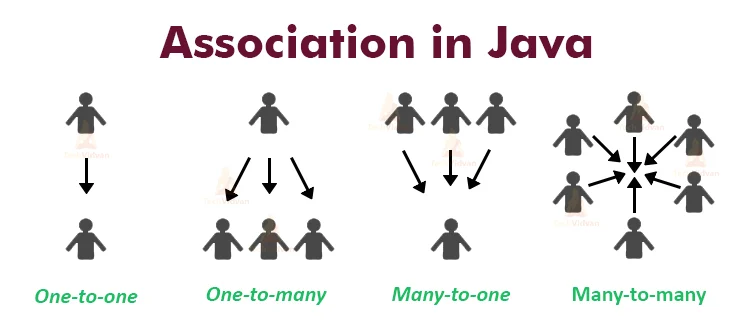
**OOP concepts**

1. Association: Connection between 2 separate classes set up through their objects. This relationship indicates how these objects know each other. Types:
   1. One to one: One person can have one passport.
   2. One to many / many to one: Every city is in one state but a state can have multiple cities.
   3. Many to many: A teacher can have multiple students and a student can have multiple teachers.



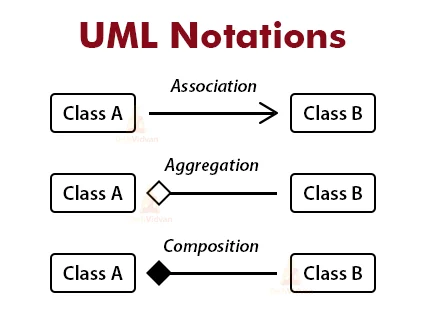
2 forms of association is as follows:

1. Aggregation:

* Has a relationship
* Weak association
* If one dies it will not affect the other on (lifecycles are not tied)
* To maintain the code reusability
* Ex: employee – bank

1. Composition:

* Has-a relationship
* Strong association
* If parent dies the child dies as well, but if child dies it will not affect the parent (lifecycles are tied)
* Ex: heart and human – heart doesn't exist if human doesn't exists



**Abstraction**

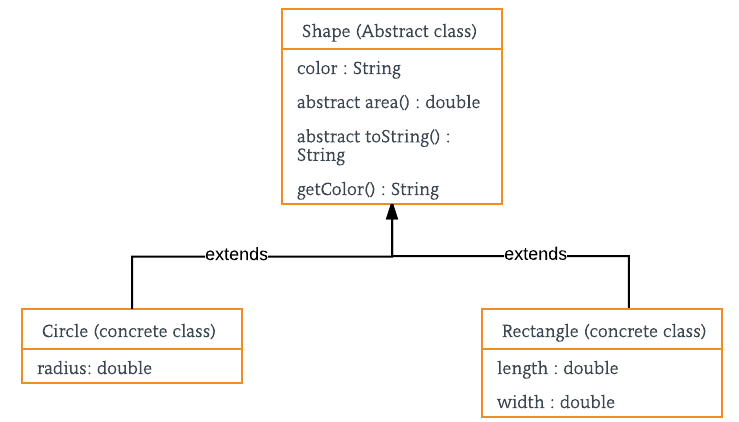
Facts:

* Data abstraction is the process of hiding certain details and showing only **essential information** to the user. Ex: A car is viewed as a car rather than its individual components.
* Can be achieved by abstract classes or interfaces
* Abstract keyword:
  + Classes
  + Methods – only in abstract classes
* Methods declared without implementation
* Need to extend from a class to implement the methods – cannot create an object of an abstract class.
* Classes can have regular methods and abstract methods. Regular ones have implementations.
* Abstraction is detail hiding(implementation hiding)

Advantages:

* Reduce the complexity of viewing things.
* Avoid code duplication
* Increase reusability
* Increase security by showing only essential info.

Use case: General case and some methods are abstract for classes to fill in accordingly.



**Encapsulation**

Facts:

* Make sure sensitive information is hidden from the user.
* Wrapping up the data under a single unit.
* Variables of a class are hidden from other classes. They can only be accessed in the same class.
* Way to achieve encapsulation:
  + Declare variables as private
  + Write public getters and setters for the variables.

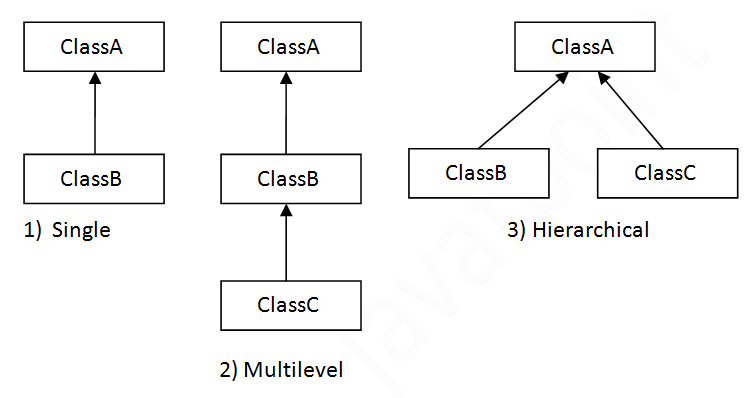
Advantages:

* Can make a class read-only or write-only
* Better control over the attributes of the class
* Increased security of the data
* Flexible: Programmers can change a part of the code without affecting other parts.
* Easier to make unit testing.

**Inheritance**

Facts:

* Creating a class from an existing class
* Is-a relationship
* Parent(super class)-child(sub class)
* “Extends” keyword
* 3 types of inheritance:



* There is also multiple inheritance but not supported in Java.
  + To reduce complexity of the language
  + May achieve the same effect through interfaces.
  + If c extends a and b & a has the same method as b then there is **ambiguity**.
* Use protected in parent class so the child can access it.
* Can override the parent class method with @Override
* Invoke constructor of parent through child’s constructor by super() keyword.

**Polymorphism**

Facts:

* Many forms
* 2 types:
  + Compile-time(static): Method overloading → call to the method is resolved in compile time.
  + Run-time(dynamic binding): Method overriding → overridden method resolved in runtime
* 2 methods to perform
  + Method overloading → multiple methods with the same name but different parameters and return types.
  + Method overriding → child has the same method as the parent