ROBOTBASIC CHEAT SHEET 📄

📋 General Notes

- **Case:** Commands are **not case-sensitive**. Variable names and labels **are** case-sensitive.
- **Screen:** `800 × 600` (width × height). Origin `(0,0)` is **top-left**. +x → right, +y → down.
- **Optional parameters** shown in `[ ]`.
- **All variables are global** by default — subroutines can access and modify them.

⚙️ Initialization & Appearance

`rLocate x, y, [heading, size, color]`

Positions the robot on-screen.

- **x, y** — center coordinates.
- **heading** — degrees (0 = up, positive = clockwise).
- **size** — diameter in pixels (default 20).
- **color** — named color (e.g., `blue`, `red`).

`rInvisible [true|false]`

Hide (`true`) or show (`false`) the robot body.

`rPen up/down`

Raise or lower the robot's pen for drawing paths.

🚙 Motion Commands

`rForward value`

Move robot forward/backward in current heading.

- Positive = forward, Negative = backward.

`rTurn angle`

Rotate by `angle` degrees (positive = clockwise).

`rSpeed value`

Control movement speed. Smaller values = faster, larger = slower.

🎨 Drawing & Graphics

Shape Commands

Command	Description
`circle x1, y1, x2, y2, [outlineColor, fillColor]`	Draws ellipse/circle bounded by `(x1,y1)` and `(x2,y2)`
`rectangle x1, y1, x2, y2, [outlineColor, fillColor]`	Draws rectangle between corners

Utilities

Command	Description
`ClearScr`	Clears screen
`SetColor <color>`	Sets drawing color
`LineWidth n`	Sets stroke thickness
`gotoxy x,y` / `LineTo x,y`	Cursor move / draw line
`XYString x,y, expr`	Display text at coordinates

Tip: Draw obstacles first, then place robot with `rLocate` for proper alignment.

📡 Sensors

Function	Returns	Common Use
`rBumper()`	0 = none, nonzero = hit	Detect collisions
`rFeel()`	0/1	Detect short-range obstacle
`rRange()`	Distance (px)	Measure distance to nearest object

`rLook()`	Color name or `NONE`	Detect color under sensor
`rBeacon(color)`	0 or distance	Detect beacon by color
`rCompass()`	Degrees	Track orientation
`rGpsX()` / `rGpsY()`	Position (px)	Locate robot
`rChargeLevel()`	%	Battery simulation

Tips:

- Poll sensors frequently in motion loops.
- Combine `rRange()` + `rTurn()` for obstacle avoidance.

🖱️ Control Inputs

Keyboard

Command	Purpose
`GetKey var`	Read current key ASCII code
`WaitKey "msg", var`	Wait for keypress
`Ascii("A")` / `Char(65)`	Convert between char/code

Mouse

Command	Purpose
`ReadMouse x,y,b`	Read cursor position `(x,y)` and button `b`

Button `b` Values

b	Meaning	Use
`0`	No click	Idle / tracking
`1`	Left click	Select / draw / move
`2`	Right click	Cancel / context

Tip: Use for click-to-move, drawing, or control panels.

🧩 Control Flow & Subroutines

Labels & Subroutines

Concept	Syntax	Notes
Label	`Name:`	Must start with a letter
Call	`Gosub Name`	Jump to subroutine
Return	`return`	End of subroutine

All variables are global — avoid name collisions by prefixing (`g_` , `tmp_`).

Conditional Structures

Type	Syntax	Example	Notes
Single-line	`if cond then stmt`	`if a>10 then print "High"`	One-liners only

Block	`if cond`
	`  stmts`
	`endif`
		`if b = 1`
		`  goSub task1`
		`endif`
			Multi-line block

If-else block	`if cond`
	`  stmts`
	`elseif cond`
	`  stmts`
	`else`
	`  stmts`
	`endif`
		-	Branching logic

Loop Structures

Type	Syntax	Notes

For Loop	`for i = a to b`
	`  stmts`
	`next`
		Counted iteration

While Loop	`while cond`
	`  stmts`
	`wend`
		Check before body, cond = `true` repeat

Repeat Loop	`repeat`
	`  stmts`
	`until cond`
		Always executes once, cond = `false` repeat

🧭 Navigation Utilities

Function	Description
`PolarA(dx,dy)`	Angle to target point
`PolarR(dx,dy)`	Distance to target point

Tip: Use `rTurn` + `rForward` in small steps with sensor checks for smooth movement.

📜 Quick Reference

Command	Purpose
`rLocate x,y,[heading,size,color]`	Place robot
`rForward value`	Move forward/backward
`rTurn angle`	Rotate robot
`rSpeed value`	Adjust speed
`rInvisible true/false`	Hide/show robot
`rPen up/down`	Toggle pen
`circle`, `rectangle`	Draw shapes
`rBumper()`	Collision detection
`rFeel()`	Proximity check
`rRange()`	Distance measurement
`rLook()`	Color detection
`rBeacon(color)`	Beacon tracking
`rCompass()`	Heading angle
`rGpsX()`, `rGpsY()`	Position data
`rChargeLevel()`	Battery level
`ReadMouse x,y,b`	Mouse input
`GetKey`, `WaitKey`	Keyboard input
`Gosub`, `return`	Subroutine control

🧰 Troubleshooting & Best Practices

⚠️ Sensors

- If `rLook()` fails: ensure color objects (`red` , `blue`) have solid fill and contrast.
- Missed bump/range events? Use smaller step size and poll often.

🎨 Drawing Alignment

- Call `rLocate` *after* drawing environment for correct alignment.

🧩 Variables

- Prefix globals consistently (`g_` , `tmp_`) to avoid overwrite.

💡 Debugging

- Slow movement with `rSpeed` .
- Print `rGpsX()` , `rGpsY()` , `rCompass()` during runs for tracking.