THE Merry Squid

ScriptCraft Drone Quick Reference

Movement

- `up(numBlocks)`
- `down(numBlocks)`
- `left(numBlocks)`
- `right(numBlocks)`
- `fwd(numBlocks)`
- `back(numBlocks)`
- `turn(numTurns)`
  - `numTurns`: number of 90 degree turns to take, clockwise

Markers

- `chkpt(name)`
  - saves the current location as a marker called `name`
- `move(name)`
  - moves to the location saved in the marker called `name`

Others

- `times(numTimes)`
  - repeats the preceding commands `numTimes` (defaults to 2)
- `copy(name, width, height, depth)`
  - copies part of the world to paste elsewhere
- `paste(name)`
  - pastes a previously copied area

Drawing

- `box(block, width, height, depth)`
  - `box0` will draw a hollow one
- `boxa(blockArray, width, height, depth)`
  - constructs a box by cycling through the blocks in `blockArray`
- `prism(block, width, length)`
  - draws a prism, which is a good shape for a roof
  - the height will be half the `length`
  - `prism0` will make a hollow prism
- `cylinder(block, radius, height)`
  - `cylinder0` will make a hollow one
- `blocktype(message, foregroundBlock, backgroundBlock)`
  - writes `message` using `foregroundBlock` filling in gaps with `backgroundBlock`
- `sphere(blockType, radius)`
  - use `sphere0` for a hollow sphere
- `hemisphere(block, radius, northSouth)`
  - draw a hemisphere
  - `northSouth` should be "`north`" or "`south`"
  - `hemisphere0` for a hollow one
- `arc({properties})`
  - The properties object can include a variety of options. `radius` and `blockType` are required
  - `radius` is the radius of the arc
  - `blockType` is the block ID
  - `orientation` (default: "`horizontal`") specifies the orientation of the arc (either "`vertical`" or "`horizontal`")
  - `stack` (default: 1) the height or length of the arc
  - `strokeWidth` (default: 1) how many blocks wide the arc is
  - `fill` if `true` fill in the arc
  - `quadrants` specifies which of the 4 quadrants of a circle to draw (the default is to draw all). Pass an object with `topleft`, `topright`, `bottomleft`, `bottomright` set to true for each quadrant to draw.
- `rand(blocks, width, height, depth)`
  - creates a random box of blocks
  - `blocks` can be an array of blocks that have an equal chance of being chosen
  - `blocks` can also be an object where the key is a block and the value is the "weight" of the block being chosen (a number)

THE Merry Squid

ScriptCraft Drone Quick Reference

Things

- `door(doorType)`
  - create a door of `doorType` (default is wood)
  - `door_iron` is a shortcut for an iron door
  - `door2` will create a double door
  - `door2_iron` for iron double doors
- `bed()`
  - yep. it places a bed.
- `ladder(height)`
  - creates a ladder `height` blocks tall
- `wallsign(message)`
  - draws a sign on the wall
  - `message` can be a string or array with string for each line
- `signpost(message)`
  - freestanding sign (see `wallsign`)
- `spiral_stairs(stairBlock, flights)`
  - `stairBlock` should be a string (in quotes) from the following list: `oak, spruce, birch, jungle, cobblestone, brick, stone, nether, sandstone, quartz`
- `stairs(block, width, height)`
  - draw a flight of stairs of type `block` (should be a block type, but must be one of the ones listed for `spiral_stairs`)
- `oak, spruce, birch, jungle`
  - create a tree
- `fort(side, height)`
  - create a fort with the given dimensions
- `castle(side, height)`
  - creates a castle built of 5 forts
- `cottage()`
  - builds a cozy little home
- `temple(side)`
  - builds a step-style temple with a base of `side` length per side
- `chessboard(whiteblock, blackblock, width, depth)`
  - create a chessboard-style grid of alternating blocks
- `dancefloor(width, depth)`
  - builds a glowstone and glass dance floor that flashes briefly
- `firework()`
  - launches a firework into the air
- `garden(width, depth)`
  - create a rectangle of random grass and flowers
- `maze(width, depth)`

Directions of Things

- `Drone.PLAYER_STAIRS_FACING`
  - Used to make stairs face the player
  - For oak stairs facing the player, use this as the block type: `blocks.stairs.oak + ":" + Drone.PLAYER_STAIRS_FACING[d.dir]` where `d` is a drone object
- `Drone.PLAYER_SIGN_FACING`
  - Used to place signs, chests, ladders, furnaces and dispensers facing toward the player
  - Used in the same way as `PLAYER_STAIRS_FACING`
- `Drone.PLAYER_TORCH_FACING`
  - Place a torch facing the player rather than just facing up
- If you want to place something facing *away* from the player, you'd use `Drone.PLAYER_STAIRS_FACING[(d.dir + 2) % 4]`