Основи об'єктно - орієнтованого програмування

- Об'єкти
- Класи
- Відносини між класами

Об'єктно - орієнтоване програмування

ООП — це цілий набір концепцій та ідей, що дозволяють осмислити завдання, що стоїть при розробці комп'ютерної програми, а потім знайти шлях до її вирішення більш зрозумілим, а, значить, і більш ефективним способом.

Задачі, що вирішуються:

- зменшення **складності** програмного забезпечення
- підвищення **надійності** програмного забезпечення
- забезпечення можливості **модифікації** окремих **компонентів** програмного забезпечення без зміни інших його компонентів.
- забезпечення можливості **повторного використання** окремих компонентів

Об'єкти

- **Об'єкт** — деяка сутність у віртуальному просторі, що володіє певним станом і поведінкою, має задані значення властивостей (атрибутів) та операцій над ними (методів)

Так що ж таке об'єкт?

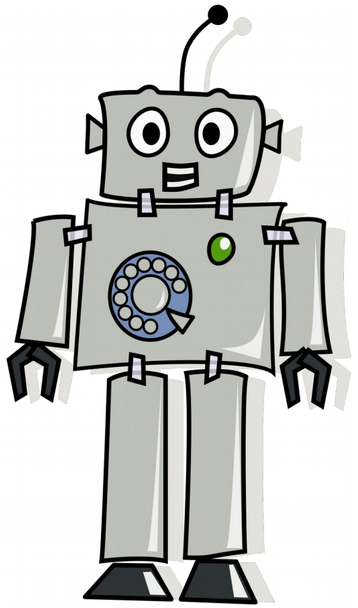
- Реальні об'єкти мають дві характеристики:
  - стан
  - поведінка
- Програмні об'єкти концептуально схожі на об'єкти реального світу: вони теж складаються зі **стану** та **поведінки**.

Стан і поведінка

- Об'єкт зберігає свій стан в **полях** (змінні в деяких мовах програмування)
- Об'єкт реалізує свою поведінку за допомогою **методів** (функцій в деяких мовах програмування
- Приховування внутрішнього стану та надання можливості працювати зі станом тільки через методи - **інкапсуляція**

На що це схоже?

Робот:



Щось тут не так...

- Чи схожі всі роботи?
- Як розрізняти об'єкти одного типу?

Унікальність

- Унікальність — це те, що відрізняє один об'єкт від інших.
- У машинному поданні під параметром унікальності об'єкта найбільш часто розуміється адреса розміщення об'єкта в пам'яті.

Плюси використання об'єктів:

- **Модульність:** програмний код об'єкта підтримується незалежно від програмного коду інших об'єктів
- **Приховування інформації:** взаємодія тільки з методами об'єктів приховує внутрішню реалізацію
- **Повторне використання коду:** Ви можете знову і знову використовувати одного разу написаний об'єкт у своїй програмі.
- **Взаємозамінність та налагодження:** якщо об'єкт Вас не влаштовує просто змініть його на інший.
- **Немає сенсу міняти всю машину в разі поломки маленького гвинтика**

Класи

- **Клас** — це креслення для створення об'єктів
- Клас — **шаблон** поведінки **об'єктів** певного типу з певними параметрами, що визначають стан. Усі примірники одного класу (об'єкти, породжені від одного класу):
  - Мають один і той же набір властивостей
  - Загальна поведінка, однаково реагують на однакові повідомлення

Знову робот

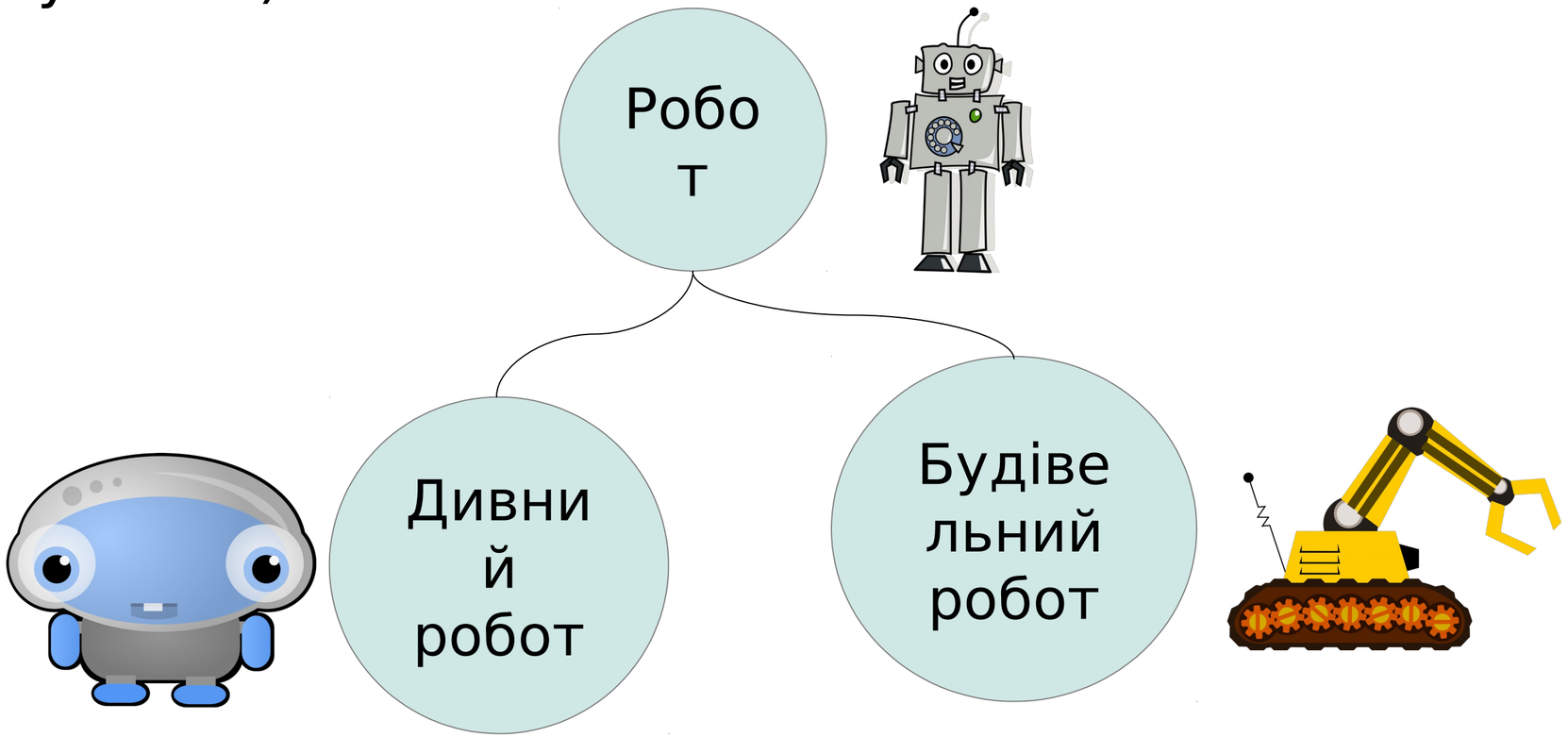
```
public class Robot {
    int xPosition;
    int yPosition;
    boolean isActive;

    void moveX(){
        this.xPosition++;
    }

    void moveY(){
        this.xPosition--;
    }

    void play(){
        isActive = true;
        moveX();
        moveY();
    }
}
```

Типи відносин між класами

- Комп'ютерна програма з використанням ООП **будується з класів**.
- Ці класи повинні **знати один про одного**, для того щоб взаємодіяти між собою і спільно виконати поставлене завдання.
- Можливі наступні **зв'язки між класами** в рамках об'єктної моделі:
  - **Агрегація** (Aggregation)
  - **Асоціація** (Association)
  - **Успадкування** (Inheritance)
  - **Метакласи** (Metaclass)

Успадкування

Успадкування (inheritance) - це відношення між класами, при якому клас використовує структуру або поведінку іншого (одиночне спадкування) або інших (множинне спадкування) класів.



Агрегація

- Відношення між класами типу "містить" або "складається з" називається **агрегацією** або **включенням**.

- **Наприклад:** якщо акваріум наповнений водою і в ньому плавають рибки, то можна сказати, що акваріум агрегує в собі воду і рибок.

Асоціація

- Якщо **об'єкти одного класу** посилаються на один або більше об'єктів **іншого класу**, але ні в ту, ні в іншу сторону відношення між об'єктами **не носить характеру "володіння"** або контейнеризації, то такі відносини називають **асоціацією** (association).

- **Наприклад:** програміст і його комп'ютер. Між цими двома об'єктами немає агрегації, але існує чіткий взаємозв'язок.

Метакласи

- Шаблон, що задає різні класи, називається **метакласом**.
- Об'єкти породжуються від класів, а класи — від метакласу.

Поліморфізм

- Поліморфізм — це можливість об'єктів з однаковою специфікацією мати різну реалізацію.
- Один інтерфейс — безліч реалізацій.

Підсумок

- **Класи** — визначення об'єктів одного і того ж абстрактного типу даних.
- **Наслідування** — виведення одного класу від іншого так, що поля і методи одного класу є частиною визначення іншого класу.
- **Поліморфізм** — здатність класів, реагувати на ту саму дію у різний спосіб.
- **Інкапсуляція** — об'єднання разом даних та поведінки в єдиний абстрактний тип даних з відкритим інтерфейсом і закритою реалізацією.
- Об'єкти мають стан, що змінюється відповідно до виклику методів, які можуть викликати інші об'єкти.