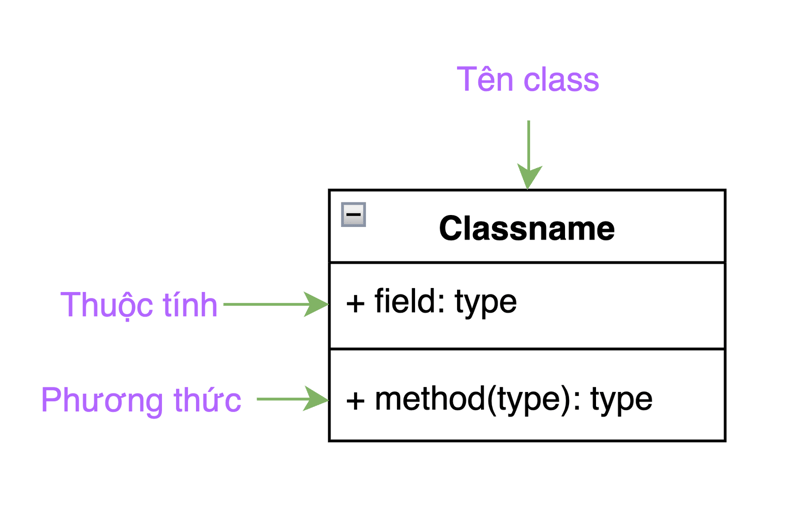
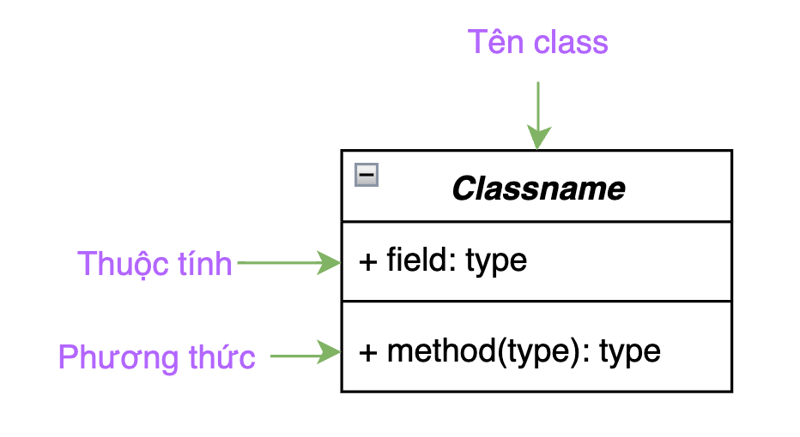
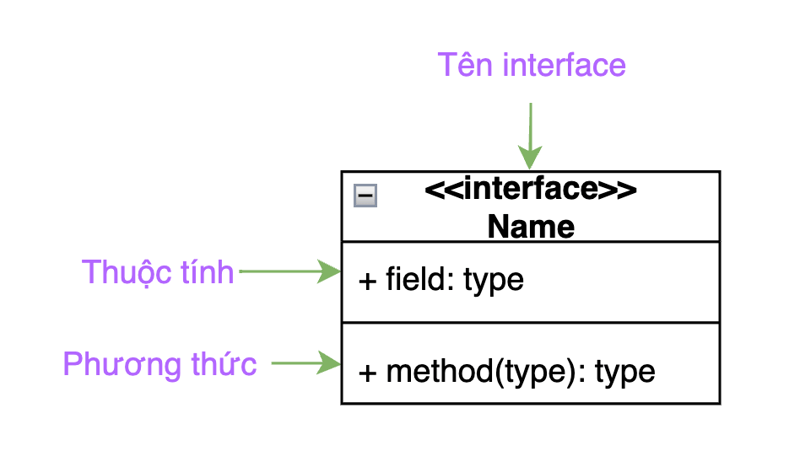
### 1. Class

****Class**** đại diện cho một nhóm các đối tượng có thuộc tính và hành vi tương tự nhau, dùng để mô hình hóa và thiết kế các thành phần của hệ thống trong lập trình hướng đối tượng, giúp các lập trình viên hiểu rõ cấu trúc và mối quan hệ giữa các đối tượng.

Một class bao gồm ba thành phần chính:

* + ****Tên class:**** Đây là tên đại diện cho nhóm đối tượng.
  + ****Thuộc tính (Attribute):**** Các biến mô tả trạng thái của các đối tượng trong class.
  + ****Phương thức (Method):**** Các hàm hoặc thủ tục thể hiện hành vi của các đối tượng trong class.

Đối với abstract class, tên class sẽ được viết in nghiêng.  
  
[](https://media2.dev.to/dynamic/image/width=800,height=,fit=scale-down,gravity=auto,format=auto/https:/dev-to-uploads.s3.amazonaws.com/uploads/articles/8bh0zw3f74j26lqphykz.png)

Đối với interface, tên interface sẽ được thêm <<interface>>.  
  
[](https://media2.dev.to/dynamic/image/width=800,height=,fit=scale-down,gravity=auto,format=auto/https:/dev-to-uploads.s3.amazonaws.com/uploads/articles/owr9610h45v091w3zywc.png)

### 2. Thuộc tính (Attribute)

* ****Thuộc tính (Attribute)**** mô tả các đặc điểm hoặc thông tin liên quan đến lớp. Mỗi đối tượng được tạo ra từ lớp sẽ có một bộ giá trị riêng cho các thuộc tính này.
* ****Đặc điểm của thuộc tính:****
  + Tên thuộc tính: Dùng để mô tả thông tin cụ thể.
  + Kiểu dữ liệu: Biểu thị loại thông tin, ví dụ như String, int, Date.

****Phạm vi truy cập:****

* + ****\* (Public)****: Có thể truy cập từ bất kỳ đâu.
  + ****- (Private)****: Chỉ truy cập được từ bên trong lớp.
  + ****# (Protected)****: Truy cập được từ lớp đó và các lớp con.

****Tầm quan trọng của thuộc tính:**** Thuộc tính giúp định nghĩa tính chất của đối tượng, từ đó làm nền tảng để hệ thống xử lý dữ liệu và thực hiện các chức năng.

### 3. Phương thức (Method)

****Phương thức (Method)**** thể hiện các hành động hoặc chức năng mà lớp có thể thực hiện trong hệ thống. Nó đại diện cho hành vi của các đối tượng được tạo ra từ lớp.

****Đặc điểm của phương thức:****

* + Tên phương thức: Biểu thị hành động cụ thể mà lớp thực hiện.
  + Tham số: Dữ liệu đầu vào mà phương thức cần để thực hiện hành động.
  + Kiểu trả về: Kết quả của hành động (nếu có).

****Tầm quan trọng của phương thức:**** Phương thức là yếu tố quyết định hành vi của lớp, giúp hệ thống thực hiện các chức năng cụ thể và tương tác với người dùng.

### 4. Mối quan hệ (Relationship)

****Mối quan hệ (Relationship)**** là yếu tố quan trọng giúp kết nối các lớp với nhau trong Class Diagram. Nó mô tả cách các lớp tương tác và phụ thuộc lẫn nhau, từ đó xây dựng nên cấu trúc hoàn chỉnh của hệ thống.

****Các loại mối quan hệ phổ biến:****

****Association (Liên kết)****

****Ký hiệu:**** Đường thẳng nối giữa hai class, có thể có mũi tên ở một hoặc cả hai đầu tùy thuộc vào chiều hướng của quan hệ.

* + Đối với ký hiệu là đường thẳng, đó là mối quan hệ 2 chiều.

[IMG_259](https://media2.dev.to/dynamic/image/width=800,height=,fit=scale-down,gravity=auto,format=auto/https:/dev-to-uploads.s3.amazonaws.com/uploads/articles/y30541v6h6z853yuesxm.png)

* + Đối với ký hiệu ở một đầu, đó là mối quan hệ 1 chiều.

[IMG_260](https://media2.dev.to/dynamic/image/width=800,height=,fit=scale-down,gravity=auto,format=auto/https:/dev-to-uploads.s3.amazonaws.com/uploads/articles/x4noz97w47zhs5u8m57z.png)

****Mô tả:**** Association biểu diễn mối quan hệ giữa hai class trong đó các đối tượng của một class liên kết với các đối tượng của class khác.

****Inheritance (Kế thừa)****

****Ký hiệu:**** Đường thẳng nối giữa hai class và kết thúc bằng một tam giác trắng.

[IMG_261](https://media2.dev.to/dynamic/image/width=800,height=,fit=scale-down,gravity=auto,format=auto/https:/dev-to-uploads.s3.amazonaws.com/uploads/articles/riuihh6xlwspmt1z94wj.PNG)

****Mô tả:**** biểu diễn mối quan hệ kế thừa giữa hai class với nhau hoặc giữa class và abstract class.

****Realization/Implementation (Triển khai)****

****Ký hiệu:**** Đường nét đứt nối giữa hai class và kết thúc bằng một tam giác trắng.

[IMG_262](https://media2.dev.to/dynamic/image/width=800,height=,fit=scale-down,gravity=auto,format=auto/https:/dev-to-uploads.s3.amazonaws.com/uploads/articles/qpsd39hsz8mysbeo3f7v.PNG)

****Mô tả:**** Biểu diễn mối quan hệ giữa một class và một interface, trong đó class cam kết thực hiện (implement) các phương thức được định nghĩa trong interface.

****Dependency (Phụ thuộc)****

****Ký hiệu:**** Đường nét đứt nối giữa hai class và thường kết thúc bằng một mũi tên thể hiện hướng phụ thuộc.

[IMG_263](https://media2.dev.to/dynamic/image/width=800,height=,fit=scale-down,gravity=auto,format=auto/https:/dev-to-uploads.s3.amazonaws.com/uploads/articles/wmv16bgkt7hrpirks85o.PNG)

****Mô tả:**** Một class phụ thuộc vào class khác, thường chỉ trong một phạm vi nhất định hoặc cho một chức năng cụ thể.

****Aggregation****

****Ký hiệu:**** Đường thẳng nối giữa hai class với một hình thoi rỗng ở đầu phía bên của class cần tổng hợp dữ liệu từ class còn lại.

[IMG_264](https://media2.dev.to/dynamic/image/width=800,height=,fit=scale-down,gravity=auto,format=auto/https:/dev-to-uploads.s3.amazonaws.com/uploads/articles/sjyi1ulttlqy3lzwf7vd.PNG)

****Mô tả:**** một class cần thông tin từ class khác, tuy nhiên, hai class này có thể tồn tại độc lập với nhau.

****Composition****

****Ký hiệu:**** Đường thẳng nối giữa hai class với một hình thoi đen ở đầu phía bên của class cần tổng hợp dữ liệu từ class còn lại.

[IMG_265](https://media2.dev.to/dynamic/image/width=800,height=,fit=scale-down,gravity=auto,format=auto/https:/dev-to-uploads.s3.amazonaws.com/uploads/articles/iun5v0tyetxygn3i7der.PNG)

****Mô tả:**** một class cần thông tin từ class khác, tuy nhiên, class chứa thông tin không thể tồn tại mà không có class kia.

* + Trong các mối quan hệ, lượng số (cardinality hoặc multiplicity) được sử dụng để biểu thị số lượng các đối tượng có thể tham gia vào một mối quan hệ giữa các lớp. Điều này giúp làm rõ ràng các ràng buộc và quy tắc trong hệ thống. Dưới đây là một số ví dụ về các lượng số phổ biến:

****1****: Một đối tượng của lớp này liên kết với một và chỉ một đối tượng của lớp khác.

****0..1****: Một đối tượng của lớp này có thể liên kết với không hoặc một đối tượng của lớp khác.

****0..\* hoặc****: Một đối tượng của lớp này có thể liên kết với bất kỳ số lượng nào của đối tượng lớp khác (bao gồm cả không có).

****1..****: Một đối tượng của lớp này liên kết với ít nhất một đối tượng của lớp khác.

****n****: Một đối tượng của lớp này liên kết với chính xác n đối tượng của lớp khác.

****m..n****: Một đối tượng của lớp này liên kết với ít nhất m và nhiều nhất n đối tượng của lớp khác.