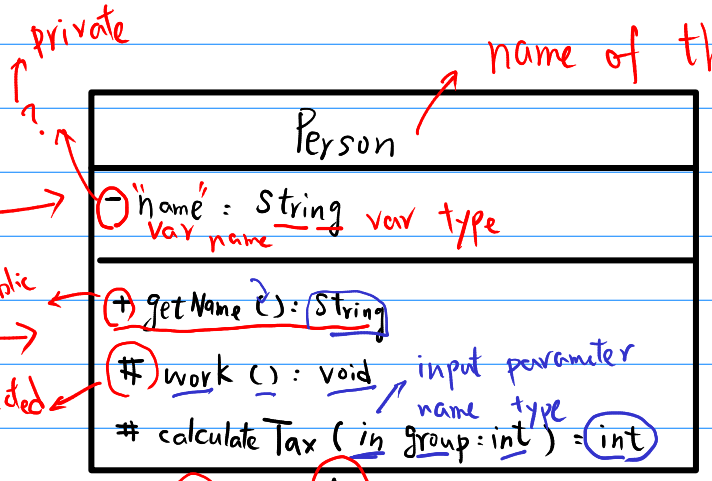
MSE 2022 SS Tutorial 2

Organization

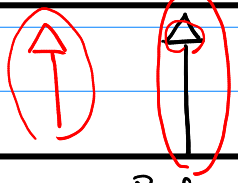
- Microphone = any noise now?
  - Please tell me immediately if something (video, audio) does not work
- Recording: we will record solutions today
  - Please turn off your video and audio during the recording!
  - Please also postpone your questions after the recording!
- Questions?

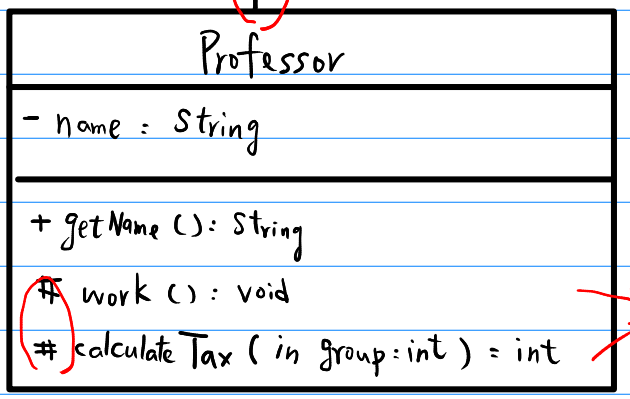
Topic UML (Unified Modeling Language)

- Class
  - Attributes
  - Methods

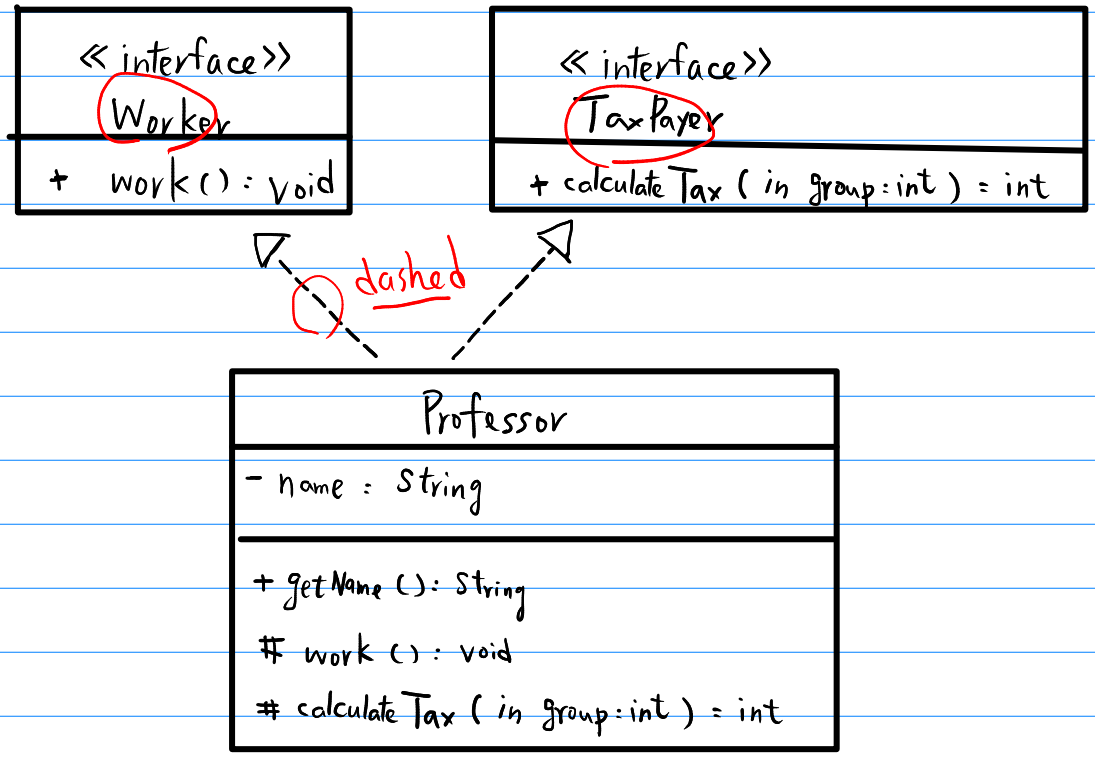


- "Inheritance" (Generalization)
  "extends" (Java)





- Interface (Realization) "implements"

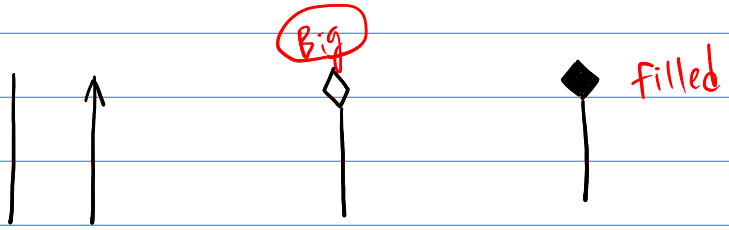


Q: Is it good to have work() in Person?
(virtual function) (base class)

- Interface : flexible
- class Baby extends Person.

- Interface Segregation (SOLID Principles)

- Association , Aggregation, Composition (Relationship between class)

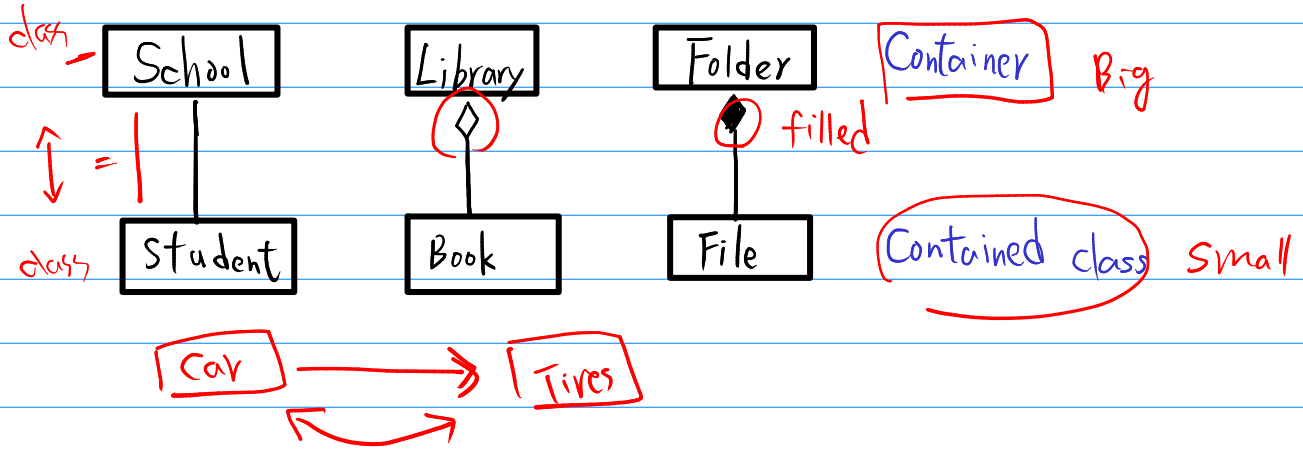


generic relationship

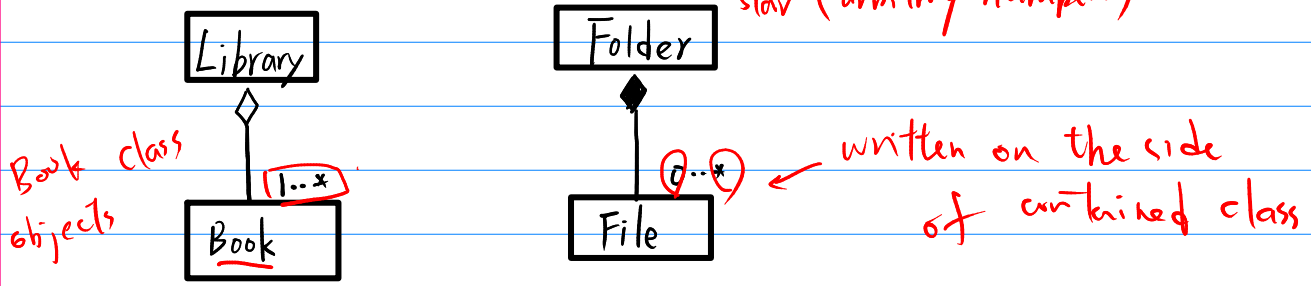
independent existence

"part-of" relationship

Note : distinction can be ambiguous, depending on applications

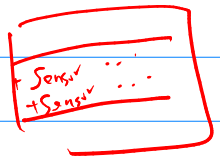


- Multiplicities : From m to n copies
  m..n , 0..* , 1..*
  ↳ star (arbitrary numbers)



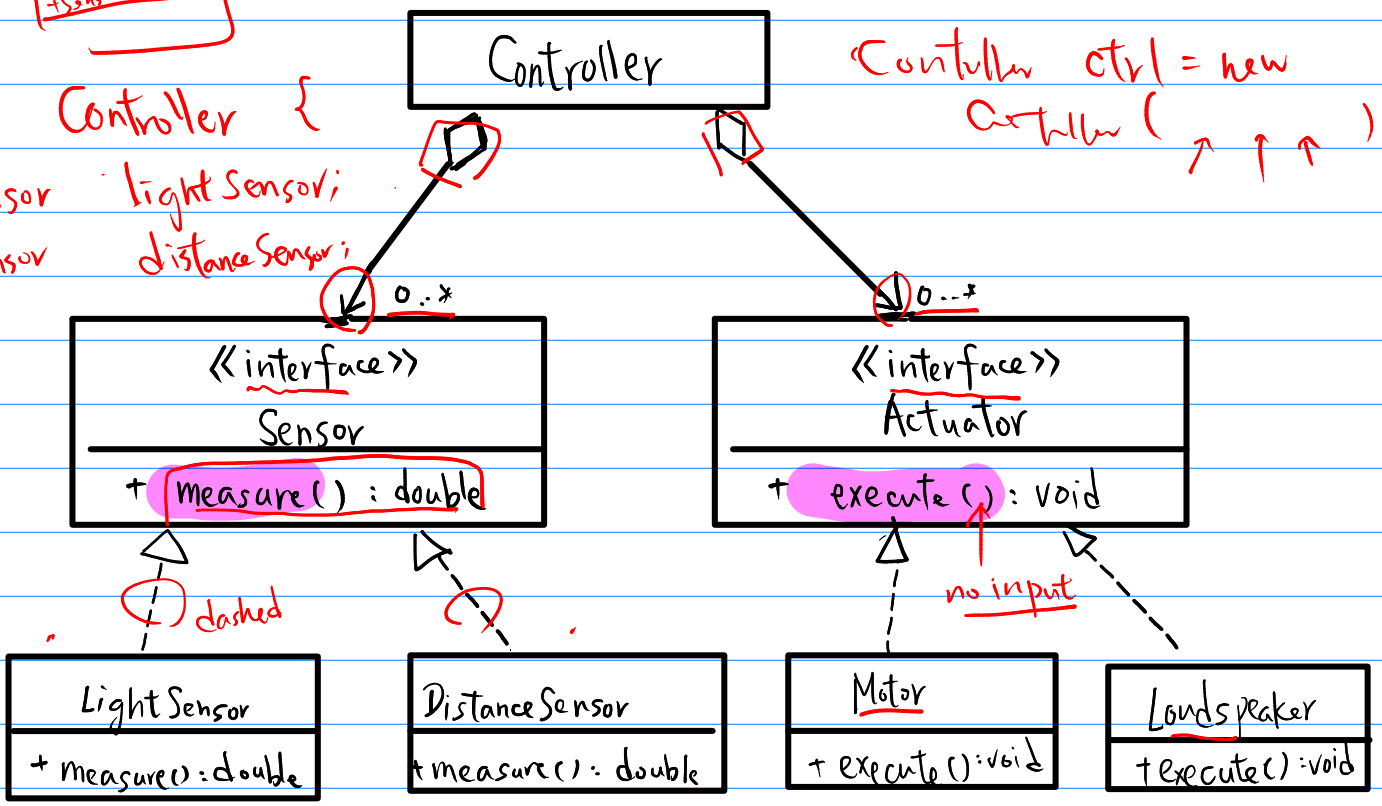
Task 1 UML of a robot (Recording, please turn off video)

- Controller
- Sensor : measurement, light, distance
- Actuator : navigate, motor, loudspeaker



class Controller {
Sensor lightSensor;
Sensor distanceSensor;
}

Controller ctrl = new Controller(↑ ↑ ↑)



- Note: There might not be a meaningful interface for Actuator because of different input parameters
- All classes are Components (no attributes, well-defined interface)

- Questions? (Pause recording!)

Topic CFG (Context-Free Grammars)

- Production rule: $A \to \alpha$ (A: nonterminal)
  - A: nonterminal symbol
  - α: a string of terminal and/or nonterminal symbols

Ex: $\Sigma = \{0, 1\}$ (terminal symbol)
$V = \{S\}$ (nonterminal =)

$S \to 0S1 \mid \varepsilon$ empty string

Q: What is the language?

A: $L = 0^n 1^n$, $n \geq 0$

$S \to \varepsilon$

$S \to 0S1 \to 01$

$S \to 0S1 \to 00S11 \to 0011$

- EBNF (Extended Backus-Naur Form)
  - A set of notations to express CFG
  - Important symbols
    - $=$: definition
    - $;$: termination
    - , : concatenation (or, without symbols) 01
    - $\{..\}$: repetition (or, Kleene star) $0^* 1^*$
    - $|$: alternation or.

Task 2 CFG for sandwich (Recording, turn off video please!)

Sol#1: Sandwich = Bread, Toppings, Bread; Terminals

repeating alternation

Toppings = { Ham | Egg | Cheese }; Non terminals

0 repetition

Q: Is this solution correct?

A: No! Because the word "Bread, Bread" is not in the language!

Sol#2: Sandwich = Bread, Toppings, Bread;

repetition

Toppings = (Ham | Egg | Cheese), { Ham | Egg | Cheese };

→ Egg, Ham, Cheese, Ham

Sol#3: Sandwich = Bread, Toppings, Bread;

Toppings = (Ham | Egg | Cheese), (Toppings | ε);

- Questions? (Pause recording!)

* : 0 or more repetitions

+ : 1 or mor..

Task 3 Represent Domain-Specific Language of a Robot

(Recording! Please turn off videos!)

EBNF:

```
program = {command ";"};
command = "turn left" |
          "turn right" |
          "move forward" number;
```

Example program: turn left; turn right; move forward 2;

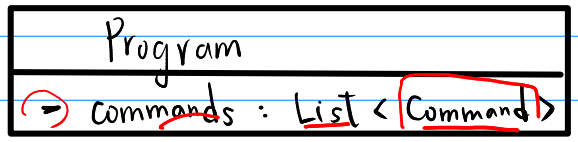
Q: What is a program in this language?

A: A list of commands

Q: What is a command?

A: turn left, turn right, or move forward

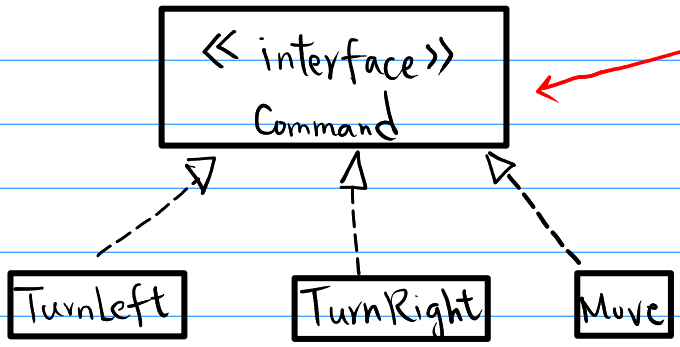
Encoded in the grammar, which tells us about the implementation



+ run() : void

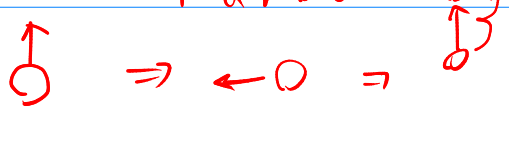
for (cmd : commands)
cmd.execute()

- Questions?
(Pause recording!)



- Why is Program a class?
(Recall shopping list)
① typedef UML* ② add run()
- What would the "state" be?
physical state in the env. of a robot

initial state → cmd → updated state → cmd



Task 4 Domain-Specific Languages (DSL) (Interactive, no recording)

- Specialized to a particular application domain
  - v.s. General-Purpose Languages (C++, Java, Python, ...)

SQL (database), R (statistics), HTML (web), Verilog (FPGA, IC)

Examples:

- DOT: graphs
- PlantUML: UML diagrams
- sed, awk: string manipulation
- SQL: databases
- HTML: webpages
- VHDL, Verilog: hardware (e.g., integrated circuits, FPGA)
- Make: build system
- LaTeX: document formatting
- XML, JSON, YAML: data encoding

(BenchExec; path finding in autonomous driving)

XML: encoder exp. config → JSON. tool use