onViz Turn on your vision!

onViz Authoring System – Features

Revised July 27, 2001

System requirements:

Authoring:

Apple Macintosh OS X Native application
Apple Macintosh OS 8.6 or above required for authoring
QuickTime v4 or above
8 MB free memory for basic application (optimal memory is based on the media assets incorporated – sound, graphics, digital video, etc.)

A Microsoft Windows compatible authoring environment will be released soon.

All new features being are presently being designed with a cross-platform authoring environment in mind to speed development of the MS-Windows based authoring system.

Runtime:
Macintosh:

Apple Macintosh OS 8.6 or above
Memory dependent on the media assets incorporated

Runtime:
Windows:

Win 95, 98, ME, NT, 2000
Note that onViz allows you to verify your users operating system and installed components (such as QuickTime, sound card, Direct X) alerting the user to any missing component that is critical to your application.

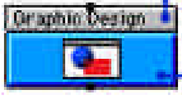
Discovery Systems International, Inc.
P.O. Box 1188 Knoxville, Tennessee 37901-1188
e-mail: info@discoverysystems.com
http://discoverysystems.com

Toll Free: 888.284.5389
Tel: +1.865.637.0311
Fax: +1.865.637.0312

How onViz Works

Nomenclature

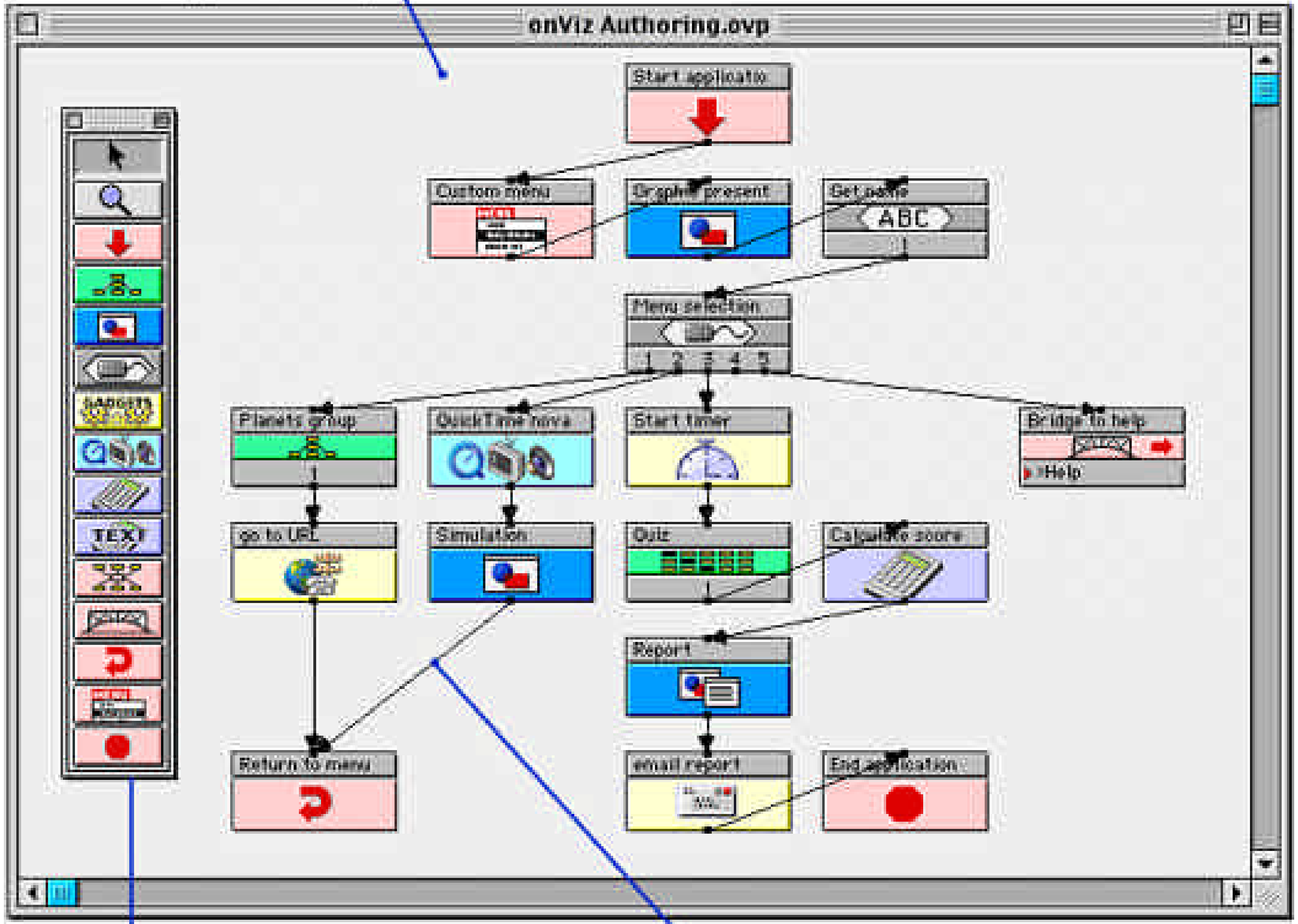
Program blocks are called "States"
States are simply pre-programmed functions displayed graphically.
For example, here is a Graphic Design Output State.



InfoCenter:
Click on the top part of the State to set up the functionality for a particular State

Presentation:
Click on the bottom part of the State to define the appearance and multimedia elements of your program

Application Map

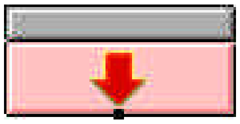


Floating Menu Bar
This is where you choose the States you want to include in your program by dragging them from this menu onto the Application Map

Path
Paths connect the States and determine the sequence in which your program will run (application flow)
These can be conditional, giving your application intelligence.

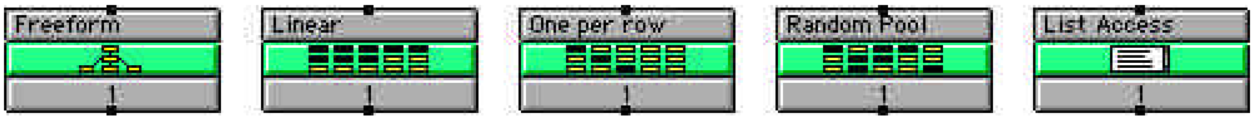
Authoring elements in onViz are called **STATES**. Each state executes specific pre-programmed functions:

Start State



The Start State defines the beginning of your program. The Start State is one of six in the larger category of states, called Flow States. Flow States direct the execution path of your application, and are color-coded pink.

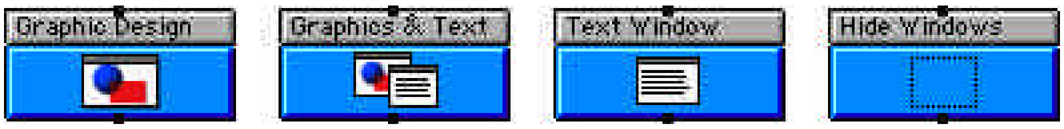
Group States



Group States are color-coded green. Group States organize your program elements into subsections; onViz provides five types of Group States:

- **Freeform** – where the author chooses the sequence
- **Linear** – where your program progresses through each group in a linear manner
- **One selection per row** – where only one choice is valid
- **Random pool** – where your program randomly chooses the output
- **List Access** – where the user is presented with a list of choices.

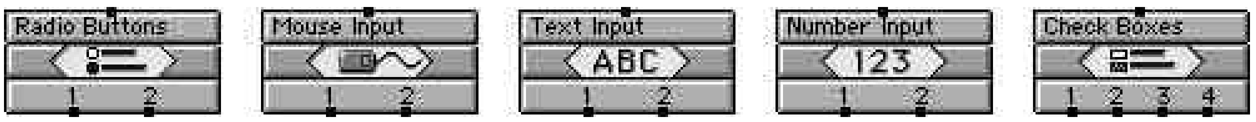
Output States



Output States are color-coded blue Output States provide the environment to add multimedia components, such as text, graphics, animation, digital video, and sound. onViz provides four types of Output States:

- **Graphic Design** – This is the basic environment for adding your multimedia elements
- **Graphics & Text** – Enables You to add a scrolling window with your graphics and text
- **Text Window** – Enable text functions only
- **Hide Windows** – Removes all onViz windows from the user's screen during runtime. For example, you could add this element to remove any onViz-generated windows from a user's screen for a set amount of time, and then restore the screen view.

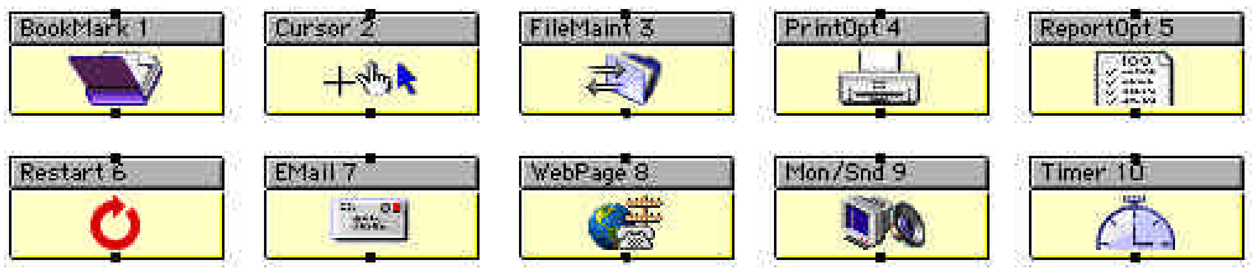
Input States



Input States are color-coded gray. Input States solicit mouse, voice, touchscreen, or keyboard input and contain features that evaluate user input, create scores based on input, and route program flow based on user response. Each State supports up to ten separate inputs. For example, the check box input State graphic shown above indicates four check boxes are enabled. onViz provides five types of Input States:

- Radio Button Input – User selects a radio button
- Mouse Input – You define a “hot area” for user selection. Any area on the screen or animation sprite can be defined as a button using this State. You can also create programs that react based on where a user places their mouse cursor (or finger in the case of touchscreens)
- Text Input – User is prompted to input text and/or numbers
- Numeric Input – User is prompted to enter numbers only
- Check Box Input – User selects check boxes

Gadget State

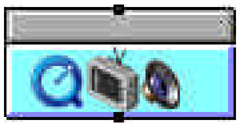


Gadget States are color-coded light yellow. Gadget States provide sophisticated control of program functions. onViz provides ten types of gadget states:

- **Bookmarking** – With bookmarking enabled, a very small application file is created to a specified directory. If you author an application with bookmarking enabled, users can choose to save their place anywhere in an application, and go back to that point later. An unlimited number of bookmarks can be set by users. You can define the default file location for saved bookmarks, auto-delete bookmark when application is complete, and whether users can edit the bookmark name.
- **Cursor Control** – Allows you to define cursor appearance from a library. You can even create your own cursor libraries, and set the operating system default cursor while your application is active.
- **File Maintenance** – You can define a drive location from which to retrieve other files used by their program. Local drives or remote servers can be specified as well as FTP access only, whether to automatically cancel downloads already in progress, and whether to automatically delete all files in a specified download directory. You can also define an error path for your program to follow if the remote file is not found, so your program can display an error message, continue your program in a different direction, or halt your program.

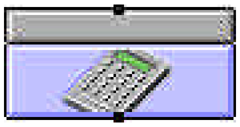
- **Printer Control** – You can define a default printer setup for their program, setting page orientation, print scale, number of copies, and whether to show a page setup or print dialog prior to printing. This can be useful for developing kiosk applications where the kiosk automatically prints out something at a specified point in your program. You can also define an error path for your program to follow if a printer is not found or if there is a printer malfunction, so your program can display an error message, continue your program in a different direction, or halt your program.
- **Report Options** – You can define whether your program will automatically save a report on exit, what data the report will contain, sort data sequentially or alphabetically, whether the format is based on user name, sequential, or date and time, and periodically save or clear a report at a preset point in your program. You can also define an error path for your program to follow if a report is not found, so your program can display an error message, continue your program in a different direction, or halt your program.
- **Restart Application** – You can define specific points in their program to restart, restart the application if idle for a specified period of time, and define a “hot key” that restarts your program. This function is useful to kiosk application developers, where the application can automatically restart in case of a power outage or if the application is inactive for a period of time.
- **Send e-mail** – Automatically launches the user’s specified e-mail client program. You can define when to automatically send e-mail, with complete control over the name, e-mail address, subject heading, message, SMTP server used, optional from name, and optional from e-mail address. You can also define an error path for your program to follow if an e-mail program is not found or malfunctions, so your program can display an error message, continue your program in a different direction, or halt your program.
- **Show Web Page** – Automatically launches the user’s specified Web browser and goes to a specified URL. You can also specify whether to automatically return to your onViz application after a preset period of time, and whether to automatically quit the user’s browser on return. You can also define an error path for your program to follow if a Web browser program is not found or malfunctions, so your program can display an error message, continue your program in a different direction, or halt your program.
- **Monitor and Sound control** – Automatically changes the user’s monitor color depth, change the Mat color (the color of the screen area not being covered by your application), and automatically adjust the sound volume. You can also define an error path for your program to follow if the user’s hardware malfunctions or if there is a device conflict, so your program can display an error message, continue your program in a different direction, or halt your program.
- **Timer control** – You define Start,, Pause, Resume, or Stop timer functions with complete control over the duration in hours, minutes, seconds, whether to count up or cont down, sound attributes on timer expiration, sound attributes on timer warning, at which point to initialize timer warning. You can also define whether to go to another program state, open another text file, application or other document on timer expiration, timer warning, or both. Font type, size, and style can be specified. Window type and color can be specified by document window, dialog window, plain rectangle, 3D window, or borderless window with complete control over where the timer window is displayed onscreen and even whether its position changes for different screen layouts.

Multimedia State



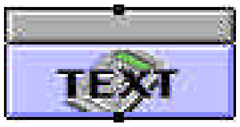
Multimedia States (there is only one at this time) are color-coded turquoise. The Multimedia State is where you define the parameters for digital video, audio, and overlay images. For example if you want to play a video or sound file within a single frame of animation, you can link to it using this State, so if a user clicks on a predefined area, a sound, movie, graphic image, or all of these activates. Future versions of onViz will interface with external hardware such as DVD-ROMs using this State.

Calculator State



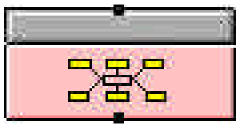
Calculator States are color-coded light purple. The Calculator State assigns values for numeric variables. This state is invaluable in your application to control sprite characteristics in Output States, scoring statistics for Input States, and a comprehensive range of general variables and high level math functions you can implement to add sophisticated features to your program.

Text Calculator State



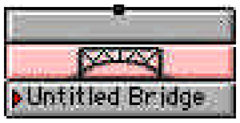
The Text Calculator State assigns values to text variables in the same manner as the Calculator State, except where the Calculator State is used for numeric variables, this state is used to assign text variables.

Junction State



The Junction State enables program paths to converge at one place so branching decisions in your program can be made. The Junction State is one of six in the larger category of states, called Flow States. Flow States direct the execution path of your application, and are color-coded pink.

Bridge State



The Bridge State accesses a Bridge Library entry – a listing of targets you define that your application can connect with. Using the Bridge Library feature, you can move or jump your application's flow to any other State within your program, within other onViz documents and programs, or launch virtually any other external file or application from your program. You can also define an error path for your program to follow if the user's hardware malfunctions or if there is a device conflict, so your program can display an error message, continue your program in a different direction, or halt your program.

Return State

The Return State is used in combination with a Bridge State in targeting a Bridge Library entry set with a Return. This State will return the project's activity flow to the Exit of the most recent Bridge State.

Custom Menu State

The Custom Menu State creates a custom menu that appears on your project's menu bar.

Stop State

The Stop State ends your application when reached in the top level Application Map, and exits a group when used within a Group State.

Notable onViz Features

- Conditional routing (branching) supports up to 64 different conditions on a single route - this allows creation of sophisticated programs such as "expert systems" that also may include multimedia elements. It also simplifies the task of creating non-linear multimedia, sophisticated courseware, e-learning programs, and games
- Top-level applications can be saved in Macintosh or Windows-compatible format. Supporting documents are common to all top-level applications (i.e. you do not need separate supporting documents such as text, graphics, video and sound for Macintosh and Windows computers)
- You only need one application to author for both Macintosh and Windows computers
- File maintenance. This is the onViz interface to a specified server or website. Applications can be designed where a user downloads only the starting application that will thereafter retrieve components or modules as they are needed. Variable substitution is supported so files to be downloaded or deleted can be dynamically modified during runtime.
- Enhanced support for playing applications from a server and new support for integration with the Internet. This product provides a hybrid solution in that a starting application can be downloaded to an individual's local computer with the application automatically downloading additional components or modules from the server or Inter/Intranet as needed.

  Programs you create with onViz can access media assets (sound, pictures, text, digital video) from specified directories on hard disk, CD-ROM, DVD-ROM, streamed from a website, Local Area Network (LAN), Peer to Peer network, or a combination of these.

  This is useful for delivering a small application over low bandwidth network connections and having centralized file directories where other programs might access the same files simultaneously through the same network.

  For example, you could provide a CD-ROM with graphics, video, and sound to your users, then deliver a small core application you create over the Internet. This core application would prompt the user to insert their CD-ROM when they start the program. The core application would then seamlessly integrate all of the media assets from the various sources you specify into a complete multimedia program.
- All media assets can optionally be incorporated into a single self-contained file (.exe on Windows, application on Macintosh). This is useful if you are distributing your application on CD-ROM, DVD-ROM, floppy disk, or want to protect the media assets by embedding them in an application
- The ability to launch other applications or documents. This also includes a "bridge error" exit so You can set a contingency path if a file is not found
- Eight different question and answer modes:
  - Text input
  - Numeric input
  - True or false
  - Multiple choice input with one answer correct

  - Multiple choice input with several answers correct
  - Correct sequence. Verifies if the question is answered in the correct sequence
  - Draggable sprites. Drag and drop a text or graphic object into a "hot" area to determine the correct answer.
  - Essay or statement. Scans for exact words, or key phrases and words in an answer
- Bookmark Control. Users of your application can bookmark their place and later go back to where they left off
- Report options include choosing when a report is saved and what information you want it to contain, an option to clear a saved report but not statistics about application's use such as input answers or time spent in the application, and a 'save milestone' feature that saves a text file containing report data up to a specific point in an application
- Restart Application. This is useful in kiosk applications where idle time or hardware failure (such as a power outage) restarts your application at a specific point
- Automatically adjust monitor color depth and resolution as well as automatically adjust the volume settings on a user's computer
- Animation features include controlling sprites visibility, controlling the visibility of a selected color, mouse event and cursor control for media assets (external file access, text, images, sound, and video), sprite image interchangeability, and preset variables
- SendMail support. Send an e-mail from any point in your application and designate an e-mail subject and message it will contain. Includes an option to attach a report
- Show Web Page. Open a currently installed browser on a user's computer and go to the specified Web address. You can also designate a timeout option to quit the browser and return to your application
- Playback and access of files from the web or a server does not require a web Browser
- Extensive variable control includes computer type, timers, support and report directories, sound, video, graphics, browser idle time, restart time, available disk space, active web browser, test for web connection, file transfer, errors, and last input state
- Sound support for cross-platform file types - .wav, aif. Your applications can use common sounds – no separate sound files for Mac and Windows are required.
- All file paths can be up to 255 characters

Details of onViz Features

Authoring/Appearance

- Auto centering for the screen display on monitors larger than the display used to author the application (or was set as the 'target screen' in the Project Attributes). The area around the display area can be blanked with a user selectable mat color of the author’s choice.
- Author and Runtime in any color depth.
- Interfaces and dialogs are optimized for authoring using one monitor. Many authoring programs have so many interfaces and dialogs visible during authoring, this either requires You to purchase another monitor just for displaying those interfaces, or be left with having to constantly move them out of the way to see the work area. By optimizing interface and dialog layouts, workflow is much more efficient and rapid.

- Variable substitution is supported in nearly all edit fields.
  By supporting variables in fields such as Report Storage Location, Subject line for Report File e-mails, and e-mail addresses, this provides the ability for an application to be updated based on a user's input
- All command keys are consistent with standard Macintosh applications.
- Option to dismiss certain alert messages and 'not show again', such as the warning that the Application Map does not have an 'undo' for 'clearing' components. Alerts can be turned back on in the Authoring Preferences dialog.

Multimedia Components

- Support for Media Folders (i.e., separate supporting files into individual sound, QuickTime, supporting docs, etc. directories) Media asset locations can be overridden for individual multimedia components to support storage and retrieval from CD-ROM, file server, Internet etc.
- Support for accessing media assets (sounds, pictures, text, and movies) from files on floppy disk, local hard drive, CD-ROM, file server, or streamed from a web site. This allows displays and information presented in the application to be changed by simply changing the information in the source directory.
- Apple's QuickTime is an integrated feature with onViz. With this integration, it is now easier to align movies for playback on any individual frame of an animation sequence, as well as control visibility of the QuickTime control interface, and take advantage of using the QuickTime "skins" feature (see Apple's QuickTime website for more information at http://www.apple.com/quicktime/.
- Support for MPEG and QTVR movies as well as movies with interactive features such as those created with Electrifier Pro
- Support for importing .pct & .jpg graphic formats.
- Image conversion - can convert the image's color depth, resolution, and from any other supported format to PICT. Also can convert any PICT image to JPEG.
- Convert sounds including format (.snd, .wav, .aif), sample rate, stereo/mono

Application Map

- Application Map (the main work area during authoring) supports click and drag from the floating tool palette to the Application Map. Holding down the Control key while clicking reveals the state type options for that tool set.
- Routes will auto-connect to a state's entrance by dragging and releasing a route anywhere on the state icon.
- All states (except Inputs) have an 'automatic return' option. This eliminates the need to have a separate 'Return State' on the Application Map. Map Icon is updated if this option has been selected
- States can be named by simply clicking on the 'info center' - no need to open dialog for naming
- If onViz detects a programming error during authoring, an error dialog is displayed, giving the author a couple of options. When 'show me' is selected from 'dead end route reached' error dialog or stop and show is selected from the run menu, the last executed state is centered in the display

Libraries

onViz draws heavily on the use of libraries. Application components are set up in a library with entries referenced throughout the application via consistent pop-up menus. Libraries include Bridge Targets, Cursor, Custom Variables, Fonts, Images (which are shown visually on a palette), Sounds, and QuickTime Movies.

An important benefit to libraries is that any changes to the library entry are reflected every place the library item has been used in the application. For example, if you've assigned a sound for a button and decide to change the sound played, this change can be accomplished by simply changing the sound referenced in the library.

Also, libraries make it easier to create and organize media components and specify how they will play within the application.

Bridge Target Library

- Set up bridges to States within the application, another application and/or its associated document, and a specific state in another document.
- When bridging to a state within the same document, a list of named states is provided. A filter allows the author to select the state types to be displayed in the list. This list can be sorted alphabetically or according to state type.
- When bridging to another application or document, authors can define separate Macintosh and Windows specific file search paths. A ‘Select’ button displays a ‘get file’ dialog to select the application that will be bridged to.

Cursor Library

Up to 24 different cursors can be designed for use within your application. This dialog contains four standard cursors that cannot be changed (showing appearance for both standard Macintosh and Windows). Other cursors are available visually from a palette and can be edited as desired. There are also empty palette spaces available to create and add your own cursor.

Custom Variable Library

- Numeric variables can be set for a variety of display types - including automatic conversion to standard date, time, and currency formats. For example - use a custom variable to convert application time, usually displayed in seconds, to be displayed in hours and minutes. Includes an option for support of European standard number display.
- Added ‘Random’ number type. Once a custom var is selected as random, edit fields are presented allowing author to set a range for the random numbers (i.e., give me a number between 5 and 20)
- An import button supports import of custom variables from another onViz document.

Font Library

- The Font Library allows authors to set up fonts to be used in the application and identify the corresponding font displayed during Windows runtime. An alternate library selection can be designated in the event that a specific font is missing from the runtime machine.
- Fonts attached to graphics and text being imported will be flagged with a user alert if not already in the font library

Image Library

- All graphics used in a application are created and edited in the image editor and stored in the Image Library. This provides the ability to reuse graphics in your application without the increased file size caused by storing multiple copies of the same image.

Also, changing the image in one place updates the image throughout your application. These images are accessed in the output window, sprite library, and new multimedia state via pop-up menus.

Movie Library

- All digital video movies used within an application are set up for access in a movie library. Movies can be accessed in the output window and new multimedia state via pop-up menus. Movie clips are shown visually in the Movie Library palette.
- The Movie Attributes dialog provides options to control how the movie will play in the application, including making the movie auto start, looping, specific frames/tracks to play or start from, movie window border, movie controls visibility, and positioning on the display.
- Movies can either be stored in the application or accessed from a file on disk or a server. Files can be stored in the 'Movie Support' folder or, if the movie will be coming from a CD-ROM or website. Individual movies can be accessed from a different location than the Movie Support folder by overriding the support path for each image.
- Storing movies in the application does two things - it allows applications to be distributed as single, self-contained files, and, protects media assets by not providing access to movie files in the support document. Of course, the trade-off is a much larger application

Sound Library

- All sounds used within an application are set up for access in a sound library. Double clicking on a particular sound in the library plays a preview of that sound. Sounds can be accessed in the output window and new multimedia state via pop-up menus.
- Support for snd, wav, aiff, and phonetic speech. Sounds saved in QuickTime or mp3 format are considered movies and stored in the movie library – this way you can take advantage of QuickTime's streaming features when delivering your application over a network.
- Sounds can be accessed by the application in two different ways - loaded into memory before the sound starts or buffered from the disk or server. Loading the sound into memory takes additional time for the sound to completely load and start, and requires the application be allocated enough memory to load sound and graphics. Buffering the sound brings the sound into the application in 'segments' to fill the sound buffer with the next sound segment automatically loaded while the current segment is playing - similar to audio streaming. The recommended method for adding sounds is as small files, loaded in memory; large sound files are buffered from disk or server.
- Option to loop sound clip for background music with continuous play until a 'Silence All Sounds' is encountered in the application.
- As with images and movies, sounds can be resident in the application or accessed from a disk file. If resident in the application, an option is provided to 'export' the sound to a sound file.
- Phonetic Speech can come from an external text file or resident in the application. By making it resident, you can type in the text to speak in the sound library entry as opposed to having to set it up in a text output.
- A pop-up menu allows selection of installed MacinTalk voices with edit fields allowing modification of rate and pitch. Phonetic Speech only functions on Macintosh computers using MacinTalk, and is not presently supported on the Windows operating system.

Sound Conversion Dialog

- New Feature - snd, wav, and aiff files can be converted to any other format using the ‘convert’ option.
- Additional options include changing from stereo to mono, decreasing the sample rate and file size.
- Note that if a sound is resident in the application, this conversion will permanently alter the sound - a warning dialog will alert the author and give a prompt to export the sound to a file first.

Image Editor

The onViz image editor has been designed to make creation of graphics easy. Features include:

- Full support of 16 and 24-bit color during both authoring and playback
- Displays the target screen boundary.
- Variable substitution is supported in all graphics – including animation sprites.
- Support for Super & Sub-script fonts
- Support for many option/command key combinations during graphic editing.
- Graphics can be nudged larger/smaller by clicking a handle and using the arrow keys.
- Resizing of graphics visually is constrained to relative height and width by holding down the ‘Shift’ key during click and dragging to resize an image or animation sprite.
- Text can be created with a single click. Auto width will be determined by 100 pixels from right edge of target frame window. To create a text object with a specific width, click and drag to define its limits.
- Any PICT graphic can be imported and edited in the graphics editor. Other images can be edited, but will be converted to PICT format when edited. A warning dialog will be presented when trying to edit these formats.
- Controls at the bottom of the graphics editor window include an edit field to name or change the image library entry name, a checkbox to designate whether the image name appears in the pop-up menu, a pop-up menu to create duplicate current image or create a new image, selecting a different image to edit.
- An Output can be selected from a popup menu to display behind the image being edited. The foreground/background images will be dimmed showing they are display only.

Conditional Routes

- The interface provides flexibility in combining conditions (AND). To combine conditions, they must be precede or follow the condition it will be grouped with. Select a condition and then select the ‘group with’ previous or next.
- Support for up to 64 different conditions in a single route.

Calculators

- The interface reveals only the most commonly used functions to reduce confusion for new users. A clickable reveal triangle expands the interface to display special use functions.

- Each calculator can contain up to 32 calculations. Calculations are performed in order they occur in calculator so you can sequence events - i.e., you can have a delay of 5 seconds (Calc 1), then show a different sprite pose (Calc 2), with another delay of 5 seconds (Calc 3), and return sprite pose to original (Calc 4).

Bridge State

- All bridge references are set up in the Bridge Target Library and selected from a pop-up menu. This gives greater flexibility in that once a bridge has been set up in the library and referenced in the application, it can be quickly updated or modified in the library with changes being made globally.
- Added a 'Bridge Error' exit on Bridge States so author can set a path for bridge failure.

Multimedia State

The multimedia state is used to start media components at any point in the application. You can launch a sound, play a QuickTime movie, start a video clip, or overlay a specific image (replaces the 'Overlay Output' option). Any media component can be set to loop or continue until some stop action terminates the component. i.e., start a background sound with the Multimedia State and let it continue in the application until a sound action in either an output or another mm state tells the sound to quit.

You can sequence up to 8 different media events. The multimedia state executes the events in the order they appear in the list. Arrows in the upper right of the dialog move the selected clip up or down in the sequence list.

Gadget State

This state provides flexibility to add custom features to onViz. Your application can start with a series of Gadget states defining behaviors for the application with behaviors modified during runtime by passing though another Gadget state. Most Gadgets include a bridge target in case of error.

Bookmark Control

Running a application through a bookmark control state can either automatically save a bookmark file or, allow the user to open a previously saved bookmark file.

Cursor Display

Use this Gadget to Show, Hide, or change the appearance of the cursor.

File Maintenance

- This is OnViz's interface to a server or web site. Using this feature, an application can be designed where a user downloads only the starting application that will thereafter retrieve components and modules as it needs them. All edit fields on this dialog support variable substitution so files to download and deleted can be dynamically modified during application runtime.

Print Options

- This dialog automatically sets up the Page Setup dialog so the user does not have to set these with each pass through an output state set to print.

Report Options

- This dialog gives alternatives for when a report is saved and the content it contains.

- A new option allows clearing of all report data - this option clears the saved report data, but not statistics about the application; such as input answers, application time, etc.
- The 'save milestone' saves a text file containing report data up to the point that this state was encountered in the application.

Restart Application

- Excellent for use in a kiosk application, this control state can restart the application based on application flow through this control, idle time or a hot key. Once idle time has been set with a control state, the idle time can be changed with a General Variable.

Send Email

- Send an email message from any point in the application. All fields support user variables to be updated dynamically during the runtime of the application.
- Designate an email subject and specific message the email will contain. Includes an option to append the current application report.

Show Web Page

- A simple dialog which will open the currently installed browser and go to the specified web address. Return options include designating timeout with the option to reset the timer if the application detects browser activity. An additional option to quit the browser application when control returns to the application.
- Includes an 'on error' bridge target in the event that Internet connection was not made.

Monitor and Sound

- This control will automatically adjust the monitor Resolution and Colors as well as changing the Mat Color selected in the Project Attributes.
- Includes a simple dialog that adjusts the sound volume of the computer. Once set, the volume can also be changed with a General Variable.
- When the application exits, the color, resolution and volume are reset to settings when the application began.

Timer

- Passing the application through a timer will start, pause, resume or stop a digital timer. An additional option allows changing timer expiration action or display attributes without affecting the timer.
- The timer can be set to count up or down and be visible or invisible. If set to be visible, you can select a display font and size as well as a window border and color. A 'sample' display shows how the timer will appear on the screen. Once the timer has been designed, it can be positioned on a selected output.
- The timer will persist until expiration at which time a selected sound can play and the application follow a specified bridge target.
- A new General variable can read the time elapsed and allow resetting the time on the current timer

Menu States

- Menu states are defined by using bridge library targets. Menu items can be disabled (shown, but dimmed and not selectable), checked, or designated as a separator line.

- Options to 'clone' the Edit and File menus allow for multi-lingual applications.

Input States

- Redesigned Info dialog combining runtime behaviors with scoring setup. Runtime features are updated based on input type.
- From this dialog, you can select an output state on which to position the input dialog. Once selected, the output is displayed allowing positioning of the dialog without having to open the lower 2/3rds of the state. This feature is particularly useful with mouse inputs where the entire display is shown along with the mouse hot spots for positioning.
- New runtime features for Radio Button, Text, and Numeric Inputs –
  Tab activates OK button – Tab, Enter, and Return will all dismiss input dialog.
  Previous Output Active – smart sprites on the output that the input dialog overlays remain active.
- A feature for Radio Button Input to include Default first radio button. With this option turned off, the input will be presented with no radio buttons preselected (not within Apple's user interface guidelines, but helpful in designing test questions where you don't want to have a preselected option).
- Questions and potential answers are entered on the 2nd and 3rd tabs. Answers designated 'correct' on the 1st tab are reflected on this tab with a checkbox to the right of each answer field and can be changed on this tab.

Output States

Info Center

- Combines attributes for both Design and Text runtime features and attributes in a single dialog.
- Option to save text in Text Output window to a text file when Output exits. This file will be saved in the same location as report files.
- Shows report information when running authoring instead of requiring the file be saved and run as an application to collect this data.

Animation/Design environment

- Background/Foreground Images - each output window can have a background image with a different foreground layered on top. Images are selected through the Background/Foreground image pop-up menus.
- The text window can be shown in any output - it is not necessary to change output type to include a text window. This is helpful when designing an application that will contain hidden information in the text window, but will be run as a design only during runtime of the application. A new General Var has been added to 'Show/Hide' Text Window at runtime.
- The Design and Animation environment is combined with animation controls for easier use. A floating window gives control over creating animation sequences on a frame-by-frame basis. Features of the animation control top to bottom/left to right:
- Animation target frame - individual frames can be set as target frames. These target frames are used by smart sprites to jump between frames of the animation sequence.

- Frame control, designates the frame of the animation sequence is being shown. Clicking on the 'Frame x' button displays a 'go to' dialog to jump to any frame in the sequence. The up/down arrows move through the sequence a frame at a time. The 'of x' button shows the total number of frames in the sequence. Clicking this button displays a dialog to add/remove/insert/delete frames.
- Timing - timing for each frame can be assigned by simply entering a number in the timing edit field - no need to open a separate dialog.
- Frame synchronization - Wait for Movie, Wait for Smart Sprite. The wait for movie waits for completion of a QuickTime movie before continuing; the wait for smart sprite holds up on the frame until the user clicks a 'Smart Sprite' (described later in the new Sprite Palette features).
- Multimedia Components - Sound, Movies, and video clips can be started from any frame of animation. When a QuickTime movie is selected, the poster frame for that movie is displayed on the animation frame as set in the movie library. Repositioning the movie on the animation will override these settings.
- Animation cycling - OnViz allows you to set up animation cycling at any point in an animation sequence. The cycling will be from any frame which has a synchronization setting (i.e., click anywhere to continue, continue button, wait for sound, etc.) and go either forward or back a number of frames as designated in the cycling field. Animation will remain in this loop until the synchronization criteria have been met.

Sprite Attributes Palette

The sprite library includes a multitude of features.

- The palette displays thumbnails images of Sprites for a grid of 6 x 4 sprites. Sprites are selected from the image library via a pop-up menu.
- The preserve size option automatically resizes the image on all animation frames if editing the sprite graphic makes it larger or smaller. The alignment option determines which point remains constant when the sprite is resized.
- Individual sprites or sprite families (sprite with all its poses) can be copied and pasted to another output. Copying the sprite will also copy any associated actions.
- An information box provides information about how the sprite is set up including any location of the sprite on the current frame and any mouse over/down/up actions that have been assigned.
- Note that sprite click sensitive areas are constrained to the actual graphic, not the bounding rectangle. (i.e., if the sprite is a circle and mouse over actions is assigned, action will happen only when the mouse is over a part of the circle itself - not any of the showthru rectangle surrounding the circle.)
- Click and drag sprites from the palette to the animation frame
- Multiple sprites can be selected for alignment
- Multi-frame editing will be handled by holding the control key to display a contextual menu when clicking a frame control, removing, adding, or moving a sprite. This menu will allow the edit to occur over the entire animation or a set number of frames.

Sprite Actions

- Clicking the reveal triangle in the upper right of the dialog displays sprite actions that can be associated with any individual sprite. These actions include a mouse over, mouse down, and mouse up. Additionally, each sprite family can have preset variables including visibility and draggability. The presets are attributes of the entire sprite family (i.e., setting a draggable affect for every pose of the sprite).

Smart Sprites

Using pull down menus, actions can be easily set for a mouse over/down/up event.

- Actions - any mouse event can trigger the application to go to the next/previous frame, go a specified number of frames forward/backward, or go to the next/last target frame or exit the output.
- Images -both mouse over and mouse down events can be set to replace the sprite displayed with another image from the image library (the replacing graphic does not have to be in the sprite palette). When this option is selected, a thumbnail of the image is displayed along with an alignment selection to designate the point the replacing graphic should use as its anchor point. Note that there is no image replacement on a 'mouse up' as this would remove the smart sprite from the frame and presumably there would be some sort of application action on a mouse up.
- Sounds - any mouse event can trigger playing a sound set up in the sound library. There is also an option to stop all sound actions.
- Cursors - displayed cursor can be changed for any mouse event. Cursors available from this pop-up are the standard plus any custom cursors created in the Cursor control state.
- Preset Variables - work on sprite families (all poses of a specific sprite number) The presets can include affecting sprite visibility and draggability. Drag can be constrained to horizontal or vertical and the x/y max/min drag positions can be set in the edit fields.

Variable Support

Output Variables

- Last Output State (read/write) - This variable makes it easier to create reusable templates. Instead of requiring a specific state name, conditional routes and calculators can be set to look at or affect output variables for the last output executed.
- Sprite (n) in bay (read only) - This variable returns a number representing the mouse input bay the sprite referenced is currently in
- Sprite (n) x min (read/write) - This variable is used to constrain the horizontal movement of sprite drags. It sets how far to the left a sprite can be dragged on the screen. This variable can also be set in the sprite library palette.
- Sprite (n) y min (read/write) - This variable is used to constrain the vertical movement of sprite drags. It sets how far up a sprite can be dragged on the screen. This variable can also be set in the sprite library palette.
- Sprite (n) x max (read/write) - This variable is used to constrain the horizontal movement of sprite drags. It sets how far to the right a sprite can be dragged on the screen. This variable can also be set in the sprite library palette.
- Sprite (n) y max (read/write) - This variable is used to constrain the vertical movement of sprite drags. It sets how far down a sprite can be dragged on the screen. This variable can also be set in the sprite library palette. This variable can also be set in the sprite library palette.

- Show Text Window (read/write) - This variable will either cause the text window of an output state to be shown or hidden regardless of whether the state was set to 'design only' or 'design and text.' Var = 1, text window is shown; Var = 0, text window is hidden. Note that if the output was set to 'text only,' setting this variable to '0' will result in no output display.
- Current Frame (read/write) - This variable reads and returns the current animation frame. Setting this variable in a calculator will update to display the number entered.
- Total Frames (read only) - This variable holds the total number of frames of the last output executed.

Input Variables

- Last Input State (read/write) - Just as the 'Last Output State,' this variable makes it easier to create reusable templates. Instead of requiring a specific state name, conditional routes and calculators can be set to look at or affect input variables for the last input executed.
- Input Answer Length (read only) - This variable returns a number representing the number of characters in the last text or numeric answer.
- # checkboxes selected (read only) - This variable returns a number representing the number of checkboxes which were selected in a checkbox input.

General Variables

- Change Computer Type to simplify 1 = 68K Macintosh; 2 = PowerMacintosh; 3 = Windows 3.x; 4 = Windows 95/98; 5 = Windows NT
- Timer (read/write) - This variable contains the current application timer as set in the Timer Control State. By using a calculator to write to this variable, the application timer will be updated with the new time.
- Timer Elapsed (read only) - Contains the amount of time elapsed since a timer was started with a Timer Control State.
- Support Folder (read/write) - reads the current location of the application support folder. Writing to this variable updates where the application will look for the support folder.
- Report Folder (read/write) - reads the current location of the report folder. Writing to this variable updates where the application will store reports.
- Sound Volume (read/write) - reads the current computer volume setting. Writing to this variable with a calculator or passing the application through a sound control panel state can change the volume.
- Browser Idle Time (read/write) - reads the time in seconds that there has been no browser activity since the application passed through a Show Web Page Control State. Writing to this variable will update the browser idle time in the web control state.
- Sound Playing (read only) - returns a 1 if a sound is playing; 0 when sound is done
- Movie Playing (read only) - returns a 1 if a QuickTime movie is playing; 0 when movie is done
- Video Playing (read only) - returns a 1 if a video clip is playing; 0 when clip is done
- Sound Loops (read/write) - counts the number of times the current sound has looped. Writing to this variable will cause the current sound to loop a specified number of times.

- Movie Loops (read/write) - counts the number of times the current movie has looped. Writing to this variable will cause the current movie to loop a specified number of times.
- Restart Time (read/write) - reads the 'idle time' set in the Restart Application Control state. Writing to this variable updates the application idle time before restart.
- Available Disk Space (read only) - used with file maintenance, this variable returns the disk space available on the volume where the support folders reside.
- QuickTime Version (read only) - returns the current installed version of QuickTime
- Active Browser (read only) - returns the current active browser. 1 = Netscape; 2 = Explorer
- Test www connect (read only) - automatically launches the browser in the background (if not already running) and tests connectivity to the Internet. Returns a 1 if connection ok, 0 if OnViz encounters an error.
- File Transfer (read only) - used to check for transfer done. Returns a 1 if currently transferring files, 0 when file transfer is done.
- Error (read only) - this variable gets set if the application encounters an error in any of the application control states.

All of these features, and we've still managed to keep the interface simple!

Discovery Systems International, Inc.

"Creating solutions that enable people to transform thought into the shared digital medium"