

```

0 \ OVERLAY PREFIX: Items routines
1 \ cursor functions, .items, .item-page
2 \ wpos: window positioning -xtrailing
3 \ write phrase
4 \ constants for file#'s, some fields
5 \ whichguy force current CREATURE attribute file
6 ( CASE STATEMENT BY CHARLES EAKER )
7 \ window definitions
8 \ window erase for the current window + cleanit
9 \ INSERT BLANK-OPTS
10 \ macros for menus
11 \ options for items activity:
12 \ final options menus
13 \ display options
14 \ options for items activity:
15 \ more options for tvehicle/tv-hold contents
16 \ display options
17 \ change-vol, ?volume
18 \ (extract)
19 \ lifeform valuation
20 \ lifeform valuation ?dist
21 \ conversion from a lifeform to a specimen or bio-
22 \ continue conversion from life to spec/bio-data
23 \ tvxy and xy!
24 \ convert a creature into a specimen
25 \ convert a creature into a specimen
26 \ pickup and drop-it
27 \ new>box: problem with multiple mineral deposits
28 \ ?deposited
29 \ drop-it
30 \ describe options for artifacts
31 \ do routines for specimens
32 \ start of options dispatcher: do.ele
33 \ options dispatcher: do.art
34 \ message action
35 \ ruin action
36 \ lifeform action
37 \ dispatcher
38 \ print scroll message at bottom of current window
39 \ this item movement
40 \ ?expand + ?contract window, reposition cursor
41 \ !!line!, set-inside : do function interface
42 \ prep? : setup window, present options and reposi
43 \ do.options ! needs refactoring
44 \ highlighting buttons
45 \ ?inside test for scrolling text up or down
46 \ scroll text up or down
47 \ item-loop , td-scroll
48 \ (?this) and try-this
49 \ try-these
50 \ find-item and gather
51 \ ?near + gather-items
52 \ return-items to this-region
53 \ ?tv-flat: use in ITEMS
54 \ (/items)

```

55 ( OVERLAY SUFFIX: ITEMS INTERFACE -----

0

3

```

0 \ OVERLAY PREFIX: Items routines          (rfg19sep85) \ write phrase          rfg06jun85)
1 9 width !
2 vocabulary items immediate
3 111 open-overlay
4 items definitions
5 2200 trans-allot
6 newt-dp
7
8
9
10
11
12
13
14
15

```

1

4

```

0 \ cursor functions, .items, .item-page    (rfg21jun85) \ constants for file#'s, some fields    rfg12jun85)
1 v: revision
2 v: ttlines \ total lines output in recent seesion
3 v: phrase-mem 399 allot
4 v: do-line#
5
6 head: (CRS-TOGGLE) ( y -- )
7 t: >R @CRS 3 R> POS. CURSORSPACE aBLT ! 7 1BLT ! 154 wBLT !
8 DK-BLUE !COLOR 1 XORMODE ! BLT !CRS xormode off t;
9
10 head: CRS-TOGGLE ( -- ) t: WTOP @ (CRS-TOGGLE) t;
11
12
13
14
15

```

2

5

```

0 \ wpos: window positioning -xtrailing    (rfg14jun85) \ whichguy force current CREATURE attribute file rfg14jun85)
1
2 head: wpos \ x,y ---
3 t: swap 4 * \ width of characters
4 wleft @ 2+ + \ horizontal offset
5 wtop @ 1- \ vertical offset
6 rot 7 * - \ width of characters
7 pos. t; \ position it
8
9 ascii . c: xtrail
10
11 head: -xtrailing
12 t: ' xtrail ! ' xtrail cfa ' -trailing 14 + !
13 -trailing ' bl cfa ' -trailing 14 + ! t;
14
15

```

```

0 \ whichguy force current CREATURE attribute file rfg14jun85)
1 transient trace @ >r trace off
2 : 20F 4 ?PAIRS COMPILE 2OVER COMPILE d= COMPILE
3 OBRANCH HERE 0 , COMPILE 2DROP 5 ; IMMEDIATE
4 resident r> trace !
5
6 head: whichguy \ force current attribute file to be referent
7 t: creature dup file# ! record# @ @record drop t;
8
9 head: reorg t: -5000 dup anchor 2! orglist t;
10
11 head: keypause \ maximal delay ---! wait for delay or key
12 t: time 2@ lkeytime 2!
13 begin dup 0 lkeytime 2@ d+ time 2@ d< 'key or
14 until drop t;
15

```

6

```

0 ( CASE STATEMENT BY CHARLES EAKER )
1 DECIMAL transient trace @ >r trace off
2 : !CSP SP@ CSP ! ;
3 : ECASE      ?COMP CSP @ !CSP 4 ; IMMEDIATE
4
5 : EOF        4 ?PAIRS COMPILE OVER COMPILE = COMPILE
6             0BRANCH HERE 0 , COMPILE DROP 5 ; IMMEDIATE
7
8 : ENDOF      5 ?PAIRS COMPILE BRANCH HERE 0 ,
9             SWAP 2 [COMPILE] THEN 4 ; IMMEDIATE
10
11 : ENDCASE    4 ?PAIRS COMPILE DROP BEGIN SP@
12             CSP @ = 0= WHILE 2 [COMPILE]
13             THEN REPEAT CSP ! ; IMMEDIATE
14 resident r> trace !
15

```

7

```

0 \ window definitions      rfg21aug85)
1
2 head: tvwindow \ terrain vehicle text
3 t: 64 3 7 38 window -1 wbottom +! t;
4
5 head: newscr \ not full screen, for doing an action
6 t: tvwindow -7 wtop +! -1 wlines +! t;
7
8 head: redisplay
9 t: >mainview .background
10 .local-icons v>display >display t;
11
12
13
14
15

```

8

```

0 \ window erase for the current window + cleanit rfg14jun85)
1
2 head: werase \ using current window
3 t: color @
4 wtop @ wleft @ wbottom @ 1- wright @ 1+
5 black poly-window-fill
6 0 0 wpos
7 !color t;
8
9 head: cleanit \ scroll up lines in TVTWINDOW
10 t: [ base @ hex ] 2001 pad ! \ 1 blank is packed string
11 ttlines @ wlines @ over - + \ total lines to clear window
12 0 do pad wline-up loop \ clear prior description
13 ttlines off [ base ! ] t; \ clear total line count
14
15

```

9

```

\ INSERT BLANK-OPTS      6nov84rd !
head: >= t: 2dup = >r >r> or t;

head: INSERT ( $ADDR,CNT --- )
t: PM-PTR @ PHRASE-MEM - TEXTC/L 1+ / \ CALC OFF INTO PM
TEXTC/L 1+ * PHRASE-MEM + \ CALC CURR LINE#
DUP C@ 3 PICK + TEXTC/L >= \ WILL EXCEED CURR LINE?
IF TEXTC/L 1+ + \ GO TO NEXT LINE
DUP 0 SWAP C! \ SET THAT LINE CNT=0
DUP 1+ PM-PTR ! 1 LINE-COUNT +! \ SET NEW PM PTR
THEN
DUP C@ 3 PICK + SWAP C! \ UPDATE LINE CNT
DUP >R PM-PTR @ SWAP CMOVE \ MOVE INTO PHRASE MEM
R> PM-PTR +! t; \ UPDATE PM PTR

```

10

```

\ macros for menus

head: .pick t: ." PICKUP" t;
head: .drop t: ." DROP" t;
head: .desc t: ." DESCRIBE" t;
head: .eras t: ." ERASE" t;
head: .reco t: ." RECORD" t;

head: lpos t: 0 swap wpos t;
\ n --- position at left, line n in tvwindow

v: #options #options off
v: option# \ use to index into case for doing the option
head: #options! t: #options ! t;

```

11

```

\ options for items activity:      rfg07may85
\ assume color preset, in tvwindow, but do their own positionin
head: te.opts \ options for elements
t: 1 dup lpos .drop #options! t;

head: ta.opts \ options for artifacts
t: 1 lpos .drop
2 dup lpos .desc #options! t;

head: tm.opts \ options for messages
t: 1 lpos ." READ"
2 dup lpos .eras #options! t;

```

12

```

0 \ final options menus
1
2 head: tl.opts \ options for lifeforms, inside hold
3 t: 1 dup lpos .desc #options! t; \ does SAYIT (same as LOOK...)
4
5 head: ts.opts \ options for specimens , on land or in hold??
6 t: 1 dup lpos .drop
7 2 lpos .desc
8 3 dup lpos .eras #options! t;
9
10 head: tb.opts \ options for bio-data, prerecorded
11 t: 1 lpos .desc
12 2 dup lpos .eras #options! t;
13
14
15

```

13

```

0 \ display options
1 head: opt.err t: delete-scroll-box
2 abort" error in display.options" t;
3
4 case t.options
5 26 is te.opts
6 28 is ta.opts
7 27 is tm.opts
8 30 is tl.opts
9 68 is tl.opts
10 40 is ts.opts
11 43 is tb.opts
12 11 is exit ( until we figure it out )
13 others opt.err
14
15

```

14

```

0 \ options for items activity;
1 \ assume color preset, in tvwindow, but do their own positionin
2 head: pe.opts \ options for elements
3 t: 1 dup lpos .pick #options! t;
4
5 head: pa.opts \ options for artifacts
6 t: 1 lpos .pick
7 2 dup lpos .desc #options! t;
8
9 head: pm.opts \ options for messages
10 t: 1 lpos .reco
11 2 dup lpos ." READ" #options! t;
12
13
14
15

```

15

```

\ more options for tvehicle/tv-hold contents
head: pr.opts \ options for ruins
t: 1 lpos ." DESCRIBE"
2 lpos ." ENTER"
3 dup lpos ." DESTROY" #options! t;

head: pl.opts \ options for lifeforms, on planet surface
t: hits c@
if 1 lpos ." CAPTURE"
else 1 lpos .pick
then 2 lpos .reco ." BIO-DATA" \ make bio-data object, in
3 dup lpos .desc #options! t; \ does SAYIT (same as LOOK..)

head: ps.opts \ options for specimens , on land or in hold??
t: 1 dup lpos .pick #options! t;

```

16

```

\ display options
case p.options
26 is pe.opts
28 is pa.opts
27 is pm.opts
30 is pl.opts
68 is pl.opts
40 is ps.opts
41 is pr.opts
11 is exit ( until we figure it out )
others opt.err

```

17

```

\ change-vol, ?volume
head: volume+! \ volume ---
t: tv-hold 1.5@ >c+s
content-vol +! iclose t;

head: ?volume \ --- volume, 1 or 0, 0 ! error if won't fit
t: @inst-class 26 = not
if inst-qty @ else elem-amt @ then dup
tv-hold 1.5@ >c+s
content-vol @ + inst-qty @ >
if drop NULL tvwindow werase revision off
." NOT ENOUGH ROOM IN CARGO HOLD " 1000 keypause
else 1
then iclose t;

```



18

```

0 \ (extract)
1 \ remove from a box within a box
2 : (extract) \ boxbox --- iaddr { item -- boxbox } !
3 ci 2swap @inst-class >r iclose \ class of item to remove
4 ( boxBOX ) >c+s iopen r> box>tocs
5 IOPEN CDROP \ ?first bug
6 >c+s (box>)
7 ?null iclose
8 if idelete then
9 iclose revision on ;
10 \ boxbox is tv-hold or surface
11 head: plextract t: (surface) 1.5@ (extract) t;
12 head: tvextract t: tv-hold 1.5@ (extract) t;
13
14
15

```

19

```

0 \ lifeform valuation
1 68 12 1 afield: niche
2 68 54 1 afield: i.level
3 68 11 1 afield: size.index
4
5 head: niche>val
6 t: niche c@ dup
7 4 mod 0= if drop 10
8 else 2 mod 0=
9 if 30 else 50 then
10 then t;
11
12
13
14
15

```

20

```

0 \ lifeform valuation ?dist
1
2 head: ?dist \ --- distance
3 t: (system) 1.5@ >c+s
4 inst-x @ 125 - dup * 0
5 inst-y @ 100 - dup * 0
6 d+ sqrt iclose t;
7
8 head: valuate \ --- monetary value
9 t: niche>val i.level c@ 10 / +
10 size.index c@ *
11 ?dist 2 * 1+ *
12 hits c@ 0= if 10 / then t; \ dead critters are worth less
13
14
15

```

21

```

rfg14jun85) \ conversion from a lifeform to a specimen or bio-rfg21aug85)
head: l$= \ count --- count,flag ! life string at pad
t: 1 spec-name 3 pick 0
do dup i + c@ pad i + c@ = not
if swap 0= swap leave then loop drop t;

head: info-setup
t: 0 5 wpos lt-blue !color t;

```

22

```

rfg12jun85) \ continue conversion from life to spec/bio-data rfg13may85)
head: $setup \ --- count ici is a lifeform
t: resembles 1.5@ 2dup or 0=
if 2drop shape 1.5@ then
>c+s phrase pad phr-cnt c@ dup >r cmove
r> iclose t;

head: $match \ --- record# !of specimen corresponding to crit
t: $setup 40 file# !
64 1 do i record# ! l$=
if i leave then loop swap drop t;
\ assumes always successfully completed

```

23

```

rfg22may85) \ tvxy and xy! (rfg06sep85)
head: tvxy \ --- x,y of terrain vehicle
t: tvehicle 1.5@ >c+s
inst-x @ inst-y @ iclose t;

head: xy! \ x,y --- ! store into current object's x,y fields
t: inst-y ! inst-x ! t;

head: .cargo
t: 116 157 pos. 3 black poly-erase-text
tv-hold 1.5@ >c+s
content-vol @ dup if 5 max 5 / then
white !color 3 .r ." % FULL"
iclose t;

```

24

```

0 \ convert a creature into a specimen ~ rfg10aug85)
1
2 head: ?>spec \ --- value,species,volume,flag ! true if tvroom
3 t: valuate \ get monetary value
4 $match \ get specimen/bio-data species
5 ?volume dup
6 if plextract >c+s
7 else 0. >c+s
8 then swap t;
9
10
11 head: ?attack \ --- flag ! true if attacking+succes or not attack
12 t: behave c@ 1 and hits c@ 0= 0= and
13 if 0 100 rrnd 10 <
14 else 1
15 then t;

```

25

```

0 \ convert a creature into a specimen rfg21aug85)
1 head: crit>spec
2 t: tv>window -7 wtop +! werase ?attack
3 if ?>spec >r
4 if -icon redisplay do-line# @ 1 = \ bio-data or specimen
5 if 40 info-setup \ phrase below, for specimens only
6 ." LIFEFORM CAPTURED AND PUT IN STASIS "
7 else swap 08 / swap 43
8 >r drop 0 >r \ bio-data has no volume
9 then swap 1create >c+s
10 inst-val ! tvxy xy! R@ INST-QTY !
11 c> tv-hold 1.5@ >c+s >box cdrop
12 else 2drop
13 then 1close >r volume+!
14 else info-setup ." THE CAPTURE ATTEMPT WAS UNSUCCESSFUL "
15 then 7 wtop +! .cargo 1000 keypause t;

```

26

```

0 \ pickup and drop-it rfg14jun85)
1 head: pickup \ { n --- n } ! put object n into (tv-hold)
2 t: ?volume
3 if plextract
4 >c+s -icon \ reorg
5 c> tv-hold 1.5@ >c+s >box 1close
6 then volume+! .cargo t;
7
8
9 head: delete.it \ from the terrain vehicle hold or land
10 t: -icon reorg 1delete revision on t;
11
12 head: pdelete t: plextract -icon 2drop revision off t;
13 head: tvdelete t: tvextract -icon 2drop t;
14
15

```

27

```

\ new>box: problem with multiple mineral deposits rfg01jul85)
head: new>box \ as in >box
t: 2dup >c+s @inst-class >r 1close
iopen 11 i class>box-spec ifind 0=
if i class>box-spec 11 swap 1 *create \ make-class-box
then >r >box>tocs
2dup >c+s @inst-class @inst-species 1close
2drop ( redundant!) ci 1insert 1close t;

```

28

```

\ ?deposited rfg01jul85)
2v: clsp
head: ?deposited \ class,species --- tflag! true if cl,sp minera
t: clsp 2! tvxy ilocal @ ?icons-at ?dup
if 0 swap 0 do dup 0=
if swap point>icon @il @ih >c+s
>r @inst-class @inst-species clsp 2@ d= >r swap
if drop 1
then 1close
else swap drop
then loop
then ( >r 2drop >r ) t;

```

29

```

\ drop-it rfg05jul85)
head: dropit \ --- \ put on planet surface
t: tvextract
>c+s
inst-qty @ negate volume+! .cargo
@inst-class @inst-species ?deposited
tvxy xy! \ position same as vehicle
' icon-param module +icon reorg
c> (surface) 1.5@ >c+s
rot if >box else new>box then
1close t;

```

30

```

0 \ describe options for artifacts
1 \ assume the artifact is TOCS
2 artifact 29 1 afield: art-anlyz
3 artifact 24 1 afield: art-text
4
5 head: arttalk \ addr --- ! print analysis, given string address
6 t: 5 0 do dup i 38 * + 38 type gcr loop drop t;
7
8 head: art.desc
9 t: werase art-anlyz c@
10 if analyze-text art-text c@ @record
11 arttalk set-current \ restore object, pause
12 else gcr ." THIS ITEM NOT ANALYZED"
13 then 3000 keypause t;
14
15

```

rfg14jun85)

33

```

\ options dispatcher: do.art
rfg10aug85)

case pa.act \ artifact on planet surface
1 is pickup 2 is art.desc others nop
case ta.act
1 is dropit 2 is art.desc others nop

head: do.art
t: inside @ if ta.act else pa.act then t;

```

31

```

0 \ do routines for specimens
1 v: inside
2 head: .spec
3 t: newscr werase
4 spec-name 16 ascii . -xtrailing type ." SPECIMEN"
5 1500 keypause t;
6
7 case ts.act
8 1 is dropit 2 is .spec 3 is tvdelete others nop
9
10 case ps.act
11 1 is pickup others nop
12
13 head: do.spec
14 t: inside @ if ts.act else ps.act then t;
15

```

rfg10aug85)

34

```

\ message action
rfg14jun85)
27 24 3 ifield: inst-text
: read.msg \ writes it to display
crs-toggle newscr cleanit
1 line-count ! 0 phrase-mem dup 1+ pm-ptr ! c!
inst-text 1.5@ >c+s
phrase phr-cnt c@ insert write-phrase iclose 3000 keypause ;

case pm.act
1 is pickup 2 is read.msg others nop
case tm.act
1 is read.msg 2 is tvdelete others nop

head: do.msg
t: inside @ if tm.act else pm.act then t;

```

32

```

0 \ start of options dispatcher: do.ele
1
2 head: set-elem-amt
3 t: inst-qty @ dup 11 < not
4 if 2/ then elem-amt ! t;
5
6 head: elem-drop
7 t: elem-amt @ set-elem-amt dropit elem-amt ! t;
8
9 head: elem-pick
10 t: elem-amt @ set-elem-amt
11 pickup elem-amt ! t;
12
13 head: do.ele \ do something with an element
14 t: drop inside @ if elem-drop else elem-pick then t;
15

```

rfg10aug85)

35

```

\ ruin action
rfg14jun85)
\ : r.desc 0 4 wpos ." THIS IS A RUIN." ;

\ head: r.enter \ WOT A MESS!
\ t: inst-x @ inst-y @
\ tvehicle 1.5@ >c+s xy!
\ ci ?icon=iaddr drop point>icon
\ inst-x @ dup !ix iclose t;

\ case do.ruin
\ 1 is r.desc 2 is beep ( r.enter) 3 is pdelete others nop

head: do.esc \ for incorrect cases, such as boxes
t: drop fquit on t;

```

36

```

0 \ lifeform action
1 head: do-say
2   t: crs-toggle 14 wtop +! werase
3     6 ttlines ! @inst-class ' sayit module
4     3000 keypause t;
5
6 case pl.act
7   1 is crit>spec 2 is crit>spec 3 is do-say  others nop
8
9 head: do.life
10 t: inside @ if drop do-say  else pl.act then t;
11
12 case do.bio
13   1 is beep 2 is tvdelete others nop
14
15

```

37

```

0 \ dispatcher
1
2 case dispatch
3 11 is do.esc
4 \ 41 is do.ruin
5 68 is do.life
6 27 is do.msg
7 26 is do.ele
8 28 is do.art
9 40 is do.spec
10 43 is do.bio
11 others do.esc
12
13
14
15

```

38

```

0 \ print scroll message at bottom of current window rfg01jul85)
1
2 head: .bottom \ --- ! position and print
3 t: 6 6 wpos color @ lt-blue !color
4   ." SCROLL ITEMS: ^\  QUIT: ]"
5   7 wbottom +! -1 wlines +!
6     !color t;
7
8 head: tvrevise
9 t: revision @
10 if -7 wtop +! werase 7 wtop +! \ preserve object of action
11 >mainview .background .local-icons v>display >display
12 lt-blue !color 0 6 wpos ." REVISING TERRAIN VEHICLE INVENTORY"
13 revision off white !color
14 then t;
15

```

39

```

\ this item movement rfg12jun85)
head: THIS-ITEM ( -- iaddr ) t: (THIS-ITEM) 1.5@ t;
head: NEXT-THIS-ITEM ( -- )
t: THIS-ITEM >c+s INEXT CI (THIS-ITEM) 1.5! ICLOSE t;
head: PREV-THIS-ITEM ( -- )
t: THIS-ITEM >c+s IPREV CI (THIS-ITEM) 1.5! ICLOSE t;
head: GET-THIS-INST ( ( -- n ) )
t: THIS-ITEM >c+s
TEXT-INST 1.5@ ICLOSE >c+s t;

```

40

```

\ ?expand + ?contract window, reposition cursor rfg15apr85
head: reposition
t: crs-toggle -7 wtop +! crs-toggle t;
head: ?contract \ contract textwindow if too few lines to fill
t: scroll-len @ 6 - dup 0< \ 6 lines really used in tvwindow
if abs 0 do black wshorten loop
else drop
then t;
head: full-text \ expand tvwindow to full size
t: [ tvwindow wbottom @ wlines @ ] literal literal
wbottom ! wlines ! t;

```

41

```

\ !!line!, set-inside : do function interface (rfg19sep85)
head: !!line! \ n --- ! increment do-line#, with bounds
t: crs-toggle do-line# +! do-line# dup
@ 1 max #options @ min swap !
tvwindow do-line# @ 7 * negate wtop +! crs-toggle t;
head: set-inside
t: super-box 1.5@ >c+s iopen
ci (surface) 1.5@ d=
if 0 else -1 then inside ! iclose iclose t;
head: do.revision
t: tvwindow full-text tvrevise delete-scroll-box
make-scroll-box super-box 1.5@ get-boxes set-inside
?contract t;

```



42

```

0 \ prep? : setup window, present options and reposirfg14jun85)
1 head: prep? \ { --- n } ! iaddr of this item
2 t: -7 wtop +! werase 7 wtop +!
3 white !color
4 get-this-inst @inst-class
5 inside @ if t.options else p.options then
6 reposition t;
7
8 head: .ITEMS { { n -- n' } --- } \ xy position prompt
9 t: .bottom WLINEs @ 0 DO
10 WLEFT @ WTOP @ 1- I 7 * - POS. TEXT>PAD PAD $. INEXT LOOP t;
11
12 head: .ITEM-PAGE ( --- )
13 t: (SCROLL-BOX) 1.5@ >c+s IOPEN CI
14 (this-item) 1.5! 0 BEGIN 1+ INEXT ?FIRST UNTIL
15 >IFONT .ITEMS IPREV CRS-TOGGLE drop t;

```

43

```

0 \ do.options ! needs refactoring rfg14jun85)
1 head: do.options \ {scrollbox,scrolltext --- same }
2 t: 1 do-line# ! revision off fquit off
3 begin ?quit not while xyscan
4 ecase 1 0 ( U0) 2of -1 !!line! endof
5 -1 0 ( DZ) 2of 1 !!line! endof
6 drop ?trig
7 if do-line# @ @inst-class dispatch
8 revision @
9 if do.revision
10 then fquit on
11 then endcase
12 repeat cdrop cdrop iclose \ close scrbox,scrltext,inst
13 tvtwindow werase ?contract white !color .item-page t;
14
15

```

44

```

0 \ highlighting buttons
1 head: svcolor t: color @ swap !color t;
2
3 head: dnbtn
4 t: svcolor black svcolor
5 85 21 pos. ." " !color 85 21 pos. ." ^" !color t;
6
7 head: upbtn
8 t: svcolor black svcolor
9 91 21 pos. ." " !color 91 21 pos. ." \ " !color t;
10
11 head: qtbtn
12 t: svcolor black svcolor
13 133 21 pos. ." " !color 133 21 pos. ." ]" !color t;
14
15 \ actually, not the buttons, merely text, but looks nice...

```

45

```

\ ?inside test for scrolling text up or down rfg15may85)
head: (?inside) \ --- -1 if ci is SURFACE, 1 if TV-HOLD, 0 other
t: get-this-inst @inst-class ll =
  if @inst-species dup 40 =
    if drop 1 else 44 =
      if -1 else 0 then then
    else 0 then iclose t;

head: inside! \ flag, sets inside accordingly
t: dup 0> if -1 inside ! then
  0< if 0 inside ! then t;

head: up-inside t: (?inside) inside! t;
head: down-inside t: (?inside) negate inside! t;

```

46

```

\ scroll text up or down rfg14jun85)
head: TD-UP ( -- )
t: green dnbtn INEXT NEXT-THIS-ITEM up-inside
TEXT>PAD PAD CRS-TOGGLE white !color
-1 wtop +! WLINE-UP 1 wtop +!
CRS-TOGGLE lt-blue dnbtn t;

head: TD-DOWN ( -- )
t: green upbtn down-inside IPREV PREV-THIS-ITEM
THIS-ITEM >c+s TEXT>PAD ICLOSE white !color
PAD CRS-TOGGLE -1 wtop +! WLINE-DOWN 1 wtop +!
CRS-TOGGLE lt-blue upbtn t;

```

47

```

\ item-loop , td-scroll rfg01jul85)
CASE TD-SCROLL
  1 IS TD-UP
  -1 IS TD-DOWN
  others nop

: (item-loop)
  BEGIN xyscan 0=
    if td-scroll
    else drop
    then ?trig
    if prep? fquit off do.options
    then fquit @
  UNTIL ;

```

48

```

0 \ (?this) and try-this
1 head: (?this) \ { iaddr --- iaddr }
2   \ aboxiaddr --- 0. or abox if within abox
3 t:  \ debugger break
4   NULL 2swap >c+s iopen
5 ?null not   \ handle empty box case
6 if begin 2dup 0= \ found one
7   ?last 0= and
8   while cj ci d= \ current is target iaddr
9     if 2drop ci' \ change flag. to exit
10    then inext
11    repeat cj ci d= if 2drop ci' then \ cover last one
12  then cdrop iclose t;
13
14
15

```

49

```

0 \ try-these
1
2 head: try-these \ { iaddr --- iaddr } --- 0. or container
3 t: 0. \ initial failure flag
4   begin 2dup or 0= ?last 0= and
5   while 11 0 ifind   \ any old box
6     if ci cj >c+s
7       (?this) iclose 2dup or \ found a match
8       if 2swap then 2drop
9       then inext
10  repeat t;
11
12 head: ?delete \ {box,item --- another box,item} delete if empty
13 t: ?null if iclose idelete iprev 0. >c+s then t;
14
15

```

50

```

0 \ find-item and gather
1
2 head: find-item \ { iaddr --- box,iaddr } searches planet tree
3 t: this-region 1.5@ (?this) 2dup or 0= \ not locally ?
4 if 2drop
5   (planet) 1.5@ >c+s
6   iopen try-these cdrop iclose
7 then c> 2swap >c+s iopen cdrop
8   >c+s t;
9
10 head: gather \ {box,item --- box,item or inext} moved to (SURFAC
11 t: iextract
12 (surface) 1.5@ >c+s new>box
13   cj >c+s
14   -1 inst-qty +! iclose t; \ decrement count in box
15

```

51

```

rfg12jun85) \ ?near + gather-items rfg12aug85)
head: ?near \ { something --- something } --- true if within
t: tvxy inst-y @ - abs 2 < swap
   inst-x @ - abs 2 < and t;

: gather-items \ moves from THIS-REGION to (SURFACE)
32000 elem-ant !
ilocal @ 0 do
  i point>icon
  @id dup >r 17 < r@ 20 23 within or r> 254 = or
  if @il @ih >c+s ?near
    if find-item gather cdrop iclose
    then iclose
  then loop 10 elem-ant ! ;
\ handles creatures, invisible, avoids flying, ruins

```

52

```

rfg06jun85) \ return-items to this-region rfg10aug85)
head: (return) \ {surface,box,item --- surface,box,inext}
t: (box>) this-region 1.5@ iinsert t;

head: (return-items) \ do return for items in box
t: iopen begin ?null not
   while (return)
     repeat iclose idelete iprev t;

: return-items
32000 elem-ant !
(surface) 1.5@ >c+s
iopen ' (return-items) all cdrop iclose
10 elem-ant ! ;

```

53

```

rfg29jun85) \ ?tv-flat: use in ITEMS rfg12jun85)
head: flat-scan \ {container,box --- same}
t: 11 15 ifind
   if iopen 28 5 ifind iclose
   else 0
   then ?flat ! t;

head: ?tv-flat \ resets ?flat if not on tv anymore.
t: tv-hold 1.5@ >c+s iopen
   flat-scan cdrop iclose t;

```

54

```
0 \ (/items)                                rfg14aug85)
1
2 : (/ITEMS) ( box-in-box,box-variable -- )
3 begin (scroll-box) 1.5@ D0=
4 if make-scroll-box gather-items
5   super-box 1.5@ get-boxes
6 then tvtwindow werase xormode off
7 white !color ?contract white !color .item-page
8 set-inside
9 (item-loop) ?quit until
10 green qbtn delete-scroll-box NULL (scroll-box) 1.5!
11 return-items cdrop iclose full-text
12 7 ttlines ! lt-blue qbtn crs-toggle
13 ?tv-flat ; \ is flat device in TV- modify flag
14
15
```

55

```
0 ( OVERLAY SUFFIX: ITEMS INTERFACE -----rfg19aug85)
1 trace @ trace off  dispose trace !
2 close-overlay
3 !!! overlay items-ov
4 items-ov
5 forth definitions
6 : /items items-ov items (/items) ;
7
8 save-buffers ov-cancel
9 31 width !
10 \ freeze
11
12
13
14
15
```