

---

# **Dazzler Deluxe 3.1**

## **User Guide**

---



---

# Contents

**DAZZLER DELUXE 3.1**

---

**CONTENTS**

---

**ACKNOWLEDGEMENTS** **1**

**CONTACTS** **3**

**WELCOME** **5**

**ABOUT YOUR DAZZLER DOCUMENTATION** **5**  
**NEW FEATURES OF DAZZLER DELUXE 3.1** **5**

**GETTING STARTED** **9**

**SYSTEM REQUIREMENTS** **9**  
**INSTALLING THE SOFTWARE** **10**  
**TESTING YOUR HARDWARE CONFIGURATION** **10**

---

---

**DAZZLER BASICS** **11**

<b>THE DAZZLER DEVELOPMENT ENVIRONMENT</b>	<b>11</b>
<b>HOW TO BUILD A COURSE</b>	<b>12</b>
WHAT ARE EVENTS?	15
WHAT ARE ACTIONS?	16
WHAT ARE RESPONSES?	21
DAZZLER RUN-TIME	23
ANALYSIS MODULE	24

---

**TUTORIALS** **25**

<b>A PICTURE, TEXT AND A BUTTON (PICTBUTN.DZL)</b>	<b>27</b>
<b>ASKING A QUESTION (QUEST.DZL)</b>	<b>47</b>
<b>GETTING AND DISPLAYING DATA</b>	<b>55</b>
<b>PICTURES AND HOT SPOTS (HOTSPOT.DZL)</b>	<b>63</b>
<b>ADDING VOICE OVER USING THE SOUND ACTION</b>	<b>67</b>
<b>CREATING A SIMPLE GRAPH (GRAPH.DZL)</b>	<b>69</b>
<b>ADDING DIGITAL VIDEO</b>	<b>79</b>
<b>VARIABLES (VARS.DZL)</b>	<b>85</b>
<b>MULTIPLE ANSWER QUESTIONS (MULTICH.DZL)</b>	<b>89</b>
<b>SCROLLING TEXT BOXES (SCRLTEXT.DZL)</b>	<b>95</b>
<b>RANDOM SELECTION (RANDOM.DZL)</b>	<b>100</b>
<b>SLIDE SHOW (SLIDEVAR.DZL)</b>	<b>107</b>
<b>USING VARIABLES TO RUN COURSE MODULES (RUNVAR.DZL)</b>	<b>113</b>
<b>USING VARIABLE FUNCTIONS (VARFILE.DZL)</b>	<b>117</b>
<b>PLAYING SOUND FILES WITH GRAPHICS EFFECTS</b>	<b>127</b>
<b>VIDEO TAPE AND DISK (VIDTAPE.DZL)</b>	<b>129</b>

---

---

# Acknowledgements

Copyright © 1996-1999 Intelamedia Ltd.

No part of this book may be reproduced without prior written permission.

Microsoft is a registered trademark and Windows is a trademark of Microsoft Corporation. DVI is a registered trademark of Intel Corporation. Kodak and Photo CD are trademarks of Eastman Kodak Company. Autodesk Animator is a registered trademark of Autodesk Inc. Tiff is a trademark of Microsoft Corporation

We are dedicated to improving and enhancing the hardware and software of our typesetting and communications systems and equipment. Consequently, the information in this manual is subject to change without notice. Reference is made to this fact during training courses.

The contents of this documentation are correct at the time of going to press.

The information contained in this manual about performance, speed as well as technical data concerning application of our products is not legally binding as it does not constitute a written contract of features.

We point out that companies, trademarks and product names mentioned in this manual fall within the regulations regarding protection of trademarks and patents.

Other product names and brands not expressly mentioned in this manual are trademarks or registered trademarks of the corresponding manufacturers.



---

# Contacts

Intelamedia Ltd.  
Claire House  
Bridge Street  
Leatherhead  
Surrey  
KT22 8BZ  
United Kingdom  
sales@dazzlersoft.com

QMark Systems  
PO Box 333  
Albion  
Queensland  
4010  
Australia  
info@qmark.com.au



---

# Welcome

Welcome to Dazzler Deluxe, the computer program designed for people who don't have time to program computers, yet who still want to create training courses and presentations which are as eye-catching and effective as the latest multi-media technology will allow. Dazzler Deluxe records results too, giving feedback to both student and trainer.

---

## About your Dazzler Documentation

This manual is divided into two parts, of which this is the first, outlining Dazzler and its features in general.

The second part is devoted to tutorials showing how to put together different kinds of courses. The Reference Manual is separate, and is also available on-line, offering a detailed description of each component of Dazzler.

---

## New Features of Dazzler Deluxe 3.1

Dazzler is designed to help you put together training courses and presentations in the shortest possible time without the need to learn a programming or authoring language. This version of Dazzler runs under Microsoft Windows 3.1 (or higher) and is designed to make the most of its features. Simply "clicking and dragging" with the mouse is all that is needed to define the tasks making up a course. Text, drawings, pictures, sound and video can all be used. When the student presses buttons on the screen in answer to a question, the replies can be logged as 'right' or 'wrong' for an automatic training report to be generated at the end of the course. Version 3.1 of Dazzler has lots of new features to help trainers create the most effective training courses:

- Full editing in the Presentation Window. Any object can be modified when the course is being built. Just click on the button, picture, text, or any other object, and move it around. Double-click to change its attributes, without having to go back to the course window. Duplicate any object from within the Presentation Window. In this way, buttons can be copied and automatically aligned with lightning speed.
- Responses can be linked directly to tasks by clicking and dragging the right hand mouse button in the Course Window. This saves time - you don't have to go to the setup dialog box of each response and select task tag numbers from the list of tasks.
- Pictures, Text and Graphics Shapes can all have effects added to them. So text can appear through an effect without having to create a bitmap of it first.
- Any object can be animated - text, pictures, and even video, can be moved along paths to create visual impact. Bezier curves can be created for the animation path as well.
- Improved button responses. Pictures can be put on buttons; full control of text, button colour, shading and depth is available.
- User 'drag and drop' response. Any object appearing in the Presentation Window can be dragged around the screen by the student. When the course is run, the user can pick up a picture of a test instrument, for example, and move it around a background picture of the equipment to be tested. Release the mouse and be scored on whether the test was done in the correct place.
- List box and combo-box responses - let the user select from a list of choices you define.
- Full control of menus. A new menu action gives you full control over the menu bar. The menu response lets you detect which menu option has been chosen by the user and to branch accordingly.
- The functionality of Dazzler Deluxe can now be extended through support for Dynamic Link Libraries (DLLs). In this way, objects and functions not already in Dazzler can be added with Microsoft Visual Basic and C++.
- OLE 2.0 action. Objects from other applications, such as graphics packages, spreadsheets and word processors can now be directly embedded in the Dazzler Presentation Window using Microsoft's new Object Linking and Embedding 2.0 architecture. Presentations and courses can be built which show spreadsheet charts, for example. Clicking the chart will bring up the menus and button bars for the

application which created the object so that the user can manipulate it directly if required.

- Greater graphic control. Objects can be layered in the Presentation Window. A new Palette action allows control over which palette is active at any time, eliminating the palette flashes which can occur when running in a 256 colour mode.
- Full file read and write. In addition, Dazzler Deluxe supports database access through Microsoft's ODBC (Open Data Base Connectivity) standard. This allows Dazzler courses to access any database which supports ODBC, such as Access, FoxPro, SQL Server, etc.
- Full support for the improved MCI Digital Video standards, including the MPEG subset.
- Preloading of graphics images - eliminate loading delays at time-critical points in a course or presentation.
- Greater grid flexibility - define horizontal and vertical size separately and view the grids on screen as you align objects, if you wish.
- A Debug trace to see which tasks were called when the course was run. Get a better idea of how your courses are structured.



---

# Getting Started

---

## System Requirements

Dazzler will run on any IBM PC compatible computer which runs Windows 3.1 or higher. This usually means at least a 386 type processor with 4MB of RAM. If you intend to do a lot of fractal compression and image manipulation, the faster the machine the better. Dazzler itself requires about 2.0 MB of hard disk space for the developer system. However, pictures, sound and particularly digital video files can take up substantially more than this, especially while developing courses, so a large hard-disk is always a good idea.

Dazzler doesn't need any additional hardware to be able to create effective courses. Nevertheless, it lets you take advantage of many of the emerging standard technologies to enhance your courses. The following items can be added:

- CD ROM drive for reading Kodak Photo CDs and for playing back courses distributed on CD. Make sure that the drive is Kodak Photo CD 'multi-session' (or at least 'single-session') compatible.
- Graphics card supporting more than the usual VGA 16 colours - for instance 256 colours. The more colours you can show on the screen, the more realistic photographs will look. Many computers now come with graphics cards capable of higher resolutions, but Windows may only be configured for VGA - consult your computer manual for details. Otherwise, one of many graphics cards can be installed in a spare slot.
- Windows compatible sound card, such as a SoundBlaster, for recording voice-overs and sound effects, and playing them back in your courses from the computer disk.

- Digital Video card, such as those based on the Video for Windows, MJPEG or MPEG standards, for recording and playback of video from the computer disk.
- CD Recording system for creating your own CD-ROMs.
- Video overlay card displaying live video from a video tape or camera within your presentations.
- Computer controllable VCR or video disk player, for adding video material at precise points in your courses.

---

## Installing the software

### *From Diskettes*

Insert the Dazzler Program diskette in a floppy drive of your computer with Windows running. From Program Manager select *Run* from the *File* menu and type in **A:SETUP** or **B:SETUP** depending on which drive the diskette is in. You will then be taken through the simple installation steps. If all is well, you will get a message saying that the installation has been completed successfully.

### *From CD*

The process is similar to that above, but you will need to run Setup from the drive associated with your CD.

---

## Testing your hardware configuration

As discussed above, Dazzler can be used with a wide range of hardware add-ons. Bringing them all together in a course can seem daunting, so it is best to check each item individually before attempting to use them within Dazzler. This will also help us in providing technical support to you. For example, if you have an audio card, test it out with the Sound Recorder which comes with Windows 3.1 in the Accessories group. Refer to the instructions which came with your hardware items for more details.

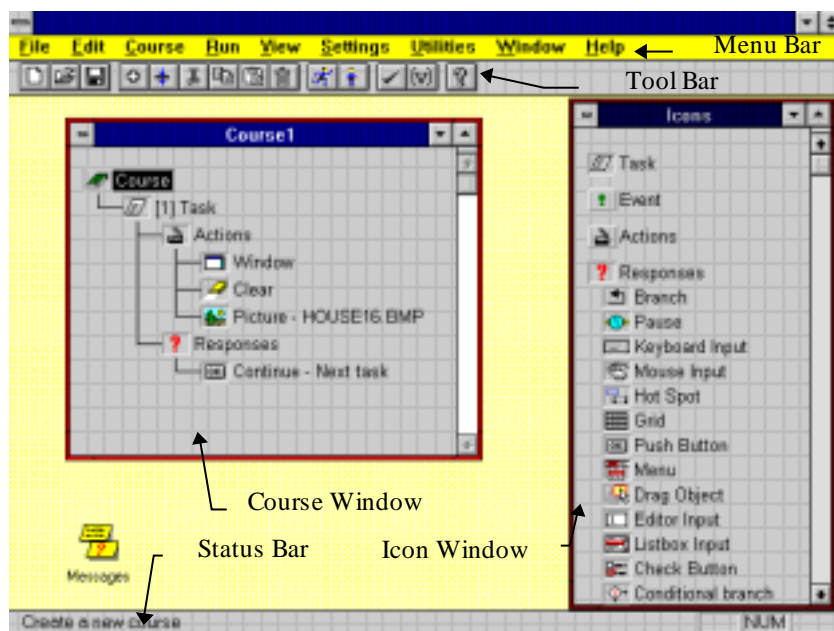
---

# Dazzler Basics

---

## The Dazzler Development Environment

To start Dazzler, just double click on its Program Manager icon in the usual Windows way. The key features of the Development Environment are as follows:

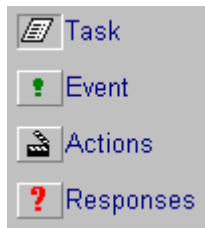



- The **menu bar** contains all the tools for opening, checking, running and saving courses, setting preferences and arranging the interface.
- The **tool bar** is a short cut to some of the more frequently used menu options.
- The **icon window** shows pre-packaged functions ready for incorporation into your course - just drag them to the **course window**. The icon window can also be configured to show only the icons you find most useful.
- Several **course windows** can be opened at one time, so that parts of one course can be copied to another, or several related courses can be developed at the same time and inter-linked.
- The **status bar** indicates both the progress of the saving and loading of courses and gives a brief description of each menu option as the mouse is dragged over it.
- The **presentation window** shows the course as the student will see it.

---

## How to Build a Course

At the heart of Dazzler is the idea that a course consist of a series of tasks - things which will happen during the course. A task consists of up to three parts - events, actions and responses. These act as the building blocks of a course and are shown in the icon window. The event, action and response icons in the Icon Window can be expanded to reveal the list of ones available by clicking once on the appropriate icon in the icon window, like a button (e.g., clicking the 'Action' icon shows the possible actions as icons below it).




Start a new course by selecting New from the File Menu or by selecting the  button from the button bar. A new course window will appear on the left hand side of the Dazzler window.

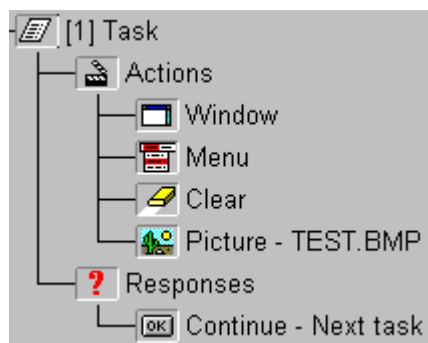
It will initially contain a course with one task, showing its action and response icons. The action icon in turn will contain three icons usually needed to set up a course - Window, Clear and Menu.

If your computer is set up to run in a 256 colour mode, there will be one additional icon already inserted, the palette action.

To add another task to a course, click on the *Task* icon in the icon window. Hold down the left mouse button and drag it to the *Course* icon in the course window. The new task will be added to the end of the course. Drop the task on top of an existing task to insert the new one before it.

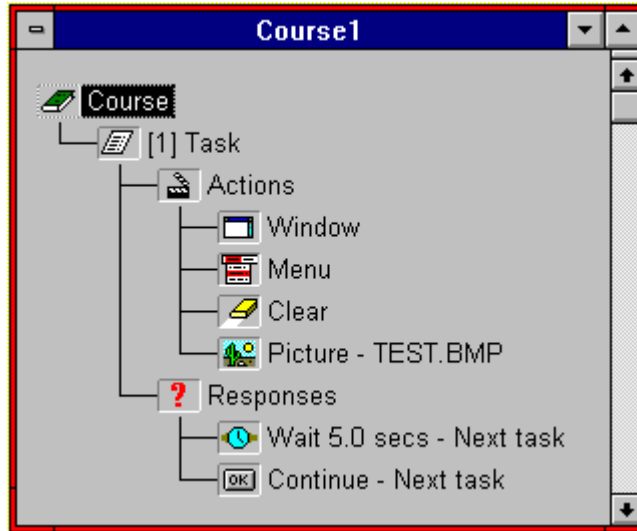
Icons can show or hide their detail. This is shown as a raised on indented icon button.

For example, if a task is shown like this:  [1] Task, click on the icon to reveal its structure :

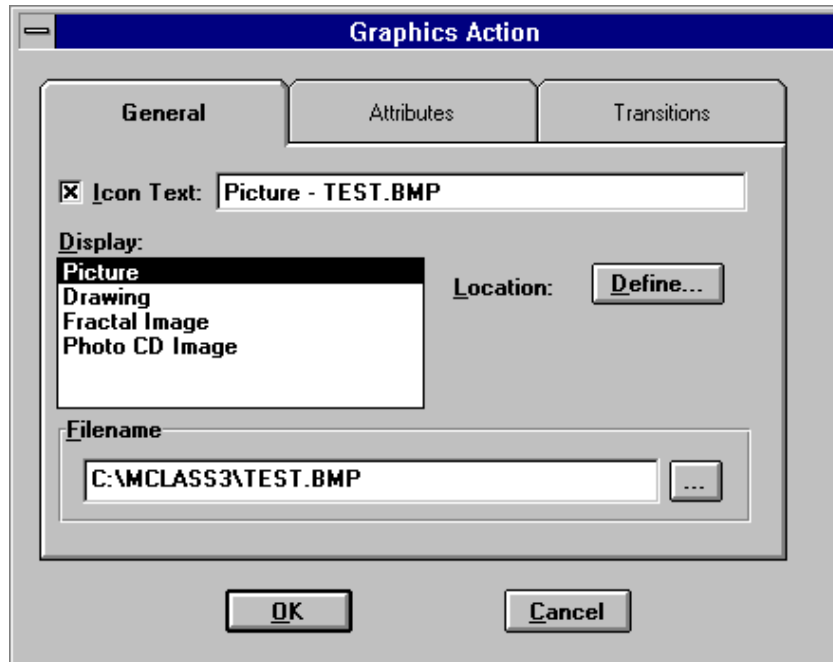


The two basic activities in building a course are as follows:

Click, drag and drop tasks, actions and so on, into your course window. You can have several actions and responses in one task.



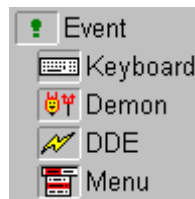
Double click on any action or response to define what it is going to do. A dialog box appears showing what options are available for that particular object. For example, double-clicking on a graphics action in a course brings up the following set-up dialog box:



Refer to the Tutorial section for a step-by-step guide to building courses.

## What are Events?

Events are used when it is necessary for part of the course to be activated by some occurrence - a message from another Windows application or a variable defined in the course reaching a certain value, or a menu option being selected by a student. Events deliberately interrupt the flow of the course.



If a task has a Menu event, then when the student selects the particular menu option in a course, this task will be carried out. Similarly with the Keyboard event, pressing a particular key, such as the Escape key, can interrupt the flow of the course and trigger the task to which it is attached.

Demons and DDE events are advanced features. Demons detect a combination of variable values being reached - for instance, a student reaching a particular score during the course. DDE events trigger their associated tasks when a particular message is received from another Windows application. In this way, training courses can be integrated into any application which supports DDE.

Since events are used infrequently, they do not automatically appear in the structure of the task. However, dragging and dropping an event icon onto the task will add it to the structure.

### **What are Actions?**

Actions are the most visible part of Dazzler. They show things in the Presentation Window, from simple text through to digital video. Despite the range of actions available, they are all set-up in the same way - drag an action from the icon window and drop it onto the appropriate task. Then double click on the action icon in the course and a dialog box appears showing the unique characteristics of that action - the position of the picture, its file name, etc. You can have several actions in one task. They will be performed in the order in which they are shown in the Course Window, top to bottom. They will all be carried out before any responses attached to the task are performed.



Here is a brief description of what the actions do. A fuller description, and step by step instructions on how to set them up, can be found in the Reference Manual.

**Window**

Use this action to define the size of the window the presentation will use, whether it runs full screen, has a menu bar, and so on.

**Menu**

Creates and modifies a menu bar in the Presentation Window.

**Clear**

Clears the contents of all, or part of, the Presentation Window.

**Graphics Image**

This action is used for displaying pictures and drawings, stored in a variety of formats (such as bitmaps) on the computer's disk, or Kodak PhotoCD from a compact disk. You can define where in the Presentation Window the picture will appear and also any effects such as wipes, dissolves, etc.

**Text**

This action displays text in any font colour and size you choose. The text can either be entered into the action itself, or read from a file.

**Graphics Shape**

Sometimes it is useful to be able to add a box, line or border to a screen without having to create a picture file for it. This is what the Graphics Shape action allows you to do, as well as defining styles and colours for the lines and boxes.

**Widget**

A sequence of pictures can be shown for cartoon-style animation. The time taken to complete one cycle of the pictures in the sequence is specified in seconds and fractions of seconds.

**Movies**

Digital video stored on computer disk can be played back using this option. Microsoft's Video for Windows, Intel's DVI, MJPEG and MPEG formats are all supported, as well as Autodesk Animator and Animation Works Interactive animations.

**OLE 2**

Dazzler 3 allows you to embed information such as pictures and text from other Windows applications which support OLE2. For example, a Microsoft Word Document can be added to the Presentation Window, complete with transition effects, as if it were a "native" Dazzler object

**Move Object**

Any action object - Picture, Text, Graphics Shape, Widget and even a Movie File can be moved along a path.

### **Palette**

The palette action's main role is to help overcome problems in the way in which Windows handles 256 colour pictures. It controls how the colours in a picture are chosen.

### **Sound**

With a Windows-compatible sound card (see System Requirements) you can play back sound files from your computer disk. In addition, you can select tracks from an audio CD in a CD-ROM drive or play MIDI music files if you have a compatible device.

### **Run Another Course**

Courses can be built as a series of modules stored as individual Dazzler courses. They can then be linked together by this action. This helps in the construction and maintenance of large courses.

### **Print**

Anything displayed in the Presentation Window can be printed out using this command.

### **Live Video**

If you have a video overlay card to show the signal from a VCR or camera on your computer screen, then this action can be used to position the video in the presentation window.

### **Tape Control**

If you have a video tape recorder or video disk player which has MCI software support, you can control it through this action. For instance, going to a particular part of the tape and going to 'play'. In conjunction with the 'Video' action, the video picture can then be shown to the student.

### **DLL Function**

Dynamic Link Libraries, or DLLs, are computer programs containing specific functions which can be called from other programs whenever they are needed. These can be accessed directly from Dazzler using the DLL Action.

### **DDE**

Dynamic Data Exchange is a powerful way for Windows programs to communicate with each other. Thus, from this command you can ask an application such as Microsoft Excel to load up a particular spreadsheet and carry out certain actions. Usually the whole 'macro' language of the package being controlled can be used as DDE commands. Refer to the application's reference manual for details of its DDE command set.

### **Launch Another Application**

This allows you to run another program at certain points in the course. For example, the Windows calculator could be called up when the student has calculations to perform during the course. DOS programs can be run as well.

### **Animation**

Pictures can be moved along paths around the screen with this action.

### **MCI Command**

Microsoft's Multimedia Control Interface is a general way of issuing commands to perform multimedia functions. Usually Dazzler does all the work of using these commands for you - the Actions take care of it. However, this action is available if you want to issue MCI instructions directly from the course yourself.

### **Variables/Functions**

As an advanced feature, variables can be used in a variety of ways. Sophisticated scoring schemes can be created by assigning different values to variables depending on which answer the student selects.

### **ODBC Action**

If you have the Database Toolkit, available separately, you can link Dazzler to any database supported by Microsoft's ODBC standard.

### **Write To File**

Dazzler keeps track of basic scoring automatically. This can then be used by the Dazzler Analysis Module. However, variable values can also be written out to a file using this action, for use in other packages, such as spreadsheets.

### **Read From File**

Text can be read from a file and stored in a variable for processing and display.

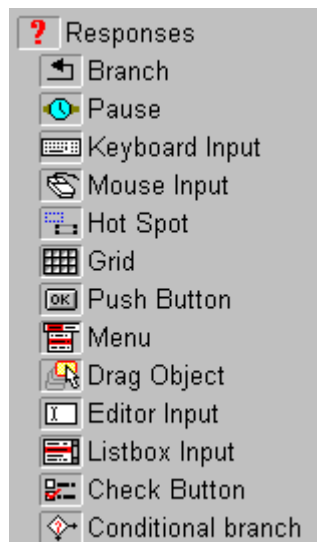
### **Simulator**

If you have the TQC Process Simulator Link Module, you can setup process control parameters and goals from this action and run the simulations themselves.

## What are Responses?

Having let the actions build the screen to show to the student (for example, a picture with some text on it), the responses are then used to specify how the student is to respond to those actions. This can include, for example, pressing a button on the screen to move to the next picture, typing answers to questions displayed, etc. Each response is linked to another task which will be carried out when that response is activated. Buttons and Hot Spots can also be assigned 'right' and 'wrong' for logging student results.

A task can have several responses. The one which is carried out is the one whose conditions are met first. For example, if there is a Pause response and a User Input response attached to the same task, if the time period defined in the Pause expires before the user has pressed the specified key, the course will branch to the task specified in the Pause response, rather than the one specified in the User Input one.



Here is a brief description of what the responses do. A fuller description, and step by step instructions on how to set them up, can be found in the Reference Manual.

### **Branch**

Use this to go immediately to another part of the course without waiting for a response from the student.

### **Pause**

Wait for a period of time, defined in seconds, before moving on to another task.

### **Keyboard**

Wait for the student to press any, or a particular key.

### **Mouse**

Wait for the student to click the mouse button.

### **Hot Spot**

Use this response to create rectangular hot-spots in the Presentation Window. If the student clicks the mouse on this spot, the course will move on to the task defined in the response. In this way, photographs can be made interactive, for example. Clicks in a region can be logged as 'right' or 'wrong' for student scoring.

### **Grid**

If many Hot Spots are needed (for instance on top of a map), the grid response provides an easy way to do this. A rectangle can be defined and divided into a number of rows and columns. Each element of the grid, when clicked on, can lead to a different task in the course.

### **Push Button**

This is one of the most commonly used responses. It places a button with text at a point on the screen. When the student presses it, the course moves on to the specified task. Button pushes can also be logged as 'right' or 'wrong' for student scoring.

### **Menu**

Trigger the response when the user selects a particular menu item.

### **Drag Object**

Using this response, the user can drag any object around the screen. A *Drop Area* can be defined which will trigger the branching specified in the response.

### **Editor Input**

This displays a standard Windows edit box on the screen so that the student can type in answers which can be assigned to variables for later use.

### **Listbox Input**

This provides a way of letting users choose from a list box or drop-down combo box.

### **Check Button**

Check boxes, or radio buttons, can be used to allow the student to select one or more possible answers.

### **Conditional Branch**

This response checks the value of variables, such as student score, and will branch to a different part of the course if a particular expression is true.

### **Frame Location**

This response is used in conjunction with the Tape Control action to wait for the tape or disk to reach a particular point.

### **Notification**

This is used in conjunction with certain actions, such as Movies, Animation and MCI, to wait for the action to complete before moving on to another task.

## **Dazzler Run-time**

Once you have created a course using the Development Environment, it can then be run directly from an icon in Program Manager using the Run-time module, DZLRUN.EXE. Create an icon in a Program Manager group pointing to DZLRUN and taking as a parameter the full path name of your course, e.g.

```
\DAZZLER\DZLRUN.EXE C:\DAZZLER\COURSES\MYCOURSE.DZL
```

If you are using the optional Dazzler Desktop application to replace Program Manager, refer to the instructions given for that application.

## **Analysis Module**

The Dazzler Analysis module allows trainers to examine the data logged in data files by Dazzler courses. It provides analysis by both student and by course, with examination of time taken to answer questions, and how many times the student got a question right and wrong. Results can be displayed and printed in both graphical and tabular form.

---

# Tutorials

The best way to learn how to build courses using Dazzler is to try it for yourself. The tutorials described here can be installed via the Tutorial option in the Dazzler installation procedure and are designed to serve two purposes:

- to teach how to use the features of Dazzler.
- to provide a series of templates which can be copied into your courses, reducing the time needed to create them.

It is assumed that you are comfortable with using Microsoft Windows. The first tutorial goes into more depth about the individual steps required to create a course. It is therefore suggested that you at least run through this first tutorial if you are new to using Dazzler.




---

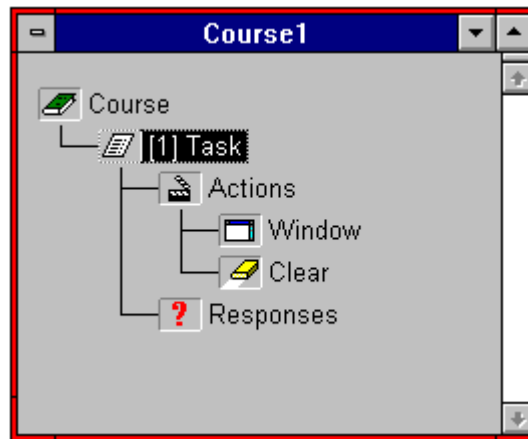
## A Picture, Text and a Button (PICTBUTN.DZL)

### Description

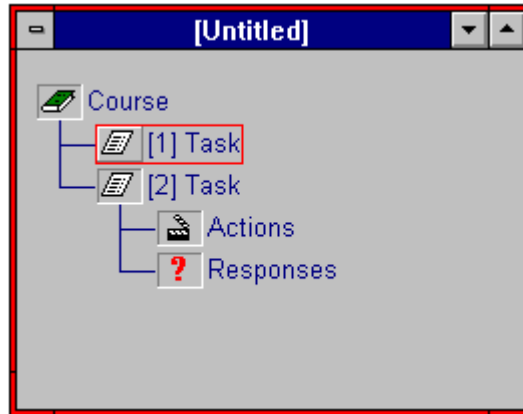
The first simple 'course' we'll build is designed to show the student a photograph with some text on top of it. A button will be added to the bottom of the picture. When the student presses this button, the course will end.

### Procedure



Start a new course, either by pressing the new course button  or by selecting *New* from the *File* menu. An untitled *Course Window* will appear with a *Course* icon and one *Task* icon:

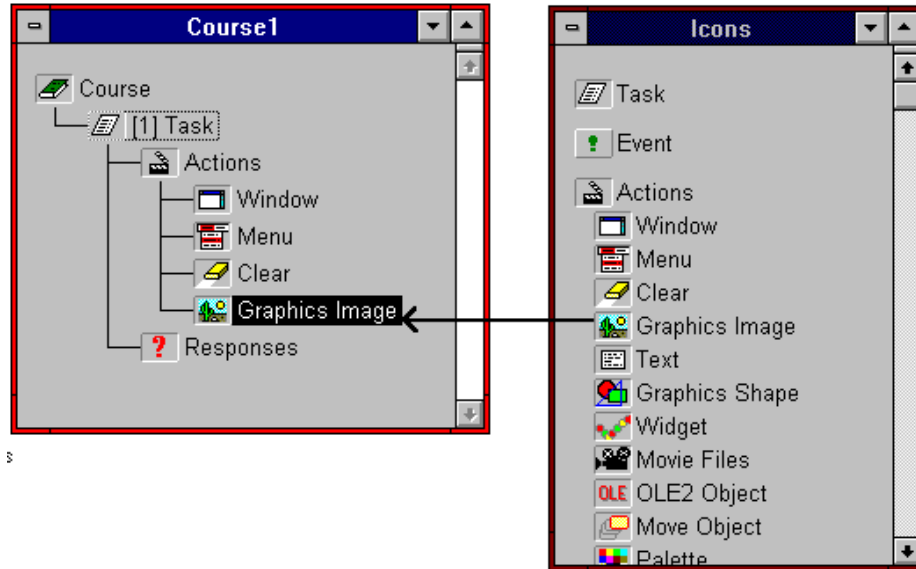


The task icon in the course window appears slightly indented, like a pushed-down button. This indicates that it has been expanded as far as it will go. Click once on the task icon. The task icon will now appear raised and its action and response icons will be hidden. In this way, the detail of the course can be hidden or examined:

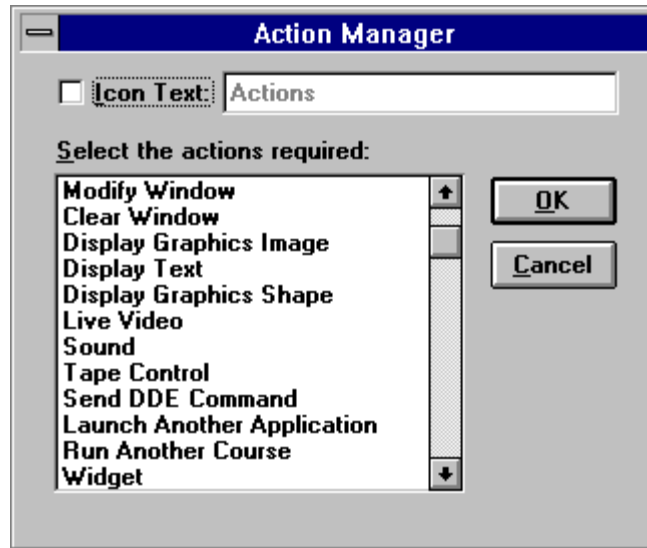


### Adding a Picture

The first task in the course now needs some actions (i.e., things that should happen at this point in the course). Move the cursor over on the *Graphics Image* icon  in the *Icon Window* and click the left mouse button. Keep the mouse button held down and drag the icon over to the *Action* icon  in the *Course Window*. Let go of the mouse button. The *Graphics Image* icon will now be dropped into the course. *Notice how the cursor changes as you drag the icon - when it shows the icon it means it is OK to drop it at this point.*

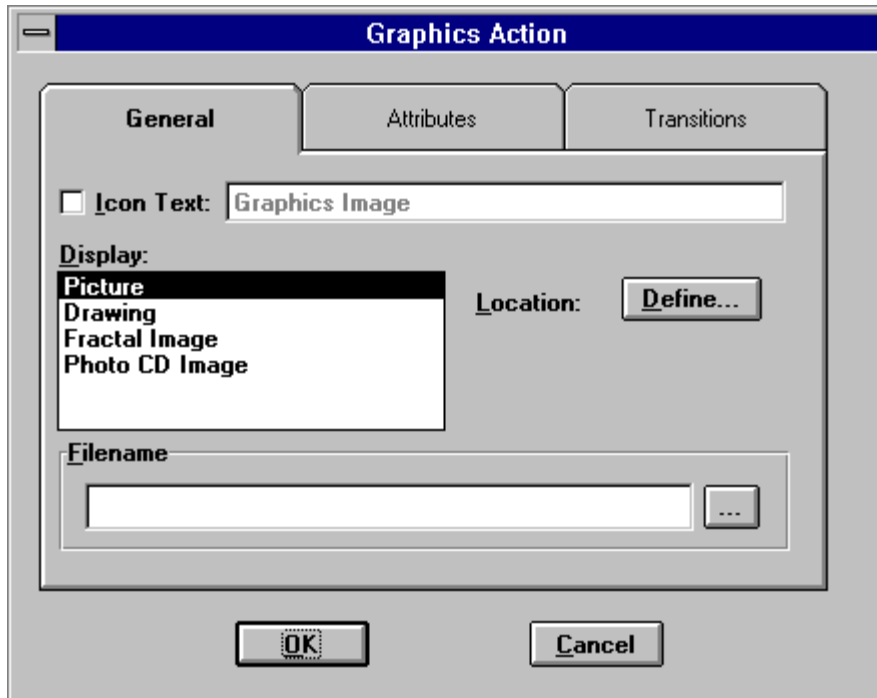


*Note: If you find it easier, rather than dragging icons from the Icon Window you can just double-click on one of the action or response icons in the course to bring up a list of possible things which can be attached to it. For example, double-clicking on the action icon will display the following list:*




*Click on the Display Graphics Image item in the list and press the OK button. The icon will now appear in the course as if it had been dragged from the Icon Window.*

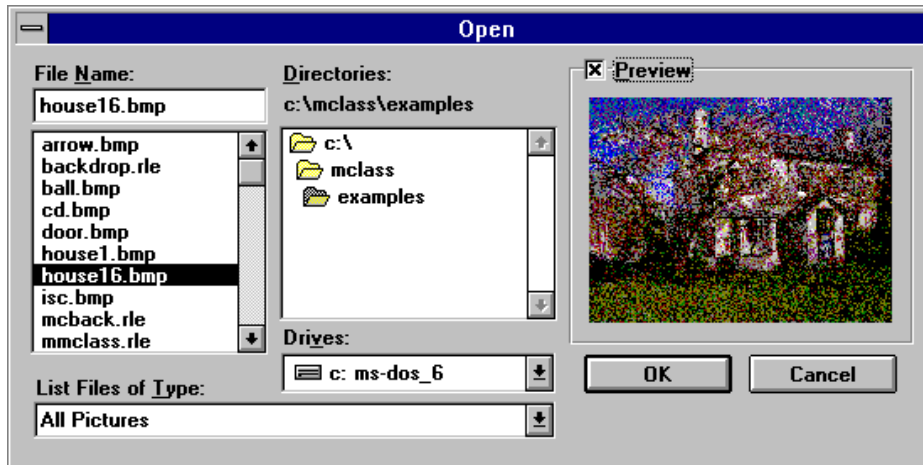
You can now specify which picture you want to display. All actions and responses are set-up by double clicking their icons in the course. Thus, double-clicking on the Graphics Image icon you have added to the course will display the following dialog box:



Displaying a picture is the default choice in the *Display* list. Refer to the Reference Manual for a description of all the formats.


There are many options which can be chosen which affect the way a picture looks in the course. However, Dazzler has been designed so that the default settings are probably good enough initially - you can always go back later and experiment with other settings. The main thing we do need to do at this stage is to select which picture file to show.

Click on the browse button  towards the bottom of the dialog box. The standard Dazzler Deluxe *File Open* dialog box will appear:



Browse through the directories in the middle column to find C:\DAZZLER\DEMO. The files in this directory will be shown on the left. Highlight HOUSE16.BMP (or HOUSE.BMP if your computer is configured for 256 colour mode or higher). The *Preview* check box can be selected to show the picture in the dialog box before it is inserted into the course. When you have found the picture you want to use, click the *OK* button. You will now be returned to the Graphics Image dialog box.

If the position of the picture is left undefined, Dazzler will assume that you want it to fill the Presentation Window. For now though, let's actually define the position of the picture. Press the *Location / Define...* button. The Presentation Window will now appear.


The cursor will change to : . Move the mouse to the point where you want the top left corner of the picture to be located. Press the left mouse button, and, while holding it down, drag the mouse to the position of the bottom corner. As you drag the mouse you will see the picture being 'painted' into the Presentation Window. Release the mouse button. The floating tool bar will appear.



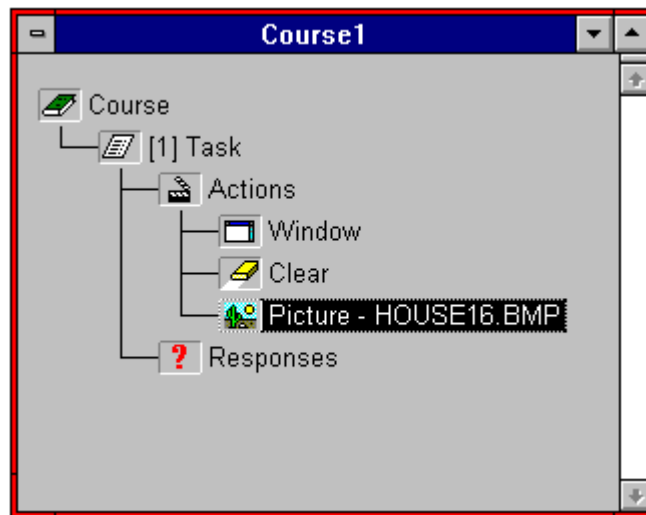
This contains tools for manipulating the objects in the Presentation Window.

The picture just inserted will have small rectangular ‘handles’ for changing the size of the area taken up by the picture. Move the mouse over them and the cursor will change to an up & down or diagonal arrow. Clicking and dragging the mouse at this point will change the size of the picture area. The cursor will change to a cross cursor when the mouse is moved into the picture area itself. Clicking and dragging at this point will move the entire picture.

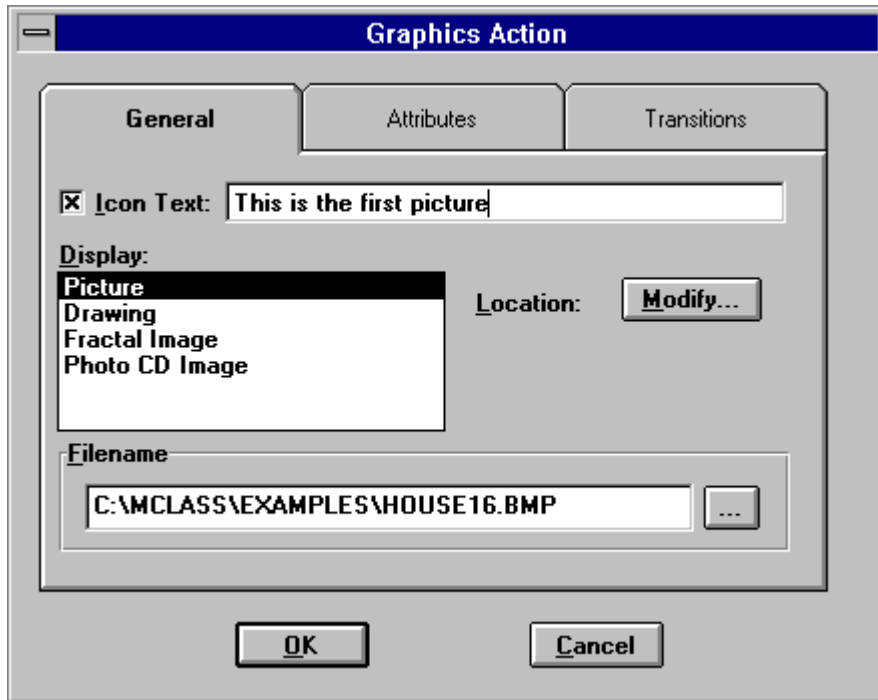
The setup dialog box for the picture, or any other object in the Presentation Window can be accessed directly by double clicking on the object itself. There is no need to go back to the Course Window and double click on the icon; it is quicker to double click on the object itself in the Presentation Window.

When you are happy with the position and size of the picture area, select the exit button from the floating tool bar , or press the *Escape* key at the top left of the keyboard. The Course Window should now look



like this.



(Note: Dazzler has automatically documented your course by indicating what the graphics action will do: - *Picture - HOUSE.BMP*. You can change the label of the icon from its setup dialog box. Double-click on the icon and check the Icon Text box. Now type in your own description of what the action is doing in the box next to it.




If you don't want Dazzler to do any documenting for you, select *Preferences* from the *Settings* menu and uncheck *Self Document Objects*. See the *Reference Manual* for more details).

To check how this action looks to the student, press the Test button  in the tool bar under the main Dazzler menu bar. The *Presentation Window* should appear, containing the picture you've chosen. To return to the Course Window, press the F3 function key, or close it in the usual Windows way by double clicking the top left hand box .

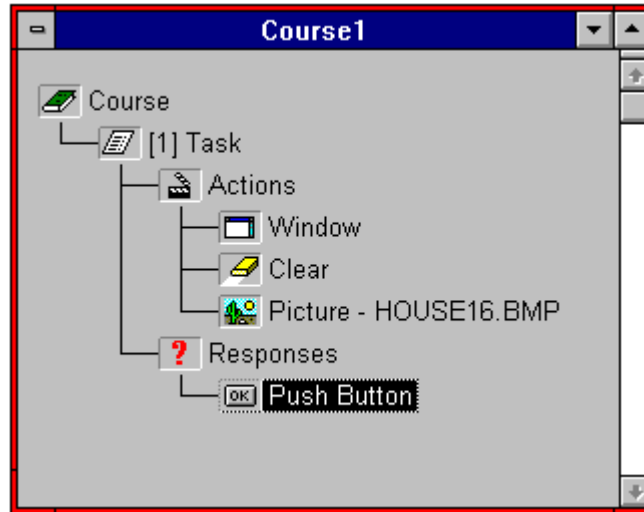
The course so far will simply show the picture and then the course will end. The next step is to add a button to the course so that the picture remains on the screen until the button is pressed by the student.


### Deleting Objects

To get rid of any task, or one of its components, highlight it by clicking on it once with the mouse. Now press the delete key, or select *Cut* from the *Edit* menu, or click the waste-basket icon  in the tool bar.

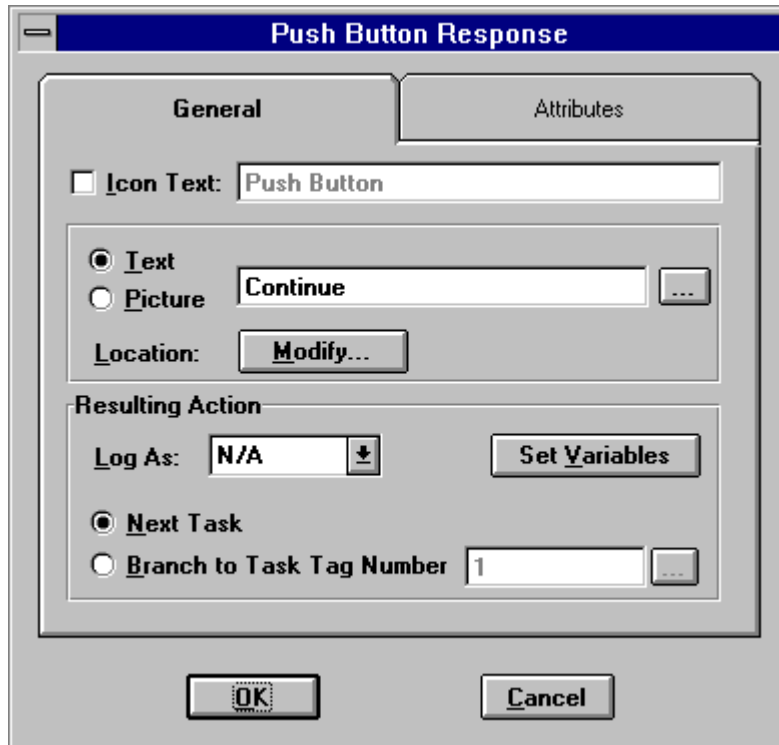
### Adding a Push Button

Click and drag the *Push Button* response from the *Icon Window* to the *Course Window*. Release the mouse button when the cursor is over the *Responses* icon of the task. The Push Button will be added to the task:



This time, don't alter any of the default settings for the push button. Instead, see how this will look to the student. Again, click on the Test  button. The Presentation Window will reappear, showing both the picture and a 'Continue' push-button at the bottom middle. This is the default, size, position and text of a push button. Now let's change it a bit.


Close the Presentation Window by pressing the F3 key. Double click on the Push Button icon in the Course Window. The following dialog box will appear:



Type in '&OK' in the button text box. The '&' in front of the 'O' text in the button text box indicates that the letter following ('O') will be the keyboard shortcut character (i.e., if the student presses 'O' on the keyboard, this will be equivalent to clicking the button with the mouse). To change the position and size of the button, click on the *Modify...* button. The Presentation Window will reappear showing the picture. The button will appear with 'handles' around it indicating that it is the currently active object:




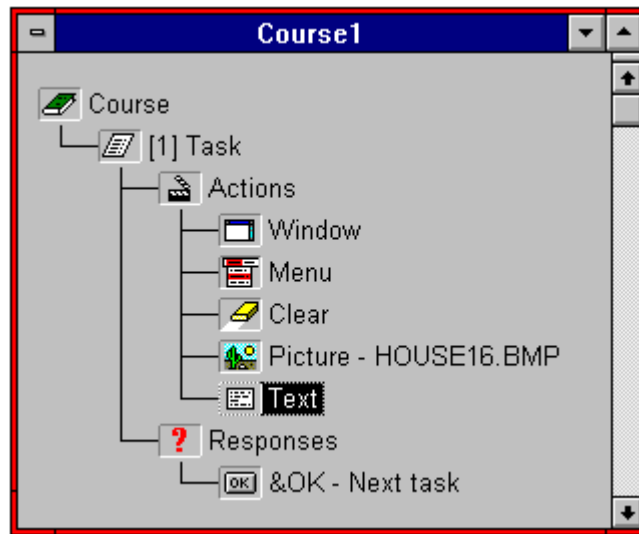
Click and drag the corner box 'handles' to change the size and shape of the button. The button can be moved around the picture by clicking in the middle of the outline box and dragging it around the Presentation Window. When you are happy with the size, shape

and position, click the Exit button  in the floating tool bar or press the Escape key on the keyboard.

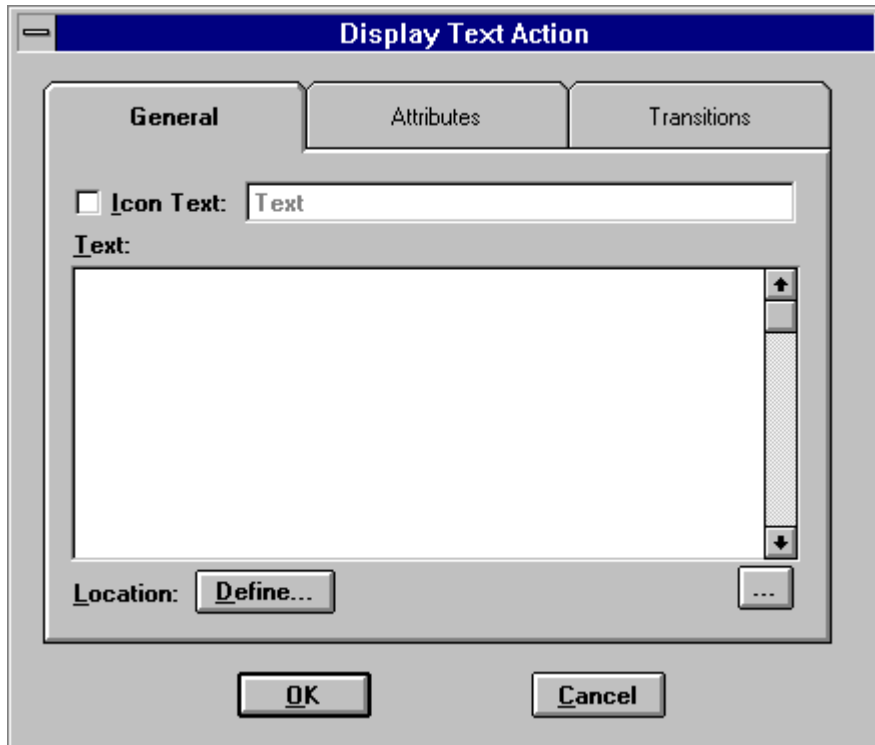
Press the *Test* button on the button bar again. The Presentation Window will show the picture with the modified button on top. Press the F3 key to return to the main Dazzler window.

### Adding Text

Finally let's add some text on top of the picture. Click and drag the *Text* icon from the *Icon Window* to the *Course Window*. To insert this action after the *Graphics Image* action , drop it on the *Actions* icon , so that the picture is shown first and then the text is put on top of it (to insert it before, drop it on the Graphics Image icon itself ).



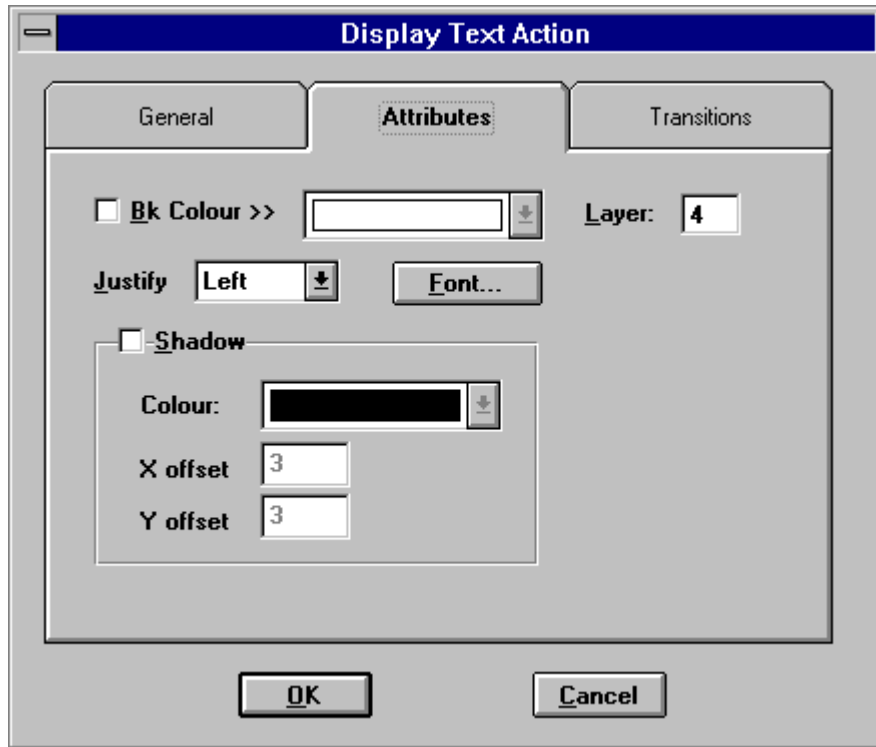
To setup the text action, double click on its icon. The following dialog box will appear:



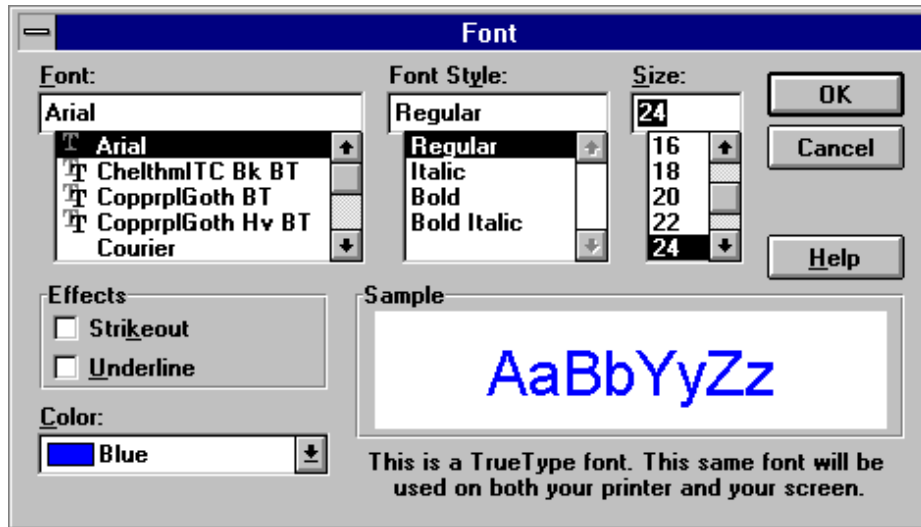
Make sure the *Text* box is ready for editing by clicking once in it. A flashing line | should appear.


Type in the text you want to appear in the *Text* edit box. You can have more than one line of text.

Click on the Attributes tab at the top of the dialog box to bring forward the attribute options:




Click the *Font...* button to select the size, colour and type of font:

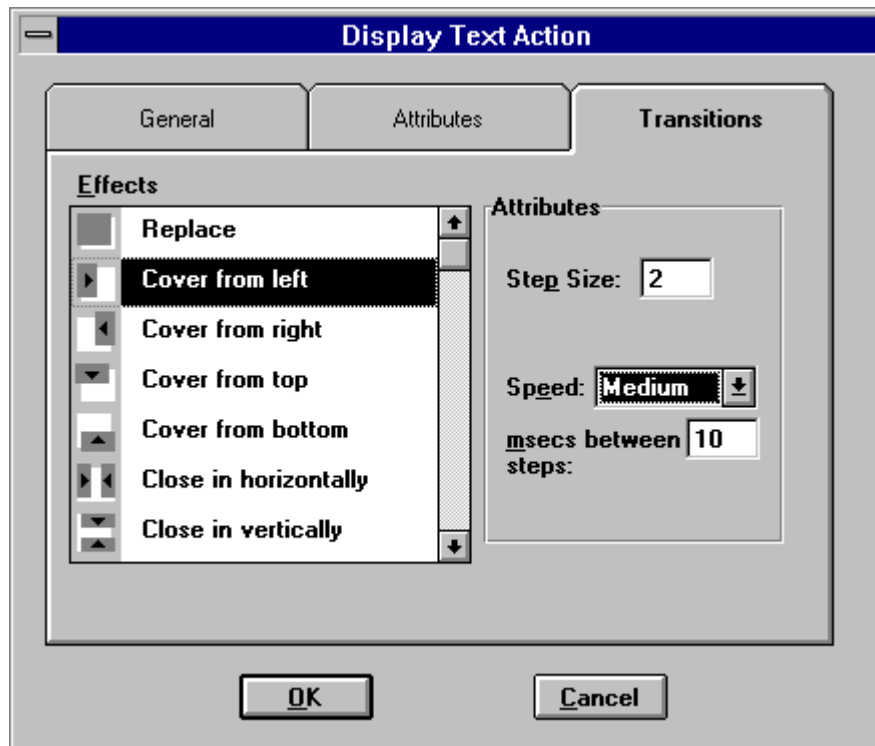



*Note: It is best to choose TrueType fonts, marked with a  in the font list, since these will look better on the screen. Also, choose one which comes as standard with Windows, such as Arial or Times New Roman if you intend to run the course on other computers.*


Click the OK button when you have selected the font. You will now be back at the *Display Text Action* dialog box. Click on the *General* tab at the top left of the box to bring forward the text entry. Click on the *Location Define...* button next to it to specify where the text is to appear in the Presentation Window.


The picture and push button added previously will be shown. The cursor will look like this: . Position the cursor at the top left point of where you want the text to appear. Hold down the left mouse button and drag the mouse to define the rectangle to hold the text. The text will appear as the mouse is dragged. Release the mouse button. The position of the text can be altered by dragging the rectangle around the screen just as you did with the push button. If there is too much text on one line to fit into the width of the area you have defined, Dazzler will wrap the text to the next line. If there is too much text to fit into the whole box, the bottom of the text will be cut off at the edge of the area. You can double-click on the text itself to bring up a setup dialog box. In this way, you can edit the wording of the text, add a transition effect or a drop-shadow to it from the

*Attributes* tab. For example, double-click on the text to bring up the setup dialog box. Click on the *Transitions* tab :



Choose *Cover from Left* in the *Effects* list and *Medium* from the *Speed* box. Click the *OK* button to return to the *Presentation Window* and then the *Exit* button  from the floating tool bar or the *Escape* key to return to the *Course Window*.

Now see how the 'course' so far will look to the student. This time, rather than just testing the actions, we'll run the course. Either select *From Start* in the *Run* menu, or click on the *Run*  button in the tool bar. The picture will appear in the *Presentation Window* with the text shown on top, 'wiped' from left to right. Clicking on the button, or pressing the button shortcut key, will end the course since we have not specified any further tasks in this course.

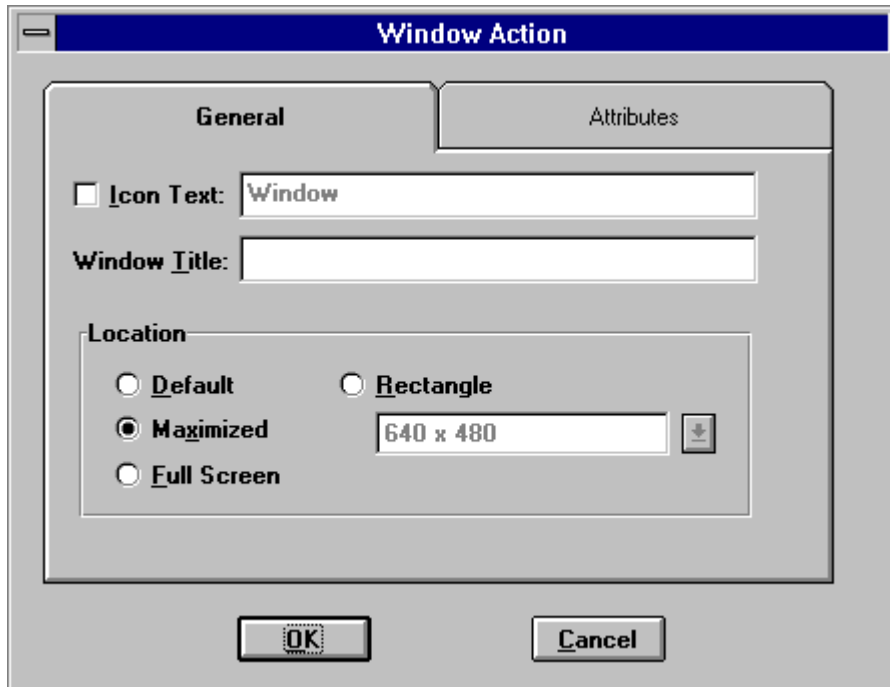
Next, save the course. Select *Save* from the *File* menu or press the *Save* button  in the tool bar. The standard Windows *File Save As* dialog box will appear. Select the directory and file name under which you want this course to be saved. Click on *OK*.

That concludes the first part of the tutorial. The procedures you have gone through show all the main features of putting together a course. There are lots more actions and responses which can be incorporated into courses, but these are all used in exactly the same way as the ones in this tutorial.

The next section will add a few more commonly used techniques to the course we have just created. Later on, we will look at more specialised features such as incorporating video and sound in courses. Feel free to experiment at any point, and refer to the Reference Manual to find out more about any particular feature.

### **Setting up the Presentation Window**


So far, when we have run the course, the Presentation Window has appeared with the default text 'Presentation Window' in its title bar and a default menu beneath that. These settings can be changed by using the *Window* action. Double click on the Window icon to show the setup dialog box:

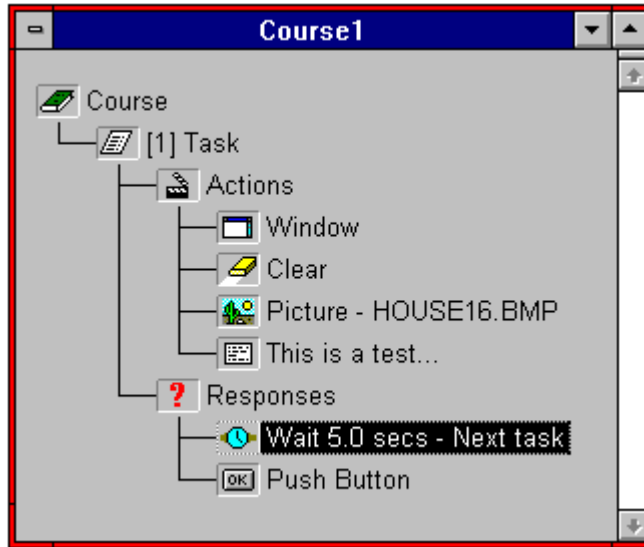


Choose *Full Screen* from the *Location* part of the dialog box. Click the *OK* button.

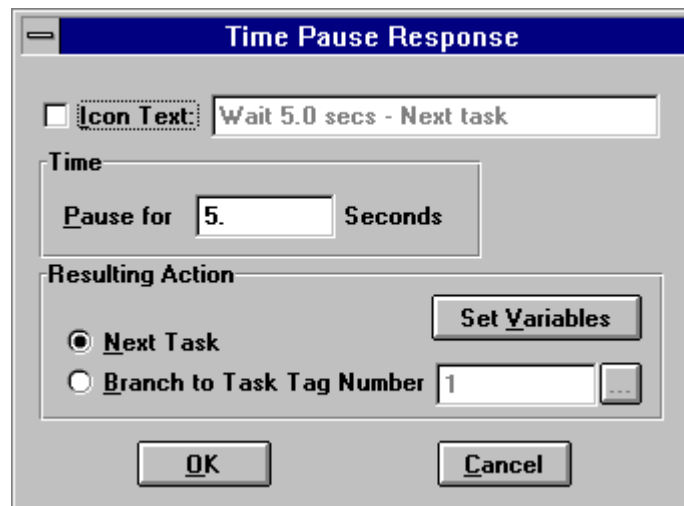
Re-run the course (from the menu or the button bar). This time the picture, text and button should appear without a Window title bar at the top of the screen. In this way, although the system is still using Windows, its complexity is hidden from the student. Press the button to end the course.

#### **Adding more than one response**

In some courses it may be useful to be able to jump to a different part of the course if the student doesn't respond in a given time period. This can be achieved by using more than one response in a task. To add a timer to your current course, click and drag the *Pause* response icon  from the Icon Window and drop it on the push button response icon added earlier. It will be inserted into the list of responses as shown:



Double-click the Pause icon to bring up the pause set-up box:



Type-in a pause, (for example 5 seconds) in the *Pause for* box and click on *OK*. We won't worry for now about branching to other tasks since our course only has one task at the moment.

Re-run the course. This time, if the button on the picture is not pressed within the specified pause period, the course will move on, in this case ending the course.

(N.B. In this case, the ordering of the responses doesn't matter - the Pause could have been added after the push button.)



---

## Asking a Question (QUEST.DZL)


The next thing to look at is adding questions to a course.

This tutorial will build a simple new course which does the following:


- asks the question "What is 2+2?"
- allows selection of an answer by pushing a button
- gives feedback to the student about whether they got it right or wrong

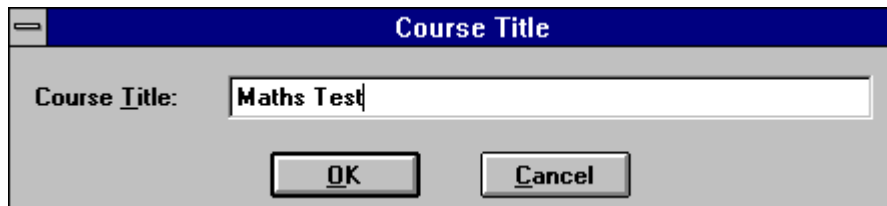
This example will then be extended to add the following features:

- prompt for the student's name
- produce a performance summary at the end.

Start a new course, either by pressing the new course button  or by selecting *New* from the *File* menu. An untitled *Course Window* will appear with a course icon and one task icon.

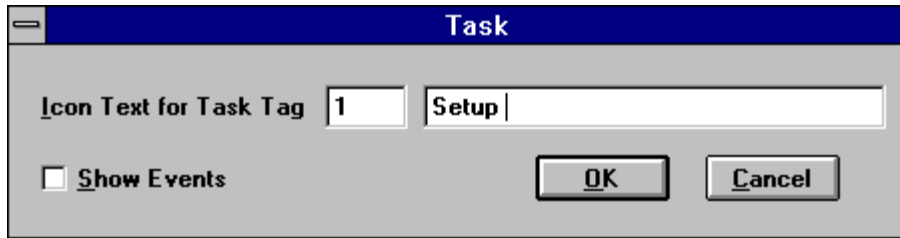
We'll take care to document this small course as fully as possible, so double click on the

*Course* text next to the icon in the course window:  *Course*. Type in a *Course Title* in the dialog box which appears, such as "Maths Test".

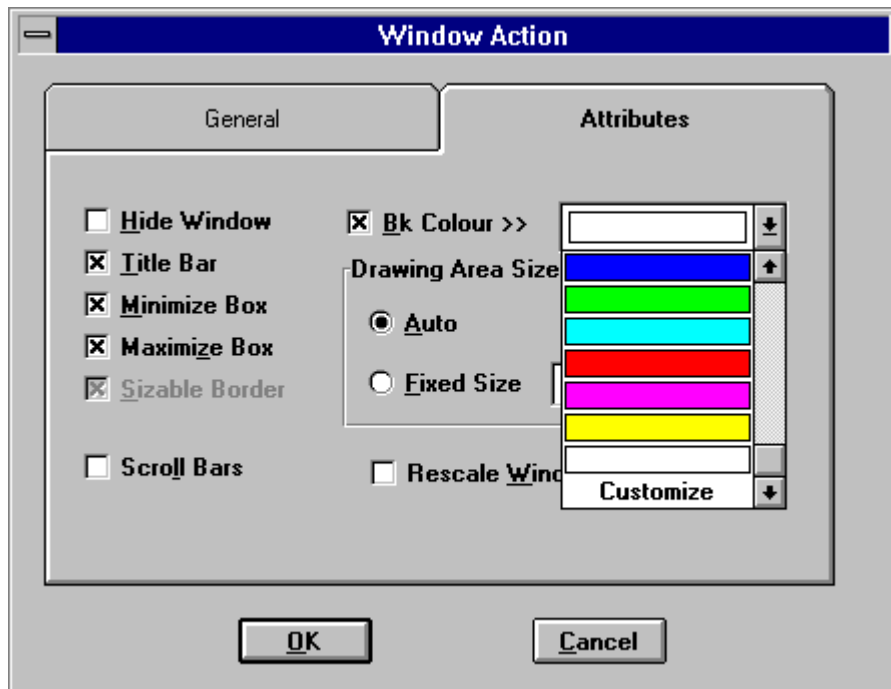


Click on the *OK* button. The course title will now appear next to the course icon.

The first task will be to set up the Presentation Window, so let's label it as such. Double click on the *[1] Task* text next to the first task icon. Type in a suitable label for this task:




Setup the Window action in the course by double clicking on its icon. Select *Full Screen* in its *Location* section. Set a background colour for the Presentation Window by clicking on the Attributes tab of the dialog box and checking the *Bk Colour* box. Pop down the list of colours next to it by clicking on it. Choose a colour from the list (you can mix your own colour by selecting the *Customize* option at the bottom of the list).



## TASK 2 - Asking the question

Drag a Task from the Icon Window to the course window. Drop it on the Course icon itself - this will add it after the first task. Dropping it on the first task would have inserted it before. Tasks and their components can be re-ordered simply by clicking and dragging them around the course window.


Give the new task a name by double clicking on its text to bring up the text edit box. Call it "Question 1", for example.

Now ask the question by dragging a *Text* action  from the Icon Window and dropping it onto the Action in Task 2, thus adding it after the Clear Window.

Double-click on the text next to the icon to show its set-up dialog box.

Type in the question text "What is  $2 + 2$  ?" into the *Text* box and press the *Font...* button in the *Attributes* section of the dialog box to select a suitable font (for example, *Arial 36pt* in red).

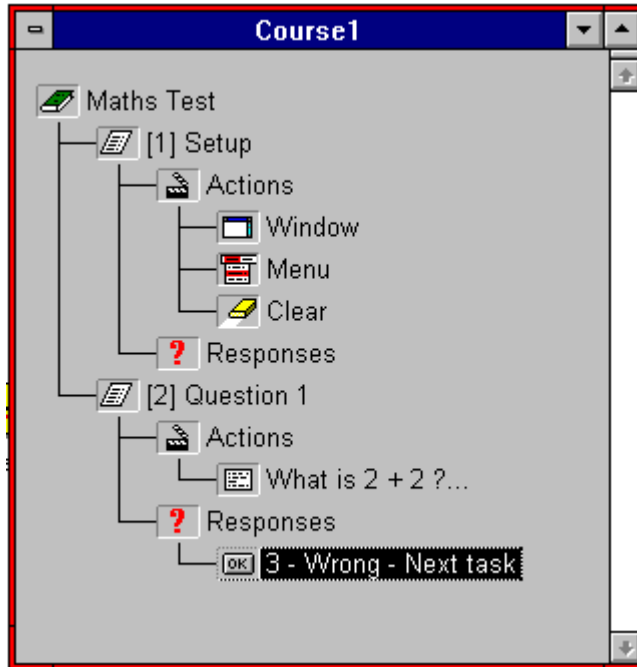
Press the *Define* button and position the text towards the top of the screen in the same way as you did in Tutorial 1. Click *OK* to return to the Course Window.

Having asked the question, you can now let the student respond by selecting from possible answers in the form of push-buttons. Drag a *Push Button* response  from the Icon Window and drop it on the Response icon of the second task.

Double click on the icon to define how the button is to behave. In the Button Text box type  $\&3$  (remember the '&' is there to provide keyboard support - pressing 3 will be the same as clicking this button with the mouse). All buttons have a default position at the bottom middle of the Presentation Window. To move the '3' button to a more suitable place, click the *Modify* button and drag the button outline around the screen. Click *OK* when it is in the right place.


In the *Log As* part of the dialog box, select *Wrong*. Click *OK* to end the setting up of the button. The links to other parts of the course will be made later.

The Course Window should now look like this:



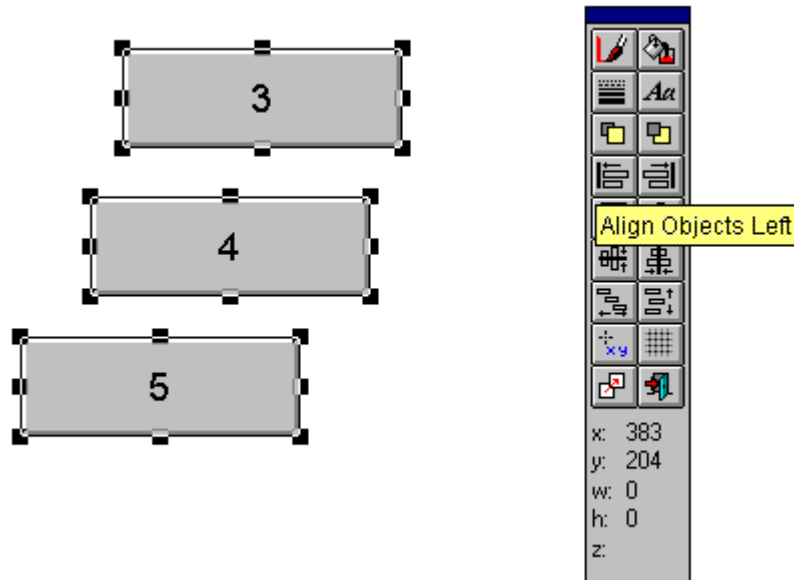
To add two more answer buttons, either drag the Push Buttons response icon from the Icon Window as you did for the first one, or copy the button just created and modify its attributes. There are two ways of doing this - in the Course Window or the Presentation Window. For both methods make sure the '3' button response is highlighted by clicking once on it.



1. Select *Copy* from the *Edit* menu (hold down the *Ctrl* key and press *C*) and then choose *Paste* (*Ctrl + V*). A copy of the button will be pasted into the response list for this task after the first button. The same process can be achieved by clicking and dragging the button in the course onto the Response icon for that task. Release the mouse button: a message will appear asking if you want to move or copy the object. Choose *Copy*.

Alternatively, the button can be copied from within the Presentation Window. Double-click on the button icon in the Course Window. Select *Modify* in the set-up dialog box. The button will be shown highlighted in the Presentation Window. Click the Copy Object button  on the floating tool bar. Click and drag the copy of the button which is now lying directly on top of the original. This process can be repeated to create a third answer button. At this stage the buttons all contain the same text. To modify the two just added,

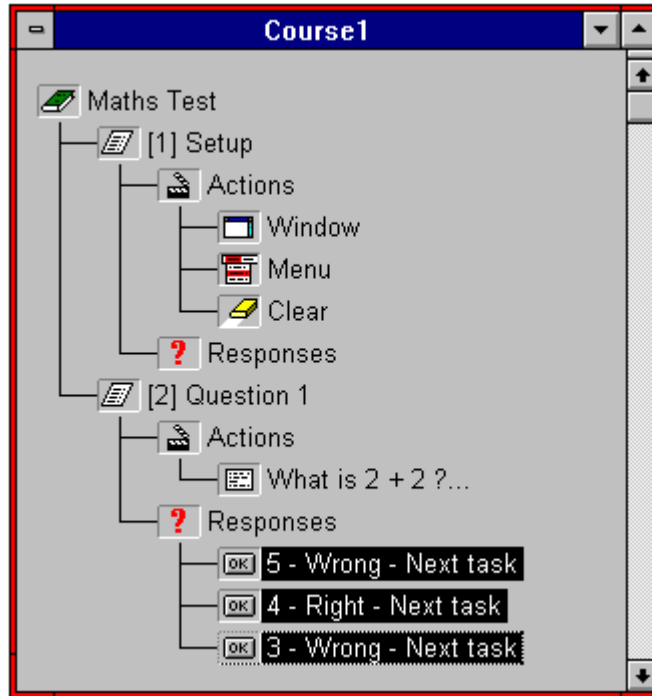
double click on the button in the Presentation Window or the Course Window. Alter the *Text* to read '&4' and '&5' respectively. *Modify* the positions so that the buttons do not overlap in the Presentation Window. Also, change the '4' button to *Log As' Right*.

The three buttons can also be aligned automatically in the Presentation Window. Click on the three buttons in turn while holding down the *shift* key on the keyboard. This will highlight all three as a group. They could also be selected by clicking and dragging the mouse to define a rectangle around all three buttons.




Now click on the Align Objects Left button  in the floating tool bar. The left hand edges of the buttons will now all be lined up. The buttons can also be spaced evenly automatically as well. Press the Space Evenly Vertically button . The spaces between the buttons will now be equal. The buttons can be moved about as a single object while they are all grouped together. Clicking and dragging any of the buttons will move them all. Position the buttons under the question text created earlier. Press the *Exit* button on the floating tool bar, or press the Escape key, to return to the Course Window.

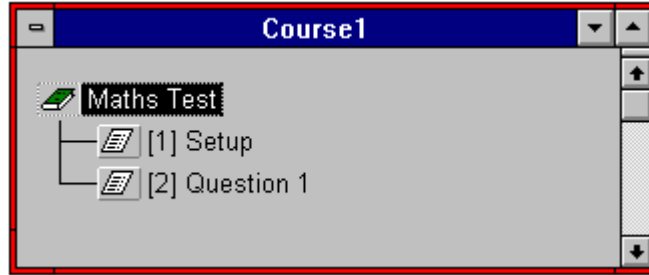
The Course Window should now look like this:



Note that Dazzler is documenting what each object does (e.g., '3 - Wrong - Next Task). If this is not happening, check that the *Self Document Objects* option is on in the *Settings/Preferences* menu item.

Run the course to see how it looks so far. If the question text looks the wrong size or colour, or the buttons aren't in the right positions, double click on the text action or button responses and modify the appropriate attributes.

Now tidy up the course window by collapsing the two tasks created so far. Click once on the Task icon  for each task- remember they act like buttons. The course window should now look like this:



The next two tasks to be added will let the students know that they've got the answer wrong or right.


### Telling the student they've got it wrong

Drag another *Task* from the Icon Window and drop it on the *Course* icon in the Course Window. It will be added to the end of the course. Double click on the task text label and give it a name such as 'Wrong'.

Drag a *Clear Window* action onto the task.

Next, drag a *Text* action and drop it onto the new task. Double-click to set up its attributes - give it some text such as 'No, that's wrong - try again', select a font for it and position the text in a rectangle on the screen.

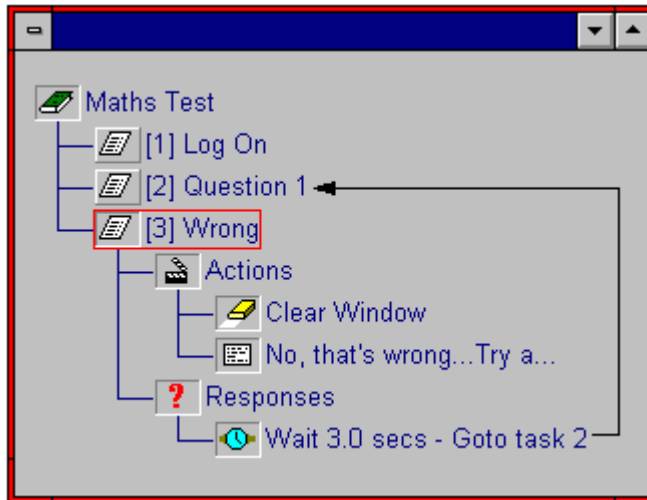
You can check how it looks by pressing the Test button in the tool bar - this just runs the current task.

After showing the student the message the course will wait a few seconds before asking the question again. Drag a Pause response icon  onto the Response icon of this third task. Double click on it and set it up to *Wait for* 3 seconds. Let's also start to deal with the links between the tasks.

Highlight the Pause response icon in the course. Click and hold down the **right** hand mouse button and drag it to the Question 1 task. Release the mouse. A link from the third task back to the second will be made automatically.

Alternatively, click on *Branch to Task Tag Number* in the Resulting Action of the dialog box. Press the *Select...* button to bring up a list of the tasks you can branch to. You can now see that this is why it is important to give the tasks names. To get this task to branch back to the question after the pause is over, just click on *Question 1* in the list. Select *OK* and return to the Pause dialog box. The number 2 has been automatically inserted in the *Branch to Task Tag Number* box. Click *OK*.

The course window should now show the new task with a branch from the pause back to the question task.



Collapse this task by clicking once on its icon.

#### **Telling the student they've got it right**

Copy the 'Wrong' task by clicking once on the 'Wrong' text label of the task and then clicking and dragging it up to the Course icon. You will be prompted to move or copy the task. Choose *Copy*. A duplicate task will be added to the end of the course.

First change its text label to 'Right'.

Now open up the task by clicking once on its icon. Note that the entire structure of the previous task has been copied.

Bring up the set-up dialog box of the text action and change the text to say 'Yes that's right'. The font and position of the text has been copied from the previous task, but it can be changed if you want.

In the *Pause* response, select *Next Task* from the *Resulting Action* part of its set-up dialog box.

Collapse this task by clicking once on its task icon.

---

## Getting and displaying data

Dazzler stores information used in a course in *variables*. Some of these, such as *CourseTitle* and *StudentName*, are already built in as part of the automatic student logging; others you can create to store information relevant to a particular course, for example *JobTitle*.

In the previous simple quiz the student was asked a question and told whether they had got the answer right or wrong. It can be very useful to record the student's details and to produce a score sheet of their performance in the training course, as well as recording those results for later analysis.

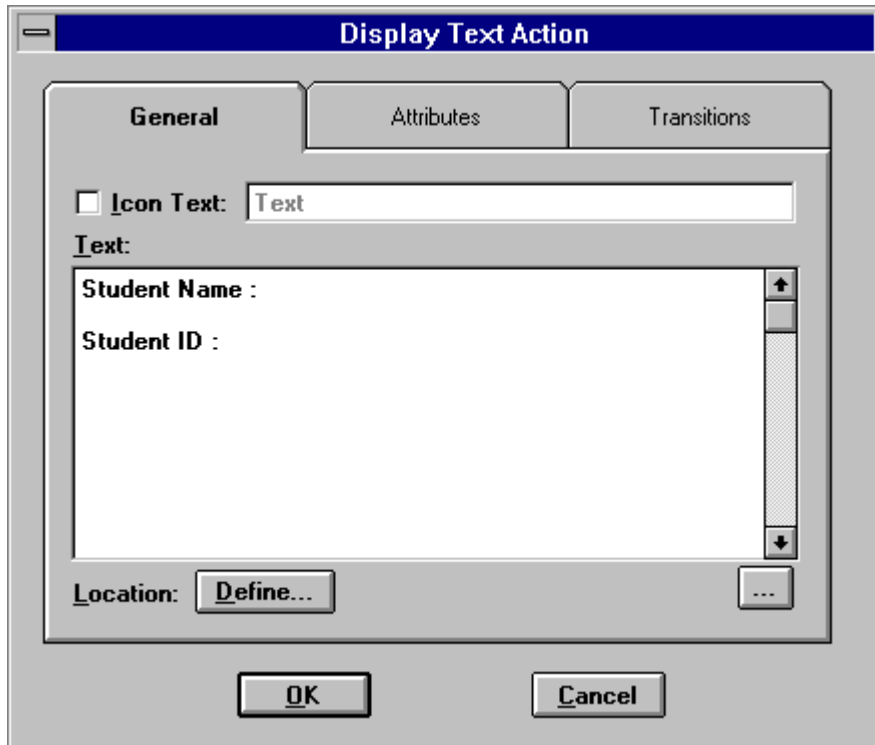
This section will look at recording the student's name and ID, as well as displaying a score sheet at the end of the course.

### Getting the student's details

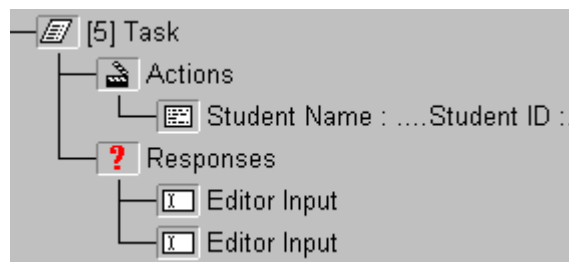
Editor Input responses will be used to get the student's name and ID. The information typed in will be stored in two built-in variables *StudentName* and *StudentID*.

Use the "What is 2+2 ?" example course created earlier. Insert a new task between the first 'Setup' task and the 'Question 1' task. Don't worry that the task numbering now appears out of sequence. Dazzler can take care of sorting out the number labels later.

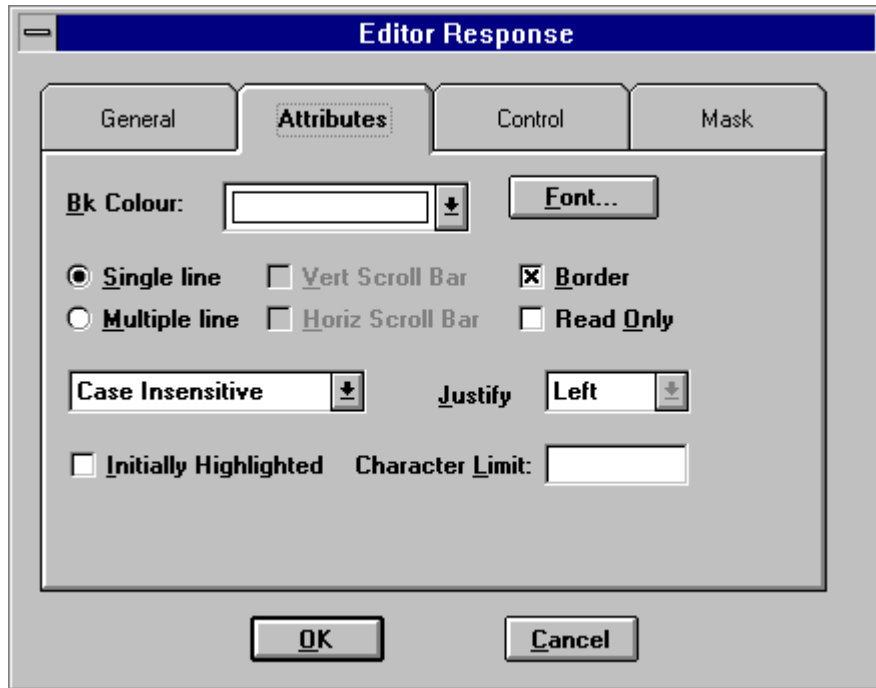
Drag a *Text* action into the new task. Set it up to display the text "Student Name" and "Student ID" on the left of the Presentation Window with enough room to place edit boxes next to them.



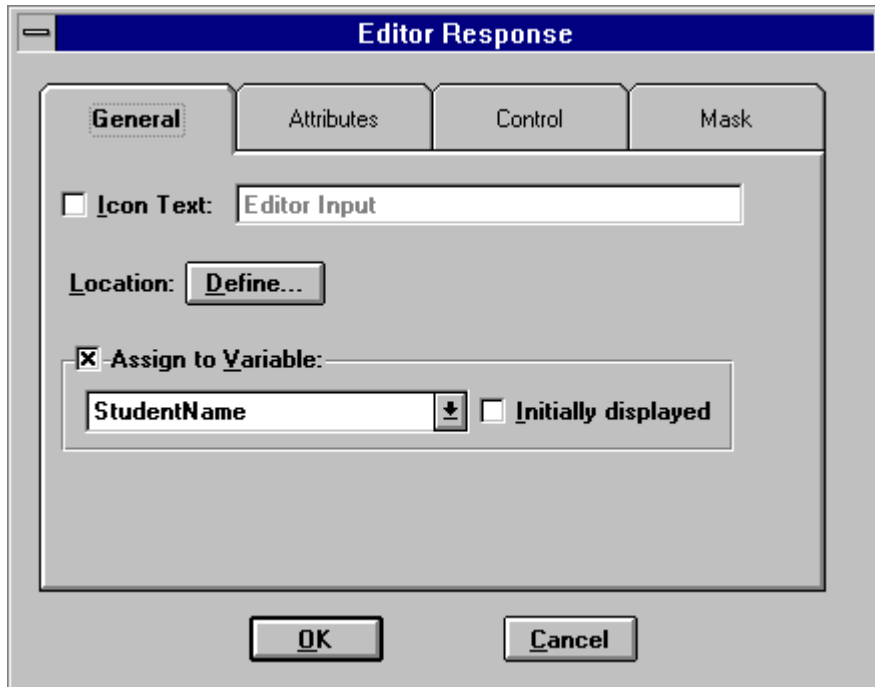
To get the information from the student and into the course, add two *Editor Input* responses by dragging them into the task from the list of Responses in the Icon Window. The task should now look like this :



Setup the Editor Input responses by double-clicking on their icons. Set them both to have borders and to be single line in the *Attributes* tab of the dialog box.



Also use the *Assign To Variable* option to store the result of the student's typing in the variables, as shown below:



Finally, also add a Push Button and position it towards the bottom of the Presentation Window. When the student presses this button, the course will move on to ask the “+” question. The names and ID typed into the edit boxes by the student will be stored in the variables store at this point.

When the course is run, the first screen to be seen should look something like this:

Student Name :

Student ID :

Continue

### Showing Results


The final task in the course will show the student how they've done.

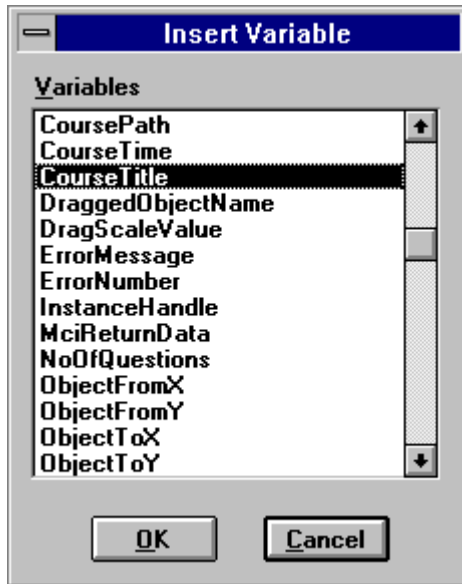
Drag a new task from the Icon Window onto the course icon in the course window. This will add it to the end.

Give it a label (by double-clicking on the text to the right of the icon), for example 'Results'.

Drag a *Clear Window* action onto the task.

Drag a *Text* icon from the action list in the Icon Window and drop it on the new task. Dazzler now has to be told what to display. This will be a combination of fixed text such as "Name of student" and information itself collected as the course runs and stored in variables, such as `StudentName`. Dazzler also generates its own information as it runs: it calculates scores and records the number of questions which were repeated, etc. Any task which contains a response which is logged as *Right* or *Wrong*, for example the answer buttons in the Maths Test course, is marked as being a question by Dazzler. The number of these questions is stored in the variable `NoOfQuestions`. Double-click on it and enter

the following in the *Text* box. The variables browse button  can be used to bring up the list of variables. This can help to prevent typing mistakes if the names of the variables are typed in directly.



Pick the names of the variables from the list, one at a time. They will be inserted in the text box :

**Course : {CourseTitle}**

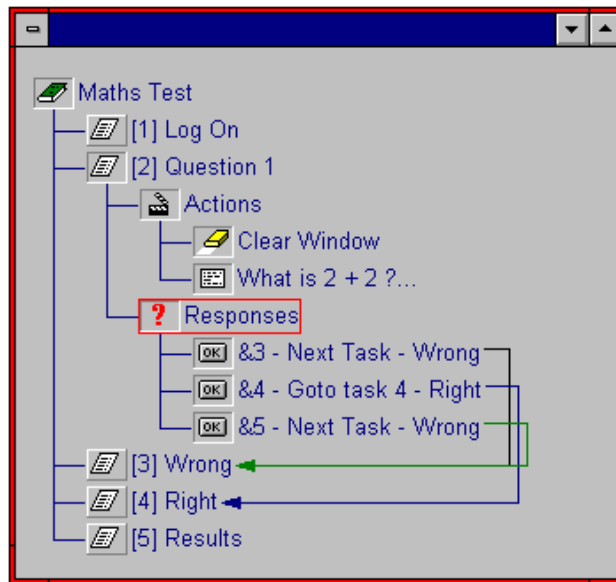
**Date : {ActualDate}**

**Student : {Student Name } ID : {StudentID}**

The parentheses {} tell Dazzler to insert the contents of that variable. The score sheet can be tailored to display whatever information you think will be relevant. The appearance can be changed with the *Font* and *Attributes* settings of the *Text* action.

Finally, drag a push button response onto the response part of the task. There is no need to alter its attributes since the default is a 'Continue' button at the bottom middle of the Presentation Window - when the student presses this, the course will end.

The course should now look like this:



Run the course and check that each stage works as you want it to.




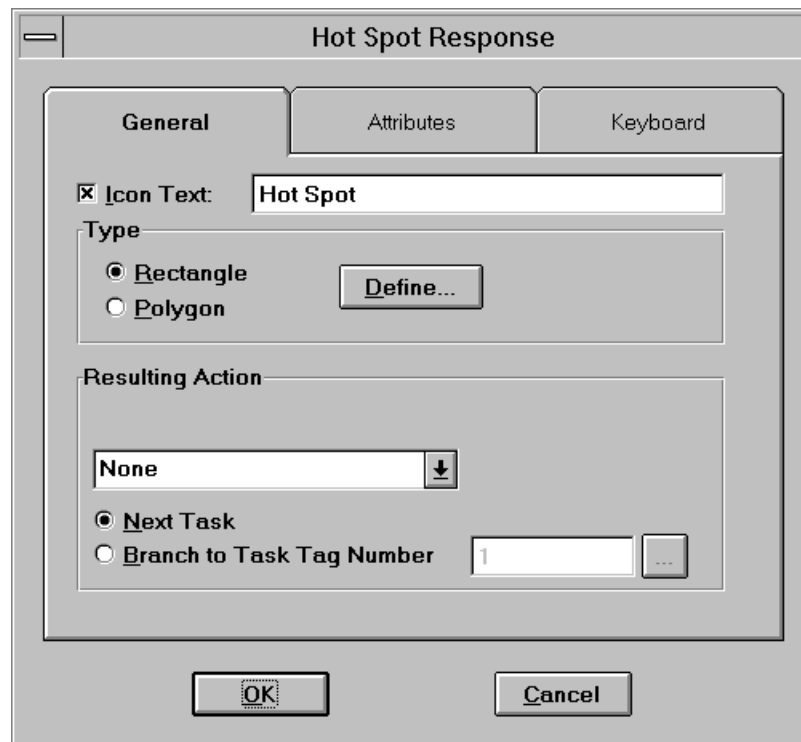
---

## Pictures and Hot Spots (HOTSPOT.DZL)

Dazzler screens can also be linked using *Hot Spots*. Areas of the *Presentation Window* can be defined as *Hot Spot* responses which act as invisible buttons - when the user clicks the mouse in these areas, or presses a defined key, the response is activated and the presentation branches accordingly. For example, a photograph of a piece a machinery could be shown and the user could click on the start switch to show a video clip of the machine operating. The hot spot regions can be either rectangular or irregularly shaped so that the hot spot can exactly fit part of a picture.

To use *Hot Spots*, add a picture to a new presentation using the *Graphics Image* action.

Now drag a *Hot Spot* response  into the presentation. Double-click on the icon to set it up. The first thing to do is to define the size and shape of the *Hot Spot*.

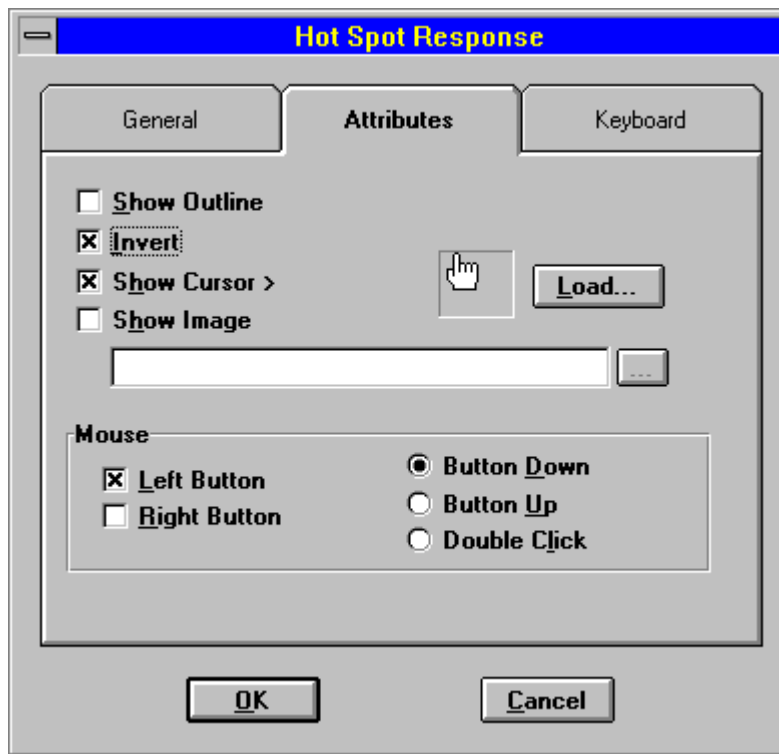


The image shows a dialog box titled "Hot Spot Response" with three tabs: "General", "Attributes", and "Keyboard". The "General" tab is active. It contains the following elements:

- Icon Text:** Hot Spot
- Type:**
  - Rectangle
  - Polygon
- Resulting Action:**
  - None
  - Next Task
  - Branch to Task Tag Number 1

At the bottom of the dialog are  and .

Select *Rectangle* from the *Type* section for your first experiment. Now click the *Define...* button. The *Presentation Window* will appear and the cursor will change. Move the cursor to the point which will become the top left corner of the *Hot Spot*. Click the left mouse button and hold it down while dragging the mouse to the bottom right point. Release the mouse button to define a rectangle. The size of the *Hot Spot* can be changed by clicking and dragging the handles shown at the edge of the rectangle formed. The *Hot Spot* can be moved by clicking anywhere inside the *Hot Spot* rectangle and dragging it to its new position. Double-click inside the *Hot Spot* to bring up the set-up box again. Click on the *Attributes* tab. You can now define the way in which this region will behave.

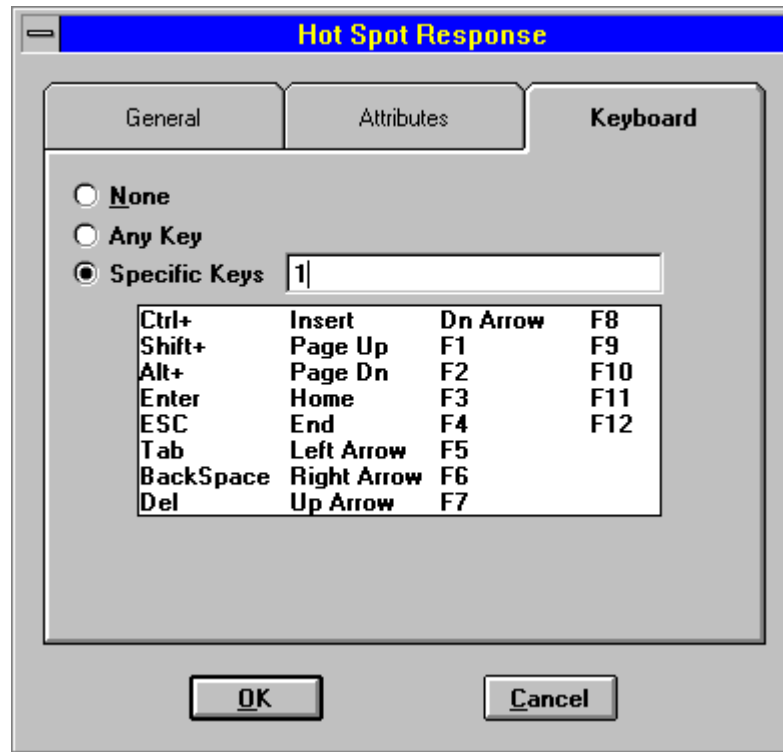



To change the cursor whenever the user moves it into the hot spot area, click on the *Show Cursor* check box. The default cursor is a pointing hand, but others can be chosen by pressing the *Load...* button. A range of different cursors are supplied with Dazzler, but any *.CUR* file can be used.


If you want the *Hot Spot* area to be more obvious when the user moves the mouse over it, then click on the *Invert* check box. The colours of the region will be inverted whenever the mouse is within the region, thus highlighting it.

The region can either be activated by a single or a double click of either mouse button - choose the appropriate radio button choice from the *Mouse* part of the set-up box.

If you want the user to be able to use the keyboard to select the region rather than, or in addition to, using the mouse, click on the *Keyboard* tab of the set-up box. For example, to trigger the region by pressing "1" on the keyboard, click on the *Specific Key* radio button and type *1* in the edit box next to it.



Press the OK button when you have finished setting up the *Hot Spot* and click the *Exit* button  in the floating tool bar to return to the *Design Window*.

To try out the result, click on the *Run*  button in the tool bar. The picture you have chosen should appear. Clicking anywhere other than in the *Hot Spot* area will have no effect. Moving the mouse into the *Hot Spot* area will change the cursor shape and the area will 'light up' if the *Invert* option was chosen in the set-up. Clicking the mouse or pressing the assigned key will end the presentation or take it on to the screen linked to it, just as if a button had been pressed.

Several *Hot Spot* responses can be used in one task, giving a 'multiple choice' facility to pictures. For example, different *Hot Spots* on the same picture could link to tasks showing close-up views of that particular part of the picture, or an explanation of the part selected using text and video.

---

## Adding voice over using the Sound Action


This tutorial shows you how to add a voice over or music to any screen. In this example, when the user is shown a picture, a voice over asks the user to click on a part of the screen to get more information. This example will add sound to the presentation created in the *Hot Spot* tutorial

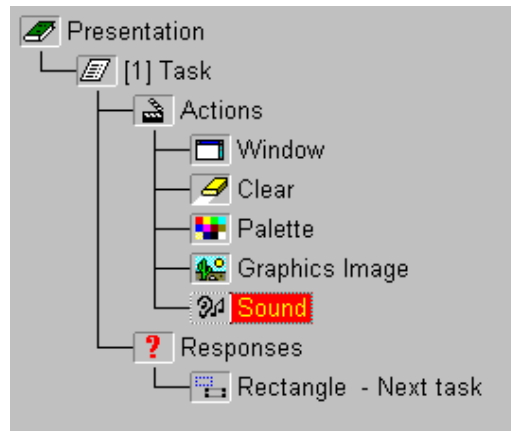
N.B. FOR THIS TUTORIAL YOU WILL NEED A SOUND CARD IN YOUR COMPUTER AND A MICROPHONE AND SPEAKERS CONNECTED TO IT.

### Step 1

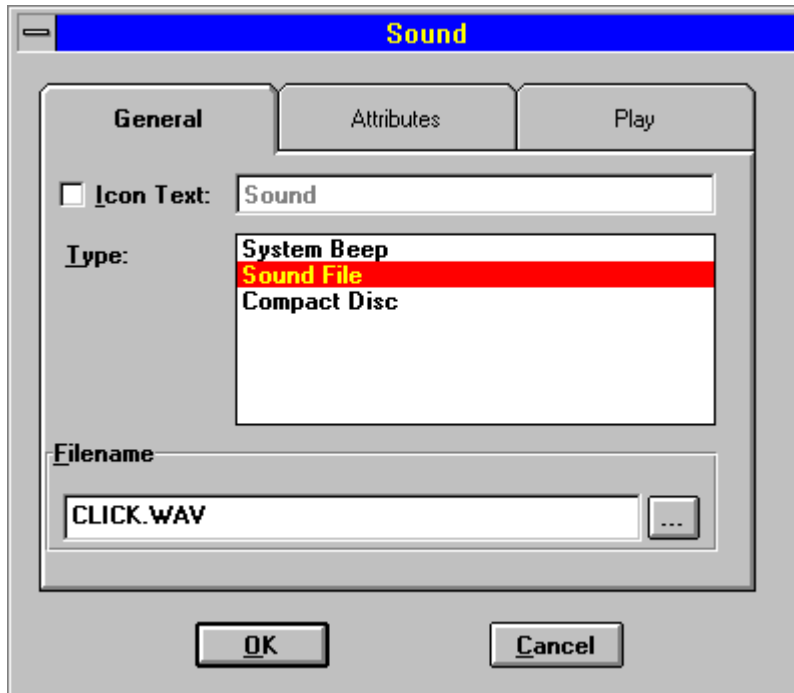
Record some voice over using the recording software supplied with your sound card or simply use the Windows Sound Recorder in the Accessories Group. The voice over could be any kind of message to the user, for instance, "Click on the .... to see more details". Refer to your Microsoft Windows User Guide for a full explanation of how to use the Sound Recorder, or select the on-line help from the menu bar in the application itself.


### Step 2


Now add a *Sound* action to the first task created in the *Hot Spot* tutorial by clicking and dragging a *Sound* action  from the *Icon Window* into the *Design Window* of your presentation.



Double-click the *Sound* icon to bring up the set-up dialog box.



Select *Sound File* from the *Type* section and browse for the sound clip using the browse button . The standard *File Open* dialog box will be shown. To hear a sound file, click the *Preview* check box and click once on the file name itself. The slider bar at the bottom right can be used to move backwards and forwards through the sound file. When you have found the right sound file, click the *OK* button. The *Attributes* and *Play* tabs of the set-up dialog box can be used to control how the sound file will behave and how much of it will be played. For now, all the default settings will be used. Click the *OK* button to return to the *Design Window*.



When the presentation is run now using the *Run* button  in the tool bar, the picture should appear and the voice over will be played.

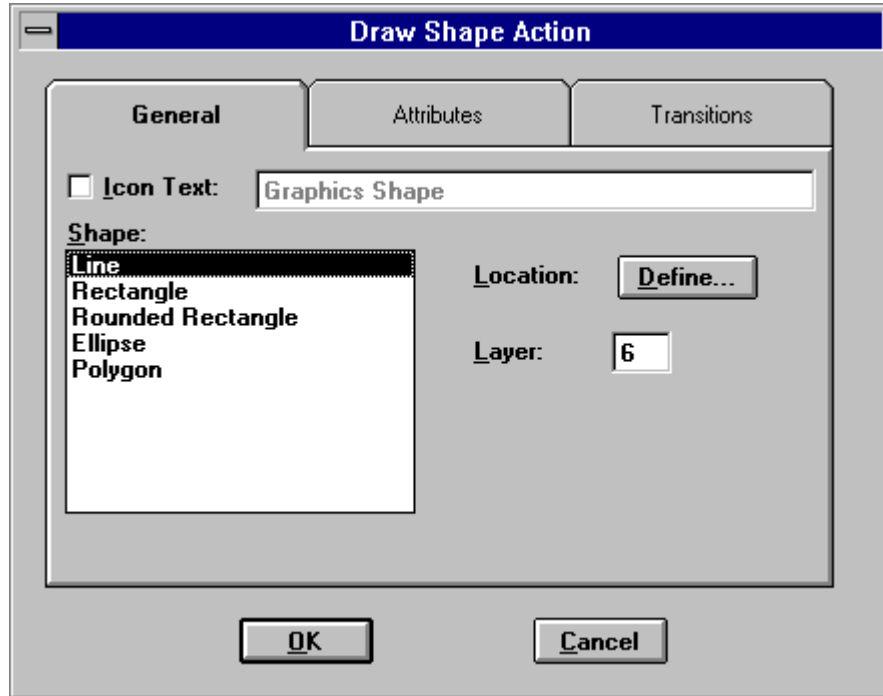
---

## Creating a simple graph (GRAPH.DZL)


This tutorial shows how the *Graphics Shape* action can be used with transition effects to create an animated graph very quickly.

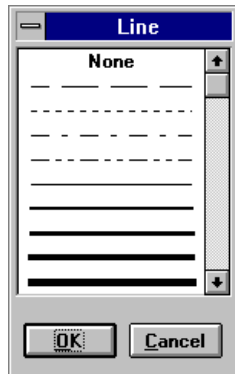
Start a new presentation using *New* from the *File* menu.



Create the axes using Dazzler's built-in graphics drawing tool. Drag and drop a *Graphics Shape* action  onto the *Actions* icon  in the presentation. This will add it after the default actions (*Window*, *Clear* etc.). Double-click to set-up the action. The following dialog box will appear:




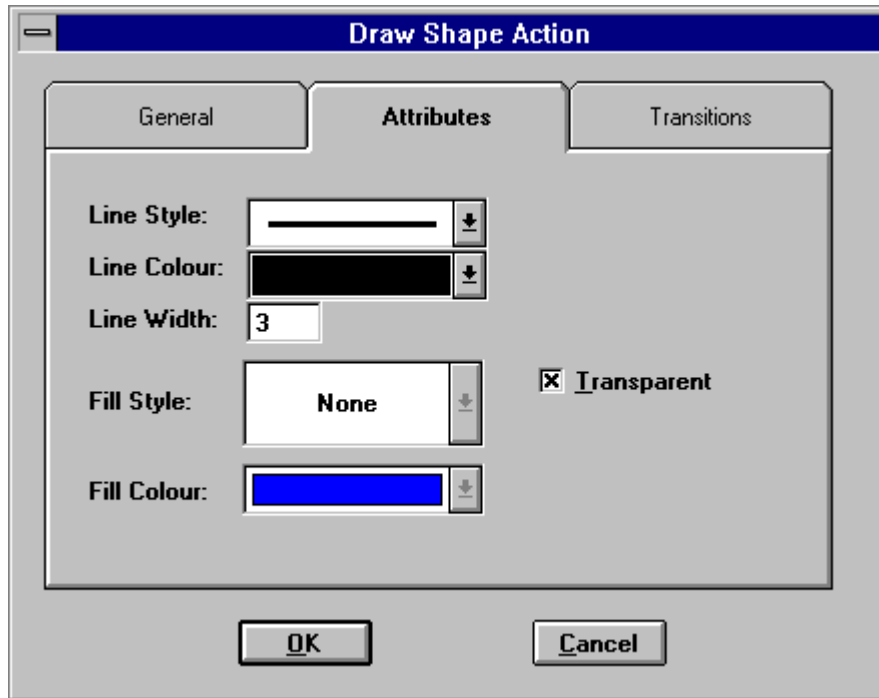
Click on *Line* and press the *Define* button. The *Presentation Window* will be shown. To draw the vertical axis, click the left mouse button to define the start point of the line towards the top left of the screen. Hold the mouse button down and move the mouse to the end of the line towards the bottom left of the screen and release the mouse button again. The line has now been drawn. Square 'handles' will appear at each end of the line

which can be moved to change the length and direction of the line. Click and drag the line itself to change its position. To change the thickness of the line press the *Line* button in the floating tool bar . This will pop-up a list of line styles and thicknesses:





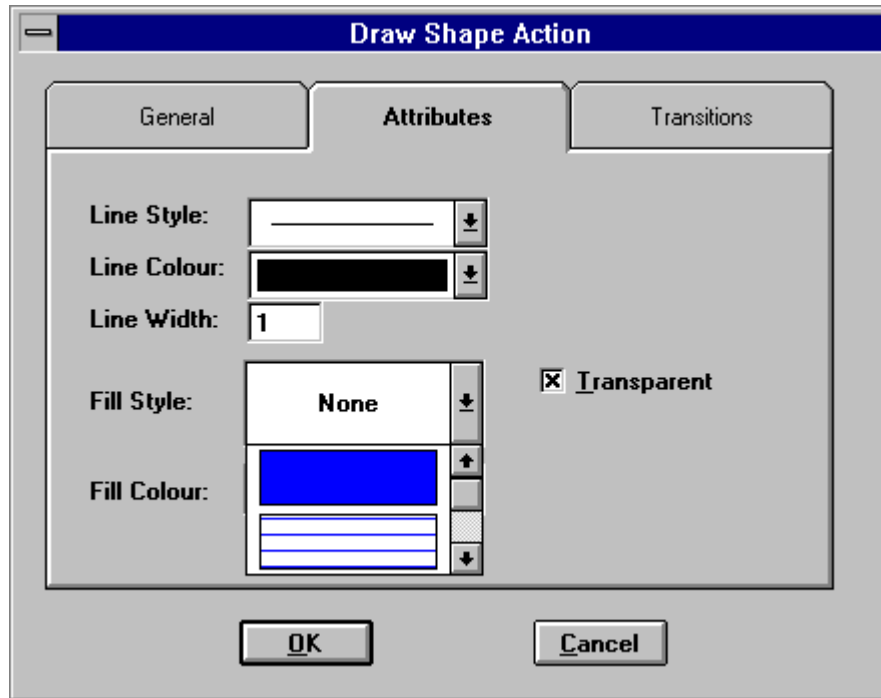
Click on the option you prefer and press the *OK* button. The colour of the line can be changed with the *Line Colour* button . When you have finished creating the line, press the *Exit* button in the floating tool bar  to return to the *Design Window*.

Add in another line for the horizontal axis by dragging another *Graphics Shape* action onto the *Actions* icon  of the first task. Double click to bring up the setup box again. The thickness of the line and its colour and style can be set directly from the *Attributes* tab of the dialog box as well as through the floating tool bar.

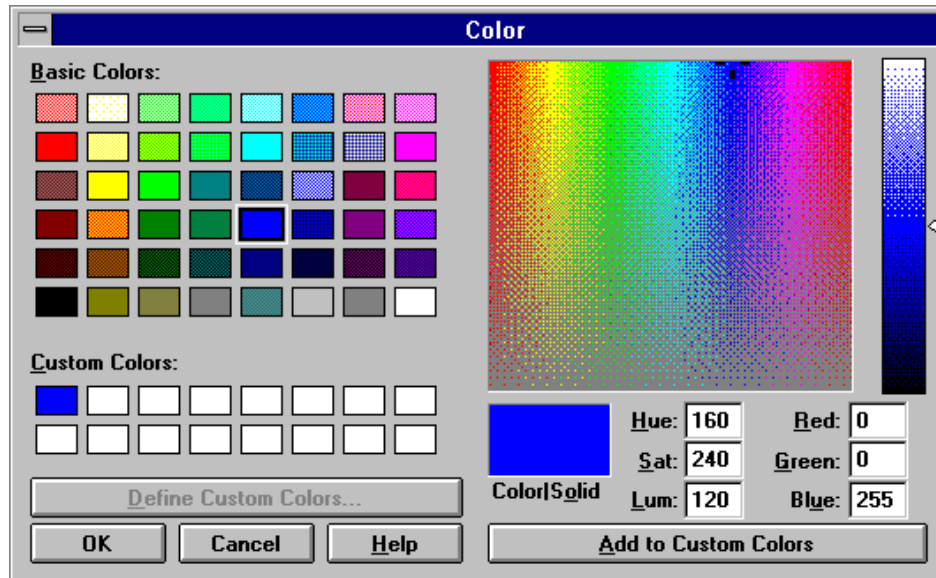


Use the *Define* button in the *General* tab to draw a line representing the horizontal axis by clicking and dragging the mouse.

Now we need to create the bars of the graph. Again, the *Graphics Shape* action  will be used. Drag and drop another one from the *Icon Window* onto the *Actions* icon  in the *Design Window*. Again double-click the icon to bring up the dialog box. Select *Rectangle* from the *Shape* list on the *General* tab. Click on the *Attributes* tab to choose the colour of the bar to be drawn. Click on the *Fill Style* drop down list box. The default item in the list is *None* meaning that the rectangle drawn will not be coloured in. To change this, click on the solid colour shown as a blue rectangle by default or scroll down the list to choose a fill pattern for the rectangular bar of the graph:

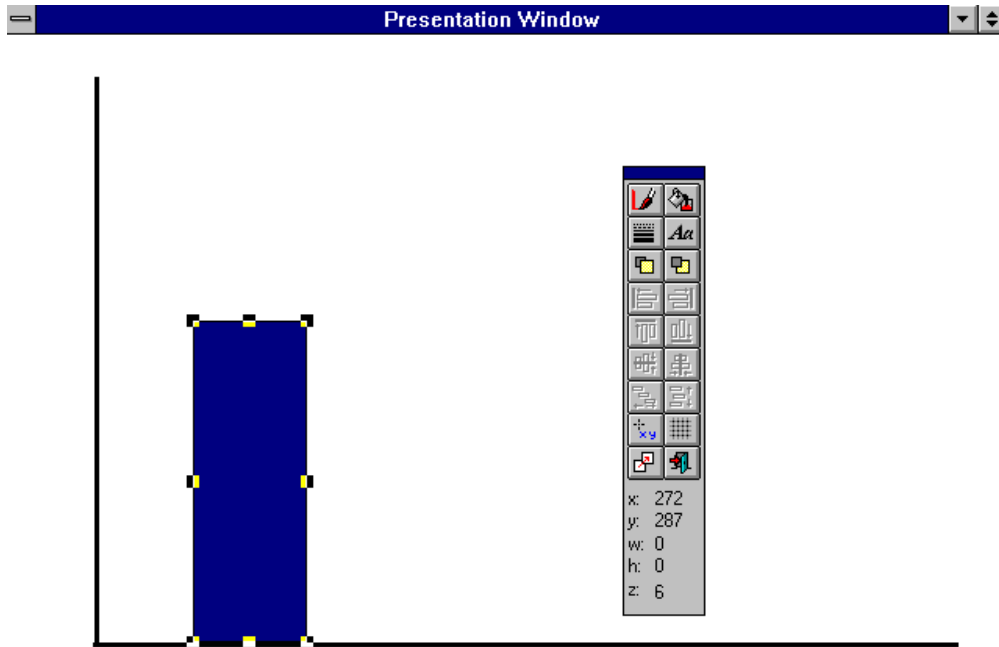


Once you have chosen the *Fill Style*, click on the drop down *Fill Colour* list box to choose a colour for the rectangle. The list box will show the 16 basic Windows colours, but you can mix your own from the *Customize* option at the bottom of the list. This will display the standard Windows colour mixer. Press *Define Custom Colours* to mix your own colour.



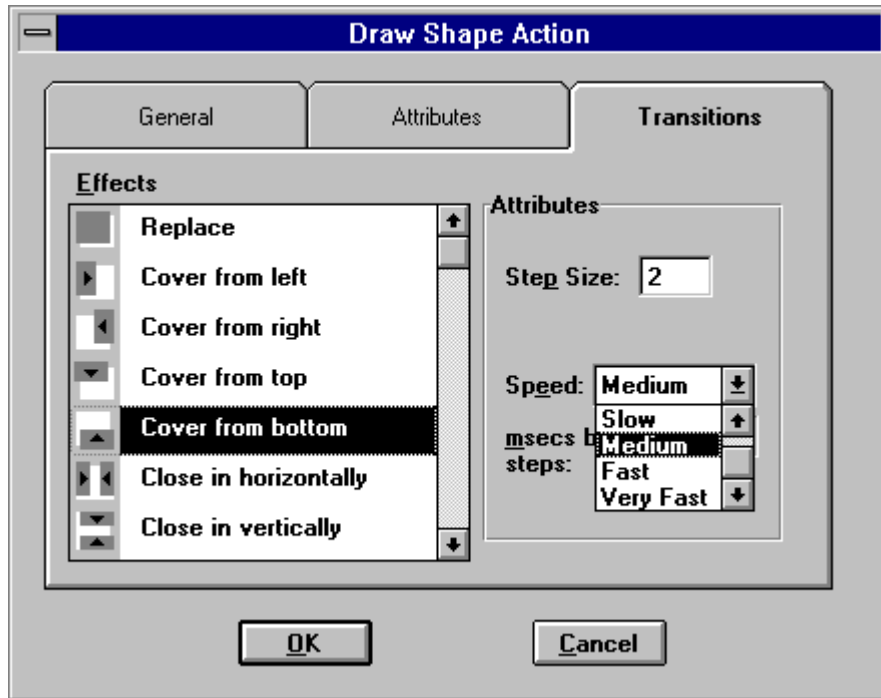
Click and drag the arrow on the right hand side to choose the brightness of the colour (setting the arrow to the top of the scale will always give white). Click and drag the mouse in the colour box to choose the colour itself. The more colours your computer is set up to display, the wider the range of colour shades you will see. Press *Add to Custom Colors* when you have mixed the colour you want to use. It will be added to the *Custom Colors* list on the left hand side. Click once on the colour to select it for use in the *Graphics Shape* action. Click *OK* to return to the *Graphics Shape* setup box.

The rectangle is now ready to be drawn. Click on the *General* tab at the top of the dialog box. Press the *Define* button. The *Presentation Window* will be shown with the two axes already drawn. Position the cursor at the point which will become the top left corner of the first bar in the graph. Click the left mouse button and hold it down. Drag the mouse towards the point which will become the bottom right corner of the bar near the horizontal axis line. Release the mouse button. A rectangle will appear, filled in with the colour and style you selected:

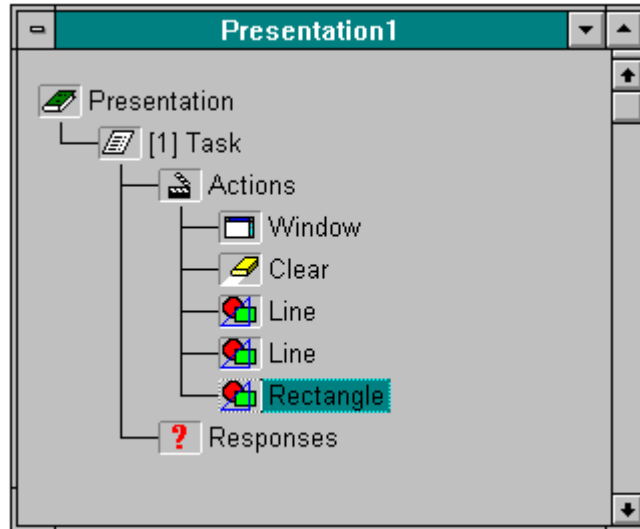



Before creating any more bars in the graph, add a transition effect to this first bar so that when the presentation is run, the bar will appear to grow out of the bottom axis.

Double-click on the rectangle you have just created to display its set-up dialog box again. Click on the *Transitions* tab. Click once on the *Cover from bottom* item in the *Effects* list. Click on the drop down *Speed* list and choose *Medium*. This controls how quickly the bar graph will appear to rise.





Click OK to accept these settings. Press the *Close Presentation Window* button in the floating tool bar to return to the *Design Window*, which should now look like this:

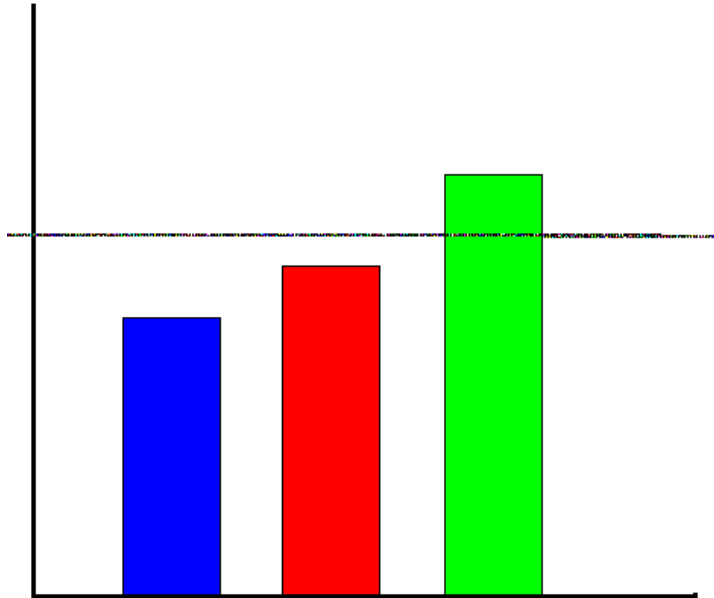


To see how the presentation looks at the moment, select the *From Start* option in the *Run* menu, or press the *Run*  button in the tool bar. If you want to change the speed of the bar rising or its colour, simply double-click on the *Graphics Shape* action labelled *Rectangle* in the *Design Window* and choose and change the appropriate setting.

Now let's add in two more bars to the graph. This could be done by dragging in more *Graphics Shape* actions from the *Icon Window* and repeating the set-up steps described above. However, there is a faster way: the rectangle you have already set-up can be copied.

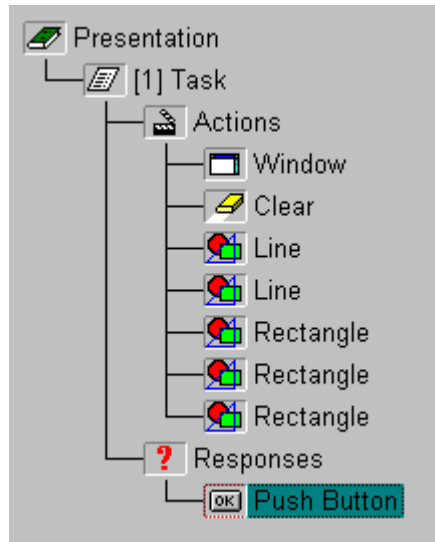
Double-click the *Graphics Shape* action labelled *Rectangle* in the *Design Window*. Press the *Modify* button to display the *Presentation Window* with the rectangle highlighted with square "handles" around it. Click once on the *Copy Objects*  button in the floating tool bar. This makes an exact copy of the object - duplicating its size, colour, position and transition effects. It lies directly on top of the original object, so to separate the two, move the cursor over to the rectangle, click and hold down the left mouse button and drag the mouse to move the new rectangle to a position next to the original one. To change the height of this bar, click and drag the square "handle" at the top of the rectangle. To change the colour of the bar, click the *Set Bk Colour* button  in the floating tool bar.

More bars can be created by repeating this process. The *Presentation Window* should now look something like this:




Press the *Exit* button in the floating tool bar  to return the *Design Window*.


To keep the graph on the screen until the user is ready to move on, a *Response* needs to be added. Click and drag the *Push Button* response from the *Icon Window* to the *Design Window*. Release the mouse button when the cursor is over the *Responses* icon of the task. The *Push Button* will be added to the task:



The text on the button can be modified by double-clicking on the *Push Button* icon in the *Task* and then typing in the new button text where it says *Continue*. The position of the button can be changed by pressing the *Modify* button. The button will appear with 'handles' around it indicating that it is the currently active object:



Click and drag the corner box 'handles' to change the size and shape of the button. The button can be moved around the picture by clicking in the middle of the outline box and dragging it around the *Presentation Window*. When you are happy with the size, shape and position, click the *Exit* button  in the floating tool bar or press the *Escape* key on the keyboard.

To run the presentation, click the *Run* button  in the tool bar. You should now see the graph being drawn, with the bars appearing one by one. Clicking the push button will end the presentation in this case, or move on to the next screen when you have added further tasks.



---



## Adding digital video

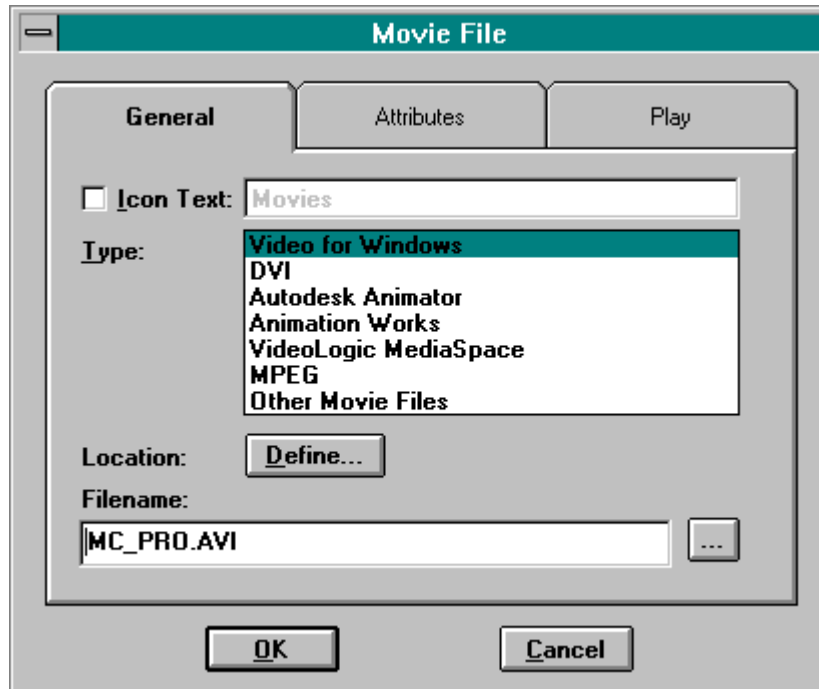
Dazzler supports a wide range of digital video formats, from Video for Windows played back with no additional hardware through to the latest high quality MPEG video, usually played back using an MPEG card added to your computer. However, whichever type you want to play back, it is exactly the same procedure in Dazzler and is as easy as displaying a still picture shown in the earlier tutorials.


First of all find a video file. You can create one yourself if you have a video capture card and suitable capture software. There are many low cost cards available - contact your dealer for advice on the latest ones.

In this tutorial, we'll use a Video For Windows (AVI) clip supplied with Dazzler.


Start a new presentation using *New* from the *File* menu or by pressing the *New* button  on the left hand side of the tool bar. Alternatively, add a new task to the end of an existing presentation by pressing the *Add Task* button  in the tool bar.

Now drag and drop a *Movie* action  from the *Icon Window* onto the *Actions* icon  in the presentation. Double-click on the icon to bring up the *Movie* set-up dialog box.



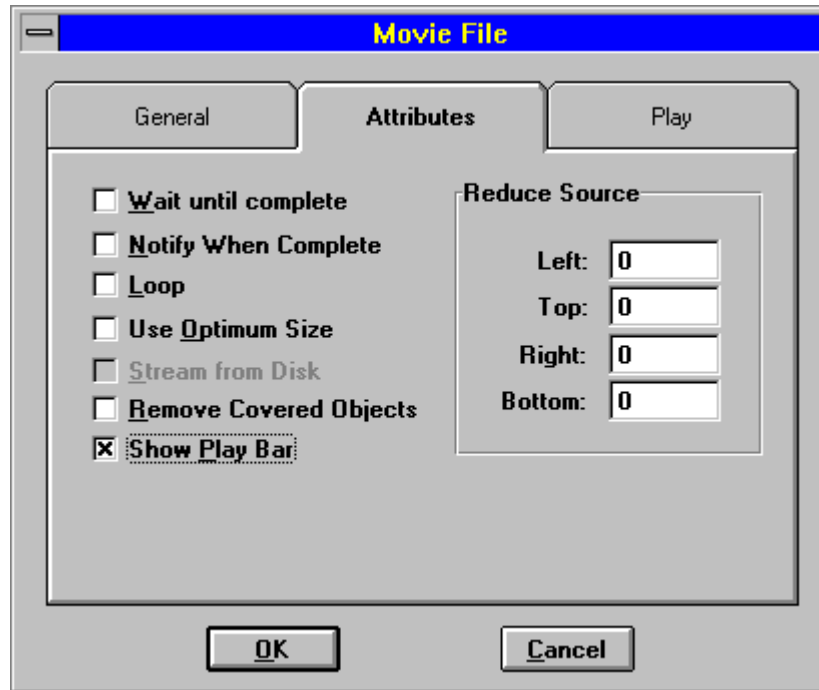
From the *Type* section choose *Video for Windows*. To select a particular file, press the browse button  towards the bottom right of the dialog box. This will display the *Movie* file open dialog box.


Browse for the movie you want to play. The movie file can be previewed using the *Show Preview* check box. Click once on the file name to play it. The *Preview* controls beneath the video clip can be used to start, stop and move through the clip to check that it is the clip you want to use. When you have found a suitable clip, press the OK button to return to the *Movie* action set-up. To define where the movie clip is to be shown in the *Presentation Window*, press the *Define...* button. The *Presentation Window* will now

appear. The cursor will change to: . Move the mouse to the point where you want the top left corner of the picture to be located. Press the left mouse button, and, while holding it down, drag the mouse to the position of the bottom right corner. As you drag the mouse you will see the first frame of the movie being 'painted' into the *Presentation Window*. Release the mouse button. The movie can be resized and moved by clicking and


dragging the square ‘handles’ at the edges of the movie or by clicking and dragging the mouse somewhere inside the frame itself.

To add a *Play Bar* so that the user can stop and start the movie themselves, double click on the movie object in the *Presentation Window* to display its set-up dialog box again. Click on the *Attributes* tab and click on the *Show Play Bar* check box at the bottom.



Press the OK button to return to the *Presentation Window*. The first frame of the video clip should now be shown with a play bar underneath. Press the *Exit* button  in the floating tool bar to return to the *Design Window*.

To keep the movie on the screen until the user is ready to move on, a *Response* needs to be added. Click and drag the *Push Button* response from the *Icon Window* to the *Design Window*. Release the mouse button when the cursor is over the *Responses* icon of the task. The *Push Button* will be added to the task: Let's just use its default text ("Continue") and position (bottom centre) so that no set-up needs to be done.

To run the presentation, click the *Run* button  in the tool bar. The *Presentation Window* should appear and the video clip will play at the position and size you defined. The video can be paused using the button at the bottom left of the clip and moved through by clicking and dragging the scroll bar next to it. Restart the video by clicking the pause button again. If the Continue button is pressed at any time the presentation will end or move on to another part if more tasks have been added.





---

## Variables (VARS.DZL)

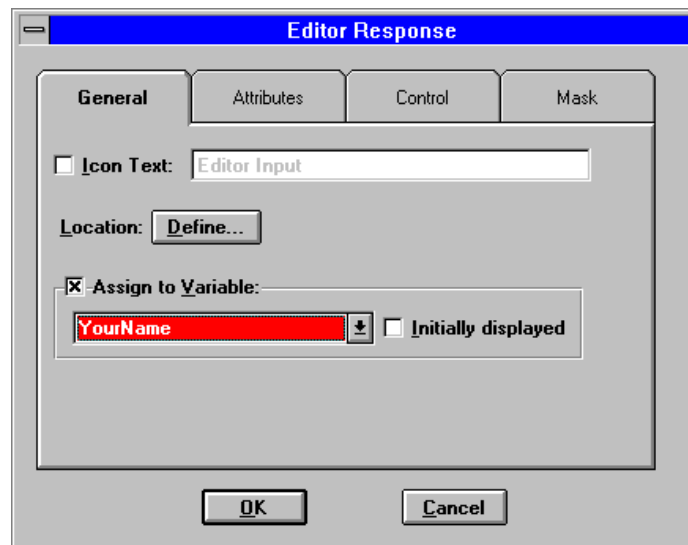
Dazzler stores information used in a course in *variables*. Some of these, such as *CourseTitle*, are already built in as part of the automatic student logging; others you can create to store information relevant to a particular course, for example *JobTitle*.

In this example (vars.dzl), the students are prompted for their name and age. A *Conditional Branch* is made based on their age. It illustrates both the use of creating your own variables and how they can then be used in actions and responses.

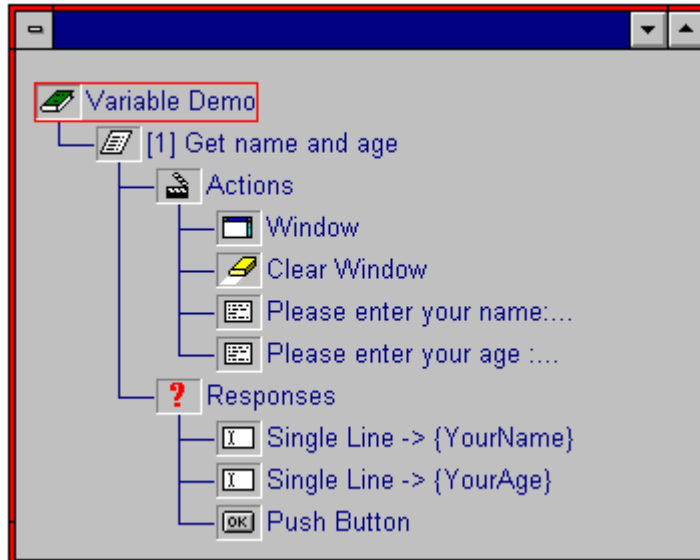
Start a new course and add two new variables to the course using the *Variables* menu item in the *Course* menu. Call one *YourName* and make it a *Text* type (refer to the paper-based or on-line *Reference Manual* for details of the various options in the Variables dialog box). Call the other one *YourAge* and make it *Numeric* with 0 decimal places.

Now drag and drop some actions into the course - set-up the window, clear it and add some text asking the student to enter their name and also their age.

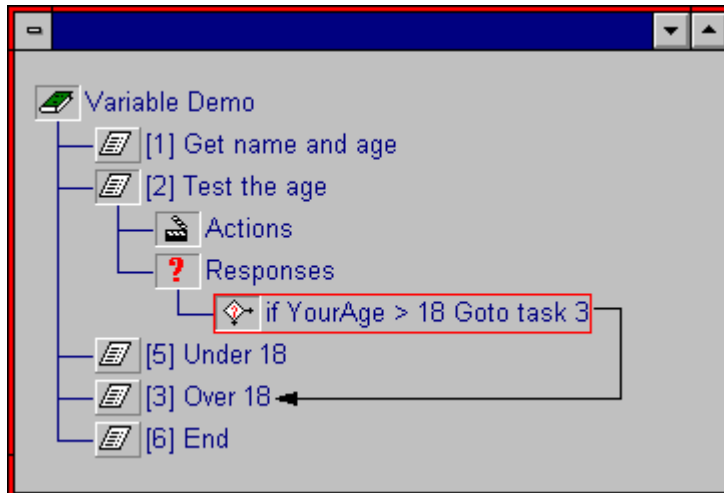
To get the information from the student and into the course, add two *Editor Input* responses. Set them both to have borders and to be single line. Also, use the *Assign To Variable* option to store the result of the student's typing in the variables just set up, as shown below:



Add a default *Push Button* response which will allow the student to continue with the course, once they have entered the details and pushed the Continue button.



The testing of the *YourAge* variable is done with a *Conditional branch* response which must be put into a separate task.



Use the *Insert Variable* part of the *Conditional branch* set-up box to create an expression to be tested:

The other tasks added to the course are linked to the conditional branch depending on whether the expression is true (therefore branch), or not (and so carry on to the next task).

---

## Multiple Answer Questions (MULTICH.DZL)

### Summary

Sometimes it is useful to present the student with a range of possible answers to a question, some of which are right (as opposed to only one answer being correct). Check boxes are used to present the choices, and a combination of variables and conditional branching is used to test whether the right choices have been made. In this example, 3 out of 5 answers will be correct.

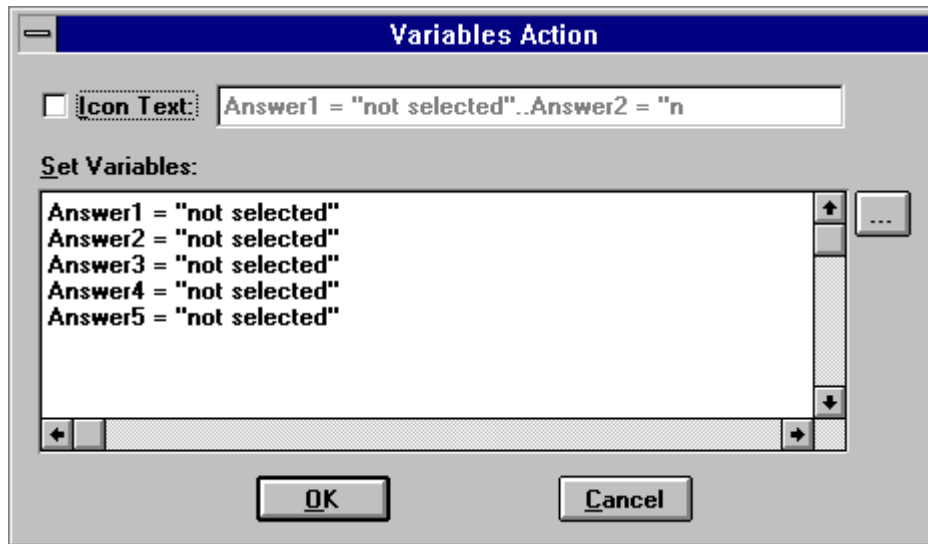
### Procedure

1. Create some variables to hold the student's choices. Use the *Course/Variables* menu option:

The screenshot shows the 'Variables' dialog box. The 'Name' field contains 'Answer5'. The 'Type' dropdown is set to 'Text'. The 'Format' section shows 'Display Numeric to' set to '2' and 'decimal places'. The 'ODBC' section has a 'Record Set:' field and a 'Setup...' button. On the right, the 'Local' radio button is selected. The list of variables includes Answer1, Answer2, Answer3, Answer4, Answer5 (highlighted), CorrectChoices, CourseDrive, CoursePath, CourseTitle, NoOfQuestions, PercentageCorrect, QuestionsAnswered, QuestionsCorrect, QuestionsRepeated, QuestionsWrong, StudentID, StudentName, and TaskTag. At the bottom are 'OK' and 'Cancel' buttons.

2. Start a new course and add a second task to initialise the variables, ask the question and record the choices made.

Add a *Variables* action and setup the variables created in the first step:



3. Create five *Check Button* responses as check boxes with the answer text entered into the *Title* field of the set-up dialog box - for example:

Which of the following statements are true ?

Answer 1

Answer 2

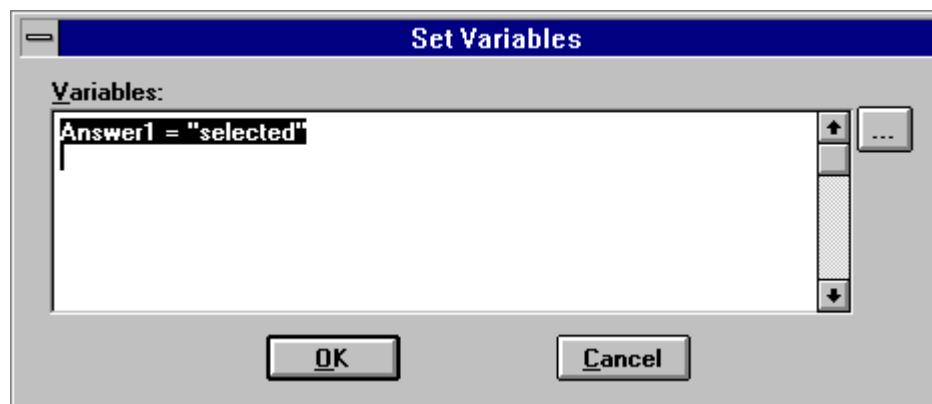
Answer 3

Answer 4

Answer 5

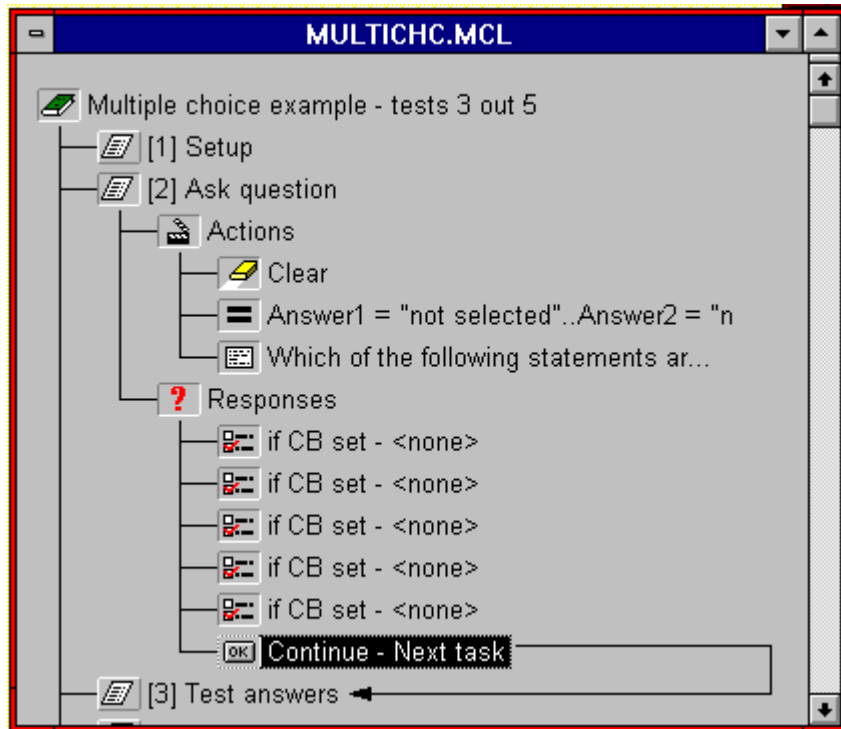
Continue

4. For each check box, use the *Set Variables* button in the *If Checked* part of the setup dialog box:

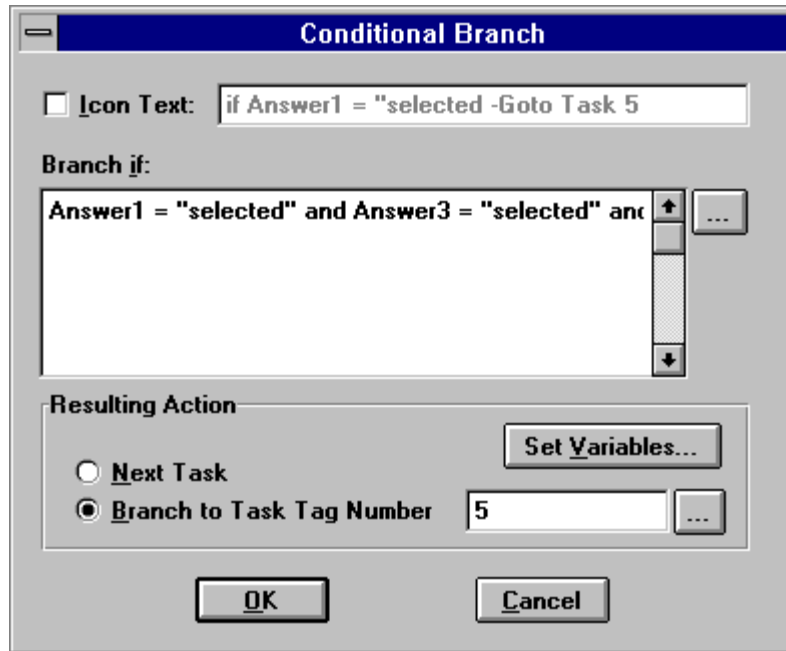


In this way, if the student has selected this option when the course moves on to the next task, the variable *Answer1* will have changed from “not selected” to “selected”, and so forth.

5. Add a *Push Button* response to the task which the student must press to accept the choices made.



6. Add another task which will be used to test if the right choices have been made. This task will consist of a single Conditional Branch response with no actions. The condition entered into the setup dialog box is:

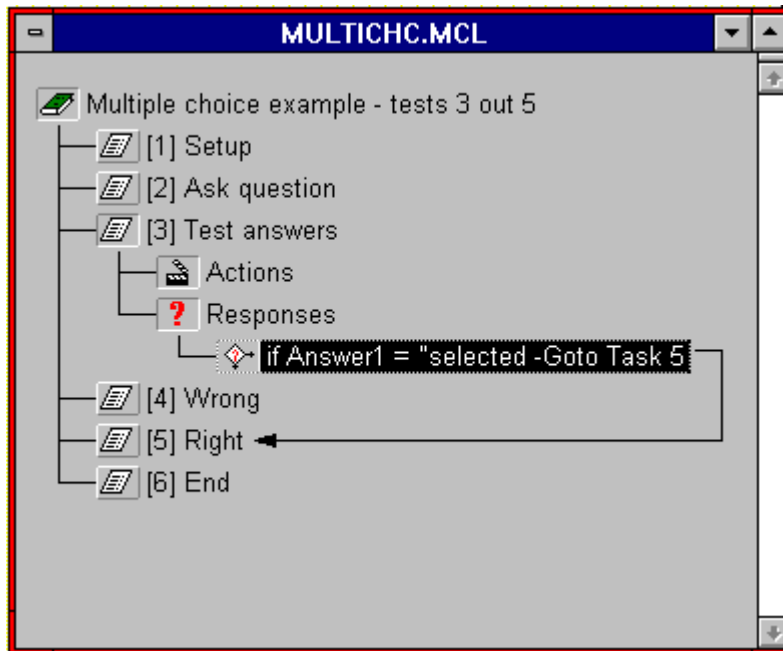


where the full expression is :

Answer1 = "selected" and Answer3 = "selected" and Answer4 = "selected" and Answer2 = "not selected" and Answer5 = "not selected"

where answers 1, 3 and 4 are the correct ones. The expression contains a check to see if the wrong answers have not been chosen, to avoid the situation where the student selects all possible options, both right and one ones.

7. Setup two more tasks, one to tell the student they made the wrong choices and to let them try again, the other to tell them that they got the choices right. Link the conditional branch response to the 'right' task. If the conditional test fails, it will fall through to the 'wrong' task.



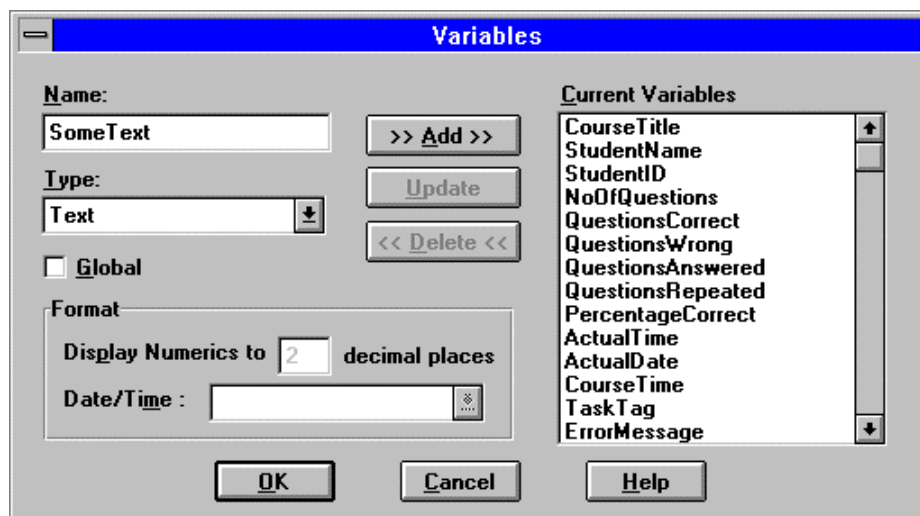
---

## Scrolling text boxes (SCRLTEXT.DZL)

The Text Action in Dazzler doesn't add scroll bars if the text falls outside the rectangle defined in the Presentation Window. Instead, the text is wrapped and then cut if it extends beyond the bottom of the rectangle. However, it is possible to create scrolling text boxes by assigning the text to a variable and displaying it using the Editor Input response as follows:

1. Create a variable to be used to store the text from the *Course/Variables...* menu.

Type in the variable name in the Name box as follows:

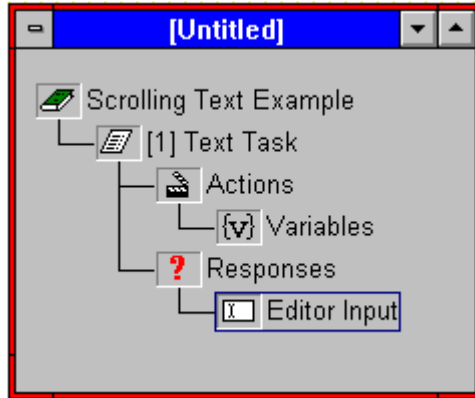


Press the *>>Add>>* button and press *OK*.

There are two ways to store the text in the variables which has been just been created. The first is add a line of text directly using the Variable action. The second is to read text from a plain text file using the Read From File action.

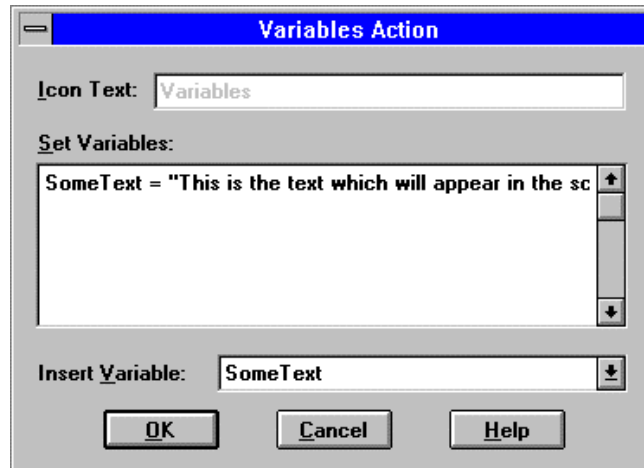
*Method 1:*

a) Add a Variable Action and an Editor Input Response to a task in the course by dragging them from the Icon window.



b). Now double-click on the Variables action to set it up. Use the *Insert Variable* button to browse for the variable created earlier. Now enter the text you want to appear in the scrolling text box, with double quotes “...” at each end.

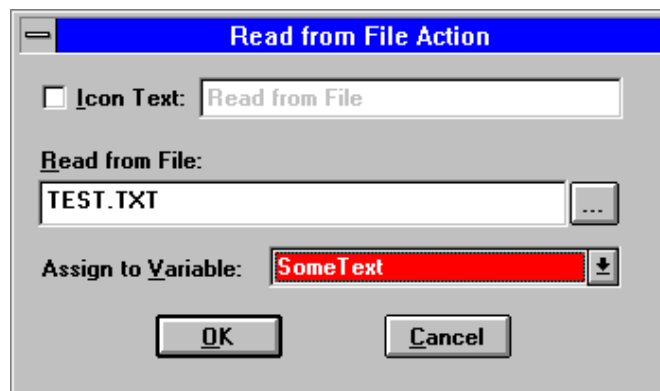
n.b. text can be pasted from other applications by using the standard Windows short-cut keys *Ctrl* and *V*.




Click OK. This variable is now ready to use in the Editor Input response.

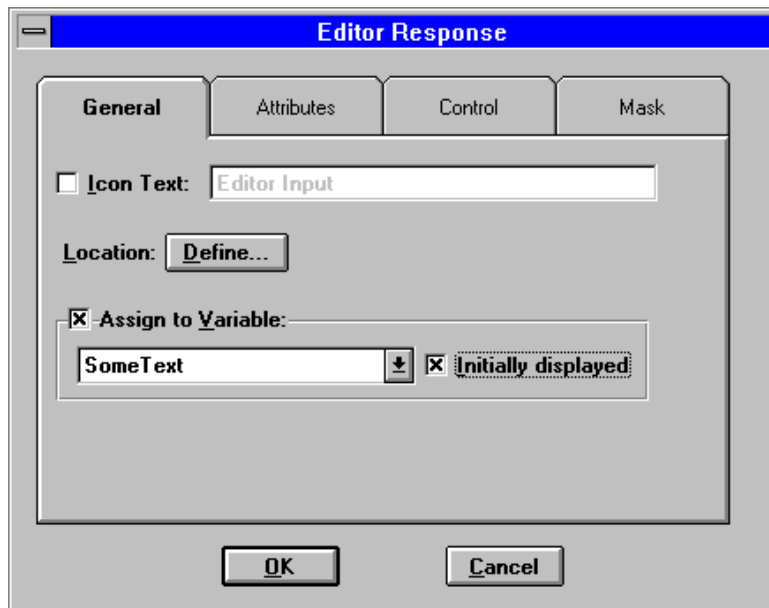
*Method 2:*

a) Drag a Read From File action into the course. Double-click on the icon to show the set-up dialog box:

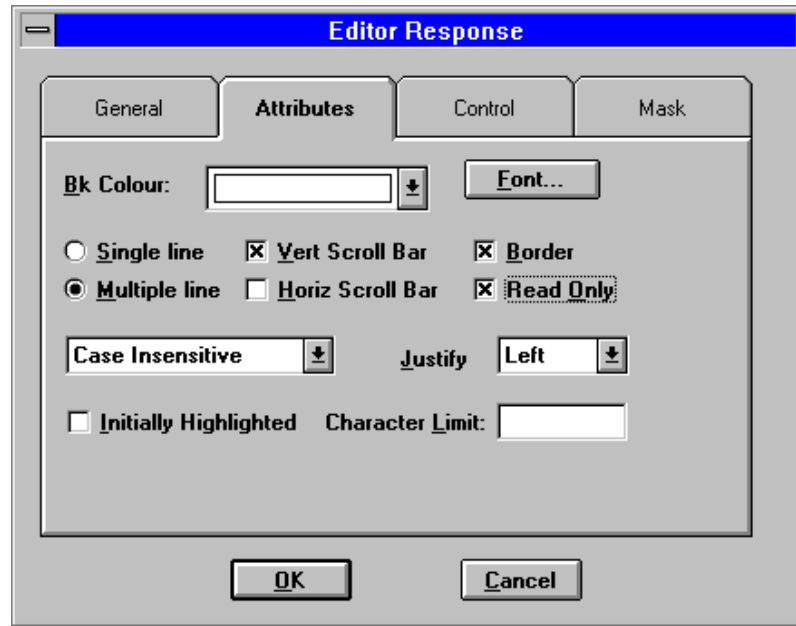


- b) Browse for the text file to be read using the browse button 
- c) Choose the *SomeText* variable created earlier from the *Assign to Variable* list.
- d) Click OK. This variable is now ready to use in the Editor Input response

4. Double click on the Editor Input response in the task.

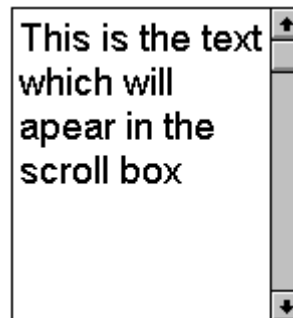


Select the *Assign to Variable* check box at the bottom and pop down the variable list.  
Choose the variable used to store the text. Select the *Initially displayed* check box.



Set the other *Attributes* options as shown: *Multiple Line*, *Border*, *Vertical Scroll Bar*, *Read Only*.

5. When the course is run, the text will appear with a scroll bar as follows:



---

## Random Selection (RANDOM.DZL)

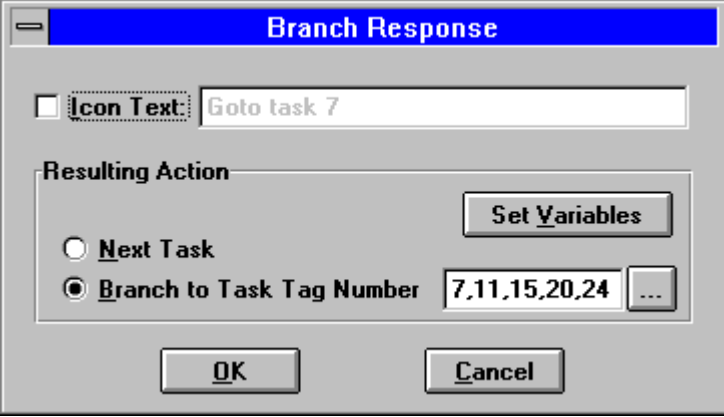
### *Summary*

Every response in Dazzler can branch either to one specific task or randomly select from a list of task numbers. This can be used to ask a student a certain number of questions at random from a set of questions.

### *Detail*

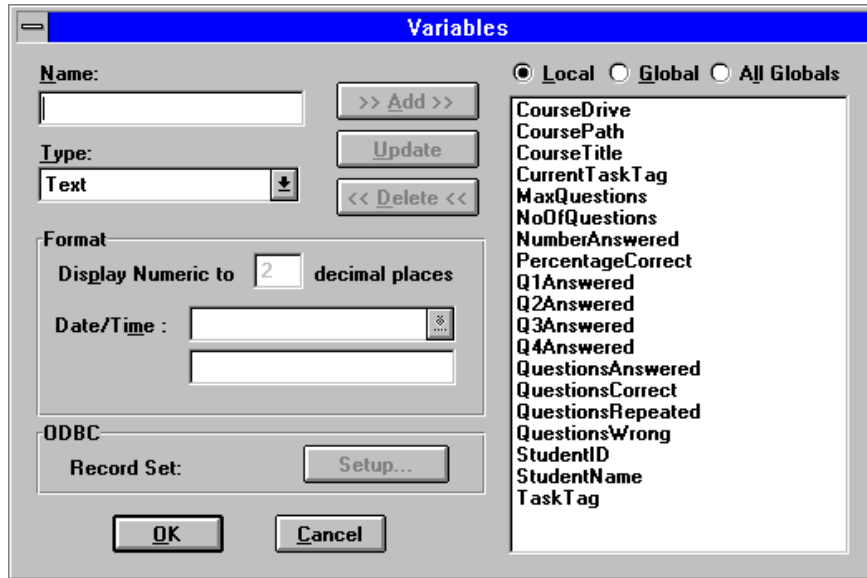
To allow random branching from a response, use the Branch to Task Tag Number box in any response and enter the Task numbers to be included in the random selection, separated by commas.

For example, the Branch Response could be setup as follows:



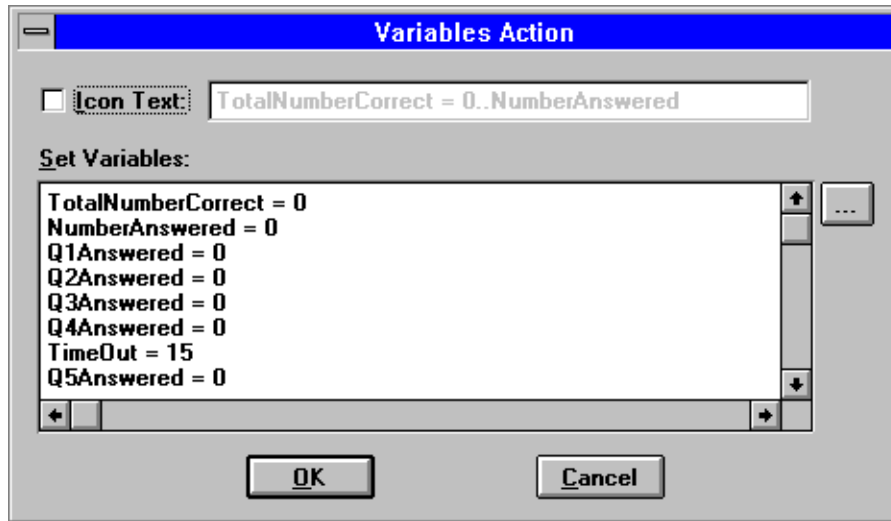
The screenshot shows a dialog box titled "Branch Response". It features a blue header bar with the title. Below the header, there is a checkbox labeled "Icon Text" which is currently unchecked, followed by a text input field containing the text "Goto task 7". Underneath this is a section titled "Resulting Action" which contains two radio button options: "Next Task" (which is unselected) and "Branch to Task Tag Number" (which is selected). To the right of the "Branch to Task Tag Number" radio button is a text input field containing the numbers "7,11,15,20,24" and a small button with three dots. Above the "Branch to Task Tag Number" radio button is a button labeled "Set Variables". At the bottom of the dialog box are two buttons: "OK" and "Cancel".

Variables can be used to test if a particular question has already been asked and whether the maximum number of questions specified has already been presented. In the example below, two out of four questions will be asked randomly:

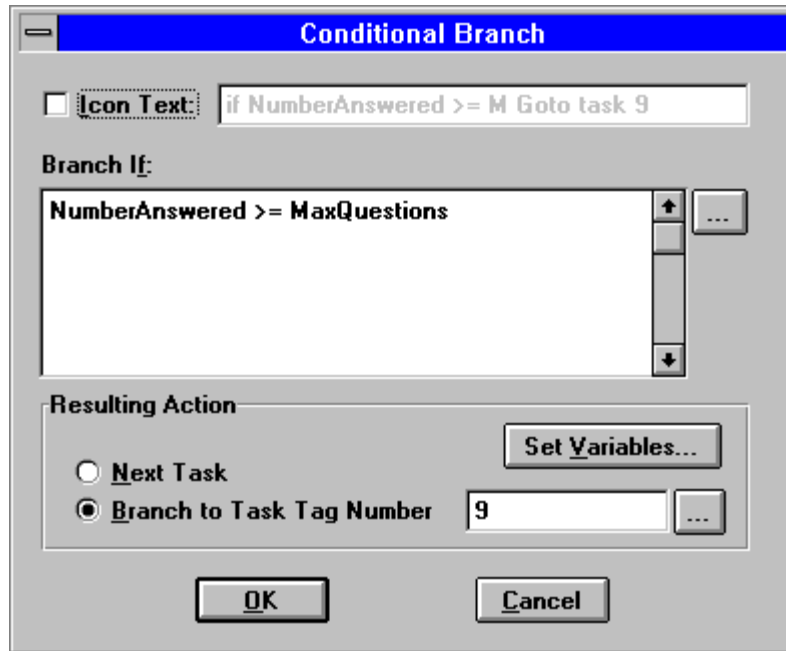


The variables created are all numeric.

In the course itself, the variables are first of all initialised using a Variable Action:

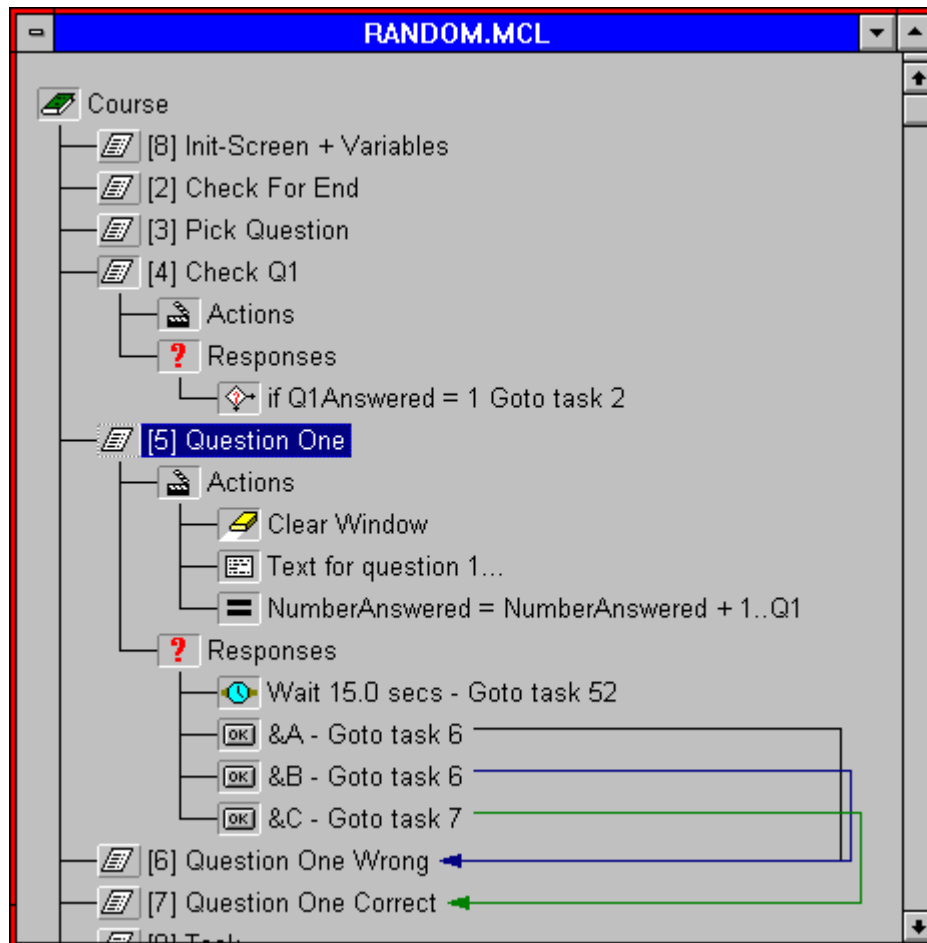


In the next task, the first one in the question loop, do a check to see if the maximum number of questions has already been asked using a Conditional Branch Response. If it has, then branch out of the loop to the end of this part of the course:

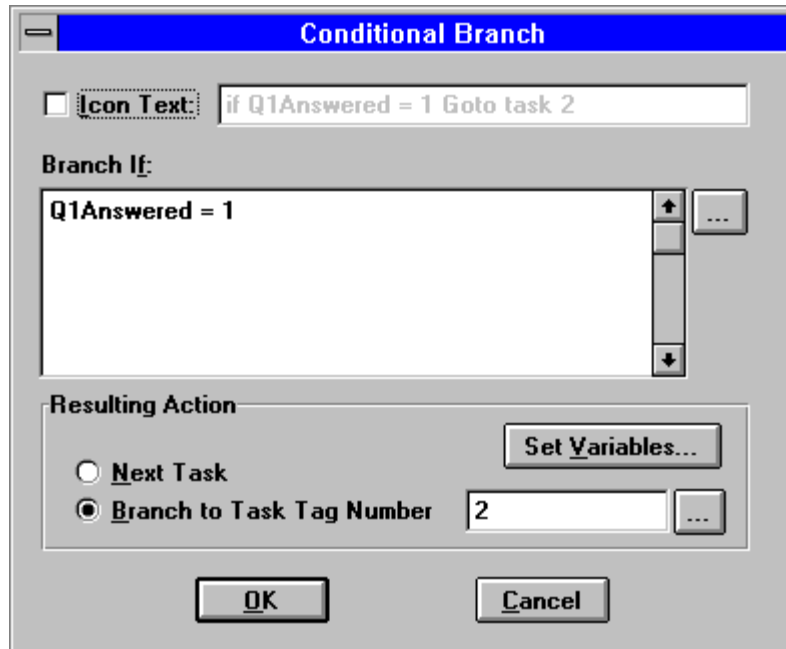


In the next task, randomly select one of the question tasks using the Branch response shown above.

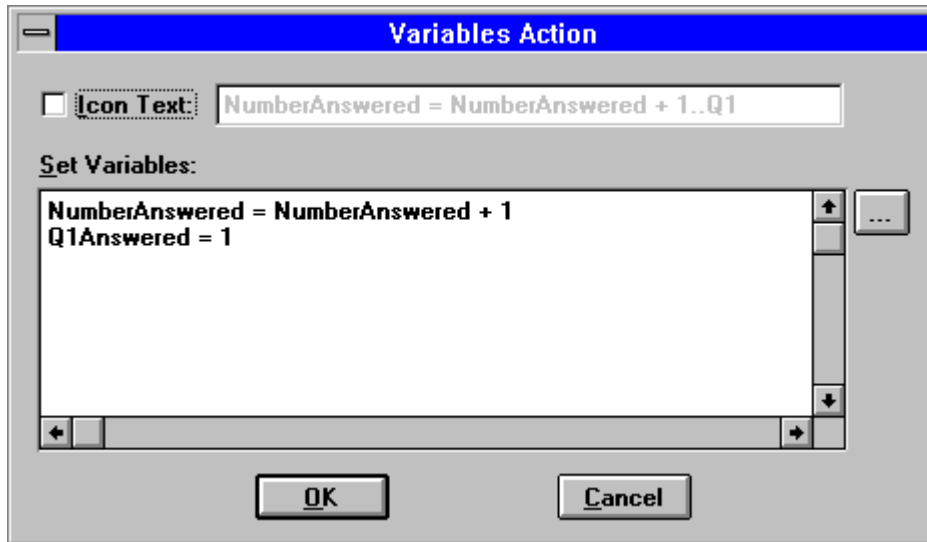
Next add in sets of tasks for the individual questions:



The first task in the sequence (Check Question 1), contains a conditional branch. This loops back to the random selection task if the Q1Answered variable has already been set, indicating that this question has already been asked.



The next task contains the questions itself, as well as a Variable action to record the fact that this question has been asked and to increment the questions asked counter:



The button responses allow the student to choose an answer. These link to the 'Right' and 'Wrong' tasks which then loop back to the task which checks if the maximum number of questions has been asked. Thus, the process of checking the number of questions asked and choosing a new question at random continues.

---

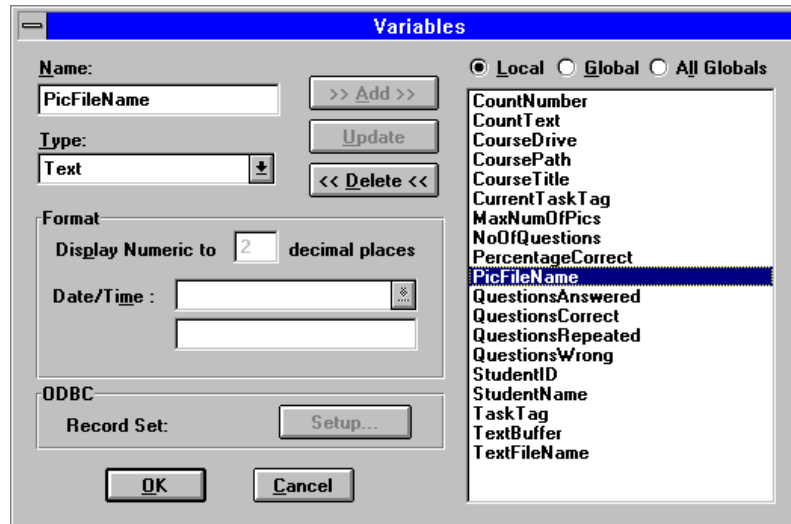
## Slide Show (SLIDEVAR.DZL)

### Summary

This demonstration course uses six tasks to create a slide show loop which will display bitmaps saved in the sequence PIC1.BMP, PIC2.BMP, etc., and overlay text associated with each picture from files TEXT1.TXT, TEXT2.TXT, etc. The slides are selected using Previous and Next push buttons. Error checking is carried out to stop the user from moving beyond the first slide or the last one as defined in the variable MaxNumOfPics. Each slide is displayed in the same position with the same transition effect, and the text has the same position, font and colour.

### Points to note:

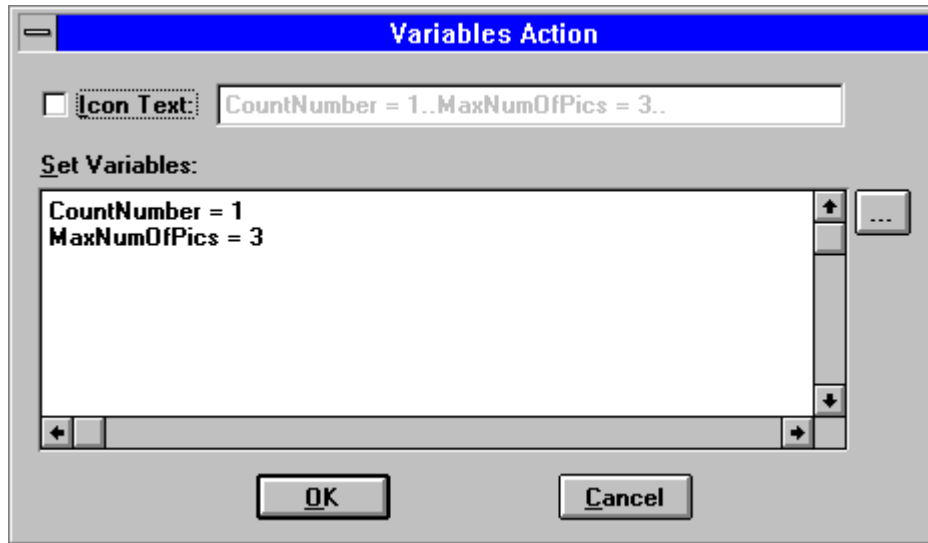
1. First of all, add the variables via the *Course/Variables* menu



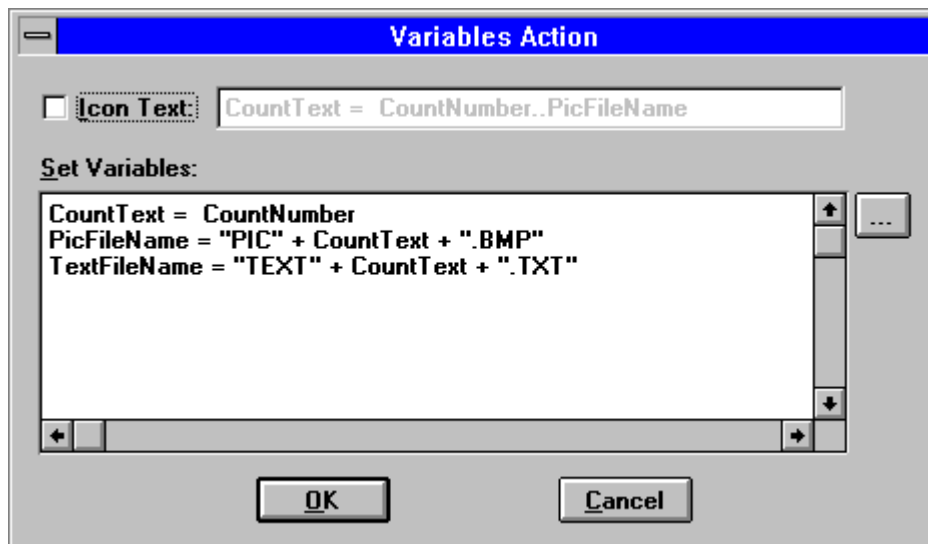
*PicFileName*, *CountText*, *TextFileName* and *TextBuffer* are text variables.

*CountNumber* and *MaxNumOfPics* are numeric with 0 decimal places.

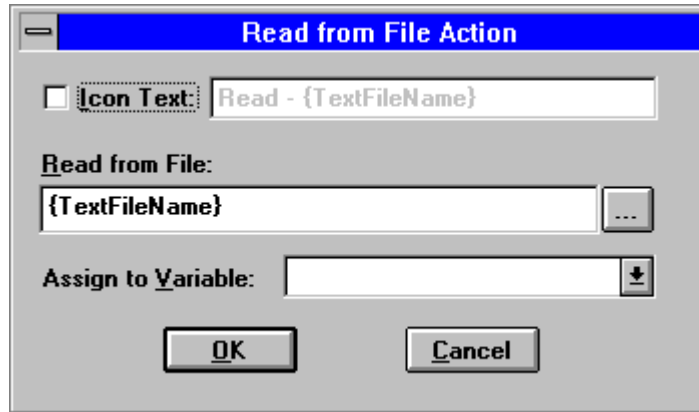
2. The first task has a variable action to initialise the variables:



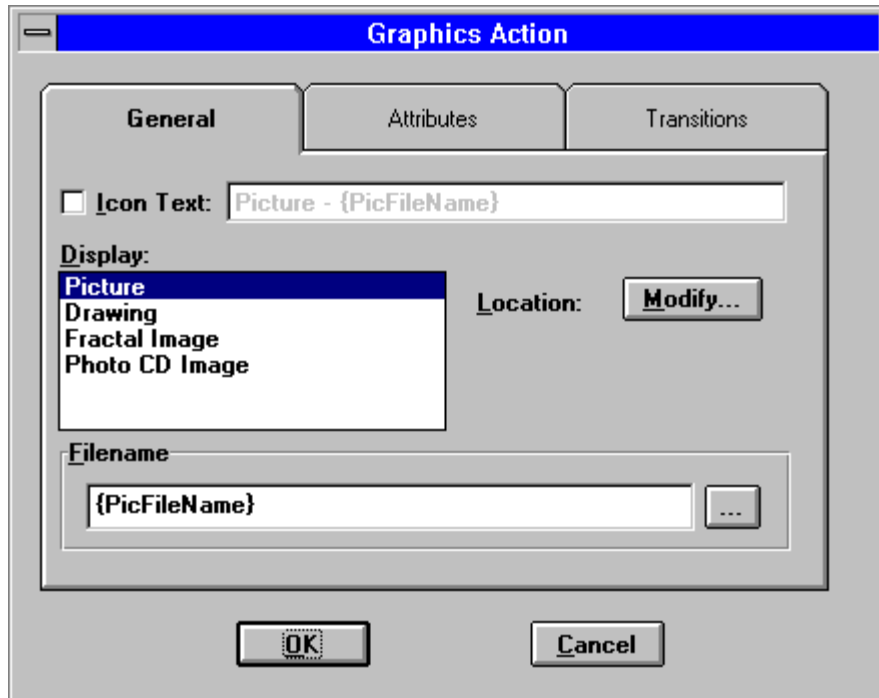
3. The main loop task has a variable action to create the picture and text file names:



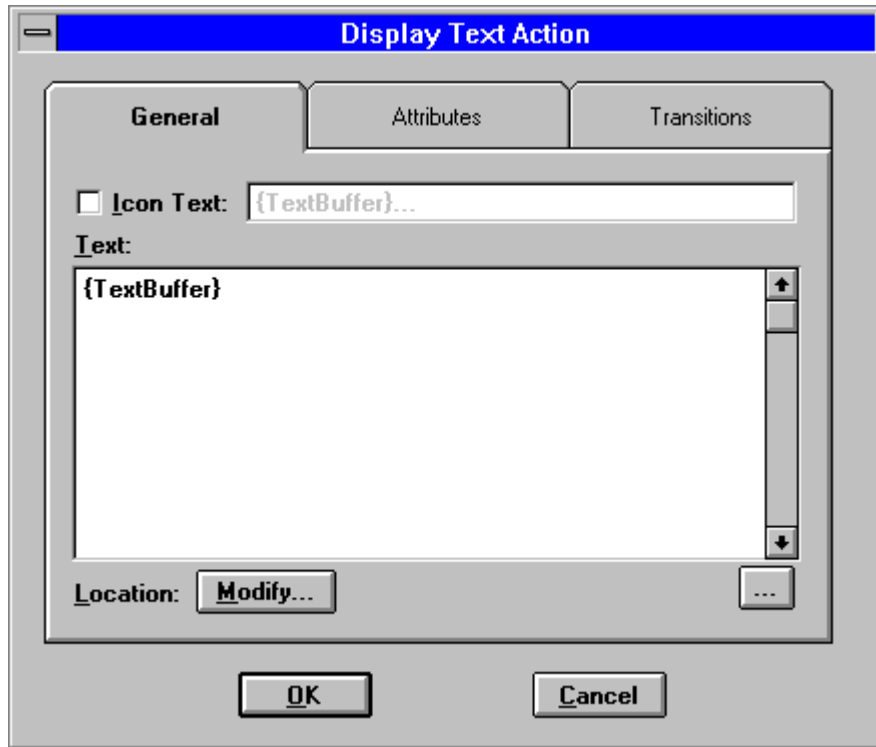
4. The *Read from File* action uses the *TextFileName* variable to identify the correct file to read and assigns the text read in to the *TextBuffer* variable:



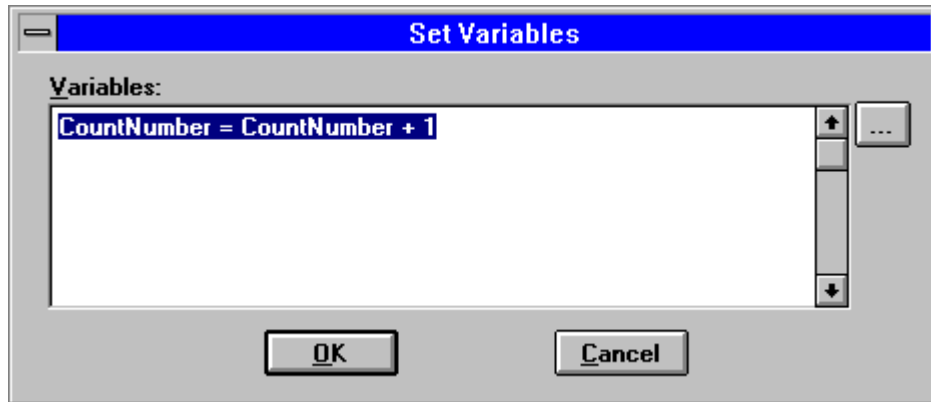
5. The picture file is specified through adding the variable name in parentheses {} in the *FileName* edit box:



6. Similarly, the text action displays the contents of the *TextBuffer* variable:



7. The push buttons, as well as providing the looping links, also increment and decrement the *NumberCount* variable through the *Resulting Action / Set Variable* part of the *Push Button* response dialog.



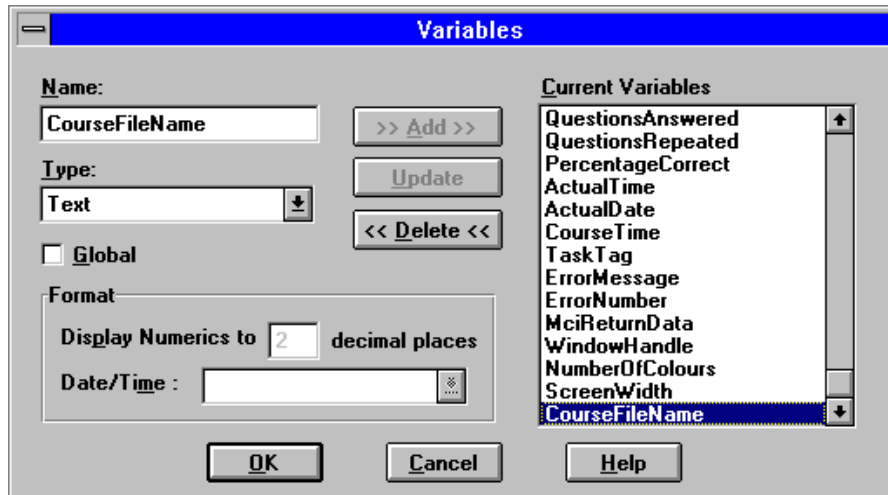
---

## Using variables to run course modules (RUNVAR.DZL)

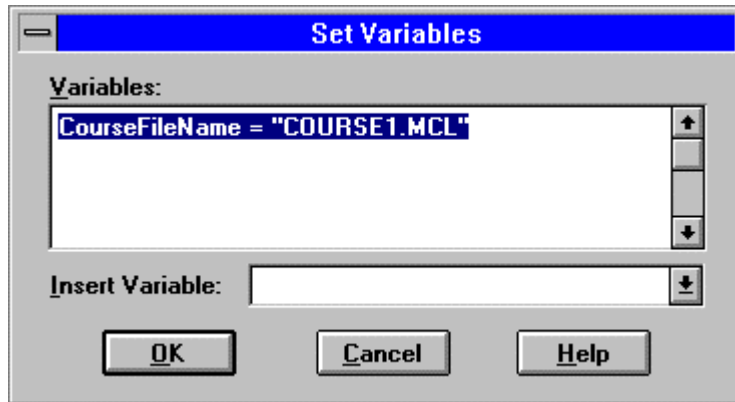
The use of variables can cut down the number of tasks needed when allowing a choice of modules to be run.

### *Techniques*

1. Create a variable from the *Course/Variable* menu.



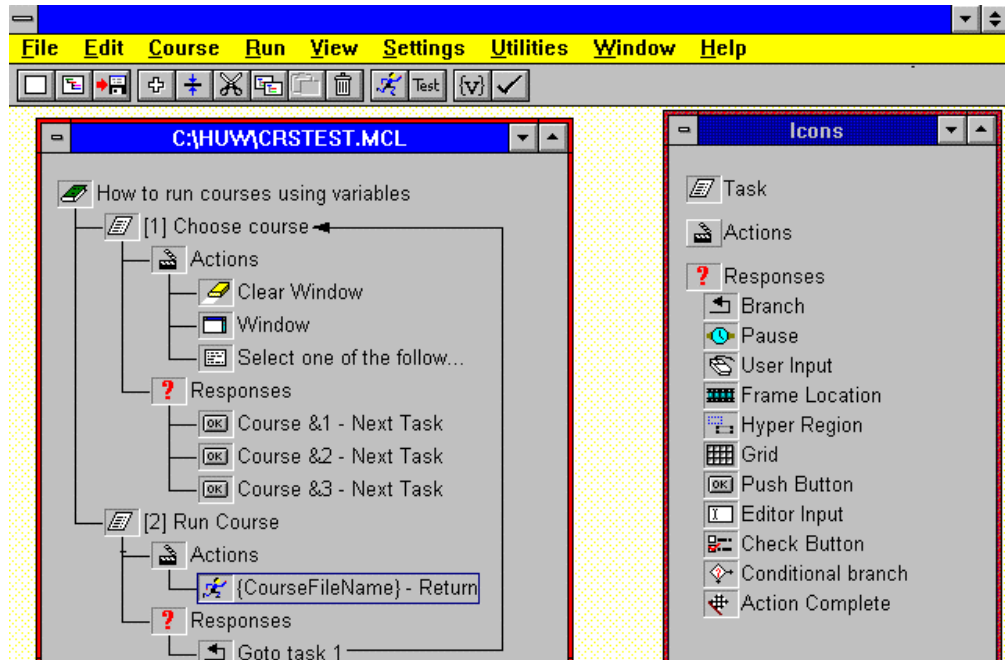
2. Use the *Set Variables* function in the *Push Button* response to set which course will be run when the button is pressed.



3. Link the button to a task which has a *Run Another Course* action, and put the variable name in the *Course Filename* box.



The finished course should look like this:





---

## Using variable functions (VARFILE.DZL)

This tutorial shows how variable functions can be used to manipulate data. The sample course does the following:

- Reads in a text file of names and IDs.
- Processes the text file to create a list box of student names using the *GetElement* function.
- When a name is chosen, the ID entered is compared with the corresponding entry stored in the file.

The student data is stored in a plain text file. The name and ID are separated by a comma

For example:

```
Joe Bloggs,12345  
John Doe,987567  
Janet Smith,675645
```

### *Step 1*

Create some variables to store information through the *Course/Variables* menu:

*Text:*

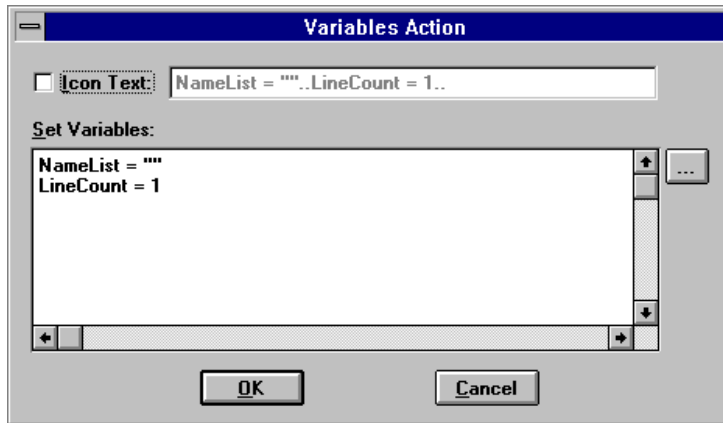
StudentFile , LineBuffer, NameBuffer, NameList

*Numeric:*

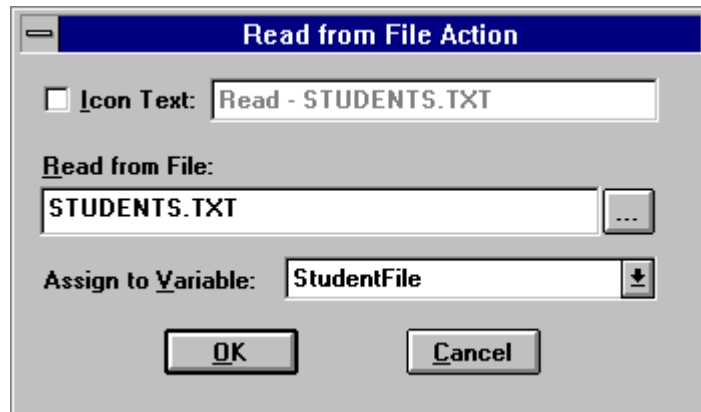
LineCount

### *Step 2*

Initialise some of the variables

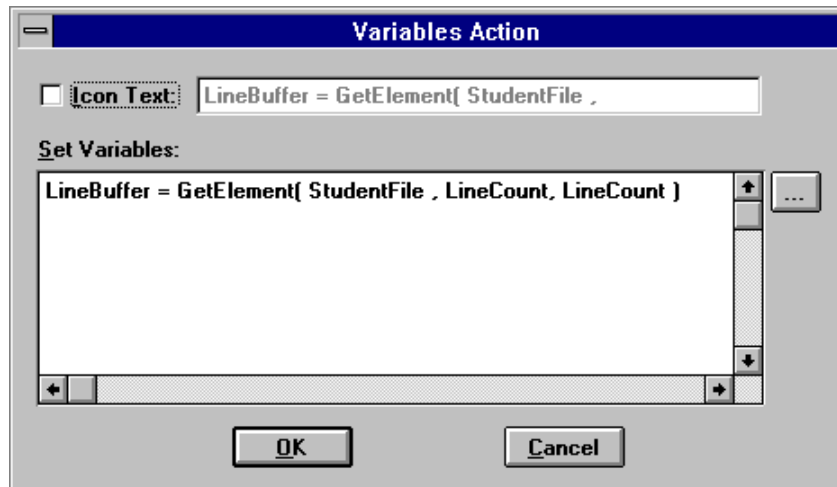


Read the file into a variable using the *Read From File* action.



*Step 3*

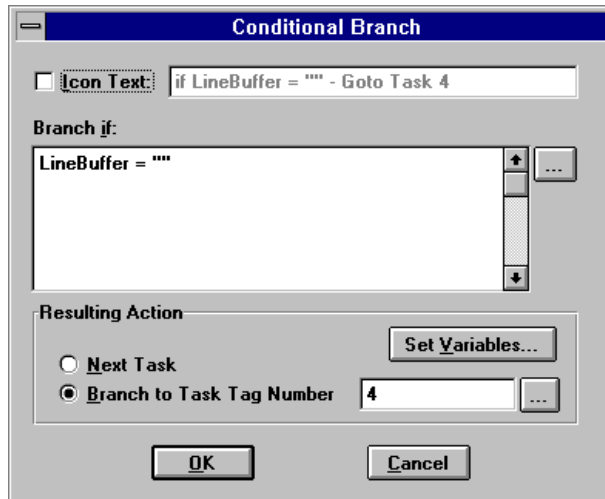
Add another task and create a *Variables* action to process the file data line by line:



*GetElement* here looks for the default *Carriage Return* (end of line) character when dividing the file up into separate components. Thus, after reading the first line, *LineBuffer* should contain

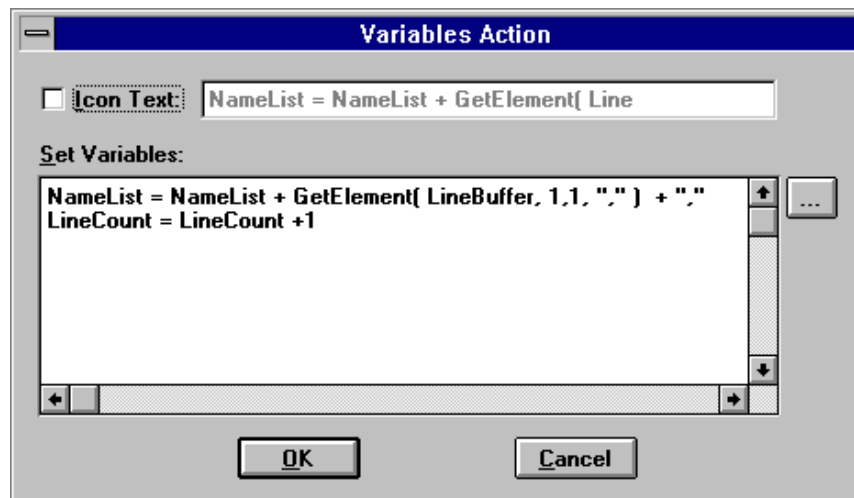
Joe Bloggs,12345

Now add a Conditional Branch response to this task to check if *LineBuffer* is empty (i.e., the end of the file data has been reached).



**Step 4**

Add another task to process the line of text. Use another Variable action as follows:

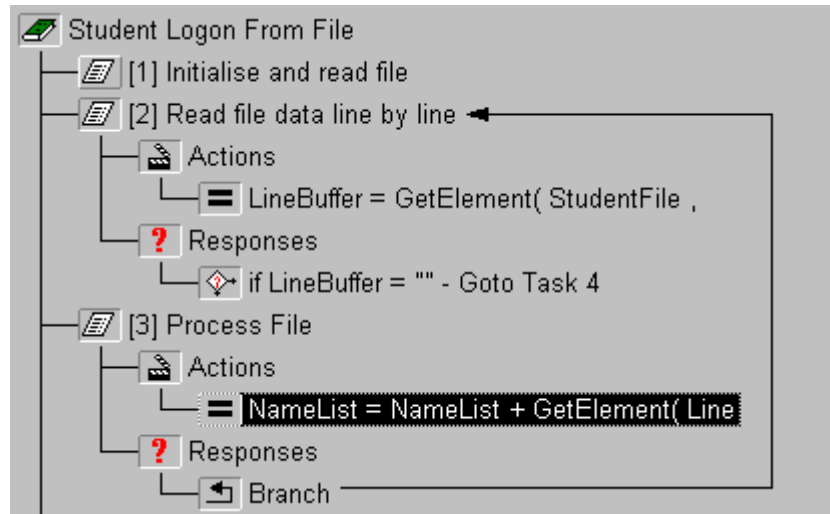


*GetElement* here is used to extract just the first part of the line (i.e., the name). The name is separated from the associated ID by a comma ",", and so a comma is put into the

*GetElement* expression to tell the function to break up the text by commas rather than by the default Carriage Return character. The *NameList* variable will contain a growing list of names, separated by the commas added as part of this variables action.

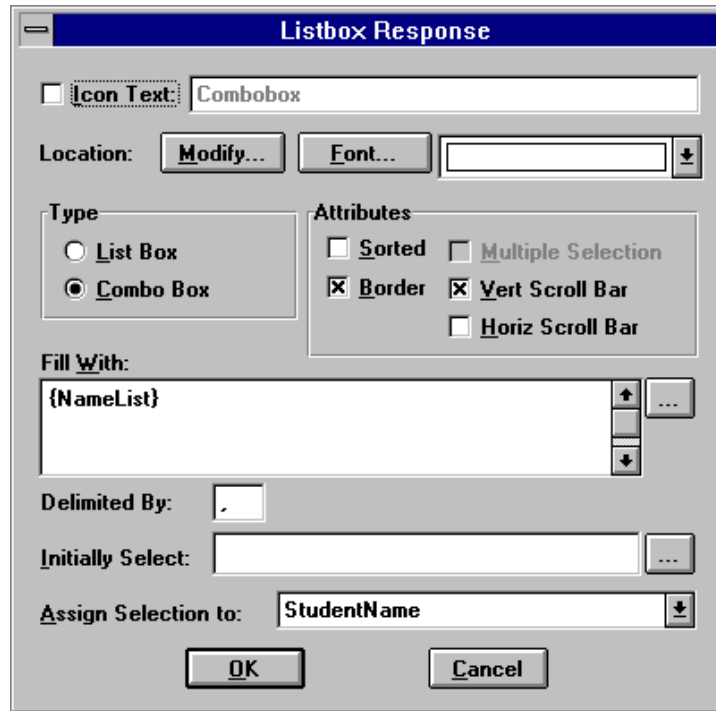
The *LineCount* variable is also incremented by this action, so the next time around the loop the next line is processed.

The response to this is simply a *Branch* response which loops back to the previous task to read the next line of data. The loop is broken when there is no more data and the *Conditional Branch* condition is met.



### Step 5

All the data from the file has been processed and a list of student names has been created. These names can now be put into a *Listbox* response in a new task:



In the *FillWith* box, the *NameList* variable is inserted using the browse button. Type in a comma “,” in the *Delimited By* box. The list box will now contain a list of the names which have been separated by commas in the *NameList* variable created in the loop described above.

Chose the built-in *StudentName* variable to store the student’s selection in the *Assign Selection To* box.

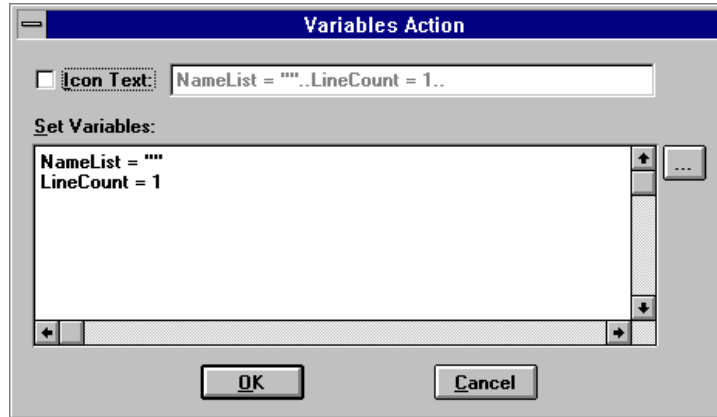
**Note** : If you simply want to read in a list of names from a file without storing and processing IDs, etc., then simply create a name file with the names on different lines in the file, and read it into a variable using the *Read From File* action. Use this in the *Listbox* response directly without doing any *GetElement* processing. The *Listbox* will simply scan the variable for items separated by Carriage Returns.

Add a push button to the list of responses.

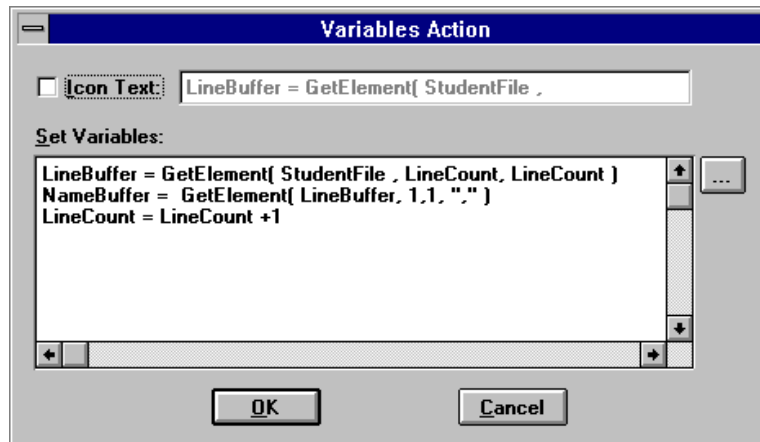
*Step 6*

Once the student has selected a name from the list and pressed the Push Button, Dazzler can now find the corresponding ID using *GetElement* and task loops.

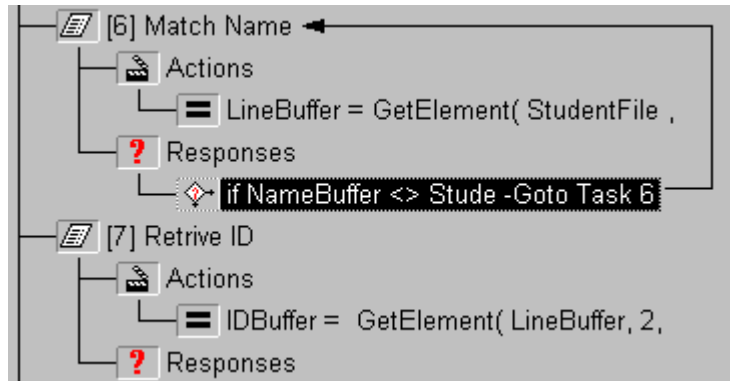
Add another task and initialise the temporary variables:



In another task below this, extract the names one by one:



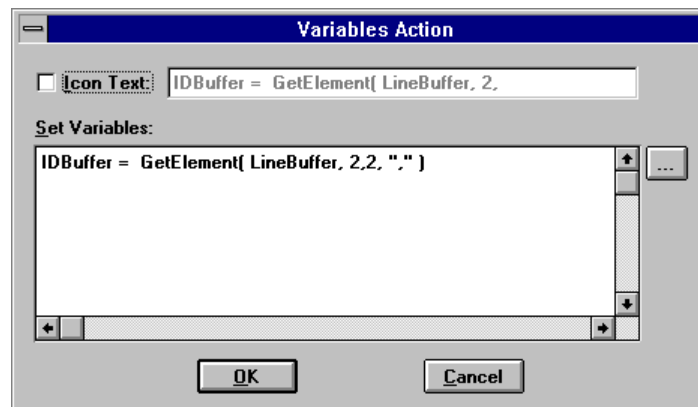
Use a Conditional Branch response to check if the current name extracted from the file is the same as the one chosen from the Listbox by the user.



If it doesn't match, loop back and process the next line.

### Step 7

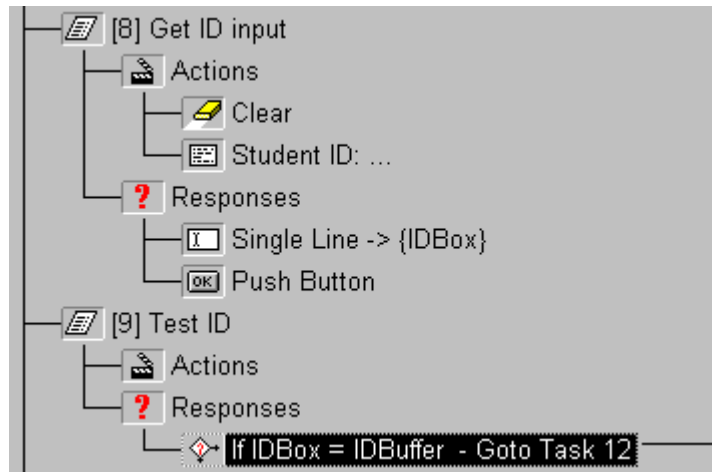
If it does match, move on to a new task and extract the Student ID which is the second part of the line.



*GetElement* here extracts the second element it finds in LineBuffer using commas as the delimiter.

*Step 8*

Dazzler has now found the ID to go with the name selected from the Listbox of student names. You can now ask the student to enter an ID which is stored in a variable *IDBox*. A check can then be done to see if it matches the one stored in the student data file.





---

## Playing sound files with graphics effects

### *Problem*

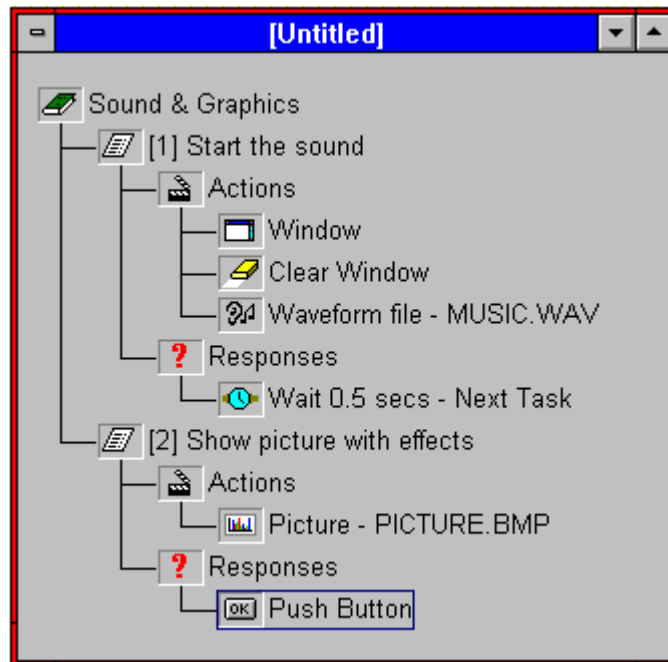
Some users have experienced difficulties playing sound files while showing pictures which have intensive graphics effects such as 'Fade'. The sound doesn't play until the picture effect has finished, even if the sound action comes first in the task.

### *Explanation*

Windows itself handles the priority of different multi-media tasks such as displaying pictures and playing sound files. Intensive processes such as drawing complex effects on the screen are given a bigger chunk of processing time than playing sound files. Thus, if Dazzler asks for the two things to be done together from the same task, the graphics effect will happen first with the sound being played when it has finished its processing.

### *Solution*

Add an extra task before the one with the graphics action to start the sound action first. This gives a higher priority to the playing of sound files.



Also, increasing the effect step size and increasing the time between the steps will free up more processing time for other actions.

---

## Video tape and disk (VIDTAPE.DZL)

This tutorial is designed to show how video material can be incorporated through the use of video tape and video disk material. Additional hardware is necessary to achieve this:

*Computer controllable video source* - this could be a video disk player or VCR. The Panasonic 7350 VCR with AGIA232TC interface is supported directly by Dazzler and its features will be referred to in this tutorial. Other devices may be used in conjunction with a suitable MCI driver - contact the manufacturer to find out if one is available for your equipment.

*Video overlay card* to display the computer picture on the computer screen. Popular cards include Fast's ScreenMachine, VideoLogic's DVA 4000 and CreativeLabs VideoBlaster.

Once you have the hardware installed and connected, make sure that Dazzler has the correct devices selected in the *Settings* menu.

### **Panasonic 7350 only**


Timecoding a tape:

The AGIA232TC add-on board available for the 7350 contains both the RS-232 control electronics and a timecode generator and reader. For each frame on the video a unique number is marked on the second (right-hand) audio channel. (Note - any audio on this channel is erased in the timecoding process). This time-code can be used by the VCR to pinpoint an exact point on a tape each time Dazzler asks it to go to a particular section of the tape. The VCR will also let Dazzler know the current tape position as it plays. In this way you can incorporate video sequences within Dazzler courses just as you have done with text and graphics in the earlier tutorials.

The first step in timecoding a tape is to make sure that the tape is not write protected - if the small tab along the spine of the video has been broken off, you will need to put a piece a sticky tape across it. Also make sure that the 7350 is set for timecoding on CH2 (using the on-screen setup - see the Panasonic Instruction Manual for this).

Start a new course by pressing the New button in the Tool Bar or the File/new menu option.

To view the tape as it is timecoded, select *View/Show Presentation Window* from the menu item (or press the *F2* key) and resize the window to about two thirds of the screen. Display the tape controls from the *View/Tape Video Controls* menu item (or press the *F4*


key). Press the timecode button  on the control panel. If you want to put an ID onto the tape, so that a course can differentiate between this tape and another, and warn the student that they are using the wrong one, type a number into the ID box. You only need to worry about the *Frame Offset* box if the course is to be used with multiple copies of the same tape, each of which may have a slightly different starting point, which the course needs to take into account.



Select *OK*. The 7350 should now rewind to the start of the tape and set itself to *Audio Dub*. A squeal on the audio channels as the timecode is written to the tape may be heard. The tape plays at normal speed as it timecodes, so it will take as long as the video material on the tape to timecode. Allow the timecoding to continue a few minutes beyond the end of the video itself. This will help the 7350 to find the correct point on the tape should it wind too far forward.

Once the timecode is written you can begin to identify the points of interest on the tape for the course you are building. Rewind the tape to the start using the Tape/Video Control Panel. Play the tape and watch it in the Presentation Window. If no video appears, check the *Settings/Video Overlay* set-up and make sure that the *Transparent Colour* is set to the default black. The timecode reference of the video being shown will be shown in the *Frames/Time* box on the Tape/Video Control panel. Make a note of the time code of the points of interest, for instance the start of a particular video sequence and its end, or points on the tape where you want something else to happen in the course. These numbers will be used in the actions and responses of the course you are about to assemble.

Hide the Presentation Window and the Tape/Video Controls by double-clicking on their close boxes ( top left hand corner).

Start the course by setting up the Presentation Window with the Window Action and add a *Clear Window* action after it.

Next add a rectangle for the video to appear in by adding a Graphics Shape action  and defining a solid filled black rectangle in the middle of the screen.

Add a Tape Control action  to send the tape to the start of the first video sequence. Select *Goto and Play* from the list of options in the set-up box. Enter the location either in frames or in seconds. To make the video actually appear in the Presentation Window when the right point on the tape is reached, add a Live Video action  next and set the *Video* and *Audio Output* levels to 100%. Define a rectangle the same size as the Graphics Shape rectangle defined previously.

Finally, add a *Frame Location* response to wait for the end of the first video sequence. Enter the location you noted while running through the tape frame or seconds.

Add a new task to stop the tape after the first task has finished by adding a *Tape Control* action and selecting *Pause* from its set-up.

When run, the course should wind itself to the start of the first section and play in a rectangle on the screen. When the end of the sequence is reached, the tape should pause.

