--:! {net: [] (:smiley:} <- **Net Remote Command Library** -> muse/docs/lib/net.md
--:neutral_face: net: *Command Line Library providing turtle operations used by remote library (effectively the UI for Muse).* -> net

--:# *Remote commands for turtles need prefixing by `farmer`, `miner`, `logger`,`porter`, or `rover` roles.*

--:# **Dispatcher `op` catches net operations that raise exceptions so turtles report errors, ready for new commandsa.**

--:# **Testing, Monitoring, and Control** (e.g., `farmer echo something something`)

--:- status level [filename] -> *Set reporting hurdle and optionally save reporting in log file.*

--:- echo arguments ... -> *For testing: just returns its arguments.*

--:- quit message -> *Set `quit` flag to message; next `core.status` throws `error` to abort operations.*

--:# **Remote Turtle and Task Operations** (for turtle API operations, e.g., `rover find`)

--:- fueling -> *Returns energy available in turtle slots.*

--:- items -> *Returns items in turtle inventory as string.*

--:- find name...? -> *Report and select first slot found [or if no name, just report inventory].*

--:< **Directions are *up, down, north, east, west, south, forward***

--:- suck direction quantity? -> *Suck quantity items [or all] into available slot.*

--:- drop item direction quantity? -> *Drop quantity of selected items [or all].*

--:- look direction -> *Detect and inspect direction, return report.*

--:- compare item direction... -> *Named item matches block in any of specified directions?*

--:- attack direction -> *Attempts attack in specified direction.*

--:- dig direction distance hoeing... -> *Direction and distance to (possibly blocked) move, hoeings directions to hoe.*

--:- put filling direction distance putting... -> *Direction, distance to move, placing filling in puttings directions.*

--:- change target filling direction distance putting... -> *Move distance in direction replacing target with filling.*

--:# **Remote Turtle Motion Commands** (e.g., `farmer come`)

--:- to place | x y z face?-> *To named place or position and face. Retry for different first direction.*

--:- go *(first letter of) directions followed by optional counts, e.g.* `r 10 u east 3 u 4 d n`. -> *Chained movement.*

--:- trace trailname -> *Move turtle along traced situations in named trail from one end of trail to the other.*

--:- come -> *rover turtle towards GPS player position.*

--:- tail rate? -> *Move `rover` every rate (default 5) seconds towards GPS player position.*

--:< **Places - Points, Locations, Trails, and Ranges of Maps**

--:- trail name label -> *Include named point at head and (current situation) tail of a new trail, update map.*

--:- point name label trail? -> *Add named labeled point, can start trail, MU updated map. (Player situation needs GPS.)*

--:- range name label point point key? value?? -> *Volume by named points, optional key and value for feature.*

--:- chart filename ... -> *Loads and runs named file in `charts` directory to create named point and associated ranges.*

--:< **Navigation in Maps: Where Are We, What's Nearby, and Where Are We Heading?**

--:- at -> *Report current (dead reckoning) turtle position and facing or player GPS position.*

--:- near place? span?? -> *Report points within span blocks (or all) of named place (or current player or turtle position).*

--:- view place -> *Report place details including name, label (if any), features and all situations.*

--:- where place? count?? -> *Report movement direction, distance to named place (or all) three (or count) closest places.*

--:- fix trail? -> *Set and report GPS turtle position for dead reckoning. Optionally begin named trailhead.*

--:- headings rate? place? count?? -> _Repeated movement report at specified rate (or every *G.Muse.rates.headings) seconds).*

--:# **Map File Operations** (e.g., `rover site ...`)

--:- erase name -> *Remove named place, broadcast Muse eXcise (MX).*

--:- sync -> *Muse Update (MU) broadcast local map to (MQ) registered units.*

--:- site name? -> *Remote operation to report or change site (persistently) after, e.g., porting `rover`.*

--:- site name? -> *Remote operation to report or change site (persistently) after, e.g., porting `rover`.*

--:- join site role -> *Set site and join landed turtle to it with specified role.*

--:# **GPS Launch Command** (e.g., `rover launch gantry ....`)

--:- launch place yD? -> *Deploy GPS launch yD or maximum y above place, report GPS at place.*

--:# **Remote Mining Operation Commands** (e.g. `miner shaft ....`)

--:- shaft minehead levels shaftPlans -> *Dig down number of levels under named minehead place using specified plans.*

--:- bore marker borePlans -> *Dig horizontally from marker using saved or specified bore and shaft plans.*

--:- post marker borePlans? -> *Go to marker (and up 1 block) from current level with saved or specified plans.*

--:- ores marker borePlans? -> *Excavate ores from side tunnel near marker, return up 1 from marker.*

--:# **Remote Volume Operations** (bound by point pairs, possibly in a `range`)

--:- cut point point -> *Quarry out blocks bound by named points (defining a rectangular solid).*

--:< *Filling and target may be one of the turtle categories or a Minecraft detail name without prefix* `minecraft:`

--:- fill point point filling ?target -> *Layer fill bounds by points; optionally swaps out only target blocks.*

--:< *Seed may be one of the turtle categories or a Minecraft detail name without the prefix* `"minecraft:"`

--:- till point point seed -> *Till the seed bounds by named points (defining a rectangular solid).*

--:- fence range [item] -> *Put item or available wooden fence from one point to another in range.*

--:# **Remote Farm Operations** (where a range feature value is a `fieldName` keyed by `field`, a string literal).
--:+ *A `fieldName` is a file name without its suffix, e.g. `cane` rather than `cane.lua`, in the `fields` directory.*
--:= farm:

--:- quarry range firstPlot? lastPlot?? -> *Dig out the field to level it.*

--:- layer range firstPlot? lastPlot?? -> *Put foundation material in place for field.*

--:- cover range firstPlot? lastPlot?? -> *Replace field material (for tree farm grid).*

--:- finish range firstPlot lastPlot?? -> *Complete field preparation for farming.*

--:- harvest range firstPlot lastPlot?? -> *Harvest (and replant if needed).*

--:- path range firstPlot lastPlot?? -> *Test harvest path (safely).*

--:# **Command Computer Setup and Port Commands** (e.g., `locate gantry launch`)

--:- locate name label? -> *Create launch point having Minecraft coordinates above* `porter`.

--:- activate range -> *Add borders from west and north to east and south of range for Minecraft forceload.*

--:- book name label from to span? item??... -> *Spanned range with (default) items as properties; return cost less bank.*

--:- port booking -> *As provided in booking, consume player inventory to teleport entities from one area to another.*