Original Keying.rkt

```
#lang racket/gui
(provide keying)
(require "sv.rkt" "av.rkt" "move.rkt"
         "key-peg.rkt")

(define (keying w k)
  "( w k -- w) Process key events."

  (case k
    [(" " "\r") (sv w 'message #f)]

    [("m") (sv w 'map (not (av w 'map)))]

    [("8")  (writeln w) w]                          ; Write out current state.
    [("9")  (sv w 'dbg-trap (not (av w 'dbg-trap)))] ; Show traps.
    [("0") (sv w 'dbg-tile (not (av w 'dbg-tile)))] ; Tile identification mode.
    [("1") (sv w 'grid (not (av w 'grid)))]         ; Display tile grid overlay.
    [("2") (sv w 'debug (not (av w 'debug)))]       ; Display current state.

    [("right") (world-move w 1 0)]
    [("left")  (world-move w -1 0)]
    [("up")    (world-move w 0 -1)]
    [("down")  (world-move w 0 1)]

    [("wheel-up") (world-scroll w 1)]
    [("wheel-down") (world-scroll w -1)]

    [else w]
    )
  )
```

Goal Keying.rkt Using PEG

```
#lang racket/gui
(provide keying)
(require "sv.rkt" "av.rkt" “move.rkt" "key-peg.rkt")
(peg pkeying “
  space : (sv w 'message #f)
  return : (sv w 'message #f)
  m:  (sv w 'map (not (av w 'map)))
  f8: [(writeln w) w]                                ; Write out current state.
  f9: (sv w 'dbg-trap (not (av w 'dbg-trap)))) ; Show traps.
  f10: (sv w 'dbg-tile (not (av w 'dbg-tile)))) ; Tile identification mode.
  f11: (sv w 'grid (not (av w 'grid)))               ; Display tile grid overlay.
  f12: (sv w 'debug (not (av w 'debug)))      ; Display current state.
  right: (world-move w 1 0)
  left:  (world-move w -1 0)
  up:    (world-move w 0 -1)
  down: (world-move w 0 1)
  wheel-up: (world-scroll w 1)
  wheel-down: (world-scroll w -1)
“
)
```

Not a major difference, but enough to test PEG-ification
(if successful)