
THINK LIKE A
PROGRAMMER
NICK DEBEURRE

AGENDA

1. Think Like a Programmer
2. Fundamentals
3. Time to Run Your First Program
4. Tic-Tac-Toe Build Along
5. Hangman Self-Exploration Challenge

BEFORE WE START

- My name is Nick
- I am Miss Audrey’s husband
- I am an AI software engineer at Qualcomm
- Coding may seem scary but I am here to help you ☺
- If you feel confused, stuck, scared, or have a question, raise your hand

PART 1

Think Like a Programmer

WHY YOU SHOULD LEARN TO CODE

- Computers are everywhere
- Computers work quickly – much faster than humans
- Computers cannot
  - Act on their own
  - Make things from scratch
- Learning to code is learning to communicate with your computer
- Programming is digital freedom

WHAT IS PROGRAMMING?

- Programming is telling a computer what to do
- Programs are like recipes for the computer to “cook” from
- Think of a restaurant:
  - The chef creates a recipe for a dish
  - When a customer orders the dish, the chef uses the recipe to make it
  - Now, the chef doesn’t need to make the dish from scratch every time
- Programming is similar:
  - You create a program for a task and save it in a file
  - When you want your computer to perform the task, you run the program

PART 2

Fundamentals

TYPES, VARIABLES, AND RESERVED WORDS

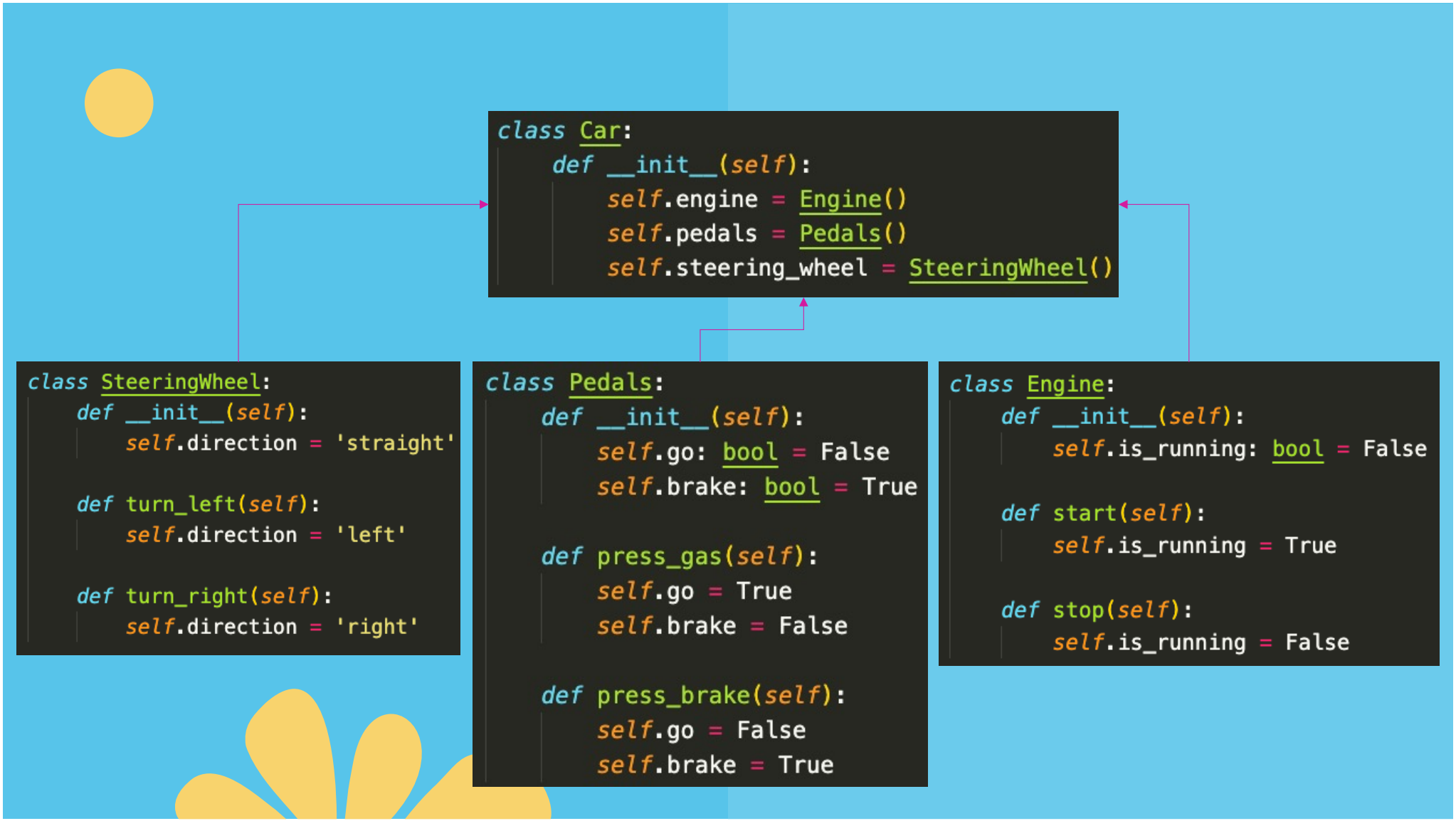
- Types are different kinds of information the computer can store
  - Integer (int) → a positive or negative whole number, for example 1
  - String (str) → a word, for example ‘bread’
  - Boolean (bool) → True or False
- Variables are how you define changing information for a computer
  - secret_word = ‘fancy sauce’
  - my_fav_color = ‘green’
- Reserved words are part of the programming language and cannot be used as variables
  - action = pet() **if** dog.is_friendly() **else** look()

CONTROL STRUCTURES AND OPERATORS

- Control structures are building blocks that can be used to control the flow of the program
  - Loops
    - while True
    - for item in list
  - Conditionals
    - If → else if → else
  - Functions
    - myFunction()
- Operators are symbols that perform operations
  - Some examples include +, -, =, ==

OBJECT ORIENTED PROGRAMMING

- Object-oriented programming (OOP) is a way of writing programs where you use digital objects to organize and handle information
- Consider a car
  - How do you steer?
  - How do you stop and go?
- Let's think about the important sub-components of a car
  - Steering Wheel
  - Pedals
  - Engine
- Now let's see it in code!


class Car:
def __init__(self):
self.engine = Engine()
self.pedals = Pedals()
self.steering_wheel = SteeringWheel()
class SteeringWheel:
def __init__(self):
self.direction = 'straight'
def turn_left(self):
self.direction = 'left'
def turn_right(self):
self.direction = 'right'
class Pedals:
def __init__(self):
self.go: bool = False
self.brake: bool = True
def press_gas(self):
self.go = True
self.brake = False
def press_brake(self):
self.go = False
self.brake = True
class Engine:
def __init__(self):
self.is_running: bool = False
def start(self):
self.is_running = True
def stop(self):
self.is_running = False

PART 3

Time to run your first program!

HELLO WORLD

Step 1: Make a workspace

a. Create folder on the desktop called [name]_workspace

Step 2: Open a terminal

a. Press cmd + space to open the search bar
b. Type ‘terminal’ and open the application

Step 3: Visual Studio Code (VS Code)

1. Go to applications and open Visual studio code
2. Right click in the workspace and click add folder
3. Choose your workspace from the Desktop

Step: 4 Make a new file

1. Right click on the directory in VS Code
2. Click ‘New File…’
3. Name it hello_world.py

Step 5: Add your code

1. name = ‘name’
2. print(’Hello World! My name is ’ + name + ‘.’)

Step 6: Run the code

1. run ‘python hello_world.py’
2. You should see the output “Hello World! My name is name.”

PART 4

Tic Tac Toe

PART 5

Hangman