Class Diagram

A **Class Diagram** is a visual representation used in object-oriented design to show the structure of a system. It represents **classes** in a system, their **attributes** (properties), **methods** (behaviors), and **relationships** between classes.

Basic Elements of a Class Diagram

1. Class

- Represents a blueprint for objects in the system.
- **Notation**: A rectangle divided into three sections:
    - **Top section**: Class name
    - **Middle section**: Attributes (properties of the class)
    - **Bottom section**: Methods (functions or behaviors of the class)

Example:

arduino

Copy code

```
+-----------------+
|    **Car**      |
+-----------------+
| - color: String  |
| - model: String  |
+-----------------+
| + drive(): void  |
| + stop(): void   |
+-----------------+
```

---

2. Attributes

- Define the properties or characteristics of a class.
- **Notation**:
    - Visibility (access modifier):
        - **+** (public): Accessible everywhere.
        - **-** (private): Accessible only within the class.

- **Attribute name**: Name of the property.
- **Data type**: Type of data (e.g., String, int).

- **Example**: - color: String

---

3. Methods (Operations)

- Define the behaviors or functions of a class.
- **Notation**:
  - Visibility (access modifier):
    - **+** (public): Method can be accessed from anywhere.
    - **-** (private): Method can only be used within the class.
  - **Method name**: Name of the method.
  - **Parameters**: List of input values.
  - **Return type**: The type of value the method returns.
- **Example**: + drive(): void

Relationships in Class Diagrams

1. Association

- **Meaning**: A connection between two classes, showing that they are related.
- **Notation**: A straight line between two classes.
- **Example**: Person and Car are associated if a person can drive a car.

Diagram:

lua

Copy code

```
+---------+         +---------+
|  Person |         |   Car   |
+---------+         +---------+
     |----------------|
```

2. Inheritance (Generalization)

- **Meaning**: One class (child) inherits properties and methods from another class (parent).
- **Notation**: A solid line with a hollow triangle pointing to the parent class.

Diagram:

lua

Copy code

```
+-----------+         +---------+
| Vehicle |◀----------|  Car   |
+-----------+         +---------+
```

3. Aggregation

- **Meaning**: A "whole-part" relationship, where the part can exist independently of the whole.
- **Notation**: A line with a white diamond near the whole class.

Diagram:

diff

Copy code

```
+----------+        ◇---------+
| Library  |        |  Book   |
+----------+        +---------+
```

4. Composition

- **Meaning**: A "whole-part" relationship where the part cannot exist without the whole.
- **Notation**: A line with a black diamond near the whole class.

Diagram:

diff

Copy code

```
+---------+        ◆---------+
| House   |        |  Room   |
+---------+        +---------+
```

5. Multiplicity

- **Meaning**: Specifies how many instances of one class relate to instances of another.
- **Notation**: Numbers like 1, 0..*, or 1..* near the association line.

Diagram Example: A person can own multiple cars, but each car belongs to only one person.

lua

Copy code

```
+---------+ 1      * +---------+
| Person |◀--------▶|   Car    |
```

+---------+ +---------+

Summary Diagram

Below is a simplified class diagram to show an example:

lua

Copy code

```
+---------+          +---------+          +---------+
| Person |◄----------► |  Car   | ◇-------- |  Engine |
+---------+          +---------+          +---------+
```

- **Person** and **Car** have an association (a person can drive a car).
- **Car** and **Engine** have an aggregation (the engine is part of the car but can exist separately).

This diagram shows how class diagrams visually organize a system's structure by representing classes and their connections.