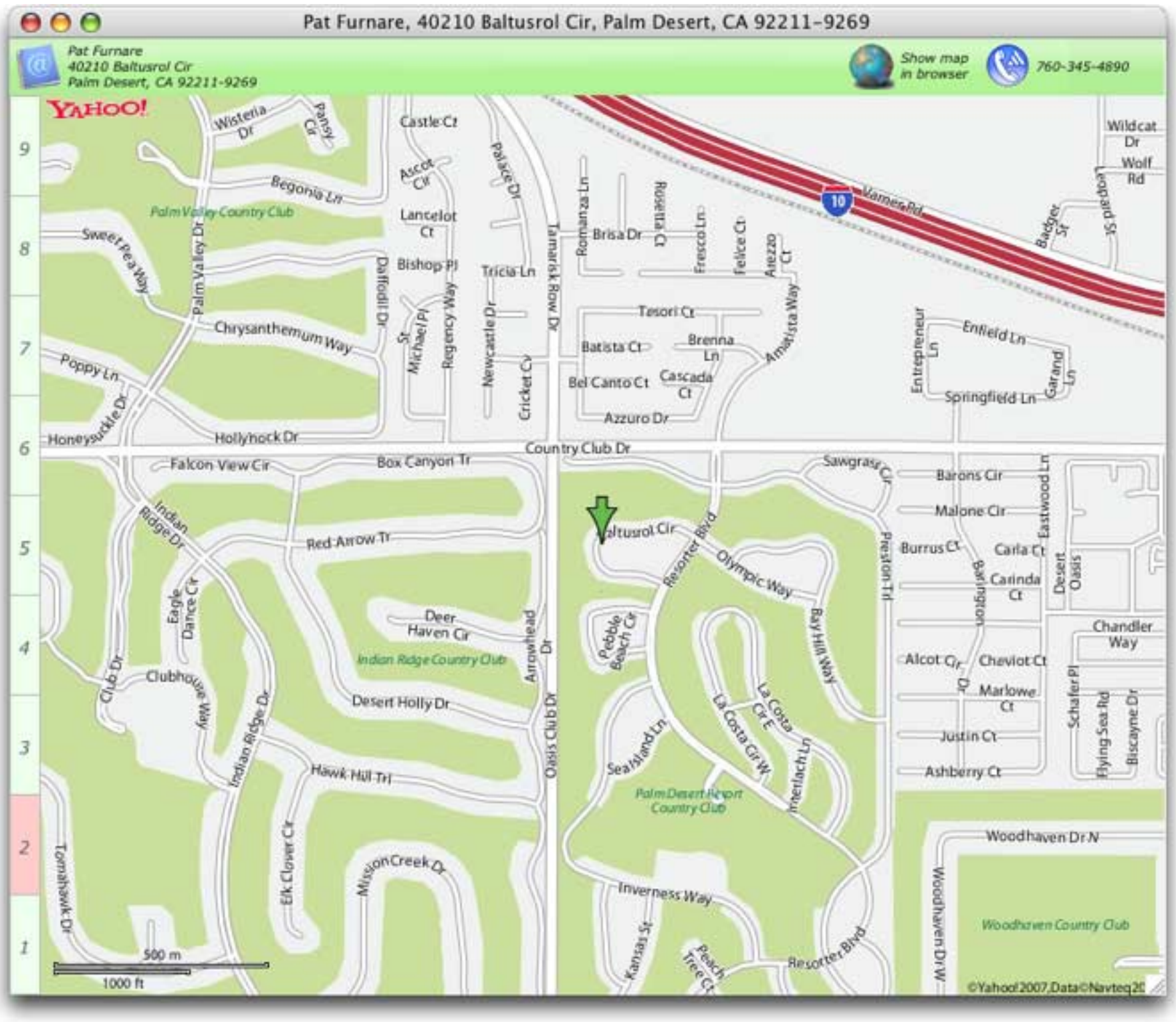
As with previous versions, holding down the **Control** key and clicking on a name in the White Pages wizard allows you to display a map of the selected address.



In previous versions the map size and scale was fixed. In Panorama 5.5 you can click on the numbers on the left to zoom in or out of the map.



You can also resize the map to show a smaller or larger area.



Mini Applications Submenu

Mini Correspondence

This wizard has three new menu items in the setup menu that allow you to easily change the default greeting, the default closing, and the default signature. See "Changing the Default Greeting, Closing and Signature" on page 735 of the *Panorama Handbook* for more details.

Preferences Submenu

This submenu has two new wizards — **Default Printer** and **Users & Groups**. There are also several new or revised channels.

Channels

Channels help Panorama communicate with the outside world. Panorama 5.5 includes three new or revised channels.

Jon's Phone Tool (Dialing). This channel has been updated to work with recent versions of Jon's Phone Tool. Jon's Phone Tool allows Panorama to dial via a wide variety of methods, including Bluetooth (cell phones), Analog Modems, Vonage; CallVantage; Speakeasy; BroadVoice; Skype; Ovolab Phlink; Parliant's PhoneValet; Asterisk (and OpenPBX.org); IP Phones from Cisco, Snom, and Others; Additional VoIP Support by Dialing Via eyeBeam, ineen, X-PRO, or X-Lite by CounterPath, the Gizmo Project, ohphoneX, LoudHush, JackenIAX, MegaFon, FRITZ!Box, and more. Jon is actively improving this application and new dialing methods are added on a regular basis. Jon's Phone Tool is a shareware application, currently the price is $22.50. Find out more and download the application from **http://www.jonn8.com/jpt/** .

XMail Channel (Email). This channel module sends e-mail via an open source AppleScript OSAX (see http://lestang.org/article63.html for information on downloading this module). The XMail OSAX is a free package that adds e-mail capability to AppleScript. You'll need access to an SMTP web server, most ISP's provide this as part of the package. Make sure that port 25 is not blocked by your firewall or by the ISP. Here at ProVUE Development XMail has become our preferred method of sending e-mail from Panorama.

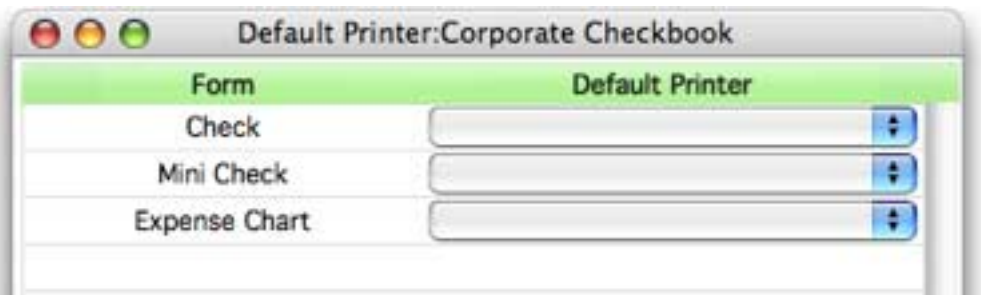
Yahoo Page Stock Channel (StockQuote). This channel module uses **finance.yahoo.com** to get a stock quote, and uses the same symbols as XMethods. This channel was submitted by Bruce De Benedictis and is provided as-is. You can use the

```
GETSTOCKQUOTE symbol,quote
```

statement to use this channel.

Default Printer Wizard

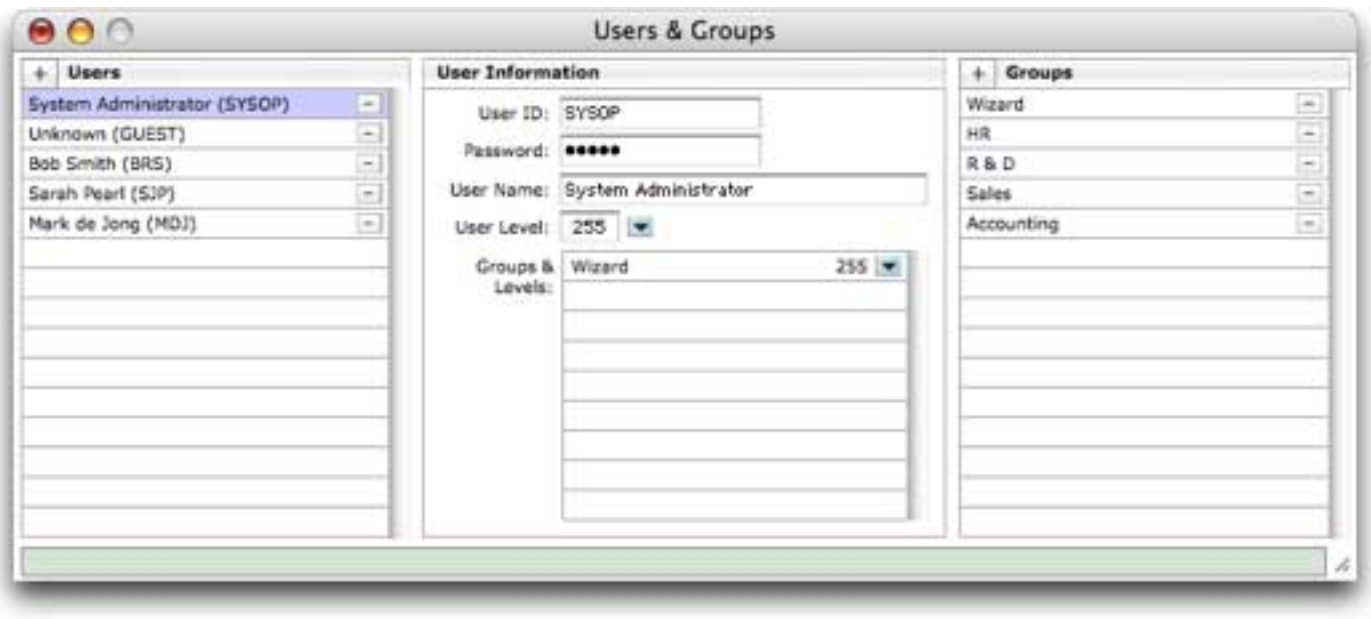
Panorama normally prints on whatever printer is currently selected in the operating system. If you are using Mac OS X you can also specify a default printer to be used with any form (this feature only works in the form view). When a default printer is set up Panorama will automatically switch to that printer when printing a form. For example you may want to print checks, envelopes or tables using a special printer. To configure this use the **Default Printer** wizard (in the **Preferences** submenu of the **Wizard** menu).



For detailed information on this feature see "Setting up Default Printers" on page 1059 of the *Panorama Handbook*.

Users & Groups

This wizard is used for set up users, groups, and access levels for accessing databases that have been locked with Panorama's security features.



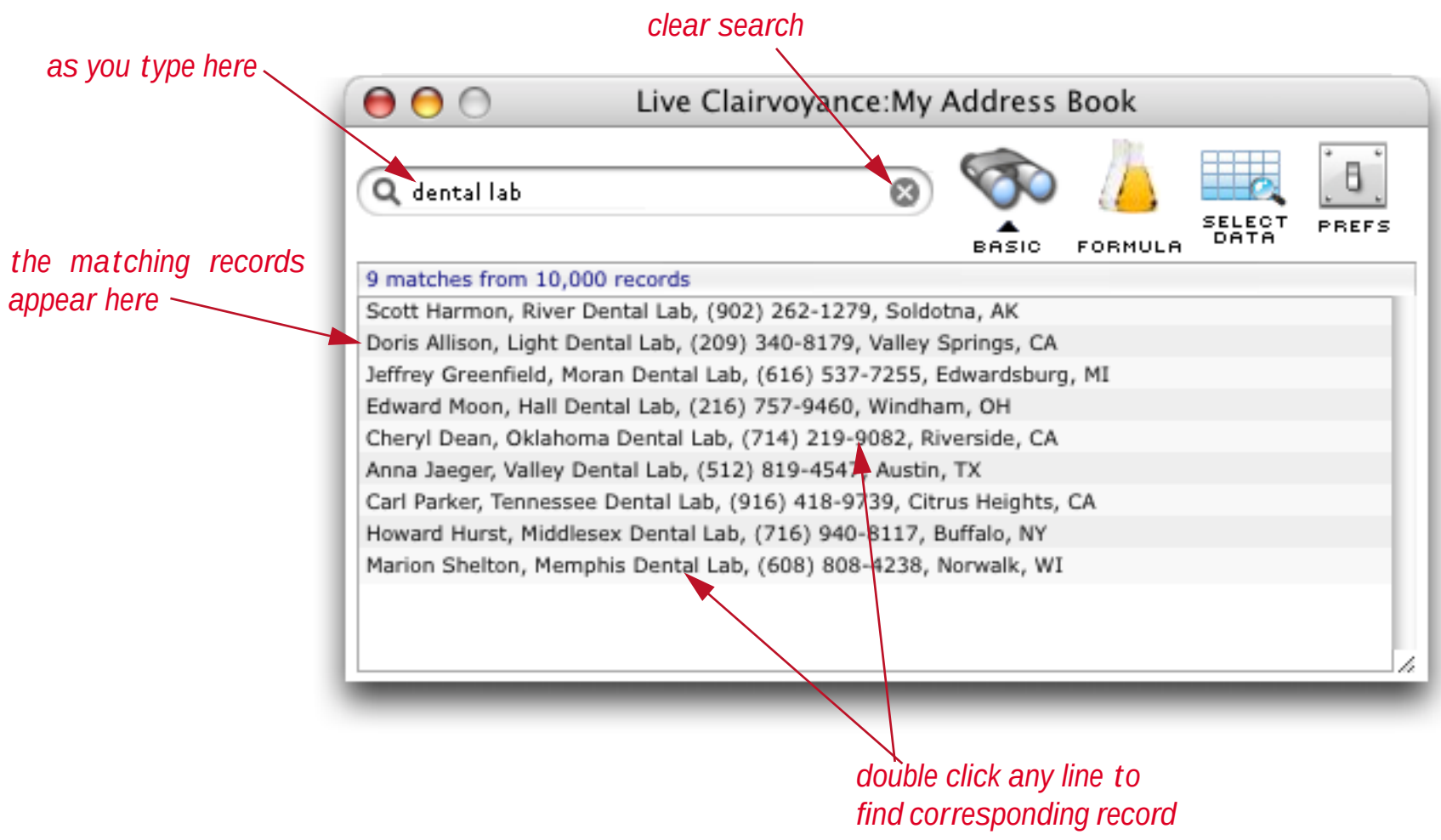
For more on this wizard and setting up database security see the Panorama Security Handbook (which much be purchased separately).

Search Submenu

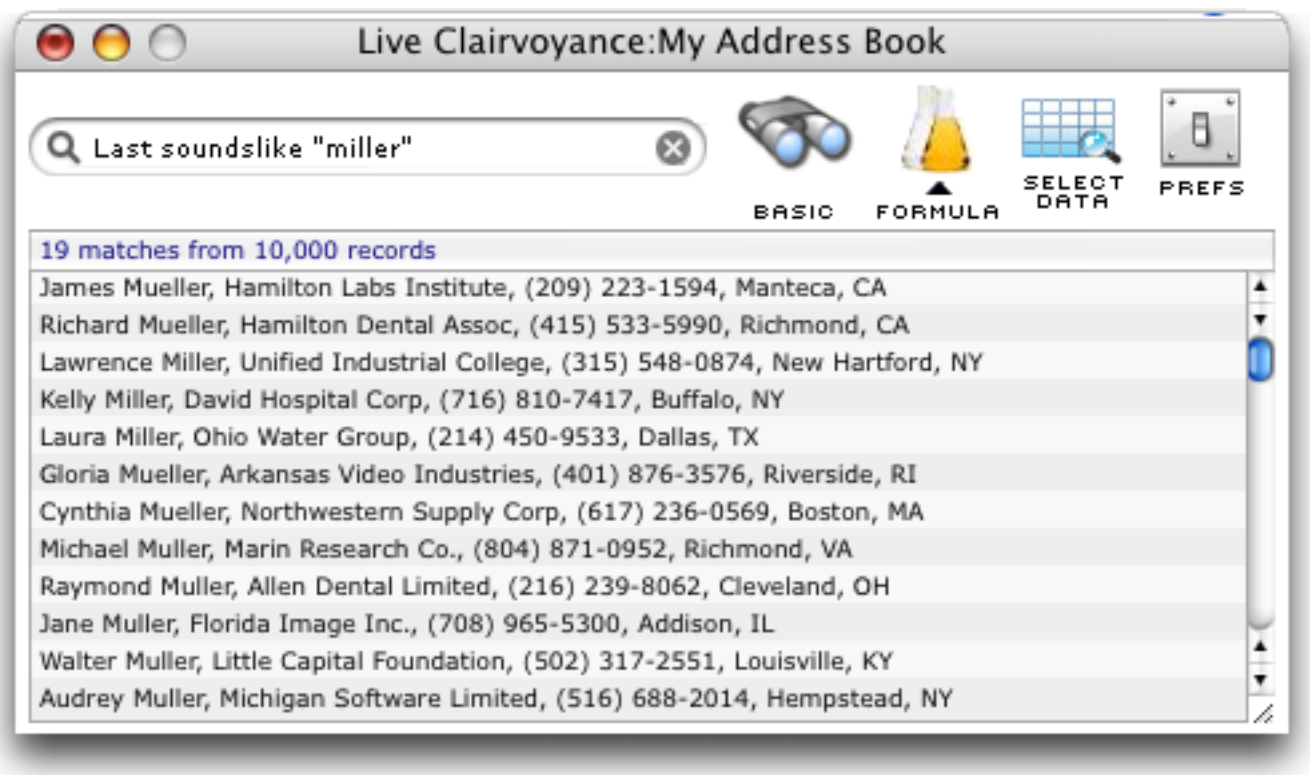
This submenu is new in Panorama 5.5. Two older wizards have been moved into this submenu (**Live Clairvoyance** and **Search All Fields**) and one new wizard has been added (**Quick Search**).

Live Clairvoyance

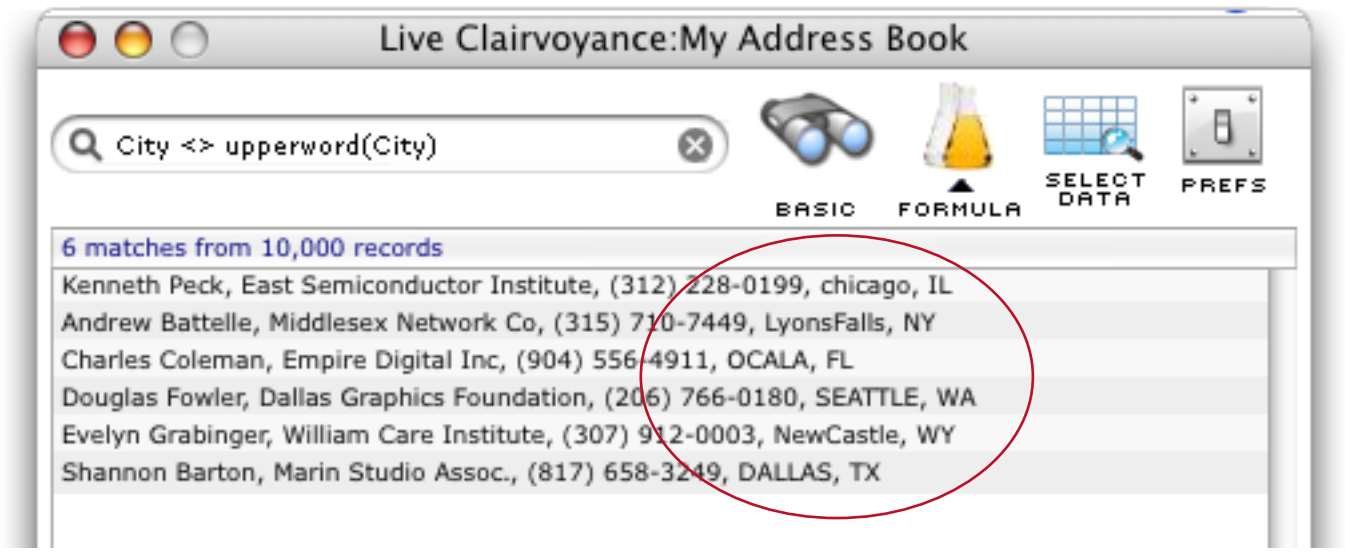
This wizard has been extensively updated. You can choose between a regular “basic” search or a formula search that can use any boolean logic. You can now select the records in the original database that match the live clairvoyance search.



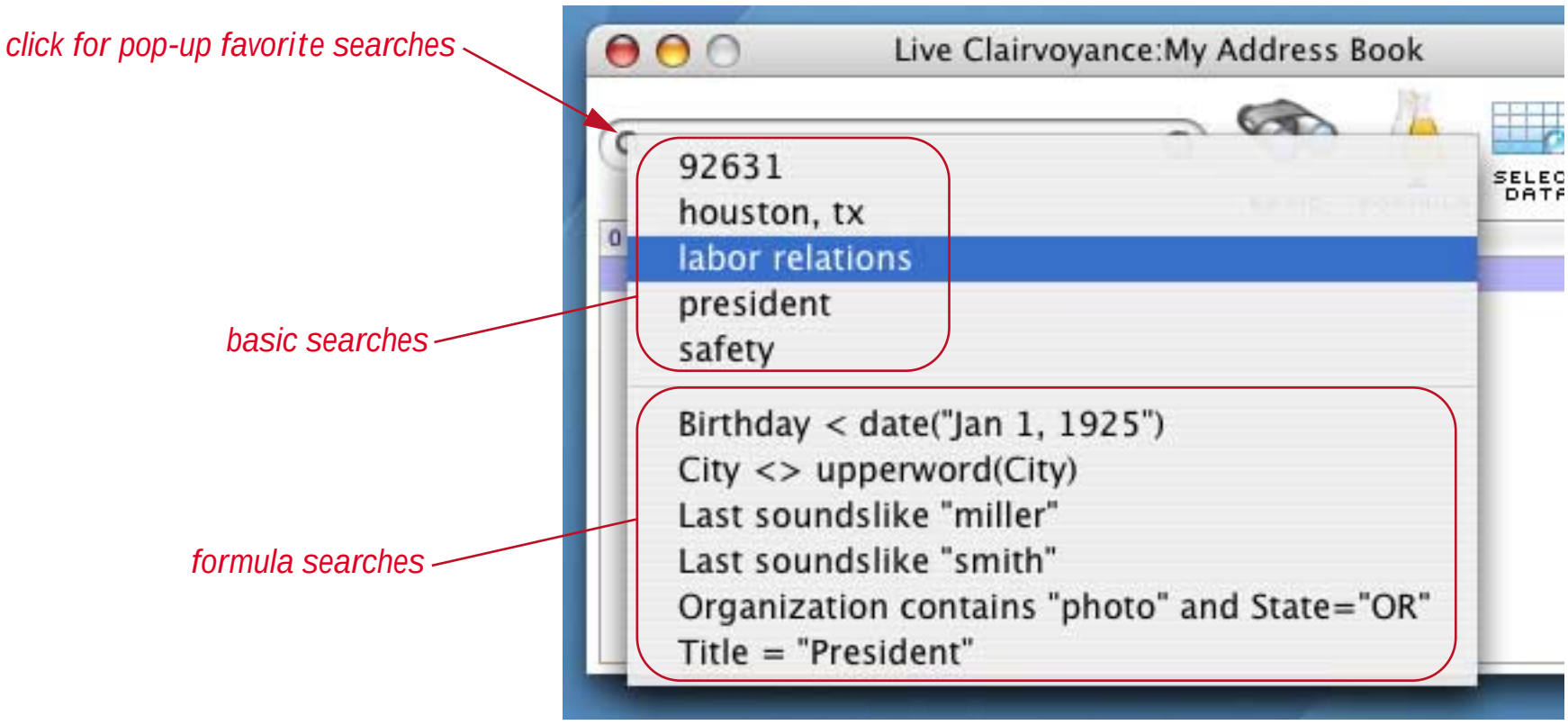
Here is an example of a new formula search that searches the database phonetically. Notice that searching phonetically for Miller will also find Mueller and other similar spellings.



Here is another example of a formula search that finds all city names that are not capitalized correctly.



The new version of this wizard allows you to keep a pop-up menu of your favorite searches. Click the icon to see the pop-up menu.

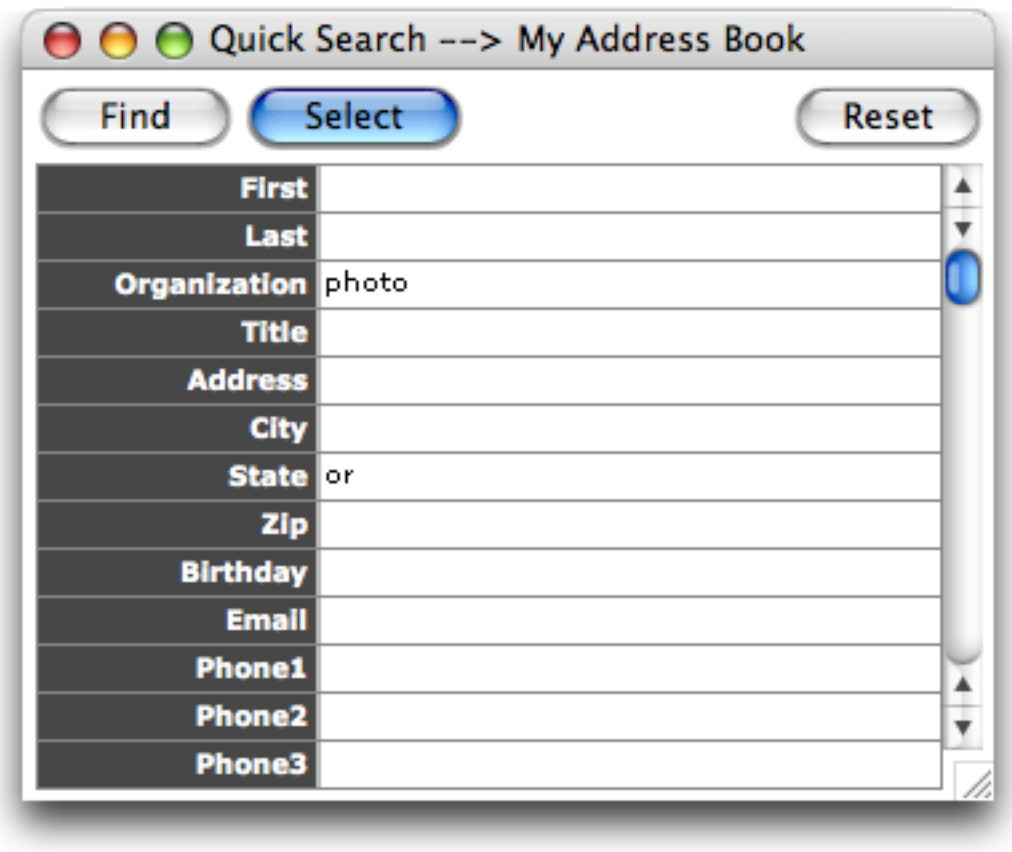


The top section of the menu lists basic searches you have saved, while the bottom section lists formula searches. To perform a search simply select it from the menu.

For more information on this wizard see "The Select Summaries Command" on page 361 of the *Panorama Handbook*.

Quick Search

This wizard provides an alternate method to locate information within databases. It allows you to easily locate data based on selection criteria in multiple fields. For example here is how you would locate all records that contained “photo” in the Organization field and “or” in the State field.



Sharing Submenu

The new wizards in this submenu are used to create and manage shared and web published databases with the Panorama Enterprise Server. If you have only a single user version of Panorama you can disregard these wizards.

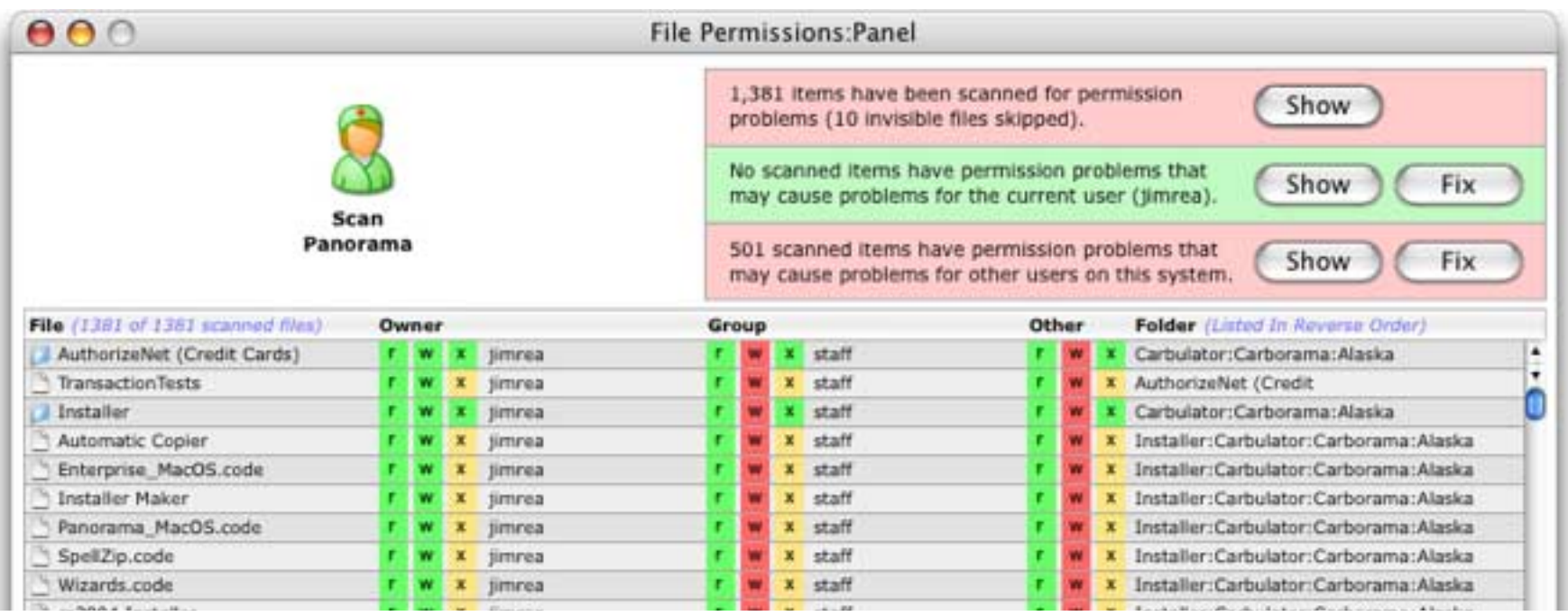
For more information on these wizards see the separate **Panorama Enterprise Handbook**.

Utilities Submenu

There are two new wizard in this submenu — **File Permissions** and **Zap Page Setup**.

File Permissions Wizard

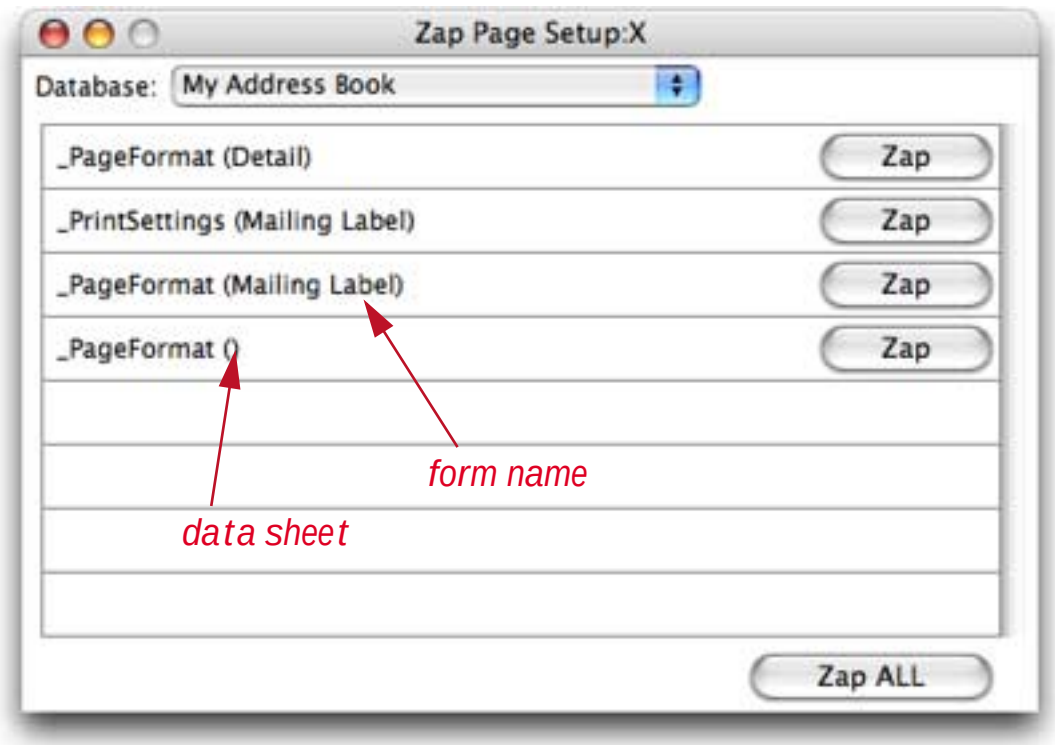
This wizard is for Mac OS X only. It is designed to scan files and folders looking for permission problems. Unlike Apple's **Disk Utility** program, which can only scan files that have been installed by a version of Apple's installer, Panorama's **File Permissions** wizard can scan any file or folder.



For more information on this wizard see "File Permissions" on page 119 of *Wizards & Demos*.

Zap Page Setup Wizard

Panorama stores page setup and print configuration information for the data sheet and for each form in each database. Normally you don't have to worry about this, but occasionally this information will become corrupted (we believe this is caused by problems with printer drivers) and Panorama will fail to print. The corrupted information cannot be recovered but you can zap it and start over. To do this open the **Zap Page Setup** wizard.

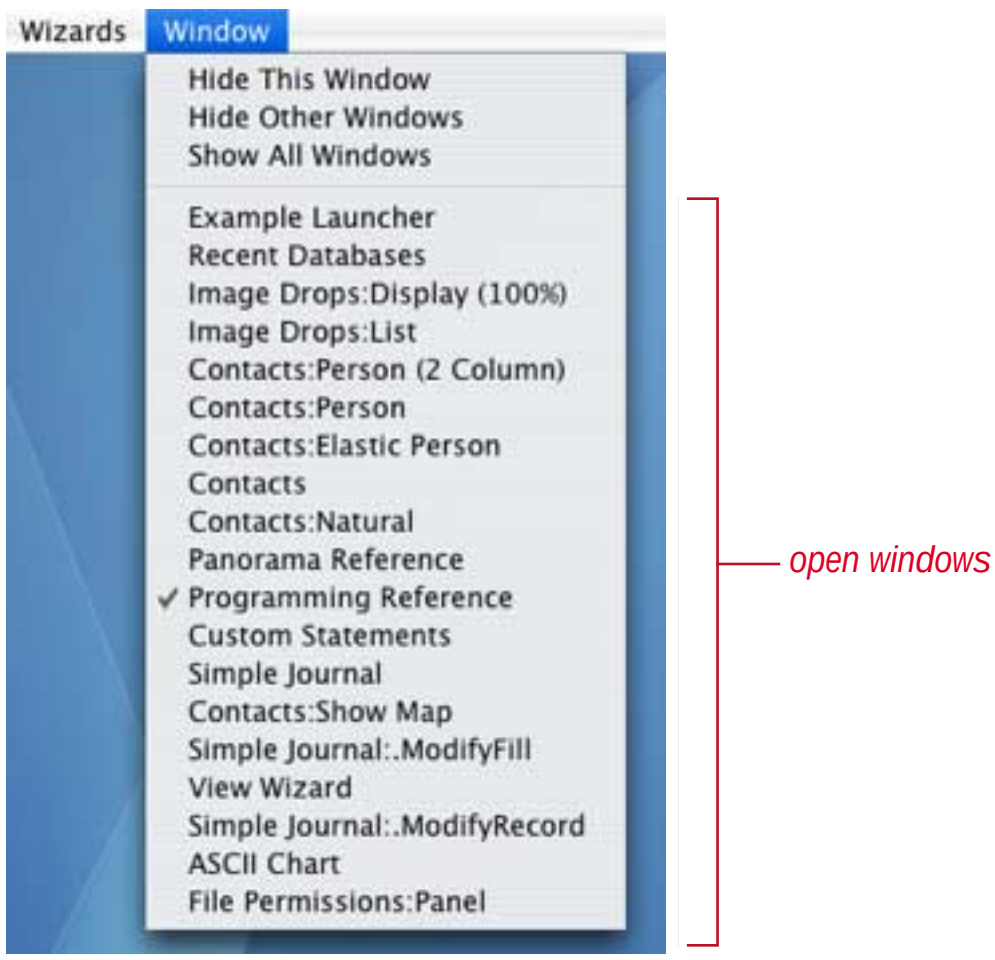


The wizard lists each entity (data sheet or form) that has page setup information. You can zap the settings for individual forms separately, or simply zap all of the page setup information for the entire database. Once you zap you'll need to use the **Page Setup** dialog to re-configure the settings for each item.

General User Interface Changes

Window Menu moved to Main Menu Bar

The **Window** menu, which used to be a submenu of the **File** menu, has been moved into the main menu bar.



Mighty Mouse and Trackpad Scrolling Support

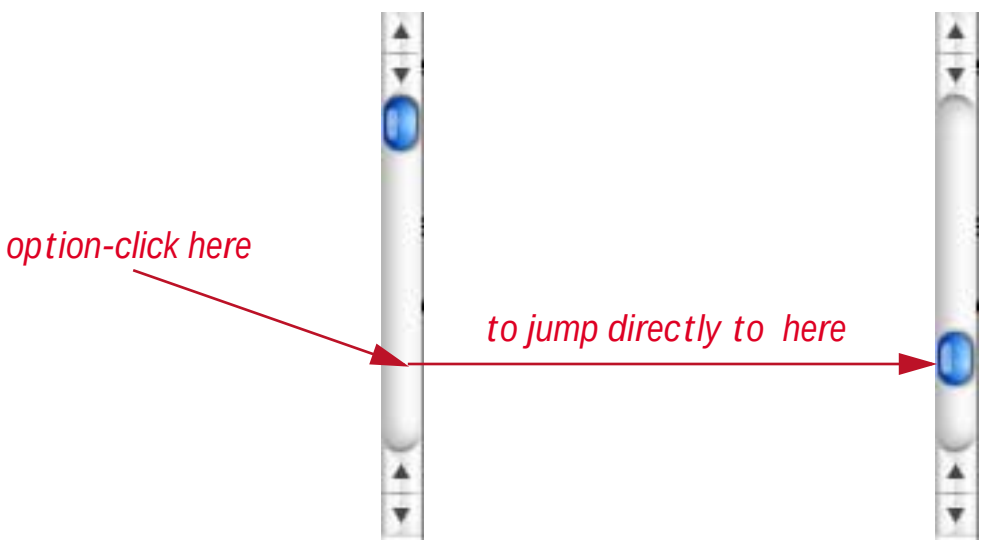
Panorama 5.5 now supports vertical scrolling with the ball on Apple's Mighty Mouse, as well as two finger scrolling on laptop trackpads. Please note, however, that horizontal and diagonal scrolling are not supported.

Horizontal Scrollwheel Scrolling with Shift Key

When using any scroll wheel holding down the **Shift** key while turning the wheel will cause the window to scroll horizontally. Note: Most Mac OS X applications support this behaviour, including Safari, Preview, GarageBand, etc.

Instant Jump to any Scroll Bar Position

To jump instantly to any position within a scroll bar hold down the **Option** key and click on the location you want to jump to.



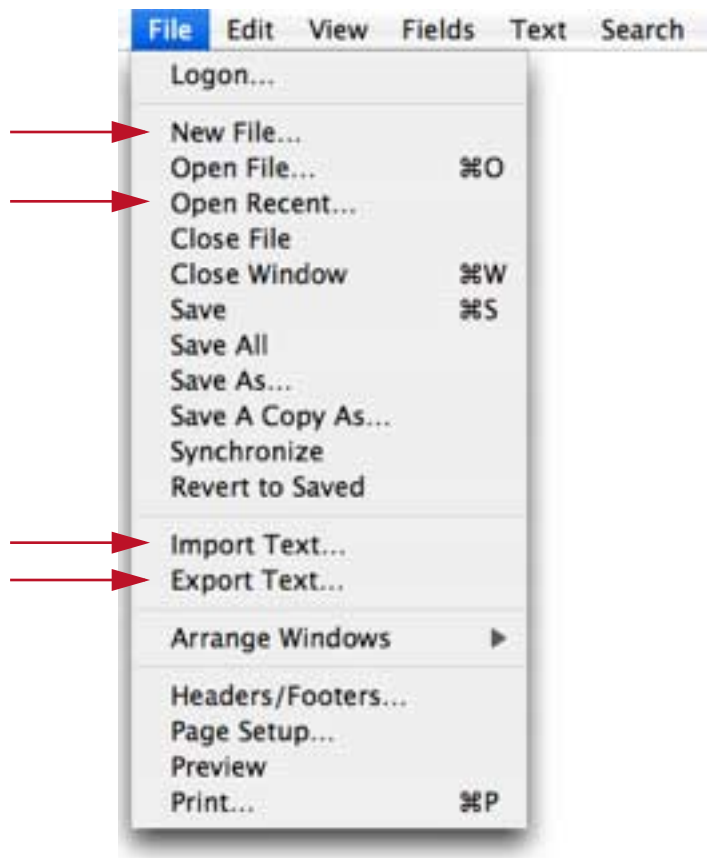
Note: Most Mac OS X applications support this behaviour, including Safari, Preview, GarageBand, etc.

Registration Command in Panorama Menu

The **Registration** command has been moved to the **Panorama** menu.

New File Menu Commands

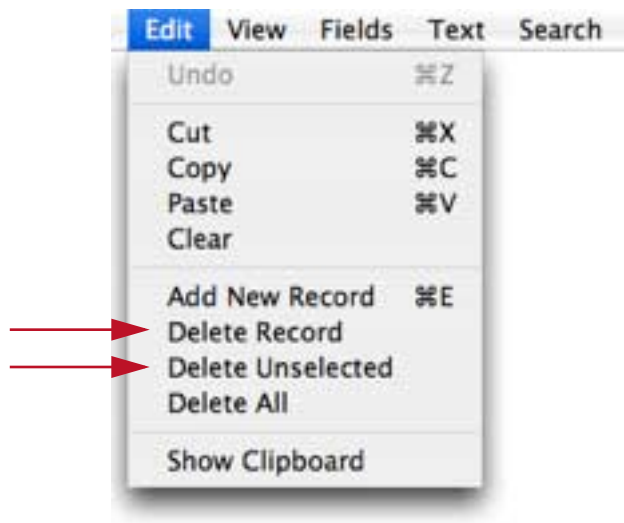
Four new commands have been added to the **File** menu.



For more information see “Using the New Database Wizard” on page 50, “The Recent Databases Wizard” on page 44, “Importing a Text File” on page 82 and “Exporting a Text File” on page 105 of the *Panorama Handbook*.

New Edit Menu Commands

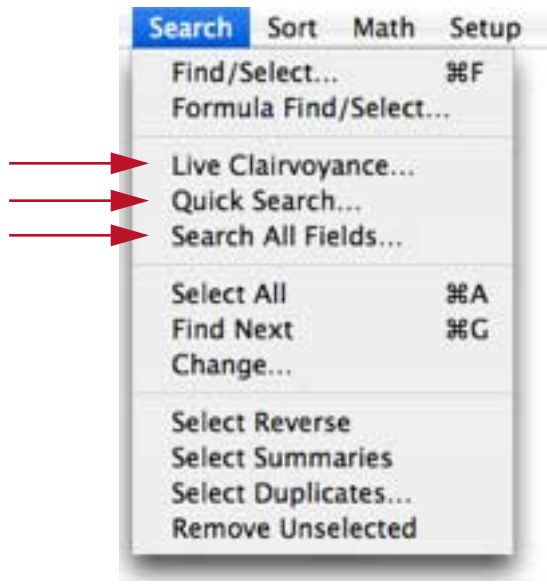
Two new commands have been added to the **Edit** menu.



For more information see “Browsing the Database With a Form” on page 488 and “Permanently Removing Unselected Data” on page 360 of the *Panorama Handbook*.

New Search Menu Commands

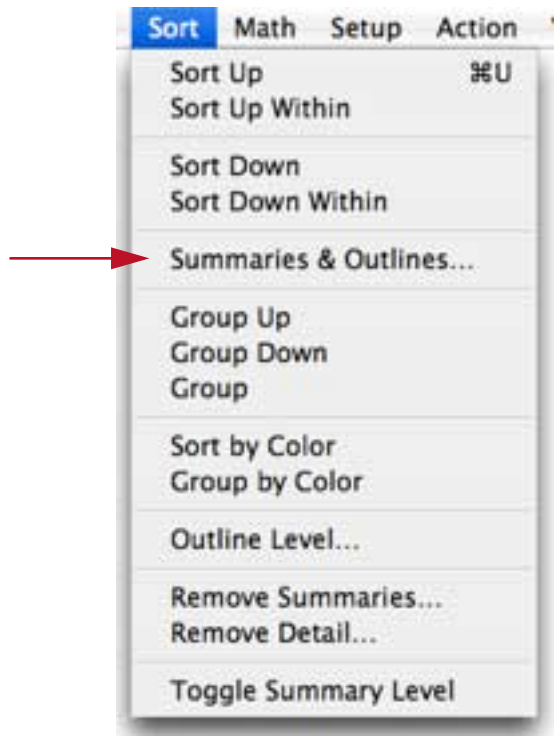
Three new commands have been added to the **Search** menu.



Note: These commands are no longer available in Panorama 6.0 and later.

New Sort Menu Commands

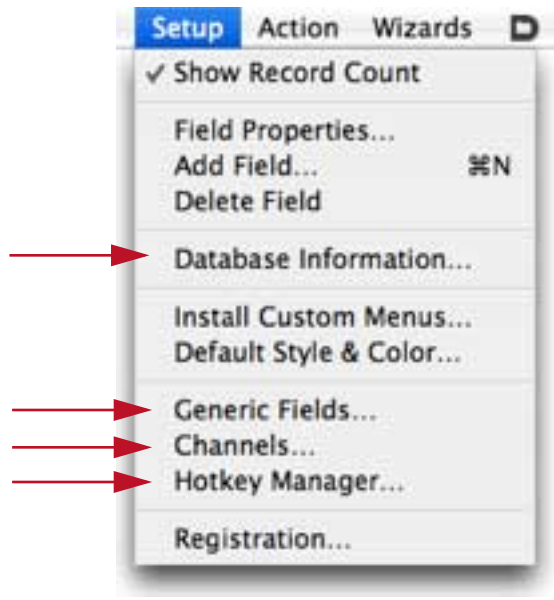
One new command has been added to the **Sort** menu.



Note: This command is no longer available in Panorama 6.0 and later.

New Setup Menu Commands

Four new commands have been added to the **Setup** menu.



For more information see "Viewing and Modifying Database Metadata" on page 73, ""Generic" Fields" on page 230, "Channels" on page 93 and "Hotkey Manager" on page 99 of the *Panorama Handbook*.

Chapter 1: Files

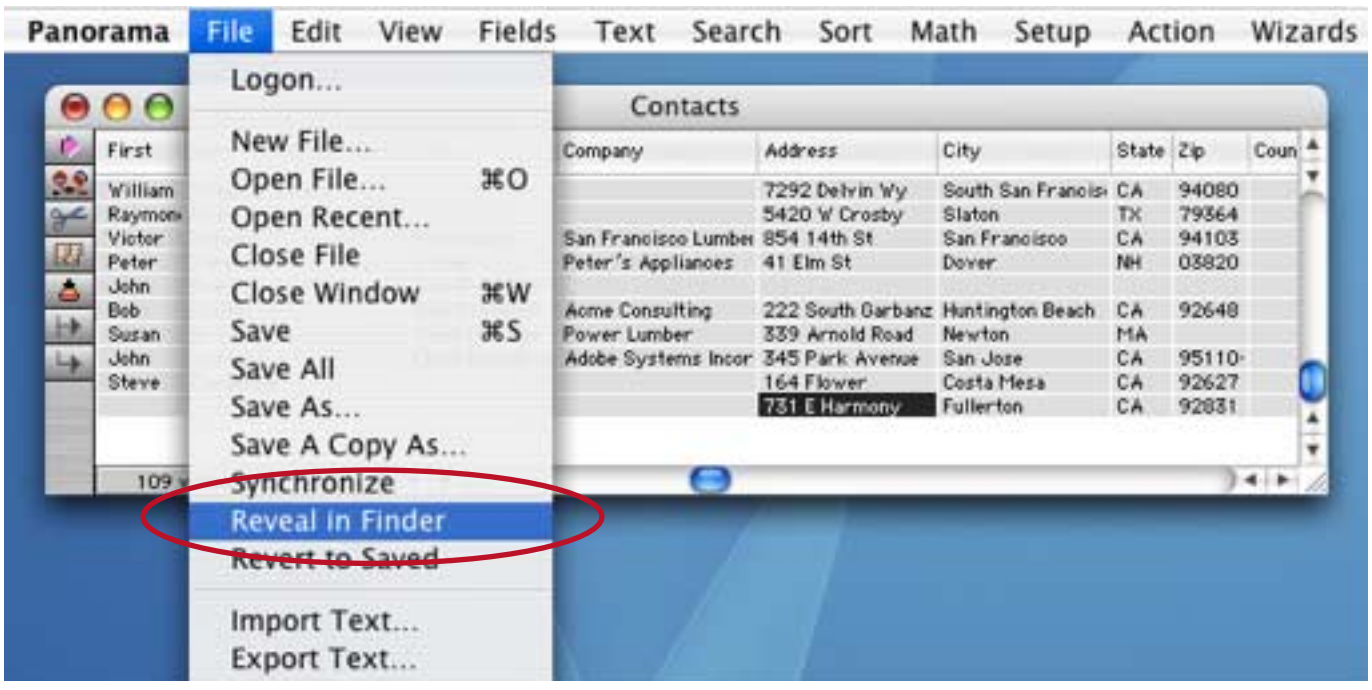
In addition to the changes listed below, Chapter 1 now includes documentation for several new commands in the File menu. For more information see "Using the New Database Wizard" on page 50, "The Recent Databases Wizard" on page 44, "Importing a Text File" on page 82 and "Exporting a Text File" on page 105 of the *Panorama Handbook*.

Saving Window Positions

When the save window positions option is selected (see "Saving Window Positions" on page 64 of the *Panorama Handbook*) Panorama no longer remembers the positions of procedure windows. Only data sheet and form windows will be re-opened.

Finding a Database on the Hard Disk

To find the location of the current database on the hard drive simply choose the new **Reveal in Finder** command from the **File** menu.



This will open the folder containing the database and highlight the database file. See "Finding a Database on the Hard Disk" on page 72 of the *Panorama Handbook*.

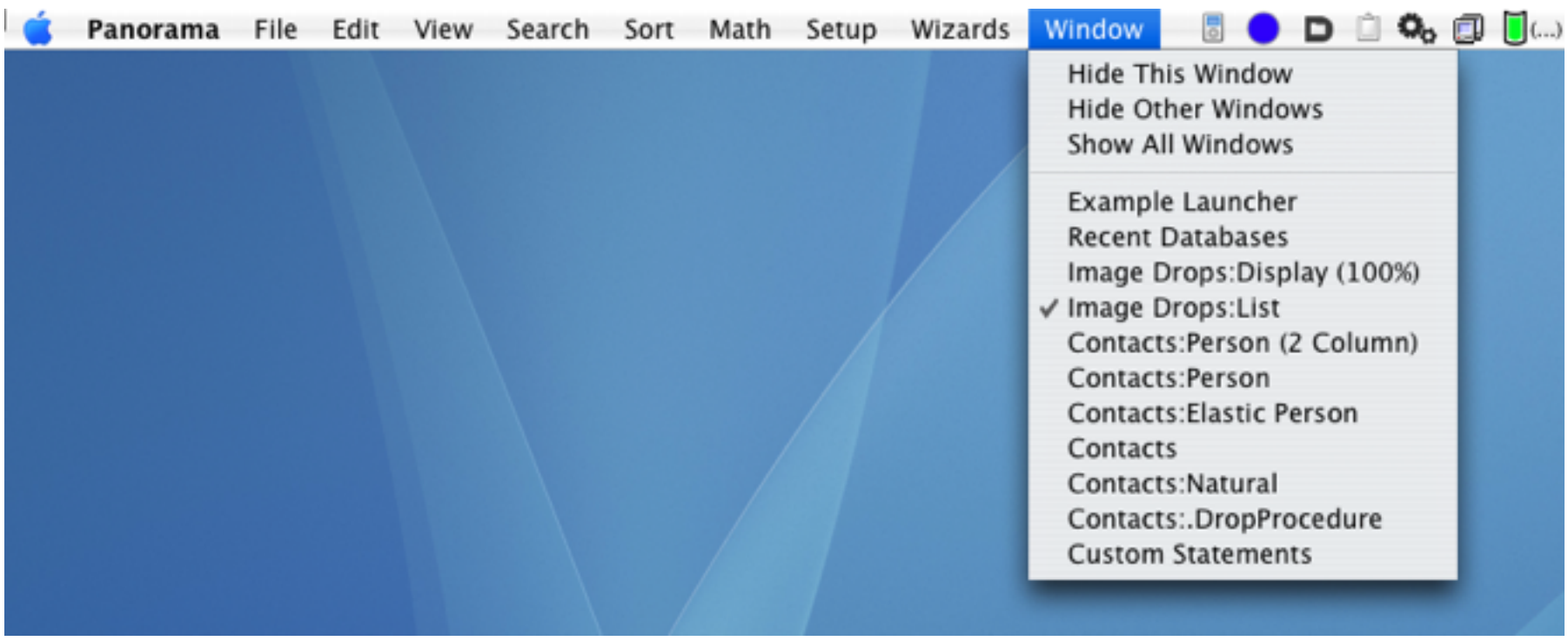
Adjusting Panorama's Memory Allocation (Windows and OS X) Beyond One Gigabyte

As shipped from the factory, Panorama normally allocates 100 megabytes of memory for databases. If your databases are larger than this you can increase this amount (see "Adjusting Panorama's Memory Allocation" on page 140 of the *Panorama Handbook*). Previous versions of Panorama would not allow memory allocations greater than 999 megabytes, but now there is no limit. (However, Panorama is a 32 bit application, so it will not be able to use more than 4 gigabytes of memory.)

Chapter 3: Windows

Improved Window Menu

The **Window** menu has been promoted from a submenu of the **File** menu into a full fledged menu on the main menu bar.



Up to 64 Open Windows

Panorama now allows up to 64 open windows at a time (the previous maximum was 32 windows).

Chapter 16: Images & Movies

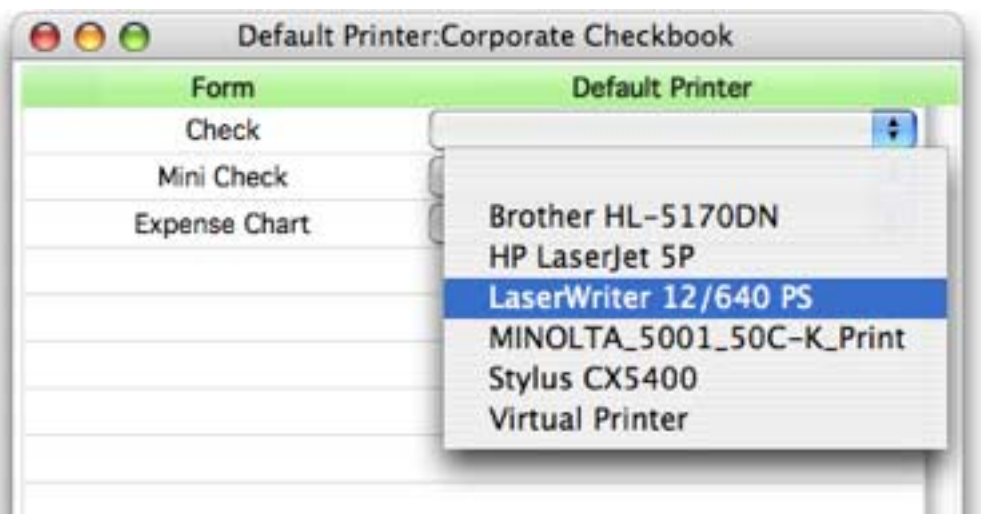
Dragging Images & Movies to a Super Flash Art object

Panorama 5.5 has a new feature that allows image & movie files to be dragged from the Finder into a database. This makes it easier than ever to work with visual media within Panorama.

To learn more about adding this feature to your databases (it only takes one line of code!) see "Flash Art Image Drag and Drop" on page 779 of the *Panorama Handbook*.

Chapter 20: Printing Basics

Panorama normally prints on whatever printer is currently selected in the operating system. If you are using Mac OS X you can now specify a default printer to be used with any form (this feature only works in the form view).



See "Setting up Default Printers" on page 1059 of the *Panorama Handbook* for detailed information on this feature.

Another new feature is the **Zap Page Setup** wizard (see "Zap Page Setup Wizard" on page 1060 of the *Panorama Handbook*). This wizard allows you to reset page setup information in case there is a printing problem.

Chapter 23: Formulas

Pipe Delimited Constants

Panorama has a special **pipe** constant delimiter that is very handy for creating contants that have other types of quotes within the constant. The constant starts with a series of pipes, and doesn't end until an equal number of pipes. For example if the constant starts with 3 pipes it should also end with three pipes.

```
|||last="Elliot" first="Suzette" address="892 Melody Lane"|||
```

You can even embed pipes within a piped constant, like this:

```
|||| language=javascript code=||alert("Hello World");|| ||||
```

As you can see pipe delimited constants are very handy for creating text constants that contain computer code.

New Functions

Panorama 5.5 contains over 150 new functions. For a complete list use the **Version** menu in the **Programming Reference** wizard. Some of the more significant new functions are listed in the table below.

Function	Reference Page	Description
adjustservervariable(database, variable,adjustvalue)		This function adjusts the value of a server variable. If adjustvalue is a number the server variable must also be numeric, and is incremented or decremented. If adjustvalue is text, it is appended to the server variable.
after(text,tag)		This function extracts all of text after a specified tag (sequence of characters. If the tag doesn't exist within the text the function returns `""`.
arrayboth(a1,a2,sep)		This statement compares two arrays. The result is a list of elements that are included in both arrays. Note: Empty array elements, if any, will be ignored. Both arrays must use the same separator.

Function	Reference Page	Description
arraybuild(sep,db,formula)		This function builds an array by scanning all records in a database. Note that the usage of this function is slightly different than the arraybuild statement in that the formula itself must be quoted. For example, instead of just upper(Name) you would need to use "upper(Name)" or {upper(Name)}.
arraycolumn(array,colnum, rowsep,colsep)		This function extracts a column from a two dimensional array. The second parameter specifieds what column to extract. The third paramter specifies the main separator between each row of the array, the fourth parameter is the sub-separator between each column.
arraydeduplicate(text,separator)		This function removes duplicate values from an array. As a by-product it also sorts the array.
arraydifference(a1,a2,sep)		This statement compares two arrays. The result is a list of elements that are in the first array but not the second. (Note: Empty array elements, if any, will be ignored.) Both arrays must use the same separator.
arraynotcontains(text,item,sep)		This function is the reverse of the arraycontains(function.
arraynumericsort(text,separator)		This function sorts an array in numerical order (instead of alphabetical order).
arraynumerictotal(text,separator)		This function totals the numerical elements of an array. For this function to work each array element must contain a number. The result is a number.
arrayrandomize(text,separator)		This function re-arranges the elements of an array in random order.
arrayselectedbuild(sep,db,formula)		This function builds an array by scanning the selected records in a database. Note that the usage of this function is slightly different than the arrayselectedbuild statement in that the formula itself must be quoted. For example, instead of just upper(Name) you would need to use "upper(Name)" or {upper(Name)}.
arraysort(text,separator)		This function sorts an array in alphabetical order.
arraytableceiling(array,key,sep, subsep,default)		This function looks up a value in a double column table, similar to the table(function but from an array instead of a database. The table must be sorted in ascending order. The first column in the table is the key value, the second column is the data value. (Note: If the sub separator is "", the first column is used as the data value also.) If there is no exact match this function will return the next higher value, if any. If the key value is numeric or contains all numbers, numeric comparisons (50 < 200) will be used instead of text comparisons (50 > 200). (Note: This function always returns a text value, even if the key is numeric.) If the key value is not within the array bounds the default value will be returned.
arraytablefloor(array,key,sep, subsep,default)		This function looks up a value in a double column table, similar to the table(function but from an array instead of a database. The table must be sorted in ascending order. The first column in the table is the key value, the second column is the data value. (Note: If the sub separator is "", the first column is used as the data value also.) If there is no exact match this function will return the next lower value, if any. If the key value is numeric or contains all numbers, numeric comparisons (50 < 200) will be used instead of text comparisons (50 > 200). (Note: This function always returns a text value, even if the key is numeric.) If the key value is not within the array bounds the default value will be returned.
arrayunpropagate(text,separator)		This function scans an array from top to bottom. If it finds two or more duplicate values in a row, it blanks out all but the first. This is similar to Panorama's UNPROPAGATE command in the Math menu, but for an array instead of a database field.

Function	Reference Page	Description
batchreplace(text,array,sep,subsep)		This function performs multiple find and replace operations on a string of text. The array must be two dimensional, it specifies what words or phrases are going to be replaced and with what. This is similar to replacemultiple(but the before and after strings are combined into a single array, making it easier to use.
before(text,tag)		This function extracts all of text before a specified tag (sequence of characters. If the tag doesn't exist within the text the function returns **""**.
bestfitrectangle(lborder,rect)		This function fits a rectangle inside a border. The original rectangle is enlarged or reduced as necessary to produce the best fit without changing the proportions. This function is useful for fitting an image (picture) inside a fixed size border.
boolstr(value)		This function converts a boolean value to text, either "true" or "false".
bytepattern(number)		This function converts a number into text. The number is treated as a number of bytes, and depending on the size will be displayed as bytes, kilobytes, megabytes, or gigabytes. This function uses SI units, meaning that 1 kB = 1000 bytes, 1 MB = 1000 kB, and 1 GB = 1000 MB. See http://en.wikipedia.org/wiki/Megabyte for more information on SI units for data size.
call(database,procedure,p1,p2,p3)		This function allows a procedure to be called as a subroutine within a formula. See "Using a Subroutine in a Formula (the CALL(function)" on page 273 of *Formulas & Programming* for detailed information on using this function.
callingdatabase()		This function returns the name of the database that contained the procedure that called this procedure as a subroutine (if any). If a procedure was called as a subroutine by another procedure this function will return the name of the database that contains the calling procedure.
callingprocedure()		This function returns the name of the procedure that called this procedure as a subroutine (if any). If a procedure was called as a subroutine by another procedure this function will return the name of the calling procedure.
callwithindatabase()		This function returns returns true if the current procedure was called by another procedure in the same database, false if it was called by a procedure in another database or called as a standalone procedure (for example with a button or the Action menu.
cardvalidate(text)		This function returns true if the text contains a credit card number that is valid, false if it is not. Credit cards have an internal checksum that allows a number to be validated for simple data entry errors (for example missing or transposed digits). This function checks to make sure that a number is a valid credit card number. Of course the function cannot tell whether this card number has actually been issued, or what the credit limit is or any other financial information about the card. It simply provides a simple check for missing or transposed digits. If this function says the card number is not valid you are sure that the number is wrong, but if the function returns true you would still need to check with the issuer to determine if this is a valid card. (Note: Unlike the cardvalidate statement, this function will remove any non-numeric data from the card number, so it's ok to leave in spaces, dashes, etc.)
checkenglish(word)		This function returns true if the specified word is in Panorama's English dictionary, false otherwise. (Note: If the optional Panorama Spell + Zip package is not installed this function will always return false.)
commastr(number)		This function converts a number into text, with a comma every third digit. The number is converted as an integer, with no places after the decimal point.

Function	Reference Page	Description
constantvalue(fieldorvariable)		This function converts a field or variable into an equivalent constant value. If the field or variable contains text the result will be quoted. The result of this function can be used in an execute statement.
countsummaries(level)		This function counts the number of summary records in the current database. The level parameter should be from 0 to 7. If 0, all summary records will be counted. If 1 to 7 then only that specific level will be counted.
currentprinter()		This function returns the name of the current printer (OS X only).
datavalue(fieldorvariable)		This function returns the value of a field or variable. The advantage of this function is that you can calculate the name of the field or variable at the time the formula is evaluated.
datevalue(year,month,day)		This function converts three integers into a date. The three integers are the year, month and date. This function provides a way to create a date that is independent of the system date settings (the date(function, which can also create dates, will produce different values in different countries depending on the date formats used in those countries).
dayvalue(date)		This function extracts the day of the month from a date as a numeric value (1 to 31).
dbcheckopen(database)		This function returns true if the specified database is currently open, false if it is not.
dbserverdomain()		This function returns the server domain for the database. If this is running on a server, it always returns the servers domain. If running on a client (presumably for testing) it will return the ip address of the server this client is hosted on.
dbshared()		This function returns true if the current database is shared, false if it is not.
dbwebpublish()		This function returns true if the current database is web published, false if it is not.
decodebase64(data)		This function decodes text that has been encoded using Base64 encoding. It is the reverse of the encodebase64(function described below. If you first use encodebase64(then decodebase64(you will get back the original data.
defaulttext(text,default)		This function returns the text value supplied in the first parameter. However, if this text value is empty (**""**) the function will return the specified default value.
defaultvalue(default,text)		This function returns a text value (the second parameter). However, if the second parameter is empty then the default value is returned (the first parameter).
emptydatabase()		This function returns true if the current database is empty, false if it is not.
emptyline()		This function returns true if the current line (all fields) is empty, false if it is not.
encodebase64(data,linelength)		This function encodes text using the Base64 algorithm. Base64 encoding is widely used on the web and in e-mail for encoding binary data and allowing it to be transmitted as plain text. For more information on Base64 encoding see http://en.wikipedia.org/wiki/Base64. The data parameter is the data to be encoded. LineLength is the maximum line length of the encoded text. A common value is around 70 characters per line - MIME data may contain an maximum of 76 characters per line.
fileexists(path)		Checks to see if a file exists. Returns true or false.
filefolder(text)		This function returns the folder ID from a combined path and filename. For example the function filefolder("Alaska:Denali:Image45.jpg") would return the folder id for the path "Alaska:Denali:".

Function	Reference Page	Description
filesuperdate(folder,file)		This function returns the modification date and time of a file as a super-date. For example this can be handy for comparing to see which of two files is newer.
foldercontains(folder,file)		Checks to see if a file exists inside the specified folder. Returns true or false.
foldercontents(folder)		This function returns a list of all of the contents in a folder (both folders and files).
foldercount(path)		This function returns the total number of files within a folder.
foldersize(path)		This function returns the total size of all of the files within a folder.
formatphone(text)		This function reformats a phone number to the standard US form number format — (aaa) nnn-nnnn.
getautonumber()		This function returns the automatically generated number for the next record that will be added to the database.
getproceduretext(database, procedure)		This function returns the contents (source) of a procedure. The first parameter is the name of the database (or "" for the current database) and the second parameter is the name of the procedure.
getwebcolor(webcolor,default)		This function will calculate an RGB color from a Netscape style web color. The first parameter specifies an HTML color, either in RRGGBB format (or #RRGGBB) or a named color (see list below). The second parameter allows you to specify what color to use if the first parameter is not recognized (for example a named color not on the list. **List of named web colors:** aliceblue, antiquewhite, aqua, aquamarine, azure, beige, bisque, black, blanchedalmond, blue, blueviolet, brown, burlywood, cadetblue, chartreuse, chocolate, coral, cornflowerblue, cornsilk, crimson, cyan, darkblue, darkcyan, darkgoldenrod, darkgray, darkgreen, darkkhaki, darkmagenta, darkolivegreen, darkorange, darkorchid, darkred, darksalmon, darkseagreen, darkslateblue, darkslategray, darkturquoise, darkviolet, deeppink, deepskyblue, dimgray, dodgerblue, firebrick, floralwhite, forestgreen, fuchsia, gainsboro, ghostwhite, gold, goldenrod, gray, green, greenyellow, honeydew, hotpink, indianred, indigo, ivory, khaki, lavender, lavenderblush, lawngreen, lemonchiffon, lightblue, lightcoral, lightcyan, lightgoldenrodyellow, lightgreen, lightgrey, lightpink, lightsalmon, lightseagreen, lightskyblue, lightslategray, lightsteelblue, lightyellow, lime, limegreen, linen, magenta, maroon, mediumaquamarine, mediumblue, mediumorchid, mediumpurple, mediumseagreen, mediumslateblue, mediumspringgreen, mediumturquoise, mediumvioletred, midnightblue, mintcream, mistyrose, moccasin, navajowhite, navy, oldlace, olive, olivedrab, orange, orangered, orchid, palegoldenrod, palegreen, paleturquoise, palevioletred, papayawhip, peachpuff, peru, pink, plum, powderblue, purple, red, rosybrown, royalblue, saddlebrown, salmon, sandybrown, seagreen, seashell, sienna, silver, skyblue, slateblue, slategray, snow, springgreen, steelblue, tan, teal, thistle, tomato, turquoise, violet, wheat, white, whitesmoke, yellow, yellowgreen
grabfieldtype(database,field)		This function gets the type of a database field in any open database. The result is an integer: 0=text, 4=date, 5=floating point, 6-10=integer.
growlrunning()		This function This routing checks to see if Growl is running, and returns true if it is (otherwise false). (Mac OS X only).
homefolder(subpath)		This function returns the folder ID of a subfolder of the current users Home folder. (Mac OS X only).
homepath(subpath)		This function returns the path of a subfolder of the current users Home folder. (Mac OS X only).

Function	Reference Page	Description
htmltabletoarray(table, rowsep,colsep,columns)		This function converts an HTML table into a two dimensional text array. You must specify two separator characters — the row separator (between lines) and the column separator (between columns). The fourth parameter is a list of table columns to include, comma separated. For example "2,5,6,10" would tell this function to include 4 of the HTML columns in the final array. If this parameter is set to "" then all of the columns in the HTML table will be included in the output array.
info("modifiedfield")		This function returns the name of the field that was just modified. The most recently modified field is not always the same as the current field (see the info("fieldname") function). For example if a checkbox or radio button has just been clicked this function will return the name of the field associated with that object rather than the current field. This function is designed to be used in the .ModifyRecord and .ModifyFill procedures, the results may not be valid if the function is used in any other context.
info("parameters")		This function returns the number of parameters passed to a procedure.
info("runninghandler")		This function returns true if the current procedure is running as a "handler" procedure. See "Event Handler Procedures" on page 394 of *Formulas & Programming* for more information on this mode.
linestrip(text)		This function removes any blank lines from the text.
listprinters()		This function returns a carriage return delimited list of the printers available on the system (Mac OS X only).
makenumberedarray(sep,start,end)		This function generates a numeric sequenced array, for example 1, 2, 3, 4, 5. You can specify the starting and ending number of the sequence.
monthvalue(date)		This function extracts the month from a date as a numeric value (1 to 12).
naturaldata(date)		This function converts a date to text in a natural format similar to how people would refer to the date, for example Today, Tue, Apr 4. If the date is more than 180 days in the past or is in the future the pattern mm/dd/yy is used. This is similar to how the Apple Finder displays dates..
nth(number)		This function converts a number into an ordinal, i.e. 1=1st, 2=2nd, 3=3rd, 4=4th, etc.
obscuredigits(number,count)		This function obscures digits (usually a credit card number) with X's. The first parameter is the text that contains the digits. The second parameter is the number of digits on the end that will NOT be obscured. For example the formula obscuredigits("1234-5678-9876-5432",4) will produce the value XXXX-XXXX-XXXX-5432. Notice that the function retains any additional formatting in the text, in this case dashes.
padzero(text,width)		This function makes the text a fixed width. If the text is shorter than the specified width, it is padded with 0's on the left (i.e. the text is right justified). If it is longer than the specified width, it is cut off on the left..
panoramasubpath(child)		This function makes it easy to reference any subpath of the Panorama folder. Or you can leave it blank ("") to reference the main Panorama folder.
pathcontains(path,file)		Checks to see if a file exists inside the specified folder. Returns true or false.
pathcontents(path)		This function returns a list of all of the contents in a folder (both folders and files).
places(number,places)		This function converts a number to text with a specific number of places after the decimal point. The value is truncated (not rounded) to the number of places specified..
quartervalue(date)		This function extracts the quarter within a year from a date as a numeric value (1 to 4).
randomline(text)		This function picks a random line from some text.

Function	Reference Page	Description
randomword(wordlist)		This function picks a random word from some text.
rangecontains(thetext,therange)		This function checks to see if the text contains any characters in the specified range. The range must be a series of character pairs, for example AZ for upper case alphabetic characters, AZaz for upper and lower case, 09 for numeric digits, etc. If the text contains any characters in the specified range the function returns true, otherwise it returns false. For example, rangecontains(Company,"09") will return true if the company name contains any numeric digits, false if it doesn't..
rectanglealign(lborder,rect)		This function aligns a small rectangle inside of a larger rectangle. The small rectangle can be aligned in one of nine positions: top left, top center, top right, left center, center, right center, bottom left, bottom center, bottom right. If an alignment axis is not specified it is assumed to be center, for example top is the same as top center. The order of the options does not matter, for example top left and left top are the same. If two conflicting specifications are made left will override right and top will override bottom. Here is an example that moves the current window to a top centered position on the main screen. zoomwindowrectangle rectanglealign(info("screenrectangle"), info("windowrectangle"), "top center") .
removeprefix(text,prefix)		This function checks to see if a text item starts with a prefix. If it does, the prefix is removed.
removesuffix(text,suffix)		This function checks to see if a text item starts with a suffix. If it does, the suffix is removed .
serverdatabasename(database)		This function returns the name of the server associated with a shared database. If the database name is "" the server name for the current database will be returned.
serverdomain()		This function returns the domain name or IP address of a server. The server must be currently available.
servername(database)		This function returns the server database name associated with a shared database. This is the name of that database on the server (which may be different from the name on a client). If the database name is "" the server database name for the current database will be returned.
serverrunning()		This function returns TRUE if this copy of Panorama is running as a server, or false if it is running in single user or as a client.
servervariable(database,variable)		This function gets the value of a server variable (a permanent variable on the server). The server variable must already be set up with the SetServerVariable statement.
sharedusers(database)		This function returns a list of users that are currently sharing a database. The specified database must currently be connected to the server on this computer. If the database name is left blank then the current database is assumed. If the database name is * then all users on the server will be listed (the current database must be a shared database). If the database is not connected, or is not a sharable database, the result will be "". If this is a connected database the result will be a carriage return separated array, with each line containing the session id, user name, and user's computer's name separated by tabs.
subpath(folder,subfolder)		This function returns the path of a subfolder of a specified folder. For example the function subpath(info("systemfolder"),"Extensions") returns the path of the Extensions folder within the System folder.

Function	Reference Page	Description
superdatepattern(number, datepattern,timepattern)		Converts a number containing a superdate to text, allowing you to specify the patterns for both the date and the time portions. For example the formula superdatepattern(supernow(),"Month ddnth yyyy @ " ,"hh:mm am/pm") will result in something like July 10th, 2008 @ 1:36 PM.
superdatesecondsstr(number)		Converts a number containing a superdate to text in standard format including (for example 4/20/03 9:56:37 AM).
updatingwindow()		This function returns true if called from a procedure that is part of displaying an object in a form, otherwise it returns false.
weblink(url,caption)		This function builds an HTML link tag. If the caption is not supplied ("") the URL will be used for the caption.
weblinknewwindow(url,caption)		This function builds an HTML link tag. The link will open the new page in a new window instead of in the same window. If the caption is not supplied ("") the URL will be used for the caption.
webtagortext(text)		This function prepares text for display on a web page. Normally it converts all characters that need special handling into HTML entities. However, if the text begins with < and ends with > it is assumed to be raw HTML and is not converted. Any occurences of <> will be removed, so you can start and begin with <> if you want to embed tags in the middle of the text.
webtext(text)		This function converts a number or regular text into text that may be displayed on a web browser. As far as possible, any special characters are converted into HTML entities so that they will display correctly (for example accented characters like å, special characters like ©, and the < and > symbols). This function goes beyond the htmlencode function in that it also encodes <, >, and carriage return (converted into).
weekvalue(date)		This function extracts the week from a date as a numeric value (this is the number of weeks since the start of the year, 1 to 52).
xtag(tag,text)		This function generates an HTML/XML tag. For example the formula xtag("italic","hello world") will generate the text <italic>hello world</italic>.
xtagvalue(text,tag)		This function extracts the data from the first matching HTML or XML tag in the text. For example the formula xtagvalue(page,"title") will extract the page title from an HTML page.
yearvalue(date)		This function extracts the year from a date as a numeric value.

Remember the table above only lists about 20% of the new functions in Panorama 5.5 — see the **Programming Reference** wizard for the rest and for additional details on the functions listed above.

ObjectInfo(Function can now be used within an object

The **`objectinfo(`** function can now be used within an object on a form — in that case it will always refer to the object currently being drawn (an object could find out its own rectangle, color or font, for example).

Time Code Calculations (Video/Film)

Panorama user and visual effects supervisor Chris Watts has built an asset management system that he has used in the production of several major motion pictures, including **Pleasantville** and **300.** In the process Chris has developed about a dozen Panorama functions that are useful in performing calculations with timecodes used in video and film, which he has donated so that other Panorama users can take advantage of them. For more information on these function see "Time Code Calculations (Video/Film)" on page 117 of *Formulas & Programming*.

Chapter 24: Procedures

Panorama 5.5 contains over 130 new statements. For a complete list use the **Version** menu in the **Programming Reference** wizard. Other changes to procedures are listed below.

Call a Procedure within a Formula

If you are a programmer then one of the most exciting new features in Panorama 5.5 is the new `call(` function that allows a procedure to be used within a formula. That's right, you can now use a procedure to calculate values on-the-fly in formulas in applications like Text Display objects, within formula fills, Live Menus, etc. Anywhere Panorama can use a formula you can now use a procedure. To learn all the details see "Using a Subroutine in a Formula (the CALL(function)" on page 273 of *Formulas & Programming*.

Single Stepping

Previous versions of Panorama did not always single step correctly through procedures that used the `select` or `find` statements. Panorama is supposed to work differently when these statements fail to match any data (see "Handling Empty Selections" on page 558 of *Formulas & Programming*) but it would not. Panorama 5.5 fixes this issue so that you single stepping now works exactly like running the procedure at full speed in these situations.

The FormulaValue Statement

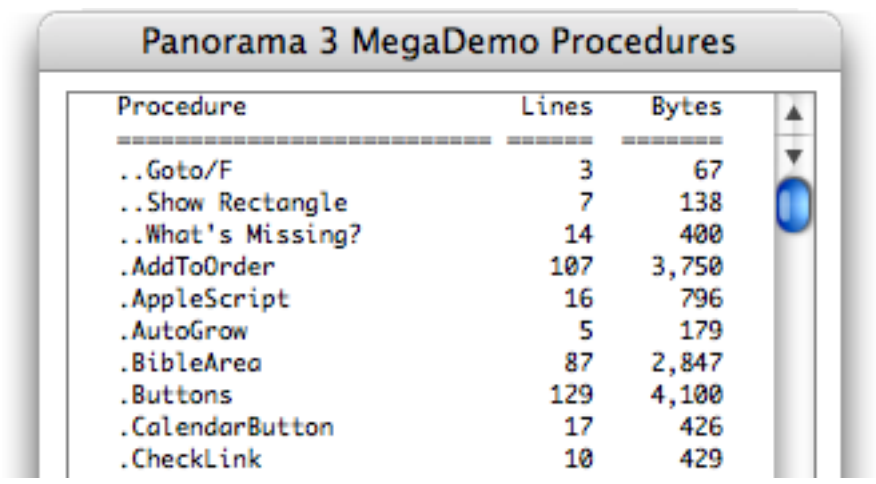
The new `formulavalue` statement is similar to a `set` statement but gives you more control over how errors are handled, allows you to specify the formula using a variable, and allows you to specify what database is to be used for the calculation. It's especially useful for processing and validating user supplied formulas. For more inrformation see "The FormulaValue Statement" on page 245 of *Formulas & Programming*.

Custom Statement Wizard

This wizard has a new menu — **Recent**. The **Recent** menu contains a list of all of the custom statements that you have recently looked at (opened the procedure window). To quickly re-open one of these statements simply choose it from the **Recent** menu. See "The Custom Statements Wizard" on page 291 of *Formulas & Programming* to learn more about this wizard.

Displaying Procedure Statistics

The **View** wizard can now display statistics (number of lines and bytes in the source code) for all of the procedures in the current database.



See "Displaying Source Code Statistics" on page 347 of *Formulas & Programming* for more information.

Exporting and Importing Procedures

The **View** wizard can now export and import procedures via text files. See "Exporting and Importing Procedure Source Code" on page 348 of *Formulas & Programming* for more information.

Suppressing Display of Text and Graphics

Panorama 5.5 now allows you to explicitly force the redisplay of the tool palette with the following statement:

```
showother «»,96
```

This feature was actually in earlier versions but was not documented. See “ShowOther field,code” on page 310 of *Formulas & Programming* for more information.

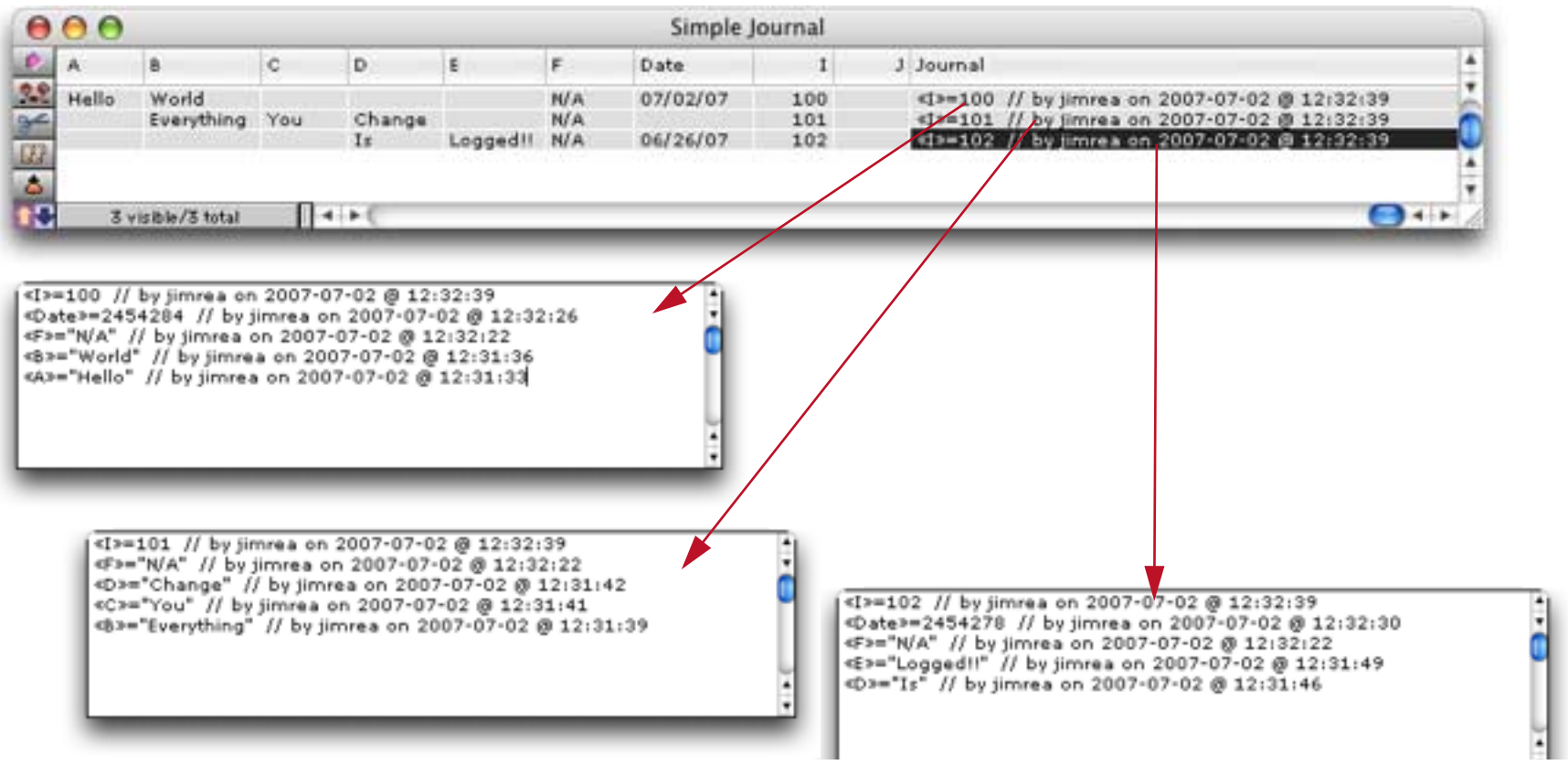
Another new feature is the ability to check whether display is currently suppressed or not. See “Checking NoShow Status” on page 310 of *Formulas & Programming* for details.

.ModifyFill Procedure

This new hidden trigger procedure complements the .ModifyRecord procedure and allows all user modifications to a database to trigger a procedure. See “.ModifyFill” on page 384 of *Formulas & Programming* of the **Panorama Handbook** for more information.

Logging Changes (Audit Trail) with .ModifyRecord, .ModifyFill and info("modifiedfield")

It’s now possible to create a log or audit trail of all changes made to a database, so you can see who changed what when. The **Simple Journal** example database included with Panorama illustrates this.



As the diagram above shows the Journal field contains a log of all changes made to the database. This log is created by the **.ModifyRecord** and **.ModifyFill** procedures in the database — See “Logging Changes (Audit Trail) with .ModifyRecord, .ModifyFill and info("modifiedfield")” on page 385 of *Formulas & Programming* for more information.

Event Handler Procedures

A new function, info("runninghandler"), allows a procedure to find out if it is running as an event handler procedure or as a normal procedures. See “Event Handler Procedures” on page 394 of *Formulas & Programming* for more information.

Chapter 25: Programming Techniques

The following updates have been made to this chapter.

Using Formulas to Work with Really Big Data Files

Panorama 5.5 has changed the way formulas and memory interact. If you are using the **fileload(** function and/or the **filesave** statement to work with really large amounts of data (more than 2 or 3 megabytes) you'll need to make a simple change to your procedures. To learn more see "Reading Really Big Data Files" on page 423 of *Formulas & Programming*.

Changing a Window's Size

The new **sizewindow** statement changes the size of a window without moving it. This statement is less "flashy" than the **zoomwindow** statement. See "Changing a Window's Position/Options" on page 450 of *Formulas & Programming*.

New Map Features

The **savewebmap** statement has new options for controlling the map size and scale (see "Generating Map Images" on page 617 of *Formulas & Programming*), while the new **openmapwindow** statement allows you to add an interactive map interface to any database with just one line of code.

See "Adding an Interactive Map Interface to a Database" on page 619 of *Formulas & Programming*.

Working with JPEG Images

Panorama 5.5 has new statements specifically for working with JPEG images. See "Working with JPEG Images" on page 695 of *Formulas & Programming*.

Taking an iSight Snapshot

If your MacOS X computer has an **iSight** camera a Panorama procedure can take a picture and save it to disk. See "Taking an iSight Snapshot" on page 696 of *Formulas & Programming*.

Buttons Within Matrix Cells

Panorama 5.5 has a new statement, **matrixbuttonhelper**, that makes it easier to implement buttons within matrix cells. See “Buttons Within Matrix Cells” on page 956 of the *Panorama Handbook*. Another new feature is the "xytoxy" command for super matrix objects, see “Super Matrix SuperObject™ Commands” on page 715 of *Formulas & Programming*.

Selecting a Printer

When running on OS X Panorama 5.5 allows you to find out what printers are available, what printer is currently selected, to change the current printer and to change the default printer for one or more forms. See “Selecting a Printer” on page 722 of *Formulas & Programming*.

Printing Directly to a PDF File

If you are using Mac OS X and have the **CodePoetry CUPS-PDF Package** installed your procedures can use the **printpdf** statement to print directly to a PDF file with no user intervention (no dialogs appear, etc.). See “Printing Directly to a PDF File” on page 727 of *Formulas & Programming* for details on using this statement and for instructions on how to install the CUPS-PDF Package.

Modifying Procedures With a Procedure!

Panorama 5.5 allows you to modify procedures with a procedure. A procedure can now modify existing procedures and create new ones. See “Accessing and Modifying Procedures” on page 736 of *Formulas & Programming*.

Chapter 26: Working with Alternate Programming Languages

This chapter used to be named “AppleScript”, but has been renamed to reflect Panorama 5.5’s more general capabilities for working with other programming language. Panorama 5.5 now supports six alternate languages.

Language	Summary	Links
AppleScript	AppleScript is an English-like language used to write script files that automate the actions of the computer and the applications that run on it. Using AppleScript you can write a program that works across multiple applications.	`http://www.apple.com/applescript/`
UNIX Shell Scripts	A shell script is a script written for the UNIX shell, or command line interpreter. Typical operations performed by shell scripts include file manipulation, program execution, and printing text, Shell scripts allow several commands that would be entered "by hand" at a command line interface to be executed automatically and rapidly.	`http://developer.apple.com/documentation/Darwin/Reference/ManPages/`
Perl	The Perl language was developed by Larry Wall. An ancient language by computing standards, the first version of Perl was contributed to the alt.comp.sources usenet newsgroup in 1987. Perl was originally developed for text processing, but today some people refer to it as "the duct tape of the internet" because it has proven itself to be so practical as a general purpose language.	`http://www.perl.org/`
Ruby	Ruby is a scripting language developed by Yukihiro Matsumoto in Japan in the early 1990's. It is a pure object-oriented language, so everything—every string, every number, every user-defined data type—all of these are objects in Ruby.	`http://www.ruby-lang.org/en/`
Python	Python is a general purpose scripting language developed in the early 1990's. It fully supports object-oriented programming, so you can create classes, virtual methods, use inheritance (including multiple inheritance), and operator overloading.	`http://www.python.org/`
PHP	Like Perl, PHP was not originally an object oriented languages but object oriented features have been added in recent releases. PHP is especially popular for use on web sites — it is commonly integrated with the popular Apache web server but can also be used separately.	`http://www.php.net/`

See “Working With Alternate Programming Languages” on page 745 of *Formulas & Programming* for more detail.

Returning Values from an AppleScript

Another new feature in Panorama 5.5 is the ability of a Panorama procedure called in response to an AppleScript to return a value to the AppleScript. This is done with the **`SetAppleEventValue`** statement.

```
setappleeventvalue value
```

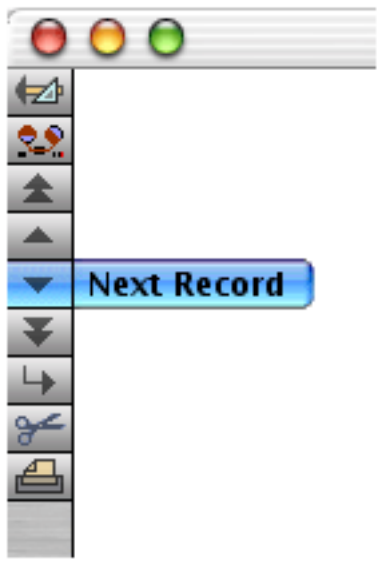
“Transferring a Value Back From Panorama to the AppleScript (Returning a Value)” on page 792 of *Formulas & Programming* to learn more about this feature.

Version 5.0.0

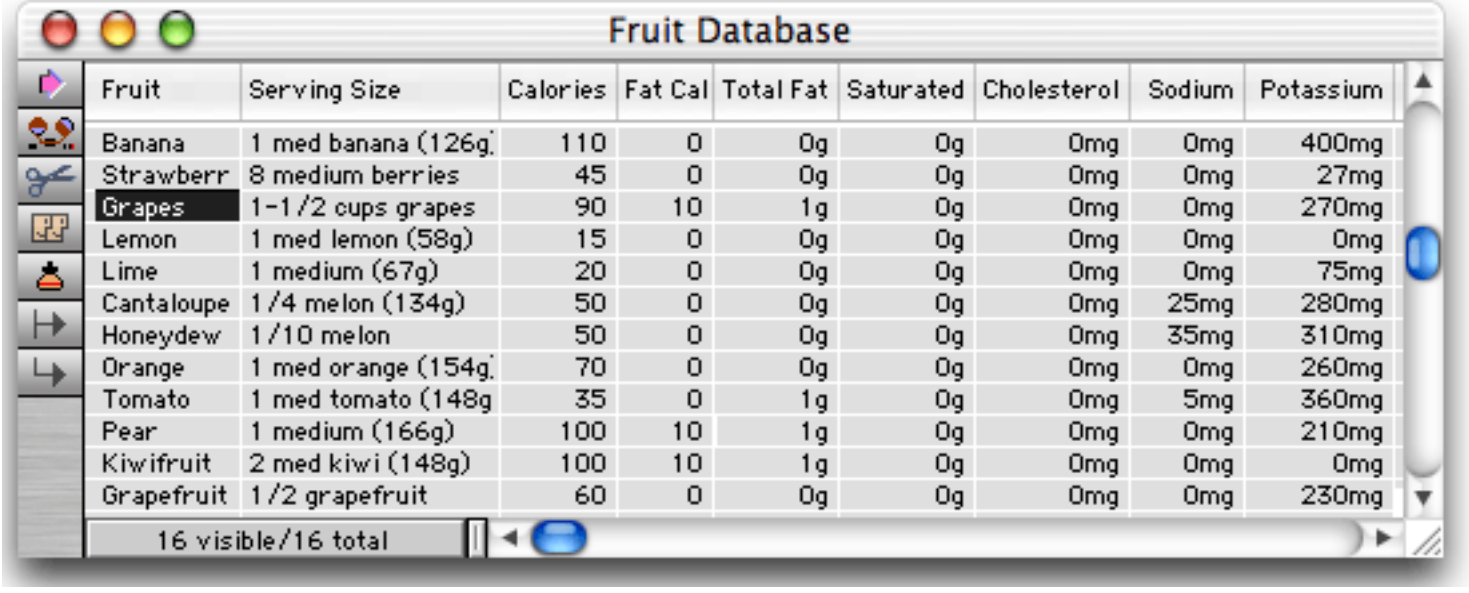
This version was released in December 2004 for Mac OS 9 and Mac OS X. This is the first version of Panorama that supports OS X in native mode, and also includes many new features.

Updated User Interface

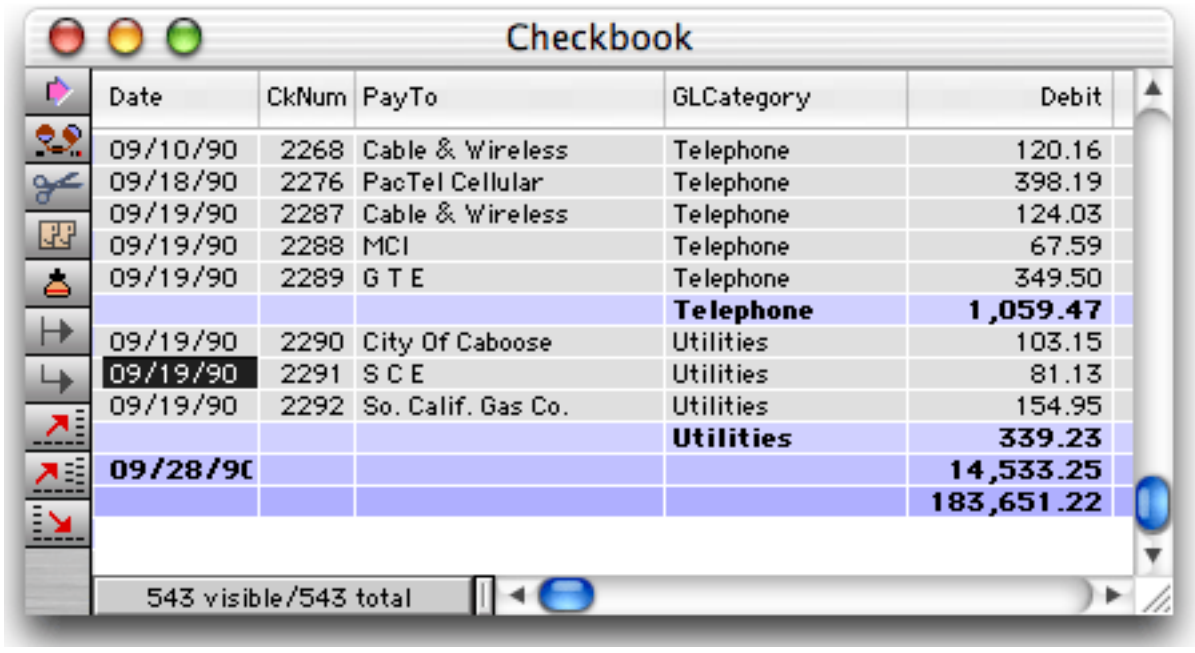
For Panorama V we've substantially upgraded the look and feel of the program. These changes were made primarily for OS X, but OS 9 and Windows users will see big improvements as well. All of the old "System 6" black and white interface elements have been eliminated. We started with a new look for the tool palette.



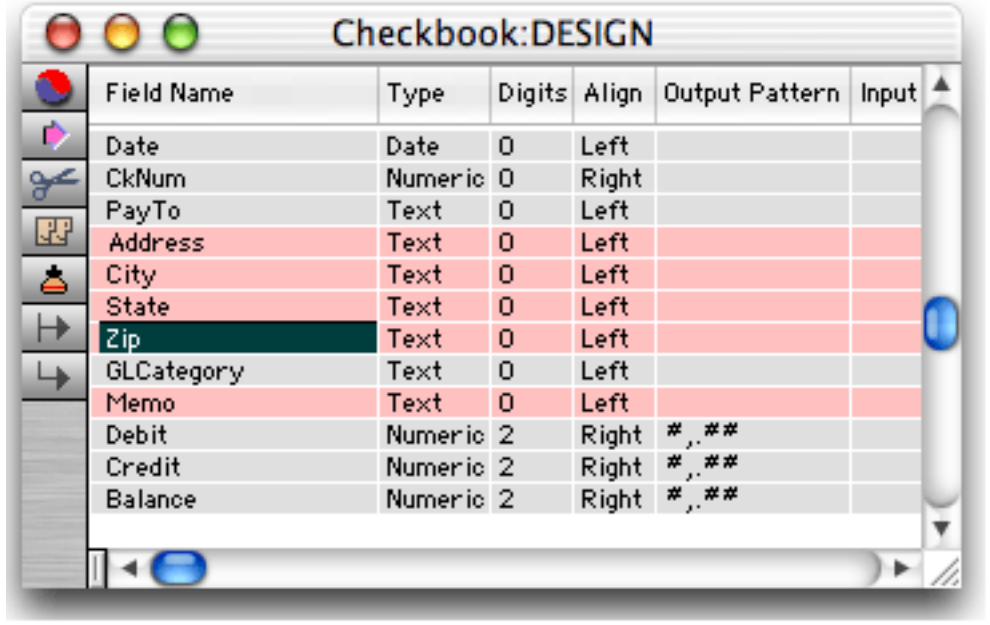
The data sheet is no longer a plain white but has a new "3D" look. The exact appearance varies depending on whether you are using OS X (shown below), OS 9 or Windows.



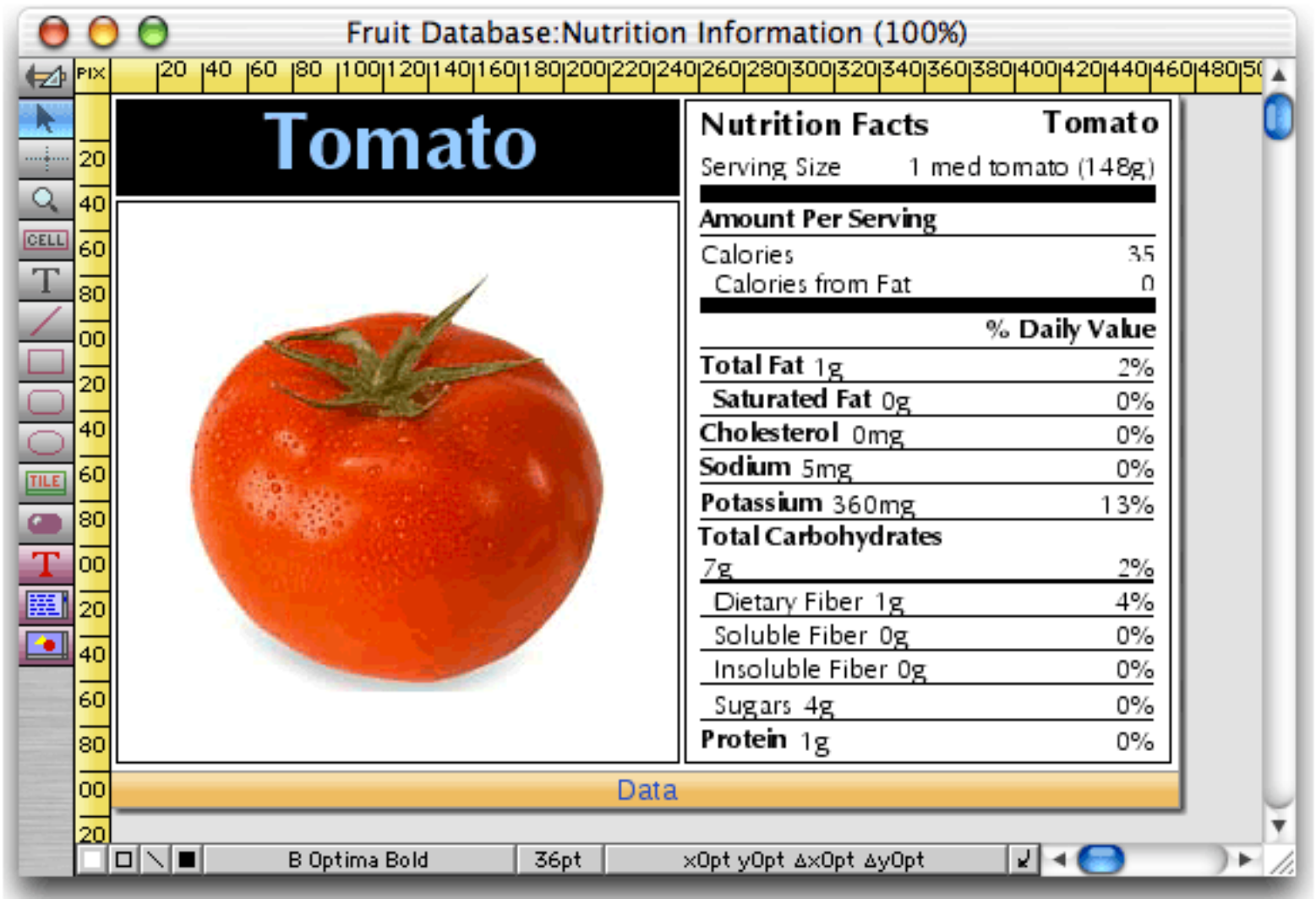
We've also changed the way summary records are handled. Instead of appearing with a + sign on the left hand edge, summary records appear with a colored background. The darkness of the color changes depending on the summary level (see below). You can no longer click on the left edge of the record to toggle a data record into a summary record, but this feature is still available in the Sort menu.



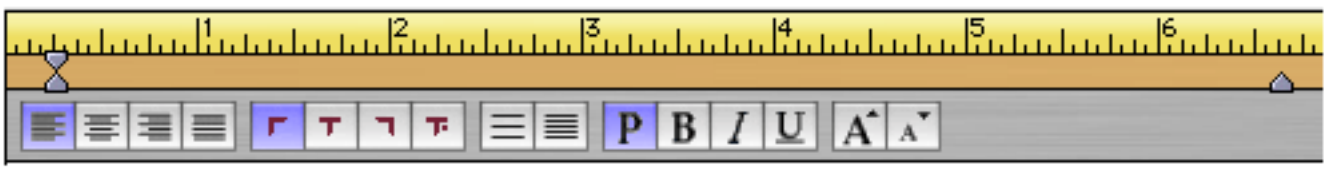
The design sheet shares the same 3D look as the data sheet, with a slight twist. Fields that already exist are shown with a gray background, while new fields appear with a pink background until you use the **New Generation** tool.



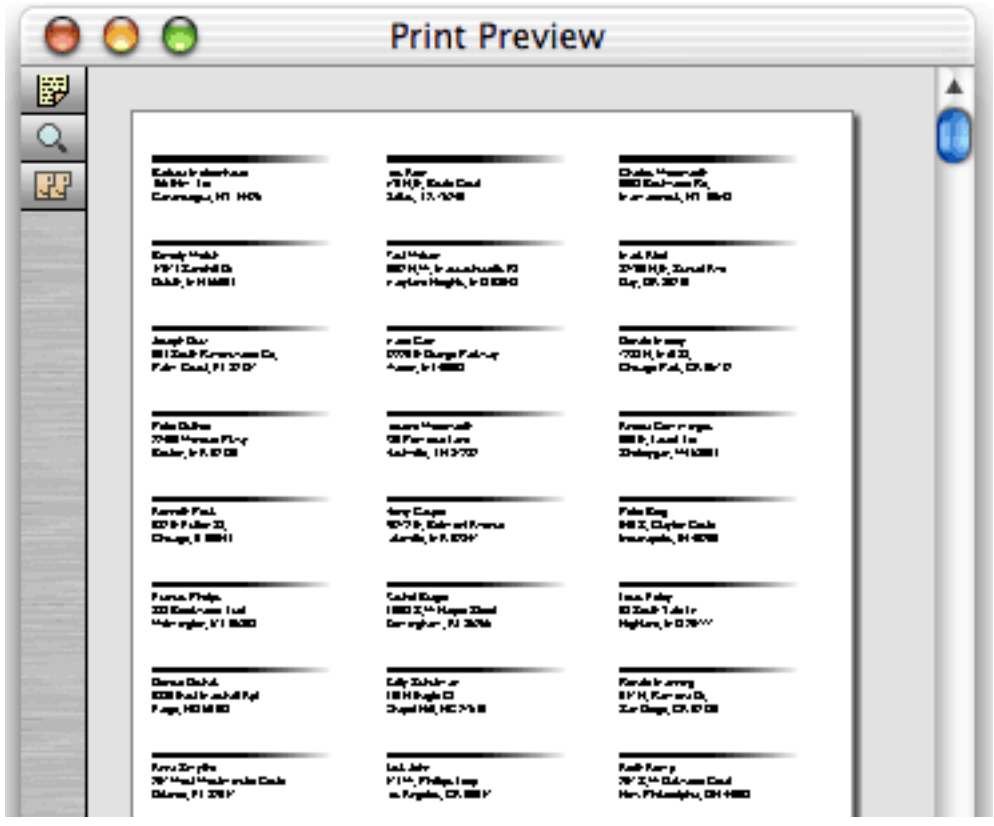
Graphics mode is basically the same, but with a cooler 3D look to rulers, tiles and the graphics control strip.



We've also upgraded the look of the word processor superobject, as well as adding several new tools for changing styles (plain, bold, italic, and underline) and font size (increase / decrease).



The print preview window has been revised to more accurately show the appearance of the printed page, including the margins and borders.

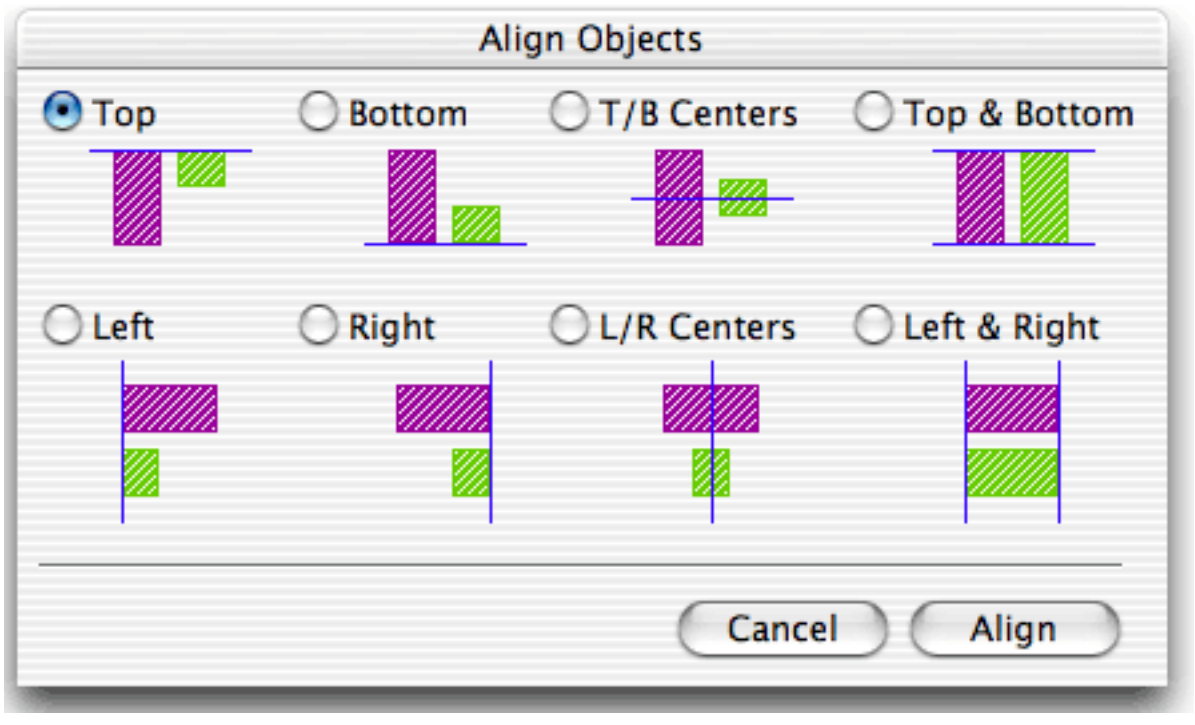


Improved Dialogs and Alerts

The biggest improvement in this area is that all dialogs and alerts are now movable and non-modal. This also means that even under OS 9 you can now click to another application when you are in a dialog.

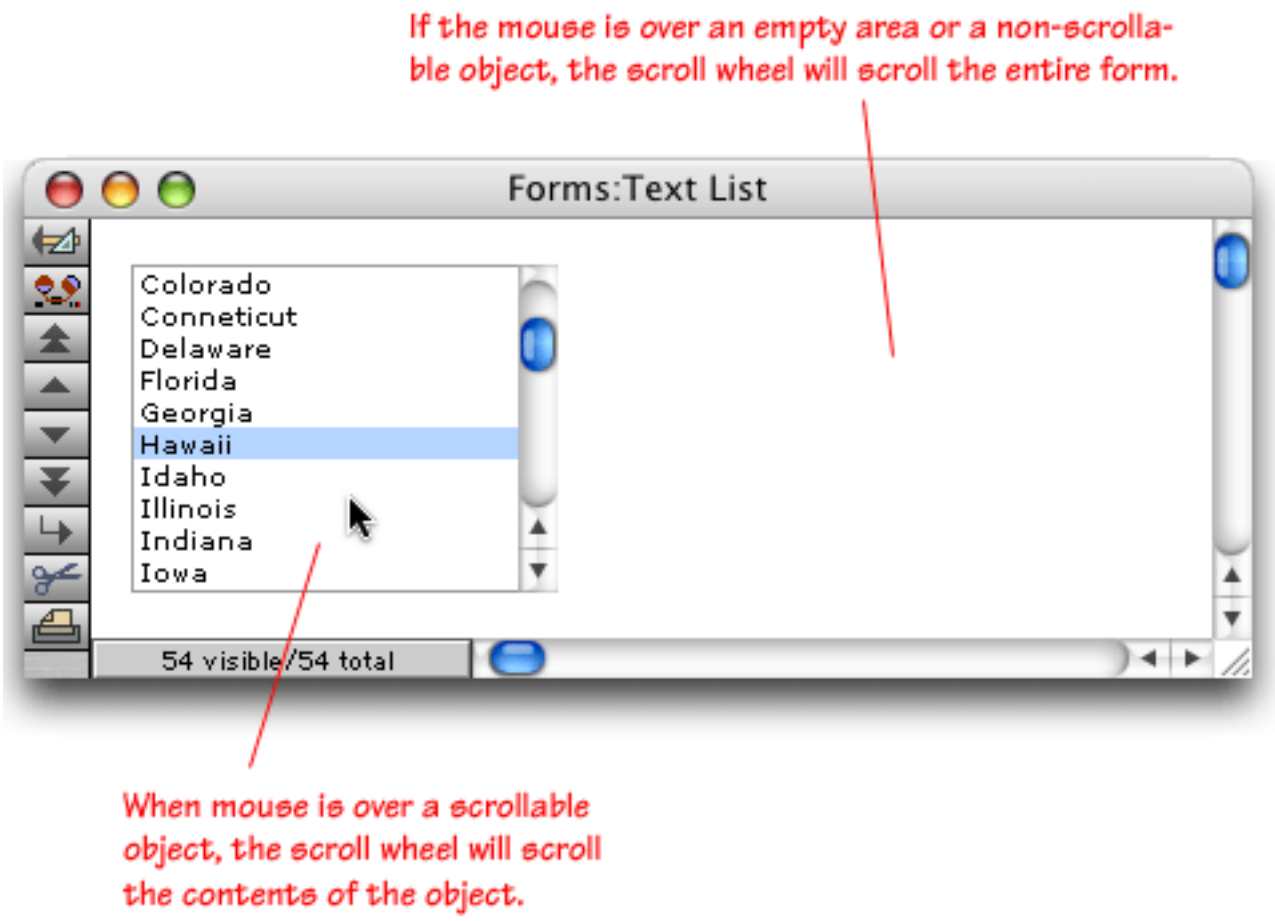
In earlier versions of Panorama all of the dialogs were designed to fit on the original "classic" Macintosh 512 by 384 pixel screen. This made for some very cramped and cluttered dialogs. All 140 of Panorama's dialogs have been redesigned for a cleaner, more organized appearance. (As a side effect, Panorama now requires a screen size of at least 600 by 800 pixels.)

A few dialogs deserve special mention. The **Find/Select** dialog no longer expands and shrinks - it always appears at its full six line maximum height. The **Tool Configuration** dialog (graphics mode) now has half a dozen buttons for setting up the most popular tool configurations. The **Align Objects** dialog now shows a graphical representation of each selection. You can double click on any of the options to align the currently selected objects.

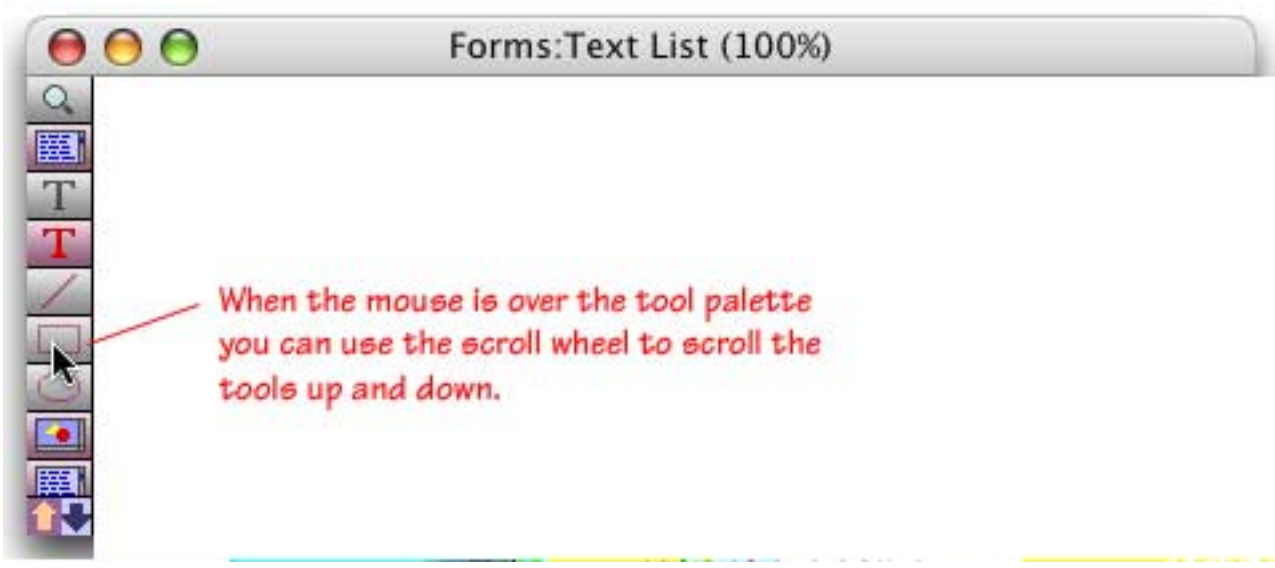


Scroll Wheel Support

When using OS X Panorama now supports scroll wheels. The scroll wheel works everywhere, including dialogs. In a form window, moving the mouse over an object with a scroll bar and turning the wheel will scroll that object instead of the entire window. (This includes the Text Editor, Text Display, Super Flash Art, List, Word Processor, and even the new scrolling Super Matrix object).



If the window is short you can move the mouse over the tool palette and scroll the tools with the wheel.

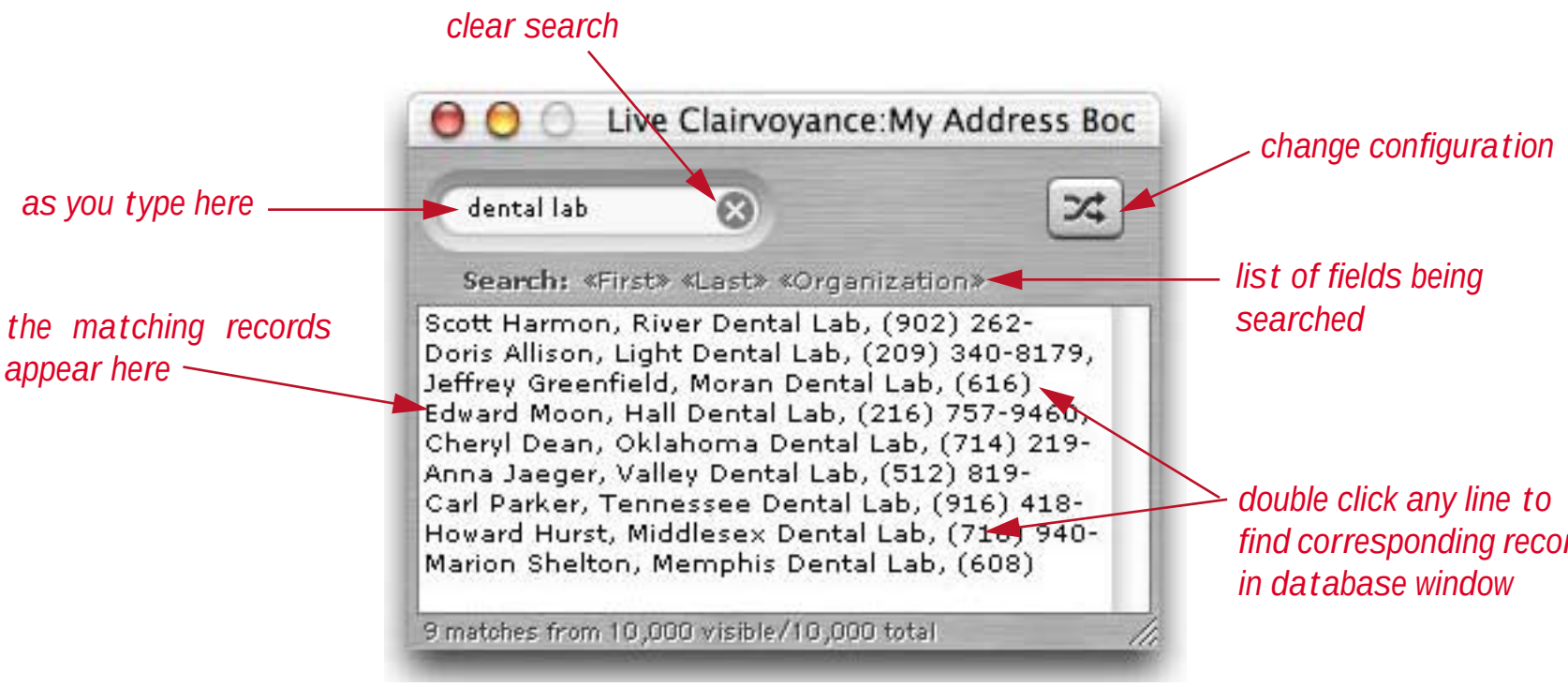


Wizards

Panorama V includes dozens of new wizards and many of the existing wizards have been improved. To make it easier to use the large number of wizards now available, the Wizard menu has been divided into sub-menus.

Live Clairvoyance™ Wizard

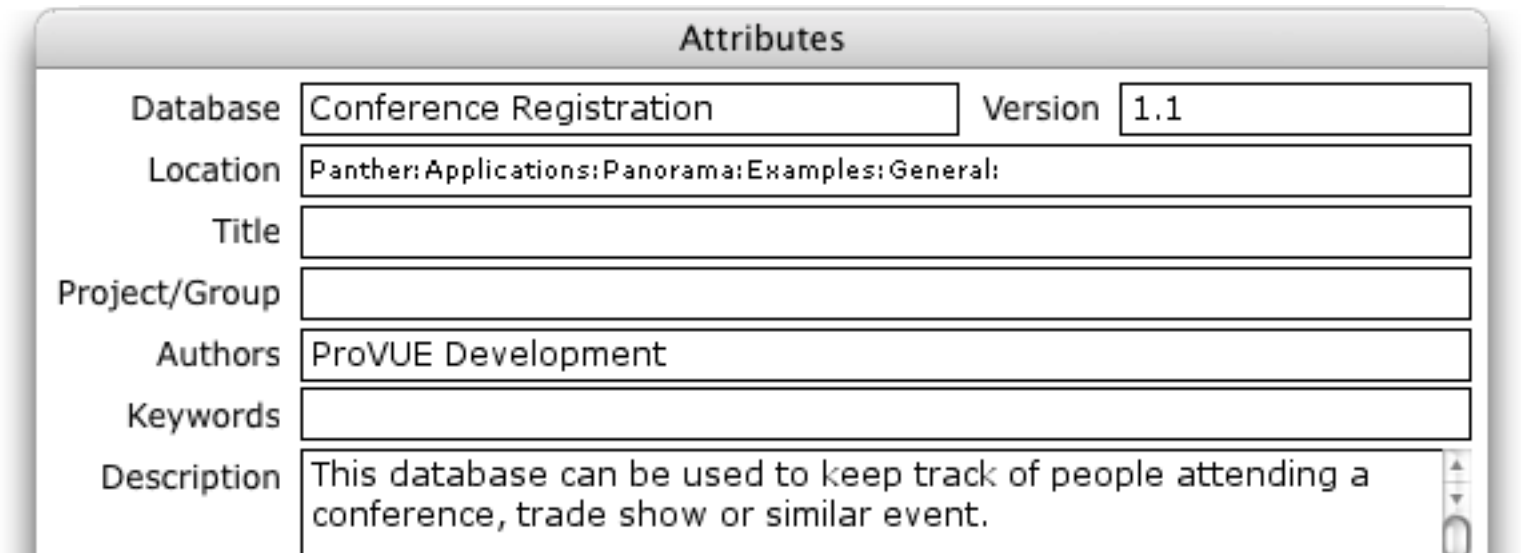
The **Live Clairvoyance**™ wizard allows you to perform “live” searches on any Panorama database. The search results are updated dynamically as you type, allowing you to “hone in” on just the information you are looking for. The search may include multiple fields or even all fields in the database being searched. (If you've used the search box in iTunes you'll find the operation of this wizard familiar.) Using the Live Clairvoyance wizard doesn't require you to do any programming or make any modifications to your existing databases. Here’s what this wizard looks like in action.



This wizard has been superseded in Panorama 6.0 and later by the new Find/Select dialog. To learn more about this wizard see the older Panorama 5.5 Handbook.

Database Information Wizard

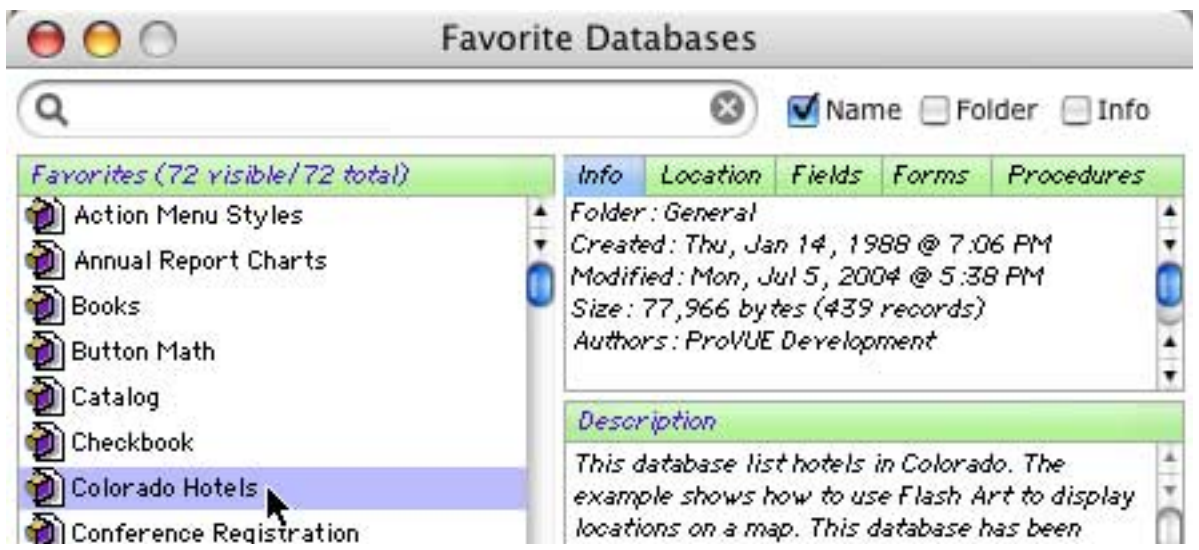
The **Database Information** wizard allows you to view and modify descriptive information about any database: the title, author name, description, keywords, etc.



To learn more about this wizard see "Viewing and Modifying Database Metadata" on page 73 of the *Panorama Handbook*.

Favorite Databases Wizard

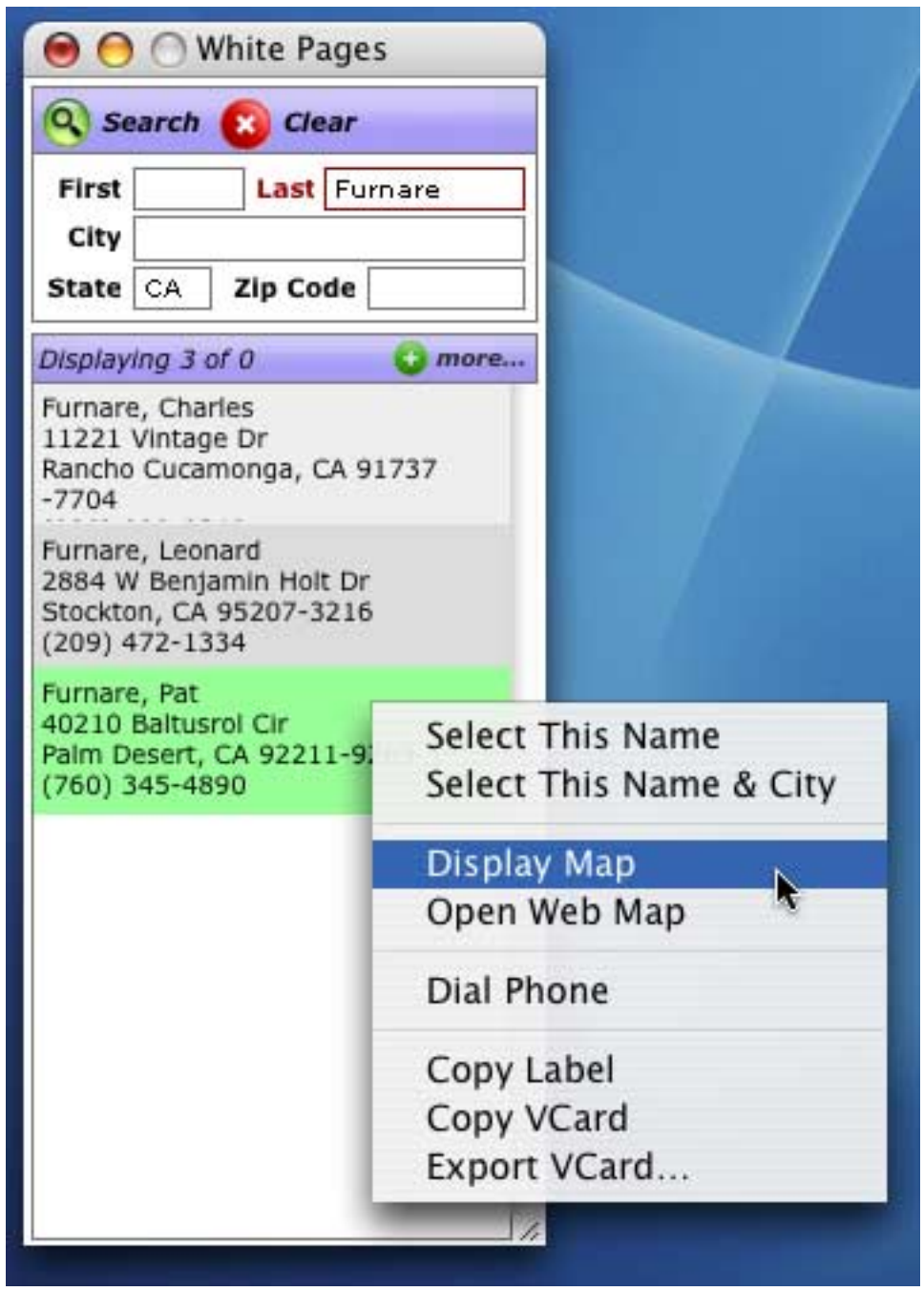
The **Favorite Databases** wizard keeps your favorite databases organized and at your fingertips. This wizard has been completely rewritten with a new streamlined interface.



This wizard has been deprecated in Panorama 6.0 and later. To learn more about the new version of this wizard see the older Panorama 5.5 documentation.

White Pages Wizard

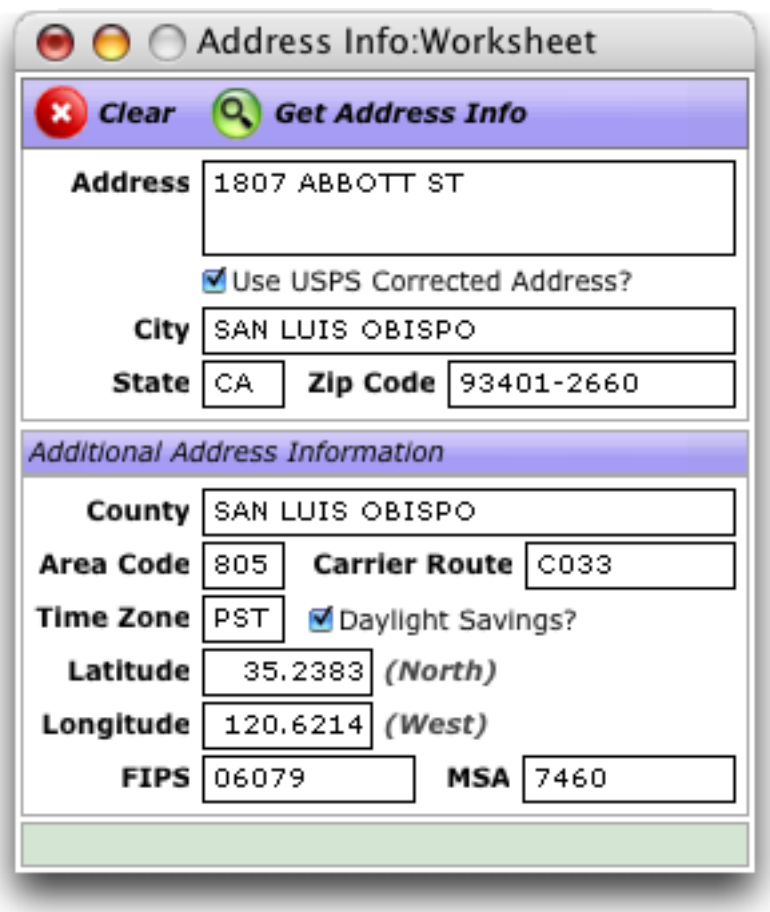
The **White Pages** wizard allows you to look up addresses and phone numbers. To search for a person, enter as much information as you know. The wizard will display any people that match the criteria you have entered. In this case there are three people named Furnare in California. Once you have located the person you want you can automatically dial them, display a map, or transfer the data to other VCard savvy applications and databases.



For more details about this wizard see "White Pages" on page 73 of *Wizards & Demos*.

Address Info Wizard

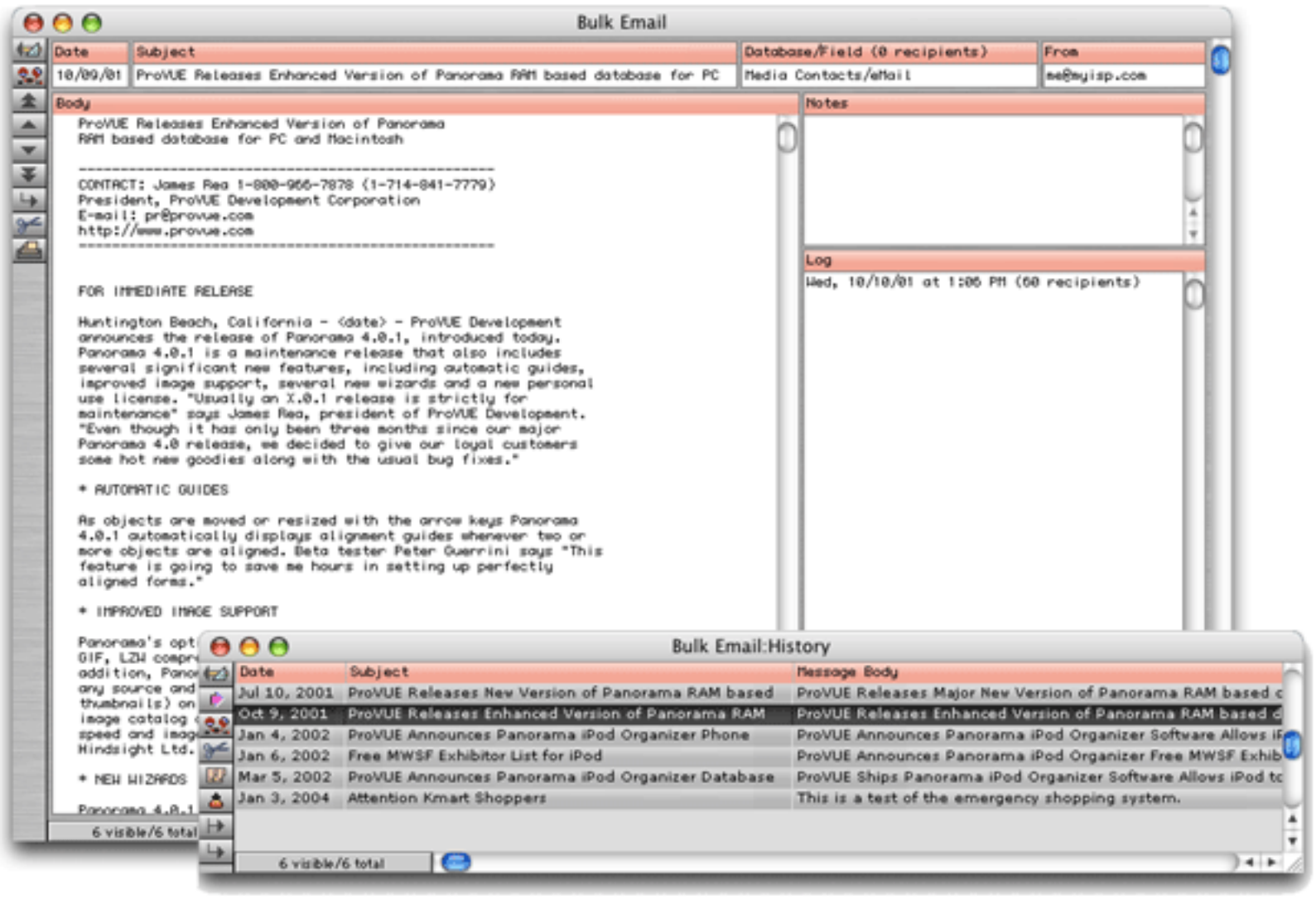
This wizard gets information about US addresses and zip codes. If you enter a zip code it will display information about that zip code, including the city, state, county, area code, time zone, latitude and longitude, and the FIPS and MSA code. If you enter a full address it will display the zip+4 code and carrier route, and will check to make sure that this is a valid address according to the US Post Office.



For more details about this wizard see “Address Info” on page 65 of *Wizards & Demos*.

Bulk Email Wizard

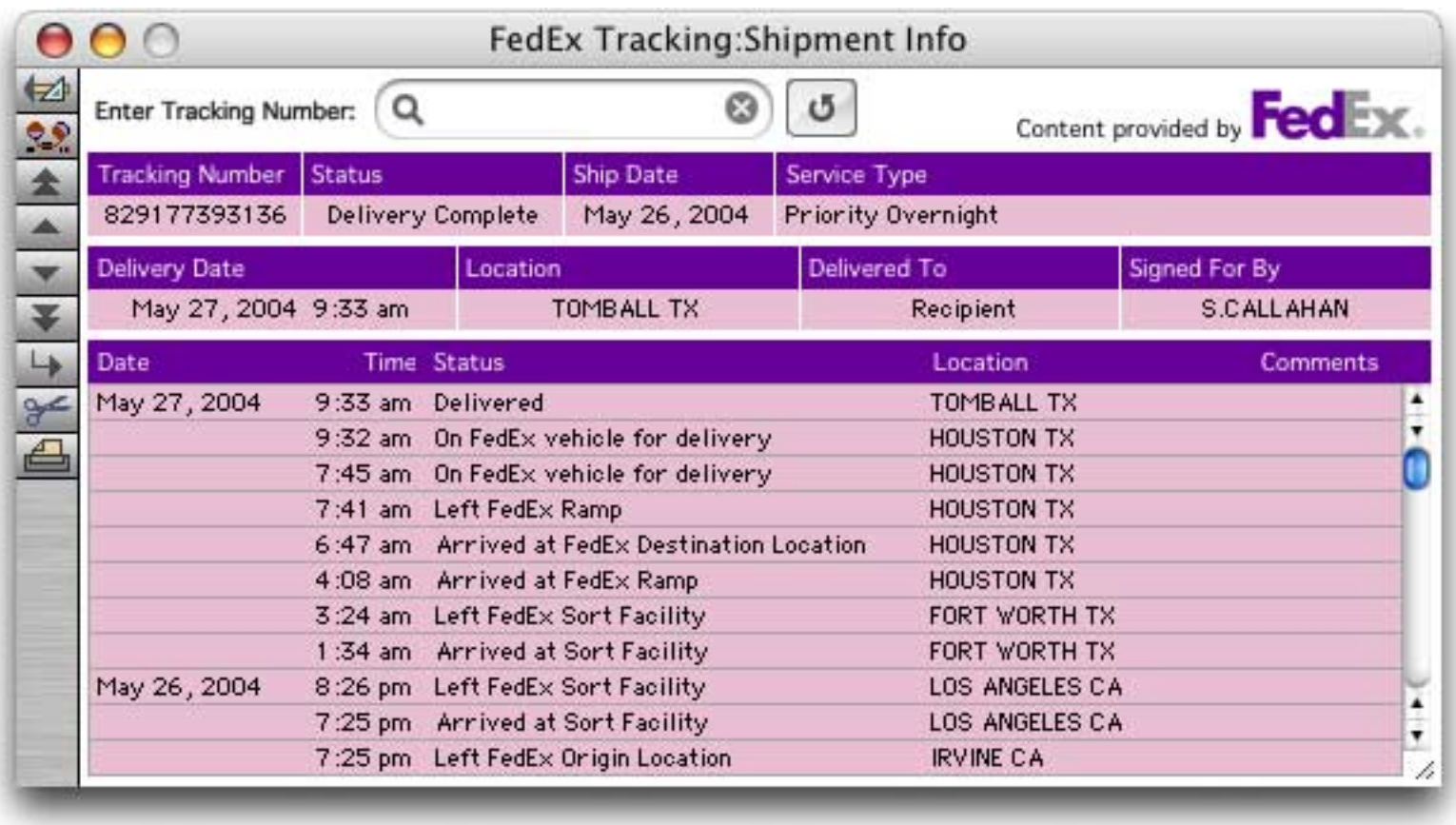
The **Bulk Email** wizard makes it easy to send and keep track of bulk emails. It keeps all of the previous e-mails you've sent organized, and can automatically extract e-mail addresses from one or more other databases. The wizard has two primary windows. The Bulk Email window displays a single e-mail message, and allows you to configure and modify that message. The History window displays a list of the previous e-mails.



For more details about this wizard see “Bulk Email” on page 68 of *Wizards & Demos*.

FedEx Tracking Wizard

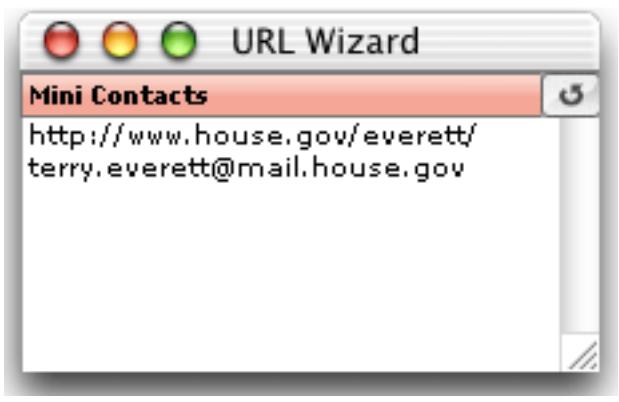
This wizard helps you track the progress of FedEx shipments. To use this wizard just enter the tracking number. The wizard will display the status of this package. In this case the package has been delivered to the recipient.



For more details about this wizard see “Fedex Tracking” on page 70 of *Wizards & Demos*.

URL Wizard

The **URL Wizard** scans all of the fields in the current record looking for URL's (web and e-mail addresses). If it finds any, you can double click on them to open the corresponding web page or create a new e-mail message.



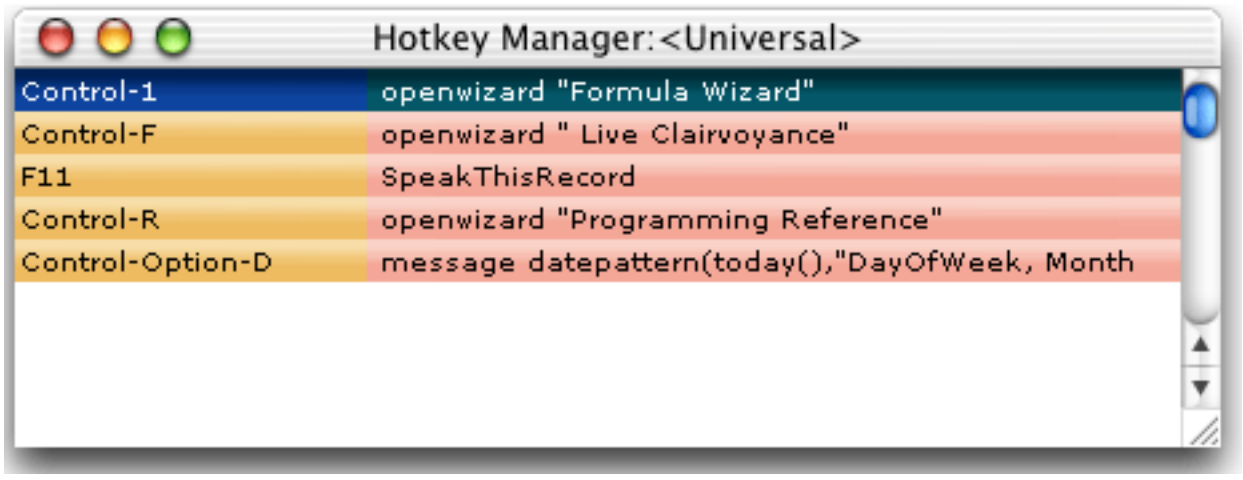
To learn more about this wizard see “URL Wizard” on page 71 of *Wizards & Demos*.

Mini Correspondence Wizard

This wizard now understands VCards, so you can transfer address to or from your correspondence and other VCard savvy databases and applications. See “Using the Mini Correspondence Wizard” on page 727 of the *Panorama Handbook* for details.

Hotkey Manager Wizard

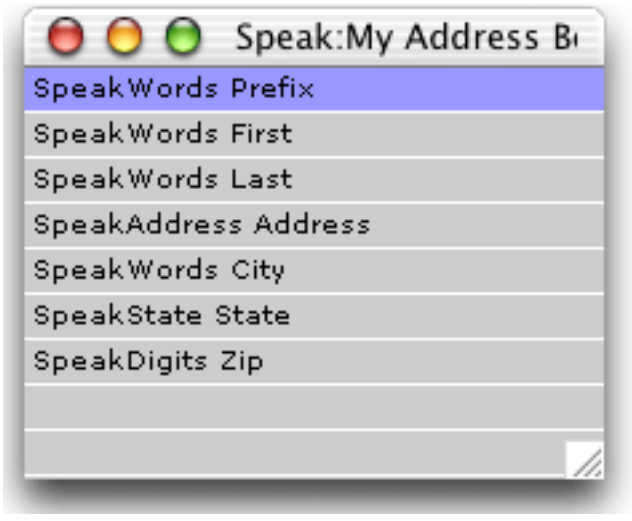
The **Hotkey Manager** Wizard allows you to set up database actions that will occur in response to different keystrokes and keystroke combinations. Each hotkey may be set up as a universal hotkey (active for all databases) or it may be made specific to a particular database.



To learn more about this wizard see “URL Wizard” on page 71 of *Wizards & Demos*.

Speech Wizard

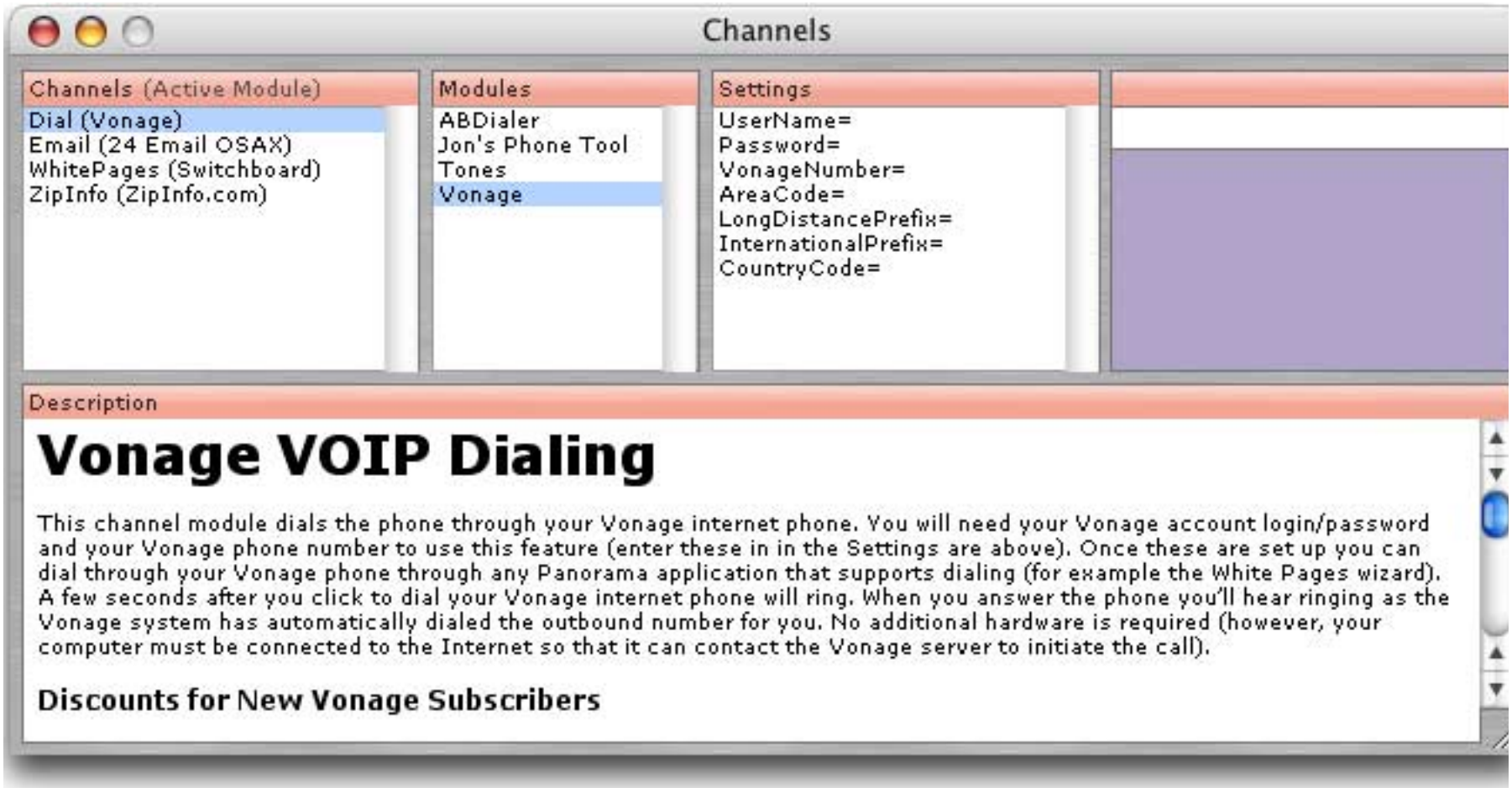
The **Speech Wizard** allows you to add speech synthesis to any Panorama database on a Macintosh computer. This allows Panorama to “read back” your data for voiced based data verification and auditing. Using a simple user interface you can create one or more scripts for reading back the information in a database.



To learn more about this wizard see “Speech Wizard” on page 103 of *Wizards & Demos*.

Channels Wizard

Some Panorama applications require a connection between Panorama and an external program or resource. Panorama allows you to set up **channels** as a conduit between Panorama and the external resource. For example, suppose you have a Panorama application that needs to send an e-mail. To do that Panorama will need to make a connection with the Internet software installed on your computer. Panorama uses the **Channel Wizard** to configure that connection.



Panorama has special programming statements that take advantage of the connections set up by different channels. The table below lists each type of channel and the statements that use the connection set up by that channel.

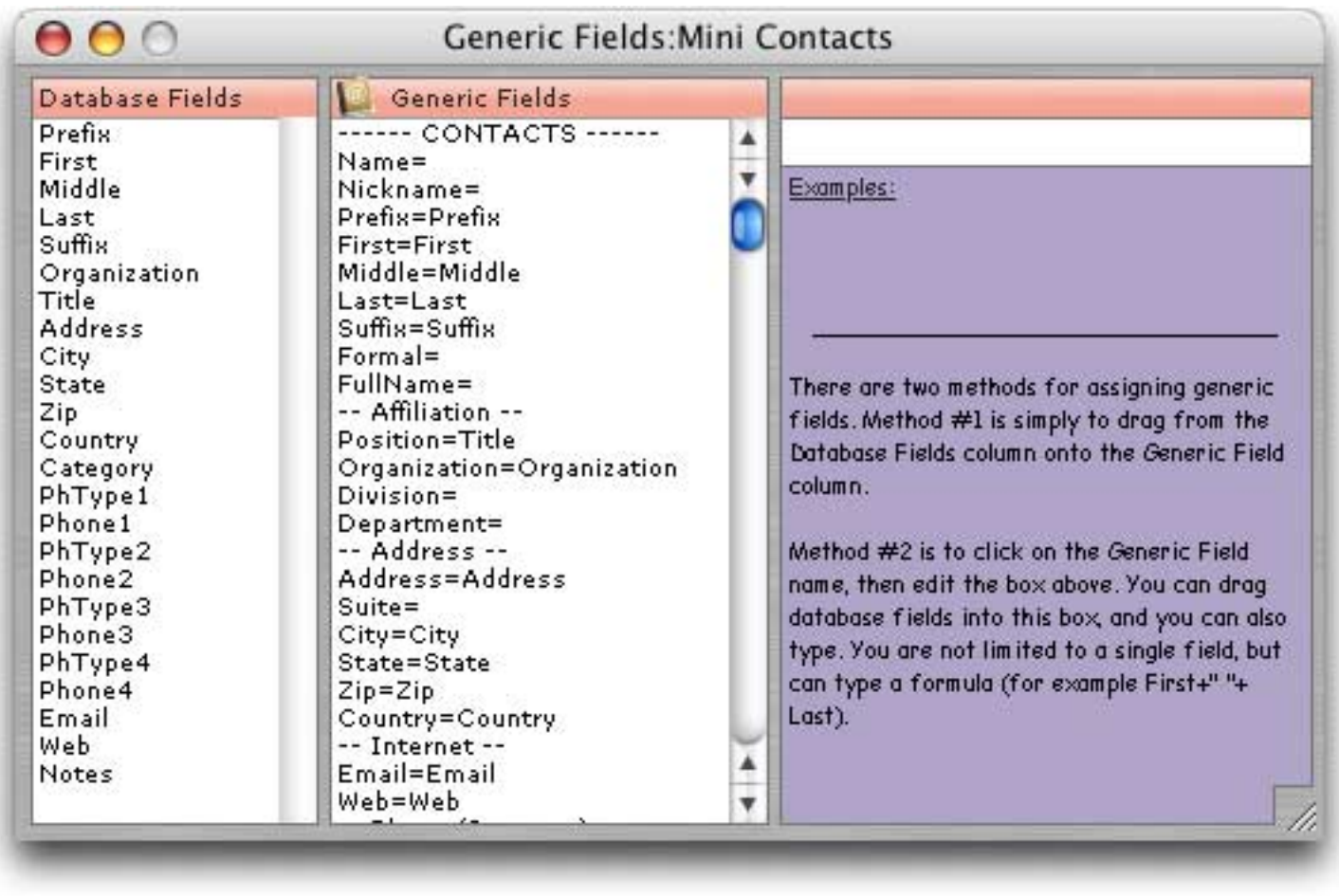
Channel	Statement	Description
Dial	dialphone	Dial the phone, adding prefixes, country codes and area codes as needed.
	dialdigits	Dial one or more digits exactly as specified.
Email	sendoneemail	Send a single e-mail message.
	sendbulkemail	Send multiple e-mail messages, one for each currently selected record in a database.
	sendarrayemail	Send multiple e-mail messages, one for each element of an array.
	sendemail	Send multiple e-mail messages with advanced options.
WhitePages	querywhitepages	Look up a person's address and / or phone number.
ZipInfo	zipinfo	Look up information about a zip code — city, state, county, area code, time zone, etc.

The exact operation of each of these statements will vary depending on how you have configured each channel. For example, the **`dialphone`** statement can dial the phone by creating tones on your computer speaker, by using a modem, with a bluetooth connection, or over the Internet if you have a Vonage phone. The programmer that writes the **`dialphone`** statement into his or her program doesn't know or care how the dialing is actually performed, he or she relies on the channel to do that job for them.

Even if you are not a programmer you may find that channels are useful for you. Many of the wizards and sample databases that are included with Panorama are already programmed to use channels. For example, the **White Pages** wizard can automatically dial the phone using the Dial channel, while the **Bulk Email** wizard will send mass e-mails using the Email channel. All you need to do is configure the channels for your needs and this wizards will be ready to go. To learn more about channels see "Channels" on page 93 of *Wizards & Demos*.

Generic Fields Wizard

Databases come in all sizes and shapes. Generic fields allow different databases to share information even if they have different field names or slightly different configurations. For example, one database may store company names in a field named Company, while another may have a similar field named Organization. By setting up generic fields for each database, you build a bridge so that Panorama knows that these two fields, though named differently, contain the same type of information. Once this bridge is built Panorama can exchange data between these two databases (for example by drag and drop), and between Panorama and other applications that can share information (for example Apple's *Address Book*). Panorama includes a special wizard for setting up generic fields for any database that contains contact information.

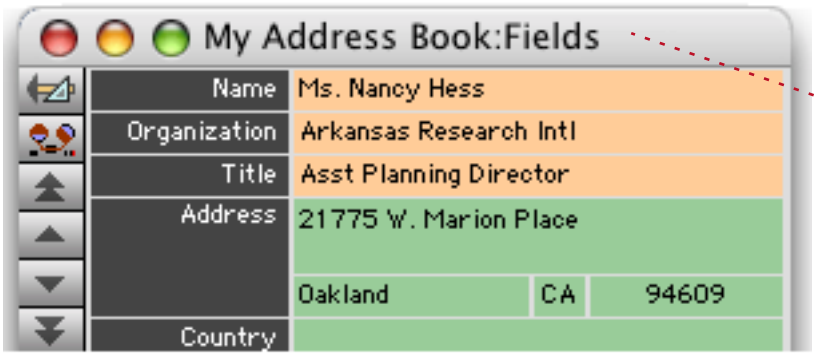


To learn how to use this wizard see "'Generic" Fields" on page 230 of the *Panorama Handbook*.

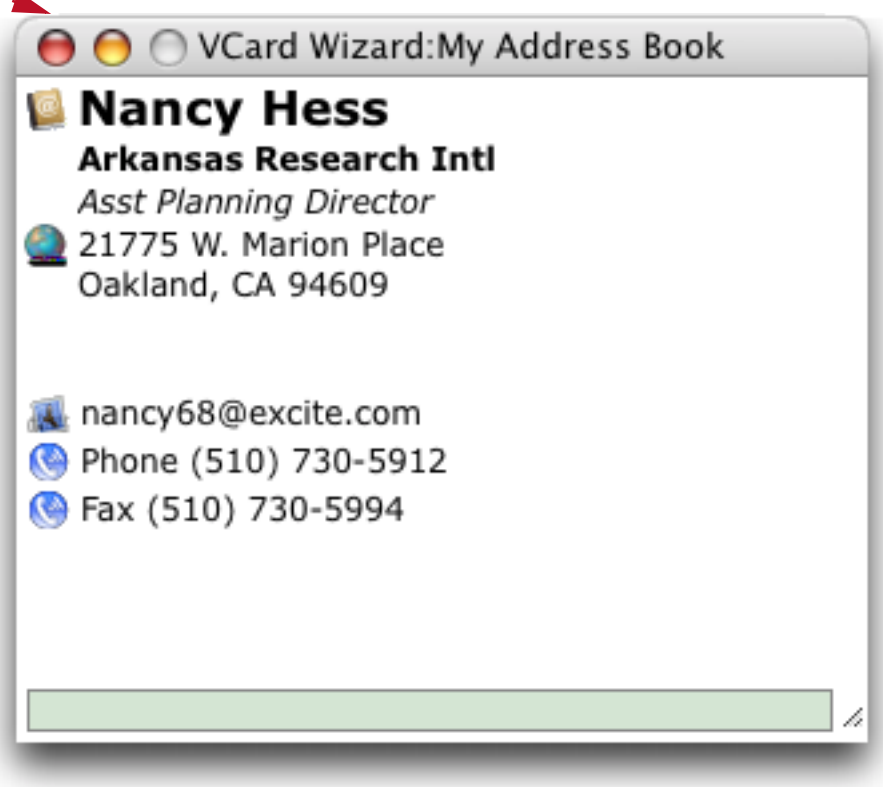
VCard Wizard

Generic fields allow you to transfer data between the database and other databases that also have generic fields, or between the database and applications that support vCards. For example an address could be copied to Apple's address book, or used to display a map. A phone number can be used to actually dial the phone, or an e-mail address to send an e-mail. The slickest way to use generic fields is to program them into your database itself (see "VCard Drag and Drop" on page 656 of *Formulas & Programming*). However, it's not necessary to do any programming to use generic fields. The **VCard Wizard** allows you to use generic fields without any programming at all. When you first open this wizard it will display the generic data from the current database, as shown below. (If the current database doesn't have any generic fields, it will display an error message.)

original database



VCard wizard displays the same information



With the **VCard Wizard** you can:

- Drag contact information back and forth between this database and other databases or applications.
- Import or export groups of VCards.
- Display a map of the currently displayed address
- Send an e-mail to the currently displayed contact
- Automatically dial the phone

To learn more about this wizard see "Using Generic Fields with the VCard Wizard" on page 237 of the *Panorama Handbook*.

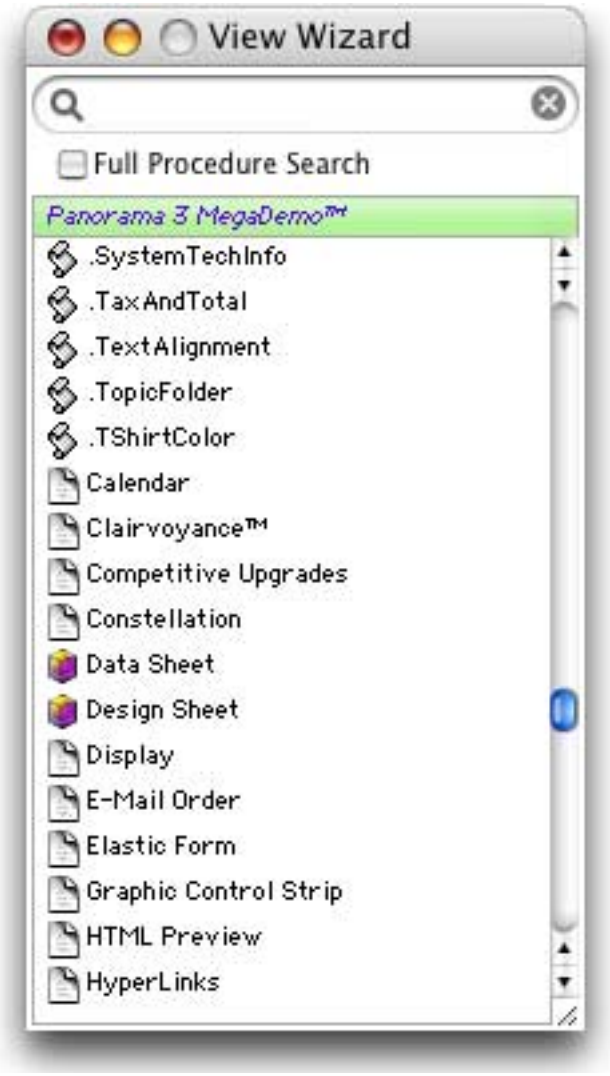
Open Database Wizard

The standard techniques for opening a database (double clicking, **Open File** dialog, **Favorite Databases** wizard, etc.) will work fine in 99,999 of 100,000 cases. Sometimes, however, you may need to use a more specialized technique to open a database. For example, if a database has lost its MacOS type/creator information (perhaps by sending it through an e-mail client that doesn't properly support this information, a common problem) the standard techniques will not work. In other cases you may need to open a database but bypass the normal initialization of that database. The **Open Database** wizard is included for these special needs.

To learn more about these wizards see "Advanced Database Opening Techniques" on page 47 of the *Panorama Handbook*.

View Wizard

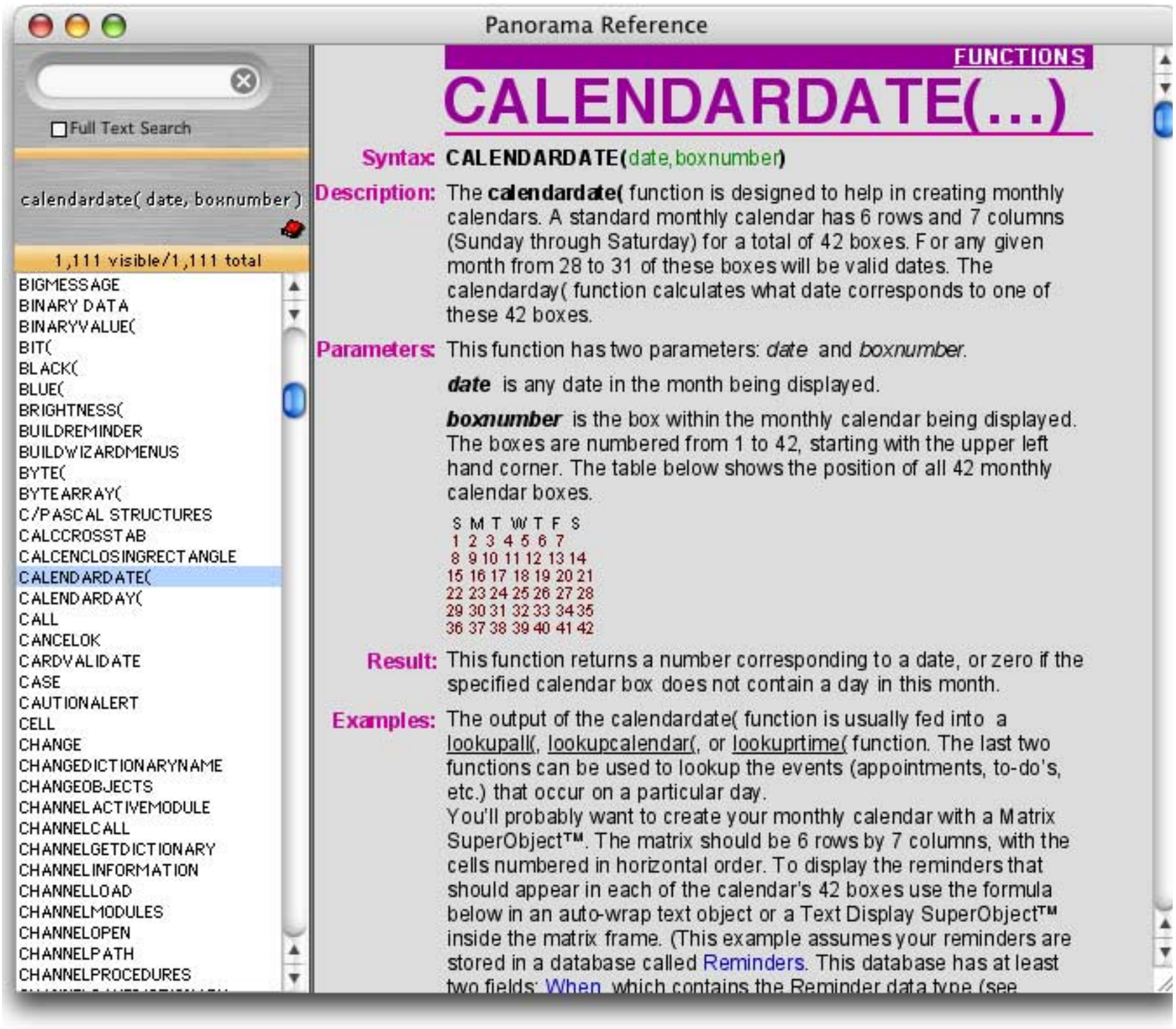
The **View Wizard** has been completely rewritten with a streamlined interface..



To learn more about the new version of this wizard see "The View Wizard" on page 173 of the *Panorama Handbook*.

Programming Reference Wizard

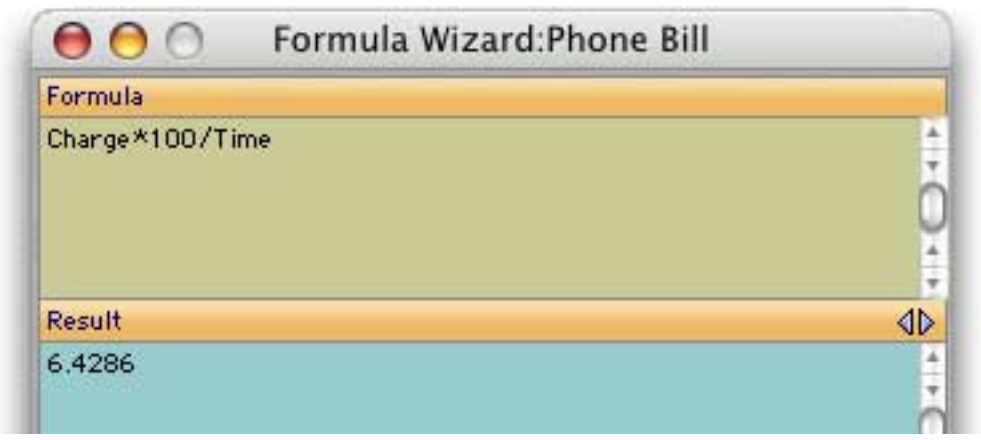
The new **Programming Reference** features live searching and an improved user interface.



For more information on using the **Programming Reference** see "Programming Reference Wizard" on page 237 of *Formulas & Programming* or simply open the wizard (the first page contains instructions).

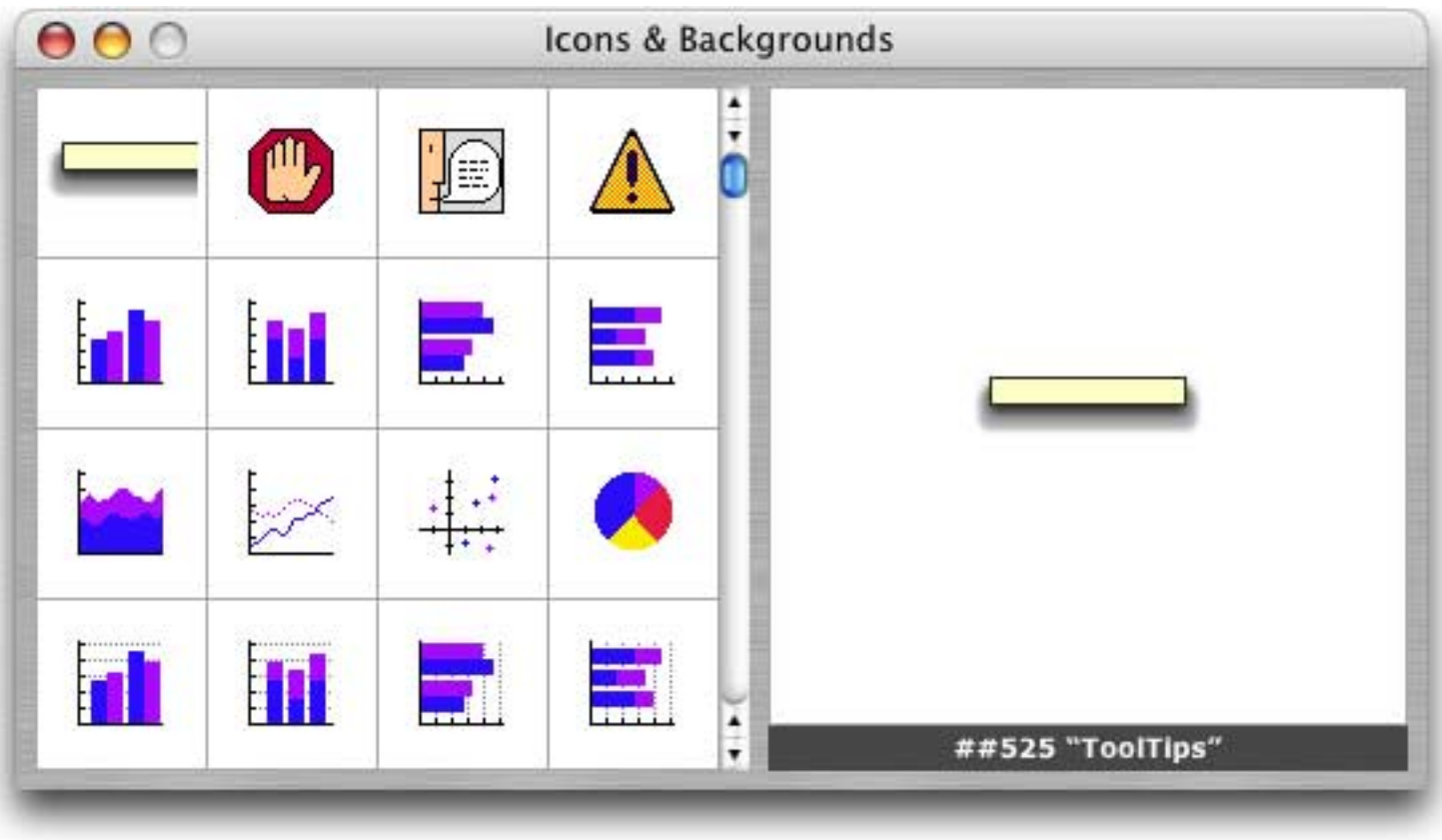
Formula Wizard

The **Formula Wizard** has an updated user interface, see "Using the Formula Wizard" on page 29 of *Formulas & Programming*.



Icons & Backgrounds Wizard

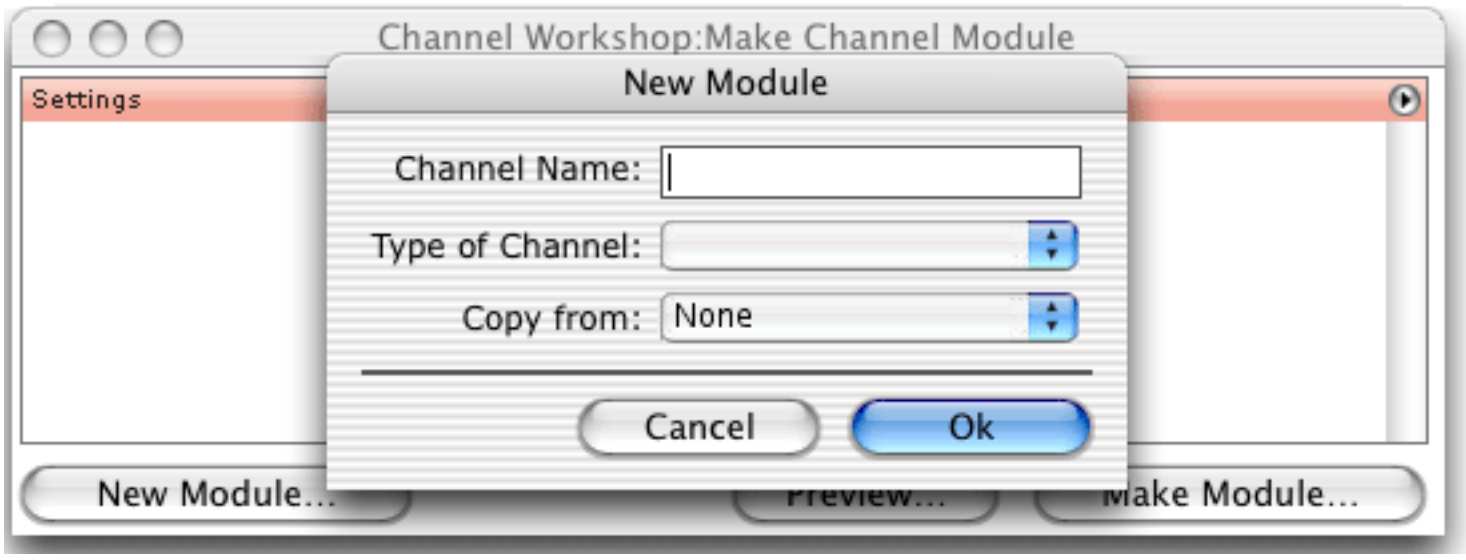
Panorama includes a number of resource based images within the application itself. Most of these are used by Panorama in various windows and dialogs, but they are available for use in your databases also. To see a list of these images open the **Icons & Backgrounds** wizard in the Form Tools submenu of the Wizard menu.



To learn more about this wizard and using these images see "Displaying Images from Resource Files" on page 802 of the *Panorama Handbook*.

Channel Workshop Wizard

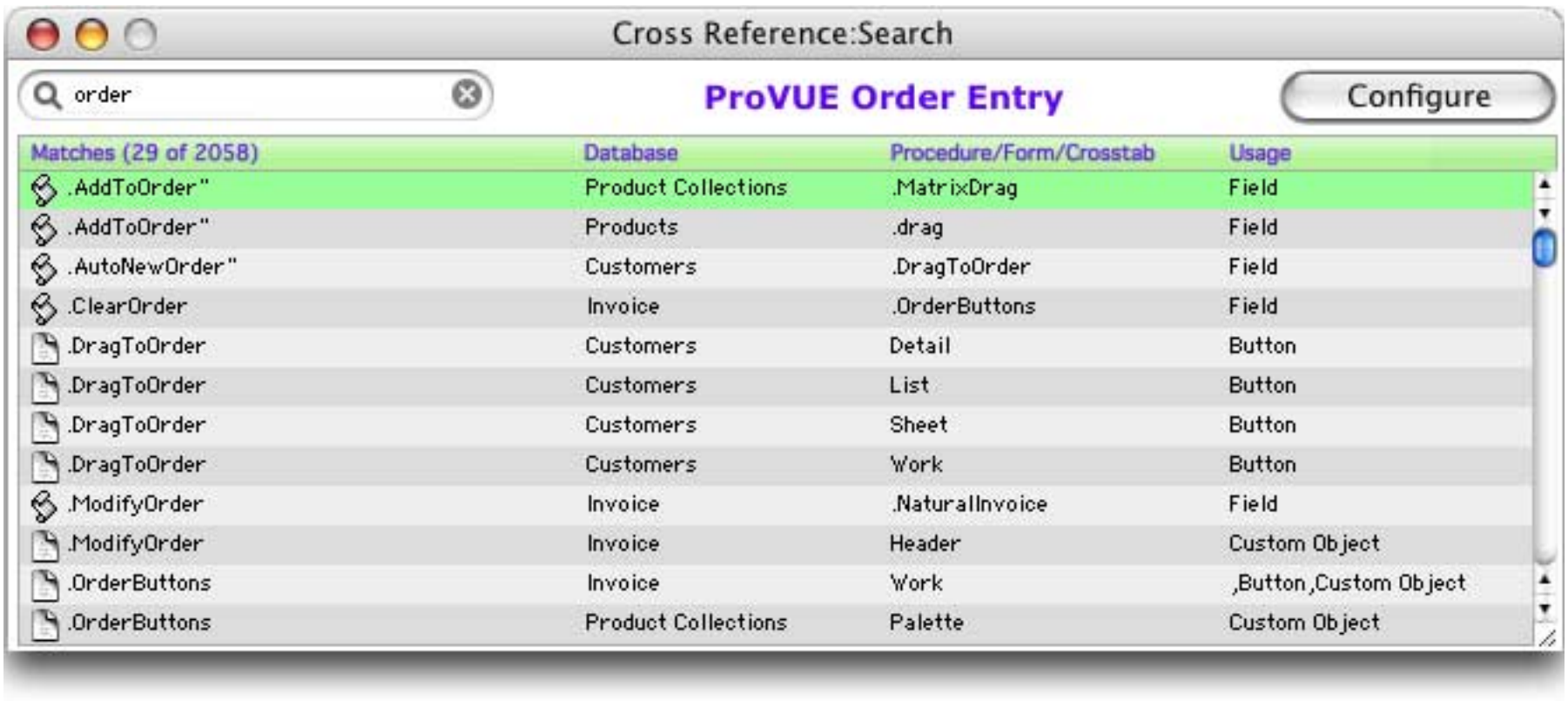
Panorama comes with a number of channel modules for sending e-mail, dialing the phone, and interfacing with other web sites and third party software. If you have programming experience you can write your own channel modules. To help make this easier we have created a **Channel Workshop** wizard that will create the core of your new module for you.



To learn more about this wizard see "Writing Your Own Channel Modules" on page 739 of *Formulas & Programming*.

Cross Reference Wizard

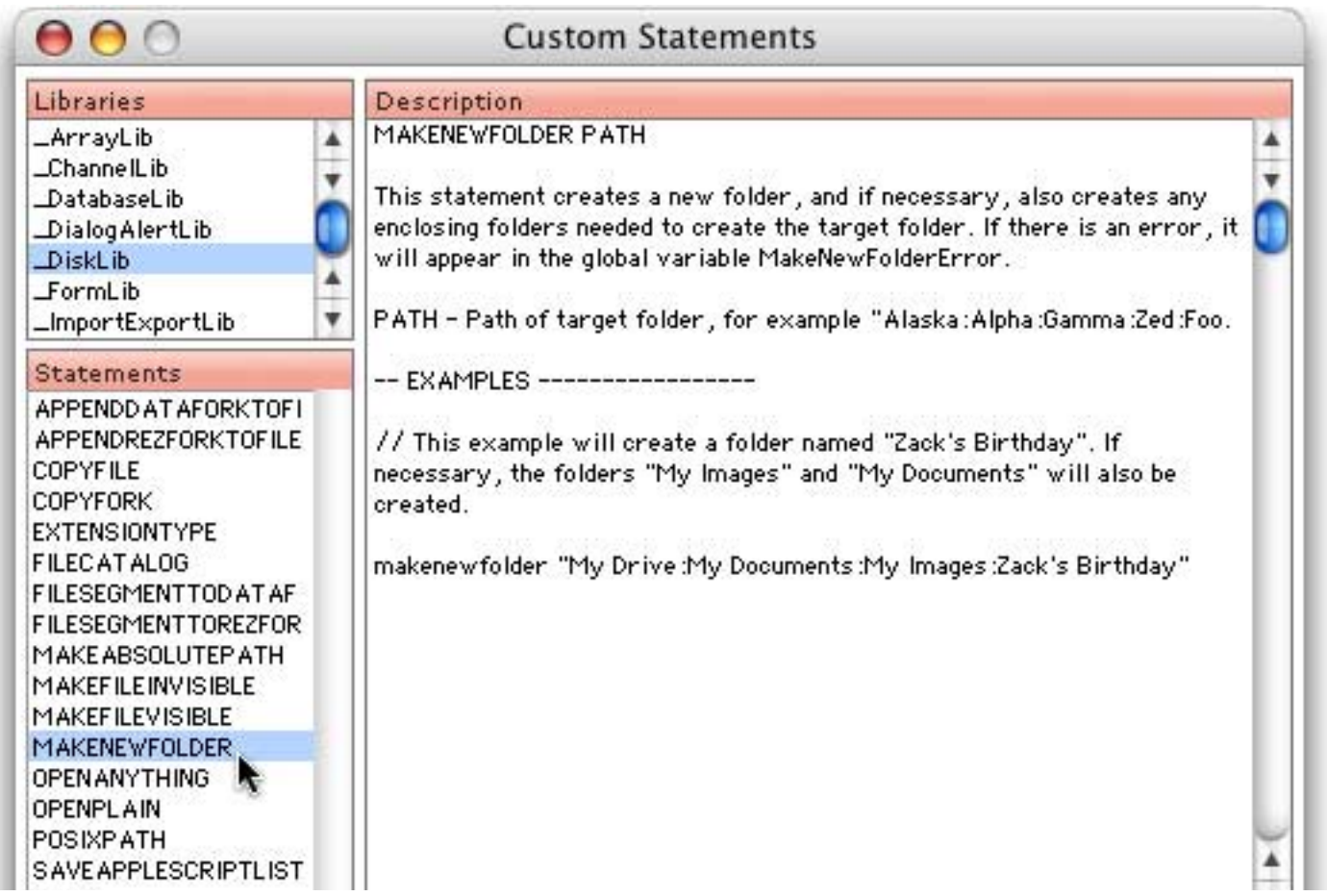
Panorama's **Cross Reference** feature has been turned into a wizard, and has an all new user interface!



To learn more about this wizard see "Cross Referencing" on page 349 of *Formulas & Programming*.

Custom Statement and Function Wizards

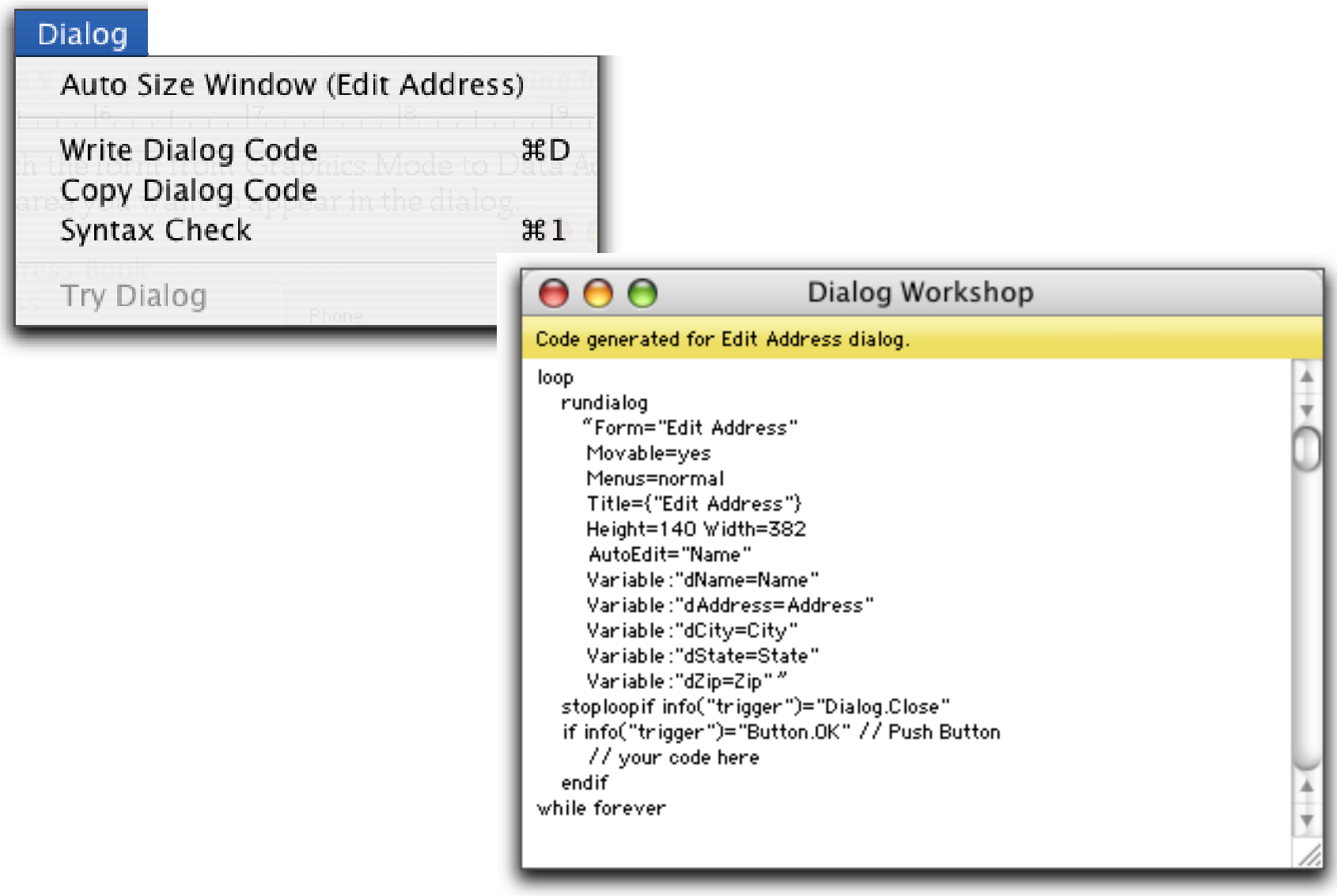
Panorama V allows you to define custom statements and functions, and there are three wizards to help.



For more information on Custom Functions see “Custom Statements” on page 289 of *Formulas & Programming* and “Custom Functions” on page 197 of *Formulas & Programming*.

Dialog Workshop Wizard

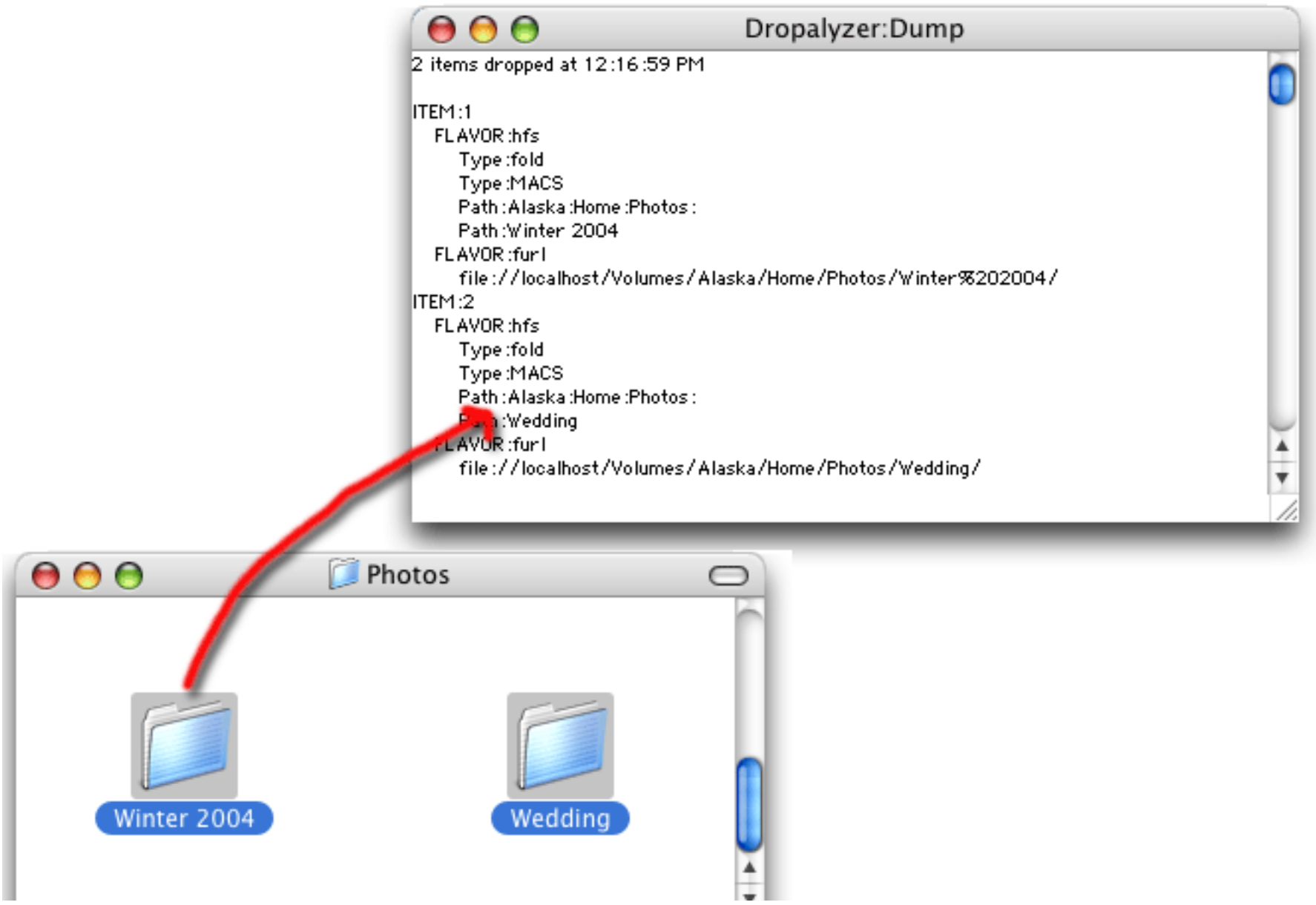
When an off the shelf dialog won't cut do, you can build your own using dialogs using standard Panorama forms and the **Dialog Workshop** wizard. This wizard analyzes your form and writes the basic code for that form for you. It also let's you try out your dialog before you actually commit the code to your database.



To learn more about this wizard see "Custom Dialogs" on page 489 of *Formulas & Programming*.

Dropalyzer Wizard

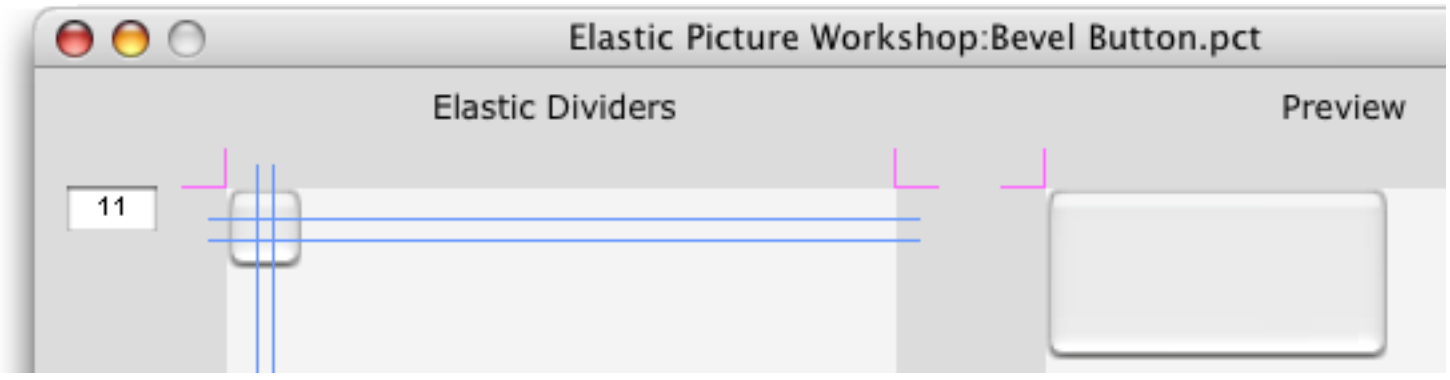
The **Dropalyzer** wizard is a handy tool for analyzing, writing and testing drag and drop procedures.When you first open this wizard it is completely blank, but you can drag anything you want onto this wizard and it will display some information about what was dropped. The illustration below shows the display if you drop two folders from the Finder onto the **Dropalyzer** wizard.



To learn more about this wizard see "The Dropalyzer Wizard" on page 651 of *Formulas & Programming*.

Elastic Picture Workshop

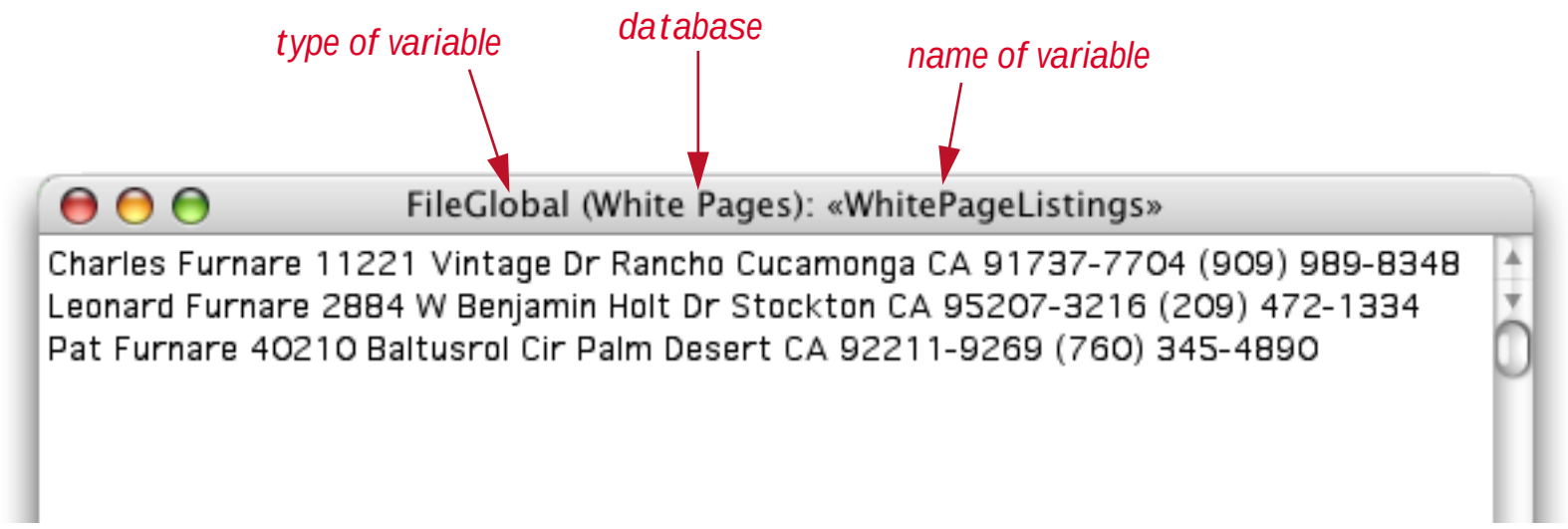
Many forms require borders, buttons and widgets need to be used over and over again but with different sizes. Any image can be stretched with the Scale to Fit option, but the result is often a distorted image. The **Elastic Picture Workshop** wizard can be used to add stretching information to an image so that it won't be distorted when it is stretched.



To learn more about this wizard see "Elastic Pictures" on page 809 of the *Panorama Handbook*.

Variables Wizard

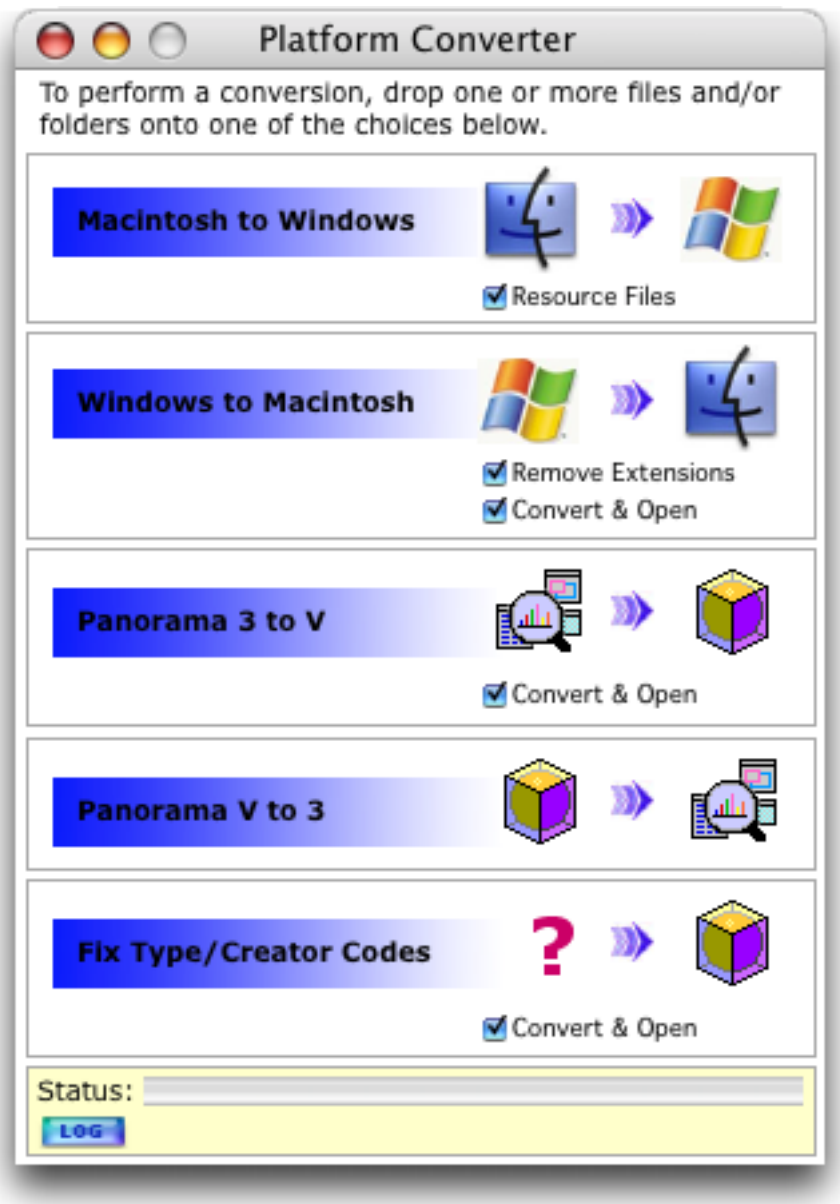
Global, **fileglobal** and **permanent** variables can be displayed and modified with the **Variables** wizard.



To learn more about this wizard see "Displaying and Changing Variables" on page 253 of *Formulas & Programming*.

Platform Converter Wizard

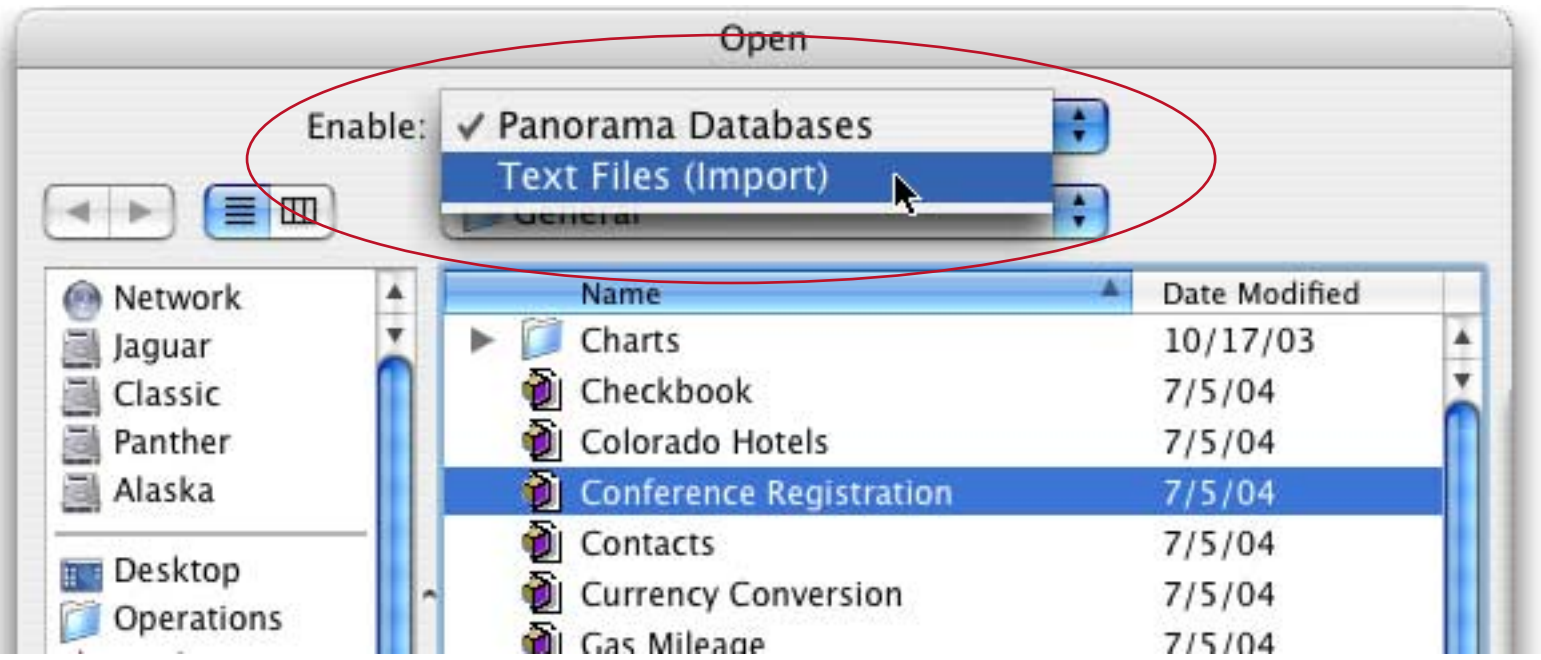
The **Platform Converter** has been completely rewritten to use drag-and-drop.



For more information about this wizard see "Platform Converter Wizard" on page 800 of *Formulas & Programming*.

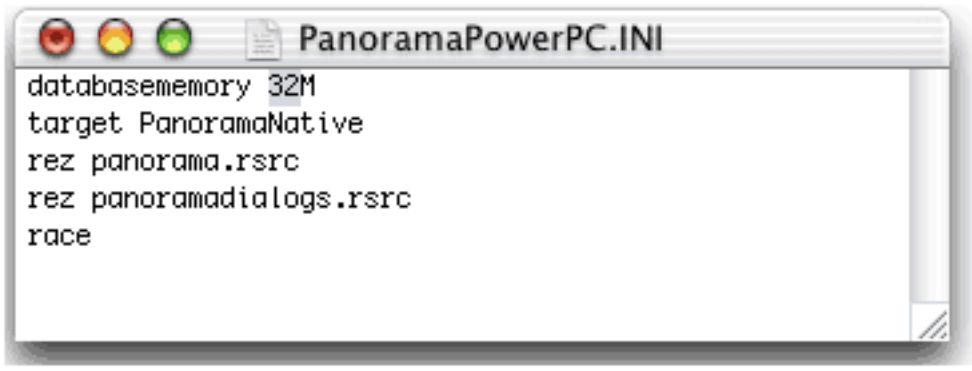
Importing with the Open File Dialog

Panorama now uses the “navigation services” dialogs for opening and saving files on Mac OS X and OS 9. These dialogs are slightly different than the older Mac OS 6/7/8 style dialogs that Panorama 4.0 and earlier versions used. Instead of using radio buttons to select the type of file to import, these new dialogs use the Enable: pop-up menu at the top of the dialog.



Panorama Memory Allocation

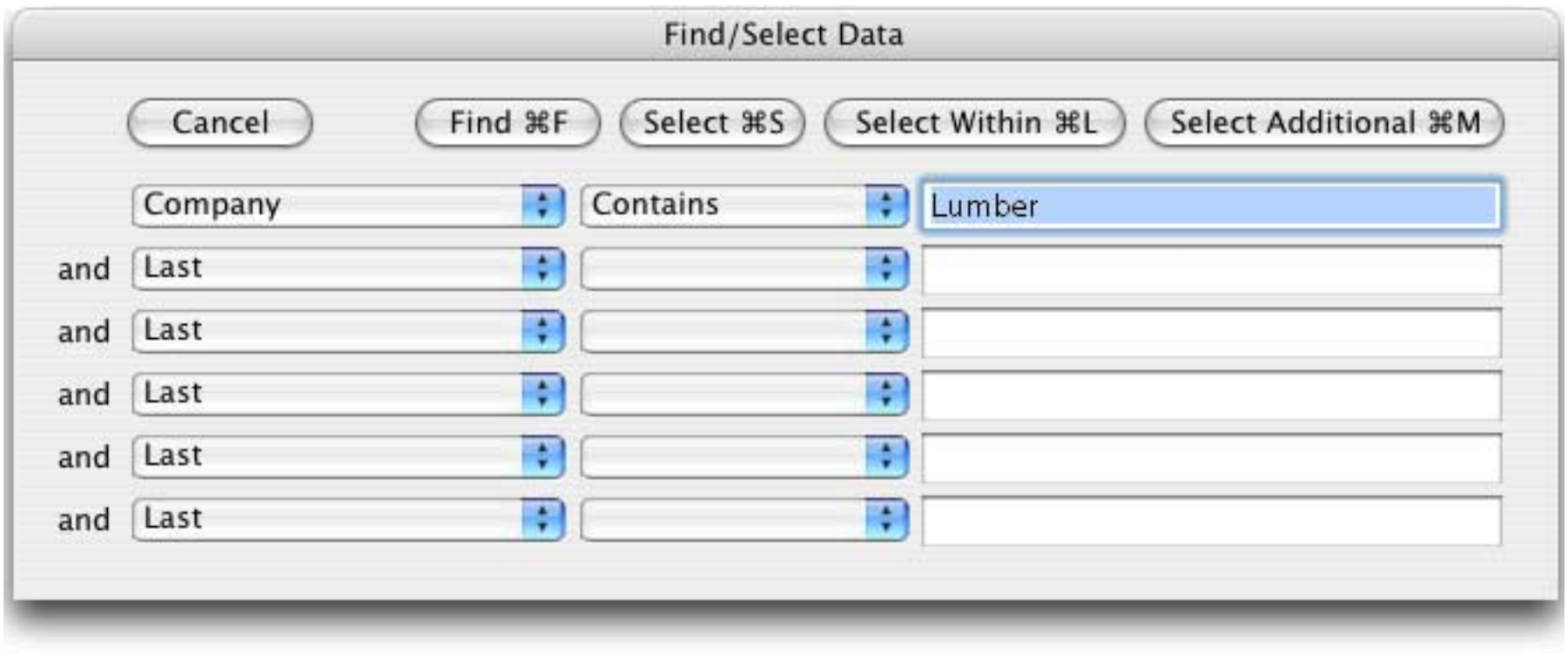
When using Mac OS 9, Panorama V's default scratch memory allocation has been increased to 2 megabytes. When using OS X memory allocation is handled differently. There is no scratch memory allocation under OS X. The default data allocation is 32 megabytes. If your databases are larger than this you must edit the PanoramaPowerPC.ini file. You can double click to open this file with the text editor. (Note: Be sure you open PanoramaPowerPC.ini and not Panorama.ini. If you open the wrong file, don’t worry, just close it and open the correct file.) The file should look something like this:



The **databasememory** line controls the amount of memory allocated by Panorama for databases. You may set this to any value from 3M to 1999M. (Don’t forget the M!). However, if you set this value to larger than the physical amount of memory available on your computer, you may reduce the amount of virtual memory available for other applications. We do not recommend opening databases that are larger than the physical memory size of your computer. Panorama will open the file and operate correctly, but its performance may be severely degraded. Once you have set the new value save and close the window, then relaunch Panorama if necessary. For more information see “Adjusting Panorama’s Memory Allocation” on page 140 of *Formulas & Programming*.

Find/Select Dialog

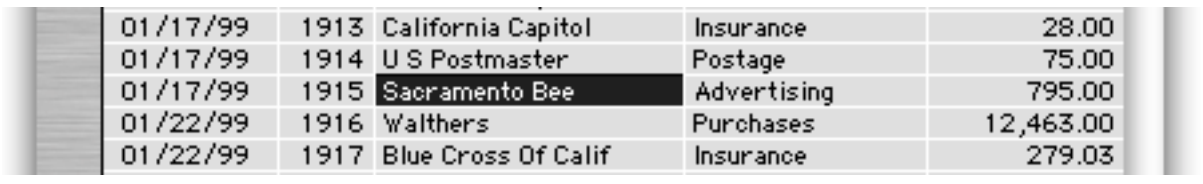
The **Find/Select** dialog no longer expands and shrinks - it always appears at its full six line maximum height, as shown here.



Manually Toggling Summary Records

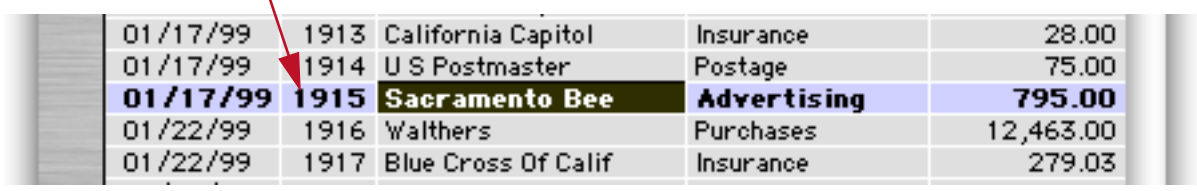
To turn a normal data record into a summary record, click anywhere in the record in the data sheet, then choose the **Toggle Summary Level** command in the Sort menu. Each time you choose this command will toggle between normal and summary. (In previous versions of Panorama you could click on the + sign at the left edge of the data sheet, but this feature has been removed because many users would click it accidentally and then wonder why their database no longer sorted correctly. Now you have to use the **Toggle Summary Level** command.)

start with an ordinary record...



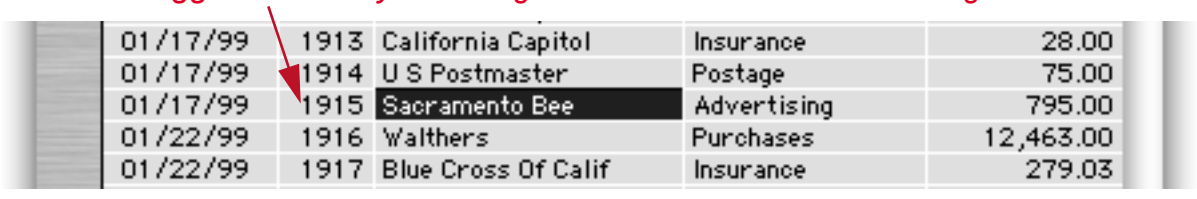
01/17/99	1913	California Capitol	Insurance	28.00
01/17/99	1914	U S Postmaster	Postage	75.00
01/17/99	1915	Sacramento Bee	Advertising	795.00
01/22/99	1916	Walthers	Purchases	12,463.00
01/22/99	1917	Blue Cross Of Calif	Insurance	279.03

Choose Toggle Summary Level to convert it to a summary record



01/17/99	1913	California Capitol	Insurance	28.00
01/17/99	1914	U S Postmaster	Postage	75.00
01/17/99	**1915**	**Sacramento Bee**	**Advertising**	**795.00**
01/22/99	1916	Walthers	Purchases	12,463.00
01/22/99	1917	Blue Cross Of Calif	Insurance	279.03

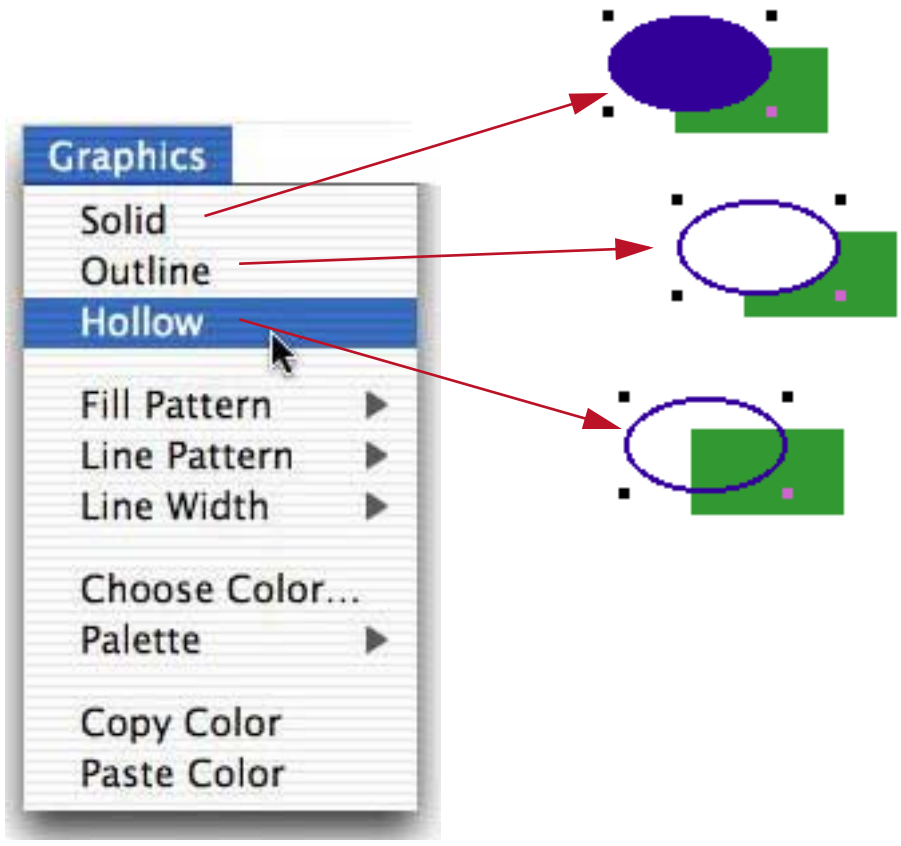
Choose Toggle Summary Level again to convert it back to a regular record



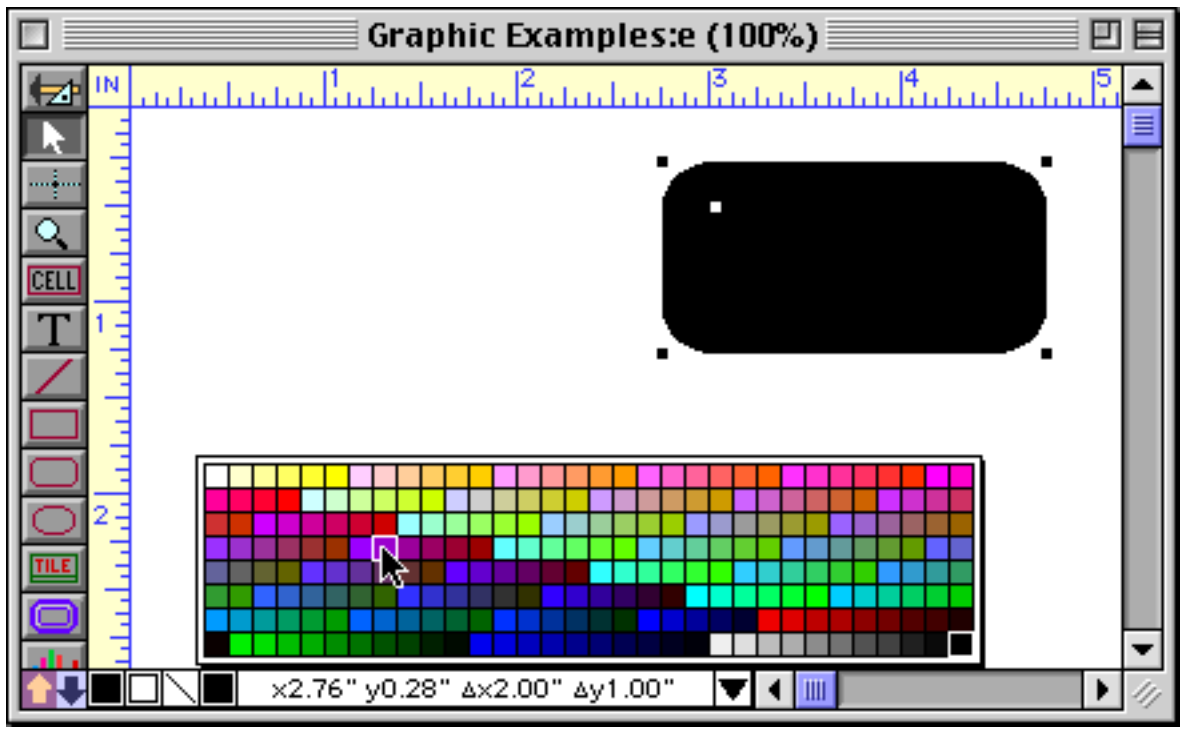
01/17/99	1913	California Capitol	Insurance	28.00
01/17/99	1914	U S Postmaster	Postage	75.00
01/17/99	1915	Sacramento Bee	Advertising	795.00
01/22/99	1916	Walthers	Purchases	12,463.00
01/22/99	1917	Blue Cross Of Calif	Insurance	279.03

Enhanced Graphics Menu

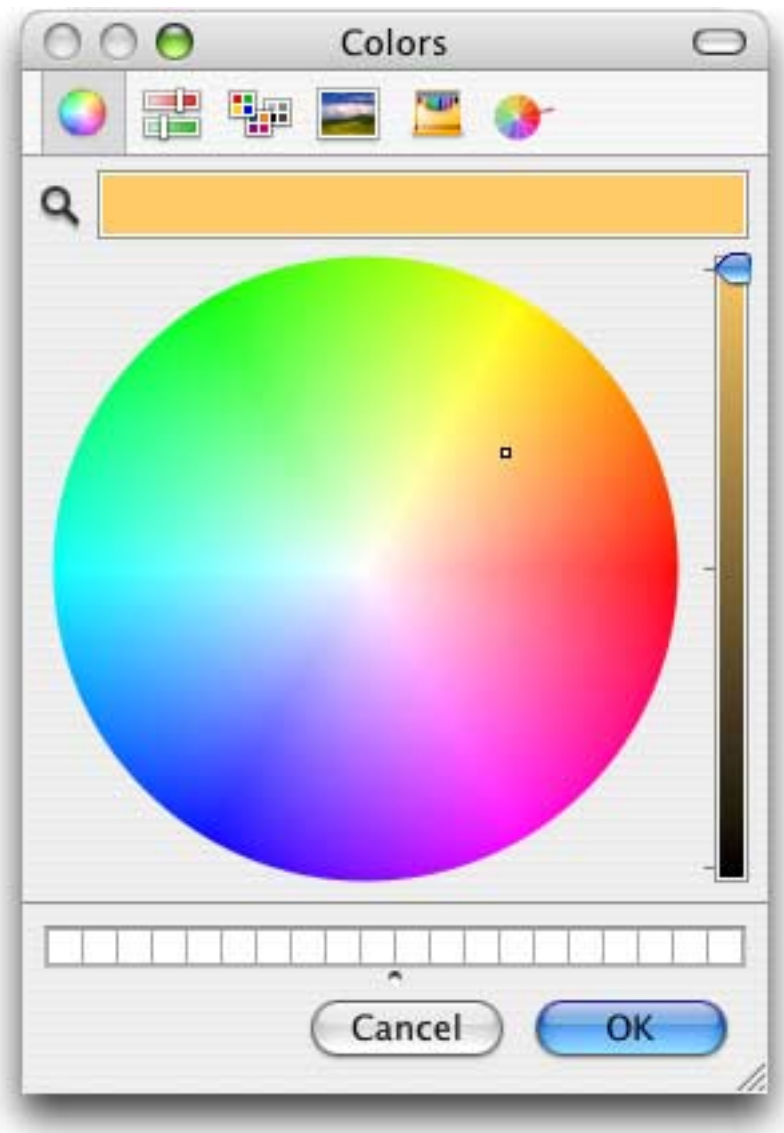
The **Graphics** menu (which appears in Graphics Mode) controls the texture and color of objects on a form. Panorama V adds several new items to this menu. The first three choices in the Graphics menu, **Solid**, **Outline** and **Hollow**, set the fill and line patterns of the selected objects to their most common choices, as shown in the diagram below (see "Solid, Outline and Hollow Objects" on page 520 of the *Panorama Handbook*).



Previous versions of Panorama required that you choose object colors from a pre-defined 256 color palette.



To choose a color that is not one of the 256 colors in the palette, use the **Choose Colors** command in the Graphics menu. The system's standard Color selection dialog will appear, allowing you to choose any of millions of colors.



You can also open this dialog by holding down the **Control** key (Mac) while clicking the color swatch in the Control strip. On PC systems you can right click the swatch. See "Color" on page 526 of the *Panorama Handbook*.

Sometimes you'll want one object to exactly match the color of another object. You can use the **Copy Color** and **Paste Color** commands to transfer a color from one object to one or more other objects.

Start by selecting the object that has the color you want, then choose the **Copy Color** command. This copies the color into a special clipboard. Now select the other objects that you want to set to this color and choose the **Paste Color** command. The selected objects will change to the same color as the original object. (Note: The color clipboard is completely separate from the normal clipboard that is used for other copy and paste operations.) See "Copying and Pasting Colors" on page 528 of the *Panorama Handbook*.

Text Display and Editing

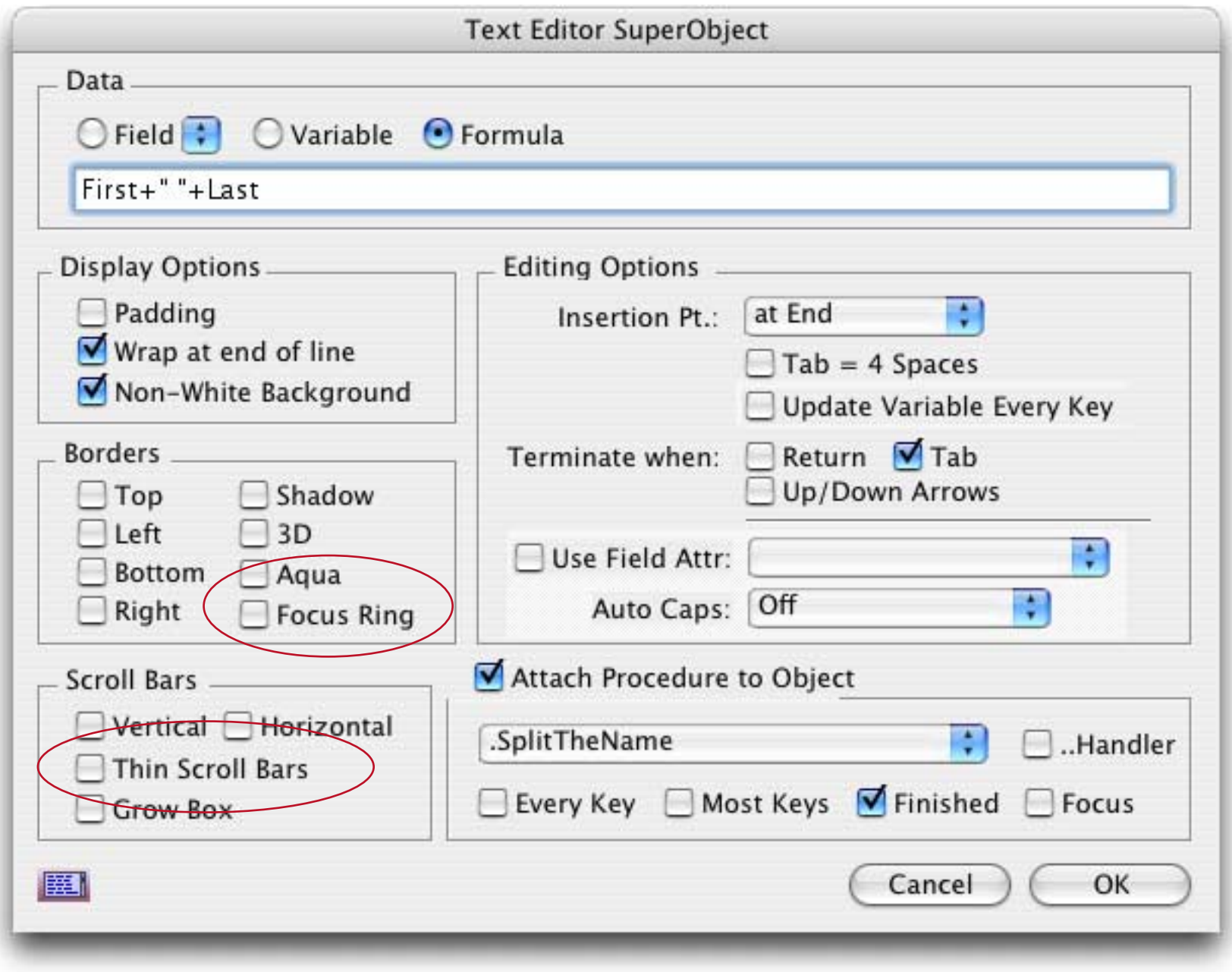
Panorama V includes minor but important enhancements to the Text Display and Text Editor SuperObjects.

Text Display "Aqua" Text Option

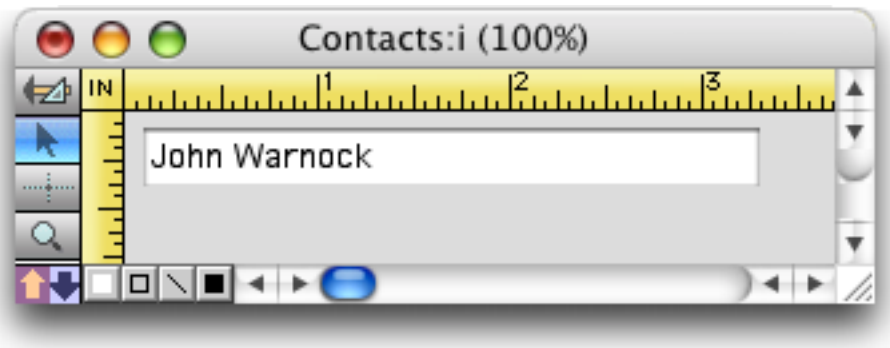
The new Aqua option is smooths the text (using anti-aliasing) if the operating system supports that feature (currently only OS X supports this). See "Text Display Options" on page 611 of the *Panorama Handbook*.

Text Editor Options

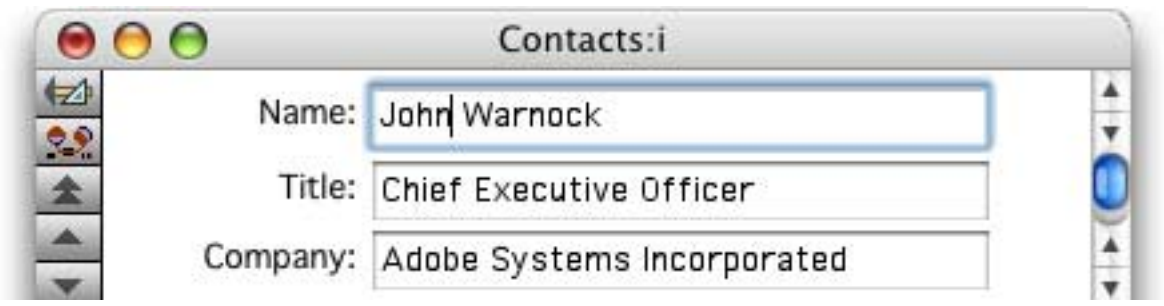
The Text Editor SuperObject has three new options: Aqua, Focus Ring, and Thin Scroll Bars (see "Text Editor Options" on page 643 of the *Panorama Handbook*).



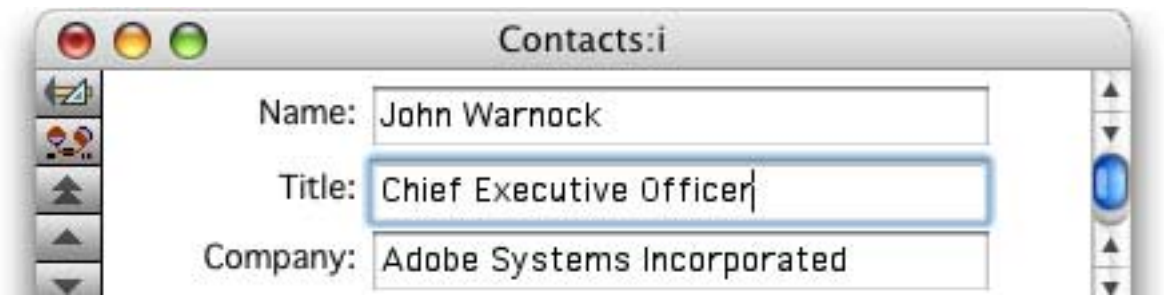
The Aqua option displays the same soft 3D borders that are used by most OS X applications. This option can work with a light gray or white background.



When the Focus Ring option is checked, Panorama will display a blue ring around the object when it is being actively edited.



When you shift to editing a different object, the blue ring will move to this new object.

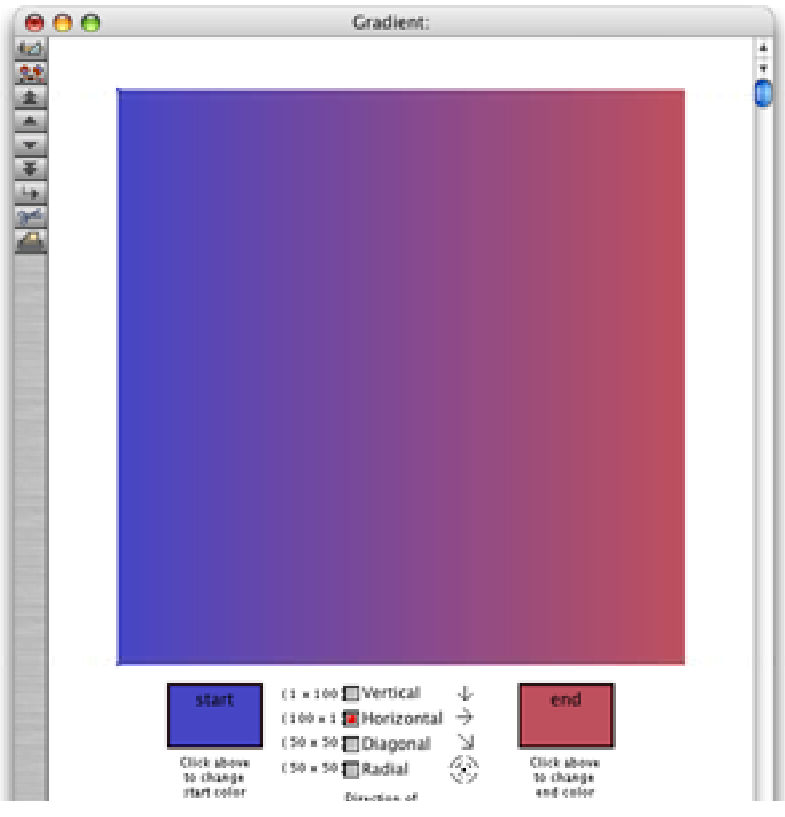


The Thin Scroll Bar option displays 11 pixel wide scroll bars instead of the standard 16.

Using Super Flash Art™ to Display a Color

Super Flash Art objects are normally used to display images. Panorama V enhances these objects with the ability to display any of 65,535 colors. No image is required - the object generates the color on it's own. See "Using Flash Art to Display a Color" on page 761 of the *Panorama Handbook*.

Panorama customer Gary Yonaites has created a cool demo that uses the solid color feature in combination with a Super Matrix object to create blends. This demo database, called Gradient, has been included with the Panorama example files.



The gradient above was not created in Photocells or some other graphics program, but within Panorama itself. By clicking on the check boxes and color selections you can change the gradient on the fly. Thanks Gary!

Buttons and Widgets

Buttons, data buttons, and pop-up menus have been enhanced to make them more compatible with OS X.

Push Button SuperObject Styles

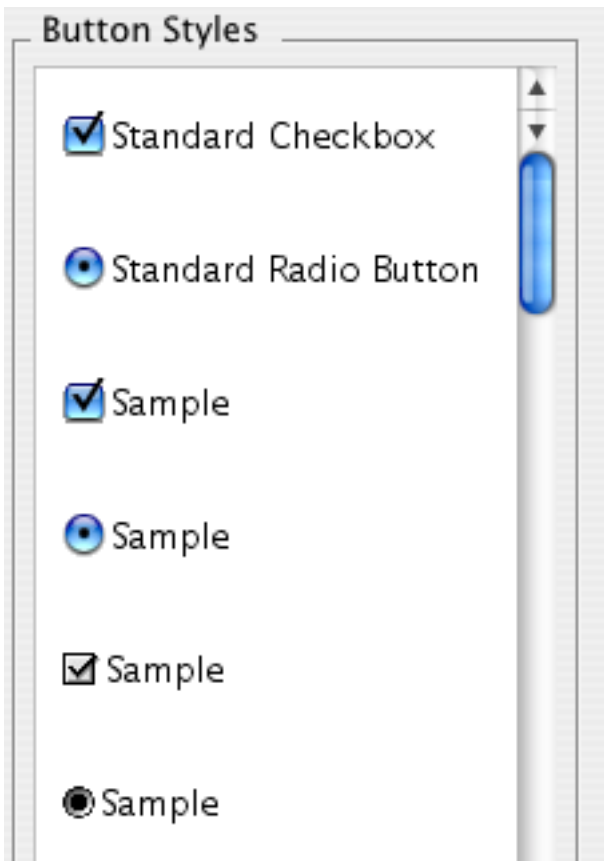
Two new styles — standard and default — make it easy to create buttons that "shape shift" to look great with whatever operating system you are using. Here's what these buttons look like in OS X.

Push Button SuperObjects now have a transparent option, which makes the button completely invisible. This removes any need to use the old "Classic" push button.

For more information see "Push Button Styles" on page 826 of the *Panorama Handbook*.

Data Button SuperObject Styles

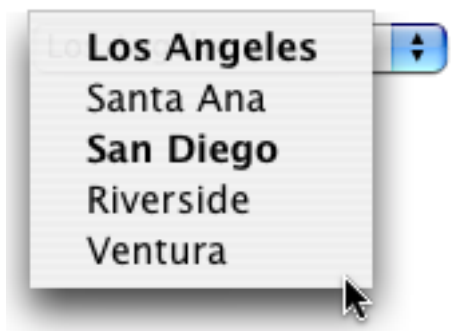
Over a dozen new data button styles have been added. The first two styles, Standard Checkbox and Standard Radio Button, are special. Data buttons created with these styles will adjust automatically depending on what operating system is being used. In other words, the same button will change appearance depending on whether the database is being used on Mac OS X, Mac OS 9, or Windows.



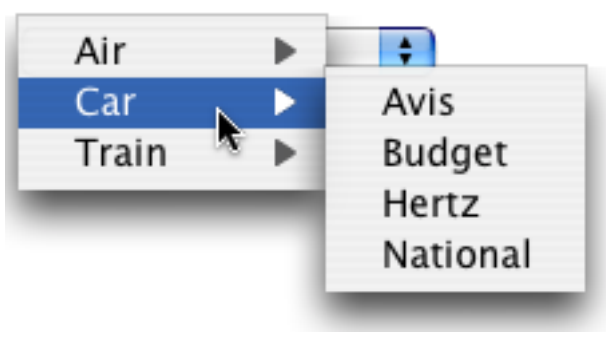
See "Super Data Button Options" on page 849 of the *Panorama Handbook* for more details.

New Pop-Up Menu Features

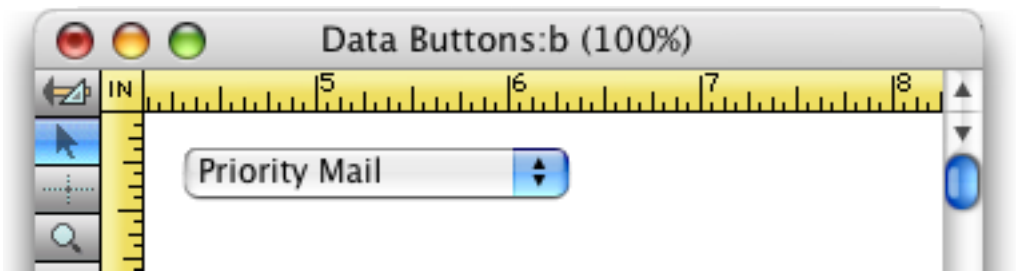
Using the new “Live Menus” feature you can create pop-up menus with different styles for each menu item (see ““Live” Pop-Up Menu Formulas” on page 864 of the *Panorama Handbook*).



You can even create a pop-up menu with submenus (see “Pop Up Submenus” on page 865 of the *Panorama Handbook*).



The new Standard Icon option is now the default choice, and displays the standard pop-up menu graphics for the operating system being used. Here is the standard icon for Mac OS X (see “Display Options” on page 869 of the *Panorama Handbook*).



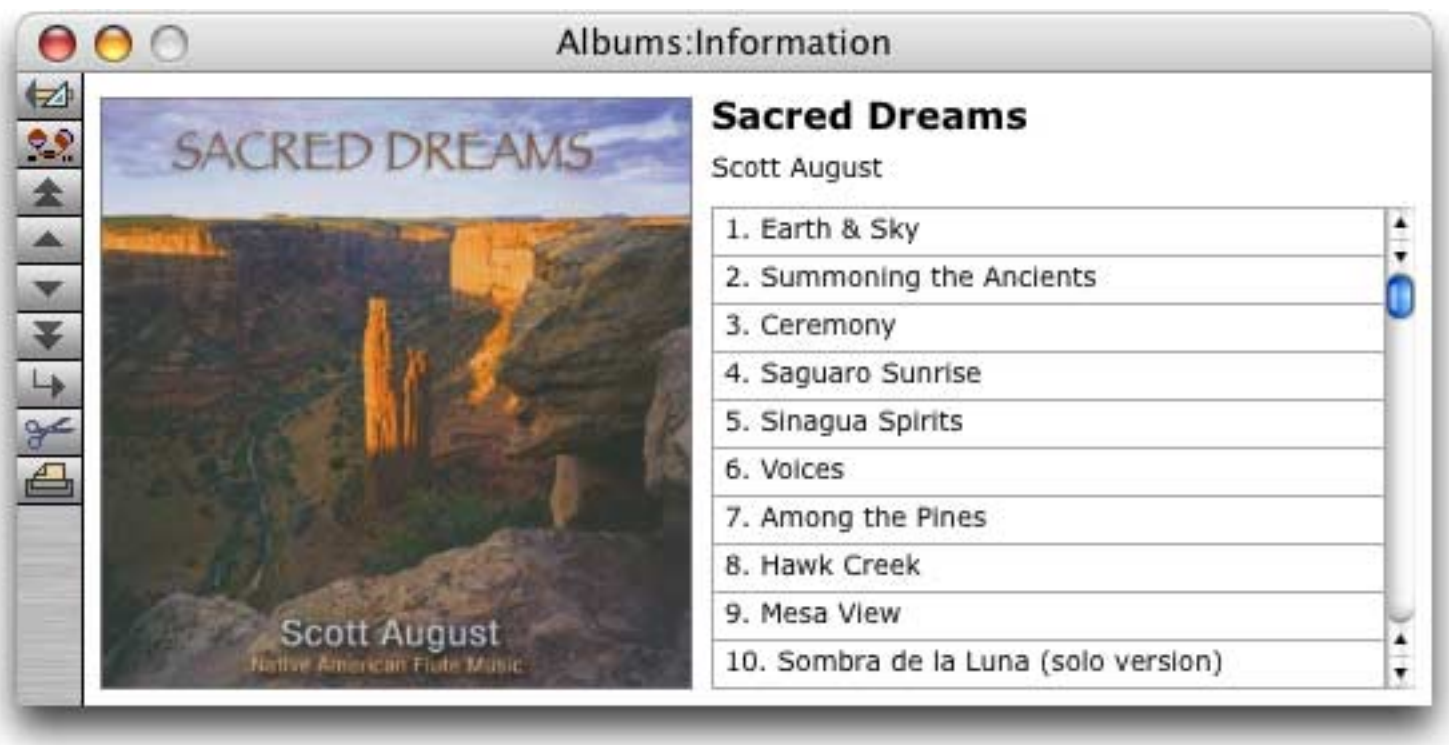
There are also several new statements that make it easier to create pop-up menu’s on the fly in a procedure. See “The PopUpButton Statement” on page 875, “The PopUpClick Statement” on page 875, “The PopUpFieldChoices Statement” on page 876 and “The PopUpDoubleFieldChoices Statement” on page 876 of the *Panorama Handbook*. The last two statements automatically build a pop-up menu (or menu and submenus) from one or two fields in the database.

List SuperObject Thin Scroll Bars

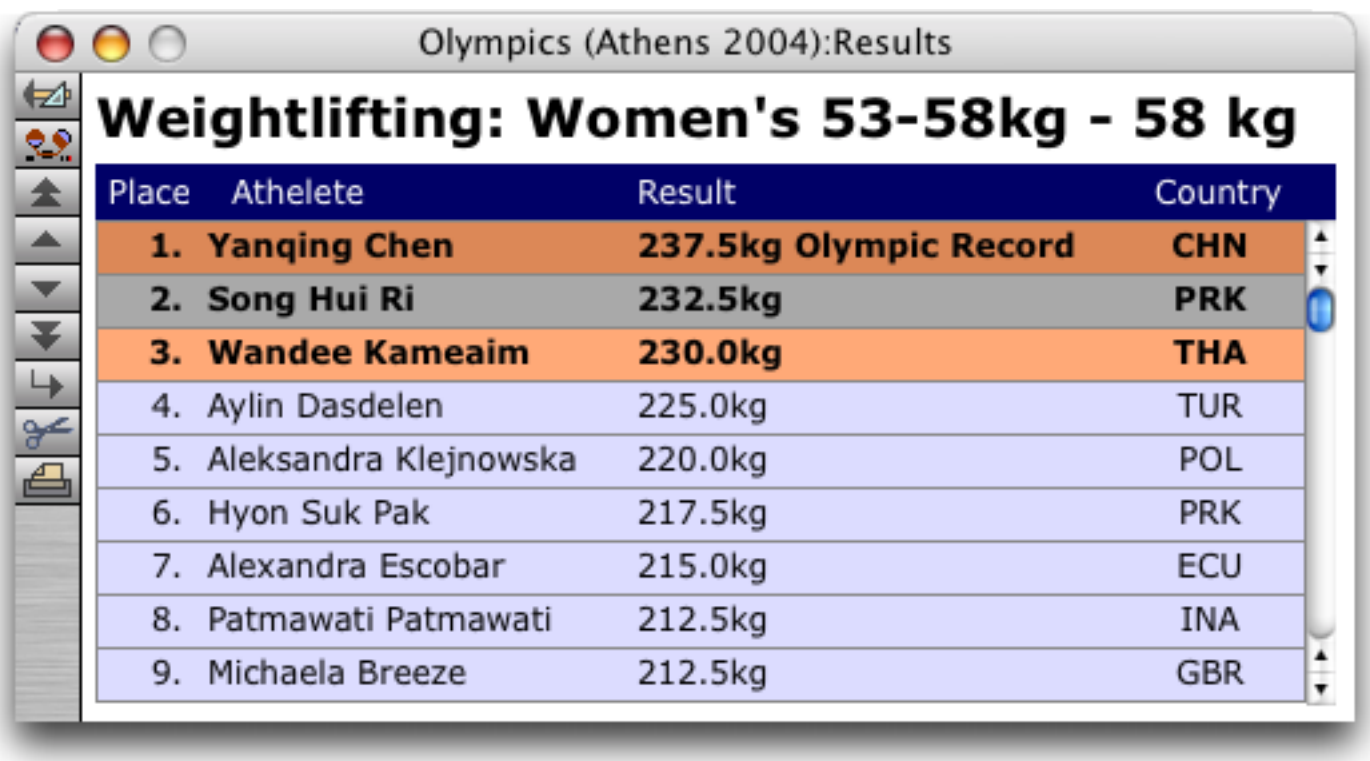
List SuperObjects are basically unchanged except for a new option to display “thin” (11 pixel) scroll bars. See “Thin Scroll Bars” on page 894 of the *Panorama Handbook*.

Scrolling Super Matrix Objects

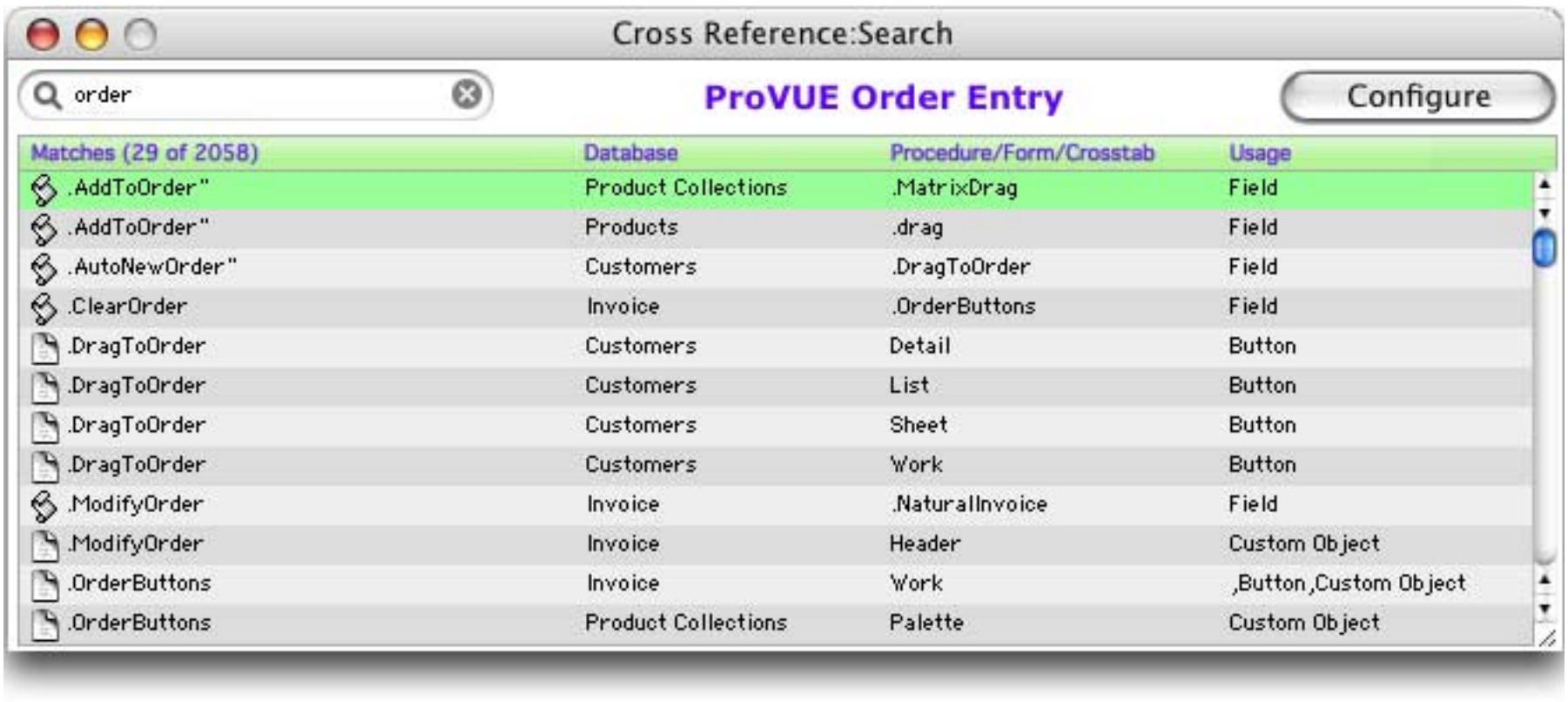
The Super Matrix Object now supports scroll bars, and can be linked to an array (see "Super Matrix Objects" on page 939 of the *Panorama Handbook*). At first glance this might appear to be a minor enhancement, but this really opens up major new possibilities in form design. For many applications, the Super Matrix object can now replace a Scrolling List object, but with full control over the graphical display. Many of the new wizards included with Panorama V use scrolling matrixes this way, including the White Pages, View Wizard, Favorite Databases, Cross Reference, and more. Here are some examples of forms that have been created with this new option. The first example shows a simple scrolling list.



This example illustrates the graphic control available. The first three items in the list are displayed in bold, with backgrounds of gold, silver and bronze. This example also demonstrates the ability to create a multi-column scrolling list using a matrix.



Here is another example with alternating background colors, multiple columns and graphics in the list.



We're looking forward to seeing the cool designs you come up with using this new feature.

Last Page Footer

Panorama has always had a **First Page Header** tile that allows you to print a title on the first page of a report (or a cover page). Panorama 4.9.5 now adds a **Last Page Footer** tile, which comes in three variations (left, center and right). These tiles work exactly like the **First Page Header** tiles except that they print at the very end of the report. You can access these new tiles through the specialized tile dialog. For more information on tiles see "Working with Tiles" on page 1062 of the *Panorama Handbook*.

Formulas and Functions

Panorama V includes hundreds of new functions, and also gives you the ability to create your own custom functions.

New Numeric Functions

These new functions manipulate numeric values.

Function	Description
numsandwich(value,extra)	This function is similar to the sandwich function, but for numbers. If the value is zero the result is zero, but if the value is not zero the result is the value plus the extra. For example, this could be useful for calculating the size of an object plus a border. If the object size is zero, the border is omitted also, for example numsandwich(20,7) will be equal to 27, but numsandwich(0,7) will be equal to zero.
randominteger(startnum,endnum)	Returns a random integer value greater than or equal to the startnumber and less than or equal to the end number.

New Functions for Taking Strings Apart

These new functions return portions of a string.

Function	Description
firstline(string)	This function extracts the first line from the text.
firstword(string)	This function extracts the first word from the text (the text up to the first space).
lastline(string)	This function extracts the last line from the text.
lastword(string)	This function extracts the last word from the text (the text from the last space to the end).
left(string,len)	Extracts characters from the left edge of the text. For example left(text,2) extracts the leftmost two characters.
mid(string,len)	Extracts characters from the middle of the text. For example mid(text,6,4) extracts four characters starting with the sixth character.
nthline(string,num)	This function extracts the nth line from the text. For example nthline(text,4) extracts fourth line.
nthword(string,num)	This function extracts the nth word from the text. For example nthword(text,7) extracts seventh word.
right(string,len)	Extracts characters from the right edge of the text. For example right(text,7) extracts the rightmost seven characters from the text.
snip(string,startposition,count)	This function removes (snips!) one or more characters from the middle of an item of text. The startposition specifies the first character removed, the count is the number of characters to remove. (Note: This function requires the startposition to be a positive number.) If count is -1 then all the text from the start position to the end of the text is snipped, otherwise the count must be a positive number.
textafter(string,tag)	This function extracts the text after the tag. The tag many be one or more characters long. If the tag doesn't occur in the text then the entire original string is returned. For example textafter("someone@isp.net","@") will return isp.net.
textbefore(string,tag)	This function extracts the text before the tag. The tag many be one or more characters long. For example textbefore("someone@isp.net","@") will return someone. If the tag doesn't occur in the text then the entire original string is returned.
trim(string,len)	This function removes characters from the right edge of the text. For example trim(text,4) removes the last four characters from the text.
trimleft(string,len)	This function removes characters from the left edge of the text. For example trimleft(text,2) removes the first two characters from the text.

New String Testing Functions

These new functions return information about the content of a string.

Function	Description
linecount(string)	This function counts the number of lines in the text.
rangematch(string,range)	This function checks text to see if the text matches the specified range. The range must be a series of character pairs, for example AZ for upper case alphabetic characters, AZaz for upper and lower case, 09 for numeric digits, etc. If it matches the function returns true, if it doesn't match, it returns false. For example, rangematch(Address,"AZaz09 ") will return true if the address contains only letters, numbers and spaces, false if it contains any other characters.
wordcount(string)	This function counts the number of words in the text.

New String Modification Functions

These new functions modify the contents of a string.

Function	Description
applescriptstring(string)	This function converts the string into an AppleScript string literal. It surrounds the text with double quote characters, and escapes any double quote and/or backslash characters within the text. This function is designed to be used with the executeapplescript statement.
connect(prefix,connector,suffix)	This function appends a prefix and suffix together with a connector in between. If either the prefix or the suffix is missing then the connector will also be left out. For example, connect(City,", ",State) combines the city and state with a comma and space in between, but if either the city or state is missing then the comma and space will also be left out. See also the sandwich(and yoke(functions in this table.
crtovtab(string)	This function converts carriage returns (ASCII 0x0D) into vertical tabs (ASCII 0x0B). Some programs (including Panorama) will convert vertical tabs into carriage returns when importing, allowing individual data cells to contain carriage returns.
fixedwidth(string,width)	This function makes the text a fixed width. If the text is shorter than the specified width, it is padded with spaces. If it is longer than the specified width, it is cut off.
fixedwidthright(string,width)	This function makes the text a fixed width. If the text is shorter than the specified width, it is padded with spaces on the left (i.e. the text is right justified). If it is longer than the specified width, it is cut off on the left.
onespace(string)	This function removes any extra spaces between words, so that there is exactly one and only one space between each word.
onewhitespace(string)	This function removes any extra whitespace between words, making sure that there is one and only one space between each word. Other whitespace characters (carriage returns, tabs) are converted to spaces and removed if there is more than one between words.
quoted(string)	This function surrounds the supplied text with double quote characters. If the text contains any double quotes they will be doubled, making this a legal string constant. This function is typically used in conjunction with the execute and executeapplescript statement.
randomletter(option)	This function returns a random letter. If the option is "U" then the letter will be between A and Z. If the option is "L" then the letter will be between a and z. If the option is any other value then the result may be either A-Z or a-z.
striphtmltags(text)	This function removes all HTML tags from the text.
stripprintable(text)	This function removes any non-displayable characters from the text.
vtabtocr(string)	This function converts vertical tabs (ASCII 0x0B) into carriage returns (ASCII 0x0D). Some programs (including Panorama) will convert vertical tabs into carriage returns when importing, allowing individual data cells to contain carriage returns. After the import you can use this function to turn the vertical tabs back into carriage returns.
yoke(prefix,joiner,suffix)	This function appends two text items (prefix and suffix) together. If both are non-blank, a joiner is placed in between. If either (or both) is blank, the joiner is not used. In some ways this is the reverse of the sandwich(function.

New Functions For Converting Between Numbers and Strings

These new functions convert numbers into strings and strings into numbers.

Function	Reference Page	Description
dollarsandcents(number)		This function converts a number to text formatted as dollars and cents (for example 98123.45 becomes Ninety eight thousand one hundred twenty three dollars and 45 cents).
hex(string)		Converts text with hex characters into a number. For example the value of hex("0C2") is 194.
hexbyte(number)		Converts a number to text formatted as a two digit hexadecimal number. For example the result of hexbyte(68) is 44.
hexlong(number)		Converts a number to text formatted as a eight digit hexadecimal number. For example the result of hexlong(68) is 00000044.
hexstr(number)		Converts a number to text formatted as a hexadecimal number. For example the result of hexstr(68) is 00000044. This function can also accept binary values with more than 4 bytes (see "Processing / Transforming Binary Data" on page 598 of *Formulas & Programming*).
hexword(number)		Converts a number to text formatted as a four digit hexadecimal number. For example the result of hexword(68) is 0044.
money(number)		Converts a number to text, formatted with commas every three digits and two digits after the decimal point (for example 98,123.45).
scientificnotation(number)		Converts a number to text, formatted in scientific notation with three places after the decimal point (for example 9.812e+4).
zbpattern(number,pattern)		This function displays a number using a pattern. Unlike the normal pattern(function, the zbpattern(function will output "" if the number is zero. (Note: zb is short for zeroblank.)

ASCII Character Constant Functions

These new functions return common ASCII characters.

Function	Description
info("lineseparator")	This function returns the line separator character on the current platform. On Macintosh systems this is a carriage return. On Windows PC systems this is a carriage return followed by a linefeed (CR-LF).
cr()	This function generates a carriage return. This is equivalent to chr(13) and is also the same as ¶.
crlf()	This function generates a carriage return line feed. This is equivalent to chr(13)+chr(10).
lf()	This function generates a line feed. This is equivalent to chr(10).
tab()	This function generates a tab character. This is equivalent to chr(9) and is also the same as ¬.
vtab()	This function generates a vertical tab character. This is equivalent to chr(11).

Working With Arrays

These new functions manipulate Text Arrays.

Function	Description
arraycontains(text,item,sep)	This procedure checks to see if any element of an array matches the specified text. For the result to be true, the array element must match the specified text exactly, including upper and lower case. Otherwise the function will return false. So checking for "Green" will only match that exact array element, not "green" or "Olive Green". (Note that this is quite different from the contains operator, which ignores upper and lower case and allows a submatch.)
arraydeletevalue(text,value,sep)	This function deletes any array elements that match the value parameter. This mush be an exact match, including upper and lower case. If the value occurs multiple times in the array, every occurrence of the value will be removed, with one exception: If the value occurs in two consecutive array elements, only the first occurrence will be deleted.
arrayfirst(text,sep)	This function extracts the first element of an array.
arraylast(text,sep)	This function extracts the last element of an array.
arraylefttrim(text,count,sep)	Removes the first elements of an array. For example arraylefttrim(text,2,",") removes the first two elements from a comma separated array.
arraylookup(text,key,mainsep, subsep,default)	This function looks up a value in a double column table, similar to a lookup. The first column in the table is the key value, the second column is the data value. Each line in the table is separated by the mainsep character, while the two columns in each line are separated by the subsep character. If no match is found the default value is returned. For example this function: arraylookup("AL.ALABAMA;AK.ALASKA; ... WY.WYOMING",State,":",".","") can be used to look up a long state name given the two letter abbreviation.

arraymerge(array1,array2, separator,joiner)	This function merges two text arrays together. This function has four parameters: array1, array2, separator and joiner . Array1 is the first text array you want to merge, array2 is the second array. Separator is the separator character for both arrays (in other words, both arrays must use the same separator). This should be a single character. For carriage return delimited arrays, use the ¶ character (option-7). For tab delimited arrays use the ¬ character (option-L). Joiner is from 1 to 10 characters of text that will be used to join the individual elements of the two arrays.
	
	The result is a new array with the elements of the original arrays joined together. This means that the first element of the first array will be joined with the first element of the second array, then the second element of the second array will be joined with the second element of the second array, and so on until both arrays are completely merged.
	
	The example could be used to display names and phone numbers from a contact database in a Text Display SuperObject.
	
	`arraymerge(`
	`lookupall( "Contacts","Company","Name",¶),`
	`lookupall( "Contacts","Company","Phone",¶),`
	`¶,", ")`
	
	The display will look something like the text shown below.
	
	John Smith , (510) 323-4905
	Susan Wilson , (510) 590-1341
	Bill Franklin , (510) 323-6781

Function	Description
arrayreplacevalue(text, oldvalue,newvalue,sep)	This function replaces any array elements that match the value parameter. This must be an exact match, including upper and lower case. If the value occurs multiple times in the array, every occurrence of the value will be replaced, with one exception. If the value occurs in two consecutive array elements, only the first occurrence will be replaced.
arrayreverselookup(text,key, mainsep,subsep,default)	This function looks up a value in a double column table, similar to a lookup. The first column in the table is the key value, the second column is the data value. However, this function reverses the function of these two columns. The key parameter is looked up in the second column, then the associated value is returned from the first column. This is the reverse of the arraylookup(function. Each line in the table is separated by the mainsep character, while the two columns in each line are separated by the subsep character. If no match is found the default value is returned. For example this function: arrayreverselookup("AL.ALABAMA;AK.ALASKA; ... WY.WYOMING",State,":",".","") can be used to look up a two letter state abbreviation given the full name of the state.
arraytrim(text,count,sep)	This function removes the last elements of an array. For example arraytrim(text,2,",") removes the last two elements from a comma separated array.

New HTML Tag and Tag Parsing Functions

This new function manipulates HTML tags.

Function	Description
striphtmltags(text)	This function removes all HTML tags from the text.

New HTML Table Parsing Functions

These new functions are specifically designed for extracting data from an HTML table.

Function	Description
htmltablecell(table,row,cell)	This function extracts the data from a cell in an HTML table. Any HTML tags in the cell are removed, leaving only the actual text. The thetable parameter must contain the body of an html table with <tr> and <td> tags (however, the actual <table> and </table> tags themselves are not required.)
htmltablecellexists(table,row,cell)	This function checks to see whether a cell in an HTML table exists or not. The result will be true if the cell exists, or false if it doesn't. The thetable parameter must contain the body of an html table with <tr> and <td> tags (however, the actual <table> and </table> tags themselves are not required.)
htmltablecellraw(table,row,cell)	This function extracts the data from a cell in an HTML table. Unlike the htmltablecell(function, any HTML tags in the cell are retained. The thetable parameter must contain the body of an html table with <tr> and <td> tags (however, the actual <table> and </table> tags themselves are not required.)
htmltableheight(table)	This function calculates the height (number of rows) in an HTML table. It assumes that the table is a regular matrix (no rowspan tags).
htmltablerowraw(table,row,cell)	This function extracts the data from a row in an HTML table. Any HTML tags in the row are retained. The the table parameter must contain the body of an html table with <tr> and <td> tags (however, the actual <table> and </table> tags themselves are not required.)
htmltablewidth(table)	This function calculates the height (number of rows) in an HTML table. It assumes that the table is a regular matrix (no colspan tags) and that all of the rows have the same number of columns as the top row in the table.

New HTML/URL Conversion Functions

These new functions help parse web URLs.

Function	Description
urlfilename(url)	This function extracts the filename from a complete url.
urlpath(url)	This function extracts the path from a url.

New HTML Generating Functions

These new functions help with generating HTML.

Function	Description
htmlbold(string)	This function takes the text and adds and tags to it.
htmlitalic(string)	This function takes the text and adds and tags to it.

New Functions for Converting Between Dates and Text

These new functions allow you to convert a date into text, or text into a date. Most of these are simply shortcuts for the **`datepattern(`** function.

Function	Description
completedatestr(number)	Convert a date to text, including the day of the week (for example Sunday, April 20th, 2003).
datestr(number)	Convert a date to text using format mm/dd/yy (for example 4/20/03).
daystr(number)	Convert a date to the day of the week (for example Sunday).
eurodatestr(number)	Convert a date to text in European format (for example 20-APR-2003).
longdatestr(number)	Convert a date to text with format Month ddnth, yyyy (for example April 20th, 2003).

New Functions for Converting Between Times and Text

This new function is a shortcut for the **`timepattern(`** function.

Function	Description
timestr(number)	Convert a number to text in am/pm time format (for example 9:34 AM).

Time Calculations with Text

These new functions operate with time values in strings. There aren't as many functions available as for times expressed as numbers, but if your input and output values will be in strings using these function saves the intermediate conversion steps.

Function	Description
texttimedifference(start,end)	This function calculates the difference between two times. Instead of being expressed as numbers, the input output times are expressed as text (for example 12:45 pm). This function works correctly even if the interval between the two times crosses over midnight. This function returns a time interval between -12 and +12 hours. See also the timeinterval(function, which returns a time interval between 0 and 24 hours. There are two parameters, start and end. Start is a string representing the starting point of the time interval. End is a string representing the ending point of the time interval. This function returns the time difference between the start and end. For example, if the start time is 9:30 PM and the end time is 2:05 AM, the difference would be 4:35. But if the parameters are reversed and the start is 2:05 AM and the end is 9:30 PM, the difference is -4:35. If the result is positive, the end is after the start. But if the result is negative, the start is after the end.
texttimeinterval(start,end)	This function calculates the time interval between two times. It works correctly even if the interval between the two times crosses over midnight. This function returns a time interval between 0 and 24 hours. See also the timedifference(function, which returns a time interval between -12 and +12 hours. There are two parameters, start and end. Start is a string representing the starting point of the time interval. End is a string representing the ending point of the time interval. This function returns the time between the start and end. For example, if the start tine is 9:30 PM and the end time is 2:05 AM, the interval would be 4:35. But if the parameters are reversed and the start time is 2:05 AM and the end time is 9:30 PM, the interval is 19:25.

New SuperDate Conversion Functions

These new functions perform common calculations and conversions with superdates.

Function	Description
superdatestr(number)	Converts a number containing a superdate to text in standard format (for example 4/20/03 9:56 AM).
supernow()	This function returns the number representing the current date and time as a superdate.

True/False Values

For purposes of calculation, Panorama treats true and false as numbers: true is -1 and false is zero. Panorama also has two new functions that directly generate these values.

Function	Description
true()	This function always returns true (-1).
false()	This function always returns false (0).

Multiple Field LookupAll Functions

Panorama V includes seven new variations on the **lookupall(** function that lookup from two to eight fields from the target database, instead of just one. Each of these functions includes a parameter for controlling the separator between records and the separator between each fields.

Function	Description
lookupalldouble(thedb, keyfield,keyvalue,datafield1, datafield2,mainsep,subsep)	This function does a lookupall for two fields (double). The lookup is of all records in thedb where the keyfield matches the supplied keyvalue. For each record datafield1 and datafield2 are joined together with the subsep separator (which may be more than one character) and each record is joined together with the mainsep characters.
lookupalltriple(thedb, keyfield,keyvalue,datafield1, datafield2,datafield3, mainsep,subsep)	This function does a lookupall for three fields (triple). The lookup is of all records in thedb where the keyfield matches the supplied keyvalue. For each record datafield1 through datafield3 are joined together with the subsep separator (which may be more than one character) and each record is joined together with the mainsep characters.
lookupallquadruple(thedb, keyfield,keyvalue,datafield1, datafield2,datafield3,datafield4, mainsep,subsep)	This function does a lookupall for four fields (quadruple). The lookup is of all records in thedb where the keyfield matches the supplied keyvalue. For each record datafield1 through datafield4 are joined together with the subsep separator (which may be more than one character) and each record is joined together with the mainsep characters.
lookupallquintuplet(thedb, keyfield,keyvalue,datafield1, datafield2,datafield3,datafield4, datafield5,mainsep,subsep)	This function does a lookupall for five fields (quintuplet). The lookup is of all records in thedb where the keyfield matches the supplied keyvalue. For each record datafield1 through datafield5 are joined together with the subsep separator (which may be more than one character) and each record is joined together with the mainsep characters.
lookupallsextet(thedb,keyfield, keyvalue,datafield1, datafield2,datafield3,datafield4, datafield5,datafield6, mainsep,subsep)	This function does a lookupall for six fields (sextet). The lookup is of all records in thedb where the keyfield matches the supplied keyvalue. For each record datafield1 through datafield6 are joined together with the subsep separator (which may be more than one character) and each record is joined together with the mainsep characters.
lookupallseptuplet(thedb,keyfield, keyvalue,datafield1, datafield2,datafield3,datafield4, datafield5,datafield6,datafield7, mainsep,subsep)	This function does a lookupall for seven fields (septuplet). The lookup is of all records in thedb where the keyfield matches the supplied keyvalue. For each record datafield1 through datafield7 are joined together with the subsep separator (which may be more than one character) and each record is joined together with the mainsep characters.
lookupalloctet(thedb,keyfield, keyvalue,datafield1, datafield2,datafield3,datafield4, datafield5,datafield6,datafield7, datafield8,mainsep,subsep)	This function does a lookupall for eight fields (octet). The lookup is of all records in thedb where the keyfield matches the supplied keyvalue. For each record datafield1 through datafield8 are joined together with the subsep separator (which may be more than one character) and each record is joined together with the mainsep characters.

New US Post Office Abbreviation Functions

These new functions return text arrays that contain lists of official US Post Office abbreviations. These functions are designed to be used with the **arraylookup(** and **arrayreverselookup(** functions.

Function	Description
stateabbreviations()	This function returns a list of state abbreviations in this format: AL:ALABAMA;AK:ALASKA; ... This table is designed to be used with the arraylookup(and arraylookupreverse(functions.
statelookup(state)	This function looks up the name of a state from the state abbreviation (for example CALIFORNIA from CA). If the parameter does not match any state then the original value is returned.
uspssecondaryunits()	This function returns a list of USPS secondary suffix designation abbreviations in this format: APT:APARTMENT;RM:ROOM; ... This table is designed to be used with the arraylookup(and arraylookupreverse(functions.
uspsstreetsuffixes()	This function returns a list of USPS street suffix abbreviations in this format: ALY:ALLEY;AVE:AVENUE; ... This table is designed to be used with the arraylookup(and arraylookupreverse(functions.

Converting Points and Rectangles to Text

These new functions convert points and rectangles to text. This is handy if you want to display the location / dimensions of the point or rectangle in a dialog or report.

Function	Description
pointstr(point)	This function converts a point value into text in the format V,H (for example 34,56).
rectanglesizestr(rect)	This function converts a rectangle into text in the format TOP,LEFT,HEIGHT,WIDTH (for example 100,20,80,30).
rectanglestr(rect)	Converts a rectangle into text in the format TOP,LEFT,BOTTOM,RIGHT (for example 100,20,180,50).

New Color Functions

These new functions help generate common colors and in converting colors from Panorama format to / from HTML format.

Function	Description
black()	This function returns black (it is equivalent to rgb(0,0,0)).
gray(saturation)	This function returns a gray color. The saturation value is between 0 (white) and 100 (black).
htmlrgb(string)	This function converts text formatted as an HTML color (for example FFFFFF for white or FF0000 for red) into a Panorama color value.
htmlrgbstr(color)	This function converts a Panorama color value into text formatted as an HTML color (for example BB4DFD).
white()	This function returns white (it is equivalent to rgb(65535,65535,65535)).

New Binary Data Functions

These new functions process bits, bytes, words, and longwords.

Function	Description
bit(number)	This function converts a bit number (1 to 32) into a number (1, 2, 4, 8, 16, etc.)
bytearray(data,index)	This function extracts a value from an array of bytes. This is not a Panorama style delimited array but a C style array of 8 bit values. The result is an integer.
chararray(data,index)	This function extracts a characters from an array of characters. This is not a Panorama style delimited array but a C style array of ASCII characters. The result is an single characters.
chunkarray(data,index, chunklength)	This function extracts a binary chunk from an array of chunks. This is not a Panorama style delimited array but a C style array of binary chunks. The result is a binary value (text).
getbit(number,bitnumber)	This function returns a true or false value by testing a bit. The bit number may be from 1 to 32. If the bit is set, the value will be true, if it is not set, the value will be false.
hex(string)	Converts text with hex characters into a number. For example the value of hex("0C2") is 194.
longwordarray(data,index)	This function extracts a value from an array of longwords. This is not a Panorama style delimited array but a C style array of 32 bit values. The result is an integer.
onescomplement(number)	This function returns the one's complement of a 32 bit number (all bits are reversed)
setbit(number,bitnumber,truefalse)	This function sets one bit within a number, without disturbing any of the other bits. The bit number may be from 1 to 32. The bit will be set based on the true/false parameter - set if true, cleared if false.
wordarray(data,index)	This function extracts a value from an array of words. This is not a Panorama style delimited array but a C style array of 16 bit values. The result is an integer.

New Functions for Accessing Disk Files and Folders

These new functions directly access files and directories on the disk. The functions below read information from the disk. For a more detailed discussion of file i/o, including writing to files, see “Directly Reading and Writing Disk Files” on page 421 of *Formulas & Programming*.

Function	Description
info("foldersepchar")	Returns the folder separator character for the current platform (: or \).
info("lineseparator")	This function returns the line separator character on the current platform. On Macintosh systems this is a carriage return. On Windows PC systems this is a carriage return followed by a linefeed (CR-LF).
filedate(folder,file)	Returns the creation date of the specified file.
fileexists(folder,file)	Checks to see if a file exists. Returns true or false.
fileextension(text)	This function extracts the extension (if any) from a complete path and filename. For example the function fileextension("Alaska:Denali:Image45.jpg") would return .jpg.
filename(text)	This function extracts the filename from a complete path and filename. For example the function filename("Alaska:Denali:Image45.jpg") would return Image45.jpg.
filepath(text)	This function extracts the path from a combined path and filename. For example the function filepath("Alaska:Denali:Image45.jpg") would return the text "Alaska:Denali:".

Function	Description
filetime(folder,file)	Returns the creation time of the current file.
filetypecreator(folder,file)	Returns 8 characters - 4 characters with the file type and 4 characters with the creator code (for example TEXTR*ch for BBEdit text files).
folderexists(folder,foldnername)	Checks to see if a folder exists. Returns true or false.
foldersepchar()	Returns the folder separator character for the current platform (: or \).
fullpath(path)	This function converts a filename or relative path (starting with the : or / symbol) into a full path, including the disk name.
listpatnoramafiles(folder)	Returns a list of Panorama database files in the specified folder (carriage return separated).
listtextfiles(folder)	Returns a list of text files in the specified folder (carriage return separated).
panoramafolder()	This function returns a **folderid** that unambiguously describes the location of the folder containing the Panorama application. This **folderid** can be used in other functions and statements.
posixpath(path)	This function converts a MacOS format path and filename into a POSIX path that can be used as a parameter to a shell command. The input parameter must be a FULL path, including the disk name. Relative pathnames are not allowed.
panoramasubfolder(subpath)	This function makes it easy to reference any subfolder of the Panorama folder. Or you can leave it blank ("") to reference the main Panorama folder.
pathexists(path)	Checks to see if a path exists. Returns true or false.
posixpath(path)	This function converts a MacOS format path and filename into a POSIX path that can be used as a parameter to a shell command. The input parameter must be a FULL path, including the disk name. Relative pathnames are not allowed.
subfolder(folder,subfolder)	This function returns the folder ID of a subfolder of a specified folder. For example the function subfolder(info("systemfolder"),"Extensions") returns the folder ID of the Extensions folder within the System folder.
urlfilename(url)	This function extracts the filename from a complete url.
urlpath(url)	This function extracts the path from a url.

New System Information Functions

These functions return information about the computer system — the keyboard, mouse, memory, clipboard and Panorama itself.

Function	Description
info("availablescreenrectangle")	This function returns a rectangle defining the edges of the main screen (the screen that contains the menu bar). Any area of the screen that is reserved for the operating system (for example the dock in OS X) is excluded from the rectangle. The rectangle is in screen relative coordinates
info("computername")	This function returns the short name of the computer (OS X only). This is the computer name that needs to be used in terminal sessions.
info("fonts")	This function returns a carriage return delimited array of all available fonts. This list is created when Panorama launches, and is not updated to include any new fonts that may have been added to the system since then.
info("unixusername")	This function returns the short user name the user has logged in under (Mac OS X only). This is the user name that needs to be used in terminal sessions.

Function	Description
os9()	This function returns true if running on Mac OS 9, otherwise false.
oswindows()	This function returns true if running on Microsoft Windows, otherwise false.
osx()	This function returns true if running on Mac OS X, otherwise false.

New User Information Functions

These new functions return information about the person that is currently using the computer.

Function	Description
info("unixusername")	This function returns the short user name the user has logged in under (Mac OS X only). This is the user name that needs to be used in terminal sessions.

New Database Information Functions

These new functions return information about the currently active database.

Function	Description
dbfolder()	This function returns the folder ID of the folder the current database is located in.
dbmetatag(database,attributes)	This function returns a piece of metadata information for the specified database (if any). This second parameter specifies the metadata item you want to retrieve - legal items are: Version, Title, Project, Authors, Keywords, Description. These are all case sensitive. For example, you can retrieve the document keywords with the formula dbmetatag(myDatabase,"Keywords").
dbmetatagclose()	This function returns the tag that appears after the metadata database information. This tag helps the Tiger (Mac OS X 10.4) metadata import function to locate the metadata.
dbmetatagdictionary(database)	This function returns the metadata database information for the specified database (if any). This metadata contains information like the database author, keyword and description.
dbmetatagopen()	This function returns the tag that appears in front of the metadata database information. This tag helps the Tiger (Mac OS X 10.4) metadata import function to locate the metadata.
dbname()	This function returns the name of the current database.
dbpath()	This function returns the path of the folder the current database is located in.
dbsubfolder(subpath)	This function returns the folder ID of a subfolder within the same folder as the current database. For example if the current database is in the folder "MyDrive:My Stuff", the function dbsubfolder("Images") returns the folder ID of the "MyDrive:My Stuff:Images" folder.

info("proceduredatabase")	This function returns the name of the database that contains the procedure itself, even if another database is currently active. It's primarily useful when the farcall statement is used to call the procedure (or when the procedure is being used as a custom statement). Here's an example that opens the Help Window form in the database that contains the procedure, even if a window from another database is currently on top,
	`window info("proceduredatabase")+":SECRET"`
	`openform "Help Window"`

Function	Description
info("visible")	This function returns true/false result based on whether or not the current record is visible. Useful for situations where Panorama may scan invisible records, for example the arraybuild and select statements, also the Scrolling List SuperObject.

New Window, Form and Report Information Functions

These new functions return information about windows, forms and reports.

Function	Description
info("matrixcelldata")	This function returns the data associated with the current matrix cell. This function is only valid if a formula has been defined for this Super Matrix Object. Note: It's up to you what to do with this data. You can display it using a Text or Flash Art object, or simply ignore it if you want. The Matrix object doesn't display this data automatically. Example: The formula below could be used in a Text Display SuperObject to display the cell number and data. `str(info("matrixcell"))+" "+info("matrixcelldata")`

info("matrixdata")	This function returns the all of the data associated with the current matrix. This function is only valid if a formula has been defined for this Super Matrix Object. Note: It's up to you what to do with this data. You can display it using a Text or Flash Art object, or simply ignore it if you want. The Matrix object doesn't display this data automatically.
	Example: The formula below could be used in a Text Display SuperObject to display the data and the cell number and position, for example Orange [4 of 17] .
	`info("matrixcelldata")+" ["`
	`+str(info("matrixcell") )+" of "+`
	`str(arraysize(info("matrixdata")+`
	`, info("matrixseparator") ) +"]"`

info("matrixseparator")	This function returns the separator associated with the current matrix. This function is only valid if a formula has been defined for this Super Matrix Object.
	Example: The formula below could be used in a Text Display SuperObject to display the data and the cell number and position, for example Orange [4 of 17] .
	`info("matrixcelldata")+`
	`" ["+str(info("matrixcell"))+`
	`" of "+str(arraysize(info("matrixdata")+`
	`, info("matrixseparator"))+"]"`

info("pagecount")	The new info("pagecount") function calculates the total number of pages that will be printed. For example, if you wanted to print Page 1 of 4, Page 2 of 4, etc. on the top of each page you would use a formula of
	`"Page "+str(info("pagenumber"))+" of "`
	`+str(info("pagecount"))`
	This function is only valid when used in an object in a form (Text Display SuperObject, auto-wrap text object, etc.) that is being printed.

The Assign Function

The **assign(** function is so different from every other Panorama V function that we gave it its own section, all by itself. Instead of calculating a new value based on its parameters, the assign(function allows you to assign the result of a partial formula to a field or variable. This is called a "side effect" because this assignment is in addition to the normal operation of the formula that contains the assign(function (or multiple assign(functions). To learn more about this function see “The Assign Function” on page 144 of *Formulas & Programming*.

Comments in Formulas

Panorama V now allows “comments” to be placed inside a formula. A comment is a note within the formula that is ignored when the formula is evaluated. A comment must start with /* and end with */. Anything between these will be ignored. (C and JavaScript programmers will recognize this style.) For example, this formula:

```
«CRatio» /* CRatio must be updated every 24 hours */ * Amount
```

will produce the same value as this one:

```
«CRatio» * Amount
```

In addition to notes to yourself, comments are also useful for temporarily disabling a section of a long formula (for example if you are trying to debug the formula).

Custom Functions

Panorama V doesn’t limit you to the built-in functions that are supplied with a Panorama. In fact for the first time you can actually create your own user defined functions that can be used in any formula. To build a custom function you assemble it from the functions and operators that already exist. For example, you could define a new **money(** function with one parameter:

```
money(number)
```

This custom function can be defined using Panorama's built in **pattern(** function, like this.

```
pattern(number,"#,.##")
```

Once the function has been defined you can use it in any formula. For example, the formula

```
money(54321.5678)
```

would result in the text value 54,321.56.

To learn how to create your own custom functions see “Custom Functions” on page 197 of the *Panorama Handbook*.

Procedures

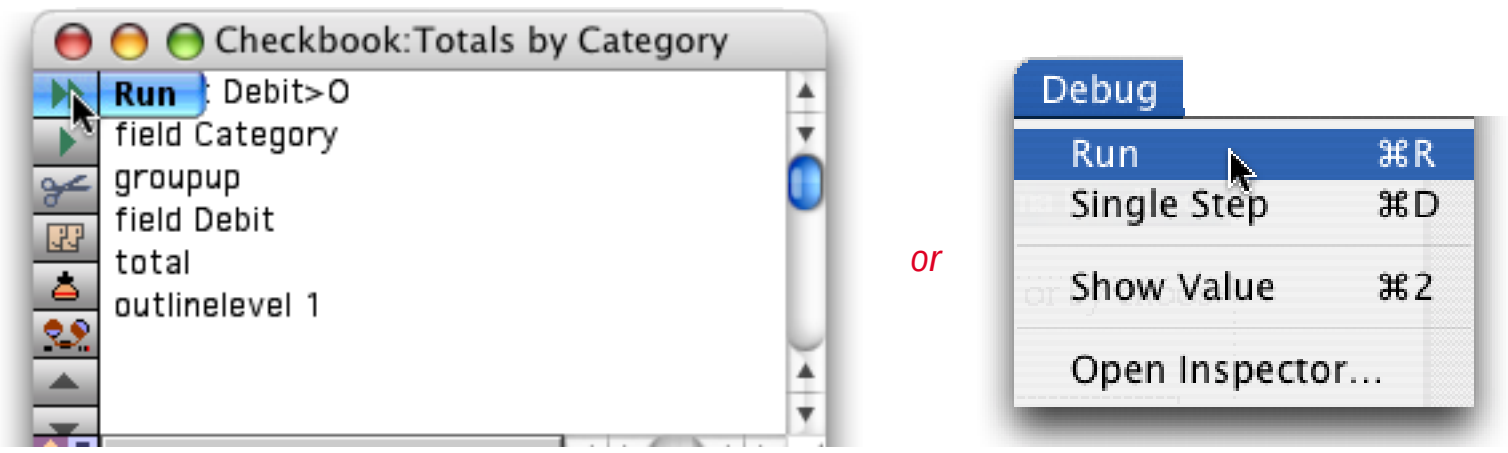
Panorama V includes hundreds of new statements as well as new features designed to make writing and debugging procedures easier. In addition to the enhancements listed below, we've also greatly enhanced several wizards relating to programming, see "View Wizard" on page 90, "Programming Reference Wizard" on page 91, "Formula Wizard" on page 92, "Icons & Backgrounds Wizard" on page 92, "Cross Reference Wizard" on page 93, "Custom Statement and Function Wizards" on page 94, "Dialog Workshop Wizard" on page 95, "Dropalyzer Wizard" on page 96, "Elastic Picture Workshop" on page 96 and "Variables Wizard" on page 97.

Undo

The procedure editing window now supports the **Undo** command. You can **Undo** any operation that modifies the text of the procedure.

Easy Procedure Triggering

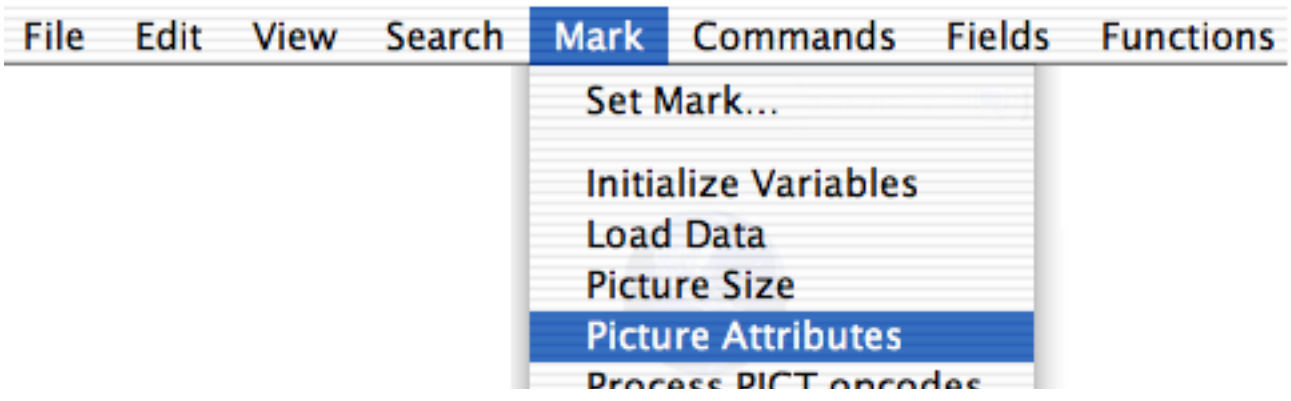
In previous versions of Panorama a procedure had to be triggered from the Action menu, from a button, or using an automatic trigger. Panorama V now allows you to trigger a procedure directly from the procedure window itself. If the procedure itself is currently the top window you can trigger it by clicking on the **Run** tool, or by choosing **Run** from the Debug menu. Using this technique you can even run procedures that would normally be impossible to trigger manually, for example procedures with names that start with period.



You can also use the **Single Step** command to start a procedure the same way. For more information see "Trying Out a Procedure" on page 218 of *Formulas & Programming*.

Organizing Large Procedures (The Mark Menu)

A single procedure may include up to 32,000 characters of text. A procedure that long would be more than 20 pages long if printed. As a procedure grows it can be difficult to navigate within the procedure itself. The **Mark** menu allows you to create "bookmarks" within the procedure. These marks are listed in the **Mark** menu.

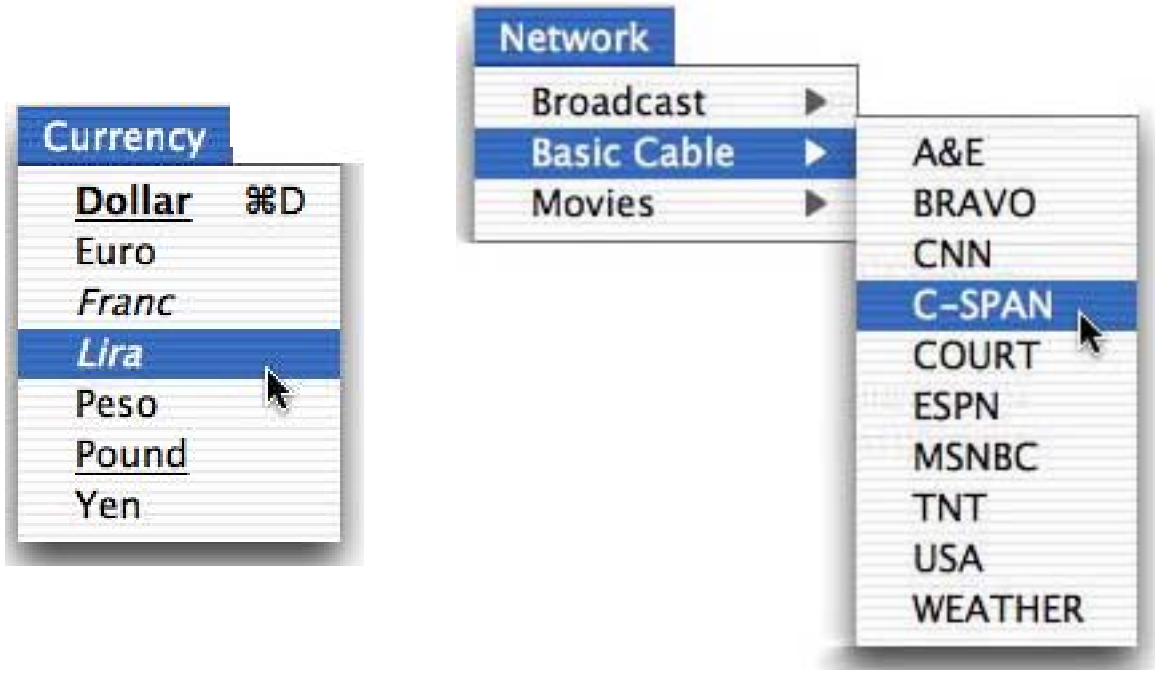


Choosing an item from this menu causes the editor to jump to the location in the procedure text corresponding to the mark. For more information see "Organizing Large Procedures (The Mark Menu)" on page 305 of *Formulas & Programming*.

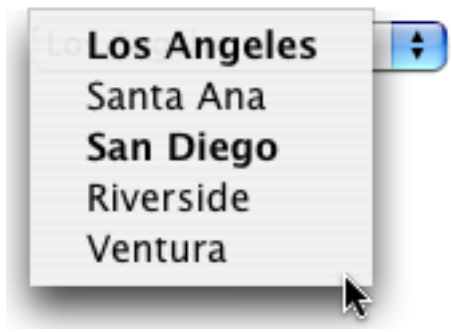
Live Menus

Prior to Panorama V there were two methods for creating menus in the menu bar: 1) Custom menus (created with resources) and 2) Action menus. Both of these methods have drawbacks in some applications. Creating custom menus with resources gives you pretty good control over the menus. Because it requires creating a separate resource file, this technique is very cumbersome. Action menus are very easy to set up, but provide only very limited control. For example, Action menus don't allow submenus, and they don't allow you to change menus on the fly.

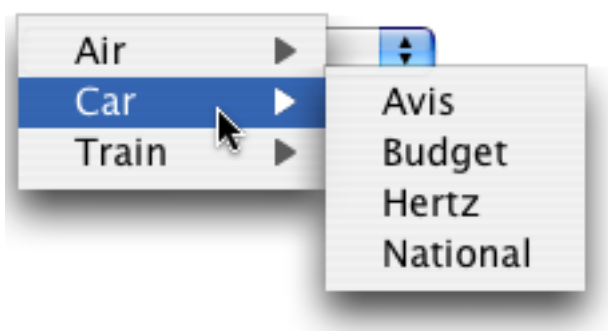
Panorama V provides a new method for designing a custom menu bar: **Live Menus**. Unlike custom menus, no separate resource file is required. Unlike Action menus, Live Menus give you complete control over the content and appearance of each menu. In fact, you have even better control than resource based custom menus. If you are creating a new database that needs it's own menus, Live Menus are the way to go. You can still use resource based custom menus, but in most cases you'll find that you want to convert to the new Live Menu system. Here are some examples of Live Menus in action.



To learn how to set up your own Live Menus see "Live Menus" on page 362 of *Formulas & Programming*. You can also create Live Pop-Up menus with different styles for each menu item (see ""Live" Pop-Up Menu Formulas" on page 864 of the *Panorama Handbook*).



You can even create a pop-up menu with submenus (see "Pop Up Submenus" on page 865 of the *Panorama Handbook*).



Important Note for Custom Menu Developers

Starting with Panorama V, resource based custom menus need to avoid the area around menu #200. Live menus use numbers from 200 up. Wizard menus use menus from 199 down. So for now that means custom menus should avoid the area between 190 and 210. For future expansion we recommend avoiding the area between 170 and 230. Or better yet, switch your custom menus to Live menus!

..CloseDatabase

The new **..CloseDatabase** procedure allows you to customize the actions that occur when a database is closed. See "..CloseDatabase" on page 397 of *Formulas & Programming*.

New Programming Statements

Panorama V includes over 200 new statements, listed below in alphabetical order (we've **highlighted** some of the most interesting statements). For more information on any statement see the **Programming Reference** wizard.

Statement	Parameters	Description
abortcheck	prompt	This statement checks to see if the user has pressed Command-Period to abort. If he or she has, info("trigger") is set to "Abort", otherwise it is set to "".
alertcanceldelete	message	This statement will produce a CancelDelete alert that uses the info("DialogTrigger") function to learn the users response. "Cancel" is the default.
alertcancelok	message	This statement will produce a CancelOK alert that uses the info("DialogTrigger") function to learn the users response instead of the clipboard.
alertdeletecancel	message	This statement will produce a DeleteCancel alert that uses the info("DialogTrigger") function to learn the users response. "Delete" is the default.
alertnoyes	message	This statement will produce a NoYes alert that uses the info("DialogTrigger") function to learn the users response instead of the clipboard.
alertok	message	This statement will produce an alert with just an Ok button (similar to the message statement).
alertokcancel	message	This statement will produce an OKCancel alert that uses the info("DialogTrigger") function to learn the users response instead of the clipboard.
alertoksmall	message	This statement will produce a small alert with just an Ok button (similar to the message statement).
alertrevertcancel	message	This statement will produce an alert with Revert and Cancel buttons. This is the same alert that appears when you choose the Revert to Saved command. The procedure can use the info("dialogtrigger") function to find out which button was pressed.
alertsavecancel	message	This statement will produce an alert with Save, DontSaveandCancelbuttons.Thisisthesamealertthatappearswhenyoucloseawindow(ifitisthelastopenwindowforadatabase)orquitPanorama.Theprocedurecanusetheinfo("dialogtrigger")functiontofindoutwhichbuttonwaspressed.'
alertyesno	message	This statement will produce a YesNo alert that uses the info("DialogTrigger") function to learn the users response instead of the clipboard.
alertyesnocancel	message	This statement will produce a YesNoCancel alert that uses the info("DialogTrigger") function to learn the users response. "Yes" is the default.

Statement	Parameters	Description
append	data,text	This statement appends a text string to a field or variable. If the field or variable doesnthaveavalueyet,thetextstringissimplyassigned-totheit.'
appenddataforktofile	source,destination, typecreator,progress, buffersize	The APPENDDATAFORKTOFILE statement copies the data fork of a file to the end of an existing file. If the destination file does not exist it is created. If the destination folder does not exist it is created.
appenddictionary-value	dictionary,name, separator,value	This statement is part of a suite that manipulates items in a key / value dictionary. This is not a language dictionary for spelling checks, but a dictionary that allows you to define words (names) and their values (definitions). The AppendDictionaryValue statement appends a value to an item, given its name.
appendline	data,text	This statement appends a text string to a field or variable on a new line. If the field or variable doesnthaveavalueyet,thetextstringissim-plyassignedtotheit.'
appendrezforktofile	source,destination, typecreator,progress, buffersize	The APPENDDATAFORKTOFILE statement copies the resource fork of a file to the end of an existing file. The data is appended to the data fork of the target file. If the destination file does not exist it is created. If the destination folder does not exist it is created.
arrayclairvoyance	input,outputlist,separa-tor,listobject, inputlist,compare	The ARRAYCLAIRVYOANCE statement searches an array to build an array. The array is updated "live" as it is built.
arraynumericsort	inputarray, outputarray,sep	This statement sorts all the numeric elements of a text array in proper numeric order.
arraynumerictotal	inputarray,sep, numerictotal	This statement totals all the numeric elements of a text array.
arrayrandomize	oldarray,newarray,sep	This statement reorders an array randomly.
arraystrip	array,separator	This statement strips blank elements from an array.
arraysubset	inputarray,outputarray, separator,formula	This statement filters an array based on a formula. It scans the array using the formula. If the formula is true, the array element will be retained in the final array. If the formula is false, the element will be removed from the output array.
arraytoggle	input,text,separator	This statement "toggles" a value within an array. If the value exists within the array, it is removed. If the value does not exist within the array, it is added to the end of the array.
arrayunpropagate	input,text,separator	This statement scans an array from top to bottom. If it finds two or more duplicate values in a row, it blanks out all but the first. This is similar to PanoramasUNPROPAGATEcommandintheMath-menu,butforanarrayinsteadofadatabasefield.'
basesixtyfourdecode	base64,raw	The BASE64DECODE statement converts BASE64 encoded data back into the original data. BASE64 is a common encoding format for e-mail and other internet protocols.
basesixtyfourencode	raw,base64	The BASE64ENCODE statement converts data into encoded BASE64 data. BASE64 is a common encoding format for e-mail and other internet protocols.
batchreplace	data,array,mainsep, subsep	The BatchReplace statement performs multiple find and replace operations on a piece of text.
bigmessage	message	This statement will display a large message alert. Keep in mind that the text in an alert is limited to 255 characters.
buildwizardmenus		This statement scans the wizard folder to build the wizard menu and submenus.
calcenclosingrectangle	formula,rectangle	This statement will calculate the rectangle surrounding all of the specified objects.

Statement	Parameters	Description
cautionalert	resource,message	The cautionalert statement allows a Panorama procedure to open an alert dialog and display a specified message within that dialog. A "warning" icon will appear in the upper left hand corner of the alert, like this:
changedictionar-yname	dictionary, oldname,newname	This statement is part of a suite that manipulates items in a key / value dictionary. This is not a language dictionary for spelling checks, but a dictionary that allows you to define words (names) and their values (definitions). The ChangeDictionaryName statement changes the name of an of the item in a dictionary. The contents (value) of the item is not changed.
channelactivemodule	channel,module	This statement returns the currently active module for a channel, if any.
channelcall	type,procedure,extras	This statement calls a procedure in the active channel library.
channelgetdictionary	type,module,dictionary	This statement loads the channel settings dictionary into a field or variable.
channelinformation	type,plugin,option,value	This statement retrieves information about the this module.
channelload	type,plugin	This statement loads a channel module.
channelmodules	channel,module	This statement returns a carriage return delimited list of modules that are available for a specified channel.
channelopen	type,plugin	This statement opens a channel module for editing.
channelpath	type,module,path	This statement returns the path that contains a channel module.
channelprocedures	type,plugin,procedures	This statement returns a list of procedures available in a specified channel.
channelsavedictionary	type,module,dictionary	This statement saves the channel settings dictionary.
channelscript	plugin,script,result, timeout,data	This statement calls an applescript in the same folder as a channel module. It can also pass parameters to the script, and waits for a response from the script.
channelswitch	type,plugin	This statement switches the active channel module.
channeltypes	types	This statement returns a list of channel types.
characterfilter	oldtext,newtext,formula	The characterfilter statement scans text on a character by character basis. As it scans the input text, it builds a new text field or variable. The contents of the new text is based on the result of the formula which is applied to each character of the original text. The formula can use the import() function to retrieve the original character, or the seq() function to retrieve the position of the character within the text..
checkword	word,status	This statement checks to see if a word is in the dictionary. If it is, the STATUS parameter is set to English.
chunkfilter	chunksize,oldtext, newtext,formula	The chunkfilter statement is almost the same as the characterfilter statement, but instead of single characters it allows you to process fixed length chunks. This statement is useful for base64 encryption, hex / ascii conversion, cryptography, etc. As it scans the input text, it builds a new text field or variable. The contents of the new text is based on the result of the formula which is applied to each character of the original text. The formula can use the import() function to retrieve the original chunks of characters, or the seq() function to retrieve the chunk number (first chunk is 1, next is 2, etc...
closelibrarywindow		This statement is intended to be used in place of the closewindow statement. Instead of closing the database, it makes it secret, keeping it in memory. Note: This command DOES NOT save the database.

Statement	Parameters	Description
closewindowkeepsecret		This statement is intended to be used in a ..CloseDatabase procedure. Instead of closing the database, it makes it secret, keeping it in memory. Especially useful for Library databases!! As a side effect, the database is also saved (since you wontgetanotherchanceonceitismadesecret).'
continueclose	yesno	The continueclose statement is designed to be used as part of a ..CloseDatabase procedure. This procedure is automatically triggered whenever Panorama is about to close a database. The continueclose statement can allow this process to continue or it can prevent Panorama from closing the database.
cookies	folder,file,options	This statement launches cURL. This command will not work on OS 9. For information on cURL see http://curl.haxx.se
copyfile	source,destination, buffersize	The COPYFILE statement copies a file to a new file. On MacOS systems it copies both the data fork and the resource fork. If the destination folder does not exist it is created.
copyfork	source,destination, buffersize	The COPYFORK statement copies a fork of a file to a new file. If the destination folder does not exist it is created. Use the statement to copy the data and resource forks separately (Mac OS only).
createpassword	characters, numbers,option, password	This statement creates a random password using characters or numbers (or a mixture) set to the chosen length.
curl	data,url	This statement launches cURL. This command will not work on OS 9. For information on cURL see http://curl.haxx.se
customnumbers	option	The customnumbers statement gives a programmer control over the formats used for numbers and dates within a procedure.
databasetogeneric	database,dictionary	This statement converts separate fields into a CommonFields dictionary for export to other database or VCards.
deletechildren	database,keyfield, keyvalue	This deletes all the records in another database that have a value in the key field that is equal to the key value. NOTE: As a side effect, all remaining records in the target database will be selected.
deletedictionaryvalue	dictionary,name	This statement is part of a suite that manipulates items in a key/value dictionary. This is not a language dictionary for spelling checks, but a dictionary that allows you to define words (names) and their values (definitions). The SetDictionaryValue statement changes the value of an item, given its name.
deleterecchildrenwarn	database,keyfield, keyvalue	This statement warns you before deleting the current record from the active database and all the records in another database that have a value in the key field that is equal to the key value.
deleterecordandchildren	database,keyfield, keyvalue	This deletes the current record from the active database and all the records in another database that have a value in the key field that is equal to the key value.
dialdigits	digits	This statement dials one or more digits. The statement sends the digits exactly as specified, without attempting to adjust for area codes, etc.
dialphone	phonenumber	This statement dials a phone number. The statement will adjust the number according to the channel settings (area codes, country codes, prefix, etc.).
dictionaryvalueexists	dictionary,name,result	This statement is part of a suite that manipulates items in a key/value dictionary. This is not a language dictionary for spelling checks, but a dictionary that allows you to define words (names) and their values (definitions). The DictionaryValueExists statement checks to see if a value exists.

Statement	Parameters	Description
displaydata	message,options	This statement will display data in a large dialog with a scroll bar. The text displayed in the alert may be any length (it is NOT limited to 255 characters).
dragcomplete		The dragcomplete statement may be used in a procedure that handles data that has been dropped on a form. This statement tells Panorama that you have finished processing the data that was dragged, allowing Panorama to reclaim the memory that was used to store the dragged information.
dragdrop	rectangle,flavor,data	The dragdrop statement is used to start a drag operation. On MacOS data can be dragged to other applications as well as within Panorama, on Windows dragging is only allowed within Panorama.
dragdropdata		The dragdropdata statement is used in combination with the dragdrop and startdragdrop statements to start a multi-flavor drag operation. The dragdropdata statement is repeated for each flavor to be included in the drag operation. On MacOS data can be dragged to other applications as well as within Panorama, on Windows dragging is only allowed within Panorama.
draglistitem	flavor	This statement allows an item to be dragged from a list to any object that can received dragged text. This should be used in response to the user clicking on the list. The list object must have the click/release option turned OFF.
dragvcard		This statement drags from the current record into VCard format.
dragwithinlist	list,list,list	This statement allows an item to be dragged within a list, rearranging the order of the items within the list. This should be used in response to the user clicking on the list. The list object must have the click/release option turned OFF.
dropfromfinder	filter,files	This statement processes any files or folders that have been dropped on a form from the finder.
dropimportvcards		This statement is designed to be used in a .DropProcedure procedure to process any vCards that were stopped.
dumpdictionary	dictionary,dump	This statement is part of a suite that manipulates items in a key/value dictionary. This is not a language dictionary for spelling checks, but a dictionary that allows you to define words (names) and their values (definitions). The DumpDictionary statement returns the a cr separated array of dictionary values, each line displaying a key=value pair.
dumpdictionary-quoted	dictionary,dump	This statement is part of a suite that manipulates items in a key/value dictionary. This is not a language dictionary for spelling checks, but a dictionary that allows you to define words (names) and their values (definitions). The DumpDictionaryQuoted statement returns the a cr separated array of dictionary values, each line displaying a key=value pair. The value is quoted (for example Name="Jim") so the dump could be used directly in an Execute statement as a series of assignments.
executeapplescript	scripttext	The executeapplescript statement compiles and runs an AppleScript script.
executeapplescript-formula	scripttemplate	This statement executes an AppleScript. Using special tags, the script can access formulas that use Panorama fields and variables. (The formulas are actually calculated in advance, then substituted into the AppleScript code as constants.)

Statement	Parameters	Description
executeasap		Whenever Panorama finishes is not doing anything else (in other words not running any other procedure) it checks for a special variable named ExecuteASAP . If it finds this variable, Panorama takes its contents and executes them, just as if the variable was part of an execute statement. The most common use for this feature is to Handler procedures to perform operations they normally wouldn't be able to do, like open or switch windows.
executelocal	code	The EXECUTELOCAL statement works exactly like the EXECUTE statement, but it shares local variables with the procedure that calls it. Note: You MUST NOT use the RTN statement inside the procedure, or if you do, it must be preceded by the UseMyLocalVariables statement.
executeunix	shellcommand,result	This statement executes a UNIX shell command.
exportvcard	database,vcard	This statement exports the current record into VCard format.
extensiontype	filename,type	This statement determines the 4 letter type code for a file based on the extension. For example .txt files are TEXT.
fedextracking	trackingnumber, shipmentinfo	This statement queries the FedEx web site to determine the status of a shipment.
filecatalog	path,wildcard, typecreator,files	This statement creates a list of files in a folder, including any subfolders of that folder
filemenubar	standardmenus, custommenus	This statement creates a menu bar for every window in this file. (Note: Only data windows get the custom menus. Other windows, for example graphics mode or procedure windows, retain their standard menu configuration.)
filesegmenttodatafork	source,spot,length, destination,typecreator, progress,buffersize	The FILESEGMENTTODATAFORK statement extracts a portion of a file to the data fork of a separate file. If the destination file does not exist it is created. If the destination folder does not exist it is created.
filesegmenttorezfork	source,spot,length, destination,typecreator, progress,buffersize	The FILESEGMENTTODATAFORK statement extracts a portion of a file to the resource fork of a separate file. If the destination file does not exist it is created. If the destination folder does not exist it is created.
findallintext	thetext,searchitem, options,founditems	This statement builds a return separated array of the positions within a text string of all search items found.
fontinformation	font,size,style,ascent, descent,leading, maxwidth	The fontinformation statement allows a program to access information about a font.
formulamerge	template,result	This statement merges data into a template. Within the template you can place formulas inside { and } characters.
generictodatabase	dictionary	This statement converts the data in a GenericFields dictionary into separate fields in the current database.
generictovcard	dictionary,vcard	This statement exports the information from a dictionary into a VCard. Elements in the dictionary are:
getdictionarykey	dictionary,value,key	This statement is part of a suite that manipulates items in a key / value dictionary. This is not a language dictionary for spelling checks, but a dictionary that allows you to define words (names) and their values (definitions). The GetDictionaryKey statement returns the key of an item, given itsvalue.ThisisthereverseoftheGetDictionaryValuestatement.Ofcoursethisonlyworksproperlyifthereisonlyoneinstanceofthevalueinthedictionary.Note:Thisstatementmaybemuchslowerthantheget DictionaryValuestatement.'

Statement	Parameters	Description
getdictionaryvalue	dictionary,name,value	This statement is part of a suite that manipulates items in a key / value dictionary. This is not a language dictionary for spelling checks, but a dictionary that allows you to define words (names) and their values (definitions). The GetDictionaryValue statement returns the value of an item, given itsname.'
getdragdata	item,flavor,data	The getdragdata statement may be used in a procedure that handles data that has been dropped on a form. The getdragdata statement allows you to determine what data was dropped.
getdragflavors	item,flavors	The getdragflavors statement may be used in a procedure that handles data that has been dropped on a form. The Mac operating system allows you to drag multiple flavors of data at a time (for example, you can drag a picture and text (perhaps the name of the picture) at the same time). The getdragflavors statement allows you to determine what flavors were dropped. The most common flavors are "TEXT" and "PICT".
getdragitemcount	count	The getdragitemcount statement may be used in a procedure that handles data that has been dropped on a form. The Mac operating system allows you to drag multiple items at a time (for example, you can select several icons in the Finder and drag them). The getdragitemcount statement allows you to determine how many items were dropped. (Note: Drag-and-drop operations that start within Panorama are limited to one item, so using this statement is unnecessary if you know the data you are receiving came from within Panorama.)
getfilepermissions	folder,filename, permissions	The getfilefinderpermissions statement (OS X only) retrieves a number that contains the permission settings for the file.
getformulatext	template,formula	Many statements require a formula as text, which usually requires using special quote characters like “ ” or { }. This can get confusing. The GETFORMULATEXT statement allows you to convert a formula into text without special quote characters.
getgenericfield	database,field,result	This statement gets the formula that corresponds to a generic field.
getgenericformula	database,field,result	This statement converts a formula using generic fields into a formula using the real database fields.
gettaglocation	page,prefix,suffix, start,length	This statement locates the first matching tag within some text.
gettaglocations	page,prefix,suffix, positions	This statement locates all matching tags within some text.
getwebcolor	rgb,webcolor,default	This statement will calculate an RGB color from a Netscape style web color.
giantmessage	message	This statement will display a giant message alert. Keep in mind that the text in an alert is limited to 255 characters.
grabformula	database,formula,result	This statement evaluates a formula using data from another database.
hardwrap	inputtext,outputtext, linesize	This statement will hard wrap text to what ever line width is set in the linesize parameter.
hexdump	binarydata,hextext	The HEXDUMP statement converts binary data into a hex dump format, with 16 bytes per line.
htmlextractimages	page,links	This statement extracts the image references from an HTML page.
htmlextractlinks	page,links,option	This statement extracts the links from an HTML page.
htmlformitemnames	page,elements	This statement returns a list of form elements (<INPUT tags, etc.).
htmltabletoarray	table,array,rowsep,colsep, colums	This statement converts an HTML table into a text array..

Statement	Parameters	Description
importvcards	vcard	This statement imports the information from one or more VCards into the current database.
increment	data,value	This statement increments a field or variable.
initializedictionary	dictionary,name,value	This statement is part of a suite that manipulates items in a key / value dictionary. This is not a language dictionary for spelling checks, but a dictionary that allows you to define words (names) and their values (definitions). The BuildDictionary statement builds a new dictionary and initializes several values. You could do this by repeatedly calling the SetDictionaryValue statement but this is faster. Note: The name and value parameters may be repeated for as many key / values as you need to initialize. You are responsible for making sure that you do not specify the same key twice, which will result in a corrupted dictionary.
listdictionarynames	dictionary,list	This statement is part of a suite that manipulates items in a key / value dictionary. This is not a language dictionary for spelling checks, but a dictionary that allows you to define words (names) and their values (definitions). The ListDictionaryNames statement returns the names of all of the items in a dictionary in a carriage return separated array.
liveclairvoyance	input,outputlist, separator,listobject, database,query,compare, template,ticks,max, options	The LIVECLAIRVOYANCE statement searches a database to build an array. The array is updated "live" as it is built.
loadlibrary	path,file	This statement loads a library database into memory and registers the procedures in that library so that they will be available as Panorama statements. Only statements that are a single, all uppercase word will become statements.
loadurl	data,url	This statement loads a URL into a field or variable. If the data in the URL contains text, any line breaks in the text are converted to Macintosh format (CR only). Also, if the data contains text, this URL will become the base URL. Subsequent downloads can be made with URLsrelativetothislocation.'
makeabsolutepath	path,absolutepath	This statement converts a relative path (relative to the current database) to an absolute path. It first checks to see if the path is already an absolute path. Note: There is no guarantee that the specified path actually exists!
makeabsoluteurl	relativeurl,absoluteurl	This statement changes the base URL. Once the base URL has been set you can access the relative URLsfromthisbase.It'susuallynotnecessarytousethisstatement,itisusedautomaticallywhenaccessingURL's.'
makefileglobals	assignment	This statement creates one or more fileglobal variables, and assigns a value to each new variable.
makefileinvisible	folder,name	This statement makes a visible file invisible.
makefilesecret	file	This statement closes all of the windows associated with a file, but keeps the file open in memory.
makefilevisible	folder,name	This statement makes an invisible file visible.
makeglobals	assignment	This statement creates one or more global variables, and assigns a value to each new variable.
makelocals	assignment	This statement creates one or more local variables, and assigns a value to each new variable.

Statement	Parameters	Description
makemergeformula	template,database, formula	This statement makes a formula from a template. Within the template you can place fields inside « and » characters and formulas inside { and } characters. Unlike FORMULAMERGE, the formula is not evaluated.
makenewalias	filefolder,filename, aliasfolder,alias,filename	The makenewalias statement creates an alias file.
makenewfolder	path	This statement creates a new folder, and if necessary, also creates any enclosing folders needed to create the target folder. If there is an error, it will appear in the global variable MakeNewFolderError.
makenumberedarray	array,separator,start,end	This statement generates a numeric sequenced array, for example 1, 2, 3, 4, 5.
markallinwpso	searchitem,searchoptions, wpsoname,markoptions, failuremessage	This statement will first change all current text to Plain text and then mark all found search items in Word Processing SuperObject with the style selected.
markdatabasechanged		This statement tells Panorama that the current database is changed (for example if you change a permanent variable.
measuretext	font,size,style,text,widths	The measuretext statement calculates the cumulative width of each character in a word or phrase.
measuretextarray	font,size,style,array, separator,widths	The measuretextarray statement calculates the display width of each element in an array.
moveobjects	objectname, nameoperator, deltav,deltah	This statement will move an object or group of objects up, down, left or right on the form. The objects are selected base upon the nameoperator (=. contains, beginswith, notcontains etc.) and the value in objectname.
noimplicitassignment		The noimplicitassignment statement disables Panoramasimplicitassignmentfeature.'
notealert	resource,message	The notealert statement allows a Panorama procedure to open an alert dialog and display a specified message within that dialog. An "information" icon will appear in the upper left hand corner of the alert, like this:
openanything	folder,file	The OpenAnything command opens any application or document. The item to open is specified by a folder and file name. This has the same effect as double clicking on the application or document in the Finder or Explorer.
openplain		This statement opens a database without opening any pre-saved windows (forms, etc.) Only the data sheet will be opened. If the database has a .Initialize procedure, it will not run. This statement may be useful for opening databases that cannot open normally for some reason. There are no parameters to this statement, instead it opens a dialog to allow the user to select the database to open.
openwebmap	street,city,state, postalcode,country	This statement takes a street address and opens your web browser to a page displaying an map of the corresponding location.
openwizard	wizard	This statement opens a wizard, just as it would be opened by selecting that wizard from the Wizard menu. If the same wizard appears in more than one submenu, the first one will be used.
originalwindow		This statement goes back to the window remembered by the REMEMBERWINDOW statement. The RememberWindow and OriginalWindow statement must be in the same procedure.
plainalert	resource,message	The plainalert statement allows a Panorama procedure to open an alert dialog and display a specified message within that dialog. No icon will appear in the upper left hand corner.

Statement	Parameters	Description
popupatmouse	list,initial,choice	This statement can be used to produce a pop-up menu in response to clicking on a form. The menu will pop-up over the location where the mouse was pressed. This statement should not be called unless the original procedure was triggered by a mouse click. The button should have the click / release option turned off.
popupbutton	list,initial,choice	This statement can be used to produce a pop-up menu in response to clicking on a button. The menu will pop-up over the button that was pressed. This statement should not be called unless the original procedure was triggered by a button. The button should have the click / release option turned off.
popupclick	list,initial,choice	This statement can be used to produce a pop-up menu in response to clicking anywhere on a form. The menu will pop-up over the current mouse location.
popupdoublefield-choices	choice,firstfield,second-field,initial	This statement can be used to produce a pop-up menu containing a list of data values in a field. The statement must be called in response to clicking on a button. The menu will pop-up over the button that was pressed. This statement should not be called unless the original procedure was triggered by a button. The button should have the click / release option turned off.
popupfieldchoices	choice,field,initial	This statement can be used to produce a pop-up menu containing a list of data values in a field. The statement must be called in response to clicking on a button. The menu will pop-up over the button that was pressed. This statement should not be called unless the original procedure was triggered by a button. The button should have the click / release option turned off.
posixpath	folder,file,path	This command converts a file and folder into a posix path.
post	mode,database,keyfield, keyvalue,datafield, datavalue,datafield2, datavalue2	This statement assigns a value to a field in another database.
posturl	data,url,name,value	This statement loads a URL into a field or variable. In addition to the data itself, you can supply one or more post arguments. These allow you to post data that would normally be supplied by an HTML form. If the data in the URL contains text, any line breaks in the text are converted to Macintosh format (CR only). Also, if the data contains text, this URL will become the base URL. Subsequent downloads can be made with URLsrelativetothislocation.Note:Thenameandvalueparametersmayberepeatedforasmanykey / valuesasyouneedtopost.Youareresponsibleformakingsurethatyoudonotspecifythesamekeytwice.'
progressbar	bounds,mercury, temp,boiling,inset	This statement updates a progress bar on the current form. The progress bar consists of two objects, the bounds object (usually displays a background) and the "mercury" object, which grows as process is made.
querywhitepages	first,last,city,state,zip,url, listings,whiteinfo,channel	This statement queries the web to obtain information about a zip code.
registerfunction	folder,name,param-count,body	The registerfunction statement defines a new custom function. This is usually done with a wizard, but a program can use this statement to do so directly. Usually this would be done in the .Initialize procedure so that the function would be available for use throughout the database.

Statement	Parameters	Description
registerlibrary	database,librarydata	The registerlibrary statement registers one or more procedures so that they can be used as custom statements. You should never need to use this command directly. During initialization, Panorama automatically scans the databases in the Extensions:Libraries folder and uses the loadlibrary statement to register all of the procedures it finds there as custom commands.
rememberwindow		This statement remembers the currently active window. You can get back to this window later with the ORIGINALWINDOW statement. The RememberWindow and OriginalWindow statement must be in the same procedure.
rtnerror		The rtnerror statement may be used to help manage errors that are occur during the execution of a subroutine. It is especially useful when designing custom statements. When this statement is encountered, Panorama checks to see if any error handling applies. (Error handling is active if an onerror statement is active, or if the statement after the call or farcall statement that started this subroutine is if error.) If there is no error handling active, the message will be displayed and then the procedure will stop. If error handling is active, the message is placed in Panoramasinternalerrormessagebuffer(itcanberetrievedusingtheinfo("error")function)andtheerrorisprocessedasifitwereaninternallygeneratederror.'
rudemessage	message	This statement will display a very tall message alert rudely. Normally this alert would be used only during debugging (to display multiple lines of data). Keep in mind that the text in an alert is limited to 255 characters.
rundialog	options	This statement uses a form as a template to display a dialog window. The statement supervises the operation of the dialog until it is closed.
saveapplescriptlist	folder,file,items	This statement saves a series of parameters as an AppleScript List. This can be useful for quickly passing a bunch of parameters to an AppleScript.
saveurl	path,url	This statement saves a URL into a file. You can check info("httperror") to make the file was downloaded successfully.<
savewebmap	path,zoom,street,city,state,postalcode,country,mapurl	This statement takes a street address and opens your web browser to a page displaying an map of the corresponding location.
scanlibrary	database,data	This statement scans the procedures in a database and builds a data structure that describes the library procedures that are available. Only procedures that have single word all uppercase names will be included as library procedures.
sendarrayemail	from,to,subject,body	This statement sends a single e-mail to multiple recipients. For other applications see also SendEmail, SendBulkEmail and SendArrayEmail
sendbulkemail	from,database,recipient,subject,body	This statement sends a single e-mail to a single recipient. For more complex applications see also SendEmail, SendOneEmail and SendArrayEmail
sendemail	options,body	This statement sends an e-mail. It gives access to the maximum number of options available. For basic applications see also SendOneEmail, SendBulkEmail and SendArrayEmail
sendoneemail	from,to,subject,body	This statement sends a single e-mail to a single recipient. For more complex applications see also SendEmail, SendBulkEmail and SendArrayEmail

Statement	Parameters	Description
setbaseurl	url	This statement changes the base URL. Once the base URL has been set you can access the relative URLsfromthisbase.It'susuallynotnecessarytousethisstatement,itisusedautomaticallywhenaccessingURL's.'
setdialogtrigger	value	The setdialogtrigger statement changes the value returned by the info("dialogtrigger") function. Usually this value is set by Panorama when a button in a dialog is clicked. The setdialogtrigger statement allows a procedure to simulate this action.
setdictionaryvalue	dictionary,name,value	This statement is part of a suite that manipulates items in a key / value dictionary. This is not a language dictionary for spelling checks, but a dictionary that allows you to define words (names) and their values (definitions). The SetDictionaryValue statement changes the value of an item, given itsname.'
setfilepermissions	folder,filename, permissions	The setfilefinderpermissions statement (OS X only) changes the permission settings for the file.
setfilevariable	database,variable,value	This statement changes the value of a fileglobal variable in another database.
setrectedges	rectangle,top,left, bottom,right	This statement will change one or more edges of the specified rectangle.
shellopendocument	document	The shellopendocument statement opens a document in another application. It can also be used to open web pages or to create e-mail messages.
speak	text	The speak statement speaks a word or phrase using the current voice. If the voice is already talking, it waits until the current text has been spoken before speaking the new text. The program continues running immediately - in other words, the next statement after the speak statement will be executed immediately, without waiting for the text to be spoken. See the info("speaking") function if you need to wait until the text has been spoken to procede. (Note: This statement is only available on MacOS computers.)
speakaddress	data	This statement buffers up an address to be spoken later by the SpeakNow statement. Various post office abbreviations are expanded, and numbers are spoken as digit pairs.
speakcharacters	data	This statement buffers up text to be spoken later by the SpeakNow statement. Instead of speaking words, each individual character is spoken.
speakcharactersslowly	data	This statement buffers up text to be spoken later by the SpeakNow statement. Instead of speaking words, each individual character is spoken separately and slowly.
speakdate	data	This statement buffers up a date to be spoken later by the SpeakNow statement. The date will be spoken in the format Month daynth, Year.
speakdigitpairs	data	This statement buffers up text to be spoken later by the SpeakNow statement. Integer numbers within the text will be spoken as pairs of digits, for example 2145 is "twenty-one fourty-five". However, if a number has more than four digits it will be spoken as individual digits. This statement is good for speaking addresses.
speakdigits	data	This statement buffers up text to be spoken later by the SpeakNow statement. Numbers will be spoken as individual digits.
speakdollars	data	This statement buffers up a number to be spoken later by the SpeakNow statement as dollars and cents.
speakdollarsandcents	data	This statement buffers up a number to be spoken later by the SpeakNow statement as dollars and cents.

Statement	Parameters	Description
speakletters	data	This statement buffers up text to be spoken later by the SpeakNow statement. Letters will be spoken individually, as upper or lower case. Numbers will be spoken as individual digits.
speaknow		This statement causes any buffered speech to be spoken.
speakphonenumber	data	This statement buffers up an address to be spoken later by the SpeakNow statement. Various post office abbreviations are expanded, and numbers are spoken as digit pairs.
speakscript	database,script	This statement speaks a script created by the Speech Wizard.
speakstate	data	This statement buffers up the name of a US state to be spoken later by the SpeakNow statement.
speakthisrecord		This statement speaks the current record in the current database. To do this, it uses the default script for this database (created by the Speech Wizard).
speakwords	data	This statement buffers up one or more words to be spoken later by the SpeakNow statement.
speechscripts	database,scripts	This statement returns a lists of speech scripts in an open database. These scripts are created by the Speech Wizard.
splitmenutrigger	menuname,menuitem	This statement splits info("trigger") into separate menu name and menu item variables.
standardclosealert		This statement is designed to be used in the ..CloseDatabase procedure. It displays the standard "Do you want to save changes " alert, and processes the result. Use this statement if you want to do additional processing before or after .
startdragdrop		The startdragdrop statement is used in combination with the dragdrop and dragdropdata statements to start a multi-flavor drag operation. The dragdropdata statement is repeated for each flavor to be included in the drag operation. On MacOS data can be dragged to other applications as well as within Panorama, on Windows dragging is only allowed within Panorama.
stopalert	resource,message	The alert statement allows a Panorama procedure to open an alert dialog and display a specified message within that dialog. Panorama will display an exlamation point icon in the upper left hand corner of the alert.
stopspeaking		The stopspeaking statement tells Panorama to shut up now! Any text that is being spoken will stop immediately in mid-sentence.
stringreverse	result	This statement reverses the characters in a string of text.
stuff	archive,sources,options	This command compresses one or more files using Stuffit. It requires that you have Stuffit Deluxe 8.0 or later installed.
superalert	message,options	This statement will display an alert. You can control the size of the alert, as well as several appearance options. The text displayed in the alert may be any length (it is NOT limited to 255 characters). If the alert has more than one button, info("dialogtrigger") will contain the name of the button that was pressed.
superarraybuild	array,sepchar,database ,conditional,generate, options,maxticks, maxrecords,skiprecords, cachelevel	The superarraybuild statement scans a database to create an array. This statement is similar to arraybuild , but with a number of additional options. This statement was primarily designed for the Live Clairvoyance™ feature, but weresurethattherewillbeotherusefulapplications.IfyoudowanttousethisforLiveClairvoyancewerecommendthatinsteadofusingtheSuperArrayBuildstatementdirectlyyouusetheliveclairvoyancecustomstatement.'

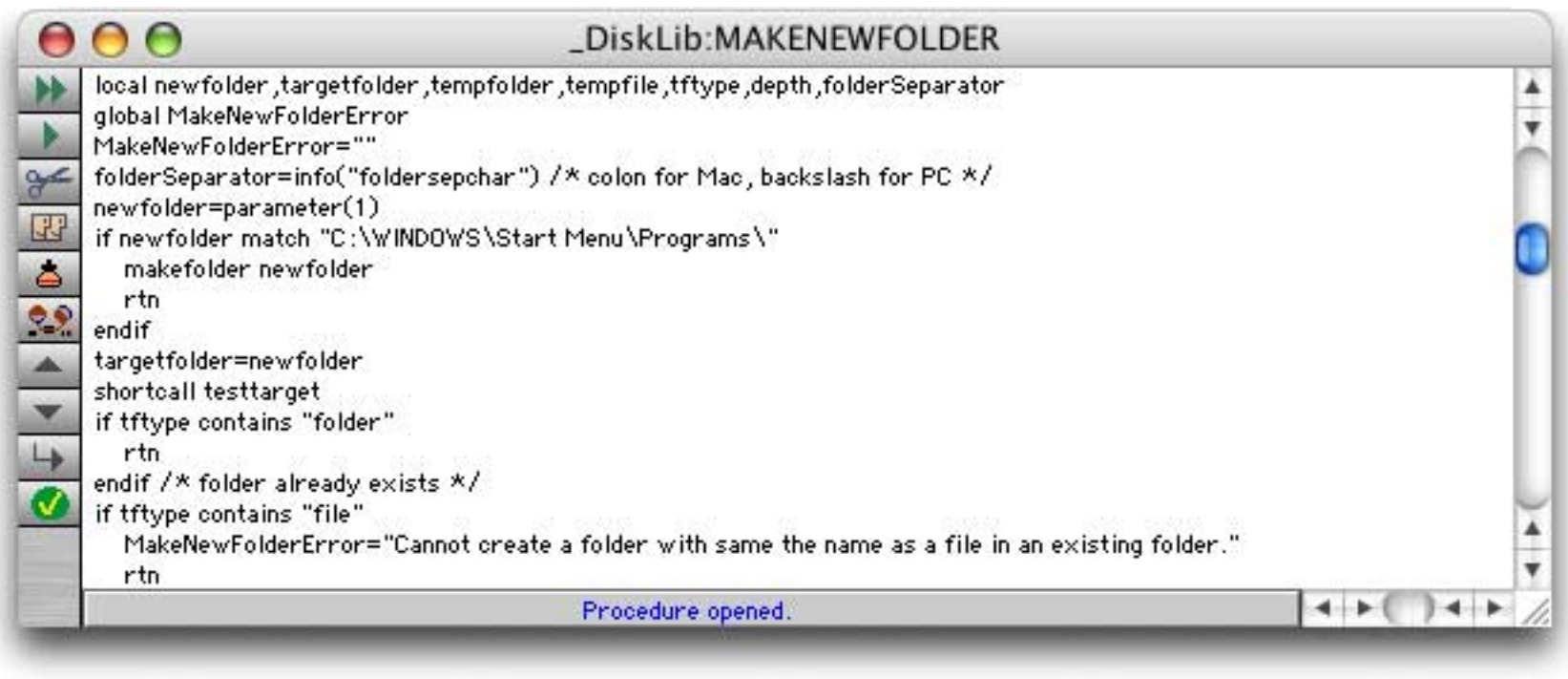
Statement	Parameters	Description
superchoicedialog	choicelist,choice,options	This statement will display a dialog with a list of choices that can be selected. You can control the size of the alert, as well as several appearance options. If the alert has more than one button, info("dialogtrigger") will contain the name of the button that was pressed.
tallmessage	message	This statement will display a very tall message alert. Normally this alert would be used only during debugging (to display multiple lines of data). Keep in mind that the text in an alert is limited to 255 characters.
textfilter	oldtext,newtext,formula	The textfilter statement is similar to the characterfilter statement, but the textfilter statement doesntnecessarilyscanthetextcharacterbycharacter.Instead,itallowsyoutoskipoverasequenceofcharactersbasedonaformulabeforeitstartsscanningcharacterbycharacteragain.'
textwidth	font,size,style,text,width	The textwidth statement calculates the width of a word or phrase in a specified font, size and style.
togglesummarylevel		The togglesummarylevel statement changes the current record from a data record into a summary record, or from a summary record into a data record.
updateballoonlocations		The updateballoonlocations statement may be used after you move objects within a form. If any balloon help objects were moved, this statement will update Panoramasballoonhelpfeaturetoreflectthenewlocations.'
usecallerslocalvariables		The usecallerslocalvariables statement allows a procedure to access the local variables of the procedure that called the current procedure.
usemylocalvariables		The usemylocalvariables statement reverses the action of the usecallerslocalvariables statement, which allows a procedure to access the local variables of the procedure that called the current procedure.
usnumbers		The usnumbers statement allows a programmer to switch to the US format for numbers and dates within a procedure, regardless of the international settings that may be defined in the operating system preferences.
vcardtogeneric	vcard,dictionary	This statement imports the information from a single VCard into a dictionary. Elements in the dictionary are:
windowcornerinitialize	edges	This statement shifts a window to an edge or corner of the main screen. However, it only does so if the screen size has changed. This is used by wizards to make sure that the window is positioned correctly the first time it is opened. But if the user decides to reposition the window, they can without this routine interfering.
windowmenubar	standardmenus, custommenus	This statement creates a menu bar for this window.
windowtocorner	edges	This statement shifts a window to an edge or corner of the main screen.
writedialogcode	code	This statement scans the current form, and writes the procedure needed to handle this form as a dialog.
zipinfo	zipcode,zipcodeinfo, channel	This statement queries the web to obtain information about a zip code.
zipinfoplus	address1,address2,city, state,zip5,zipplusinfo	This statement queries the web to obtain information about a zip code.

Custom Statements

Panorama comes with hundreds of ready to use built-in statements, but you aren't limited to these built-in statements. If you don't find the statement you need you can build your own! If you are planning on using a particular sequence of steps frequently then it might pay to create a custom statement that you can use in any database.

As you might have guessed from the phrase "sequence of steps" in the previous paragraph, custom statements are very similar to subroutines. In fact, custom statements actually are Panorama subroutines, written using Panorama's standard procedure editor and the Panorama programming language. The only difference is that these subroutines are placed in a special location, where Panorama automatically loads them so. As it loads the database containing these special subroutines, Panorama also adds them to the programming language so that they are always available to procedures in any database.

To illustrate this, consider the subroutine shown below, which will make a new folder. This procedure is called MAKENEWFOLDER and is contained in the _DiskLib database.



Even without the ability to create custom statements you can still call this subroutine from another database using the line shown below. (see "Calling a Subroutine in Another Database" on page 269 of *Formulas & Programming*).

```
farcall "_DiskLib","MAKENEWFOLDER","My Drive:My Documents:MyImages:Zack:"
```

This technique has some drawbacks — you have to remember that this subroutine is in the _DiskLib database, you have to make sure that the _DiskLib database has actually been opened, and you have to use the exact capitalization MAKENEWFOLDER because MakeNewFolder or makenewfolder won't work. Converting this procedure into a custom statement fixes these drawbacks. Here is the same subroutine used as a custom statement:

```
MakeNewFolder "My Drive:My Documents:MyImages:Zack:"
```

To learn how to create your own custom statements, see "Custom Statements" on page 289 of *Formulas & Programming*.

Programming Techniques

The chapter "Programming Techniques" on page 399 of *Formulas & Programming* includes many new techniques introduced in Panorama V.

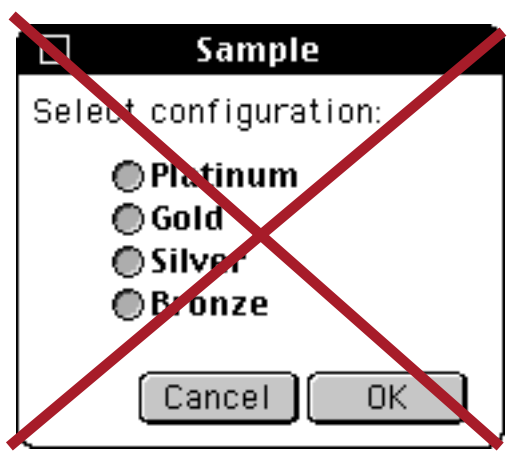
Files

You can use the **openanything** statement to open any document on your hard drive in its native application. See "Opening a Document in Another Application" on page 416 of *Formulas & Programming*.

The MacOS X operating system is based on UNIX, and uses UNIX style file permissions. File permissions allow the system administrator to control who can read and write any file or folder. Panorama can access and modify these permissions with the **getfilepermissions** and **setfilepermissions** statements. To learn more about these statements see the **Programming Reference** wizard.

Windows

In previous versions of Panorama window type 16 was a rounded corner window with a black drag bar. This window type is no longer available.



See "Non Standard Window Styles" on page 449 of *Formulas & Programming*.

Alerts

There are three new variations on the **Message** statement — **BigMessage**, **GiantMessage** and **TallMessage**. See "Displaying a Message" on page 464 of *Formulas & Programming*.

There are a number of new statements for displaying standard alerts — **alertyesno**, **alertnoyes**, **alertyesnocancel**, **alertokcancel**, **alertcancelok**, **alertdeletecancel**, **alertcanceldelete**, **alertsavecancel**, **alertrevertcancel**, **alertok** and **alertoksmall**. See "Alerts With Multiple Buttons" on page 466 of *Formulas & Programming*.

The new **superalert** statement displays a configurable alert. You can control over a dozen different options, including the overall alert size (height and width), the font, text size, title, color, background color, and more. You can even place images on the alert and specify up to three buttons on the alert. In addition, the alert may display up to 32,768 characters (it is *not* limited to 255 characters like the other alerts described in this section) and can even have a scroll bar. See "The SuperAlert Statement" on page 472 of *Formulas & Programming*.

The **displaydata** statement is based on the **superalert** statement, and provides an easy way to display large amounts of information for debugging. See "The DisplayData Alert" on page 479 of *Formulas & Programming*.

Dialogs

The **supergettext** statement displays a configurable dialog for entering text. You can control over a dozen different options, including the overall dialog size (height and width), the font, text size, title, color, background color, and more. See "The SuperGetText Statement" on page 482 of *Formulas & Programming*.

The **supergetchoice** statement displays a configurable dialog for choosing an item from a list. You can control over a dozen different options, similar to the **supergettext** statement. See "The SuperChoiceDialog Statement" on page 485 of *Formulas & Programming*.

The custom dialog system introduced in Panorama 4 has been enhanced and is now built into Panorama, along with a new Custom Dialog wizard. Using this system you can turn any form into a dialog.See “Custom Dialogs” on page 489 of the *Panorama Handbook*.

Moving Data Between Files

The `post` statement assigns values to one or more fields in another database. It’s kind of like a reverse lookup. See “The Post Statement” on page 548 of the *Panorama Handbook*. The `postadjust` statement adjusts the value of a numeric field in another database, by adding (or subtracting) a number. For example, you could use this to subtract from the quantity on hand field of an inventory database as an invoice is processed. See “The PostAdjust Statement” on page 549 of the *Panorama Handbook*.

Finding Information

Panorama V introduced a new search method — **Live Clairvoyance™**. Live Clairvoyance allows you to perform "live" searches on any Panorama database. The search results are updated dynamically as you type, allowing you to "hone in" on just the information you are looking for. The search may include multiple fields or even all fields in the database being searched. (If you've used the search box in iTunes you'll find Live Clairvoyance familiar, although the underlying technology is not related.)

The easiest way to use Live Clairvoyance is with the wizard that comes with Panorama. This wizard performs the neat trick of allowing you to use Live Clairvoyance without doing any programming, and in fact without making any modification at all to your databases. However, for some applications you may want to build Live Clairvoyance into your forms. This allows you to customize it exactly for your needs, and to integrate it with other database operations. To learn how to do this see “Live Clairvoyance™” on page 559 of *Formulas & Programming*.

Processing and Transforming Binary Data

Panorama V includes several new functions for handling bits (see “Bits” on page 598 of *Formulas & Programming*).

Function	Description
bit(number)	This function converts a bit number (1 to 32) into a number (1, 2, 4, 8, 16, etc.)
getbit(number,bitnumber)	This function returns a true or false value by testing a bit. The bit number may be from 1 to 32. If the bit is set, the value will be true, if it is not set, the value will be false.
onescomplement(number)	This function returns the one's complement of a 32 bit number (all bits are reversed)
setbit(number,bitnumber,truefalse)	This function sets one bit within a number, without disturbing any of the other bits. The bit number may be from 1 to 32. The bit will be set based on the true / false parameter - set if true, cleared if false.

One DImensional Arrays of Binary Values

Programmers familiar with languages like C and Pascal will be familiar with the concept of an array as a sequence of binary values, for example a sequence of bytes, words, longwords, or some other length of binary data. This is quite different from Panorama's usual concept of arrays as variable length items separated by a special character. Panorama V has several functions for extracting a specific element from an array of binary values. The function you pick depends on the type of binary value in the array. See "One Dimensional Arrays of Binary Values" on page 600.

Function	Reference Page	Description
bytearray(data,index)		This function extracts a value from an array of bytes. This is not a Panorama style delimited array but a C style array of 8 bit values. The result is an integer.
chararray(data,index)		This function extracts a characters from an array of characters. This is not a Panorama style delimited array but a C style array of ASCII characters. The result is an single characters.
chunkarray(data,index, chunklength)		This function extracts a binary chunk from an array of chunks. This is not a Panorama style delimited array but a C style array of binary chunks. The result is a binary value (text).
longwordarray(data,index)		This function extracts a value from an array of longwords. This is not a Panorama style delimited array but a C style array of 32 bit values. The result is an integer.
wordarray(data,index)		This function extracts a value from an array of words. This is not a Panorama style delimited array but a C style array of 16 bit values. The result is an integer.

Panorama also has three new statements designed for rapidly filtering and / or processing arrays of binary or textual data. See "The CharacterFilter Statement" on page 600, "The ChunkFilter Statement" on page 601 and "The TextFilter Statement" on page 601 of *Formulas & Programming*.

Data Dictionaries

A conventional dictionary (Webster's, for example) contains a list of entries. Each entry comes as a pair — the word itself, and the definition of the word.

Panorama V supports a data structure called a **data dictionary**. Like a conventional dictionary, a data dictionary contains a list of entries, and each entry comes as a pair — the entry **key** (which identifies the entry) and the entry **value** (some data associated with this entry). Sometimes these entries are referred to as **key/value pairs**. If you know the key, you can find out what the value is. A data dictionary allows you to combine any number of these key / value pairs into a single structure that can be stored in a single variable, a single field or a single procedure parameter. For example, in a database you could use a data dictionary to store additional, seldom used data that you don't want to devote an entire field to, or to store information that you didn't anticipate when you created the database. When a procedure (or custom statement) has a large and variable number of parameters, you can combine these into a data dictionary that can be passed back and forth easily (many of Panorama's Internet access statements work this way). Data dictionaries are also an excellent way to store preferences, and they are used that way by many of Panorama's wizards (for example the **Hot Key** wizard and any application that works with Generic fields (see "Generic" Fields" on page 230 of the *Panorama Handbook*).)

To learn how to create, access and update data dictionaries see "Data Dictionaries" on page 602 of *Formulas & Programming*.

General Internet Access

In this millennium no computer is an island. Panorama allows you to tap into the Internet to automatically retrieve data and graphics from the web, submit information to web servers, and to send e-mail based on database information (see "Basic Web Access" on page 607 of *Formulas & Programming*). The **loadurl** statement fetches web pages from any server (see "Fetching Web Pages" on page 607 of *Formulas & Programming*). There are a number of new statements and function for parsing web pages (see "Parsing Web Pages" on page 608 of *Formulas & Programming*). The **saveurl** statement fetches images from the web (.jpg, .gif, etc., see "Fetching Images" on page 613 of *Formulas & Programming*). The **posturl** statement allows a Panorama program to submit a form to a web server and fetch the response (see "Submitting Forms" on page 614 of *Formulas & Programming*). Panorama can also access web sites that require cookies (see "Cookies" on page 616 of *Formulas & Programming*).

Accessing Web Content

The **savewebmap** statement makes it easy to download a map for any US address (see "Generating Map Images" on page 617 of *Formulas & Programming*). The **zipinfo** statement queries the web to get general information about a zip code: city, state, area code, etc. (see "General Zip Code Information" on page 622 of *Formulas & Programming*). Given a US street address, the **zipinfoplus** statement uses the US Post Office web site to find out the zip+4, carrier route, and other information (see "Street Address Information" on page 623 of *Formulas & Programming*). The **querywhitepages** statement queries the web to look up white page information (see "White Pages" on page 624 of *Formulas & Programming*) The **fedextracking** statement queries the FedEx web site to determine the status of a shipment (see "FedEx Shipment Tracking" on page 625 of *Formulas & Programming*).

Controlling Web and E-Mail Clients

In addition to directly accessing web pages (see above) and sending e-mail (see below) Panorama can control your default web and e-mail client to view pages and set up outgoing e-mail. To display a web page use the **shellopendocument** statement (see "Displaying a Web Page" on page 626 of *Formulas & Programming*). To display a web page that is on your local hard drive (not on the web) use the **openanything** statement (see "Looking Up a Dictionary Key Given Its Value" on page 606 of *Formulas & Programming*). Use the **openwebmap** statement to display the map for any US address (see "Generating Map Images" on page 617 of *Formulas & Programming*).

In addition to displaying web pages, the **shellopendocument** statement can also be used to initiate the process of creating a new e-mail. See "Generating Map Images" on page 617 of *Formulas & Programming*.

Sending E-Mail

Panorama cannot send e-mail by itself. However, through the use of a channel Panorama can interface with external software to send e-mails automatically. Panorama comes with several ready to use e-mail channels, and it is also possible to write your own channels. See "Channel Configuration" on page 630 of *Formulas & Programming* to learn how to prepare an e-mail channel for your system. Once the channel is configured you can use the **sendoneemail** statement to send a single e-mail to a single recipient (see "Sending a single e-mail" on page 631 of *Formulas & Programming*). The **sendbulkemail** and **sendarrayemail** statements send multiple copies of the same e-mail to different recipients (see "Sending multiple e-mails" on page 631 of *Formulas & Programming*). Use **sendbulkemail** if the recipient e-mail addresses are in a database field. Use **sendarrayemail** if the recipient e-mail addresses are in a text array.

Drag and Drop

Panorama supports the ability to drag data from one location to another (usually called drag-and-drop). On MacOS computers you can drag data within Panorama, and also drag data from other applications to Panorama or drag data from Panorama to other applications. On Windows systems you can only drag within Panorama, drag and drop between Panorama and other applications is not supported. To learn how to add drag and/or drop capabilities to your databases see "Drag and Drop" on page 650 of *Formulas & Programming*.

VCard Drag and Drop. Once generic fields have been set up for a database (see ""Generic" Fields" on page 230 of the *Panorama Handbook*) it is very easy to add drag and drop support so that you can exchange data with Apple's Address Book (or other databases or VCard enabled applications) without using a wizard. See "VCard Drag and Drop" on page 656 of *Formulas & Programming* for more information.

Speech Synthesis

Speech synthesis is an exciting new feature introduced in Panorama V. (Note: Speech synthesis is available only on Macintosh systems. It is not available on Windows systems.) There are several new statements and functions that support speech synthesis — see "Speech Synthesis" on page 719 of *Formulas & Programming* of the **Panorama Handbook**.

Writing Your Own Channel Modules

Panorama comes with a number of channel modules for sending e-mail, dialing the phone, and interfacing with other web sites and third party software. If you have programming experience you can write your own channel modules. To help make this easier we have created a **Channel Workshop** wizard that will create the core of your new module for you. See "Writing Your Own Channel Modules" on page 739 of *Formulas & Programming* for more information.

Improved AppleScript Support

The **executeapplescript** statement allows you to build and execute an AppleScript on-the-fly within a Panorama procedure. This means that you don't have to keep track of a bunch of separate script files and you can easily change the script to match changing conditions. This statement also makes it possible for Panorama to easily access the result of a script. See the **Programming Reference** wizard for more information on this statement. (Note: This statement has now been superseded by the **applescript** statement, see "AppleScript" on page 745 of *Formulas & Programming*.)

Using the UNIX Command Line

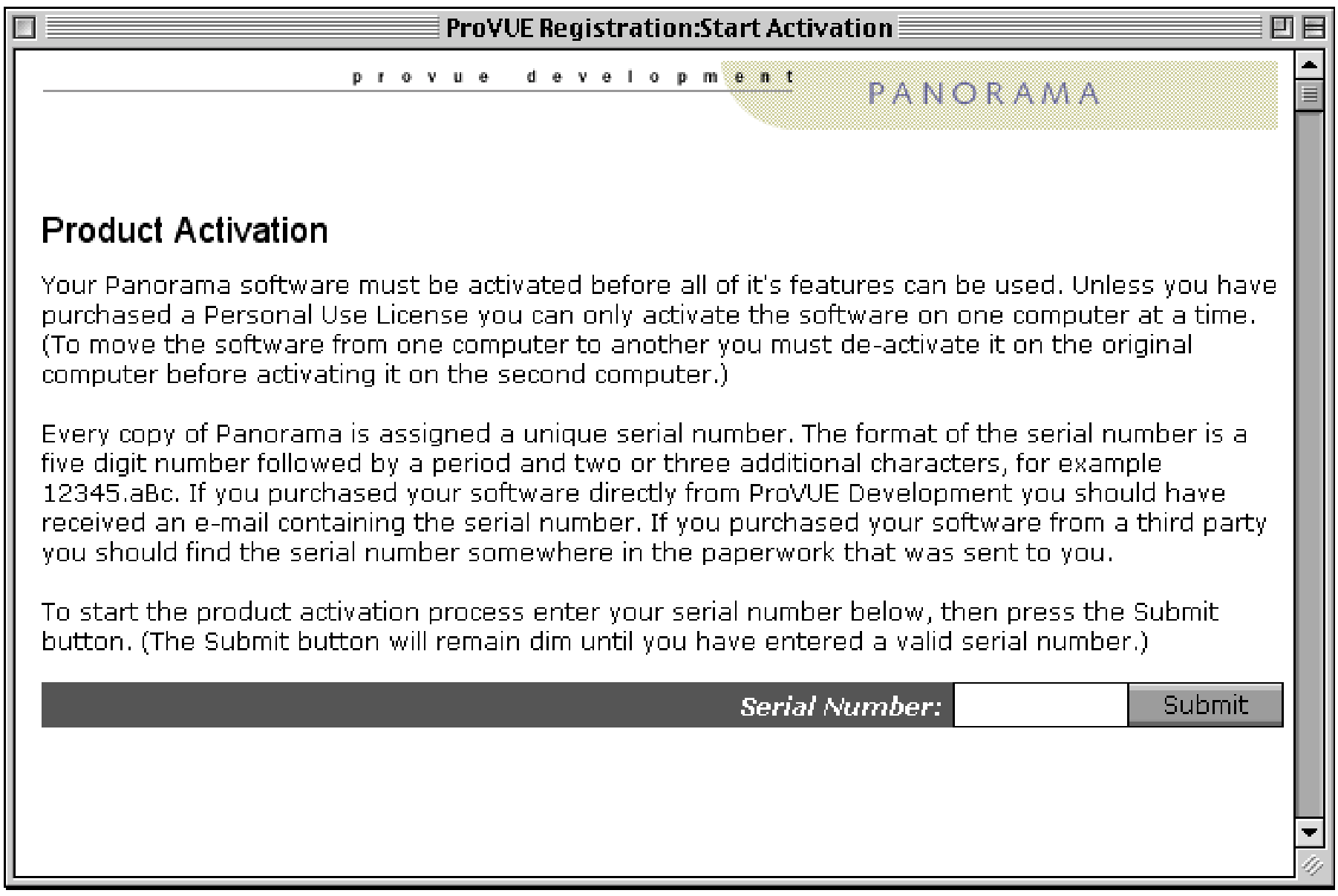
The **executeunix** statement executes a UNIX shell command. The **executeunix** statement opens the entire universe of the power of UNIX to Panorama programmers. See the **Programming Reference** wizard for more information on this statement. (Note: This statement has now been superseded by the **shellscript** statement, see "AppleScript" on page 745 of *Formulas & Programming*.)

Version 4.0.2

This version was released in September 2002. It was primarily a bug fix release, with only a few new features.

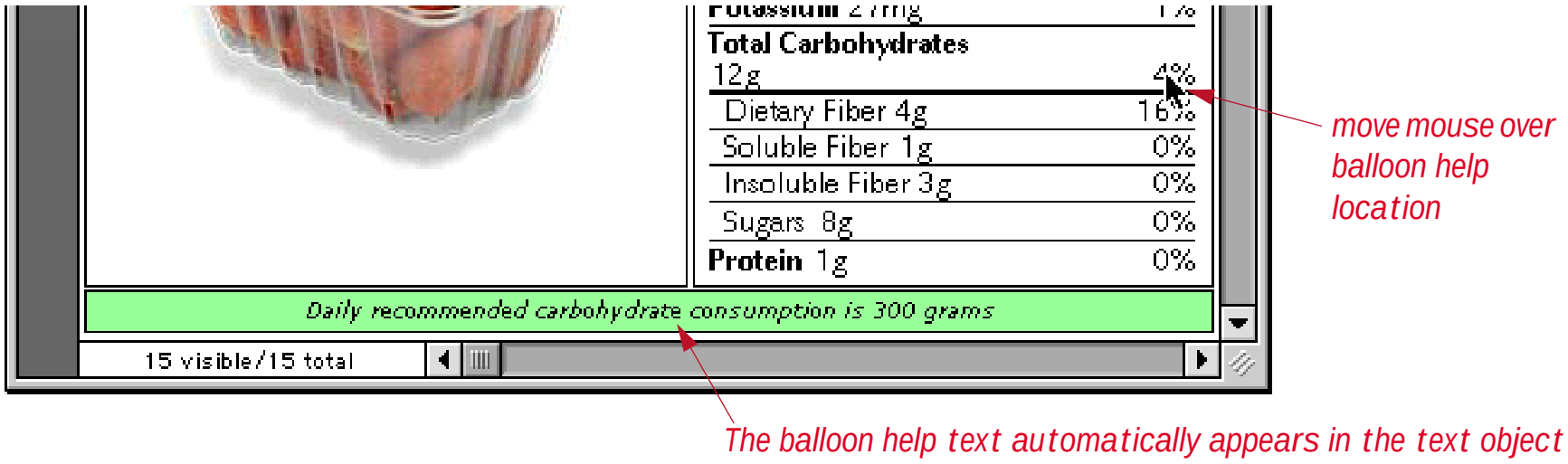
New Activation Dialog

Panorama 4.0.2 includes a new version of the **ProVUE Registration.pan** database. This new version almost completely automates the process of activating Panorama. If your computer has an internet connection you no longer need to type product codes, activation codes, or personal license information. All you need to type is the serial number and Panorama will use your web browser to get the additional information from ProVUE's web server.



Displaying Balloon Help Text Directly on the Form

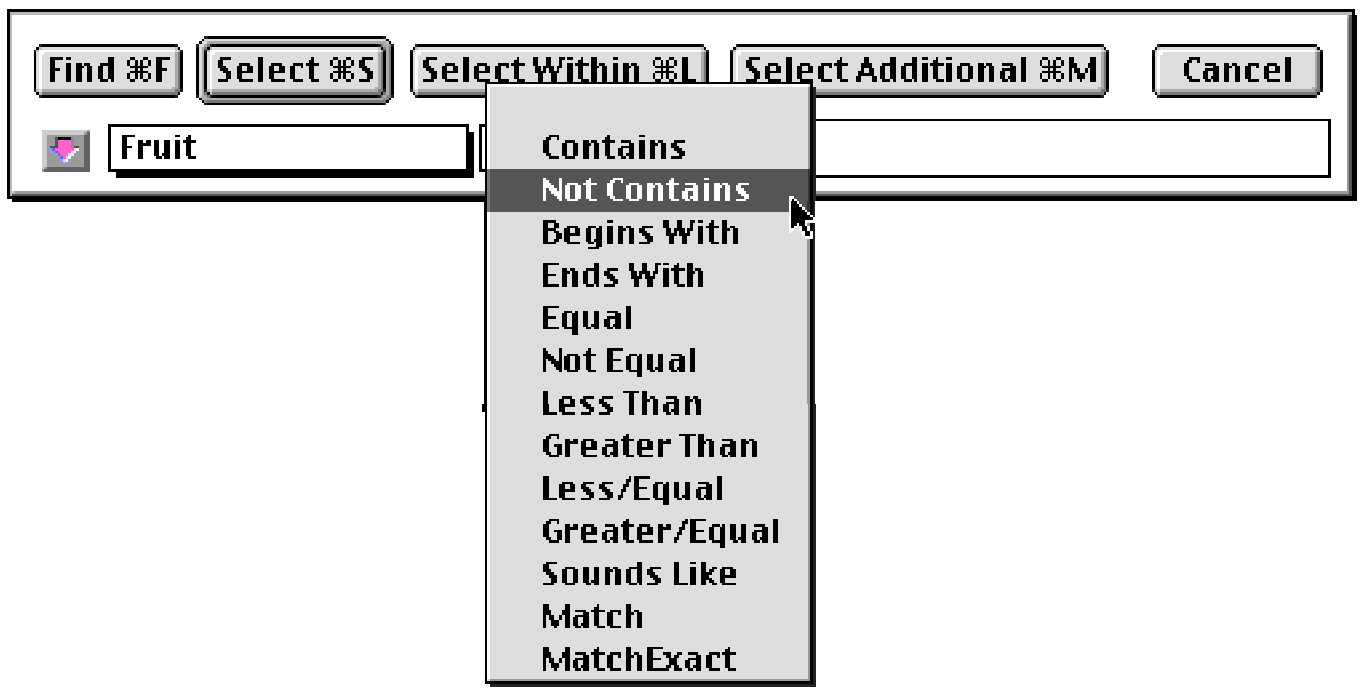
Balloon Help text can now be displayed on the form itself as well as using the Macintosh **Show Balloons** option (see "Displaying Balloon Help Text Directly on the Form" on page 992 of the *Panorama Handbook*).



When used this way the help text is visible even if the **Show Balloons** option is not turned on. This option also allows the balloon help text to be displayed on PC systems (kind of like tool tips).

New Search Option

The **Find/Select** dialog includes a new option, **Not Contains**.



This option works exactly the opposite from the **Contains** option. "The Find/Select Dialog" on page 336 of the *Panorama Handbook* for more information on this dialog.

Separate Close File & Close Window Commands

Panorama 4.0.2 now includes two separate close commands in the **File** menu.

The **Close File** command closes the entire database. The new **Close Window** command closes just the current window (just like clicking on the window's close box).

Assorted User Interface Fixes and Improvements

Panorama 4.0.2 includes a number of small changes for improving the general operation of the program.

Fixed a bug that sometimes caused clairvoyance to fire off prematurely.
Files sets ending in .pnz now work properly on the Macintosh as well as Windows. The databases in the set must have filenames that end with .pan. This improvement allows Mac and Windows computers to use the same file set on a shared server.
Fixed pop-up menu problem reported by some users.
When you double click on a data cell that contains more than 32k of text an alert appears (earlier versions of Panorama would often crash in this situation).
Fixed the Windows PC version so that windows can open partially off screen. In previous versions of Panorama 4 if a window was partially off screen it would be pushed completely off the screen, leaving you wondering “where is my window?”
Fixed the forward delete key in Windows, now it only deletes the next character.
Fixed problem with pasting text from another application into the word processor.
Fixed the Mini Correspondence wizard so it doesn't get hunter errors sometimes when opening.
Previous versions of the New Database Wizard wouldn't import a text file properly if the first line was longer 1000 characters. In Panorama 4.0.2 this limit has been increased to 10000 characters.
The Text Export Wizard now works properly even if the array of fields is scrolled.
Fixed append of Panorama files with .pan extension.
Attempting to create a Flash Sound object no longer crashes the program.
Removed font error message from the Panorama Handbook Wizard.
Previous versions of Panorama 4 had a problem if you copied graphic objects (or an entire form) onto the clipboard and then attempted to paste into text (data cell, procedure, etc.). The possibility of accidentally erasing data has been removed.
In the Design Sheet equation column whitespace (blanks) is now allowed after a procedure name. The whitespace will be ignored.

Improved Formula Calculations

Panorama 4.0.2 corrects a number of problems with numeric calculations and functions.

The min(function now works properly with negative numbers.
Fixed a rounding problem when dividing two integers. This problem also affected the **Average** command.
Panorama now correctly checks for divide by zero in floating point calculations, and produces an error if encountered.
Adjusted several scientific functions to give floating point error if an invalid input value is supplied, including log(, log10(, sqr(, arccos(, arccosh(, arcsin(and arctanh(.
When using a numeric field in an auto wrap text object it would not display 0's after the decimal point. For example 100.00 would be 100 while 100.10 would be 100.10. This was incompatible with Panorama 3.1, now it works the same.
Fixed the urldecode(function so that it works with hex values with letters (%4A, etc.) and so that it works properly with most 8 bit ASCII values.
Fixed the urlencode(function so that it works with 8 bit ASCII values (accents, special characters etc.) These characters are encoded as %nn. If a character cannot be expressed in a URL it is converted to %00.
On Windows systems, the folder(function now works correctly when the specified folder is C:\.
Fixed the exportcell(, sizeof(, fieldstyle and fieldmax(functions so that they can be used in the arraybuild statement.

Improved Procedures and Programming

Panorama 4.0.2 corrects a number of problems with program statements and tools.

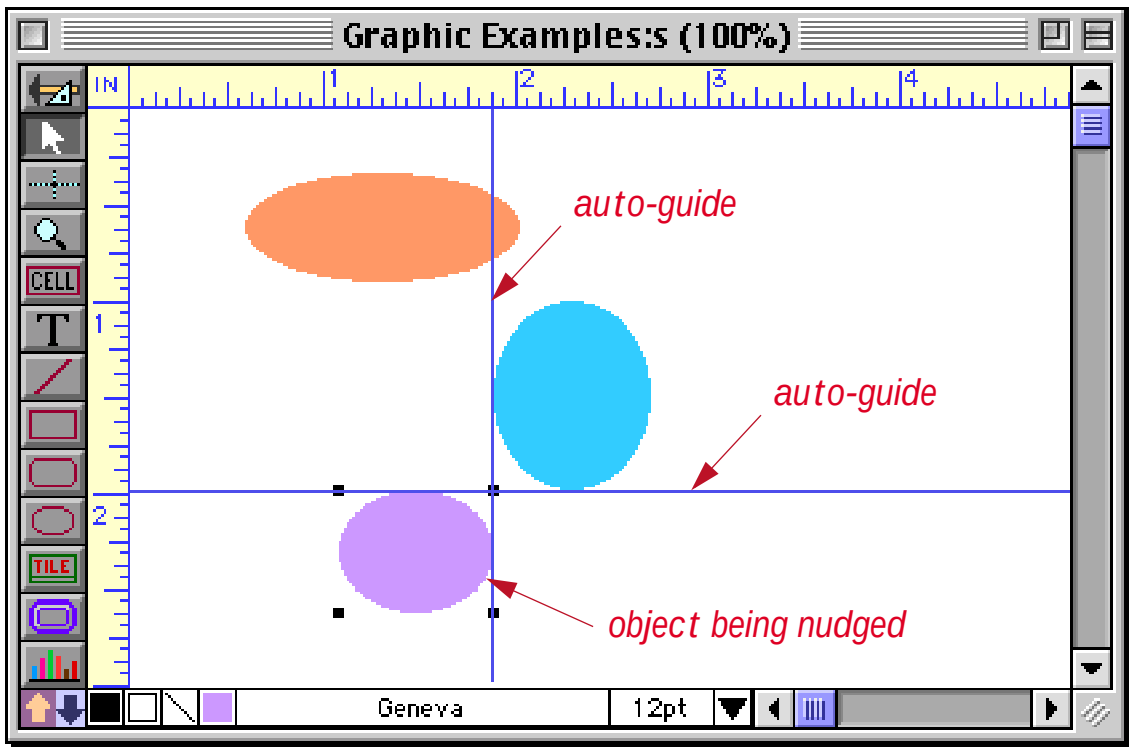
The AppleScript EXECUTE statement worked in Panorama 4.0 but was broken in Panorama 4.0.1. It now works again.
Subroutines may now be nested up to 32 levels (previous maximum was 8). If your program exceeds 32 levels of recursion it will stop with an error message without crashing.
The Command and Function help menus have been re-arranged into a more logical order.
Fixed the ZoomWindow and SetWindow statements so that they work properly when a rectangle function is used as one of the parameters, for example Zoomwindow 10,10,rheight(info("ScreenRectangle")),450,""
Panorama 4 did not switch windows properly when running a procedure in the background. Now it does, allowing it to work properly in server applications.
If a Panorama database is open the OpenFile statement now makes sure that a data window is open before continuing with the procedure. If a procedure or form window is the top window in the database being opened, this statement will automatically switch to the topmost open form or data sheet in that database. This greatly reduces accidental occurences of the dreaded “You can’t do that in this window” error message.
Fixed the SaveACopyAs statement so that it can write over an existing Panorama file (this bug was in the Windows PC version only).
The superobject "TextObject","Find" statment now works as documented in Text Editor SuperObjects (previously it worked in Word Processor SuperObjects but not Text Editor SuperObjects).
Fixed the ChangeObjects statement. Previouslly the EXPANDABLE and EXPANDSHRINK options did not work with a zero parameter. In other words, you could not make an object have a fixed size under program control.
Fixed several problems associated with the ONERROR statement. These fixes will make Panorama 4.0 more reliable in server applications.

Version 4.0.1

This version was released in September 2001, and includes an unusual number of new features for a x.0.1 release. In addition to the new features listed below version 4.0.1 also includes a number of bug fixes, see our website at **www.provue.com** if you are interested in the exact list.

Automatic Guides when Nudging Graphic Objects

As you nudge the location or size of a graphic object (or objects) Panorama 4.0.1 now checks to see if the edges of the nudged object align with any other objects on the form. If so, guides appear to show the alignment.



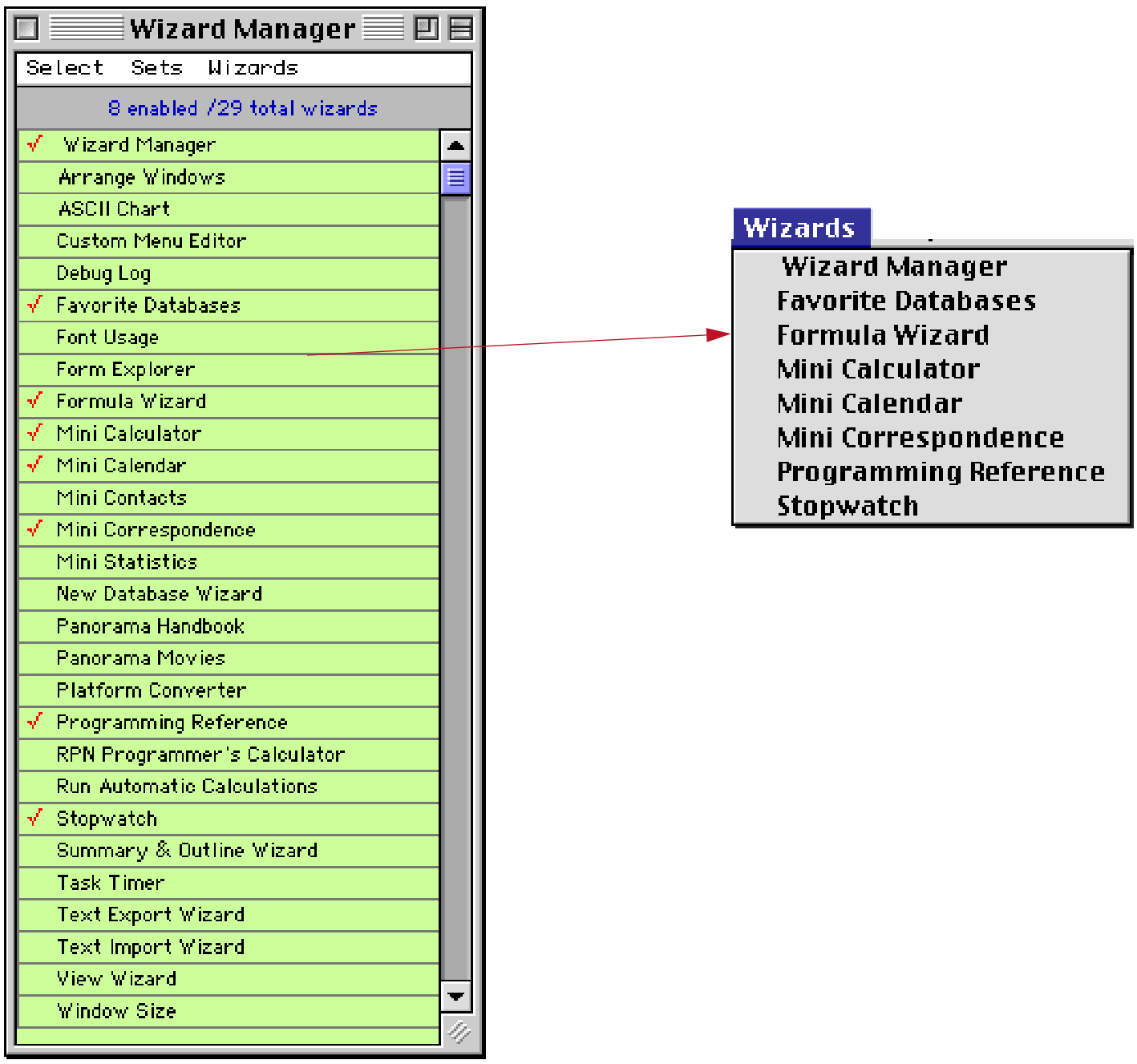
This is a great feature for quickly setting up perfectly aligned forms. See “Nudge “Auto Guides”” on page 510 of the *Panorama Handbook* for all the details.

Improved Enhanced Image Pack

The **Enhanced Image Pack** has been rewritten to use Apple Quicktime to display and convert images (previously we used a library licensed from another third party vendor). Because of this, support for some infrequently used image types has been dropped, and support for GIF, LZW compressed TIFF and Photoshop files has been added. In addition, the Enhanced Image Pack is now more compatible with JPEG images from any source and can now print sharp reduced images (for example thumbnails) on Macintosh systems. For more information see See “Displaying Non PICT Images (Enhanced Image Pack)” on page 775 and “Converting Between Image Formats” on page 694 of the *Panorama Handbook*.

New Wizard Manager

Wizards here, wizards there, I see wizards everywhere! Seriously, the number of wizards is growing, and you can also create your own wizards. To help you keep all these wizards organized and manageable we've added a new **Wizard Manager** that gives you complete control over the **Wizard** menu.

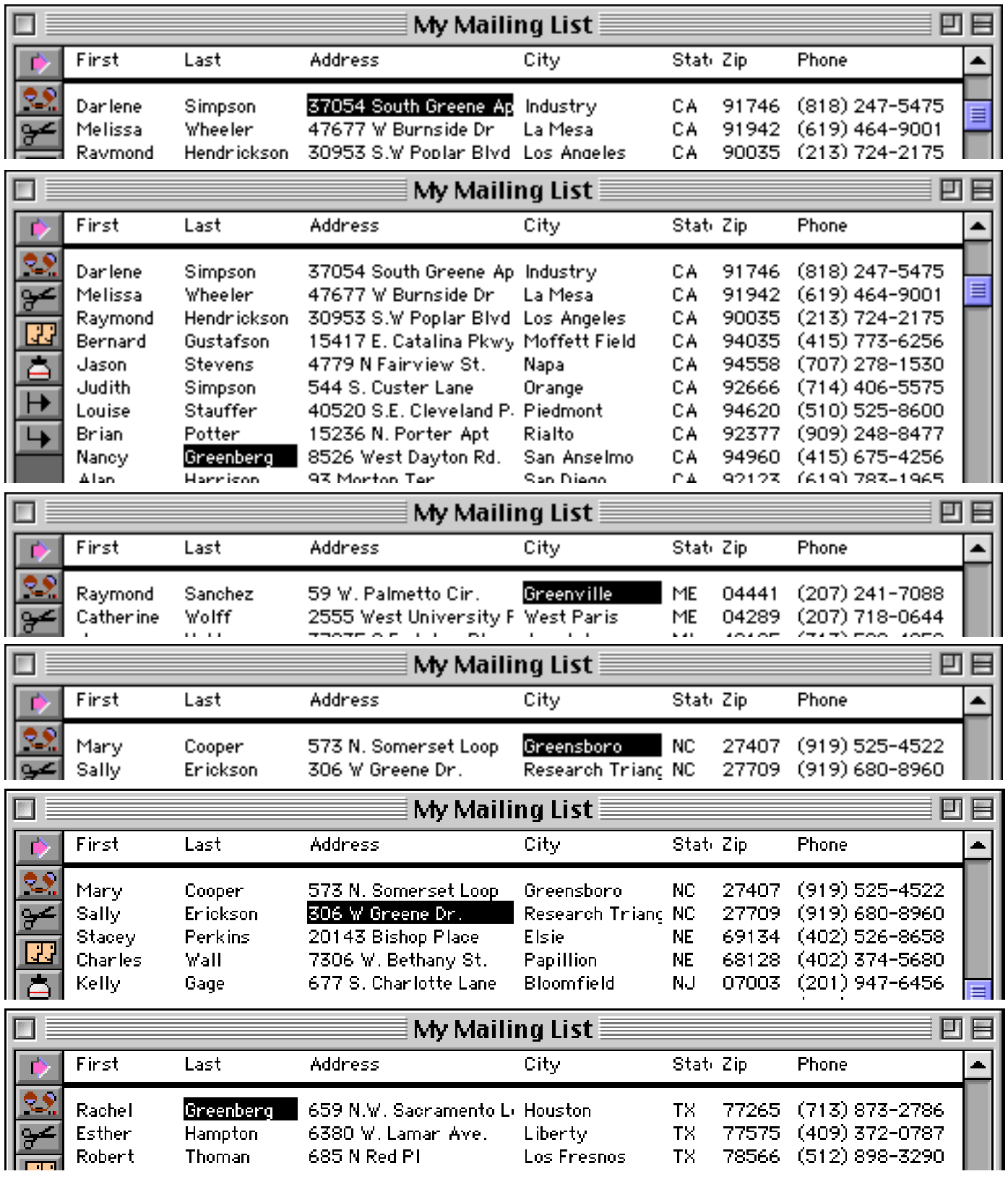


See "Primary Wizards" on page 12 of *Wizards & Demos*.

New Search All Fields Wizard

The **Search All Fields** wizard makes it easy to search all of the fields in a database at once instead of one field at a time. Simply enter the word or phrase you want to locate and press either the **Find** or **Select** button.

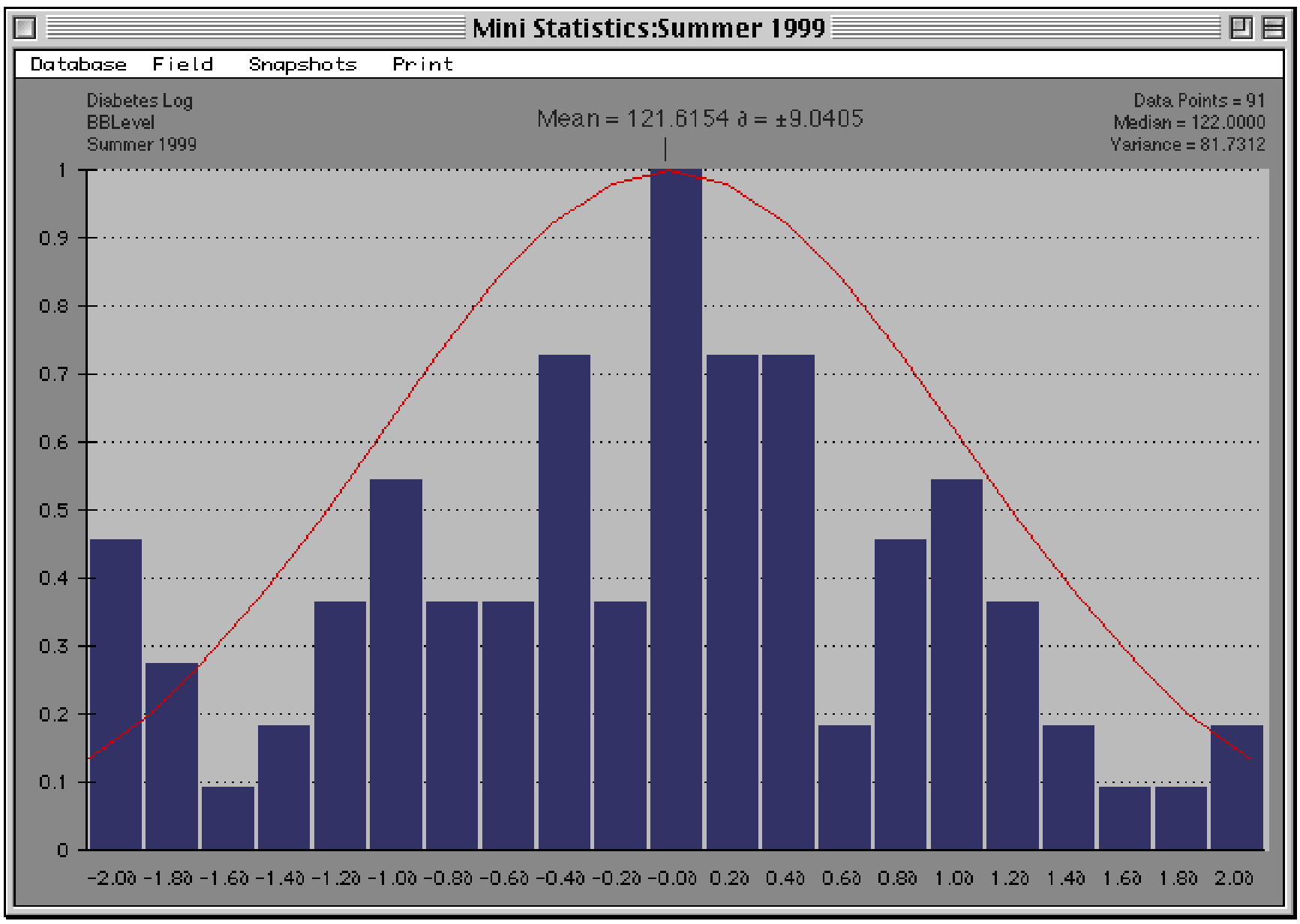
The wizard will locate the word or phrase no matter what field it is located in. If you use the **Find** button you can jump through the database with the **Next** button to locate every occurrence of the word or phrase (in this case Green).



For more information on this wizard see "Search All Fields Wizard" on page 115 of *Wizards & Demos*.

New Mini Statistics Wizard

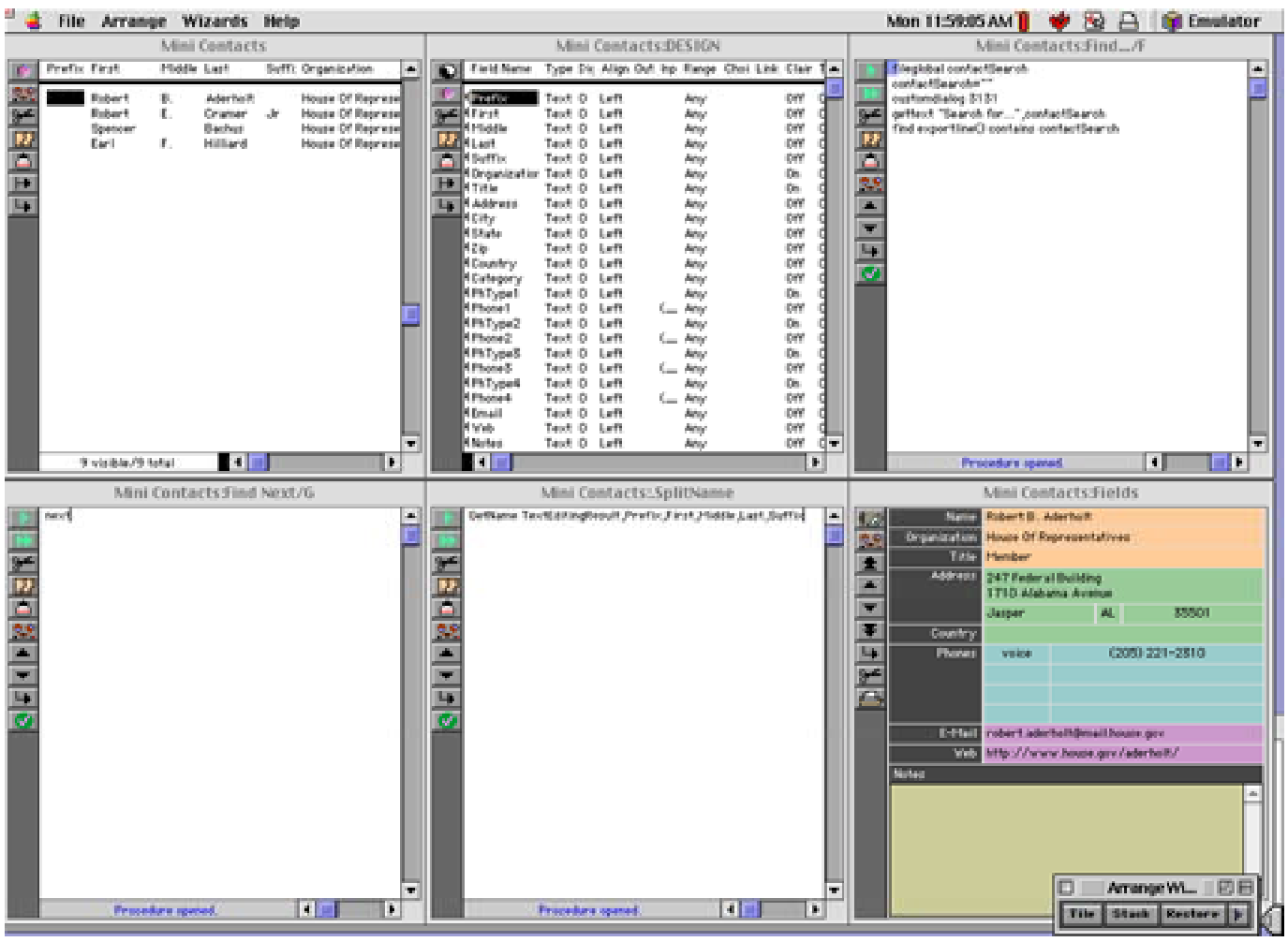
The new **Mini Statistics** wizard can automatically calculate the mean (average), median, and standard deviation of a data set. In addition the wizard can plot a normalized chart showing how the data is distributed around the mean. You can easily see how this distribution compares with the standard gaussian distribution (the famous bell shaped curve). Here is an example of an analysis performed by this wizard.



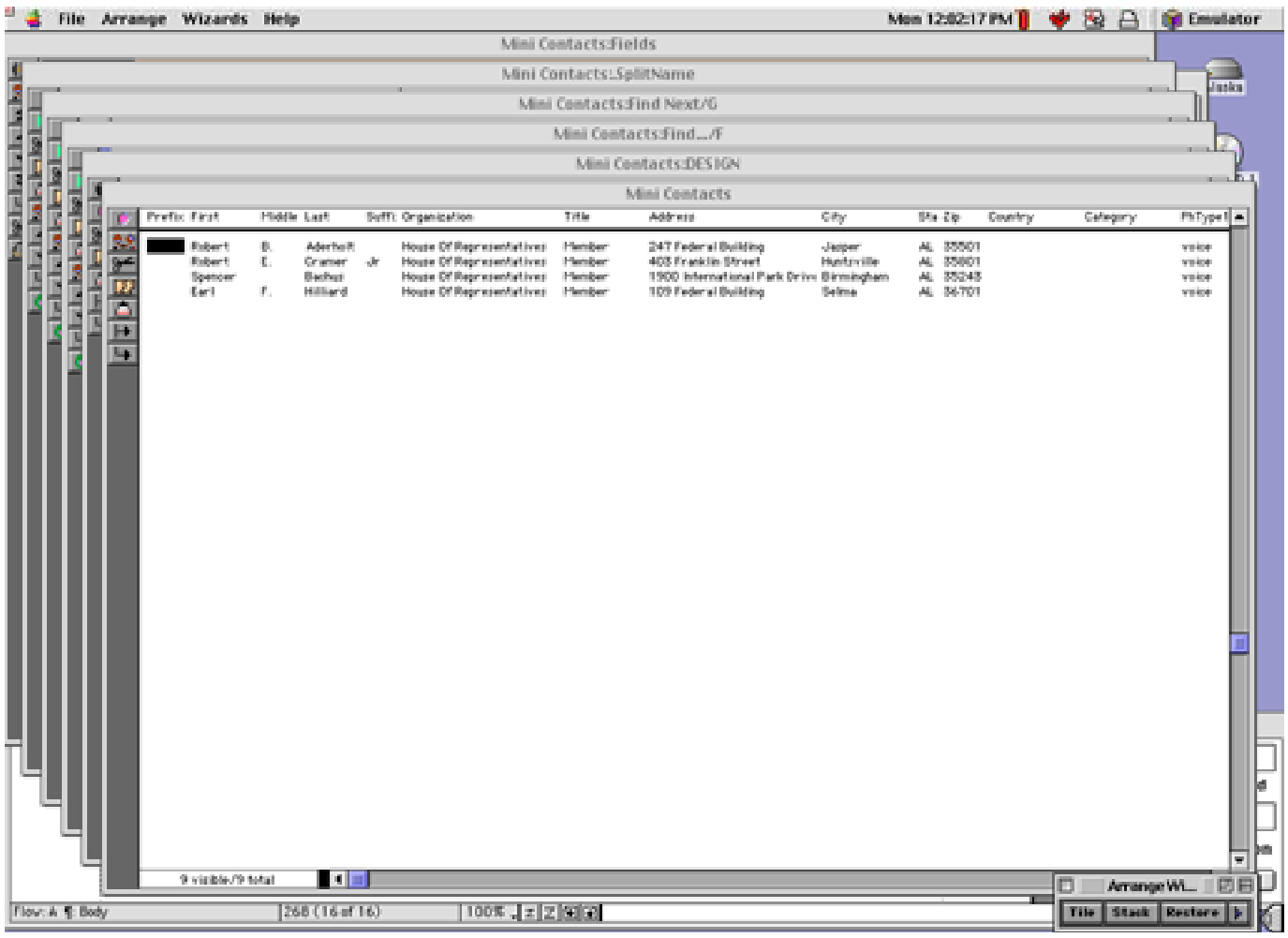
See “The Mini Statistics Wizard” on page 410 of the *Panorama Handbook* for more information.

Tiling and Stacking Windows

The **Arrange Windows** wizard makes it easy to tile or stack all of the open windows. This illustration shows an example of window tiling.



Here is an example of window stacking.



See “Arranging All Open Windows at Once (Tiling and Stacking)” on page 156 of the *Panorama Handbook* for more information.

Personal Use License

Panorama is normally licensed for use on a single computer. Panorama 4.0.1 now supports a new personal use license which allows a single Panorama serial number to be used by a single person on multiple computers.

See our web site and "Using Panorama With a Personal Use License" on page 22 of *Welcome to Panorama* for more information about this license.

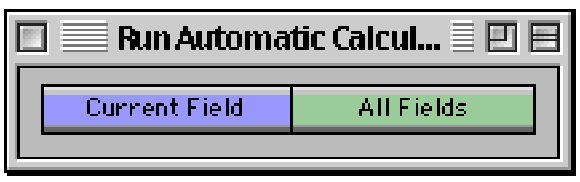
Setting Exact Window Dimensions

The **Window Size** wizard has been enhanced to allow you to specify the exact dimensions for any window.

See "Setting Exact Window Dimensions" on page 153 of the *Panorama Handbook* for more information.

Run Automatic Calculations Wizard

This new wizard allows the formulas set up in the design sheet to be applied to existing data.



See "The Run Automatic Calculations Wizard" on page 316 of the *Panorama Handbook* for more information.

Hiding Windows

The **Hide This Window**, **Hide Other Windows** and **Show All Windows** commands (in the **Arrange** menu) now work properly on Windows PC systems. See "Hiding Windows" on page 147 of the *Panorama Handbook* for more information.

More Complex Charts

Panorama 4.0.1 allows charts with up to 5,000 data points (the previous maximum was 500 points). See "Maximum Number of Chart Points" on page 1014 of the *Panorama Handbook* for more information.

Alternate Key for Opening New Windows

When using the **View** menu on the Macintosh, Panorama 4.0.1 allows either the **Control** key or the **Option** key to open an additional window (see "Opening More Than One Window Per Database" on page 169 of the *Panorama Handbook*).

Using the Esc Key to Cancel Data Entry

When editing a data cell pressing the **Esc** closes the Input Box without updating the cell (see "The Input Box" on page 272 of the *Panorama Handbook*). Previously this could be done only by pressing **Command-Period** (Mac) or **Control-Period** (Windows).

Using the Esc Key to Toggle Form Modes

Panorama 4 added the ability to use the Esc key to toggle between Data Entry and Graphic Design modes when using a form window. In Panorama 4.0.1 this feature is disabled if the window does not have a tool palette (see "Using the Keyboard to Select Common Tools" on page 500 of the *Panorama Handbook*).

Using the Option/Alt Key to Zoom Out

When using the **Zoom** tool in a form, pressing the **Shift** key has always shifted to "zoom out" mode. Many other applications use the **Option** key for zoom out, so Panorama 4.0.1 now allows you to use either. You can also use the **Space Bar**. See "Magnification and Reduction" on page 579 of the *Panorama Handbook* to learn more.

Simulating Panorama Direct and Panorama Engine

Sometimes when developing a database for in-house or commercial distribution it may be useful to test the database on a copy of Panorama Direct or Panorama Engine. Panorama 4.0.1 now allows this testing to occur on your development system using the `simulatedirect` and `simulatengine` statements.

New Page Numbering for Panorama Reference

To help make it easier to follow references between the main **Panorama Handbook** and the **Panorama Reference**, all page numbers in the reference now start with 5000. Any page number in the 5xxx range refers to the **Panorama Reference**.

Documentation Code Sample Corrections

Due to a font rendering problem when creating the PDF file, the code samples in the Panorama 4.0 documentation sometimes contained incorrect special symbols. For example the ≠ symbol was frequently displayed as **>**, and other special symbols were also incorrect. This has been corrected throughout this new version of the documentation.

New KeyNow Statement Simulates Keystrokes Immediately

Panorama 4.0.1 includes a new statement: `KeyNow`. This statement is similar to the `key` statement but processes the key immediately, making it useful for automatic demos.

New info("imagepack") function

The new `info("imagepack")` function checks to see if the Enhanced Image Pack is installed.

Displaying Images and Icons from Resource Files

Super Flash Art objects can display images and icons stored in resource files (see “Displaying Images from Resource Files” on page 802 of the *Panorama Handbook*). This feature actually has been in Panorama for some time but was not previously documented.

Version 4.0

First released in July 2001, Panorama 4.0 has been completely re-engineered to make it cross platform. This is the first version of Panorama that runs on Windows PC systems and the first version that is native on Power Macintosh computers.

Cross Platform Compatibility

Panorama 4.0 allows the same database to be moved back and forth between Power Macintosh and Windows PC systems, or even accessed over a cross platform network. To learn how to set up your databases for cross platform compatibility see “Platform Converter Wizard” on page 800 of *Formulas & Programming*.

Performance Enhancements

The Macintosh version of Panorama 4.0 is now Power Mac native, which means that it is now optimized for speed on Power Macintosh computers. (It also means that Panorama 4.0 will not run on older 68K based Macintosh systems. We will continue to sell Panorama 3.1.5 for these older computers, but the 68K version will no longer be updated.) Depending on the operation being performed Panorama 4.0 is up to twice as fast as Panorama 3.1 on the same computer. We especially concentrated on the speed of sorting, searching and selecting data.

In addition to the overall speed improvements there is also a tremendous improvement in the speed of SuperMatrix objects (for calendars, photo thumbnails, etc.). The display speed of matrixes is up to 10 times faster than the previous version of Panorama (depending on the formulas being used).

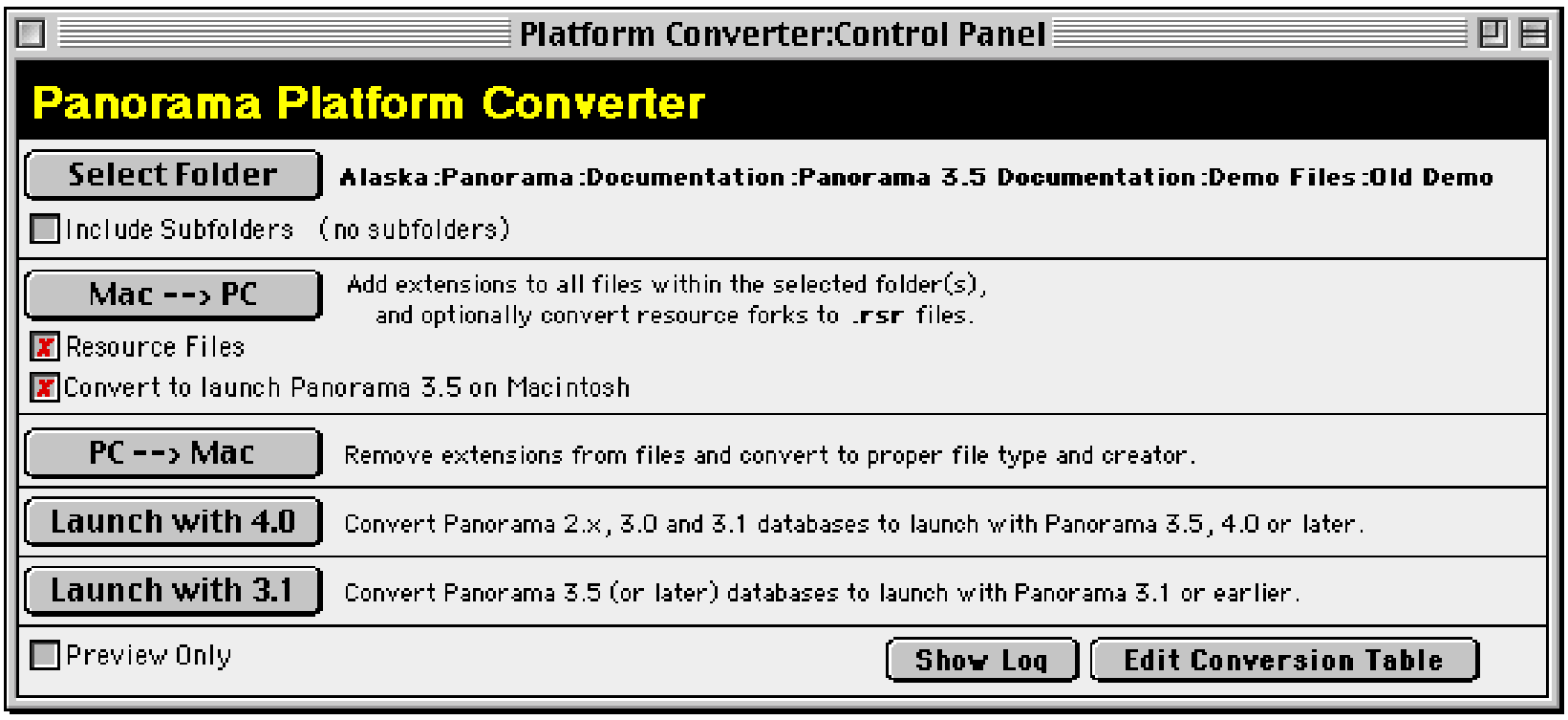
Converting from Panorama 3.x to 4.0 (Macintosh)

Panorama 3.1 and Panorama 4.0 can both co-exist on the same Power Macintosh computer, and in fact both can be running at the same time! Databases initially created with Panorama 3.1 will automatically open Panorama 3.1 when double clicked; databases created with Panorama 4.0 will automatically open Panorama 4.0 when double clicked. Panorama 4.0 can open databases created with Panorama 3.1 or earlier simply by dragging these files onto Panorama 4.0 or by using the **Open File** dialog. If you want Panorama 4.0 to launch automatically when a Panorama 3.1 database is double clicked, you must convert it using the Platform Converter (see “Platform Converter Wizard” on page 800 of *Formulas & Programming*). After the database is converted it's icon will change from the old Panorama icon to a new icon.

old icon (Panorama 3.1 or earlier)

new icon (Panorama 4.0 or later)

To convert all the databases in a folder so that they will automatically launch Panorama 4.0 instead of 3.1 you must use the Panorama Platform Converter. First, select the folder. Then choose the **Launch with 4.0** button. The Converter will scan the files and convert any Panorama 3.1 files to 4.0.



If you want to go back to 3.1, use the **Launch with 3.1** button. This converts the files so that they will automatically launch Panorama 3.1 (or whatever earlier version of Panorama you have installed). Note: If a database was lasted opened on a Windows computer Panorama 3.1 will not be able to open the file, and it will not be converted.

Wizards

Panorama 4.0 includes a number of pre-built databases that you can use as is, modify for your own purposes, or simply use as learning tools. With only a few exceptions these pre-built databases are completely accessible so that you can not only use them as is but also take them apart and see how they work. All of these databases can be opened with the **Wizards** menu, which is just to the left of the **Help** menu and several of them can be accessed with the **Start** or **Apple** menu even when Panorama is not open The table below lists the wizards included with Panorama 4.0 and provides a short description of each one.

Category	Wizard	Page	Description
General Productivity	Mini Contacts	Page 81	Basic name & address database.
	Mini Calendar	Page 85	Basic calendar / event database.
	Mini Calculator	Page 23	Basic math calculator.
	Mini Correspondence	Page 87	Basic correspondence / mail merge database. Can be linked with any database that contains names and addresses to create individual letters or mail merge letters.
	Stopwatch	Page 88	Simple timer.
	Task Timer	Page 89	Keep track of time spent on different tasks.

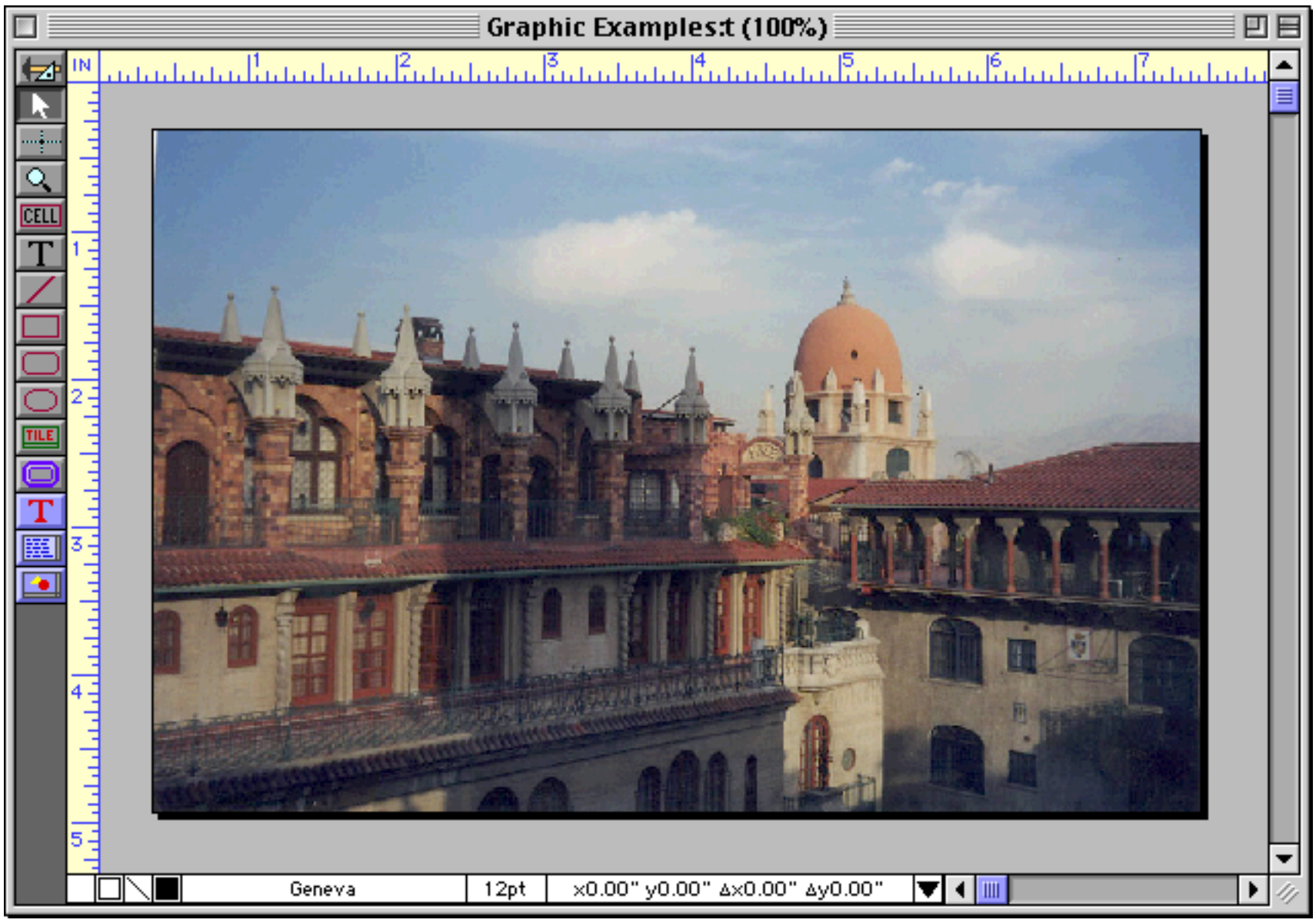
Category	Wizard	Page	Description
Database Operation	New Database Wizard	Page 13	Helps to design and create the field structure for a new databases. You can simply type in the field names you want or select from about two dozen templates.
	Favorite Databases Wizard	Page 13	Helps to keep track of your frequently used databases. We've preloaded this with several dozen sample and tutorial databases.
	Summaries & Outline	Page 125	Categorize and subtotal database information with a single click.
	Text Export	Page 62	Export data into text files. You can control the format of the exported data, including exporting data as HTML tables. Export formats can be saved for later use.
	Text Import	Page 63	Imports data from text files into existing databases. Drag and drop to define how columns in the text file are mapped to database fields, and save configurations as templates for later use.
Programming & Development	ASCII Chart	Page 28	Displays a table of ASCII characters.
	Custom Menu Editor	Page 42	Edits custom menu resource files. This wizard allows you to create and modify custom menus without a separate resource editor program.
	Debug Log	Page 33	Traces the internal operation of a PanTalk procedure for quick debugging
	Font Usage	Page 50	Display list of fonts used in forms.
	Form Explorer	Page 50	Display / edit information about form objects.
	Formula Wizard	Page 23	Workbench for experimenting with formulas. Allows you to test formulas before you use them for real.
	RPN Programmers Calculator	Page 24	Calculator for decimal, hex, octal and binary.
	View Wizard	Page 22	Opens form and procedure windows (an alternate to the View menu). Also allows you to search all procedures in a database for a word or phrase.
	Window Size	Page 51	Display size of any window.
	Window Tweak	Page 51	Disable and enable window tool palettes and scroll bars without changing the window size.

Font Management across Multiple Computers and Platforms

Panorama 4.0 keeps track of font usage within each database and checks that the necessary fonts are installed in the system each time the database is opened. As long as fonts have the correct name they may be used on any computer — even when switching platforms from Macintosh to PC and back. For more information see "Maintaining Fonts across Multiple Computers and Platforms" on page 529 of the *Panorama Handbook*.

Enhanced Image Pack

The optional **Enhanced Image Pack** allows a database form to display advanced format images including JPEG, TIFF (except for LZW compressed TIFF), PNG and many others. To learn how to use this package see "Displaying Non PICT Images (Enhanced Image Pack)" on page 775 of the *Panorama Handbook*.

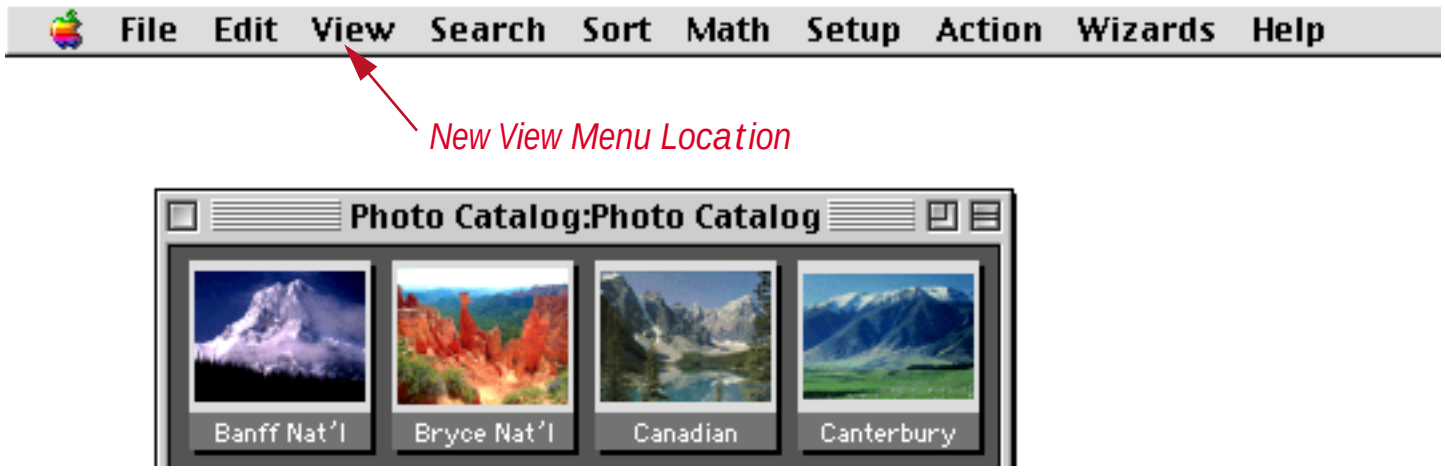


The **Enhanced Image Pack** also adds two new programming statements to Panorama — **`convertimage`** and **`imagequality`**. These two statements give Panorama the ability to convert images from one format to another (for example from TIFF to JPEG or from PICT to PNG). You can also change the size (height and width) of an image. For example, you can take an entire folder of images and create tiny thumbnails for them automatically. See "Converting Between Image Formats" on page 694 of *Formulas & Programming* to learn how to use these statements.

View Menu Moved to Menu Bar

In previous versions of Panorama the **View** Menu appeared in the title bar of each window (as shown in the illustration below). The menu appeared when you clicked on the small menu location

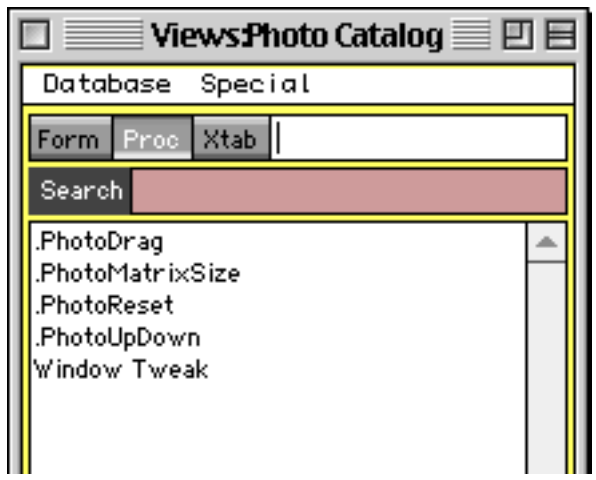
In Panorama 4.0 the **View** menu has been moved into the menu bar. The **View** menu always appears just after the **Edit** menu.



For more information on the **View** menu see "Switching Between Views" on page 168 of the *Panorama Handbook*.

View Wizard

Panorama 3 also provides an alternate method for opening views, the **View Wizard**.



The **View Wizard** is especially handy for complex databases with hundreds of forms and / or procedures. The Wizard lists the forms and procedures in alphabetical order and allows you to search for the form or procedure you want. For more information on this wizard see "The View Wizard" on page 173 of the *Panorama Handbook*.

Using the View Menu with Custom Menus

When using custom menus Panorama automatically places the **View** menu immediately after the **Edit** menu, but it can also be customized.

Graphics Mode Keyboard Shortcuts

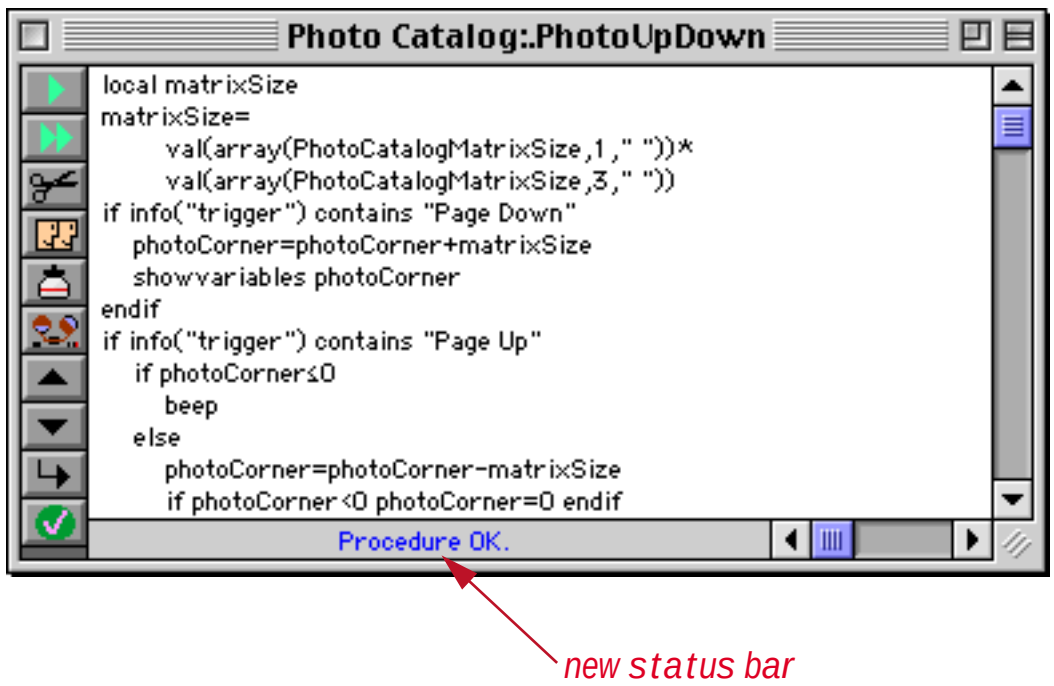
When editing a form you can now use the keyboard to select several common tools (pointer, text, rectangle, etc.). This can save many trips to the tool palette and back. See "Using the Keyboard to Select Common Tools" on page 500 of the *Panorama Handbook* for the details.

Improved Procedure Editor

We've modified the procedure editor window to make it easier than ever to create and modify procedures.

Status Bar

A new status bar at the bottom of each procedure window shows the current status of the procedure both when editing and debugging (see “Improved Debugging Tools” on page 165).



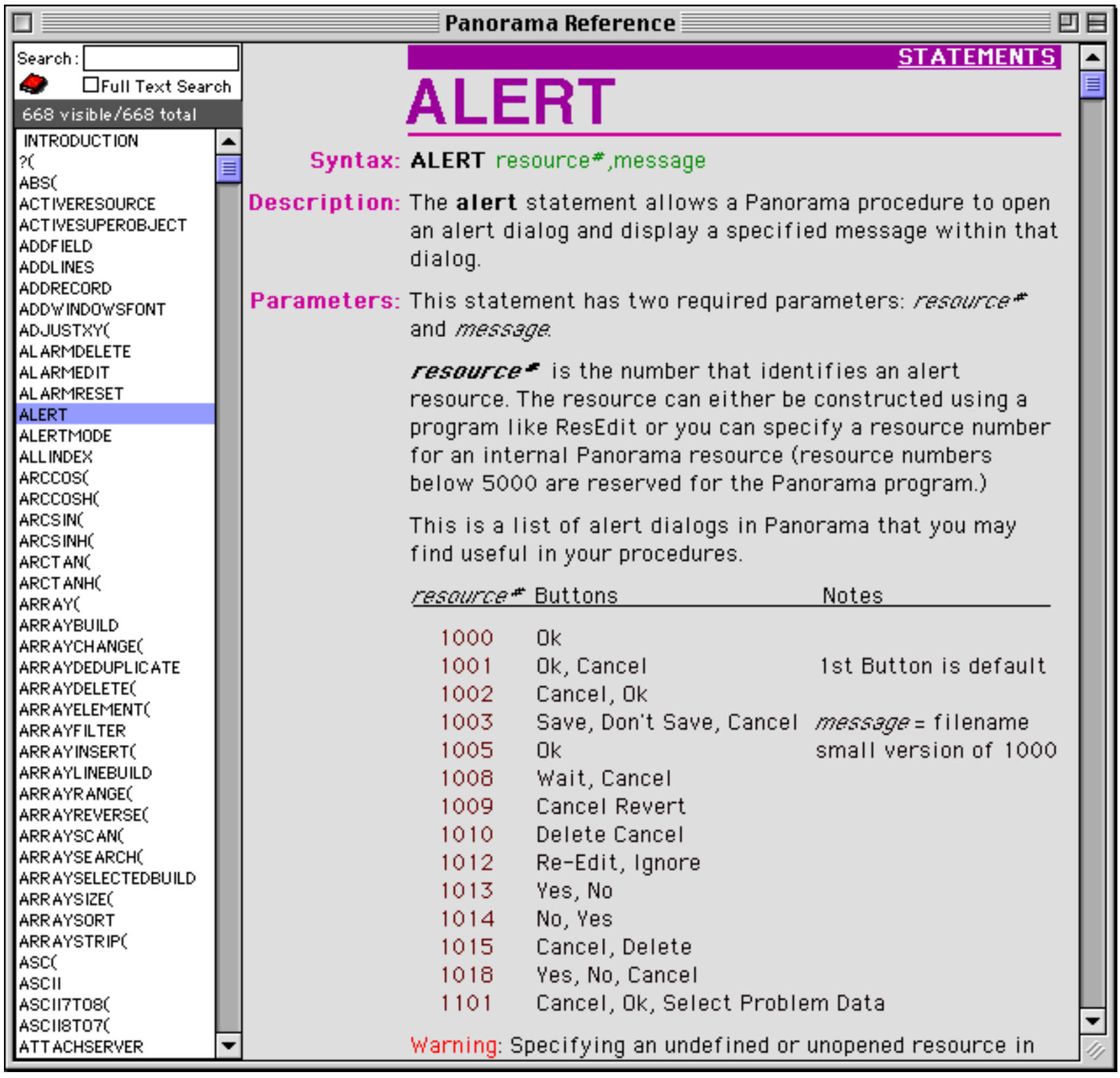
When a procedure contains an error Panorama no longer displays an alert. Instead, the error appears in the status bar (see “Checking for Mistakes” on page 220 of *Formulas & Programming*). This makes it much easier to continue if you want to ignore the error and come back to it later, since you don’t have to bother pressing the **OK** button to continue your work.

Shifting a Block of Text Left or Right

The **Edit** menu now contains commands that can shift an entire block of text left or right 4 spaces. This makes it easy to adjust the indentation of your program when you add another **`if`**, **`case`** or **`loop`** statement. See “Program Formatting” on page 301 of *Formulas & Programming* to learn more about this new editing tool.

On-Line Programming Reference

Panorama now includes a complete on-line reference to all programming statements and functions. This on-line reference is searchable and includes numerous links between topics (see "Programming Reference Wizard" on page 237 of *Formulas & Programming*).

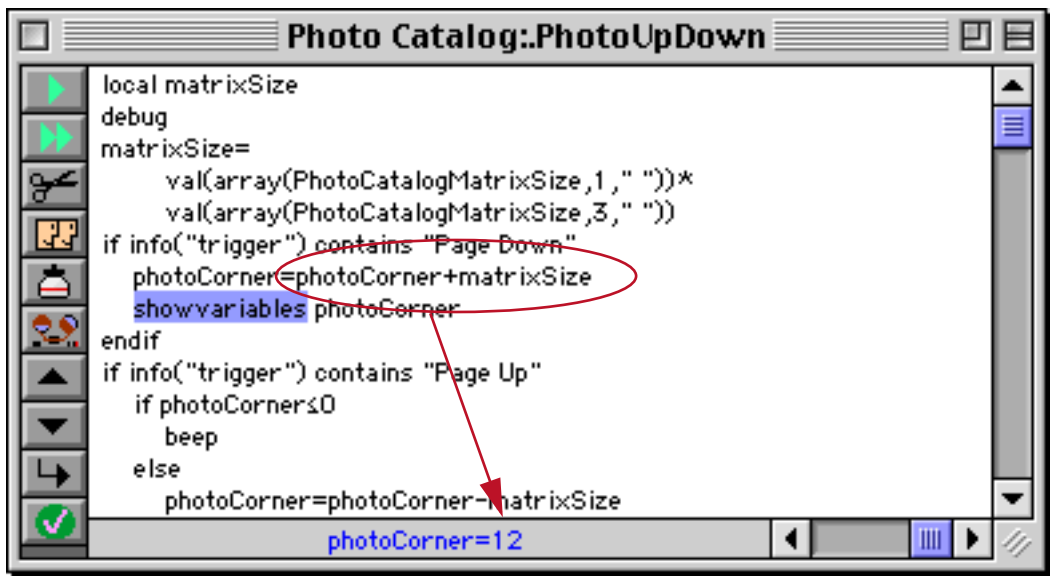


Improved Debugging Tools

Debugging your procedures has never been easier or faster.

Displaying Values While Single Stepping

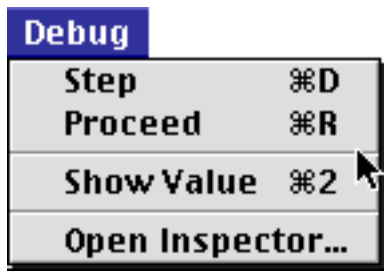
The new procedure status bar automatically displays the result of every assignment statement while single stepping. This makes it much easier to "follow the action" as Panorama executes each step. See "Single Stepping" on page 316 of *Formulas & Programming* to learn more about single stepping.



If the value you need to see is not the result of an assignment statement you can add one or more **`statusmessage`** statements to your procedure. This statement displays the result of any formula in the status bar, see "Watching Computations" on page 318 of *Formulas & Programming*.

New Command Key Equivalents (Shortcuts) for Debugging

In previous versions of Panorama the command key equivalents for debugging were the same as for the Jasik Debugger. Panorama 4.0 now uses the same command key equivalents as CodeWarrior.



Debug Log

The debug log allows you to make a record of every step made by a procedure. If the procedure doesn't work you can review the log to see where it went wrong — even if the computer crashed. We created the debug log to help us debug Panorama itself. Once we started using it we discovered that it is such a valuable tool that we had to share it with you. To learn more about this powerful tool see "Procedure Debug Log" on page 336 of *Formulas & Programming*.

Hot Keys

Panorama 4.0 allows you to assign any procedure to any keystroke. You can assign hot keys to a window, a database, or globally across all databases. See "Hot Key Procedures" on page 390 of *Formulas & Programming*.

Triggering a Procedure Every Minute or Second

Panorama 4.0 allows you to set up procedures that are triggered automatically every second or every minute. You can use these repeating procedures to create timers, alarms, and animations. See "Triggering a Procedure Every Second" on page 391 of *Formulas & Programming*.

Credit Card Data Entry Validation

Credit cards have an internal checksum that allows a number to be validated for simple data entry errors (for example missing or transposed digits). The new **`cardvalidate`** statement checks to make sure that a number is a potentially valid credit card number. See "Validating a Credit Card Number" on page 533 of *Formulas & Programming*.

Calculating Time Intervals Smaller Than One Second

The new **`info("tickcount")`** function allows you to calculate time intervals and delays as small as 1/60th of a second. See "Calculating Time Intervals Smaller Than One Second" on page 116 of *Formulas & Programming*.

Elastic View-As-List Forms

Previously only regular forms could be made elastic, now view-as-list forms can be elastic too! See "Elastic View-As-List Forms" on page 937 of the *Panorama Handbook* to learn how to create an elastic view-as-list form.

New QuickTime Features

We've upgraded Panorama's QuickTime capabilities. Instead of only showing the QuickTime controls when you click on a movie the controls now appear whenever the form is visible. Existing databases that use movies will have to be slightly re-designed, see "Displaying Movies in a Form" on page 819 of the *Panorama Handbook* to learn how to use movies with Panorama 4.0.

A Panorama 4.0 procedure can exercise complete control over a movie. The procedure can start or stop a movie, control the playback speed, or even jump to a pre-defined spot in a movie. See "Super Flash Art Commands (Including Movie Control)" on page 690 of *Formulas & Programming* to learn how to program a movie.

SuperObject Enhancements

A number of SuperObjects have been enhanced in Panorama 4.0. One change that affects almost all types of SuperObjects is the ability to access and modify the configuration of most SuperObjects (see "Accessing and Modifying a SuperObject's Internal Data" on page 669 of *Formulas & Programming*).

Two other improvements apply to the **superobject** statement. First, you can now specify what objects are controlled "on-the-fly" (see "Program Control of SuperObjects™" on page 666 of *Formulas & Programming*). In addition you can also now use this statement to send commands in any open window, not just the current window (see ""Magic" Windows" on page 456 of *Formulas & Programming*).

Text Display SuperObject

The Text Display SuperObject no longer displays error messages when the form is in data mode (only in graphics mode). This means that if you use a Text Display SuperObject to display a variable Panorama won't display an "undefined field or variable" message when the form first opens (before the variable is initialized with a value).

Flash Art SuperObject

If the first character of the formula is @ Panorama treats the rest of the line as a variable name instead of a formula. The actual formula is stored in the variable. This allows you to easily change the formula on the fly, and also allows formulas longer than 255 characters. See "Formula in a Variable" on page 786 of the *Panorama Handbook*.

Panorama 4.0 also allows a procedure to change the configuration of a Super Flash Art object on the fly. See "Super Flash Art Internal Data" on page 693 of the *Panorama Handbook*.

List SuperObject

The List SuperObject hasn't actually changed, but there is a slight change in the documentation for changing the list formula on the fly - you must use { } around the formula. See "List SuperObject™ Commands" on page 708 of *Formulas & Programming*.

SuperMatrix SuperObject

A couple of minor programming revisions for this object. First there is a new programming command:

```
SuperObject "Object","scroll",RowDelta,ColDelta
```

This command allows a procedure to scroll the matrix in any direction (see "Super Matrix SuperObject™ Commands" on page 715 of *Formulas & Programming*).

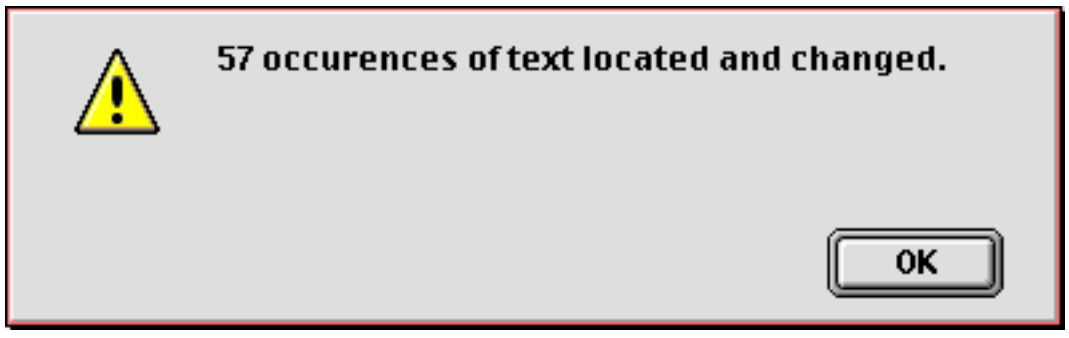
A new Super Matrix option, Sync Up / Dn, makes it easy to update the matrix as different records in the database are clicked on. See "Updating the Matrix Display" on page 957 of the *Panorama Handbook*.

Form Preferences Dialog

This dialog has a new option — FileGlobal Variables. When this option is enabled any variables created by SuperObjects in the form will be fileglobal variables instead of global variables (see "Creating Variables with a SuperObject" on page 251 of *Formulas & Programming*).

Change Command Reports Changes

The **Change** command (in the Search menu) now tells you what it did, if anything.



This command has also been modified to allow replacing text in fields containing up to 32,768 characters (this limit was 10,000 characters in previous versions).

Stop Cursor Flashing

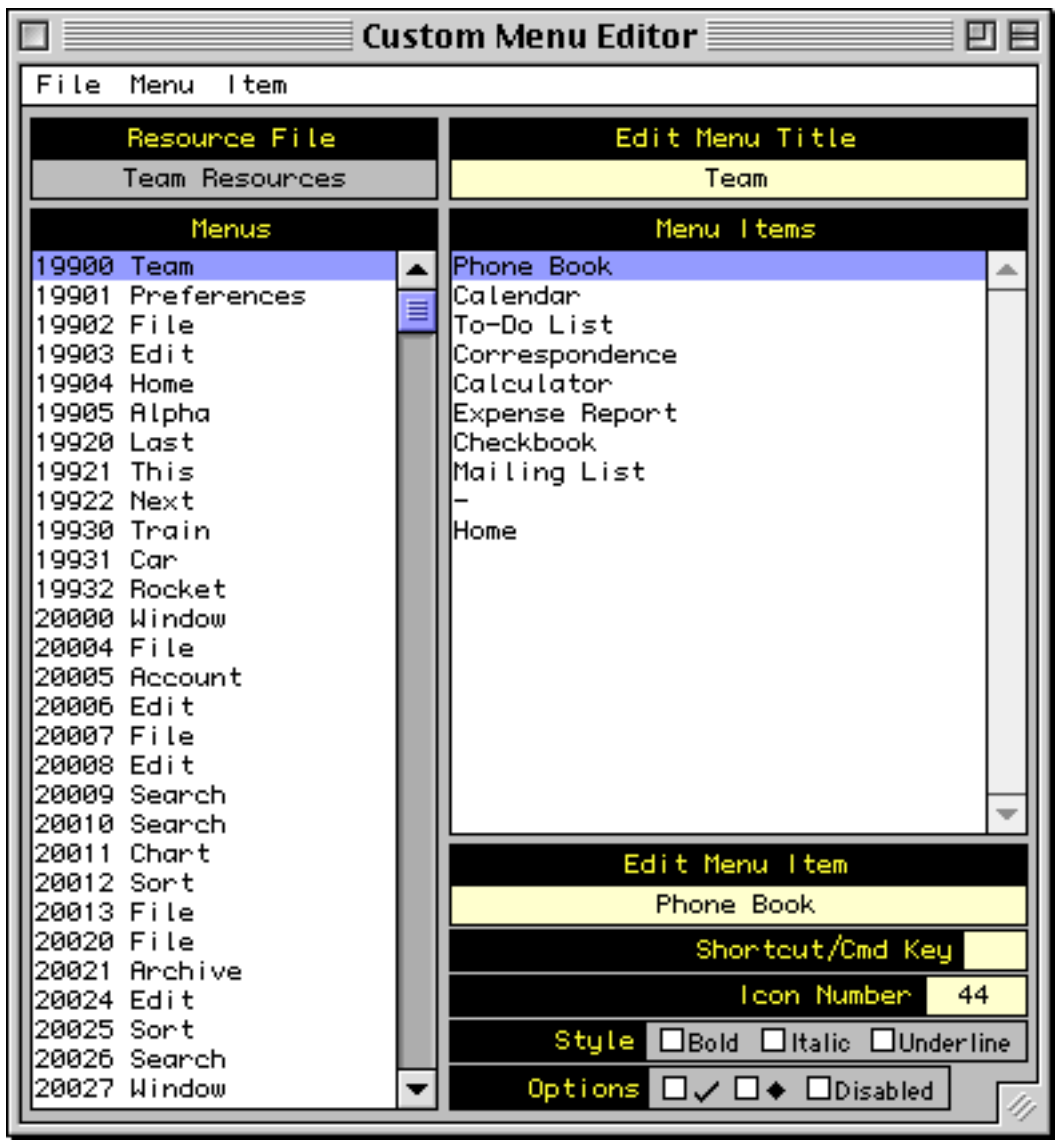
Panorama normally flips the mouse cursor from an arrow to a watch or pie chart when performing an operation that may take a while. With today's faster computers this often isn't necessary, and it can be somewhat annoying. To disable the watch and pie chart cursors during a procedure you can use the new **`nowatchcursor`** statement. See "Disabling the Watch Cursor" on page 310 of *Formulas & Programming*.

Destroy Variables At Any Time

The new **`undefine`** statement allows a program to destroy a variable that has been created with the local, windowglobal, fileglobal or global statement. "Destroying a Variable" on page 249 of *Formulas & Programming* to learn how to do this.

Improved Resource Editing Tools

Previous versions of Panorama had no capability to edit resource files. If you wanted to create custom menu resources you had to use a separate editing program like ResEdit or Resourcerer. Panorama now includes a menu resource editor that you can use within Panorama on both Macintosh and Windows PC systems.



The custom menu resource editor was built using new statements and functions available with Panorama 4.0. If you need to modify resources in your PanTalk procedure see "Writing a Resource" on page 437 of *Formulas & Programming*, "Deleting a Resource" on page 438, "Renumbering a Resource" on page 438 and "Working with Multiple Resource Files" on page 440.

Opening Documents with Other Applications

Since Panorama 3 the Macintosh version of Panorama has had the ability to open documents in other applications (using an AppleScript). The new **`shellopendocument`** statement extends this capability to the Windows operating system (see "Opening a Document in Another Application" on page 416 of *Formulas & Programming*).

Windows Registry

If you don't know what the Window's Registry is you probably shouldn't be messing with it! Propellerheads (you know who you are) should turn straight to "Accessing the Windows Registry" on page 441 of *Formulas & Programming*.

Memory Allocation on Windows PC Systems

The default memory allocation for Panorama on Windows PC systems is 32 megabytes, which is enough for all but the largest databases. See "Adjusting Panorama's Memory Allocation" on page 140 of the *Panorama Handbook* if your databases are larger than this. (Unlike the Macintosh there is no Scratch Memory allocation on Windows PC systems.)

Autoload File Set

On PowerPC systems the name of the .AutoLoad file set has been changed to AutoLoad. (no period). On Windows PC systems the name of this file must be AutoLoad.pnz. See "The AutoLoad File Set" on page 77 of the *Panorama Handbook*.

Working with Files

Two new statements allow a program to access and modify the position, creation date, modification date and operating system flags for any file. For example, these statements can be used to make a file invisible. See "Getting and Setting Additional File Information" on page 431 of *Formulas & Programming*.

Windows PC systems introduce a new concept not needed on Macintosh computers, the file extension. (The file extension is the three or four character suffix at the end of the file name, for example .pan or .txt). There are several new statements and functions to help you work with file extensions. See "Suppressing the Default Extension" on page 405 of *Formulas & Programming*, "Importing a Text File into an Existing Database" on page 409 (look for the **opentext** statement) and "INFO("DATABASEFILENAME")" on page 5366.

New Procedure Statements

The table below lists the new procedure statements available in Panorama 4.0.

Statement	Reference	Description
activeresource	Page 5011	Select which open resource file to work with.
addwindowsfont	Page 5019	Install font (Windows only).
cardvalidate	Page 5090	Validate credit card number
convertimage	Page 5120	Convert image file to a different format / resolution.
deleteresource	Page 5163	Delete resource item.
drawobjects	Page 5183	Draw selected objects.
getfilefinderinfo	Page 5296	Get information about file position, date, etc.
getproceduretext	Page 5310	Get text of procedure (source code).
imagequality	Page 5351	JPEG compression setting for convertimage.
logmessage	Page 5494	Write formula result to log file (for debugging).
magicformwindow	Page 5517	Work with alternate window.
magicwindow	Page 5518	Work with alternate window.
nodefaultextension	Page 5540	Temporarily disable automatic .pan extension.
nowatchcursor	Page 5549	Temporarily disable watch and pie chart cursors
opentext	Page 5585	Open text file with any extension (.ini, .html, etc.)
registrydelete	Page 5639	Delete registry value, subkey or key.

Statement	Reference	Description
registrywrite	Page 5641	Modify registry value.
renameresouce	Page 5659	Change number and / or name of resource item.
scratchmemorytemporary	Page 5704	Temporarily change scratch memory allocation.
setfilefinderinfo	Page 5741	Set information about file position, date, etc.
statusmessage	Page 5795	Display message in status bar (for debugging).
shellopendocument	Page 5757	Open document in another application (Windows)
undefine	Page 5866	Destroy one or more variables.
watchcursor	Page 5889	Enable watch and pie chart cursors
writeresource	Page 5908	Modify value of resource item.

Revised Procedure Statements

A new function, **`info("changecount")`**, can be used to find out how many items were located and changed by the **`change`** statement. See "Change (Find and Replace)" on page 584 of *Formulas & Programming*.

There are six new resource templates that can be used with the **`gettext`** statement — 3121, 3122, 3123, 3125, 3120 and 3131 (see "Basic Text Entry Dialogs" on page 480 of *Formulas & Programming*). These templates contain **OK** and **Cancel** buttons (instead of **OK** and **Stop**). If you use one of these templates the procedure will not stop even if the **Cancel** button is pressed. The procedure can use the **`info("dialogtrigger")`** function to find out what button was pressed.

The dialog opened by the **`alarmedit`** statement now allows years up to 2020 A.D. (see "ALARMEDIT" on page 5023).

New Functions

The table below lists the new formula functions available in Panorama 4.0.

Statement	Reference	Return Value
info("applemenufolder")	Page 5359	Location of Apple Menu folder.
info("changecount")	Page 5362	Number of changes made by last **change** statement.
info("databasefilename")	Page 5366	Actual file name of current database.
info("desktopfolder")	Page 5369	Location of desktop folder.
info("magicwindow")	Page 5390	Name of magic window, if any.
info("matrixname")	Page 5393	Name of matrix that was clicked on.
info("openresourcefiles")	Page 5402	List of open resource files.
info("panoramabuild")	Page 5404	Time and date Panorama application was created.
info("preferencesfolder")	Page 5409	Location of Preferences folder.
info("startupfolder")	Page 5425	Location of Startup Items folder.
info("tempfolder")	Page 5431	Location of folder for temporary items.
info("windowoptions")	Page 5444	Window options (scroll bars, tool palette, drag bar).
info("windowview")	Page 5450	Type of window.
registryinfo(	Page 5640	Information about registry key or value.

Custom Dialog Wizard

The **Custom Dialog Wizard** was introduced at the ProVUE 98 conference (in August of 1998). This wizard is actually a set of procedures that you can copy into your database to make creating custom dialogs with forms a snap. The wizard actually writes the code for you! We are now including this wizard as part of Panorama, see Page 489. (Note: If you attended the ProVUE 98 conference (or purchased the CD set afterwards) the version of the Custom Dialog Wizard included with Panorama 4.0 has been updated. You should update to the new version if you plan to create new dialogs.)

New Documentation

The new Panorama documentation describes a number of topics in much greater detail than previous versions of the manual. Here is a list of some of these expanded topics.

"Drag and Drop (Obsolete Method)" on page 658
"Program Control of SuperObjects™" on page 666
"Programming a HyperText Engine" on page 699
"Printing Data in an Array" on page 726

If you have read the **Panorama Real World Programming Guide** some of these topics will be familiar, while others are completely new.

Unsupported Panorama 3.1 Features

Almost all Panorama 3.1 features are supported by Panorama 4.0 on both Macintosh and Windows PC platforms. However, external procedures (xcalls) are not supported by Panorama 4.0 on either platform. This feature will be added in a subsequent version of Panorama.

The Windows PC version of Panorama does not support Panorama's sound playing ability. This includes phone dialing through either the speaker or serial ports. This feature is on our list of possible future enhancements.

The Windows PC version of Panorama does not support the ability to customize the open file dialog and save file dialog with the `customdialog` statement.

The Windows PC version of Panorama does not support making windows "invisible" by opening them in an off screen location. It also does not support the **Hide This Window**, **Hide Other Windows** and **Show All Window** commands.

The Windows PC version of Panorama does not support use of the Butler SQL server for multi-user Partner/Server database operation. A future version of Panorama will support multi-user Partner/Server application with a new, cross platform, server.

The Windows PC version of Panorama does not support AppleScript (since AppleScript is not available for Windows PC systems!).

Version 3.1.5

This version was released on December 16, 1998, and is the latest version available for 68K computers (early Macintosh systems).

Mac OS 8.5 Bug Fix

Several Panorama dialogs did not work (crashed) when run under Mac OS 8.5, including: Custom Tool Palette Dialog, Quick Report Dialog, Access Privileges Dialog, Rearrange Form/Crosstab/Procedure Dialogs, Crosstab Setup Dialog and Rearrange Flash Art Dialog. These dialogs are all fixed.

Improved Butler/SQL Performance

Improved performance when adding new records to a Butler SQL database (up to 400% improvement). Thanks to Mark Sanchez for discovering how to do this.

New FileTypeCreator Statement

This version includes a new statement, filetypecreator. See "Changing a File's Type and Creator" on page 429 of *Formulas & Programming*.

Version 3.1.4

This version was released on August 29, 1998. This is a simple update, just one small but critical bug fix, plus a couple other minor cleanups. The critical bug would sometimes cause Panorama to ask the user to logon even when the database did not use security. This bug has been fixed. The `repeatloopif` statement has been fixed to work properly (see "Restarting a Loop in the Middle" on page 261 of *Formulas & Programming*).

Version 3.1.3

This version was released on December 31, 1997.

Special Keyboard Support

Panorama now supports the following special keys on keyboards that have them.

Page Up

Page Down

Home

End

Forward Delete

COMMAND-RIGHT ARROW to end of line

COMMAND-LEFT ARROW to start of line

COMMAND-UP ARROW (same as Home)

COMMAND-DOWN ARROW (same as End)

Holding down the SHIFT key while using any of the arrow key combinations selects the text.

Update Server Every Cell Option

When this option in the **SQL Local Setup** dialog (design sheet) is enabled, Panorama will immediately write your data to the server as soon as you enter it. (Normally, Panorama does not write the data until you move to another record.) If you have been experiencing problems with SQL, you might want to try this option. Be sure to set the option on every one of your client databases (this is a client option, not a server option). Using this option will make data entry somewhat slower, since there will be a delay after each cell you edit.

MakeFolder Statement

This new statement allows a procedure to create new folders (see “Creating a New Folder” on page 429 of *Formulas & Programming*).

Minimum Window Size (Elastic Forms)

The **`GetMinSize`** command allows a procedure to find out what the minimum size of a window is, while the **`SetMinSize`** command allows the minimum size to be modified. See “Auto Grow SuperObject™ Commands (Elastic Forms)” on page 714 of *Formulas & Programming*.

AlertMode Statement

This new statement allows you to disable all alerts when Panorama is running. For example, you might want to do this when Panorama is running on a server. See “ALERTMODE” on page 5028.

Info("FreeMemory") Function

This new function returns the amount of free memory available (not including scratch memory). See “INFO("FREEMEMORY")” on page 5382.

New Action Menus Security Option

This option, found in the **Access Privileges** dialog, allows you to disable Action Menus if the user does not have a high enough security level. For example, you might set up custom menus for all users, with Action Menus only for high level users (or even only for the database developer).

OS 8 Bug Fixes

Fixed bug that caused Panorama to crash if you clicked on the windowshade icon. This bug only occurred if Panorama was in the background, and only with OS 8. Fixed the **New Form** (QuickLabel) options (Mac OS 8 bug).Fixed the **FormSelect** and **FormComment** dialogs (Mac OS 8 bug).

Version 3.1.2

This version was released on October 6, 1997.

Info("Abort") Function

Added **`info("abort")`** function that can be used with the **`disableabort`** statement (see "Controlling the Abort Process" on page 279 of *Formulas & Programming*).

Long Window Names

The **`windowname`** statement (see "Changing the Name of a Window" on page 451 of *Formulas & Programming*) now works with window names up to 100 characters long. (Previous limit was 48 characters).

SetPlugAndRun Statement

The new **`setplugandrun`** statement allows you to change some of the parameters in the **Server>Local Setup** dialog via a procedure. See "SETPLUGANDRUN" on page 5749. Also added the **`info("plugandrun")`** function. This allows a procedure to find out what the settings in the **Server>Local Setup** dialog are. See "INFO("PLUGANDRUN")" on page 5408.

Disabling Up/Down Arrows in a Form

Added the Enable Up/Down Arrows options to the **Form Preferences** dialog. If this is turned off (the default is on) pressing the **up** or **down** arrows will not move to the next or previous record. For example, suppose you are creating a dialog using a form. In that case, you probably don't want the user to be able to get to the next or previous record with the arrow keys, and this option disables that capability on a form by form basis.

Window Management

The new **Hide This Window**, **Hide Other Windows** and **Show All Window** commands allow one or more windows to be hidden temporarily. See "Hiding Windows" on page 147 of the *Panorama Handbook*. In addition, the maximum number of open windows has been raised to 32 (was 24).

Version 3.1.1

This version was released on August 23, 1997.

New HTML Parsing Functions

The new functions **`tagparameter(`** and **`tagparameterarray(`** are handy for parsing the options in an HTML tag. See "Tag Parameter Functions" on page 103 of *Formulas & Programming*.

Sleep Statement

This new statement puts Panorama to sleep for a while, letting other programs do their thing. This parameter should be followed by a number. Bigger numbers make Panorama sleep longer. According to the documentation for the Apple system call used by this command, the sleep time should be the number specified multiplied by 1/60 second, for instance 120 for 2 seconds. However, it doesn't work that way in real life. Nevertheless, the statement is very useful for allowing programs like Netscape Navigator to do their thing while Panorama is waiting for them to finish something.

Here is an example of how to use this statement. The AppleScript tells Netscape to save a disk file. However, Netscape returns to Panorama immediately without finishing this task, so Panorama must wait for the file to be finished. Without the sleep statement, Netscape will run very very slowly.

```
startscript "Grab Netscape Page"
loop
   websource=fileload("",theFile)
   if error
      sleep 10000
   else
      stoploopif 1=1
   endif
while forever
```

Version 3.1

This version was released on July 17, 1997. Major new features include HTML table import, improved user interface and enhanced programming capabilities. This version was bundled with a copy of our SurfScout URL management utility.

HTML Table Import

Panorama's text import capability has been enhanced to allow the import of HTML tables. Panorama automatically checks any text file you import for a **<table>** tag. If the text file contains a **<table>** tag Panorama will parse the HTML and input the data in the table. See “Importing HTML Tables” on page 87 of the *Panorama Handbook*.

HTML Tag Parsing Functions

Panorama 3.1 has six functions for working with text that contains data delimited by tags. These functions are not actually specific to HTML, and you may find other uses for them. See “HTML Tag and Tag Parsing Functions” on page 101 of the *Panorama Handbook*.

HTML/URL Conversion Functions

Several new functions allow text to be converted from normal ASCII to the special format used by HTML or URL’s. See “HTML Table Parsing Functions” on page 104 of *Formulas & Programming*.

Window Clones

Panorama normally allows only a single window per form. However, Panorama 3.1 allows a single form to be opened over and over again into multiple windows. This is called window "cloning." See “Window Clones” on page 457 of *Formulas & Programming* to learn how to set this up.

Dragging To/From a List

Panorama 3.1 adds new features that make it possible to drag-and-drop to or from a List SuperObject. See “Using Drag and Drop to Change the Order of Items in a List” on page 712 of *Formulas & Programming*.

Suppressing Display of Text and Graphics

Previous versions of Panorama (up to 3.0) included the **hide** and **show** commands that allowed a programmer to turn off the display of text and graphics while the procedure was running. Unfortunately these commands did not give accurate control over the display, and worse, they could even crash if you attempted to use them across multiple windows.

Panorama 3.1 includes 7 new statements and 2 slightly modified old statements for suppressing and enabling the display of text and graphics. These statements are:

```
NoShow
EndNoShow
ShowPage
ShowLine
ShowFields field,field,…,field
ShowVariables var,var,…,var
ShowColumns field,field,…,field
ShowRecordCounter
ShowOther field?,op
```

See "Suppressing Display of Text and Graphics" on page 307 of *Formulas & Programming* for detailed information on how to use these statements.

Unlisted Procedures

Panorama has always had the capability of unlisted procedures by starting the procedure name with a period (for example .Initialize or .ButtonClick). Panorama 3.1 adds the capability to unlist an entire group of procedures all at once, even if they don't start with a period. See ""Unlisted" Procedures" on page 361 of *Formulas & Programming*.

Disabling Command-Period

Pressing **Command-Period** normally stops any procedure right in its tracks, no matter what the procedure is doing. This is normally an important safety valve, but you may have a procedure that should not be stopped in the middle with a job halfway done. For these types of cases Panorama 3.1 now allows you to disable **Command-Period** during some or all of a procedure. See "Controlling the Abort Process" on page 279 of *Formulas & Programming*.

Text Editor Padding and Grow Box Options

Two new options allow you to customize a Text Editor SuperObject — Padding and Grow Box. See "Text Editor Options" on page 643 of the *Panorama Handbook*.

Working With Complex Formulas

The new **`formulabuffer`** statement allows you to avoid the Expression Too Complicated error message. See "Working With Extremely Complex Formulas" on page 59 of *Formulas & Programming*.

ReplaceMultiple(Function

The **`replacemultiple(`** function is similar to the **`replace(`** function. However, instead of simply replacing one word or phrase with another, the **`replacemultiple(`** function takes an entire list of words or phrases and replaces them with the corresponding words and phrases in a second list. See "REPLACEMULTIPLE(" on page 5667.

ExportCell(Function

This new function takes any database field and converts it to text. You do not have to know the type of data in the field (text, number or date). See "EXPORTCELL(" on page 5209.

OnError Statement

The **`onerror`** statement can be used to catch all errors that are not trapped by **`if error`** statements. This has two benefits: 1) It allows the programmer to easily eliminate all error alert dialogs. This is very important for server applications because an alert dialog requires human intervention to get the server going again. 2) It makes it easy to build a log of errors. See "Catching Program Errors (Especially for Web and other Server Applications)" on page 288 of *Formulas & Programming*.

Customizing the About Panorama Menu Item

Panorama 3.1 allows you to customize the first item in the Apple Menu. This item normally says **About Panorama...** or **About Panorama Direct...**, but you can customize it to display any text you want when your database is active, for example **About This Database...** . See "..CustomAbout" on page 397 of *Formulas & Programming*.

SuperObjectClose Statement

This statement closes (terminates editing) the Text Editor SuperObject or Word Processor object currently being edited. See "The Active SuperObject" on page 667 of *Formulas & Programming*.

Customizing the Open File Dialog and Save File Dialog

Panorama has two statements that display a dialog for selecting a folder and file name: **`openfiledialog`** and **`savefiledialog`**. Panorama 3.1 extends this capability by allowing you to customize these dialogs (note — these dialogs cannot be customized on Windows PC systems). The customization options you have include changing the layout of the dialog, adding extra text to the dialog and adding extra push buttons to the dialog. See "Customizing the Standard File Dialogs" on page 405 of *Formulas & Programming*.

Loading/Saving Multiple Variables

Panorama 3.1 has the ability to combine multiple variables into a single array, or to take an array and split it into many separate variables. This capability can be useful for editing arrays (each array element can be edited in a separate variable) and for saving a collection of variables in a single disk file (for example to store preferences). See "Copying Between Multiple Variables and an Array" on page 595 of *Formulas & Programming*.

Version 3.0

This major version was released in January 1996. Major new features include Client / Server multiuser operation, SuperObjects, built in Word Processing, Elastic Forms and AppleScript support. If you are upgrading from Panorama 2.x we've summarized these changes here with links to the new documentation.

Client/Server

Panorama 3 introduces a whole new dimension in client / server database management. Instead of a "dumb" client that simply displays forms and allows data to be edited, our **Partner/Server™** system combines the best of Panorama's incredibly fast single user RAM based database technology with an industry standard SQL server for co-ordinating data sharing across multiple computers. (This feature requires an optional SQL server, sold separately. The Client / Server system is documented in the separate **Panorama Partner/Server Handbook**, which is included with your optional SQL server software.)

SuperObjects™

Our new SuperObject technology allows you to rapidly develop even the most complicated forms with prebuilt elements like 3D buttons and checkboxes, pop-up menus, scrolling lists, matrices, scalable text, scalable / scrollable pictures, hypertext, text editing and word processing. Any SuperObject can be automatically linked to any field or variable, and most have dozens of options for controlling the appearance and operation of the object. Now you can create virtually any user interface you want. See "SuperObjects" on page 501 of the *Panorama Handbook*.

Word Processing

Panorama 3 includes full built in word processing, with multiple fonts / styles / sizes in a single data cell, four tab stop styles, over 16 different text styles, multiple colors, discontinuous and rectangular selections, and mail merge. See "Word Processor SuperObject" on page 673 of the *Panorama Handbook*.

Graphics/Forms

Improvements include the customizable tool palette (see "Customizing the Tool Palette" on page 497 of the *Panorama Handbook*) and the Graphic Control Strip with instant display of object specifications and pop-up menus for patterns, colors, fonts, sizes, and more (see "The Graphic Control Strip" on page 505 of the *Panorama Handbook*).

Elastic Forms

Panorama 3 can automatically resize and rearrange the elements of a form as the window containing the form is resized or zoomed. The form adapts automatically to different window sizes. See "Elastic Forms" on page 922 of the *Panorama Handbook*.

Reports

Reports can now include text, word processing, or pictures that are longer than a single page, selectively printing different pages of a multi-page form (see "Printing Data that Overflows a Page" on page 1116 of the *Panorama Handbook*).

Duplicates

The new **Select Duplicates** command makes it easy to find and eliminate duplicate records (see "Select Duplicates" on page 362 of the *Panorama Handbook*).

AppleScript

Now Panorama can work with and exchange data with other applications automatically. AppleScript can access and modify Panorama windows, fields and variables, and can launch Panorama procedures (macros). Panorama procedures can also launch an AppleScript as part of their operation.

Programming Language

Panorama's programming language has been vastly upgraded with over 100 new statements and over 100 new functions. New features include permanent variables (see "Long Life Variables" on page 249 of *Formulas & Programming*), improved loop control (see "Stopping a Loop in the Middle" on page 260 and "Restarting a Loop in the Middle" on page 261), parameters to subroutines (see "Passing Values to a Subroutine (Parameters)" on page 263 and "Passing Values Back From a Procedure" on page 264), pause / resume (see "Custom Dialogs" on page 489), direct form manipulation (see "Programming Graphic Objects on the Fly" on

page 633), and more access to the Macintosh toolbox (see "System and Database Information Functions" on page 175). There are also new data types for arrays (see "Text Arrays" on page 93), superdates (see "Super-Dates (combined date and time)" on page 118), graphic elements including points (see "Points" on page 147), rectangles (see "Rectangles" on page 149), colors (see "Colors" on page 154) and more.

Development Tools

Panorama 3 includes a built-in interactive debugger (see "The Panorama Interactive Debugger" on page 315 of *Formulas & Programming*), complete cross-reference tools ("Cross Referencing" on page 349 of *Formulas & Programming*), and improved procedure editing.

Import/Export

Data can be rearranged and processed on the fly as it is being imported (see "Importing a Text File into an Existing Database" on page 409 of *Formulas & Programming*), and database columns can be matched by field name instead of position (see "Appending One Database to Another" on page 78 of *Formulas & Programming*).

Security

Panorama 3's built-in, flexible security system safeguards your data automatically but does not interfere with authorized database use. (The security system is documented in the **Panorama Security Handbook**, sold separately.)

Version 2.1

This version was released in December 1992. Features included the ability to use up to 256 colors in a form (see "Color" on page 526 of the *Panorama Handbook*), QuickTime support (see "Displaying Movies in a Form" on page 819 of the *Panorama Handbook*), Balloon Help (see "Balloon Help" on page 985 of the *Panorama Handbook*), and Custom paper sizes including DayRunner / DayTime organizer notebook pages (see "Special Paper Options" on page 1166 of the *Panorama Handbook*).

Version 2.0

This major version was released in April 1991. Over 450 new features were added, including pop-up data entry window (see "The Input Box" on page 272 of the *Panorama Handbook*), Smart Dates™ (see "Entering Dates" on page 255 of the *Panorama Handbook*), view-as-list forms (see "View-As-List Forms" on page 899 of the *Panorama Handbook*), instant labels (see "The QuickLabel Dialog" on page 1169 of the *Panorama Handbook*) and reports (see "The QuickReport Dialog" on page 1111 of the *Panorama Handbook*), auto-save (see "Auto-Save" on page 65 of the *Panorama Handbook*), and improved programmability.

Version 1.0, 1.1 and 1.5

Panorama 1.0 was released in November 1988. Frankly, we don't remember much else about those versions!

General Corporate History

Founded in 1978, ProVUE has been developing interactive productivity tools for over two decades. From the beginning, ProVUE products have been known for their innovation and performance. Since 1978 over 100,000 customers have used ProVUE products on the CP/M, Alpha Micro, PC, and Macintosh platforms. ProVUE was one of the first commercial developers of Macintosh software, shipping our first Mac program, OverVUE, in August of 1984 (only 8 months after the debut of the original Mac 128).

PolyVUE

In 1978 ProVUE (then called Micro Concepts) was founded to market PolyVUE, a text editor for the CP/M operating system. That same year we attended our first trade show, the West Coast Computer Faire.

SuperVUE and DataVUE

In 1980 ProVUE introduced SuperVUE, a WYSIWIG word processor for Alpha Micro minicomputers. This was one of the first word processors on any machine to include “what you see is what you get” on screen formatting, and quickly became the leading word processor in the Alpha Micro market. A PC version of SuperVUE was released in 1986. In 1983 ProVUE introduced DataVUE, a RAM based database for Alpha Micro minicomputers.

OverVUE

In August of 1984 ProVUE introduced OverVUE, a RAM based database program for the Macintosh. OverVUE pioneered many of the unique features that later made Panorama unique, including ultra fast sorting, selecting, calculating and import, Clairvoyance™, macros, and charts. In 1985 MacUser magazine awarded OverVUE the very first “Eddy” award for best database. In January of 1985 we exhibited at the very first MacWorld Expo (in Brooks Hall in San Francisco) and we’ve been at almost every MacWorld Expo ever since.

Panorama

In 1986 work was begun on an all new database designed specifically for the Macintosh. The result, Panorama, was first released in November 1988. Panorama retained the incredible RAM based speed of OverVUE while adding a full graphical multi-window interface, MacDraw like forms, Flash Art™, mail-merge, outlining, crosstabs and lookups. The result? In 1988 MacUser magazine awarded Panorama an “Eddy” award for best new database.

Power Team

In 1993 ProVUE introduced Power Team, a complete organizer that combines seven modules into one integrated package: Phone Book, Calendar/To-Do List, Correspondence, Checkbook, Calculator, Expense Report, and Mailing List. All seven modules are designed from the ground up to work together and share data seamlessly. MacWorld magazine calls it a “Well thought out, easy to use package” while MacUser says that Power Team “Breaks new ground in the PIM and contact management arena.”

SurfScout

In the summer of 1997 ProVUE shipped our first Internet product, SurfScout. SurfScout makes web bookmarks manageable by putting an ultra-fast searchable bookmark database right at your fingertips! SurfScout also imports bookmarks from the web, and imports data tables too!

SiteWarrior

In the fall of 1997 ProVUE started a new HTML industrial revolution with the introduction of SiteWarrior. Instead of concentrating on flashy WYSIWYG features for newbies, SiteWarrior is designed to maximize the productivity of HTML experts. SiteWarrior combines database technology with web authoring to turn the craft of web site creation on its head.

iPod Organizer

In 2002 ProVUE made information mobile with the Panorama iPod Organizer. This package combines the power of Panorama's unique RAM based database technology with the portability and flexibility of the iPod. It allows you to store phone numbers, email addresses, flight numbers, appointment times ... all the important information you need to access on the go, then easily transfer the information to your iPod for instant access at home, at work or in the field.

Panorama Enterprise Server

In 2007 the Panorama Enterprise Server debuted, bringing Panorama’s amazing RAM based technology to local networks and even the Internet.