

BUDDY

Brain Using Desk Device for You

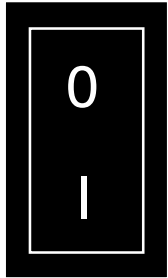
User Guide

Table of Contents

User Requirements	2
Key Words	3
Faces	4
Lessons.....	6

User Requirements

To power on your BUDDY simply flip the switch from 0 to 1 on the back of BUDDY.



Then open the desktop app and your BUDDY is ready to use!

If your BUDDY needs a recharge, simply plug in the charger. You can still use the BUDDY just don't say "trash" (it will get tangled in the cord).

Key Words

In any case, the robot has a listening face displayed for active listening and a thinking face if it is processing the speech to text. Faces shown on Page 4. If thinking did not appear, then the phrase/word was lost. Try saying these words within a sentence so BUDDY can understand you better! (i.e. "Get the Trash" or "I have a Question")

Weather - Displays the outside temperature and weather conditions based on your location

Time – Displays a digital clock with local time

Question- Ask it any question and get a short response in the companion app using the power of Google's Gemini

Trash / Garbage - The BUDDY unit scans for trash then pushes it towards the edge of the table

Calendar- Opens the calendar in the companion app

Faces

BUDDY has many expressive faces to display what they are thinking. Here is a list of what those faces mean



Idle - BUDDY is on standby
as you code!



Listing - BUDDY is listening
for key words



Question - BUDDY is
thinking how to answer the



Thinking - BUDDY is
processing the key phrase



Trash - BUDDY is upset
that you littered

Lessons

1. Say Hello

What the lesson teaches:

The program should display a simple message on the screen. It's the first step in learning to output text.

Example Code:

```
print("Hello, world!")
```

2. Variables

What the lesson teaches:

Store information in a variable and use it in a sentence.

Example Code:

```
name = "Alex"  
print("Hello, " + name + "!")
```

3. Math & Print

What the lesson teaches:

Do math with variables and print the result.

Example Code:

```
a = 7  
b = 5  
print("Sum:", a + b)
```

4. If (Decisions)

What the lesson teaches:

Use an if/else statement to choose between two outputs.

Example Code:

```
score = 75
if score >= 70:
    print("PASS")
else:
    print("TRY AGAIN")
```

5. For Loop

What the lesson teaches:

Repeat an action a specific number of times.

Example Code:

```
for i in range(1, 6):
    print(i)
```


6. While Loop

What the lesson teaches:

Repeat actions until a condition becomes false.

Example Code:

```
n = 3
while n > 0:
    print(n)
    n -= 1
print("GO!")
```

7. Functions

What the lesson teaches:

Create reusable code blocks that return values.

Example Code:

```
def shout(msg):
    return msg.upper() + "!"
print(shout("ready"))
```

8. Lists (Moves)

What the lesson teaches:

Store multiple items in a list and loop through them.

Example Code:

```
moves = ["up", "up", "left", "down", "right"]  
for m in moves:  
    print("move:", m)
```

9. Grid Coordinates

What the lesson teaches:

Track an (x, y) position and update it based on movement instructions.

Example Code:

```
x, y = 0, 0
moves = ["up", "up", "right", "right", "down"]
for m in moves:
    if m == "up":
        y -= 1
    elif m == "down":
        y += 1
    elif m == "left":
        x -= 1
    elif m == "right":
        x += 1
print("final:", x, y)
```

10. Turtle Square

What the lesson teaches:

Use turtle graphics to draw shapes.

Example Code:

```
import turtle as t
t.speed(3)
for _ in range(4):
    t.forward(100)
    t.left(90)
t.done()
```

11. Dictionaries (Key → Action)

What the lesson teaches:

Map keys to actions using a dictionary.

Example Code:

```
keys = ["W", "A", "S", "D", "W"]
keymap = {"W": "up", "A": "left", "S": "down", "D": "right"}
for k in keys:
    print("do:", keymap.get(k, "?"))
```

12. Nested Loops (Grid Print)

What the lesson teaches:

Use loops inside loops to build grids or tables.

Example Code:

```
for r in range(3):  
    line = ""  
    for c in range(3):  
        line += f"({r},{c}) "  
    print(line)
```

13. Input Guard (Clamp)

What the lesson teaches:

Limit a value so it stays within allowed bounds.

Example Code:

```
speed = 14  
speed = max(0, min(10, speed))  
print("speed:", speed)
```

14. Random Route

What the lesson teaches:

Use randomness to create unpredictable behavior.

Example Code:

```
import random
moves = [random.choice(["up", "down", "left", "right"]) for _ in range(5)]
print("moves:", moves)
```

15. Classes (Robot)

What the lesson teaches:

Create objects with data and behaviors.

Example Code:

```
class Robot:
    def __init__(self):
        self.x = 0
        self.y = 0
    def move(self, m):
        if m == "up": self.y -= 1
        elif m == "down": self.y += 1
        elif m == "left": self.x -= 1
        elif m == "right": self.x += 1
    def pos(self):
        return (self.x, self.y)
r = Robot()
for m in ["up", "right", "right"]:
    r.move(m)
print("pos:", r.pos())
```

16. Turtle Triangle

What the lesson teaches:

Drawing shapes using angles and movement.

Example Code:

```
import turtle as t
t.speed(4)
for _ in range(3):
    t.forward(120)
    t.left(120)
t.done()
```


17. Path Replay (Turtle)

What the lesson teaches:

Use a list of actions to draw a path in order.

Example Code:

```
import turtle as t
t.speed(5)
path = [
    ("forward", 80),
    ("left", 90),
    ("forward", 80),
    ("right", 90),
    ("forward", 80)
]
for action, value in path:
    if action == "forward": t.forward(value)
    elif action == "left": t.left(value)
    elif action == "right": t.right(value)

t.done()
```

18. Functions with Params

What the lesson teaches:

Create a function with input parameters that returns a calculation.

Example Code:

```
def move(x, y, direction):  
    if direction == "up": return x, y - 1  
    if direction == "down": return x, y + 1  
    if direction == "left": return x - 1, y  
    if direction == "right": return x + 1, y  
    return x, y  
x, y = 0, 0  
for d in ["up", "up", "right", "down"]:  
    x, y = move(x, y, d)  
print("final:", x, y)
```

19. Obstacles & Bounds

What the lesson teaches:

Prevent movement into forbidden spaces or outside the grid.

Example Code:

```
x, y = 0, 0
rock = (2, 2)
W = H = 5
for d in ["right", "right", "down", "down", "down"]:
    nx, ny = x, y
    if d == "up": ny -= 1
    elif d == "down": ny += 1
    elif d == "left": nx -= 1
    elif d == "right": nx += 1
    if 0 <= nx < W and 0 <= ny < H and (nx, ny) != rock:
        x, y = nx, ny

print("safe at:", x, y)
```

20. Save & Load (JSON)

What the lesson teaches:

Save data to a file and load it back in.

Example Code:

```
import json

data = {"score": 123}
with open("tiny_save.json", "w") as f:
    json.dump(data, f)

with open("tiny_save.json", "r") as f:
    loaded = json.load(f)

print("loaded score =", loaded["score"])
```