User selections

Synopsis

```
select <NAME>; do
  <LIST>
done
```

```
select <NAME> in <WORDS>; do
  <LIST>
done
```

```
# alternative, historical and undocumented syntax

select <NAME>
{
  <LIST>
}

select <NAME> in <WORDS>
{
  <LIST>
}
```

Description

This compound command provides a kind of menu. The user is prompted with a *numbered list* of the given words, and is asked to input the index number of the word. If a word was selected, the variable `<NAME>` is set to this word, and the list `<LIST>` is executed.

If no `in <WORDS>` is given, then the positional parameters are taken as words (as if `in "$@"` was written).

Regardless of the functionality, the *number* the user entered is saved in the variable `REPLY`.

Bash knows an alternative syntax for the `select` command, enclosing the loop body in `{...}` instead of `do ... done`:

```
select x in 1 2 3
{
  echo $x
}
```

This syntax is **not documented** and should not be used. I found the parser definitions for it in 1.x code, and in modern 4.x code. My guess is that it's there for compatiblity reasons. This syntax is not specified by POSIX(R).

Examples

```
# select <NAME> in <WORDS>; do
#   <LIST>
# done


#   meaning    e.g.:

clear
echo
echo  hit number key 1 2 or 3 then ENTER-key
echo  ENTER alone is an empty choice and will loop endlessly until Ct
rl-C or Ctrl-D
echo

select OPTIONX in beer whiskey wine liquor ; do

  echo  you ordered a   $OPTIONX
  break # break avoids endless loop -- second line to be executed alw
ays

done

# place some   if else fi    business here
# and explain how it makes sense that $OPTIONX is red but OPTIONX is
 black
# even though both are variables
```

Portability considerations

See also

Discussion