IconAuthor Test Drive

Click logo to see Movie

January, 1996
AimTech Corporation
20 Trafalgar Square
Nashua, NH 03063
http://www.aimtech.com

Click here for Help

Click here to Continue

Table of Contents

Introduction to the Test Drive

The IconAuthor 7.0 Test Drive lets you experience, first hand, how easy it is to develop applications using IconAuthor. Once you've finished this Test Drive, you will have completed a customer service training application for employees of a fictitious bicycle manufacturer called Wild Planet Cycles. The file name of the IconAuthor application you will be working on is "EVALBIKE.IWM". We have removed approximately 10% of the functionality in EVALBIKE.IWM, and you will use IconAuthor to add the remaining functionality to complete the application.

Before You Begin

What do you have to do before you begin working with the IconAuthor 7.0 Test Drive?

First, before you must install the IconAuthor 7.0 Evaluation CD. See the Installation Instructions card that came with your CD or the README.WRI file on the CD for information on the installation procedure. During installation, make sure that you chose the following components: Authoring System, MCI Drivers, ODBC Driver, and Test Drive. Otherwise, the Test Drive will not run properly. The IconAuthor Evaluation CD must remain in the drive when working with the Test Drive since all of the audio files are stored on the CD-ROM.

Second, this tutorial assumes you are familiar with your windowing environment. You need to know about conventions such as pull-down menus and mouse techniques. If you are new to your windowing environment, be sure to spend some time reviewing the documentation that accompanies it.

Third, your system <u>must</u> be running in 640 x 480 screen resolution when you work on the Test Drive application. Because we have partially created some files for you on a system running in 640 x 480 resolution, you need to finish the files in the same resolution. Later on, when you are working on your own applications, you can work in any resolution. Note that when using IconAuthor, although you create an application in one resolution, you will be able to play back that application in any resolution.

Fourth, we recommend that you explore the IconAuthor 7.0 Guided Tour on this CD before you begin your work on the IconAuthor 7.0 Test Drive. The Guided Tour provides a 60-90 minute comprehensive tour of IconAuthor 7.0 with an interactive, self-explanatory review of the complete product.

What if you make mistakes?

They can be fixed. We'll try to make suggestions about how to correct mistakes as you go along. However, every once in a while, you'll run your application to see how it performs. When you run the application, we'll tell you what it <u>should</u> do, and you'll have to make a mental note if there are any problems.

If something isn't working correctly, you'll stop running the application (by pressing the ESC key) and review the instructions you followed up to that point. If you find that you completed an instruction improperly, make the necessary corrections. Next, you'll run the application again to see whether you corrected the problem. The process of detecting and fixing errors is called debugging, and debugging is a normal part of authoring.

see whether you corrected the problem. The process of detecting and fixing errors is called debugging, and debugging is a normal part of authoring.

If you find you can't track down the source of a problem, you can open the completed working copy of the application that you are going to create (provided with this CD) to compare your application to the final one.

What do the conventions used in this manual mean?

Steps that you must complete appear in bold, Arial type and are preceded by a right-pointing arrow head (▶). Explanatory material appears in un-bolded, New Times Roman type. For example, an instruction and explanation might appear as follows:

▶ Click on OK.

The dialog box is removed and the color of the screen changes to blue.

Note that unless otherwise specified, any actions to be taken with the mouse refer to the left mouse button. That is, in the instruction "Click on OK," you click on OK with the left mouse button.

Previewing the Completed Test Drive

Rather than explain all the details of what EVALBIKE.IWM will do, we have given you another file called BIKE.IWM which is a completed, working copy of the application. We suggest that before you begin to work on the EVALBIKE.IWM application, run the completed Wild Planet Cycles application.

▶ Double-click on the Wild Planet Cycles icon in the IconAuthor Evaluation Program Group.

Wild Planet Cycles

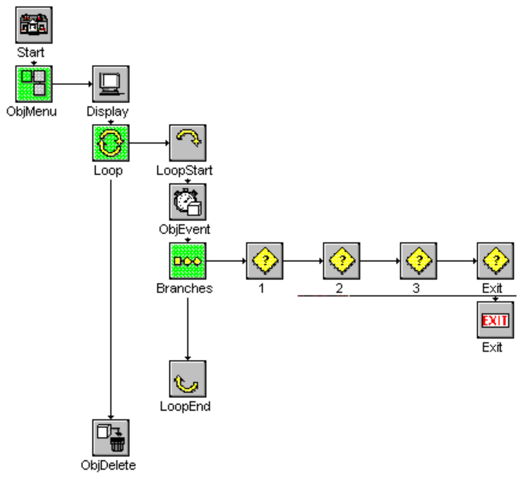
This is exactly how your application, EVALBIKE.IWM, will look once you've completed this Test Drive. Before we get started, let's review some basic concepts and terminology that should help make this development experience more enjoyable.

Basic Concepts and Terminology

Icons

Icons are the building blocks of an IconAuthor application. Each icon is a small picture that represents a function that can be performed. As an example, the icon to the right is a Display icon that allows your application to display a file (such as a graphic) on the screen.

You use IconAuthor to combine icons into a logical sequence or flowchart that depicts the flow of your application. The flowchart you create is called the structure of your application. The figure to the right shows an example of a structure created with IconAuthor.



The Start icon in the top left corner is automatically built into every structure and marks the beginning of the application. You build any other icons that are required by your application below and to the right of the Start icon.

The icons that you use to build your structure are always readily available in an on-screen, scrollable area called the Icon Library. In the library, you find the icon you want to use, drag it to the appropriate point in the structure, and drop it into place.

Objects

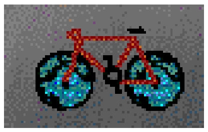
Objects are as essential to applications as icons. While icons make up the structure of your application, controlling which action takes place in what order, objects also play a key role in determining the appearance and performance of your application. Here are some examples of objects that you can use throughout your applications.

Button Objects

Button objects let you create the variety of buttons that you see in applications: push buttons, picture push buttons, check boxes, radio buttons, and group boxes. The figure on the right shows an example of a push button.

Graphic Objects

Graphic objects let you display graphics. The graphics can be actual size, stretched or compressed, or tiled within a particular area. You can designate a color in the graphic to be transparent so that it won't show at runtime.

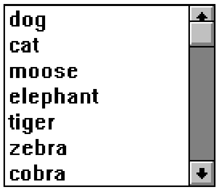


Audio Objects

Audio objects let you include sound files or CD audio selections within your applications. You can set up audio so that it plays automatically, or when the user clicks on a button.

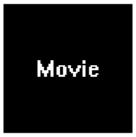
List Box Objects

A user can make selections from list boxes. If the choices that you provide do not fit in the list box, the list box automatically creates a scroll bar. You decide which items appear in the list. You also select the font and color for the items as well as whether the items appear in alphabetical order.


dog
cat
moose
elephant
tiger
zebra
cobra

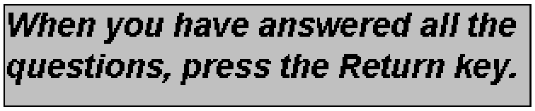
Movie Objects

The Movie object allows you to include digital video or an animation in your applications. You can provide the user with a control bar, or you can cause the object to play automatically at desired points in the application.



Text Objects

Text objects let you display text. This can be text that you provide for display purposes (such as the figure to the right) or it can be text the user inputs at runtime.


When you have answered all the questions, press the Return key.

Setting up Objects

You use IconAuthor's SmartObject Editor to create a page that contains several objects. Each SmartObject file can have several pages. In the editor you position the objects exactly where needed and you customize their appearance. For example, first you draw a Button, then you move it and resize it, then you can set its label so that is says "OK."

Defining how objects look and perform is called setting properties. Every object has properties (characteristics) that you can set. For example, in order to turn a Button object into an "OK" button, you set the object's Label property to OK.

All the objects you create are live by default. When an object is live, the user can interact with it at runtime. For example, a Button object is live and therefore a user can click on it. Also, when an object is live, your application can manipulate it. For example, if a Button object is live you can change its Label property at runtime so that it changes from a "Play" button to a "Pause" button.

You have the option of making some of the objects static, instead of live. However, once static objects are displayed, they behave as if they are part of the background. Your application cannot set properties of static objects at runtime.

Icons and Objects

How are icons and objects connected? After you create a SmartObject file you use a Display icon to display the file and its objects. After the Display icon executes, other icons manipulate the objects. The icons that control and manipulate objects all begin with the prefix “Obj”. For example, a Display icon can display a “Play” Button. After the user clicks on the Button, an ObjSet icon can reset the object’s Label property so that it says “Pause.” Later when the user clicks again, another ObjSet icon can set the Label property back to “Play.”

In this Test Drive you will be working with most of the icons and objects and basic concepts described above to complete the EVALBIKE.IWM application.

Now you’re ready to begin your first development project using IconAuthor. It is very important that you complete every instruction in the sequence in which it is given. If you omit a step, the EVALBIKE.IWM application will not perform properly.

The following five steps will assist you in adding the remaining 10% to the Wild Planet Cycles application. After you are finished, you will be ready to move on to your next development project using IconAuthor.

Step 1: Displaying the Welcome Screen

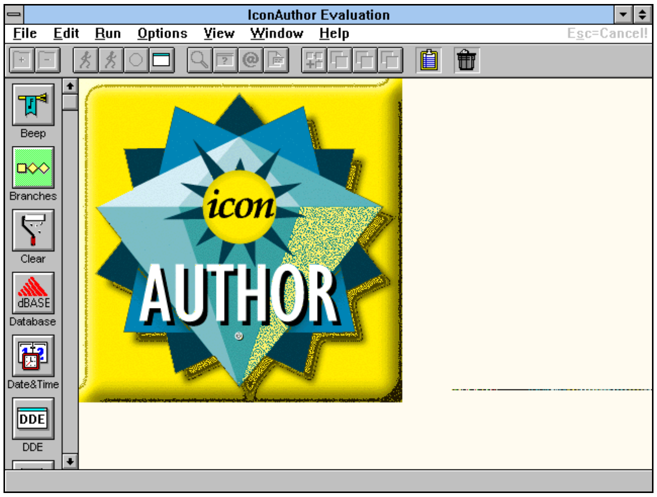
In this section you will add a Display icon to the EVALBIKE.IWM structure to show the welcome screen. This screen contains a background graphic and plays an audio clip while an animated bicycle moves across the screen. The Display icon is essential to IconAuthor applications because it is used to create several kinds of screen displays. Specifically, each Display icon you use can display a bitmap graphic, an animation script, or a SmartObject file. The Display icon is also used to pre-load graphics into memory which will increase the speed at which they are displayed. Let's begin.

Opening the Application

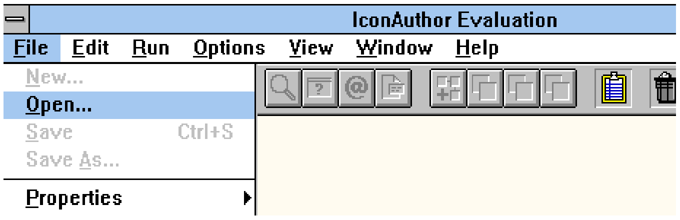
First you need to run IconAuthor.

- Double-click on the IconAuthor icon in the IconAuthor Evaluation program group.

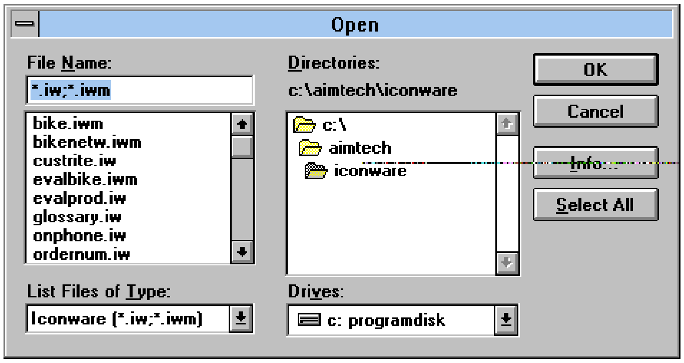
The initial IconAuthor display appears as follows. IconAuthor appears in color or in black and white depending on the capabilities of your system.



▶ Choose Open... from the File menu.



The Open dialog box appears similar to the following.

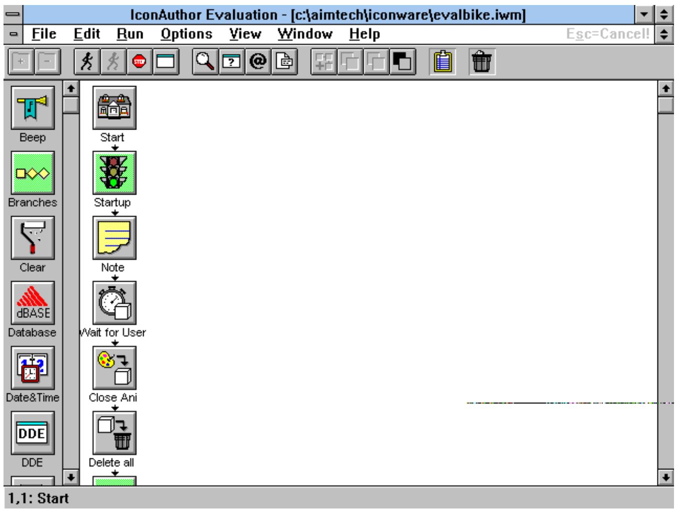


Note: You may get an error message about a missing file called EVALBIKE.PTH when the file is opened. Click on OK to continue.

▶ Click on EVALBIKE.IWM in the File Name list box. Click on OK.

The application work area and its icons will appear in a window. Use the Windows Maximize/Minimize button to make the work area full-screen.

The application EVALBIKE.IWM appears similar to the following:



As mentioned previously, the Start icon is automatically built into the top of every application. Although this tutorial won't go into detail on every icon, it will provide focus on enough key icons to give you a good idea of how IconAuthor works.

Adding a Display Icon

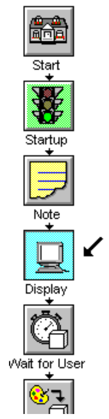
All of the icons in the structure perform a specific task. For example, some display information on the screen and others allow the user to interact. In the simplest situation, icons execute one after another, from the Start icon on down. That's the way the icons at the top of this application work. Later you'll learn more about how other rules of execution order can apply.

You're ready to add the Display icon that will show the Welcome screen. The basic procedure is to find the icon in the Icon Library, use the mouse to drag it to the appropriate position in the application structure, and drop it into place.

The Icon Library is the vertical list of icons on the far left side of the IconAuthor window. By using the scroll bar to the right of the library, you can find the icon you want to use.

- Scroll to find the Display icon in the Icon library.

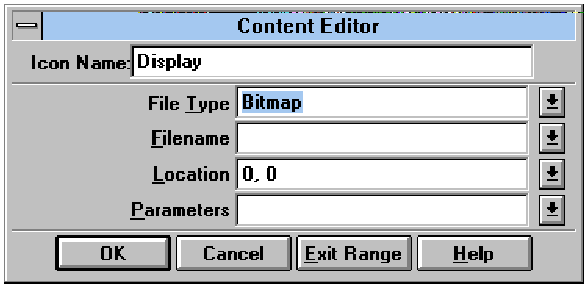
- Position the cursor over the Display icon and press and hold the left mouse button.
- Drag the icon (actually a copy of the icon) so that it is just below the first Note icon in the structure.
- Release the left mouse button to drop the icon into the structure.



The top of the application structure appears similar to the figure on the left. If you accidentally drop the icon somewhere else in the structure, simply use the same drag and drop technique to move the icon to the correct position.

In order to set up the icon, you need to open its dialog box (called a Content Editor). Every icon has a unique Content Editor. This is where you define how the icon performs at runtime.

▶ Double-click on the Display icon to open its Content Editor.



Icon Name: Display
File Type: Bitmap
Filename:
Location: 0, 0
Parameters:

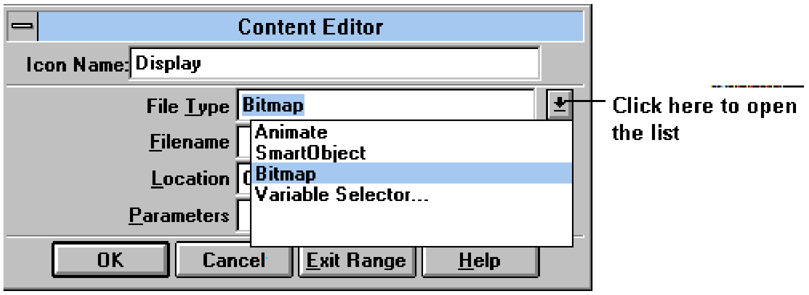
The Icon Name field controls how the icon is labeled in the structure. You'll change this field to make it a little more specific. This practice helps you to tell, at a glance, the purpose of a particular icon in the structure.

▶ Double-click on Display in the Icon Name field to select the value. Type the new value: Welcome Screen.

Next you'll work with the File Type field which identifies the kind of file you plan to display. By default, this value is set to Bitmap. However, you need to change it to SmartObject because you will be displaying a SmartObject file that we previously created for you.

This time, rather than typing the value in the field, you'll select it from the field's drop-down list.

▶ Click on the small down-arrow to the right of the File Type field and click on the SmartObject item.

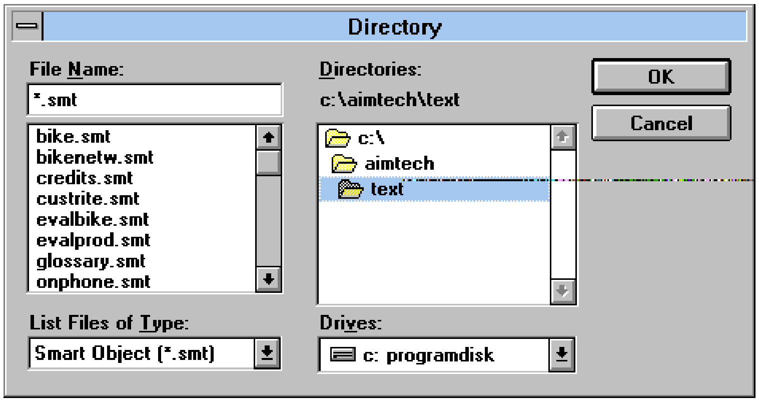


When you click on the SmartObject item, the list closes and the value is automatically returned to the File Type field. This saves you time and eliminates the risk of typing errors.

Next you'll use the Filename field to indicate the specific file you want to display. You'll select an item called Directory from the drop-down list. The Directory lets you find the appropriate filename, select it and automatically return it to the Content Editor field.

▶ Select Directory... from the drop-down list of the Filename field.

The Directory dialog box appears similar to the following:



SmartObject files use an .SMT extension. Because you specified earlier (in the File Type field) that you plan to display a SmartObject file, the File Name field contains an *.SMT filter. This filter causes all of the SmartObject filenames in your text subdirectory to appear in the File Name list box. The file you will be using is called EVALBIKE.SMT.

▶ Click on EVALBIKE.SMT in the File Name list box. Click on OK.

The dialog box closes and the filename EVALBIKE.SMT is automatically returned to the Filename field in the Content Editor.

▶ Leave the Location field set to 0,0.

The Location field specifies the position on the screen where you want the file to appear. To specify a position on the screen you need to understand that your screen is actually made up of tiny dots called pixels. As an example, a system that is referred to as having 640 x 480 screen resolution means that the screen is 640 pixels wide and 480 pixels tall.

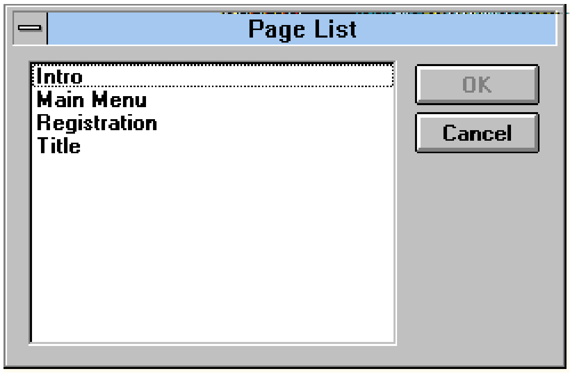
When you need to specify a position on the screen, you enter the coordinates of that point. For example, the top left corner of the screen is the point called 0,0. The bottom corner is called 640,480. Typically, you want a fullscreen file to display from the top left corner. That is why the default for the Location field is 0,0.

The last field you need to complete is the Parameters field. This field uses a different type of value depending on what type of file you are displaying. For example, when you display a bitmap, the Parameters field is often set to a special display effect. In this case, you will be setting the field to the name of a page in the SmartObject file. Briefly, pages are the discrete displays that can make up a SmartObject file. While some SmartObject files contain a single page, others (like EVALBIKE.SMT) contain multiple pages.

In this case, the page you'll be using is called Title.

▶ Select Page List... from the drop-down list of the Parameters field.

The Page List dialog box appears as follows:

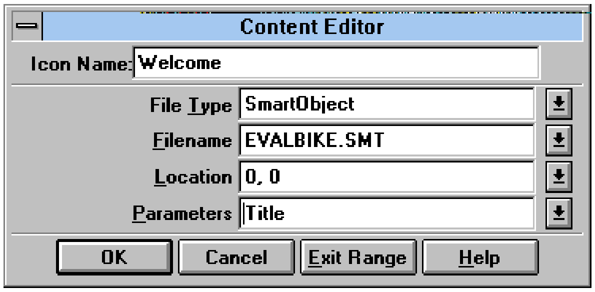


Intro
Main Menu
Registration
Title

The Page List displays the names of all the pages within the SmartObject filename that you specified in the Filename field. Note that the SmartObject file EVALBIKE.SMT has four pages: Intro, Main Menu, Registration, and Title.

▶ Click on Title and click on OK.

The page name is automatically returned to the Content Editor.



Icon Name: Welcome
File Type: SmartObject
Filename: EVALBIKE.SMT
Location: 0, 0
Parameters: Title

To review, this Content Editor directs the Display icon to display the Title page of the SmartObject file EVALBIKE.SMT in the upper left hand corner of the screen.

▶ Click on OK to close the Content Editor and accept the current values.

You're ready to save the changes that you've made and try running the application to see the impact of the new Welcome Screen.

▶ Choose Save from the File menu.

▶ Choose Application From Top from the Run menu.

The first screen you see is the Welcome Screen you just added to the application. Now stop the running application and return to the authoring window.

▶ Click on the Exit button and then on Yes to return to the authoring window.

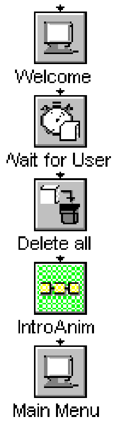
If your application is not working correctly, you can stop running the application (by pressing the ESC key) and review the instructions you followed up to this point. If you find that you completed an instruction improperly, make the necessary corrections. Then, run the application again to see whether you corrected the problem.

If you find you can't track down the source of the problem, run the completed working copy “Wild Planet Cycles application” to compare it to this one. To open the completed Wild Planet Cycles application, choose Open.. from the File menu and double-click on the file called BIKE.IWM.

Important: Do not continue to the next step of this Test Drive until you have successfully debugged EVALBIKE.IWM.

Step 2: Finishing the Main Menu

After the welcome screen, and introductory animation screen, EVALBIKE.IWM uses a Display icon (renamed "Main Menu") to show the main menu.



The main menu is actually a page called Main Menu in the SmartObject file EVALBIKE.SMT. (This is the same file that contained the page called Title that you used in Step 1.)

In Step 2, you will finish the Main Menu page by completing the following steps:

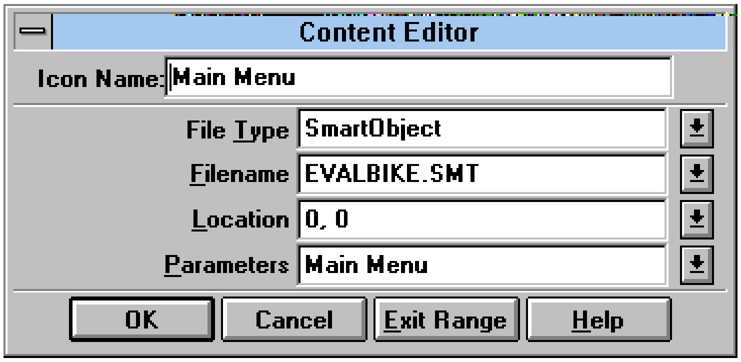
- adding a Picture Push Button used to launch a help application
- adding selectable text to provide a menu option called "Exit Course"
- creating a selectable graphic with a transparent color
- adding a Keyboard object that responds to a user selecting the F1 key

There are two general steps required to make these additions. First, you will run the SmartObject Editor, open the Main Menu page of EVALBIKE.SMT, and complete the screen by adding the remaining objects. Second, you will make the necessary changes to EVALBIKE.IWM so that the structure contains the icons that will execute when the user invokes the new options.

Opening the SmartObject File

To modify the main menu screen you need to run the SmartObject Editor. You can run the editor directly from the "Main Menu" icon in the structure.

▶ Double-click to open the Main Menu icon


Content Editor

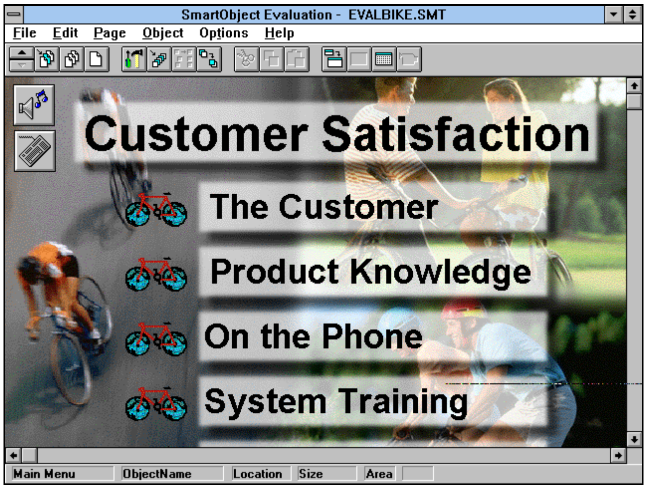
Icon Name: Main Menu

File Type: SmartObject
Filename: EVALBIKE.SMT
Location: 0, 0
Parameters: Main Menu

Like the Display icon that you used in Step 1, this icon displays a page from a SmartObject file in the top left corner of screen.

▶ In the FileName field, click on the down-arrow to open the drop-down list. Select the SmartObject Editor... item.

The SmartObject Editor appears and the Main Menu page of EVALBIKE.SMT displays automatically.



Only a portion of the current page may be visible within the work area. Use the scroll bars along the bottom and side of the SmartObject Editor to view information too extensive to be displayed at one time. Or you can view the whole page by working in "full screen" mode. In full screen mode, the work area takes up the entire screen and the scroll bars are removed from the screen.

- To change to full screen mode, select Options from the menu bar and choose Full Screen.

This SmartObject page contains a number of different objects. The object that provides most of the colorful visual information on this page is a Graphic object. The Graphic object contains a file called MAINMNU.BMP. The following figure shows how that graphic looks on its own.

The menu items (labeled “The Customer”, “Product Knowledge”, and so on) are Text objects. They are borderless, contain black text and have a transparent background. The transparent background makes the text appear to sit directly on the background graphic.

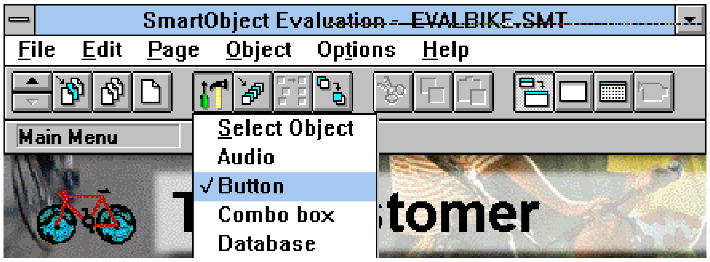
Creating Objects: Buttons, Text, Graphics, Keyboard Object

Now you are ready to finish the Main Menu screen by adding some additional objects to the screen. The first object that you will add is a PicturePush Button. A PicturePush Button is a graphic button that has up to three different states: enabled, selected, and disabled. A PicturePush Button can provide a user with visual feedback when it is selected, or disabled. We will use a small graphic of a question mark with a bike super-imposed on it to create graphical button for launching the application’s Help function.

Adding a Help Button Object

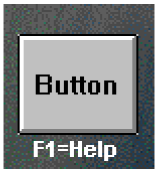
In this step you will start by creating a new object.

- Click on the Tools button in the ribbon bar.



- Select Button.

The cursor takes the shape of a cross-hair. Now that you have selected the appropriate tool, you are ready to draw. You will draw the button in the lower left area of the page, just above the words F1=Help. It should look similar to the following:



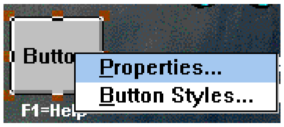
- Position the cursor where you want the top left corner of the button to be.
- Press and hold the left mouse button.
- Drag the mouse diagonally down and to the right until you are satisfied with the size. (Don’t worry about the exact position of the button- you can move it later.)
- Release the left mouse button.

The Button is drawn. The next step is to select the appropriate style for the button.

Although the Button object always appears in Push Button style by default, you can change this. For example, you can make it a Picture Push Button, a Check Box, or a Radio Button. In this case, you will change the style of the object so that it is a Picture Push Button. A Picture Push Button contains a graphic rather than a text label.

▶ Click on the object with the right mouse button.

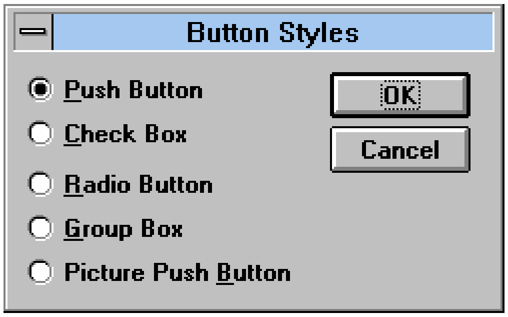
A pop-up menu appears.



The pop-up menu varies from one object to another but always begins with the Properties... command. Before setting any object properties, you will change the object's style. It's a good idea to set the style first because the properties of an object may vary depending on the selected style.

▶ Choose Button Styles... from the pop-up menu.

The Button Styles dialog box appears as follows:



▶ Select Picture Push Button and click on OK.

Notice that the button is now solid gray because it no longer contains the text label. Later in this step you will load the appropriate graphic into the object.

Setting Properties

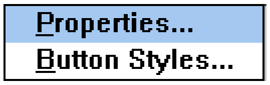
Now you are ready to change one of the characteristics of an object so that it will perform properly and be interactive at runtime. Objects actually have several characteristics that you can pre-set in the editor (and re-set as necessary in the application structure). These characteristics are called properties. As an example, a Button object has a property called CursorName that is initially set to Default. When the property is set Default, the cursor appears as an arrow over that button. If you were to set the CursorName property to Indexed Hand, the cursor would show up as an indexed hand when the user moved his or her cursor over that button at runtime.

Many properties like the Editable property serve a specialized purpose and are only used by a limited number of objects. You'll also find that each object has several properties, some of which are common to other objects. Most properties are set to a default.

Let's see how you view and change the properties of an object.

▶ Right click on the button object.

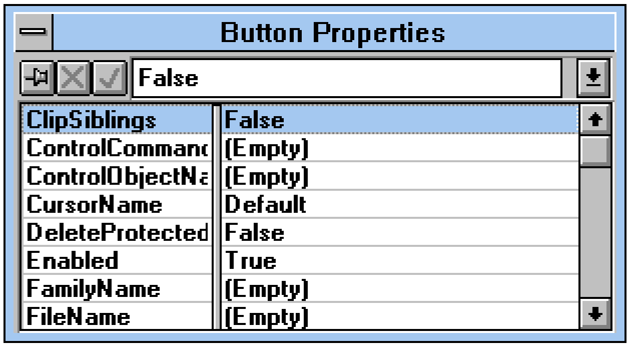
A pop-up menu appears.



The Properties command displays the dialog box where you view and change the properties of the selected object.

▶ Choose Properties...

A Properties dialog box appears. The title of this dialog box varies from one class of object to another. For example, if you choose Properties... for a Button object, the Button Properties dialog box appears as follows:


Button Properties

False

ClipSiblings	False
ControlCommanc	(Empty)
ControlObjectNa	(Empty)
CursorName	Default
DeleteProtected	False
Enabled	True
FamilyName	(Empty)
FileName	(Empty)

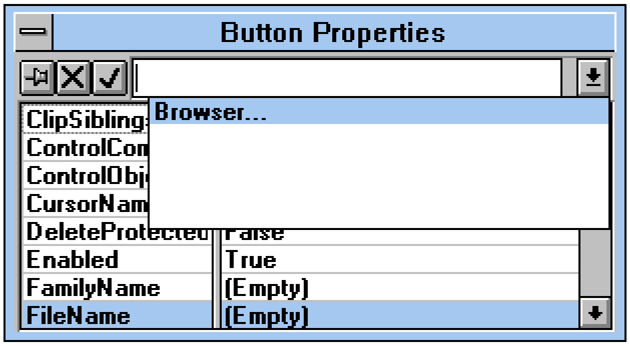
The Properties dialog box identifies all of the properties available for the currently selected object. The property names are displayed on the left and the property values are displayed on the right. The edit box at the top of the dialog box contains the value(s) currently set for the highlighted property. If you look down the list you'll see the CursorName property. Notice that it is set to Default. The default cursor on top of a button is an arrow. There are many other cursors available to you, but for now we'll stick with the default.

The next property we'll look at is FileName. Because this is a Picture Push Button it will contain a graphic rather than a text label. To accomplish this, you'll set the object's FileName property. You will use a file called HELP.BMP that we have provided for you.

▶ Scroll to the FileName property and click on it.

▶ Click on the down-arrow to the right of the edit box.

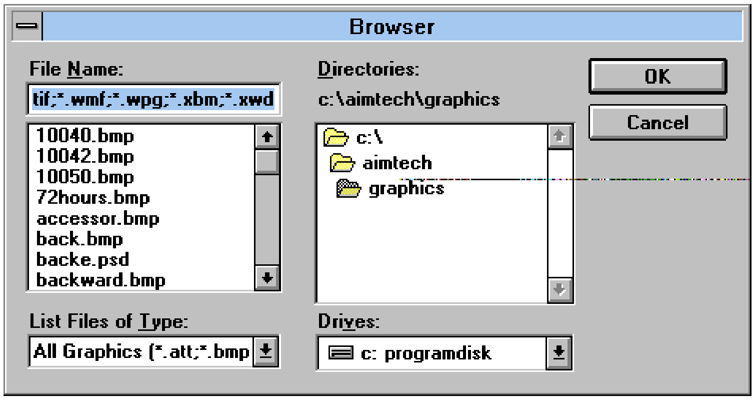
The Properties Dialog Box appears as follows:



A tool called the Browser is available in the drop-down list. The Browser lets you search for the file you want to use.

▶ Click on Browser...

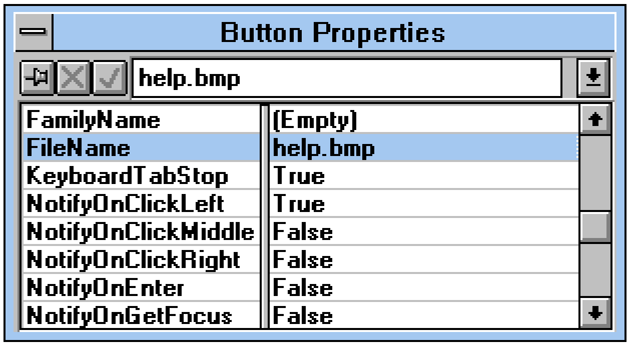
The Browser appears similar to the following:



▶ Select the file HELP.BMP from the Graphics sub-directory and click on OK.

The Browser closes and the file name you selected is automatically returned to the edit box in the Button Properties dialog box.

The Properties dialog box appears as follows:


Button Properties

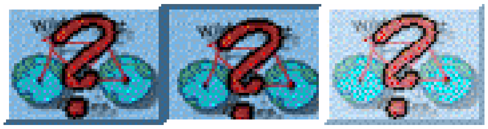
help.bmp

FamilyName	(Empty)
FileName	help.bmp
KeyboardTabStop	True
NotifyOnClickLeft	True
NotifyOnClickMiddle	False
NotifyOnClickRight	False
NotifyOnEnter	False
NotifyOnGetFocus	False

Notice also that the graphic file has automatically been loaded into the object. The button now appears as follows:

The next property you will review is called ButtonStates. This property lets you identify how many different states your button has. IconAuthor's Picture Push button supports up to three states: ready to be clicked (Up), being clicked (Down), and unavailable (Disabled).

You will use the ButtonStates property to allow all three states: Up, Down, and Disabled. Note that IconAuthor does not actually change the graphic to reflect the button's different states. Rather, the graphic that you specify for the FileName property should contain side by side images of all the possible button states as follows. In the case of your three state button (UpDownDisabled), the original graphic has been built as follows:



- Scroll to the ButtonStates property and click on it.

The default value is UpDownDisabled.

- Leave the ButtonStates property as is.

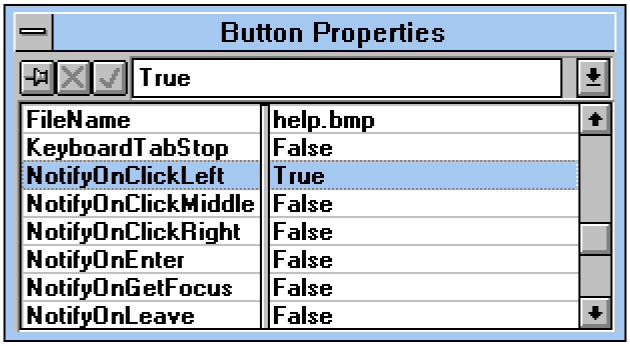
If you like, you can try changing the button states property to see the effect changing the number of states has on the appearance of your button.

Let's look at a couple of other properties before we proceed. There are two ways to move through the properties list. You can use the scroll bar on the right side of the dialog box. Or, you can press an alphabetical key on the keyboard to jump to the next property that begins with that character.

▶ Press the "N" key once to go to the NotifyOnClickLeft property.

Notice that it set to True by default.

It is up to the author to decide which of the user's actions are recognized by the application as events. When the NotifyOnClickLeft property is set to true, the application recognizes a left mouse click on the button as an event. An application can than branch based upon which object generates the event.


Button Properties

True

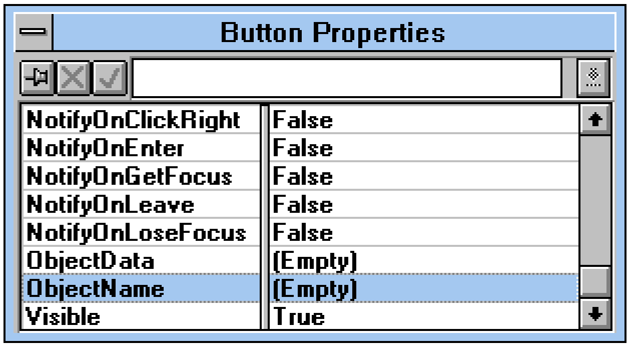
FileName	help.bmp
KeyboardTabStop	False
NotifyOnClickLeft	True
NotifyOnClickMiddle	False
NotifyOnClickRight	False
NotifyOnEnter	False
NotifyOnGetFocus	False
NotifyOnLeave	False

▶ Press the "N" key again to view the NotifyOnClickMiddle and NotifyOnClickRight properties.

Notice that these properties are set to False by default. Because they are false, if a user clicks on the object with the middle or right mouse button, the action is not recognized as an event.

Now that you've seen how to view properties, you will set a final very important property: ObjectName. The ObjectName property lets you assign a unique label to an object so that your application can manipulate the object at runtime. Once the application detects that an event has occurred, it will need to know exactly which object was clicked on.

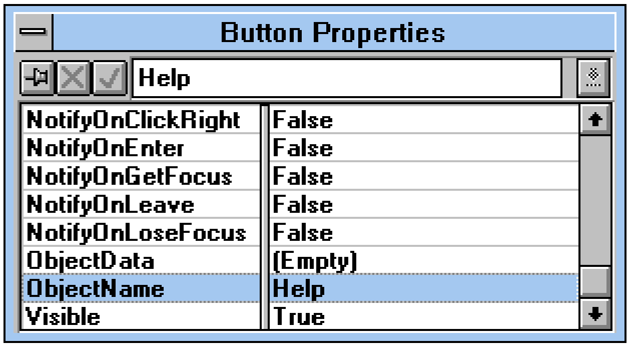
▶ Use the scroll bar and click on the ObjectName property.


Button Properties

NotifyOnClickRight	False
NotifyOnEnter	False
NotifyOnGetFocus	False
NotifyOnLeave	False
NotifyOnLoseFocus	False
ObjectData	(Empty)
ObjectName	(Empty)
Visible	True

The current ObjectName is set to (Empty) because we haven't assigned it a name yet.

- To change the current setting for this property, click in the edit box at the top of the dialog box. Type the new value Help.
- Press Enter or click on the Check Mark button to accept the change.


Button Properties

Help

NotifyOnClickRight	False
NotifyOnEnter	False
NotifyOnGetFocus	False
NotifyOnLeave	False
NotifyOnLoseFocus	False
ObjectData	(Empty)
ObjectName	Help
Visible	True

Now if the user clicks on the object, the application can not only detect that an event occurred, but it will have a way of knowing which object was affected. You'll learn more about the specific role of the ObjectName property when you return to IconAuthor.

The Help button is complete. Later, you will return to IconAuthor and set up the necessary icons that will launch a help application when the user selects this button.

- Choose Save from the File Menu.

Adding Selectable Text

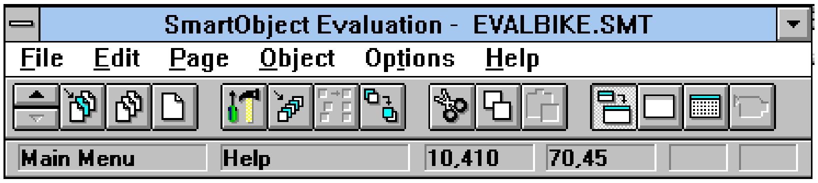
The second addition we will make to our screen is a menu option for the user to exit the course. We will do this by adding a selectable text to our screen. Instead of creating a brand new text object from scratch, we will accelerate the design process by copying one of the previously created menu options and modifying it to fit our needs.

To copy an object, you must first select it.

- Click on the text object that says 'System Training' with your mouse.

Resizer handles to appear around the object as follows:

- To Copy the object, go to the ribbon bar and click on the Copy button.


SmartObject Evaluation - EVALBIKE.SMT

Main Menu | Help | 10,410 | 70,45

▶ Next, click on the Paste button also found in the ribbon bar just to the right of the copy button.

The paste button will put an exact copy of the text object you just copied into the upper left hand corner of your screen. (Hint: another way to copy objects is by using a Windows convention. Simply select an object by clicking on it once with your mouse. Then, hold down the Control key, and drag and drop a copy of the selected object to the target location using your mouse).

▶ Drag the copied text and position it over the empty space on the menu graphic.

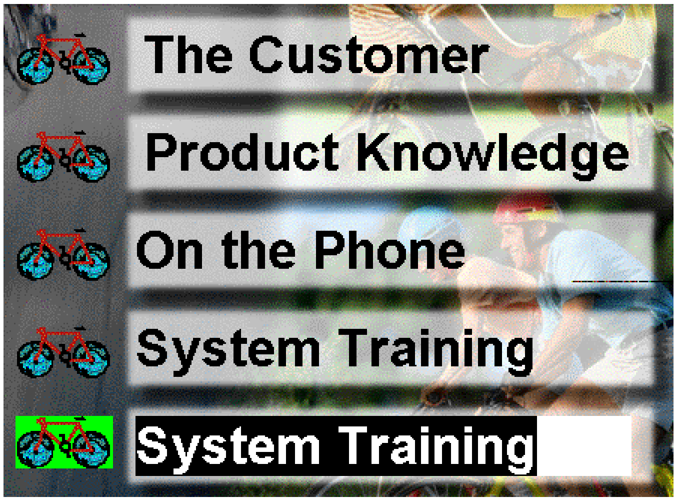
Now, since you don't need two menu options that say the same thing, you need to modify the text to say 'Exit Course'.

▶ Double-click on the text block to enter text edit mode.

You will know when you are in text edit mode when your cursor shows up as a blinking I-Beam inside the text box and the text background appears as white instead of transparent.

▶ Highlight the text to be changed by holding down the left mouse button and dragging over the text.

Your screen should appear as follows:



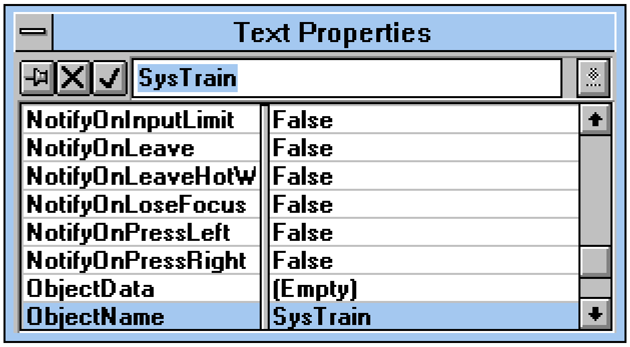
▶ Type in the new text, Exit Course.

We also need to change the ObjectName property for our new text object. Every Text Object in the main menu typically has its ObjectName property set to a unique value. We have already assigned Object Names to the other menu items as follows:

Menu Item:	ObjectName:
"The Customer"	CustRight
"Product Knowledge"	ProdKnow
"On the Phone"	OnPhone
"System Training"	SysTrain

You will set the "Exit Course" item's ObjectName to Exit Course.

- Right click on the "Exit Course" object to bring up the pop-up menu.
- Choose Properties...
- Use the scroll bar to find the ObjectName Property, then click on it.



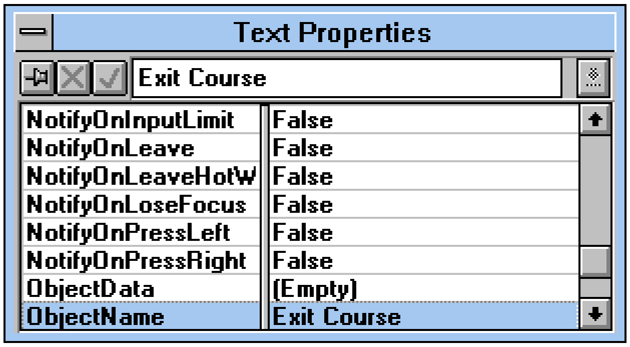
Text Properties

SysTrain

NotifyOnInputLimit	False
NotifyOnLeave	False
NotifyOnLeaveHotW	False
NotifyOnLoseFocus	False
NotifyOnPressLeft	False
NotifyOnPressRight	False
ObjectData	(Empty)
ObjectName	SysTrain

The current ObjectName is set to SysTrain since our new text is a copy of the "System Training" text from the menu.

- To change the current setting for this property, click in the edit box at the top of the dialog box. Type the new value Exit Course.
- Press Enter or click on the Check Mark button to accept the change.



Text Properties

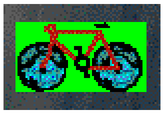
Exit Course

NotifyOnInputLimit	False
NotifyOnLeave	False
NotifyOnLeaveHotW	False
NotifyOnLoseFocus	False
NotifyOnPressLeft	False
NotifyOnPressRight	False
ObjectData	(Empty)
ObjectName	Exit Course

Now if the user clicks on the object, the application can not only detect that an event occurred, but it will have a way of knowing which object was affected. You'll learn more about the specific role of the ObjectName property when you return to IconAuthor.

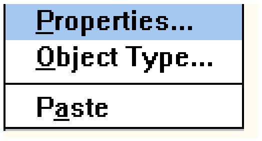
- Choose Save from the File Menu.

The next thing you'll do is to give your users the option of clicking on a graphic to indicate where they want to go next. As you can see, the previous author copied a graphic of a bicycle next to the Exit Course option, but did not finish editing its properties. The graphic appears as follows:



You need to make two modifications to finish editing this graphic object: first, give this graphic object an object name and second, make the lime background transparent.

- Right click on the graphic of the bicycle object to bring up the pop-up menu.



- Choose Properties...
- Use the scroll bar to find the ObjectName Property, then click on it.

The current ObjectName is set to (Empty) because we haven't assigned a name to it yet.

- To change the current setting for this property, click in the edit box at the top of the dialog box. Type the new value Exit Course.

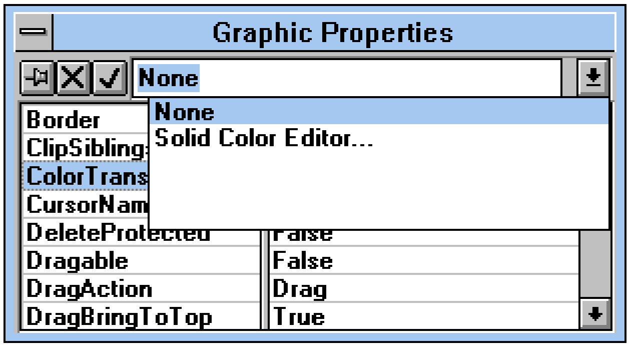
The graphic of the bicycle will share the same object name as our Exit Course text block so that the user can click on either the text or the graphic to leave the course.

- Press Enter or click on the Check Mark button to accept the change.
- With the Property Selector still open, hit the "N" key once to move the NotifyOnClickLeft property. Confirm that the property value is set to True.

You have just confirmed that the application will receive a notification if a user clicks on the graphic. Your final change will be to make the background color of lime a transparent color.

- Use the scroll bar (or hit the "C" key twice) to move to the ColorTransparent property. By default, the ColorTransparent property is set to None.
- Click on the down-arrow to the right of the edit box.

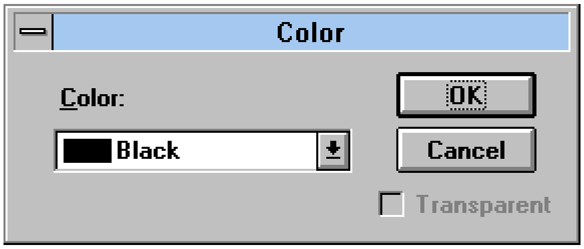
Your dialog box should appear as follows:



Border	
ClipSibling:	
ColorTrans	
CursorNam	
DeleteProtected	False
Dragable	False
DragAction	Drag
DragBringToTop	True

▶ Select the Solid Color Editor... option.

A color selection dialog box appears similar to the following:



Color: Black

▶ Click on the down-arrow to the right of the edit box, and select Lime from the drop-down list.

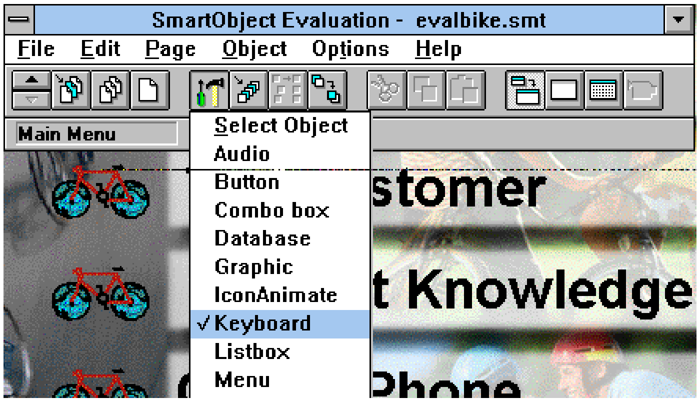
▶ Click on OK to close the dialog box.

▶ Choose Save from the File Menu.

Recognizing Keyboard Input

The final modification you will make to the screen is the addition of a Keyboard Object to capture a user striking the F1 key to get help. This will allow your application to respond to both a mouse, and to the keyboard. Following the same general process we've used before, we need to add a new object to the screen, and then set its properties.

- Click on the Tools button in the ribbon bar.



- Select Keyboard.
- Position the cursor where you want to place the top left corner of the object.
- Drag and draw a KeyBoard object somewhere in the upper left corner of the screen. (Don't worry about the exact size or position of the object).

The KeyBoard object looks like this:

When drawn, it snaps to a default size. Feel free to re-position it on the screen anywhere you find it convenient. It doesn't really matter where you place it since when the application is run, the object does not appear on the screen. It shows up now because as an author, you need to know that you've added keyboard functionality.

The next step is to set the KeyBoardList property so that the application knows which key strokes to recognize.

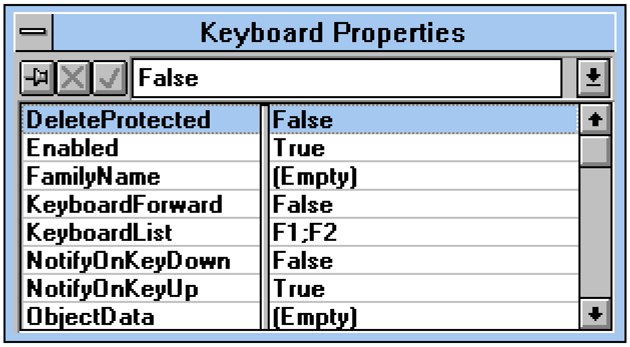
- Click on the object with the right mouse button.

Properties...

- Choose Properties... from the pop-up menu.

Our first step is to protect the Keyboard object from being deleted, so that a user can select the F1 key to get help at any point in the application. To do this, we need to set a property called DeleteProtected to True. DeleteProtected is the first property in your list and should be highlighted when you open up the Properties dialog box.

- Make sure that the DeleteProtected property is highlighted.



False

DeleteProtected	False
Enabled	True
FamilyName	(Empty)
KeyboardForward	False
KeyboardList	F1;F2
NotifyOnKeyDown	False
NotifyOnKeyUp	True
ObjectData	(Empty)

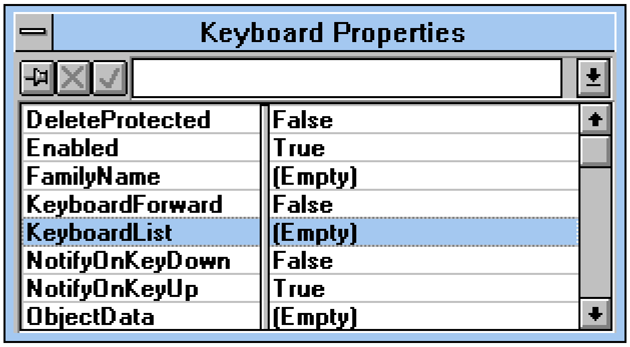
- Click on the down-arrow to the right of the edit box to show the options.
- Select True.

The next step is insure that the keyboard object is ready to respond to a user's keystrokes. You will do this by setting the Enabled property to True.

- Highlight the Enabled property by clicking on it.
- Click on the down-arrow to the right of the edit box to show the options.
- Select True.

Now you can set the KeyboardList property. This property determines the keys or key stroke combinations to which the keyboard object will respond.

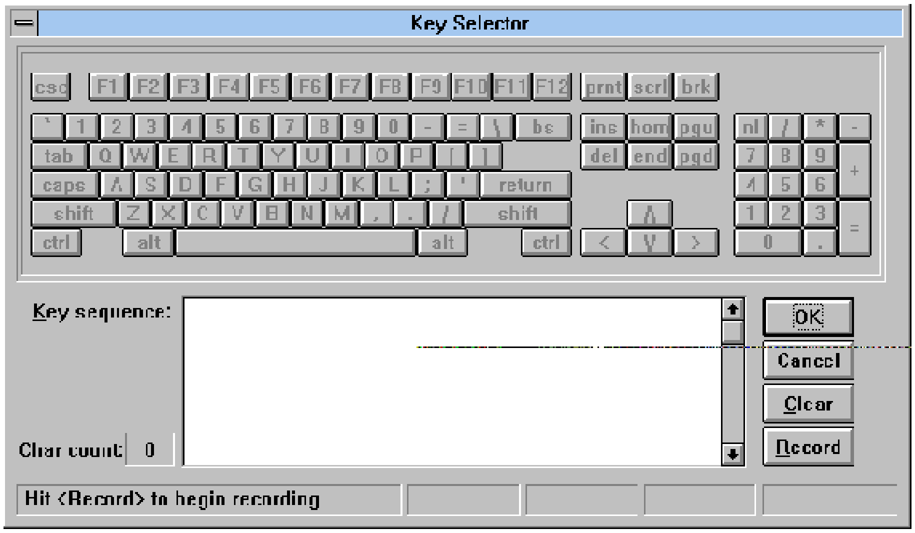
- Use the scroll bar (or hit the "K" key twice) to move the KeyBoardList property.



DeleteProtected	False
Enabled	True
FamilyName	(Empty)
KeyboardForward	False
KeyboardList	(Empty)
NotifyOnKeyDown	False
NotifyOnKeyUp	True
ObjectData	(Empty)

- Click on the down-arrow to the right of the edit box.
- Select the Key Selector... option.

A Key Selector dialog box appears similar to the following:


Key Selector
esc F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11 F12 prnt scrl brk
` 1 2 3 4 5 6 7 8 9 0 - = \ bs ins hom pgu nl / * -
tab Q W E R T Y U I O P [ ] del end pgd 7 8 9 +
caps A S D F G H J K L ; ' return 4 5 6
shift Z X C V B N M , . / shift 1 2 3
ctrl alt alt ctrl 0 . =
Key sequence:
Char count: 0
OK
Cancel
Clear
Record
Hit <Record> to begin recording

▶ Click on the Record button.

▶ Use your mouse to select the F1 key.

▶ Click on Stop to finish recording.

▶ Click on OK to close the dialog box.

Finally, review your Notify properties before you finish up.

▶ Scroll or Press the "N" key once to go to the NotifyOnKeyDown property.

Notice that it set to False by default. The next property, NotifyOnKeyUp is set to True. This means that that the Keyboard object will generate a notification (event) when the user lets go of one of the keys specified in the KeyboardList property. The application can then respond to that keystroke event anyway the author wants.

Now you will set that final very important property: ObjectName.

▶ Use the scroll bar to find the ObjectName Property, then click on it.

The current ObjectName is Empty.

- Click in the edit box at the top of the dialog box. Type the new value Keyboard.
- Press Enter or click on the Check Mark button to accept the change.

You have created a PicturePush button, added a textual menu option, added a selectable graphic with a transparent background, and permitted the application to respond to keystrokes with a Keyboard object. You have now finished the Main Menu.

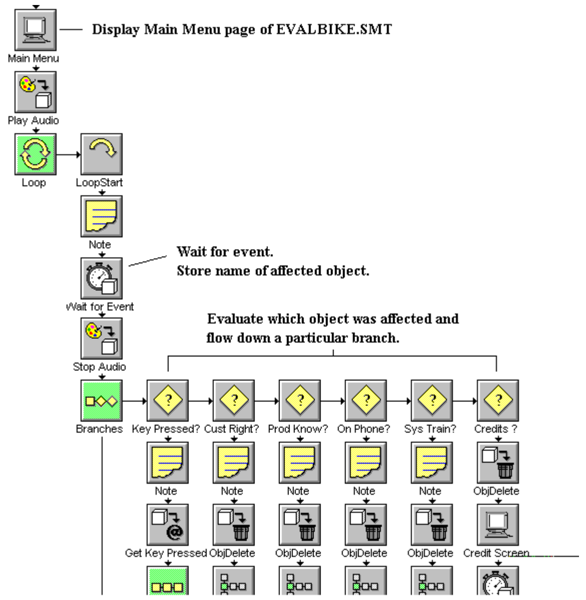
- Choose Save from the File Menu.
- Choose Exit from the File Menu.

Once again, you are back at the Display icon from which you opened the Smart Object Editor. Next, you’ll work on the icons that actually cause the application to exit when the user clicks on the “Exit Course” button.

Step 3: Modifying the Application Structure

Overview of the Menuing Structure

Before you make more changes to EVALBIKE.IWM, you need to know a little about how menuing works. The following figure shows the portion of the application that we'll be working on.



▶ Scroll the window so that you can view this general part of the application.

Don't be concerned with every icon in the structure. For now, we'll focus on the icons that form the backbone of the menu structure.

First, there is the "Main Menu" Display icon that you already know about. It displays the Main Menu page of EVALBIKE.SMT. The next key icon is the "Wait for Event" ObjEvent icon. It is set up to wait indefinitely for an event to occur. An event is an action that is recognized by your application. When the ObjEvent icon detects that an event occurs, it lets execution flow down to the next icon. It's up to the author to decide which actions are recognized as events by the application. In this case, we have set up the events so that a left click is an event, but a right click is not. (This is accomplished elsewhere in the structure.)

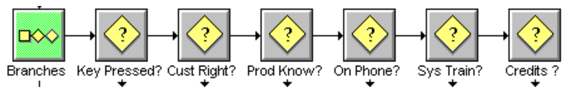
This is a good time to learn about another essential function performed by the ObjEvent icon. In addition to letting execution continue on when an event occurs, the icon also stores the ObjectName of the affected object in a system variable called @_OBJECT_NAME. Once the value is stored in @_OBJECT_NAME, it is up to the author to set up subsequent icons to manipulate the variable. In this case, we have set up the structure that manipulates the variable for you.

The Branches

The variable @_OBJECT_NAME is manipulated in a part of the structure referred to as branches. Up to now, you have seen the most basic rule of execution flow where execution passes from one icon down to the next icon, and so on. At this point, you will learn about some of the important exceptions to this rule.

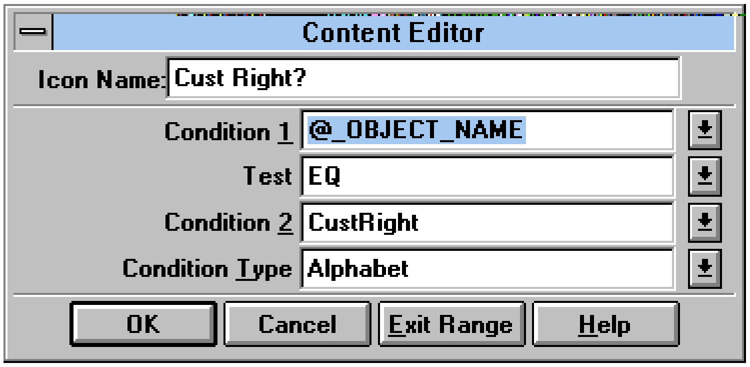
Branching, as its name implies, is where an application branches off in several directions. In EVALBIKE.IWM, when a user clicks on an item in the Main Menu, the application will branch to a different part of the structure (with a different series of icons) depending on the selected item.

After the "Wait for Event" ObjEvent icon executes, the next key icon is the one labeled Branches. When execution arrives at the Branches icon, flow is directed to the right rather than downward. The six icons to the right of the Branches icon are where the evaluation of the ObjectName occurs.



These icons are If icons, although they have all been renamed in EVALBIKE.IWM. The purpose of an If icon is to ask whether a condition is true and then direct execution downward (if the result is true) or to the right (if the result is false). In this case, each If icon asks whether a specific ObjectName is currently stored in @_OBJECT_NAME.

▶ Double-click to open the If icon named "Cust Right?"


Content Editor

Icon Name: Cust Right?

Condition 1 @_OBJECT_NAME

Test EQ

Condition 2 CustRight

Condition Type Alphabet

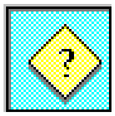
The values ask if @_OBJECT_NAME is equal (EQ) to CustRight. (Remember that CustRight is the ObjectName of the menu item labeled "The Customer.") If the user has clicked on the menu item labeled "The Customer" this If icon would evaluate to true and execution would flow downward to the icons that present the appropriate part of the application. If the user has not clicked on the item labeled "The Customer," execution flows to the right, to the next If icon.

▶ Click on OK to close the "Cust Right?" icon Content Editor.

Adding an If Icon

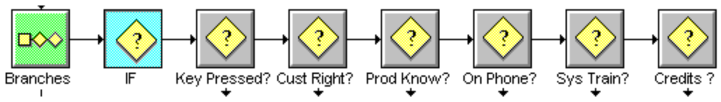
As mentioned previously, right now there are six If icons. The first If icon performs a test to determine whether the user pressed a key on the keyboard. The other If icons test which of the first four Main Menu items was clicked. You will now add another If icon that tests whether the user clicked on the “Exit Course” item.

- Find the If icon in the Icon library.

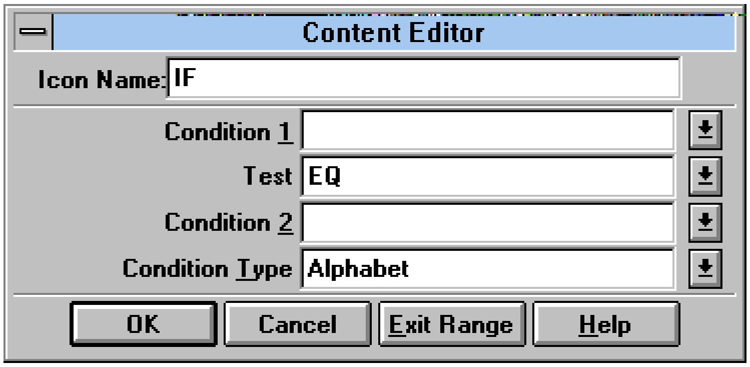


- Drag it directly to the right of the Branches icon and drop it in place.

The structure appears as follows:



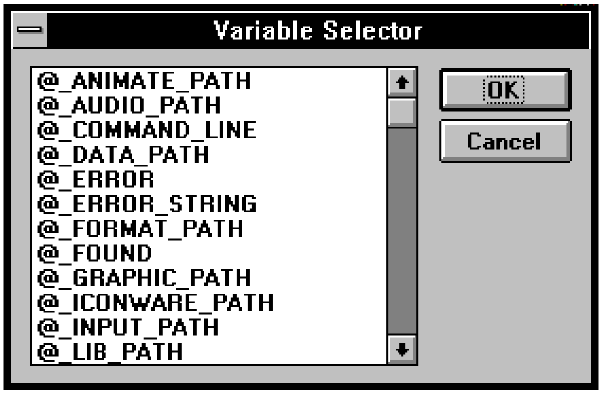
- Double-click on the If icon to open the Content Editor.



Icon Name: IF
Condition 1:
Test: EQ
Condition 2:
Condition Type: Alphabet

- Change the Icon Name to "Exit Course?"
- In the Condition 1 field, select Variable Selector... from the drop-down list box.

The Variable Selector appears as follows:



@_ANIMATE_PATH
@_AUDIO_PATH
@_COMMAND_LINE
@_DATA_PATH
@_ERROR
@_ERROR_STRING
@_FORMAT_PATH
@_FOUND
@_GRAPHIC_PATH
@_ICONWARE_PATH
@_INPUT_PATH
@_LIB_PATH

▶ Scroll down in the list and select @_OBJECT_NAME. Click on OK.

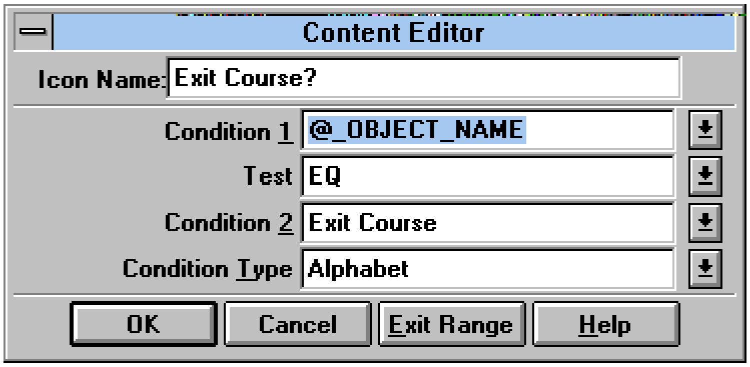
The value @_OBJECT_NAME is automatically returned to the Condition 1 field.

▶ Leave the Test field as is, set to EQ.

▶ Type Exit Course in the Condition 2 field.

▶ Leave the Condition Type field as is, set to Alphabet.

The Condition Type field identifies the kind of value that you are evaluating (alphabetical versus numeric). The If Icon Content Editor appears as follows:



Icon Name: Exit Course?

Condition 1 @_OBJECT_NAME
Test EQ
Condition 2 Exit Course
Condition Type Alphabet

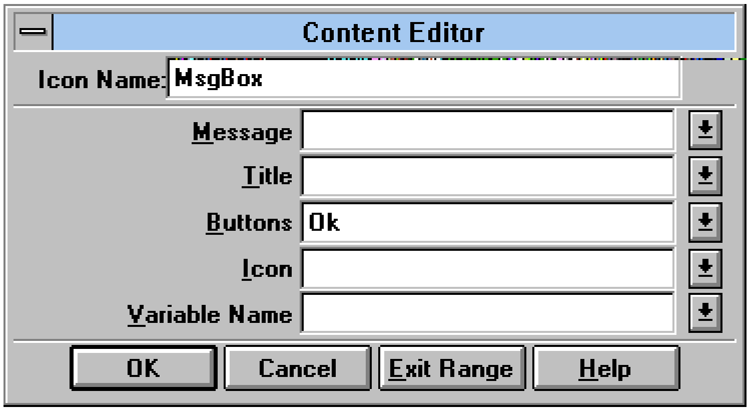
To review, this icon will compare the value Exit Course with the value currently stored in @_OBJECT_NAME. If the user clicks on the "Exit Course" item, execution will flow down this branch. If the user clicks on another item, execution will flow to the right to the next If icon.

▶ Click on OK to close the Content Editor.

Adding a Message Box

Now you'll begin adding the icons that should execute if the "Exit Course?" If icon evaluates to true. The first icon that you will add to this branch is a MsgBox icon. It will display a Message Box that advises the user that he/she is about to exit the course.

- Find the MsgBox icon in the Icon Library.
- Drag a MsgBox icon immediately below the "Exit Course?" If icon.
- Open the MsgBox icon's Content Editor.



Icon Name: MsgBox

Message

Title

Buttons: Ok

Icon

Variable Name

- In the Message field type: You have chosen to exit the course.

Next, use the Title field to specify the title that will appear on the Message Box.

- Type Exit Course in the Title field.

Use the Buttons field to specify which combination of buttons to display. You can display a variety of buttons, for example, Yes and No, or OK and Cancel.

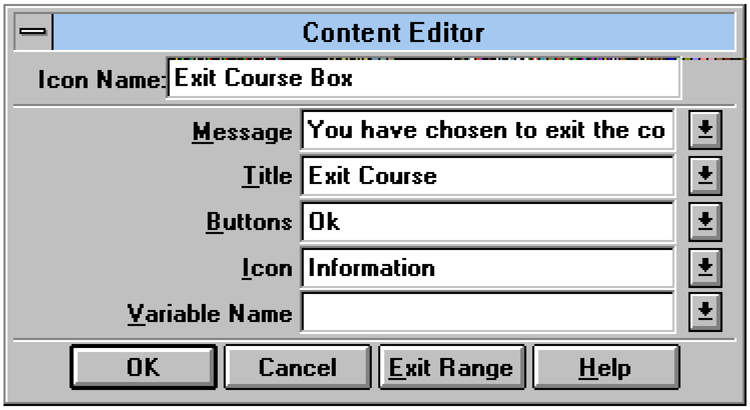
- Select "OK" from the drop-down list of the Buttons field.

The Icon field lets you specify what kind of symbol the Message Box should contain (for example an exclamation point or a question mark). Each symbol is intended to provide immediate visual information to the user about the nature of the message. In this case you'll use an informational symbol.

- Select "Information" from the drop-down list of the Icon field.

In the last field, you can specify a variable in which the user's selection (such as Yes or No) is stored. Since our message box is only informational, we do not need to capture the user's response for evaluation. We will leave this field blank.

The Content Editor should appear as follows:



Icon Name: Exit Course Box

Message: You have chosen to exit the co

Title: Exit Course

Buttons: Ok

Icon: Information

Variable Name:

To review, this icon displays the informational message “You have chosen to exit the course”. It presents the user with an “OK” button. When the user clicks on the OK Button, the application flow will continue on to the next icon in the structure.

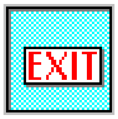
The Variable Name field allows you enter the name of a variable in which a user’s selection from the message box can be stored. It’s important to capture the user’s response when there are multiple buttons in the Message Box (YesNoCancel) and you need to know which one the user selected.

- Click on OK to close the Content Editor

Adding an Exit Icon

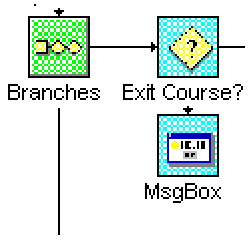
The last step in building the exit branch is to add an Exit icon. Depending on how you set it up the Exit icon can cause execution flow to exit from one of many types of structures. In this case, you’ll provide values so that the Exit icon causes the application to complete.

- Find the Exit icon in the Icon library.

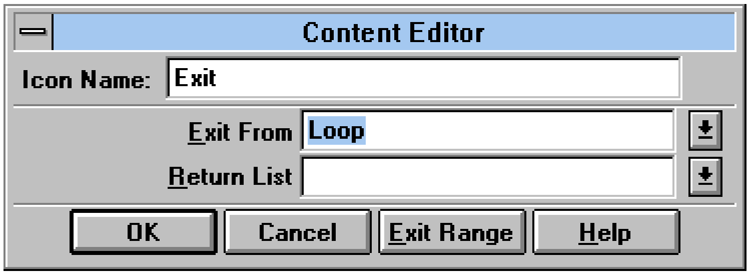


- Drag it immediately below the Message Box icon and drop it in place.

The application structure appears as follows:



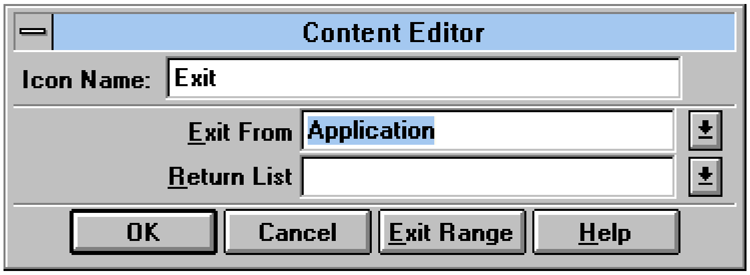
▶ Double-click to open the Exit icon Content Editor.



Icon Name: Exit
Exit From: Loop
Return List:

▶ Select "Application" from the drop-down list of the Exit From field.

The Content Editor appears as follows:



Icon Name: Exit
Exit From: Application
Return List:

The Exit icon will cause the user to exit the entire application. It will execute when the user clicks on the OK button to acknowledge the message.

Congratulations, you've just completed the exit branch of the EVALBIKE.IWM application. Now, let's run the application to see how it works.

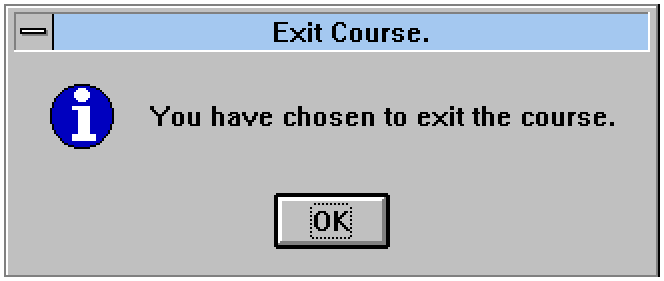
▶ Choose Save from the File menu to save your work.

▶ Choose Application From Top from the Run menu.

▶ Follow the on-screen instructions to until you get to the Main Menu.

▶ Click on the "Exit Course" item from the Main Menu.

The Message Box appears as follows:



Exit Course.

You have chosen to exit the course.

OK

- Click on OK in the Message Box.

The application executes the next icon in the structure which exits the application and returns you to the authoring window.

Adding Another Branch

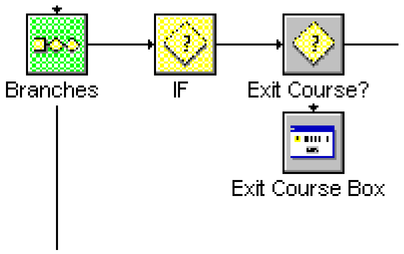
Next, you will add the icons to launch a Help application. When the user clicks on the Help button that you added to the Main Menu screen, Windows program called WINTUTOR.EXE will execute.

First, you need to add an If icon to test when the user selects this object.

- Find the If icon in the Icon library.

- Drag it to the right of the icon labeled "Branches" icon.

The application structure appears as follows:



- Double-click on the If icon to open the Content Editor.
- Change the Icon Name field to Help Button?

Like the other If icons in this part of the structure, the "Help Button?" If icon will compare the value currently stored in @_OBJECT_NAME with the name of a specific object (in this case the object named HELP).

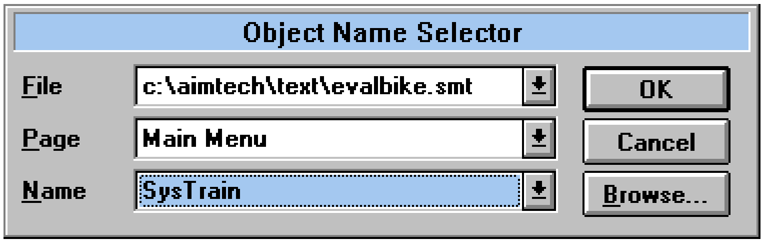
Rather than type in the variable name, this time you will use a tool called the Variable Selector that lets you select and automatically return the variable name. This removes the chance of a typing error.

- In the Condition 1 field, select Variable Selector... from the drop-down list box.
- Scroll down in the list and select @_OBJECT_NAME. Click on OK.

The Variable Selector is removed and the value @_OBJECT_NAME is automatically returned to the Condition 1 field.

- Leave the Test field as is, set to EQ.
- In the Condition 2 field, select Object Name Selector... from the drop-down list box..

The Object Name Selector should appear as follows:



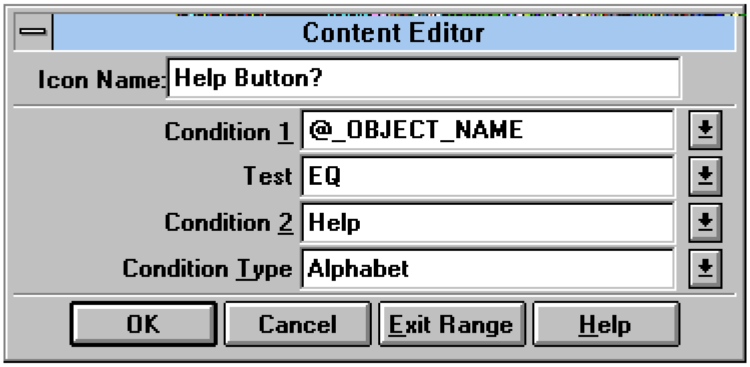
File: c:\aimtech\text\evalbike.smt
Page: Main Menu
Name: SysTrain

Make sure that the entry in the File field shows the EVALBIKE.SMT since that is the name of the file you modified. If the file name is not correct, you can user the browser to find the correct file. The Page field should read Main Menu. If you need to select another page in your EVALBIKE.SMT Smart Object file, you can view other available pages by clicking on the drop-down arrow for that field.

- For the Name field, use the drop down list box to find and select the object named Help. Click on OK to close the Object Name Selector dialog box.
- Leave the Condition Type field as is, set to Alphabet.

The Condition Type field identifies the kind of value that you are evaluating (alphabetical versus numeric).

The If Icon Content Editor appears as follows:


Content Editor

Icon Name: Help Button?

Condition 1: @_OBJECT_NAME

Test: EQ

Condition 2: Help

Condition Type: Alphabet

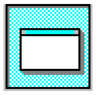
To review, if the user clicks on the Help button, this icon will evaluate to true and execution will flow downward. Otherwise, if the user interacted with another object, execution flows to the right.

▶ Click on OK to close the Content Editor.

Adding a Program Icon

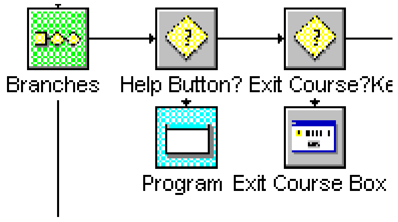
Next you'll add a Program icon just below the "Help Button?" If icon. This icon will run the external program called WINTUTOR.EXE when the user clicks on the Help button. WINTUTOR.EXE is an external program provides help on how to use Windows. IconAuthor allows you to run any Windows or DOS executable, such as WINTUTOR.EXE, from within an IconAuthor application. When the external program finishes, the icon below the Program icon is executed.

▶ Find the Program icon in the Icon library.

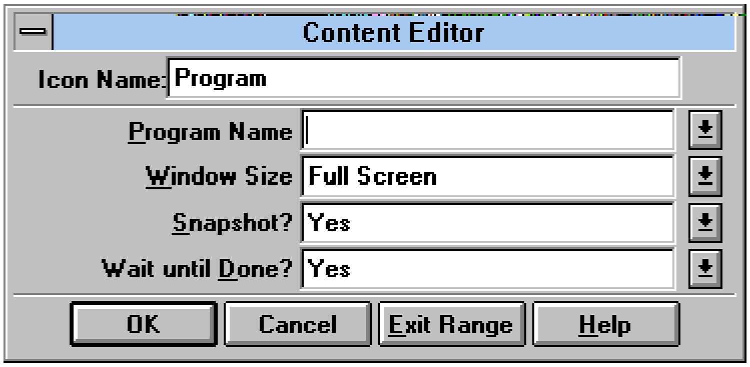


▶ Drag the Program icon directly below the "Help Button" If icon and drop it into place.

The application structure appears as follows:



- Double-click on the Program icon to open its Content Editor.



Content Editor

Icon Name: Program

Program Name:

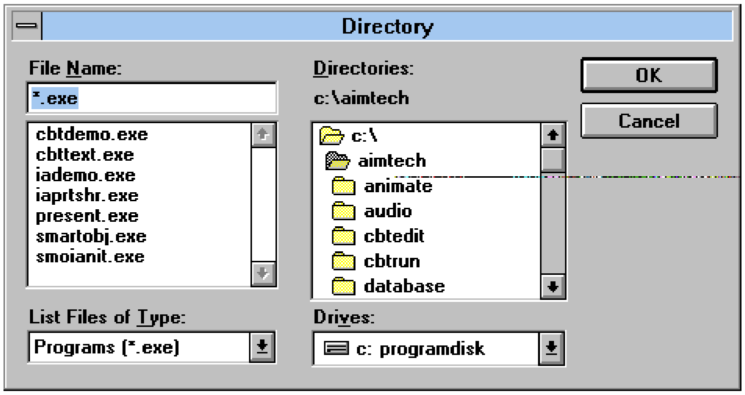
Window Size: Full Screen

Snapshot? Yes

Wait until Done? Yes

- Select Directory... from the drop-down list box in the Program Name field.

A Directory dialog box appears similar to the following:



Directory

File Name: *.exe

cbtdemo.exe
cbttext.exe
iademo.exe
iaprtshr.exe
present.exe
smartobj.exe
smoianit.exe

Directories: c:\aimtech

c:\
aimtech
animate
audio
cbtedit
cbtrun
database

List Files of Type: Programs (*.exe)

Drives: c: programdisk

The File Name field contains an *.EXE filter. By default the Directories list box is pointing to the ia_eval directory, where the Test Drive files were installed.

- Change to the Windows directory and highlight the filename WINTUTOR.EXE.
- Click on OK.

The dialog box is closed and the filename WINTUTOR.EXE is automatically entered in the Program Name field.

- Leave the Window Size value at Full Screen.

The Window Size field specifies the size of the window in which the IconAuthor application appears when the new program runs.

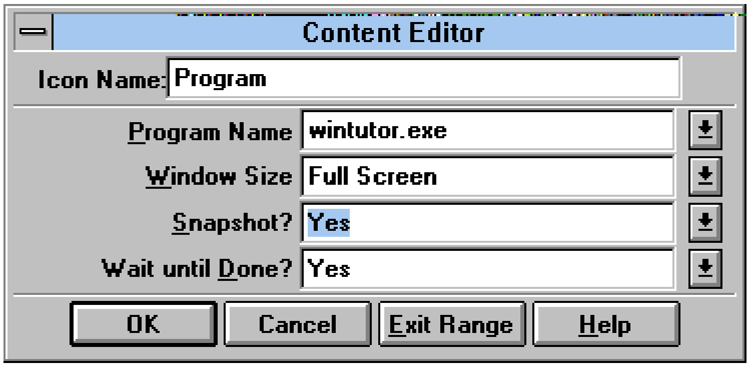
Note: This setting does not control the size and position of the new program you are running. These factors are controlled by the windowing system.

▶ Leave the Snapshot? value as is.

The Snapshot? field specifies whether you want the IconAuthor application context (the currently displayed information) to be restored after the user exits the program run by the Program icon.

▶ Leave the Wait until Done? value as is.

The Wait until Done? field specifies whether you want the IconAuthor application to continue executing while the new program is executing. The Program Icon Content Editor appears as follows:


Content Editor

Icon Name: Program

Program Name: wintutor.exe
Window Size: Full Screen
Snapshot?: Yes
Wait until Done?: Yes

This Program icon will run the Windows application WINTUTOR.EXE, while leaving the IconAuthor application in full screen mode. The IconAuthor application will not continue until the WINTUTOR.EXE application has been completed. Subsequently, the Main Menu screen will be restored.

▶ Click on OK.

▶ Choose Save from the File menu to save your work.

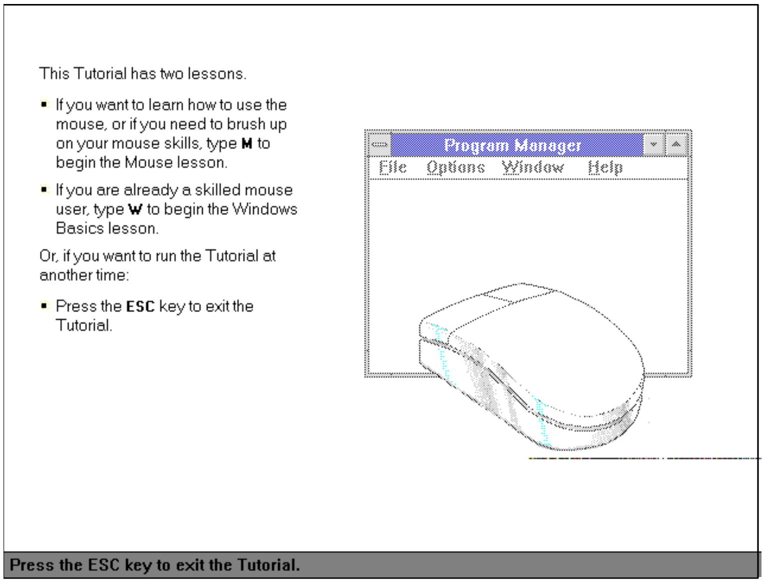
Now that you've completed the Help button logic, let's run the application to see how it works:

▶ Choose Application From Top from the Run menu.

Notice that the Main Menu displays the Help button that you added.

▶ Click on the Help button in the Main Menu.

The system executes the WINTUTOR.EXE application.



This Tutorial has two lessons.

- If you want to learn how to use the mouse, or if you need to brush up on your mouse skills, type **M** to begin the Mouse lesson.
- If you are already a skilled mouse user, type **W** to begin the Windows Basics lesson.

Or, if you want to run the Tutorial at another time:

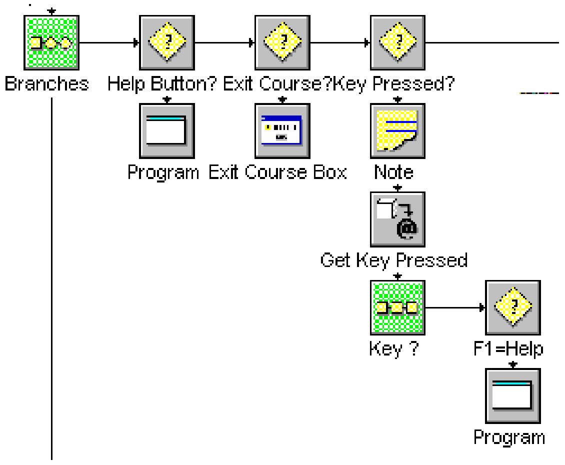
- Press the **ESC** key to exit the Tutorial.

Press the ESC key to exit the Tutorial.

- To exit online help press Esc.
- To exit the EVALBIKE.IWM click on the Exit Course option in the Main Menu.
- When the Message Box appears, click on OK.

The authoring window returns to view. The branch that runs the Help program is complete.

The last thing you need to check is whether your keyboard object is functioning properly. If you review your application structure, you will see that a branch responding to a key stroke is already in place. In this graphic, the Key? composite icon has been expanded so you can see all the icons in this structure.



Just like the Help button, this branch will launch the Windows Tutorial (WINTUTOR.EXE), however, it responds to keyboard rather than mouse input. By including a keyboard object, you have created an application that will respond to both.

You can test this functionality by running your application again.

- Choose Application From Top from the Run menu.
- At the Main Menu, hit the F1 key to run the Help application.

The system executes the WINTUTOR.EXE application. Notice that only the F1 key is enabled since that's the key you selected when setting up the keyboard object.

- To exit online help press Esc.
- To exit the EVALBIKE.IWM click on the Exit Course option in the Main Menu.
- When the Message Box appears, click on OK.

If something isn't working correctly, stop running the application (by pressing the ESC key) and review the instructions you followed up to this point. If you find that you completed an instruction improperly, make the necessary corrections. Then, run the application again to see whether you corrected the problem.

If you find you can't track down the source of the problem, open BIKE.IWM and compare it to your own application.

- Choose Save from File menu to save your changes.

Important: Do not continue to the next step of this Test Drive until you have successfully debugged EVALBIKE.IWM.

Step 4: Adding a Login Routine

In Step 4, you will add a login routine to the application by re-using one that was built for a similar application. To bring the login into your application, you will take advantage of IconAuthor's ability to open multiple applications at once, and to cut and paste between them.

Opening a Second Application File

To cut and paste icons between applications, you first need to open the second application from which you will be borrowing.

- From the IconAuthor top level menu, select File-Open.

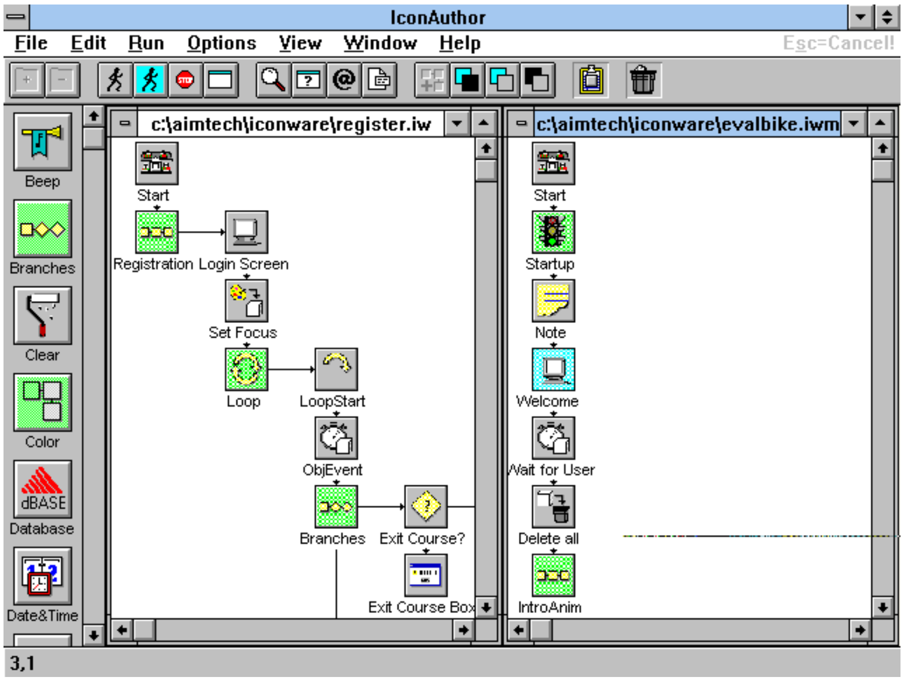
Make sure that you are looking in the C:\AIMTECH\ICONWARE directory.

- Select the file REGISTER.IW and select OK.

Your now have two applications open. Your next step will be to tile the applications so that you can easily see both of them at the same time.

- From the top level IconAuthor menu, select the Windows menu option, then choose the option Tile.

You should now be able to see both applications open as follows:



To copy icons from one application to another, we will use the IconAuthor clipboard. (Note: you can also drag and drop icons directly from one application window to another).

First, you need to select the icon or range of icons you want to copy. Selected icons are visually represented by their light blue color; icons which are not selected appear as gray (or green in the case of composite icons). When you select the Label Icon of a composite, all the icons in that module are selected.

When you drag and drop icons in IconAuthor, your cursor changes its appearance depending on whether you are over a valid drop area. When you are over a valid drop target, the icon appears as either a clipboard (if you are dragging and dropping from the clipboard) or a series of icons:

or

if your are not over a valid drop area, the icon looks like this:

▶ Click on the Registration Icon in REGISTER.IW file you just opened. The entire module should appear selected (light blue).

▶ Hold down the Control (Ctrl) key to indicate a copy (and not a move) operation.

▶ Drag and drop a copy of the registration module to the IconAuthor clipboard.

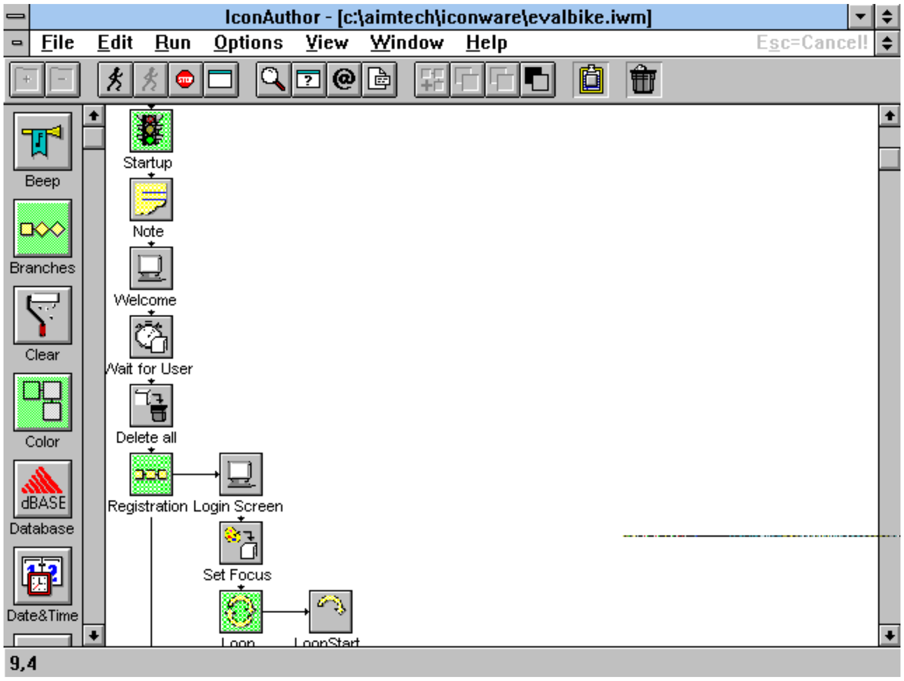
You have stored a copy of the register module to the clipboard. A clipboard which contains content appears as follows:

You can now drag and drop the icons in the clipboard. The clipboard is both useful for moving icons and structures between applications as well as to new locations within the same application. Before you attempt to drag and drop the Registration module, you should make sure that you can see where you want to put it.

▶ In the EVALBIKE.IWM file, scroll until you locate the IntroAnim icon.

▶ Drag from the IconAuthor clipboard and drop the registration immediately above the IntroAnim icon in the structure we are working on.

Your structure should now look as follows:



The Registration icon is a good example of a composite icon. Composite icons have two basic states: expanded and compressed. Your registration structure is currently fully expanded. Since it's taking up a fair amount of room on your screen, let's collapse it so it takes up less space. The button in the IconAuthor ribbon bar that allows you to compress a composite icon, looks like this:

▶ Collapse the Registration Module by clicking on the Collapse button in IconAuthor's ribbon bar (and if you click it again, the icon will expand).

Hint: sometimes composite icons contain other composites within them. When this is true, clicking on the expand button in the ribbon bar, only expands the top level composite in the icon structure. To open all the composites in the structure, hold down the shift key while clicking on the expand button.

While we won't spend any more time on composites in this tutorial, you should know that you can create your own custom composite icons, and even add those composites to your icon library.

▶ Choose Run from Top from the Application menu.

After the opening screen and animation, you should be presented with a registration screen. Go ahead and login and then click on OK. After registration, the application should move on to the introductory screens of the application. Verify that the student registration is working, and then save your work.

▶ Choose Save from the File menu.

One more step remains before you have completed your application. In step 5 of this tutorial, you will finish up by adding digital video to your application

Important: Do not continue to the next step of this Test Drive until you have successfully debugged EVALBIKE.IWM.

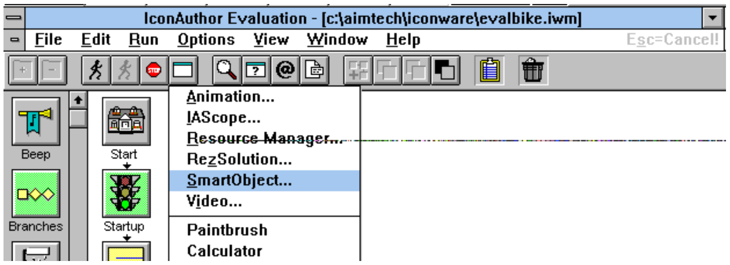
Step 5: Adding Multimedia to the Parts Screen

In this last step you'll see how easy it is to play a movie file in your application. You do this by adding a Movie object to a SmartObject file.

Unlike previous steps, when you worked on the Main Menu page, this time you'll be working with a page called Part Info in a SmartObject file called EVALPROD.SMT. This page displays a part of the branch that executes when the user clicks on the Product Knowledge item in the Main Menu. Don't worry about the precise part of the structure that displays the page. You won't need to add any icons to the structure to make the Movie object work. It will be completely set up within the SmartObject file.

Although in the past you ran the SmartObject Editor from within a Display icon, this time you'll learn to run it from the IconAuthor ribbon bar.

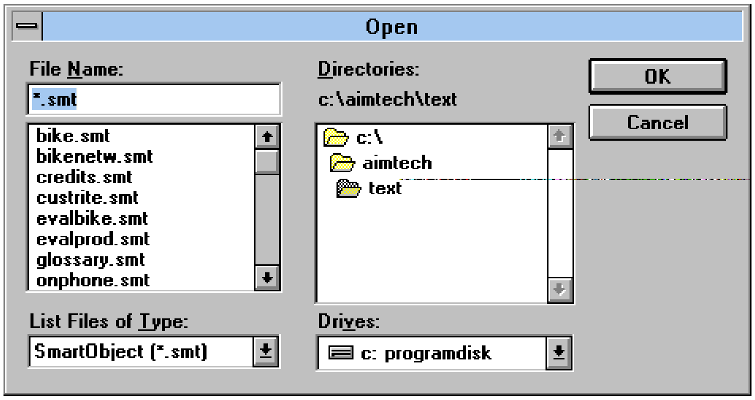
▶ Click on the Editors button in the IconAuthor ribbon bar and choose SmartObject... from the pull-down menu.



The SmartObject Editor appears. This time, it's up to you to open the appropriate file and page.

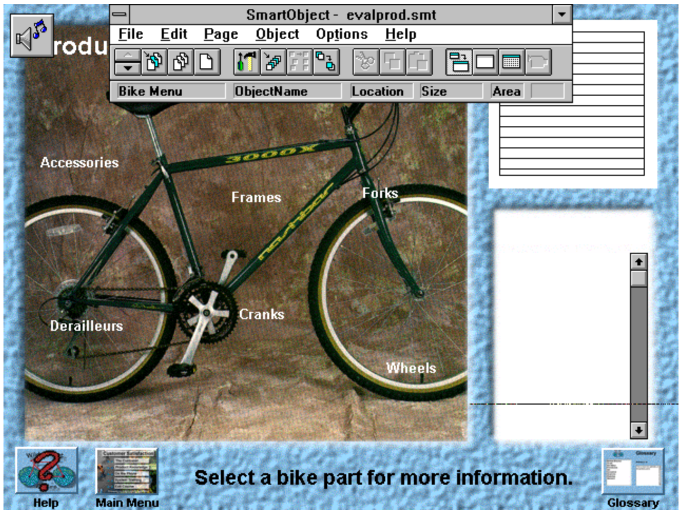
▶ Choose Open... from the File menu of the SmartObject Editor.

The Open dialog box appears.



▶ Click on EVALPROD.SMT in the File Name list box. Click on OK.

The EVALPROD.SMT SmartObject file displays as follows:

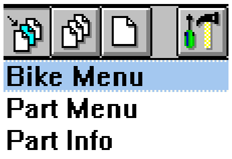


This is the page called Bike Menu. You'll use the Page Selector dialog box to show the Part Info page.

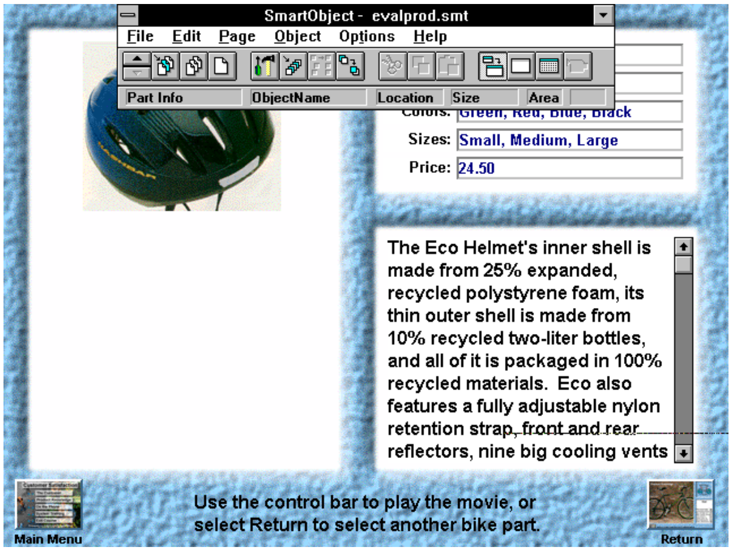
- Click on the Page Selector button or choose Go To from the Page menu.

The Page Selector menu appears.

- Select the Part Info page.



The Part Info page displays:



Colors: Green, Red, Blue, Black
Sizes: Small, Medium, Large
Price: 24.50

The Eco Helmet's inner shell is made from 25% expanded, recycled polystyrene foam, its thin outer shell is made from 10% recycled two-liter bottles, and all of it is packaged in 100% recycled materials. Eco also features a fully adjustable nylon retention strap, front and rear reflectors, nine big cooling vents

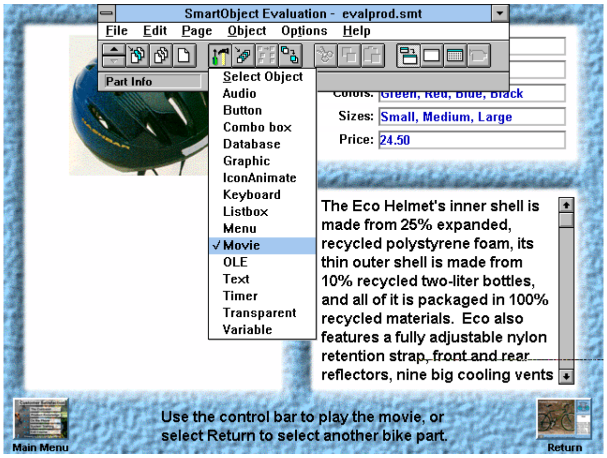
Use the control bar to play the movie, or select Return to select another bike part.

Adding a Movie Object

For your last task, you will see how easy it is to add a Movie object to the Part Info page. The object will have a control bar so that the user can play the movie at any time.

Use the Tools button in the ribbon bar to create a Movie object on the Part Info page.

▶ Click on the Tools button in the ribbon bar.



Colors: Green, Red, Blue, Black
Sizes: Small, Medium, Large
Price: 24.50

The Eco Helmet's inner shell is made from 25% expanded, recycled polystyrene foam, its thin outer shell is made from 10% recycled two-liter bottles, and all of it is packaged in 100% recycled materials. Eco also features a fully adjustable nylon retention strap, front and rear reflectors, nine big cooling vents

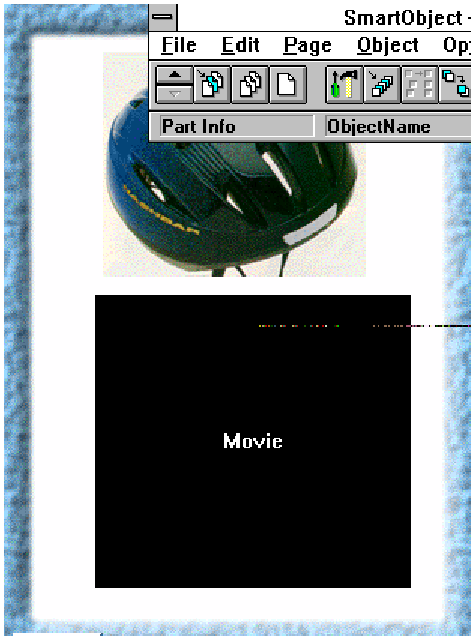
Use the control bar to play the movie, or select Return to select another bike part.

▶ Select Movie from the pull-down menu.

The cursor takes the shape of a cross-hair. You are now ready to draw the Movie object.

- Move the mouse in the white space directly under the helmet.
- Press and hold the left mouse button.
- Drag the mouse diagonally down and to the right to draw a square that is approximately 2”x 2”.
- Release the left mouse button.

The Movie object appears similar to the following:



Next you’ll set properties for this object so that it plays automatically at runtime.

- Click on the object with the right mouse button and select Properties...

The Movie Properties dialog box appears.

- Click on the CommandOnCreation property.
- Click on the drop-down arrow in the edit box and select Play.

The CommandOnCreation property executes an object’s command as soon as the page displays at runtime. In this case, you are instructing IconAuthor to automatically play the movie when the Part Info page displays.

Next you’ll use the FileName property to specify the particular movie you want to play.

- Scroll to the FileName property and select it.
- Click on the down-arrow to the right of the edit box.
- Click on Browser...

The Browser dialog box appears. The File Name list displays the available movie files. You'll be using a Video for Windows (.AVI) file.

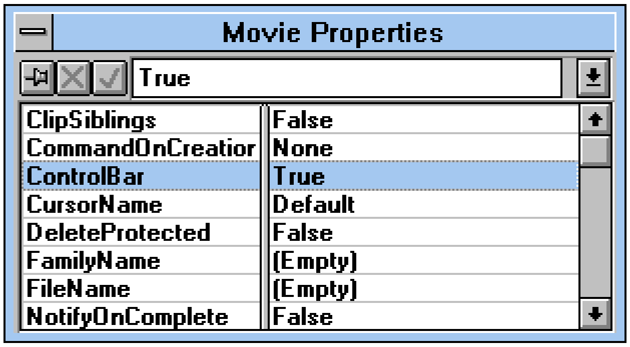
- Select the file ECOHELMT.AVI from the Movie subdirectory and click on OK.

The file name is automatically returned to the edit box in the Movie Properties dialog box. Note that despite the dimensions of the black box you drew, the movie object is automatically resized to the resolution of ECOHELMT.AVI. The ResizeToFile property is set to True by default which causes the movie object to automatically resize to the proper dimensions.

The last property you need to set is the ControlBar property. It adds a control bar to the object so that the user can play and pause the movie at will. The control bar also lets you (the author) preview the file within the SmartObject Editor.

- Scroll to the ControlBar property and click on it.
- Open the drop-down list in the edit box and click on True.

The Properties dialog box appears as follows:

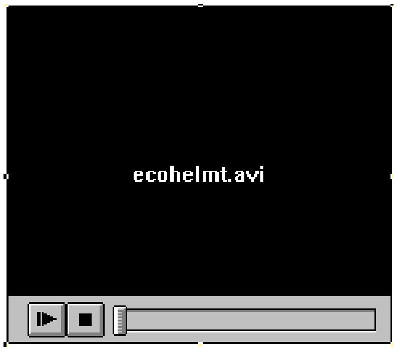

Movie Properties

True

ClipSiblings	False
CommandOnCreatior	None
ControlBar	True
CursorName	Default
DeleteProtected	False
FamilyName	(Empty)
FileName	(Empty)
NotifyOnComplete	False

- Click the left mouse button on the Movie object to close the Movie Properties dialog box.

The Movie object appears similar to the following:



▶ Choose Save from the File Menu.

Before exiting the editor, you'll preview the movie.

▶ Click on the Play button (the right-pointing arrow) to play the file.

To stop the movie, click on the square stop button. Use the scrollbar to move to a particular part of the movie before commencing play. The SmartObject file is complete.

▶ Choose Exit from the File Menu.

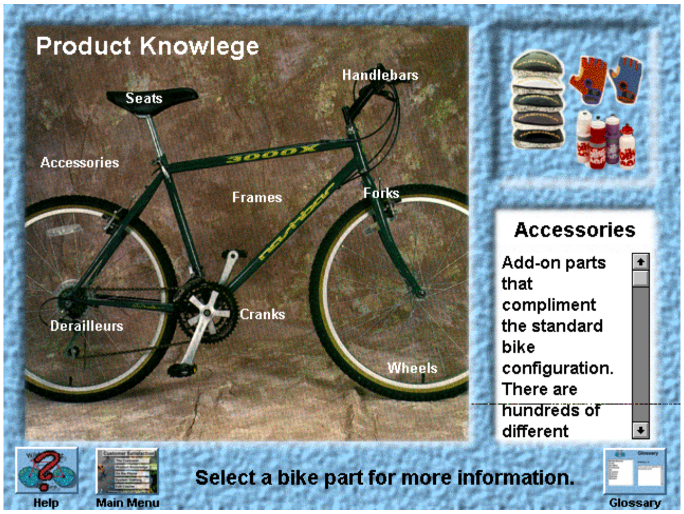
Now that you've completed the last step to the Test Drive, let's run the application to see how the video works.

▶ Choose Application From Top from the Run menu.

▶ Click on the Product Knowledge item in the Main Menu.

▶ Click on Accessories.

An Accessories graphic appears in the upper right-hand corner.



Accessories

Add-on parts that compliment the standard bike configuration. There are hundreds of different

Select a bike part for more information.

- Click on the Accessories graphic.
- Click on the picture of the helmets.

The movie plays automatically when the page displays. Because the Movie object has a control bar, you can play it repeatedly if you wish.

- Press Esc or Click on the Exit Course option in the Main Menu.

The authoring window returns to view.

Now that you've successfully completed the EVALBIKE.IWM application, you can further your development experiences with IconAuthor by completing the IconAuthor Tutorial. The Tutorial covers the fundamentals of IconAuthor and steps you through an entire development project, from the Start icon to the last subroutine. Once you've completed the IconAuthor Tutorial you will be well equipped to begin working on your own multimedia development projects in today's sophisticated, interactive learning and training environment.

IconAuthor and the Internet

Thousands of multimedia developers have taken advantage of the power of IconAuthor’s object authoring technology to create computer-based training, interactive presentations, self-service kiosks, electronic performance support, commercial titles, and more. Now it’s time to bring that power to the Internet and World Wide Web.

Now, Internet authoring with IconAuthor 7.0 takes you to a whole new level of interactivity without any programming or scripting. If you know how to build a flowchart, you can easily create interactive applications that leverage the power of the Internet. Multimedia authoring on the Internet, just got a whole lot easier.

With version 7.0, IconAuthor now has the ability to create interactive multimedia applications for delivery on the Internet. Authors can create applications that access content located on any server on the Internet anywhere in the world. In every place within IconAuthor that access an external file, users can now specify a location anywhere on the Internet.

Create Complete Web Applications

Internet authoring with IconAuthor goes beyond the creation of web pages; it allows developers to create interactive multimedia applications for delivery on the Internet. Applications developed with IconAuthor deliver a higher level of interactivity than traditional web authoring tools providing authors with ways to use items such as buttons, pull-down and pop-up menus, drag and drop, timers and combo-boxes in their applications. IconAuthor applications can contain “hotspots” that do more than just display HTML pages. IconAuthor’s hyperlinks can be used to display graphics, display text, start an audio or video clip or any other “action” created by the author. IconAuthor applications can be more than just an HTML page-turning application because it handles complex interactivity much better than today’s web browsers.

IconAuthor’s Internet support also gives developers the ability to provide a richer level of content in web applications. Special effects for text and graphics, complete 2D and 3D animation clips, and audio and video files are easily incorporated into IconAuthor applications.

IconAuthor also supports HTML. With the HTML object, authors can integrate existing web documents into their IconAuthor applications.

Applications developed with IconAuthor deliver a higher level of interactivity and richer level of content over traditional web authoring tools . You can now create a visually exciting custom interface by adding your own graphics, corporate colors and logos or hot buttons. Here’s just a few ideas to get you started:

- Deliver on the promise of electronic commerce. Design an electronic catalog of all your products and services
- Launch audio and video using our new “Hybrid Media” technique
- Add pull-down and pop-up menus to show lists of options
- Create fun and interactive “drag and drop exercises” and games
- Drop in new advertising messages on the bottom of your Home Page every 20 seconds

- Make text and graphics change color and move across the screen
- Create a buzz with 2D and 3D animations. Get your logo to entertain and inform
- Shuffle your web content to constantly display new product information
- Survey your customers online and capture their responses to a database
- Conduct product or remote sales training over the Internet
- Create "behind the firewall" information systems for internal use only
- Design an interactive HR kiosk that outlines recent changes in employee benefits

Our unique Universal Media Access enables your application's content files to be stored among multiple locations, whether the content resides locally on your hard drive, CD-ROM or LAN, or remotely on the Internet. This "hybrid media" capability helps you manage "thin media" vs. "fat media" and allows you to update content without the need to go back into IconAuthor. If you want faster playback than what today's web browsers can deliver, let IconAuthor's open architecture and object authoring technology increase application playback performance and decrease file transfer time over the Internet.

Running an Internet Application

Hybrid Application

The first Internet application you will run will use some files located on the Internet and other files that reside on your hard disk. IconAuthor gives you the flexibility to manage your application according to your needs; if you have a slow Internet connection (14,400 bps modem or less) you may not want to spend your time waiting to download every single graphic, audio and movie file. The sample application you are about to run in this section will let you decide what application files you want to come from the Internet.

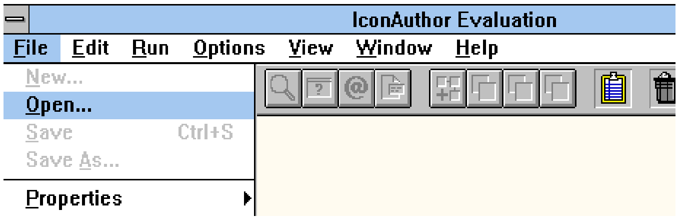
Note: IconAuthor does not provide your computer with a connection to the Internet. If you want to use the Internet capabilities of IconAuthor, your computer must be connected with the Internet. If you don't know whether your system is equipped to run Internet applications, please consult your System Administrator.

When you are connected to the Internet, you are ready to run our sample Internet application. The Wild Planet Cycles application will be run and you will tell IconAuthor where to get the content files.

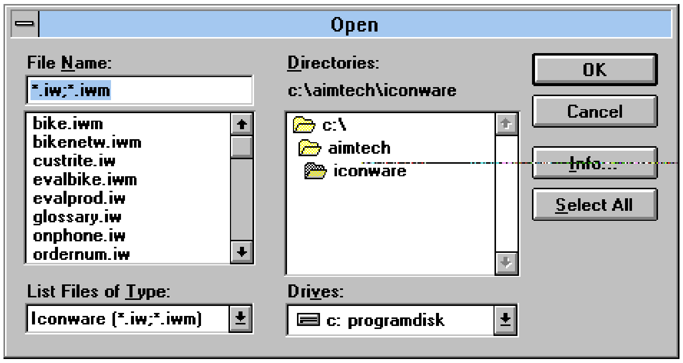
Let's begin by starting IconAuthor.

- Double-click on the IconAuthor icon in the AimTech Test Drive program group.

- Choose Open... from the File menu.



The Open dialog box appears similar to the following.



File Name: *.iw;*.iwm

bike.iwm
bikenetw.iwm
custrite.iw
evalbike.iwm
evalprod.iw
glossary.iw
onphone.iw
ordernum.iw

Directories: c:\aimtech\iconware

c:\
aimtech
iconware

List Files of Type: Iconware (*.iw;*.iwm)

Drives: c: programdisk

Note: You may get an error message about a missing file called BIKENETW.PTH when the file is opened. Click on OK to continue.

- Click on BIKENETW.IWM in the File Name list box. Click on OK.

To run this application, click on the Running Man icon from the IconAuthor Toolbar.

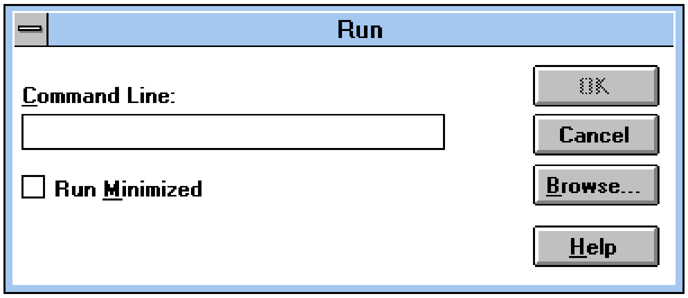
The first thing this application will do is check to see if the Internet connection to AimTech's web site is available. If the application can't connect with AimTech, you will have the option of exiting the application or proceeding. If you continue, all content files will be retrieved locally and this application will be no different than the one you just built. If the Internet connection is working, the application will ask you for the location of different content files needed to run the Wild Planet Cycles application. No changes were made to this application; the only change we are making is telling IconAuthor where to get the files it needs to run this application. We've put a copy of this application on AimTech's web site, so when you choose to have the content of this application come from the Internet, it is being displayed from our web site in Nashua, NH USA.

Note: To hear audio, this sample application assumes the Test Drive CD-ROM is in a local CD-ROM drive. For users with "slow" Internet access (14,400 bps modems or less), choose only the IconAuthor application files and Text files for download for maximum performance. See Table A for a list of estimated transfer times for the Internet.

Complete Internet Application

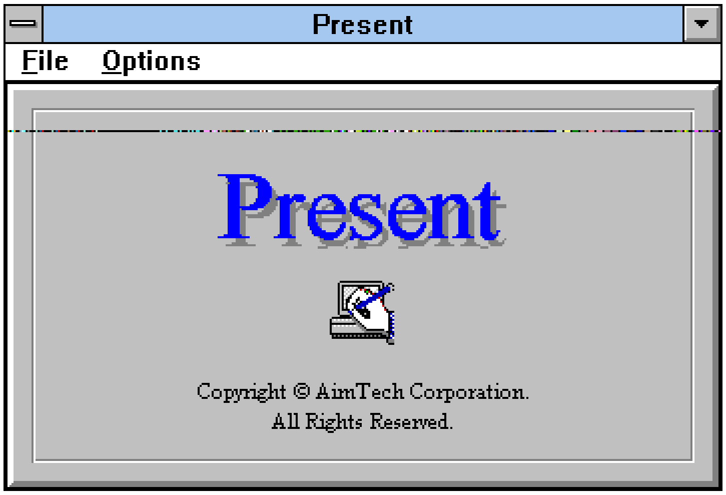
IconAuthor applications can also be run entirely from the Internet. We've created another sample application and put it on our web server. To run this application, you will use the Runtime version of IconAuthor, called Present.

- Choose Exit... from the File Menu to close IconAuthor.
- Choose Run... from the File Menu of the Windows Program Manager.



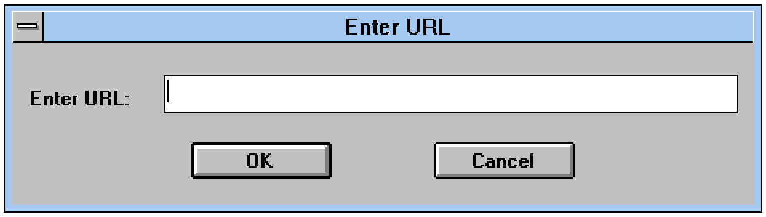
- Type C:\AIMTECH\PRESENT.EXE in the Command Line field. Or browse to the location where you installed the Test Drive.

IconAuthor Present should appear similar to the following:



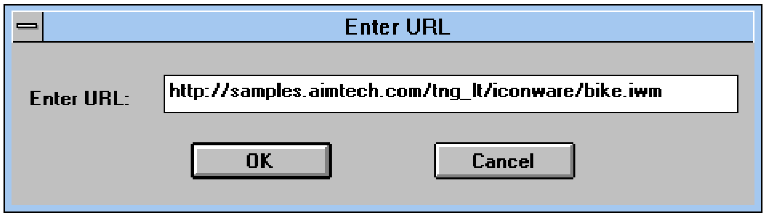
You will now tell IconAuthor where to find the sample application. This location can be a local hard disk, CD-ROM, Local Area Network or anywhere on the Internet.

▶ Choose Open URL... from the File Menu to open the sample application.



A Uniform Resource Locator or URL is the standard way of representing Internet locations for files. For this sample application, the file is located on AimTech's web server in Nashua, NH.

▶ Type http://samples.aimtech.com/tng_lt/iconware/bike.iwm and click the OK button.


Enter URL

Enter URL: http://samples.aimtech.com/tng_lt/iconware/bike.iwm

If your Internet connection is configured properly, the first screen of this application should appear shortly.

Follow the instructions on the screen to go through the application.

For more sample applications and the latest information on IconAuthor 7.0 and its Internet capabilities, be sure to check in at AimTech's Web Site at http://www.aimtech.com.

Table A
Estimated File Transfer Times Over the Internet

Multimedia Content on the Internet

Transfer times for different Internet Access Methods

Content Files	Size (bytes)	14.4K	28.8K	T1 or Single speed CD-ROM
Small graphics	30 K	20 secs	10 secs	<1 second
Large graphics	100-200 K	1-2 mins	30-60 secs	1 second
5 second audio (8 bits, 22 Khz, Mono)	100 K	1 min	30 secs	1 second
5 second audio (16 bits, 22 Khz, Mono)	200 K	2 mins	60 secs	2 seconds
10 second video clip (15 fps)	1,000 K (1 MB)	8-12 mins	4-6 mins	6 seconds