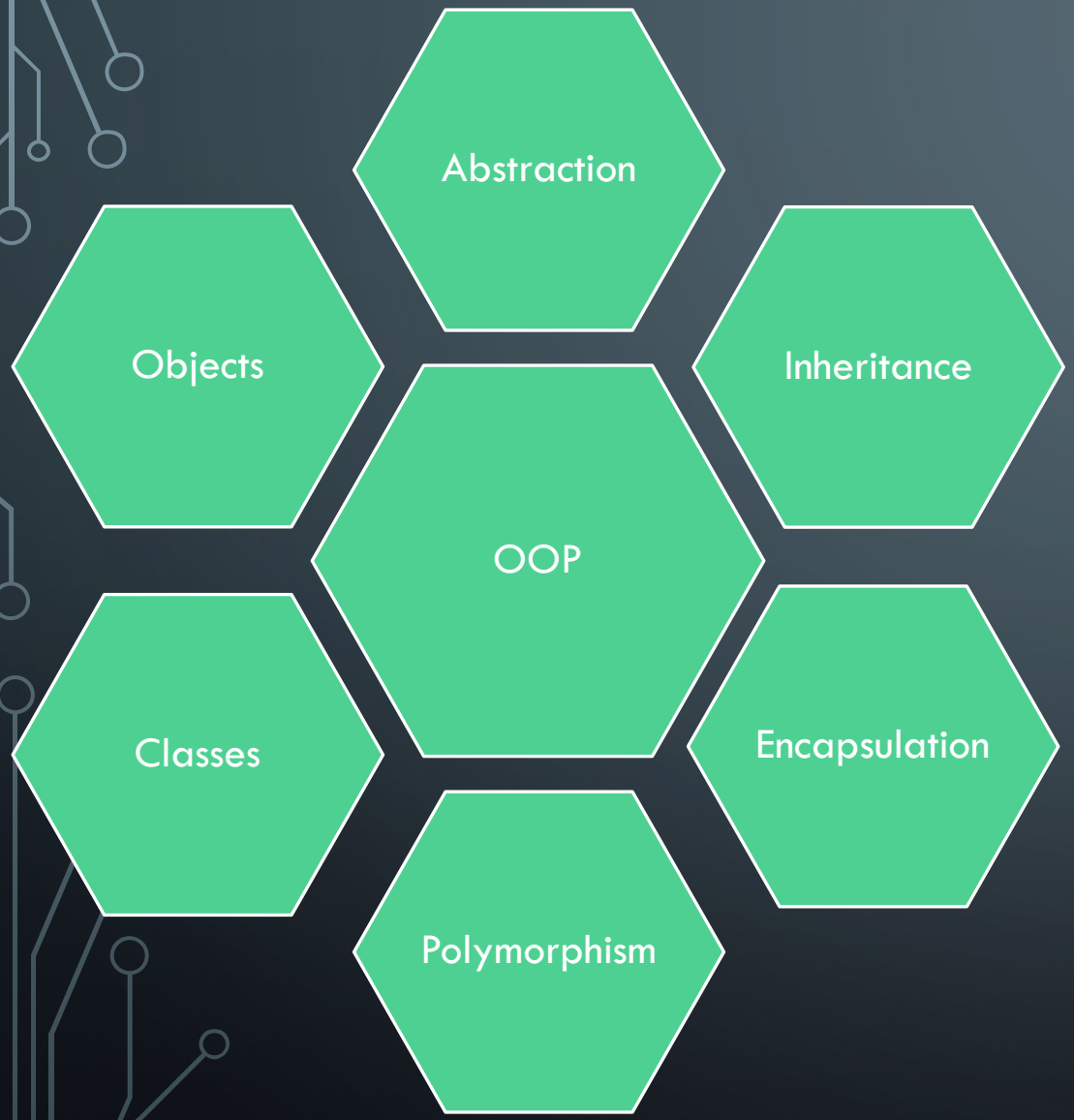
# ASD00- BACKGROUND
## OOP- OBJECT ORIENTED PROGRAMMING

SARINA TILL

# OBJECT ORIENTED PROGRAMMING




**Object Oriented programming (OOP)** is a programming paradigm that relies on the concept **of classes and objects.** It is used to structure a software program into simple, **reusable pieces of code blueprints** (usually called classes), which are used to create individual instances of objects. There are many object-oriented programming languages including **JavaScript, C++, Java, and Python.**

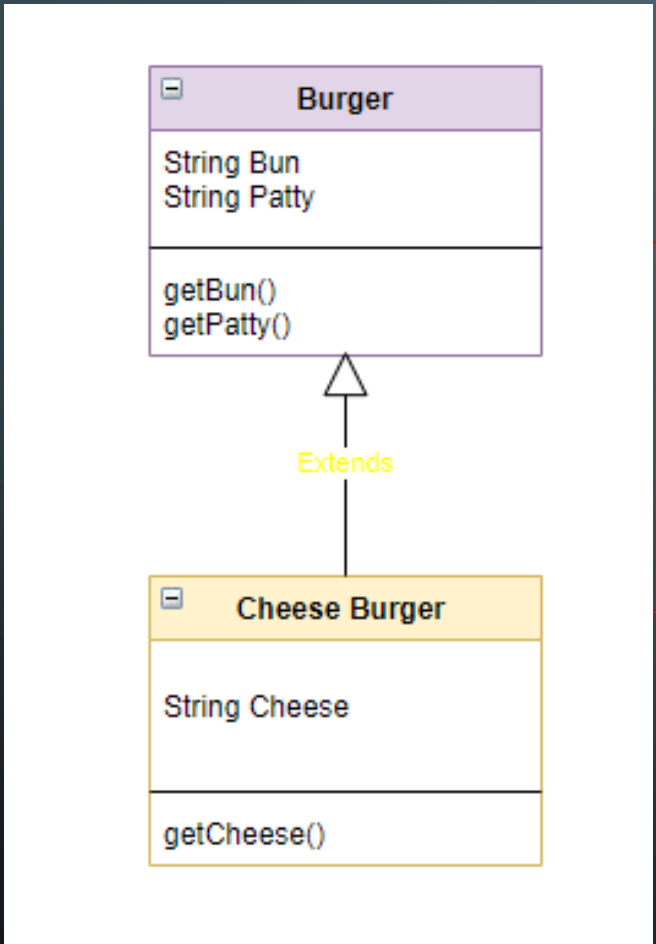
**OOP is made up of 4 Principles:**
**Abstraction**
**Inheritance**
**Encapsulation**
**Polymorphism**

# PRINCIPLES OF OOP:

https://www.youtube.com/channel/UCPy9ky3ocbIJDR3izWABMvQ

# ABSTRACTION

- Presenting data / a concept in its most generic form:

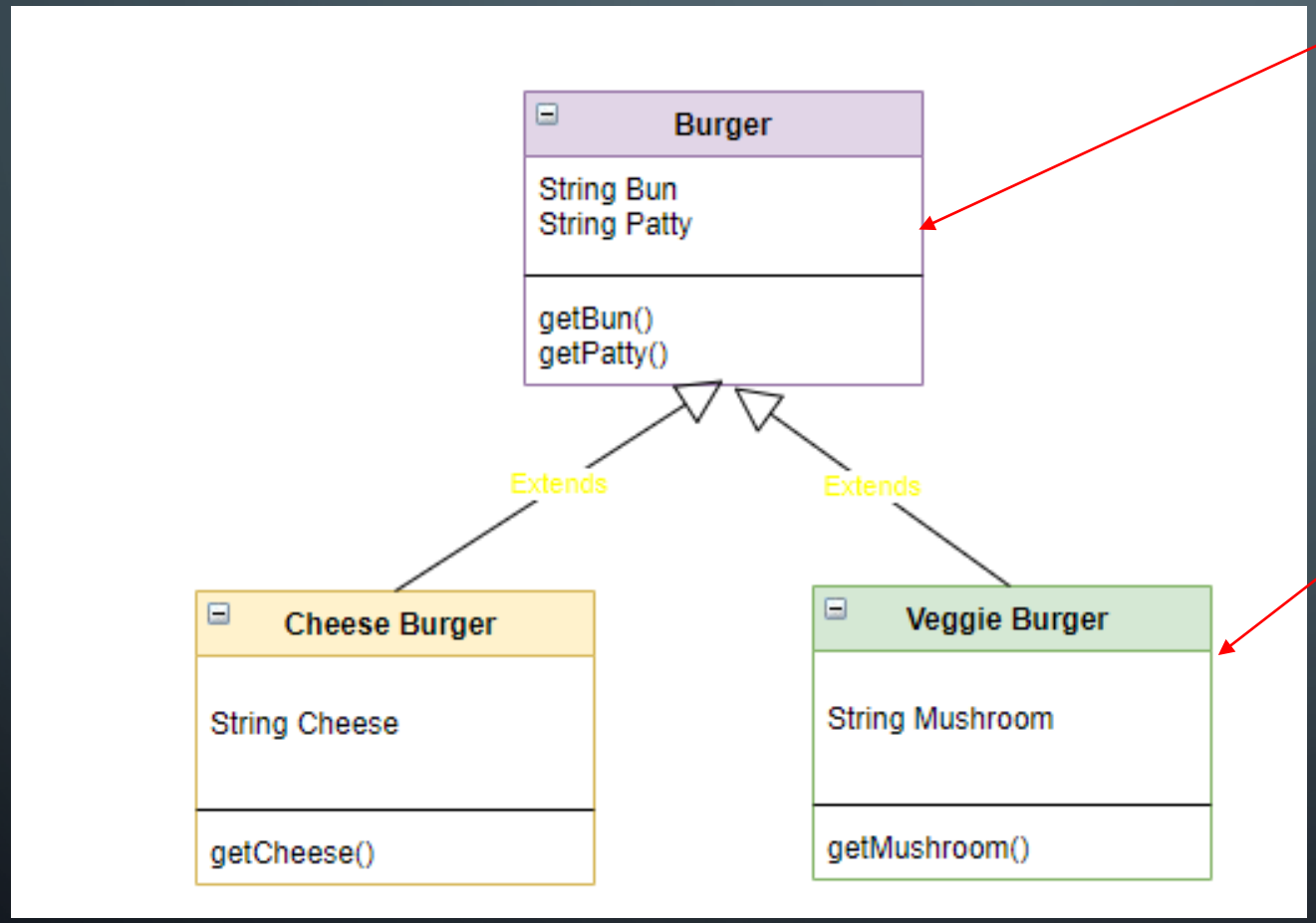



Burger is an **Abstraction** of CheeseBurger class: All burgers have buns and patties

The Cheese Burger class then adds the cheese attribute and the getCheese method which is Specifig to the CheeseBurger class .

This means that we can reuse all the methods and attributes from the burger class in the cheeseburger class

# INHERITANCE




Burger is the Generic class that contains the generic methods applicable to all the classes that extend it ( inherit from it)

**Code Reuse**
Cheese and Veggie burger will get access to all the methods contained in the Burger class through inheritance and will add their specific attributes and methods

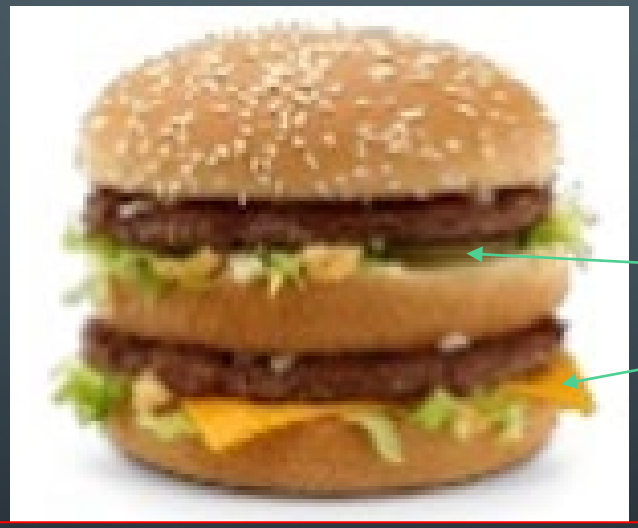
This also gives an IS A relationship - A cheeseburger IS A Burger, A Veggie Burger IS A Burger

Unless all your burgers will have cheese on it – Cheese and getCheese does not go in this class

***Just Ham Burger**

**-String bun**
**-String patty**

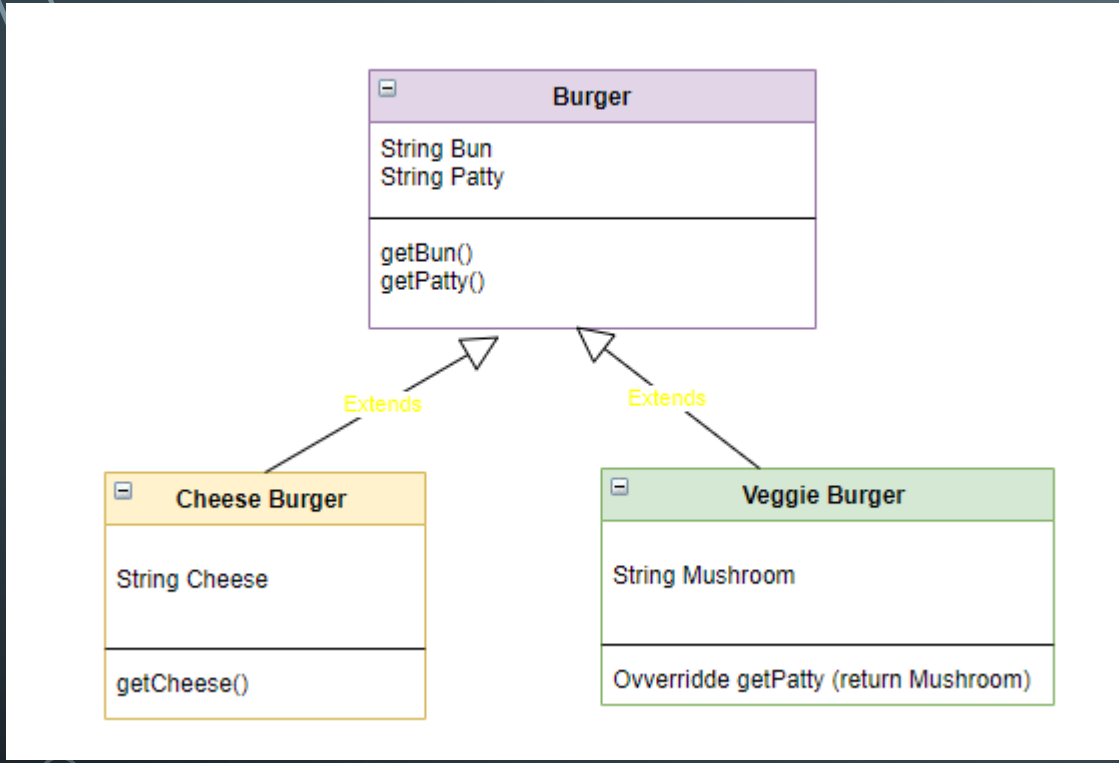
**+getbun()**
**+getPatty()**



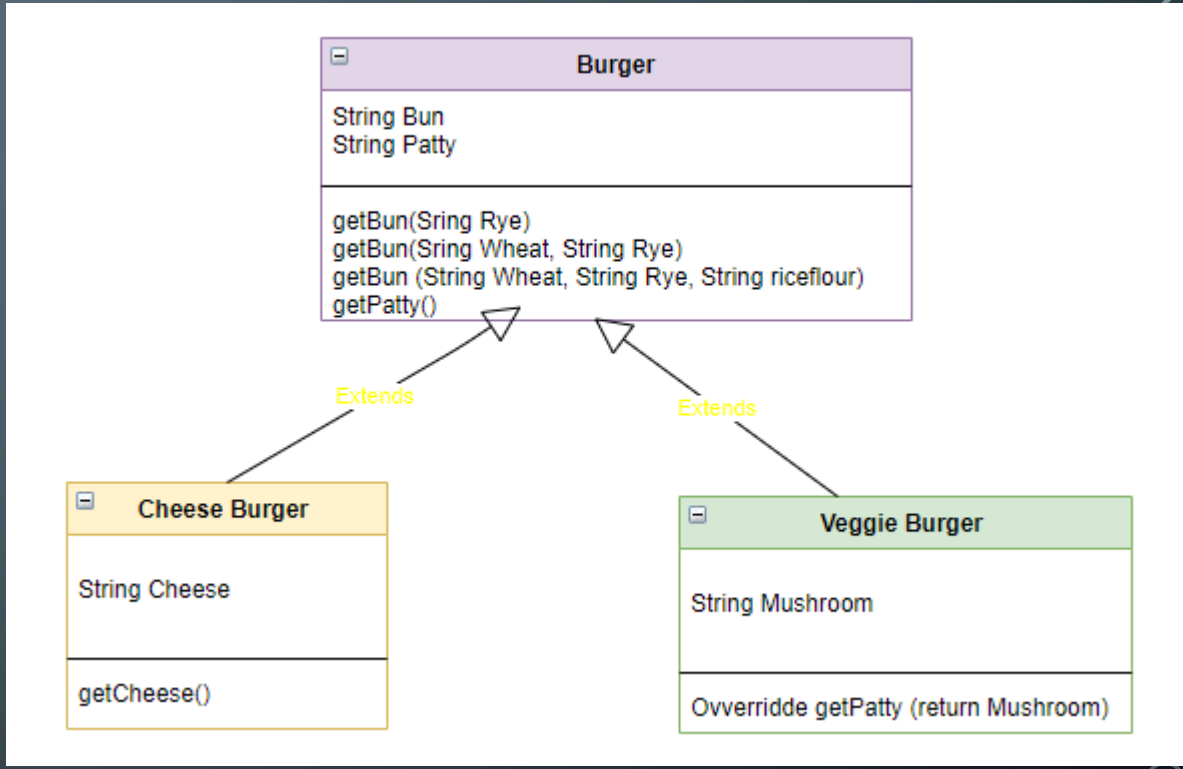
**CheeseBurger** class will inherit from the hamburger class and add its cheese, lettuce etc – NON Generic

**Methods do not belong in an abstract class if you are not going to inherit them – All methods in an abstract class does not have to be abstract**

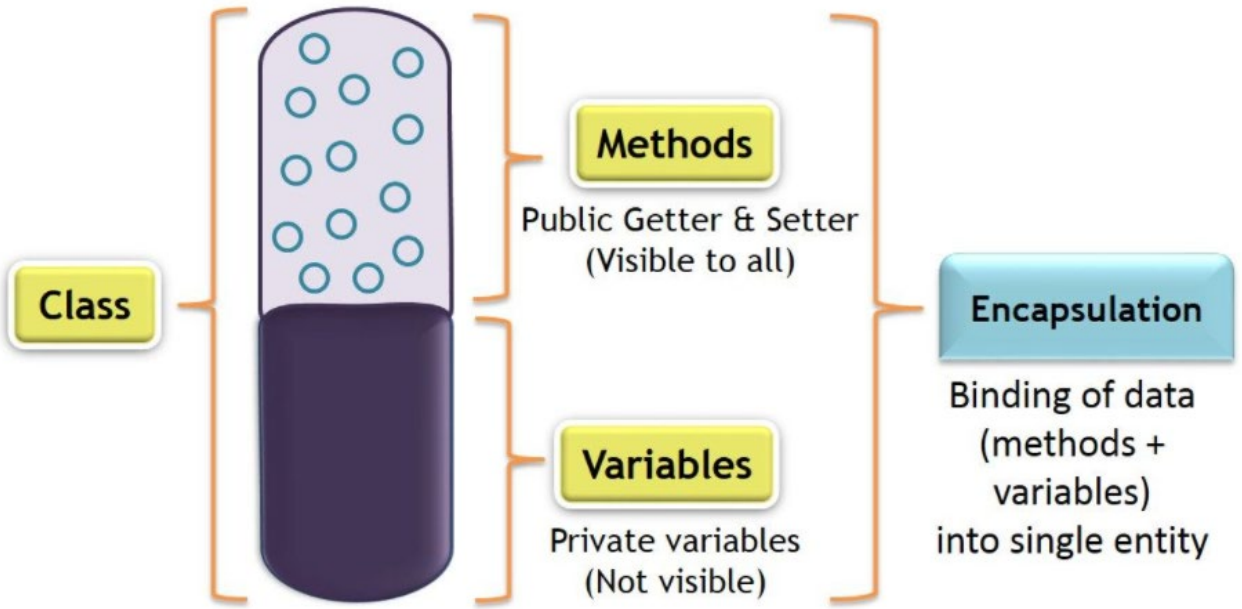
# POLYMORPHISM




**Overriding :** Changing the function of a method in the base class: Here we override the getPatty() method to return a Mushroom rather than a beef patty

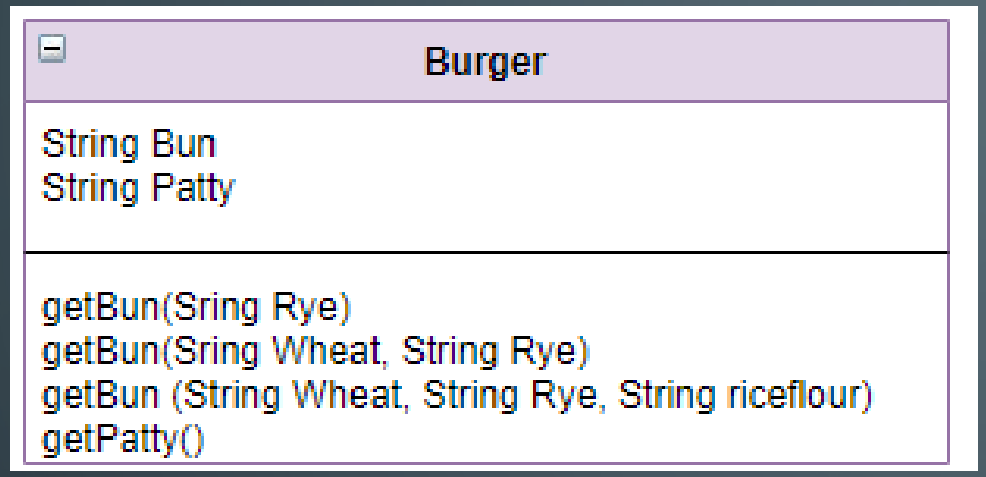



**Overloading :** Providing different parameters to the same method – here we are overloading the getBun method – you can either pass one parameter for a plain Rey bun, or add Wheat , or add Wheat and Rice flour




# ENCAPSULATION:

- Encapsulation means containing all important information **inside an object,** and only exposing selected information to the outside world.




RAM (Memory)

plainBurger:
Bun = Wheat
Patty = Beef

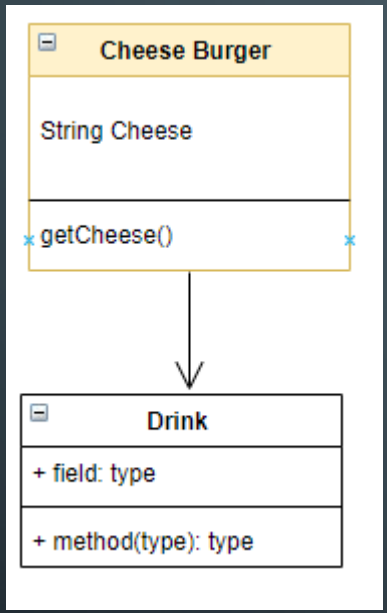
```
Burger plainBurger = new Burger()
plainBurger.getBun();
```

Encapsulation

We use the burger class to create the plainBurger object – all the attributes and methods are encapsulated in this class/object and can only be accessed through this class/object

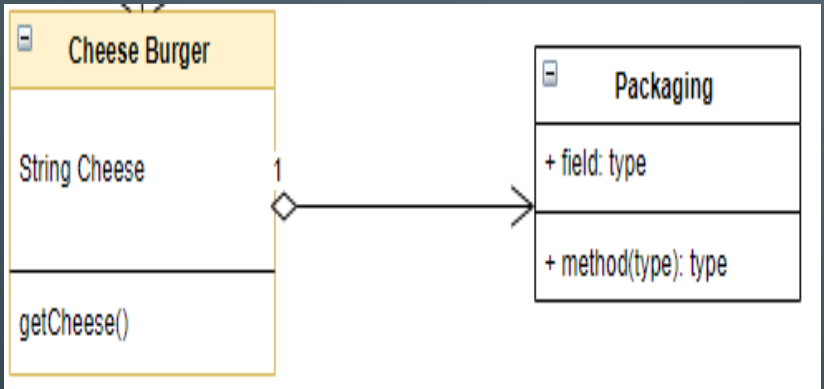
# PART / WHOLE RELATIONSHIPS

## Association



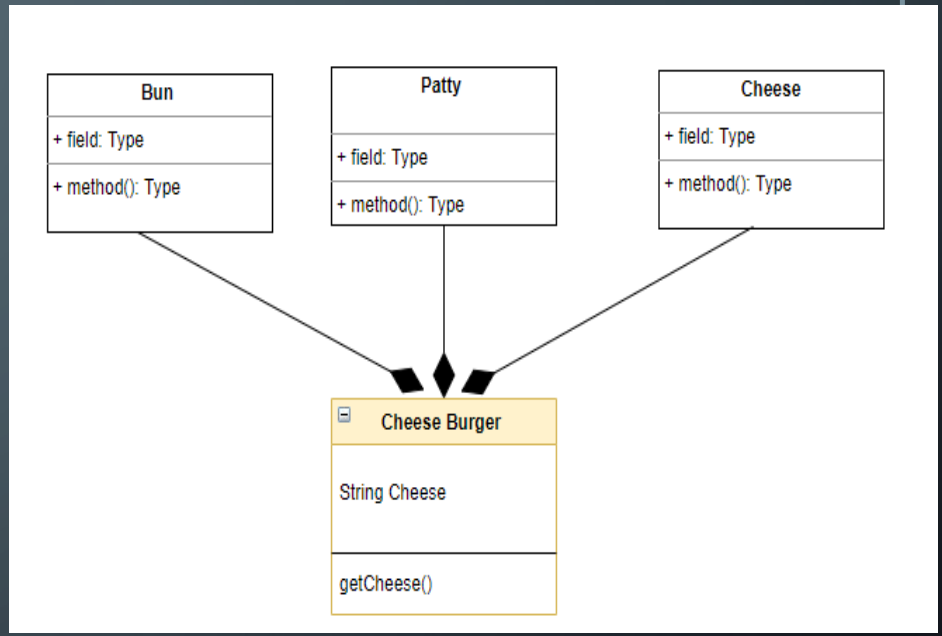

Only a relationship between classes

## Aggregation




CheeseBurger and Packaging is related, but one can exist without the other

## Composition




A cheeseburger is composed (made off) a bun, a patty and cheese and cannot exist without these elements.

https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-aggregation-vs-composition/