[illegible]

[illegible]

1. [illegible]

Red[illegible]Red[illegible]Red[illegible]Red[illegible]Red[illegible] `[illegible]directives[illegible]` [illegible]directive[illegible]

- Red[illegible]
- `do` [illegible]Red[illegible]
- block[illegible] `expand-directives` [illegible]

[illegible]LOAD[illegible]Red[illegible]

[illegible]

- [illegible] [illegible]
- [illegible]

- □□□□□ □□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□□□□□ # □□□□□□□□□□ issue! □□□□□□□□□□□□□□□

□□□

NOTE	Red/System□□□□□ □□□□□□□□□ □□□

1.1. Config□□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ `config` □□□ □□□ □□□□□□

□□

```
#if config/OS = 'Windows [#include %windows.red]
```

NOTE: □□□□□□□□□□□□□□□□□□□□□□□Red□□□□□□□□□□□□□□□□□□□□□□□□□□□□ `config` □□□□□□□□□□□□□□□□□□□□□□□□□ `config` □□□□□□□□□□□□□□□□□□□□□□□□Red□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

1.2. □□□□□□□□□□□

□□□set-word□□□□□□ `#do` □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

Tips:

- □□□

  ```
  #do [probe self]
  ```

- □□

  ```
  probe preprocessor/exec
  ```

1.3. □□□□□□□□

□□Rebol2□□Red□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□Red□□Red□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

2. [illegible]

Red[illegible] [illegible] [illegible]Red[illegible]Red[illegible]

NOTE:[illegible]Parse[illegible]

[illegible]

2.1. [illegible]

[illegible]

```
#macro name: func [arg1 arg2... /local word1 word2...][...code...]
```

[illegible] `name` [illegible]word[illegible]

1. [illegible]
2. [illegible]
3. [illegible]
4. [illegible]

NOTE:[illegible]

[illegible]

```
Red []
#macro make-KB: func [n][n * 1024]
print make-KB 64
```

[illegible]

```
Red []
print 65536
```

[illegible]

```
Red []
#macro make-KB: func [n][n * 1024]
#macro make-MB: func [n][make-KB make-KB n]

print make-MB 1
```

□□□□□□□□□□□□□□□□□□

```
Red []
print 1048576
```

2.2. □□□□□□□□□□□□

□□□□□□□□□□word□□□□□□□□□□□□□□□□□□Parse□□

□□□□□□□□□□□□□□□□□□□□□□□□□ `[manual]` □□

□□□□□□□□□□□□□□□□□□□□□□□□□□

```
#macro <rule> func [<attribute> start end /local word1 word2...][...code...]
```

`<rule>` □□□□□□□□□□□□□□□□□□□

- lit-word!□□□□□□□□word□□□□□□□□□□□□
- word!□□□Parse□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ □□□□□□□□□□□□ □skip□□□□□□
- block!□□□□Parse□□□□□□□□□□□

□□ `start` □ `end` □□

`<attribute>` □□ `[manual]` □□□

□□

```
Red []

#macro integer! func [s e][s/1 + 1]
print 1 + 2
```

□□□□□□□□□□□□□□□□□□

```
Red []
print 2 + 3
```

[illegible]

```
Red []

#macro integer! func [[manual] s e][s/1: s/1 + 1 next s]
print 1 + 2
```

block[illegible]

```
Red []
#macro ['max some [integer!]] func [s e][
    first maximum-of copy/part next s e
]
print max 4 2 3 8 1
```

[illegible]

```
Red []
print 8
```

3. [illegible]

3.1. #if

[illegible]

```
#if <expr> [<body>]

<expr> : [illegible]
<body> : if <expr> [illegible]true[illegible]
```

[illegible]

[illegible]true[illegible] `<body>` [illegible]

[illegible]

```
Red []

#if config/OS = 'Windows [print "OS is Windows"]
```

Windows[illegible]

```
Red []

print "OS is Windows"
```

Windows[illegible]

```
Red []
```

#do [illegible]word[illegible]

```
Red []

#do [debug?: yes]

#if debug? [print "running in debug mode"]
```

[illegible]

```
Red []

print "running in debug mode"
```

3.2. #either

[illegible]

```
#either <expr> [<true>][<false>]

<expr>  : [illegible]
<true>  : if <expr> [illegible]true[illegible]
<false> : if <expr> [illegible]false[illegible]
```

[illegible]

[illegible]

[illegible]

```
Red []

print #either config/OS = 'Windows ["Windows"]["Unix"]
```

Windows[illegible]

```
Red []

print "Windows"
```

Windows[illegible]

```
Red []

print "Unix"
```

3.3. #switch

[illegible]

```
#switch <expr> [<value1> [<case1>] <value2> [<case2>] ...]
#switch <expr> [<value1> [<case1>] <value2> [<case2>] ... #default [<default>]]

<valueN>  : [illegible]
<caseN>   : [illegible]
<default> : [illegible]
```

[illegible]

[illegible]

[illegible]

```
Red []

print #switch config/OS [
    Windows ["Windows"]
    Linux   ["Unix"]
    MacOSX  ["macOS"]
]
```

Windows[illegible]

```
Red []

print "Windows"
```

3.4. #case

□□

```
#case [<expr1> [<case1>] <expr2> [<case2>] ...]

<exprN> : □□□
<caseN> : □□□□true□□□□□□□□□□□□□□□
```

□□

□□

□

```
Red []

#do [level: 2]

print #case [
    level = 1  ["Easy"]
    level >= 2 ["Medium"]
    level >= 4 ["Hard"]
]
```

□□□□□□□□□□□□□□□□□

```
Red []

print "Medium"
```

3.5. #include

□□

```
#include <file>

<file> : □□□□Red□□□□□ □file!□
```

□□

□□Red□□□□□□□□□
□□□□□□□□□□□□□□□□ do □□□□□□□□□□□□□□□□□□□□

3.6. #do

□□

```
#do [<body>]
#do keep [<body>]

<body> : □□□Red□□□□
```

□□

□□□□□□□□□□□□□□□body□□□□□□□□□□□ `keep` □□□□□□□□□ `body` □□□□□□□□□□□□□□□□□□□□□□□□□

□

```
Red []

#do [a: 1]

print ["2 + 3 =" #do keep [2 + 3]]

#if a < 0 [print "negative"]
```

□□□□□□□□□□□□□□□□□□□

```
Red []

print ["2 + 3 =" 5]
```

3.7. #macro

□□

```
#macro <name> func <spec> <body>
#macro <pattern> func <spec> <body>

<name>    : □□□□□□□□ □set-word!□
<pattern> : □□□□□□□□□□□□□□□□□block!, word!, lit-word!□
<spec>    : □□□□□□□□□□□□□□
<body>    : □□□□□□□□□□□□□
```

□□

□□□□□□□□□□□□

□□□□□□□□□□spec□□□□□□□□□□□□□□□□□□□□□body□□
□□□□□□□□□□□□□□□

□□□□□□□□□□□□□□spec□□□□□ 2□ □□ `func [start end]` □□□□□□□□ `func [s e]` □□ □□□□□□□□□□□□□□ □□□□□□□□□ □□□□□□□□□□spec□□□□□ `[manual]` □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ `func [[manual] start end]` □□□ □□□ □□□□□□□□□□ `start` □□□□□□□□□□□□□□□□□□□ □□□□ □□□□□□□□ `end` □□□

□□□□□□□□□□□□□□□□□□□□□□□□

- block□Parse□□□□□□□□□□□□□□□□□□□□□□□□
- word□Parse□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ □□□□□□□□□□□□ □skip□□□□□□
- lit-word□□□□□word□□□□□□□□□□□□

□

```
Red []
#macro pow2: func [n][to integer! n ** 2]
print pow2 10
print pow2 3 + pow2 4 = pow2 5
```

□□□□□□□□□□□□□□□□□□

```
Red []
print 100
print 9 + 16 = 25
```

□□□□□□□□□□□□□

```
Red []
#macro [number! '+ number! '= number!] func [s e][
    do copy/part s e
]

print 9 + 16 = 25
```

□□□□□□□□□□□□□□□□□□

```
Red []
print true
```

□□□□□□□□□□□□□□□□□□□□□

```
Red []
#macro ['sqrt number!] func [[manual] s e][
    if negative? s/2 [
        print [
            "*** SQRT Error: no negative number allowed" lf
            "*** At:" copy/part s e
        ]
        halt
    ]
    e              ;-- □□□□□□□□□□□□□□□□□□□□□□
]

print sqrt 9
print sqrt -4
```

□□□□□□□□□□□□□□□□□□□

```
*** SQRT Error: no negative number allowed
*** At: sqrt -4
(halted)
```

3.8. #local

□□

```
#local [<body>]

<body> : □□□□□□□□□□□□□□□□Red□□□
```

□□

□□ `#local` □ `body` □□□□□□□□□□□□□□□□□□□□

□

```
Red []
print 1.0
#local [
    #macro float! func [s e][to integer! s/1]
    print [1.23 2.54 123.789]
]
print 2.0
```

□□□□□□□□□□□□□□□□□

```
Red []
print 1.0
print [1 3 124]
print 2.0
```

3.9. #reset

□□

```
#reset
```

□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□word□□□□□□□□□□□□□□

3.10. #process

□□

```
#process [on | off]
```

□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□Red□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ `#process off` □□□□□□□ `#process on` □□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□

```
Red []

print "Conditional directives:"
#process off
foreach d [#if #either #switch #case][probe d]
#process on
```

□□□□□□□□□□□□□□□

```
Red []

print "Conditional directives:"
foreach d [#if #either #switch #case][probe d]
```

3.11. #trace

[illegible]

```
#trace [on | off]
```

[illegible]

[illegible]Red[illegible]

4. [illegible]API

Red[illegible] `do` [illegible] `file!` [illegible] [illegible] `do` [illegible] `do load %file`

4.1. expand-directives

[illegible]

```
expand-directives [<body>]
expand-directives/clean [<body>]

<body> : [illegible]Red[illegible]
```

[illegible]

[illegible] `/clean` [illegible]

[illegible]

```
expand-directives [print #either config/OS = 'Windows ["Windows"]["Unix"]]
```

[illegible]Windows[illegible]

```
[print "Windows"]
```