



## Enforcer® Version 5.0

TechNote Number: ENF005  
Last updated: August 1999

### Controlling by Default

Control Files are the essential core of Enforcer and a thorough understanding of their function is a crucial step towards the efficient and effective use of the product as a labour-saving tool. In this issue we examine the **Default Control File**. To fully appreciate the format of the Default Control File, it is advisable to review the evolution of control files.

In the early days the standard control file consisted of the following comma-separated columns:

*Label*  
*Level*  
*Colour*  
*Weight*  
*Style*  
*Cell*  
*Scale*  
*Font*  
*Text Height*  
*Text Width*  
*Element*  
*Command*  
*Feature Description.*

Since then, at different stages some optional columns were introduced between the Command and Feature Description columns. They were:

*Text*  
*Line Spacing*  
*Text Justification*  
*Angle*  
*Pattern/Hatching*  
*Element Class*  
*Inter-Character Spacing.*

Older control files without these columns were still supported as long as the first character in the Feature Description column was a colon or semi-colon. This special “comment” character told Enforcer not to expect any more columns to follow, other than the Feature Description. Obviously this meant that the user could include as many of the optional columns as they wanted, as long as they appeared in order from left to right and as long as a special comment character preceded the Feature Description.

The next phase in the development took the concept of optional columns one step further through the introduction of the Default Control File. Since then users have been able to define their own control file layouts. They have also been able to preset default values for columns, eliminating the need to hard-code the default value repeatedly for a particular column.

In its simplest form the Default Control File consists of two special "Plus" statements that tell Enforcer where the layout starts and ends. Between these two statements the columns are listed in order, one per line. The column names are fixed: always upper case and coded exactly as shown below. After each column name and a comma, the default value of that column may be specified. This information is optional. A standard Default Control File with all columns defined, would look like this:

```
:  
: DEFAULT.CTL  
:  
: Column Name      , Default Value  
:  
+START_CONTROL_LAYOUT  
:  
LABEL              ,  
LEVEL              ,  
COLOUR             ,  
WEIGHT            ,  
STYLE              ,  
CELL               ,  
SCALE              ,  
FONT               ,  
TEXT_HEIGHT       ,  
TEXT_WIDTH        ,  
ELEMENT            ,  
COMMAND           ,  
TEXT               ,  
TEXT_LS           ,  
TEXT_JUST         , +U=LT  
ANGLE              ,  
PAT_HAT_GROUP     ,  
ELEMENT_CLASS     , +U=0  
TEXT_CHAR_SPACE  ,  
FEATURE           ,  
:  
+END_CONTROL_LAYOUT
```

Note that a special rule applies to Text Justification and Element Class: default values **must** be defined for these two columns.

The general rule that applies to all Default Control Files is that when Enforcer encounters one, its layout remains current until another one changes the status quo.

We'll illustrate this rule with an example: Imagine that I have a master control file (*master.ctl*), and one each for structural, electrical, plumbing and roofing features, called *struct.ctl*, *elec.ctl*, *plumb.ctl* and *roof.ctl* respectively. I also have a special control file for miscellaneous text features, called *text.ctl*. The majority of my control files have the same format. The exceptions are *text.ctl* and *roof.ctl* that each has a unique layout. I included all possible columns in the structural, electrical and plumbing control files, consequently the associated Default Control File would be identical in layout to the standard one listed above.

The roofing features are all simple linear ones, so I have no need for columns associated with text or cells, whereas the text file does not need the cell column, pattern hatching or element class. To employ all of these control files effectively and efficiently within the same project, I should create three Default Control Files, which I will call *default.ctl*, *def\_text.ctl* and *def\_roof.ctl*.

In *master.ctl* the section where I include the “sub” control files should be modified to the following:

```
+INCLUDE_CTL    DEFAULT.CTL
+INCLUDE_CTL    STRUCT.CTL
+INCLUDE_CTL    PLUMB.CTL
+INCLUDE_CTL    ELEC.CTL
+INCLUDE_CTL    DEF_ROOF.CTL
+INCLUDE_CTL    ROOF.CTL
+INCLUDE_CTL    DEF_TEXT.CTL
+INCLUDE_CTL    TEXT.CTL
+INCLUDE_CTL    DEFAULT.CTL
```

Of course, it is possible to include each Default Control File from within each sub control file, but with the recommended method illustrated above, it is easier to see which layout is current at a particular time. By including *default.ctl* again at the end of the list I have ensured that the layout is reset to my standard default format, something that I may forget to do when I add another file to the list later.

In short, the Default Control File enables users to break from the mould and to define their own layout for their own control files.

### **Australian Data Systems**

55 Morrison Road, Midland, Western Australia 6056

Ph: +61 8 9250 8044 Fax: +61 8 9250 8344

[ads@ozdata.com.au](mailto:ads@ozdata.com.au)

[www.ozdata.com.au](http://www.ozdata.com.au)