These are (most) all the boolean options that **indent** offers.
;parameter.options.rtfd;Parameter options;¬ For parameter options.
;buttons.rtfd;Buttons;¬ For button info.
;typedef.panel.rtfd;Setting typedef values;¬ Setting typedef values

Force Blank line after Block Declaration
If this option is specified, a blank line is forced after every block of declarations.

Force Blank line after procedure body
If this option is specified, a blank line is forced after every procedure body.

Force Blank line before block Comment
If this option is specified, a blank line is forced before every block comment.

Force New Line after comma declaration
If this option is specified, then a newline is forced after each comma in a declaration.

Line up compound statements

Specifying this option lines up compound statements that look like this:

```
if (...)
{
    code
}
```

So they look like this:

```
if (...) {
    code
}
```

Enable Blank line comments

This option enables the placement of comment delimiters on blank lines. With this option enabled, comments look like this:

```
/*
 * this is a comment
 */
```

Rather than like this:

```
/* this is a comment */
```

This only affects block comments, not comments to the right of code.

Nestle elses with preceding '}'

Enables (disables) forcing `else's to cuddle up to the immediately preceding `}'.

Left justify declarations

This option left justifies declarations. Otherwise indent indents declarations the same as code.

Indent "if's" as much as "if/else's"

This option enables special else-if processing. If enabled, ifs following elses will have the same indentation as the preceding if statement.

Format comments starting in column 1

This option enables the formatting of comments that start in column 1. Often, comments whose leading `/' is in column 1 have been carefully hand formatted by the programmer. In such cases, this option should be used.

Indent parameter declarations

This option enables the indentation of parameter declarations from

the left margin.

Justify up parenthesis text in split lines

This option lines up code surrounded by parenthesis in continuation lines. If a line has a left paren which is not closed on that line, then continuation lines will be lined up to start at the character position just after the left paren.

Convert =- to -=

If this option is used, old style assignment operators (`=-', `=*', and so on) are considered to be tokens, and are converted to the newer form (`-=', `*=').

Note: I couldn't ever get gcc to compile a line with the text '=-' in it, but I find this option usefull for not splitting '==' tokens.

Put a space before leading '('s

If this option is used, all procedure calls will have a space inserted between the name and the `('.

Put spaces arount '->'s

If this option is used, the pointer following operator `->' will be surrounded by spaces on either side.

Put procedure names in column 1

If this option is used, the names of procedures being defined are placed in column 1 - their types, if any, will be left on the previous lines.

Put '*' on the left edje of all commnets

This option enables the placement of asterisks (`*'s) at the left edge of all comments.

Delete extra blank lines

If this option is specified, indent will swallow optional blank lines. You can use this to get rid of blank lines after declarations.

Verbose mode (tell you what happened)

This option turns on `verbose' mode. When in verbose mode, indent reports when it splits one line of input into two or more lines of output, and gives some size statistics at completion.