floor = math.floor ceil = math.ceil

Maze = Maze.$_{index=MazeMaze.NORTH=90Maze.SOUTH=270Maze.EAST=0Maze.WEST=180}$

Maze.new = function(r, c) local i, j, row, cell local obj = setmetatable(obj, Maze) obj.rows = r obj.cols = c obj:reset() return obj end

Maze.reset = function(self) local i, j, row, cell self.cells = for i = 1,self.rows do row = for j = 1,self.cols do cell = visited=false, visitable=true, north=true,south=true,east=true, west=true, meta=" table.insert(row, cell) end table.insert(self.cells, row) end end

Maze.visitable = function(self, i, j) return self.cells[i] and self.cells[i][j] and self.cells[i][j].visitable end

Maze.isVisitable = function(self, i, j) return self.cells[i] and self.cells[i][j] and self.cells[i][j].visitable end

Maze.setVisitable = function(self, i, j, visitable) self.cells[i][j].visitable = visitable self.cells[i][j].north = visitable self.cells[i][j].south = visitable self.cells[i][j].east = visitable self.cells[i][j].west = visitable end

Maze.getMeta = function(self, i, j) return self.cells[i][j].meta end

Maze.setMeta = function(self, i, j, meta) self.cells[i][j].meta = meta end

Maze.create = function(self, start) local n, N, x, y, a, b, k, stack N = self.rows * self.cols for i = 1,self.rows do for j = 1,self.cols do if not self.cells[i][j].visitable then N = N - 1 end end end

start = start or x=1, y=1 x, y = start.x, start.y stack = table.insert(stack, x=x, y=y) self.cells[y][x].visited = true visited = 1 while visited ¡ N do x = stack[stack].x y = stack[stack].y

n = if self:isVisitable(y+1, x) and not self.cells[y+1][x].visited then table.insert(n, Maze.NORTH) end if self:isVisitable(y-1, x) and not self.cells[y-1][x].visited then table.insert(n, Maze.SOUTH) end if self:isVisitable(y, x+1) and not self.cells[y][x+1].visited then table.insert(n, Maze.EAST) end if self:isVisitable(y, x-1) and not self.cells[y][x-1].visited then table.insert(n, Maze.WEST) end

if n == 0 then table.remove(stack) else k = n[math.random(n)] if k == Maze.NORTH then a, b = x, y+1 self.cells[y][x].north = false self.cells[b][a].south = false elseif k == Maze.SOUTH then a, b = x, y-1 self.cells[y][x].south = false self.cells[b][a].north = false elseif k == Maze.EAST then a, b = x+1, y self.cells[y][x].east = false self.cells[b][a].west = false elseif k == Maze.WEST then a, b = x-1, y self.cells[y][x].west = false self.cells[b][a].east = false end table.insert(stack, x=a, y=b) self.cells[b][a].visited = true visited = visited + 1 end end end

m = Maze.new(70,55) for i = 1,m.rows do for j = 1,m.cols do m:setVisitable(i, j, false) end end

– tree tree$_data$ = $1, 1, 1, 3, 3, 3, 5, 5, 7, 7, 9, 11, 11, 13, 15, 17, 19, 23, 27, 21, 13, 15, 17, 17, 19, 21, 23, 27, 29, 33, 25, 19, 21, 21, 23, 25, 27, 29, 33, 35, 1, xdom : setVisitable(m.rows - i - 5, floor((m.cols - x)/2) + j, true)m : setMeta(m.rows - i - 5, floor((m.cols - x)/2) + j,' green')endend$

– decorations c = floor(m.cols/2) colors = 'red', 'orange', 'purple' ncolors = 3 for q = 0,3 do for j = c-15, c+15 do if j ¡ c then k = 3 else k = 1.625 end for r = -1,1 do m:setMeta(floor((j - c) / k) + 15 + q * 10 + r, j, colors[math.random(ncolors)]) end end end

– star star$_data$ = $1, 1, 1, 3, 3, 5, 9, 15, 9, 5, 3, 3, 1, 1, 1 for i, xinipairs(star_data) do for j = 1, xdom : setVisitable(m.rows - i + 1, floor((m.cols - x)/2) + j, true)m : setMeta(m.rows - i + 1, floor((m.cols - x)/2) + j,' yellow')endend$

– trunk trunk$_data$ = $7, 7, 9, 11 for i, xinipairs(trunk_data) do for j = 1, xdom : setVisitable(m.rows−i−61, floor((m.cols−x)/2)+j, true)m : setMeta(m.rows− i − 61, floor((m.cols - x)/2) + j,' brown')endend$

– snow for i = 1,5 do for j = 1, m.cols do m:setVisitable(i,j, true) m:setMeta(i,j, 'white') end end

m:create(x=1, y=1) m.cells[1][1].west = false m.cells[m.rows][ceil(m.cols/2)].north = false