NAME

case − Evaluate one of several scripts, depending on a given value

SYNOPSIS

case *string* ?**in**? *patList body* ?*patList body* ...?

case *string* ?**in**? {*patList body* ?*patList body* ...?}

DESCRIPTION

Note: the **case** *command is obsolete and is supported only for backward compatibility. At some point in the future it may be removed entirely. You should use the* **switch** *command instead.*

The **case** command matches *string* against each of the *patList* arguments in order. Each*patList* ar gument is a list of one or more patterns. If any of these patterns matches *string* then **case** evaluates the following *body* argument by passing it recursively to the Tcl interpreter and returns the result of that evaluation. Each *patList* argument consists of a single pattern or list of patterns. Each pattern may contain any of the wildcards described under **string match**. If a*patList* ar gument is **default**, the corresponding body will be evaluated if no *patList* matches *string*. If no*patList* ar gument matches *string* and no default is given, then the **case** command returns an empty string.

Two syntaxes are provided for the *patList* and *body* arguments. The first uses a separate argument for each of the patterns and commands; this form is convenient if substitutions are desired on some of the patterns or commands. The second form places all of the patterns and commands together into a single argument; the argument must have proper list structure, with the elements of the list being the patterns and commands. The second form makes it easy to construct multi-line case commands, since the braces around the whole list make it unnecessary to include a backslash at the end of each line. Since the *patList* arguments are in braces in the second form, no command or variable substitutions are performed on them; this makes the behavior of the second form different than the first form in some cases.

SEE ALSO

switch(n)

KEYWORDS

case, match, regular expression