7

Text Handling

This chapter is about DTP's way with words – and with letters and characters generally. We shall see how even simple text can be made to look special if you give it a touch of style: changes of fonts, sizes and layouts are all valuable resources at your disposal. There are also typographical niceties that separate the amateurish approach from the professional finish, and special effects with text are not overlooked.

Text Alone

Text Entry

The simplest use of DTP software is as a word processor, though that seems rather a waste of all the other facilities. Nevertheless it is feasible provided that you set the text size large enough for it to be easily legible when the line occupies the screen width; if text is too small you get eyestrain, and simply enlarging the view results in the annoying effect of only part of a line being on screen at any one time so that the area visible keeps flicking back and forth as you type. There is nothing to stop you from making the type size small again just before printing, once you have checked your typing on screen. This gives you the advantages of a word processor without the disadvantages of an unreadable screen display.

Not all DTP packages are equally suitable for use as a word processor, though; in particular, *Acorn DTP* doesn't add pages automatically as you type and is really designed for the addition of text confined to one frame only. In this case you are much better off preparing your text beforehand on a word processor and loading it into DTP. In fact this is a very common practice for a variety of reasons: you may be responsible for putting together a publication with several authors who send you their texts on disk, or you may simply be used to the way your favourite word

processor works, or the text editing facilities on your DTP software may not do what you want. Whatever the reason, the process of *importing* text has many advantages.

Text Importing

All DTP packages have the ability to assimilate text from another source; some cast their net wider than others and some can do more with what they are given. However, the principle is simple enough even if the precise details vary: just drag a file icon onto a document and the text will be loaded, in much the same way as described in Chapter 4 for using *Edit* with *Draw*.

Text Filters. One of the most helpful jobs that DTP software can do is accepting text from a wide variety of word processors, preferably in such a way as to retain as much as possible of any information about text styles or layout. *Acorn DTP* has a built-in ability to translate *1st Word Plus* files so that text styles such as bold and italic are preserved. Some of the other DTP packages have additional modules that you load as you need them: they act mainly as filters to remove word processing codes that might otherwise cause confusion or, worse, that might force you to search through the entire text and weed them out yourself. The most common cause of frustration with imported text is the unwanted Return characters that are added at the end of every line by some word processors. Always having to do a search and replace for them on long documents is tedious, so it is fortunate that *Impression*, for example, supplies a module to do this for you automatically during the importation process.

The advantage of using modules in this way is that after one has been loaded it can be ignored because it works invisibly, and if you want a different type of filtering action you merely double-click on the relevant file icon. In this way the application program can be kept more general (and therefore shorter) while leaving you free to add only the functions that you need at any one time. It also means that software houses can supply further modules as time goes by to broaden the range of word processors supported. This apparently simple system is therefore really very powerful.

Text preparation. The fact that *Acorn DTP* isn't really intended for use as a word processor for entering text makes a word processor an essential adjunct; importing text is thus the norm with this software. With the others you have the choice of whether or not to use a separate word processor. Here is some general advice on preparing text for subsequent import: heeding it should minimise any difficulties in the absence of any text filters specific to the word processor in use, although some word processor–DTP combinations are more compatible than others.

- Don't use any of the word processor's own formatting or typographic

codes. In particular, don't apply justification or hyphenation if your word processor provides them.

- Don't type more than one space after punctuation such as colons or full stops. Although this is widely considered to be good secretarial practice it can cause formatting problems in justified text once it has been imported into DTP. Multiple spaces should in general be avoided.

- Unless you are using *Acorn DTP*, use a single Return to signify the end of a paragraph, as the other DTP software doesn't convert double Returns to a single one without help from a filter module. As is generally advised with word processors, don't type Return at the end of every line.

- Tabs may or may not be preserved, but even if they are you will need to do some work on the positioning of tabulated matter after importation. Tabbing in *Edit*, and other word processors that insert spaces instead of tab characters, should be avoided.

The benefit of multitasking under RISC OS is that you can (if you have enough computer memory at your disposal) test which of the above points can be ignored for your word processor–DTP pair by running them both simultaneously and 'saving' text from one to the other directly, without going via disk. In this way you can quickly establish the best conditions for working with your particular system, and you can also build up a set of instructions for use by anyone else who might be preparing text for you to assemble by DTP.

Text Flow

Text Overflow. One of the most important things about importing text as a story into DTP is to know whereabouts it will go when the process is complete. It will of course be poured into a text frame nominated by you, but there may be more text than will fit into that frame. The outcome of such *text overflow* depends on what sort of frame it is: with reference to the frame types described in Chapter 6, if it is a *master* or *principal* frame, as many new frames will be created automatically (except in *Acorn DTP*) as are necessary to contain the whole story. If it is a *local* frame you will be warned in some way that the overflow has occurred: *Ovation* and *Impression* place a downward-pointing arrow at the bottom right of the frame, whereas *Acorn DTP* shows the bottom of the frame as a broken line. It is all too easy to overlook this telltale sign if you are in a hurry or if the broken frame line is obscured by an applied border, so try to develop the habit of checking the bottom of the last frame whenever you are importing text into a document. Even the professionals can forget this sometimes, and losing the end of a story is risking the scorn of your readers.

Linking Frames. To cope with text overflow, every DTP package has a way to link together more than one frame so that the story flows smoothly and continuously between them. Mostly this is done by subtle clicks of the correct mouse buttons, presumably to avoid links being created by accident, although *Ovation* has a special linking tool that must be selected first. You don't have to wait until you have achieved text overflow with an imported story before making the links: they may already exist in cases where there is a fairly rigid page design, such as a news sheet, and linking text frames on the master page is one way of producing multiple columns.

Remember that the order in which you link frames is important. That may sound obvious as you read this, but linking frames in the wrong order is easier than you might think, and will cause some momentary head-scratching. *Impression* and *Ovation* have mechanisms for showing the direction of text flow (Figure 7.1 shows *Ovation*'s text flow arrows), which is useful for ensuring that you have the order right.

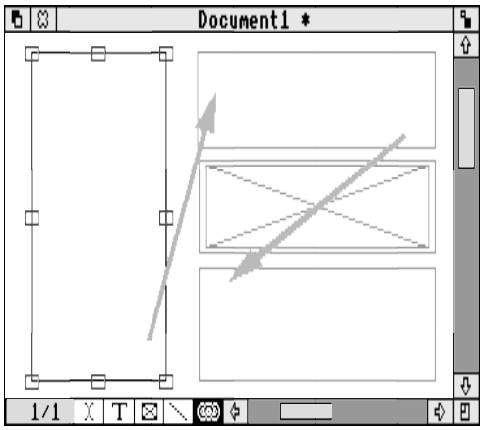


Figure 7.1 *A set of linked text frames in Ovation, with the link tool selected to show text flow between two columns and across a graphic frame*

Text Manipulation

Editing. As already mentioned, all the Archimedes DTP software has, to a greater or lesser degree, word processing facilities so that you can edit an imported story. This may be necessary because you have at the last minute spotted some grievous spelling error, or you may have to cut the story down to fit a limited space (this is one of the jobs of an editor, and although vital it is a way of losing friends if not done with care and tact!).

Apart from the use of the Delete and Copy keys for single-letter deletions backwards and forwards respectively, text, like frames, can be cut or copied to the clipboard and then pasted back elsewhere, in either the same document or a different one (if the software allows more than one document in memory at once). Just as with frames, though, the DTP program has to be told what text you intend to cut or copy, so you must select or *mark* the necessary letters, words or sentences.

There are several ways of marking text, of which two are common to all the DTP packages on the Archimedes: the *drag* and the *Select–Adjust* methods. The first simply involves clicking and holding Select while the mouse pointer is at one end of the text to be marked, then dragging until the passage has been selected as indicated by its reversed or *highlighted* appearance. In the second routine you click Select to move the text cursor to one end of the required passage, then move the mouse pointer and click Adjust at the other end. Figure 7.2 shows a highlighted region of text; the techniques and the appearance will both be familiar to you if you have used *Edit* to any extent.

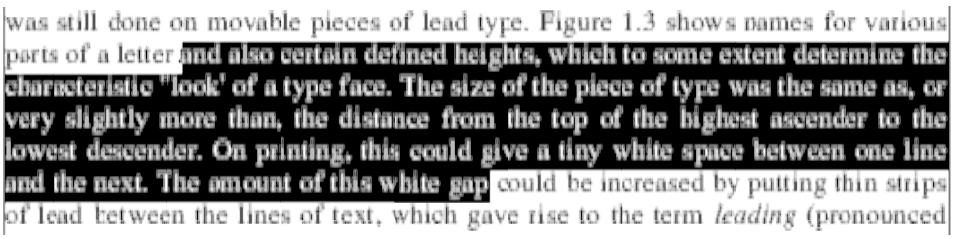


Figure 7.2 *A selected (or highlighted, or marked) region of text*

Now for a couple of refinements: with either method you can move the end of the marked section by clicking or dragging with Adjust. Secondly, you don't need to drag forwards in the text (or click Select at an earlier point than Adjust); you can also drag backwards. In both cases it is the place where you stopped dragging (or clicked Adjust) which can be moved by clicking or dragging with Adjust.

Dragging is more likely to be useful if the passage is all visible on screen, whereas Select–Adjust comes into its own if you have to scroll around the document between marking the start and the finish points.

You may think that the foregoing has made a bit of a meal of such a basic thing as selecting a piece of text: with these methods it is in fact very much easier to do than to describe. The different DTP programs available also offer their own individual ways of marking commonly needed regions with different numbers of button clicks; they all use a double click to select the current word, but beyond that the details vary. In addition, any number of key shortcuts exist to make your text editing task easier –

too many (and too inconsistent between packages) to list or even hint at here; familiarity with your chosen software is the only answer to this (or you can use the menu options, which is a slower way but more certain). Remember, too, that there are quick ways to move around your document which can save a lot of time if it is several pages or chapters long: the Home, Page up, Page down and cursor keys are worth investigating for a start, with or without Ctrl and/or Shift.

Cutting a marked passage to the clipboard can always be done with the Delete key, but *Impression* and *Ovation* have an extra feature: if you press any letter or number key while there is a marked section, that section will be cut to the clipboard and replaced with what you have just typed. This is really useful, for instance in conjunction with double-clicking to select a word, for making very rapid textual substitutions. (Note that when you select a word the software may also select the space before or after it, so be sure to type the space again if necessary.) Yes, it is very unnerving when it happens by accident and an unobserved marked section vanishes silently and unexpectedly, but don't panic: the disappeared passage will stay on the clipboard until another cut is made, so you can paste it back where it came from.

Most of the other aspects of text editing I shall leave for you to discover in your own time with your own software, as each has its own peculiarities.

Margins, Indents and Tabs. As with any word processor, DTP must give you control over the positioning of text. The horizontal part of this is governed by margins (the spaces between the text and the left and right edges of its frame), indents (traditionally, extra space between the usual margin and text; in DTP it is normally the space between the left edge of the frame and the first line of a paragraph, but see below) and tabs (column positions or *stops* that come into action when you press the Tab key: each time you press Tab the cursor moves one tab stop along). In traditional typesetting the distance between the left and right margins, in other words the width of the text, was (and still is) called the *measure*.

The relationship between margins and indents is slightly complex, and terminology varies slightly from one piece of software to another. How you view margins and indents at any one time can depend on the text layout you are trying to achieve. For *flush text*, where all the lines are the same length, things are simple: there are no indents. For a paragraph with an indented first line the picture is as explained in the previous paragraph. The reverse situation, however, is where the first line of a paragraph sticks out to the left of the others: this is called a *hanging indent* (or, more colloquially, an outdent). The traditional view of a hanging indent would be that the longest line represented the text measure and the shorter lines were indented; in DTP terms it makes more sense, not least because it is the way you would do it in practice, to regard the left edge of second and subsequent lines in a paragraph as the left margin, with the first line having a smaller margin, putting it further to the left.

Combining a hanging indent and a tab stop at the same place as the left margin is a good way of giving a neat finish to a list containing several items, be they numbered or 'bulleted' in some way. Figure 7.3 gives two examples of such lists.

1 This is the first item in a short but fascinating numbered list.
2 This is the second item; the first line of each item starts at the 'first line margin' position.
3 Straight after the number is a Tab character, which moves the cursor to the first tab stop.

- This is the first item in a brief 'bulleted' list; we shall see later how bullets come to be.
- The first tab stop is positioned at the same place as the left margin so that any text following the first line aligns itself neatly.
- This is the last item in the list.

Figure 7.3 *Numbered and bulleted lists arranged tidily by using margins and tabs*

Four types of tab stop are available in DTP: left, right, centre and decimal, as demonstrated in Figure 7.4. Left and right tabs, as their names imply, align with the left and right edges of text. Typing against a left tab is like typing against the left margin, whereas text typed at a right tab moves the opposite way from usual, pushing previously typed letters leftwards. A centre tab shares any text equally either side of the tab position, and a decimal tab works like a right tab until a full stop is typed, when it starts acting as a left tab; the effect of this is to line up all decimal points for presentation of numerical tables.

left:	*centred:*	*right:*	*decimal:*
the text on	the text on	the text on	123.45
this tab stop	this tab stop	this tab stop	1.2345
is ranged	is arranged	is ranged	123456
to the left	centrally	to the right	.123

Figure 7.4 *The four types of tab; the position of each tab is shown by a vertical line*

Margins, indents and tab stops can be set up in two ways: some programs allow both, others only one. The easiest and most intuitive method uses a graphical representation of a ruler along the top of the frame, with different symbols standing for margins, indents and tabs that you can slide along to where you want them to be. The other way is more exact but a little harder to get right first time: you have a table of values that you fill in according to the required positions and types of item. The programs (*Impression* and *Ovation*) that provide both facilities give you the best of both worlds, because you can position your stops visually on the ruler to get them more or less right, then use the value tables to put them exactly where they should

be. Both methods, from *Impression* version 2.05, are illustrated in Figure 7.5, which is the ruler used for the columns in Figure 7.4. The different symbols used for margins and tabs are shown in a box at the top of the ruler, and their positions along the ruler scale are given in numbers in the 'Ruler setup' dialogue box underneath.

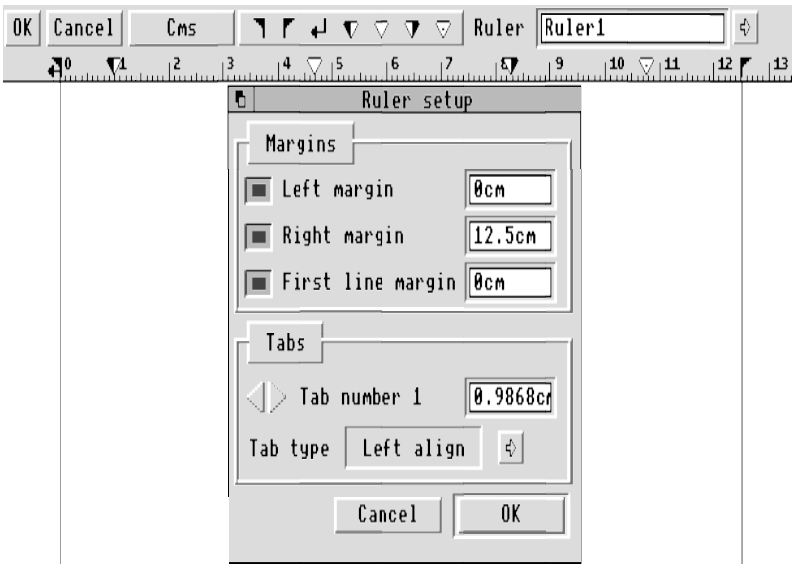


Figure 7.5 *Margins and tabs in Impression: two approaches to the same specification*

Vertical Spacing. Control over the vertical positioning of text in DTP takes two forms: the line spacing applied equally to a group of lines (which may of course be the whole text) and extra spacing applied between paragraphs. The line spacing can be viewed as a norm from which you would seldom, if ever, deviate for any one type of material, whereas paragraph spacing automatically gives you consistency in the white space between paragraphs and above and below headings. Note the phrase 'above and below': the space above a paragraph can be set to a different value from that below. The two are independent.

In typing or word processing you might insert two or three extra carriage returns above a heading and one below it, so that there is a clear break in the text; for the same reason it is common practice to put two Returns to mark the end of each paragraph. In DTP this is not only unnecessary but is actually counterproductive, because using full line spaces in this way prevents the kind of fine control over spacing that DTP offers, and having to remove them is a nuisance. The spacing of a heading is part of the attributes you define for it, in the same way as changes in font or size. We shall see later, when considering the appearance of text, how to set up

and apply the spacing controls.

Checking Spelling. These days any word processor worthy of the name must be able to check your spelling and query any words with which it is unfamiliar. As DTP has begun to assume the mantle of word processing it has had to follow suit: of the Archimedes DTP packages only *Acorn DTP* doesn't provide a spelling checker. The principle is fairly simple in that a word is compared with a dictionary, and if a match is not found an error is generated; depending on the conditions you were operating the checker under, the program may offer you a list of guesses at what you might have meant. The suggestions are usually pretty good for simple typing errors, and choosing the right word usually involves no more than two mouse clicks, one on the word and one on a 'Replace' or 'Insert' button.

Alternatively the word might have been correct as typed and you would like the software to record it as a valid one. In that case you can add it to the vocabulary that will be recognised and then save it to disk so that it will be available in all future work sessions, which over time will build into a personalised set of words that are important to you. A third possibility is that you might not wish to add the word to the list, but because it occurs several times you don't want it to be queried later; an 'Ignore' or 'Continue' button is provided for this purpose. *Impression* lets you save an Ignore dictionary so that you needn't be troubled by proper names again.

You have several checking options: you can check the whole of a story once it has been typed in or imported, you can see if the word at the cursor is correctly spelt or you can have words checked as you type. For incompetent typists or abysmal spellers the last option may be too distracting, causing continual stoppages.

It is as well to recognise the limitations of a spelling checker. Like all computing devices it is only as good as the information it is given, so a complete vocabulary is important; bear in mind, though, that the longer your dictionary the more memory it takes, reducing the room for documents. This is not too great a problem on the Archimedes unless you are profligate with words, when you will find that too big a dictionary also slows down the checking process. Another drawback is that a word may be correctly spelt but be the wrong one entirely, as in 'He was to late too help'. A spelling checker can't pick up this sort of contextual error; some additional human intervention is always a good idea.

Other Dictionaries. There are two other refinements on the dictionary theme. The first, an abbreviation expansion dictionary, is unique to *Impression* and is a real boon to lazy or consistently inaccurate typists (I fall into both categories). It works like this: for common words or phrases you prepare a list of abbreviations and their complete counterparts, such as 'ar' for 'Archimedes', and provided that you have selected the appropriate option, 'Expand as you type', every time you type the

abbreviation followed by a space or a Return it will be replaced by its expansion. It is also useful for ensuring consistency (I always spell 'disc' thus; the publisher's house style for this book is 'disk', so my dictionary translates one to the other) and for correcting common typing errors.

The last dictionary type, the hyphenation exception dictionary, is worth mentioning here even though an explanation of its context must wait until later. An exception dictionary is used by those programs that perform automatic hyphenation, and is a list of words that don't conform to the rules used to decide where words should be broken. You can build up your own exception dictionary if you feel strongly about the mistakes your DTP software makes.

Mail Merge. One of the most widely used features of word processors in the business world is the ability to take a form letter and print it dozens or hundreds of times so that each copy is addressed 'personally' to someone different. You must at some time have received special offers or prize draw letters with your name and address inserted in the appropriate places. Originally the form letter was printed with gaps and the personalisation details were added (unconvincingly) afterwards, but the power and speed of modern computers is such that a letter and a file of names can be combined (*merged*) so that each letter is ready for printing separately. I suppose we are still not fooled into thinking that the letter is genuinely personal, but it looks more realistic.

On the Archimedes a form of mail merge, though something of a fiddle, used to be possible only with *Impression*; later versions (after 2.10) can be used with a mail-merge utility called *Importer*. Unfortunately I haven't so far been able to get mail merging to work in any practical way with *Acorn DTP* or version 1.22 of *Ovation*, so there is little point in going into detail here.

Text and its Appearance

Styles and Effects

Up to now we have been considering text as merely the words themselves, with DTP acting more or less as a word processor. We now come to the parting of the textual ways between word processors and DTP: giving plain words a special appearance that gets close to, and in some senses even a little beyond, 'real' typesetting. This versatility takes many forms, from different fonts and sizes to the use of special effects.

Paragraphs and *ad-hoc*ery. You should now be familiar with the concept of a paragraph as a basic unit, and we have looked at marked text in word processing

terms. In a similar sort of way there is a distinction in DTP between large-scale adjustments to the appearance of paragraphs or other major units of text, and purely local fine-tuning of words or small sections. The fact that the philosophies of the various packages are rather different makes generalisation a slightly thorny matter, but the underlying concepts are valid for all, regardless of the details of how they are achieved.

It would be as well to try to clear up a point of possible confusion of terminology, as usage is so variable from one piece of software to another. In the DTP context the word 'style' has several different connotations: it can mean as little as the font and size of text; or it may be used in a specific sense to denote something that isn't the font name or size or the way the text is laid out but, for example, the weight or italicisation or whether it is expanded, condensed or underlined; or it can be the actual means of applying a defined set of typographical features to a selected piece of text or a paragraph. In the rest of this section I shall use the the words *style* and *effect* with the following meanings (unless otherwise indicated).

- Style: a named specification (font, size, underlining or whatever, or any combination of these) given to anything from a single letter, through a whole paragraph or over a complete document; editing the definition of a style will alter the specifications of all text containing that style.

- Effect: a single part of a specification (font, size, etc.) applied to a marked portion of text; it can only be altered by reselecting that text. Effects are useful when you don't want or need to go to the trouble of setting up a style for an occasional task: hence my use of the term *ad-hoc*ery to describe such improvisation.

There is also, unfortunately, a wide variation in the scope of application of styles: in some software a style has to be applied over the whole of a paragraph, whereas other programs allow any marked text to be given one or more styles. Effects, on the other hand, are applied to a marked region, no matter what software you use.

It may be that in trying to shed light on this issue I have succeeded only in clouding it, but all should become clear as we proceed.

Effects. Let us start with these, as they are simpler and generally used one by one. The effects provided by all DTP packages are as follows:

- text font;
- text size;
- line spacing (not necessarily independently of text size);
- variant within font (bold, italic);

- subscript or superscript;
- underline; *and*
- text colour (even if only white).

In addition, the more recent software offers other effects, such as:

- condensed text (like this);
- expanded text (like this);
- text alignment or justification (which we shall consider later); *and*
- margins, indents and tabs (which we have already encountered).

The main benefits of effects are their speed and simplicity of application: they are intended to be used and then forgotten about. No preparation is required; you just choose what you need from the list provided and carry on.

Applying an effect can be done either while you are typing or afterwards. Suppose you are entering text at the keyboard and want to change to italics for a few words: just choose the appropriate effect (on the browser in *Acorn DTP*, in the Style submenu or using Shift-Ctrl-I in *Ovation*, under 'Text font' in the Effect submenu in *Impression*) and continue typing, changing back when you have finished with it.

Alternatively, if you have imported some text and want to underline occasional words, you first have to mark the required text passages one at a time and then apply the underline effect. However, underlining is a hangover from the typewriter era: as a means of emphasis it is becoming defunct now that DTP gives access to 'traditional' typesetting conventions such as italic or bold. So you may decide later that you want to change all your underlines to bold type. That is no real problem: you just have to reselect every passage, removing the underline effect and adding a bold one each time. Well, this isn't hard but it could get tedious. And that is one place where styles come in useful.

Another drawback with effects arises if you want to change more than one typographical feature of the text at once, say from 12pt Trinity Medium to 16pt Homerton Bold. You may have to go through more than one menu and make individual choices, one for size, one for font and one for weight. After each choice the intermediate result has to be calculated by the font manager and plotted to the screen: you may have to pass through 16pt Trinity Medium and 16pt Homerton Medium before you reach your goal. Not only does this take time, but having to store all these successive but unwanted fonts wastes precious memory in the font cache. A way round this is to adjust the window size until the marked section is no longer visible and then make the changes: as the intermediate stages are thus never shown on screen the unnecessary calculations are avoided.

Something else to bear in mind when using effects to change to italic or bold type in the middle of a paragraph is that different programs have different ways of identifying the fonts within a family. *Acorn DTP* and *Ovation* let you specify a base font such as Trinity or Homerton, and it will appear as roman type. To change to italics you just choose Italics from the appropriate menu; if a font of the same name plus '.Italic' or '.Oblique' is available it will be provided. You don't have to give the full font name because the program intelligently assumes you want the italic of the font that is already there. With *Impression*, on the other hand, you choose from the complete list of fonts, so you would have to specify Trinity.Medium.Italic or Homerton.Medium.Oblique explicitly.

So far, so good, but suppose you now change your mind and decide to put the whole paragraph into bold type. *Acorn DTP* and *Ovation* add the attributes together, so that the roman type becomes bold roman and the italic becomes (correctly) bold italic: the two elements, bold and italic, are treated independently. If you try this with *Impression*, however, the whole paragraph becomes bold roman as the specification for the italic font is overwritten by the new font name, Trinity.Bold or Homerton.Bold. This is not intended as a criticism, as each philosophy has its merits and its problems.

So, although effects are in general a quick and easy way of changing the appearance of text, they are not the whole story: something more powerful is needed.

Styles. At the centre of all DTP work is the notion of an underlying style of text, which is essentially its default appearance when you open a fresh document and start typing. It is normally the one used most in a document, so in something with longish stories, such as a book or magazine, it would be the *body text*, the main reading matter. You may find it called Body text (*Acorn DTP*), BodyText (*Ovation*) or BaseStyle (*Impression*). Other styles can be defined that use this as a basis, merely changing perhaps the size for use as a heading, or they can redefine everything for a complete contrast. A style is in essence a group of effects that has been given a readily identifiable name, but it is much more besides.

Once styles have been set up you can apply them anywhere in the document and achieve uniformity with ease. Note that *Acorn DTP*'s styles can be applied **only** in paragraph mode, and similarly *Ovation*'s paragraph styles act on whole paragraphs. If your main headings are in Homerton Bold at 18pt with a 6pt space below, your subheadings in Homerton Medium at 16pt with 4pt space below, and your body text is in Trinity Medium at 12pt on 14pt with 4pt space between paragraphs, by applying the necessary styles to the headings all the definitions are taken care of in the one step rather than having to apply several different effects one by one. Styles not only save you time and make sure that similar things are treated consistently, but they have the added advantage that they act in the same way over the whole document. If

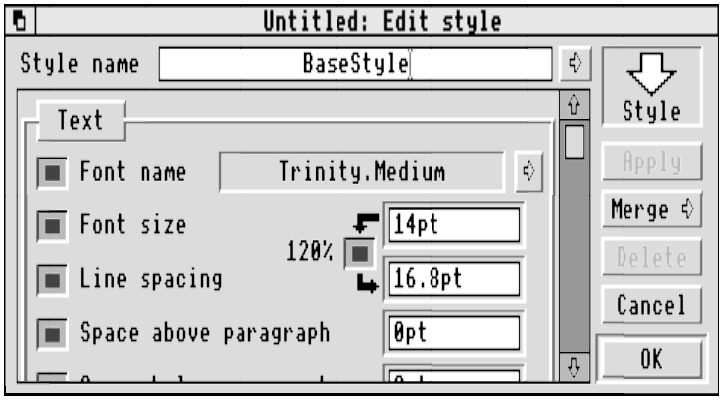
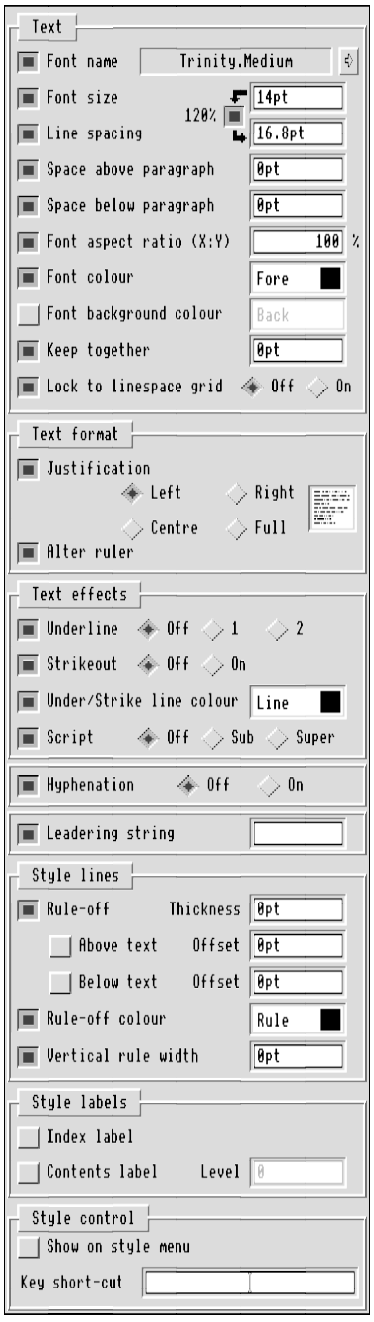


Figure 7.6 *The 'Edit style' dialogue box from Impression; note the inset window and scroll bar*

you define all your headings with effects rather than styles, what happens if you change your mind about the font at a late stage? You have to go through them all, changing the font for each individually. With styles you just alter the definition, and every occurrence of the style will be updated automatically.

The only real problem with styles is that they require some forethought and patience in their definition. To give you an idea of just how much you can do with styles, and therefore how much you may have to consider when setting them up, Figures 6.6 and 6.7 show the 'Edit style' dialogue box from *Impression*; other programs offer most of these functions in different ways, and more besides, usually over several dialogue boxes. So not only is there a wealth of potential for control over your text's appearance, there is also great scope for disaster unless you know what you are doing.

In Figure 7.7 we see a definition of BaseStyle as it is set up by default on my Archimedes. Wide-awake readers will spot that this statement implies two things. First, in the definition of BaseStyle almost every entry has a value or a selected option: this is because BaseStyle underlies all other applied styles and all those entries **must** have some sort of decision made about them – for instance hyphenation must be switched on or off. The second point is that by altering the definition of BaseStyle in a document the whole basic character of the document is changed; it is possible to force the program to start up with a new default BaseStyle by saving a suitably amended blank document under the name !Default in a special place – in the directory !Impress.Auto. The other DTP programs have similar ways of letting you start up with a different default text style from that supplied originally, although what you save may be a *stylesheet* (see later) rather than an empty document.



It is worth taking a quick look at some of the options in Figure 7.7 as this is the best place to explain them. In the box headed Text you will see a number of familiar items. The button marked '120%' is used to set the line spacing to the DTP standard of 20% more than the font size; unless you turn the button off, any alterations to the line spacing will be ignored. We have also already met the 'Space above paragraph' and 'Space below paragraph' options, but unexpected things can happen even with this apparently simple scheme. Just remember that if your software allows you to apply a style to a paragraph by selecting it (as opposed to only letting you apply a style in paragraph mode, for example as with *Acorn DTP*) the Return character that follows it needs to have the style applied to it as well if the value shown for the space below it is to appear correctly.

The term 'Font aspect ratio' may be new to you. All it signifies is the degree of stretching or squashing that is applied to the letters as they are printed. This is a technique that is very much a part of the DTP experience, as it was completely unthinkable for those using metal type! An aspect ratio of 100% means that the scale in the horizontal (X) direction is the same as that in the vertical (Y) direction; if the ratio is 50% the letters are only half as wide as they are high (X:Y = 0.5), and if it is 200% the width is twice the scale of the height (X:Y = 2). Below are some samples showing the effect of different aspect ratios:

ratio 60%

ratio 100%

ratio 200%

Figure 7.7 *The inset window from Figure 7.6 'unrolled' to show all the options*

I have used Homerton, a sans serif font, to emphasise that the relative widths of horizontal and vertical lines are affected by these transformations. This makes them appear different from 'true' condensed and expanded fonts, which are designed so that the weights of horizontals and verticals are comparable. *Ovation* also offers you a 'Small capitals' style (it looks a bit LIKE THIS), which for the same reason doesn't quite look right because the line thickness of the letters is reduced in comparison with the normal size.

The aspect ratio is set in some software by giving the font a different width (in points) from its height; the effect is the same, though a little harder to calculate consistently for different sizes.

The use of font colour is fairly easily imagined, and if available is defined by a colour picker (see Figure 6.6) or similar device. The font background colour, which is in any event an option only in *Impression*, can be ignored for normal purposes; you would only need to worry about it if you changed the background colour of a text frame (if you didn't alter the font background colour to match you will find odd edge effects around the letters). In contrast, *Ovation* seems to set the font background colour to the frame background colour automatically, which is far simpler, even if not quite as flexible.

Text Formats. An understanding of these is fundamental to good DTP practice, and this is as good a place to deal with them as any. As you can see from the contents of the box headed 'Text formats' there are four ways in which text can be set out between the left and right margins (or left and right indents, if your software uses that terminology; I shall continue to use the word 'margins'). Three of the names should be strangely familiar unless you skipped the illustration of tab stops in Figure 7.4. The Left, Right and Centre entries refer to *left aligned, right aligned* and *centred* text, and typing text in these formats is much the same as in their tab counterparts as long as you substitute 'margin' for 'tab stop'. As in Figure 7.4, one or both edges of the text end up uneven because the lines tend to be of different lengths; left aligned setting is therefore sometimes described as *ragged right* and right aligned as *ragged left*. Some other pseudonyms, which are self-explanatory, are shown in Figure 7.8.

The last option, Full, is used to choose *fully justified* text. In contrast with ragged and centred setting, where all the spaces are the same width, the spaces in justified text are expanded to make the widths of the lines constant (in other words the right-hand end of each line is pulled to the right margin, stretching every space in the line). Different lines may need different amounts of adjustment, so the spaces between words will tend to be unequal from line to line. Automatic justification can be a problem, or it can be turned to a design advantage, as we shall see later.

Hyphenation and Justification. It is quite apparent from even a quick look at the justified text sample in Figure 7.8 that full justification can cause problems of

The text in this frame is *left aligned*, also known as *unjustified*, *flush left* or *ragged right* because of the uneven right edge; all the spaces between words are the same width. It gives a 'modern', friendly-to-read feel to body text but short lines can look rough.	The text in this frame is *centred*; as the name implies, it is equally distributed about the centre line between the left and right margins; all the spaces between words are equal. It is useful for largish displayed text but looks 'faddish' used for body text.	The text in this frame is *right aligned*, which is otherwise known as *flush right* or *ragged left* because of the uneven left edge; all the spaces between words are equal. Text set ragged left can be harder to read as the lines start in different places.	The text in this frame is *fully justified*, or simply *justified*; the spaces between words are no longer equal, as any excess on a line is shared between the words. This produces neat edges but sometimes messy middles; it looks quite traditional.

Figure 7.8 *The four types of text alignment or format: each has its own characteristics*

spacing between words. Although the sample was deliberately fabricated to give some unsightly lines, the difficulty is not uncommon. To some extent the ragged settings suffer too, but not as badly because the eye doesn't have to traverse wide spaces in trying to read a line. The trouble is caused by an unfavourable combination of three factors: the text measure, the text size and the lengths of the words you are using. Looked at another way, it is related to the number of letters that will go into each line and the number of spaces there are to go round. If a word fails to fit at the end of a line, the program takes it over to start the next one (this is termed *word wrap*); if that word is particularly long and only just missed fitting, there will be a lot of space to allocate and the line will be *open*. A potential answer lies in trying to split or *hyphenate* the long word at a suitable place.

The close relationship between justified text and the need for hyphenation has been recognised for a very long time: so close is the association, indeed, that the term *hyphenation and justification* is often shortened to the initials *h&j*. Decent DTP programs are able to hyphenate text automatically according to a set of rules (the *algorithm*), though the power and accuracy of the algorithm used can vary from one package to another. Some of the Archimedes software, notably *Acorn DTP*, also gives you control over how narrow the spaces in a line can become before a word has to be carried over, and how wide they are allowed to become before hyphenation is attempted. DTP software normally lets you choose whether automatic hyphenation is switched on or off (*Impression*'s hyphenation buttons are about two-thirds of the way down Figure 7.7). In preparing the text of this book I left hyphenation switched off because of the relatively wide measure and fairly small text.

In practice, automatic hyphenation by algorithm is not the whole answer, because there will always be times when the rules of logic used by the program break down and it makes mistakes. To be sure, hyphenating the English language is a real

minefield (next time you are reading a newspaper or magazine, take a look at the way words are split: you are likely to see even worse examples than *at-heist*, *lig-hthouse*, *mate-rial*, *telep-hone*, *typew-riter* or *contemp-t*, and now you are aware of it you may be consciously irritated instead of subconsciously disorientated every time you come across a bad break!). Over the years traditional printers built up a set of rules for determining where word breaks should be allowed, based on word stems, pronunciation and meaning. In contrast, a DTP program can't learn from its mistakes, but you can: this is where the *exception dictionary* comes in. It is essentially a list of words that don't conform to the algorithm, showing permissible breaks. If a word must not be split automatically, it is entered into the list as the full word; otherwise it is included with one or more hyphens at appropriate places. This will then override the automatic choice of hyphenation point. Interestingly, with *Impression* you can allocate different values of desirability to different breaks, so a word may be split at one point in preference to another.

When all else fails and the line remains too open, you still have some choices. First, you can insert a hyphen manually; however, if you then alter the measure or rewrite part of the paragraph before that point you will end up with an embarrassing hyphen in the middle of a word. Secondly, you can insert an optional (or discretionary, or *soft*) hyphen instead: if it is needed for a word break it will appear just like a normal one, but if it isn't required it will not be printed. Thirdly, if you really can't find a suitable place to break a word, *Ovation* offers a useful facility: altering the space between letters by a process called *tracking*. This has more than one use, but here it can help you by letting you squeeze together the letters in the words on one line if the problem word almost fits, or separate them slightly to reduce the space between the words to less unsightly gaps. Figure 7.9 shows both effects used as a last resort: the first line gets negative tracking applied, and the second has positive tracking.

One point to look out for: in setting where there are only a few characters to a line, hyphenation may become frequent. For aesthetic reasons you should avoid having a run of successive lines ending in hyphens: three would be a normal maximum. In *Impression* you can prevent hyphenation in a specific case without adding a word to the exception dictionary by defining a style (call it NoHyphen) that does nothing but turn off automatic hyphenation, and applying it to the offending word. Alternatively, you can use a soft hyphen to prevent hyphenation: in *Acorn DTP*, place the cursor at the start of the word and press F10, and in *Impression* put the cursor at the end of the

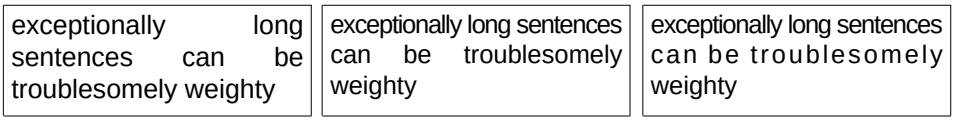


Figure 7.9 *The use of tracking in Ovation: left, the original; centre, with negative tracking applied to the first line; right, with positive tracking in the second line as well*

word and type Ctrl-hyphen. Each of these methods will override the automatic hyphenation if it is turned on.

Other Style Options. Returning briefly to Figure 7.7, there are a few remaining options that are worth looking at here.

- Underline: fairly obvious, but note the option in this instance for two underlines as well as just one.
- Strikeout: this is a bit like a raised underline, and the only place I have ever seen it is in learned journals describing original manuscripts to show where the author crossed something out after a change of mind (it looks like this: ~~STRIKEOUT~~ ~~strikeout~~).
- Script: this is really only shorthand for subscript (small characters below the line) and superscript (above), which are useful for chemical and mathematical formulae like H_2SO_4 or $E = Mc^2$, or for footnote numbers.
- Leadering string: this is one or more characters that are used to fill the gap when the cursor moves to a new tab stop; it is not often used in tables but can be useful in widely-spaced contents lists to help the eye follow from the entry to the page number, as below:

1.What is DTP? 1
From Manuscript to Type 2
The Computing Perspective 2

Compare that with the following, where it may be hard to tell which page number goes with which contents entry (but a page full of dots can look decidedly odd, so there is a trade-off):

1. What is DTP? 1
From Manuscript to Type 2
The Computing Perspective 2

The next two options in Figure 7.7 will be dealt with elsewhere, the ‘Style lines’ box in the next chapter and ‘Style labels’ later in this chapter and in Appendix 1. The first item in the ‘Style control’ box has a counterpart in *Ovation*: it avoids cluttering up lists of available styles. The second option is available in all the Archimedes DTP programs: with this you can define a time-saving key shortcut for applying a style at the cursor or to a marked section (or, in some programs, a paragraph).

Stylesheets

When we were considering master pages in Chapter 6 I mentioned that most of the Archimedes DTP packages allowed you to preserve the master pages from a

document separately from any material in the document itself by saving a *stylesheet*; I glossed over how a stylesheet is used. It can be looked on as a sort of skeleton document containing the page format and any master pages and styles that have been defined, so if you will be producing a series of documents which all need to look consistent you can set up the first one and save its stylesheet. You shouldn't forget to save the document itself as well, of course: the stylesheet contains the template information but none of the pages themselves.

If you want to store something like a letterhead as a stylesheet, remember that, as only the master pages are saved, any text or logos must be on a master page. This has the side-effect that the letter heading, being on the master page, will also appear on page 2, page 3 and so on, whereas it is common practice for such follow-on pages to be blank. There may be occasions when you can take advantage of this fact, but otherwise you are probably better off saving a form document consisting of a blank master page and the letterhead on page 1. This is similar to the way that *Impression* works: it doesn't use stylesheets as such so you have to save a whole blank document instead.

Loading Stylesheets. Dragging a stylesheet's file icon onto the DTP icon on the icon bar causes a new document to be opened containing the master pages and styles defined in the stylesheet. In addition, *Ovation* lets you merge stylesheets so that styles (but not master pages) can be accumulated in one document.

The reason *Impression* doesn't have a stylesheet function is that it doesn't need one: if you want to add the styles from a document on disk to one you have open, you drag the document icon from the directory display onto the big downward arrow marked 'Style' at the top right of the 'Edit style' dialogue box (see Figure 7.6). Any styles in the document on disk with different names from those in the document in memory will be added to the list of style definitions.

Defaults. On page 126 I mentioned setting up a default text size in *Impression*. You may of course want your DTP software to start up with other settings convenient to you, such as page size, paragraph styles and spelling checker options (if available). *Impression* has a Preferences menu and a !Default document in the !Impress.Auto directory; *Ovation* lets you save your choices as a stylesheet called Default in the !Ovation directory; *Acorn DTP* has a stylesheet called dtp_style in the !DTPWork directory. They all do much the same thing, but the more powerful the software, the more choices you have.

Style Management. It isn't a good idea to have too many styles in one document, by the way, because choosing from a long list slows you down; your document will also probably look very bitty if you put it together from lots of styles. If you have built up a long list of them over time, see if there are some that you can delete because they

are no longer needed, or perhaps you can combine the effects of styles that you always use together.

Style Tags in Imported Text

So far we have treated text for importing as a sort of bland paste to which various flavourings (styles and effects) are added after it is loaded into DTP. Most users of DTP will find this convenient enough for their purposes, but some may be able to make use of the capability of *Acorn DTP* and *Impression* to interpret special commands typed in the text as style changes and certain other functions. In a way this is analogous to the coding system used by *Draw*, described in Chapter 4, but is optional rather than obligatory. You might at first think that this would only be of interest to the sort of people who enjoy the challenge of learning new programming languages, as indeed it might; but a different use, and one used in the commercial world, is where the text is being word processed by several different people for assembly into documents of consistent format by someone else. So if you have a magazine that many people contribute to by sending their texts on disk, this could be for you: all you have to do is provide the contributors with a list of commands and how to use them.

The flexibility of this system, sometimes known as *tagging* the text, varies from program to program, as do the command structures. For instance, *Acorn DTP* uses angle brackets, < and >, round the names of paragraph styles, which means that (i) they can be applied only to a paragraph at a time and (ii) a tag in the middle of a paragraph isn't recognised as such: it is merely imported as text in angle brackets. Any tag that doesn't already exist as a style will be created as a style with the same characteristics as Body Text. *Impression*, on the other hand, uses curly brackets, { and }, that can enclose a style name, change a single text attribute, define an entire style or merely insert a non-keyboard character; they can be placed anywhere in the text.

If tagging sounds as though it could be useful to you but it all sounds dauntingly complex, you can relax: there is no need to commit yourself fully at the outset. Try using a few simple tags or commands at first and work up from there. You may surprise yourself. As typing errors in tags can cause problems, it is best if the word processor used allows the entry of pre-defined text strings with the function keys, as they can then be programmed in advance with the most common tags – or, better, all of them: keep the number down to avoid confusion. If nothing else, it may force you to design your DTP document around only a few styles; simplicity can be elegant if it is well thought out.

Exporting Text

After hours of planning and executing a DTP document, arranging the elements so that everything is balanced and fits perfectly, you might think it a complete waste of time if you then simply save the text as a plain *Edit* file. But this isn't as stupid as it sounds, and both *Impression* and *Ovation* allow you to do it. Only one story at a time can be saved, but that isn't surprising as this is also the way that they are imported.

Text saved from *Ovation* will be completely plain, innocent of style information, containing only line feeds (control code 10 or &0A (Ctrl-J)) in place of new line, new paragraph or new page commands, and control code 9 (Ctrl-I) instead of tabs.

In contrast, *Impression* lets you save text without style information if you want, but much more useful is the ability to save the complete style definitions with the text and all the points where they are turned on and off throughout the story. This means that you can re-import it into a blank document wherein all the style changes will be reproduced ... well, almost: any styles in the blank document will not be redefined by styles of the same name in the imported text. Deleting all the styles in the blank document would overcome this – but you can't delete BaseStyle, the single most important one. I leave it for you to work out how to get round this: it is possible.

Saving a blank *Impression* document separately from its text is a way of economising on disk space if you constantly use the same basic design, because the text file takes up less room than a complete document. As long as you don't use illustrations you can keep the texts of, say, letters or newsletters in a compact form in the knowledge that the documents themselves can easily be reconstituted.

Advanced Text Features

Once you have got the hang of the basics of word handling in your DTP work, there are a number of typographical refinements that you can make use of to add that touch of class – and variety. This is another area where DTP is a major advance over typewriting, offering you facilities that were once the exclusive province of typesetters: on a typewriter they were either fudged or impossible.

Non-keyboard Characters

The range of characters you can print in any font is far greater than is available by pressing the letter keys on the keyboard, with or without Shift. Acorn have specified that in fonts for use on the Archimedes series all the extra characters should be as defined in an international standard (currently the ISO Latin 1 alphabet, though this may change in due course), which for the code numbers 143 and above are as shown

143	•	144	‘	145	’	146	‹	147	›	148	“	149	”	150	„
151	–	152	—	153	−	154	Œ	155	œ	156	†	157	‡	158	fi
159	fl	160		161	¡	162	¢	163	£	164	¤	165	¥	166	¦
167	§	168	¨	169	©	170	ª	171	«	172	¬	173	-	174	®
175	¯	176	°	177	±	178	²	179	³	180	´	181	µ	182	¶
183	·	184	¸	185	¹	186	º	187	»	188	¼	189	½	190	¾
191	¿	192	À	193	Á	194	Â	195	Ã	196	Ä	197	Å	198	Æ
199	Ç	200	È	201	É	202	Ê	203	Ë	204	Ì	205	Í	206	Î
207	Ï	208	Ð	209	Ñ	210	Ò	211	Ó	212	Ô	213	Õ	214	Ö
215	×	216	Ø	217	Ù	218	Ú	219	Û	220	Ü	221	Ý	222	Þ
223	ß	224	à	225	á	226	â	227	ã	228	ä	229	å	230	æ
231	ç	232	è	233	é	234	ê	235	ë	236	ì	237	í	238	î
239	ï	240	ð	241	ñ	242	ò	243	ó	244	ô	245	õ	246	ö
247	÷	248	ø	249	ù	250	ú	251	û	252	ü	253	ý	254	þ
255	ÿ														

Figure 7.10 *Characters with numbers from 143 to 255, as defined by Acorn*

in Figure 7.10. To access one of them, hold down the Alt key while typing the appropriate code number on the numeric keypad at the right of the keyboard. When you release the Alt key the letter should appear. This technique is in fact quite general and can be used to choose any character in a font: try the numbers 72, 105 and 33, releasing the Alt key after each.

It is just possible that all that will happen when you do this is that a string of numbers will appear. Don’t worry, your computer isn’t broken: it is just that the software module built into it that responds to the Alt technique has been disabled, probably by some naughty software. To cure it, press F12 to leave the Desktop for a moment and at the * prompt type:

```
RMREINIT INTERNATIONALKEYBOARD
```

(There should be no space in the second word, long though it is.) Then press Return twice to re-enter the Desktop.

Now let us look at the specific uses of some of these characters which can make your finished products stand out from the crowd.

‘Real’ Quotes. Characters 144 and 145, and 148 and 149, are matched pairs of opening and closing quotation marks, single and double respectively, as used by typesetters (145 is also the proper apostrophe). Typing quotation marks from the key

next to Return on the keyboard will give you only undifferentiated objects that serve as both openers and closers, the so-called ‘sexless’ quotes (' and ").

If you can, get used to inserting the proper quote marks: they look much more professional. Using the Alt key can be rather laborious, though, and there are three shortcuts you can adopt instead. Two are general and are discussed a little later; the other is specific to quotes and works in *Impression*: they can be accessed with Ctrl-] for ‘, Ctrl-\ for ’, Shift-Ctrl-] for “ and Shift-Ctrl-\ for ”. This is unfortunately not very memorable, although you may find it easier than remembering numbers, and also less disruptive to your flow while you are typing.

The use of proper quotes is so important that a function has been built into the standard text-importing part of Archimedes DTP software that automatically translates sexless quotes in the text into opening and closing ones. Such a ‘smart quotes’ feature is valuable if you want it; if you don’t, you can turn it off in *Impression* by altering the Preferences dialogue box. Be aware that there is no perfect smart quotes procedure: it can be fooled sometimes and put in the wrong sort. Try importing the word *’tis* or the possessive *Saturn 4’s* from a text file.

Dashes. On a typewriter the minus key has to serve a wide variety of purposes: as a hyphen, as a longer dash between words or numbers, as a grammatical dash, sometimes with a space each side – like that – as well as a minus sign, which is different again. Using each one correctly is a mark of professionalism, even if your readers don’t notice them consciously. It is sad but true that quality is not always in the eye of the beholder.

The hyphen is the commonest and shortest of these dashes and is what you get when you press the minus key next to the figure 0. This is not the place for a grammar lesson: look it up in one of the well-thumbed, paper-covered, grease-stained books in your local library.

The en dash, –, is number 151 and is used in two ways: (1) to join two compatible words in such a way as to mean ‘and’ or ‘to’, as in ‘the Canadian–American border’ or ‘the Paris–Nice route’; and (2) to join together two numbers to indicate a range, as in ‘15–21 °C’ (note the use of character 176, the degree sign) or ‘1968–1972’. In the second usage the dash should be read as ‘to’, not ‘and’, so the phrase ‘between 15–21 °C’ would be wrong.

The em dash, —, is number 152 and is rarer than the en dash, as well as being twice as long. It used to be the standard grammatical dash and was printed without a space either side—like that. Modern usage has moved towards a spaced en dash for this purpose, a practice that I have followed in this book.

The minus sign, −, which is number 153, looks very like an en dash but is fractionally (almost invisibly) longer. Used as a *binary operator* in equations (such as $2x - y = 16$) it should be spaced on either side with a fixed space (see below); however, used on its own (a *unary operator*) it should be tight against the number or algebraic variable that follows it (for example, −16, −6*z*, −*t*). I could go on at length, but that is probably enough maths for now.

Ligatures. If you look closely at characters 158 and 159 you will see that each of them is two letters joined together. These are the last vestiges of a long and honourable line of *ligatures*, groups of letters that for sound practical reasons (or occasionally as an ornamental flourish) were set as one character in the days of hot-metal typesetting. Certain common letter combinations would simply not fit together, particularly fi, fl, ff, ffi and ffl, but also *gy* in some italic fonts. The ligatures were therefore designed to avoid breakage of the type. (Some uncommon combinations such as fk as in Stiffkey caused problems, but were rare enough not to warrant ligatures.) The fi, etc., ones still persist in 'proper' setting because they look better than the separate letters in fonts of traditional design, but most modern fonts are designed not to need them so they are falling into disuse. Acorn have unfortunately provided space in each font for only two of the bare minimum set of three, so their usefulness is limited. Use them by all means to experience the warm glow of self-satisfaction, but don't expect anyone to notice. This book has them all the way through: can you honestly say that they registered with you?

I separate the foregoing typographical ligatures from the phonetic ones Œ, œ, Æ and æ principally because if you use the latter they will definitely be noticed! These ligatures are commonly (and arguably incorrectly) called *diphthongs*. They are quite rare in English usage now, though they were once more common in words like Ægean, mediæval and œcology (now spelt ecology). They still crop up in Scandinavian words, though.

Fixed Space. Character 160 is an oddity because nothing is visible in the table. This a space, but no ordinary one: it is a *fixed space*. As far as the eye and the printer is concerned, it is a non-printing character (i.e. a space) of the same width as a normal space in unjustified text, but to word-processing or DTP software it looks like an ordinary letter. There are a couple of reasons why this is important: (1) putting a fixed space in a large number or a telephone number stops it from being split over a line (£100 at the end of one line and 000 at the start of the next looks very unprofessional), and (2) if you are *really* fussy there may be spaces that mustn't be expanded during justification – for instance, large numbers again. In Chapter 3 I referred to *hard spaces*; they are the same thing, but had a different function there.

Bullets. You have probably noticed that lists in this book have usually had a blob character, •, at the start of each item; this is called a *bullet*. Its character number is

143, which means that you can enter it by hand, but *Acorn DTP* (and *EasiWriter*, for that matter) let you define a bulleted style that automatically starts each paragraph in that style with a bullet character of your choice (some others are • ✔ ✘ ⌘ ✳ ☆ ★ ❒ ➧). In *Impression*, Shift-Backspace (the grey key below F12) inserts a bullet.

Non-Monospaced Digit. It is not widely known that all the numerals in 'proper' fonts are designed on the same width; they are therefore called *monospaced*, as opposed to proportionally spaced (where characters are different widths). These numbers line up properly in tables, but as a result the number 1 looks as though it has a small space on either side. For work in text you may want to avoid this, and most fonts from EFF provide a non-monospaced figure 1 as character 131.

Other Means of Access. Looking up character numbers in a table, even one close to the computer, is a bind and really slows you down; although you will quickly learn some of the more useful numbers the Alt method is rather tedious. Fortunately there are a couple of ways of speeding up the use of these non-keyboard characters.

A number of applications are available to let you pick a letter from a table such as that shown in Figure 7.11: you just point and click. The two most common are *Chars*, which is provided with *Impression*, with *PipeDream* or on Careware disk 7 from Norwich Computer Services, and *CharSel*, which comes with *Ovation*. The font shown is Oxford.Bold from the Electronic Font Foundry. (Incidentally, if you look carefully you will notice that to the left of the pointer tip are a set of characters not in Figure 7.10: this is because EFF thoughtfully provide them as extras.) These character-picking aids still have the drawback that they don't avoid disruption to the flow of your work at the keyboard: you still have to break off to use the mouse. However, the advantage is that you don't need to trouble with all those codes.

A less rigorous way to get at the top set, because it involves more than one stage and

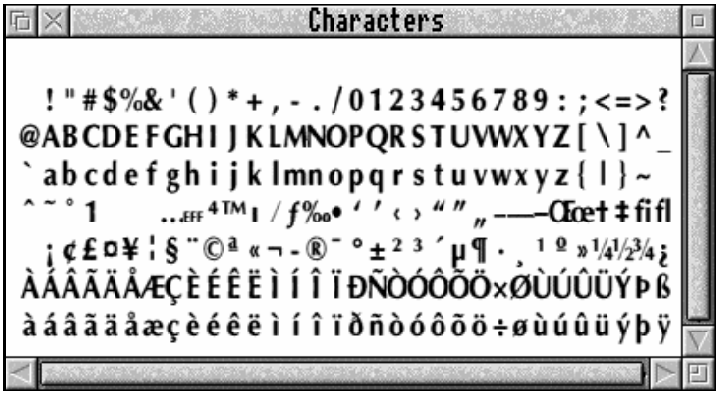


Figure 7.11 *The application Chars, used for choosing awkward characters*

could get forgotten about if you are in a rush, is to use a process of substitution: where you need a curious character just type an otherwise unused one in its place, and only when you have finished the whole document do you search for all the place-holders and replace them with the correct ones. Of course, you should make a list of what you are using and you *must* be consistent. Nevertheless, this can be a valuable technique and is one that I used extensively in preparing the text of this book; it is of most help for long documents. My main substitutions were double quote for opening quote and single quote for closing quote (which also meant that apostrophes were correctly handled), and double hyphen for en dash. The ligatures fi and fl were dealt with automatically in that no substitution was needed: I just searched for the separate letters.

The Last Word on Alt. Yes, there is more. The Archimedes has a small number of undocumented key shortcuts involving Alt, mainly for little-used characters. There is no systematic arrangement, but the shortcuts are designed to be memorable in case you need them; the available characters have some relation to the Alt-letter that produces them. Figure 7.12 lists those that I have found: maybe you will find others. On the left of each pair is the key you press with Alt, and on the right is the character that results. If you get different effects it may be because you have a different *keyboard handler module* loaded, as the characters shown are what the UK handler produces.

Alt-	C ¢	M µ	R ®	S §	X »	Y ¥	Z «
	~ ¬	1 ¹	2 ²	3 ³	9 ±	0 °	
	< ×	> ÷	space = hard (fixed) space				

Alt-Shift- C ©

Figure 7.12 *Characters available by pressing Alt and a key simultaneously*

Kerning and Tracking

Kerning. Take a look at the following letter pairs, chosen more or less at random:

AY ET FA LW QU

Although each pair was typed normally, without justification or any space between the letters, the visual effect of the letter combinations is such that they each appear to have different amounts of space between them. This can be quite subtle, as you can

see by comparing the ET and QU pairs, or it can be striking, as between AY and ET. This effect arises because there is really no such thing as the 'correct' width for a letter: a designer of a font has to find a compromise width for each that will be almost right for as many letter-pair combinations as possible. This means that as the letters are printed larger and larger the compromise wears thinner – for some combinations faster than others. In most cases this is best corrected by closing together slightly, or *kerning*, individual letter pairs to balance the overall visual effect, although it is possible to apply reverse kerning and increase the spacing of the closer pairs slightly. The amount of kerning to apply is very much a subjective matter and depends on which letters are involved and also on their context: see which of the two sets below you prefer, given that each pair is in this instance more or less in isolation:

AY ET FA LW QU
AY ET FA LW QU

To my eye the first pairs are better than the second, because the latter have been kerned excessively, bringing the letters too close together for normal use, and resulting in poorer readability. Of course, you might want to achieve a special visual impact by squeezing letters right up against each other, but this should be a conscious design decision rather than over-enthusiastic kerning.

The need for kerning only arises in displayed material such as titles and headings; there is no point in trying to kern body text manually. On computers other than the Archimedes, some DTP software provides a table of letter-pair kerning adjustments so that even body text can be kerned automatically. Not every possible combination of letter pairs is included (it would slow processing down too much and would also make the table very large); about 500 are normally enough for good results. Each font needs a separate table because the letters in different fonts fit together in different ways. The end result is that body text looks 'tighter', and larger sizes such as headings need less manual adjustment. At the time of writing, such tables cannot be used with the Acorn outline fonts on the Archimedes, but a new font manager currently under development will have this facility.

Tracking. We have already seen tracking in action, on page 130, where it was used in *Ovation* to force a justified line to fit, or to prevent immoderate space between words. It is more properly defined as an automatic adjustment to the spacing between characters to suit the point size, which is necessary because the inter-character spacing is designed to be best at a certain size. Smaller text needs spacing out

slightly overall and larger text needs closing together, quite apart from the needs of individual letter combinations for separate treatment. Like kerning tables, automatic tracking is not yet available on the Archimedes, but the manual tracking facility can be used in this way for headings in capitals if you feel the need (it was traditional in 'quality' typesetting to add small spaces between capital letters in headings to make them appear less heavy), or it can be used to tighten them for emphasis. If your DTP program doesn't have tracking, kerning can be used with the same effect, although it takes longer to do.

Special Effects with Fonts

As we are nearing the end of the chapter on text and approaching that on graphics, in this section we shall be looking at ways in which text can be used as the basis for graphic-like treatment. Here we shall begin to see the need for self-control, the conscious decision **not** to use all the facilities on offer simply because they are there. The key to good design lies in rejecting what is superfluous: anything that is surplus to the effective communication of your message will be a barrier between you and your reader.

Expanded and Condensed Type

This is little more than a note or reminder that even with your DTP program (unless it be *Acorn DTP*) you can modify the appearance of a font by making the letters wider or narrower with the 'aspect ratio', 'horizontal scale' or 'font width' controls. This is the simplest of the text manipulation tools, yet it is often overlooked. Also, don't forget that by condensing the body text slightly you can fit more words in a given area.

Creating Graphics from Text

If you have ever experimented with the Text option in *Draw* (the capital T on the toolbox) you will no doubt have been disappointed to find that once text has been entered it can only be stretched or squeezed by selecting it as an object and pulling the relevant 'ear'; there is no facility for other distortion or general playing around with the letter shapes. Fortunately there is a good range of software for converting text into a form in which it can be manipulated as one or more *Draw* objects with editable paths rather than as a single text object. We shall look at importing the image itself into DTP in Chapter 8, but meanwhile here are some of the possibilities.

FontDraw. The first package to consider is also the simplest: *FontDraw*, which is supplied with *Impression*, allows you to type in a line of text in any font you have available and convert it into a simple *Draw* file, with different outline and fill

colours. Figure 7.13 shows the program's dialogue box, and Figure 7.14 a sample letter loaded into *Draw*. In (*a*) the outline of the letter has been thickened and the fill colour altered to make a contrast. The next step was to choose 'Ungroup' in the Select submenu (this seems to be necessary whether there are several letters or only one) and then 'Edit' to show the object's paths, as in Figure 7.14(*b*). Once you have the letter in this form it can be reshaped by hand (see Figure 7.15) or by software.

DrawBender. The next step up in complexity, and in this case leading directly on from *FontDraw*, is *DrawBender* from Ian Copestake. This takes a *Draw* object (not necessarily a letter or a word, of course) and distorts it within an outline mould of your choice or design. It isn't lightning fast but it is entertaining, and it comes with three fonts. A selection of the effects achievable with words is shown in Figure 7.16.

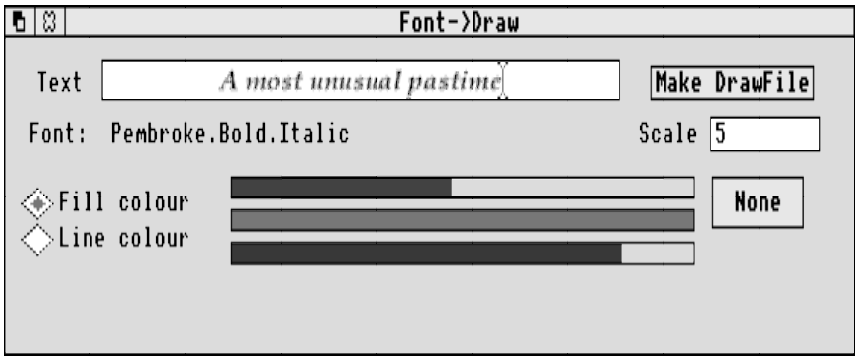


Figure 7.13 *FontDraw's dialogue box*

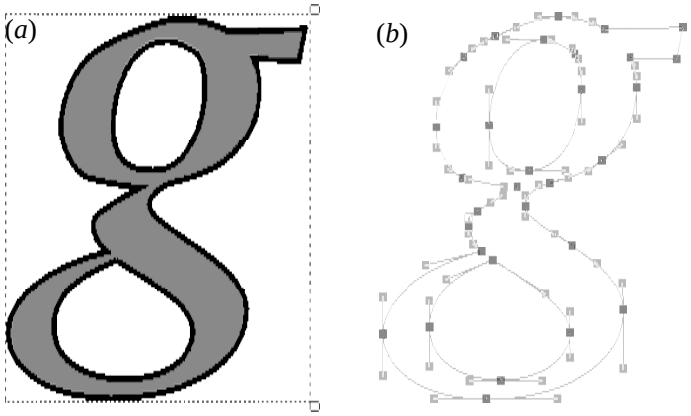


Figure 7.14 *(a) A letter selected as an object in Draw; (b) the same letter as a set of lines and control points, ready for editing*

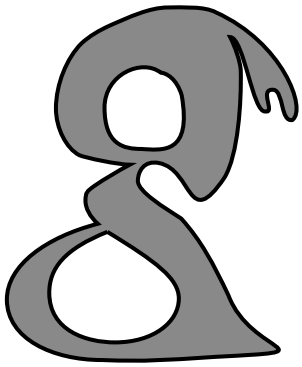


Figure 7.15 *A 'melted' letter g created by editing its shape in Draw*

Figure 7.16 *Some of the manipulations that can be performed with DrawBender*

FontFX. We now move on to software that accepts text as an input and converts it to a *Draw* file, simultaneously adding a variety of flourishes or appendages. Such a program is *FontFX* from the Data Store. The dialogue box in Figure 7.17 indicates the range of options, and Figure 7.18 shows some of the results that can be obtained.

Both *DrawBender* and *FontFX* are inexpensive (around £10 each), and together with *FontDraw* provide many of the creative tools that the average DTP designer will need. You may need others, though, and so we move on to...

Poster. At the top of the range of software for manipulating *Draw* images is *Poster* from 4Mation Educational Resources. To a large extent it is like *Draw* itself, but with many more alternatives; like *Draw*, it can also be used as a DTP package in that it will accept text prepared beforehand in a word processor and combine it with

graphics. For the purposes of this chapter it is enough to note that *Poster* can take small amounts of textual input, or the output from one of the other programs men-

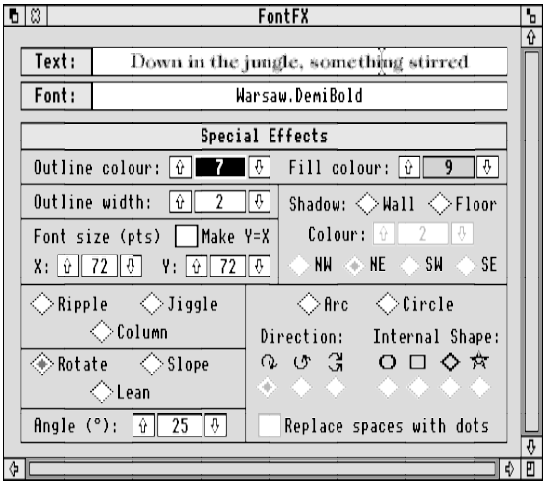


Figure 7.17 *FontFX's dialogue box, demonstrating its scope*

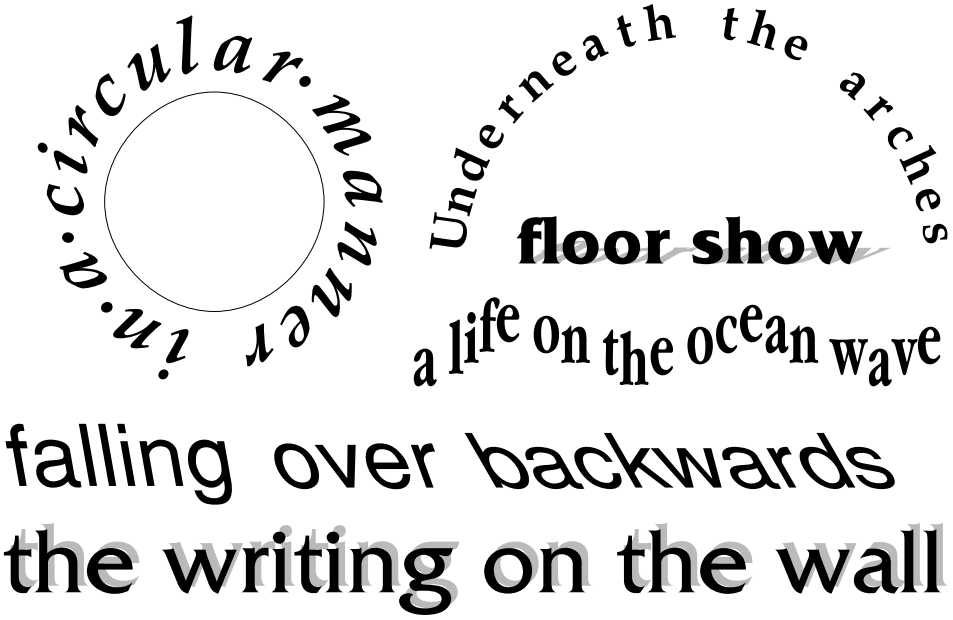


Figure 7.18 *Some simple uses of FontFX's facilities; many more can be achieved by combining different effects*

tioned above, and mould, shape and shadow it in a wide variety of ways. Figure 7.19 shows a few of the simpler effects that you can achieve; once you know what you are doing, and provided that you have enough imagination, you can design your own moulds and really go to town. However, as the figure demonstrates, for the purposes of legibility you should try to restrain yourself.

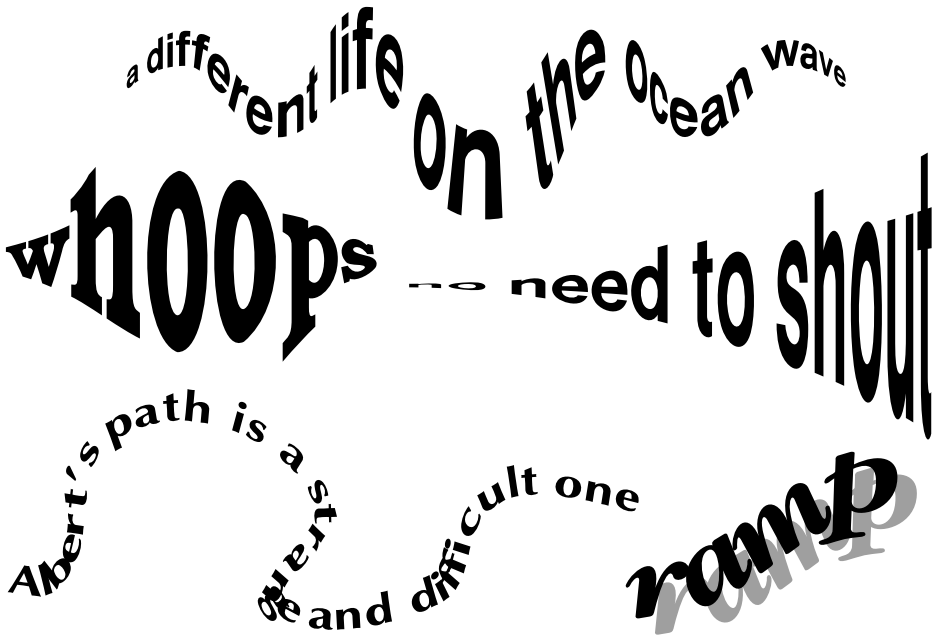


Figure 7.19 *Using Poster to shape and adjust text*

Equasor. A rather specialised piece of software for DTP purposes is *Equasor* from Computer Concepts. It is designed to let you enter complex mathematical formulae and include them as *Draw* files in your text. You don't have to be an expert at mathematics to use it, but some basic knowledge helps. Figure 7.20 gives examples of what can be done with it.

While we are thinking about mathematical work, I should mention a promised all-round package from Icon Technology called *TechWriter*. This is expected to be an advanced version of *EasiWriter* (see Chapter 3) with the addition of mathematical typesetting and equation-building facilities. The mathematical part was developed from *Formulator* on the Apple Macintosh and should even allow the output of work created on it as a document complete with TEX codes. (TEX, pronounced more or less like *tek*, is a language widely used in mathematical typesetting, so the ability to produce such output greatly increases the compatibility of the Archimedes with the

Integrate $\frac{(1+\sqrt{x})^2}{x}$

$$\int \frac{(1+\sqrt{x})^2}{x}\,dx = \int \frac{1+2x+x}{x}\,dx$$

$$= \int \frac{1}{x} + 2x^{-\frac{1}{2}} + 1\,dx$$

$$= \log x + 4x + x + c$$

IF $f(x) = 3x^2$ *evaluate* $\lim_{h\to 0} \frac{f(x+h)-f(x)}{h}$

$$\lim_{h\to 0} \frac{f(x+h)-f(x)}{h} = \frac{3(x+h)^2 - 3x^2}{h}$$

$$= \frac{3x^2 + 6xh + 3h^2 - 3x^2}{h}$$

$$= 6x + 3h$$

$$\lim_{h\to 0} f(x) = 6x$$

$$L^{-1}\left(\frac{2s^2+7}{(s^2+4)(s+3)^3}\right) = \left(\frac{-23}{2197} - \frac{6}{169}t + \frac{25}{26}t^2\right)e^{-3t} + \frac{23}{2197}\cos(2t) + \frac{9}{4394}\sin(2t)$$

Figure 7.20 *A few examples of mathematical setting with Equasor*

wider mathematical fraternity in this respect.) It isn't DTP, though, by the definition I use.

Shadow Text within DTP

In the previous section we have seen some remarkable effects created by using extra software. Before you get carried away, let us look briefly at some quite acceptable, though simpler, designs that can be built up with the facilities provided by the DTP programs themselves. The technique used is perfectly general and will work with all the DTP packages on the Archimedes.

Figure 7.21 *Three forms of shadow text easily produced without artificial aids*

Several different effects can be achieved, three of which are shown in Figure 7.21, all from the same basic principle. The idea is to have two frames containing text, one over the other but displaced slightly. Just create a frame, add the text, make a copy of the frame and play around with the colours of the text and the frame background or transparency in each. The most important thing is to remember to make the one on top non-repelling to text.

So, in the first example the upper frame is set transparent and the text colour is black, and the lower frame's text is grey. The second is a more subtle effect, and for it to be successful you really have to use a bold font and only a small displacement (less than the line thickness of the letters): the colours used are obvious. To produce the third example rapidly I had to cheat slightly and put a third frame, whose colour was grey, behind the other two, both of which were made transparent. However, if you were routinely going to be preparing headings in this way you should be able to manage with only two, setting up the correct positioning of the text within the lower frame with styles and effects.

As ever, the caution against overuse of such obviously 'DTP' artefacts applies, unless you are trying to achieve that sort of appearance for reasons of your own.

A Finishing Touch: Contents and Indexes

A document of any size is going to need some sort of contents list and perhaps also an index. If you have *Impression* you are fortunate because facilities for generating these are built in; otherwise, although DTP in general isn't really geared to this, even without specific facilities in your application there are quicker, or at least more accurate, ways of preparing contents and indexes than writing them on a sheet of paper and typing them in again. With *Acorn DTP*, which allows only one document in memory at a time, you will have to create some space, such as a spare page, in your document so that you can build up the contents or index text; if you are an *Ovation* user, just open a fresh document. Then just go through the final version of the document, marking the required entries one by one, copying and pasting them to the area set aside for them and adding page references as you go along. In this way you can be sure that the entries will agree exactly with what is in the text (including, naturally, any mistakes). Remember to use key or mouse shortcuts rather than menus wherever possible; also, try working with one hand on the keyboard and one on the mouse, because there is not much typing involved, only manipulation.

The contents and indexes facility in *Impression* is centred on the 'Style labels' box in the Edit style dialogue box (see Figure 7.7). In principle all you have to do for a contents list is to incorporate a contents level number into each heading style and choose the 'Compile contents' option in the Misc submenu. For indexes you can

define a separate style with only the 'Index label' button and perhaps a font background colour of yellow (giving you an effect on screen like a highlighter pen), then choose 'Compile indexes'. In both cases the software then works through the document and picks out all the text pieces containing the relevant flags, assembling them at the start or end of the document, as appropriate. If you do highlight the index style, remember to set the font background colour to white again before printing the document out. A little more detail on this topic can be found in Appendix 1.

Hints and Tips

Save, Save, Save. You can never save a document to disk too often. Cultivate the habit of saving every five minutes or so. Sometimes problems can be caused by a software error or a hardware crash, leading to the application vanishing or a Reset being your only option – and there goes your document. Or you might accidentally delete something crucial instead of cutting it to the clipboard. It is unbelievably frustrating to find that you have just lost the fruits of an hour's labour and there is no other record of the changes or additions that you made during that time. Software is becoming less likely to crash now that relatively bug-free versions are being released, but there is no such thing as absolute security in computer memory: so take heed. Fortunately, *Ovation* and the later versions of *Impression* have auto-save features, so you don't even have to remember this advice. You can just set up the program to perform a timed save every few minutes.

Back it up. Similarly, just as computer memory isn't secure, disks aren't infallible. At the end of a DTP session you should make a security copy of any documents you have changed (especially the important ones) onto a separate disk. If you are lucky enough to have a hard disk, keep a backup or archive copy on a floppy disk; better still, use the 'grandparent–parent–child' system so you always have the two previous versions as well as the current one in case of dire accidents. The way I do it is to have three directories (Grandad, Dad and Son) for each document, and on backing up I avoid extensive copying by:

- deleting the Grandad directory (and therefore the oldest version as well);
- renaming the Dad directory to Grandad;
- renaming the Son directory to Dad;
- creating a new Son directory and opening it; and
- copying the latest version into the Son directory.

Renaming a directory takes considerably less time than copying files around, especially if they are long ones (like this book, which as I write already occupies 2.5Mb on disk – even a hard disk has to work hard at that).

Even Line Spacing. There is a curious effect in *Impression* that is sometimes visible if the first character on a line is in a different font from the lines around it; take a look at this, which is a reproduction of part of Figure 4.1:

`\L`*n*	line spacing (points)
`\M`*left right*	left and right margins
`\P`*n*	space between paragraphs (points)
`\U`*p t*	underline position and thickness
`\U.`	turn off underlining
`\V`*move*	vertical move

You can readily see that the fourth and sixth lines seem lower than all the others; this is because they start with characters in Trinity font rather than Corpus and this throws the line spacing out. The solution is simple but effective: put the cursor at the start of the affected line and choose the same font as the lines around it, then use the Alt key to enter a character that hasn't been defined in that font (try number 128, 131, 136 or 139). In this way you will place an invisible character without any width at the beginning of the line, and the spacing will be evened up.

Shift in *Chars* and *CharSel*. The character selection utilities *Chars* and *CharSel* have a secondary method for picking characters instead of clicking on them: if you press Shift the character under the pointer is inserted at the caret in the currently active document. While this can be a boon it can also be annoying if you forget that the pointer is over the selection window when you are trying to type a capital letter, because odd characters will suddenly appear in your text instead. To get round this, if you are feeling courageous you can load the Basic program !RunImage from either application directory (!Chars or !CharSel) and alter it so that the program uses a different key, such as Ctrl, which you are less likely to press by mistake. In *Chars* the relevant line, which is early in the program, reads:

```
HotKey%=-1
```

and all you have to do is change the −1 to −2 to make Ctrl do the trick. In *CharSel* the line (number 1670 in the version I have) reads:

```
IF INKEY-1 THEN
```

Again, just change −1 to −2 for the desired effect. A complete list of internal key numbers is given on pages 474–480 of the *Programmer's Reference Manual*: if you want to choose a different active key just substitute a different number. Calculate it by adding 1 to the number in the table and then put a minus sign in front.